UI Editor

User Manual

V2.30

Version	Date	Description
V2.30_02	Mar-20-2025	English Version Release

www.buydisplay.com

EastRising Technology Co.，Ltd.

© EasSemiconductor Co, Ltd.

Contents

1 Introduction

UI_Editor-II is a 2nd generation UI editing tool for UartTFT panels. It is designed for Uart TFT displays. This manual is to illustrate how users can utilize this tool to implement UI designs. If you would like to learn about UI Editor II through videos, please visit this link.We have recorded detailed tutorial videos for almost every widget application.

(https://www.buydisplay.com/blog/Tutorial-Video-Collection-for-UI-Editor-II.html)

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

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-II, a simulation tool for UI_Editor-II projects. Refer to *UI_Emulator-II;*

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

2 UI_Editor-II Installation

2.1 UI_Editor-II Tool Kits

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

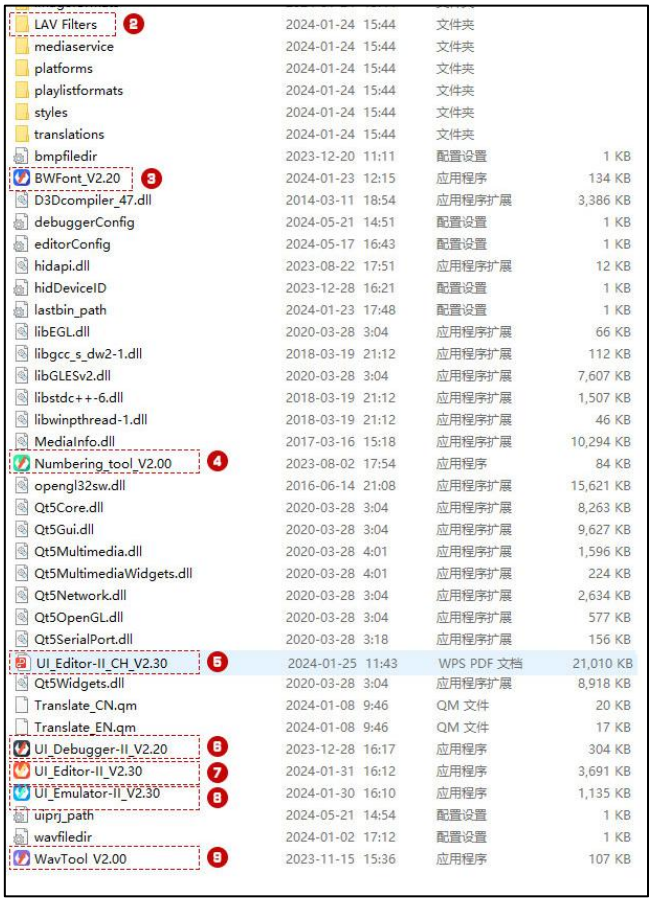


Figure 2-5: UI_Editor-II File Folder

❶ **Example Folder:** Three demo projects are available in this folder.

❷ **LAV Filters Folder:** A video decoder tool used by UI_Emulator-II is stored in this folder.

❸ **BWFont_Vx.x.exe:** A tool for customizing Fonts, refer to *Font Tool*。

❹ **Numbering_Vx.x.exe**: A tool for numbering the material files, refer to *Numbering Tool*。

❺ **UI_Debugger-II_Vx.xx.exe:** A tool for testing and monitoring the communication between PC and the UartTFT controller, refer to *UI_Debugger-II* and *Uart Communication*

❻ **UI_Editor-II_ENG_V2.xx.pdf:** User manual

❼ **UI_Editor-II_Vx.xx.exe:** Main program. Please right click on it and select [Run as administrator]

❽ **UI_Emulator-II:** An emulator for the projects created by UI_Editor-II, refer to *UI_Emulator-II*。

❾ **WavTool_Vxx.exe:** A tool for converting Wav files to bin files, refer to *WavTool*。

2.2 Activate UI_Editor-II

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

Figure 2-6: Locate UI_Editor-II-II_Vx.xx

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

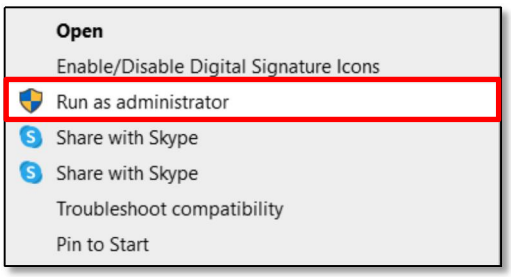


Figure 2-7: Select [Run as administrator]

3 UI_Editor-II Menu & Operation

3.1 Main Screen

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

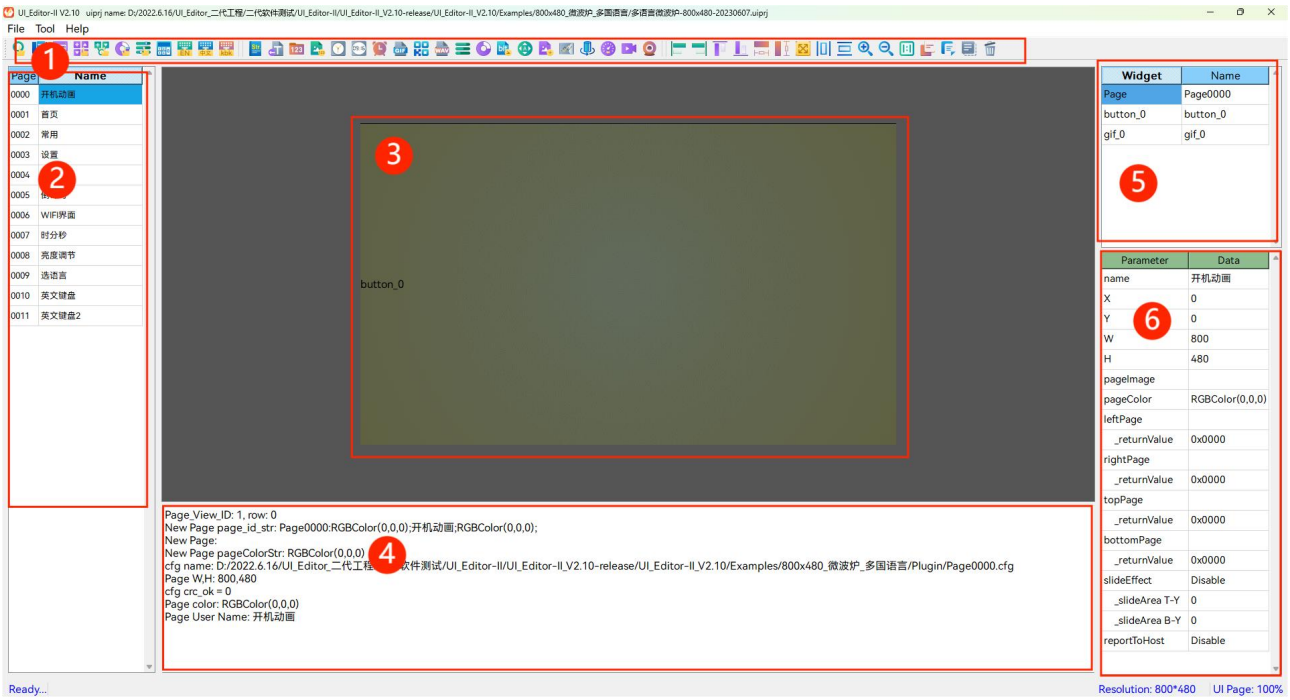


Figure 3-1: Main Screen

1、Tool bar

As shown in Figure 3-1, ❶, 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

❹ Widgets for delete/copy operations

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, ❷, 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, ❸, developers may edit (e.g. adding widgets) within the base map.

4、Status Window

As shown in Figure 3-1, ❹, 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, ❺, 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, ❻, 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

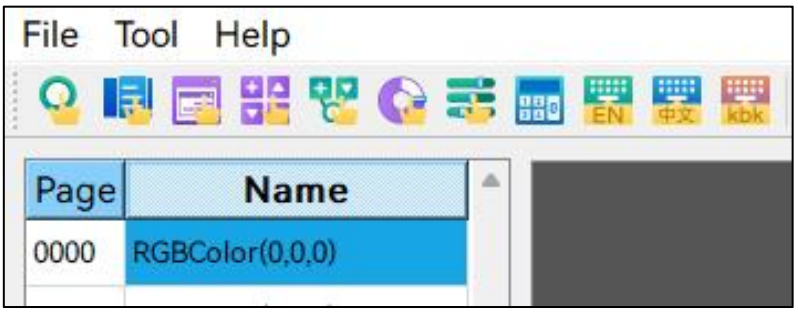


Figure 3-3: Function Menus

3.2.1 File

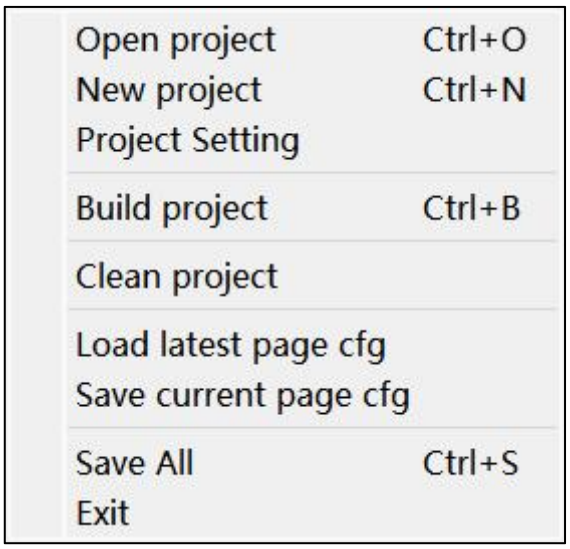


Figure 3-4: File

1. Open project: Open an existed project

2. New project: Create a new project

3. Project Setting: Refer to *Project Setting*

4. Build project: Compile the current project and export the UartTFT-II_Flash.bin

5. Clean project: Deleted all bin files (except for font and wav bin files)

6. Load latest page cfg: Load the latest cfg file

7. Save current page cfg: Save the parameters to a cfg file

8. Save All: Save all changes

9. 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

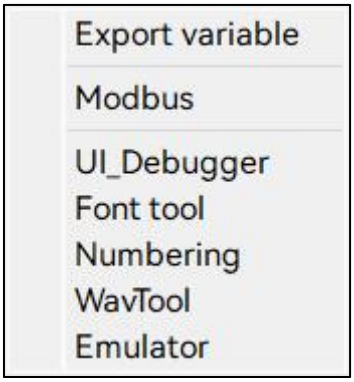


Figure 3-5: Tool

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

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

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

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

7. Emulator: Click to open emulator tool. Refer to *UI_Emulator-II*

3.2.3 Help

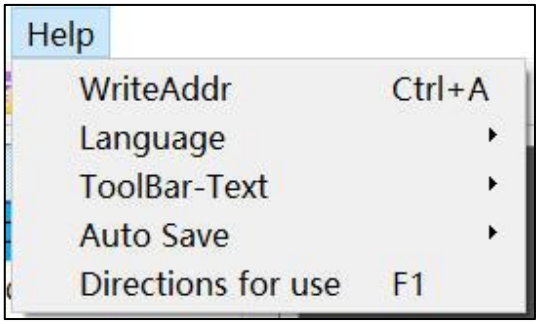


Figure 3-7：Help

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.

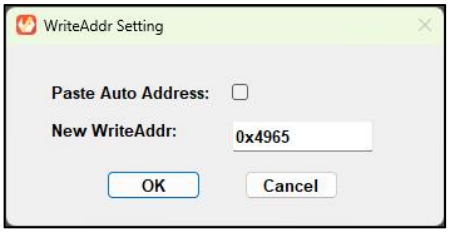


Figure 3-8：writeAddr

2. Language: Options for Chinese and English menu.

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

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

5. Direction for use: Open the user manual

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-II and should not be changed.

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

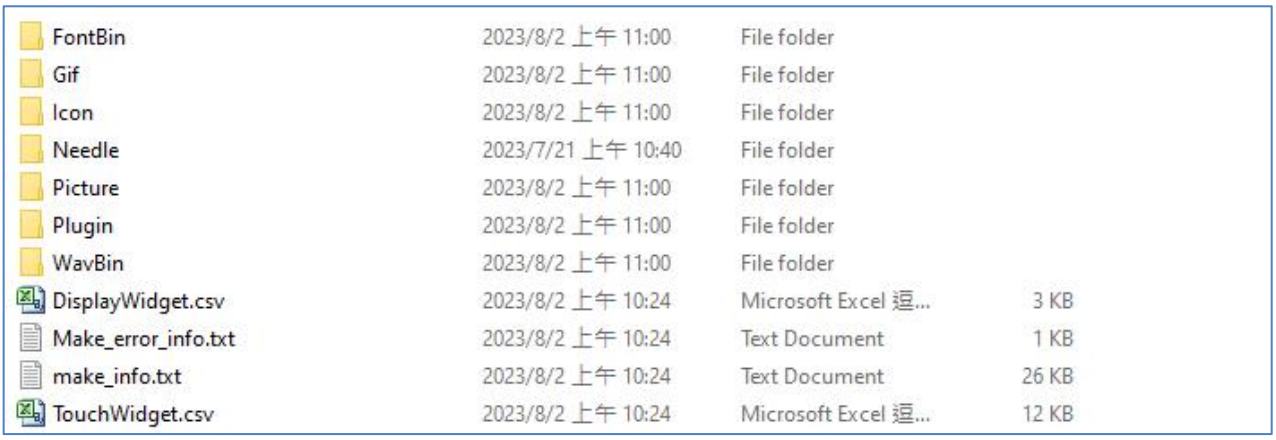


Figure 4-1: File Folders

4.1.2 Material Format

4.1.2.1 Picture Folder

Contents: Page pictures, Popupbox 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:

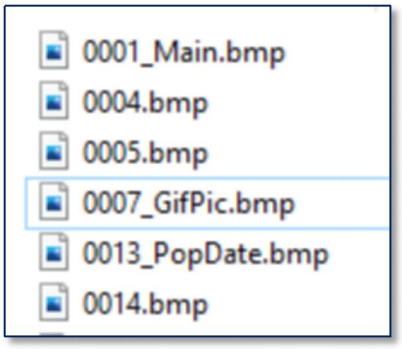


Figure 4-2: Name a Picture

Note:

1、Each number can only be used once

2、If the maximum picture number is 0010, and there is only 6 pictures in the folder, then UI_Editor-II 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 amount 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-II, 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 EastRising to quickly number the pictures. Refer to *Numbering Tool*

4.1.2.2 Icon

Contents: Icons, Graphic Number Display, SlideMenu, 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:

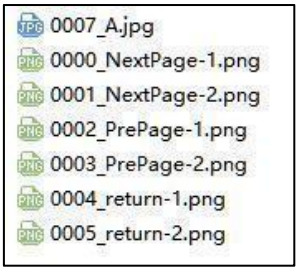


Figure 4-3: Name an Icon

Note:

1、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:

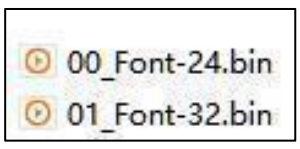


Figure 4-4: Name a Font

Note:

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

4.1.2.4 Gif

Contents: Gif

Format: Gif

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

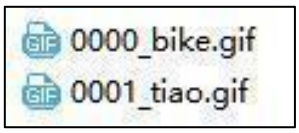


Figure 4-5: Name a Gif

4.1.2.5 WavBin

Contents: Wave bin files

Format: bin

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

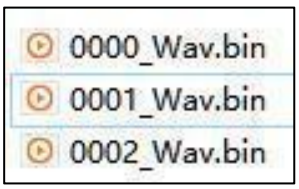


Figure 4-6: Name a Wav

4.1.2.6 Music

Contents: Audio files

Format: mp3

Naming: Number the Audio files by 0001 ~ 0099, and name them as "**00xx**".

Note: This folder stores mp3 files for LT3688 applications.

4.1.2.7 Needle

Contents: Picture generated by needle widgets

Format: png、bin

Note: The contents are generated by the widget automatically.

4.2 Create a New Project

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

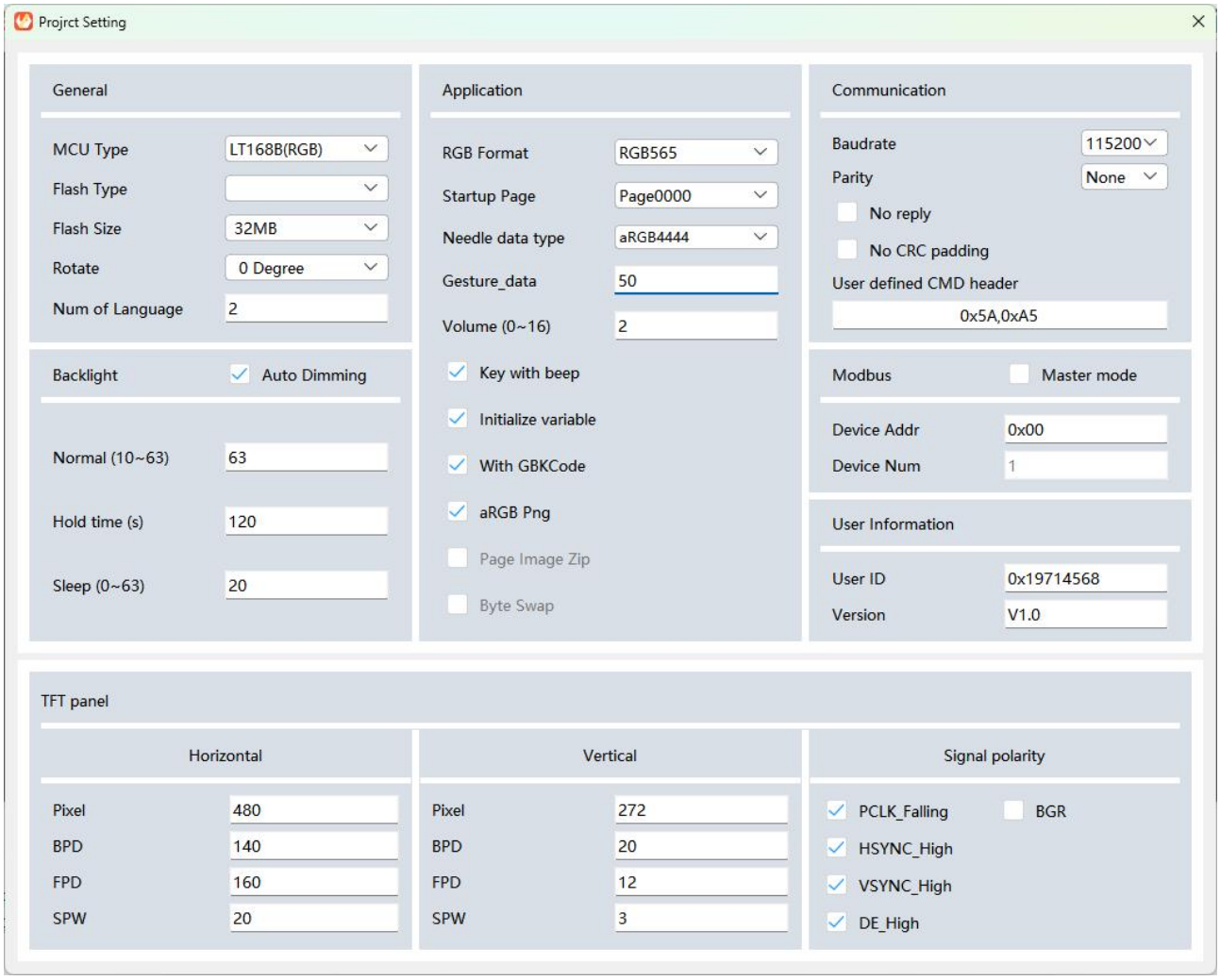


Figure 4-11: Project Setting

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 amount 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. LT7689 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-II_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
	Page Image Zip	:	Only available for LT776. If checked, the page pictures will be compressed to reduce the file size of UartTFT-II_Flash.bin
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.

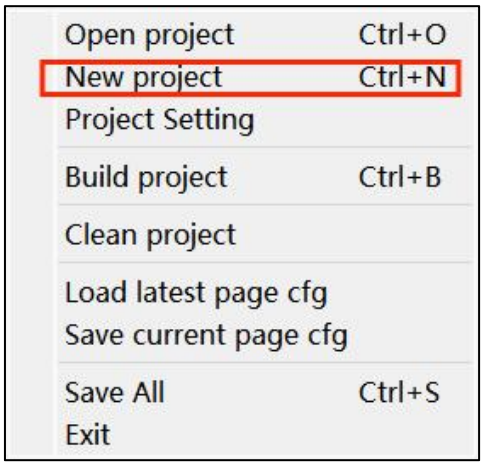


Figure 4-12: Create a New Project

Step 1: Create the file folders as described in *Materials Preparation*, 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:

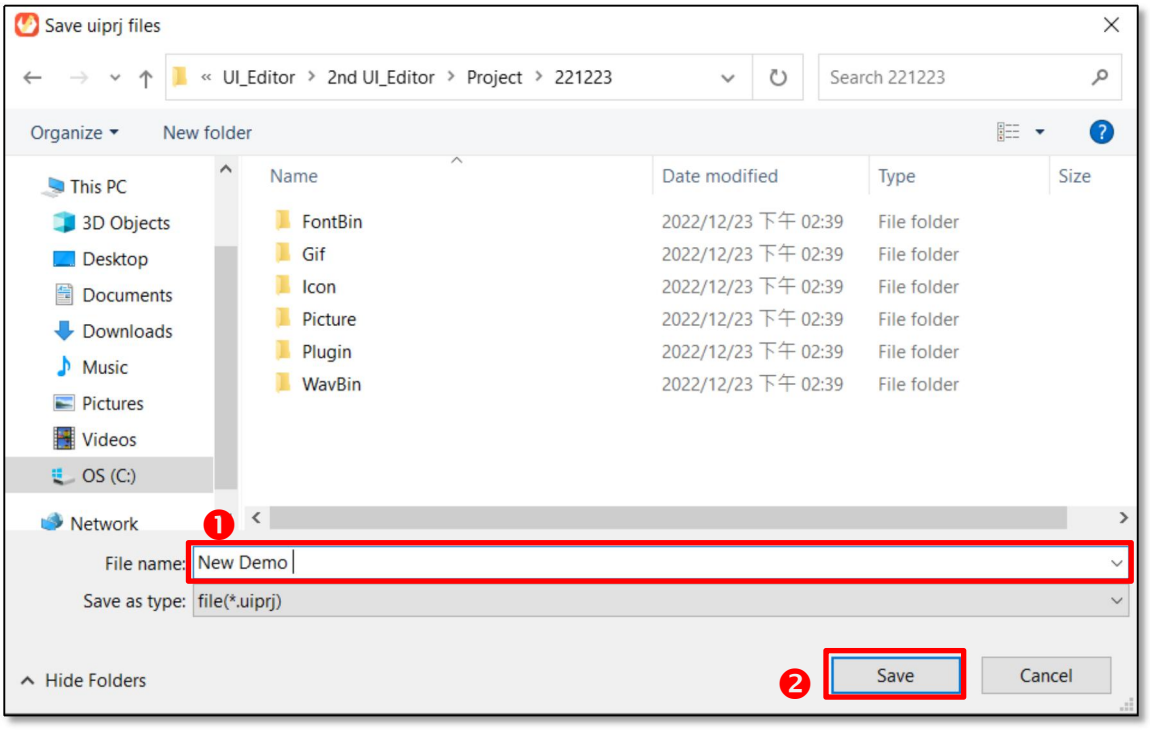


Figure 4-13: Enter Project Name and Save it

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].

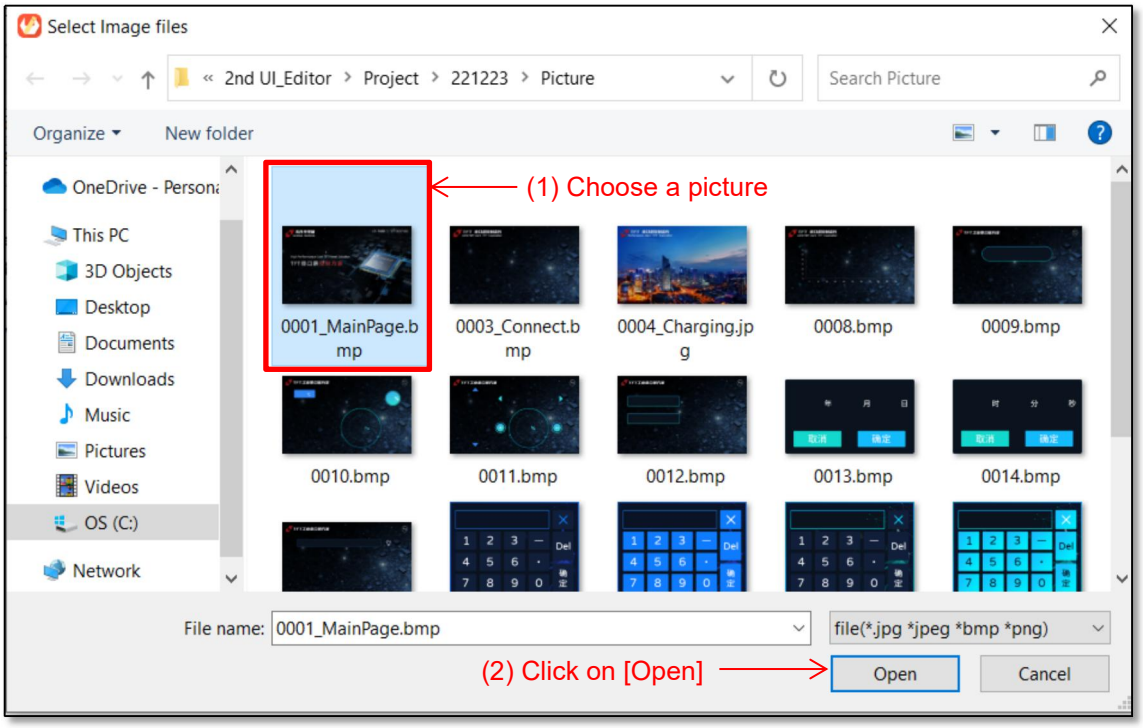


Figure 4-14: Choose a Picture

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

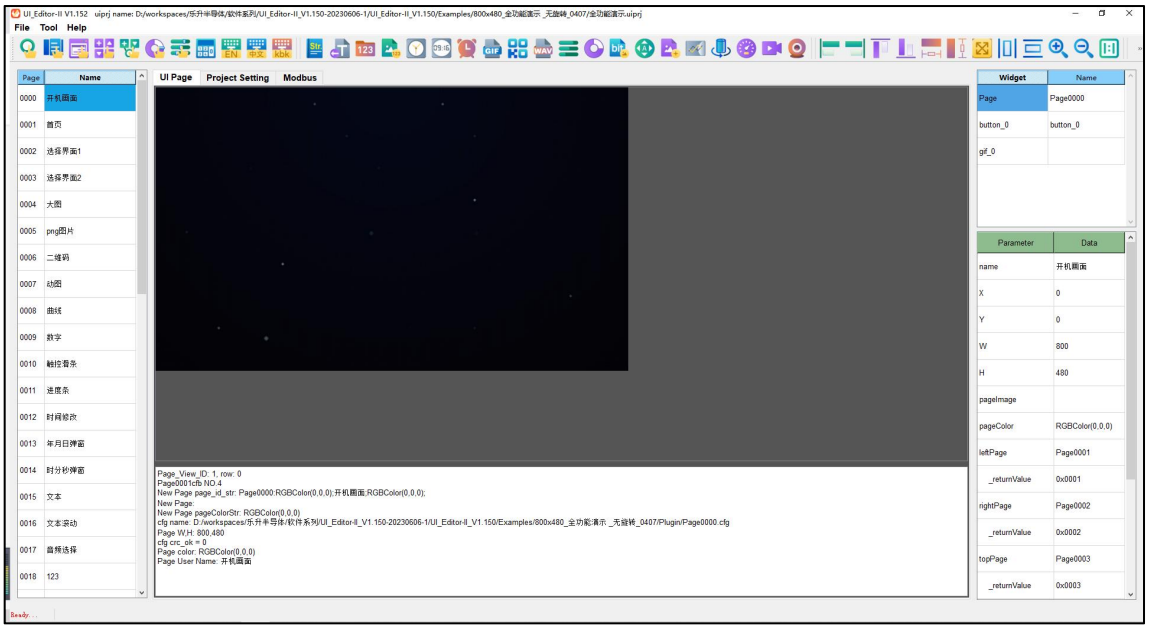


Figure 4-15: UI Page View

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.

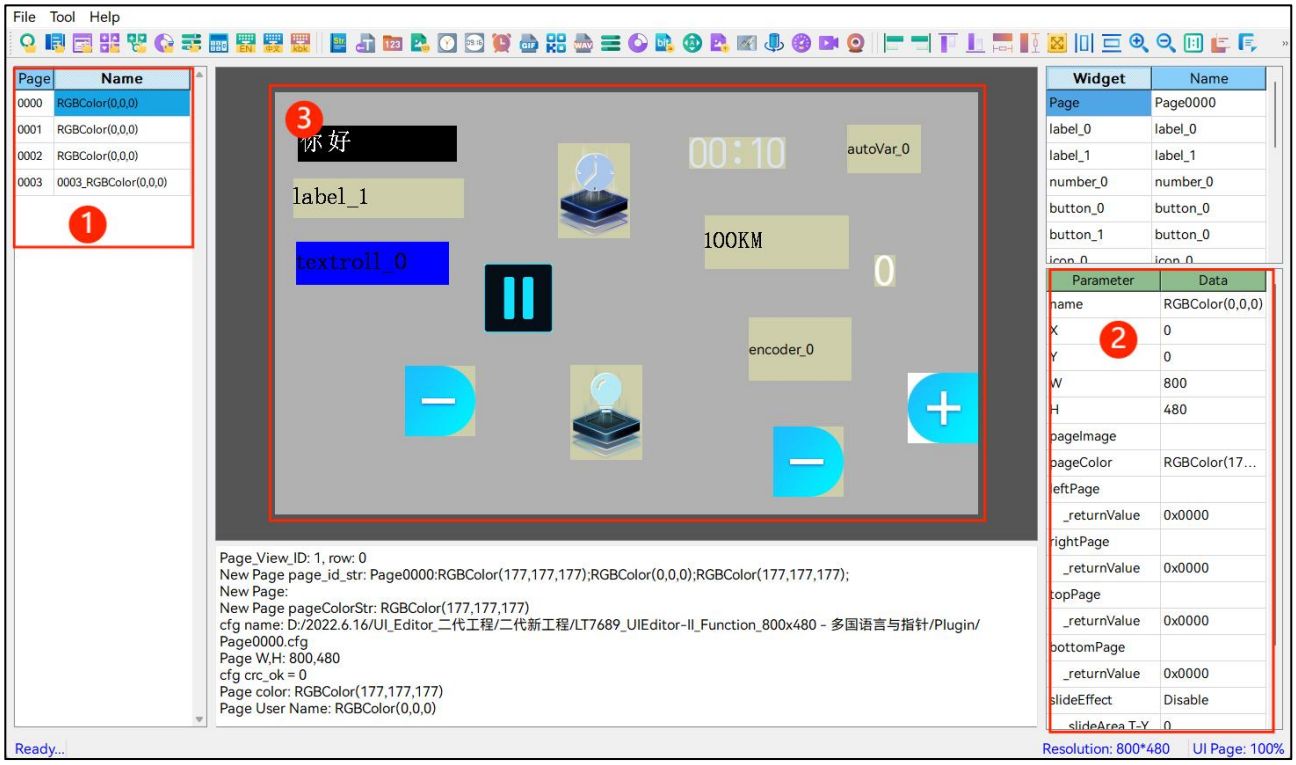


Figure 5-1: Check Page Parameters

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

MCU Number	Slide to Jump
LT7689 (7689)	Support
LT168A (168A)	Not Support

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 Multi-Variable Button widgets, refer to *Multi-Variable 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. Issue 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.

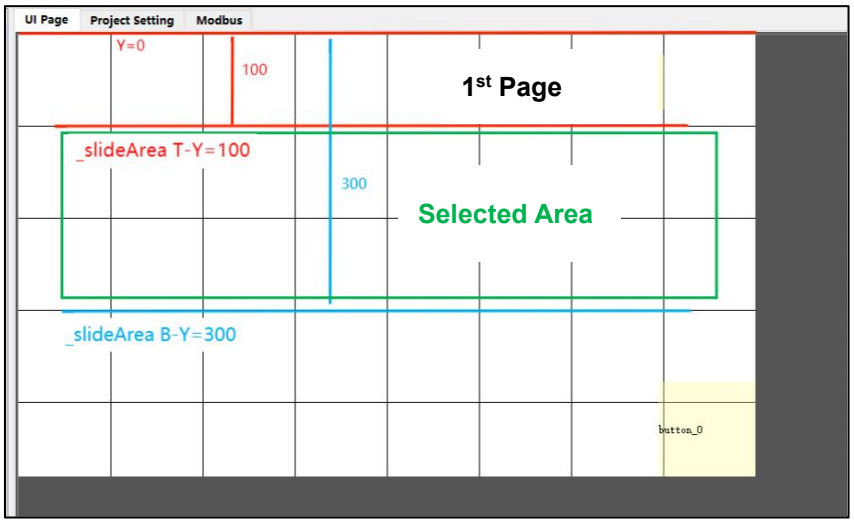


Figure 5-5: Sliding Area

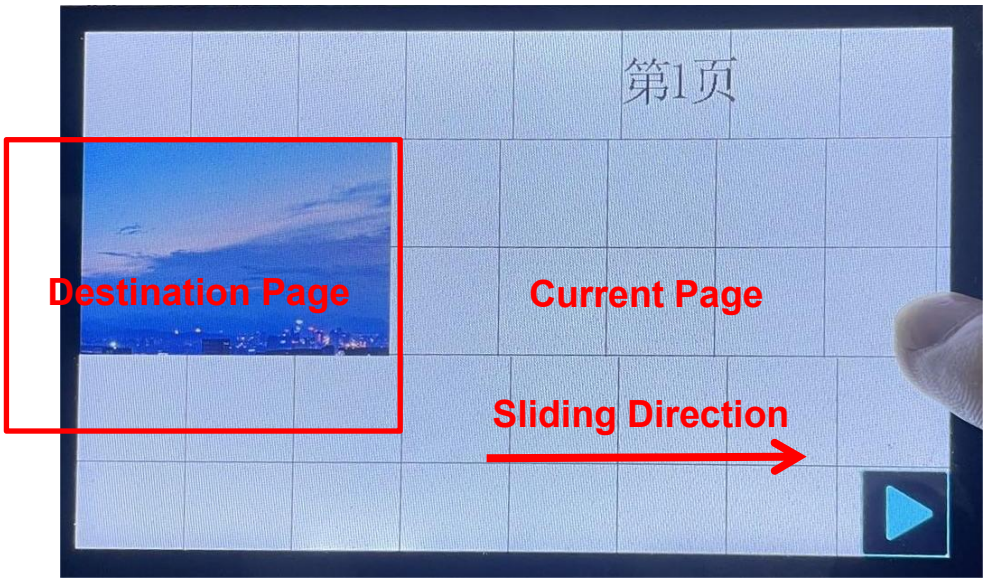


Figure 5-6: Demonstration on Sliding Area

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 shown below:

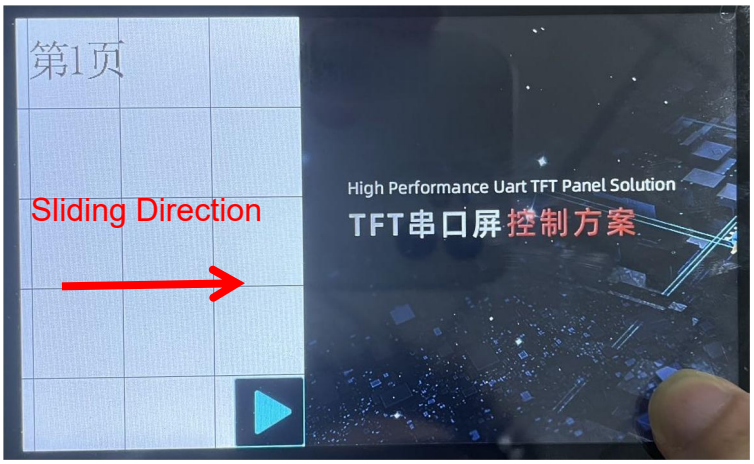


Figure 5-7: Sliding the whole page

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 Y resolution

3. If RGB format is set as RGB565, the HMI displays do not support the sliding effect on panels with resolution of 1024x600 or above.

   If RGB format is set as RGB888, the HMI displays do 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.

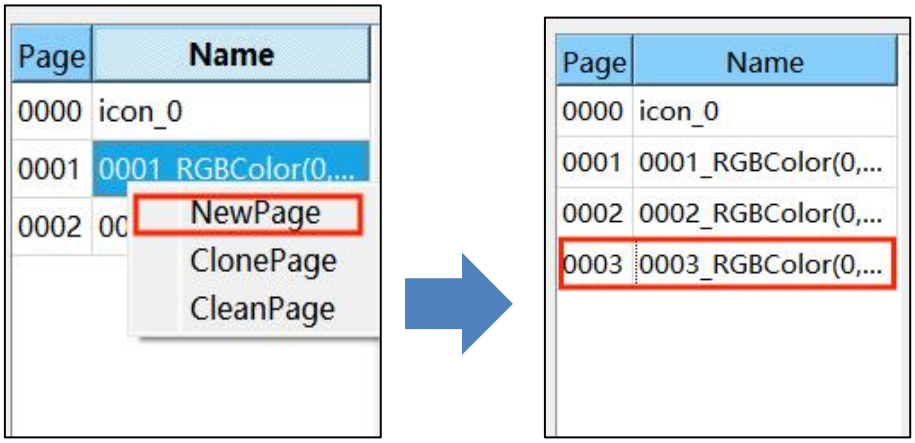


Figure 5-8: Add a New Page

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.

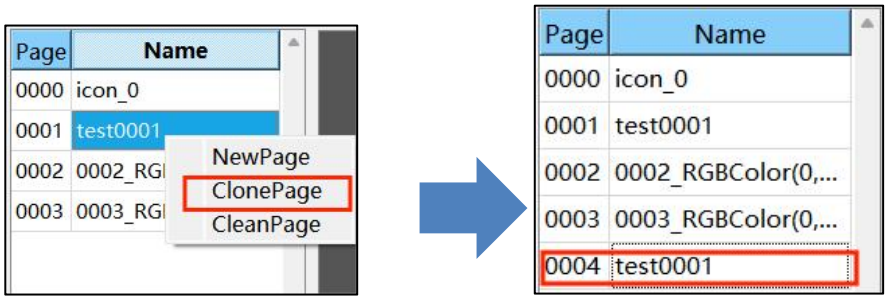


Figure 5-9: Clone a Page

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.

