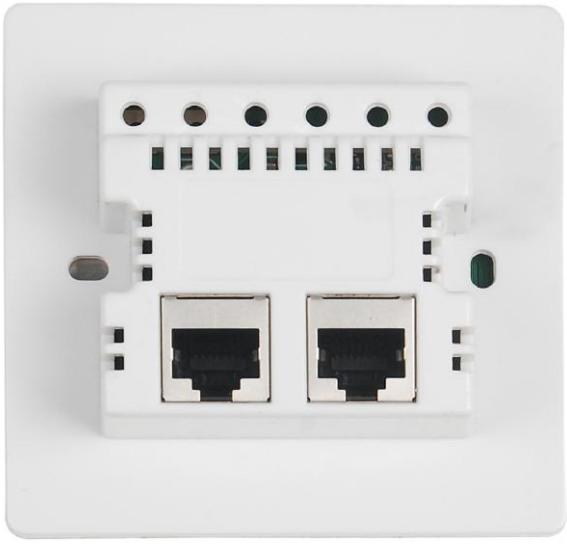
大彩 86 系列串口屏数据手册

版本：V1.0



版本	描述	日期	审查
V1.0	文档编写	2014-05-18	李 勇

86 串口屏应用环境

- 替换传统机械式开关，添加智能开关灯模式；
- 替换传统电容触摸调光器面板；
- 替换传统黑白温控器显示，升级到彩屏触摸功能；
- 智能家居开关面板组网应用；
- 中控系统控制面板；




86 串口屏硬件架构图

系统处理器采用 Cortex-M3+高速 FPGA 双核设计，M3 主要进行协议解析和图片下载，FPGA 主要实现 Nandflash 的图片读取和 TFT 控制显示。整个系统无运行任何 OS 操作系统，纯硬件驱动，稳定可靠，上电即可运行。内部结构如图 A 所示。

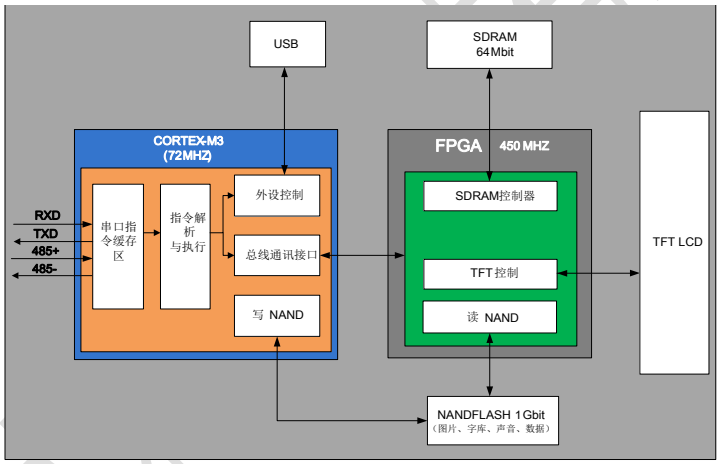


图 A 经济型串口屏内部结构图

系统在无指令接收的状态下，M3 处于空闲状态，所有大负荷的数据读取和显示刷新工作全部由 FPGA 纯硬件完成，该架构特点从某种程度上确保了系统的健壮性和抗干扰性。

由于 FPGA 内部是采用并行处理机制，所以 Nandflash 的数据读取、SDRAM 的写入和 RGB 数据输出全部在一个时钟节拍下执行，确保了系统图片刷新速度快，响应迅速等特点。

设备内部有 4.7K 字节的指令缓冲区，用户主机可无等待、一次性发送多条指令后退出串口程序。整个过程操作简单，程序代码量大大降低。

操作与创新-- 30 分钟完成界面设计

任何大彩用户只需要 3 个步骤，即可 30 分钟内完成复杂的人机交互设计。

1. 准备美工素材。

安排美工人员将开机画面、文本背景、按钮图标和提示框等产品所需图片设计好。

2. 利用配套的 VisualTFT 软件进行画面编辑、控件配置和图片下载。

首先利用配套的上位机 VisualTFT 软件，将预先设计好的美工图片进行界面排版和控件配置，然后运行 “虚拟串口屏”进行模拟仿真，最后通过 SD/UART 将整个工程信息下载到

串口屏内部存储器中。PC 软件会对工程中的每个画面、图片和控件分配一个唯一的 ID 号。

3. **用户单片机监听和发送相应的串口指令控制画面显示。**

工程下载到屏内后，一旦按下画面某个按钮，用户 MCU 串口就会收到屏幕上传的按钮 ID 信息或坐标值。通过对 ID 号进行解析，用户即可获取当前按钮的画面位置和功能属性，这样就可以控制相关外围设备动作或画面更新显示。

对于无触摸产品，用户单片机无需监听按钮 ID 上传的信息，只需发送相关指令进行画面切换和文本图片显示等。

什么是“虚拟串口屏”？

“虚拟串口屏”是广州大彩光电科技有限公司(www.gz-dc.com)自主创新开发的国内首款串口屏模拟仿真器。

用户新建好工程、配置和编译无误后，就可以运行它来测试界面设计是否正确，仿真结果与真实串口屏一模一样。

即便用户在没有购买硬件的前期下，通过自己单片机 RS232 串口与“虚拟串口屏”直接相连，也可以进行相互通信，鼠标点击按钮就会立刻上传按钮控件 ID 或坐标信息，如图 C 所示。一旦“虚拟串口屏”调试通过，真实硬件无需再调试，直接下载即可。

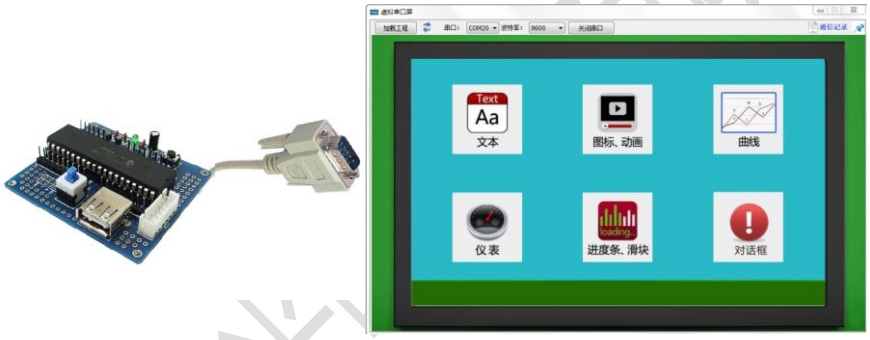


图 C 用户单片机串口与“虚拟串口屏”联机调试

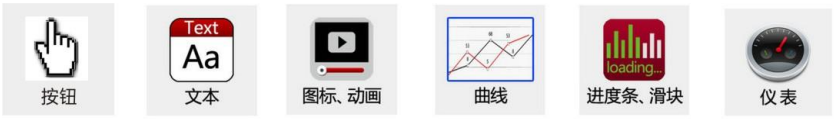
为了进一步提高开发效率，用户还可以通过 Keil 开发环境与“虚拟串口屏”进行绑定 Debug 调试。程序单步调试时，所有运行结果都可以在“虚拟串口屏”上呈现，大大节省工程师程序开发时间。

产品特征

- 型号和尺寸

	型号	接口	尺寸	分辨率	备注
86 串口屏	DC86C035_01T/NW	RS485	3.5〃	320*240	--T 带触摸 --N 不带触摸
	可以 ODM 设计，接收任意尺寸定制、协议定制、RS232 和 WIFI 接口等				

- 核心处理器单元
  - 采用 32 位 Cortex-M3 内核 + 高速 FPGA 架构
  - 无操作系统，纯硬件驱动显示，上电即刻运行，显示速度快
- 工作电压
  - 标准 5-26V；
- 图片存储容量
  - 内置 1GBit Flash，可存储 666 张全屏 320*240 张图片
- 通信接口
  - 与用户主机 MCU 通信接口：RS232/485/WIFI
  - 图片下载接口：USB
- 硬件功能特性
  - 16 位真彩色 RGB 显示(64K 色)；
  - 内置标准 8*12、8*16、12*24、16*32、32*64 ASCII 字库，12*12、16*16、24*24GBK 和 32*32、64*64 GB2312 字库，可自定义任意 windows 字库；
  - 支持 BMP、JPEG、WMF、PNG 和 GIF 等图片格式下载；
  - 支持任意位置文本、图片、GIF 动画和常用 GUI 显示；
  - 支持多种组态控件：按钮控件、文本控件、进度条、仪表、曲线和滑块控件等；
  - 外部 4 个按钮可自定义配置指令输出；
  - 内置人体感应传感器，人体靠近后液晶背光自动点亮；
  - 支持系统弹出内置虚拟键盘输入；
  - 支持中文输入法(需定制)；
  - 支持所有画面按钮自定义指令输出；
  - 波特率设置范围 1200-38400bps；
  - 255 级背光调节和自动屏保模式；
  - 可 ODM 设计，按需定制，满足用户一切需求。
- 组态控件特点



  - **按钮控件**

    通过上位机 VisualTFT 软件对工程画面中的所有按钮进行配置，即可在无用户程序参与下，自动实现画面的切换，按钮的按下和弹起，自定义指令输出等功能。画面中任何按钮被按下，屏幕都会将此按钮的 ID 编号上传给用户，无需计算坐标值。

  - **文本控件**

    上位机预先将文本在工程画面中的位置、颜色、字体和背景色设置好，用户只需对相应的文本 ID 进行读写数据，无需关心坐标值、颜色和字体等信息。

- **动画控件**

  用户可任意控制 gif 动画的播放、停止、暂停、上下帧和指定帧播放。

- **图标控件**

  用户可轻松实现同一位置不同状态图标显示。

- **曲线控件**

  用户只需发送 AD 采样数据，屏幕会自动进行拟合，动态推移显示。

- **仪表控件**

  通过上位机预先对仪表盘进行设置，用户只需要发送相应数值即可实现指针转动。假如表盘刻度范围为 0-180，用户只需发送数值 90，指针将自动转到 90 刻度处。

- **进度条控件**

  通过上位机预先对进度条的方向、起始值、终止值、前景和背景图等参数进行设置，用户只需要发送对应的显示数值，即可实现进度条递增或递减。

- **滑块控件**

  通过上位机预先对滑块的游标大小、刻度值、起始值、终止值和背景图等参数进行设置，用户拖动游标或点击某个刻度处时，设备将立刻上传当前游标所在位置值。

- **系统内置键盘**

  上位机预先对画面某位置处进行弹出键盘设置，点击该区域后，系统将自动弹出内置键盘。用户选择中英文输入法后点击确定，输入的字符全部以 ASCII 码形式上传到主机。

- 显示速度和功耗

尺寸		3.5〃
单一颜色清屏(ms,业界最快)		1.4
全屏图片更新时间(ms,业界最快)		12
16*16 文本显示速度(ms,业界最快)		0.35
功耗大小 (mA@5V)	背光最亮	170
	关闭背光	71

- LCD 特性
  - 显示器类型：TFT LCD
  - 背光灯管：LED
  - 背光灯寿命（平均）：大于 30,000 小时
  - 对比度/亮度/视角：(选用常规参数、与屏级别相关)
- 触摸屏特性
  - 触控类型：4 线精密电阻式
  - 透光率: 80%
  - 触控单点次数：>100 万次
  - 工作温度：-20℃～70℃
  - 可选配电容式触摸，默认电阻式触摸；
- 工作环境
  - 工作温度：-20℃～70℃
  - 相对湿度：10%-90%@40℃,无凝露
  - 震动测试：10 to 25Hz(X,Y,Z 方向 2G　30 分钟)

- ESD 标准：Air= ±8KV，Contact= ±4KV，class B (该参数与屏类型有关)

- 定制服务
  - 可定制 Zigbee、433M、POE 等通信接口；
  - 定制 PCB 尺寸及厚度、添加用户电路、选用指定高亮度、特殊规格 TFT 屏；
  - 根据用户需要定制特殊指令或控件，降低主机开销；
  - 可定制电阻或电容触摸板；
  - 可提供图片美工及产品结构设计服务；
  - 一次性下单 500PCS 可免费进行定制，包含用户程序、PCB 贴标等；
  - 按需定制，满足用户一切需求；

VisualTFT 软件特性

VisualTFT 是广州大彩光电科技有限公司(www.gz-dc.com)自主创新开发的一款功能强大的串口屏开发调试软件。用户新建工程后，导入设计好的美工图片，然后对每个画面中的按钮和其它控件进行配置，模拟仿真正确后，最后将整个工程下载到串口屏中。设备与 PC 连接成功后，还可以进行同步和调试显示，如图 D 所示。

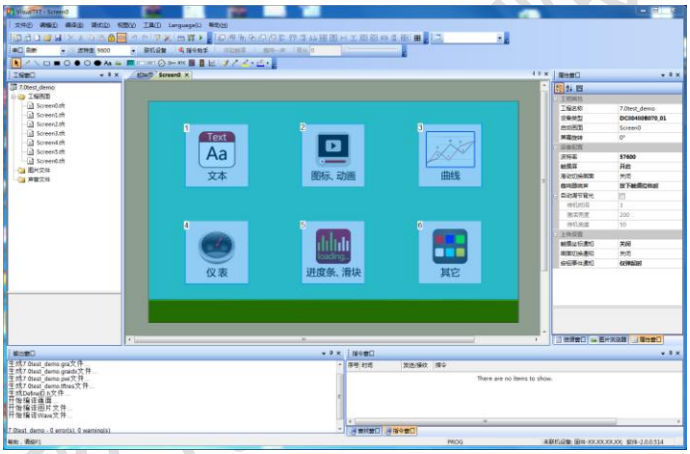


图 D VisualTFT 主界面

软件主要功能如下：

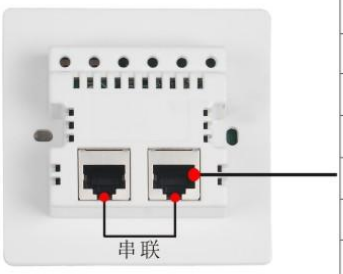
- VisualTFT IDE 环境操作人性化，界面美观大方；
- 软件集成了常用图标、按钮和键盘等矢量图，降低美工难度；
- 画面支持图片缩放、文字编辑和元素拷贝等常用操作功能；
- 工程编译后为每个画面、图片和控件分配唯一 ID 编号；
- 支持各种组态控件：按钮控件、文本控件、进度条、仪表、曲线等；
- 内嵌“虚拟串口屏”模拟仿真器，节省开发周期；
- 支持二进制文件烧录，量产更方便安全。
- 根据用户需求，定制特殊的 PC 软件功能；

