串口屏模块

BM32D4021-1

版本：V1.00　日期：2024-05-17

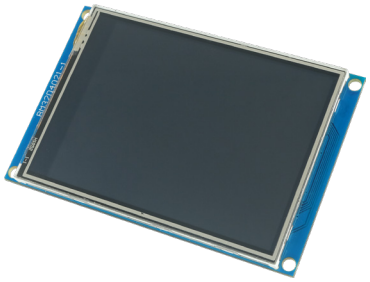
www.bestmodulescorp.com

目录

特性

串口屏模块系列现有 BM32D4021-1 (2.8 寸) 型号。总共拥有一种型号。下表是具体的模块信息。

特性内容		串口屏模块型号
		BM32D4021-1
TFT-LCD 屏	尺寸	2.8 寸
	分辨率	240×320
	颜色	RGB565
	显示方向	0° 横屏，90° 竖屏
模块数据	主频	240MHz
	Flash 容量	16M (Bytes)
	串口缓存区	256 (Bytes)
更新下载方式	SD 卡离线更新	√
	串口在线更新	√
	上位机 IAP 固件在线更新	√
低电耗	待机电流	<300μA
模块对外接口	串口	√
	SD 卡	√



概述

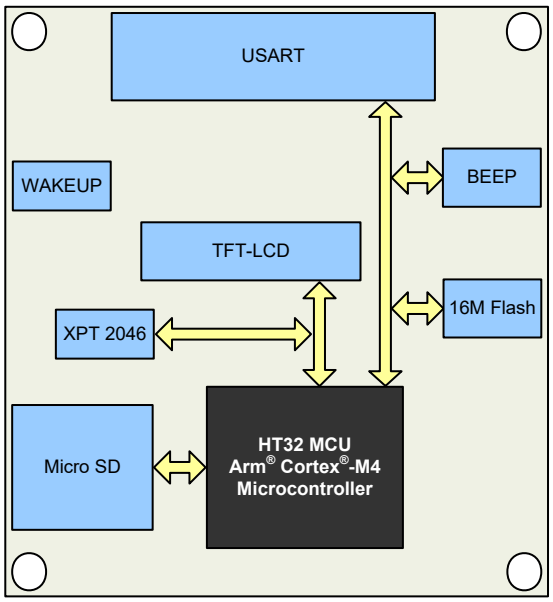
串口屏模块主控为 Holtek 32-bit Arm® Cortex®-M4 高性能单片机，可用于各种场景下需要屏幕显示和触控互动需求的中小型电子产品。

串口屏模块可使用相对应的上位机软件进行显示界面开发，上位机软件提供多种控件和多种指令，可让用户根据产品需求设计目标使用界面。在使用模块时，用户只需连接 VDD、GND 和串口线。在此基础上，串口屏模块提供 SD 卡 (离线下载) 和上位机串口 (在线下载) 两种方式对模块显示内容进行更新。

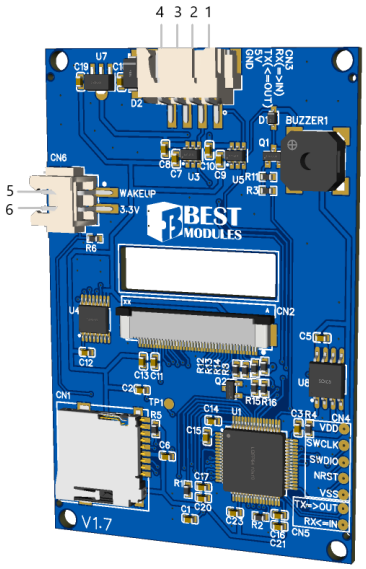
应用领域

- 医疗美容
- 智能家电
- 充电桩
- 共享设备
- 工业自动化

方框图


USART
WAKEUP
BEEP
TFT-LCD
16M Flash
XPT 2046
Micro SD
HT32 MCU
Arm® Cortex®-M4
Microcontroller

引脚图


4 3 2 1
5
6
BUZZER1
WAKEUP
3.3V
BEST MODULES
SWCLK
SWDIO
NRST
VSS
V1.7

引脚说明

引脚	功能	类型	说明
1	RX	I	模块串口 RX 信号线 (连接主控 TX 信号线)
2	TX	O	模块串口 TX 信号线 (连接主控 RX 信号线)
3	VDD	PWR	电源供电
4	GND	PWR	负电源
5	WAKEUP	I	按键 WAKEUP 唤醒功能脚 (将按键一端接入 WAKEUP)
6	3.3V	PWR	按键 WAKEUP 电源供电脚 (将按键另一端接 3.3V)

注：PWR：电源；　　I：数字输入；　　O：数字输出

技术规格

直流电气特性

Ta=25℃，V_{DD}=5V

参数	测试条件	最小	典型	最大	单位
工作电压 (V_{DD})	—	3.3	5.0	5.5	V
电流损耗	正常模式下工作	70	100	110	mA
	正常模式下蜂鸣器鸣响	130	164	175	mA
	睡眠模式下	—	51	—	mA
	深度睡眠	—	11	—	mA
	待机模式	—	300	—	μA

模块信息

参数内容	参数
工作频率	240MHz
RAM	224K (Bytes)
ROM	16M (Bytes)
工作温度	-25°C~85°C
尺寸	51mm×77.2mm
屏幕尺寸	2.8 寸
屏幕分辨率	240×320
颜色	RGB565
显示方向	0° 横屏，90° 竖屏

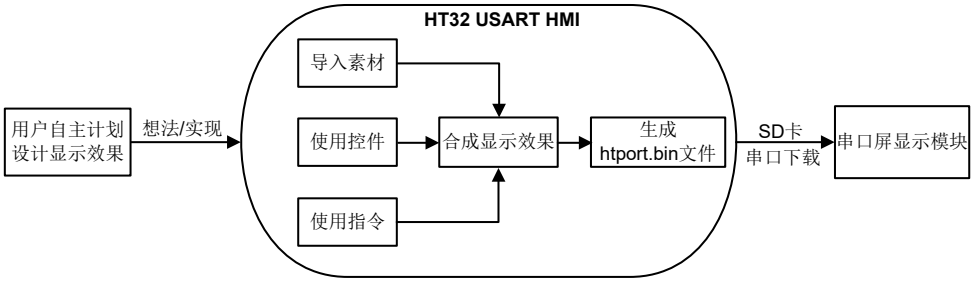
功能描述

系统描述

串口屏模块需要用户根据自身产品的显示需求，通过 HT32 USART HMI 进行使用界面的开发。

工作原理

用户在自主设计显示效果后，使用HT32 USART HMI上位机，通过导入的素材，加上控件和指令组合出预期的屏幕显示效果和使用方式，之后在 HT32 USART HMI 中生成 htport.bin 文档。通过 SD 卡或串口下载至串口屏显示模块即可。



串口屏模块使用流程图

休眠模式

为节省系统功耗，串口屏有三种方式进入休眠待机模式，进入不同的休眠方式需开启相对应的唤醒方式。

休眠方式	需开启的唤醒方式	唤醒方式
睡眠模式	开启触控屏幕和串口通信唤醒	触控屏幕和串口通信
深度睡眠	开启触控屏幕唤醒方式	触控屏幕
待机模式	接上 WAKEUP 唤醒按键	按下唤醒按键

通信接口

串口屏模块与上位机和主控通过 USART 进行通信。通信内容为自定义的串口协议包。

发送数据协议

数据包结构

帧头	设备地址	数据长度	负载数据	CRC 校验	帧尾
2-byte	2-byte	1-byte	N-byte	4-byte	2-byte

数据包结构解析

结构名称	结构说明	说明
帧头	固定的 0x55、0xAA	用于识别一个数据包的起始位置
设备地址	串口屏的设备地址	用于区分不同的设备 无效地址：0x0000 广播地址：0xFFFF
数据长度	数据包中负载数据的长度	负载数据的字节数
负载数据	传输的数据内容(使用GB2312编码)	具体传输的指令数据
CRC 校验	校验负载数据	采用的是 CRC32/MPEG-2
帧尾	固定的 0x0A、0xA0	用于识别一个数据包的结束

注：数据包的总大小不超过255个字节(包括帧头、帧尾、设备地址、数据长度、CRC校验值)。

CRC 校验示例

CRC 校验采用的是 CRC32/MPEG-2 的参数模型，以下是该模型具体的一些校验示例。

表格中的数据全部使用十六进制的格式表示。

需要校验的数据	校验计算结果
AE	83 A9 49 B0
B0 A1	47 27 5A 69
70 61 52 50	CE 7E F9 8C
63 74 72 6C 5F 72 65 20 62 31 2E 78 3D 62 74 30 2E 78 20	63 FC 74 30

数据包封包的流程

1. 在将数据封包之前，需要为数据包申请一个暂用的缓存空间。
2. 将帧头和设备地址赋值到缓存空间的对应地方。
3. 将指令数据转成 ASCII 码放置到负载数据部分，并计算负载数据的长度。
4. 将计算出来的数据长度赋值到缓存空间的数据长度部分。
5. 对指令数据进行 CRC 校验，并将校验计算结果赋到缓存空间的 CRC 校验段。
6. 为数据包赋上帧尾，保证一个数据包的完整。

封包中需要注意的事项

1. 帧头先 0x55 再到 0xAA，帧尾先 0x0A 再到 0xA0 结束。
2. 设备地址以高位在前，如 A1B1，在数据包中 A1 靠近帧头。
3. CRC 校验只校验负载数据，其余数据不在校验范围内。

封包示例

以下使用向设备地址为 A1B1 的串口屏发送 page page0 指令作为例子进行封包解析。

封包需要提供以下的数据：

设备地址：A1B1

负载数据：page page0

封包数据如下表所示：

帧头	设备地址	数据长度	负载数据	CRC 校验	帧尾
55 AA	A1 B1	0A	70 61 67 65 20 70 61 67 65 30	D1 5F 04 63	0A A0

封包后的数据	55 AA A1 B1 0A 70 61 67 65 20 70 61 67 65 30 D1 5F 04 63 0A A0

回传数据协议

数据包结构

帧头	设备地址	数据长度	数据类型	错误码	负载数据	CRC 校验	帧尾
2-byte	2-byte	1-byte	1-byte	1-byte	N-byte	4-byte	2-byte

数据包结构解析

结构名称	结构说明
帧头	固定的 0x55、0xAA
设备地址	当前设备的设备地址
数据长度	数据包中的数据的字节数 (包含数据类型和错误码)
数据类型	数据包中的负载数据类型，包含 3 种类型：1：错误码；2：字符数据；3：数值数据；当数据包中没有附带字符或数值数据时，返回的数据类型显示为错误码，如果有附带字符或数值数据时则返回对应的数据类型
错误码	数据包中返回的错误码，错误码说明请参考错误码章节。仅当系统中设置了需要返回错误码的情况才会返回指令执行的对应错误码，否则返回 0xFF
负载数据	串口屏回传的数据内容
CRC 校验	采用的是 CRC32/MPEG-2
帧尾	固定的 0x0A、0xA0

串口屏回传数据包示例

以下用读取页面中数字控件 num1 的 Y 坐标值的回传数据作为例子。

发送指令	ctrl_read num1.Y
回传数据	55 AA 00 01 04 03 01 35 30 D5 56 E3 42 0A A0

根据回传数据包结构对数据进行分解。

帧头	设备地址	数据长度	数据类型	错误码	负载数据	CRC 校验	帧尾
55 AA	00 01	04	03	01	35 30	D5 56 E3 42	0A A0

数据解析后得到如下结果：

设备地址：0x0001

数据长度：4 个数据 (包括 1 个数据类型，1 个错误码和 2 个负载数据)

数据类型：0x03 (负载中的数据为数值型数据)

错误码：0x01 (参照指令执行结果编码，该值为指令成功执行的返回码)

负载数据：0x35 0x30 (负载数据为 GB2312 编码，两个数值解码后为 50)

CRC 校验码：校验的数据包括数据类型，错误码和负载数据

上位机软件

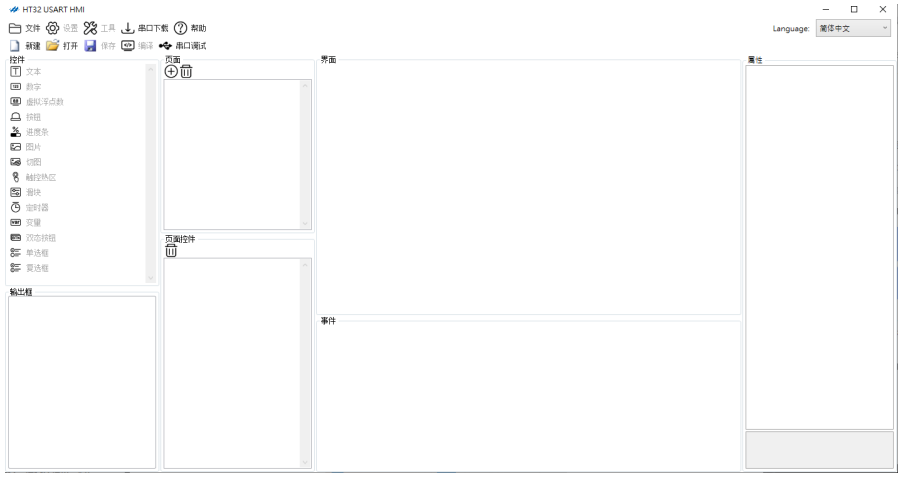
HT32 USART HMI 是一个 HT32 串口屏模块的辅助开发软件，是为了让用户更方便的使用 HT32 串口屏模块而开发的一款应用程序。用户可以通过图形化操作来完成屏幕界面的设计等工作。由于其简单易用的特性，可大大缩短开发者的开发时间。HT32 USART HMI 需搭配串口屏模块一起使用。

