



AGN Series Embedded LCD Development Guide

MODULE NO:

AGN320240A00-3.5N12NSH-R

AGN480272A00-4.3N12NSM-R

AGN800480A00-7.0N12NSM-R

REVISION NO: V1.1

Version	Date	Note
V1.0	2019.2.13	The first release
V1.1	2019.2.26	The second release

Contents

1	Quick Demo Guides	2
1.1	Components of Demo Set	2
1.2	Connection to Power Supply	2
1.3	Basic Feature Demo	3
1.4	Connection to PC	3
1.5	Configure UART Interface in PC	3
2	System Development	4
2.1	Introduction to the Development Environment	4
2.2	The development advantages of Embedded LCD	5
2.3	Comparison of Development Examples	6
3	Configuration of Embedded LCD	7
3.1	Format Requirements for files	7
3.1.1	Picture File	7
3.1.2	Icon Library File	7
3.1.3	Font Library File	7
3.1.4	Data Format	8
3.1.5	Color Data Format	8
3.2	Configuration Files	8
3.3	File / Data Storage	8
3.3.1	FLASH Storage	9
3.3.2	RAM Storage	10
3.3.3	Configuration Register	12
3.3.4	Curve Buffer	12
3.4	Configuration File	12
3.4.1	System Configuration File (CONFIG.txt)	12
3.4.2	Variable Initialization File (22*.bin)	16
3.5	Device Configuration	16
3.5.1	Touch Panel Calibration	16
3.5.2	Download File by SD card	17
4	UART Communication	18
4.1	UART Testing	19
4.2	Command	19

4.3	Troubleshooting of UART Communication	20
5	Configuration Register	21
5.1	Configuration Register Table	21
5.2	Examples of register applications	24
5.2.1	Read/Write RTC	24
5.2.2	Read Front Library	25
5.2.3	Read/Write Database	25
5.2.4	Key Code Control.....	26
6	Touch/keying Configuration File.....	27
6.1	Touch / Keying Function List	28
6.2	Data Input from Touch Screen	28
6.2.1	Number Input	29
6.2.2	Text Input	30
6.3	Pop-up Menu	33
6.4	Incremental Input	34
6.5	Drag Input.....	35
6.6	RTC Control.....	35
6.7	Key Value	36
6.8	Hardware parameter configuration.....	36
6.9	Touch Synchronous Data Return	38
6.10	Rotation input	39
7	Variable Display Configuration File	40
7.1	Variable Display Function List	41
7.2	Icon Display.....	42
7.2.1	Variable Icon Display (0x00).....	42
7.2.2	Animation Icon Display	43
7.2.3	Progress Bar Display.....	43
7.2.4	Artistic character display	44
7.2.5	Picture Animation Display.....	44
7.2.6	Icon Rotation Display	45
7.2.7	Variable Bits Display	46
7.3	Text Display.....	47
7.3.1	Numeric Variable Display	47
7.3.2	Text Display.....	48
7.3.3	Real Time Clock (RTC) Display	49

7.3.4	HEX Data Display	50
7.3.5	Text Scrolling Display	50
7.4	Graph Display	51
7.4.1	Curve Display(0x20).....	51
7.4.2	Basic Graphical Display	52
7.4.3	Table Display.....	55
7.4.4	2D QR Code Display	57
8	OGUS Software User Guide	58
8.1	Establish a development environment.....	58
8.1.1	Install OGUS software	58
8.1.2	Create new project.....	59
8.2	Prepare development material	60
8.3	Work flow of project development.....	60

Symbols and Special Nouns

Symbols / Nouns	Note
"0x"	Numbers with "0x" prefixes to represent hexadecimal numbers
"H"	Numbers with "H" suffixes to represent hexadecimal numbers
Variable	Display/touch function object
Variable Address	The first address of a space in RAM that stores variable.
Description Pointer	The first address of a space in RAM that stores a description of the variable properties.
High/Low Byte	The command and data used in UART communication are all hexadecimal numbers and high byte send first (MSB). For example, 0x1234 will be sent as 0x12, 0x34. The 0x12 is high byte and 0x34 is low byte.
System Configure Register	The registers used to configure Embedded LCD which can be set by file config.txt. The register is represented by capital letter R and hexadecimal number, for example: R2, RC etc.
Variable Storage	RAM space used to store variable address and data which Description Pointer point to. The data is not reserved when power off.
Font Library Space	The space stores configure file, font library, icon file.
Picture Space	The space stores interface picture.
Database Space	User database
Register Space	The register space can be read/write through UART interface and the data is saved in hexadecimal.

1 Quick Demo Guides

1.1 Components of Demo Set

The components of an Embedded LCD demo set include: Embedded LCD, AC/DC adapter, USB/UART adapter board, mini-USB cable and Flexible Flat Cable (FFC). The picture is as below:



Figure 1.1 Embedded LCD Demo set

1.2 Connection to Power Supply

The Embedded LCD can connect to power supply directly and demonstrates program stored in it. The picture below shows the connection.



Figure 1.2 Embedded LCD connect power supply

1.3 Basic Feature Demo

Embedded LCD provide a basic feature demo which integrated multiple functions, such as, Artistic Words, Dash Board, Progress Bar, Increment Adjustment, Animation, QR Code, etc.



Figure 1.3 Function Demo

1.4 Connection to PC

Embedded LCD can connect PC by Mini-USB cable. User can control Embedded LCD by sending command though UART interface.



Figure 1.4 Embedded LCD connect PC

1.5 Configure UART Interface in PC

With the connection mentioned in section 1.4, user can send command to Embedded LCD by UART interface to control the display operations and configure Embedded LCD's status. In this connection, the connector on USB/UART adapter board (in red circle in figure1.5) should be disconnection.



Figure 1.5 USB/UART adapter board

The configuration of UART is shown s below:

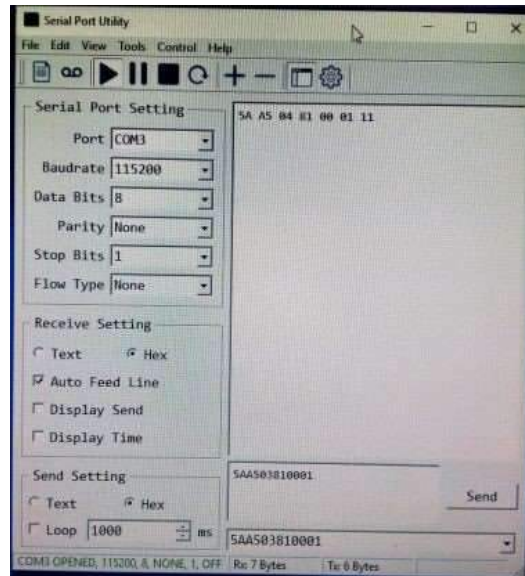


Figure 1.6 UART configuration

As shown in Figure1.6, user can send command “5A A5 03 81 00 01” to get the software version of Embedded LCD. If the correct information (such as 5A A5 04 81 00 01 11) can be received, the communication between the PC and the LCD is ready.

2 System Development

2.1 Introduction to the Development Environment

The Orient Display Embedded LCD Development Environment include hardware and software two parts. The hardware part refers to Section1: Quick Demo Guides. The software part is Orient display Graphic Utilized Software (OGUS). The structure of the development environment as the figure2.1 below.

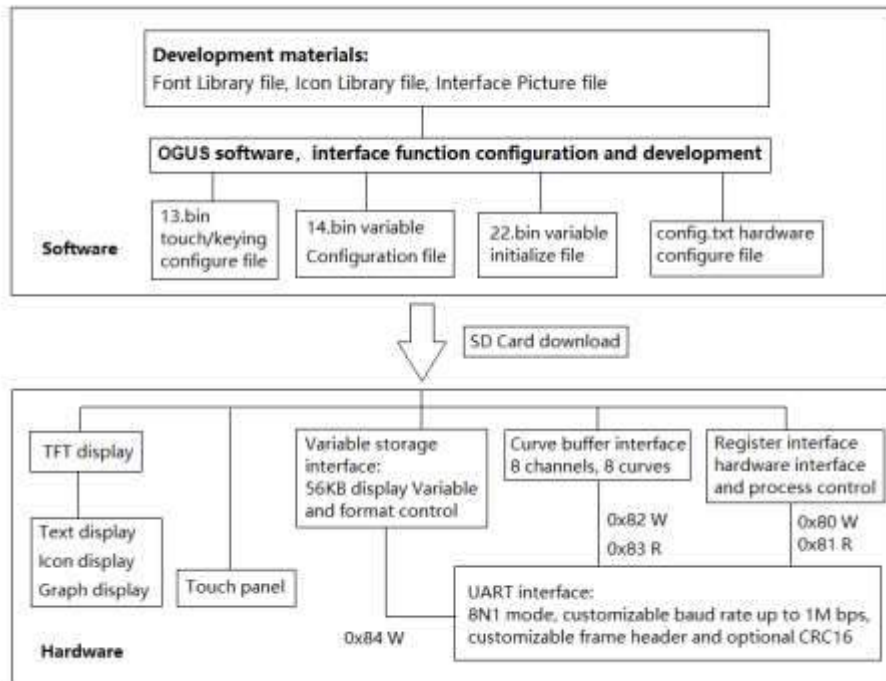


Figure 2.1 The structure of the development environment

The development materials (such as font library files, icon library files and picture files) are combined and generate configure files (such as 13*.bin, 14*.bin, 22*.bin and config.txt) by OGUS software. The materials are saved to SD card and download into Embedded LCD hardware. The detail information please refer to Section9: OGUS software user guide.

2.2 The development advantages of Embedded LCD

Unlike the traditional LCM display through timing or instruction control, the LCD adopts the direct variable drive display mode, all the operation is based on the pre-set variable matching file, greatly reduces the user's development difficulty.

The advantages include:

1. Decompose each page into multiple controls and users can use a feature simply by adding the appropriate control to the configuration file.
2. A variety of selectable controls, including data display, touch input, sound playback, etc.
3. 56K RAM space, 8 channels curve data storage, fast variables display response speed (up to 80ms maximum).
4. 256 bytes configuration registers for hardware control which can be read/written through UART interface.
5. Each page can have up to 128 display controls (overlay that support display controls) and any number of touch controls

6. Use an SD card to configure hardware parameters, download pictures, and upgrade software which provides convenience of production Archives management in mass production.
7. Integrate multiple functions, such as: RTC, backlight control, etc.
8. Support capacitive touch panel
9. Support custom data base which can be stored in picture space
10. The integrated OGOS platform, which allows users to run part of the code on Embedded LCD, makes the user's development simple and allows users to implement more complex functions with a wealth of instructions. This also makes it possible for Embedded LCD to be used as the host device of the system.
11. The OGOS platform integrates instructions such as mathematical operations (including MAC, CRC), data storage (including Flash database Read and write), serial communication, Common communication protocol processing (such as Modbus protocol, DT/T465 Power Meter reading protocol, etc.), serial peripheral (such as printer) drive, OGUS process Control, etc. Typical application cases include Modbus bus management, Power meter reading, Bill printing, POS equipment and so on.

2.3 Comparison of Development Examples

Suppose we need to make a touchscreen thermostat that requires real-time temperature to be displayed on the current page and can be switched to the settings page for parameter settings by clicking on the touchscreen. The following figure2.2 shows the traditional LCM development process and the development process of OGUS. By comparison, it can be seen that the development process of OGUS greatly reduces the difficulty of user development.

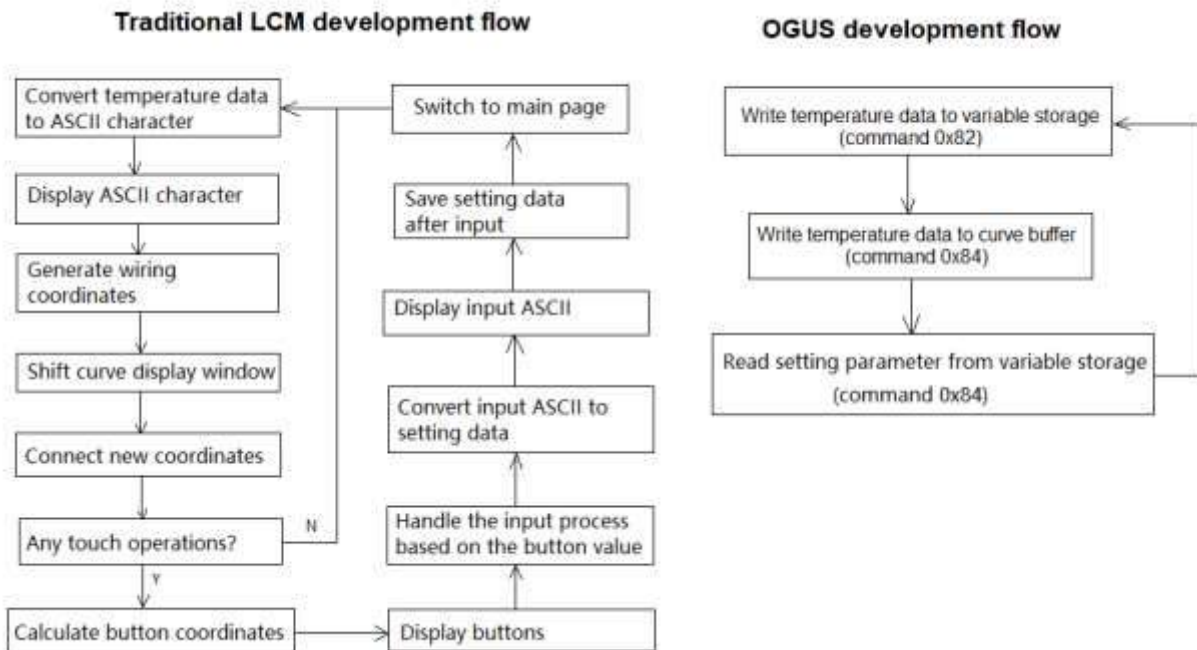


Figure2.2 Comparison of development process

3 Configuration of Embedded LCD

3.1 Format Requirements for files

As introduction before, multiple types of file needed in system development, such as: picture file, icon file and font library file etc. Embedded LCD systems identify different files through file IDs, which should be written as numbers at the beginning of the file name. Such as 0_poweron.bmp, 01.bmp, 30switch.ico, 48GBK24.HZK, 0_hello.wav are all valid filenames. The part of filename that follows the file ID is optional and each file ID can be used only once in one file type.

3.1.1 Picture File

Picture file can be used as user interface which stored in picture space. To get the better effect, the picture should have the same resolution with LCD screen and format of 24-bits color BMP file. The picture file should also include the icons of buttons and menus that will be used to defined functions.

3.1.2 Icon Library File

The icon file starting with the file ID should have the size no more than 255*255 pixels. It is important to note that usually an icon file has the size larger than 256KB and occupied more than one space (FLASH space participation refer to section 3.3). Therefore, the IDs of icon files should not be a successively number.

For example:

If the icon file occupied two spaces for each, the file IDs should be increased by 2, such as 30note.ico and 32single.ico.

If the icon file occupied three spaces for each, the file IDs should be increased by 3, such as 31error.ico and 34.ico.

3.1.3 Font Library File

Embedded LCD support multiple font encoding: 8bits, ASCII, GBK, GB2312, Unicode. ASCII library (ID=0) is pre-installed which include all ASCII characters with pixel matrix of 4*8 to 64*128. Embedded LCD supports font library file types of Bin, DZK, HZK.

3.1.4 Data Format

Data Type	Range
Integer	-32768[0x8000] ~ +32767[0x7FFF]
Unsigned Integer	0[0x0000] ~ 65535[0xFFFF]
Long Integer	-2147483648[0x80000000] ~ +2147483647[0x7FFFFFFF]
Extra Long Integer	-9223372036854775808 ~ +9223372036854775807

Decimal is stored in integer format and user-defined decimal digits.

For example: When the number of decimal digits is defined as 2, 0x4D2 (1234) means 12.34

3.1.5 Color Data Format

Embedded LCD support 16-bits (2 bytes, 65k colors, RGB 5-6-5-bit) color data, as beolow:

Bit	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
Color	R4	R3	R2	R1	R0	G5	G4	G3	G2	G1	G0	B4	B3	B2	B1	B0
	RED					Green						Blue				

3.2 Configuration Files

The configuration files include:

- 13*.bin -- touch/keying configuration file
- 14*.bin – variable configuration file
- 22*.bin – variable initialization file
- 23*.bin – OS code
- Config.txt – screen parameter configuration file

3.3 File / Data Storage

Embedded LCD supports FLASH (1GB/2GB) storage, RAM (56KB) storage, configuration registers(256Byte), curve buffer(16KB).

The storage location of files is as below:

File	File Type	Location
Picture file	Bmp	Picture space in FLASH storage
Font library file	Bin, dsk, hzk	Font library space in FLASH storage
Icon file	Ico	Font library space in FLASH storage
Configuration file	Bin, txt	Font library space in FLASH storage

3.3.1 FLASH Storage

FLASH is used to store parameter configuration file(config.txt), picture file, font library file, icon library file, touch/keying configuration file, variable configuration file, OS code and user data etc. Data in FLASH is not loss when power off. 32MB flash is reserved as font library space to store system pre-stored file, part of configuration files, icon library files and customized font library.