目 录

第1章 接口定义及开发工具

1.1 通讯接口定义

产品通讯接口如图 1.1 所示。



定义	描述
1. TXD	接用户MCU TXD引脚
2. RXD	接用户MCU RXD引脚
3. VCC	5-26V
4. VCC	5-26V
5. GND	无
6. GND	无
7. 485+	用户485+/A
8. 485-	用户485-/B

图 1.1 产品引脚定义

1.2 开发工具

86 串口屏开发所需工具清单如下：

1. 5V/12V 电源适配器。建议电流 1A 以上，保证足够电流供应；
2. 网口转接板。用于给屏供电、PC 联机和图片下载，如图 1.2 所示。
3. Miniusb 电缆线，主要用于下载图片。

图 1.2 网口转接调试板

产品出厂时的标选配清单如表 1.1 所示。

表 1.1 产品出厂标配/选配清单

类型	产品	数量	备注
标配	86 串口屏	1 个	--
必需另购买件	网口调试板(含 0.5m 长网线)	1 个	将 RJ45 口转为独立的电源、485 和 232 口
	Miniusb 电缆线	1 个	用于下载图片
选购	电源适配器 5V	1 个	设备供电

第2章 产品外观及机械尺寸图

本章主要介绍常用尺寸的外观和机械尺寸图，一些特殊型号尺寸图可以登录公司网站 www.gz-dc.com 进行查阅和下载。

2.1 产品外观和机械尺寸

产品外观如图 2.1 所示。

图 2.1 产品外观和机械尺寸图

第3章 串口指令集

3.1 指令格式

一条完整的串口指令帧格式如表 3.1 无 CRC 校验指令帧格式 所示。如果指令参数大于 1 个字节，则高字节在前、低字节在后。指令的最大长度为 1024 字节(包含帧头和帧尾)，数值均为十六进制。串口格式：8 位数据位、1 位停止位、无效验位。

表 3.1 无 CRC 校验指令帧格式

指令	EE	XX	XX XX…XXX	FF FC FF FF
说明	帧头	指令	指令参数	帧尾

若用户需要指令支持 CRC 格式效验，则指令帧格如表 3.2 所示。

表 3.2 带 CRC 校验的指令帧格式

指令	EE	XX	XX XX…XXX	CRC16	FF FC FF FF
说明	帧头	指令	指令参数	校验位	帧尾

(1) 切换画面。若用户需要在屏幕上显示如图 3.1 所示的画面，则用户主机发送的串口指令如下：

图 3.1 切换画面显示

单片机发送命令	EE 【B1 00 00 01】FF FC FF FF
命令解析	EE 表示帧头
	B1 00 表示切换画面指令
	00 01 表示需要显示的目标画面 ID，2 个字节
	FF FC FF FF 表示帧尾
提示	每个画面的 ID 编号均由上位机编译后生成

(2) 按钮 ID 上传。若用户点击触摸图 3.2 中的“文本”按钮，假设当前画面 ID 为 2，文本按钮 ID 为 1，则串口屏上传串口指令如下：

图 3.2 按钮控件 ID 值上传

屏幕上传指令	EE【B1 11 00 02 00 01 10 00 00】FF FC FF FF
命令解析	EE 表示帧头
	B1 11 表示上传的组态控件指令；
	00 02 00 01 表示当前画面 ID 为 2，控件 ID 为 1
	10 表示控件为按钮控件
	00 表示按钮控件属性为切换画面
	00 保留
	FF FC FF FF 表示帧尾
提示	所有按钮控件的 ID 号均由上位机配置生成
说明	用户单片机串口接收到以上指令数据就可以解析出当前哪个画面的哪个按钮被按下

(3) 文本显示。若用户需要在图 3.3 中当前电压显示数字 220，假设画面 ID 为 1，文本控件 ID 为 7，则用户主机发送指令如下：

图 3.3 文本控件显示

单片机发送指令	EE【B1 10 00 01 00 07 32 32 30】FF FC FF FF
命令解析	EE 表示帧头
	B1 10 表示发送的组态控件指令；
	00 01 00 07 表示画面 ID 为 1，控件 ID 为 7

	32 32 30　表示数字 220 的 ASCII 码； FF FC FF FF　表示帧尾
说明	由于字体的颜色、大小、位置都在 PC 上预先进行了配置，所以用户单片机只需针对 ID 号发送数据即可

(4) 系统键盘输入。若用户需要在图 3.4 中的机组号显示“123”，假设画面 ID 为 1，文本控件为 1，则只需点击文本框处，然后在弹出的系统键盘中录入输入 123，最后点击确定，这样录入的文本将自动嵌入到文本框中显示，同时上传指令如下：

图 3.4 键盘录入参数显示

屏幕上传指令	EE【B1 11 00 01 00 01 11 31 32 33 00】FF FC FF FF
命令解析	EE　表示帧头
	B1 11　表示接收的组态控件指令；
	00 01 00 01 表示画面 ID 为 1，控件 ID 为 1
	11　表示控件为文本控件
	31 32 33　表示数字 123 的 ASCII 码
	00　表示字符结束
	FF FC FF FF　表示帧尾
说明	用户单片机接收到屏幕上传的 ASCII 码即可获取输入信息

(5) 自定义指令输出。用户可以设置按下某个按钮后，设备上传自己定义的数据串列。例图 3.5 所示，用户可以设置按下“自定义指令”按钮后，屏幕下发指令：FF 01 AA FF。

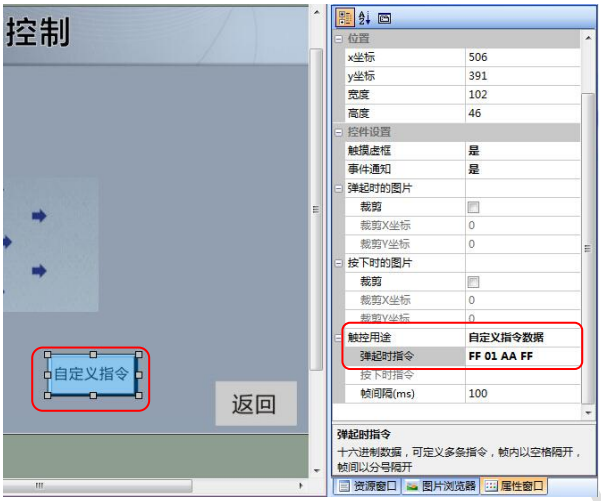


图 3.5 自定义指令数据

(6) 图标显示。若用户想实现某个状态图标变化显示，例如在运行时候，显示“运行图标”，停止时显示“停止图标”，复位时图标消失，异常时两个图标来回闪烁，此时可以使用图标控件来解决。

先用软件内置的图标生成器将所有状态图标生成一个 ICON 文件(ICON 文件包含了所有状态帧图)，然后单片机发送指令控制 ICON 的帧播放。例如图 3.6 所示，用户要将“停止图标”更换为“启动图标”，则发送指令如下：

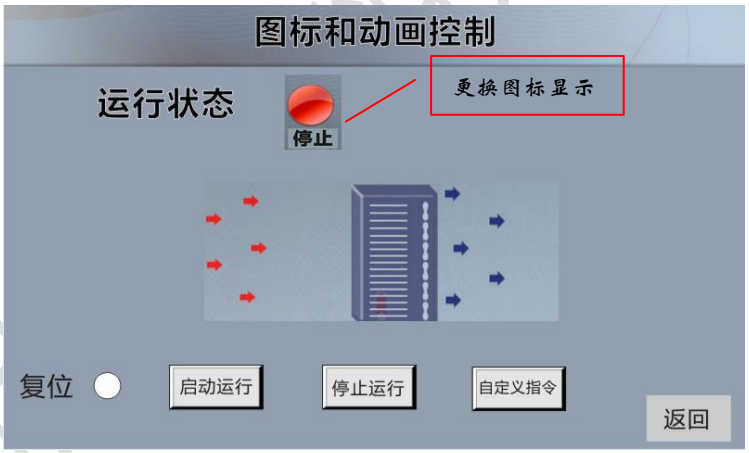


图 3.6 图标显示

单片机发送指令	EE【 B1 23 00 03 00 01 01】FF FC FF FF
命令解析	EE 表示帧头
	B1 23 表示图标控件指令；
	00 03 00 01 表示画面 ID 为 3，控件 ID 为 1
	01 代表“运行图标”的帧 ID
	FF FC FF FF 表示帧尾
说明	ICON 文件里面有 2 个图片帧，停止运行是帧 ID 0，启动运行是帧 ID 1，这样播放帧 1 就是显示运行图标，播放帧 0 就是显示停止图标。

(7) 仪表显示。若用户需要将图 3.7 中表盘 1 的指针转到度 5 处，假设画面 ID 为 4，仪表控件 ID 为 1，则用户单片机发送指令如下：

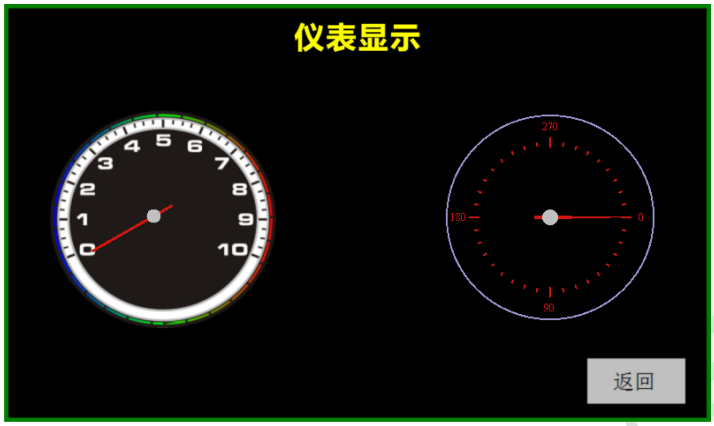


图 3.7 仪表控件显示

设备上传指令	EE【B1 10 00 04 00 01 00 00 00 32】FF FC FF FF
命令解析	EE　表示帧头
	B1 10　表示发送的组态控件指令
	00 04 00 01　表示画面 ID 为 4，控件 ID 为 1
	00 00 00 32　表示数值 50
	FF FC FF FF　表示帧尾
说明	由于 PC 预先设置了表盘的起始值 0，终止值 100，发送数值 50，指针正好指向刻度 5 方向处

(8) 曲线显示。若用户需要实现图 3.8 中曲线显示，用户主机只需发送 AD 采样序列值，设备将会自动进行缩放、平移推进显示。

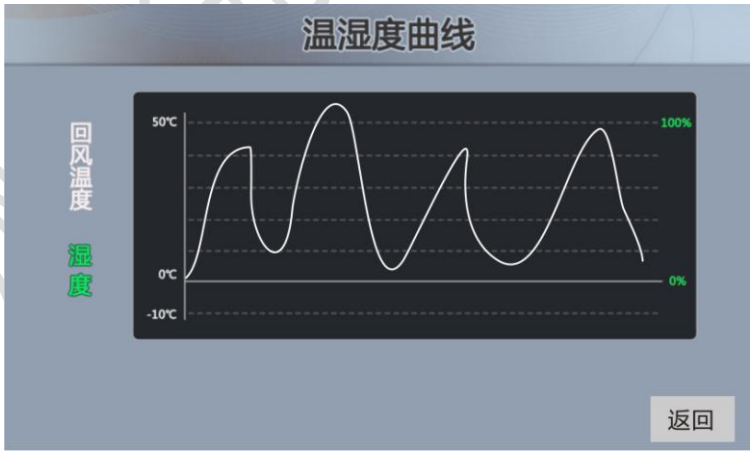


图 3.8 曲线控件显示

(9) 进度条和滑块显示。若需要实现图 3.9 所示的进度条滚动显示，用户主机只需发送显示的进度值即可。假设 PC 上配置的进度条起始值为 0，终止值为 100，画面 ID 为 3，进度条控件 ID 为 1，则用户需要将进度条显示在中央，发送指令如下：

图 3.9 进度和滑块显示

设备上传指令	EE【B1 10 00 03 00 01 00 00 00 32】FF FC FF FF
命令解析	EE 表示帧头
	B1 10 表示发送的组态控件指令
	00 03 00 01 表示画面 ID 为 3，控件 ID 为 1
	00 00 00 32 表示数值 50
	FF FC FF FF 表示帧尾
说明	由于 PC 预先设置了进度条的起始值 0，终止值 100，发送数值 50，进度条正好显示在中央位置

3.2 颜色格式

设备共支持 2^16=65536 种颜色(简称 65K 色)，RGB 为 565 格式，其高低字节分配如表 3.3 所示。

表 3.3 RGB 颜色分配格式

位数(Bit)	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
颜色分配	R					G						B				

举例说明：纯红色=F800H，纯蓝色=001FH

注：用户可以通过 visualTFT 上位机软件调试并获取期望的颜色和 RGB 值。

3.3 组态指令集与基本指令集区别

指令集分两部分：组态控件指令集和基本指令集。两者主要区别是：基本指令集可以理解为最底层的指令集，大部分操作必须包含坐标、颜色、字体等参数信息；组态指令集直接是面向对象 ID 操作，而这些对象的相关参数全部在上位机软件中进行配置。