安装

- HT32 USART HMI 安装于 Microsoft Win7/Win8/Win10/Win11 桌面操作系统
- 系统需要安装 .NET Framework 4.8 或者更新版本。一般 Win10/Win11 都已经随系统安装，若没有安装，请于 Microsoft 官网下载安装：https://dotnet.microsoft.com/download
- .NET Framework 4.8 具体安装过程，按提示一直点“下一步”就可以完成安装
- HT32 USART HMI 上位机软件于网盘中进行获取
- 网盘链接：https://wwyz.lanzoul.com/b04w7km5g
- 网盘密码：8rl6

功能介绍

在这个章节，将介绍工具的基本操作，主界面如下图所示。

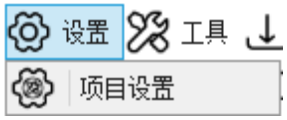


菜单栏

“文件”菜单如下图：

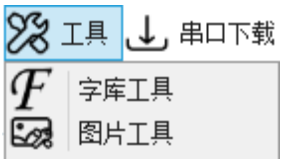
–新建：创建新的工程
–打开：打开已经创建的工程
–保存：对正在编辑的工程进行存档
–另存为：对正在编辑的工程另外进行存档
–生成 Bin 文件：生成工程的 Bin 文档
–退出：退出软件

“设置”菜单：



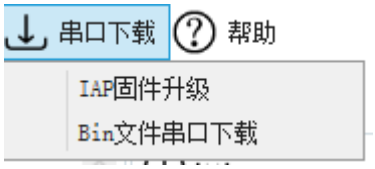
–项目设置：查看和设置串口屏的型号、显示方向、编码方式、文件写入次数和设备地址

“工具”菜单：



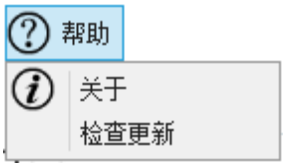
–字库工具：添加字库
–图片工具：添加图片

“串口下载”菜单：



–IAP 固件升级：用于升级串口屏的固件信息
–Bin 文件串口下载：用于下载串口屏的显示信息

“帮助”菜单：



–关于：Holtek 官方网址和建议反馈邮箱
–检查更新：用于更新最新版本的 HT32 USART HMI

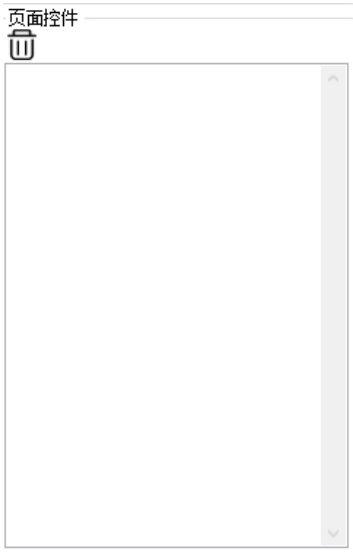
控件种类框

控件种类框部分列出了目前所有可用的控件，用户可在这里点击所需的控件来添加到页面。

页面设置框

页面设置框可以进行添加、删除页面操作。

页面控件设置框

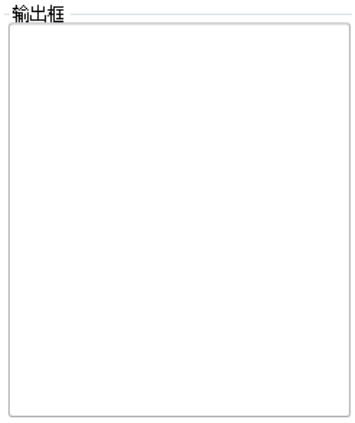


在页面控件框中可以选择已添加的控件来进行控件的属性修改和删除不需要的控件。

事件框

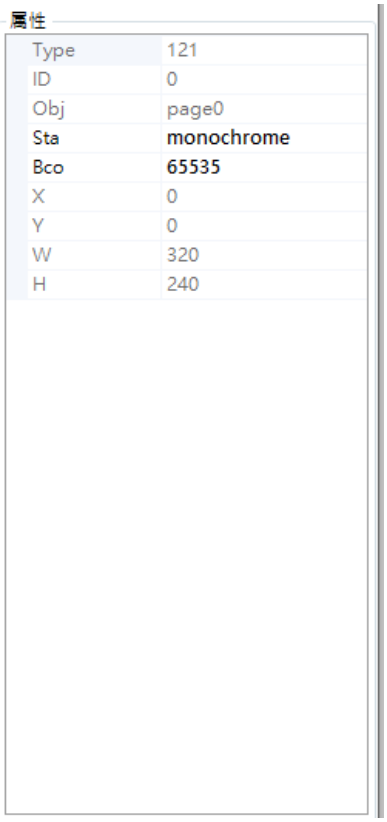
用户可以根据自己的需求在对应的事件中编辑需要执行的指令。

输出框



用户点击编译功能后，输出框中会输出各页面和素材所占用的字节数，并提醒用户是否编译成功。

属性框



当用户在页面控件框中选中控件时，属性框中会显示被选中控件的属性参数。用户可以通过选中不同的控件，然后在属性框中对该控件的属性参数进行修改，从而达到不同的显示效果。

界面框

用户在页面控件框所选中的控件，会根据控件属性，在界面框中进行显示。

IAP 固件升级

IAP 固件升级需要用户提前下载好需要更新的固件，使用串口工具将串口屏与电脑进行连接，选择对应的串口号以及需要更新的固件，点击 IAP 固件更新对串口屏进行固件更新。

Bin 文件串口下载

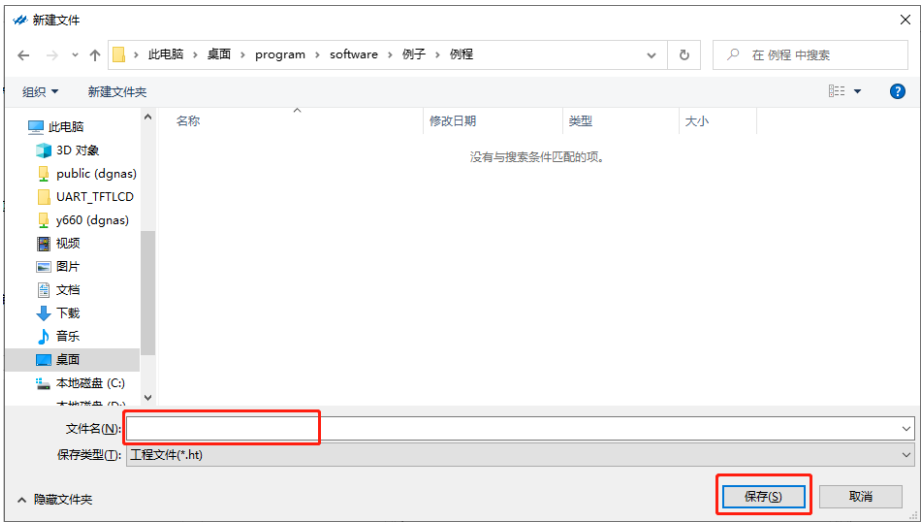
Bin 文件串口下载主要是将用户当前编辑的页面显示效果下载到串口屏中。在将显示文件下载到串口屏之前需要先将当前编辑的页面进行编译，只有编译成功后才能进行显示文件下载。在串口下载菜单中选择 Bin 文件串口下载功能，上位机会弹出上图所示的串口下载界面，使用串口工具将串口屏与电脑进行连接，并选择对应的串口，点击更新下载，上位机即可自动进行显示文件下载。同时串口下载窗口中会显示当前下载的显示文件 (Bin 文件) 的总大小以及下载的进度。