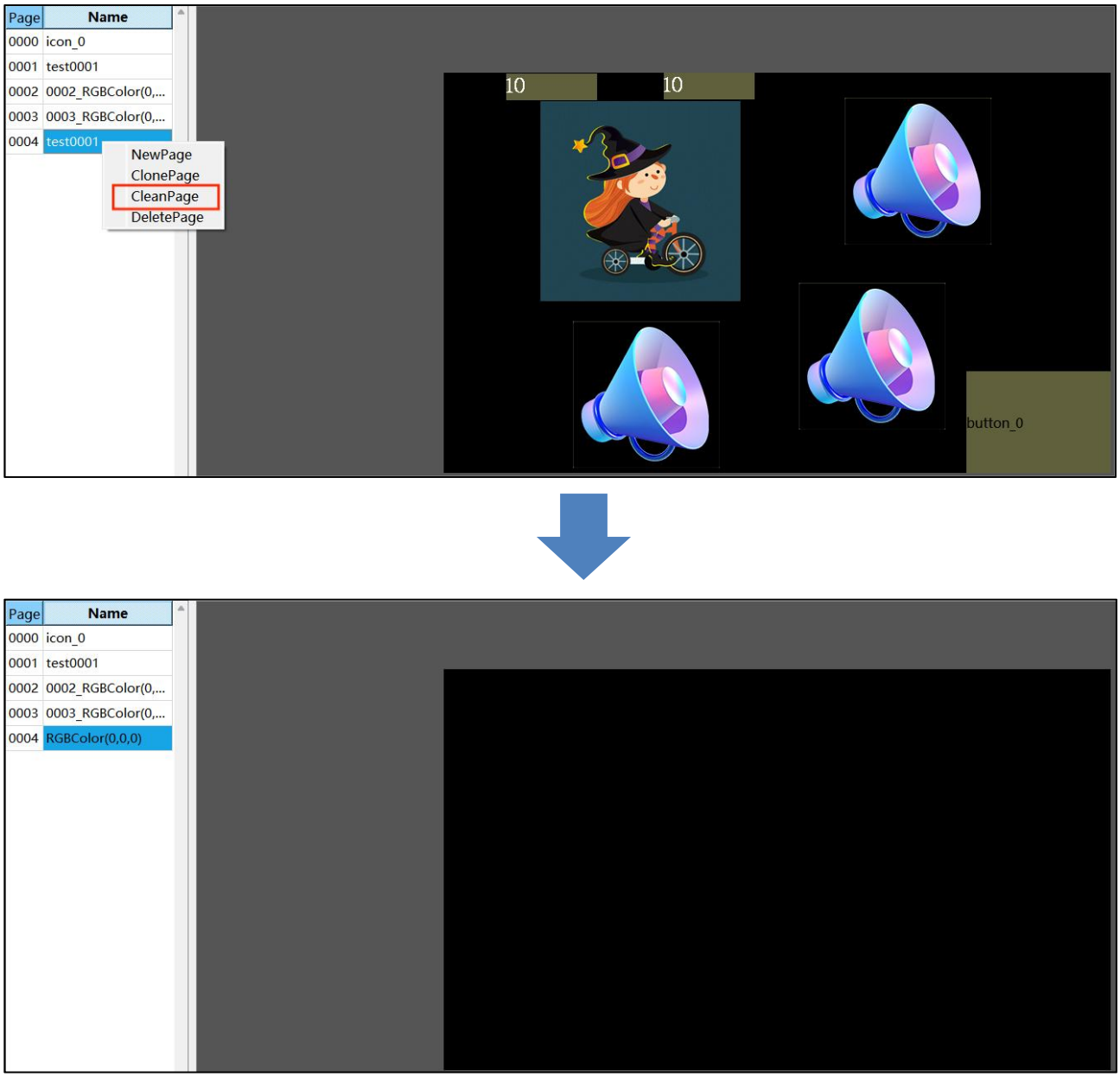


Figure 5-10: Clean Page

5.6 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.

5.7 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.

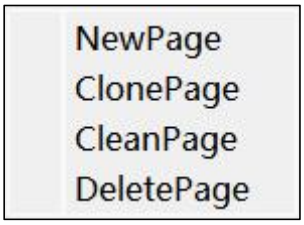


Figure 5-11: Delete the Last Page

5.8 Basic Operation

5.8.1 Add a Widget

Step I: Click on the target widget icon, the cursor will be switched to "Corss" style.

Figure 5-12: Add a Widget

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

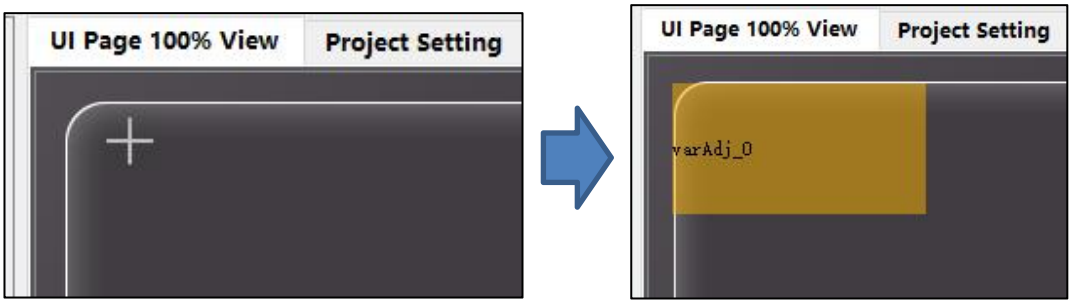


Figure 5-13: Generate a Widget

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.

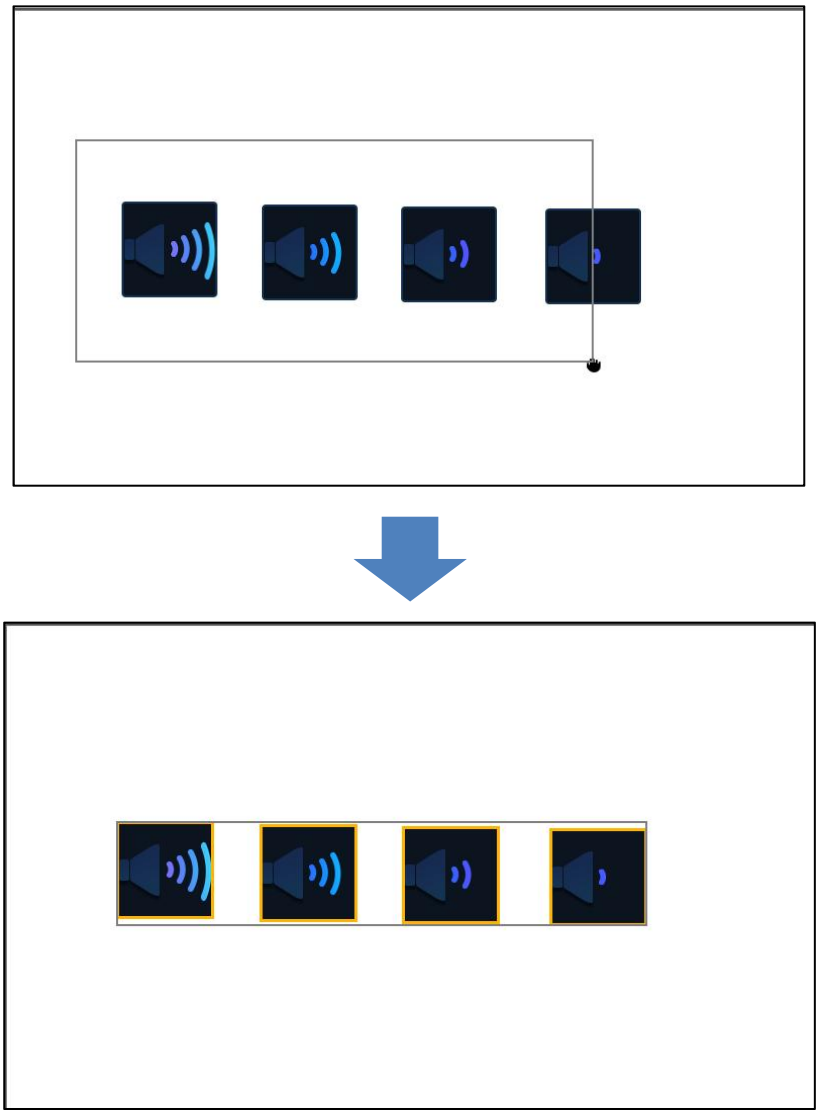


Figure 5-14: Frame Selection

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.

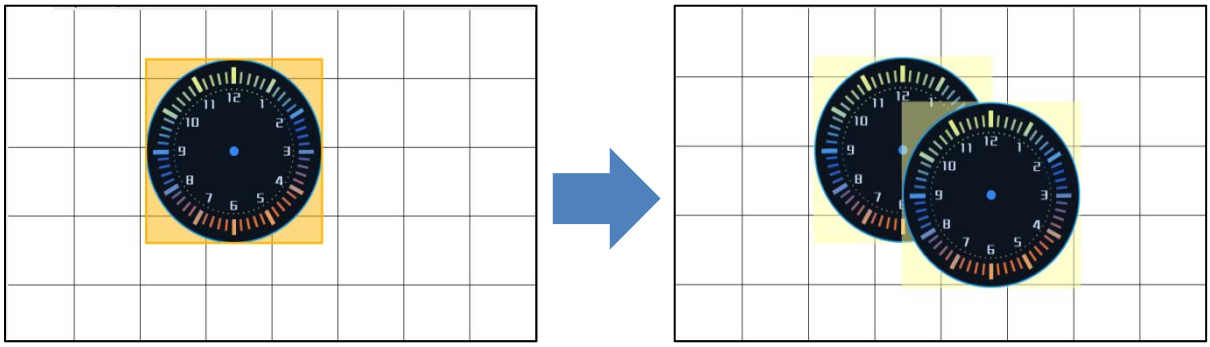


Figure 5-15: Clone a Widget

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.

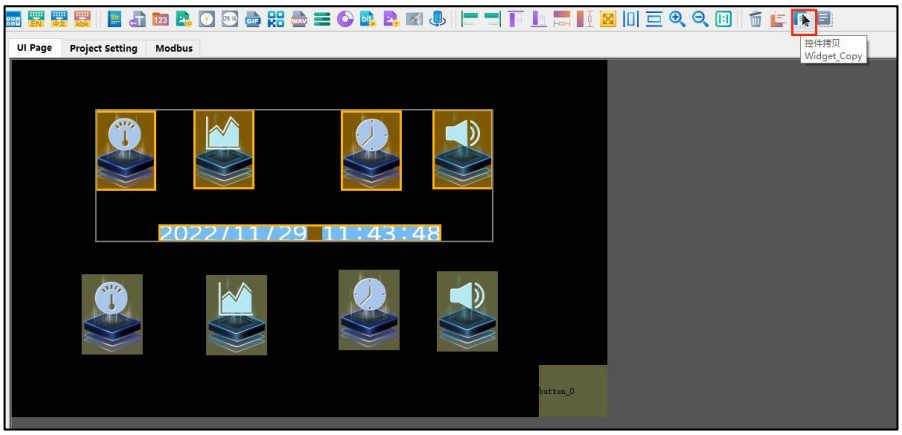


Figure 5-16: Copy Widgets

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

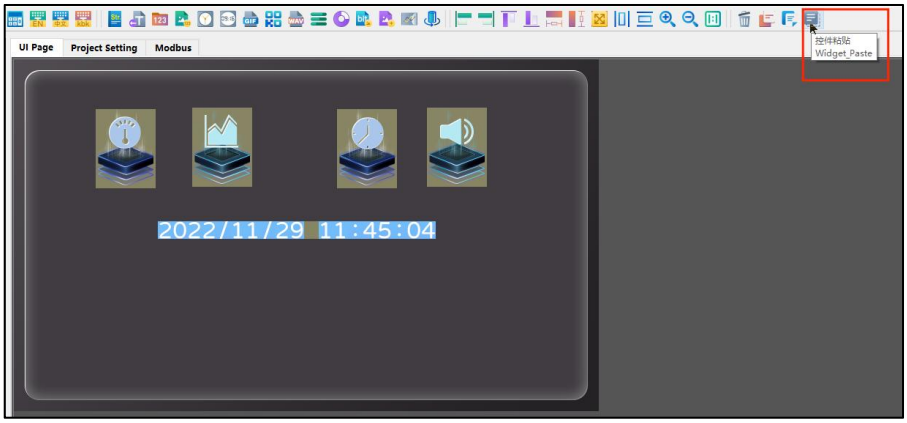


Figure 5-17: Paste the Widgets

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.

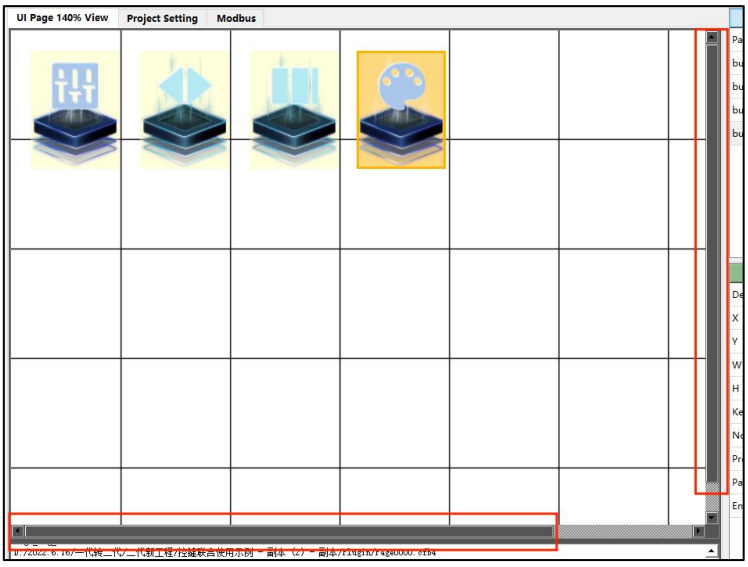


Figure 5-18: Fine-tune Widget Location

5.8.7 Load previous step and Load next step

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.

Figure 5-19: Load [Previous] & [Next] step

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 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

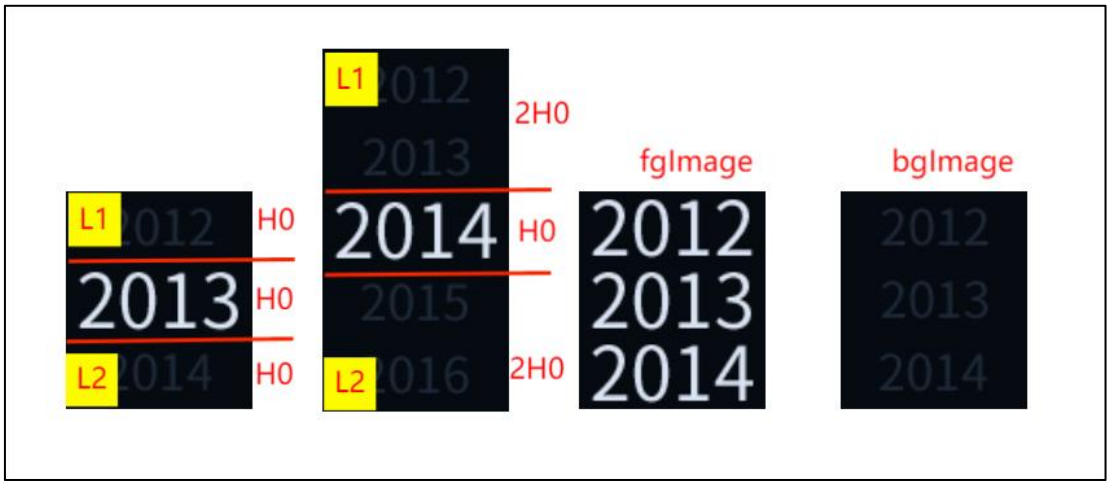


Figure 6-3: Example of SlideMenu

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 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.

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 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 Multi-Variable Button

ICON: 

Function : To control the most 8 variables by a single button

name : Name of the widget, user-definable

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

W & H : The width and height of the Multi-Variable Button. 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.

pageGoto : Setup which page to jump to. Leave it empty if no page jump needed.

writeAddr0~7 : Address of the variable

_value9~7 : Value of the variable (Hexadecimal)

reportToHost : Set [Enable] to report the 8 writeAddr and their values 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 button.

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

Figure 6-6: Multi-Variable Button

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

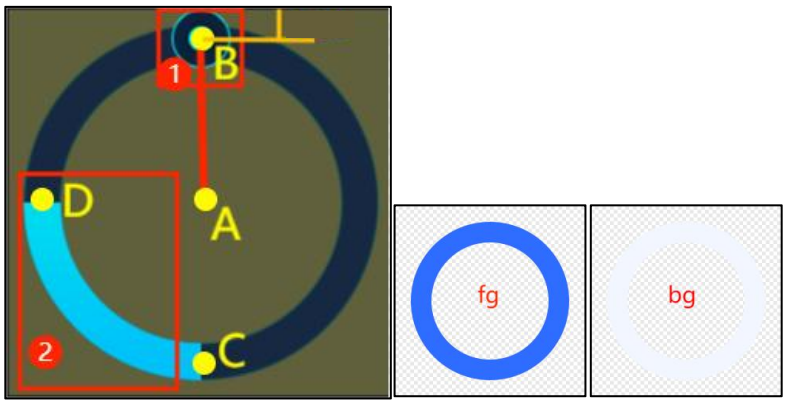


Figure 6-7: Circular Touch

Parameter	Data
name	rtouch_0
writeAddr	0x0002
X	367
Y	87
W	147
H	125
foreground	
background	
slideButton	
slide_R	50
touch_R	50
minValue	0
maxValue	100
defaultValue	0
startAngle	0
finalAngle	359
promptNum_x	73
promptNum_y	62
integerDigit	3
decimalDigit	0
alignment	Left
fontID	
fontColor	0x000000
firstIcon	
lastIcon	
digitDisplayMode	NULL
reportToHost	Disable

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-II will auto calculate the value.

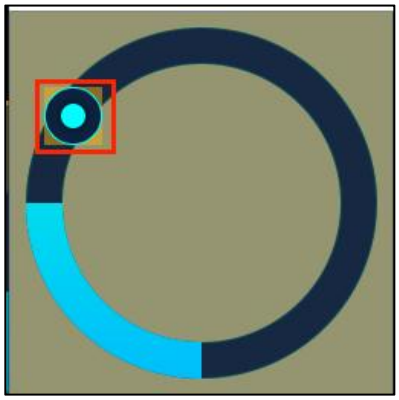


Figure 6-9: slide_R

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.

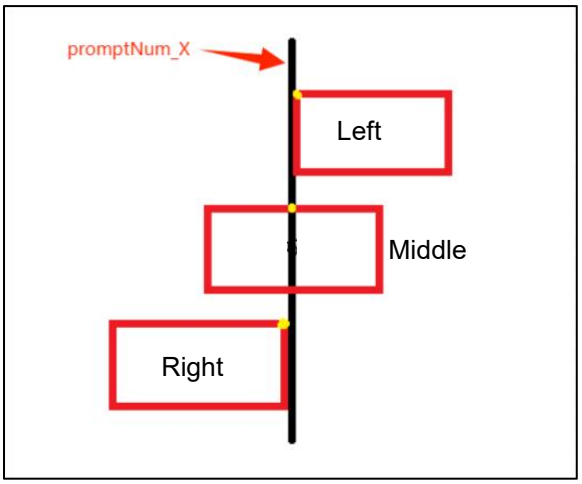


Figure 6-10: Prompt Number Alignment

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.

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.

Figure 6-11: Slider Bar Example

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

Figure 6-12: Slider Bar

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 6-11

slideButton : Slider button, as the rhombus shape shown in Figure 6-11. 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 6-11. 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.

6.8 Keyboard

6.8.1 Setup 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:

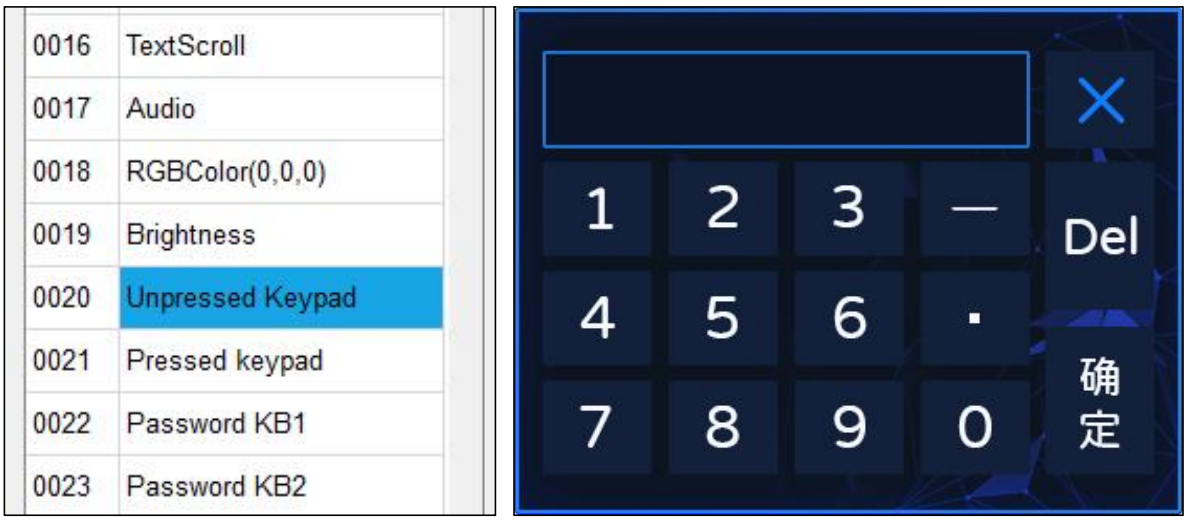


Figure 6-14: Add Keyboard Picture (unpressed state)

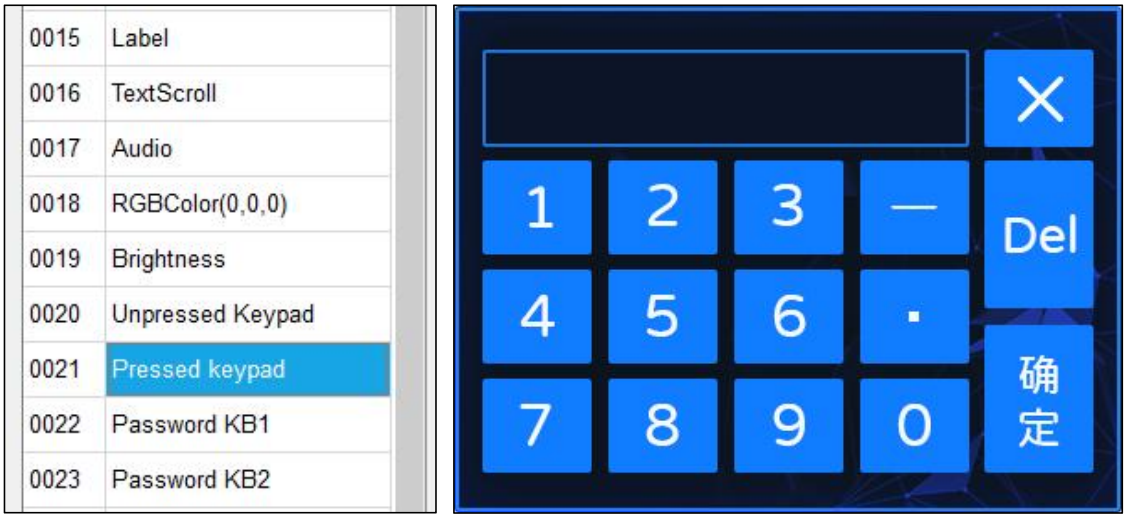


Figure 6-15: Add Keyboard Picture (pressed state)

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.

dataType	short
pageID	Page0020
fontColor	0x000000

Figure 6-18: Setup Keypad Widget

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

Figure 6-20: Numeric Keypad

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

(1) Left: 

(2) Right: 

cursorColor : Set the cursor color

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 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: Example of Input Range Limitation

intNum	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
long long	-2^63 ~ 2^63-1	18	18

Note: digit number = integerDigit + decimalDigit

6.8.4 EN_KeyBoard

ICON: 

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

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.

Parameter	Data
name	enkeyB_0
writeAddr	0xFFFF
wordLength	16
X	263
Y	108
W	205
H	106
fontID	
fontWidth	16
fontHeight	16
cursorColor	White
fontColor	0x000000
entryBox_X	0
entryBox_Y	0
pageID	
keyboard_X	0
keyboard_Y	0
inputMode	New
displayFormat	normal
reportToHost	Disable
backgroundPage	
hostControl	Disable
_triggerValue	0x0000

Figure 6-21: EN_Keyboard

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

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 Set the left-top coordinate of the PinYin

Parameter	Data
name	cnkeyB_0
writeAddr	0xFFFF
wordLength	16
X	316
Y	110
W	172
H	130
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	White
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
_triggerValue	0x0000

和 Y prompt character, as ❷ shown in Figure 6-23. The reference point (0, 0) is the left-top coordinate of the keyboard page.

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.

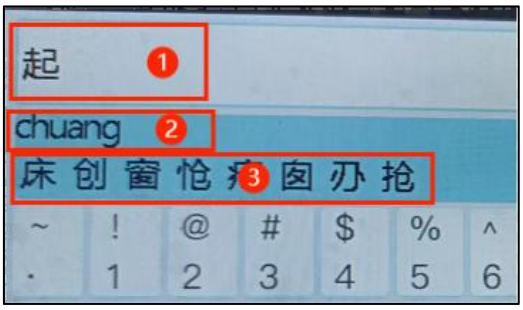


Figure 6-23：Example of PinYin Characters

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 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 last Chinese character list (for Pinyin Chinese Character)

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

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 String_Label

ICON: 

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: 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.

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

Figure 6-24: String_Label

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

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.

6.10 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.

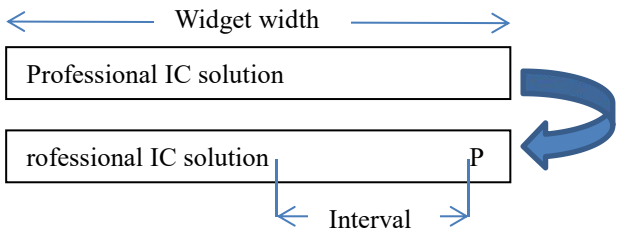
fontColor : Set the font color

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 6-25: 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.

6.11 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
parameterAddr	0xFFFF
writeAddr	0x0038
byteLength	2
X	274
Y	122
W	253
H	202
fontWidth	65535
fontID	
encoding	
alignment	Left
integerDigit	4
decimalDigit	0
dataType	short
unitSymbol	
_length	0
fontColor	0x000000
defaultNumber	0
leadingZero	Disable

Figure 6-26: 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.

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.

6.12 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
parameterAddr	0xFFFF
writeAddr	0x003C
byteLength	2
X	315
Y	116
W	234
H	230
integerDigit	4
decimalDigit	0
dataType	short
alignment	Left
firstIcon	
lastIcon	
defaultNumber	100
leadingZero	Disable

Figure 6-27: Graphics Number

alignment : Alignment mode. Options includes 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.

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.

6.13 Real Time Clock

6.13.1 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 6-30.

hourHand_S : The length of the hour hand on the shorter side. See example depicted in Figure 6-30.

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 6-29)

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	

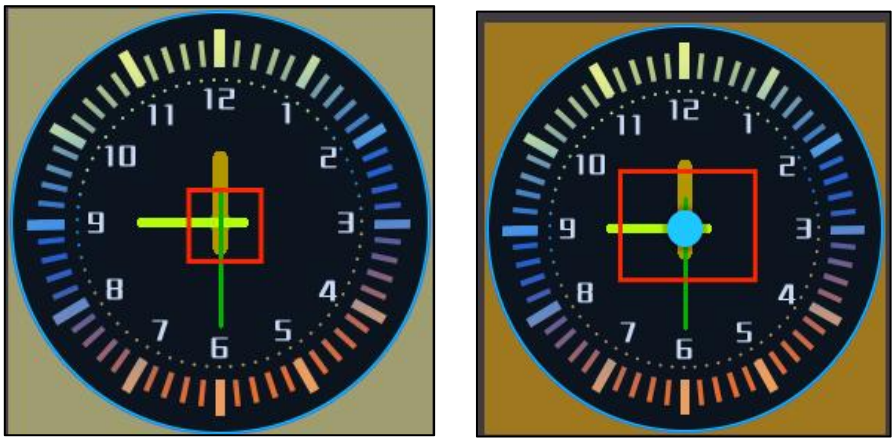


Figure 6-29: Example of Analog Clock

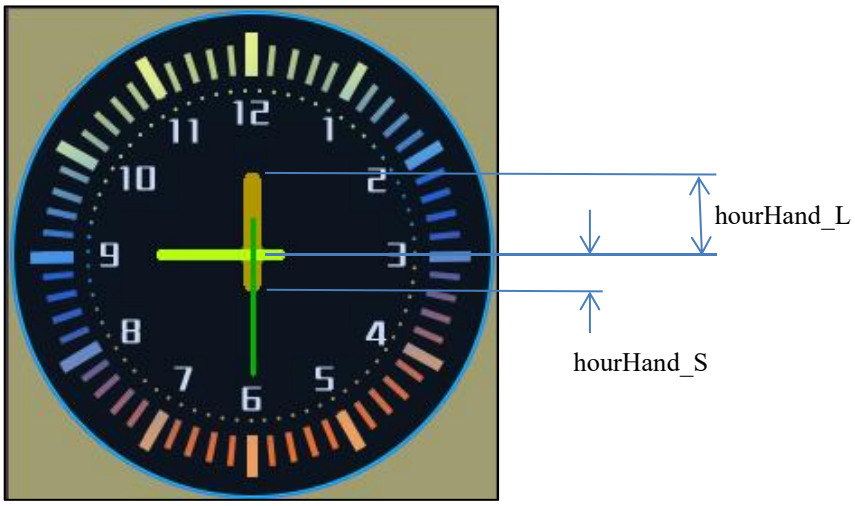


Figure 6-30: hourHand_L and hourHand_S

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 RTC circuit is available.

6.13.2 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 6-31: Digital Clock

displayFormat : Display options, as shown in Figure 6-32.

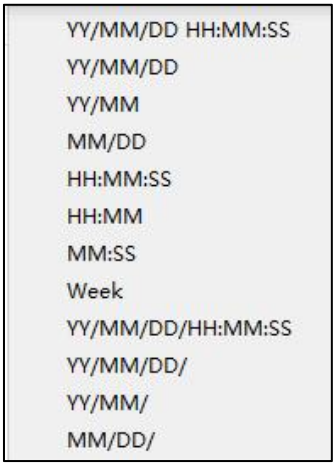


Figure 6-32: Display Options of Digital Clock

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, Thur, 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 RTC circuit is available.

6.13.3 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.

6.14 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.

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.

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

Figure 6-33: Timer

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

writeAddr0~7 : The address of the operation that will be executed after the counting is done.

_value0~7 : 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 − CalcValue, 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 RTC circuit is available.

6.15 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.

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 : Click to add Gif

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

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

Figure 6-34: Gif

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

effects	:	Set [Enable] if the Gif is overlapped with graphic number, icon (αPNG), or text widgets. Set [Disable] otherwise.
writeAddr0~7	:	The address of the operation that will be executed once the Gif is done playing.
_value0~7	:	The value to be assigned to the address of the operation once the Gif is done playing.

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. For LT268x & 269, when using PNG Gif, the gif resolution should meet the requirement of W*H <= 40000.
7. For LT168A&B and LT268x&LT269, Gif cannot be overlapped with PNG numbers or text widgets.
8. Refer to *UartTFT-II_Flash.bin* for the explanation about the size of the bin file converted from Gif.

6.16 QR Code

ICON: 

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 6-35: QRCode

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.

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.

6.17 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 6-36: 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 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)

6.18 Progress Bar

ICON: 

Function : To display a progress bar.

name : Name of the widget, user-definable.

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

writeAddr : Variable address of the progress bar.

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

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

bar_X & bar_Y : Left-top coordinate of the progress bar. The reference point (0,0) is the left-top coordinate of the background picture. If no background picture is used, then the two parameters must be set to 0.

direction : The progress direction, from small to large.

barIcon : The picture of the progress bar.

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

Figure 6-37: Progress Bar

minValue & maxValue : Define the range of the progress bar. -32768 ~ 32767

defaultValue : Default value (initial position) of the progress bar.

background : Assign the background picture.

6.19 Circular Progress Bar

ICON: 

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

Function : To display a circular progress bar.

name : Name of the widget, user-definable.

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

writeAddr : Variable address of the circular progress bar.

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

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

foreground : Assign a foreground picture.

background : Assign a background picture.

minValue & maxValue : Define the range of the progress bar. -32768 ~ 32767

defaultValue : Default value of the progress bar

startAngle : Start angle

finalAngle : Final angle

promptNum_X ,promptNum_Y : The coordinate of the number shown in the widget. The reference point (0, 0) is the left-top coordinate of the widget.

integerDigit : The digit number of the integer number

decimalDigit : The digit number of the decimal number

alignment : Alignment mode. Refer to *Circular Touch* for more detail.

fontID : Click to select a font

fontColor : Set font color

Figure 6-38: Circular Progress Bar

firstIcon : Select the picture of "0".

lastIcon : Select the picture of decimal point.

digitDisplayMode : [FontNum]: display font numbers; [IconNum]: display PNG numbers; [NULL]: not showing numbers

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

6.20 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
parameterAddr	0xFFFF
writeAddr	0x00AA
bitIndex	bit0
X	451
Y	111
W	144
H	253
offStateIcon	
onStateIcon	
overlap	Disable

Figure 6-39: 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.

overlap [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.

6.21 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
parameterAddr	0xFFFF
writeAddr	0x00AC
byteLength	2
X	535
Y	161
W	160
H	236
firstIcon	
lastIcon	
dataFormat	
defaultDisplayID	
minDisplayID	0
maxDisplayID	0
overlap	Disable

Figure 6-40: 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.

overlap : 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.

6.22 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 6-42, unit: pixel.

_referenceValue : The value represented by the baseline. Refer to Figure 6-42, 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 6-41: Trend Graph

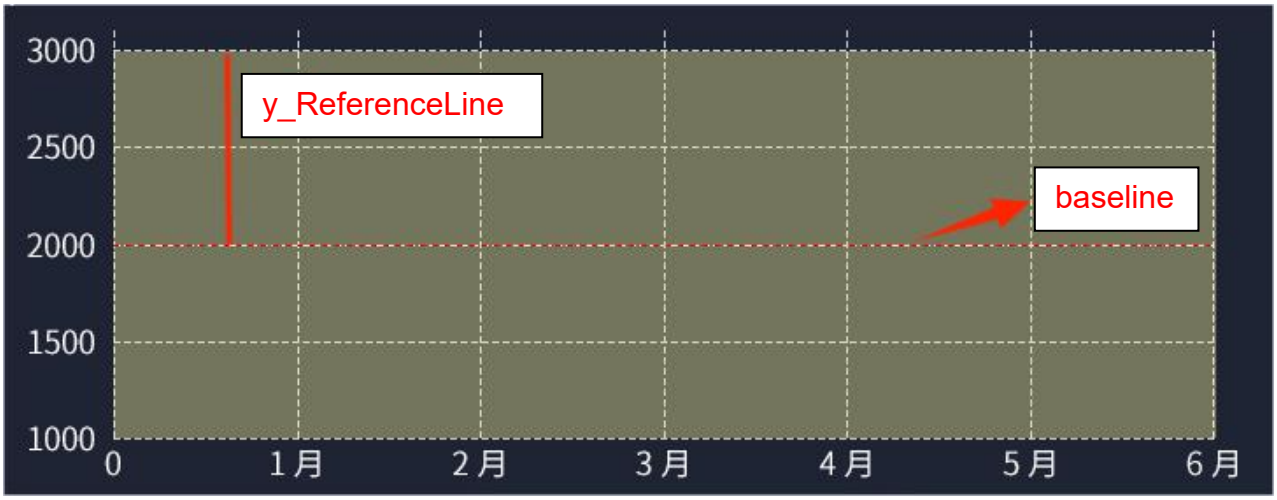


Figure 6-42: y_Reference Line and baseline

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 6-43, where host sends two data (0x00C8, 0x0064) to channels with different direction settings.

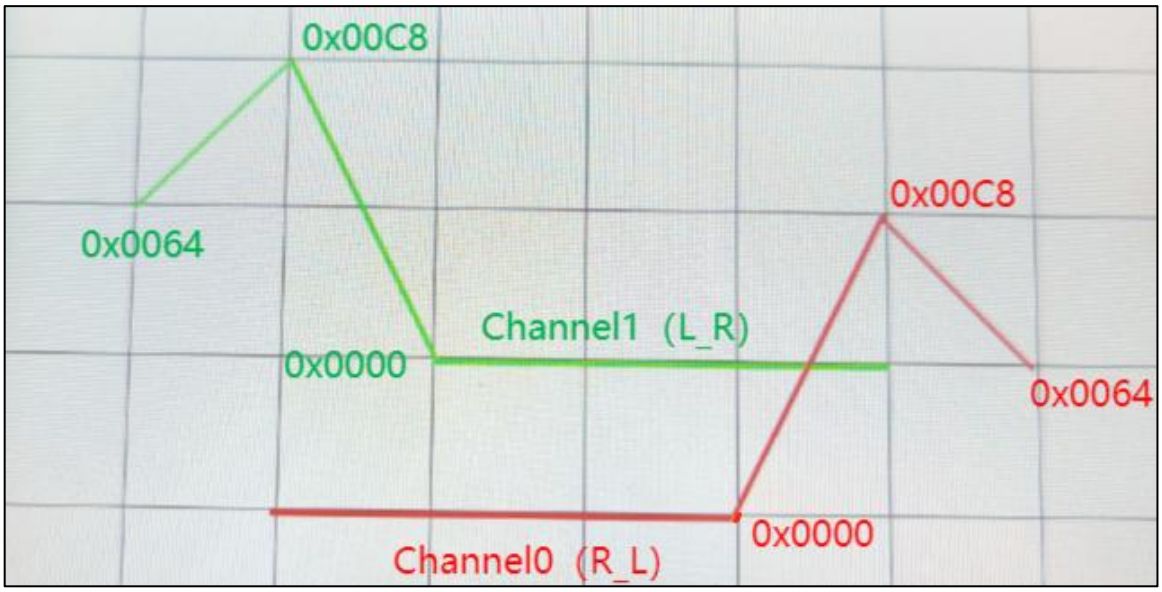


Figure 6-43: Example of Direction Settings

maxData : The maximum value that the widget area represents.

minData : The minimum value that the widget area represents.

As shown in Figure 6-44, based on the widget area, maxData is set to 3000, and minData is set to 1000.

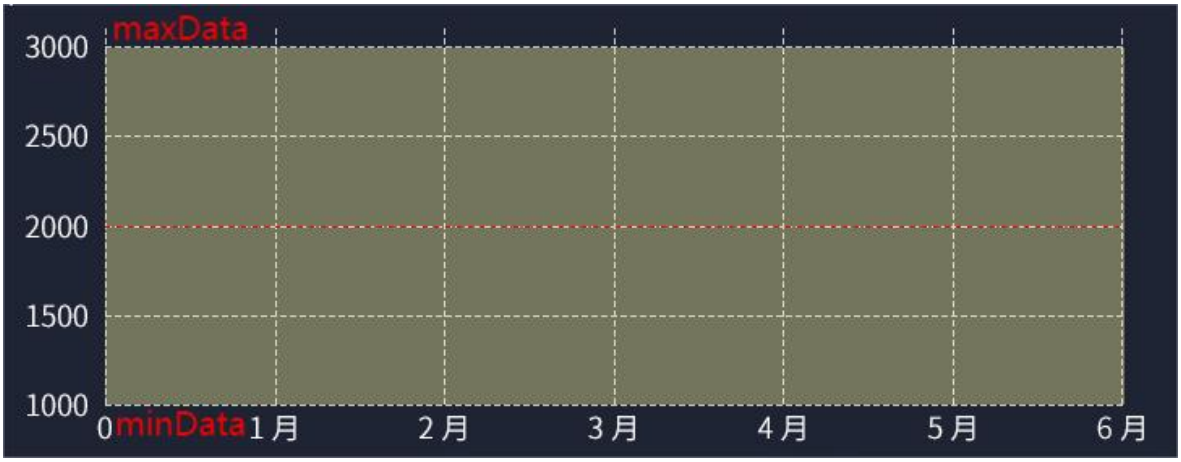


Figure 6-44: Example of maxData and minData

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.

6.23 Encoder

ICON: 

MCU Number	Encoder
LT7689 (7689)	Support
LT168A (168A)	Not Support

Function : To operate the display by an Encoder, instead of a touch panel. A knob part will be needed for implementation.

name : Name of the widget, user-definable.

writeAddr : Encoder Address. The icon used to control the encoder should be assigned to the same address.

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 4 options. Each page is allowed to have the most one encoder widget, which means each page may have the most 16 operations through the encoder widget.

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

Figure 6-45: Encoder

6.23.1 Encoder: Operation Principle

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 operation. 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 modes to apply the Encoder widget:

Mode 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.

Setting Example of Mode 1:

Figure 6-46 and Figure 6-47 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	25
Y	98
W	238
H	152
firstIcon	0350.png
lastIcon	
dataFormat	
defaultDisplayID	
minDisplayID	2
maxDisplayID	2
overlap	Disable

Parameter	Data
name	icon_1
parameterAddr	0xFFFF
writeAddr	0x5101
byteLength	2
X	281
Y	100
W	238
H	152
firstIcon	0351.png
lastIcon	
dataFormat	
defaultDisplayID	
minDisplayID	3
maxDisplayID	3
overlap	Disable

Parameter	Data
name	icon_2
parameterAddr	0xFFFF
writeAddr	0x5101
byteLength	2
X	542
Y	100
W	238
H	152
firstIcon	0352.png
lastIcon	
dataFormat	
defaultDisplayID	
minDisplayID	4
maxDisplayID	4
overlap	Disable

Figure 6-46: Icon Settings for Encoder Application (Mode 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 6-47: Encoder Settings (Mode 1)

Mode 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 paramerter 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.

Setting Example of Mode 2:

Figure 6-48 and Figure 6-49 illustrate 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

Figure 6-48: Icon Settings (Mode 2)

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 6-49: Encoder Settings (Mode 2)

There are 4 operating options of an encoder widget, including knob turn, click, double click, and long-press. Besides the operation of knob turn, developers may also utilize either [Click], [Double Click], or [Long-Press] to choose and execute the linked operation (item). Each item of an encoder has 4 sub-options (Mode1 ~ 4). Mode 1 ~ 3 can be operated by [Click], whereas, Mode 4 can be set to be operated by either [Click], [Double Click], or [Long-Press].