组态指令集满足 99%的用户需求，开发简单，真正的“零”代码编程。除此之外，在组态指令集无法满足的情况下，用户还可以将基本指令和组态相结合，完成所期望显示功能。

3.4 指令列表

表 3.4 组态指令集

类别	指令	指令参数	说明
切换画面	0xB1+0x00	Screen_id	从当前屏幕切换到目标画面显示 Screen_id(2 个字节)：目标画面ID 该指令主要实现切换画面显示
读取画面	0xB1+0x01	无	读取当前画面的 ID 值 指令返回格式：EE B1 01 Screen_id FF FC FF FF Screen_id(2 个字节)：当前画面的编号 该指令主要用于获取当前屏幕处于哪个画面显示
按钮控件 ID 值上传	无	无	按下某个按钮后，设备主动上传的 ID 信息格式 上传格式：EE 【B1 11 Screen_id Control_id Control_type Slidervalue】 FF FC FF FF Screen_id(2 个字节)：画面编号 Control_id(2 个字节)：控件编号 Control_type(1 个字节)：固定值 0x10，表示为按钮控件类型 Subtype(1 个字节)：按钮控件的子类型 0x00：画面切换，表示当前按下的是画面切换按钮 0x01：开关类型，表示当前按下的是开关按钮 0x02：自定义键值，表示当前按下的自定义键值按钮 Status (1 个字节)：按钮状态 0x00：按钮由按下变成弹起状态 0x01：按钮从弹起变成按下状态 该指令主要告诉用户当前哪个画面上第几个按钮被按下了
自定义按钮控件 ID 上传值	无	无	通过上位机自定义按下某个按钮控件后上传的ID指令，多个指令之间用分号隔开
设置按钮弹起和按下状态	0xB1+0x10	Screen_id+Control_id+Status	将某个按钮设置为弹起或按下的显示状态 Screen_id(2 个字节)：画面编号 Control_id(2 个字节)：控件编号 Status (1 个字节)：按钮状态 0x00：按钮由按下变成弹起 0x01：按钮由弹起变成按下 该指令主要用于改变按钮的显示状态
读取按钮控件状态	0xB1+0x11	Screen_id+Control_id	查询某个按钮当前是按下还是弹起状态 Screen_id(2 个字节)：画面编号 Control_id(2 个字节)：控件编号 返回指令格式：EE B1 11 Screen_id Control_id Control_type Subtype Status FF FC FF FF Status（1 个字节）：0x00 弹起状态 0x01 按下状态 注：返回参数的定义同按钮控件值上传指令一致

更新文本控件数值	0xB1+0x10	Screen_id+Control_id+Strings	对指定的文本控件写入文本数据 Screen_id(2 个字节)：画面编号 Control_id(2 个字节)：控件编号 Strings（不定长）： 用户写入的文本数值； 该指令主要用于实现文本数据显示
读取文本控件数值	0xB1+0x11	Screen_id+Control_id	获取某个文本控件当前显示的数值 Screen_id(2 个字节)：画面编号 Control_id(2 个字节)：控件编号 返回指令格式：EE B1 11 Screen_id Control_id Control_type Strings FF FC FF FF 返回参数：Control_type(1 个字节)：固定值 0x11，表示为文本控件类型 Strings（4 个字节）：当前显示的文本值，文本后面附加 1 个 0x00 做为结束符
更新进度条控件数值	0xB1+0x10	Screen_id+Control_id+Progressvalue	对指定的进度条控件写入显示的数据 Screen_id(2 个字节)：画面编号 Control_id(2 个字节)：控件编号 Progressvalue（4 个字节）：用户写入的新进度条值 该指令主要用于实现进度条的递增或递减
读取进度条控件数值	0xB1+0x11	Screen_id+Control_id	获取指定进度条控件的当前数值 Screen_id(2 个字节)：画面编号 Control_id(2 个字节)：控件编号 返回指令格式：EE B1 11 Screen_id Control_id Control_type Progressvalue FF FC FF FF 返回参数：Control_type（1 个字节）：固定值 0x12，表示为进度条控件类型 Progressvalue（4 个字节）：当前进度条值
滑动条控件上传格式	无	无	当拖动滑动条时，设备不断上传游标数值如下： 上传格式：EE 【B1 11 Screen_id Control_id Control_type Slidervalue】 FF FC FF FF Screen_id(2 个字节)：画面编号 Control_id(2 个字节)：控件编号 Control_type(1 个字节)：固定值 0x13，表示为滑动条控件 Slidervalue (4 个字节)：表示当前游标数值
更新滑动条控件数值	0xB1+0x10	Screen_id+Control_id+Slidervalue	对指定的滑动条控件写入显示的数据 Screen_id(2 个字节)：画面编号 Control_id(2 个字节)：控件编号 Slidervalue(4 个字节)：用户写入的新滑动条值 该指令主要用于控制滑动条游标的位置
读取滑动条控件数值	0xB1+0x11	Screen_id+Control_id	获取指定滑动条控件的当前游标数值 Screen_id(2 个字节)：画面编号 Control_id(2 个字节)：控件编号

			返回指令格式：EE B1 11 Screen_id Control_id Control_type Slidervalue FF FC FF FF 返回参数：Control_type（1 个字节）：固定值 0x13，表示为滑动条控件类型 Slidervalue（4 个字节）：当前显示的滑动条值
更新仪表控件数值	0xB1+0x10	Screen_id+Control_id+ Metervalue	对指定的仪表控件写入显示的数据 Screen_id(2 个字节)：画面编号 Control_id(2 个字节)：控件编号 Metervalue(4 个字节): 用户写入的新仪表值 该指令主要实现仪表指针的转动
读取仪表控件数值	0xB1+0x11	Screen_id+Control_id	获取指定仪表控件的当前数值 Screen_id(2 个字节)：画面编号 Control_id(2 个字节)：控件编号 返回指令格式：EE B1 11 Screen_id Control_id Control_type Metervalue FF FC FF FF 返回参数：Control_type（1 个字节）：固定值 0x14，表示为仪表控件类型 Metervalue（4 个字节）：当前显示的仪表值 注：返回参数的定义同更新仪表控件数值一致
设置光标焦点	0xB1+0x02	SCREEN_ID + CONTROL_ID + ENABLE	在指定的文本控件上显示光标（仅适用自定义键盘的模式下） Screen_id(2 个字节)：画面编号 Control_id(2 个字节)：控件编号 Enable(1 个字节) ：光标显示使能 0x00：关闭显示； 0x01：开启显示
手动禁止/使能屏幕更新	0xB3	ENABLE	手动禁止/使能屏幕更新 Enable(1 个字节) ：更新使能 0x00：禁止更新； 0x01：使能更新 该命令主要解决某一画面中实时动态更新的控件数目过多，导致屏幕更新速度慢的问题。 使用方法：用户先发送禁止屏幕更新指令，然后发送整个画面中需要更新的内容，最后再使能屏幕更新。
屏蔽/隐藏控件	0xB1+0x03	SCREEN_ID + CONTROL_ID + ENABLE	将某个控件屏蔽或显示隐藏。 Screen_id(2 个字节)：画面编号 Control_id(2 个字节)：控件编号 Enable(1 个字节) ：屏蔽或隐藏使能 0x00：屏蔽或隐藏控件； 0x01：屏蔽/隐藏解除 该指令常用于某一时刻将指定按钮控件功能失效，也可用于将某个控件隐藏无法显示
	0xB1+0x20	SCREEN_ID + CONTROL_ID	启动动画播放 Screen_id(2 个字节)：画面编号 Control_id(2 个字节)：控件编号 启动播放后，动画每次从帧头 0 开始播放，格式仅支持 GIF。

动画控件显示	0xB1+0x21	SCREEN_ID + CONTROL_ID	停止动画播放 Screen_id(2 个字节)：画面编号 Control_id(2 个字节)：控件编号 执行停止播放后，下次将从帧头 0 开始播放
	0xB1+0x22	SCREEN_ID + CONTROL_ID	暂停动画播放 Screen_id(2 个字节)：画面编号 Control_id(2 个字节)：控件编号 执行暂停后，下次将从暂停帧开始继续播放
	0xB1+0x23	SCREEN_ID + CONTROL_ID + FlashImgae_ID	指定帧播放 Screen_id(2 个字节)：画面编号 Control_id(2 个字节)：控件编号 FlashImgae_ID(1 个字节)：某一动画帧 ID 指定从某一帧开始播放
	0xB1+0x24	SCREEN_ID + CONTROL_ID	播放上一帧 Screen_id(2 个字节)：画面编号 Control_id(2 个字节)：控件编号
	0xB1+0x25	SCREEN_ID + CONTROL_ID	播放下一帧 Screen_id(2 个字节)：画面编号 Control_id(2 个字节)：控件编号
动画控件值上传	无	0XB1+0x26 + SCREEN_ID + CONTROL_ID+ Status + FlashImgae_ID	按下或滑动某个动画控件时候，设备上传的相关信息： Screen_id(2 个字节)：画面编号 Control_id(2 个字节)：控件编号 Status (1 个字节)：0x00 表示触摸按下，0x01 表示弹起； FlashImgae_ID (1 个字节)：表示按下屏幕时刻显示的动画帧； 备注：用户可以在 PC 配置禁止/使能动画控件值上传。
图标控件显示	0xB1+0x23	SCREEN_ID + CONTROL_ID +IconImgae_ID	用户先使用上位机自带的图标生成器，把所有不同状态的图片合成 1 个 ICON 文件，然后主机指定某一图标帧显示。 Screen_id(2 个字节)：画面编号 Control_id(2 个字节)：控件编号 IconImgae_ID (1 个字节)：某一图标帧 ID 该指令常用于在画面同一个位置显示几种不同状态的图片。 注：若用户需要图标消失显示，可以做一帧透明的 PNG 图片即可达到消失效果。
图标控件值上传	无	0XB1+0x26 + SCREEN_ID + CONTROL_ID+ Status + IconImgae_ID	按下或滑动某个图标控件时候，设备上传的相关信息： Screen_id(2 个字节)：画面编号 Control_id(2 个字节)：控件编号 Status (1 个字节)：0x00 表示触摸按下，0x01 表示弹起； IconImgae_ID (1 个字节)：表示按下屏幕时刻显示的图标帧； 备注：用户可以在 PC 配置禁止/使能图标控件值上传。
	0xB1+0x30	SCREEN_ID + CONTROL_ID +CHANNEL+COLOR	添加指定数据通道 Screen_id(2 个字节)：画面编号 Control_id(2 个字节)：控件编号

曲线控件			CHANNEL(1 个字节)：共 8 个数据通道，编号范围（0~7） COLOR(2 个字节)：数据通道颜色
	0xB1+0x31	SCREEN_ID + CONTROL_ID+CHANNEL	删除指定数据通道 Screen_id(2 个字节)：画面编号 Control_id(2 个字节)：控件编号 CHANNEL(1 个字节)：共 8 个数据通道，编号范围（0~7）
	0xB1+0x32	SCREEN_ID + CONTROL_ID+ CHANNEL+DATA_LEN+DATA	指定通道末尾添加新数据 在指定数据通道末尾添加新数据，当数据长度超过缓冲区长度时，旧数据左移。 Screen_id(2 字节)：画面编号 ；Control_id(2 字节)：控件编号 CHANNEL(1 个字节)：共 8 个数据通道，编号范围（0~7） DATA_LEN(2 个字节)：数据长度 DATA ：不定长数据，长度由 DATA_LEN 指定 格式：数据代表是 Y 轴方向的值，X 轴方向的会根据水平缩放系数自动递增，例如当水平缩放系数为 1 的时候，每插入一个点 X 轴自动加 1，当水平缩放系数为 5 的时候，每插入一个点 X 轴自动加 5
	0xB1+0x33	SCREEN_ID + CONTROL_ID + CHANNEL	清空指定通道的数据 Screen_id(2 字节)：画面编号； Control_id(2 字节)：控件编号 CHANNEL(1 个字节)：共 8 个数据通道，编号范围（0~7）
	0xB1+0x34	SCREEN_ID + CONTROL_ID+ XOFFSET + XMUL + YOFFSET+YMUL	指定垂直/水平的缩放/平移 Screen_id(2 个字节)：画面编号 Control_id(2 个字节)：控件编号 XOFFSET (2 个字节)：水平偏移数据点数，左移为正，右移为负 XMUL (2 个字节)：水平缩放系数，单位 0.01 YOFFSET (2 个字节)：垂直偏移数值，下移为正，上移为负 YMUL (2 个字节)：垂直缩放系数，单位 0.01 采样点与坐标点的计算公式：第 N 个采样点的数值为 V X 坐标 = (N-XOFFSET)*XMUL*0.01 Y 坐标 = (V-YOFFSET)*YMUL*0.01
	0xB1+0x35	SCREEN_ID + CONTROL_ID+ CHANNEL+DATA_LEN+DATA	指定通道前端添加新数据 在指定数据通道最前端插入新数据，当数据长度超过缓冲区长度时，旧数据右移 Screen_id(2 字节)：画面编号 Control_id(2 字节)：控件编号 CHANNEL(1 个字节)：共 8 个数据通道，编号范围（0~7） DATA_LEN(2 个字节)：数据长度 DATA ：不定长数据，长度由 DATA_LEN 指定
设置按钮对内发送指令	无	无	通过 PC 设置按下某个按钮后对内执行的指令。例如按下 A 按钮后可执行弹起 B 按钮、播放某个动画控件、显示某个图标等
弹出系统键盘	无	无	通过 PC 设置按下某个文本框后弹出系统键盘，然后用户可以输入中英文字符到文本框显示
滑动切换画面	无	无	通过 PC 设置滑动切换画面使能，左右滑动即可顺序切换画面

用户也可以直接打开上位机 VisualTFT 软件，点击导航栏的“指令助手”，如图 3.10

所示，里面有包含了所有屏幕相关指令，方便用户更快捷的熟悉指令操作。另外，一些常规的波特率设置、校准触摸和 RTC 时间设置都可以直接使用指令助手来完成，如所示。

图 3.10 指令助手

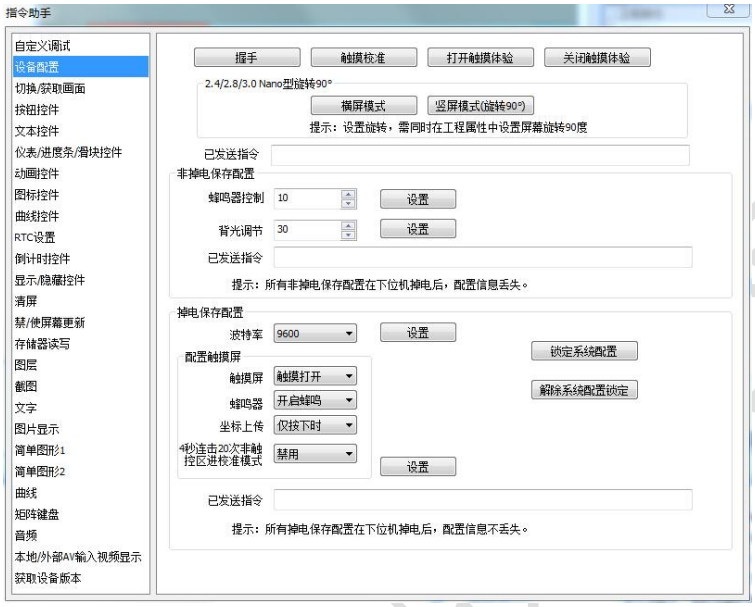


图 3.11 指令助手

在组态指令无法满足显示情况下，用户可以通过基本指令集来实现，指令集如下表所示。

表 3.5 基本指令集

类别	指令	指令参数	说明
握手	0x04	无	设备接收握手命令后，返回 55 给主机以示握手成功或在线状态。 返回格式：EE 55 FF FC FF FF
复位报告	无	无	一旦设备上电、意外重启或监控芯片复位，将立即上传相关数据，告知用户当前设备已被复位 返回格式：EE 07 FF FC FF FF
复位设备	0X07	0x35+0x5a+0x53+0xa5	主机通过串口指令复位设备 返回格式：EE 07 FF FC FF FF
设置前景色	0x41	Fcolor	前景色用于点、线、圆、图形和文字的颜色的指定 Fcolor(2 个字节) ：RGB 颜色值
设置背景色	0x42	Bcolor	背景色用于清屏、文字底色和曲线背景等颜色的指定 Bcolor(2 个字节) ：RGB 颜色值
清 屏	0x01	无	按照指定的颜色进行清屏 备注：清屏颜色取决于背景色设置，默认为蓝色
设置文字行列间距	0x43	Y_W+ X_W	Y_W(1 个字节) 以点为单位的行间距，取值 00~3F X_W(1 个字节) 以点为单位的列间距，取值 00~3F
设置文本框	0x45	Enable+Width+Hight	限制文本显示区域，以便自动换行显示