操作流程

新建工程

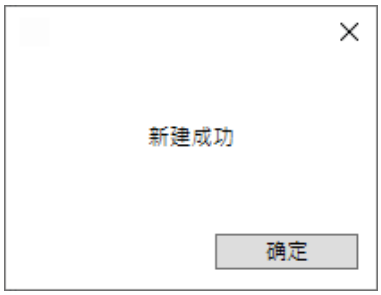
点击新建选项。

会弹出新建文件界面，输入文档的名称点击保存即可。



保存文档后会弹出设置界面，在设置界面选择对应的串口屏型号等信息，点击确认就完成新建工程。

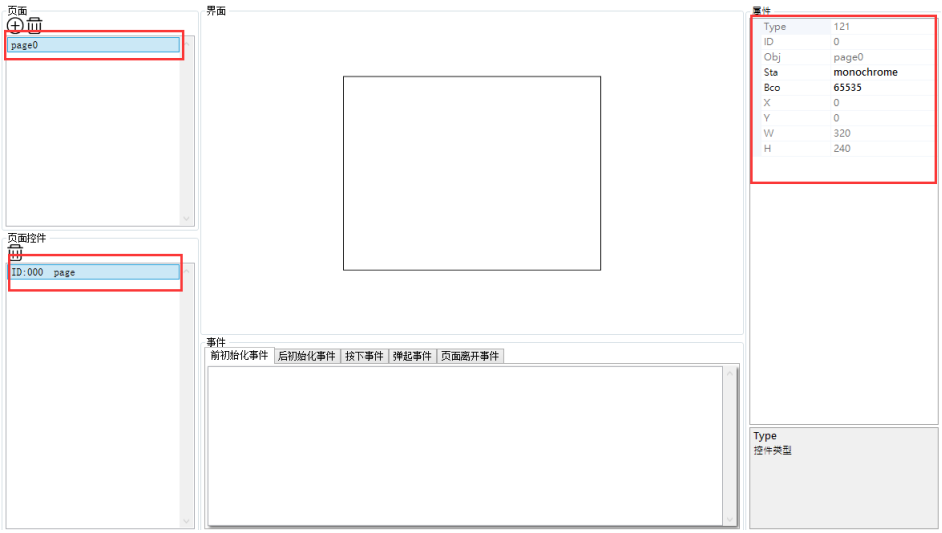
界面会弹出新建成功提醒。



添加页面

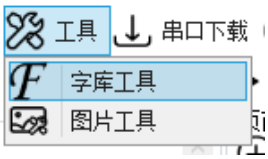
在页面设置框中点击“+”便可添加页面。

添加好页面后会分别在页面设置框，页面控件框，属性框显示页面的信息。

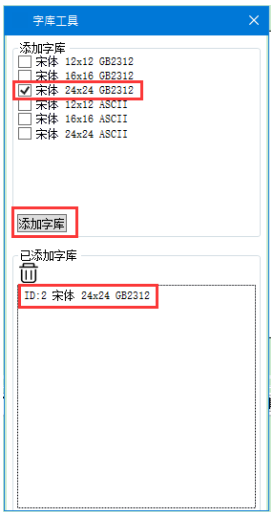


添加字库

点击工具栏，点击字库工具会弹出添加字库的界面。

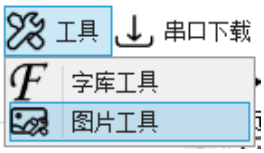


选择需要添加的字库类型，点击添加字库，下方已添加框内会出现添加的字库表示添加成功。

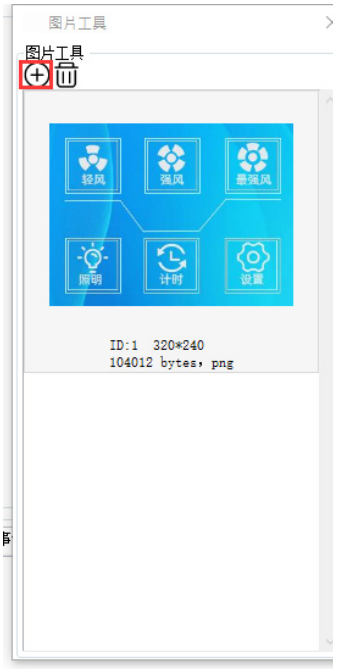


添加图片

点击工具栏，点击图片工具会弹出添加图片的界面。

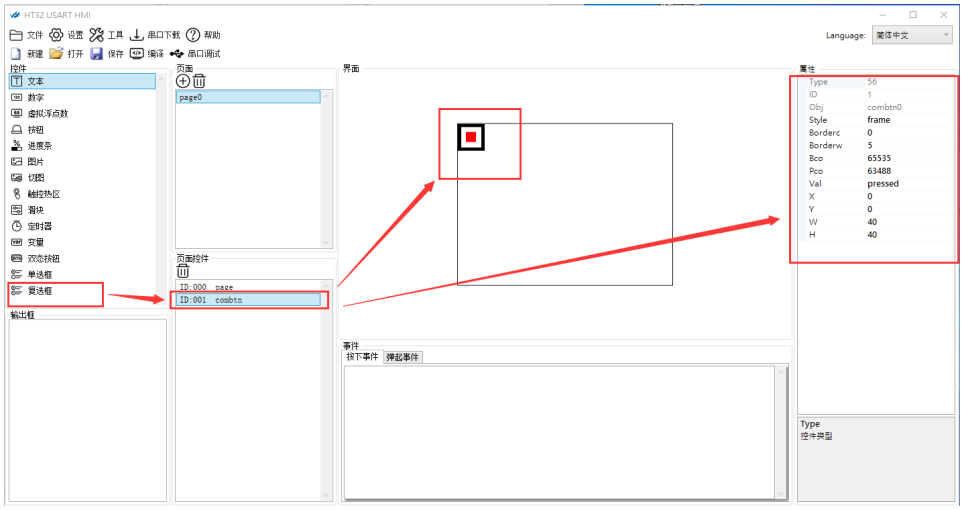


点击“+”选择需要添加的图片文件即可。



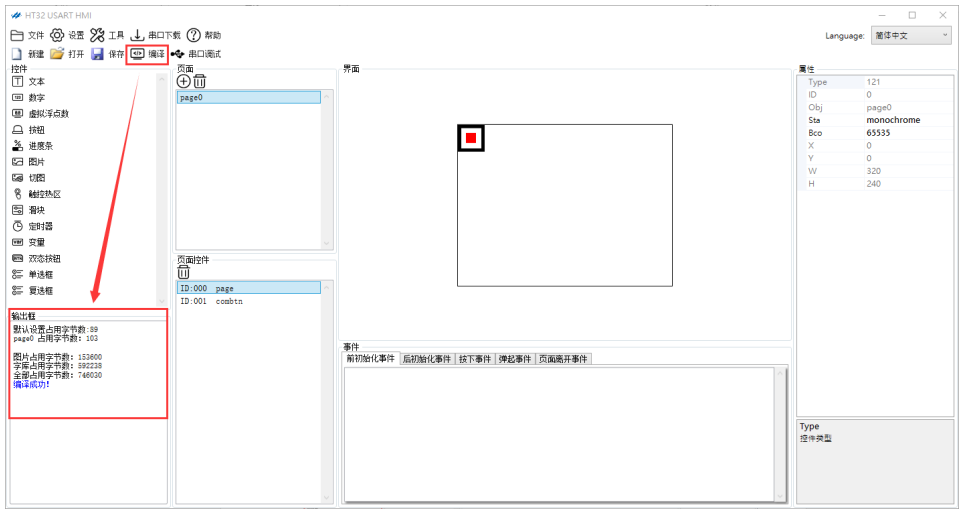
添加控件

添加控件需要在控件列表中点击需要添加的控件，点击后在页面控件框中会显示添加的控件。在属性框中设置添加控件的参数，详细参数解释将在后面控件属性章节进行介绍。



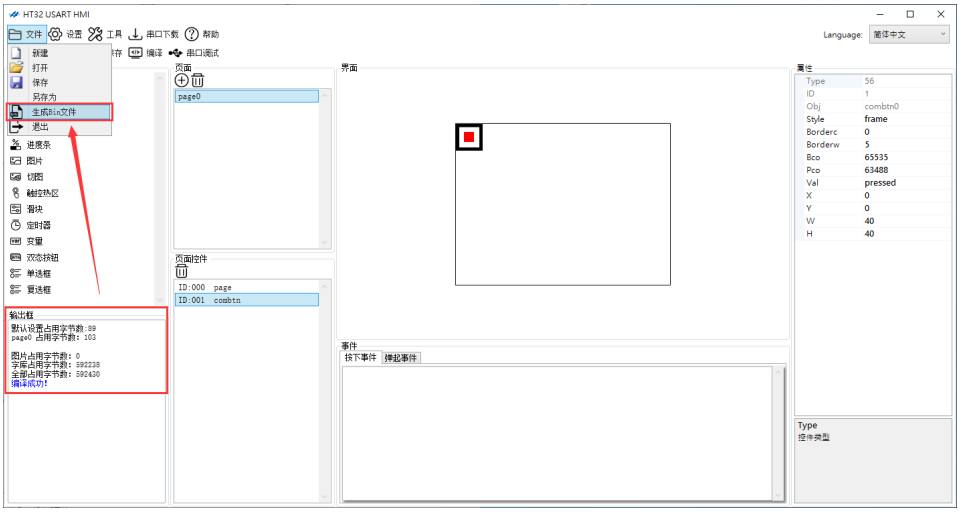
编译

工程设计完成后，可以点击编译。编译成功便可进行 Bin 文档生成。



生成 Bin 档

在文件目录下，点击生成 Bin 文档，Bin 文档生成成功后会保存在工程文件相同的文件夹中。



控件详解

认识控件

A. 控件是什么

控件是串口屏想实现相关功能的“功能元件”，需要串口屏显示什么内容就使用相对应的控件。例如：

显示文本，用【文本控件】

显示图片，用【图片控件】

显示进度，用【进度条控件】

实现按键功能，用【按钮控件】

实现滑动功能，用【滑块控件】

在串口屏上位机的界面编辑窗口中，一切你能看到的，都是控件。也有一些控件是看不见的，比如定时器等控件。

B. 控件事件是什么

控件事件指的是这个控件被操作时要执行的功能。

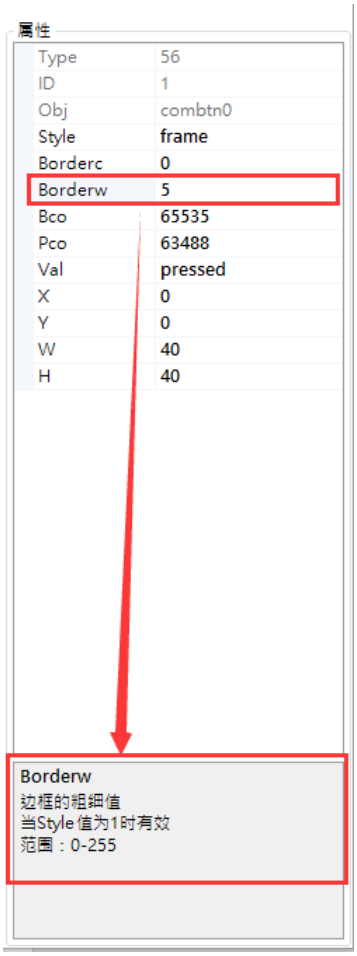
目前各种控件综合起来被操作的方式有以下几种类型：

1. 触控被按下：对应名称叫做【按下事件】
2. 触控被按下后弹起：对应名称叫做【弹起事件】
3. 滑块控件被滑动：对应的名称叫做【滑动事件】
4. 定时器定时运行：对应的名称叫做【定时事件】

C. 控件属性解析

控件属性是控件自己的一些设置项，上面提到想要什么功能就选择对应的控件。比如想要显示文本，就用文本控件，但是选择文本控件后，显示什么内容？什么字体？字体什么颜色？文本背景什么颜色？字体间距多少等等这些信息怎么设置呢？这就需要属性来定义了，这些信息都属于这个文本控件的属性，每个控件都有很多属性可以设置，用来定义它的显示效果。

选中属性框的任一属性，最底部会显示该属性的注释。



D. 注意事项

1. 由于很多控件需要用到字库，所以在使用串口屏时不要忘记添加字库。
2. 串口屏的型号有多种，在新建工程时要选择对应的型号和其他参数。

页面控件

页面控件拥有以下五个事件：

前初始化事件

每次在执行 page 命令时，在执行页面刷新操作以前，串口屏会自动执行一次“前初始化事件”中的代码。

后初始化事件

每次在执行 page 命令时，在页面刷新操作完成以后，串口屏会自动执行一次“后初始化事件”中的代码。

页面按下事件

显示区域内且没有控件的区域，触控被按下的瞬间，串口屏会自动执行一次页面的“按下事件”中的代码。

页面弹起事件

显示区域内且没有控件的区域 (以触控按下时的坐标为准)，触控按下以后松开触控的瞬间，串口屏会自动执行一次页面的“弹起事件”中的代码。

页面离开事件

每次在执行 page 切换新的页面前，串口屏会自动执行一次当前页面的“页面离开事件”中的代码。

- **页面控件 – 属性详解**

名称	范围	备注	说明
Type	121	控件类型	用于识别当前控件的类型，不可读写。
ID	0	控件 ID	用于区分同一个页面中控件所处的不同层，不可读写。
Obj		控件的名称	当需要操作控件时，该属性用于定位当前页面中指定的控件，不可读写。
Sta	0~2	背景填充方式 0：无背景 1：单色 2：图片	该属性决定了控件的背景填充模式，可通过上位机修改，但无法使用指令来读写。 注：无背景填充方式，在跳转到新页面时不会将旧页面擦除。
Bco	0~65535	背景色，当 Sta 为 1 时有效	用于填充背景的颜色值，可使用上位机或指令进行读写。
Picid		背景图片，当 Sta 为 2 时有效	用于填充控件背景的图片，可通过上位机修改，但无法使用指令来读写。 注：系统如果没有查找到指定的图片，将自动以无背景填充方式对控件的背景进行填充。
X	0	控件的原点 X 坐标	页面控件的原点 X 坐标为固定值，无法通过上位机或指令进行读写。
Y	0	控件的原点 Y 坐标	页面控件的原点 Y 坐标为固定值，无法通过上位机或指令进行读写。
W		控件的宽度	页面控件的宽度为固定值，根据屏幕的分辨率来决定。 无法使用上位机或指令进行读写。
H		控件的高度	页面控件的高度为固定值，根据屏幕的分辨率来决定。 无法使用上位机或指令进行读写。

注：页面控件以左上角的顶点作为控件的原点。

文本控件

用于在串口屏上显示文本信息，包括数字，字母，符号，汉字和其他各国语言，使用前需要提前导入字库。

文本控件有两个控件事件：

按下事件

弹起事件

- 文本控件 – 属性详解

名称	范围	备注	说明
Type	116	控件类型	用于识别当前控件的类型，不可读写。
ID		控件 ID	用于区分同一个页面中控件所处的不同层，不可读写。
Obj		控件的名称	当需要操作控件时，该属性用于定位当前页面中指定的控件，不可读写。
Sta	0~2	背景填充方式 0：切图 1：单色 2：图片	该属性决定了控件的背景填充模式，可通过上位机修改，但无法使用指令来读写。
Piccid		切图背景，当 Sta 为 0 时有效	作为控件切图背景的图片，可通过上位机修改，但无法使用指令来读写。 注：切图图片必须为全屏图片。
Style	0~1	单色显示风格，当 Sta 值为 1 时有效 0：平面 1：边框	该属性决定了控件在单色填充模式下的显示样式，可通过上位机或指令进行读写。
Borderc	0~65535	边框的颜色值，当 Style 值为 1 时有效	用于决定控件边框的颜色值，可通过上位机和指令进行读写。
Borderw	0~255	边框的粗细值，当 Style 值为 1 时有效	用于决定控件边框的粗细值，可通过上位机和指令进行读写。
Bco	0~65535	背景颜色值，当 Sta 值为 1 时有效	用于填充控件的背景颜色值，可使用上位机或指令进行读写。
Picid		背景图片，当 Sta 为 2 时有效	用于填充控件背景的图片，可通过上位机修改，但无法使用指令来读写。
Fontid		控件调用的字库	用于决定当前控件所使用的字库，不同字库显示的字体和字形也不同，可通过上位机修改，但无法使用指令来读写。
Pco	0~65535	字体颜色值	用于决定控件中显示的字体颜色，可使用上位机或指令进行读写。
Xcen	0~2	水平对齐 0：靠左 1：居中 2：靠右	用于决定控件中显示的字体的水平位置，可使用上位机或指令进行读写。
Ycen	0~2	垂直对齐 0：靠上 1：居中 2：靠下	用于决定控件中显示的字体的垂直位置，可使用上位机或指令进行读写。

名称	范围	备注	说明
Pw	0~1	是否显示为密码 0：否 (正常显示) 1：是 (显示成 *)	用于决定控件中字体的显示模式，可使用上位机或指令进行读写。
Isbr	0~1	是否自动换行 0：否 1：是	用于决定当控件中的字体超出显示的范围时是否自动进行换行操作，可使用上位机或指令进行读写。
Spax	0~255	两个字符的水平间距	用于决定控件显示的字符的水平间距，可使用上位机或指令进行读写。
Spay	0~255	两个字符的垂直间距	用于决定控件显示的字符的行间距，可使用上位机或指令进行读写。
X		控件的原点 X 坐标	控件的 X 坐标值加上控件的宽度值不能超过显示区域的最大宽度，可使用上位机或指令进行读写。
Y		控件的原点 Y 坐标	控件的 Y 坐标值加上控件的高度值不能超过显示区域的最大高度，可使用上位机或指令进行读写。
W		控件的宽度	控件的宽度值加上 X 坐标值不能超过显示区域的最大宽度，可使用上位机或指令进行读写。
H		控件的高度	控件的高度值加上 Y 坐标值不能超过显示区域的最大高度，可使用上位机或指令进行读写。
Txt_maxl		保存 Txt 内容的字节数	上位机自动识别输入的 Txt 字符所需的字节数，无法通过指令读写，可通过修改 Txt 间接修改。
Txt		控件显示的字符	控件中需要显示的字符，可使用上位机或指令进行读写。 注：使用指令修改时需要确保修改后的字符串长度不超过 Txt_maxl 所限定的长度。

注：文本控件以左上角的顶点作为控件的原点。

数字控件

用于在串口屏上显示整型的数据，使用前需要提前导入字库。

数字控件有两个控件事件：

按下事件

弹起事件

- **数字控件 – 属性详解**

名称	范围	备注	说明
Type	54	控件类型	用于识别当前控件的类型，不可读写。
ID		控件 ID	用于区分同一个页面中控件所处的不同层，不可读写。
Obj		控件的名称	当需要操作控件时，该属性用于定位当前页面中指定的控件，不可读写。
Sta	0~2	背景填充方式 0：切图 1：单色 2：图片	该属性决定了控件的背景填充模式，可通过上位机修改，但无法使用指令来读写。
Piccid		切图背景，当 Sta 为 0 时有效	作为控件切图背景的图片，可通过上位机修改，但无法使用指令来读写。 注：切图图片必须为全屏图片。