Note: The address of the Item parameter must NOT be set as same as the Encoder’ s writeAddr.

6.23.2 Encoder: Setup item parameter

Double click the data column (NULL) of the item0, a pop-up window will show up as Figure 6-50:

Mode: 4 options available

New: Create new operation based on the selected Mode.

Clear: Clear/delete the created Mode

OK: Confirm the modification

Cancel: Exit without modification

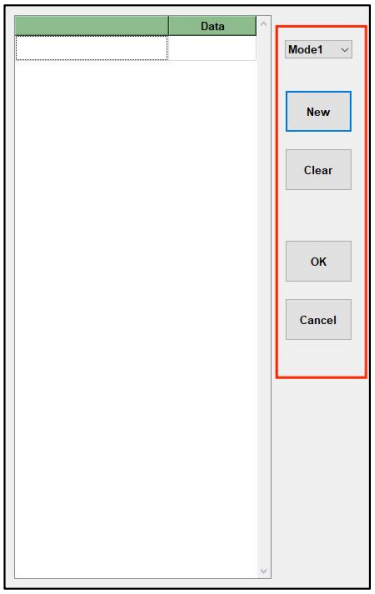


Figure 6-50: Setup item parameter

6.23.2.1 Encoder: Mode1

Function	:	(1) Jump to designated page; (2) Assign values to the designated addresses This function is triggered by clicking the knob.
controlMode	:	Function name, user-definable.
reportToHost	:	Set [Enable] to report writeAddr and its value through Uart port, set [Disable] otherwise. Refer to *Touch Returned Message* for more detail.
PageGoto	:	Set the target page to jump to.
writeAddr (0 ~ 7)	:	Assign the variable address
_value- (0 ~ 7)	:	Assign the variable value

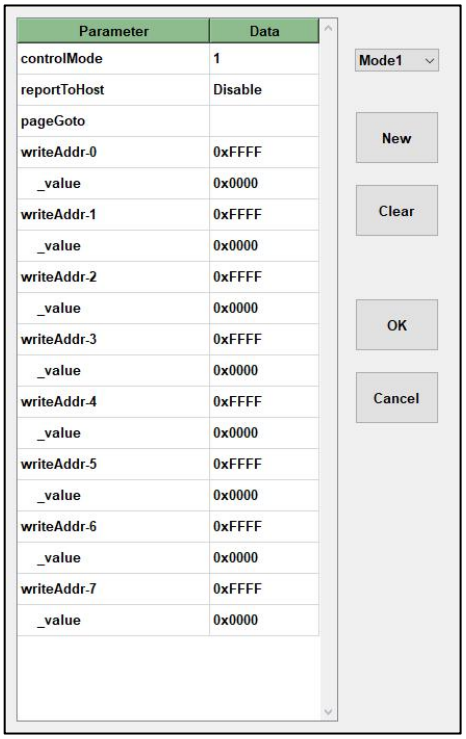


Figure 6-51: Encoder Function - Mode1

6.23.2.2 Encoder: Mode2

Function	:	Assign variable values to designated address when the knob of the encoder is clicked.
controlMode	:	Function name, user-definable.
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
minValue & maxValue	:	Adjustable range, from -32768 ~ 32767
adjStep	:	Incremental / decrement value of each clicking

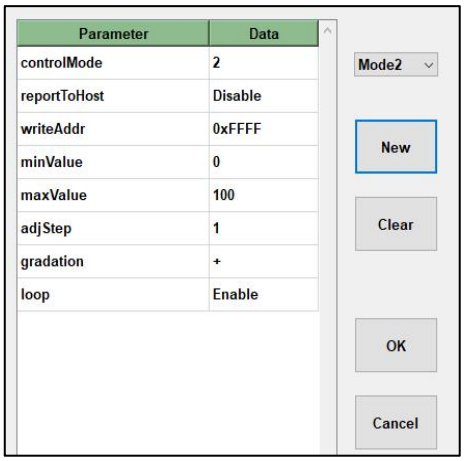


Figure 6-52: Encoder Function - Mode2

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.

6.23.2.3 Encoder: Mode3

Function : Assign variable values to designated address by turning the knob of the encoder.
Click to enter / exit the mode.

controlMode : Function name, user-definable.

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

minValue & maxValue : Adjustable range, from -32768 ~ 32767

adjStep : Incremental / decrement value when turning the knob.

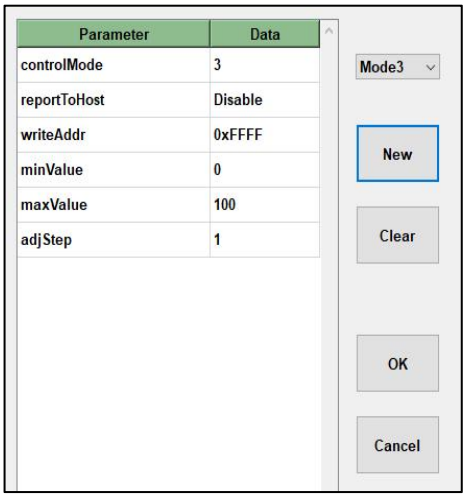


Figure 6-53: Encoder Function – Mode3

Note: The knob must be clicked once to enable the function, and then users may start adjusting the value by turning the knob. As soon as the adjustment is done, click the knob again to exit the function.

6.23.2.4 Encoder: Mode4

Function	:	Execute the designated operations. The function can be triggered by [Click], [Double Click] and [Long-Press]
controlMode	:	Function name, user-definable.
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(0~7)	:	Assign the variable address.
_value (0~7)	:	Assign the variable value.
doubleClickPageGoto	:	Set the target page to jump to, when [Double Click].
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 (0~7)	:	Assign the variable address.
_value (0~7)	:	Assign the variable value.
longPressPageGoto	:	Set the target page to jump to, when [Long-Press]

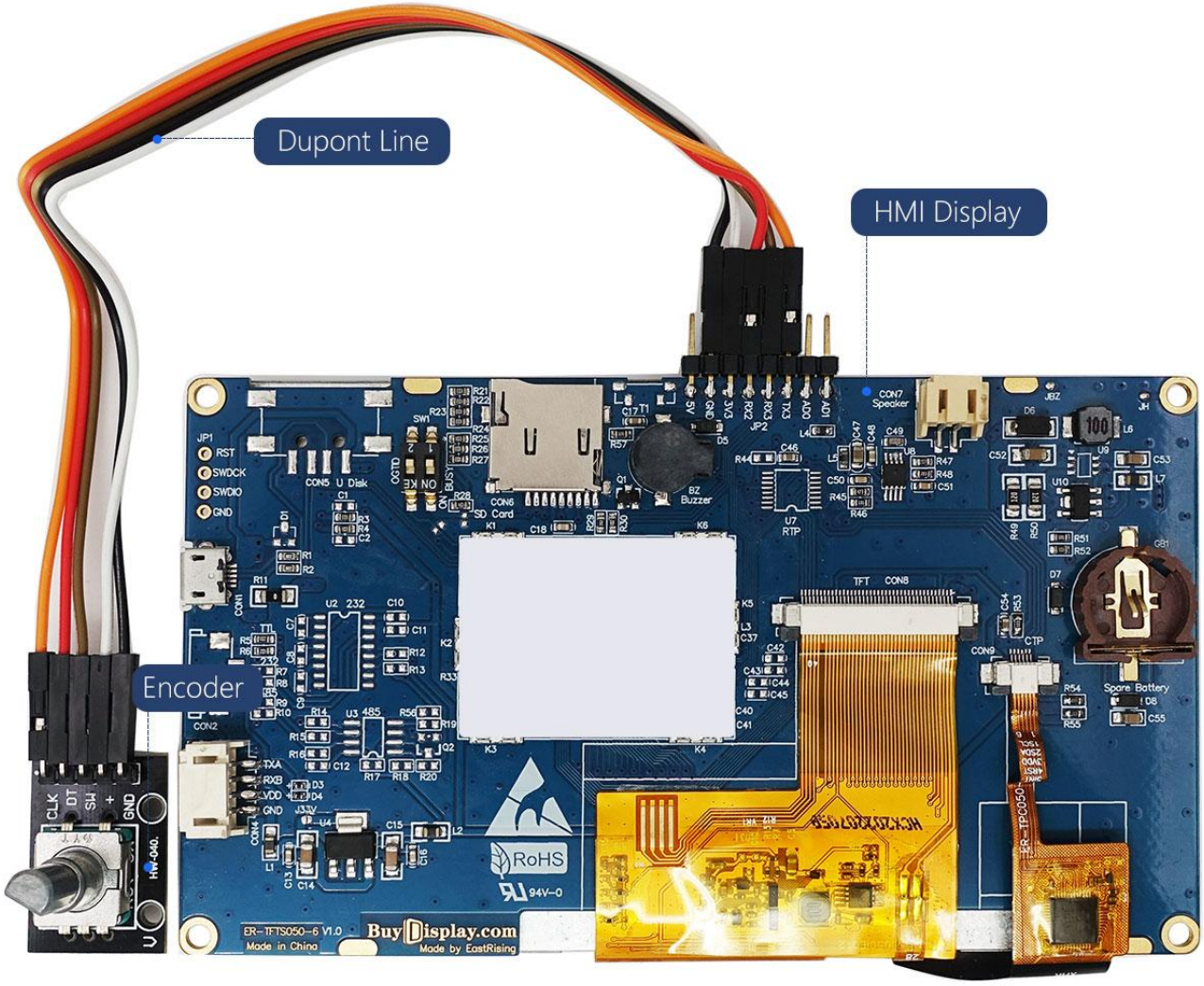
Parameter	Data
controlMode	4
singleClickPageGoto	
singleClickReport	Disable
singleClick_Addr0	0xFFFF
_value	0x0000
doubleClickPageGoto	
doubleClickTimeGap(50ms)	2
doubleClickReport	Disable
doubleClick_Addr0	0xFFFF
_value	0x0000
longPressPageGoto	
longPressDuration(50ms)	20
longPressReport	Disable
longPress_Addr0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

Figure 6-54: Encoder Function – Mode4

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 (0~7)	:	Assign the variable address
_value (0~7)	:	Assign the variable value

6.23.3 Connect Encoder to HMI Display

Use DuPont line to connect the encoder interface of HMI display with encoder board.



6.24 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.)

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.

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

Figure 6-57: Automatic Variable

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 0~7 : Variable address

_value : The value to be assigned to the corresponding writeAddr. After the widget is executed, the value will be assigned to the designated writeAddr.

6.25 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.

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

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.

Figure 6-58: 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.

6.25.1 Parameter: step

When the **needleType** is set to Animation, UI_Editor-II 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:

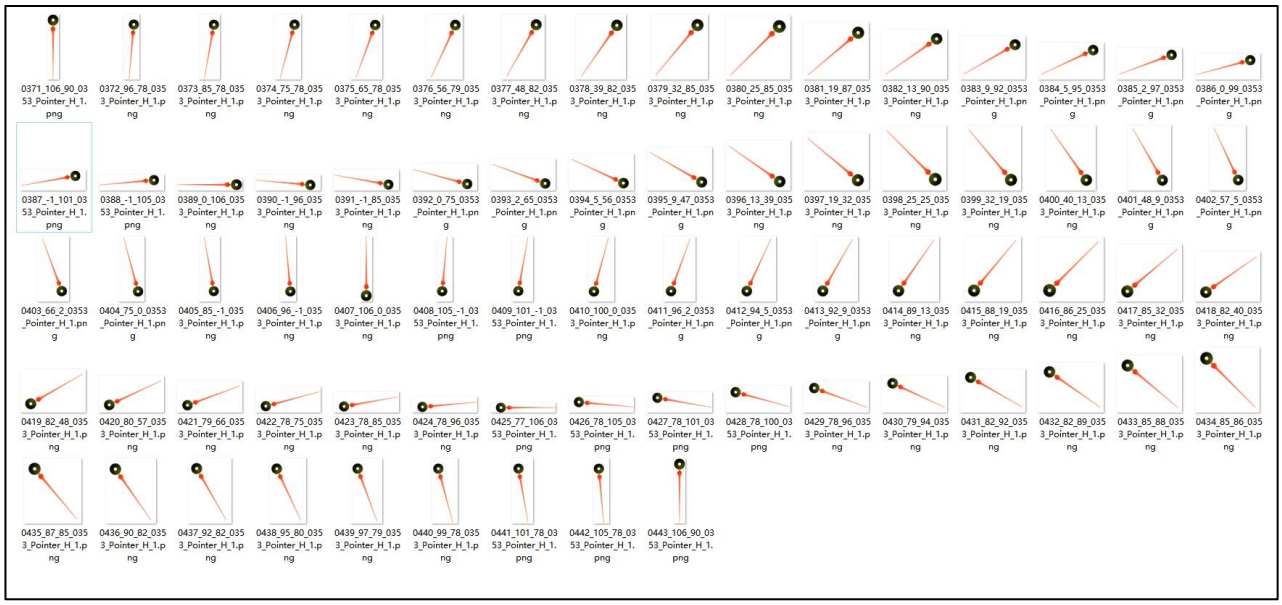


Figure 6-59: Needle icons with different angles

6.25.2 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°

6.25.3 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 needle icon 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.

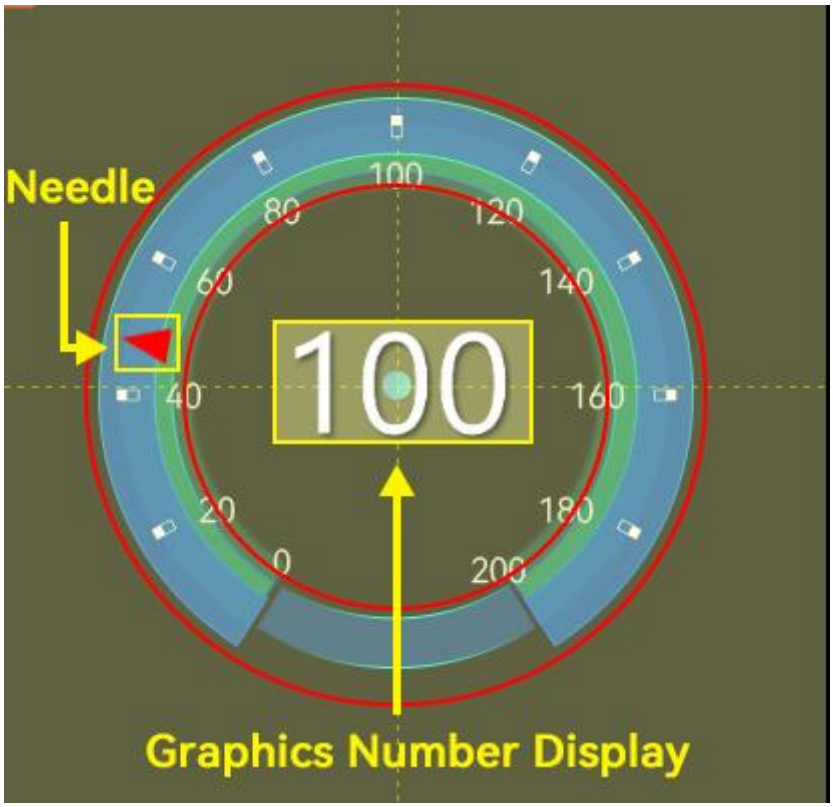


Figure 6-60: Needle – Icon type

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.

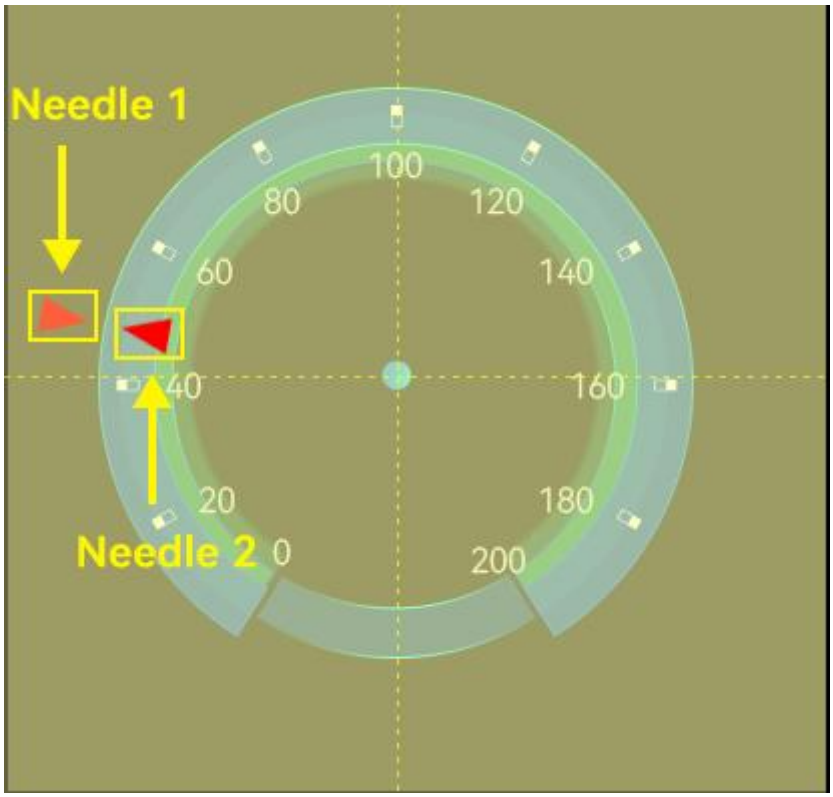


Figure 6-61: Multiple Needles – Icon type

6.25.4 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.

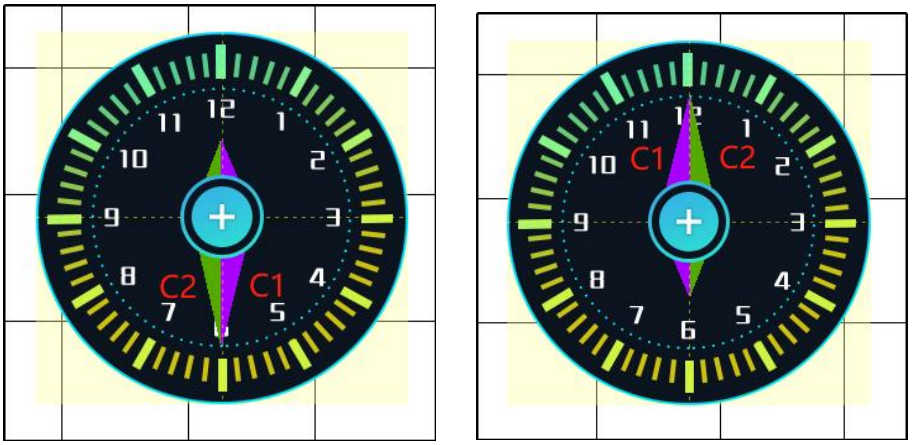


Figure 6-62: Needle Color

6.25.5 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.

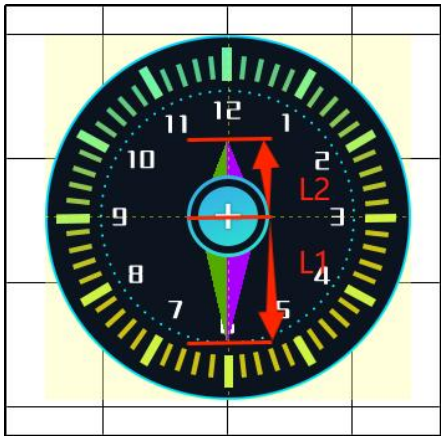


Figure 6-63: Needle Length

6.26 Layout Widgets

Figure 6-64: 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.

6.26.1 Left_Align

ICON: 

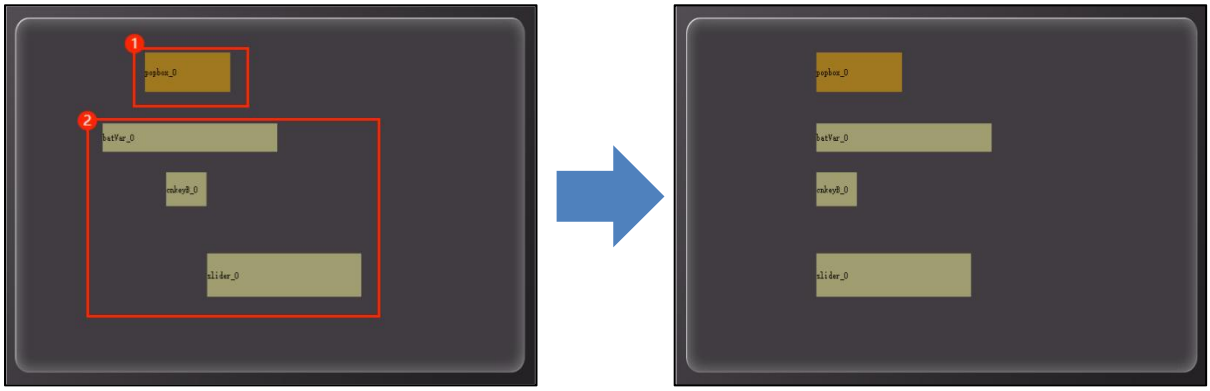


Figure 6-65: Example of Left_Align

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.

6.26.2 Right_Align

ICON: 

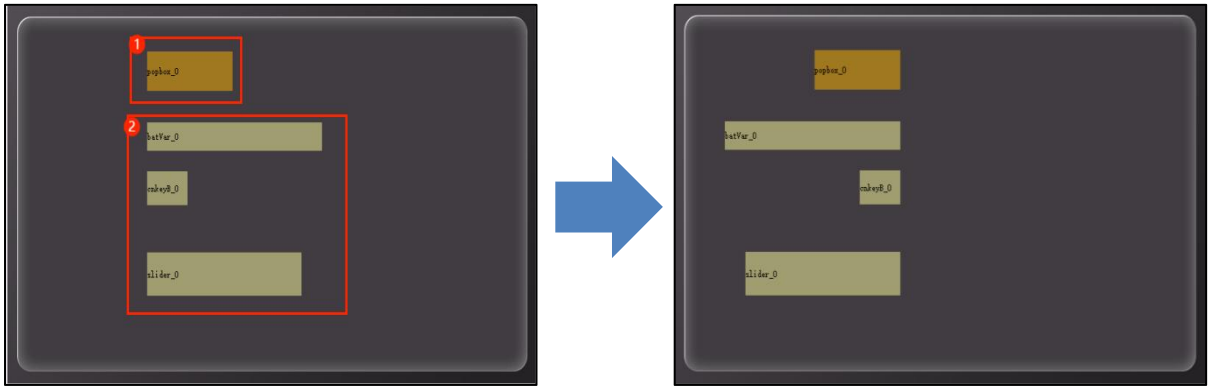


Figure 6-66: Example of Right_Align

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.

6.26.3 Top_Align

ICON: 

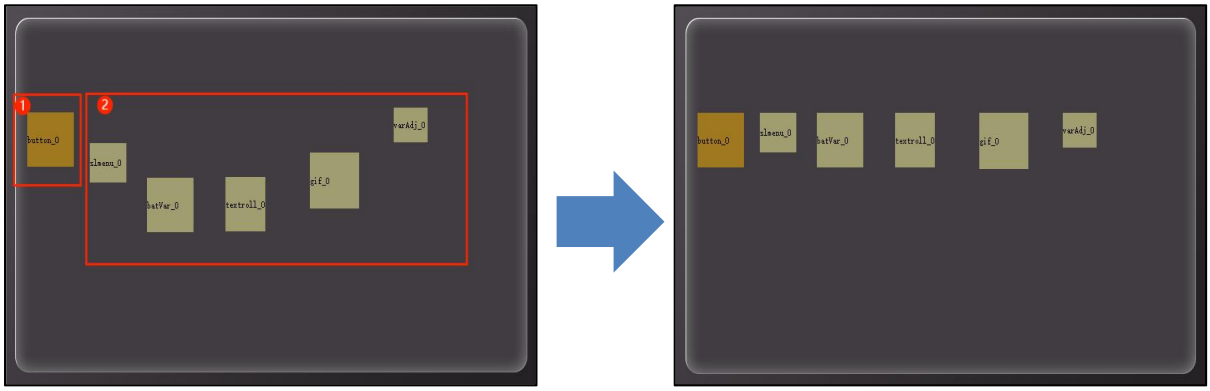


Figure 6-67: Example of Top_Align

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.

6.26.4 Bottom_Align

ICON: 

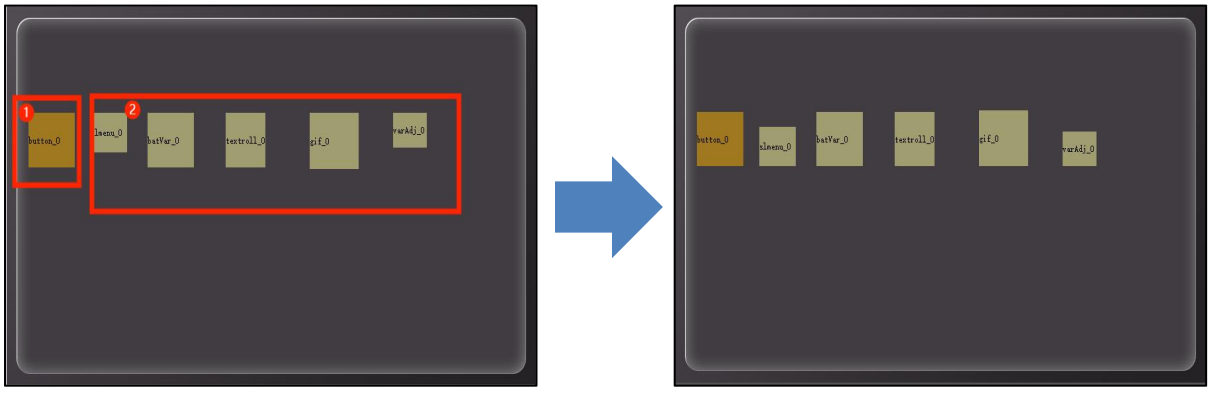


Figure 6-68: Example of Bottom_Align

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.

6.26.5 Width_Align

ICON: 

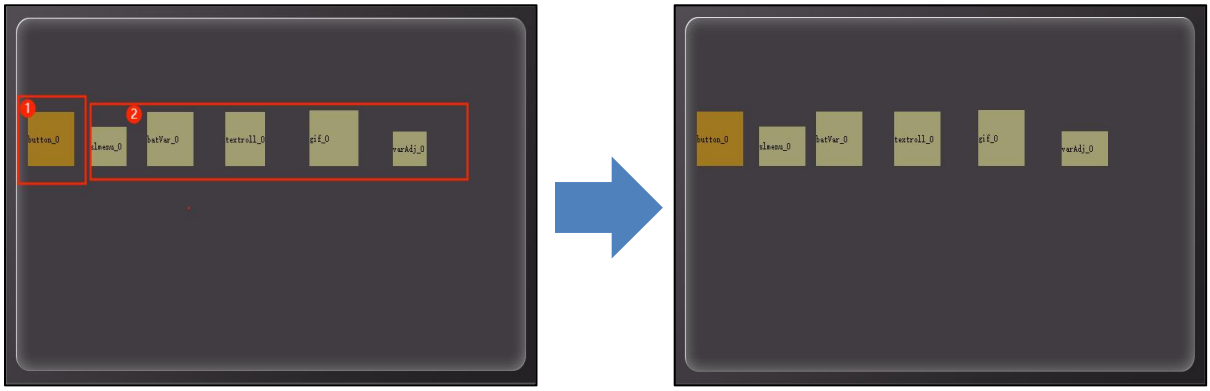


Figure 6-69: Example of Width_Align

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.

6.26.6 Height_Align

ICON: 

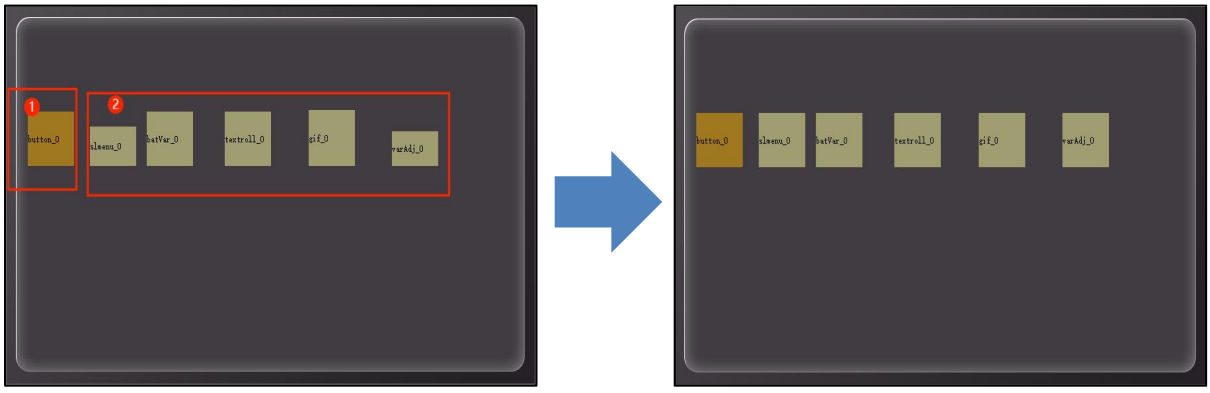


Figure 6-70: Example of Height_Align

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.

6.26.7 Shape Consistent

ICON: 

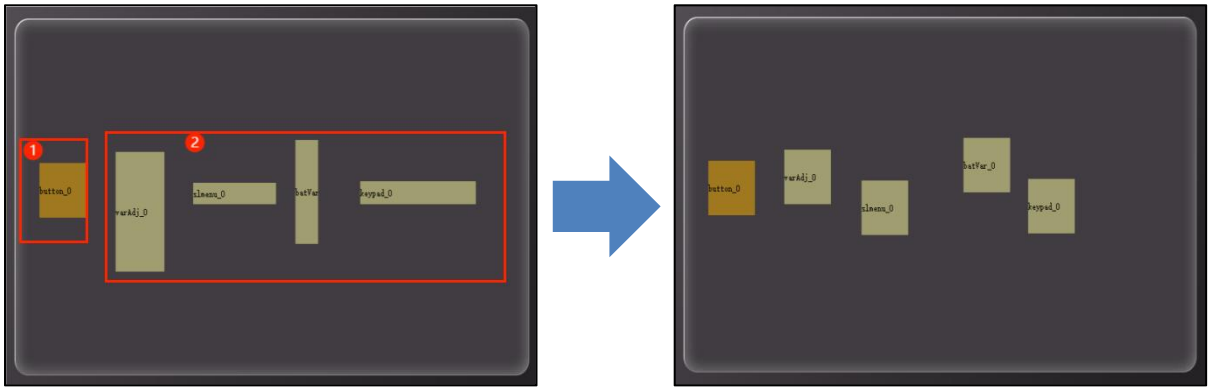


Figure 6-71: Example of Shape Consistent

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.

6.26.8 Horizontal Equidistance

ICON: 

Function: To reallocate the selected widgets in horizontal equidistance.

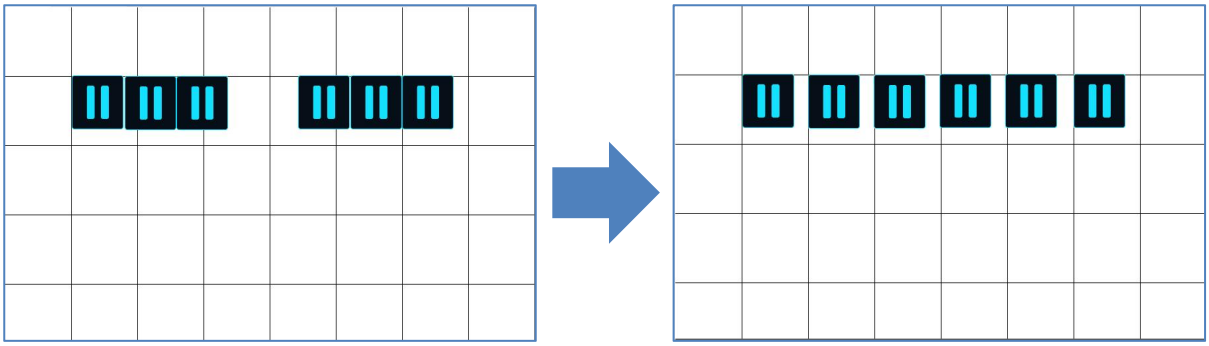


Figure 6-72: Example of Horizontal Equidistance

6.26.9 Vertical Equidistance

ICON: 

Function: To reallocate the selected widgets in vertical equidistance.

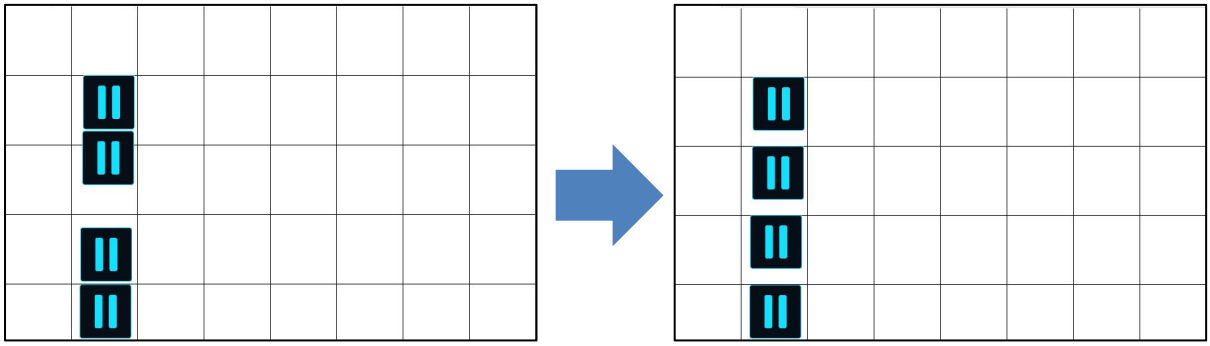


Figure 6-73: Example of Vertical Equidistance

6.26.10 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 6-74: Zoom In

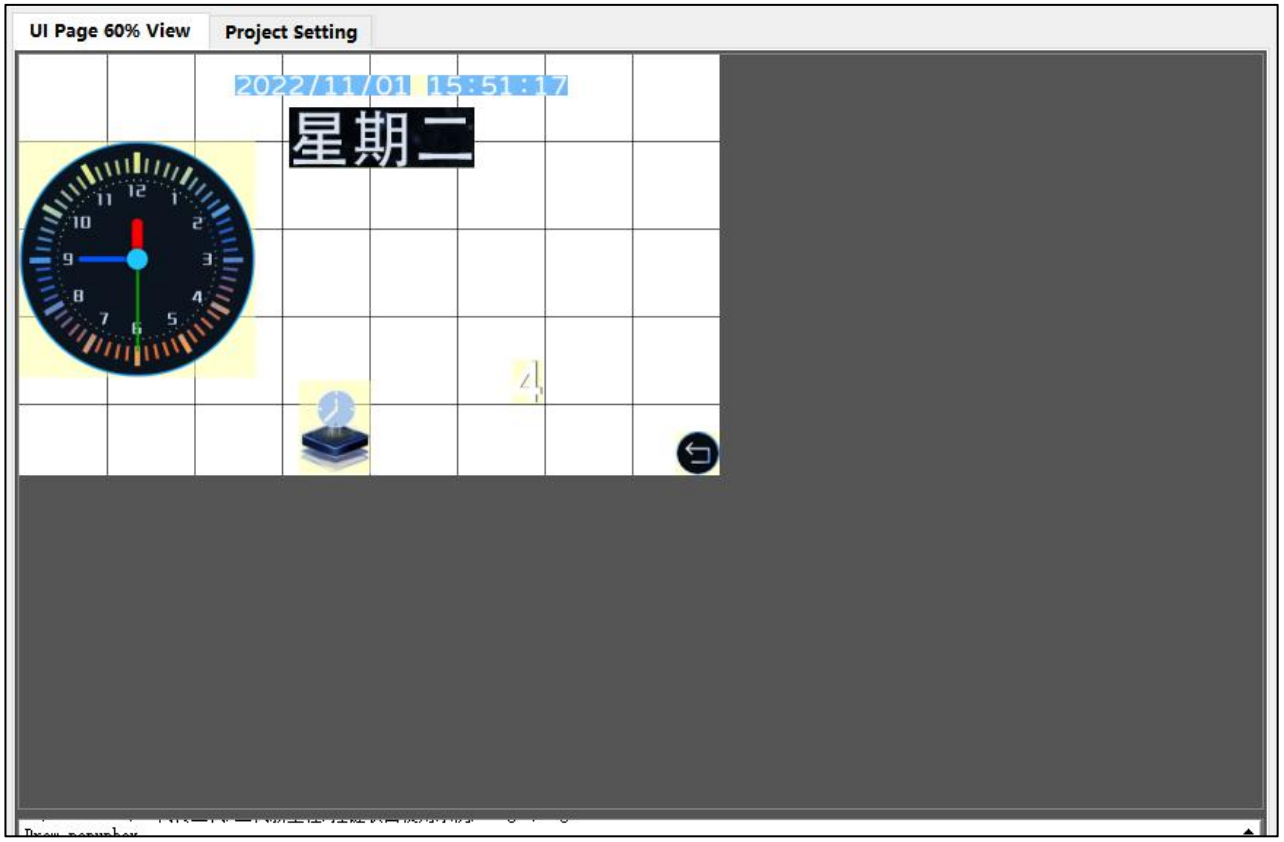


Figure 6-75: Zoom Out

7 Variable Address

7.1 RAM

MCU Number	RAM	Address Range
LT7689 (7689)	48KB	s 0x0000 ~ 0x5FFF
LT168A (168A)	16KB	0x0000 ~ 0x1FFF

In UI_Editor-II, 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, 旭日东方, 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	旭日东方

Variable Address	Stored Data	Chinese
0x1000	0xD0F1	旭
0x1001	0xC8D5	日
0x1002	0xB6AB	东
0x1003	0xB7BD	方

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 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
7002	Year	10 ~ 99
7003	Month	01 ~ 12
7004	Day	01 ~ 31
7005	Hour	00 ~ 23
7006	Minute	00 ~ 59
7007	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
7008	0	1		2					Year
			3						Month
									Day
		4				5			Hour
				6					Minute
									Second

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. Video_play : Address 0x7012~7027, used for controlling the video widget. Refer to *Video Registers – 0x7012 ~ 0x7027* for more details.

14. Multi-Language : Address 0x703F, write designated value to switch languages.

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-II_V2.30 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 first (1) Set the number of the languages used in Project Setting page; (2) create the icons of different languages; (3) store the icons in 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.

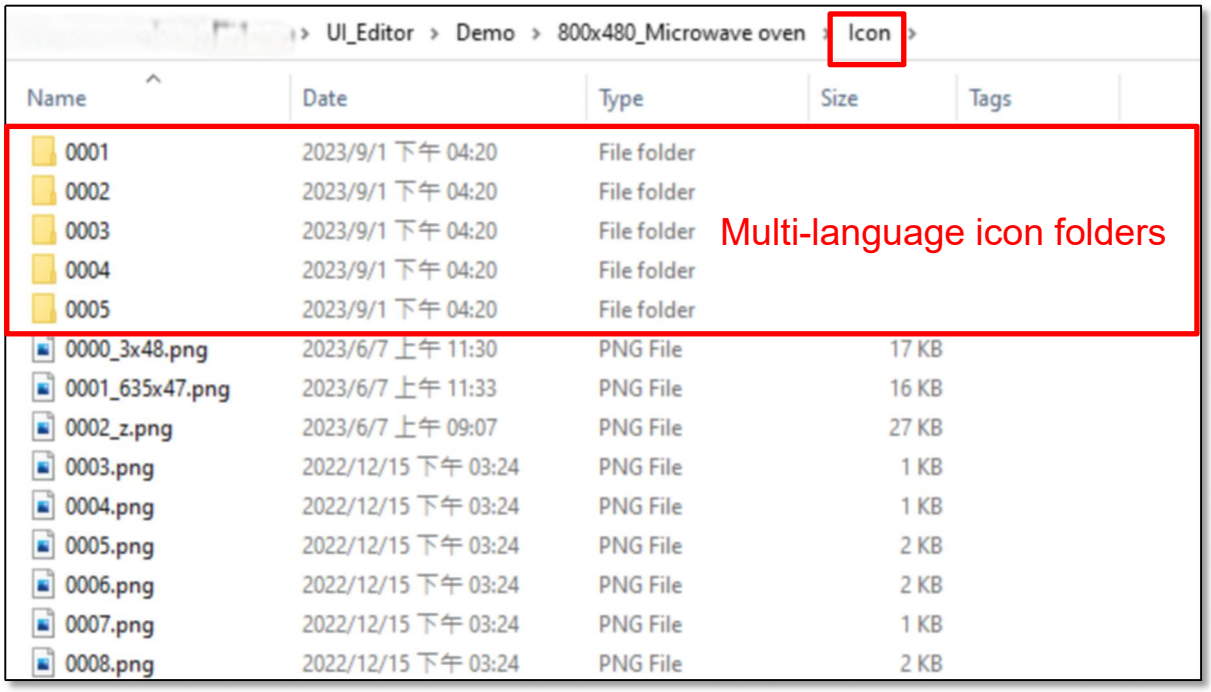


Figure 8-1: Setup Icon Folders for Multi-Language

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.