FLASH storage, depending on the content of the store, can be divided into picture space and database space. In picture space, 32MB space is reserved as font library space which is divided into 128 parts with 256KB for each.

Font library space partition

Location (space ID)	File	Size (KB)	Note
0-11	ASCII font library	256*12	System files of 6MB
12	Thesaurus for Input method	256	
13	Touch/keying configuration file	256	
14-21	Variable configuration file	256*8	
22	Initialization file	256	
23	OS code	256	
24-127	Customized font library file	256*n	
	Icon library file		
64-127	Customized font library file	256*n	Can be called by command
	Icon library file		

Font library space is divided into 128 parts with 256KB for each. The file is stored in the appropriate space according to the file ID. As the table above, Space 0-11 store ASCII font library (ID=0) and Space 13 store touch/keying configuration file (13*.bin). The files in Space 0-23 (system files, total 6MB) is fixed and can not be used to store other files. Therefore, the file ID number for customized font library and icon library is 24 to 127, inclusive. And only Space 64-127 can be called by command.

The file stored in font library space should start with file ID number which corresponding to the file location. For example, the file name of a customized font library file to be stored to Space 35 in the font library space can be like "35*.HKZ" (where * are optional characters). the file names such as "35hanzi.HKZ", "35.HKZ", "35 test.HKZ" are usable. The file name "ziku35.HKZ" is unusable.

Files in font library space

Location	File type	Name format	Sample	Note
0-11	ASCII font library	0*.HZK	0_ASClib.HZK	Pre-installed
12	Thesaurus for Input method	12*.BIN	12_PYLib.BIN	
13	Touch/keying configuration	13*.BIN	13touch.BIN	Load from SD card
14-21	Variable configuration	14*.BIN	14Var.BIN	

22	Variable initialization	22*.BIN	22Ini.BIN	
23	OS code	23*.BIN	23WCL.BIN	System generate
24-127	Font library	ID*.bin/HKZ/DZK	35hanzi.DZK	Load from SD card
	Icon library	ID*.ICO	51tubiao.ICO	

The remaining space outside the font library space in the picture space is used to store interface pictures and customer database. For different FLASH space, the maximum available picture number as below:

Picture storage (MB)	Database storage (MB)	Available picture number with different flash						
		320x480	480x272	640x480	800x480	800x600	1024x600	1024x768
VersionA (128)	51.75	370	370	148	123	92	74	61
VersionB (256)	179.5	868	868	289	289	217	173	144

3.3.2 RAM Storage

RAM storage has a size of 56KB and is divided into sub-spaces with 2 bytes physical space and addresses from 0x0000 to 0x6FFF. Generally, it is recommended to set description pointer range to 0x4000 to 0x6F00 and to set variable address range to 0x0000 to 0x4000 to avoid conflict. As the variable address and variable description pointer always point to the start address of a record and the size of each record is different, so it needs to be calculated carefully to avoid address overlap.

Note: the RAM from 0x6F00 to 0x6FFF is reserved for hardware parameter and can not be used by user.

3.3.2.1 Variable Address

Variable address is the start address of one or more sub-spaces in RAM, these sub-spaces are used to store variable encoding or status value that to be displayed.

For example:

A text "123456" is saved in RAM as below and each character occupy two bytes:

Variable address	Data	Character
...
0x1000	0x0031	"1"
0x1001	0x0032	"2"
0x1002	0x0033	"3"
0x1003	0x0034	"4"
0x1004	0x0035	"5"
0x1005	0x0036	"6"
...

To change character of the text, it just needs to change the encoding that saved in corresponding address. User can change the encoding value by command:

Command sent to Embedded LCD: 5A A5 05 82 1001 0031

Parameter "1001": the address of the character "2"

Parameter "0031": the encoding value of "1" which used to replace character "2" to be displayed

Then, the text is changed to "113456". By this way, user can change display data, start/stop animation, toggle icon, etc.

3.3.2.2 Variable Description Pointer

Variable description pointer is a start address of a sub-space of RAM which stores the property of a variable such as: display coordinate, color, font, etc. As variable description pointer shares the RAM space with variable address, it needs to calculate carefully to avoid address overlap when assigning address.

For example, a variable description is saved in RAM as below:

RAM address	Data	Note
0x4FFF_L	
0x5000_H	10	Variable address: 0x1000
0x5000_L	00	
0x5001_H	00	Start position to display, coordinates of the upper-left corner (x,y)
0x5001_L	6E	
0x5002_H	00	
0x5002_L	82	
0x5003_H	84	Color (0x8400) of text
0x5003_L	00	
0x5004_H	

Usually, user sends command to change the property of display variable by changing the value in variable description.

For example: command "5A A5 05 82 5003 F800" is used to change display color to red

Parameter "5003": the address of color value

Parameter "F800": value of the new color (red) to be changed to

Examples of other applications

Function	Command	Result
Change display position	5A A5 07 82 5001 0000 0000	Change display position, upper-left corner, to (0,0)

Change ASCII pixel matrix size	5A A5 05 82 500A 30 60	Change pixel matrix size to 48*96. Note: both x and y direction are all need to change
Hide character	5A A5 05 82 5008 0000	Set character length to "0" to hide character
Change font library file	5A A5 07 82 5009 003C 10 10	Load font library (ID=60) with matrix size 16*16. The font size is also changed if font library file changed

3.3.3 Configuration Register

Configuration Register has a size of 256 Byte. Different with storage, the registers are used to store status value. For example, the time of real time clock (RTC) and the real time brightness of backlight. The host can change values of the registers by command though UART interface and implement information interaction and control.

3.3.4 Curve Buffer

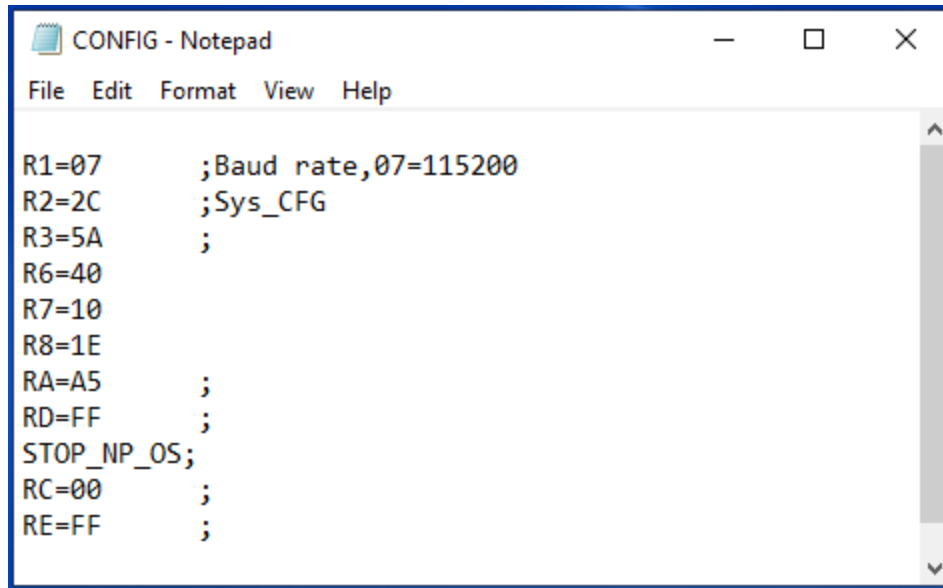
Embedded LCD supports 16KB curve buffer in which 8 curves can be stored. Sending data to curve buffer according specified format by command can implement rapidly display of curves. The data in curve buffer is in 16-bit unsigned integer.

3.4 Configuration File

3.4.1 System Configuration File (CONFIG.txt)

File CONFIG.txt which contains register parameters and downloaded from SD card is used to implement system parameter configuration of Embedded LCD. CONFIG.txt is a text file in which registers are described in a manner similar to the scripting language: each register description occupied one line and parameter is not written if it need not to be modified.

The register description in CONFIG.txt should follow the format: R? = HH, in which, "?" is register ID, "HH" is a hexadecimal number and letters should be capital. A typical sample as below:



```
CONFIG - Notepad
File Edit Format View Help

R1=07      ;Baud rate,07=115200
R2=2C      ;Sys_CFG
R3=5A      ;
R6=40
R7=10
R8=1E
RA=A5      ;
RD=FF      ;
STOP_NP_OS;
RC=00      ;
RE=FF      ;
```

Register configuration description:

For R2=2C, bit definition (0x2C=00 10 11 00) from left to right as below:

- 0: normal display
- 0: normal display
- 1: backlight controlled by Embedded LCD status
- 1: enable CRC16 in UART communication
- 0: disable automatic upload parameter of touch panel input
- 1: initialized variable by file 22
- 00: period is 200ms

For R6=40, the brightness of the screen after clicking on the touchscreen is 64 (0x40).

For R7=10, after a period of time without clicking on the touchscreen, the screen brightness turns to 16(0x10).

For R8=1E, after 30s(0x1E) without clicking on the touchscreen, the screen brightness turns to the value set in R7.

For RC=00, bit definition from left to right as below:

- 0: system reserved, set "0"
- 0: do not run OS program
- 0: reserved
- 0: the number of display variables is 64 per page
- 0: disable automatic acknowledge of result of frame CRC verification
- 0: using 3-point calibration mode
- 0: gesture recognize function off
- 0: undefined, set "0"

3.4.1.1 Set the physical resolution of the screen(R0)

The physical resolution of the screen is set by register R0. It is set up at the factory and does not need to be configured by the user.

3.4.1.2 Set Display Clock Phase

With different TCON, there two types of relationship between display data and display clock phase which set by register R4 which is set up at the factory and does not need to be configured by the user.

R4 value	Relationship to display clock phase
00	Display data is locked by the rising edge of display clock
FF	Display data is locked by the falling edge of display clock

3.4.1.3 Set UART baud rate (R1, R5, R9)

The UART baud rate is set by registers R1, R5, R9 and the default setting is R1=07, baud rate=115200bps. If value of R1 is in range of 0x00-0x12, R5 and R9 are invalid and the corresponding baud rate refer to the table below:

R1 value	0x00	0x01	0x02	0x03	0x04	0x05	0x06	0x07
Baud(bps)	1200	2400	4800	9600	19200	38400	57600	115200
R1 value	0x08	0x09	0x0A	0x0B	0x0C	0x0D	0x0E	0x0F
Baud(bps)	28800	76800	62500	125000	250000	230400	345600	460800
R1 value	0x10	0x11	0x12	0x12 - 0xFD	0xFE	0xFF		
Baud(bps)	625000	691200	921600	Reserved	Custom	Reserved		

When R1=0xFE, the baud rate is calculated as R5:R9 = 6250000/target baud rate. R5:R9 construct a two bytes parameter, R5 is high byte and R9 is low byte.

For example, when baud rate is to be set to 10Kbps, R5:R9 = 6250000/10000=625=0x0271, so the setting should be R5=02, R9=71

3.4.1.4 Set Frame Header (R3, RA)

The frame header has two functions:

- 1 To identified and synchronous data frame.
- 2 Used as a device address when multiple Embedded LCDs work in parallel

The default frame header is 0x5AA5 (R3=5A, RA=A5) and it can be changed by setting registers R3 and RA.

For example, if set R3=AA, RA=BB, only command with frame header "AABB" can be identified and received by Embedded LCD.

3.4.1.5 Set Software Operation Mode (R2, RC)

R2 and RC is defined in bits, used to configure software operation mode.

Description of Register R2 (default value is 0x00):

BIT	Weight	Definition	Note				
.7	0x80	VDS	0: normal display 1: rotate 90°				
.6	0x40	Reserved	Recommend to set "0"				
.5	0x20	TP_Led	0: disable backlight energy saving mode 1: enable backlight energy saving mode, parameters are set in R6/R7/R8 in file config.txt				
.4	0x10	FCRC	0: disable CRC-16 verification in UART communication 1: enable CRC-16 verification in UART communication				
.3	0x08	TPSAUTO	0: Disable automatic upload touch/keying input parameters, requiring user access 1: Enable automatic upload touch/keying input parameters				
.2	0x04	L22_InitEn	0: Set 56K RAM all to be "0" in initialization 1: Set 56K RAM according font library file 22 in initialization				
.1	0x02	FRS1	Scanning period, the smaller the scanning period, the more device sensitive, but the lower the processing capacity. The scan period affects the animation speed displayed by the animation icon. Note: for resolution 1024*768, the recommend scanning period is greater than 120ms.				
.0	0x01	FRS0	Scanning period	80ms	120ms	160ms	200ms
			FRS1	1	1	0	0
			FRS0	1	0	1	0

For example: to implement automatic upload of input parameters (set R2.3, weight 0x08), scanning period of 120ms (set R2.1, weight 0x02 and clear R2.0, weight 0) and all other bits keep default value "0". Then register R2 = 0x08+0x02 = 0x0A (=0000 1010)

Description of Register RC (default value is 0x00):

BIT	Weight	Definition	Note
.7	0x80	Reserved	Recommend to set "0"
.6	0x40	RunOSEnable	0: Disable running OS 1: Enable running OS
.5	0x20	Reserved	Reserved
.4	0x10	Page128Enable	0: 64 variables displayed per page 1: 128 variables displayed per page
.3	0x08	CRCAckEnable	0: Disable automatic acknowledge of CRC results 1: Enable automatic acknowledge of CRC results
.2	0x04	Reserved	Recommend to set "0"
.1	0x02	GestureEnable	0: disable gesture recognition 1: enable gesture recognition
.0	0x01	Reserved	Recommend to set "0"

For example, to implement 128 variables displayed per page (set RC.4, weight 0x10), enable automatic acknowledge of CRC results (set RC.3, weight 0x08), enable gesture recognition

function (set RC.1, weight 0x02) and all other bits keep default value “0”. Then RC = 0x10 + 0x08 + 0x02 = 0x1A (0001 1010)

3.4.1.6 Set Backlight in standby and wakeup (R2.5, R6, R7, R8)

When R2.5 set to “1”, backlight energy saving mode enabled. Embedded LCD will standby after a period of time during which there is no touch operation and wakeup by touch operation. This function can be achieved by registers R6, R7, R8 working together. R6 is brightness of backlight when Embedded LCD wakeup (brightness range is 0 to 64, that is 0x00 to 0x40). R7 is brightness of backlight when Embedded LCD standby (brightness range is 0 to 64, that is 0x00 to 0x40). R8 is the waiting time that no operation occurred. Note: the first touch operation does not trigger any action when backlight standby.

Register	Range	Note
R6	0x00 – 0x3F	After backlight control function enabled, the brightness of the backlight after wakeup
R7	0x00 – 0x3F	After backlight control function enabled, the brightness of the backlight that standby after a period of time during which there is not any touch operation.
R8	0x01 – 0xFF	After backlight control function enabled, the time waiting to standby, unit: seconds

3.4.2 Variable Initialization File (22*.bin)

22*.bin (* is optional name string) file is used to initialize the variables. 22*.bin is a binary file and data per two bytes corresponds to a variable address of the Embedded LCD. In default setting, system disable this function. User can enable it by adding a line of code as “R2=04” in to CONFIG.txt. 22*.bin and CONFIG.txt can be downloaded by SD card.

3.5 Device Configuration

3.5.1 Touch Panel Calibration

There are 3 methods to enable the touch panel calibration.

Method 1:

Step1: In 4 seconds, continuous click on the non-touch area (the area in which no function defined) more than 20 times

Step2: Get into calibration mode and click the specified location of the touch panel which indicated by the cross lines

Step3: Calibration finished and return back automatically.

Method 2:

Add “TP_CORRECT” into file CONFIG.TXT, Embedded LCD will start one calibration process.

Method 3:

In normal mode, writing 0x5A into register (0xEA) through UART will start one calibration process. Please be noted that multiple invalid calibrations may cause physical damage to the touch panel, such as line breakage, touchpad damage ...