名称	范围	备注	说明
Style	0~1	单色显示风格，当 Sta 值为 1 时有效 0：平面 1：边框	该属性决定了控件在单色填充模式下的显示样式，可通过上位机或指令进行读写。
Borderc	0~65535	边框的颜色值，当 Style 值为 1 时有效	用于决定控件边框的颜色值，可通过上位机和指令进行读写。
Borderw	0~255	边框的粗细值，当 Style 值为 1 时有效	用于决定控件边框的粗细值，可通过上位机和指令进行读写。
Bco	0~65535	背景颜色值，当 Sta 值为 1 时有效	用于填充控件的背景颜色值，可使用上位机或指令进行读写。
Picid		背景图片，当 Sta 为 2 时有效	用于填充控件背景的图片，可通过上位机修改，但无法使用指令来读写。
Fontid		控件调用的字库	用于决定当前控件所使用的字库，不同字库显示的字体和字形也不同，可通过上位机修改，但无法使用指令来读写。
Pco	0~65535	字体颜色值	用于决定控件中显示的字体颜色，可使用上位机或指令进行读写。
Xcen	0~2	水平对齐 0：靠左 1：居中 2：靠右	用于决定控件中显示的字体的水平位置，可使用上位机或指令进行读写。
Ycen	0~2	垂直对齐 0：靠上 1：居中 2：靠下	用于决定控件中显示的字体的垂直位置，可使用上位机或指令进行读写。
Val		用于显示的数值	控件中显示的数值，可使用上位机或指令进行读写。
Lenth	0~10	显示数值的长度 0：自动显示 1~10：显示 1~10 位	用于决定控件中数值显示的长度，当该属性为 0 时，系统会自动适配数值的长度；当该属性为 1~10 时，系统会根据设置的长度进行显示，不足的位置中会自动补 0。可使用上位机或指令进行读写。
Format	0~2	数值显示的格式化类型 0：数值 1：货币 2：Hex	用于决定控件显示数值的格式，当使用货币显示格式时，数值的前面会增加一个 $ 符号；当使用 Hex 显示格式时，系统会将数值转换成十六进制后再显示。可使用上位机修改，但无法通过指令读写。
Spax	0~255	两个字符的水平间距	用于决定控件显示的字符的水平间距，可使用上位机或指令进行读写。
X		控件的原点 X 坐标	控件的 X 坐标值加上控件的宽度值不能超过显示区域的最大宽度，可使用上位机或指令进行读写。
Y		控件的原点 Y 坐标	控件的 Y 坐标值加上控件的高度值不能超过显示区域的最大高度，可使用上位机或指令进行读写。
W		控件的宽度	控件的宽度值加上 X 坐标值不能超过显示区域的最大宽度，可使用上位机或指令进行读写。
H		控件的高度	控件的高度值加上 Y 坐标值不能超过显示区域的最大高度，可使用上位机或指令进行读写。

注：数字控件以左上角的顶点作为控件的原点。

虚拟浮点数控件

用于在串口屏上显示浮点型 (float) 类型的数据，但是本质上仍然是整型 (int)，使用前需要提前导入字库。

虚拟浮点数控件有两个控件事件：

按下事件

弹起事件

- **虚拟浮点数控件 – 属性详解**

名称	范围	备注	说明
Type	59	控件类型	用于识别当前控件的类型，不可读写。
ID		控件 ID	用于区分同一个页面中控件所处的不同层，不可读写。
Obj		控件的名称	当需要操作控件时，该属性用于定位当前页面中指定的控件，不可读写。
Sta	0~2	背景填充方式 0：切图 1：单色 2：图片	该属性决定了控件的背景填充模式，可通过上位机修改，但无法使用指令来读写。
Piccid		切图背景，当 Sta 为 0 时有效	作为控件切图背景的图片，可通过上位机修改，但无法使用指令来读写。 注：切图图片必须为全屏图片。
Style	0~1	单色显示风格，当 Sta 值为 1 时有效 0：平面 1：边框	该属性决定了控件在单色填充模式下的显示样式，可通过上位机或指令进行读写。
Borderc	0~65535	边框的颜色值，当 Style 值为 1 时有效	用于决定控件边框的颜色值，可通过上位机和指令进行读写。
Borderw	0~255	边框的粗细值，当 Style 值为 1 时有效	用于决定控件边框的粗细值，可通过上位机和指令进行读写。
Bco	0~65535	背景颜色值，当 Sta 值为 1 时有效	用于填充控件的背景颜色值，可使用上位机或指令进行读写。
Picid		背景图片，当 Sta 为 2 时有效	用于填充控件背景的图片，可通过上位机修改，但无法使用指令来读写。
Fontid		控件调用的字库	用于决定当前控件所使用的字库，不同字库显示的字体和字形也不同，可通过上位机修改，但无法使用指令来读写。
Pco	0~65535	字体颜色值	用于决定控件中显示的字体颜色，可使用上位机或指令进行读写。
Xcen	0~2	水平对齐 0：靠左 1：居中 2：靠右	用于决定控件中显示的字体的水平位置，可使用上位机或指令进行读写。
Ycen	0~2	垂直对齐 0：靠上 1：居中 2：靠下	用于决定控件中显示的字体的垂直位置，可使用上位机或指令进行读写。
Val		用于显示的数值	控件中显示的数值，可使用上位机或指令进行读写。

名称	范围	备注	说明
Vvs0	0~10	整数位数 0：自动显示 1~10：显示 1~10 位	用于决定控件中整数值显示的长度，当该属性为 0 时，系统会自动适配数值的长度；当该属性为 1~10 时，系统会根据设置的长度进行显示，不足的位置中会自动补 0。可使用上位机或指令进行读写。
Vvs1	0~10	小数位数 0：不显示小数 1~8：显示 1~8 位	用于决定控件中小数显示的长度，当该属性为 0 时，控件不显示小数；当该属性为 1~8 时，系统会根据设置的长度进行显示，不足的位置中会自动补 0。可使用上位机或指令进行读写。
Spax	0~255	两个字符的水平间距	用于决定控件显示的字符的水平间距，可使用上位机或指令进行读写。
X		控件的原点 X 坐标	控件的 X 坐标值加上控件的宽度值不能超过显示区域的最大宽度，可使用上位机或指令进行读写。
Y		控件的原点 Y 坐标	控件的 Y 坐标值加上控件的高度值不能超过显示区域的最大高度，可使用上位机或指令进行读写。
W		控件的宽度	控件的宽度值加上 X 坐标值不能超过显示区域的最大宽度，可使用上位机或指令进行读写。
H		控件的高度	控件的高度值加上 Y 坐标值不能超过显示区域的最大高度，可使用上位机或指令进行读写。

注：虚拟浮点数控件以左上角的顶点作为控件的原点。

按钮控件

用于在串口屏上显示按钮，使用前需要提前导入字库。

按钮控件有两个控件事件：

按下事件

弹起事件

- **按钮控件 – 属性详解**

名称	范围	备注	说明
Type	98	控件类型	用于识别当前控件的类型，不可读写。
ID		控件 ID	用于区分同一个页面中控件所处的不同层，不可读写。
Obj		控件的名称	当需要操作控件时，该属性用于定位当前页面中指定的控件，不可读写。
Sta	0~1	背景填充方式 0：单色 1：图片	该属性决定了控件的背景填充模式，可通过上位机修改，但无法使用指令来读写。
Style	0~1	单色显示风格，当 Sta 值为 0 时有效 0：平面 1：边框	该属性决定了控件在单色填充模式下的显示样式，可通过上位机或指令进行读写。
Borderc	0~65535	边框的颜色值，当 Style 值为 1 时有效	用于决定控件边框的颜色值，可通过上位机和指令进行读写。
Borderw	0~255	边框的粗细值，当 Style 值为 1 时有效	用于决定控件边框的粗细值，可通过上位机和指令进行读写。

名称	范围	备注	说明
Bco	0~65535	背景颜色值，当 Sta 值为 0 时有效	用于填充按钮弹起状态时的背景颜色，可使用上位机或指令进行读写。
Bco2	0~65535	按下背景颜色值，当 Sta 值为 0 时有效	用于填充按钮按下状态时的背景颜色，可使用上位机或指令进行读写。
Picid		背景图片，当 Sta 为 1 时有效	用于填充按钮弹起状态时的背景图片，可通过上位机修改，但无法使用指令来读写。
Pic2id		按下背景图片，当 Sta 为 1 时有效	用于填充按钮按下状态时的背景图片，可通过上位机修改，但无法使用指令来读写。
Fontid		控件调用的字库	用于决定当前控件所使用的字库，不同字库显示的字体和字形也不同，可通过上位机修改，但无法使用指令来读写。
Pco	0~65535	字体颜色值	用于决定按钮弹起状态下显示的字体颜色，可使用上位机或指令进行读写。
Pco2	0~65535	按下字体颜色值	用于决定按钮按下状态下显示的字体颜色，可使用上位机或指令进行读写。
Xcen	0~2	水平对齐 0：靠左 1：居中 2：靠右	用于决定控件中显示的字体的水平位置，可使用上位机或指令进行读写。
Ycen	0~2	垂直对齐 0：靠上 1：居中 2：靠下	用于决定控件中显示的字体的垂直位置，可使用上位机或指令进行读写。
Isbr	0~1	是否自动换行 0：否 1：是	用于决定当控件中的字体超出显示的范围时是否自动进行换行操作，可使用上位机或指令进行读写。
Spax	0~255	两个字符的水平间距	用于决定控件显示的字符的水平间距，可使用上位机或指令进行读写。
Spay	0~255	两个字符的垂直间距	用于决定控件显示的字符的行间距，可使用上位机或指令进行读写。
X		控件的原点 X 坐标	控件的 X 坐标值加上控件的宽度值不能超过显示区域的最大宽度，可使用上位机或指令进行读写。
Y		控件的原点 Y 坐标	控件的 Y 坐标值加上控件的高度值不能超过显示区域的最大高度，可使用上位机或指令进行读写。
W		控件的宽度	控件的宽度值加上 X 坐标值不能超过显示区域的最大宽度，可使用上位机或指令进行读写。
H		控件的高度	控件的高度值加上 Y 坐标值不能超过显示区域的最大高度，可使用上位机或指令进行读写。
Txt_maxl		保存 Txt 内容的字节数	上位机自动识别输入的 Txt 字符所需的字节数，无法通过指令读写，可通过修改 Txt 间接修改。
Txt		控件显示的字符	控件中需要显示的字符，可使用上位机或指令进行读写。 注：使用指令修改时需要确保修改后的字符串长度不超过 Txt_maxl 所限定的长度。

注：按钮控件以左上角的顶点作为控件的原点。

进度条控件

用于在串口屏上显示进度。

进度条控件有两个控件事件：

按下事件

弹起事件

● 进度条控件 – 属性详解

名称	范围	备注	说明
Type	106	控件类型	用于识别当前控件的类型，不可读写。
ID		控件 ID	用于区分同一个页面中控件所处的不同层，不可读写。
Obj		控件的名称	当需要操作控件时，该属性用于定位当前页面中指定的控件，不可读写。
Sta	0~1	背景填充方式 0：单色 1：图片	该属性决定了控件的背景填充模式，可通过上位机修改，但无法使用指令来读写。
Bco	0~65535	背景颜色值，当 Sta 值为 0 时有效	用于填充进度条控件走过之前的背景颜色，可使用上位机或指令进行读写。
Pco	0~65535	前景颜色值，当 Sta 值为 0 时有效	用于填充进度条控件走过之后的背景颜色，可使用上位机或指令进行读写。
BPicid		背景图片，当 Sta 为 1 时有效	用于填充进度条控件走过之前的背景图片，可通过上位机修改，但无法使用指令来读写。
PPicid		前景图片，当 Sta 为 1 时有效	用于填充进度条控件走过之后的背景图片，可通过上位机修改，但无法使用指令来读写。
Dez	0~1	进度条前进的方向 0：横向 (从左向右) 1：纵向 (从下往上)	该属性决定了进度条前进的方向，可通过上位机修改，但无法使用指令来读写。
Val	0~100	进度值	走过的进度条占整个进度条的百分比，可使用上位机或指令进行读写。
X		控件的原点 X 坐标	控件的 X 坐标值加上控件的宽度值不能超过显示区域的最大宽度，可使用上位机或指令进行读写。
Y		控件的原点 Y 坐标	控件的 Y 坐标值加上控件的高度值不能超过显示区域的最大高度，可使用上位机或指令进行读写。
W		控件的宽度	控件的宽度值加上 X 坐标值不能超过显示区域的最大宽度，可使用上位机或指令进行读写。
H		控件的高度	控件的高度值加上 Y 坐标值不能超过显示区域的最大高度，可使用上位机或指令进行读写。

注：进度条控件以左上角的顶点作为控件的原点。

图片控件

用于在串口屏上显示图片。

图片控件有两个控件事件：

按下事件

弹起事件

- 图片控件 – 属性详解

名称	大小	备注	说明
Type	112	控件类型	用于识别当前控件的类型，不可读写。
ID		控件 ID	用于区分同一个页面中控件所处的不同层，不可读写。
Obj		控件的名称	当需要操作控件时，该属性用于定位当前页面中指定的控件，不可读写。
Picid		显示的图片	图片控件需要显示的图片，可通过上位机修改，但无法使用指令来读写。
X		控件的原点 X 坐标	控件的 X 坐标值加上控件的宽度值不能超过显示区域的最大宽度，可使用上位机或指令进行读写。
Y		控件的原点 Y 坐标	控件的 Y 坐标值加上控件的高度值不能超过显示区域的最大高度，可使用上位机或指令进行读写。
W		控件的宽度	控件的宽度值加上 X 坐标值不能超过显示区域的最大宽度，可使用上位机或指令进行读写。
H		控件的高度	控件的高度值加上 Y 坐标值不能超过显示区域的最大高度，可使用上位机或指令进行读写。