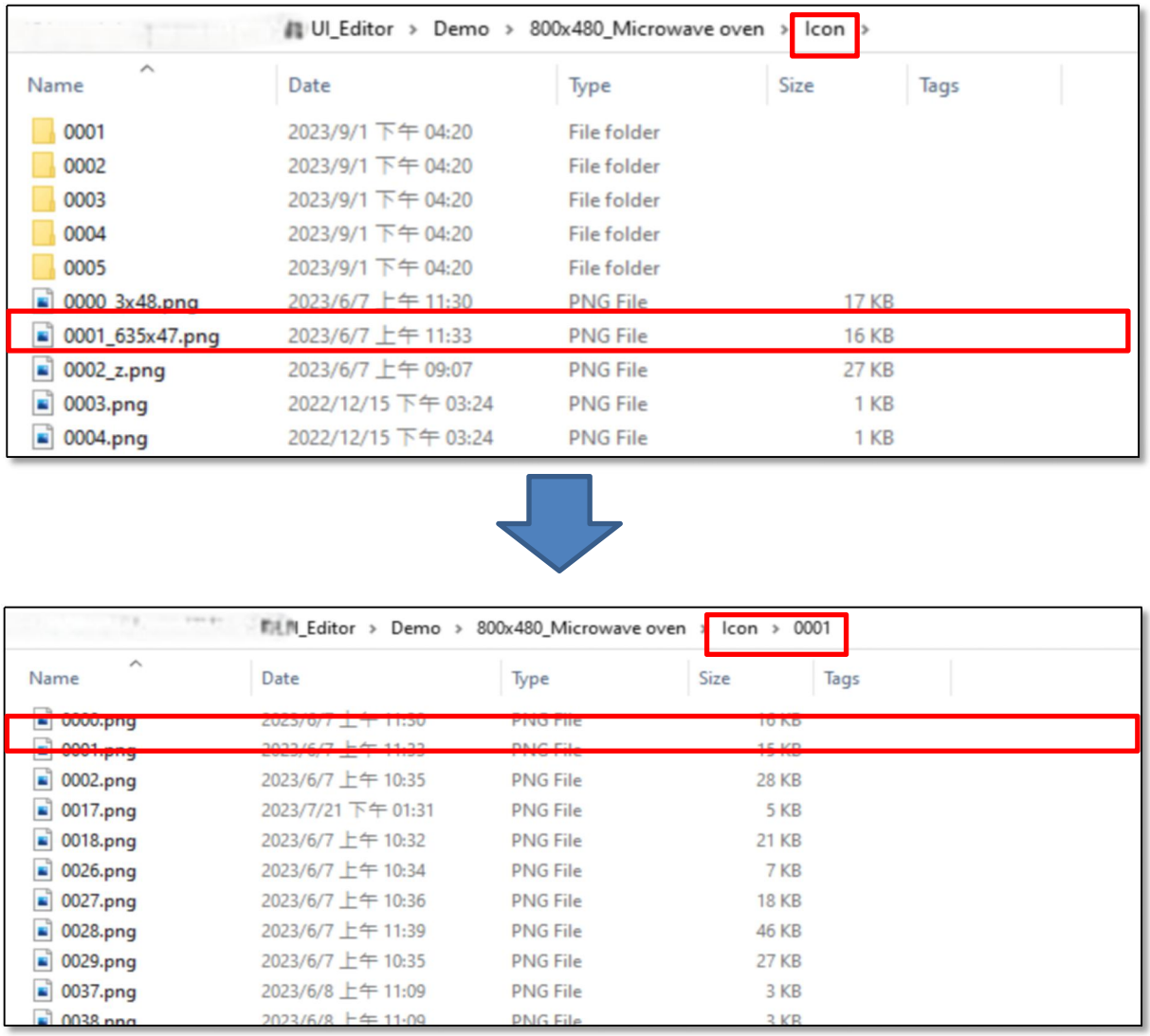


Figure 8-2: Icons of Different Languages

8.1.3 Wigets 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.

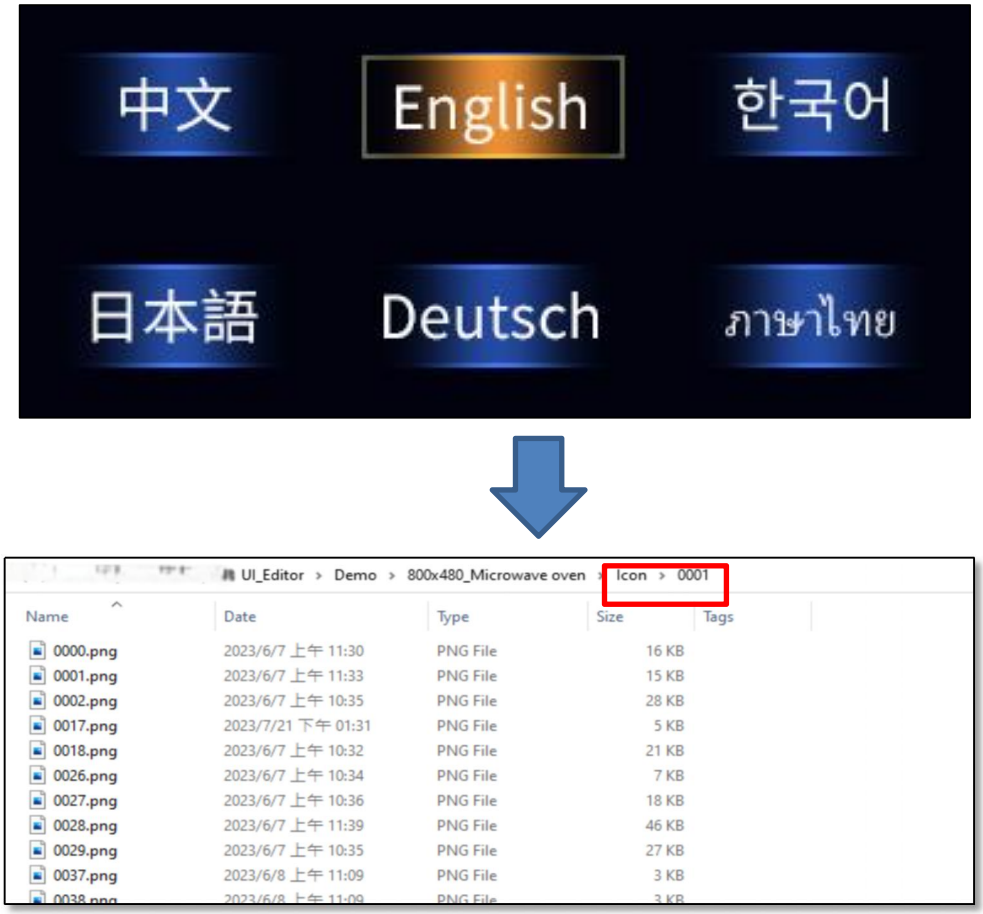


Figure 8-3: Switching to English Icons

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:

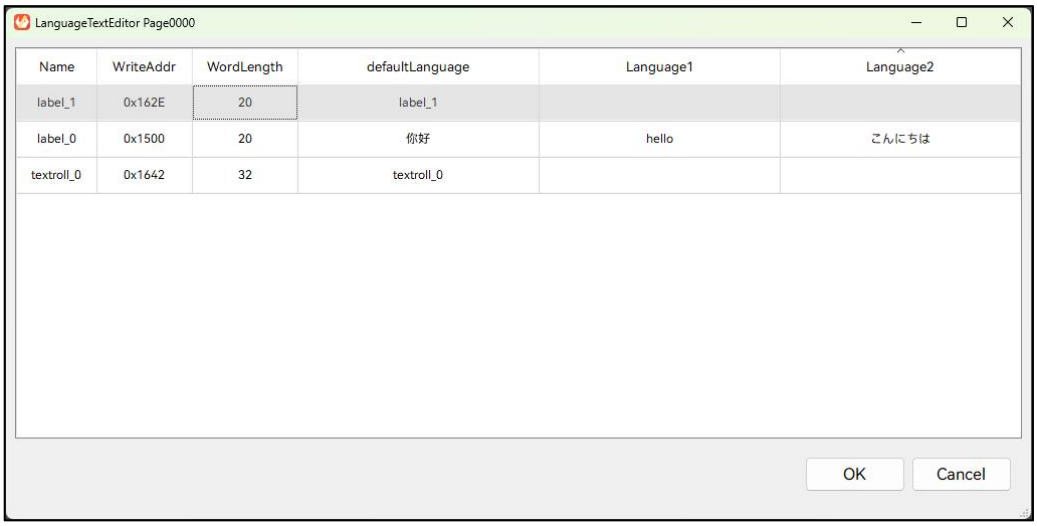


Figure 8-4: Input Multi-Languages

3. Enter the texts in corresponding languages to the entry boxes. Note that each character is represented by 2Bytes of data in Unicode. For example, 'A' is represented by 0x0041.

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

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

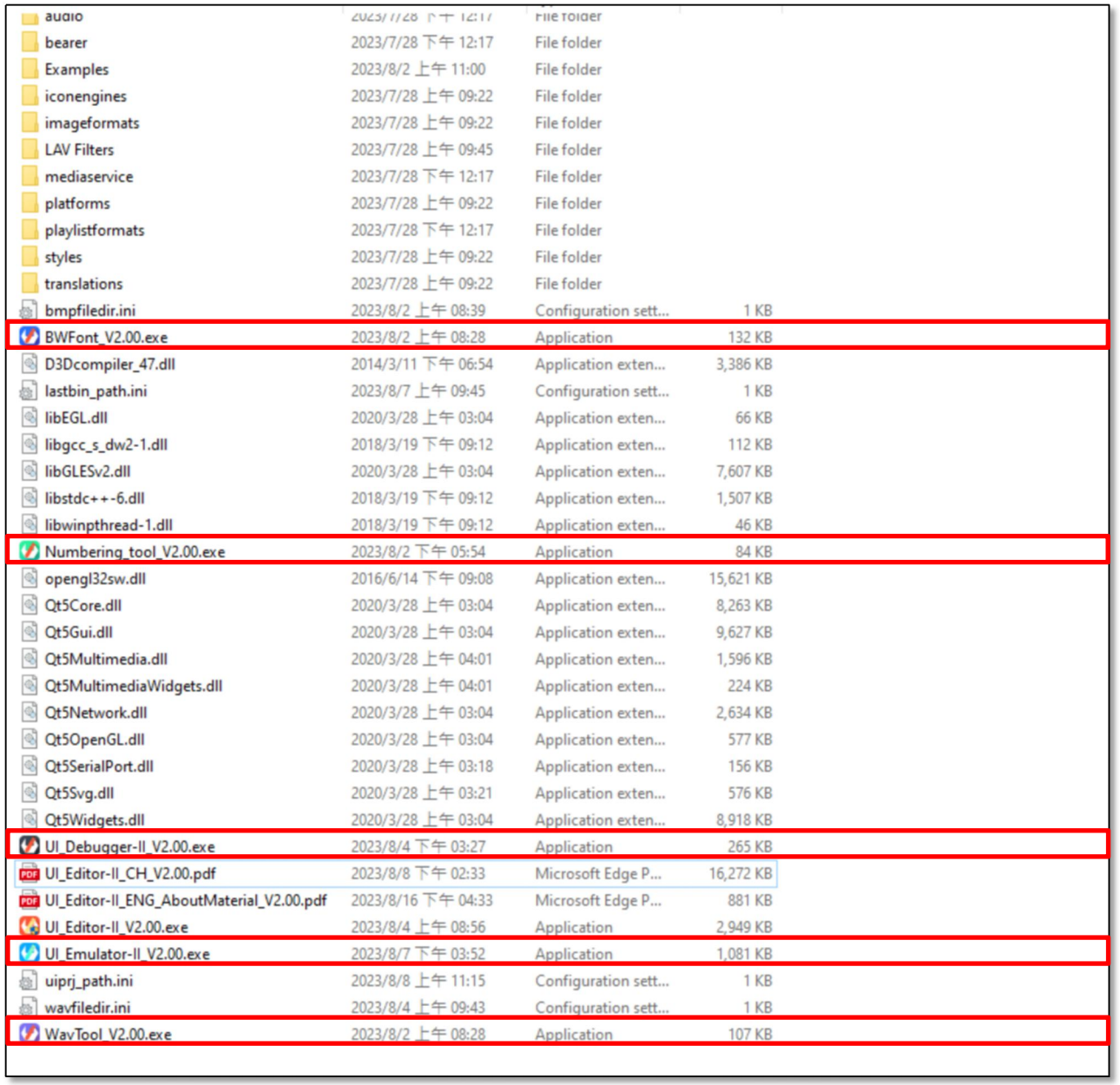


Figure 9-1: Tools for UI_Editor-II

9.1 UI_Emulator-II

9.1.1 Activate UI_Emulator-II

UI_Emulator-II 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-II will automatically import UartTFT-II_Flash.bin to start the emulation. Note UartTFT-II_Flash.bin will be generated after the UI project is compiled.

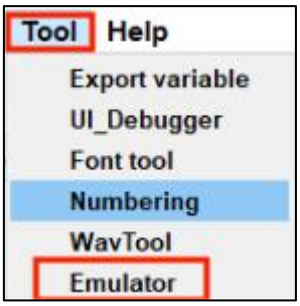


Figure 9-2: Activate UI_Emulator-II (1)

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

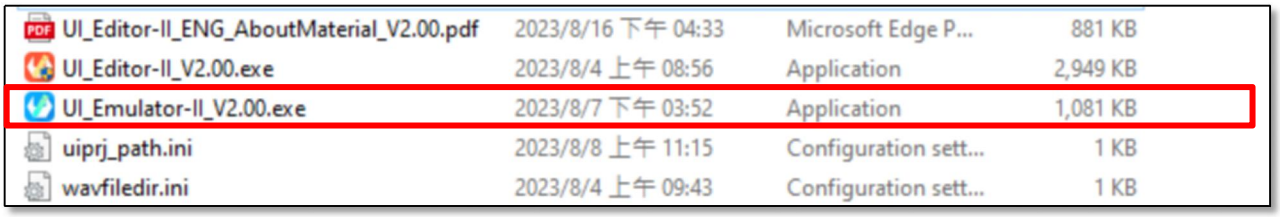


Figure 9-3: Activate UI_Emulator-II (2)

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

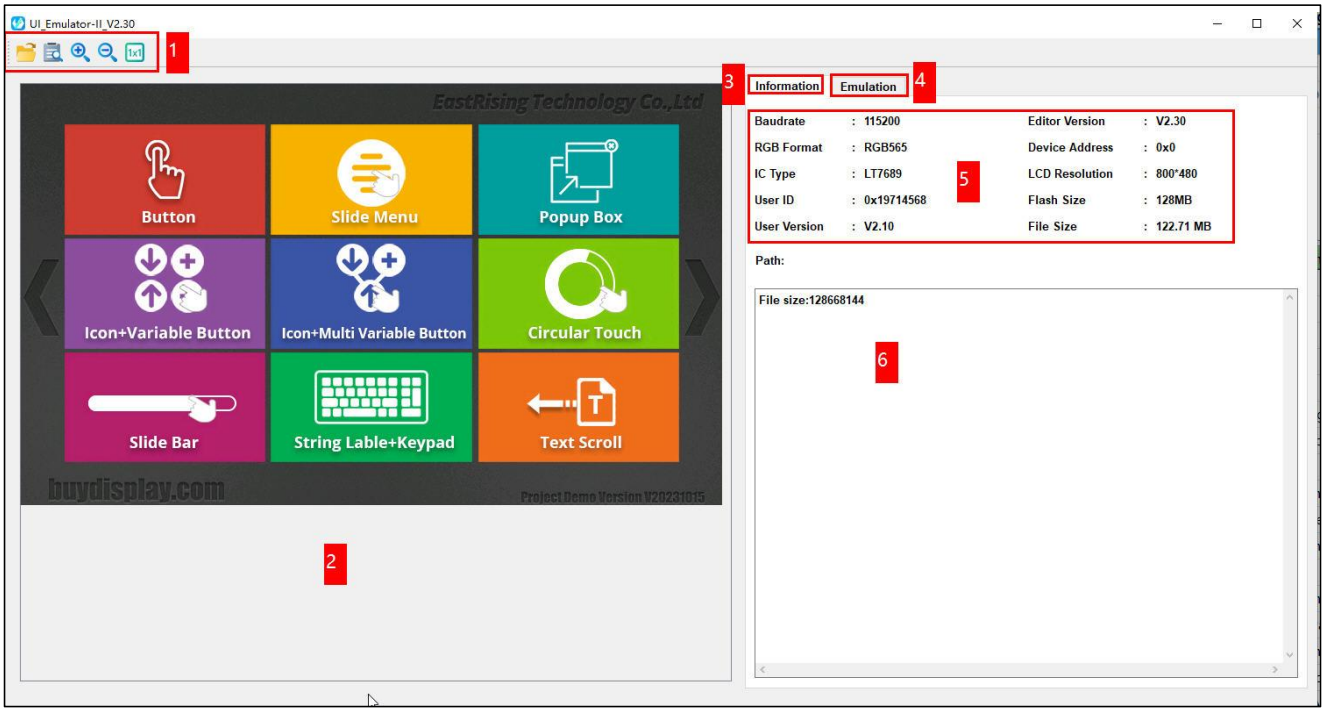


Figure 9-5: UI_Emulator-II Main Screen

❶ **Function bar:** For importing UartTFT-II_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 project information.

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

❺ **Information area:** Display project information for quick review

❻ **Operation record:** For listing the import/operation record.

9.1.2 Variable Operation

Click on [Emulation] to enter variable operation page:

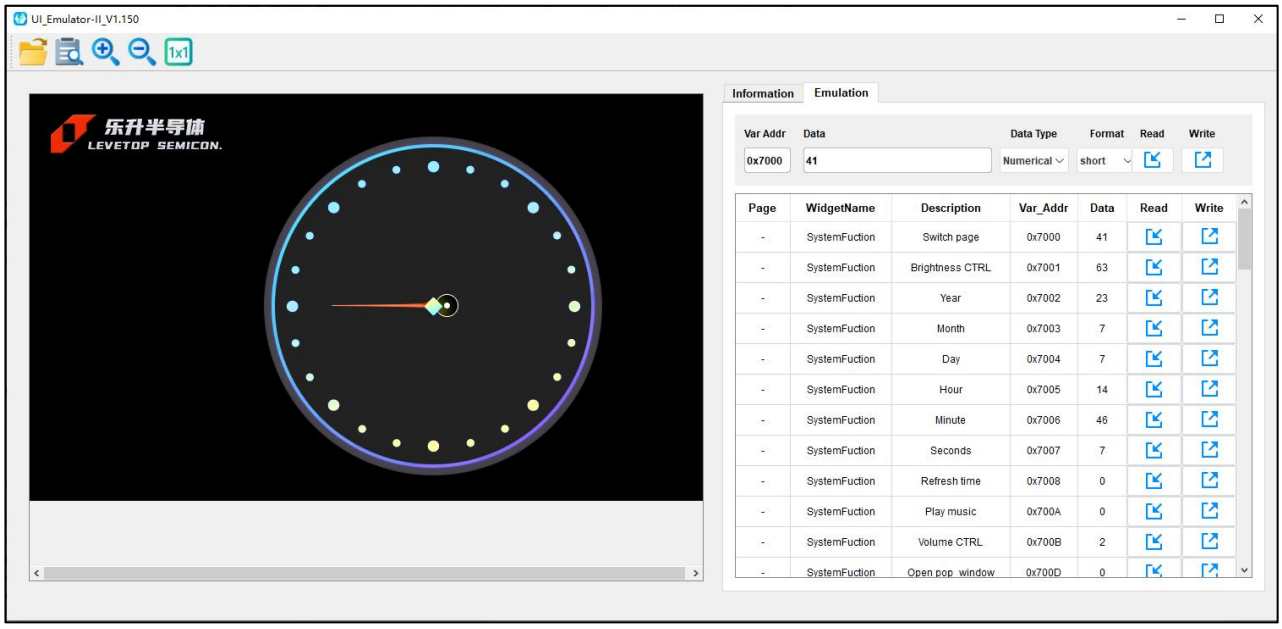


Figure 9-6: Variable Operation Page

9.1.2.1 Setting Bar

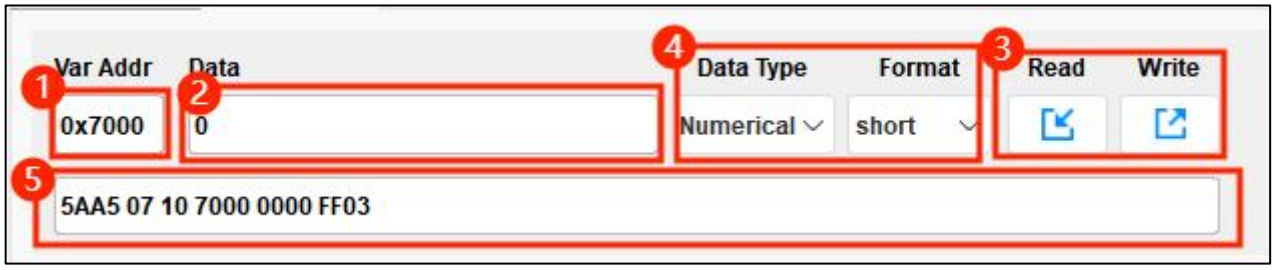


Figure 9-7: Setting Bar

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-II* for more details.

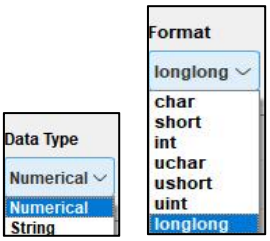


Figure 9-8：Numerical Data Type

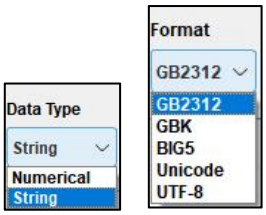
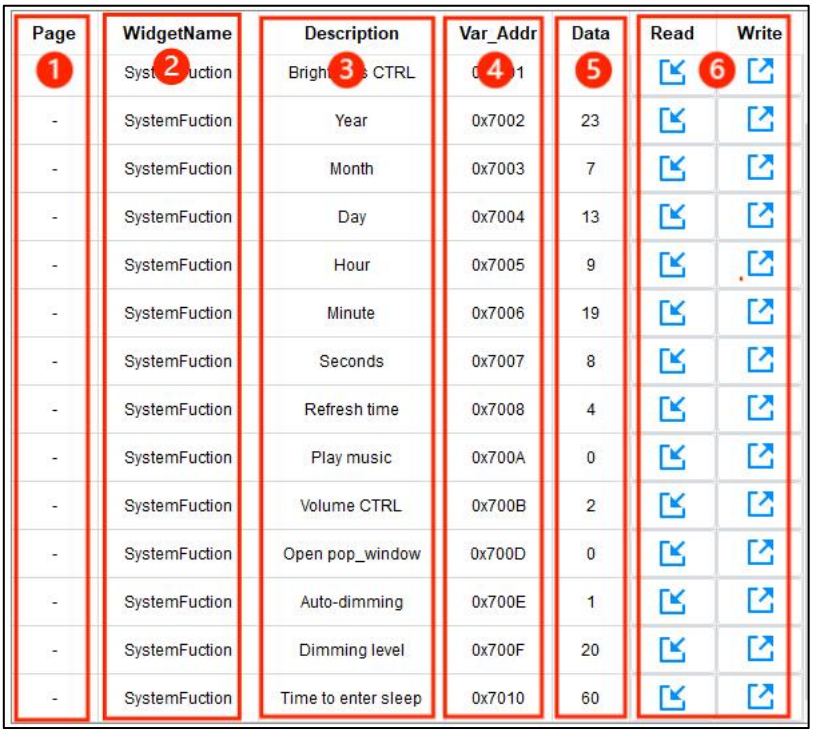


Figure 9-9：String Data Type and Format

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-II 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:

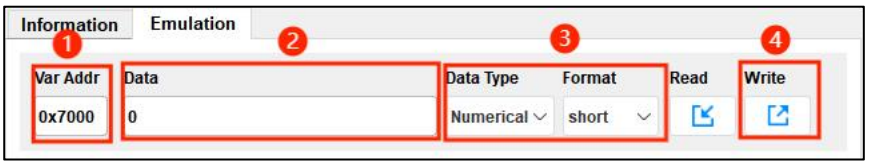


Figure 9-11: Write Numeric Data to Variable Address

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 Encoders Emulation

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-II 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-II_Flash.bin to be loaded by UI_Emulator-II. 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.)

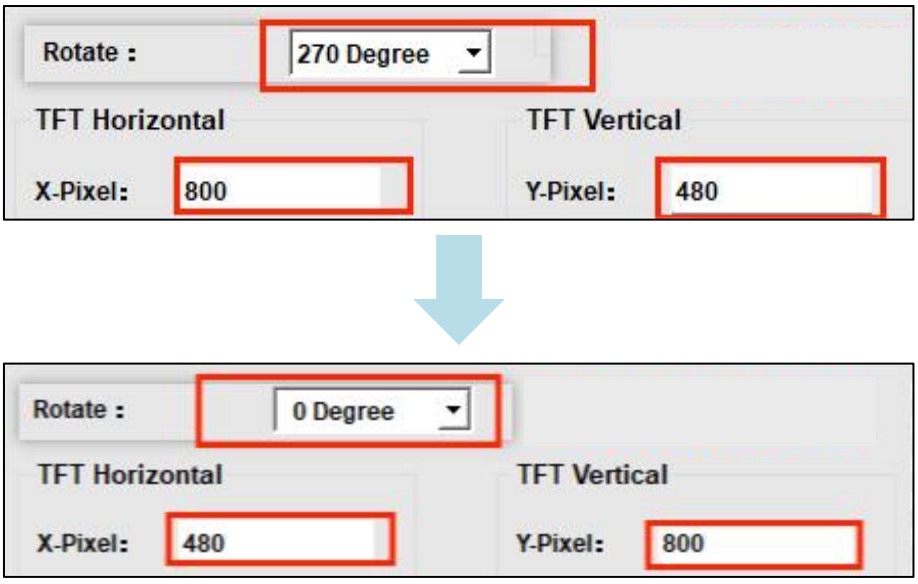


Figure 9-13: Reset Angle and Switch X/Y Resolution

9.1.6 Limitations of UI_Emulator-II

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

2、Key with beep – not supported.

9.1.7 Sending Data by UI_Emulator-II

Table 9-1: Sending Data by UI_Emulator-II

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
Multi-Variable Button	-	-	Audio Play	-	-
Circular Touch	2	-	Progress Bar	2	short
Slider Bar	2	-	Circular Progress 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	Camera	2	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			

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

9.2 UI_Debugger-II

UI_Debugger-II 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:

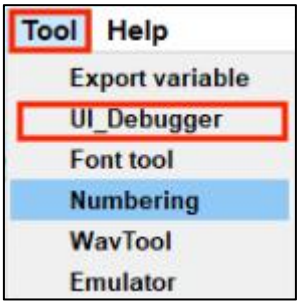


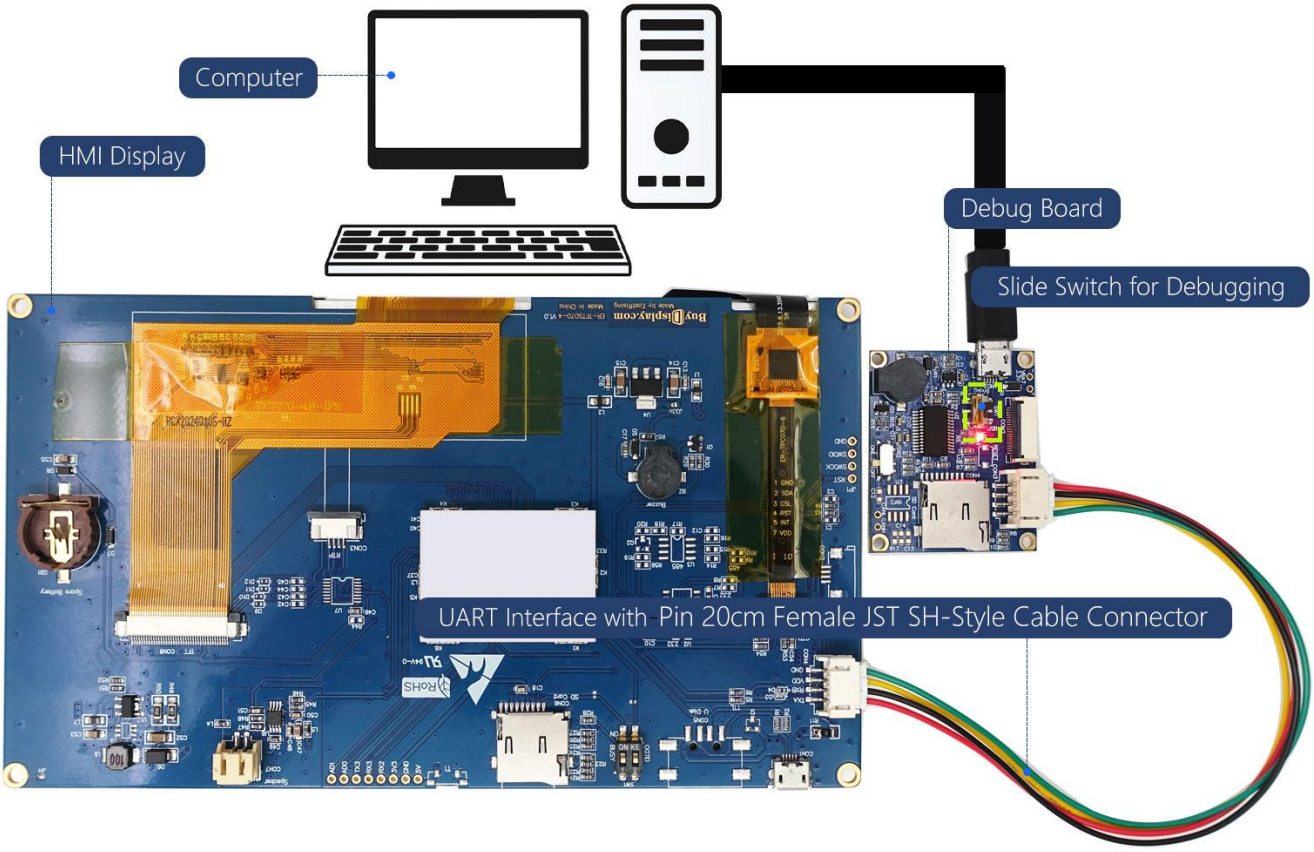
Figure 9-14: Activate UI_Debugger-II

9.2.1 Connect Debug Board

1) Use Female JST SH-Style Cable to connect Uart interface of HMI display with Uart interface of debug board.

2) Use USB cable to connect debug board to computer

3) Move the slide switch to the debug side.



9.2.2 Main Screen

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

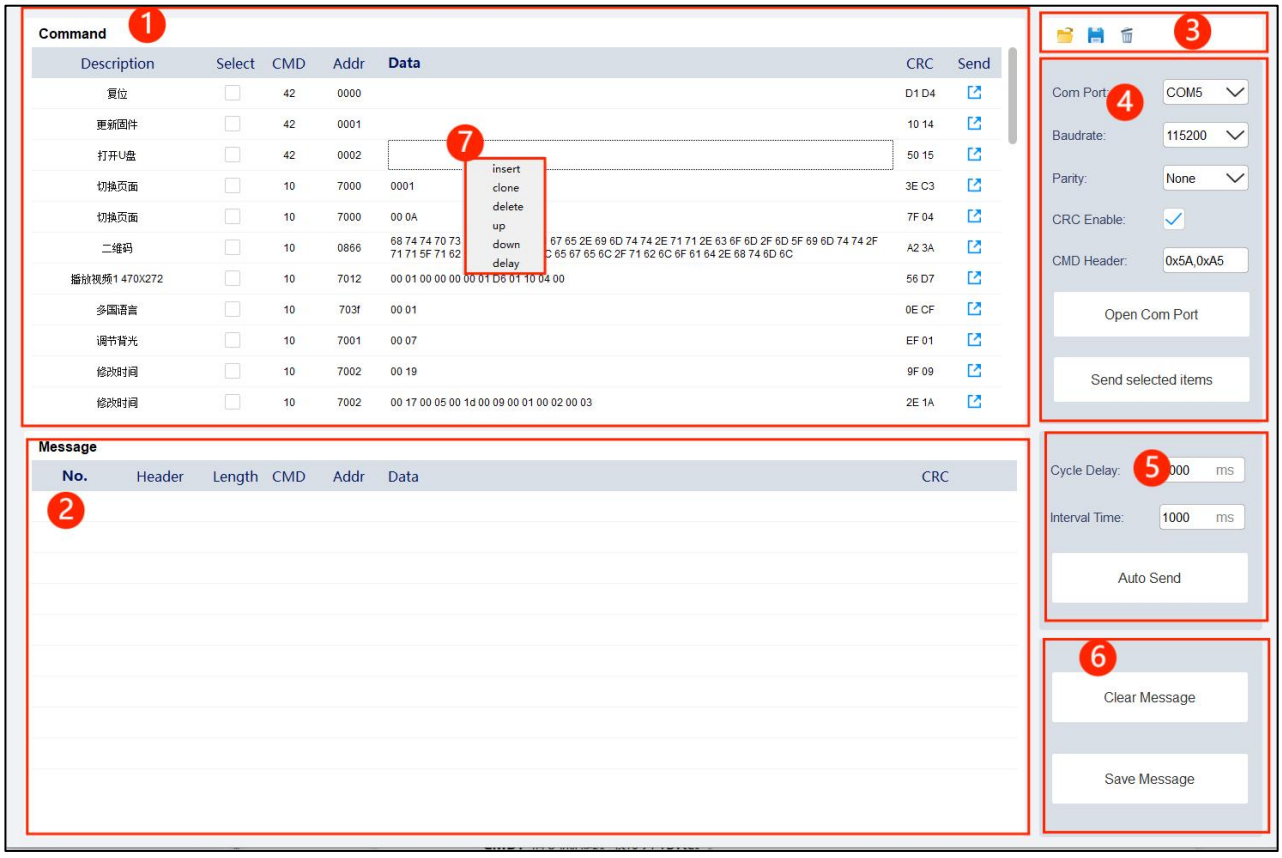


Figure 9-15: UI_Debugger-II Main Screen

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 (Refer to *Special Commands* for more detail). 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 = number 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 debug 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:

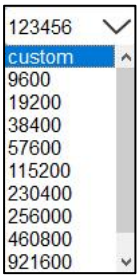


Figure 9-16: User-defined Baudrate

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

9.2.3 Tutorial – Send Commands

1、Send one 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 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 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

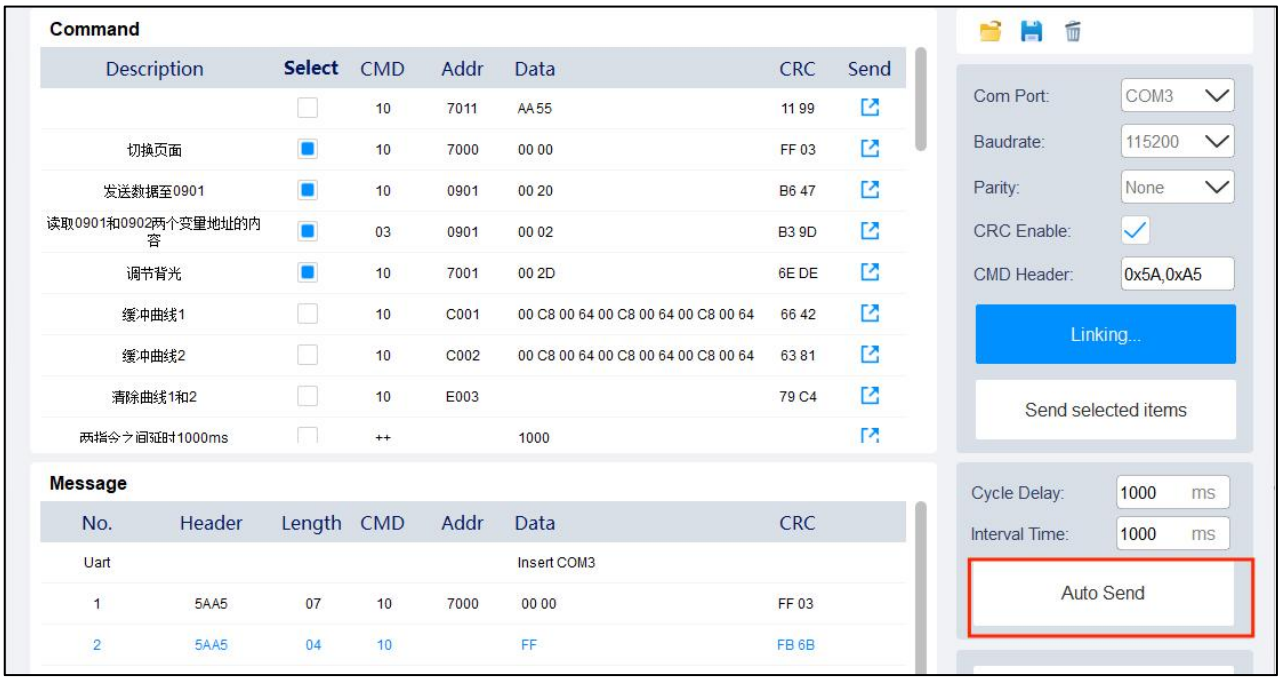


Figure 9-21: Send Multiple Commands

(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.

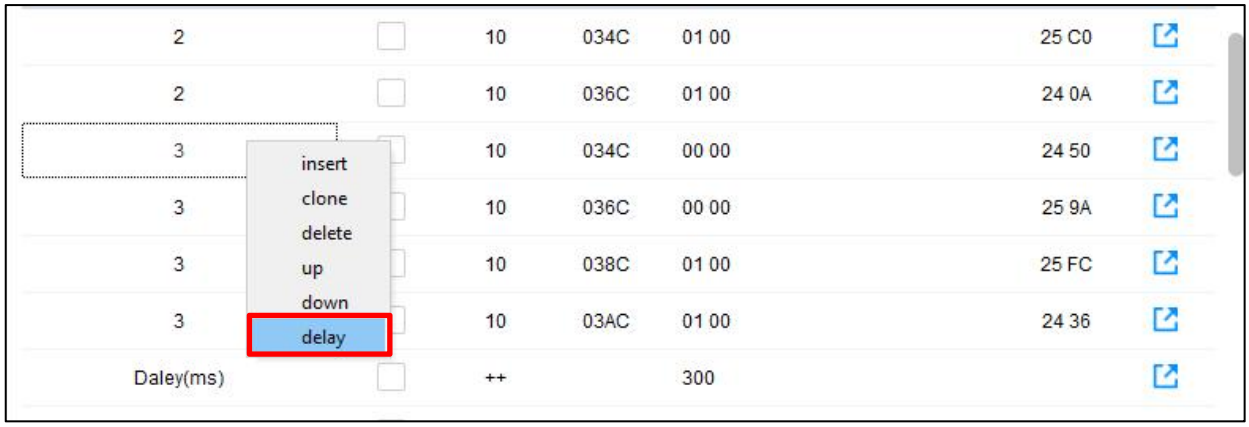


Figure 9-22: Add a delay command

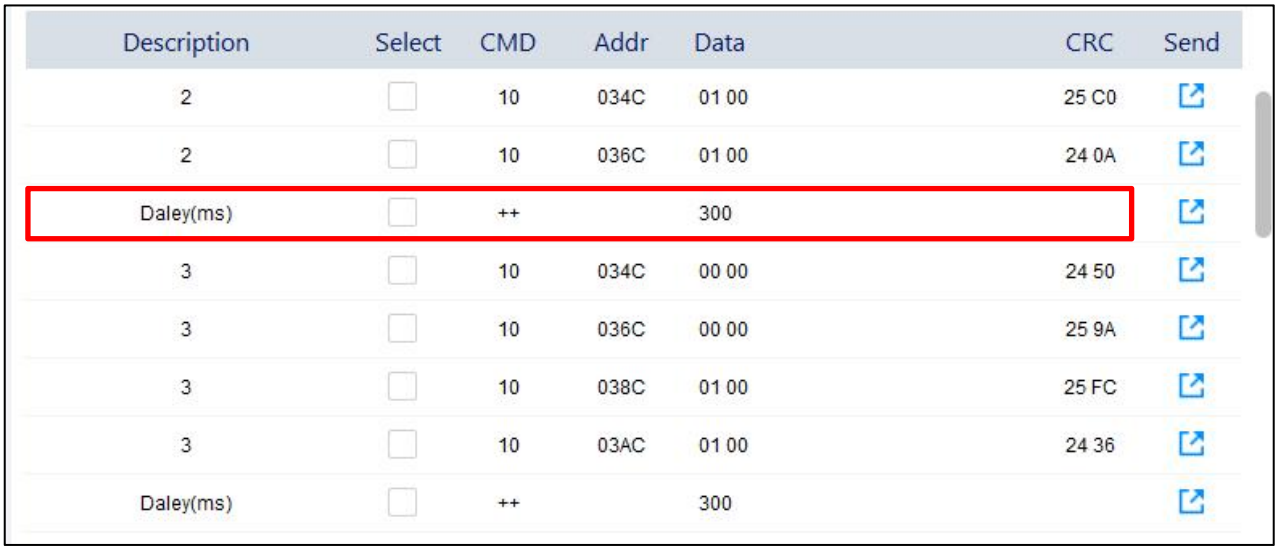


Figure 9-23: Setup delay time

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.