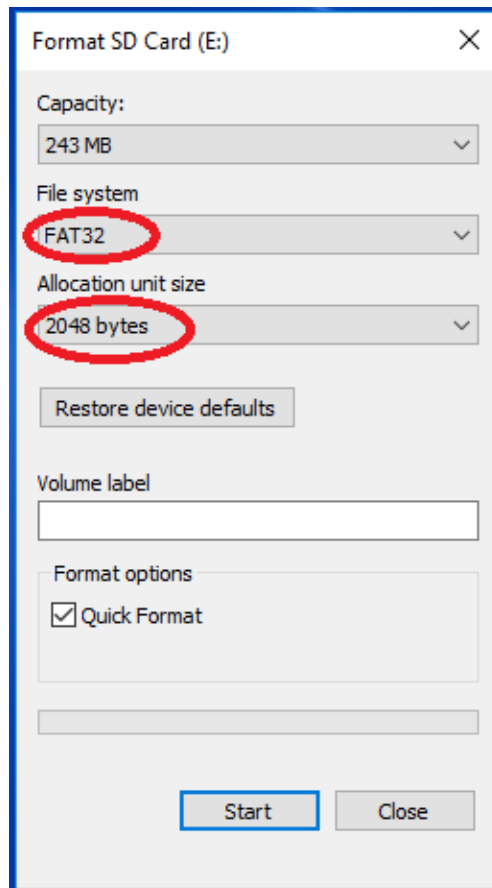
3.5.2 Download File by SD card

3.5.2.1 Format SD card

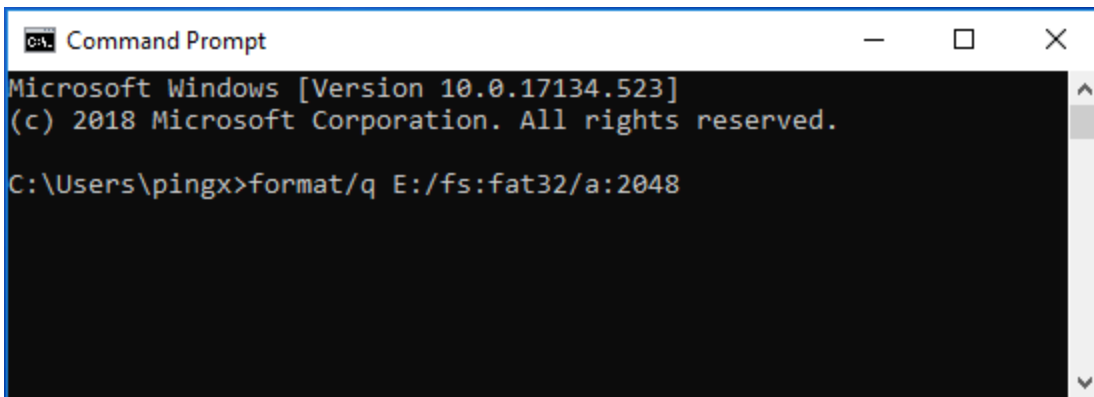
For compatibility reasons, it is recommended to use a card with a capacity of less than or equal to 4GB. All hardware parameter settings, program downloads, and software upgrades on the Embedded LCD are done through the SD/SDHC interface.

Before using the SD card for the first time, it is recommended to format the SD card and set its file system to: FAT32 format, the allocation unit size (that is, cluster size) must be selected 2048 or less. Format operation can be done either in windows system or in DOS system.

➤ Format in windows system



➤ Format in DOS



```
Command Prompt
Microsoft Windows [Version 10.0.17134.523]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\pingx>format/q E:/fs:fat32/a:2048
```

In DOS system, command “**format/q E:/fs:fat32/a:2048**” can be used to format the SD card. Where **E**: is the drive letter for the SD card, which can be replaced according to the actual situation.

Note: If your card does not support formatting of 2048 allocation unit, replace the card with a smaller capacity (this format is supported for cards below 4GB) or apply to the factory for a card.

3.5.2.2 Files downloading process

- 1 Create a folder named as NP_GUI under the SD card root directory
- 2 Copy pictures, font libraries, and configuration files that need to be downloaded into NP_GUI folder
- 3 Power off the Embedded LCD and insert the MicroSD (TF) card;
- 4 Power on the Embedded LCD and it will automatically load the contents of the NP_GUI folder, and save to the Embedded LCD device;
- 5 The user can power off the Embedded LCD and unplug the MicroSD (TF) card and re-power on the Embedded LCD to go back to normal mode.

Note : The minimum interval between two hot-swappable is 6 seconds, otherwise the embedded LCD will consider it as the same SD card and does not download data.

4 UART Communication

The UART interface is used for data transfer and configuration.

Asynchronous/Full duplex UART works in mode 8n1. Each byte consists of 10 bits,

- 1 start bit
- 8 data bits, always LSB (Least Significant Bit) is sent first
- 1 stop bit
- No parity bit

The communicate data rate can be configured in MicroSD/TF memory and the default value is 115200bps. All data transferred is 8-bit bytes (HEX) and the high byte is sent first.

For example, Data 0xF800 is sent as two 8-bit bytes 0xF8 (first), 0x00(second)

4.1 UART Testing

After making sure the UART connection is correct, user can check the UART communication by send command “5A A5 03 81 00 01”. This command is used to get the software version number of Embedded LCD and the return frame should be like “5A A5 04 81 00 01 **”, where “**” is the software version number.

4.2 Command

Commands are used to control display operations and work status of Embedded LCD by changing the values of variables. The description of command set as below:

Command	Code	Data
Write Register	0x80	TX: Address (0x00 ~ 0xFF) + data.
Read Register	0x81	TX: Address (0x00 ~ 0xFF) + data length (0x00 ~ 0xFF)
		RX: Address (0x00 ~ 0xFF) + data length (0x00 ~ 0xFF) + data
Write Variable Table	0x82	TX: Address (0x0000 ~ 0x6FFF) + data
Read Variable Table	0x83	TX: Address (0x0000 ~ 0x6FFF) + data length (0x00 ~ 0xFF)
		RX: Address (0x0000 ~ 0x6FFF) + data length (0x00 ~ 0xFF) + data
Write Curve Buffer	0x84	<p>Mode + Data0 + + Datan Mode: Channel order definition Parameter Mode is 8-bit data which defines the channel order, each bit corresponds one channel, Mode.0 corresponds to channel 0 Mode.1 corresponds to channel 1 Mode.7 corresponds to channel 7</p> <p>When the bit is set to “1”, the data should be written into the corresponding channel. When the bit is set to “0”, no data will be written into the corresponding channel. Data is assembled from low to high in order of channel number. For example: When Mode is 0x45(01000101B), the data should be written into channel0, channel2 and channel6, so the data assembling sequence should be (channel0+ channel2+ channel6) + (channel0+ channel2+ channel6) +.....+ (channel0+ channel2+ channel6)</p> <p>Datan: 2-byte curve data</p>

Note: Embedded LCD provides

256 Bytes registers which addressing in Bytes.

A 28K-word (56K Bytes) Variable table which addressing in words.

8K-word buffer to store the data which are up to 8 curves. The data is saved as 16-bit unsigned integers.

Command frame is used in communication which consists of Header, Data length, Command code, Data and CRC (cyclic redundancy check) bytes. The frame structure is shown as:

Frame	Header	Data length	Command code	Data	CRC value
Length (Byte)	1	1	1	N	2
Note	Defined by R3/RA in configure file (CONFIG.TXT)	The length Include Command code, Data and CRC	0x80 ~ 0x84		Enabled/Disabled by R2.4 in configure file (CONFIG.TXT)

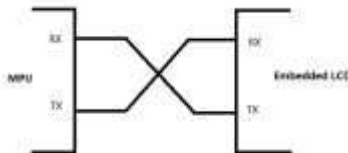
The maximum valid data length of a frame is 254 bytes (with CRC) or 252 bytes (without CRC). CRC check does not include header byte and data length byte, only for command byte and data by CRC-16 Algorithm (Polynomial representations is $X^{16}+X^{15}+X^2+1$, initial value is 0xFFFF). When CRC check is enabled (R2.4 and RC.3 are set), Embedded LCD will automatically return CRC results, the acknowledge frame is as below:

Status	Data Frame
CRC correct	Header+0x02+command code+ 0xFF + CRC value
CRC incorrect	Header+0x02+command code+ 0x00 + CRC value

4.3 Troubleshooting of UART Communication

Case1: hardware connection error

Make sure the hardware connections of UART RX/TX are as below:



Case2: TTL/RS232 selection error

The connector on USB/UART adapter board (in red circle) should be disconnection. Embedded LCD support RS232 interface.



Case3: Serial Port is occupied

Closed the software running in PC which occupying the COM port.

Case4: COM port parameter setting error

The default baud rate of Embedded LCD is 115200bps and the data format is 8N1. The baud rate can be configured by config.txt file.

Case5: Data frame header setting error

Make sure the header of data frame is correct (the default value is 5AA5) which can be configured by config.txt file.

5 Configuration Register

Configuration registers are used to store current status, such as RTC (real time), backlight brightness, etc.

5.1 Configuration Register Table

Address	Register	R/W	Length (Byte)	Note
0x00	Version	R	1	Version Number, BCD code
0x01	LedLm	R/W	1	LCD backlight brightness, 0x00 ~ 0x3F
0x02	Reserved	W	1	Reserved
0x03	PicIndex	R/W	2	Read: index ID of current display page Write: Pre-switch to the index number of the specified page
0x05	TPFlag	R/W	1	Touch panel flag. After reading, touch data will not be updated until this flag is cleared. 0x5A: update coordinates of touch other: not update coordinates of touch
0x06	TPStatus	R	1	0x01: the first tap 0x02: tap off 0x03: tap continue other: invalid
0x07	TPPosition	R	4	Coordinates of touch: Xh, Xl, Yh, Yl, Xh: high byte of X value in coordinate, Xl: low byte of X value in coordinate, Yh: high byte of Y value in coordinate, Yl: low byte of Y value in coordinate
0x0B	TPCEnable	R/W	1	0x00: Disable Touch function other: Enable Touch function, default value is 0xFF
0x0C	RunTime	R	4	Running time after power-on, BCD code, data format: HHHH: MM: SS Example: 9999: 59: 59
0x10	R0 ~ RC	R	13	Mapping of configuration registers of SD card, read only

0x1D	Config_Enable	W	1	Flag of R1~ RC register re-set	
				0x5A: R1 ~ RC reset and save	
				0xA5: R1 ~ RC reset and do not save	
0x1E	LedLm_Now	R	1	Current backlight brightness	
0x1F	RtcComAdj	W	1	0x5A: RTC setting flag, cleared after RTC is set	
0x20	RtcNow	R/W	7	YY: MM: DD: WW: HH: MM: SS	
0x27	Reserved	—	25	Reserved	
0x40	EnLibOP	R/W	1	0x5A: Enable Font Library operation, cleared after execution	
0x41	LibOPMode	W	1	0xA0: copy the specified data in font library space to variable space	
0x42	LibID	W	1	Index of font library, value range: 0x40 ~ 0x7F	
0x43	LiaAddress	W	3	The first address of the data block in font library space, value range: 0x000000 ~ 0x01FFFF	
0x46	VP	W	2	The first address of the data block in variable space, value range: 0x0000 ~ 0x6FFF	
0x48	OPLength	W	2	The length of data to be operated, value range: 0x0001 ~ 0x6FFF	
0x4A	Timer0	R/W	2	16-bit software timer	Software Timer0~3, Unit: 4ms, automatically reduced to zero and stop
0x4C	Timer1	R/W	1	8-bit software timer	
0x4D	Timer2	R/W	1	8-bit software timer	
0x4E	Timer3	R/W	1	8-bit software timer	
0x4F	KeyCode	W	1	Touch key code for triggering file 13*.bin Value range: 0x01 ~ 0xFF, 0x00 means invalid. This register will be cleared after execution.	
0x50	Reserved	W	3	Reserved	
0x53	VolumeAdj	W	2	Volume adjustment. Format: 0x5A VOL	
				0x5A: Enable Volume adjustment	
				VOL: Volume = VOL/64, default value of VOL is 64.	
0x55	Reserved	—	1	undefined	
0x56	EnDBLOP	R/W	1	0x5A: Enable font library operations, cleared after execution	
0x57	OPMode	W	1	0x50: copy data from variable space to database space	
				0xA0: copy data from database space to variable space	
0x58	DBLAddress	W	4	Database address (in word) is 0x00000000 ~ 0x01C1FFFF, maximum database space is 450MW (900MB, depends on Flash in Kernel). The database starts from the physical address of 64MB and coincides with the picture space, with each data occupying 2 bytes of physical storage.	
0x5C	VP	W	2	the first address (in word) of the specified data block in variable space, range: 0x0000 ~ 0x6FFF	
0x5E	OPLength	W	2	Data length, range 0x0001 ~ 0x6FFF	
0x60	Reserved	—	138	Reserved	
0xEA	TPCalTrigger	W	1	Write 0x5A to enable touch screen calibration and be cleared after calibration.	
0xEB	TrendlineClear	W	1	Writes a specific value to clear the corresponding curve buffer data.	

				0x55: Clear buffer data for all 8 curves
				0x56 ~ 0x5D: Clear curve buffer data for CH0-CH7 channels, respectively
				When the curve buffer data is cleared, the register is cleared
0xEC	Reserved	—	2	Reserved
0xEE	RstSystem	W	2	Write 0x5AA5 to reset the system once
0xF0	Reserved	—	16	Reserved

Example 1: Query and adjustment of backlight brightness value

Query current backlight brightness:

Command frame	5A A5	03	81	01	01
Note	Frame header	Frame Data length	Read register command code	Register address	Data length to return

Responding frame:

Answer frame	5A A5	03	81	01	01	40
Note	Frame header	Frame Data length	Command code	Register address	Data length	Brightness value of backlight

The returning brightness value of backlight is 0x40 (brightness value is in range of 0x00-0x40).

The command below can change the brightness value of backlight to 0x10,

Command frame	5A A5	03	80	01	10
Note	Frame header	Frame Data length	Write register command code	Register address	Data

User can use query command again to verify if the brightness is changed correctly.

Note: when system configuration register R2.5=1 (enable standby and wakeup functions), the commands 5AA5 03 80 01 10 and 5AA5 03 80 01 3F should be sent together to change the backlight brightness.

Example 2: Get the current page ID and change page

Current page ID is saved in register 0x03, the command below can be used to read it.

Command frame	5A A5	03	81	03	02
Note	Frame header	Frame Data length	Read register command code	Register address	Data length to return

Responding frame:

Answer frame	5A A5	04	81	03	02	00 00
Note	Frame header	Frame Data length	Command code	Register address	Data length	Page ID

The ID of current page is 0x00. And the command below can change current page to page (ID = 0x10).

Command frame	5A A5	03	80	03	00 10
Note	Frame header	Frame Data length	Write register command code	Register address	Data

Other configurations,

Functions	Command	Note
Reset	5A A5 04 80 EE 5A A5	Equivalent to power off then power on
Touch panel calibration	5A A5 03 80 EA 5A	
Disable touch function	5A A5 03 80 0B 00	
Enable touch function	5A A5 03 80 0B 01	

5.2 Examples of register applications

5.2.1 Read/Write RTC

5.2.1.1 Read RTC

The RTC information is stored in 7 registers which address from 0x20 to 0x26.

Command to read full data (YY: MM: DD: WW: HH: MM: SS): 5AA5 03 81 20 07

Command to read time only (HH: MM: SS): 5AA5 03 81 24 03

5.2.1.2 Write RTC

Step1: Enable RTC calibration, write 0x5A into register 0x1F: 5A A5 03 80 1F 5A

Step2: Write data into registers from 0x20 to 0x26. For example, if it is set to 17:05:23, January 25, 2016, the command frame should be 5A A5 09 80 20 16 01 25 00 17 05 23.

A combined command is 5A A5 0A 80 1F 5A 16 01 25 00 17 05 23.

Note: when write RTC, the system will calculate data of week automatically, so it is written as "0x00" in the command.

5.2.2 Read Front Library

Address	Register	R/W	Length (Byte)	Note
0x40	EnLibOP	R/W	1	0x5A: Enable Font Library, cleared after execution
0x41	LibOPMode	W	1	0xA0: copy the specified font library to variable table
0x42	LibID	W	1	Index of font library, value range: 0x40 ~ 0x7F
0x43	LiaAddress	W	3	The first address of the font library to be read, value range: 0x000000 ~ 0x01FFFF
0x46	VP	W	2	The first address of the variable space to be written, value range: 0x0000 ~ 0x6FFF
0x48	OPLength	W	2	The length of data to be copied, value range: 0x0001 ~ 0x6FFF

The font libraries (index ID from 64 to 127, total 64 font libraries, 16MB) can be copied to variable space by command. And user can read the font library from the variable space by command 0x83.

Example: Copy font library with index 64, 4KW(0x1000), start address 0x000000 to variable space with start address 0x1000. The command frame is "5AA5 0C 80 40 5A A0 40 00 00 00 10 00 10 00"

The data to be copied should not exceed the font library space, $(LiaAddress + OPLength) \leq 0x020000$

5.2.3 Read/Write Database

Configuration registers of database,

Address	Register	R/W	Length (Byte)	Note
0x56	EnDBLOP	R/W	1	0x5A: Enable font library operations, cleared after execution
0x57	OPMode	W	1	0x50: copy data from variable space to database space 0xA0: copy data from database space to variable space
0x58	DBLAddress	W	4	Database address is 0x00000000 ~ 0x01C1FFFF, maximum database space is 450MW (900MB, depends on Flash in Kernel). Each byte occupying 2 bytes of physical FLASH.
0x5C	VP	W	2	Specifies the first (word) address of the data in variable space, range: 0x0000 ~ 0x6FFF
0x5E	OPLength	W	2	Operation Data length, range 0x0001 ~ 0x6FFF

With different flash storage space, the picture space and the available database space has different sizes which listed in the table below:

Picture storage (MB)	Database storage (MB)	Available picture number with different flash						
		320x480	480x272	640x480	800x480	800x600	1024x600	1024x768
VersionA (128)	51.75	370	370	148	123	92	74	61
VersionB (256)	179.5	868	868	289	289	217	173	144

The user database is physically composed of several database pages of 64KW (128KB) in size, but the address in the read/write operation is continuous, the system automatically handles paging situations. Each page writes a lifetime of 100, 000 times, that is, each write operation reduces the number of times once.