注：图片控件以左上角的顶点作为控件的原点。

切图控件

用于在串口屏上显示切图，切图控件的作用是将全屏图片的一部分抠出来显示，请注意切图控件 Piccid 属性对应的一定要是全屏图片。

切图控件有两个控件事件：

按下事件

弹起事件

- 切图控件 – 属性详解

名称	大小	备注	说明
Type	113	控件类型	用于识别当前控件的类型，不可读写。
ID		控件 ID	用于区分同一个页面中控件所处的不同层，不可读写。
Obj		控件的名称	当需要操作控件时，该属性用于定位当前页面中指定的控件，不可读写。
Piccid		切图图片	作为控件切图背景的图片，可通过上位机修改，但无法使用指令来读写。 注：切图图片必须为全屏图片。
X		控件的原点 X 坐标	控件的 X 坐标值加上控件的宽度值不能超过显示区域的最大宽度，可使用上位机或指令进行读写。

名称	大小	备注	说明
Y		控件的原点 Y 坐标	控件的 Y 坐标值加上控件的高度值不能超过显示区域的最大高度，可使用上位机或指令进行读写。
W		控件的宽度	控件的宽度值加上 X 坐标值不能超过显示区域的最大宽度，可使用上位机或指令进行读写。
H		控件的高度	控件的高度值加上 Y 坐标值不能超过显示区域的最大高度，可使用上位机或指令进行读写。

注：切图控件以左上角的顶点作为控件的原点。

触控热区控件

触控热区控件可以理解为一个看不见的按钮控件。

触控热区控件有两个控件事件：

按下事件

弹起事件

- **触控热区控件 – 属性详解**

名称	大小	备注	说明
Type	109	控件类型	用于识别当前控件的类型，不可读写。
ID		控件 ID	用于区分同一个页面中控件所处的不同层，不可读写。
Obj		控件的名称	当需要操作控件时，该属性用于定位当前页面中指定的控件，不可读写。
X		控件的原点 X 坐标	控件的 X 坐标值加上控件的宽度值不能超过显示区域的最大宽度，可使用上位机或指令进行读写。
Y		控件的原点 Y 坐标	控件的 Y 坐标值加上控件的高度值不能超过显示区域的最大高度，可使用上位机或指令进行读写。
W		控件的宽度	控件的宽度值加上 X 坐标值不能超过显示区域的最大宽度，可使用上位机或指令进行读写。
H		控件的高度	控件的高度值加上 Y 坐标值不能超过显示区域的最大高度，可使用上位机或指令进行读写。

注：触控热区控件以左上角的顶点作为控件的原点。

滑块控件

用于在串口屏上滑动调整数值，例如亮度，音量。

滑块控件有三个控件事件：

按下事件

弹起事件

滑动事件

● 滑块控件 – 属性详解

名称	范围	备注	说明
Type	109	控件类型	用于识别当前控件的类型，不可读写。
ID		控件 ID	用于区分同一个页面中控件所处的不同层，不可读写。
Obj		控件的名称	当需要操作控件时，该属性用于定位当前页面中指定的控件，不可读写。
Mode		滑行方向 0：横向 1：纵向	该属性决定了控件是该横向滑行或者是纵向滑行，可通过上位机修改，但无法使用指令来读写。
Sta	0~1	背景填充方式 0：单色 1：图片	该属性决定了控件的背景填充模式，可通过上位机修改，但无法使用指令来读写。
Bco	0~65535	前景颜色值，当 Sta 值为 0 时有效	用于填充滑块控件滑过之后的背景颜色，可使用上位机或指令进行读写。
Bco1	0~65535	背景颜色值，当 Sta 值为 0 时有效	用于填充滑块控件滑过之前的背景颜色，可使用上位机或指令进行读写。
Picid		背景图片，当 Sta 为 1 时有效	用于填充滑块控件滑过之前的背景图片，可通过上位机修改，但无法使用指令来读写。
Pic1id		前景图片，当 Sta 为 1 时有效	用于填充滑块控件滑过之后的背景图片，可通过上位机修改，但无法使用指令来读写。
Psta	0	游标的填充方式 0：单色	该属性决定了滑块控件的游标的填充方式，不可读写。
Pco	0~65535	游标的背景色	该属性决定了滑块控件的游标的背景填充颜色，可通过上位机修改，但无法使用指令来读写。
Wid	0~255	游标的宽度 255：自动 0：无游标	该属性决定了游标的宽度，当该值为 255 时，游标的宽度值等于滑块控件的最小边长，可使用上位机或指令进行读写。
Hig	0~255	游标的高度 255：自动 0：无游标	该属性决定了游标的高度，当该值为 255 时，游标的高度值等于滑块控件的最小边长，可使用上位机或指令进行读写。
Val	Minval~ Maxval	当前值	该属性决定了当前游标所在的位置，也会根据游标的移动而变动，可使用上位机或指令进行读写。
Maxval	Minval~ 65535	最大值	该属性决定了滑块控件的最大值，可使用上位机或指令进行读写。
Minval	0~ Maxval	最小值	该属性决定了滑块控件的最小值，可使用上位机或指令进行读写。

名称	范围	备注	说明
X		控件的原点 X 坐标	控件的 X 坐标值加上控件的宽度值不能超过显示区域的最大宽度，可使用上位机或指令进行读写。
Y		控件的原点 Y 坐标	控件的 Y 坐标值加上控件的高度值不能超过显示区域的最大高度，可使用上位机或指令进行读写。
W		控件的宽度	控件的宽度值加上 X 坐标值不能超过显示区域的最大宽度，可使用上位机或指令进行读写。
H		控件的高度	控件的高度值加上 Y 坐标值不能超过显示区域的最大高度，可使用上位机或指令进行读写。

注：滑块控件以左上角的顶点作为控件的原点。

定时器控件

定时器控件用于定时执行某些代码，或者延时执行某些代码，当定时被使能时，定时器里的代码会定时执行。

定时器控件有一个控件事件：

定时事件

- **定时器控件 – 属性详解**

名称	范围	备注	说明
Type	51	控件类型	用于识别当前控件的类型，不可读写。
ID		控件 ID	用于区分同一个页面中控件所处的不同层，不可读写。
Obj		控件的名称	当需要操作控件时，该属性用于定位当前页面中指定的控件，不可读写。
Tim	50~65534	定时时间 单位：ms	该属性决定了多长时间执行一次定时器的定时事件，可使用上位机或指令进行读写。
En	0~1	使能开关 0：关闭 1：开启	该属性决定了是否要开启定时器进行计时并执行定时器事件，可使用上位机或指令进行读写。

变量控件

变量控件位于特殊控件栏上。

- **变量控件 – 属性详解**

名称	大小	备注	说明
Type	52	控件类型	用于识别当前控件的类型，不可读写。
ID		控件 ID	用于区分同一个页面中控件所处的不同层，不可读写。
Obj		控件的名称	当需要操作控件时，该属性用于定位当前页面中指定的控件，不可读写。
Sta	0~1	数据类型 0：数值 1：字符串	该属性决定了变量控件中存储的是数值变量还是字符串变量，可通过上位机修改，但无法使用指令来读写。
Val		变量控件存储的数值，当 Sta 为 0 时有效	该属性用于存放变量控件的数值，可使用上位机或指令进行读写。
Fontid		控件调用的字库，当 Sta 为 1 时有效	用于决定当前控件所使用的字库，不同字库显示的字体和字形也不同，可通过上位机修改，但无法使用指令来读写。

名称	大小	备注	说明
Txt_maxl		保存 Txt 内容的字节数，当 Sta 为 1 时有效	上位机自动识别输入的 Txt 字符所需的字节数，无法通过指令读写，可通过修改 Txt 间接修改。
Txt		变量控件存储的字符串，当 Sta 为 1 时有效	该属性用于存放变量控件的字符串，可使用上位机或指令进行读写。 注：使用指令修改时需要确保修改后的字符串长度不超过 Txt_maxl 所限定的长度。

双态按钮控件

用于在串口屏上显示按钮，此按钮类似于自锁开关，使用前需要提前导入字库。

双态按钮控件有两个控件事件：

按下事件

弹起事件

- **双态按钮控件 – 属性详解**

名称	范围	备注	说明
Type	53	控件类型	用于识别当前控件的类型，不可读写。
ID		控件 ID	用于区分同一个页面中控件所处的不同层，不可读写。
Obj		控件的名称	当需要操作控件时，该属性用于定位当前页面中指定的控件，不可读写。
Sta	0~1	背景填充方式 0：单色 1：图片	该属性决定了控件的背景填充模式，可通过上位机修改，但无法使用指令来读写。
Style	0~1	单色显示风格，当 Sta 值为 0 时有效 0：平面 1：边框	该属性决定了控件在单色填充模式下的显示样式，可通过上位机或指令进行读写。
Borderc	0~65535	边框的颜色值，当 Style 值为 1 时有效	用于决定控件边框的颜色值，可通过上位机和指令进行读写。
Borderw	0~255	边框的粗细值，当 Style 值为 1 时有效	用于决定控件边框的粗细值，可通过上位机和指令进行读写。
Bco	0~65535	背景颜色值，当 Sta 值为 0 时有效	用于填充双态按钮未按下状态时的背景颜色，可使用上位机或指令进行读写。
Bco2	0~65535	按下背景颜色值，当 Sta 值为 0 时有效	用于填充双态按钮按下状态时的背景颜色，可使用上位机或指令进行读写。
Picid		背景图片，当 Sta 为 1 时有效	用于填充双态按钮未按下状态时的背景图片，可通过上位机修改，但无法使用指令来读写。
Pic2id		按下背景图片，当 Sta 为 1 时有效	用于填充双态按钮按下状态时的背景图片，可通过上位机修改，但无法使用指令来读写。
Fontid		控件调用的字库	用于决定当前控件所使用的字库，不同字库显示的字体和字形也不同，可通过上位机修改，但无法使用指令来读写。
Pco	0~65535	字体颜色值	用于决定双态按钮未按下状态下显示的字体颜色，可使用上位机或指令进行读写。
Pco2	0~65535	按下字体颜色值	用于决定双态按钮按下状态下显示的字体颜色，可使用上位机或指令进行读写。

名称	范围	备注	说明
Xcen	0~2	水平对齐 0：靠左 1：居中 2：靠右	用于决定控件中显示的字体的水平位置，可使用上位机或指令进行读写。
Ycen	0~2	垂直对齐 0：靠上 1：居中 2：靠下	用于决定控件中显示的字体的垂直位置，可使用上位机或指令进行读写。
Val	0~1	双态按钮的当前状态 0：未按下 1：按下	该属性决定了双态按钮当前的状态是按下或者未按下，可使用上位机或指令进行读写。
Isbr	0~1	是否自动换行 0：否 1：是	用于决定当控件中的字体超出显示的范围时是否自动进行换行操作，可使用上位机或指令进行读写。
Spax	0~255	两个字符的水平间距	用于决定控件显示的字符的水平间距，可使用上位机或指令进行读写。
Spay	0~255	两个字符的垂直间距	用于决定控件显示的字符的行间距，可使用上位机或指令进行读写。
X		控件的原点 X 坐标	控件的 X 坐标值加上控件的宽度值不能超过显示区域的最大宽度，可使用上位机或指令进行读写。
Y		控件的原点 Y 坐标	控件的 Y 坐标值加上控件的高度值不能超过显示区域的最大高度，可使用上位机或指令进行读写。
W		控件的宽度	控件的宽度值加上 X 坐标值不能超过显示区域的最大宽度，可使用上位机或指令进行读写。
H		控件的高度	控件的高度值加上 Y 坐标值不能超过显示区域的最大高度，可使用上位机或指令进行读写。
Txt_maxl		保存 Txt 内容的字节数	上位机自动识别输入的 Txt 字符所需的字节数，无法通过指令读写，可通过修改 Txt 间接修改。
Txt		控件显示的字符	控件中需要显示的字符，可使用上位机或指令进行读写。 注：使用指令修改时需要确保修改后的字符串长度不超过 Txt_maxl 所限定的长度。

注：双态按钮控件以左上角的顶点作为控件的原点。

单选框控件

用于在串口屏上显示单选框。

单选框控件有两个控件事件：

按下事件

弹起事件

- **单选框控件 – 属性详解**

名称	范围	备注	说明
Type	57	控件类型	用于识别当前控件的类型，不可读写。
ID		控件 ID	用于区分同一个页面中控件所处的不同层，不可读写。
Obj		控件的名称	当需要操作控件时，该属性用于定位当前页面中指定的控件，不可读写。
Bco	0~65535	边框颜色值	该属性决定了单选框的边框颜色，可使用上位机或指令进行读写。
Pco	0~65535	前景颜色值，选中时的颜色	该属性决定了单选框选中时选中点的颜色，可使用上位机或指令进行读写。
Val	0~1	当前状态 0：未选中 1：选中	该属性决定了当前单选框控件是否被选中，属性值会随着选中和取消选中而被修改，可使用上位机或指令进行读写。
X		控件的原点 X 坐标	控件的 X 坐标值加上单选框控件的半径不能超过显示区域的最大宽度，可使用上位机或指令进行读写。 注：单选框以控件的中心为原点。
Y		控件的原点 Y 坐标	控件的 Y 坐标值加上单选框控件的半径不能超过显示区域的最大高度，可使用上位机或指令进行读写。 注：单选框以控件的中心为原点。
R	5~20	控件的半径	该属性决定了单选框控件的半径大小，可使用上位机或指令进行读写。
Group	0~10	控件的组名	该属性决定了当前单选框所属的组，同一组的单选框只能有一个被选中，如果需要选择组内的其他单选框需要先将当前选中的单选框取消再去选择其他单选框，使用上位机或指令进行读写。