			Enable(1 个字节) 0x01：打开文本框限制使能，0x00：关闭文本框限制使能 Width (2 个字节) :文本显示框的宽度 Hight (2 个字节) :文本显示框的高度
设置图片过滤色	0x44	FilterColor	图片中的颜色与过滤色值相同时不予显示 FillColor (2 个字节): 过滤色 RGB 值
文本显示	0x20	X+Y+Back+Font+String	任意坐标处显示指定大小的文本内容 X (2 个字节) :以点为单位的 X 轴坐标值 Y (2 个字节) :以点为单位的 Y 轴坐标值 Back (背景色，1 个字节) 0x01:打开背景色显示 0x00：关闭背景色显示 Font (字库编码,1 个字节) 0x00: 8x12 点阵 (ASCII) 0x01: 8x16 点阵(ASCII) 0x02: 12x24 点阵(ASCII) 0x03: 16x32 点阵(ASCII) 0x04 12 x 12 点阵(GBK) 0x05: 16 x 16 点阵（GBK） 0x06: 24 x 24 点阵(GBK) 0x07: 32 x 32 点阵(GB2312) 0x08: 32 x 64 点阵(ASCII) 0x09: 64 x 64 点阵(GB2312) Strings：用户写入的字符串（高字节在前） 备注：文字字体颜色与前景色一致，底色为背景色
光标显示	0x21	Enable+X+Y+ Width+Hight	任意坐标处显示指定大小的光标 Enable(1 个字节) ：光标使能信号 0x00：关闭 0x01：开启 X(2 个字节): 以点为单位的 X 轴坐标值 Y(2 个字节): 以点为单位的 Y 轴坐标值 Width (1 个字节) :光标宽度 Hight (1 个字节) : 光标高度 备注：光标的颜色与当前光标区域起点颜色相反，闪烁时间默认 1 秒
全屏图片显示	0x31	Image_ID+MaskEn	全屏显示某张图片，起始位置固定(0, 0）坐标 Image _ID (2 个字节)：图片编号 MaskEn(1 个字节)：过滤使能 0x00:颜色不过滤 ;0x01 执行颜色过滤 备注：被过滤的颜色取决于图片过滤色的设置，下载的图片分辨率不能超过当前屏幕的分辨率，否则不能显示。
区域图片显示	0x32	X+Y+ Image_ID+MaskEn	任意坐标处显示某张图片 X (2 个字节): 以点为单位的 X 轴坐标值 Y(2 个字节) : 以点为单位的 Y 轴坐标值

			Image _ID (2 个字节)：图片编号 MaskEn(1 个字节)：过滤使能 0x00:颜色不过滤; 0x01 执行颜色过滤 备注：被过滤的颜色取决于过滤色的设置。
图片剪切显示	0x33	X+Y+Image_ID+Image_X+Image_Y+Image_W+Image_H+MaskEn	任意坐标处显示从某张图剪切过来的图片 X(2 个字节):以点为单位的 X 轴坐标值 Y(2 个字节):以点为单位的 Y 轴坐标值 Image_ID (2 字节): 要剪切的图片编号 Image_X (2 字节): 被剪切的图片起点 X 坐标 Image_Y (2 字节): 被剪切的图片起点 Y 坐标 Image_W (2 字节) : 剪切的宽度 Image_H (2 字节) : 剪切的高度 MaskEn(1 个字节)：过滤使能 0x00:颜色不过滤 ;0x01 执行颜色过滤 备注：被过滤的颜色取决于过滤色的设置
动画显示	0x80	X+Y+FlashImgae_ID+Enable+Playnum	任意坐标处显示 GIF 动画 X (2 个字节): 以点为单位的 X 轴坐标值 Y (2 个字节): 以点为单位的 Y 轴坐标值 FlashImage_ID (2 个字节) : 动画编号 Enable(1 个字节) ：播放使能 0x00：关闭动画播放； 0x01：开启动画播放 Playnum(1 个字节)：播放次数 0x00：重复播放； 0x01~0xFF：播放指定次数后停止 串口屏上传 EE 02 FF FC FF FF 表示动画播放结束 备注：动画只支持*gif 格式，一个画面只支持一个动画播放。如果期望一个画面同时播放多个 gif 动画，以及动画暂停、停止、播放上/下帧等功能，请使用组态动画控件，参见如下组态指令表。
前景色画点	0x50	X+Y	在屏幕上显示一个点，点的颜色取决于前景色设置 X (2 个字节) :以点为单位的 X 轴坐标值 Y (2 个字节) :以点为单位的 Y 轴坐标值
背景色画点 （删除点）	0x58	X+Y	在屏幕上显示一个点，点的颜色取决于背景色设置 X (2 个字节) :以点为单位的 X 轴坐标值 Y (2 个字节) :以点为单位的 Y 轴坐标值 备注：主要配合前景色画点使用，可用于清除前景色画的点
画线	0x51	X0 +Y0+X1+Y1	将指定的两个坐标点连接起来 X_0 (2 个字节) :以点为单位的直线 X 轴起点坐标值 Y_0 (2 个字节) :以点为单位的直线 Y 轴起点坐标值 X_1 (2 个字节) :以点为单位的直线 X 轴终点坐标值 Y_1 (2 个字节) :以点为单位的直线 Y 轴终点坐标值 备注：线的颜色值取决于前景色设置
按照等间隔 X 坐标	0x59	X_0+Xspace+Y_0+…+Yn	将指定的多个等间隔 X 轴坐标点快速连接起来

用前景色连线			X (2 个字节) ：以点为单位的 X 轴坐标值 Xspace(2 个字节)：以点为单位的 X 轴间隔值，相邻前后点固定间距为 Xspace Y (2 个字节) ：以点为单位的 Y 轴坐标值 备注：由于该指令不需发送 X 坐标值，指令发送时间节省一半，绘制速度提高一倍。线的颜色值取决于前景色设置
按照坐标偏移量用前景色连线	0x75	$(X,Y)_0$+（X1o,Y1o）+..+(Xno,Yno)	将指定的多个偏移量坐标点用前景色快速连接起来 X (2 个字节)：以点为单位的 X 轴坐标值 Y (2 个字节)：以点为单位的 Y 轴坐标值 X1o(1 个字节) ：以点为单位的 X 轴偏移量 Y1o(1 个字节) ：以点为单位的 Y 轴偏移量 备注：(X,Y)为第 1 点绝对坐标，后面的每一个点分别由前一个点的绝对坐标加上当前偏移量组成。偏移量的最高位为符号位，0 代表正偏移量，1 代表负偏移，最大偏移量值为正负 127 个点。
按照坐标偏移量用背景色连线	0x76	$(X,Y)_0$+（X1o,Y1o）+..+(Xno,Yno)	将指定的多个偏移量坐标点用背景色快速连接起来 X (2 个字节)：以点为单位的 X 轴坐标值 Y (2 个字节)：以点为单位的 Y 轴坐标值 X1o(1 个字节) ：以点为单位的 X 轴偏移量 Y1o(1 个字节) ：以点为单位的 Y 轴偏移量 备注：(X,Y)第 1 点为绝对坐标，后面的每一个点分别由前一个点的绝对坐标加上当前偏移量组成。偏移量的最高位为符号位，0 代表正偏移量，1 代表负偏移，最大偏移量值为正负 127 个点。
将指定的坐标点用前景色连线	0x68	$(X,Y)_0$+$(X,Y)_1$…+(X,Y)n	将指定的多个坐标点用前景色连接起来 X (2 个字节)：以点为单位的 X 轴坐标值 Y (2 个字节)：以点为单位的 Y 轴坐标值 备注：线的颜色值取决于前景色设置
将指定的坐标点用背景色连线	0x69	$(X,Y)_0$+$(X,Y)_1$…+(X,Y)n	将指定的多个坐标点用背景色连接起来 X (2 个字节)：以点为单位的 X 轴坐标值 Y (2 个字节)：以点为单位的 Y 轴坐标值 备注：线的颜色值取决于背景色设置
画空心圆	0x52	X_0+Y_0+R	任意坐标处画一个半径为 R 的空心圆 X_0 (2 个字节) :以点为单位的圆心 X 坐标值 Y_0 (2 个字节) :以点为单位的圆心 Y 坐标值 R (2 个字节) :空心圆的半径 备注：颜色值取决于前景色设置
画实心圆	0x53	X_0+Y_0+R	任意坐标处画一个半径为 R 的实心圆 X_0 (2 个字节) :以点为单位的圆心 X 坐标值 Y_0 (2 个字节) :以点为单位的圆心 Y 坐标值 R (2 个字节) : 空心圆的半径 备注：颜色值取决于前景色设置
画圆弧	0x67	X_0+Y_0+R+EA+SA	任意坐标处画一个圆弧 X0 (2 个字节) :以点为单位的圆心 X 坐标值

			Y0 (2 个字节) :以点为单位的圆心 Y 坐标值 R (2 个字节) :圆的半径 EA(2 个字节): 结束角度 SA(2 个字节): 起始角度 备注：钟表 3 点方向为 0 度，逆时针计算；颜色值取决于前景色调色板设置
画空心矩形/局部清屏	0x54	X_0+Y_0+X_1+Y_1	任意位置画一个空心矩形，也可用于局部清屏使用 X_0(2 个字节) :以点为单位的空心矩形左上角 X 坐标值 Y_0 (2 个字节) :以点为单位的空心矩形左上角 Y 坐标值 X_1 (2 个字节) :以点为单位的空心矩形右下角 X 坐标值 Y_1 (2 个字节) :以点为单位的空心矩形右下角 Y 坐标值 备注：颜色值取决于前景色设置
画实心矩形	0x55	X_0+Y_0+X_1+Y_1	任意位置画一个实心矩形 X_0(2 个字节) :以点为单位的实心矩形左上角 X 坐标值 Y_0 (2 个字节) :以点为单位的实心矩形左上角 Y 坐标值 X_1 (2 个字节) :以点为单位的实心矩形右下角 X 坐标值 Y_1 (2 个字节) :以点为单位的实心矩形右下角 Y 坐标值 备注：颜色值取决于前景色设置
画空心椭圆	0x56	X_0+Y_0+X_1+Y_1	任意位置画一个空心椭圆 X0(2 个字节):以点为单位的空心椭圆最左端 X 坐标值 Y0(2 个字节):以点为单位的空心椭圆最上端 Y 坐标值 X1(2 个字节):以点为单位的空心椭圆最右端 X 坐标值 Y1(2 个字节):以点为单位的空心椭圆最下端 Y 坐标值 说明：颜色值取决于前景色设置
画实心椭圆	0x57	X_0+Y_0+X_1+Y_1	任意位置画一个实心椭圆 X0(2 个字节):以点为单位的实心椭圆最左端 X 坐标值 Y0(2 个字节):以点为单位的实心椭圆最上端 Y 坐标值 X1(2 个字节):以点为单位的实心椭圆最右端 X 坐标值 Y1(2 个字节):以点为单位的实心椭圆最下端 Y 坐标值 说明：颜色值取决于前景色设置
背光调节	0x60	Light_level	设置背光亮度值 0x00：背光最亮 0xFF：背光关闭
自动屏保模式	0X77	Enable+BL_ON+BL_OFF +BL_ON_Time	设置屏保模式的背光亮度值和时间值。一段时间无触摸动作，屏幕自动降低亮度，进入节能模式直至触摸被按下时唤醒。 Enable(1 个字节)：使能信号 0x00：关闭省电模式 0x01：开启省电模式 BL_ON(1 个字节): 触摸激活后背光的亮度值 BL_OFF(1 个字节): 进入省电模式后背光的亮度值 BL_ON_Time(2 个字节)：在无触摸动作下，多长时间后进入省电模式，单位 1s 提示：只适合带触摸的型号，不带 TP 产品需要程序控制背光
蜂鸣器控制	0x61	Time	Time(1 个字节): 讯响时间，单位为 10ms

配置触摸屏	0x70	Cmd	Cmd(1个字节)：配置参数 BIT0: 1表示触摸屏打开，0表示触摸屏关闭； BIT1: 1表示触摸时蜂鸣器自动响， 0表示不响； BIT4~BIT2：触摸上传方式 000: 表示按下触摸屏时才上传一次坐标 001: 表示触摸屏被按下直至释放后才上传一次坐标 010: 触摸屏一直被按下时，每隔100ms定时上传坐标，释放时也上传一次坐标 011: 表示触摸屏被按下和释放时分别上传一次坐标 BIT5: 1表示在4秒内连续点击某个区域20下，屏幕进入触摸校准模式，0表示禁止此功能； BIT7-BIT6：保留 触摸上传格式： 按下时，上传格式： EE 01 X Y FF FC FF FF 释放时，上传格式： EE 03 X Y FF FC FF FF X 坐标、Y 坐标均为 2 个字节，高字节在前
触摸屏校准	0x72	无	校准完毕后，设备返回 EE 04 FF FC FF FF 或者在非触控区域某一点 4 秒内快速点击 20 下，自动进入触摸校模式，校准完成后自动返回当前显示画面
触摸屏体验	0x73	Enable	Enable：使能信号 0x00：关闭体验 0x01：体验使能 用户按下触摸后，屏对应坐标处显示一个红色实心圆。用于测试触摸屏精确度。
设置波特率	0xA0	Baudset	Baudset(单位 bps,1 个字节)，波特率编序： 0x00: 1200 0x01: 2400 0x02: 4800 0x03: 9600 0x04: 19200 0x05: 38400 0x06: 57600 0x07: 115200 0x08:1M 0x09:2M 0x0A : 218750 0x0B : 437500 0x0C : 875000 0x0D : 921800
写数据到 FLASH	0x87	Addr + Data0… +Datan	将数据保存在指定的 FLASH 地址中，当做 EEPROM 使用 Addr（4 个字节）：数据写入的起始地址 Datan（1 个字节）：写入的数据 存储空间为 128K 字节， 地址范围是 0~0x1FFFF 写入成功后返回： EE 0C FF FC FF FF
读取保存在 FLASH 中的数据	0x88	Addr + Length	将写入随机或顺序存储器中的数据读出 Addr（4 个字节）：数据读起始地址 Length(2 个字节)：读取数据的长度，单位为字节 返回的数据格式为： EE 0B Data0 … Datan FF FC FF FF
清除图层	0x05	Layer	清除某个图层内容 Layer(1 个字节)：写入的图层 （取值范围 0~1）
切换画面时自动清	0x06	Enable	设置切换画面时是否需要自动清除当前用户层