9.2.4 Save Commands

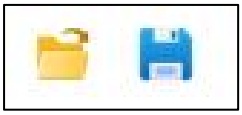


Figure 9-24: Load & Save Commands

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

9.2.5 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-II.

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

Figure 9-26: Example of Message Information File

9.3 Font Tool

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

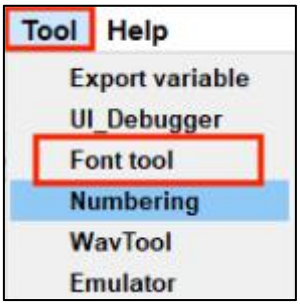


Figure 9-27: Activate BWFont

The main screen of BWFont is shown and explained below:

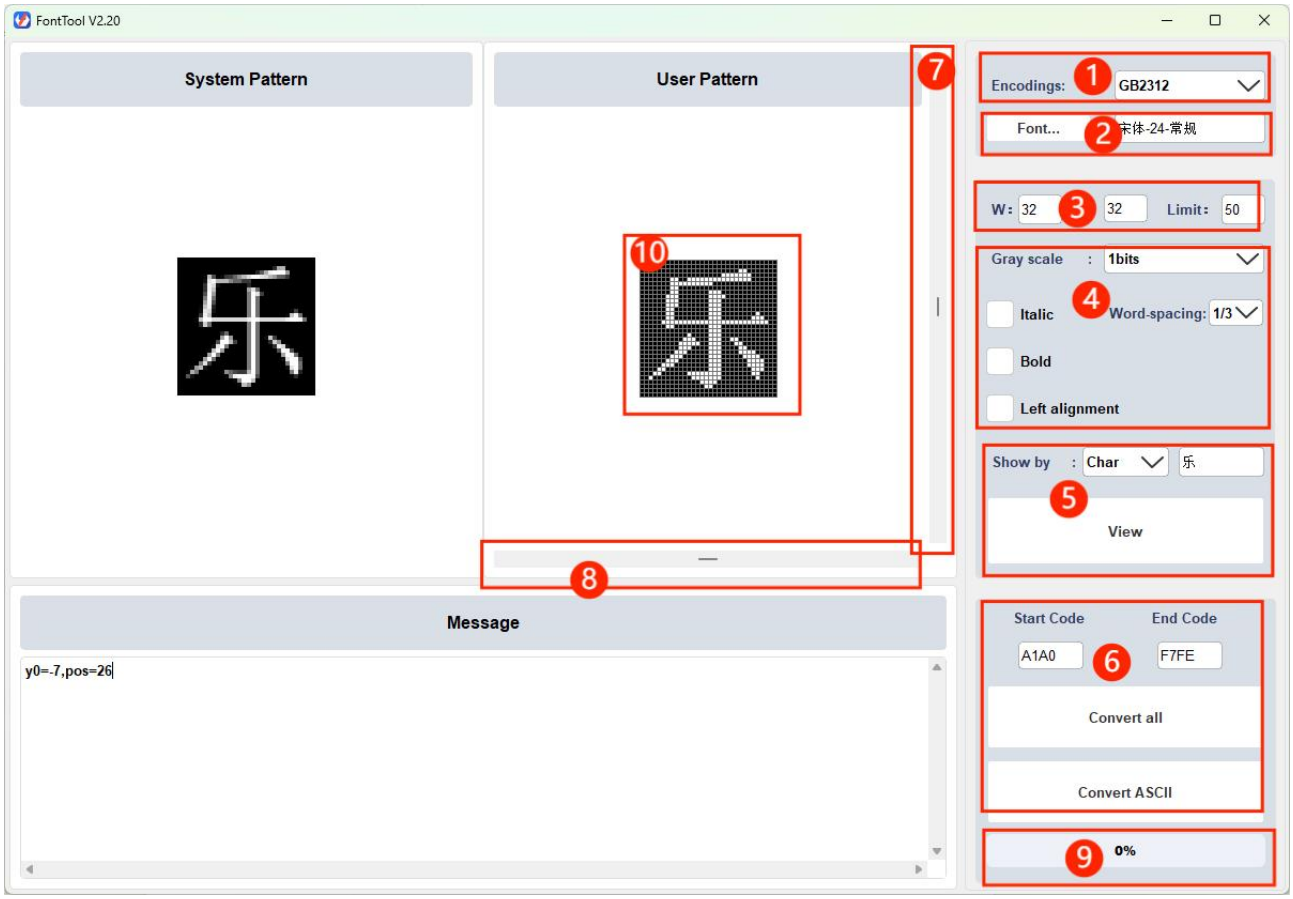


Figure 9-28: Main Screen of BWFont

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.

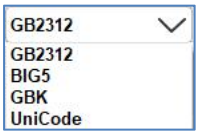


Figure 9-29: Encoding Types

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

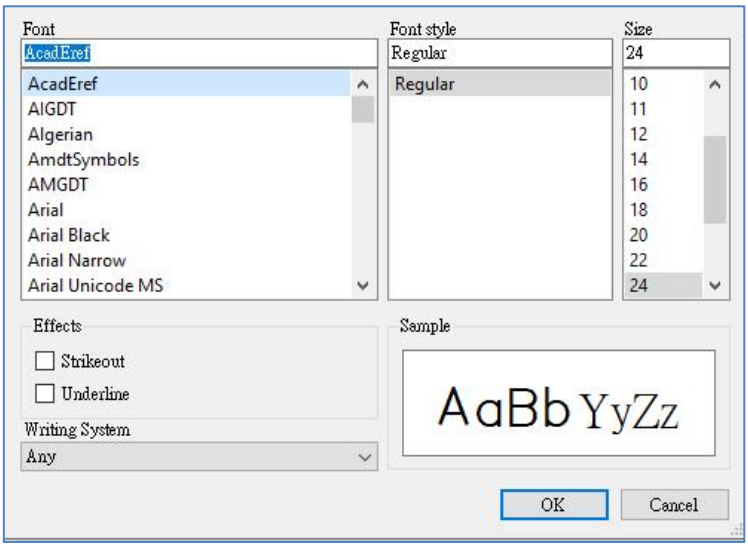


Figure 9-30: Select Font / Font Style / Size

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.

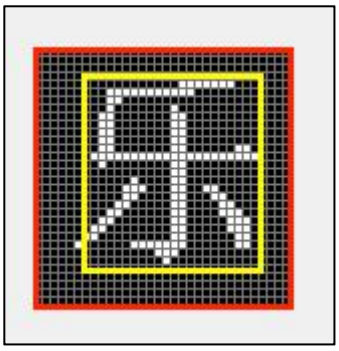


Figure 9-31: Character Example

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 EastRising 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 spaicing 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 Cick 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 Save the file to designated path. The file name must not include " * " , refer to *FontBin* for more details.

❽ Click **[Save]** to save the file

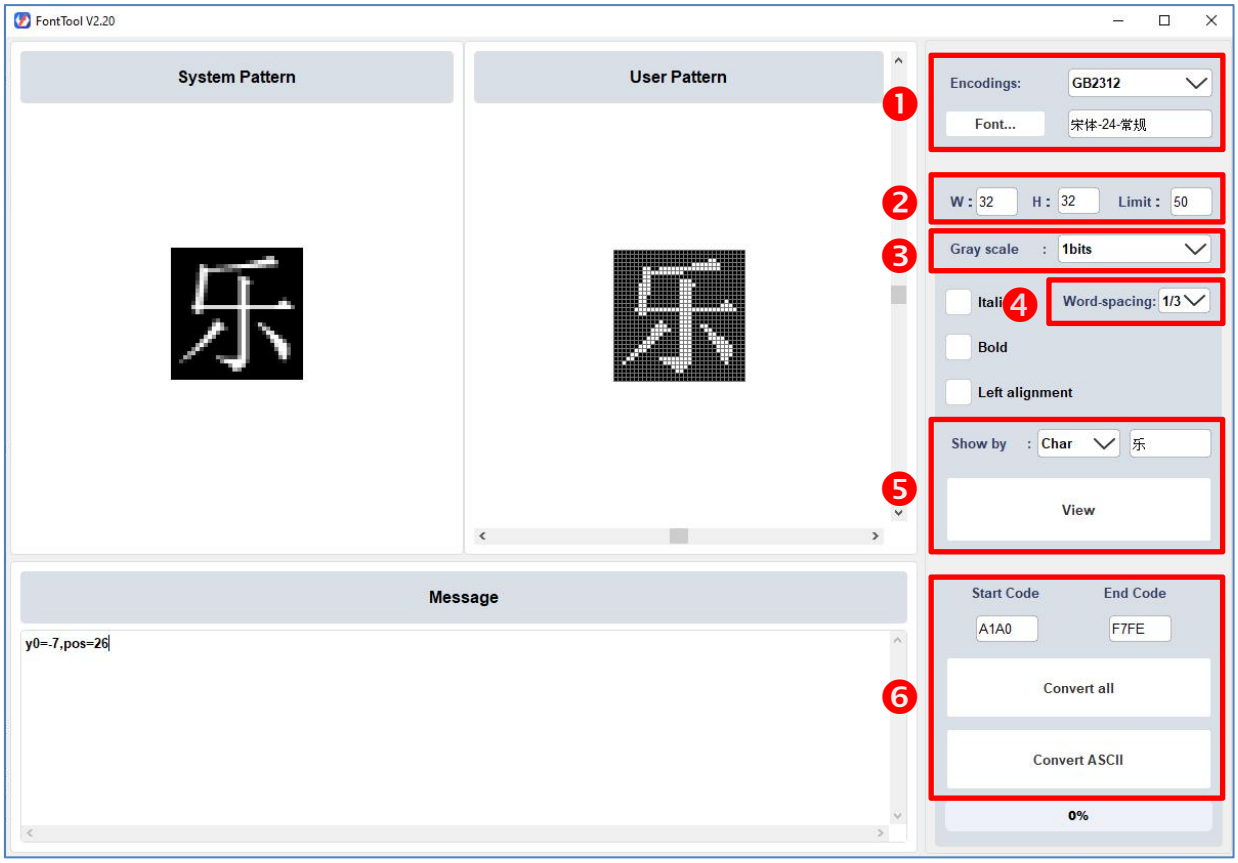


Figure 9-32: Steps of making a font (1)

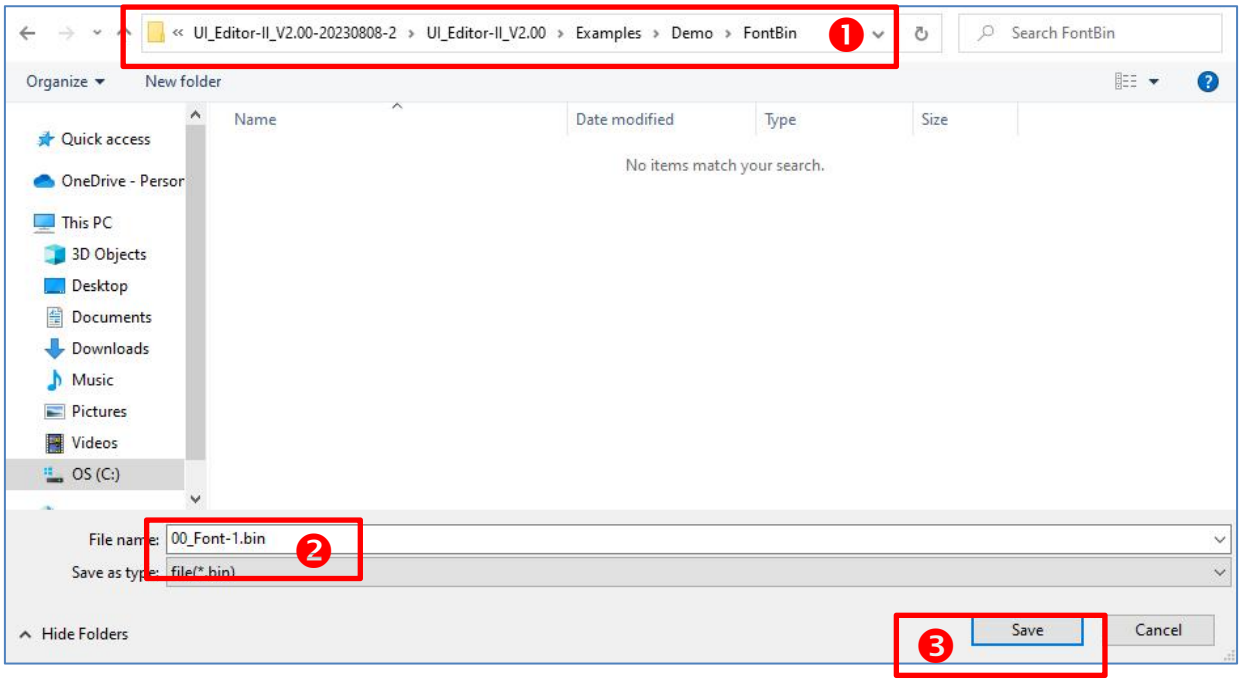


Figure 9-33: Steps of making a font (2)

9.4 Numbering Tool

Numbering_tool is designed to number the pictures and icons that will be used in UI_Editor-II. To activate the tool, simply click on the [Tool] menu and then click on [Numbering], as shown below:

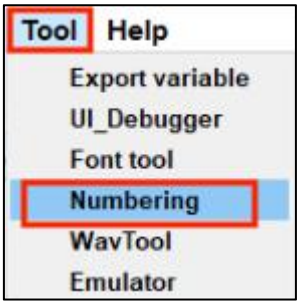


Figure 9-34: Activate Numbering_tool

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:

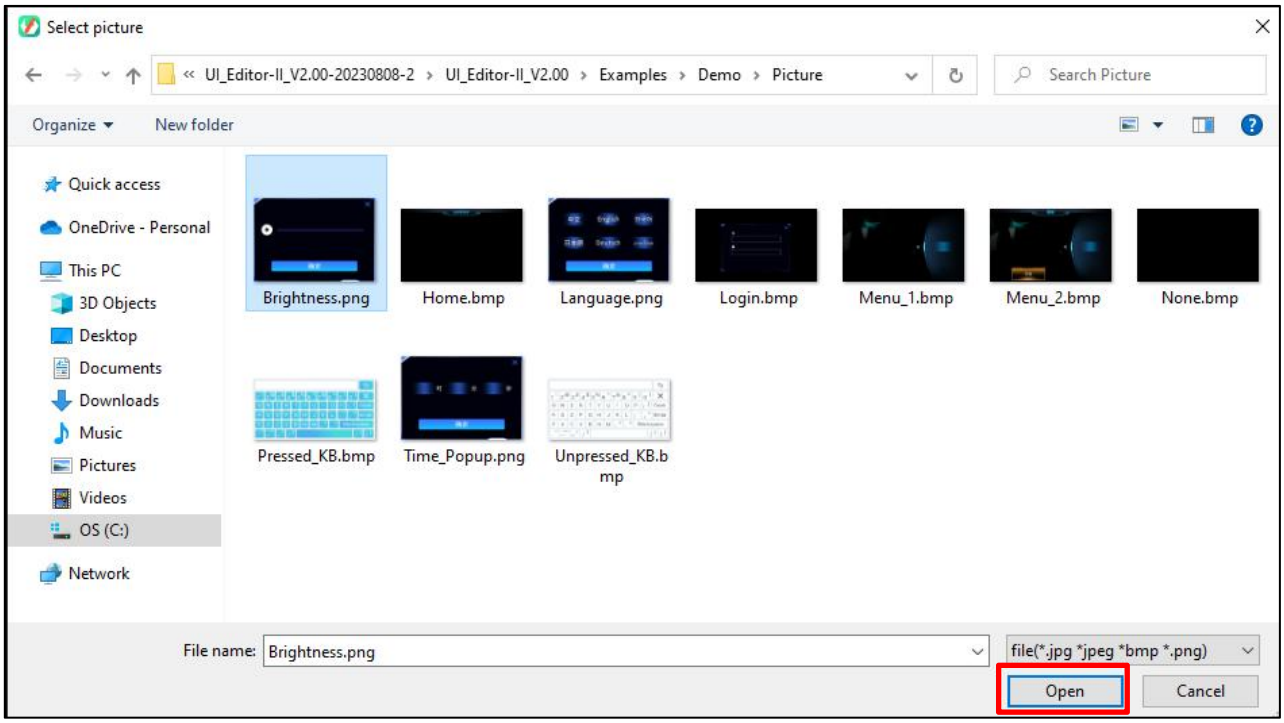


Figure 9-36：Select a Picture

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.

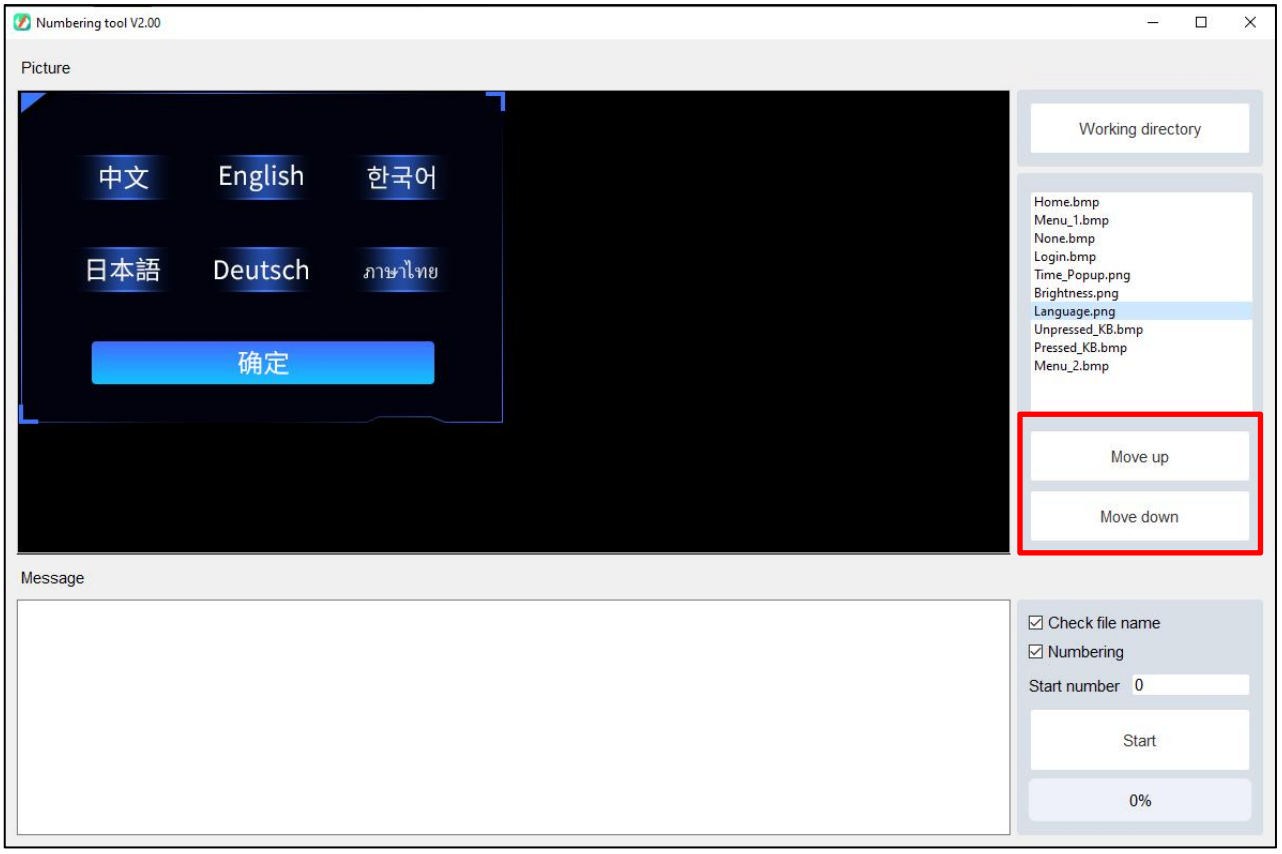


Figure 9-37：Adjust the picture order

3、Click on [Start] to number the pictures.

Figure 9-38：Process done

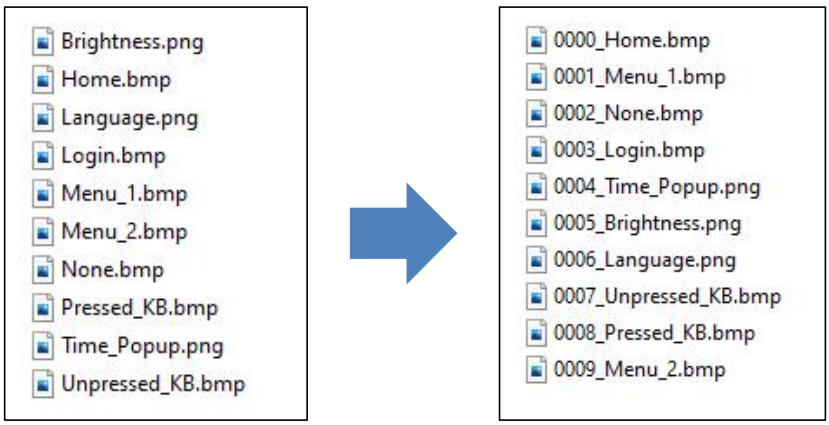


Figure 9-39：Final Result

9.5 WavTool

9.5.1 Make a Wave file

If an audio file is not in Wave format, developers will need to convert it into Wave format in order to use the related functions in UI_Editor-II.

9.5.2 Convert Wave to Bin

WavTool is designed to convert wave files into bin files. To activate the tool, simply click on the [Tool] menu and then click on [WavTool], as shown below:

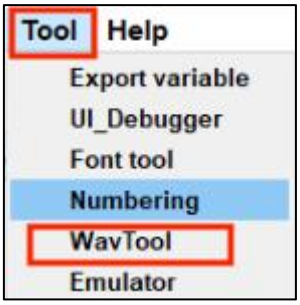


Figure 9-40: Activate WavTool

The main screen of WavTool is shown and explained below:

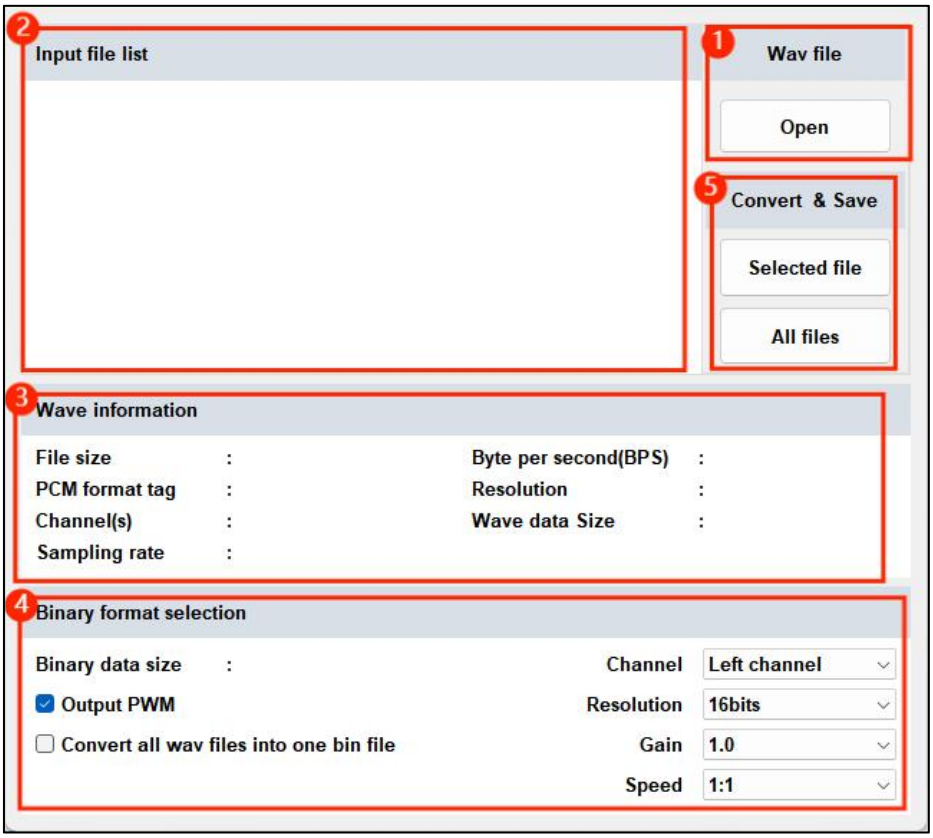


Figure 9-41: Main Screen of WavTool

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 wave file into a bin file.

All files: Click to convert all wave files into bin files.

Steps of converting wave files to bin files:

1、Activate WavTool and click on **[Open]** to add wave files. Select a wave file in the popup window, and then click **[Open].** As shown below:

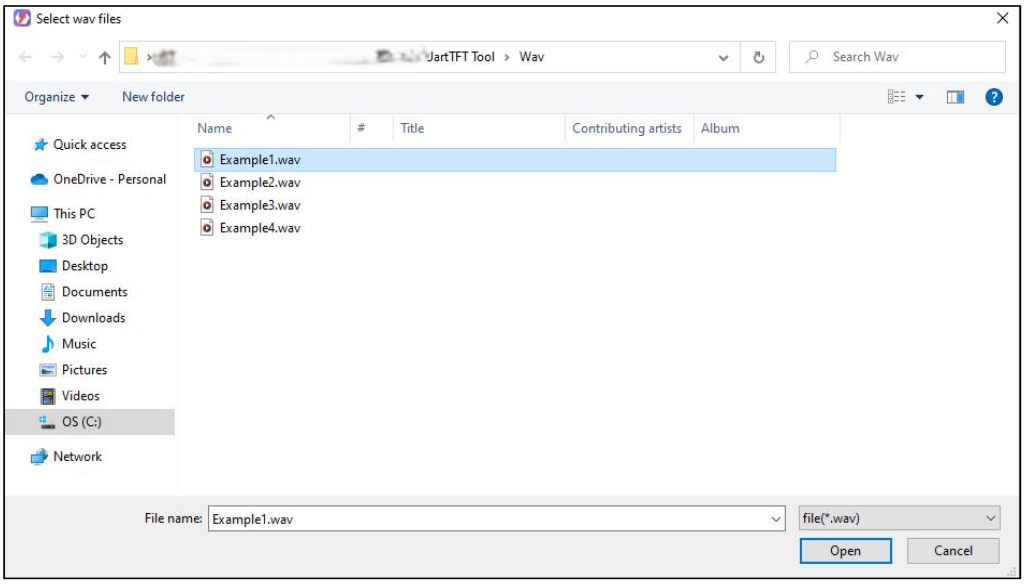


Figure 9-42: Select a Wave File

2、As shown in Figure 9-43, all the wave 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 wave file by 22050 sampling rate. Click on **[Selected file]** or **[All files]** to generate the bin file(s).

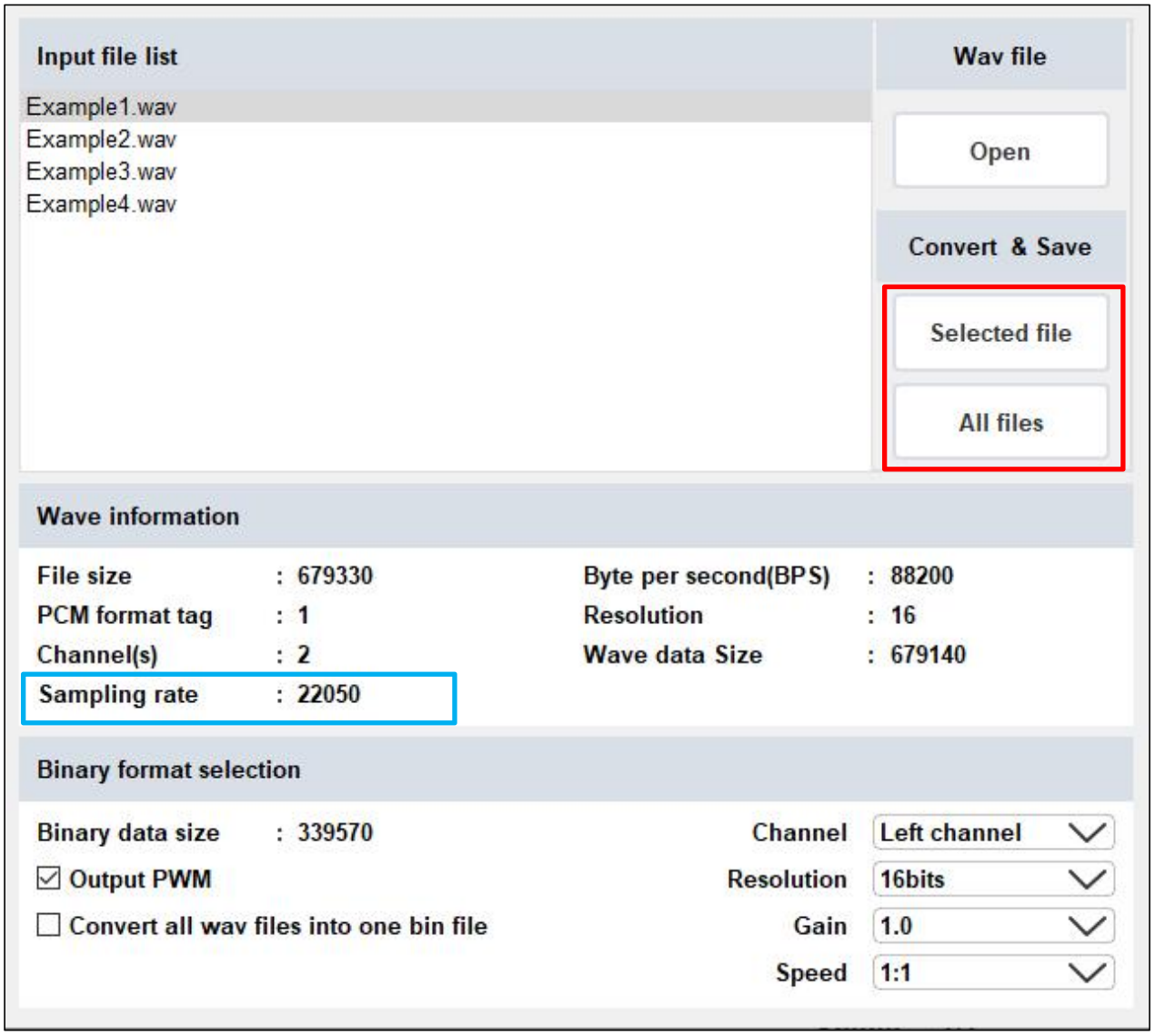


Figure 9-43: Convert Wave to Bin

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.

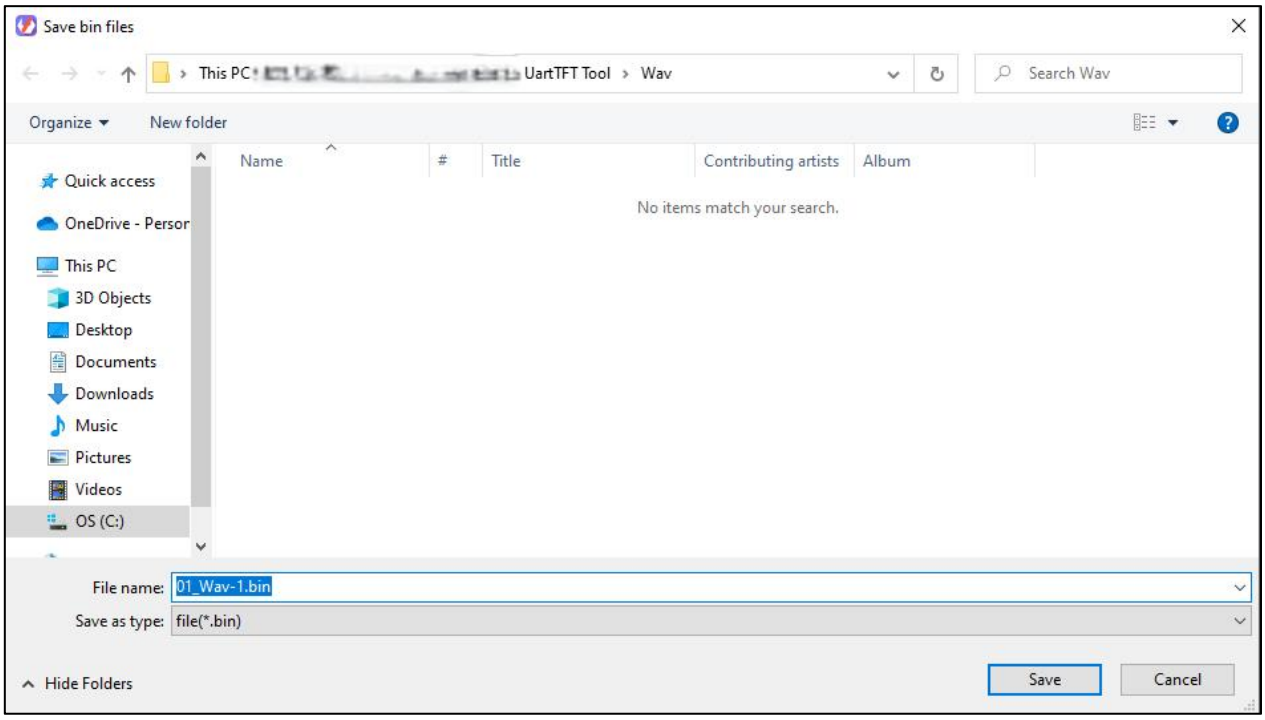


Figure 9-44: Save the bin file

Note: The file name should not include special characters such as **\ / : * ? " < > |**

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-II* for more details.

Table 10-1: Command Formats

Host write data	**Header 0xXXXX**	**Length 0xXX**	**Write Command 0x10**	**Address 0x0000 ~ 0x5FFF**	**Write Data 0xXXXX.....0xXXXX (2*n Bytes)**		**CRC 0xXXXX**
Host read data	**Header 0xXXXX**	**Length 0xXX**	**Read Command 0x03**	**Address 0x0000 ~ 0x5FFF**	**Read Word amount 0xXXXX**		**CRC 0xXXXX**
Returned message	**Header 0xXXXX**	**Length 0xXX**	**Read Command 0x03**	**Address 0x0000 ~ 0x5FFF**	**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-II.

2、**Length:** Command length. **Length = Command (1) + Address(2) + Write Data(2*N) + CRC(2)**

3、**Write/Read Command:** Command type. 0x01: Write; 0x03: Read; 0x41: Touch returned message

4、**Address:** Variable/Register address. Data length: 2Bytes

5、**Data:** Data to be written / Data amount to be read.

6、**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 register address. Each register has its own purpose. Refer to *Variable Address* for more detail.

Host may write maximum 250Bytes of data to UartTFT controller at a time.

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 and the Returned Message

Host write data	**Header 0xXXXX**	**Length 0xXX**	**Write Command 0x10**	**Address 0x0000 ~ 0x5FFF**	**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	NULL	0xFF	0x4C30
CRC Fail	0x5AA5	0x04	0x10	NULL	0x00	0x0C70

Example (using UI_Debugger-II):

1、Write 0x1020 to the address of 0x2001: **0x10 0x2001 0x1020**

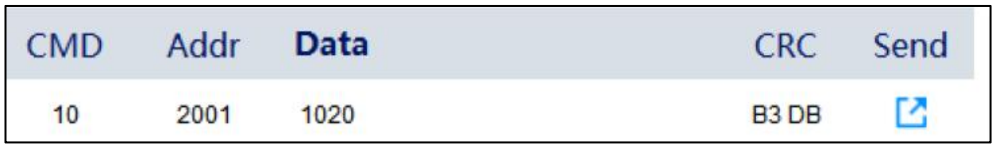


Figure 10-1: Example of Write Command (1)

2、Write 0x1020 and 0x2022 to the address of 0x2001: **0x10 0x2001 0x1020 0x2022**

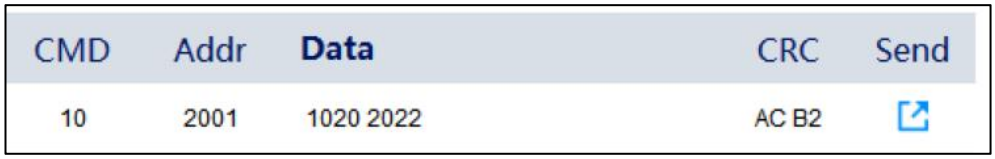


Figure 10-2: Example of Write Command (2)

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-II 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 widgets

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

Please note that a 2 bytes data, 00 00, must be added to the end of the text data as an ending sign

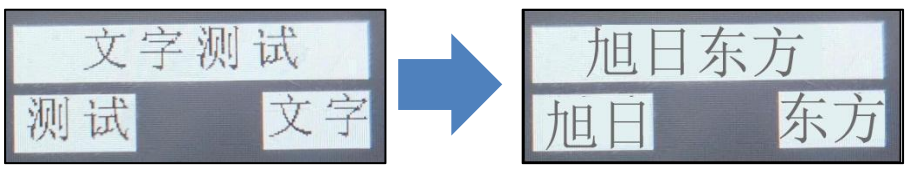


Figure 10-4: Write Command to Change Texts

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 10-5: Text Number & Graphics Number Widgets

As shown in Figure 10-5, 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

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 10-6 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**

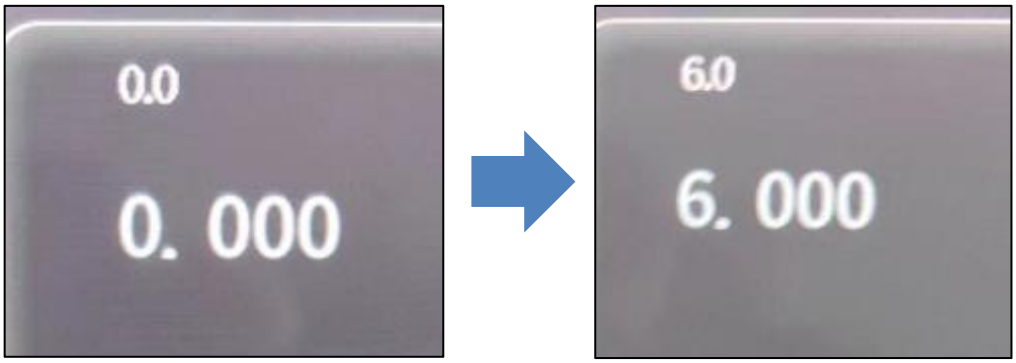


Figure 10-6: Write Command to Change Numbers

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 10-7: QRCode Parameters

As shown in Figure 10-7, 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

The display result is as shown in Figure 10-8.

Figure 10-8: 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 10-9: Bit Status Parameters

As shown in Figure 10-9, 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

The display result is as shown in Figure 10-10. (0: White circle; 1: Blue circle)

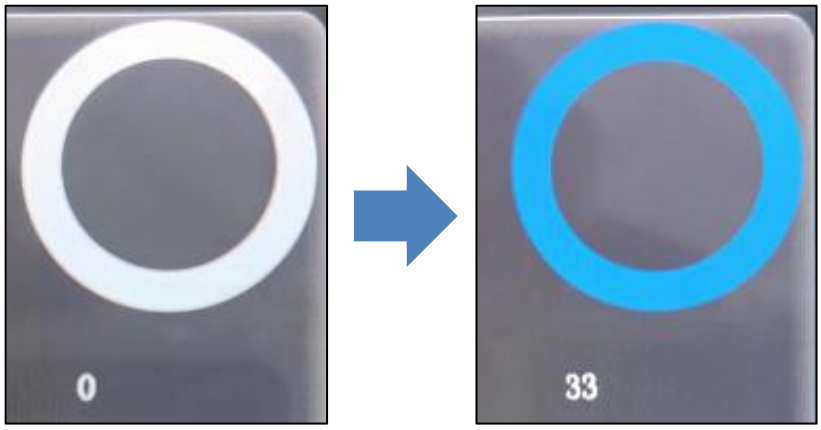


Figure 10-10: Write Command to change Bit Status

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 10-11: Icon Parameters

As shown in Figure 10-11, 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

The display result is as shown in Figure 10-12.