注：单选框控件以控件中心作为控件的原点。

复选框控件

用于在串口屏上显示复选框。

复选框控件有两个控件事件：

按下事件

弹起事件

- **复选框控件 – 属性详解**

名称	大小	备注	说明
Type	56	控件类型	用于识别当前控件的类型，不可读写。
ID		控件 ID	用于区分同一个页面中控件所处的不同层，不可读写。
Obj		控件的名称	当需要操作控件时，该属性用于定位当前页面中指定的控件，不可读写。
Style	0~1	显示风格 0：平面 1：边框	该属性决定了复选框控件的显示风格，可通过上位机或指令进行读写。
Borderc	0~65535	边框的颜色值，当 Style 的值为 1 时有效	用于决定复选框控件边框的颜色值，可通过上位机和指令进行读写。
Borderw	0~255	边框的粗细值，当 Style 值为 1 时有效	用于决定控件边框的粗细值，可通过上位机和指令进行读写。
Bco	0~65535	背景颜色值	该属性决定了复选框的背景颜色值，可使用上位机或指令进行读写。
Pco	0~65535	前景颜色值，选中时的颜色	该属性决定了复选框选中时选中点的颜色，可使用上位机或指令进行读写。
Val	0~1	当前状态 0：未选中 1：选中	该属性决定了当前复选框控件是否被选中，属性值会随着选中和取消选中而被修改，可使用上位机或指令进行读写。
X		控件的原点 X 坐标	控件的 X 坐标值加上控件的宽度值不能超过显示区域的最大宽度，可使用上位机或指令进行读写。
Y		控件的原点 Y 坐标	控件的 Y 坐标值加上控件的高度值不能超过显示区域的最大高度，可使用上位机或指令进行读写。
W		控件的宽度	控件的宽度值加上 X 坐标值不能超过显示区域的最大宽度，可使用上位机或指令进行读写。
H		控件的高度	控件的高度值加上 Y 坐标值不能超过显示区域的最大高度，可使用上位机或指令进行读写。

注：复选框控件以左上角的顶点作为控件的原点。

指令使用说明

注：所有指令都使用小写字符。

dp 指令

说明：dp 在系统中是用于保存当前页面名称的一个变量，可通过定义好的指令格式对该变量进行读取操作。

读取指令格式：

指令
dp

写入指令格式：

该指令权限为只读，不可写入。

指令参数：

参数	说明
无	该指令不需要参数

指令的返回值：

该指令返回的是当前页面的名称。

例子：

读取当前的页面名称：dp

dim 指令

说明：dim 在系统中是用于保存串口屏当前屏幕背光状态的一个变量，可通过定义好的指令格式对该变量进行读写操作。通过该指令修改过后的值会被暂存于 RAM 中，当系统掉电后，该值会消失。

读取指令格式：

指令
dim

写入指令格式：

指令	间隔符	参数
dim	空格	1

指令参数：

参数	说明
0	关闭串口屏的屏幕背光
1	开启串口屏的屏幕背光

指令的返回值：

该指令返回的是当前屏幕背光的状态 (0/1)。

例子：

读取 dim 的值：dim

写入 dim 值：dim 1

dims 指令

说明：dims 在系统中是用于保存串口屏屏幕背光初始化状态的一个变量，可通过定义好的指令格式对该变量进行读写操作。修改过后的值会被保存到 Flash 中，等下次开机时系统会以该值来初始化串口屏的屏幕背光状态。

读取指令格式：

指令
dims

写入指令格式：

指令	间隔符	参数
dims	空格	1

指令参数：

参数	说明
0	关闭串口屏的屏幕背光
1	开启串口屏的屏幕背光

指令的返回值：

该指令返回的是默认屏幕背光的状态 (0/1)。

例子：

读取 dims 的值：dims

写入 dims 值：dims 1

baud 指令

说明：baud 在系统中是用于保存串口屏串口波特率的一个变量，可通过定义好的指令格式对该变量进行读写操作。通过该指令修改过后的值会被暂存于 RAM 中，当系统掉电后，该值会消失。

注：在使用 baud 更改系统波特率的过程中，系统会暂时关闭串口并修改波特率，该过程大概需要 10ms，请在此期间不要使用串口跟串口屏进行通信，避免发送的信息无法被串口屏接收到。

读取指令格式：

指令
baud

写入指令格式：

指令	间隔符	参数
baud	空格	115200

指令参数：

参数	说明
2400	将串口波特率设置为 2400
4800	将串口波特率设置为 4800
9600	将串口波特率设置为 9600
19200	将串口波特率设置为 19200
38400	将串口波特率设置为 38400

参数	说明
57600	将串口波特率设置为 57600
115200	将串口波特率设置为 115200
230400	将串口波特率设置为 230400
256000	将串口波特率设置为 256000
512000	将串口波特率设置为 512000
921600	将串口波特率设置为 921600

注：以上的参数为串口波特率的参考值，也可以设置为其他参数，为避免设置的串口波特率影响到串口的正常通信，建议以参考的串口波特率值为准进行设置。

指令的返回值：

该指令返回的是串口屏当前的波特率值。

例子：

读取 baud 的值：baud

写入 baud 值：baud 115200

bauds 指令

说明：bauds 在系统中是用于保存串口屏默认串口波特率的一个变量，可通过定义好的指令格式对该变量进行读写操作。修改过后的值会被保存到 Flash 中，等下次开机时系统会以该值来初始化串口屏的串口波特率。

读取指令格式：

指令
bauds

写入指令格式：

指令	间隔符	参数
bauds	空格	115200

指令参数：

参数	说明
2400	将串口波特率设置为 2400
4800	将串口波特率设置为 4800
9600	将串口波特率设置为 9600
19200	将串口波特率设置为 19200
38400	将串口波特率设置为 38400
57600	将串口波特率设置为 57600
115200	将串口波特率设置为 115200
230400	将串口波特率设置为 230400
256000	将串口波特率设置为 256000
512000	将串口波特率设置为 512000
921600	将串口波特率设置为 921600

注：以上的参数为串口波特率的参考值，也可以设置为其他参数，为避免设置的串口波特率影响到串口的正常通信，建议以参考的串口波特率值为准进行设置。

指令的返回值：

该指令返回的是串口屏默认的波特率值。

例子：

读取 bauds 的值：bauds

写入 bauds 值：bauds 115200

ussp 指令

说明：ussp 在系统中是用于保存无串口数据的休眠时间的一个变量 (该变量的单位是秒)，可通过定义好的指令格式对该变量进行读写操作。当无串口数据休眠功能开启时，系统会根据 ussp 变量中记录的时间来进行倒计时，在倒计时的这个过程中，如果系统没有接收到新的串口数据，系统会进入休眠状态。串口屏还有一个无触控休眠的功能，该功能的休眠时间由 thsp 变量指定。当这两个功能同时开启时，系统会从 ussp 和 thsp 中选取最小的值作为系统进入休眠状态的倒计时间。通过该指令修改过后的值会被暂存于 RAM 中，当系统掉电后，该值会消失。

注：在进入休眠模式之前请根据 lowpower 的值对系统的唤醒方式进行设置，否则串口屏无法进入休眠模式。普通休眠可以通过串口唤醒也可以通过触控唤醒，深度休眠只能通过触控唤醒。

读取指令格式：

指令
ussp

写入指令格式：

指令	间隔符	参数
ussp	空格	500

指令参数：

参数	说明
0~2	关闭无串口数据休眠功能
3~65535	开启无串口数据休眠功能，并设定无串口数据休眠时间为输入的参数值

指令的返回值：

该指令返回的是无串口数据休眠的时间。

例子：

读取 ussp 的值：ussp

写入 ussp 值：ussp 1000

thsp 指令

说明：thsp 在系统中是用于保存无触控休眠时间的一个变量 (该变量的单位是秒)，可通过定义好的指令格式对该变量进行读写操作。当无触控休眠功能开启时，系统会根据 thsp 变量中记录的时间来进行倒计时，在倒计时结束的这个过程中，如果系统没有接收到触控操作的信号，系统会进入休眠状态。串口屏还有一个无串口数据休眠功能，该功能的休眠时间由 ussp 变量指定。当这两个功能同时开启时，系统会从 ussp 和 thsp 中选取最小的值作为系统进入休眠状态的倒计时间。通过该指令修改过后的值会被暂存于 RAM 中，当系统掉电后，该值会消失。

注：在进入休眠模式之前请根据 lowpower 的值对系统的唤醒方式进行设置，否则串口屏无法进入休眠模式。普通休眠可以通过串口唤醒也可以通过触控唤醒，深度休眠只能通过触控唤醒。

读取指令格式：

指令
thsp

写入指令格式：

指令	间隔符	参数
thsp	空格	500

指令参数：

参数	说明
0~2	关闭无触控休眠功能
3~65535	开启无触控休眠功能，并设定无触控休眠时间为输入的参数值

指令的返回值：

该指令返回的是无触控休眠的时间。

例子：

读取 thsp 的值：thsp

写入 thsp 值：thsp 1000

thup 指令

说明：thup 在系统中是用于保存休眠模式下触控自动唤醒功能是否开启的一个变量，可通过定义好的指令格式对该变量进行写操作。通过该指令修改过后的值会被暂存于 RAM 中，当系统掉电后，该值会消失。

注：关于休眠模式下触控自动唤醒功能，该功能可以唤醒处于普通休眠或深度休眠模式下的系统。

读取指令格式：

该指令权限为写，不可读取。

写入指令格式：

指令	间隔符	参数
thup	空格	0

指令参数：

参数	说明
0	关闭休眠模式下触控自动唤醒功能
1	开启休眠模式下触控自动唤醒功能

指令的返回值：

无

例子：

写入 thup 值：thup 1

usup 指令

说明：usup 在系统中是用于保存休眠模式下串口数据自动唤醒功能是否开启的一个变量，可通过定义好的指令格式对该变量进行写操作。通过该指令修改过后的值会被暂存于 RAM 中，当系统掉电后，该值会消失。

注：关于休眠模式下串口数据自动唤醒功能，该功能只能唤醒处于普通休眠的系统。

读取指令格式：

该指令权限为写，不可读取。

写入指令格式：

指令	间隔符	参数
usup	空格	0

指令参数：

参数	说明
0	关闭休眠模式下串口数据自动唤醒功能
1	开启休眠模式下串口数据自动唤醒功能

指令的返回值：

无

例子：

写入 usup 值：usup 1

sleep 指令

说明：sleep 是系统用于进入休眠模式的一条指令，当系统接收到该指令后，系统会自动进入休眠模式。

注：在进入休眠模式之前请根据 lowpower 的值对系统的唤醒方式进行设置，否则串口屏无法进入休眠模式。普通休眠可以通过串口唤醒也可以通过触控唤醒，深度休眠只能通过触控唤醒。

指令格式：

指令
sleep

指令参数：

参数	说明
无	该指令不需要参数

指令的返回值：

无

例子：

系统进入休眠模式：sleep

lowpower 指令

说明：lowpower 在系统中是用于保存系统进入普通休眠或深度休眠标志的一个变量，可通过定义好的指令格式对该变量进行写操作。通过该指令修改过后的值会被暂存于 RAM 中，当系统掉电后，该值会消失。

注：lowpower 仅在系统正常运行的状态下才能进行修改。普通休眠可以通过串口唤醒也可以通过触控唤醒，深度休眠只能通过触控唤醒。

读取指令格式：

该指令权限为写，不可读取。

写入指令格式：

指令	间隔符	参数
lowpower	空格	0

指令参数：

参数	说明
0	进入普通休眠模式
1	进入深度休眠模式

指令的返回值：

无

例子：

进入普通休眠模式：lowpower 0

进入深度休眠模式：lowpower 1

bkcmd 指令

说明：bkcmd 在系统中是用于保存串口指令执行结果反馈的一个变量，可通过定义好的指令格式对该变量进行写操作。系统在执行所有指令时会根据 bkcmd 的参数，判断是否返回结果和返回的结果类型。通过该指令修改过后的值会被暂存于 RAM 中，当系统掉电后，该值会消失。

注：返回的结果编码可参考错误码章节中的指令执行结果编码及说明。

读取指令格式：

该指令权限为写，不可读取。

写入指令格式：

指令	间隔符	参数
bkcmd	空格	0

指令参数：

参数	说明
0	不返回结果
1	只返回成功结果
2	只返回失败结果
3	成功或失败都返回结果

指令的返回值：

无

例子：

不返回指令执行结果：bkcmd 0

只返回指令执行成功结果：bkcmd 1

只返回指令执行失败结果：bkcmd 2

sendxy 指令

说明：sendxy 在系统中是用于保存是否实时发送触控坐标的一个变量，可通过定义好的指令格式对该变量进行写操作。开启了实时发送触控坐标功能后，每次触控产生的坐标都会通过串口发送出去。通过该指令修改过后的值会被暂存于 RAM 中，当系统掉电后，该值会消失。

读取指令格式：

该指令权限为写，不可读取。

写入指令格式：

指令	间隔符	参数
sendxy	空格	0

指令参数：

参数	说明
0	关闭实时发送触控坐标功能
1	开启实时发送触控坐标功能

指令的返回值：

无

实时坐标返回格式：

x,y

例子：

关闭实时发送触控坐标功能：sendxy 0

开启实时发送触控坐标功能：sendxy 1

rand 指令

说明：rand 在系统中是用于保存获取到的随机数的一个变量，可通过定义好的指令格式对该变量进行读操作。当系统获取到该指令后会在 randmin 和 randmax 之间获取一个随机值，并通过串口返回获取到的值。该变量的信息会被暂存于 RAM 中，当系统掉电后，该值会消失。