除当前图层			Enable（1 个字节）：使能信号 0x01：自动清除图层　0x00：禁止清除图层 默认切换画面时全部清除用户两个图层里面的全部内容。
截取当前屏幕并保存在 FLASH 中	0x46	Image_ID	将当前屏幕显示内容保存到设备 FLASH 中 Image_ID (1 个字节)：用户自定义保存在存储器中的画面编号
显示保存在 FLASH 中的截取画面	0x47	Image_ID	显示保存在设备 FLASH 中的截取画面 Image_ID (1 个字节)：用户自定义保存在存储器中的画面编号
RTC 显示设置 （需硬件支持）	0x85	Cmd+DisMode+Font +Color +X+Y	RTC 显示设置 Cmd (1 个字节)：参数配置 BIT0：使能信号 0：RTC 关闭　1：RTC 开启 BIT7-BIT1：保留 DisMode(1 个字节)：显示模式 0x00：格式 HH:MM:SS 0x01：格式 20XX-MM-DD　HH:MM:SS Font (1 个字节)：字体选择 0x00: 8x12 点阵 (ASCII)　0x01: 8x16 点阵(ASCII) 0x02: 12x24 点阵(ASCII)　0x03: 16x32 点阵(ASCII) 0x04 12 x 12 点阵(GBK)　0x05: 16 x 16 点阵（GBK） 0x06: 24 x 24 点阵(GBK)　0x07: 32 x 32 点阵(GB2312) 0x08: 32 x 64 点阵(ASCII)　0x09: 64 x 64 点阵(GB2312) Color (2 个字节)：显示颜色 X (2 个字节): 以点为单位的 X 轴坐标值 Y (2 个字节): 以点为单位的 Y 轴坐标值 建议用户直接使用时钟控件来完成 RTC 时间显示和校准。
RTC 时间设置 （需硬件支持）	0x81	Sec+Min+Hour+Day +Week+Mon+Year	时间参数设定 Sec:　秒设置　Min: 分设置； Hour:　小时设置　Day: 日设置 Week:　星期设置　Mon: 月设置 Year:　年设置 建议用户直接使用时钟控件来完成 RTC 时间显示和校准。 备注：各 1 个字节，以 BCD 码表示，星期天设置为 0x00
读取 RTC 时钟 （需硬件支持）	0x82	无	数据输出格式：EE +0xF7 +Year +Mon +Week +Day +Hour +Min +Sec +FF FC FF FF 备注：各 1 个字节，以 BCD 码表示 建议用户直接使用时钟控件来完成 RTC 的显示和时间的校准。
锁定系统配置	0x08	0xa5+0x5a+0x5f+0xf5	防止在系统运行过程中，收到错误指令帧导致系统配置意外修改。一旦配置被锁定，设备将无法接收外部串口命令进行修改，直到系统被解除。 配置参数包括：波特率、触摸和矩阵键盘工作模式、自动背光调节参数。 返回格式：　EE 17 FF FC FF FF

解除系统配置锁定	0x09	0xde+0xed+0x13+0x31	一旦解除系统配置锁定，设备可以重新接收外部串口命令来修改配置参数。 返回格式 EE 18 FF FC FF FF

第4章 组态指令集详述

以下章节主要介绍组态指令集功能和用法，内容中涉及到的驱动函数库可在参考程序范例中查看。程序范例可以在网站中进行下载。

4.1 切换画面

指令格式：EE【B1 00 Screen_id 】FF FC FF FF

参数说明：Screen_id(2 个字节)：画面编号

虽然通过按钮控件可设置按下某个按钮后自动切换到哪个画面，但有些场合需要主机做出逻辑判断后去控制目标画面显示。

程序参考代码：

```
{
  ......
  SetScreen(2);  // 切换到 Screen_id =2 的画面
}
```

4.2 读取画面

指令格式：EE【B1 01 】FF FC FF FF

参数说明：无

该指令主要用于获取当前画面的 ID 值。在可靠性环境中，主机通过发送该指令来再次确定画面是否切换成功。

指令返回格式：EE B1 01 Screen_id FF FC FF FF

其中，Screen_id(2 个字节)：当前画面的编号

程序参考代码：

```
{
  ......
  GetScreen( );  // 获取当前画面的 ID 值
 }
```

4.3 按钮控件 ID 值上传

按钮控件有 4 种用途：切换画面、开关描述、自定义按键和自定义指令数据。

(1) 切换画面。切换画面是指按下某个按钮后屏幕自动切换到另外一个画面显示。

例如图 4.1 所示，用户要实现点击“对话框”图标后切换到画面 Screen1 显示，则首先将整个按钮设置为触控区域，然后在属性窗口选择：触控用途→切换画面；目标画面→Screen1，最后运行“虚拟串口屏”进行效果验证。关于上位机详细操作见《大彩组态串口屏快速入门手册》。

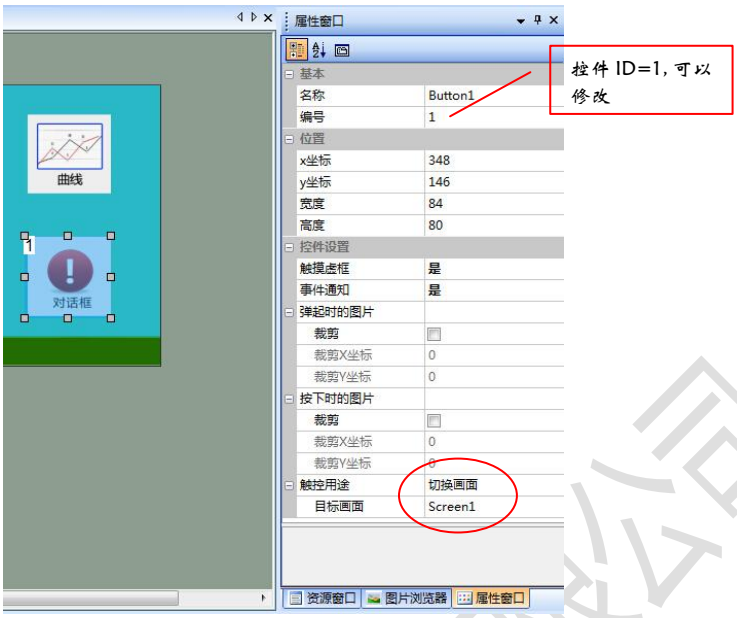


图 4.1 按钮控件-切换画面配置图

画面切换按钮上传的指令格式：

EE 【B1 11 Screen_id Control_id Control_type Subtype Status】FF FC FF FF

参数说明：Screen_id(2 个字节)：画面 ID

Control_id(2 个字节)：控件 ID 编号

Control_type(1 个字节)：固定值 0x10，表示为按钮控件

Subtype(1 个字节)：固定值 0x00，表示当前按下的是画面切换按钮

Status (1 个字节)：保留

(2) 开关描述。开关描述是指按钮作为一个按下或弹起的开关功能，共分为 4 种风格，内容如下：

a) 瞬变。按下后，开关自动弹起，类似轻触开关功能；

b) 开关。按下后，开关由弹起变成按下或由按下变成弹起，类似带锁开关功能；

c) 置位。开关只能由按下变成弹起；

d) 复位。开关只能由弹起变成按下；

例如图 4.2 所示，用户需要将“停止运行”按钮作为一个开关功能，则首先将整个按钮设置为触控区域，然后在属性窗口选择：触控用途→开关描述；操作风格→开关；按下时的图片→选择按下时的图片 UI，最后运行“虚拟串口屏”进行效果验证。

图 4.2 按钮控件-开关类型配置图

开关类型按钮上传格式：

EE 【B1 11 Screen_id Control_id Control_type Subtype Status】FF FC FF FF

参数说明：Screen_id(2 个字节)：画面编号

Control_id(2 个字节)：控件编号

Control_type(1 个字节)：固定值 0x10，表示为按钮控件

Subtype(1 个字节)：固定值 0x01，表示当前按下的是开关描述按钮

Status (1 个字节)：按钮状态

0x00：按钮由按下变成弹起状态

0x01：按钮从弹起变成按下状态

(3) 自定义键值。自定义键值是指按钮作为一个键值来使用，键值可设为任意 1 个 ASCII 字符。

例如图 4.3 所示，用户需要将“数字 9”按钮作为一个键值 9 使用，则首先将整个按钮设置为触控区域，然后在属性窗口选择：触控用途→自定义键值；类型→字符；字符→写入 9，最后运行“虚拟串口屏”进行仿真测试。另外，键值类型除了字符外，还可以选择“Enter”、“Clear”、“Backspace”、“Esc”或“Shift”特殊功能键。

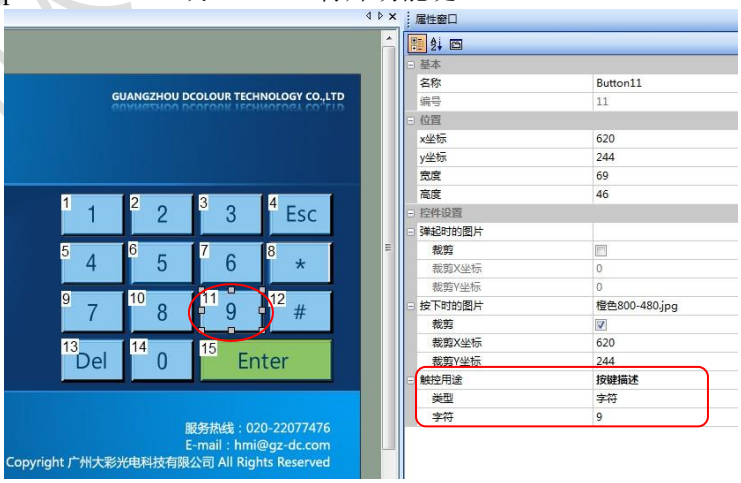


图 4.3 按钮控件-自定义键值配置图

自定义键值按钮上传格式：

EE 【B1 11 Screen_id Control_id Control_type Subtype Key_value Status】FF FC FF FF

参数说明：Screen_id(2 个字节)：画面编号

Control_id(2 个字节)：控件编号

Control_type(1 个字节)：固定值 0x10，表示为按钮控件

Subtype(1 个字节)：固定值 0x02，表示当前按下的是自定义键值按钮

Key_value(1 个字节)：用户自定义的键值，以 ASCII 码表示

Status (1 个字节)：保留

(4) 自定义指令数据。用户可以设置按下某个按钮后，设备上传自己定义的数据串列。例图 4.4 所示，用户可以设置按下“自定义指令”按钮后，屏幕下发指令： FF 01 AA FF。

提示：自定义指令中不能包含 FF FC FF FF 组合字符，否则会与常规指令冲突引起错误。

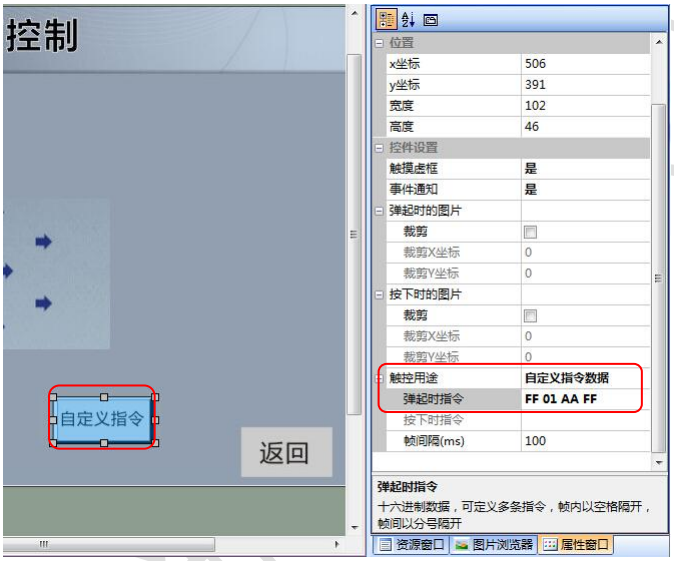


图 4.4 自定义指令数据

4.4 设置按钮弹起或按下状态

将画面某个按钮设置为弹起或按下状态，有两种方式： (1) 通过串口指令进行改变；(2) 通过 PC 进行按钮互斥配置。

1. 通过串口指令进行改变。

指令格式：EE 【B1 10 Screen_id Control_id Status 】FF FC FF FF

参数说明：Screen_id(2 个字节)：画面编号

Control_id(2 个字节)：控件编号

Status(1 个字节) ：按钮状态

0x00：按钮由按下变成弹起状态

0x01：按钮从弹起变成按下状态

该指令主要用于将“按下”的按钮变成“弹起”或将“弹起”的按钮变成“按下”状态。在某些场合中，按钮之间存在互斥性，当某个按钮被按下后，另外一个按钮必须弹起，则可以使用该指令来实现。

例如图 4.5 所示，用户按下“启动运行”按钮后，需要将“停止运行”按钮(画面 ID 为 2，控件 ID 为 4)由“按下”变成“弹起”状态，则单片机发送指令：EE B1 10 00 02 00 04 00 FF FC FF FF。