As the database space shared the storage space with picture storage space, the address offset should be calculated to avoid data overlap. Image index number corresponding to the physical address (64MB) of the database space as below:

Resolution		320x480	480x272	640x480	800x480	800x600	1024x600	1024x768
Adjustment coefficient (K)	VersionA	2	2	5	6	8	10	12
	VersionB	1	1	3	3	4	5	6
The number of pictures	VersionA	126	126	50 ~ 51	42	31 ~ 32	25 ~ 26	21
	VersionB	124	124	41 ~ 42	41 ~ 42	31	24 ~ 25	20 ~ 21

Note: When accessing data by using database, users need to avoid picture pages.

The database offset address calculation formula is shown below (in words):

$A = (Q * K - M) * T * C$ when $(Q * K) > M$

$A = 0$ when $(Q * K) \leq M$

A: Represents a relative offset address (in word), that is, the length of the address that needs to be offset from the start address of the database

Q: The total number of pre-saved pictures

M: Picture quantity adjustment coefficient (constant value). Adjustment coefficient of Version A is 252, Adjustment coefficient of Version B is 124.

K: Adjustment coefficient

T: Correction coefficient, the correction coefficient of version A is 1, the correction coefficient of version B is 2.

C: Constant, $64 * 1024$

Example 1: 100 pictures with resolution 800x480 saved in Embedded LCD (Part No.

NP800480C070_GASR00), system version is A, $Q = 100$, $M = 252$, $K = 6$, $T = 1$, then address offset of the database operation is $(100 * 6 - 252) * 1 * 64 * 1024 = 22806528[0x015C0000]$

Example 2: 200 pictures with resolution 800x600 saved in Embedded LCD (Part No.

NP800480C070_GBSR00), system version is B, $Q = 200$, $M = 124$, $K = 4$, $T = 2$, then address offset of the database operation is $(200 * 4 - 124) * 2 * 64 * 1024 = 88604672[0x05480000]$

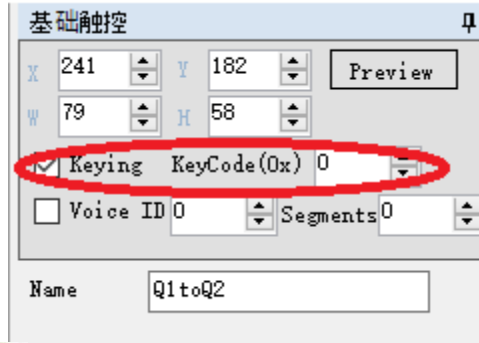
5.2.4 Key Code Control


Address	Register	R/W	Length (Byte)	Note
0x4F	KeyCode	W	1	Touch key code for triggering file 13*.bin. Value range: 0x01 ~ 0xFF, 0x00 is invalid value. This register will be cleared after execution.

Embedded LCD provide 0x4F (Key code control) Register instead of keyboard interface. User can perform a pre-defined touch process which described in (13*.bin) by writing corresponding key code into register (0x4F).

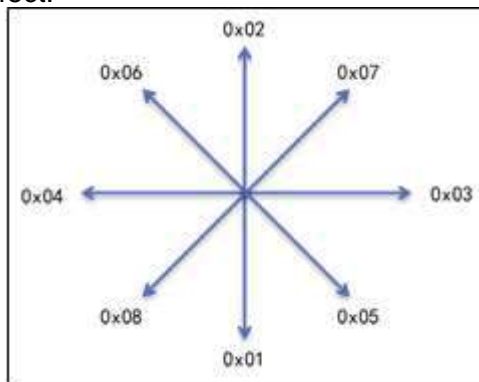
For example, in touch configure file, key code 0x01 on page 10th means turn to data input interface, when the command frame "5AA5 03 80 4F 01" is sent on page 10th, the Embedded LCD will turn to data input interface.

Key-code triggers and touch-screen triggers can be triggered in parallel, so they can be used together. Key-code trigger function can be defined in 13*.bin which is generated though OGUS.



In OGUS, set BasicTouch  variable's property as Keying and the KeyCode can be set from 0x00 to 0xFF which support 256 keying functions. If a BasicTouch variable is set as "keying", it can not be controlled by screen touch. Putting another BasicTouch variable which is not set as "keying" at the same place can implement the key-code/touch double control. Overlapping positions do not affect functions.

Further more, the key code 0x01 to 0x08 are reserved for gesture control. As the figure shown below, Embedded LCD will write the corresponding code into register 0x4F automatically when the gesture is recognition. If RC.1=1 (gesture recognition enabled), Embedded LCD will implement the pre-defined effect.



6 Touch/keying Configuration File

Touch/keying features are configured with touch profiles 13*.bin. The file consists of one or more instructions described in accordance with the touch/key control function, each of which is fixed in 16, 32, or 48 bytes of space; A touch instruction consists of 6 parts.

NO.	Name	Length (Byte)	Note
1	TPID	2	Index ID of function picture
2	TPArea	8	Valid area of touch: Top left corner coordinates (Xs, Ys), Lower right corner coordinates (Xe, Ye). When Xs = 0xFFFF, the function is triggered by writing key code into register 0x4F and tap effect should be disabled, then Ys_H is key code value and Ys_L / Xe / Ye are undefined.
3	TraID	2	Switch picture index ID ; 0xFFxx: no switching
4	AniID	2	Tap effect picture index ID; 0xFFxx: no effect to be displayed
5	TP_Code	2	Key code

			0xFFxx: invalid key code 0xFExx (0xFDxx): function key code. For example, 0xFE00: enable variable screen input function. The "00" is touch/keying code, refer to section 6.1 Touch/Keying Function List Any change of 0xFExx function key can be upload automatically by setting the R2.3. 0xFDxx function keys are not upload automatically. Other value: key code in ASCII. For example, 0x0031: key "1"
6	TPFun	32	Description of function key when TP_Code has value of 0xFExx

6.1 Touch / Keying Function List

No.	Code	Function	Note
1	0x00	Data Input	Input data (integer, decimal, etc.) to variable space.
2	0x01	Menu Action	Click to trigger a pop-up menu and return key code of the menu
3	0x02	Incremental Input	Click the button to make a self-added (++) or self-subtraction (- -) operation on the specified variable. The step length and upper/lower limits can be set. Setup loop adjustment with range 0~1 to achieve column check box function.
4	0x03	Drag Input	Data input by the way of dragging and the scale range is configurable.
5	0x04	Set RTC	Set Real Time Clock by soft keyboard, data of a full calendar day (year, month, date, hour, minute, second) is needed.
6	0x05	Return Key code	Click the button to return the key value directly to the variable, and support returning variable bit
7	0x06	Text Input	Input text in format of ASCII
8	07_00	Register to variable	Provides a way to overwrite the register space through the touchscreen, to control the hardware indirectly.
9	07_01	Variable to register	For example, the backlight register read to the variable, adjust the variable and then write back to adjust the backlight brightness.
10	07_02	Convert to monochrome (portrait)	Convert color image in specified area to monochrome bitmap (portrait) and store in variable space which pointed by VP.
11	07_05	Convert to monochrome (landscape)	Mainly used to print what is displayed on the current screen.
12	07_03	Send data to COM1	Click on the touchscreen to send data from the specified area pointed by VP to serial port (COM1).
13	07_04	Send data to COM2	Click on the touchscreen to send data from the specified area pointed by VP to the extended serial port (COM2).
14	07_06	Send coordinates to COM2	Click on the touchscreen to send click position coordinates to the extended serial port (COM2).
15	0x08	Return touch data	Click the touch panel and return pre-set data to variable space or UART.
16	0x09	Rotation input	Input data by turning the knob and scale range configurable.

6.2 Data Input from Touch Screen

The input data which includes different types such as number, letter, character, etc. is to be saved to corresponding location in variable space. Therefore, the input content must be mapped to a key code.

6.2.1 Number Input

Address	Definition	Length (byte)	Note
0x00	TPID	2	Index ID of function picture
0x02	TPArea	8	Valid area of touch: Top left corner coordinates (Xs, Ys), Lower right corner coordinates (Xe, Ye).
0x0A	TraID	2	Switch picture index ID ; 0xFFxx: no switching
0x0C	AniID	2	Tap effect picture index ID; 0xFFxx: no effect to be display
0x0E	Code	2	0xFE00/0xFD00
0x10	0xFE	1	0xFE
0x11	*VP	2	Variable address pointer which corresponds to the input data
0x13	VType	1	Return data type
			0x00: 2 bytes, Integer: -32768~+32767 unsigned integer: 0~65535
			0x01: 4 bytes, long integer -2147483648 ~ +2147483647 unsigned long integer 0 ~ 4294967295
			0x02: *VP high byte, unsigned integer, 0 ~ 255
			0x03: *VP low byte, unsigned integer, 0 ~ 255
			0x04: 8 bytes, super long integer: -9223372036854775808 ~ +9223372036854775807
0x14	NInt	1	Number of integer bits of input data. For data 1234.56, NInt is set 0x04
0x15	NDot	1	Number of decimal bits of input data. For data 1234.56, NDot is set 0x02
0x16	x, y	4	Display location of input: right alignment, (x, y) is the upper right coordinate of the last character of the string
0x1A	Color	2	Color of the input characters
0x1C	LibID	1	ASCII Font Library index ID of characters displayed
0x1D	FontHor	1	Font size, number of dot-matrix in X-axis direction
0x1E	CusorColor	1	Cursor Color, 0: Black; other: White
0x1F	HideEn	1	0x00: Hide Input, displayed as ""
			other: display input character
0x20	0xFE	1	0xFE
0x21	KBSource	1	0x00: Keyboard on the current page
			other: Keyboard is not in the current page
0x22	KBPicID	2	The index ID of the page on which the keyboard resides. Valid when KBsource is not 0x00
0x24	KBArea	8	Keyboard area: Upper-left coordinate (xs, ys), lower right coordinate (xe, ye). Valid when KBsource is not 0x00
0x2C	KBPosition	4	The keyboard position on the current page, the upper-left coordinates, and is valid when KBsource is not 0x00
0x30	0xFE	1	0xFE
0x31	LimitEn	1	0xFF: Enable input range limit, overflow input invalid (equivalent cancellation)
			Other: no input range limit
0x32	Vmin	4	Input lower limit, 4 bytes (Long integer or unsigned long integer)

0x36	Vmax	4	Input upper Limit, 4 bytes (Long integer or unsigned long integer)
0x3A	Reserved	6	Recommend to set "0"
The valid key codes: 0x0030(0)~0x0039(9), 0x002E(.), 0x00F0(cancel), 0x00F1 (enter), 0x00F2 (BackSpace)			

6.2.2 Text Input

The typical text key code definition is shown as below:

Key code	Lower case	Capital	Key code	Lower case	Capital	Key code	Lower case	Capital	Key code	Lower case	Capital
7E60	`	~	5171	q	Q	4161	a	A	5A7A	z	Z
2131	1	!	5777	w	W	5373	s	S	5878	x	X
4032	2	@	4565	e	E	4464	d	D	4363	c	C
2333	3	#	5272	r	R	4666	f	F	5676	v	V
2434	4	\$	5474	t	T	4767	g	G	4262	b	B
2535	5	%	5979	y	Y	4868	h	H	4E6E	n	N
5E36	6	^	5575	u	U	4A6A	j	J	4D6D	m	M
2637	7	&	4969	i	I	4B6B	k	K	3C2C	,	<
2A38	8	*	4F6F	o	O	4C6C	l	L	3E2E	.	>
2839	9	(5070	p	P	3A3B	;	:	3F2F	/	?
2930	0)	7B5B	[{	2227	'	"	2020	SP	SP
5F2D	-	_	7D5D]	}	0D0D	Enter	Enter			
2B3D	=	+	7C5C	\							

Note: For the key code in table, the low byte of two-byte key code represents a lowercase key code, and the high byte represents a capital key code. The key code of a text keyboard must be less than 0x80 (ASCII code). 0x0D Key code entry will be automatically converted to 0x0D 0x0A; Key codes 0x00 and 0xFF are disabled.

Special features keyboard key code definition

Key Code	Definition	Note
0x00F0	Cancel	Cancel input and return, do not change variables
0x00F1	Return	Confirm input and return, input text is saved to the specified variable position
0x00F2	Backspace	Delete 1 character forward
0x00F3	Delete	Delete 1 character backward
0x00F4	CapsLock	Uppercase lock. If enabled, the corresponding button must define the effect of button click
0x00F7	Left	The cursor moves a character forward and is also used for paging in GBK Chinese character entry
0x00F8	Right	The cursor moves a character backward and is also used for paging in GBK Chinese character entry

When using the keyboard (the key code saved in the register 0x4F) to do text entry, define the animation area of the button in the area where you need to prompt "CapsLock", so that when you send the CapsLock key, the corresponding position of the screen will automatically display the "CapsLock" area icon prompt.

6.2.2.1 ASCII Input

Address	Definition	Length (Byte)	Note	
0x00	TPID	2	Index ID of function picture	
0x02	TPArea	8	Valid area of touch: Top left corner coordinates (Xs, Ys), Lower right corner coordinates (Xe, Ye).	
0x0A	TraID	2	Switch picture index ID ; 0xFFxx: no switching	
0x0C	AniID	2	Tap effect picture index ID; 0xFFxx: no effect to be display	
0x0E	Code	2	0xFE00/0xFD00	
0x10	0xFE	1	0xFE	
0x11	*VP	2	Variable address pointer which corresponds to the input data.	
0x13	VPLenMax	1	The maximum length of the text variable, the range of the number of words: 0x01 ~ 0x7B. When the text is saved to the specified address, 0xFFFF is added at the end of the text automatically. The input text variable may actually occupy the maximum variable space as VPLenMax +1.	
0x14	ScanMode0	1	Input Mode control	
			0x00	Re-input
			0x01	Open the original text and modify it
0x15	LibID	1	Display the current ASCII font library location, 0x00 = default font library	
0x16	FontHor	1	Font size, number of matrix dot in X-direction	
0x17	FontVer	1	Font size, number of matrix dot in Y-direction (when LibID =0x00, number of matrix dot in Y-direction must be doubled)	
0x18	CusorColor	1	Cursor Color	
			0	Black
			Other	White
0x19	Color	2	Text Display color	
0x1B	ScanAreaStart	4	Input text display area left upper corner coordinates (Xs, Ys)	
0x1F	ScanReturnMode	1	0X55: Save the input end tag and the valid data length at the * (VP-1) Location	
			* (VP-1) High byte, 0x5A: end tag, 0x00: processing tag	
			* (VP-1) Low byte, valid input data lengths, one byte	
0x20	0xFE	1	0xFE	
0x21	ScanAreaEnd	4	Input text display area lower right corner coordinates (Xe, Ye)	
0x25	KBSource	1	0x00: Keyboard on the current page	
			Other: Keyboard is not on the current page	
0x26	KBPicID	2	The index ID of the page on which the keyboard resides. Valid when KBSource is not 0x00	
0x28	KBArea	8	Keyboard area: Upper-left coordinate (Xs, Ys), lower right coordinate (Xe, Ye). Valid when KBSource is not 0x00	
0x30	0xFE	1	0xFE	

0x31	KBPosition	4	The keyboard position on the current page, the upper-left coordinates, and is valid when KBSource is not 0x00
0x35	DisplayEn	1	0x00: The input displayed normally 0x01: Display "*" for password input
0x36	Reserved	9	Recommend to set "0"
0x3F	ScanMode1	1	0x00 (ASCII)

Note: Embedded LCD pre-installed font library (ID=0) which contains 4*8 ~ 64*128 dot matrix of all ASCII pattern.