注：在使用 rand 指令之前请先使用 randmin 和 randmax 指令对随机数的范围进行设置，如不进行随机数的范围设置，则随机数的范围以上次设置的范围为准。

读取指令格式：

指令
rand

写入指令格式：

该指令权限为读，不可写入。

指令参数：

参数	说明
无	该指令不需要参数

指令的返回值：

返回的是获取到的随机数值。

例子：

获取随机数值：rand

randmin 指令

说明：randmin 在系统中是用于保存随机数范围的最小值的一个变量，可通过定义好的指令格式对该变量的值进行设置。当执行 rand 指令时，系统会在 randmin 和 randmax 之前获取一个随机数并通过串口发送。通过该指令修改过后的值会被暂存于 RAM 中，当系统掉电后，该值会消失。

注：设置的 randmin 值必须要小于 randmax 的值，否则会设置失败。

读取指令格式：

该指令权限为写，不可读取。

写入指令格式：

指令	间隔符	参数
randmin	空格	50

指令参数：

参数	说明
0~65534	在参数范围内的数值都可以设置

注：randmin 能设置成功的前提不仅要符合 randmin 的基本参数要求，且设置的值必须小于 randmax 的值。

指令的返回值：

无

例子：

将 randmin 设置为 30：randmin 30

randmax 指令

说明：randmax 在系统中是用于保存随机数范围的最大值的一个变量，可通过定义好的指令格式对该变量的值进行设置。当执行 rand 指令时，系统会在 randmin 和 randmax 之前获取一个随机数并通过串口发送。通过该指令修改过后的值会被暂存于 RAM 中，当系统掉电后，该值会消失。

注：设置的 randmax 值必须要大于 randmin 的值，否则会设置失败。

读取指令格式：

该指令权限为写，不可读取。

写入指令格式：

指令	间隔符	参数
randmax	空格	50000

指令参数：

参数	说明
1~65535	在参数范围内的数值都可以设置

注：randmax 能设置成功的前提不仅要符合 randmax 的基本参数要求，且设置的值必须大于 randmin 的值。

指令的返回值：

无

例子：

将 randmax 设置为 1000：randmax 1000

tch0 指令

说明：tch0 在系统中是用于保存当前触控坐标 X 的值的一个变量，可通过定义好的指令格式对该变量进行读操作。每次触控产生新的 X 坐标都会被系统更新到 tch0 这个变量中。该变量的信息会被暂存于 RAM 中，当系统掉电后，该值会消失。

读取指令格式：

指令
tch0

写入指令格式：

该指令权限为读，不可写入。

指令参数：

参数	说明
无	该指令不需要参数

指令的返回值：

返回的是当前触控获取到的 X 轴的值。

例子：

获取当前触控坐标 X 的值：tch0

tch1 指令

说明：tch1 在系统中是用于保存当前触控坐标 Y 的值的一个变量，可通过定义好的指令格式对该变量进行读操作。每次触控产生新的 Y 坐标都会被系统更新到 tch1 这个变量中。该变量的信息会被暂存于 RAM 中，当系统掉电后，该值会消失。

读取指令格式：

指令
tch1

写入指令格式：

该指令权限为读，不可写入。

指令参数：

参数	说明
无	该指令不需要参数

指令的返回值：

返回的是当前触控获取到的 Y 轴的值。

例子：

获取当前触控坐标 Y 的值：tch1

tch2 指令

说明：tch2 在系统中是用于保存上次触控坐标 X 的值的一个变量，可通过定义好的指令格式对该变量进行读操作。每次触控产生新的 X 坐标后，系统都会将旧的 X 坐标保存到 tch2 中。该变量的信息会被暂存于 RAM 中，当系统掉电后，该值会消失。

注：tch2 只保留上一次的 X 坐标记录，当触控产生新的坐标后，原本保存在 tch0 中的 X 坐标就会被覆盖到 tch2 中。

读取指令格式：

指令
tch2

写入指令格式：

该指令权限为读，不可写入。

指令参数：

参数	说明
无	该指令不需要参数

指令的返回值：

返回的是上一次触控获取到的 X 轴的值。

例子：

获取上次触控坐标 X 的值：tch2

tch3 指令

说明：tch3 在系统中是用于保存上次触控坐标 Y 的值的一个变量，可通过定义好的指令格式对该变量进行读操作。每次触控产生新的 Y 坐标后，系统都会将旧的 Y 坐标保存到 tch3 中。该变量的信息会被暂存于 RAM 中，当系统掉电后，该值会消失。

注：tch3 只保留上一次的 Y 坐标记录，当触控产生新的坐标后，原本保存在 tch1 中的 Y 坐标就会被覆盖到 tch3 中。

读取指令格式：

指令
tch3

写入指令格式：

该指令权限为读，不可写入。

指令参数：

参数	说明
无	该指令不需要参数

指令的返回值：

返回的是上一次触控获取到的 Y 轴的值。

例子：

获取上次触控坐标 Y 的值：tch3

addr 指令

说明：addr 在系统中是用于保存设备地址的一个变量，可通过定义好的指令格式对该变量进行读写操作。用户可以通过设置不同的 addr 值，从而向指定的 addr 值的串口屏发送指令。使用 addr 指令修改过的 addr 值会被保存到 Flash 中，等下次开机时系统会以该值作为设备的地址值。

注：系统中有两个地址是比较特殊的，第一种 0x0000 作为无效地址使用。第二种 0xFFFF 作为广播地址使用，即在总线上的设备都能接收到指令。这两个地址都不能设置为 addr 的值。

读取指令格式：

指令
addr

写入指令格式：

指令	间隔符	参数
addr	空格	41393

指令参数：

参数	说明
1~65534	可作为设备地址的参数

注：当以上的数值作为 addr 指令的参数时，需要以十进制的形式作为指令的参数。

指令的返回值：

返回的是设备的地址信息。

例子：

获取设备的地址：addr

将设备的地址设置成 0xA1B1：addr 41393

date 指令

说明：date 指令可用于设置或者获取系统日期，需要通过定义好的指令格式和处于特定的情况下才能对系统的日期进行获取或修改。系统内部有一个关于系统日期和时间状态的标志位 (rtc_flag)。当该状态位为 0 时，可正常通过 date 指令获取到系统的日期，但无法通过 date 指令修改系统的日期。当该状态位为 1 时，可通过 date 指令修改系统的临时日期缓存区。当 rtc_flag 的由 1 变为 0 时，系统会将临时日期缓存区中的日期写入到系统的 RTC 模块中。

注：当 rtc_flag 的值为 1 时，系统暂停获取日期，使用 date 指令读取到的值为暂停时的日期，如有对日期进行修改，则读取到的是修改后的日期。

读取指令格式：

指令
date

写入指令格式：

指令	间隔符	参数
date	空格	2023/02/03

指令参数：

年的参数以及范围：

参数	说明
1970~2099	可作为年的参数范围

月的参数以及范围：

参数	说明
1~12	可作为月的参数范围

日的参数以及范围：

参数	说明
1~31	可作为日的参数范围

注：

1. 参数的格式需要按照 (年 / 月 / 日) 来书写。
2. 有一些月份没有 30 号或 31 号的，系统会根据多出的天数对日期进行延长。如 2023 年的 4 月没有 31 号，使用指令设置日期为 2023/04/31 的话，在写入系统的 RTC 模块后，模块自动将 31 号这天计算到下一个月。即使用 2023/04/31 设置的时间实际为 2023/05/01。

指令的返回值：

返回的是芯片内部的 RTC 模块的日期，格式为 (年 / 月 / 日)。

例子：

获取当前日期：date

设置日期为 2023/05/10：

rtc_flag 1

date 2023/05/10

rtc_flag 0

time 指令

说明：time 指令可用于设置或者获取系统时间，需要通过定义好的指令格式和处于特定的情况下才能对系统的时间进行获取或修改。系统内部有一个关于系统日期和时间状态的标志位 (rtc_flag)。当该状态位为 0 时，可正常通过 time 指令获取到系统的时间，但无法通过 time 指令修改系统的时间。当该状态位为 1 时，可通过 time 指令修改系统的临时时间缓存区。当 rtc_flag 的由 1 变为 0 时，系统会将临时时间缓存区中的时间写入到系统的 RTC 模块中。

注：当 rtc_flag 的值为 1 时，系统暂停获取时间，使用 time 指令读取到的值为暂停时的时间，如有对时间进行修改，则读取到的是修改后的时间。

读取指令格式：

指令
time

写入指令格式：

指令	间隔符	参数
time	空格	22:20:50

指令参数：

时的参数以及范围：

参数	说明
0~23	可作为时的参数范围

分的参数以及范围：

参数	说明
0~59	可作为分的参数范围

秒的参数以及范围：

参数	说明
0~59	可作为秒的参数范围

注：参数的格式需要按照 (时 : 分 : 秒) 来书写。

指令的返回值：

返回的是芯片内部的 RTC 模块的时间，格式为 (时 : 分 : 秒)。

例子：

获取当前时间：time

设置时间为 20:50:30：

rtc_flag 1

time 20:50:30

rtc_flag 0

week 指令

说明：week 指令可用于获取当前日期具体为周几，通过定义好的指令格式可直接读取当前日期具体为周几。该变量仅在 rtc_flag 为 0 时才能保证读取到的周几为正确的。当 rtc_flag 为 1 时，系统停止从 RTC 模块中获取最新的时间，week 也不会跟随 date 的设置而改变。

注：当 rtc_flag 的值为 1 时，系统暂停获取时间，使用 week 指令读取到的值为暂停时的周几。

读取指令格式：

指令
week

写入指令格式：

无，该指令为仅读指令。

指令参数：

无，该指令不需要参数。

指令的返回值：

返回的是芯片内部的 RTC 模块计算出来的周几，格式 1~7。

例子：

获取当前的周几：week

rtc_flag 指令

说明：rtc_flag 在系统中是用于保存系统日期和时间状态的标志位，可通过定义好的指令格式对该标志位的值进行修改。当该标志位的值为 0 时，可通过 date 和 time 指令对系统的日期和时间进行读取。当该标志位的值为 1 时，可通过 date 和 time 指令将需要修改的日期和时间写入到系统的临时缓存空间中，再将 rtc_flag 的值改为 0，即可将设置的日期和时间写入到系统的 RTC 模块中。

读取指令格式：

该指令权限为写，不可读取。

写入指令格式：

指令	间隔符	参数
rtc_flag	空格	1

指令参数：

参数	说明
0	RTC 模块进入获取日期和时间模式
1	RTC 模块进入设置日期和时间模式

指令的返回值：

无

例子：

RTC 模块进入设置日期和时间模式：rtc_flag 1

standby 指令

说明：standby 是系统用于进入待机模式的一条指令，当系统接收到该指令后，系统会自动进入待机模式。系统进入待机模式后只能通过 WAKEUP 接口才能唤醒。

指令格式：

指令
standby

指令参数：

参数	说明
无	该指令不需要参数

指令的返回值：

无

例子：

系统进入待机模式：standby

page 指令

说明：page 指令主要用于实现在不同页面之间进行切换的功能，该指令需要以页面的名称作为指令的参数。系统会根据指令的参数，在保存的页面信息中寻找对应的页面，并将该页面刷新到屏幕中。

注：page 指令以页面的名称作为指令的参数，仅当页面信息中存在该名称的页面，page 指令才能进行跳转。

指令格式：

指令	间隔符	参数
page	空格	page1

指令参数：

参数类型	参数示例	说明
页面名称	page1	作为参数的页面名称必须是存在于页面信息中

例子：

跳转到页面名称为 page1 的页面：page page1

redraw 指令

说明：redraw 指令用于控件的重新绘制，其主要作用是将处于其他控件底部的控件提到顶部来显示，该指令的参数为当前显示页面中的控件名称。当页面中的某个控件需要临时在页面的最顶层显示时，就以该控件的名称作为参数放置于 redraw 指令的参数区。使用 redraw 指令后显示在最顶层的控件，会因为 page、ref_star、ctrl_rev、ctrl_write 指令重新回到之前的位置。

注：只有处于当前页面中的控件才能作为 redraw 指令的参数被重新绘制。

指令格式：

指令	间隔符	参数
redraw	空格	text0

指令参数：

参数类型	参数示例	说明
控件名称	text0	作为参数的控件名称必须是处于当前显示的页面中的

例子：

重新绘制按钮控件 btn0：redraw btn0

prints 指令

说明：prints 指令主要用于向串口输出一些信息，该输出主要分为自定义的字符信息、自定义的数值信息、控件的可变属性。自定义的字符信息指的是，用户自行定义的需要向串口输出的字符串信息。自定义的数值信息指的是，用户自己编写的需要向串口输出的数值信息。控件的可变属性指的是，控件的一些可变更的属性的值。以上三种参数都需要遵循它们各自的格式。

注：输出控件的可变属性时只能获取当前页面中的控件的可变属性。

指令格式：

指令	间隔符	参数
prints	空格	“abcde”
prints	空格	535400
prints	空格	btn0.X

指令参数：

参数类型	参数示例	说明
自定义字符信息	“abcde”	自定义的字符信息需要包含在英文的双引号中
自定义数值信息	535400	自定义的数值信息必须是纯数值的，不包含任何数字以外的字符信息
控件的可变属性	btn0.X	控件的可变属性参数由控件的名称和可变的属性组成，中间使用“.”隔开，属性统一首字母大写，控件名称为当前页面中的控件的名称

例子：

发送字符串 A1B1：prints “A1B1”

发送数值 5663：prints 5663

发送按钮控件 bt0 的 Y 值：prints btn0.Y

vis 指令

说明：vis 指令实现了控件的隐藏和显示功能。该指令有两个参数，一个参数指定了需要被操作的控件，该参数以当前页面中的控件名称作为参数值；另一个参数指定了被操作的控件是显示或是隐藏。