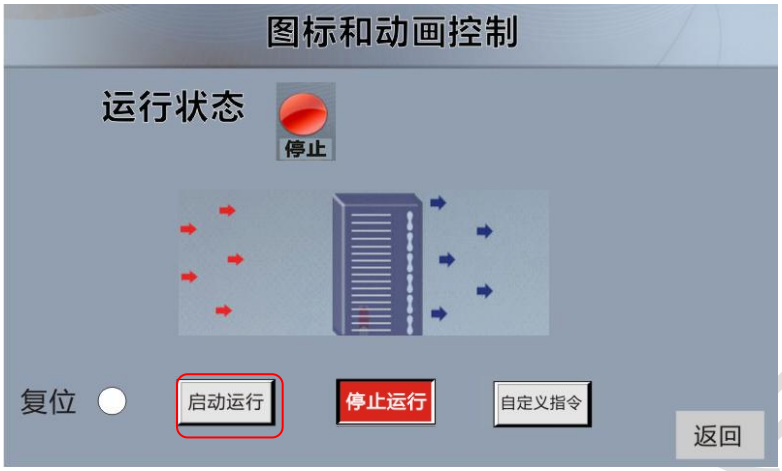


图 4.5 设置按钮按下和弹起状态

程序参考代码：

```
{
  .....                          //检测到启动运行被按下
  SetButtonValue(2,4,0)；// 将画面 2，控制 ID 位 4 的停止运行按钮设置为弹起状态
 }
```

2. 通过 PC 进行按钮互斥配置。

同样如图 4.6 所示，若用户只要按下“启动运行”按钮后，必须将“停止运行”按钮(画面 ID 为 2，控件 ID 为 4)进行弹起，则可以直接在 PC 中进行配置。属性窗口选择：选弹起时对内指令→EE B1 10 00 02 00 04 00 FF FC FF FF，这样就无需单片机再发送额外指令处理。

当然除了可以配置按下“启动运行”弹起停止运行外，还可以在“弹起时对内指令”框内输入多条其它的指令，例如启动 gif 动画，改变运行状态标志等，多条指令用分号进行隔开即可。

图 4.6 按钮互斥配置

4.5 读取按钮控件状态

指令格式：EE 【B1 11 Screen_id Control_id】FF FC FF FF

参数说明：Screen_id(2 个字节)：画面编号

Control_id(2 个字节)：控件编号

该指令主要用于查询某个按钮控件当前是“按下”还是“弹起”状态。

返回指令格式：EE B1 11 Screen_id Control_id Control_type Subtype Status FF FC FF FF

参数说明：Screen_id(2 个字节)：画面编号

Control_id(2 个字节)：控件编号

Control_type(1 个字节)：固定值 0x10，表示为按钮控件

Subtype(1 个字节)：固定值 0x01，表示按钮为开关描述类型

Status (1 个字节)：按钮状态

0x00：按钮为弹起状态

0x01：按钮为按下状态

4.6 更新文本控件数值

文本更新方式有三种：用户主机输入、弹出系统键盘输入和自定义键盘输入。

(1) 用户主机输入。屏幕上显示的数据来自于用户单片机输入。

指令格式：EE【B1 10 Screen_id Control_id Strings 】FF FC FF FF

参数说明：Screen_id(2 个字节)：画面编号

Control_id(2 个字节)：控件编号

Strings（不定长）： 用户写入的字符串

该指令主要用于文本显示。由于系统对所有文本变量分配了内存地址，当屏幕从其它界面再次返回到文本显示时，用户主机无须再次重新刷新数据。

使用文本控件时，用户首先在上位机进行控件相关参数配置，譬如字体、前景色、背景色和文本输入方式等如图 4.7 所示，然后主机只需要直接对相应 ID 写入变化的数据即可。

图 4.7 文本控件参数配置

例如用户需要对第 5 个画面、控件 ID 为 3 处写入数值 20，程序代码如下。

程序参考代码：

```
{
```

```
.....
SetTextValue(5,3,20)；//对画面 5 的第 3 个文本控件写入 20
}
```

(2) 弹出系统键盘输入。当点击文本框时，系统自动弹出内置键盘，用户输入完所需字符后点击确定，设备会将输入的字符显示在文本框内，同时将字符的 ASCII 码上传到主机。

例如，用户预先设置好文本控件，输入方式选择“弹出键盘输入”，如图 4.8 所示。点击文本屏幕将自动弹出系统小键盘，录入“123”后点击 Enter，此时输入的数字将自动显示到文本框内，同时上传录入字符的 ASCII 码，上传格式与下面即将介绍的“读取文本控件数值”返回格式一样。用户单片机解析上传的指令即可知道录入的数据。

图 4.8 弹出系统小键盘

若用户需要录入中英文，只需键盘类型选择全键盘，如图 4.9 所示。注意：因为显示区域的原因，只有 3.5 寸以上（不包括 3.5 寸）的尺寸才支持全键盘和输入法。

图 4.9 弹出系统全键盘显示

(3) 自定义键盘输入。文本框内的数据来自于同一画面内的键盘输入。如图 4.10 所示，若用户需要实现密码输入，操作时先利用按钮控件的自定义键值功能，将每个按钮定义为对应的 ASCII 字符（参考图 4.3 所示），然后在密码框内放置一个文本控件，将属性输入方式设置为“自定义键盘输入”。这样点击文本框后，界面将自动出现光标闪动，然后点击右边键盘输入，对应的数字就自动显示在文本框中，同时输入的字符 ASCII 码也将上传给主机。

提示：配置完毕后，建议运行“虚拟串口屏”，查看按下每个按钮后屏幕上传的信息。

图 4.10 自定义键盘输入

4.7 读取文本控件数值

指令格式：EE 【B1 11 Screen_id Control_id】FF FC FF FF

参数说明：Screen_id(2 个字节)：画面编号

Control_id(2 个字节)：控件编号

该指令主要用于获取当前文本控件的数值。对于一些重要参数，用户可以采用获取文本控件数值命令来重新校验。

返回指令格式：EE B1 11 Screen_id Control_id Control_type Strings FF FC FF FF

返回参数：Screen_id(2 个字节)：画面编号

Control_id(2 个字节)：控件编号

Control_type(1 个字节)：固定值 0x11，表示为文本控件

Strings(不定长)：当前显示的文本值

4.8 更新进度条控件数值

指令格式：EE 【B1 10 Screen_id Control_id Progressvalue】 FF FC FF FF

参数说明：Screen_id(2 个字节)：画面编号

Control_id(2 个字节)：控件编号

Progressvalue（4 个字节）：新的进度条值

该指令主要实现进度的递增或递减。

使用进度条控件时，用户首先在上位机进行控件相关参数配置，譬如前景图、背景图、数值最大值和最小值等，如图 4.11 所示，然后主机只需要对相应的 ID 写入进度条数值即可实现进度条滚动。

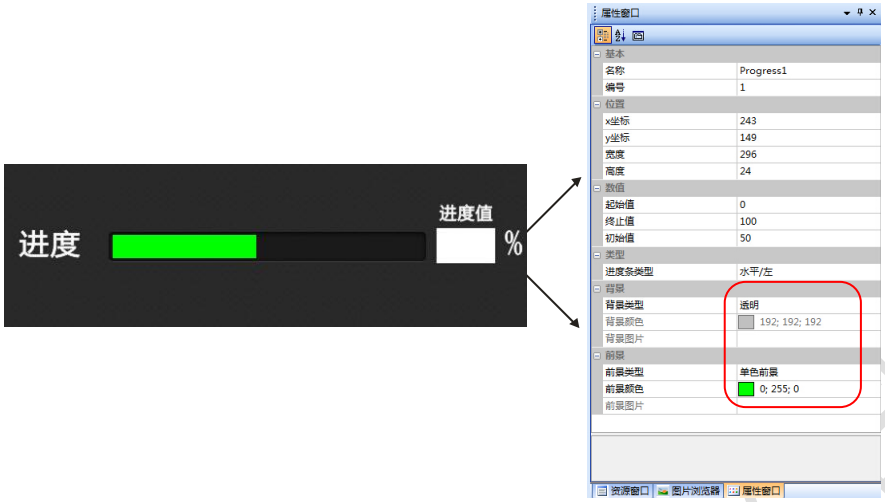


图 4.11 进度条控件配置信息

例如用户需要对第 3 个画面、控件 ID 为 1 的进度条控件写入进度值 50，程序代码如下。
程序参考代码：

```
{
  ......
  SetProgressValue(3,1,0,50)   //对画面 3 的控件 ID 为 1 的进度条控件写入 50
 }
```

4.9 获取进度条控件值

指令格式：EE 【B1 11 Screen_id Control_id】FF FC FF FF
参数说明：Screen_id(2 个字节)：画面编号
Control_id(2 个字节)：控件编号
该指令主要用于获取当前进度条的数值。
返回指令格式：EE B1 11 Screen_id Control_id Control_type Progressvalue FF FC FF FF
返回参数：Screen_id(2 个字节)：画面编号
Control_id(2 个字节)：控件编号
Control_type(1 个字节)：固定值 0x12，表示为进度条控件
Progressvalue(4 个字节)：当前进度条数值

4.10 滑动条控件上传格式

使用滑块控件时，用户首先在上位机进行控件相关参数配置，譬如标尺长度、方向、游标图片、背景图和数值等，如图 4.12 所示，当用户拖动游标时，设备将不断上传当前游标数值给主机，通过判断游标数值即可知道当前滑块所在位置。

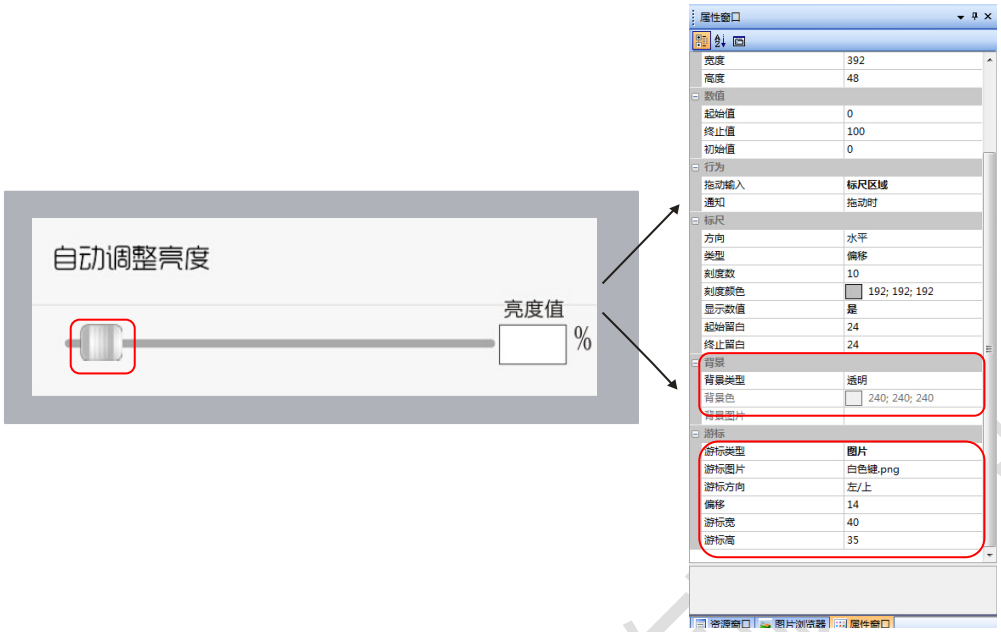


图 4.12 滑块控件配置图

滑块控件上传格式：

EE 【B1 11 Screen_id Control_id Control_type Slidervalue】FF FC FF FF

参数说明：Screen_id(2 个字节)：画面编号

Control_id(2 个字节)：控件编号

Control_type(1 个字节)：固定值 0x13，表示为滑块控件

Slidervalue (4 个字节)：表示当前游标数值

4.11 更新滑动条控件数值

指令格式：EE 【B1 10 Screen_id Control_id Slidervalue】 FF FC FF FF

参数说明：Screen_id(2 个字节)：画面编号

Control_id(2 个字节)：控件编号

Slidervalue（4 个字节）：新的游标数值

该指令主要用于控制滑块游标显示的位置。用户主机可以发送相应的指令控制游标强制在某一个位置显示。

例如用户需要对第 2 个画面、控件 ID 为 5 的滑动条控件写入数值 50，程序代码如下。

程序参考代码：

```
{
  .....
  SetSliderValue (2,5,0,50) //对画面 2 的控件 ID 为 5 的滑块控件写入数值 50
}
```

4.12 读取滑动条控件值

指令格式：EE 【B1 11 Screen_id Control_id】FF FC FF FF

参数说明：Screen_id(2 个字节)：画面编号

Control_id(2 个字节)：控件编号

该指令主要用于获取当前游标所在的数值。

返回指令格式：EE B1 11 Screen_id Control_id Control_type Slidervalue FF FC FF FF

返回参数：Screen_id(2 个字节)：画面编号

Control_id(2 个字节)：控件编号

Control_type(1 个字节)：固定值 0x13，表示为滑块控件

Slidervalue（4 个字节）：表示当前游标数值

4.13 更新仪表控件数值

指令格式：EE 【B1 10 Screen_id Control_id Metervalue】 FF FC FF FF

参数说明：Screen_id(2 个字节)：画面编号

Control_id(2 个字节)：控件编号

Metervalue（4 个字节）：新的仪表值

使用仪表控件时，用户首先在上位机进行控件相关参数配置，譬如表盘、刻度、指针和数值等，如图 4.13 所示，然后主机只需要发送相应的数值就可以实现仪表指针的转动。

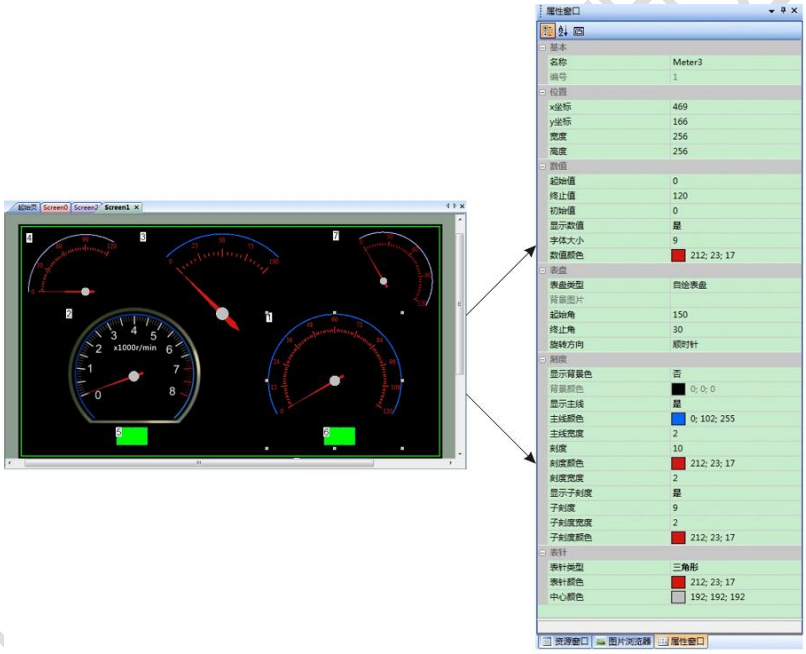


图 4.13 仪表控件配置信息

例如用户需要对第 0 个画面、控件 ID 为 4 的仪表控件写入数值 100，程序代码如下。

程序参考代码：

```
{
  ......
  SetMeterValue (0,4,0,100) //对画面 0 的控件 ID 为 4 的仪表控件写入数值 100
 }
```