6.2.2.2 GBK Input

Address	Definition	Length (Byte)	Note	
0x00	TPID	2	Index ID of function picture	
0x02	TPArea	8	Valid area of touch: Top left corner coordinates (Xs, Ys), Lower right corner coordinates (Xe, Ye).	
0x0A	TraID	2	Switch picture index ID ; 0xFFxx: no switching	
0x0C	AniID	2	Tap effect picture index ID; 0xFFxx: no effect to be displayed	
0x0E	Code	2	0xFE00/0xFD00	
0x10	0xFE	1	0xFE	
0x11	*VP	2	Variable address pointer which corresponds to the input data.	
0x13	VPLenMax	1	The maximum length of the text variable, the range of the number of words: 0x01 ~ 0x7B. When the text is saved to the specified address, 0xFFFF is added at the end of the text automatically. The input text variable may actually occupy the maximum variable space as VPLenMax +1.	
0x14	ScanMode0	1	Input Mode control 0x00: Re-input 0x01: Open an existing text and modify it	
0x15	LibGbk0	1	The GBK font library index ID, default ASCII Font ID is 0x00	
0x16	LibGbk1	1	The index ID of GBK font library used in input process	
0x17	FontDot0	1	LibGbk0: Font size, number of matrix dot	
0x18	FontDot1	1	LibGbk1: Font size, number of matrix dot	
0x19	CusorColor	1	Color of cursor. 0x00: Black Other: White	
0x1A	Color0	2	Color of text entered	
0x1C	Color1	2	Color of text in the input	
0x1E	PyDisMode	1	Pinyin hints and the display mode of Chinese characters	
			0x00	Pinyin tips on the top, Chinese characters show another line below. Pinyin tips and Chinese characters show left alignment, line spacing is ScanDis.
			0x01	Pinyin tips on the left, Chinese characters on the right; Chinese characters begin to show X position in Scan1AreaStart + 3*FontDot1 +ScanDis

0x1F	ScanReturnMode	1	0xAA: Save the input end tag and the valid data length at the * (VP-1) Location	*(VP-1) Low bytes, valid input data length in byte.		
			0xFF: Do not return input end tags and lengths	*(VP-1) High bytes, entry status flag	0x5A	Input end
				0x00	In input process	
0x20	0xFE	1	0xFE			
0x21	Scan0AreaStart	4	Input text display area upper-left corner coordinates (Xs, Ys)			
0x25	Scan0AreaEnd	4	Input text display area lower right corner coordinates (Xe, Ye)			
0x29	Scan1AreaStart	4	The upper-left coordinate of the text display area of the voice prompt during the input process			
0x2D	ScanDis	1	The spacing of Chinese characters is shown in the input process, up to 8 Chinese characters per line			
0x2E	Reserved	1	Recommend to set "0x00"			
0x2F	KbSource	1	Keyboard location selection. 0x00: Keyboard on the current page; other: Keyboard not on current page			
0x30	0xFE	1	0xFE			
0x31	PicKb	2	The following data is valid only if Kbsource is not 0x00. The index ID of the page where the keyboard resides			
0x33	AreaKb	8	Keyboard area coordinates: Upper-left corner (Xs, Ys) Lower-right corner (Xe, Ye)			
0x3B	AreaKbPosition	4	the location of keyboard area which pasted on the current page, the upper-left coordinates.			
0x3F	ScanMode1	1	0x02(Pinyin Input Method)			

Note: Pinyin "bd" corresponds to all GBK encoded fullwidth punctuation marks
Embedded LCD pre-installed font library (ID=0) which contains 4*8 ~ 64*128 dot matrix of all ASCII pattern.

6.3 Pop-up Menu

Address	Definition	Length (byte)	Note
0x00	TPID	2	Index ID of function picture
0x02	TPArea	8	Valid area of touch: Top left corner coordinates (Xs, Ys), Lower right corner coordinates (Xe, Ye).
0x0A	TraID	2	Switch picture index ID ; 0xFFxx: no switching
0x0C	AniID	2	Tap effect picture index ID; 0xFFxx: no effect to be display
0x0E	Code	2	0xFE01/0xFD01
0x10	0xFE	1	0xFE
0x11	*VP	2	Variable address pointer which corresponds to the input data. The returning data is determined by VType
0x13	VType	1	0x00: Write the key code 0x00** to the VP address (integer number)
			0x01: Write the Key code 0x** to the high-byte of the VP address (VP_H)
			0x02: Write the Key code 0x** to the low-byte of the VP

			address (VP_L)
			0x10-0x1F: Writes the lowest bit (1bit) of the key code 0x** to the specified bit of the data in VP address (0x10 modifies VP.0, 0x1F modifies the VP. F)
0x14	KBPicID	2	The index ID of the page where the keyboard resides
0x16	KBArea	8	Keyboard area: upper-left coordinate (Xs, Ys), lower right coordinate (Xe, Ye)
0x1E	KBPositionX	2	The keyboard position on the current page, the upper-left coordinates
0x20	0xFE	1	0xFE
0x21	KBPositionY	2	The keyboard position on the current page, the upper-left coordinates
0x23	Reserved	13	Recommend to set "0"
Valid Key code: 0x0000 ~ 0x00FF, where 0x00FF means cancel (return directly without parameter).			

6.4 Incremental Input

Address	Definition	Length (byte)	Note
0x00	TPID	2	Index ID of function picture
0x02	TPArea	8	Valid area of touch: Top left corner coordinates (Xs, Ys), Lower right corner coordinates (Xe, Ye).
0x0A	TraID	2	Switch picture index ID ; 0xFFxx: no switching
0x0C	AniID	2	Tap effect picture index ID; 0xFFxx: no effect to be display
0x0E	Code	2	0xFE02/0xFD02
0x10	0xFE	1	0xFE
0x11	*VP	2	Variable address pointer which corresponds to the input data. The returning data is determined by VType
0x13	VType	1	0x00: Adjust VP address (integer)
			0x01: Adjust the high-byte address of the VP word address (1byte unsigned integer, VP_H)
			0x02: Adjust the low-byte address of the VP word address (1byte unsigned integer, VP_L)
			0x10-0x1F: Adjust the specified bit of the VP address (0x10 corresponds to VP.0, 0x1F corresponding to the VP. F), the adjustment range must be set to 0 ~ 1.
0x14	AdjMode	1	Adjust mode
			0x00: -- other: ++
0x15	ReturnMode	1	Overflow processing
			0x00: stop other: cyclic adjustment
0x16	AdjStep	2	Step length adjustment, 0x0000 ~ 0x7FFF
0x18	VMin	2	Lower limit: 2 bytes integer (only low bytes valid when VPmode is 0x01 or 0x02)
0x1A	VMax	2	Upper limit: 2 bytes integer (only low bytes valid when VPmode is 0x01 or 0x02)
0x1C	KeyMode	1	0x00: Continuous adjustment when holding down the button
			0x01: Adjust only once when holding down the button
0x1D	NULL	3	Recommend to set "0"

6.5 Drag Input

Address	Definition	Length (byte)	Note
0x00	TPID	2	Index ID of function picture
0x02	TPArea	8	Valid area of touch: Top left corner coordinates (Xs, Ys), Lower right corner coordinates (Xe, Ye).
0x0A	TraID	2	Switch picture index ID ; 0xFFxx: no switching
0x0C	AniID	2	Tap effect picture index ID; 0xFFxx: no effect to be display
0x0E	Code	2	0xFE03/0xFD03
0x10	0xFE	1	0xFE
0x11	*VP	2	Variable address pointer which corresponds to the input data. The return data is determined by VType
0x13	VType	1	High 4 bits define the data return format
			0x0*: Adjust data in VP address (integer)
			0x1*: Adjust the high-byte of the VP Word address (1byte unsigned integer, VP_H)
			0x2*: Adjust the low-byte of the VP Word address (1byte unsigned integer, VP_L)
Low 4 bits define the drag mode	0x*0: Drag horizontally		
0x*1: Drag vertically			
0x14	AreaAdj	8	Adjusting area: (Xs, Ys), (Xe, Ye); must be consistent with TPArea (touch area)
0x1C	VBegain	2	Return value corresponding to the start position, integer
0x1E	VEnd	2	Return value corresponding to the end position, integer

6.6 RTC Control

Address	Definition	Length (byte)	Note
0x00	TPID	2	Index ID of function picture
0x02	TPArea	8	Valid area of touch: Top left corner coordinates (Xs, Ys), Lower right corner coordinates (Xe, Ye).
0x0A	TraID	2	Switch picture index ID ; 0xFFxx: no switching
0x0C	AniID	2	Tap effect picture index ID; 0xFFxx: no effect to be display
0x0E	Code	2	0xFE04/0xFD04
0x10	0xFE	1	0xFE
0x11	NULL	3	0x00 00 00
0x14	x, y	4	Input display location: Right alignment, (x, y) is the upper-right coordinate of the last character of the string
0x18	Color	2	Color of the input characters
0x1A	LibID	1	ASCII Font Library index ID of characters displayed
0x1B	FontHor	1	Font size, number of matrix dot in X-direction
0x1C	Reserved	1	Recommend to set "0"
			0x00: Keyboard on the current page

0x1D	KBSource	1	other: Keyboard is not on the current page
0x1E	KBPicID	2	The index ID of the page on which the keyboard resides which is valid when KBsource is not 0x00
0x20	0xFE	1	0xFE
0x21	KBArea	8	Keyboard area: upper-left coordinate (Xs, Ys), lower-right coordinate (Xe, Ye) which is valid when KBsource is not 0x00
0x29	KBPosition	4	The keyboard position on the current page, the upper-left coordinates which is valid when KBsource is not 0x00
0x2D	NULL	3	Recommend to set "0"

6.7 Key Value

Address	Definition	Length (byte)	Note
0x00	TPID	2	Index ID of function picture
0x02	TPArea	8	Valid area of touch: Top left corner coordinates (Xs, Ys), Lower right corner coordinates (Xe, Ye).
0x0A	TraID	2	Switch picture index ID ; 0xFFxx: no switching
0x0C	AniID	2	Tap effect picture index ID; 0xFFxx: no effect to be display
0x0E	Code	2	0xFE05/0xFD05
0x10	0xFE	1	0xFE
0x11	*VP	2	Variable address pointer which corresponds to the input data. The return data is determined by VType
0x13	VType	1	0x00: Key code saved in VP address (integer)
			0x01: The low byte of the key code is saved at the high-byte of the VP word address (VP_H)
			0x02: The low byte of the key code is saved at the low-byte of the VP word address (VP_L)
0x14	Code	2	The key code returned
0x16	NULL	10	Recommend to set "0"

6.8 Hardware parameter configuration

Hardware parameter configuration enables multiple functions, including data transfer between registers and variables, image conversion to monochrome diagrams, sending data to serial ports, and so on. Its instruction format is shown as following table.

Address	Definition	Data length	Note
0x00	Pic_ID	2	Page ID
0x02	TP_Area	8	Area of button: (Xs,Ys) (Xe,Ye)
0x0A	Pic_Next	2	Target page ID to switch to, 0xFF** means no switching
0x0C	Pic_On	2	Page ID of press effect, 0xFF** means no effect
0x0E	TP_Code	2	0xFE07
0x10	0xFE	1	0xFE
0x11	Mode	1	Operation mode, refer to table below
0x12	Data_Pack	14	Data pack, refer to table below

The operating mode of the hardware parameter configuration is shown in the following table. In the OGUS development software, the button icon that adds the feature is



Mode	Data_Pack	Note	Function																																																							
0x00	NA	NA	Load register space data to 0X6F00-0X6FFF variable space (occupy low bytes).																																																							
0x01	NA	NA	Load 0x6F00-0x6FFF variable space data to register space (occupy low bytes) and modified corresponding register R1-R3 , R5-RASD/SDHC interface configuration.																																																							
0x02	Tran_Area	Coordinates of target area : (Xs,Ys) (Xe,Ye)	<p>Converts the specified area into a monochrome bitmap (longitudinal mode print bitmap format) and saves it to the data memory to which the VP pointer points.</p> <ol style="list-style-type: none"> The width of the area (Xe-Xs+1) should be even. The height of the area (Ye-Ys+1) should be a multiple of 8. The data format to which *VP pointer points: <p>*VP: status, set to 0x5555 after processing. *VP+1: horizontal word length=(Xe-Xs+1) & 0xFFFE/2. *VP+2: number of data segment =(Ye-Ys+1) & 0xFFF8/8. *VP+3: start of bitmap data, MSB mode.</p> <p>If auto upload function is enabled, (R2.3=1), then after conversion, data in (*VP) will be modified to 0x5555 and send out a message automatically. This command is primarily used for printing screen.</p>																																																							
	*VP	The start addresses of the buffer for converted bitmap data																																																								
	<table border="1"> <thead> <tr> <th></th> <th>X=0</th> <th>X=1</th> <th>X=2</th> <th>X=3</th> <th>...</th> <th>X=126</th> <th>X=127</th> </tr> </thead> <tbody> <tr> <td>Y=0</td> <td>D0.15</td> <td>D0.7</td> <td>D1.15</td> <td>D1.7</td> <td>...</td> <td>D63.15</td> <td>D63.7</td> </tr> <tr> <td>...</td> <td>...</td> <td>...</td> <td>...</td> <td>...</td> <td>...</td> <td>...</td> <td>...</td> </tr> <tr> <td>Y=7</td> <td>D0.8</td> <td>D0.0</td> <td>D1.8</td> <td>D1.0</td> <td>...</td> <td>D63.8</td> <td>D63.0</td> </tr> <tr> <td>Y=8</td> <td>D64.15</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> </tr> <tr> <td>...</td> <td>...</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> </tr> <tr> <td>Y=15</td> <td>D64.8</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> </tr> </tbody> </table>			X=0	X=1	X=2	X=3	...	X=126	X=127	Y=0	D0.15	D0.7	D1.15	D1.7	...	D63.15	D63.7	Y=7	D0.8	D0.0	D1.8	D1.0	...	D63.8	D63.0	Y=8	D64.15													Y=15	D64.8						
	X=0	X=1	X=2	X=3	...	X=126	X=127																																																			
Y=0	D0.15	D0.7	D1.15	D1.7	...	D63.15	D63.7																																																			
...																																																			
Y=7	D0.8	D0.0	D1.8	D1.0	...	D63.8	D63.0																																																			
Y=8	D64.15																																																									
...	...																																																									
Y=15	D64.8																																																									
0x03	*VP	Data pointer	Send data which start at address pointed by *VP and has Tx_Len data length to serial port.																																																							
	Tx_Len	The length of data which to be sent	A word storing data length with range of 0x0001-0xFFFF.																																																							
0x04	Same as 0x03 but send data to COM2 (system reserved serial port).																																																									
0x05	Tran_Area	Coordinates of target area: (Xs,Ys) (Xe,Ye)	Converts the specified area into a monochrome bitmap (transverse mode print bitmap format) and saves it to the data memory to which the VP pointer points.																																																							
	*VP	The start addresses of the buffer for converted bitmap data																																																								

			<p>1. The width of the area (Xe-Xs+1) should be a multiple of 16.</p> <p>2. The data format to which *VP pointer points: *VP: status, set to 0x5555 after processing. *VP+1: horizontal word length=(Xe-Xs+1) & 0xFFF0/16. *VP+2: number of data segment =(Ye-Ys+1). *VP+3: start of bitmap data, MSB mode.</p> <p>If auto upload function is enabled, (R2.3=1), then after conversion, data in (*VP) will be modified to 0x5555 and send out a message automatically. This command is primarily used for printing screen.</p>
0x06	Frame_Head	Frame head (2 bytes)	Send the coordinates of the current click position to COM2 (system reserved serial port) in format: Frame_Head + X + Y + Check (the one-byte sum of X and Y) + Frame_end.
	Frame_End	Frame end (2 byte)	

6.9 Touch Synchronous Data Return

Address	Definition	Length (Byte)	Note
0x00	TPID	2	Index ID of function picture
0x02	TPArea	8	Valid area of touch: Top left corner coordinates (Xs, Ys), Lower right corner coordinates (Xe, Ye).
0x0A	TraID	2	Switch picture index ID ; 0xFFxx: no switching
0x0C	AniID	2	Tap effect picture index ID; 0xFFxx: no effect to be display
0x0E	Code	2	0xFE08/0xFD08
0x10	0xFE	1	0xFE
0x11	TP_OnMode	1	Data return mode for the first tap: 0x00: No State 0x01: Copy data which pointed by VP1S with the length of LEN1 to the variable space pointed by VP1D 0x02: Copy data which pointed by VP1S with the length of LEN1 to UART interface 0x03: Copy data which pointed by VP1S with the length of LEN1 to the register space pointed by VP1D Other: reserved
0x12	VP1S	2	Address of data to be read for the first tap
0x14	VP1D	2	Address of data to be written for the first tap
0x16	NULL	1	0x00

0x17	LEN1	1	Length of returning data in byte. When data returning mode is 0x01, the value must be even.
0x18	0xFE	1	0xFE
0x19	TP_ContinueMode	1	Data returning mode for the long tap: 0x00: No State 0x01: Copy data which pointed by VP1S with the length of LEN2 to the variable space pointed by VP1D 0x02: Copy data which pointed by VP1S with the length of LEN2 to UART interface 0x03: Copy data which pointed by VP1S with the length of LEN2 to the register space pointed by VP1D Other: reserved
0x1A	VP1S	2	Address of data to be read for the long tap
0x1C	VP1D	2	Address of data to be written for the long tap
0x1E	NULL	1	0x00
0x1F	LEN2	1	Length of returning data in byte. When Data return mode is 0x01, the value must be even.
0x20	0xFE	1	0xFE
0x21	TP_OffMode	1	Data return mode for tap off: 0x00: No State 0x01: Copy data which pointed by VP1S with the length of LEN3 to the variable space pointed by VP1D 0x02: Copy data which pointed by VP1S with the length of LEN3 to UART interface 0x03: Copy data which pointed by VP1S with the length of LEN3 to the register space pointed by VP1D Other: reserved
0x22	VP1S	2	Address of data to be read for tap off
0x24	VP1D	2	Address of data to be written for tap off
0x26	NULL	1	0x00
0x27	LEN3	1	Length of returning data in byte. When Data return mode is 0x01, the value must be even.
0x28	0x00	8	0x00

The three states of the Tap-On:



6.10 Rotation input

The rotation adjustment function (that is, Icon Rotation Display) realizes variable data entry by turning the knob, and its instruction storage format is shown in the following table. In the OGUS development software, the button icon to add the feature is



Address	Definition	Length (byte)	Note
0x00	0x5A05	2	Fixed value
0x02	*SP	2	Variable description pointer 0xFFFF: loaded by configuration file
0x04	0x000C	2	Fixed value
0x06	0x00	2	Variable pointer, variable format determined by VPMode

0x08	0x01	IconID	2	The specified icon index ID	
0x0A	0x02	IconXc	2	Rotation center position of the icon: X coordinates	
0x0C	0x03	IconYc	2	Rotation center position of the icon: Y coordinates	
0x0E	0x04	Xc	2	Rotation center position of the page: X coordinates	
0x10	0x05	Yc	2	Rotation center position of the page: Y coordinates	
0x12	0x06	VBegain	2	Variable values corresponding to the begin rotation angle, in integer, overflow data is not displayed.	
0x14	0x07	VEnd	2	Variable values corresponding to the end rotation angle, in integer, overflow data is not displayed.	
0x16	0x08	ALBegain	2	Begin rotation angle, 0 ~ 720(0x000-0x2D0), Unit: 0.5°	
0x18	0x09	ALEnd	2	End rotation angle, 0 ~ 720(0x000-0x2D0), Unit: 0.5°	
0x1A	0x0A: H	VPMODE	1	0x00: *VP point to an integer variable	
				0x01: *VP point to high-byte data of an integer variable	
				0x02: *VP point to low-byte data of an integer variable	
0x1B	0x0A: L	LibID	1	Icon library index ID	
0x1C	0x0B: H	Mode	1	Icon Display mode	0x00: Transparent (no background displayed)
					other: Display icon background

Note: Rotation is always assumed to be "clockwise" rotation, that is, ALEnd must be greater than ALBegain. Rotation input do not support keying control (register 0x4F).