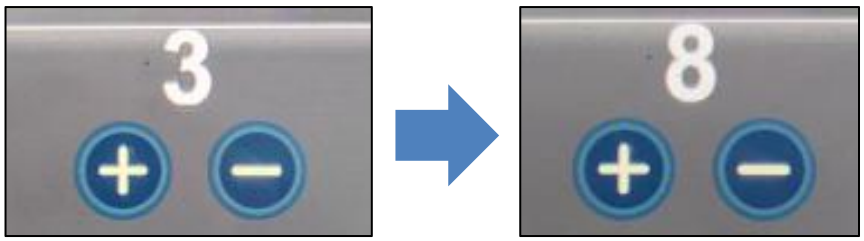


Figure 10-12: Write Command to Switch Icon

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 clear 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 10-3 & Table 10-4 show the address definition.

Table 10-3: 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 10-4: 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 10-13: Trend Graph Parameters

Figure 10-13 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 10-5: 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

The display result is as shown in Figure 10-14.

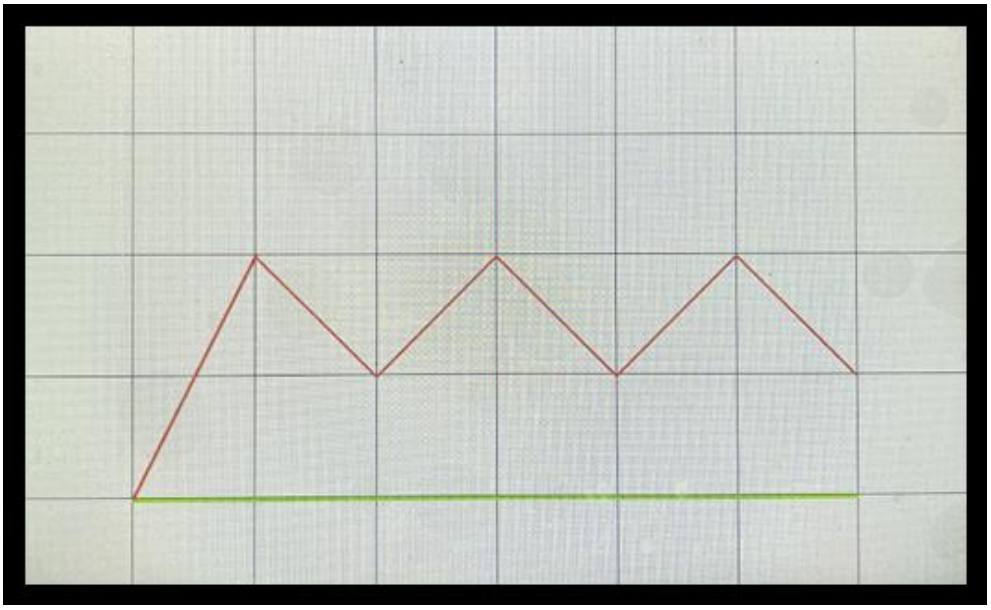


Figure 10-14: Display Result of Channel 0

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

The display result is as shown in Figure 10-15.

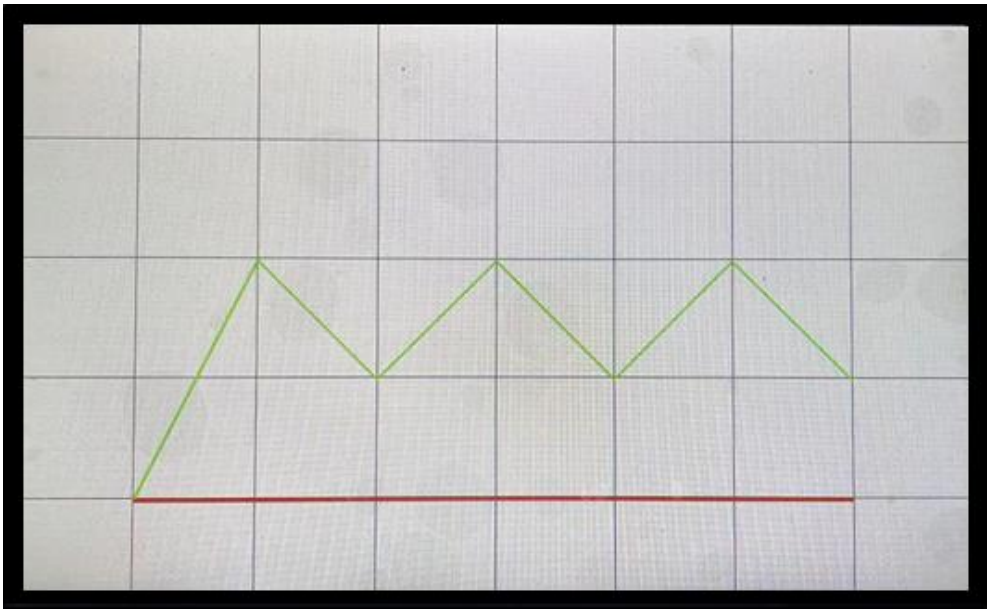


Figure 10-15: Display Result of Channel 1

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

Channel 0 will receive 3 sets of 200, and channel 1 will receive 3 sets of 100. The display result is shown in Figure 10-16.

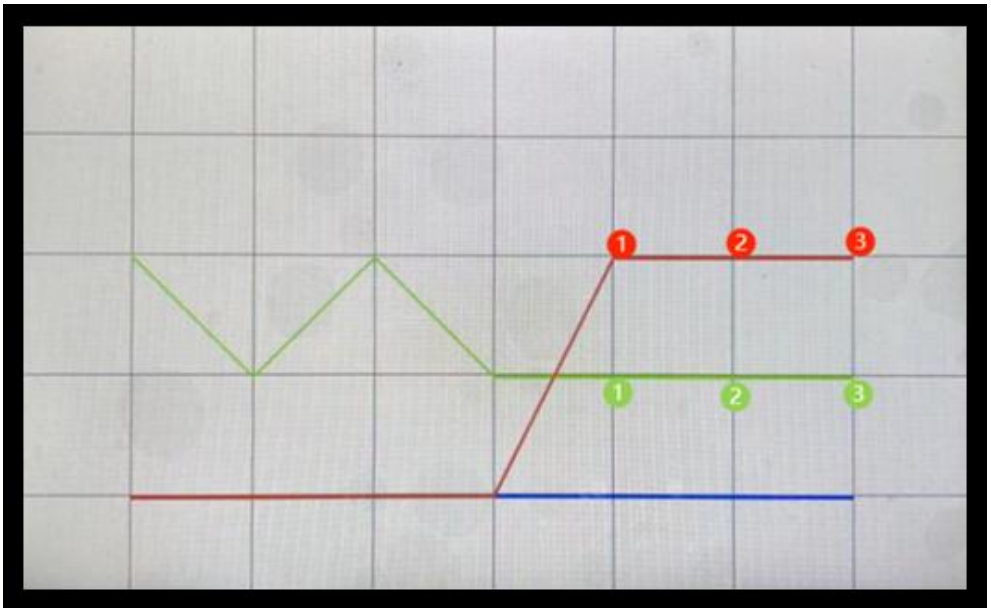


Figure 10-16: Display Result of Multi-Channels (Channel 0 & Channel 1)

Example: Clear the data in channel 0, the command is as below,

Header + 05 10 E0 01 + CRC

The display result is as shown in Figure 10-17.

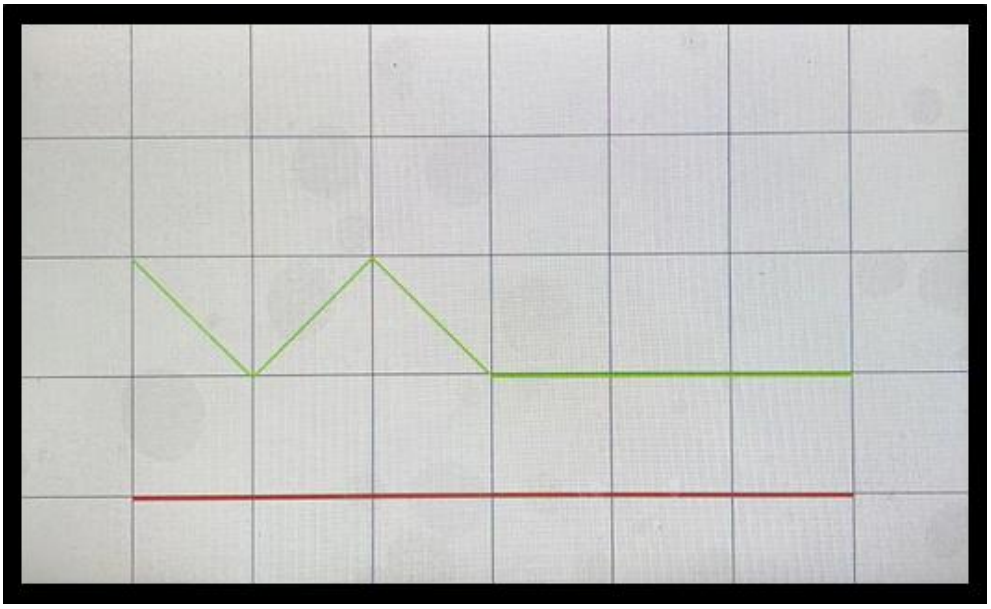


Figure 10-17: Clear the Data of Channel 0

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 ~ 0x5FFFF/0x1FFF, based on various IC models. Refer to *Registers Addresses by IC Models* 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

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

10.1.2.3 Time Registers – 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

Note: Updating time through Uart command does not need to write value to 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

0x0001: Play the 1st audio (0000.bin) in the folder

Header + 07 10 70 0A 00 01 + CRC

0x8001: Play the 1st audio (0000.bin) in the folder in loop

Header + 07 10 70 0A 80 01 + CRC

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

10.1.2.6 RTP Calibration Register – 0x700C

Write 0x005A to execute RTP calibration.

Command example: **Header + 07 10 70 0C 00 5A + CRC**

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**

Disable: **Header + 07 10 70 0E 00 00 + CRC**

10.1.2.9 Dimming Value Register – 0x700F

Value range: 0 ~ 63 (Same as **Sleep** parameter)

Example: Set brightness to 10 → **Header + 07 10 70 0F 00 0A + CRC**

10.1.2.10 Register for setting the wait- time to enter sleep mode – 0x7010

Unit: Second. Same as **Hold time** parameter.

Example: Set sleep time to 20 seconds → **Header + 07 10 70 10 00 14 + CRC**

10.1.2.11 Register for setting the Uart upgrade mode – 0x7011

Write 0xAA55 to enter upgrade mode (designated Bootloader is required)

Example: **Header + 07 10 70 11 AA 55 + CRC**

See *Download bin files through Uart port* for more details.

10.2 Read Command

Host may send a "Read command" to designated 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 10-11: 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)**	**NULL**	**CRC 0xXXXX**
e.g. Read 2*2Bytes from address 0x2050	0x5AA5	0x07	0x03	0x2050	0x0002	NULL	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 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**:**

CMD	Addr	Data	CRC	Send
03	2050	0002	EA10	

Figure 10-18: 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**

CMD	Addr	Data	CRC	Send
03	0220	0002	E1 B3	

Figure 10-19: 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-II is used, the write commands must be complete, including **Header**, **Length**, and **CRC** data, as described in Table 10-11

2、The returned messages listed in Table 10-11 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 Figure 10-20.

Figure 10-20: Enable reportToHost

The command code is 0x41 for Touch Returned Message. See the format below:

Table 10-12: 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 10-13: Widgets with reportToHost parameter

<table>
<tr><th>Widget Name</th><th>Header (2 Bytes)</th><th>Length (1 Bytes)</th><th>Command (1 Bytes)</th><th>Address/Registers(2 Bytes)</th><th>returnValue / Data</th><th>CRC (2 Bytes)</th></tr>
<tr><td>Page Slide to Jump</td><td rowspan="15">0x5AA5</td><td>0x07</td><td>0x41</td><td>0xFFFF</td><td>returnValue 0xXXXX (2 Bytes)</td><td rowspan="15">0xXXXX</td></tr>
<tr><td>Page Slide to Jump (with effect)</td><td>0x07</td><td>0x41</td><td>0xFFFF</td><td>returnValue 0xXXXX (2 Bytes)</td></tr>
<tr><td>Button</td><td>0x07</td><td>0x41</td><td>0xFFFF</td><td>returnValue 0xXXXX (2 Bytes)</td></tr>
<tr><td>Popupbox</td><td>0x07</td><td>0x41</td><td>0xFFFF</td><td>returnValue 0xXXXX (2 Bytes)</td></tr>
<tr><td>Variable Button / Encoder sub-function 2 and 3</td><td>0x07</td><td>0x41</td><td rowspan="9">Address/Registers 0xXXXX</td><td>Data 0xXXXX (2 Bytes)</td></tr>
<tr><td>Slider Bar</td><td>0x07</td><td>0x41</td><td>Data 0xXXXX (2 Bytes)</td></tr>
<tr><td>SlideMenu</td><td>0x07</td><td>0x41</td><td>Data 0xXXXX (2 Bytes)</td></tr>
<tr><td>Circular Touch</td><td>0x07</td><td>0x41</td><td>Data 0xXXXX (2 Bytes)</td></tr>
<tr><td>Numeric Keypad</td><td>0xXX</td><td>0x41</td><td>Data 0xXXXX.....0xXXXX (2*n Bytes)</td></tr>
<tr><td>EN_Keyboard</td><td>0xXX</td><td>0x41</td><td>Data 0xXXXX...0xXXXX (2*n Bytes)</td></tr>
<tr><td>CN_Keyboard</td><td>0xXX</td><td>0x41</td><td>Data 0xXXXX.....0xXXXX (2*n Bytes)</td></tr>
<tr><td>Multi-VariableButton / Encoder sub-function 1 and 4</td><td>0x23</td><td>0x41</td><td colspan="2">(Address/Register + Data) * 8
0xXXXX 0xXXXX.....0xXXXX 0xXXXX
(4*8 Bytes)</td></tr>
<tr><td>Timer</td><td>0x23</td><td>0x41</td><td colspan="2">(Address/Register + Data) * 8
0xXXXX 0xXXXX.....0xXXXX 0xXXXX
(4*8 Bytes)</td></tr>
</table>

Automatic Variable		0x07	0x41	Target Address 0xXXXX	Value 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 10-17: CRC Calculation for Write/Read Command

Type	Header	Length	Command	Address	Data / Data amount	CRC
Write Command	0xXXXX	0xXX	0x10	0~0x5FFF (2 Bytes)	0xXXXX……0xXXXX (2*n Bytes)	CRC 0xXXXX
Read Command	0xXXXX	0xXX	0x03	0~0x5FFF (2 Bytes)	0xXXXX (2 Bytes)	CRC 0xXXXX

As shown in Table 10-17, 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 10-18: CRC Calculation for Returned Result of Read Command

Returned Result	Header	Length	Command	Address	Data amount	Returned Data	CRC
Returned Result	0xXXXX	0xXX	0x03	0~0x5FFF (2 Bytes)	n 0xXXXX (2 Bytes)	0xXXXX (2*n Bytes)	CRC 0xXXXX

As shown in Table 10-18, the data of the green part will be used for calculating CRC,

Data to be used for calculating CRC → **Command Address Data amount (Word) Returned Data**

10.4.3 CRC Calculation for Touch Returned Message

Table 10-19 shows the 4 types of touch returned message. See *Touch Returned Message* for more information.

Table 10-19: 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 Table 10-19, 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 10-20 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 10-20: 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	Audio	N
SingleKey	N	Progress Bar	Y
		Circular Progress Bar	Y
		Bit Status	Y
		Automatic Variable	Y
		Icon	Y
		Trend Graph	Y
		Encoder	N
		Needle	Y

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 10-21 for the data length needed by various widgets.

Table 10-21: Data Length of Various Widget parameterAddrs

Widget Name	Data Length/Bytes	Occupied Spaces
String_Label	25	ParameterAddr + 0x000D
Text Number Display	17 + N	ParameterAddr+0x0009+N/2
Text Scroll	29	ParameterAddr + 0x000F
Graphics Number Display	15	ParameterAddr + 0x0008
Analog Clock	27	ParameterAddr + 0x000E
Digital Clock	10	ParameterAddr + 0x0005
Timer	49	ParameterAddr + 0x0019
Gif	54	ParameterAddr + 0x001B
QRCode	10	ParameterAddr + 0x0005
Progress Bar	20	ParameterAddr + 0x000A
Circular Progress Bar	35	ParameterAddr + 0x0012
Bit Status	13	ParameterAddr + 0x0007
Icon	15	ParameterAddr + 0x0008
Automatic Variable	65	ParameterAddr + 0x0021
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.

3、The data length of parameterAddr = Len value + 1, where Len is explained in below sections.

4、The below explanation will be using 0x2000 as an example address of the parameterAddr

10.5.2 String: parameterAddr

Table 10-22 shows the parameterAddr related content of a String_Label widget. As an example, the address of parameterAddr is 0x2000, where 0x2000H represents the higher byte of 0x2000, and 0x2000L represents the lower byte of 0x2000. Note that lower byte data is saved ahead of higher byte data in RAM.

Table 10-22: parameterAddr Related Content of String_Label

Parameter Name	Data Length/Bytes	Feature	Address
Len	1	Unchangeable	0x2000H
writeAddr	2	Changeable	0x2000L
			0x2001H
wordLength	2	Changeable	0x2001L
			0x2002H
Xs	2	Changeable	0x2002L
			0x2003H
Ys	2	Changeable	0x2003L
			0x2004H
Xe	2	Changeable	0x2004L
			0x2005H
Ye	2	Changeable	0x2005L
			0x2006H
fontWidth	1	Unchangeable	0x2006L
fontHeight	1	Unchangeable	0x2007H
fontID	1	Changeable	0x2007L
encoding	1	Unchangeable	0x2008H
alignment	1	Changeable	0x2008L
_color	3	Changeable	0x2009H
			0x2009L
			0x200AH
fontColor	3	Changeable	0x200AL
			0x200BH
			0x200BL
Mode	1	Changeable	0x200CH
0x00			0x200CL

Parameter Name	RGB565	RGB888	Address
Len	0x18	0x18	0x2000H
writeAddr	0x00	0x00	0x2000L
	0X03	0X03	0x2001H
wordLength	0x14	0x14	0x2001L
	0X00	0X00	0x2002H
Xs	0xC8	0xC8	0x2002L
	0X00	0X00	0x2003H
Ys	0x64	0x64	0x2003L
	0x00	0x00	0x2004H
Xe	0x8F	0x8F	0x2004L
	0x01	0x01	0x2005H
Ye	0xC7	0xC7	0x2005L
	0x00	0x00	0x2006H
fontWidth	0x20	0x20	0x2006L
fontHeight	0x20	0x20	0x2007H
fontID	0x00	0x00	0x2007L
encoding	0x00	0x00	0x2008H
alignment	0x01	0x01	0x2008L
_color	0x40	0x00	0x2009H
	0xFD	0xAA	0x2009L
	0x00	0xFF	0x200AH
fontColor	0xFF	0xFF	0x200AL
	0x57	0xFF	0x200BH
	0x00	0x55	0x200BL
Mode	0x03	0x03	0x200CH
NULL			0x200CL

Parameter	Data
name	label_0
parameterAddr	0x2000
writeAddr	0x0300
wordLength	20
X	200
Y	100
W	200
H	100
fontWidth	32
fontHeight	32
fontID	00_Font-2312...
encoding	GB2312
alignment	Middle
backgroundCo...	Enable
_color	0xFFAA00
fontColor	0x55FFFF
defaultText	label_0
passwordMode	Enable
multiLanguage	Enable

Figure 10-22: parameterAddr Contents vs. Widget Parameters

As shown in Figure 10-22,

Len: The total number of data bytes calculated from writeAddr to Mode (Len itself is not included). The value of Len is 0x18 (Decimal: 24) for a String_Label widget.

Xs、Ys: The left-top coordinate of the widget

Xe、Ye: The right-bottom coordinate of the widget, where Xe(Ye) = Xs(Ys) + W(H) -1

encoding: 0x00 = GB2312; 0x01 = GBK; 0x02 = BIG5; 0x03 = UNICODE; 0x04 = ASCII; 0x06 =

UNICODE

fontID: Font ID. To modify this parameter, make sure that (1) the new Font is included in the current UartTFT-II_Flash.bin; (2) fontWidth, fontHeight, and encoding should be modified accordingly too; (3) The widget Width and Height may not fit in with the new Font.

alignment: There are 9 alignment modes, which is numbered from 0x00 to 0x08, as shown in Figure 10-23 below:

Figure 10-23: Alignment Modes

fontColor: As shown in Figure 10-22, 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, 0xFF is stored in 0x200AL, 0xFF is stored in 0x200BH, and 0x55 is stored in 0x200BL. If the UI project is set to RGB565, 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	(Get rid of the bits in red)
RGB565 (Bin) :	**01010**	**111111**	**11111**	
RGB565 (Hex) :	57FF			

Next, swap the converted RGB565 data (57FF → FF57), and then add another byte of 0x00 to the end of the color data (FF57 → FF5700). Finally, write 0xFF to 0x200AL, 0x57 to 0x200BH, and 0x00 to 0x200BL. See the example shown in the left table of Figure 10-22.

_color: The description is the same as fontColor above.

Mode: This parameter can be set through an 8bits data,

1、bit0 → Enable/Disable backgroundColor: bit0 = 1, Enable; bit0 = 0, Disable

2、bit1→ Enable/Disable passwordMode: bit1 = 1, Enable, bit0 = 0, Disable

Example: To modify the font color to "Blue" (RGB888), the "Write Command" will be as below:

0x10 (parameterAddr+0x000A) 0xNN 0xFF 0x00 0x00 (0xNN is the original lower byte of _color data)

Note:

1、In a "Write Command", the amount of data (Data column) must be 2*n Bytes. (The amount of data cannot be odd.)

2、For a parameter whose data length > = 2Bytes, when using "Write Command" to update its contents, the lower byte of the data must be placed before the higher byte of the data. As shown in Figure 10-23, to change the content of writeAddr to 0x1234, the "Write Command" will be as

0x10 0x2000 0x18 0x34 0x12 0x14

: Data updated from address 0x2000 to 0x2001

10.5.2.1 Example – Modify the Parameters of String_Lable Widget

Assume a String_Label widget is set as below:

Parameter Name	RGB565	RGB888	Address
Len	0x18	0x18	0x2000H
writeAddr	0x00	0x00	0x2000L
	0X03	0X03	0x2001H
wordLength	0x14	0x14	0x2001L
	0X00	0X00	0x2002H
Xs	0xC8	0xC8	0x2002L
	0X00	0X00	0x2003H
Ys	0x64	0x64	0x2003L
	0x00	0x00	0x2004H
Xe	0x8F	0x8F	0x2004L
	0x01	0x01	0x2005H
Ye	0xC7	0xC7	0x2005L
	0x00	0x00	0x2006H
fontWidth	0x20	0x20	0x2006L
fontHeight	0x20	0x20	0x2007H
fontID	0x00	0x00	0x2007L
encoding	0x00	0x00	0x2008H
alignment	0x01	0x01	0x2008L
_color	0x40	0x00	0x2009H
	0xFD	0xAA	0x2009L
	0x00	0xFF	0x200AH
fontColor	0xFF	0xFF	0x200AL
	0x57	0xFF	0x200BH

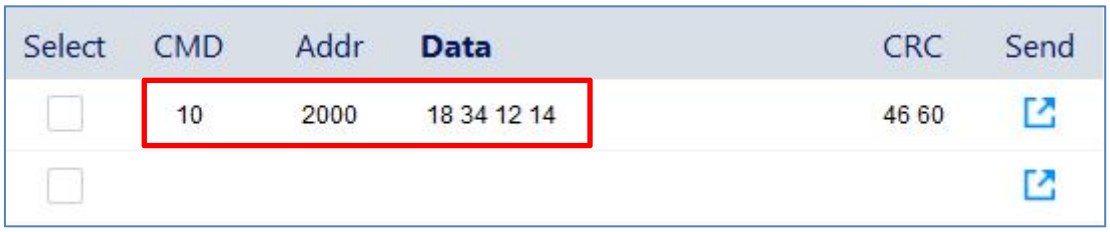
Parameter	Data
name	label_0
parameterAddr	0x2000
writeAddr	0x0300
wordLength	20
X	200
Y	100
W	200
H	100
fontWidth	32
fontHeight	32
fontID	00_Font-2312...
encoding	GB2312
alignment	Middle
backgroundCo...	Enable
_color	0xFFAA00
fontColor	0x55FFFF
defaultText	label_0
passwordMode	Enable
multiLanguage	Enable

	0x00	0x55	0x200BL
Mode	0x03	0x03	0x200CH
NULL			0x200CL

1. Modify writeAddr through Uart command

To do: Change the "writeAddr" from 0x0300 to 0x1234

The Uart command is as shown below:



Each code is explained as following:

10: Command type. 10 represents Write command

2000: ParameterAddr

18: the value stored in 0x2000H. (Parameter: Len)

34: new "writeAddr" to be written to 0x2000L, lower byte first

12: new "writeAddr" to be written to 0x2001H, higher byte

14: the value " stored in 0x2001L. (Parameter: wordLength)

Since writeAddr data are located at 0x2000L and 0x2000H respectively, when updating writeAddr, the data located in 0x2000H and 0x2001L have to be written too.

2. Modify widget location through Uart command

To do: Change the widget location from (200, 100) to (0, 0)

The Uart command is as shown below:

Each code is explained as following:

10: Command type. 10 represents Write command

2002: Start address for writing the data

00: the value stored in 0x2002H (Parameter: wordLength)

0000: new "Xs" value to be written to 0x2002L and 0x2003H, lower byte first.

0000: new "Ys" value to be written to 0x2003L and 0x2004H, lower byte first.

c700: new "Xe" value to be written to 0x2004L and 0x2005H, lower byte first.

6300: new "Ye" value to be written to 0x2005L and 0x2006H, lower byte first

20: the value stored in 0x2006L (Parameter: fontWidth)

The original Xe value is 0x8F01 (see the previous table, lower byte first), since the coordinate is changed from (**200**, 100) to (**0**, 0), the updated Xe value should be 0x018F – 200(decimal) = 0x00C7. As the data are stored in a lower byte first order, the new value of Xe becomes 0xC700.

The original Ye value is 0xC700 (see the previous table, lower byte first), since the coordinate is changed from (200, **100**) to (0, **0**), the updated Ye value should be 0x00C7 – 100(decimal) = 0x0063. As the data are stored in a lower byte first order, the new value of Ye becomes 0x6300.

3. Modify widget width through Uart command

To do: Modify the widget width from 200 pixels to 300 pixels.

The command is as shown below:

Select	CMD	**Addr**	Data	CRC	Send
☐	10	2004	00 F301 C7	4C 50	
☐					

Each code is explained as following:

10: Command type. 10 represents Write command

2004: Start address for writing the data

00: the value stored in 0x2004H (Parameter: Ys)

F301: new "Xe" value to be written to 0x2004L and 0x2005H, lower byte first.

C7: the value stored in 0x2005L (Parameter: Ye)

The new Xe value (F310) is derived by below calculation:

Original Xe = 0x018F = 399 (decimal)

To modify the widget width from 200pixels to 300pixels, Xe must be increased 100 pixels, that is,

New Xe = 399 + 100 = 499 = 0x01F3 (hexidecimal)

As the data are stored in a lower byte first order, the new data should be written as 0xF301

4. Modify alignment setting through Uart command

To do: Modify the alignment mode from Middle to Left.

The command is as shown below:

Select	CMD	Addr	Data	CRC	Send
☐	10	2008	00 00	6F C1	
☐					

Each code is explained as following:

10: Command type. 10 represents Write command

2008: Start address for writing the data

00: the value stored in 0x2008H (Parameter: encoding)

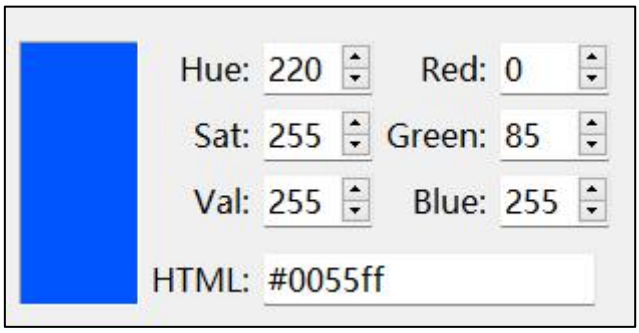
00: the new "alignment" data to be written to 0x2008L

0x01 represents Middle alignment.

0x00 represents Left alignment.

5. Modify font color through Uart command

To do: Modify the font color as shown in the below figure,



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 00 to complete the fontColor update.

The command is as shown below:

Select	CMD	Addr	Data	CRC	Send
☐	10	200A	00 BF 02 00	A5 24	
☐					

Each code is explained as following:

10: Command type. 10 represents Write command

200A: Start address for writing the data

00: the value stored in 200AH (Parameter: _color)

BF: the new "fontColor" data to be written to 0x200AL

02: the new "fontColor" data to be written to 0x200BH

00: the new "fontColor" data to be written to 0x200BL

10.5.3 Text Number Display: parameterAddr

Table 10-23: parameterAddr Related Content of Text Number Display

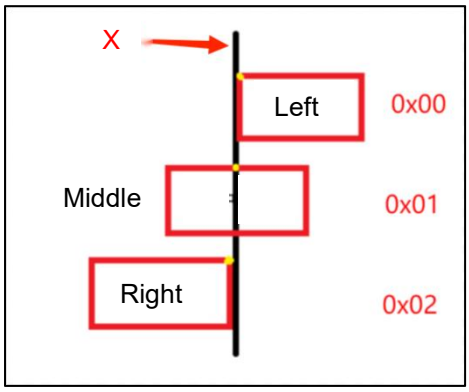
Parameter Name	Data Length/Bytes	Feature	Address
Len	1	Unchangeable	0x2000H
writeAddr	2	Changeable	0x2000L
			0x2001H
X	2	Changeable	0x2001L
			0x2002H
Y	2	Changeable	0x2002L
			0x2003H
fontColor	3	Changeable	0x2003L
			0x2004H
			0x2004L
fontID	1	Changeable	0x2005H
fontWidth	1	Unchangeable	0x2005L
alignment	1	Changeable	0x2006H
integerDigit	1	Changeable	0x2006L
decimalDigit	1	Changeable	0x2007H
dataType	1	Changeable	0x2007L
_length	1	Changeable	0x2008H
uniSymbol	N（not fixed）	Changeable	0x2008L

Len: The value of Len is 16+N Bytes，where N is the data length of UniSymbol.

X: X = the left-top X coordinate of the widget + W/2, where W is the widget width.

Y: The left-top Y coordinate of the widget.

alignMode: There are 3 alignment modes. 0x00: Left; 0x01: Middle; 0x02: Right. In addition, the bit7 of this parameter is used to control the setting of leadingZero, where bit7= 1 is Enable, and bit7=0 is Disable.



dataType: 0x80 : char; 0x00: uchar; 0x81: short; 0x01: ushort; 0x82: int; 0x02: uint; 0x03: longlong

_length: The data length of uniSymbol, a character = 1byte

uniSymbol: Name of the units such as Km, Kg, and ml. The characters are in ASCII code.

Note: Since uniSymbol is not a fixed parameter, the value of Len is not fixed. For example, if the uniSymbol is set as KM, then Len = 18. If uniSymbol is not set, then Len = 16.

10.5.4 Text Scroll: parameterAddr

Table 10-24: parameterAddr Related Content of Text Scroll

Parameter Name	Data Length/Bytes	Feature	Address
Len	1	Unchangeable	0x2000H
writeAddr	2	Changeable	0x2000L
			0x2001H
Xs	2	Changeable	0x2001L
			0x2002H
Ys	2	Changeable	0x2002L
			0x2003H
Xe	2	Changeable	0x2003L
			0x2004H
Ye	2	Changeable	0x2004L
			0x2005H
wordLength	2	Changeable	0x2005L
			0x2006H
fontID	1	Changeable	0x2006L
fontWidth	1	Unchangeable	0x2007H
fontHeight	1	Unchangeable	0x2007L
encoding	1	Unchangeable	0x2008H
alignment	1	Changeable	0x2008L
scrollMode	1	Changeable	0x2009H
fontColor	3	Changeable	0x2009L
			0x200AH
			0x200AL
interval(10ms)	1	Changeable	0x200BH
backgroundColor	3	Changeable	0x200BL
			0x200CH
			0x200CL
interval(pixels)	2	Changeable	0x200DH
			0x200DL
transparency	1	Changeable	0x200EH
0x00			0x200EL

Len: The total number of bytes calculated from writeAddr to transparency (Len itself is not included), which is 28Bytes (0x1C).

alignment: There are three alignment modes: 0x00 = Left; 0x01 = Middle; 0x02 = Right。

scrollMode: 0 = Disable; 1 = Enable

transparency: 0 = Disable; 1 = Enable

10.5.5 Graphics Number Display: parameterAddr

Table 10-25: parameterAddr Related Content of Graphics Number Display

Parameter Name	Data Length/Bytes	Feature	Address
Len	1	Unchangeable	0x2000H
writeAddr	2	Changeable	0x2000L
			0x2001H
dataType	1	Changeable	0x2001L
X	2	Changeable	0x2002H
			0x2002L
Y	2	Changeable	0x2003H
			0x2003L
integerDigit	1	Changeable	0x2004H
decimalDigit	1	Changeable	0x2004L
alignment	1	Changeable	0x2005H
firstIcon	2	Changeable	0x2005L
			0x2006H
lastIcon	2	Changeable	0x2006L
			0x2007H
0x00			0x2007L

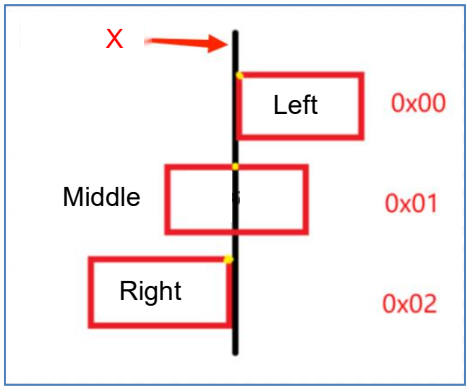
Len: The total number of bytes calculated from writeAddr to lastIcon (Len itself is not included), which is 14Bytes(0x0E).

dataType: 0x80 = char; 0x00 = uchar; 0x81 = short; 0x01 = ushort; 0x82 = int; 0x02 = uint; 0x03 = longlong Before changing dataType, developers must make sure there is sufficient consecutive RAM spaces.

X: X = the left-top X coordinate of the widget + W/2, where W is the widget width.

Y: The left-top Y coordinate of the widget.

alignMode: There are 3 alignment modes. 0x00: Left; 0x01: Middle; 0x02: Right. In addition, the bit7 of this parameter is used to control the setting of leadingZero, where bit7= 1 is Enable, and bit7=0 is Disable.



firstIcon & lastIcon: To modify these parameters, note that (1) the new pictures have to be included in the current UartTFT-II_Flash.bin; (2) the designated Icon number (4-digit decimal number) has to be convert 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 10-26: parameterAddr Related Content of Analog Clock

Parameter Name	Data Length/Bytes	Feature	Address
Len	1	Unchangeable	0x2000H
X	2	Changeable	0x2000L
			0x2001H
Y	2	Changeable	0x2001L
			0x2002H
background	2	Changeable	0x2002L
			0x2003H
hourHand_L	1	Changeable	0x2003L
hourHand_S	1	Changeable	0x2004H
hourHand_W	1	Changeable	0x2004L
hourHandColor	3	Changeable	0x2005H
			0x2005L
			0x2006H
minuteHand_L	1	Changeable	0x2006L
minuteHand_S	1	Changeable	0x2007H
minuteHand_W	1	Changeable	0x2007L
minuteHandColor	3	Changeable	0x2008H
			0x2008L
			0x2009H
secondHand_L	1	Changeable	0x2009L
secondHand_S	1	Changeable	0x200AH
secondHand_W	1	Changeable	0x200AL
secondHandColor	3	Changeable	0x200BH
			0x200BL
			0x200CH
centerIcon	2	Changeable	0x200CL
			0x200DH
0x00			0x200DL

Len: The total number of bytes calculated from X to centerIcon (Len itself is not included), which is 26Bytes(0x1A).

10.5.7 Digital Clock: parameterAddr

Table 10-27: parameterAddr Related Content of Digital Clock

Parameter Name	Data Length/Bytes	Feature	Address
Len	1	Unchangeable	0x2000H
X	2	Changeable	0x2000L
			0x2001H
Y	2	Changeable	0x2001L
			0x2002H
firstIcon	2	Changeable	0x2002L
			0x2003H
lastIcon	2	Changeable	0x2003L
			0x2004H
displayFormat	1	Changeable	0x2004L

Len: The total number of bytes calculated from X to displayFormat (Len itself is not included), which is 9Bytes(0x09).

displayFormat: Refer to below picture for the data vs. displayFormat,

10.5.8 Timer: parameterAddr

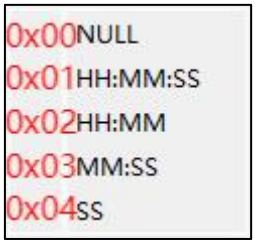
Table 10-28: parameterAddr Related Content of Timer

Parameter Name	Data Length/Bytes	Feature	Address
Len	1	Unchangeable	0x2000H
presetAddr	2	Changeable	0x2000L
			0x2001H
countAddr	2	Changeable	0x2001L
			0x2002H
controlAddr	2	Changeable	0x2002L
			0x2003H
X	2	Changeable	0x2003L
			0x2004H
Y	2	Changeable	0x2004L
			0x2005H
firstIcon	2	Changeable	0x2005L
			0x2006H
displayFormat	1	Changeable	0x2006L
countMode	1	Changeable	0x2007H
globalCounting	1	Changeable	0x2007L
reportToHost	1	Changeable	0x2008H
writeAddr0	2	Changeable	0x2008L
			0x2009H
_value0	2		0x2009L
			0x200AH
writeAddr1	2		0x200AL
			0x200BH
_value1	2		0x200BL
			0x200CH
.....			.....
writeAddr7	2		0x2016L
			0x2017H
_value7	2		0x2017L
			0x2018H
0x00			0x2018L

Len : The total number of bytes calculated from presetAddr to _value7 (Len itself is not included), which is 48Bytes(0x30).

firstIcon: To modify this parameter, developers should make sure the new Icon has been included in the current UartTFT-II_Flash.bin

displayFormat: Refer to below picture for the data vs. displayFormat settings,



countMode: 0x00 = counterclockwise; 0x01 = clockwise

globalCounting: 0x00 = Disable; 0x01 = Enable

reportToHost: 0x00 = Disable; 0x01 = Enable

10.5.9 Gif: parameterAddr

Table 10-29: parameterAddr Related Content of Gif

Parameter Name	Data Length/Bytes	Feature	Address
Len	1	Unchangeable	0x2000H
writeAddr	2	Changeable	0x2000L
			0x2001H
X	2	Changeable	0x2001L
			0x2002H
Y	2	Changeable	0x2002L
			0x2003H
W	2	Changeable	0x2003L
			0x2004H
H	2	Changeable	0x2004L
			0x2005H
gifName	2	Changeable	0x2005L
			0x2006H
playAtStart	1	Changeable	0x2006L
interval(10ms)	1	Changeable	0x2007H
startCode	2	Changeable	0x2007L
			0x2008H
stopCode	2	Changeable	0x2008L
			0x2009H
mode	1	Reserved	0x2009L
playOnceCode	2	Changeable	0x200AH
			0x200AL
writeAddr0	2	Changeable	0x200BH
			0x200BL
_value0	2		0x200CH
			0x200CL
writeAddr1	2		0x200DH
			0x200DH
_value1	2		0x200EH
			0x200EH
.....			.....
writeAddr7	2		0x2019H
			0x2019L
_value7	2		0x201AH
			0x201AL

Len: The total number of bytes calculated from writeAddr to _value7 (Len itself is not included), which is 53Bytes(0x35). In addition, the bit7 of this parameter is used to control the setting of "effects" , where bit7= 1 is Enable (Len = 0xB5), and bit7=0 is Disable (Len = 0x35).

gifName: To modify this parameter, developers should make sure the new Gif has been included in the current UartTFT-II_Flash.bin, and the W & H parameters should also be updated accordingly to avoid abnormal display.

playAtStart: 0x00 = Disable; 0x01 = Enable

mode: Reserved

10.5.10 QRCode: parameterAddr

Table 10-30: parameterAddr Related Content of QRCode

Parameter Name	Data Length/Bytes	Feature	Address
Len	1	Unchangeable	0x2000H
writeAddr	2	Changeable	0x2000L
			0x2001H
byteLength	2	Changeable	0x2001L
			0x2002H
X	2	Changeable	0x2002L
			0x2003H
Y	2	Changeable	0x2003L
			0x2004H
size	1	Changeable	0x2004L

Len: The total number of bytes calculated from writeAddr to size (Len itself is not included), which is 9Bytes(0x09)

size: QRCode size, unit: 50pixels. For example, if the value is 0x02, the QRCode size = 100x100 pixels.