4.14 读取仪表控件数值

指令格式：EE 【B1 11 Screen_id Control_id】FF FC FF FF

参数说明：Screen_id(2 个字节)：画面编号

Control_id(2 个字节)：控件编号

该指令主要用于获取当前仪表显示的数值。

返回指令格式：EE B1 11 Screen_id Control_id Control_type Metervalue FF FC FF FF
返回参数：Screen_id(2 个字节)：画面编号
Control_id(2 个字节)：控件编号
Control_type(1 个字节)：固定值 0x14，表示为仪表控件类型
Metervalue（4 个字节）：当前显示的仪表值

4.15 动画控件显示

通过动画控件调用 gif 动画显示，可在同一画面内支持多个 gif 显示，用户发送相应的指令就可以控制动画播放、停止、暂停和上/下帧等功能，指令介绍如表 4.1 所示。

表 4.1 动画控件指令表

功能	指令格式
启动动画播放	EE 【B1 20 Screen_id Control_id】FF FC FF FF
	Screen_id(2 个字节)：画面编号 Control_id(2 个字节)：控件编号 启动播放后，动画每次从帧头 0 开始播放
停止动画播放	EE 【B1 21 Screen_id Control_id】FF FC FF FF
	停止播放后，下次将从帧头 0 开始播放
暂停动画播放	EE 【B1 22 Screen_id Control_id】FF FC FF FF
	暂停后，下次将从暂停帧开始继续播放
指定帧播放	EE 【B1 23 Screen_id Control_id】FF FC FF FF
	指定从某一帧开始播放
播放上一帧	EE 【B1 24 Screen_id Control_id】FF FC FF FF
	显示当前帧的上一帧内容
播放下一帧	EE 【B1 25 Screen_id Control_id】FF FC FF FF
	显示当前帧的下一帧内容
动画控件值上传	EE 【B1 26 Screen_id Control_id Status FlashImgae_ID】 FF FC FF FF
	按下某个动画控件时候，设备上传的如下信息： Status (1 个字节)：0x00 表示触摸按下，0x01 表示弹起 FlashImgae_ID (1 个字节)：表示按下屏幕时，此刻显示的画面帧 ID 备注：用户可以在 PC 软件配置禁止/使能动画控件值上传。

使用时，用户首先通过动画控件调用 gif 动画进行显示，然后设置是否需要触摸通知(也就是点击动画是否需要上传动画控件 ID 值)和自动播放，如图 4.14 所示。动画播放的间隔时间自动从 gif 原文件中提取。

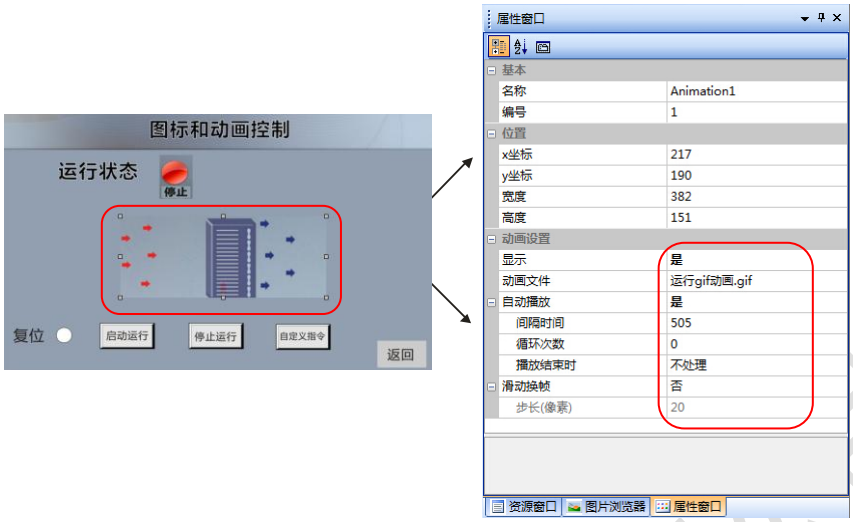


图 4.14 动画控件配置图

4.16 图标控件显示

指令格式：EE 【B1 23 Screen_id Control_id IconImgae_ID】FF FC FF FF

参数说明：Screen_id(2 个字节)：画面编号

Control_id(2 个字节)：控件编号

IconImgae_ID (1 个字节)：ICON 文件中某一图标帧 ID

该指令主要解决画面内同一位置不同状态图的切换或消失显示。

如何生成 ICON 图标？使用前用户先点击 VisualTFT 软件的工具菜单，选择→图标生成，弹出如图 4.15 所示的画面，然后将预先做好的 3 张 62x82 像素图片(停止、运行和透明 PNG)添加进去，按照排列的顺序，分别第 1 帧为停止帧、第 2 帧为运行和第 3 帧为透明图片，最后点击生成图标。这样一个新的 ICON 文件就生成了，里面包含了 3 帧图片。

图 4.15 图标生成工具

使用时，用户首先通过图标控件调用新生成的 ICON 进行显示，然后设置是否需要触摸通知(也就是点击图标控件是否需要上传 ID)和自动播放，如图 4.16 所示。画面默认显示 ICON 的第 1 帧，若需要切换状态图显示，主机程序只需发送 ICON 编号和第几帧即可完成显示。

透明帧其实就是一个空的 PNG 图片，显示透明帧就可以达到图标消失或隐藏的效果。

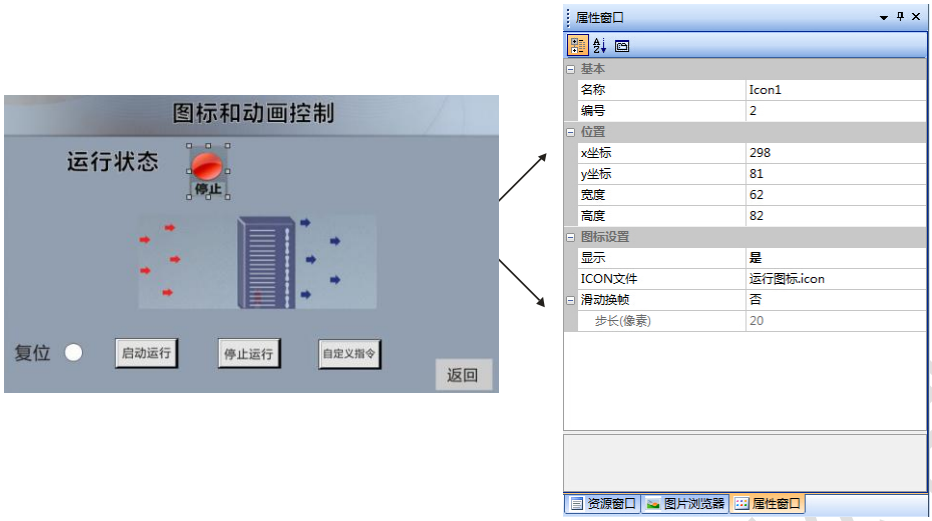


图 4.16 图标控件配置图

4.17 图标控件值上传

指令上传格式：EE 【B1 26 Screen_id Control_id Status IconImgae_ID】FF FC FF FF

参数说明：Screen_id(2 个字节)：画面编号

Control_id(2 个字节)：控件编号

Status (1 个字节)：0x00 表示触摸按下，0x01 表示弹起；

IconImgae_ID (1 个字节)：ICON 文件中某一图片帧 ID

若在 PC 软件选择了图标控件触摸通知，用户点击图标后，设备将上传给图标控件的 ID 和当前图片帧的 ID。

4.18 曲线控件显示

若需要实现动态图表数据采集，自动左右平移，曲线控件将给用户带来极大便捷。指令介绍如表 4.2 所示，上位机软件配置图如图 4.17 所示。

表 4.2 曲线控件指令表

功能	指令格式
添加数据通道	EE 【B1 30 SCREEN_ID CONTROL_ID CHANNEL COLOR】FF FC FF FF
	指定数据通道和曲线颜色 Screen_id(2 个字节)：画面编号 Control_id(2 个字节)：控件编号 CHANNEL(1 个字节)：共 8 个数据通道，编号范围（0~7） COLOR(2 个字节)：数据通道颜色
删除指定数据通道	EE 【B1 31 SCREEN_ID CONTROL_ID CHANNEL】FF FC FF FF
	CHANNEL(1 个字节)：共 8 个数据通道，编号范围（0~7）

指定垂直/水平的缩放/平移	EE 【B1 34 SCREEN_ID CONTROL_ID XOFFSET XMUL YOFFSET YMUL】FF FC FF FF
	指定垂直/水平的缩放/平移 XOFFSET (2 个字节)：水平偏移数据点数，左移为正，右移为负 XMUL (2 个字节)：水平缩放系数，单位 0.01 YOFFSET (2 个字节)：垂直偏移数值，下移为正，上移为负 YMUL (2 个字节)：垂直缩放系数，单位 0.01 采样点与坐标点的计算公式：第 N 个采样点的数值为 V X 坐标 = (N-XOFFSET)*XMUL*0.01 Y 坐标 = (V-YOFFSET)*YMUL*0.01
指定通道末尾添加新数据	EE 【B1 32 SCREEN_ID CONTROL_ID CHANNEL DATA_LEN DATA】FF FC FF FF
	在指定数据通道末尾添加新数据，当数据长度超过缓冲区长度时，旧数据左移。 CHANNEL(1 个字节)：共 8 个数据通道，编号范围（0~7） DATA_LEN(2 个字节)：数据长度 DATA ：不定长数据，长度由 DATA_LEN 指定 格式：数据代表是 Y 轴方向的值，X 轴方向的会根据水平缩放系数自动递增，例如当水平缩放系数为 1 的时候，每插入一个点 X 轴自动加 1，当水平缩放系数为 5 的时候，每插入一个点 X 轴自动加 5
指定通道前端添加新数据	EE 【B1 35 SCREEN_ID CONTROL_ID CHANNEL DATA_LEN DATA】FF FC FF FF
	在指定数据通道最前端插入新数据，当数据长度超过缓冲区长度时，旧数据右移 CHANNEL(1 个字节)：共 8 个数据通道，编号范围（0~7） DATA_LEN(2 个字节)：数据长度 DATA ：不定长数据，长度由 DATA_LEN 指定
清空指定通道的数据	EE 【B1 33 SCREEN_ID CONTROL_ID CHANNEL】FF FC FF FF
	清空指定通道的数据 CHANNEL(1 个字节)：共 8 个数据通道，编号范围（0~7）

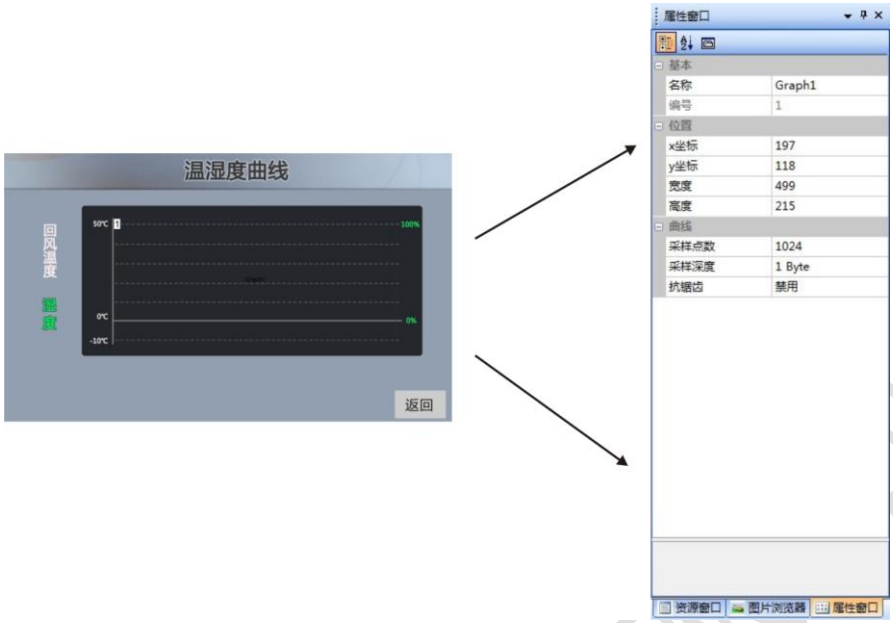


图 4.17 曲线控件配置图

4.19 设置光标焦点

指令格式：EE 【B1 02 Screen_id Control_id Enable】FF FC FF FF

参数说明：Screen_id(2 个字节)：画面编号

Control_id(2 个字节)：控件编号

Enable (1 个字节)：0x00 表示关闭，0x01 表示开启显示；

该指令主要将光标符号移植到指定的文本控件上显示，更直观提醒用户录入文本。

提示：仅在自定义键盘中有效。

4.20 手动禁止/使能屏幕更新

指令格式：EE 【B3 Enable】FF FC FF FF

参数说明： Enable (1 个字节)：0x00 表示禁止更新，0x01 表示使能更新；

该命令主要解决某一画面中实时动态更新的控件数目过多，导致屏幕更新速度慢的问题。由于系统默认 100ms 自动刷新屏幕一次，而有些场合用户数据量庞大，系统自动更新无法满足时时性要求，所以可以采用手动更新屏幕指令。