7 Variable Display Configuration File

Variable display function configured by file 14*.bin which consists of one or more instructions that describes the function of a variable display.

Each instruction is fixed to occupy 32 bytes.

Each page is fixed with a 2KB or 4KB (0x1000 or 0x0800) variable storage space, and each page can be configured to display 64 or 128 variable functions (64/128 are selected by RC.4 in file CONFIG.TXT).

Variable display configuration file (14*.bin) is up to 2MB, configurable up to 1024 pages (Up to 512 pages in 128 display functions mode).

For variables of the same type, the lower the storage location, the higher the display priority.

A display variable instruction consists of 6 parts, refer to the table below:

No.	Definition	Length (Byte)	Note
1	0x5A	1	Fixed value
2	Type	1	Variable type
3	*SP	2	Destination address pointer for Variable description file which loaded from flash to data store. 0xFFFF: Do not load to data store.
4	Len_Dsc	2	The length of variable description
5	*VP	2	Variable address, 0x0000 ~ 0x6FFF, 0x0000 is used for the variable which need not specifying an address. This instruction will be canceled when the variable address has high bytes of 0xFF.
6	Description	N	Variable description

7.1 Variable Display Function List

No.	Code	Function	Note
01	0x00	Variable icon display	Linearly corresponds the range of a data variable to a set of icons. When the variable changes, the icon also automatically switches accordingly, mostly for fine dashboards, progress bars
02	0x01	Animation icon display	A constant data variable corresponds to 3 different icons indicating the status: do not display, display a fixed icon, and display an animation icon. Mostly used for alarm tips.
03	0x02	Progress Bar display	The range of a data variable is represented by the position of an icon (slider). Mostly used for liquid level, dial, schedule indication
04	0x03	Artistic character display	Replace the font library with icons to display variable data
05	0x04	Animation display	Play a set of full-screen pictures at the specified speed; Mostly for the boot interface or screensaver
06	0x05	Icon rotation display	The range of a data variable is linearly corresponding to the angle data, and then an icon is rotated according to the corresponding angle data, which is mostly used for display of pointer meter, etc.
07	0x06	Bit variable icon display	The state of each bit of a data variable corresponds to two of the 8 different display scenarios, which are displayed with icons (or icon animations). Mostly used for switch status display, such as fan operation (animation), stop (still icon).
08	0x10	Numeric Variable display	Display a numeric variable in the specified format (integer, decimal, unit) with a number of the specified font and size
09	0x11	Text display	Displays the string in the specified format (determined by the selected font library), displayed in the specified area
10	0x12_00	Text Format RTC display	Display RTC in text with customize format
11	0x12_01	Dial format RTC display	Rotate the icon and display the RTC in the form of a pointer dial
12	0x13	HEX data display	Separate variable data by byte (HEX) and display with specified ASCII characters. Mostly for timing display, such as showing 1234 as 12: 34
13	0x14	Scroll text display	Displays the specified content in the specified area using scrolling
14	0x20	Curve display	Auto Match command 0x84 write curve buffer data to display real-time curves (trend graph which can specify display area, center axis coordinates, display scale (zoom in/out)).
15	0x21	Drawing	Draw Basic Graphics, drawing command list as below.
16	0x22	List display	Display the data which defined by the two-dimensional array separately in tables.
17	0x25	2D barcode display	Displays the specified content in the specified location by means of 2D barcode

Drawing command list

Command	Function	Description
0x0001	Set point	Set point (x, y, color)
0x0002	Draw line	Multi-point Connection (color, (x0, y0), ... (xn, yn))
0x0003	Draw a rectangular box	Display a rectangular box with customize size, area, location
0x0004	Draw a rectangular area	Populates the specified area with the specified color
0x0005	Draw a circle box	Display a circle box with customize size, area, location
0x0006	Picture clipping display	Cut an area from the specified page to the current page
0x**07	Icon Display	Gets the icon from the specified icon library and displays
0x0008	Monochrome Area Fill	Specify the color fill in the specified area
0x0009	Draw a vertical line	Display vertical lines based on variable data, with customize colors, locations
0x000A	Draw Segments	Customize colors, locations, and display connections based on variable data
0x000B	Draw Arcs	Display arcs with customize radius, color, starting and ending angles
0x000C	Character display	Display characters based on variable data
0x000D	Reverse color display in rectangular area	Reverse color display of the specified area
0x000E	Two-color bitmap display	Display the appropriate color by bit based on variable data
0x000F	Bitmap display	Statement variable data display

7.2 Icon Display

7.2.1 Variable Icon Display (0x00)

Address	Definition	Length (byte)	Note
0x00	0x5A00	2	
0x02	*SP	2	Variable description pointer 0xFFFF: loaded by configuration file
0x04	0x0008	2	Fixed value
0x06	0x00 *VP	2	Variable pointer, variable is in integer
0x08	0x01 x, y	4	Variable display position, upper-left coordinates of icons
0x0C	0x03 Vmin	2	Variable lower limit, overflow data is not displayed
0x0E	0x04 Vmax	2	Variable upper limit, overflow data is not displayed
0x10	0x05 IconMin	2	Icon ID corresponding to Vmin
0x12	0x06 IconMax	2	Icon ID corresponding to Vmax
0x14	0x07: H IconLib	1	Icon Library location
0x15	0x07: L Mode	1	Icon display mode 0x00: Transparent (no background displayed) other: Display icon background

7.2.2 Animation Icon Display

Address		Definition	Length (byte)	Note
0x00		0x5A01	2	Fixed value
0x02		*SP	2	Variable description pointer 0xFFFF: loaded by configuration file
0x04		0x000A	2	Fixed value
0x06	0x00	*VP	2	Icon variable pointer, variable in double word, low word is reserved, high word is unsigned number (0x0000-0x0FFFF). Animation icon display is controlled by user data.
0x08	0x01	x, y	4	Variable display location, upper-left coordinates of icons
0x0C	0x03	0x0000	2	Fixed value
0x0E	0x04	VStop	2	Stop animation display when the variable equals this value
0x10	0x05	VStart	2	Automatically display the animation icon when the variable equals this value
0x12	0x06	IconStop	2	The icon corresponding to VStop
0x14	0x07	IconStart	2	When the variable equals VStart value, the icon is automatically displayed from IconStart to IconEnd to form an animation
0x16	0x08	IconEnd	2	
0x18	0x09: H	IconLib	1	Icon Library location
0x19	0x09: L	Mode	1	Icon Display Mode: 0x00: Transparent (no background displayed) other: Display icon background

Note: (VP +1) is a reserved bit that cannot be used. Icons or animations will not be displayed when variables are not equal to Vstop or Vstart.

7.2.3 Progress Bar Display

Address		Definition	Length (byte)	Note
0x00		0x5A02	2	Fixed value
0x02		*SP	2	Variable description pointer 0xFFFF: loaded by configuration file
0x04		0x000A	2	Fixed value
0x06	0x00	*VP	2	Icon variable pointer, variable in double word, low word is reserved, high word is unsigned number (0x0000-0x0FFFF). Animation icon display is controlled by user data.
0x08	0x01	VBegain	2	Variable values corresponding to the begin scale
0x0A	0x02	VEnd	2	Variable values corresponding to the end scale
0x0C	0x03	XBegain	2	Start scale X coordinates (Y coordinates in vertical)
0x0E	0x04	XEnd	2	End scale X coordinates (Y coordinates in vertical)
0x10	0x05	IconID	2	Icon ID of the progress bar
0x12	0x06	Y	2	The Y coordinates of the icon (X coordinates in vertical)
0x14	0x07: H	XAdj	1	The X coordinates offset of the icon (Y coordinates in vertical)
0x15	0x07: L	Mode	1	Scale mode
				0x00: Horizontal Bar 0x01: Vertical Bar
0x16	0x08: H	IconLib	1	Icon Library location
0x17	0x08: L	DisplayMode	1	Icon Display mode 0x00: Transparent (no background displayed)

				other: Display icon background
0x18	0x09: H	DataMode	1	0x00: *VP points to an integer variable
				0x01: *VP points to high-byte data of an integer variable
				0x02: *VP points to low-byte data of an integer variable

7.2.4 Artistic character display

Address	Definition	Length (byte)	Note	
0x00	0x5A03	2	Fixed value	
0x02	*SP	2	Variable description pointer 0xFFFF: loaded by configuration file	
0x04	0x0007	2	Fixed value	
0x06	0x00	*VP	2	Variable pointer
0x08	0x01	X, Y	4	Start position for display Right alignment mode, upper-right coordinate of the string. Left alignment mode, upper-left coordinate of the string.
0x0C	0x03	Icon0	2	The IconID corresponds to 0, the order is 0123456789 ...
0x0E	0x04: H	IconID	1	Icon library index ID
0x0F	0x04: L	IconMode	1	Icon Display mode 0x00: Transparent (no background displayed) other: Display icon background
0x10	0x05: H	Nint	1	Number of integer bits of data to be displayed.
0x11	0x05: L	NDot	1	Number of decimal bits of data to be displayed.
0x12	0x06: H	VType	1	0x00: integer (2 bytes): -32768 ~ +32767 0x01: long integer (4 bytes): -2147483647 ~ +2147483647 0x02: *VP high byte, unsigned integer, 0 ~ 255 0x03: *VP low byte, unsigned integer, 0 ~ 255 0x04: super long integer (8 byte): -9223372036854775808 ~ +9223372036854775807 0x05: unsigned integer (2 bytes), 0 ~ 65535 0x06: unsigned long integer (4 byte), 0 ~ 4294967295
0x13	0x06: L	Mode	1	0x00: Left alignment 0x01: Right alignment

7.2.5 Picture Animation Display

Address	Definition	Length (byte)	Note	
0x00	0x5A04	2	Fixed value	
0x02	*SP	2	Variable description pointer 0xFFFF: loaded by configuration file	
0x04	0x0004	2	Fixed value	
0x06	0x00	0x0000	2	Fixed value
0x08	0x01	PicBegin	2	Index ID of the first picture
0x0A	0x02	PicEnd	2	Index ID of the last picture
0x0C	0x03: H	FrameTime	1	Display time of one frame, unit: 8ms

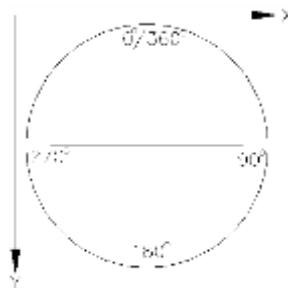
Note: The Begin picture index ID must be less than the End picture index ID. Picture loop display can be achieved by setting the picture animation variable of the end page. During the image animation process,

the picture animation can be ended by switching the page through the 0x80 command or touch instruction.

7.2.6 Icon Rotation Display

Address		Definition	Length (byte)	Note	
0x00		0x5A05	2	Fixed value	
0x02		*SP	2	Variable description pointer 0xFFFF: loaded by configuration file	
0x04		0x000C	2	Fixed value	
0x06	0x00	*VP	2	Variable pointer, variable format determined by VPMode	
0x08	0x01	IconID	2	The specified icon index ID	
0x0A	0x02	IconXc	2	Rotation center position of the icon: X coordinates	
0x0C	0x03	IconYc	2	Rotation center position of the icon: Y coordinates	
0x0E	0x04	Xc	2	Rotation center position of the page: X coordinates	
0x10	0x05	Yc	2	Rotation center position of the page: Y coordinates	
0x12	0x06	VBegain	2	Variable values corresponding to the begin rotation angle, in integer, overflow data is not displayed.	
0x14	0x07	VEnd	2	Variable values corresponding to the end rotation angle, in integer, overflow data is not displayed.	
0x16	0x08	ALBegain	2	Begin rotation angle, 0 ~ 720(0x000-0x2D0), Unit: 0.5°	
0x18	0x09	ALEnd	2	End rotation angle, 0 ~ 720(0x000-0x2D0), Unit: 0.5°	
0x1A	0x0A: H	VPMode	1	0x00: *VP point to an integer variable	
				0x01: *VP point to high-byte data of an integer variable	
				0x02: *VP point to low-byte data of an integer variable	
0x1B	0x0A: L	LibID	1	Icon library index ID	
0x1C	0x0B: H	Mode	1	Icon Display mode	0x00: Transparent (no background displayed)
					other: Display icon background

Note: Rotation is always assumed to be "clockwise" rotation, that is, ALEnd must be greater than ALBegain. The definition of angle as below:



7.2.7 Variable Bits Display

Address		Definition	Length (byte)	Note		
0x00		0x5A06	2	Fixed value		
0x02		*SP	2	Variable description pointer 0xFFFF: loaded by configuration file		
0x04		0x000C	2	Fixed value		
0x06	0x00	*VP	2	Variable pointer, variable format determined by VPMODE		
0x08	0x01	*VpAux	2	Auxiliary variable pointer, double byte, cannot be accessed by user software		
0x0A	0x02	ActBitSet	2	Bits that equals 1 indicate the corresponding bits of the *VP needs to be displayed.		
0x0C	0x03: H	DisplayMode	1	When DisplayMode is 0x02, the Icon0S icon is displayed when the corresponding variable (*VP) has a certain bit of "0"		
				DisplayMode	Variable Bit Value	
					0	1
				0x00	Icon0S	Icon1S
				0x01	Icon0S	not display
				0x02	Icon0S	Icon0S ~ Icon0E Animation
				0x03	not display	Icon1S
				0x04	not display	Icon0S ~ Icon0E Animation
				0x05	Icon0S~ Icon0E Animation	not display
				0x06	Icon0S~ Icon0E Animation	not display
0x07	Icon0S~ Icon0E Animation	Icon0S ~ Icon0E Animation				
0x0D	0x03: L	MoveMode	1	Bitmap Icon Arrangement		
				0x00: X++, do not keep DisMov position for the bit which specified non-displayed by ActBitSet.		
				0x01: Y++, do not keep DisMov position for the bit which specified non-displayed by ActBitSet.		
				0x02: X++, keep DisMov position for the bit which specified non-displayed by ActBitSet.		
0x03: Y++, keep DisMov position for the bit which specified non-displayed by ActBitSet.						
0x0E	0x04: H	IconMode	1	Icon Display mode 0x00: Transparent (no background displayed) other: Display icon background		
0x0F	0x04: L	IconLib	1	Icon library index ID		
0x10	0x05	Icon0S	2	Not display animation mode, bit 0, Icon index ID. Display animation mode, bit 0, start Icon index ID of icon animation.		
0x12	0x06	Icon0E	2	Display animation mode, bit 0, end Icon index ID of icon animation.		
0x14	0x07	Icon1S	2	Not display animation mode, bit 1, Icon index ID. Display animation mode, bit 1, begin start Icon index ID of icon animation.		