注：被操作的控件必须是当前显示页面中的控件。

指令格式：

指令	间隔符	参数 1	间隔符	参数 2
vis	空格	btn0	空格	disable

指令参数：

参数 1		
参数类型	参数示例	说明
控件名称	btn0	作为参数的控件名称必须是处于当前显示的页面中的
参数 2		
参数类型	参数示例	说明
	enable	显示控件
	disable	隐藏控件

例子：

隐藏按钮控件 btn0：vis btn0 disable

显示按钮控件 btn0：vis btn0 enable

tsw 指令

说明：tsw 指令实现了控件的触控使能和除能的功能。该指令有两个参数，一个参数指定了需要被操作的控件，该参数以当前页面中的控件名称作为参数值；另一个参数指定了被操作的控件的触控功能为使能还是除能。

注：被操作的控件必须是当前显示页面中的控件。

指令格式：

指令	间隔符	参数 1	间隔符	参数 2
tsw	空格	btn0	空格	disable

指令参数：

参数 1		
参数类型	参数示例	说明
控件名称	btn0	作为参数的控件名称必须是处于当前显示的页面中的
参数 2		
参数类型	参数示例	说明
	enable	使能控件的触控功能
	disable	除能控件的触控功能

例子：

除能按钮控件 btn0 的触控功能：vis btn0 disable

使能按钮控件 btn0 的触控功能：vis btn0 enable

touch_j 指令

说明：touch_j 指令用于串口屏屏幕的触控校准功能。使用该指令后，系统会进入触控校准界面并依次在屏幕的四个角显示“+”符号，根据提示依次点击“+”。如果校准成功，屏幕中会显示“touch adjust finish”表示校准完成；如果校准失败，屏幕会显示“please try again”，系统会重新进行校准。如果 3 次校准都失败，屏幕会显示“adjust fail,using default mode”并退出校准功能。

指令格式：

指令
touch_j

指令参数：

参数类型	参数示例	说明
无	无	该指令不需要参数

例子：

进入串口屏屏幕的触控校准功能：touch_j

ref_stop 指令

说明：ref_stop 指令用于暂停屏幕的刷新功能。当使用 ref_stop 指令暂停了屏幕的刷新功能后，屏幕会停留在最后刷新的页面上。该指令仅暂停了屏幕的刷新功能，其他功能依然可以正常执行。使用 ref_star 指令恢复屏幕刷新功能后，系统会将最新的屏幕显示效果刷新到屏幕中。该功能仅在当前页面有效，如使用 page 指令切换其他页面后，暂停屏幕刷新功能将会被恢复。

指令格式：

指令
ref_stop

指令参数：

参数类型	参数示例	说明 .
无	无	该指令不需要参数

例子：

暂停屏幕的刷新功能：ref_stop

ref_star 指令

说明：ref_star 指令主要用于恢复屏幕的刷新功能。使用该指令恢复屏幕的刷新功能时，系统会将最新的屏幕显示效果刷新到屏幕中。这其中包括屏幕刷新暂停期间的指令执行，事件执行和触控执行的效果。

指令格式：

指令
ref_star

指令参数：

参数类型	参数示例	说明
无	无	该指令不需要参数

例子：

恢复屏幕的刷新功能：ref_star

com_stop 指令

说明：com_stop 指令主要用于暂停串口指令的执行功能。当串口指令的执行功能被暂停后，新接收到的串口指令会被缓存到串口指令缓存区，该区域有 256 个字节。当剩余的空间不足以存储下新接收到的指令时，该指令会被丢弃。在使用 com_star 指令恢复串口的指令执行功能时，系统会依次执行指令缓存区中的指令。

注：指令是否缓存到指令缓存区是根据指令的长度和指令缓存区所剩的空间来决定的，当接收到的指令长度大于缓存区所剩的空间大小，该指令就会被丢弃；当接收到的指令长度小于缓存区所剩的空间大小，该指令就会被缓存到指令缓存区。

指令格式：

指令
com_stop

指令参数：

参数类型	参数示例	说明 .
无	无	该指令不需要参数

例子：

暂停串口指令的执行功能：com_stop

com_star 指令

说明：com_star 指令主要用于恢复串口指令的执行功能。该指令仅在串口指令的执行功能被暂停时有效，其无法在正常的指令执行过程中使用。当使用 com_star 指令将串口指令的执行功能从暂停中恢复时，系统会首先检查指令缓存区中是否存在需要被执行的指令，如果存在则首先执行指令缓存区中的指令，如果不存在，则执行系统的其他操作。

注：该指令仅在串口指令的执行功能被暂停时才有效，在正常的指令执行过程中，该指令被限定为无效的指令。

指令格式：

指令
com_star

指令参数：

参数类型	参数示例	说明
无	无	该指令不需要参数

例子：

恢复串口指令的执行功能：com_star

com_c 指令

说明：com_c 指令主要用于清除串口指令缓存区。该指令仅在串口指令的执行功能被暂停时有效，其无法在正常的指令执行过程中使用。在串口指令的执行功能被暂停的情况下，使用 com_c 指令即可清除指令缓存区中的所有指令。

注：该指令仅在串口指令的执行功能被暂停时才有效，在正常的指令执行过程中，该指令被限定为无效的指令。

指令格式：

指令
com_c

指令参数：

参数类型	参数示例	说明
无	无	该指令不需要参数

例子：

清空串口指令缓存区：com_c

reset 指令

说明：reset 指令用于串口屏的系统复位功能。

注：reset 指令只有被执行的时候才会复位系统，如果仅将其存储到串口指令缓存区中时是不会复位系统的。当使用 com_star 指令恢复串口指令的执行功能时，串口指令缓存区中存在 reset 指令，该指令被执行并复位系统。

指令格式：

指令
reset

指令参数：

参数类型	参数示例	说明
无	无	该指令不需要参数

例子：

复位系统：reset

setlayer 指令

说明：setlayer 指令主要用于两个控件之间的图层互换。在同一个位置的两个控件，顶层的控件会掩盖底层的控件。该指令需要两个参数，这两个参数必须是当前显示页面中的控件的名称，且不能是页面控件。

注：该指令只能互换当前页面中的控件，且页面控件不包含在可互换的控件范围内。

指令格式：

指令	间隔符	参数 1	间隔符	参数 2
setlayer	空格	btn0	空格	btn1

指令参数：

参数 1		
参数类型	参数示例	说明
控件名称	btn0	作为参数的控件名称必须是处于当前显示的页面中的
参数 2		
参数类型	参数示例	说明
控件名称	btn1	作为参数的控件名称必须是处于当前显示的页面中的

例子：

互换按钮控件 btn0 和 btn1 的图层顺序：setlayer btn0 btn1

beep 指令

说明：beep 指令主要用于实现蜂鸣器鸣叫的功能。该指令有一个参数，该参数指定了蜂鸣器鸣叫的时长，以毫秒 (ms) 为单位。

指令格式：

指令	间隔符	参数
beep	空格	500

指令参数：

参数	说明
1~65535	蜂鸣器鸣叫的时长

例子：

让蜂鸣器鸣叫 200ms：beep 200

ctrl_rev 指令

说明：ctrl_rev 指令主要用于将一个控件的可变属性赋值给另一个控件的可变属性。该指令需要两个参数，一个为目标参数，一个为数据源。两个参数都需要是当前页面中的控件的可变属性。在赋值的过程中，系统会根据可变属性自身的限定范围对新获得的值进行范围限制处理。如一个可变属性的范围为 2~5，此时将一个 10 赋给该可变属性，由于其范围的限制，最终得到的结果是该可变属性的值为 5。

注：两个参数都需要是当前页面中的控件的可变属性，两个参数的类型必须是相同的，字符与数值不能互相赋值，两个参数之间使用的间隔符为“=”。目标参数处于等号的左边，数据源处于等号的右边。

指令格式：

指令	间隔符	参数 1	间隔符	参数 2
ctrl_rev	空格	btn0.H	=	btn1.X

指令参数：

参数 1		
参数类型	参数示例	说明
控件的可变属性	btn0.H	控件的可变属性参数由控件的名称和可变的属性组成，中间使用“.”隔开，属性统一首字母大写，控件名称为当前页面中的控件的名称

参数 2		
参数类型	参数示例	说明
控件的可变属性	btn1.X	控件的可变属性参数由控件的名称和可变的属性组成，中间使用“.”隔开，属性统一首字母大写，控件名称为当前页面中的控件的名称

注：控件的可变属性可以参考控件的属性表。

例子：

将按钮控件 btn0 的 X 值赋给按钮控件 btn1 的 X：ctrl_rev btn1.X=btn0.X

ctrl_read 指令

说明：ctrl_read 指令主要用于获取控件的可变属性的值。系统获取到控件的可变属性值后，通过串口输出该值。

注：ctrl_read 指令只能获取到处于当前页面中的控件的可变属性的值。

指令格式：

指令	间隔符	参数
ctrl_read	空格	btn0.W

指令参数：

参数类型	参数示例	说明
控件的可变属性	btn0.W	控件的可变属性参数由控件的名称和可变的属性组成，中间使用“.”隔开，属性统一首字母大写，控件名称为当前页面中的控件的名称

注：控件的可变属性可以参考控件的属性表。

例子：

获取按钮控件 btn1 的 W 值：ctrl_read btn1.W

ctrl_write 指令

说明：ctrl_write 指令主要用于实现向控件的可变参数中写入数据的功能。该指令需要两个参数，一个为目标参数，一个为需要写入的数据。目标参数必须是当前页面中的控件的可变属性，写入的数据需要跟目标参数的数据类型一样。在赋值的过程中，系统会根据可变属性自身的限定范围对新获得的值进行范围限制处理。如一个可变属性的范围为 2~5，此时将一个 10 赋给该可变属性，由于其范围的限制，最终得到的结果是该可变属性的值为 5。

注：写入的数据类型必须跟目标参数的数据类型一致，向数值型的目标参数中写入字符串会出现错误。两个参数之间使用的间隔符为“=”时，目标参数处于等号的左边，需要写入的数据处于等号的右边。

指令格式：

指令	间隔符	参数 1	间隔符	参数 2
ctrl_write	空格	btn0.H	=	100
ctrl_write	空格	btn0.H	+	+
ctrl_write	空格	btn0.H	-	-
ctrl_write	空格	btn0.H	+	10
ctrl_write	空格	btn0.H	-	10

指令参数：

参数 1		
参数类型	参数示例	说明
控件的可变属性	btn0.H	控件的可变属性参数由控件的名称和可变的属性组成，中间使用“.”隔开，属性统一首字母大写，控件名称为当前页面中的控件的名称
参数 2		
参数类型	参数示例	说明
数值型数据	120	数值型数据必须是纯数值
字符型数据	“abcde”	字符型数据需要包含在英文的双引号内部

注：控件的可变属性可以参考控件的属性表。数值型数据暂时不支持小数的识别。

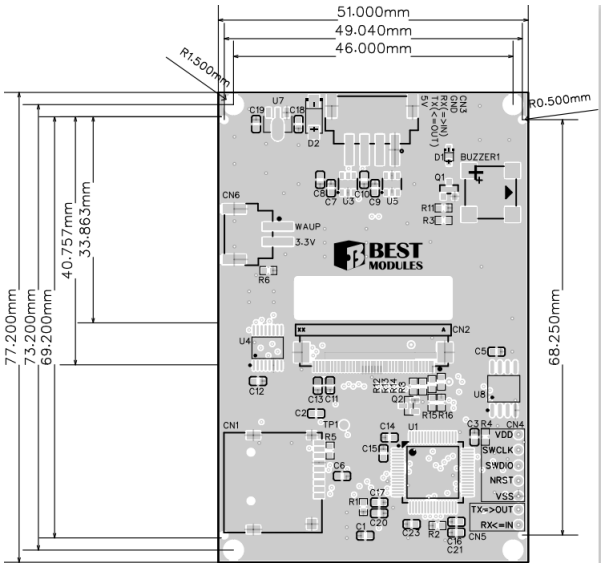
例子：

将进度条控件 probar0 的 Val 值改成 80：ctrl_write probar0.Val=80
将文本控件 text0 的 Txt 值改成“下一页”：ctrl_write text.Txt=“下一页”
数字控件 num1 的 Val 值自增 1：ctrl_write num1.Val++
数字控件 num1 的 Val 值自减 1：ctrl_write num1.Val--
数字控件 num1 的 Val 值加 10：ctrl_write num1.Val+10
数字控件 num1 的 Val 值减 10：ctrl_write num1.Val-10

错误码

指令执行结果编码	编码说明
0x00	无效指令
0x01	指令执行成功
0x02	数据读取成功
0x03	数据写入成功
0x04	参数错误，缺少参数
0x05	参数错误，多余的参数
0x06	参数错误，不是数值
0x07	参数错误，输入的参数不符合参数要求
0x08	参数错误，参数超出规定范围
0x09	该指令为仅读指令
0x0A	该指令为仅写指令
0x0B	该控件不存在
0x0C	该属性不存在或者不可更改
0x0D	没有找到对应的页面
0x0E	无法进入休眠状态，没有正确设置唤醒方式
0xFF	无错误码返回

尺寸图



参考信息

修订历史

日期	作者	版本	修改信息
2024.03.05	洪嘉阳	V1.00	更新网盘链接并修改上位机软件章节相关内容
2023.08.25	洪嘉阳	V1.00	第一版

开发工具

上位机软件：HT32 USART HMI

网盘链接：https://wwyz.lanzoul.com/b04w7km5g

网盘密码：8rl6

相关文档

串口屏模块应用手册

在线购买

倍创科技