10.5.11 Progress Bar: parameterAddr

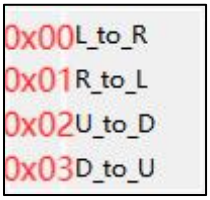
Table 10-31: parameterAddr Related Content of Progress Bar

Parameter Name	Data Length/Bytes	Feature	Address
Len	1	Unchangeable	0x2000H
writeAddr	2	Changeable	0x2000L
			0x2001H
bar_X	2	Changeable	0x2001L
			0x2002H
bar_Y	2	Changeable	0x2002L
			0x2003H
barIcon	2	Changeable	0x2003L
			0x2004H
direction	1	Changeable	0x2004L
maxValue	2	Changeable	0x2005H
			0x2005L
minValue	2	Changeable	0x2006H
			0x2006L
X	2	Changeable	0x2007H
			0x2007L
Y	2	Changeable	0x2008H
			0x2008L
background	2	Changeable	0x2009H
			0x2009L

Len: The total number of bytes calculated from writeAddr to background (Len itself is not included), which is 19Bytes(0x13).

bar_X、bar_Y: The left-top coordinate of the barIcon. Note that the reference point (0, 0) is the left-top coordinate of the panel, not the left-top coordinate of the widget.

direction: Refer to below picture for the data vs. direction settings,



10.5.12 Circular Progress Bar: parameterAddr

Table 10-32: parameterAddr Related Content of Circular Progress Bar

Parameter Name	Data Length/Bytes	Feature	Address
Len	1	Unchangeable	0x2000H
writeAddr	2	Changeable	0x2000L
			0x2001H
X	2	Changeable	0x2001L
			0x2002H
Y	2	Changeable	0x2002L
			0x2003H
foreground	2	Changeable	0x2003L
			0x2004H
background	2	Changeable	0x2004L
			0x2005H
minValue	2	Changeable	0x2005L
			0x2006H
maxValue	2	Changeable	0x2006L
			0x2007H
startAngle	2	Changeable	0x2007L
			0x2008H
finalAngle	2	Changeable	0x2008L
			0x2009H
promptNum_X	2	Changeable	0x2009L
			0x200AH
promptNum_Y	2	Changeable	0x200AL
			0x200BH
integerDigit	1	Changeable	0x200BL
decimalDigit	1	Changeable	0x200CH
alignment	1	Changeable	0x200CL
fontID	1	Changeable	0x200DH
fontColor	3	Changeable	0x200DL
			0x200EH
			0x200EL
firstIcon	2	Changeable	0x200FH
			0x200FL
lastIcon	2	Changeable	0x2010H
			0x2010L

digitDisplayMode	1	Changeable	0x2011H

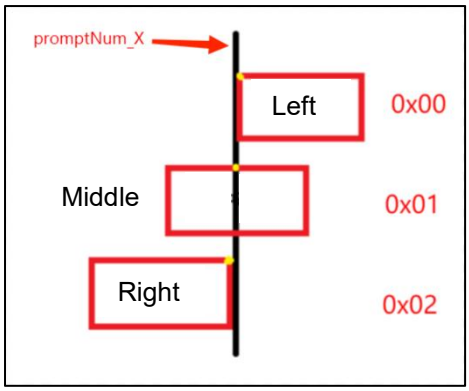
Len : The total number of bytes calculated from writeAddr to digitDisplayMode (Len itself is not included), which is 34Bytes(0x22).

startAngle、finalAngle: Range: 0° <= startAngle < finalAngle <= 360°

promptNum_X、promptNum_Y: The left-top coordinate of the promptNum. Note that the reference point (0, 0) is the left-top coordinate of the panel, not the left-top coordinate of the widget.

integerDigit、decimalDigit: The sum of integer digit and decimal digit should be < 5.

alignment: There are 3 alignment modes using promptNum_X as the baseline, as shown below:
0x00: Left; 0x01: Middle; 0x02: Right



fontColor: Valid only when fontID is set

digitDisplayMode:

0x00 = null → Do not display number;

0x01 = FontNum → Display font number;

0x02 = IconNum → Display icon number

10.5.13 Bit Status: parameterAddr

Table 10-33: parameterAddr Related Content of Bit Status

Parameter Name	Data Length/Bytes	Feature	Address
Len	1	Unchangeable	0x2000H
writeAddr	2	Changeable	0x2000L
			0x2001H
X	2	Changeable	0x2001L
			0x2002H
Y	2	Changeable	0x2002L
			0x2003H
bitIndex	1	Changeable	0x2003L
offStateIcon	2	Changeable	0x2004H
			0x2004L
onStateIcon	2	Changeable	0x2005H
			0x2005L
mode	1	Reserved	0x2006H
0x00			0x2006L

Len: The total number of bytes calculated from writeAddr to mode (Len itself is not included), which is 12Bytes(0x0C). In addition, the bit7 of this parameter is used to control the setting of "overlap", where bit7= 1 is Enable (Len = 0x8C), and bit7=0 is Disable (Len = 0x0C).

bitIndex: 0x00 ~ 0x0F vs. bit 0 ~ bit 15 。

10.5.14 Icon: parameterAddr

Table 10-34: parameterAddr Related Content of Icon

Parameter Name	Data Length/Bytes	Feature	Address
Len	1	Unchangeable	0x2000H
writeAddr	2	Changeable	0x2000L
			0x2001H
X	2	Changeable	0x2001L
			0x2002H
Y	2	Changeable	0x2002L
			0x2003H
firstIcon	2	Changeable	0x2003L
			0x2004H
eImage	2	Changeable	0x2004L
			0x2005H
minDisplayID	2	Changeable	0x2005L
			0x2006H
maxDisplayID	2	Changeable	0x2006L
			0x2007H
0x00			0x2007L

Len: The total number of bytes calculated from writeAddr to maxDisplayID (Len itself is not included), which is 14Bytes(0x0E). In addition, the bit7 of this parameter is used to control the setting of "overlap" , where bit7= 1 is Enable (Len = 0x8E), and bit7=0 is Disable (Len = 0x0E).

firstIcon、lastIcon: If these parameters are not set in UI_Editor-II, then the content will be 0xFFFF. Otherwise, the content will be the icon number (Hexadecimal).

10.5.15 Automatic Variable: parameterAddr

Table 10-35: parameterAddr Related Content of Automatic Variable

Parameter Name	Data Length/Bytes	Feature	Address
Len	1	Unchangeable	0x2000H
targetAddr	2	Changeable	0x2000L
			0x2001H
_dataType	1	Changeable	0x2001L
presetAddr	2	Changeable	0x2002H
			0x2002L
loopCode	2	Changeable	0x2003H
			0x2003L
onceCode	2	Changeable	0x2004H
			0x2004L
stopCode	2	Changeable	0x2005H
			0x2005L
minValue	8	Changeable	0x2006H
			0x2006L
			0x2007H
			0x2007L
			0x2008H
			0x2008L
			0x2009H
			0x2009L
maxValue	8	Changeable	0x200AH
			0x200AL
			0x200BH
			0x200BL
			0x200CH
			0x200CL
			0x200DH
			0x200DL
stepValue	2	Changeable	0x200EH
			0x200EL
interval(10ms)	2	Changeable	0x200FH
			0x200FL
writeAddr0	2	Changeable	0x2010H
			0x2010L

_value0	2	Changeable	0x2011H
			0x2011L
.....		Changeable	.....
gradation	1	Changeable	0x2020H
reportToHost	1	Changeable	0x2020L

Len: The total number of bytes calculated from targetAddr to reportToHost (Len itself is not included), which is 65Bytes(0x41).

targetAddr: Target variable address.

_dataType: 0x80 = char; 0x00 = uchar; 0x81 = short; 0x01 = ushort; 0x82 = int; 0x02 = uint; 0x03 = longlong Before changing _dataType, developers must make sure that there is sufficient consecutive RAM spaces, starting from the address designated by targetAddr.

minValue & maxValue: Both parameters require 8 bytes (64bits) data length. If the data type is set as char, short, int, or longlong the input value can be negative. For other data types (uchar, ushort, and uint), the input value should be >= 0. In addition, negative number must be converted to two's complement. Refer to the examples show below:

Data Type	minValue/maxValue	Data read back from UartTFT IC
char	2	0x 00 00 00 00 00 00 00 02
	-2	0x FF FF FF FF FF FF FF FE
short	2	0x 00 00 00 00 00 00 00 02
	-2	0x FF FF FF FF FF FF FF FE
int	2	0x 00 00 00 00 00 00 00 02
	-2	0x FF FF FF FF FF FF FF FE

In the above table, the available digits for different data types are marked in green, and other digits are marked in red.

Example 1: If a _dataType is set as [short], to update the minValue to -9, the command will be as below:

0x10 parameterAddr + 0x0006 0xFFFF 0xFFFF 0xFFFF 0xFFF7

Example 2: If a _dataType is set as [short], to update the maxValue to 9, the command will be as below:

0x10 parameterAddr + 0x000A 0x0000 0x0000 0x0000 0x0009

gradation: 0x01 = ' + ' ; 0x00 = ' - '

reportToHost: 0x01 = Enable; 0x00 = Disable

10.5.16 Trend Graph: parameterAddr

Table 10-36: parameterAddr Related Content of Trend Graph

Parameter Name	Data Length/Bytes	Feature	Address
Len	1	Unchangeable	0x2000H
Xs	2	Changeable	0x2000L
			0x2001H
Ys	2	Changeable	0x2001L
			0x2002H
Xe	2	Changeable	0x2002L
			0x2003H
Ye	2	Changeable	0x2003L
			0x2004H
y_ReferenceLine	2	Changeable	0x2004L
			0x2005H
_referenceValue	2	Changeable	0x2005L
			0x2006H
Zoom	2	Reserved	0x2006L
			0x2007H
lineColor	3	Changeable	0x2007L
			0x2008H
			0x2008L
channel	1	Changeable	0x2009H
x_Spacing(Pixels)	1	Changeable	0x2009L
lineWidth	1	Changeable	0x200AH
0x00			0x200AL

Len: The total number of bytes calculated from Xs to linewidth (Len itself is not included), which is 20Bytes(0x14).

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.

Channel: For modifying the direction and designated channels. bit7: direction. 0x80 = L-R; 0x00 = R-L; bit0~6: channels. 0x00 = channel0; 0x01 = channel1; 0x02 = channel2, ...0x07 = channel7

10.5.17 Needle: parameterAddr

Table 10-37: parameterAddr Related Content of Needle

Parameter Name	Data Length/ Bytes	Feature	Address	Parameter Name	Data Length/ Bytes	Feature	Address
Len	1	Unchangeable	0x2000H	_promptNum_X	2	Changeable	0x200FH
writeAddr	2	Changeable	0x2000L				0x200FL
			0x2001H	_promptNum_Y	2	Changeable	0x2010H
background	2	Changeable	0x2001L				0x2010L
			0x2002H	_firstIcon	2	Changeable	0x2011H
X	2	Changeable	0x2002L				0x2011L
			0x2003H	_lastIcon	2	Changeable	0x2012H
Y	2	Changeable	0x2003L				0x2012L
			0x2004H	_alignment	1	Changeable	0x2013H
W	2	Changeable	0x2004L	_integerDigit	1	Changeable	0x2013L
			0x2005H	_decimalDigit	1	Changeable	0x2014H
H	2	Changeable	0x2005L	needleType	1	Unchangeable	0x2014L
			0x2006H	needle_W	2	Changeable	0x2015H
pivot_X	2	Changeable	0x2006L				0x2015L
			0x2007H	needle_L1	2	Changeable	0x2016H
pivot_Y	2	Changeable	0x2007L				0x2016L
			0x2008H	needle_C1	3	Changeable	0x2017H
startAngle	2	Changeable	0x2008L				0x2017L
			0x2009H				0x2018H
finalAngle	2	Changeable	0x2009L	needle_L2	2	Changeable	0x2018L
			0x200AH				0x2019H
step	2	Unchangeable	0x200AL	needle_C2	3	Changeable	0x2019L
			0x200BH				0x201AH
swing	1	Changeable	0x200BL				0x201AL
pivotIcon	2	Changeable	0x200CH	Pointer_sid	2	Unchangeable	0x201BH
			0x200CL				0x201BL
showNumber	1	Changeable	0x200DH	Pointer_eid	2	Unchangeable	0x201CH
_numberAddr	2	Changeable	0x200DL				0x201CL
			0x200EH	needleIcon	2	Unchangeable	0x201DH
_dataType	1	Changeable	0x200EL				0x201DL

Len: The total number of bytes calculated from writeAddr to needleIcon (Len itself is not included), which is 59 (0x3B) Bytes.

Background: To modify this parameter, developers should make sure the designated picture has been included in the current UartTFT-II_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. **showNumber**: 0x00 = Disable; 0x01 = Enable

_numberAddr: The address of the Graphics Number

_dataType: 0x80 = char; 0x00 = uchar; 0x81 = short; 0x01 = ushort; 0x82 = int; 0x02 = uint; 0x03 = longlong Before changing _dataType, developers must make sure that there is sufficient consecutive RAM spaces, starting from the address designated by _numberAddr.

_promptNum_X & _promptNum_Y: The coordinate of the prompt number.

_alignment: There are 3 alignment modes. 0x00: Left; 0x01: Middle; 0x02: Right. In addition, the bit7 of this parameter is used to control the setting of leadingZero, where bit7= 1 is Enable, and bit7=0 is Disable.

10.6 Widget Trigger: triggerValue

Table 10-38 shows the list of widgets that can be triggered by Host through the parameter, triggerValue:

Table 10-38: 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
SlideMenu	N	Timer	N
Popupbox	Y	Gif	N
Variable Button	Y	QRCode	N
Multi-Variable Button	Y	Audio Play	N
Circular Touch	N	Progress Bar	N
Progress Bar	N	Circular Progress Bar	N
Numeric Keypad	Y	Bit Status	N
CN_Keyboard	Y	Icon	N
EN_Keyboard	Y	Trend Graph	N
SingleKey	N	Encoder	N
String_Label	N	Video Play	N
Text Scroll	N	Camera	N
Text Number Display	N	Automatic Variable	N
Graphics Number Display	N	Needle	N
Analog Clock	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.

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.

EastRising 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 Master.

3. UartTFT controller only supports Register operation if it is used as Slave.

11.1Create a ModBus Command File

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

Click on Tool menu, and select [Modbus] to enter Modbus command setting page, as shown below:

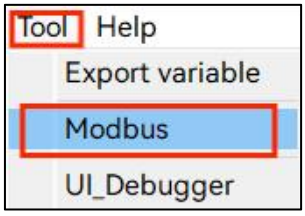


Figure 11-2: Enter ModBus Command Setting Page

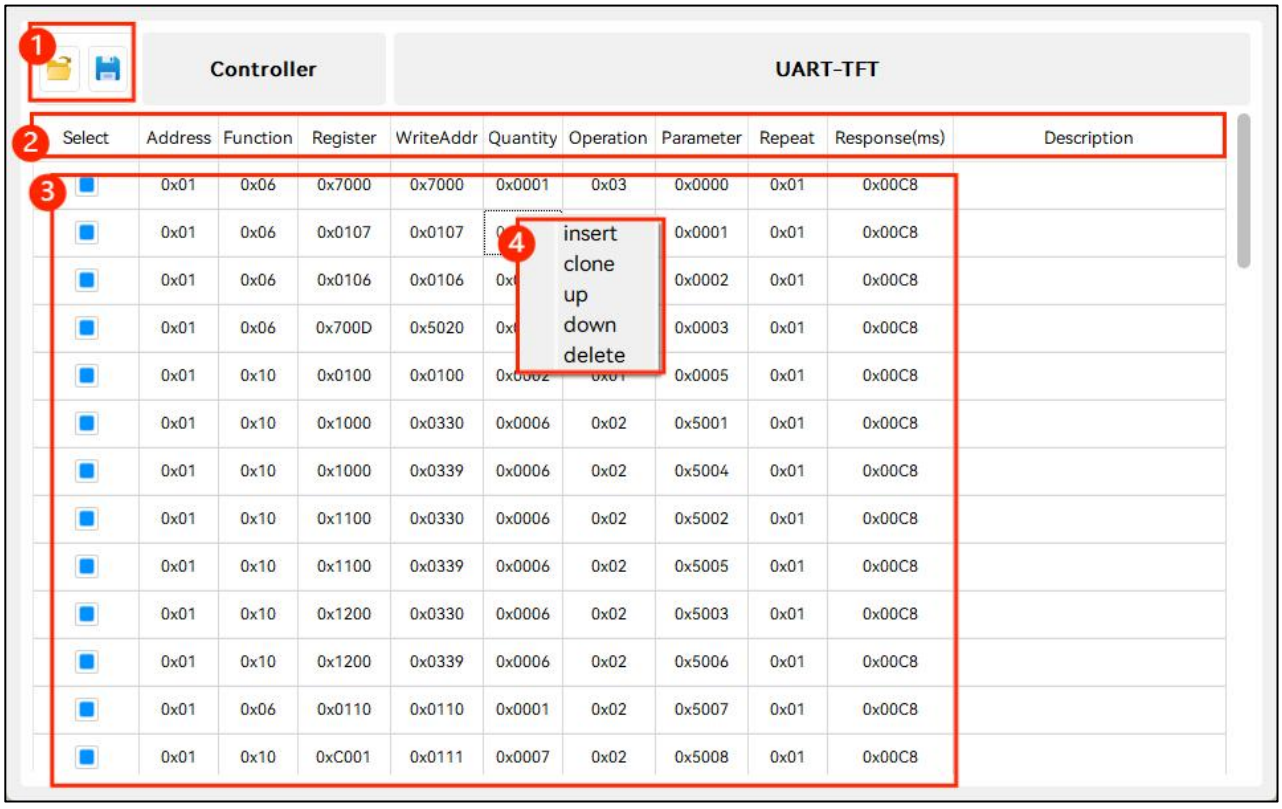


Figure 11-3: ModBus Command Editing Page

❶ Click on  to import the command.list. Click on  to save as 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

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 cliking on  before exporting UartTFT-II_Flash.bin

11.3 ModBus Command Structure

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 Parameter	Command Mode	Repeat Times	Response Time
Bytes	1	1	2	2	2	2	1	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 Write/Read operation) of the slave.

Master variable address : The variable starting address of 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.

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**	**0xC8**

This command will be sent to the Slave whose address is 0x01. The Slave will then responds 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.

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	0xC8

Command 0x04 is used by Master to read input register from 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**	**0xC8**

This command will assign the 2 bytes data stored in the designated address (0x0020) of 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**	**0xC8**

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**	**0xC8**

This command will read 10 (0x000A) coils status starting from the designated Slave coil address. The received data will be allocated to Master variable address calculated as below:

(1) Slave coil address % 0x10 = 0x0009 % 0x10 = 9 → the read data will be stored to 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	0xC8

The allocation method of the read data is the same as the one described above in "Master Read Coil Status"

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**	**0xC8**

This command will write data to a designated Slave coil address. The written data is based on the content of 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	Function	Coil address	Coil status	CRC

	address (1 Byte)	code (1 Byte)	(2 Bytes)	(2 Bytes)	(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**	**0xC8**

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 Master variable address 0x0001; and Slave coil address, 0x0017, is related to bit7 of 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 010**0 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 MCU_Code needs to be set accordingly.

Set the device address (Device Addr, **must NOT be 0x00**) in the Project Setting page, as shown below:

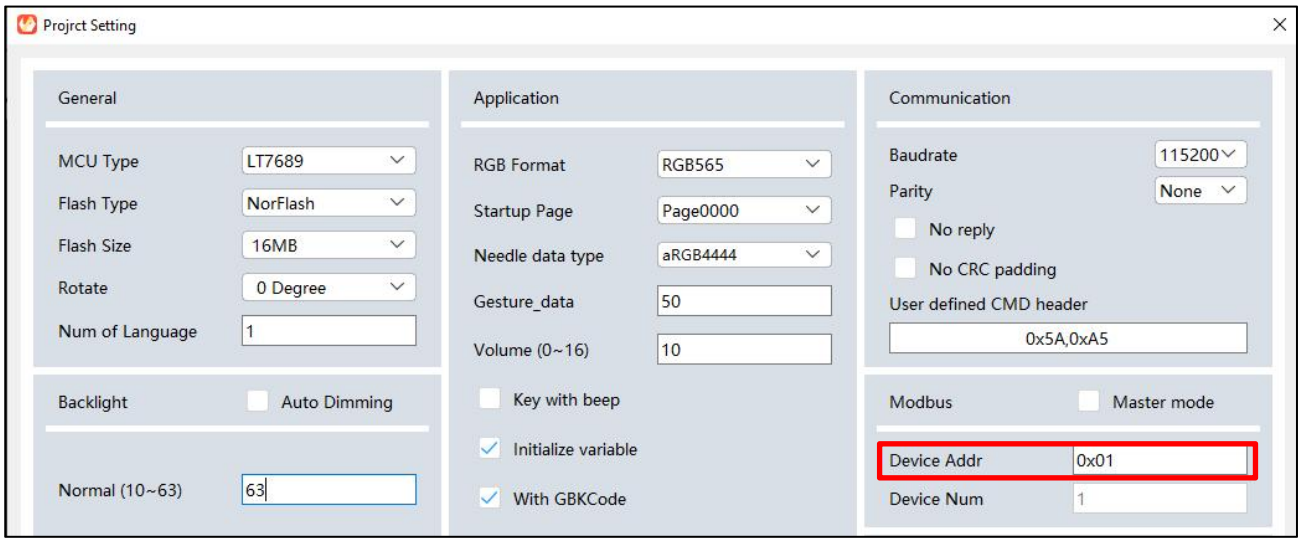


Figure 11-4: Setting Slave Mode

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 MCU_Code needs to be set accordingly.
Check the [Master mode] in the Project Setting page. No need to set the device address. As shown below:

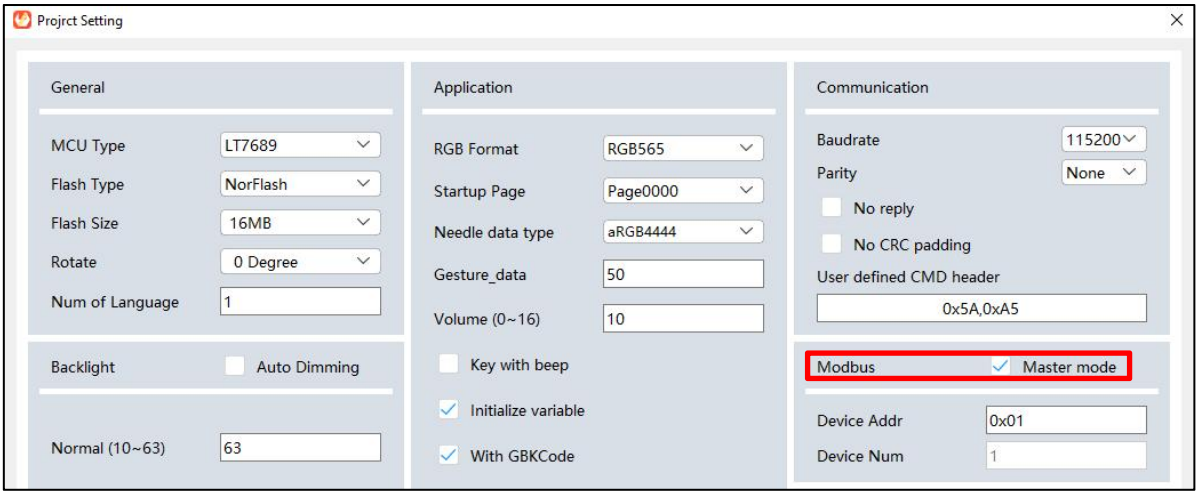


Figure 11-8: Setting Master Mode

11.7 Modbus Operation Mode Setting Tutorial

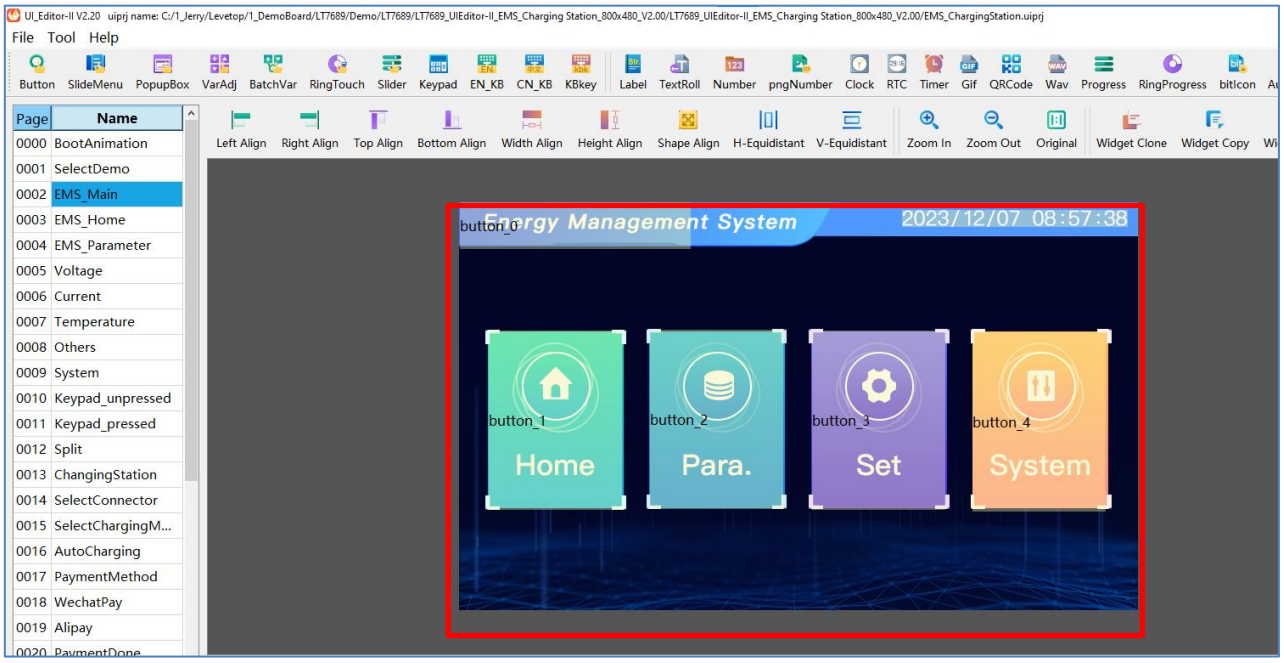
11.7.1 Operation Mode – 0x00

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

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:



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

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

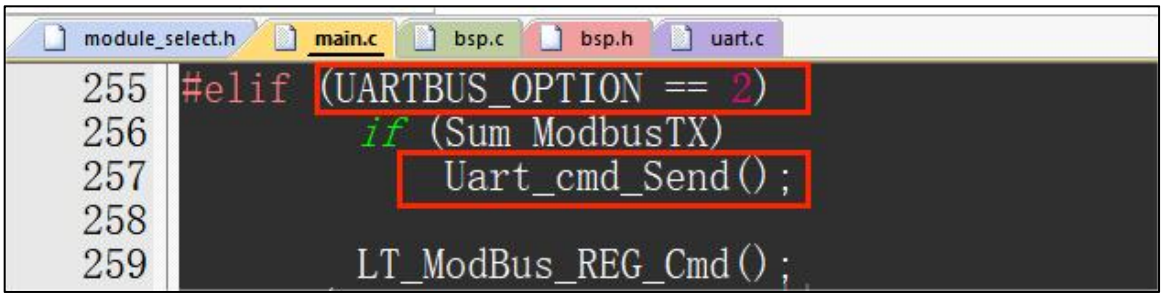
11.7.4 Operation Mode – 0x03

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

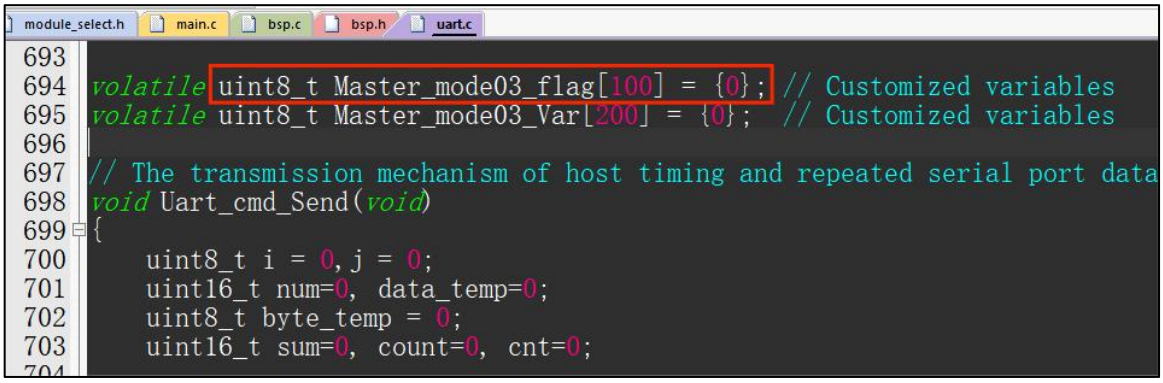
As shown in the above table, the [Operation] mode is 0x03, which means the command will be executed when the value of the designated location is set to 1 in the MCU_Code. The location is assigned to [Parameter], which is 0x0001 in this example.

Following is a MCU_Code setting example:

1. In the MCU_Code, locate the function: Uart_cmd_Send()



2. Find the location array, 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.

```
module_select.h | main.c | bsp.c | bsp.h | uart.c
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
```