0x16	0x08	Icon1E	2	Display animation mode, bit 1, end Icon index ID of icon animation.
0x18	0x09	x, y	4	Starting bit variable display position, upper-left coordinate of icon
0x1C	0x0B: H	DisMov	2	Coordinate interval to the next icon
0x1E		Reserved	2	Recommend to set "0"

7.3 Text Display

7.3.1 Numeric Variable Display

Address	Definition	Length (byte)	Note	
0x00		0x5A10	2	Fixed value
0x02		*SP	2	Variable description pointer 0xFFFF: loaded by configuration file
0x04		0x000D	2	Fixed value
0x06	0x00	*VP	2	Variable pointer
0x08	0x01	x, y	4	Display position, upper-left coordinates of strings
0x0C	0x03	Color	2	Display color
0x0E	0x04: H	LibID	1	ASCII Font location
0x0F	0x04: L	FontSize	1	Number of matrix dot in X-direction of Character
0x10	0x05: H	Mode	1	0x00: Left alignment
				0x01: Right alignment
				0x02: Center
0x11	0x05: L	NIInt	1	Number of integer bits displayed
0x12	0x06: H	NDot	1	Number of decimal bits displayed
0x13	0x06: L	VType	1	0x00: integer (2 bytes), -32768 ~ +32767
				0x01: long integer (4 bytes), -2147483647 ~ +2147483647
				0x02: *VP high byte, unsigned integer, 0 ~ 255
				0x03: *VP low byte, unsigned integer, 0 ~ 255
				0x04: super long integer (8 bytes), -9223372036854775808 ~ +9223372036854775807
				0x05: unsigned integer (2 bytes), 0 ~ 65535
				0x06: unsigned long integer (4 bytes), 0 ~ 4294967295
0x14	0x07: H	Len	1	The length of the unit (for fixed string), 0x00: no unit to be display
0x15	0x07: L	StringUnit	Max11	Unit string (ASCII)

7.3.2 Text Display

Address		Definition	Length (byte)	Note	
0x00		0x5A11	2	Fixed value	
0x02		*SP	2	Variable description pointer 0xFFFF: loaded by configuration file	
0x04		0x000D	2	Fixed value	
0x06	0x00	*VP	2	Variable pointer	
0x08	0x01	X, Y	4	Display position, upper-left coordinates of string	
0x0C	0x03	Color	2	Display color	
0x0E	0x04	Area	8	Text box (Xs, Ys) (Xe, Ye)	
0x16	0x08	Textlength	2	Text length in byte. Text display ends at data 0xFFFF/0x0000 or Textbox tail	
0x18	0x09: H	Font0ID	1	The index ID of ASCII font library when encoding mode is 0x01~0x04	
0x19	0x09: L	Font1ID	1	The index ID of non-ASCII font library when encoding mode is 0x01~0x04 and the index ID of font library when encoding mode is 0x00/0x05.	
0x1A	0x0A: H	FontXDots	1	Number of matrix dot in X-direction of font (in mode 0x01 ~ 0x04, the X value of the ASCII character is calculated according to X/2)	
0x1B	0x0A: L	FontYDots	1	Number of matrix dot in Y-direction of the font	
0x1C	0x0B: H	Encode	1	[6: 0]: encoding mode	
				Value	Mode
				0	8bit encoding
				1	GB2312
				2	GBK
				3	BIG5
				4	SJIS
				5	UNICODE
				[7]: character spacing adjustment	
				Value	Note
0	Auto adjust				
1	not auto adjust				
0x1D	0x0B: L	HorDis	1	Character Horizontal interval	
0x1E	0x0C: H	VerDis	1	Character Vertical interval	
0x1F	0x0C: L	Reserved	1	Recommend to set "0"	

Note: The Y-direction matrix dot of the character must be an even number. The pre-installed font library (ID=0) includes 4*8 ~ 64*128 dot matrix ASCII characters.

7.3.3 Real Time Clock (RTC) Display

7.3.3.1 Text mode

Address		Definition	Length (byte)	Note
0x00		0x5A12	2	Fixed value
0x02		*SP	2	Variable description pointer 0xFFFF: loaded by configuration file
0x04		0x000D	2	Fixed value
0x06	0x00	0x0000	2	Fixed value
0x08	0x01	X, Y	4	Display position, upper-left coordinates of string
0x0C	0x03	Color	2	Display color of font
0x0E	0x04: H	LibID	1	Font library index ID
0x0F	0x04: L	FontSize	1	Number of matrix dot in X-direction of the font
0x10	0x05	StringCode	MAX16	String that displayed is defined by RTC encoding rules. For example: current time is 2014-03-22 16: 18: 50 Saturday, then: Mode1: Y-M-D H: M: S 0x00 display as: 2014-03-22 16: 18: 50 Mode2: M-D W H: M 0x00 display as: 03-22 SAT 16: 18

RTC encoding rules

Information	Code	Display Format
Year	Y	2000 ~ 2099
Month	M	01 ~ 12
Day	D	01 ~ 31
Hour	H	00 ~ 23
Minute	Q	00 ~ 59
Second	S	00 ~ 59
Week	W	SUN MON TUE WED THU FRI SAT
End sign	0x00	

7.3.3.2 Dial mode

Address		Definition	Length (byte)	Note
0x00		0x5A13	2	Fixed value
0x02		*SP	2	Variable description pointer 0xFFFF: loaded by configuration file
0x04		0x000D	2	Fixed value
0x06	0x00	0x0001	2	Fixed value
0x08	0x01	x, y	4	The center of the dial
0x0C	0x03	IconHour	2	The index ID of the hour hand Icon, 0xFFFF: do not display
0x0E	0x04	IconHourCentral	4	Rotation center position of the hour hand Icon
0x12	0x06	IconMinute	2	The index ID of the minute hand Icon, 0xFFFF: do not display

0x14	0x07	IconMinuteCentral	4	Rotation center position of the minute hand Icon
0x18	0x09	IconSecond	2	The index ID of the second hand Icon, 0xFFFF: do not display
0x1A	0x0A	IconSecondCentral	4	Rotation center position of the second hand Icon
0x1E	0x0C: H	IconLib	1	The index ID of Icon library
0x1F		Reserved	1	Recommend to set "0"

7.3.4 HEX Data Display

Address		Definition	Length (byte)	Note
0x00		0x5A13	2	Fixed value
0x02		*SP	2	Variable description pointer 0xFFFF: loaded by configuration file
0x04		0x000D	2	Fixed value
0x06	0x00	*VP	2	Variable pointer of start Address of data string. Variable is saved as BCD(HEX) code. For example: Data 0x32 displays as 32, Data 0xBF displays BF
0x08	0x01	X, Y	4	Display position, upper-left coordinates of strings
0x0C	0x03	Color	2	Font color
0x0E	0x04: H	ByteNum	1	Byte number showed in high byte of *VP pointer, 0x01 ~ 0x0F
0x0F	0x04: L	LibID	1	Font library index ID, halfwidth mode for Chinese character. If LibID is not 0, the font library should be 8-bit coding
0x10	0x05: H	Fontx	1	Number of matrix dot in X-direction of the font
0x11	0x05: L	StringCode	MAX15	Encode strings, combined with time variables, to get the display format required. Special characters are defined as follows: 0x00: Invalid, not display 0x0D: Line breaks, X=Xs, Y=Y+FontX*2

7.3.5 Text Scrolling Display

Address		Definition	Length (byte)	Note
0x00		0x5A13	2	Fixed value
0x02		*SP	2	Variable description pointer 0xFFFF: loaded by configuration file
0x04		0x000B	2	Fixed value
0x06	0x00	*VP	2	Text pointer, the first three bytes are reserved and the text start at (VP+3), stop at 0xFF/0x00
0x08	0x01: H	RollingMode	1	Rolling mode, from right to left
0x09	0x01: L	RollingDis	1	Scroll interval, number of pixels for each text scrolling cycle
0x0A	0x02: H	AdjustMode	1	Display Adjustment mode: 0x00: Left alignment

				0x01: Center 0x02: Right alignment
0x0B	0x02: L	RunControl	1	Scrolling control: 0x00: Run 0x01: Pause 0x02: Stop 0x03: Initialization (static display)
0x0C	0x03	Color	2	Text color
0x0E	0x04	(Xs, Ys) ~ (Xe, Ye)	8	Text box
0x16	0x08: H	Font0ID	1	Font library of ASCII character. The default font library index ID is 0 when coding mode is 0x00/0x05.
0x17	0x08: L	Font1ID	1	Font library index ID for Non-ASCII characters
0x18	0x09: H	FontXDots	1	Number of matrix dot in X-direction of Character, automatically calculated as X/2 when it is ASCII characters
0x19	0x09: L	FontYDots	1	Number of matrix dot in Y-direction of Character,
0x1A	0x0A: H	EncodeMode	1	[6: 0]: encode mode 0x00: 8Bits 0x01: GB2312 0x02: GBK 0x03: BIG5 0x04: SJUS 0x05: UNICODE [7]: character spacing adjustment 0: auto adjust 1: not auto adjust
0x1B	0x0A: L	TextDis	1	Character interval
0x1C	0x0B	Reserved	4	Recommend to set "0"

Note: The number of matrix dot in Y-direction of the character must be even number. The pre-installed font library (ID=0) includes 4*8 ~ 64*128 dot matrix ASCII characters.

7.4 Graph Display

7.4.1 Curve Display(0x20)

Address	Definition	Length (byte)	Note
0x00	0x5A20	2	Fixed value
0x02	*SP	2	Variable description pointer 0xFFFF: loaded by configuration file
0x04	0x000A	2	Fixed value
0x06	0x0000	2	undefined
0x08	0x01	8	Curve window: Top left corner coordinates (Xs, Ys) Lower right corner coordinates (Xe, Ye) The overflow data will not be displayed
0x10	0x05	2	Curve Center Axis location

0x12	0x06	VDCentral	2	The value of curve center axis: generally, it is average of maximum and minimum data
0x14	0x07	Color	2	Curve color
0x16	0x08	MuLY	2	Amplification Multiples of Y axis, Unit: 1/256, 0x0000 ~ 0x7FFF
0x18	0x09: H	Channel	1	Data source channel: 0x00 ~ 0x07
0x19	0x09: L	DisHor	1	Horizontal axis Interval: 0x01 ~ 0xFF

7.4.2 Basic Graphical Display

Address		Definition	Length (byte)	Note
0x00		0x5A21	2	Fixed value
0x02		*SP	2	Variable description pointer 0xFFFF: loaded by configuration file
0x04		0x0008	2	Fixed value
0x06	0x00	*VP	2	Variable data pointer
0x08	0x01	Area	8	Display area: from upper-left coordinates to lower-right corner coordinates and the overflow data will not be displayed. Available for command 0x0001 ~ 0x0005, 0x0009 ~ 0x000B.
0x10	0x05	Reserved	18	Recommend to set "0"

Note: A "drawing board" is defined in file 14*.bin firstly, the specific drawing operation is determined by the content in the variable space pointed to by *VP. Users implement different drawing functions by sending different data frames.

Variable data frame structure in variable space

Address	Definition	Note
VP	CMD	Drawing command
VP+1	Data_Pack_Num_Max	Maximum number of data packages, connection command (0x0002), number of connection lines is defined as (number of vertex - 1)
VP+2	Data_Pack	Data package

Drawing command format

Command (CMD)	Function	Command Frame structure			
		Address (Relative)	Length (Byte)	Definition	Note
0x0001	Draw dot	0x00	2	(x, y)	Coordinate of Dot, high byte of x parameter is condition.
		0x02	1	color	Dot color
0x0002	Draw line	0x00	1	color	Line color
		0x01	2	(x, y)0	Coordinate of endpoint0, high byte of x parameter is condition*.
		0x03	2	(x, y)1	Coordinate of endpoint1, high byte of x parameter is condition*.

	
		0x01+2*n	2	(x, y) n	Coordinate of endpointn, high byte of x parameter is condition*.
0x0003	Draw Rectangular box	0x00	2	(x, y) s	Upper-left corner coordinates of the box, high byte of x parameter is condition*.
		0x02	2	(x, y) e	Lower-right corner coordinates
		0x04	1	color	Rectangular box color
0x0004	Draw Rectangular Area	0x00	2	(x, y) s	Upper-left corner coordinates, high byte of x parameter is condition*.
		0x02	2	(x, y) e	Lower-right corner coordinates
		0x04	1	color	Rectangular area color
0x0005	Draw circle	0x00	2	(x, y)	Center point coordinates, high byte of x parameter is condition*.
		0x02	1	rad	Radius
		0x03	1	color	Circle color
0x0006	Cut picture area	0x00	1	Pic_Index	Picture index ID, high byte of x parameter is condition*.
		0x01	2	(x, y) s	Upper-left corner coordinates of cut area
		0x03	2	(x, y) e	Lower-right corner coordinates of cut area
		0x05	2	(x, y)	Upper-left corner coordinates of the target area to be pasted in current page.
0x**07	Icon display	0x00	2	(x, y)	Location of Icon, high byte of x parameter is condition*.
		0x02	1	Icon_Index	Icon index ID. The icon library position is indicated by the high byte of the command. Icon is displayed without background color.
0x0008	Area fill	0x00	2	(x, y)	Seed point coordinates, high byte of x parameter is condition*.
		0x02	1	color	Filled color
0x0009	Spectrum display	0x00	1	color	Connect (x0, y0s) (x0, y0e) with color lines, high byte of x0 is condition*.
		0x01	3	x0, y0s, y0e	
0x000A	Line segment display	0x00	1	color	Connect (x, y) s and (x, y) e with color lines, high byte of xs is condition*
		0x01	2	(x, y) s	
		0x03	2	(x, y) e	
0x000B	Arc display	0x00	1	color0	Arc color
		0x01	2	(x, y)0	Center point coordinates, high byte of x parameter is condition*.
		0x03	1	rad0	Radius
		0x04	1	Deg_s0	Starting angle, unit 0.5°, 0 ~ 720

		0x05	1	Deg_e0	Termination angle, unit 0.5°, 0 ~ 720	
0x000C	Character display	0x00	1	color0	Character color	
		0x01	2	(x, y)0	Starting position, high byte of x parameter is condition*.	
		0x03H	0.5	Lib_Index	Font library Index ID	
		0x03L	0.5	mode	Encoding mode: 0: 8bit 1: GB2312 2: GBK 3: BIG5 4: SJIS 5: UNICODE	
		0x04H	0.5	x_dots	Number of matrix dot in X-direction of Character	
		0x04L	0.5	y_dots	Number of matrix dot in Y-direction of Character	
		0x05	1	text0	Character data	
0x000D	Reverse color display in rectangular area	0x00	2	(x, y) s	Upper-left corner coordinates, high byte of x parameter is condition*.	
		0x02	2	(x, y) e	Lower-right corner coordinates	
		0x04	1	color	Color to be displayed, 0xFFFF: reverse current color.	
0x000E	Two-color bitmap display	0x00	2	(x, y) s	Upper-left corner coordinates of the rectangular area, high byte of x parameter is condition*.	
		0x02	1	x_dots	Number of matrix dot in X-direction	
		0x03	1	y_dots	Number of matrix dot in Y-direction	
		0x04	1	color0	Color corresponding to bit "0"	
		0x05	1	color1	Color corresponding to bit "1"	
		0x06	N	DataPack	Data to be displayed, MSB mode, word alignment: each line should start at a new word.	
0x000F	Bitmap display	0x00	2	(x, y) s	Upper-left corner coordinates of the rectangular area, high byte of x parameter is condition*.	
		0x02	1	x_dots	Number of matrix dot in X-direction	Limited by variable space, the maximum display bitmap is 196x146(4: 3)/226x126(16: 9)
		0x03	1	y_dots	Number of matrix dot in Y-direction	

		0x04	N	DataPack	Data to be displayed. Data format: MSB, 5R6G5B
--	--	------	---	----------	--

*Condition:

0xFF: the end of drawing operation

0xFE: ignore this operation

7.4.3 Table Display

Address		Definition	Length (byte)	Note	
0x00		0x5A22	2	Fixed value	
0x02		*SP	2	Variable description pointer 0xFFFF: loaded by configuration file	
0x04		0x000C	2	Fixed value	
0x06	0x00	*VP	2	Variable data pointer	
0x08	0x01: H	Tab_X_Num	1	Number of columns, 0x01 ~ 0xFF	
0x09	0x01: L	Tab_Y_Num	1	Number of rows, 0x01 ~ 0xFF	
0x0A	0x02: H	Tab_X_Start	1	Position of the first column, 0x00 ~ 0xFF	
0x0B	0x02: L	Tab_Y_Start	1	Position of the first row, 0x00 ~ 0xFF	
0x0C	0x03: H	Unit_Data_Num	1	0x01 ~ 0x7F	Data length for cell (in word), all cells have the same data length.
				0x00	The variable space pointed by the *VP pointer defines the data length (in word) of different column cells: table data content storage location corresponding deferred (UNIT_DATA_NUM/2) up to the entire word address For example: When *VP0=x1000, and Tab_X_Num = 7, 0x1000 ~ 0x1003 stores the table data length of column 0 to 6 in turn, Where the 0x1003 low bytes are not used; The 0x1004 address begins to store the table contents.
0x0D	0x03: L	Encode_Mode	1	[7]: Character interval adjustment 0: auto adjust 1: not auto adjust	
				[6]: Cell format 0: text format 1: defined by the first two words of the data	
				[5]: Border display setting 0: display border 1: not display border	

				[4]: undefined, recommend to set "0"
				[3: 0]: text encoding mode 0: 8bit 1: GB2312 2: GBK 3: BIG5 4: SJIS 5: UNICODE
0x0E	0x04	Area	8	Table area definition: Upper-left corner coordinates and Lower-right corner coordinates The table is always displayed from upper-left corner and ended when overflow.
0x16	0x08	color_line	2	Border line color
0x18	0x09	color_text	2	Text color
0x1A	0x0A: H	fontID0	1	ASCII font library index ID when encoding mode is 0x01 ~ 0x04
0x1B	0x0A: L	fontID1	1	Non-ASCII font library index ID when encoding mode is 0x00/0x05 or 0x01 ~ 0x04
0x1C	0x0B: H	Font_X_dots	1	Number of matrix dot in X-direction of Character, automatically calculated as X/2 when it is ASCII characters
0x1D	0x0B: L	Font_Y_dots	1	Number of matrix dot in Y-direction of Character,
0x1E	0x0C: H	Tab_X_Adj_Mod	1	When Tab_X_Start not equal 0x00, 0x00: not display first column 0x01: display first column
0x1F	0x0C: L	Tab_Y_Adj_Mod	1	When Tab_Y_Start not equal 0x00, 0x00: not display first row 0x01: display first row

When Encode_Mode.6 is set to "1", the cell format is defined by the first two words of the cell data as below

Function	Location	value	Definition	Note
Data type	High bytes of the first word	0x00	Integer, 2 bytes	-32768 ~ 32767
		0x01	Long integer, 4 bytes	-2147483648 ~ 2147483647
		0x02	High byte of *VP, unsigned integer	0 ~ 255
		0x03	Low byte of *VP, unsigned integer	0 ~ 255
		0x04	Super long integer, 8 bytes	-9223372036854775808 ~ 9223372036854775807
		0x05	Unsigned integer, 2 bytes	0 ~ 65536

		0x06	Unsigned long integer, 4 bytes	0 ~ 4294967295
		0x10	Time format 1	BCD string as 12: 34: 56
		0x11	时间格式 2	BCD string as 12-34-56
		0x12	时间格式 3	BCD string as YYYYER-MM-DD HH: MM: SS
		0xFF	Text	Text format
Data format	Low bytes of the first word	Data format of the decimal (data type is 0x00 ~ 0x06)		The high 4 bits is the bit number of integer and the low 4 bits is the bit number of the decimal.
		Time BCD string, (data type is 0x10 ~ 0x11)		Length of the BCD string
		others	undefined	
Text color	The second word	The color of the text displayed		

Note: If the actual length of the table is shorter than the length specified by unit_data_num, use 0xFFFF as the cell text terminator. For particularly large tables, through the touch screen operation to modify the Tab_x_start, Tab_y_start value can be very convenient to achieve the table positioning or dragging.