In Basic_touch(), locate the below coniditon 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)  // The button is touched and released
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           // Enter when the returnValue = 0x0022
12348       {
12349       Master_mode03_flag[0x0001] = 1;  // Set the designated location to 1
12350       }
12351   }
```

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

12 Additional Information

12.1 Codes & Documents

Followings are the codes and documents related to a UI_Editor-II project:

bootloader: This is the code that enables UartTFT controller to download MCU_Code.bin and UartTFT-II_Flash.bin.

MCU_Code.bin: This is the code that enables UartTFT controller to implement the display functions and operations edited on UI_Editor-II. 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-II_Flash.bin: This file is generated by UI_Editor-II after compilation. It includes all the required materials and settings that developers design on UI_Editor-II. UartTFT-II_Flash.bin is programmed to external SPI Flash. Its size varies according to the imported materials.

12.2 Using Existed Project to Create 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 13-1

(3) Create a new project with the copied material

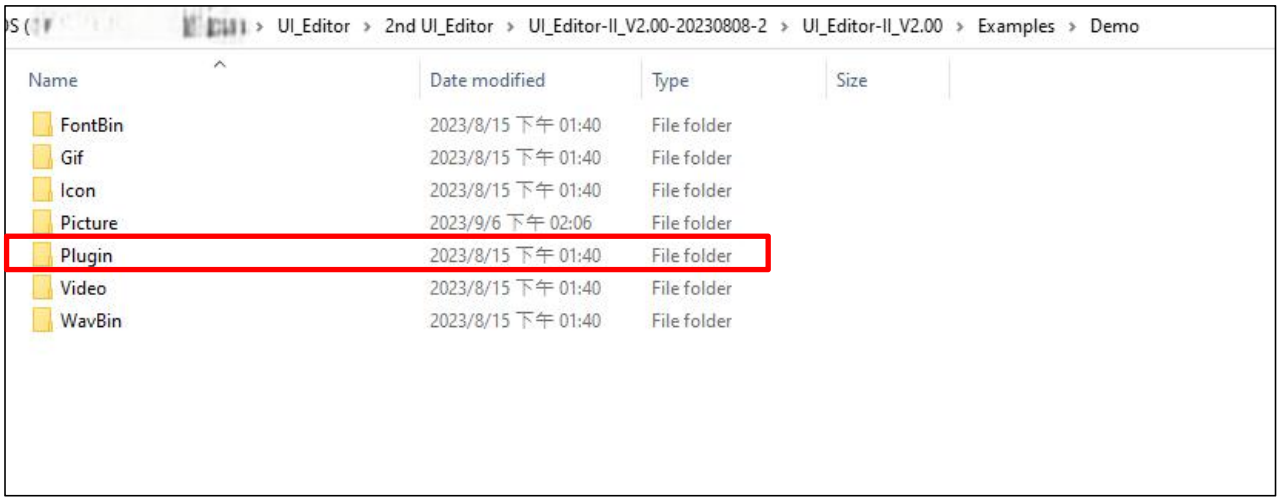


Figure 13-1: Delete the Files in Plugin Folder

12.3 Screen Rotation

12.3.1 Screen Rotation for MCU LT7689

The method of rotating a screen differs based on the MCU used. For HMI displays utilizing the MCU LT7689, achieving screen rotation can be as straightforward as adjusting the Rotate parameter within the Project Setting section of the UI Editor software, while keeping other settings unchanged.

Note:

1、No need to modify the panel resolution settings.

2、Rotating direction is clockwise

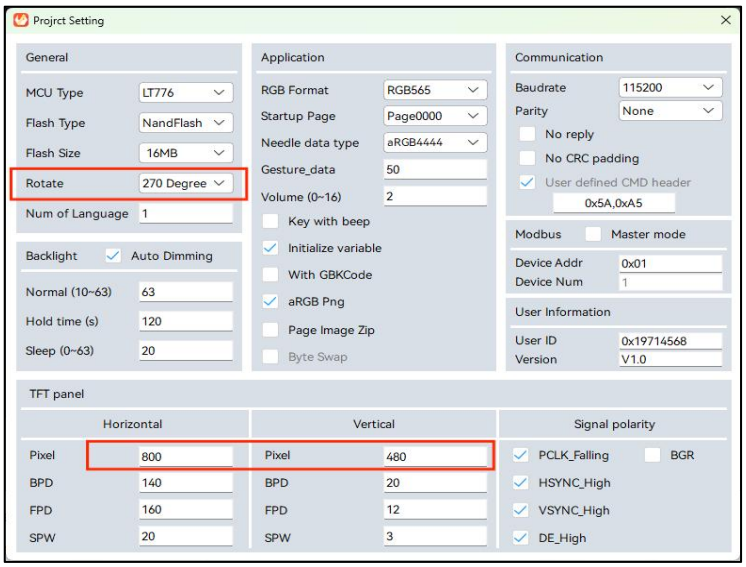


Figure 13-2: Set [Rotate] Parameter

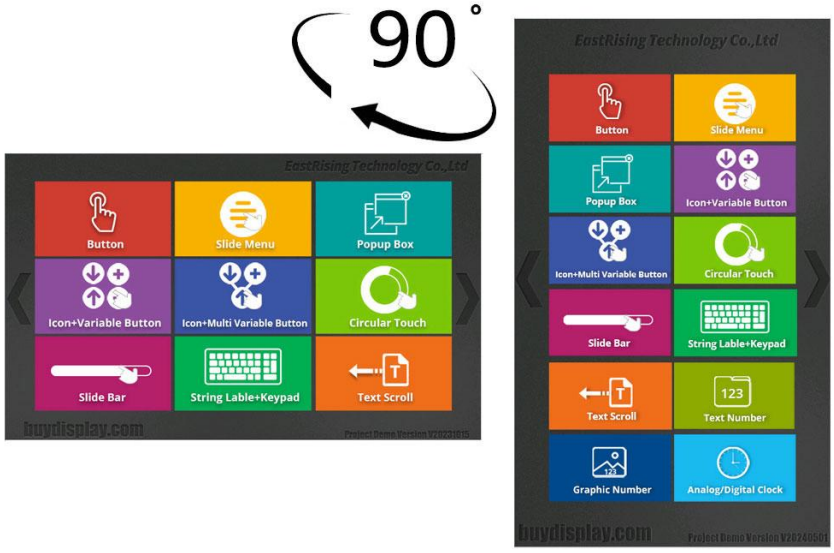


Figure 13-3: Rotate 90° Clockwise

12.3.2 Screen Rotation for MCU LT168A

For HMI displays utilizing the MCU LT168A， you need to update MCU code for screen rotation. The related program can be downloaded from our tutorial page online,then follow 13.1 programming for next operation.

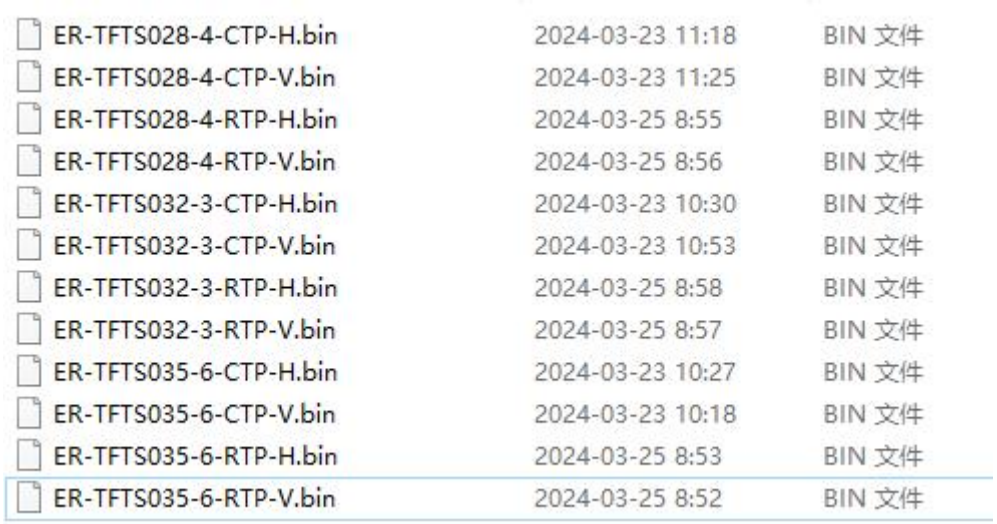


Figure 13-3-1：Screenshot for MCU Code

Note: CTP stands for Capacitive Touch Panel, RTP for Resistive Touch Panel, H for Horizontal, and V for Vertical. Please select the appropriate MCU program for updating based on your actual situation.For example, file ER-TFTS035-6-RTP-V is the MCU code for HMI display ER-TFTS035-6 with resistive touch panel and vertical screen display.

12.4 UartTFT-II_Flash.bin

A UartTFT-II_Flash.bin contains font, Gif, pictures, Wav, and page information. Since the size of a UartTFT-II_Flash.bin varies according to the materials used, developers should make sure if the SPI Flash has enough room for storing the UartTFT-II_Flash.bin

Among the materials used in UI_Editor-II, 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-II. Each frame is taken as a picture. Therefore, a Gif with high frame count will consume a great amount of spaces. As shown in Figure 13-4, the size of the converted bin file is over 8 times bigger than the original one.

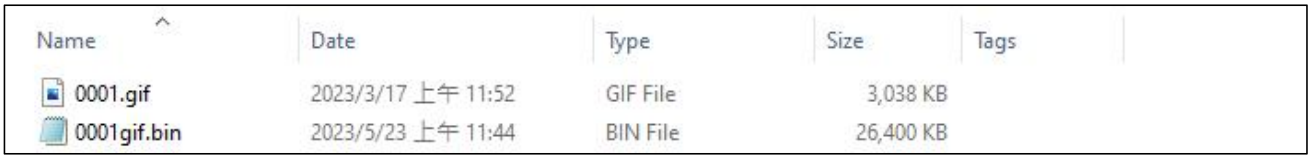


Figure 13-4：Gif converted to bin

12.5 Data Type

Table 13-1: Data Type List

Type	Address	Length	Max. Value	Range
long long	0x0000	**8bytes**	0x7FFF	-2^63 ~ 2^63-1
	0x0001		0xFFFF	
	0x0002		0xFFFF	
	0x0003		0xFFFF	
int	0x0004	**4bytes**	0x7FFF	-2^31 ~ 2^31-1
	0x0005		0xFFFF	
uint	0x0006	**4bytes**	0xFFFF	0 ~ 2^32-1
	0x0007		0xFFFF	
short	0x0008	**2bytes**	0x7FFF	-2^15 ~ 2^15-1
ushort	0x0009	**2bytes**	0xFFFF	0 ~ 2^16-1
char	0x000AH	-	0x00	-
	0x000AL	**1byte**	0x7F	-2^7 ~ 2^7-1
unchar	0x000BH	**-**	0x00	-
	0x000BL	**1byte**	0xFF	0 ~ 2^8-1

12.6 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: 5 digits, int: 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.

12.7 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、Thur、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.

12.8 Widget Initial Setting

When multiple widgets share the same variable address, their initial settings should be the same as well. For example, the parameter, default Number, should be the same for all the Text Number Display widgets with the same variable address.

12.9 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.

12.10 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 13-5;

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 13-5: Locate the Image Item

(2) Double click on the image item, and a file manager window will pop up, as shown in Figure 13-6;

(3) Click on [Cancel] to close the window;

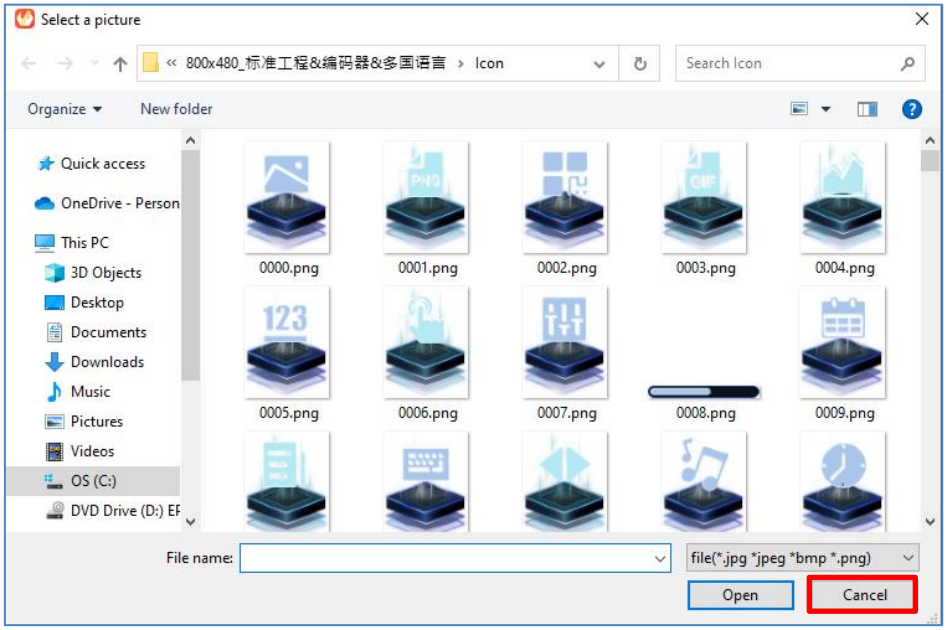
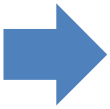


Figure 13-6: File Manager Window

(4) Delete the image name in the Parameter Setting Window, as shown in Figure 13-7, and then click on [Enter] to confirm the operation. The final result is as show in Figure 13-8

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



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	
pressedIcon	
reportToHost	Disable

Figure 13-7: Delete the Selected Image

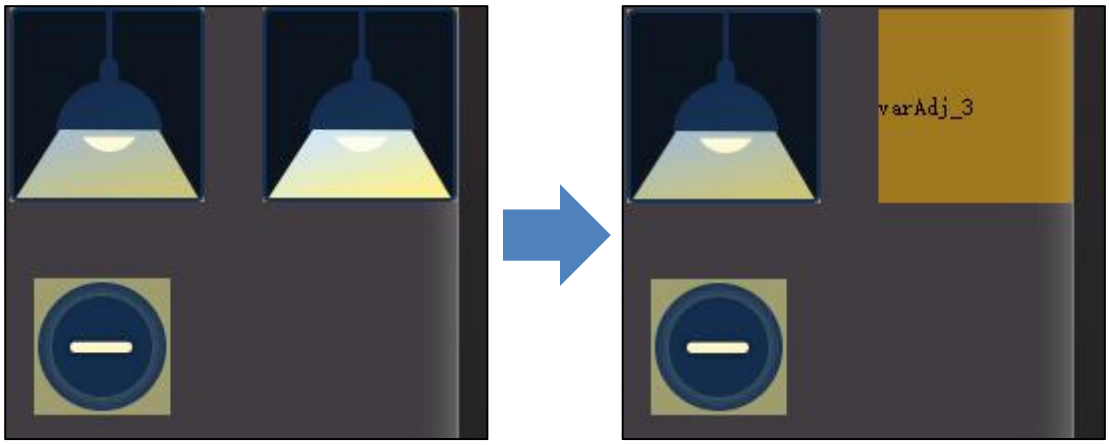


Figure 13-8: Operation Result

12.11 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 13-2, 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 13-2: 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

12.12 Widget Overlap

To avoid false operations, widgets with touch functions cannot be overlapped with each other.

12.13 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)

12.14 Display Scaling

Due to various computer resolutions, UI_Editor-II may not be displayed properly for certain cases, as shown in Figure 13-9. Developers may improve it by adjusting the display scaling, as described below. (Only available in Win10)

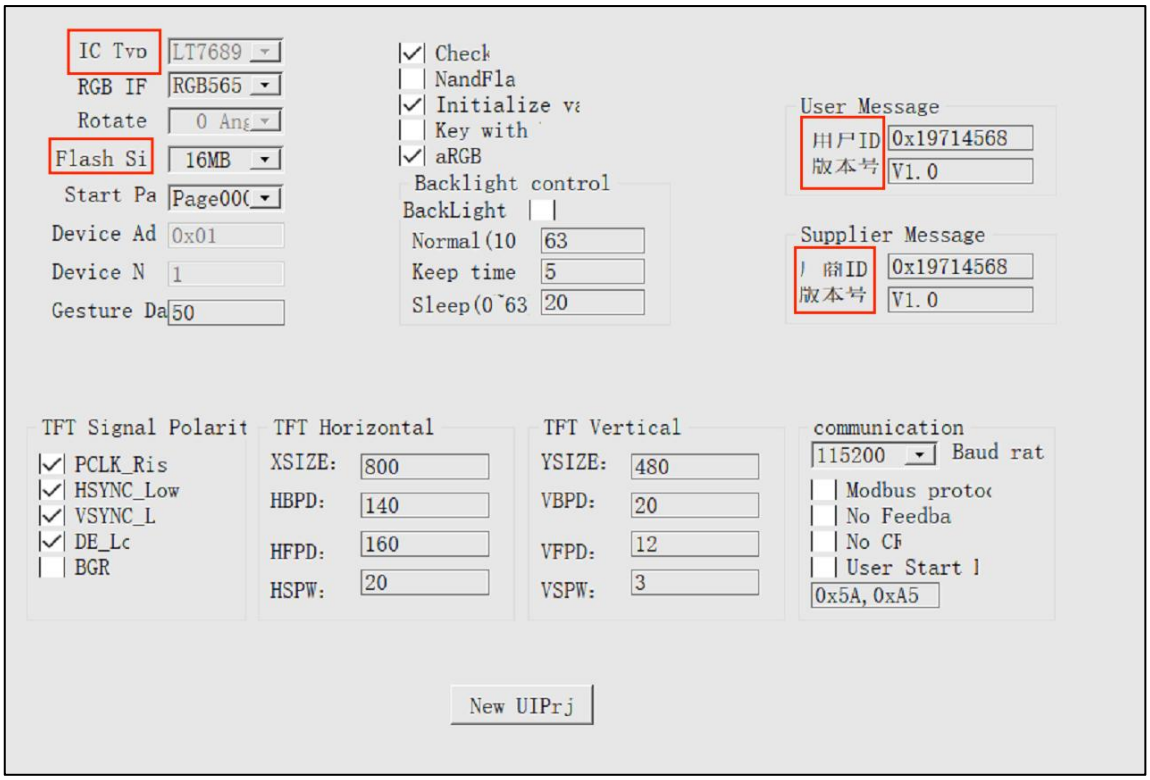


Figure 13-9: Program Display Issue

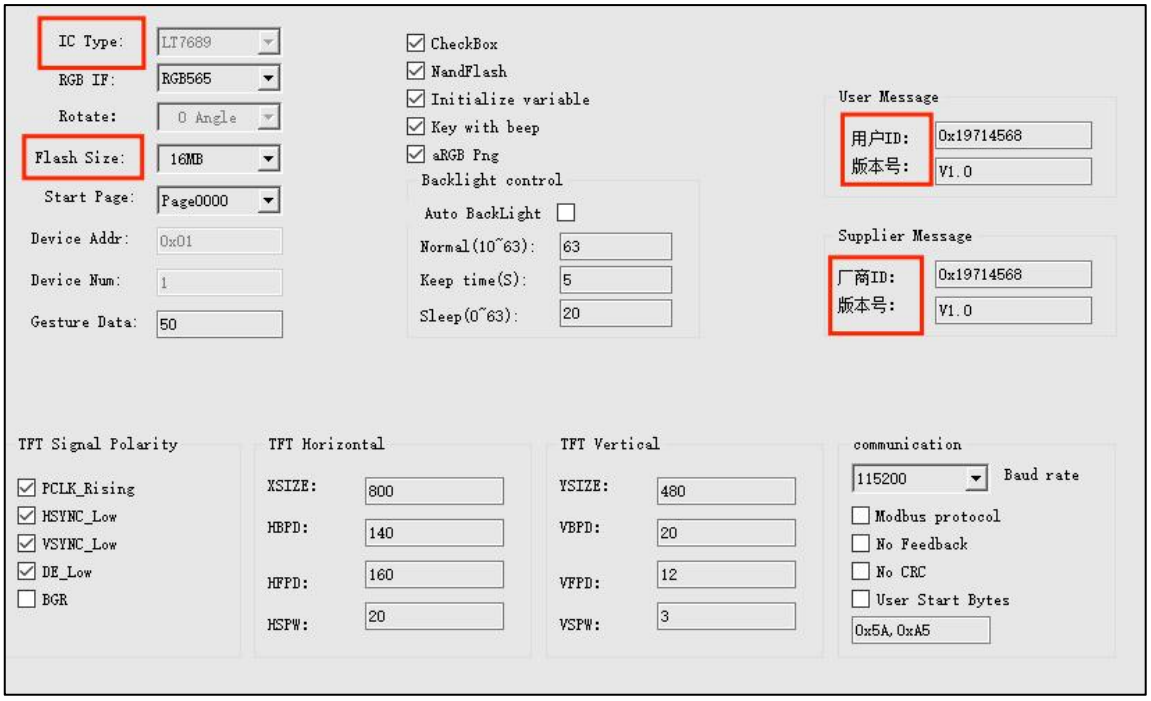


Figure 13-10：Normal Display

Step 1: Close UI_Editor-II, and then right click on the EXE file. Select [Properties] from the pop-up window.

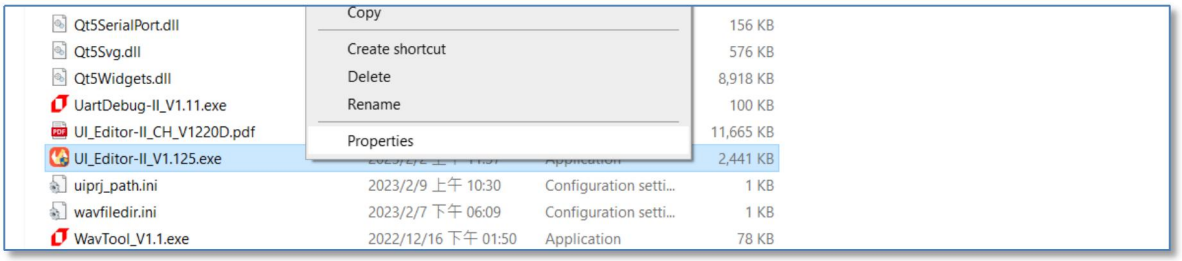


Figure 13-11：Open [Properties] Window

Step 2: Click on [Compatibility] page, and then click on [Change high DPI settings]

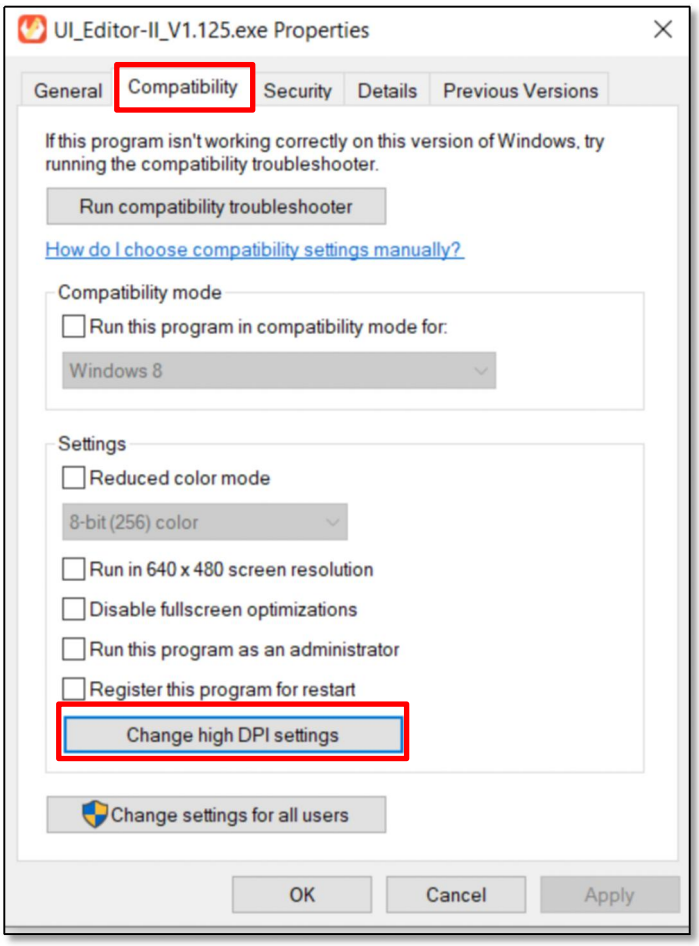


Figure 13-12: Change DPI Setting (1)

Step 3: Check [Override high DPI scaling behavior], and then select [System (Enhanced)]. Next, click [OK] to confirm the operation.

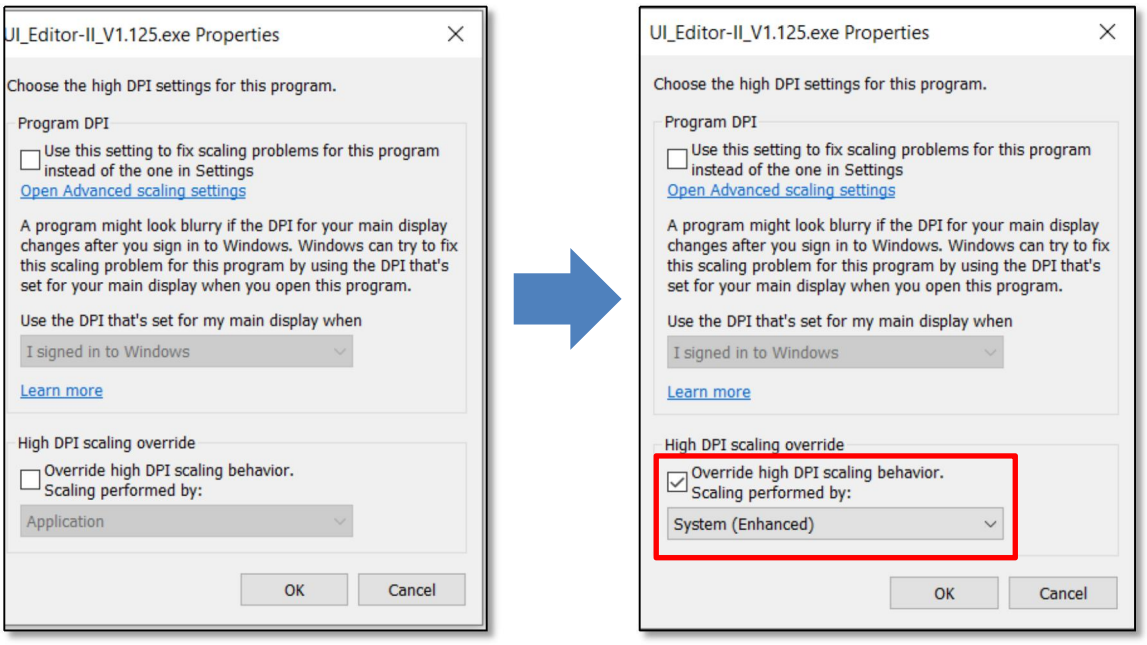


Figure 13-13: Change DPI Settings (2)

Step 4: Click on the [OK] button in the [Compatibility] page to finish the setting.

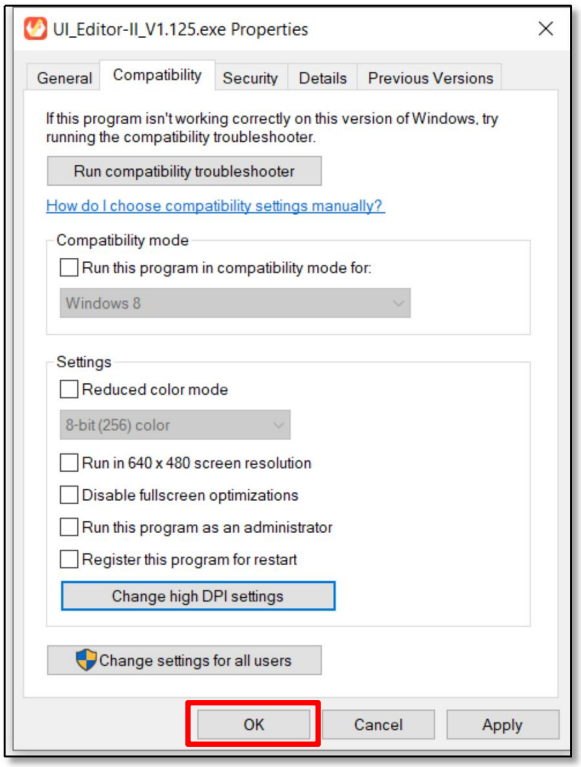


Figure 13-14: Confirm the Change

12.15 Computer OS

Preferred OS: Win10 or above. It is suggested that developers operate UI_Editor-II in Full Screen mode.

12.16 Naming Rule

The names of material, widgets, pages, and projects should not include special characters as shown in Table 13-3. There is only one decimal point "." allowed before the file suffix.

Table 13-3: Illegal Symbol List

Mode	EN	EN	EN	CN/EN	CN/EN	EN	EN	CN/EN	CN	CN
Symbol	\	/	:	*	?	<	>	|	、	，

12.17 Material Library

EastRising provides a public Material Library which contains various icons, and pictures etc. Developers can download in free from this link.

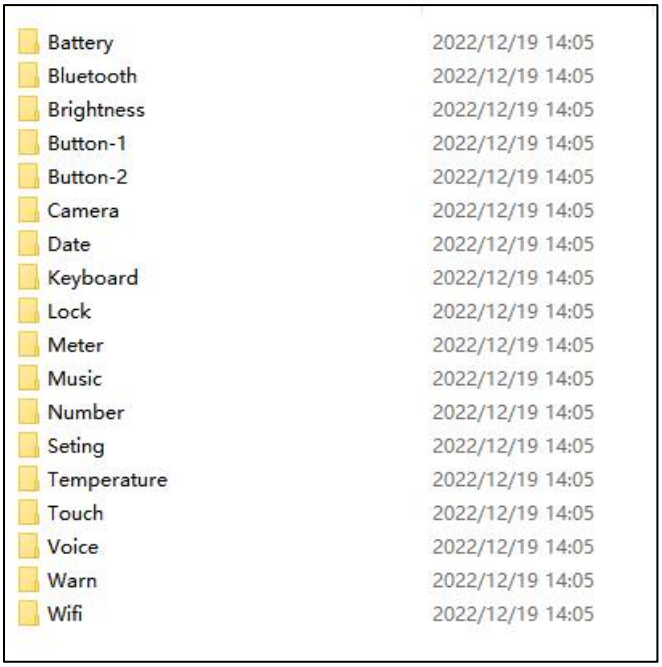


Figure 13-15: Material Library

12.18 dataFormat

12.18.1 Structure of Various dataFomat

1、dataFormat supported by LT7689:

The dataFormat described below is based on a single Pixel.

RGB888: Each pixel is represented by 24bits data, as the structure shown in Table 13-4:

Table 13-4: RGB888 Data Structure

dataFormat	Red	Green	Blue
RGB888	bit 23~16	bit 15~8	bit 7~0
	R7~R0	G7~G0	B7~B0

RGB565: Each pixel is represented by 16bits data, as the structure shown in Table 13-5:

Table 13-5: RGB565 Data Structure

dataFormat	Red	Green	Blue
RGB565	bit 15~11	bit 10~5	bit 4~0
	R7~R3	G7~G2	B7~B3

Softpng: Each pixel is represented by 16bits data. The data structure is the same as RGB565. In addition, each pixel data will be converted by UI_Editor-II, according to α value of the PNG picture. If α value of a pixel >= 127, the pixel data will be saved as the original RGB565 format. If α value of a pixel < 127, then the pixel data will be saved as 0x0000.

αRGB4444: Each pixel is represented by 16bits data, as the structure shown in Table 13-6:

Table 13-6: αRGB4444 Data Structure

dataFormat	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%.

2、dataFormat supported by LT269/LT268C/LT268D/LT776/LT3688:

RGB565: Same as described above

softpng: Same as described above

αRGB4444: Same as described above

RGB565_zip: Zip format of RGB565, compressed for saving spaces.

Softpng_zip: Zip format of Softpng, compressed for saving spaces.

αRGB4444_zip: Zip format of αRGB4444, compressed for saving spaces

αRGB8565: Each pixel is represented by 24bits data, as the structure shown in Table 13-7:

Table 13-7: αRGB8565 Data Structure

dataFormat	Transparency α	Red	Green	Blue
αRGB8565	bit 23~16	bit 15~11	bit 10~5	bit 4~0
	α 7~0	R7~R3	G7~G2	B7~B3

12.18.2 dataFormat – Icon and Gif

When generating the UartTFT-II_Flash.bin, UI_Editor-II will convert the imported pictures based on the dataFormat settings.

When dataFormat is not set → BMP and JPG pictures will be converted to RGB888 or RGB565 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 RGB565, RGB888, or RGB555_zip

2、No need to set dataFormat for BMP and JPG pictures.

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.

13 Appendix

13.1 Programming

Please refer to the below procedure or learn from video tutorial page online:

13.1.1 Programming by SD Card

Users can program MCU.code or UartTFT_Flash.bin to HMI display through an SD card.

(1) Format an SD card / FAT32

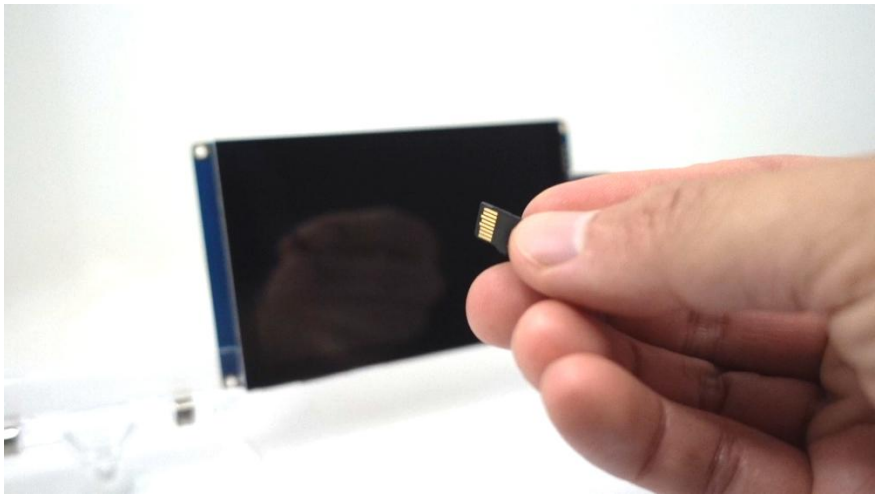
(2) Make two directories and name them as [MCU_Code], and [UartTFT_Flash] respectively.

(3) Save the bin files that will be programmed to the corresponding directories, as shown below:

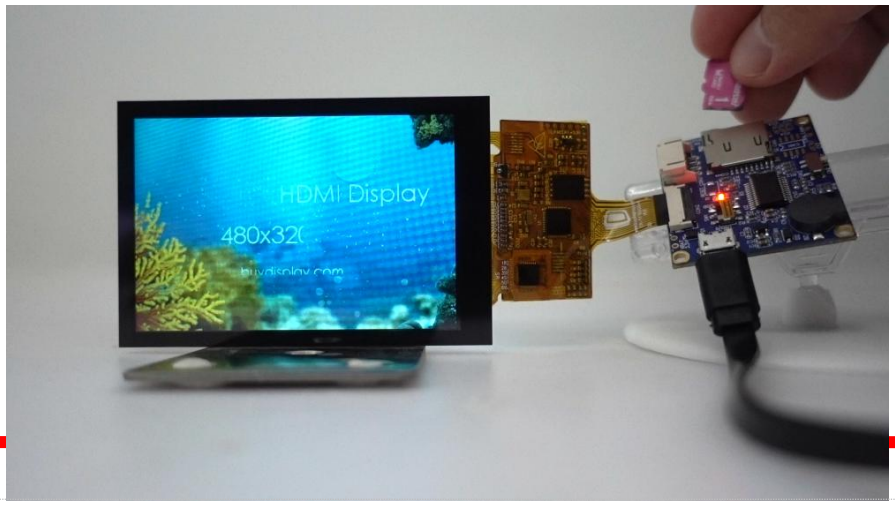
(The name of the files and directories MUST be exactly the same as shown in Figure 14-4)

Figure 14-4: Make two directories

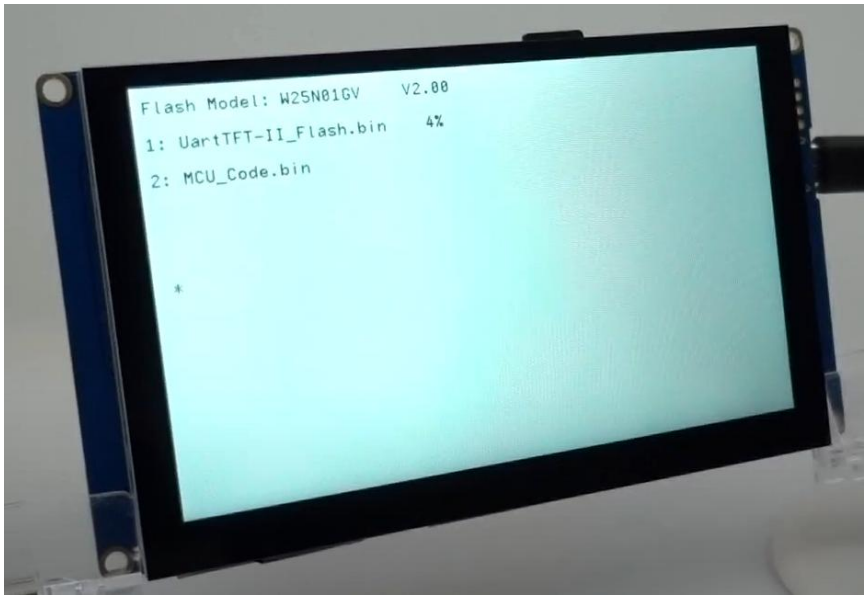
(4) Make sure that HMI display is power off, and then insert the SD card if the MCU is **LT7689**.



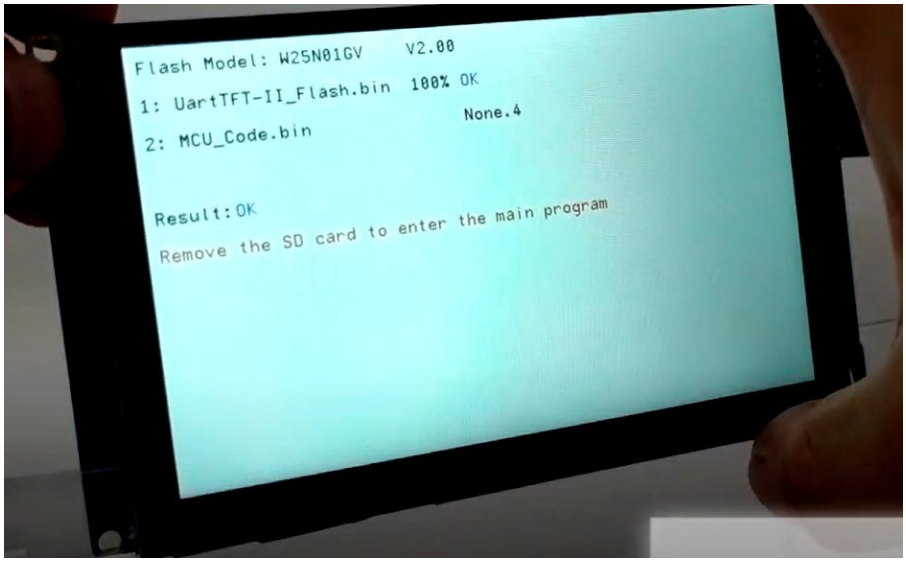
HMI display should be power on, and then insert the SD card to debugging board if the MCU is **LT168A**.



(5) After inserting the SD card, powering on the HMI Display will automatically initiate the update process.

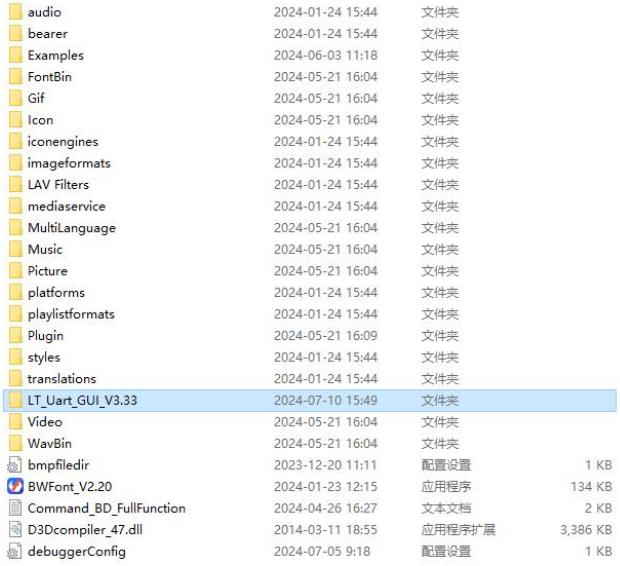


(6) As soon as the CRC checking is passed, a message of "removing the USB disk or SD card" will be prompted. Once the USB disk or SD card is removed, HMI display will enter the main program.

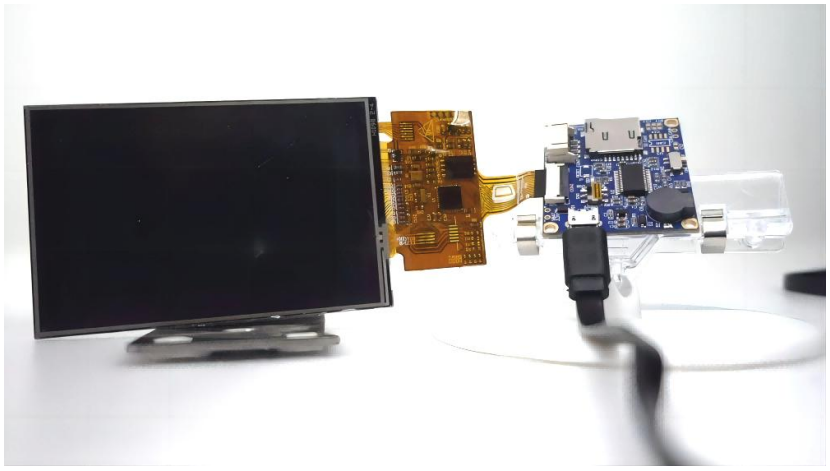


13.1.2 Programming by UART Port

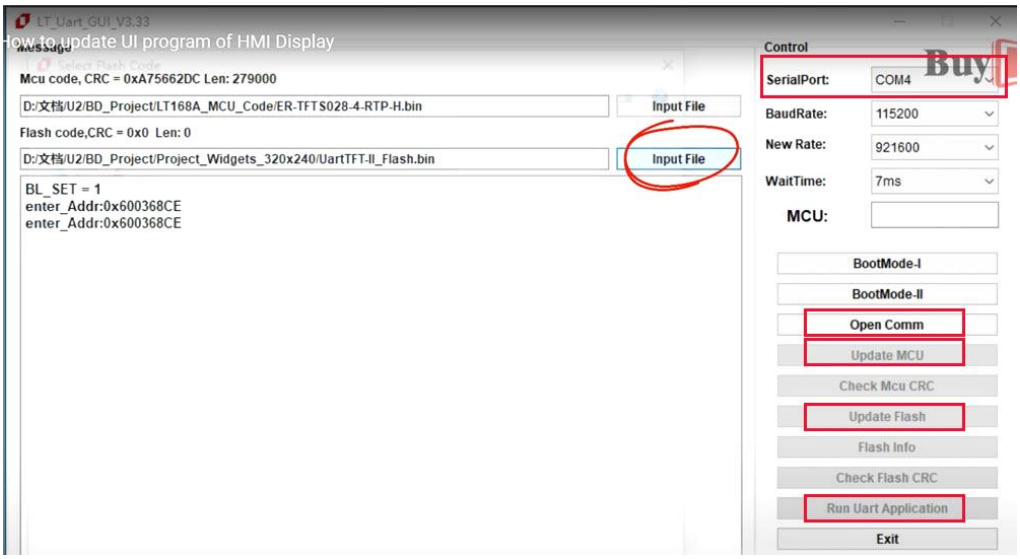
Users can program UartTFT_Flash.bin to HMI display through UART port.
To update via the UART port, you will require the LT_Uart_GUI software and a debugging board. The LT_Uart_GUI software is already included in the download package of the UI Editor, while the debugging board needs to be purchased separately.



To initiate the process, launch the LT UART GUI and connect the HMI Display to the debugging board. Then, insert the USB cable extending from the debugging board into a USB port on your PC. It is recommended that your PC's operating system be Windows 10 or above, as the system will automatically install the necessary drivers upon recognizing the debugging board. This recognition will subsequently trigger changes in the SerialPort configuration.



Next, click on "Open Comm" to establish a connection with the open port. Afterward, navigate to "Input File" and specify the file paths for both the MCU Code and Flash Code that you intend to program. Once the paths are correctly set, execute the "Update MCU" and "Update Flash" commands sequentially to program each component individually. Upon successful programming, click on "Run Uart Application" to initiate the application. Once the application is running, the HMI Display will display the intended graphics or interface, indicating a successful burn process.



Note: When programming UartTFT_Flash.bin, it usually takes a little more time to go through the [erase] and [write] operations because of the characteristics of the SPI Flash.

13.2 Setting Limits

MCU Model	LT168A	LT7689
User address range	0x0000 ~ 0x1FFF	0x0000 ~ 0x5FFF
PNG size limitation	W*H <= 480*320	No limitation
Circular touch/progress bar	W=H<= Y resolution of the panel	W=H<= Y resolution of the panel
Analog Clock	W=H<= Y resolution of the panel	W=H<= Y resolution of the panel
Trend graph area	W*H <= 480*320	No limitation
Picture format	RGB565	RGB565 RGB888
Picture size limit for Keybpard widget	W*H <= 480*320	No limitation
Area limits for SlideMenu widget	W*H <= 480*320	No limitation
Picture size limit for SlideMenu	No limitation	W*H < 384000
Slide to jump – with sliding effects	NA	Support[1]
PopupBox background dimming	NA	Support
Page Picture Compression	Support	NA
Icon & Gif Compression	Support	NA

13.3 Maximum Amount of Widgets in a Single Page

The amount 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 amount 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 14-3: Maximum Amount of Widgets in a Single Page

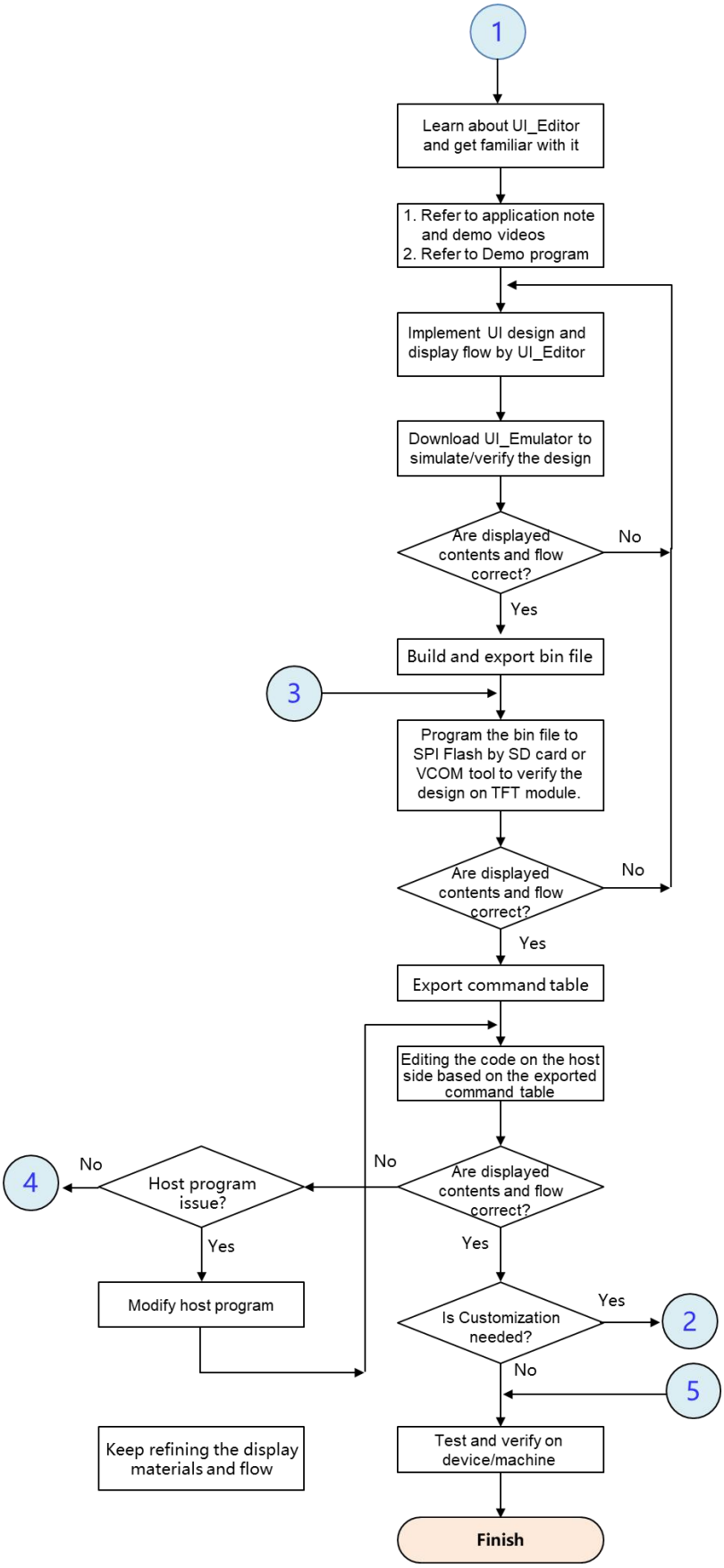
IC Model	LT168A/ LT168B	LT7689
Button	20	20
SlideMenu	6	6
PopupBox	8	8
Variable Button	10	10
Multi-Variable Button	20	20
Circular Touch	4	4
Slider Bar	4	4
SingleKey	60	60
Numeric Keypad	20	20
EN_KeyBoard	10	10
CN_KeyBoard	10	10
String_Label	200	200
Text Scroll	4	4
Text Number Display	30	30
Graphics Number Display	30	30
Analog Clock	2	2
Digital Clock	6	6
Gif	20	20
QRCode	16	16
Audio Play	1	1
Progress Bar	4	4
Circular Progress Bar	4	4
Bit Status	64	64
Icon	64	64
Trend Graph	8	8
Encoder	1	1
Timer	8	8
Automatic variable	4	4
Needle	4	4

Table 14-4: Registers Addresses by IC Models

13.4 Registers Addresses by IC Models

IC Model	LT168A/ LT168B		LT7689		
Range of User Address	0x0000 ~ 0x1FFF		0x0000 ~ 0x5FFF		
Page Register	0x7000				
Backlight Register	0x7001				
Time Registers	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 upgrade mode	0x7011				
Registers for Video Play	0x7012~0x702D				
Register for multiple language	0x703F				

13.5 Development Flow



14 Copyright

This document is the copyright of EastRising Technology co.,ltd. No part of this document may be reproduced or duplicated in any form or by any means without the prior permission of EastRising.

The information appearing in this Datasheet is believed to be accurate at the time of publication. However, EastRising assumes no responsibility arising from the use of the specifications described.

The applications mentioned here in are used solely for the purpose of illustration and EastRising makes no warranty or representation that such applications will be suitable without further modification, nor recommends the use of its products for application that may present a risk to human life due to malfunction or otherwise.

EastRising's products are not authorized for use as critical components in life support devices or systems.

EastRising reserves the right to alter its products without prior notification. For the most up-to-date information, please visit our web site at https://www.buydisplay.com