7.4.4 2D QR Code Display

Address		Definition	Length (byte)	Note	
0x00		0x5A12	2	Fixed value	
0x02		*SP	2	Variable description pointer 0xFFFF: loaded by configuration file	
0x04		0x0004	2	Fixed value	
0x06	0x00	*VP	2	2D code data pointer, data length up to 458 bytes, with 0x0000/0xFFFF as the end flag	
				Greater than or equal to 1 byte and less than 155 bytes	45x45 unit pixels
				Greater than or equal to 155 bytes and less than or equal to 458 bytes	73x73 unit pixels
0x08	0x01	X, Y	4	The upper-left position coordinates of the area	
0x0C	0x03	Size	2	Physical pixel dot size for each QR code unit pixel: 0x01 ~ 0x07. For example: when Size=2, each unit pixel will be displayed as a 2x2 dot matrix	
0x0D		Reserved	18	Recommend to set "0"	

8 OGUS Software User Guide

Orient display Graphic Utilized Software (OGUS) is a development tool running in PC to develop project which running in Embedded LCD. By which, we can design page layout, define functions, set display variables and generate configuration files downloaded to Embedded LCD by SD card.

In this unit, we will talk about,

- How to establish a development environment.
- How to prepare the development material.
- Start our development work with an example.

8.1 Establish a development environment.

8.1.1 Install OGUS software

To establish a development environment, we need to get the OGUS firstly. The last version of the software package can be downloaded from Orient Display website: <http://orientdisplay.com/> After that, extract the software package to the specified location and we can see,



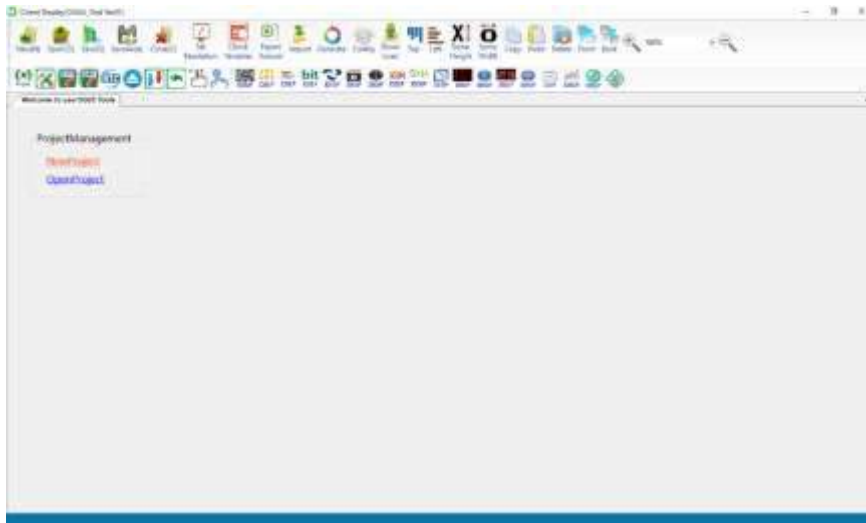
Name	Date modified	Type	Size
Config	2018-08-01 10:18 ...	File folder	
OD_0Font	2018-08-01 10:31 ...	Application	101 KB
OD_JCON	2018-08-01 10:31 ...	Application	176 KB
OD_ImgConversion	2018-08-01 10:31 ...	Application	411 KB
OGUS_Ver01	2018-08-01 10:31 ...	Application	7,486 KB

Where, OGUS_Ver01 is the executable file to start the OGUS.

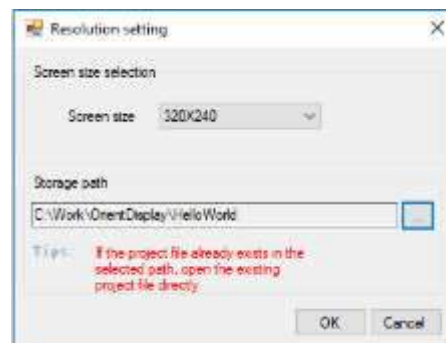
If OGUS can not be started, please check your setting of PC, the NetFramework2.0 should be installed previously. According to the system type (32bits/64bits) which can be found in File Explorer -> This PC -> Properties, install NetFramework2.0(x86) for 32bit system or NetFramework2.0(x64) for 64bit system respectively.

8.1.2 Create new project

When OGUS software installed correctly, start it we can see the work window as below:

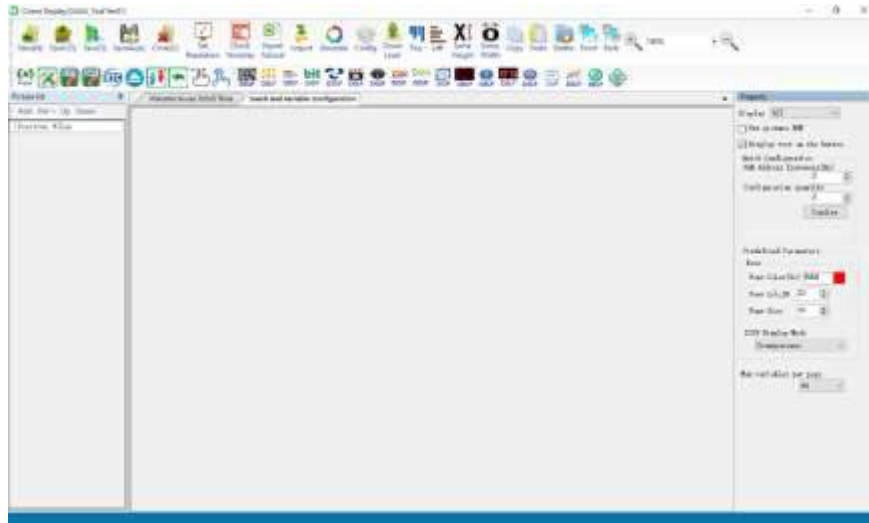


Click “NewProject” label to create a new project.



In Resolution setting dialog box,

1. Select Screen size which corresponds to hardware screen.
2. Set project path. In this case, we create a new folder which named as “HelloWorld”.
3. Click “OK” button to go to work interface as below.



After new project is created, a set of folders and files are created in project folder automatically.

Name	Date modified	Type	Size
ICON	2019-02-08 11:08 ...	File folder	
image	2019-02-08 11:08 ...	File folder	
OD_GUI	2019-02-08 11:08 ...	File folder	
TFT	2019-02-08 11:08 ...	File folder	
ODprj.html	2019-02-08 11:08 ...	HTML File	1 KB
ODprj.tft	2019-02-08 11:08 ...	TFT File	3 KB

8.2 Prepare development material

The material of a project usually includes Background pictures, Icon images. In project folder:

- The background pictures are usually stored in the image folder. The size of background picture should be adjusted to the same resolution with the hardware screen.
- The Icon images are usually stored in the ICON folder.
- All files that need to be download to Embedded LCD should be copied to OD_GUI folder and other configuration files such as 13*bin, 14*bin, config.txt which be created automatically also saved in this folder. OD_GUI folder stores all files of a project, it can be copied and downloaded by SD card easily in mass production.
- TFT folder is used to store .tft files which used by OGUS.

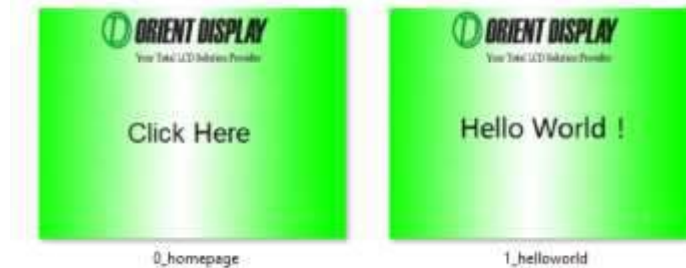
8.3 Work flow of project development

In this section, we introduce the work flow of development by creating a simple case. We would like to create an application which can show “Hello world!” when clicked on screen and return to the previous page when click on the “Hello World!”. By this way, the basic work flow will be shown step by step.

Step1: Prepare background pictures.

To achieved the target effect, two background pictures are needed. One for home page, shows information “Click Here” which named as “0_homepage.bmp”. The other one shows words “Hello World!” and named as “1_helloworld.bmp”.

The picture’s name should start with Index ID number and the Embedded LCD will display the

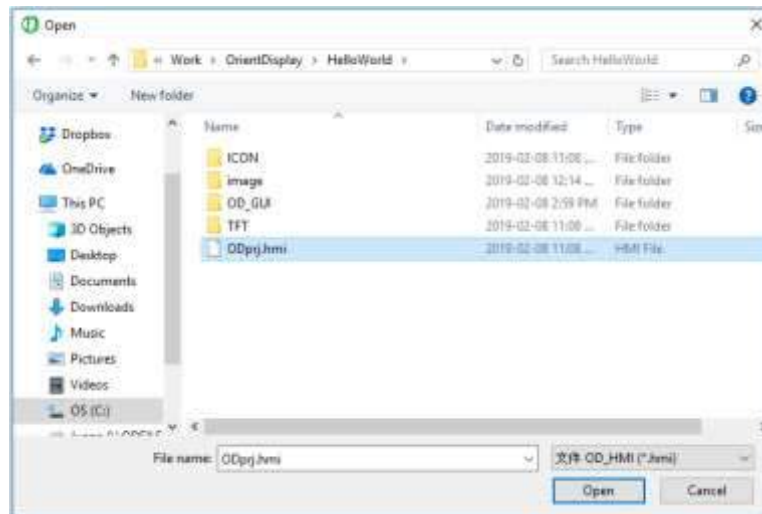


pictures from ID=0. As the resolution of this project is 320x240 (please refer to section 8.1.2), the size of picture should be adjusted to 320x240 pixels.

After adjustment, copy both pictures to folder OD_GUI.

Step2: Import background pictures

Start OGUS software and click “OpenProject” to open the existing project “HelloWorld” by



opening file “ODprj.hmi” in project folder.


Click the “Add” label in “Picture list” tool bar, select both pictures in OD_GUI folder and click “Open” button to import the background pictures. The figure is shown as below.



Then, the OGUS will display the first (ID=0) background picture in work area which under the label of “touch and variable configuration” and designer can select other background in “Picture list”.

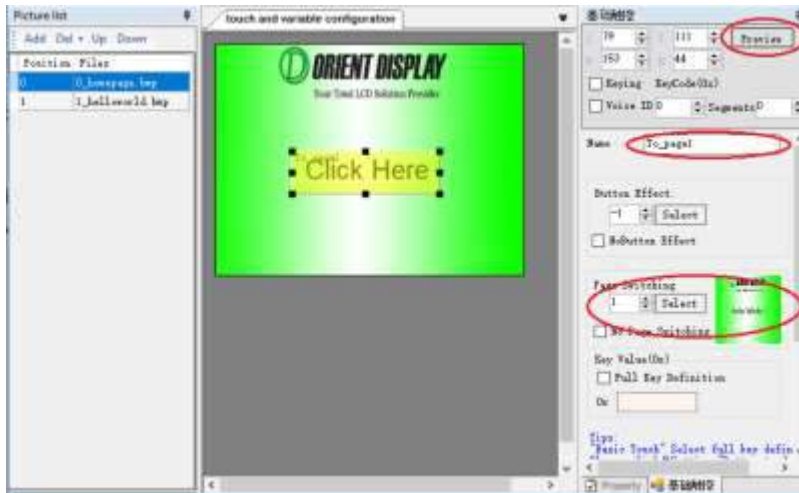


Step3: Add functions

Click the icon  and draw a function area overlap the words “Click Here”. Then, when user click this area on screen, Embedded LCD will jump to the page which indicated in the setting of this function.

As the figure below (in red circles), change the variable name and set the target page to be switched to.

The function can be demonstrated by clicking the button “Preview” in upper-right corner of the screen.



Repeat the same operations on page “1_helloworld”: set variable name as “To_page0” and set the target page as “0_homepage”.

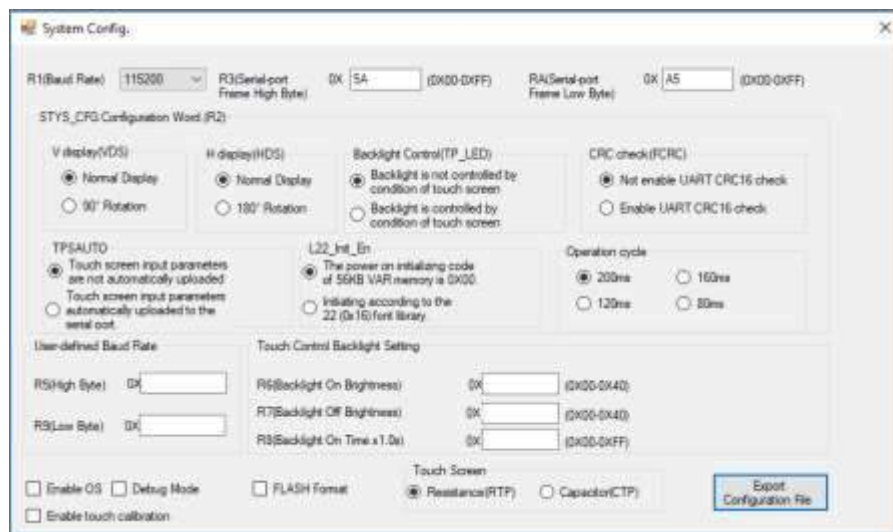
Step4: Generate configuration files

After adding all functions, the configuration files need to be generated.



Open the “system config” window by clicking label [Config](#).

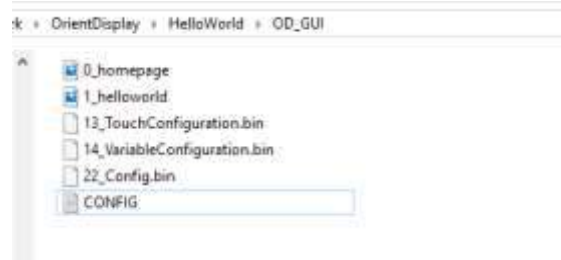
Click button “Export Configuration File” to generate configuration file “CONFIG.txt”.



Click label to generate the configuration files:

- 13_TouchConfiguration.bin
- 14_VariableConfiguration.bin
- 22_Config.bin

All configuration files can be found in folder OD_GUI.



Step5: Copy project folder to SD card.

Copy OD_GUI folder to root directory of SD card.

Step6: Download project to Embedded LCD

Plug SD card into SD slot of Embedded LCD and power on it. The file in folder OD_GUI will be downloaded into Embedded LCD automatically.



Then the project can be demonstrated now.
The first project “Hello world” finished!

Note: More teaching videos can be found in <http://orientdisplay.com/knowledge-base/embedded-lcds/>.