使用方法：程序先发送禁止屏幕更新指令，然后发送整个画面中需要更新的内容，最后再使能屏幕更新，这样更新的数据立刻显示在屏幕上。

4.21 屏蔽/隐藏控件

指令格式：EE 【B1 03 Screen_id Control_id Enable】FF FC FF FF

参数说明：Screen_id(2 个字节)：画面编号

Control_id(2 个字节)：控件编号

Enable (1 个字节)：0x00 表示隐藏控件，0x01 表示显示控件；

该指令常用于某一时刻将指定按钮控件功能失效，直至屏蔽被解除，也可用于将某个动画/图标控件显示消失。

第5章 用户单片机程序

5.1 用户单片机程序参考范例

本节主要简要介绍用户程序开发的一般流程，详细的程序代码可以在官网 www.gz-dc.com 进行下载。

1. 准备配套的美工素材，如图 5.1 所示。

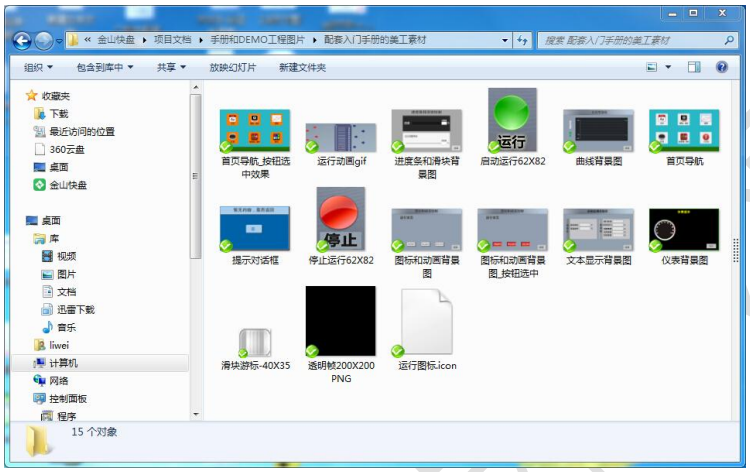


图 5.1 配套的美工素材

2. 使用 PC 软件进行界面配置。预先利用上位机 VisualTFT 对每个画面进行配置，然后运行“虚拟串口屏”进行仿真，确保每个界面按钮逻辑关系的正确性，最后将工程下载到串口屏中。上位机软件如图 5.1 所示。

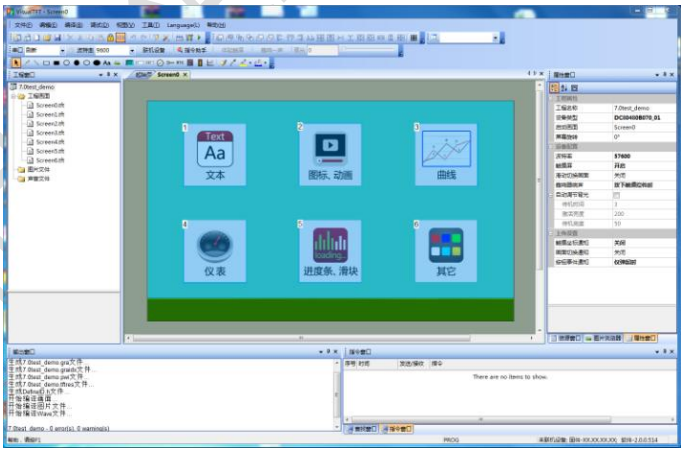


图 5.2 使用 PC 进行界面配置

3. 用户程序结构分析

将配置好的工程下载到串口屏后，用户单片机就可以接收或发送指令控制显示了。我司提供的单片机(51、STM32、AVR 等)程序驱动架构如图 5.3 所示。其中最下层为用户处理器串口的硬件驱动，往上一层为为串口屏的命令帧驱动，再上面一层是串口消息响应处理函数，最上面为用户应用代码。

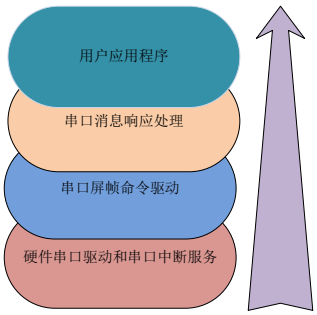


图 5.3 程序结构图

4. 程序工作流程图。整个串口屏工作流程如所示，无论用户当前使用什么 MCU 平台，只需替换串口底层的收发函数即可。

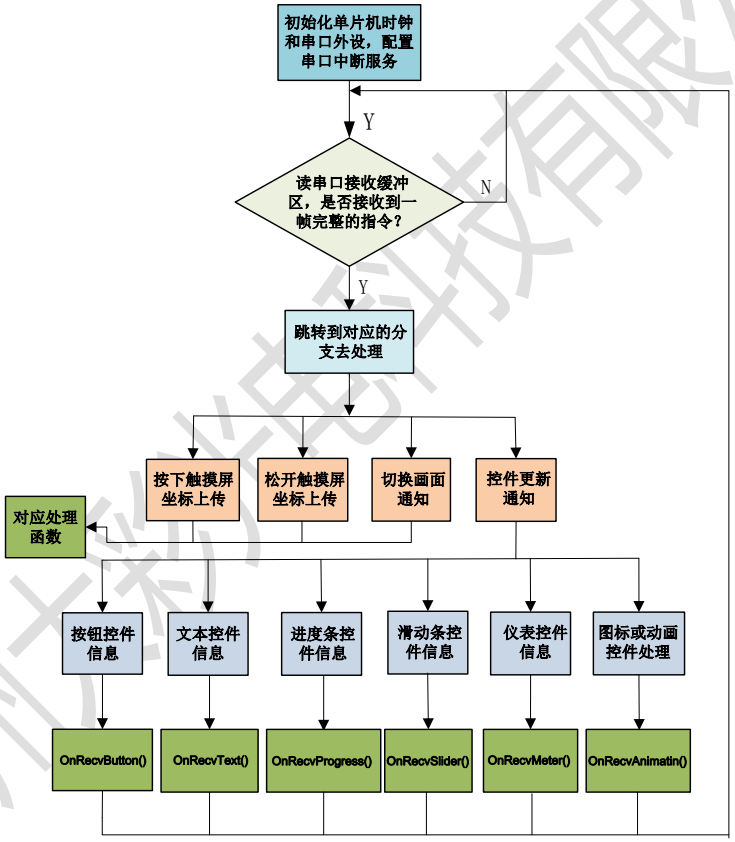


图 5.4 程序工作流程图

5. Keil 程序参考模板。我司已提供标准的串口屏开发平台模板，用户只需在对应的消息事件中添加应用程序即可，如图 5.5 所示。

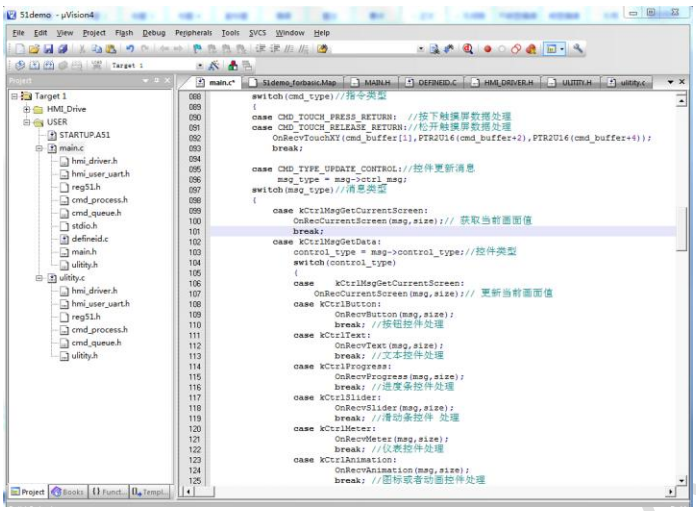


图 5.5 KEIL 程序模板

程序清单 6.1

```
 switch(cmd_type)//指令类型
{
case CMD_TOUCH_PRESS_RETURN:     //按下触摸屏坐标上传
case CMD_TOUCH_RELEASE_RETURN://松开触摸屏坐标上传
      OnRecvTouchXY(cmd_buffer[1],PTR2U16(cmd_buffer+2),PTR2U16(cmd_buffer+4));
      break;
case CMD_TYPE_UPDATE_CONTROL://控件更新消息
     msg_type = msg->ctrl_msg;
      switch(msg_type)//消息类型
      {
                                case kCtrlMsgGetCurrentScreen:
                                      OnRecCurrentScreen(msg,size);//  获取当前画面值
                                      break;
                                case kCtrlMsgGetData:
                                      control_type = msg->control_type;//控件类型
                                      switch(control_type)
                                      {
                                      case   kCtrlMsgGetCurrentScreen:
                                            OnRecCurrentScreen(msg,size);//  更新当前画面值
                                      case kCtrlButton:
                                                  OnRecvButton(msg,size);
                                                  break; //按钮控件处理
                                      case kCtrlText:
                                                  OnRecvText(msg,size);
```

```
                    break; //文本控件处理
                case kCtrlProgress:
                    OnRecvProgress(msg,size);
                    break; //进度条控件处理
                case kCtrlSlider:
                    OnRecvSlider(msg,size);
                    break; //滑动条控件  处理
                case kCtrlMeter:
                    OnRecvMeter(msg,size);
                    break; //仪表控件处理
                case kCtrlAnimation:
                    OnRecvAnimation(msg,size);
                    break; //图标或者动画控件处理
                default:
                    break;
                }
            break;
    }
```

6. 使用 Keil 与虚拟串口屏绑定连调。为了更方便工程师开发，避免每次修改界面都需要重新下载图片到设备，用户可以直接将 KEIL 与虚拟串口屏绑定，轻松实现单步、多步 Debug 调试，节省开发时间，如图 5.6 所示。

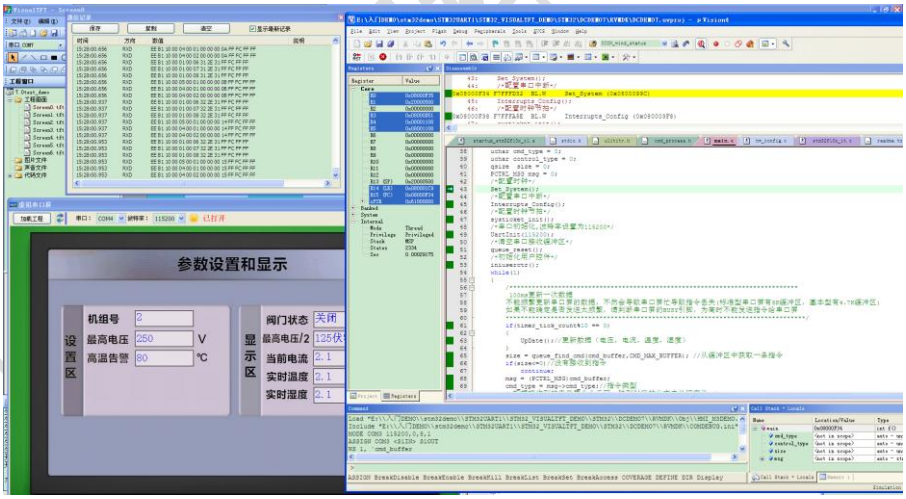


图 5.6 虚拟串口屏与 KEIL 绑定调试

7. 详细的开发流程和程序范例请参考《串口屏快速入门文档》。

第6章 附录 A 基本指令集详述

由于组态指令集可以满足 95%的工程应用，而且操作简单，大部分配置都在 PC 中完成，因此我们并不推荐用户使用基本指令集。相对组态指令集，基本指令集属于最底层指令，灵活性更强，在组态指令无法完成的情况下，可以使用基本集指令来实现。详细的指令集说明如下列章节所述。

6.1 握手

指令格式：EE【04】FF FC FF FF
设备返回：EE 55 FF FC FF FF
握手指令主要用于判断设备是否上电初始化完毕、通信是否正常和是否在线状态等。发送指令后设备返回 55 表示握手成功。

6.2 复位报告

指令格式：无
设备返回：EE 07 FF FC FF FF
一旦设备上电启动、意外重启或监控芯片复位，将立即上传相关数据，告知用户设备已被复位。用户主机检测到设备意外复位后，需控制程序重新从头初始化执行。

6.3 复位设备

指令格式：EE【07 35 5A 53 A5】FF FC FF FF
设备返回：EE 07 FF FC FF FF
主机在运行的过程中通过串口指令来复位设备。建议主机初始化设备时增加该指令，以便主机意外复位后，设备也跟着复位。

6.4 设置前/背景色

指令格式： 设置前景色：EE【41 Fcolor】FF FC FF FF
设置背景色：EE【42 Bcolor】FF FC FF FF
设置前景和背景色：EE【40 Fcolor Bcolor】FF FC FF FF
参数说明：Fcolor、Bcolor (2 个字节)分别是前景色和背景色的 RGB 值
前景色主要用于指定文本、点、线和圆等显示的颜色，背景色主要用于清屏和文字底色颜色的指定。比如通过设置前/背景色的指令可完成显示图 6.1 的所示内容。
参考程序：

```
{
    SetBcolor(31);          //设置背景色为蓝色，用于指定清屏颜色
    GUI_CleanScreen( );   //清屏
    SetFcolor(65516);      //设置前景色为黄色，用户指定文本显示颜色
    SetBcolor(63488);      //设置背景色为红色，用于指定文本背景色（底色）
    DisText(50, 50, 1, 6, "你好，色彩！");  //在坐标(50,50)写入字符串，有背景色
    DisText(50, 90, 0, 6, "你好，串口屏！");  //在坐标(50,90)写入字符串，无背景色
    SetFcolor(2016);                  //重新设置前景色为绿色
    GUI_RectangleFill(256, 57, 370, 116);      //画实心矩形
}
```

图 6.1 背景前景色说明

6.5 清屏

指令格式：EE【01】FF FC FF FF

参数说明：无参数

该指令用于实现指定颜色清屏，清屏颜色取决背景色的设置。若用户未进行背景色设置，清屏色默认为蓝色。

6.6 设置文字行列间距

指令格式：EE【43 Y_W X_W】FF FC FF FF

参数说明：Y_W(1 个字节) 是以点为单位的行间距，取值范围 00~3F；
X_W(1 个字节) 是以点为单位的列间距，取值范围 00~3F。

该指令用于设置文字之间的行列距。如果文本只有一行，行间距则为 0。例如在屏幕上显示 2 行 32*32 的字符串，设置行间距 24，列间距 16，程序如下所示。

参考程序：

```
{
    SetFcolor(65504);       //设置前景色为黄色，指定文本显示颜色
    SetBcolor(63488);       //设置背景色为红色，指定文本底色颜色
    SetTextSpace(24,16);    //设置文本行间距为 24，列间距为 16
    DisText(50, 51, 1, 7, "广州大彩科技工业串口屏");  //显示文本字符串
}
```

6.7 设置文本框

指令格式：EE【45 Enable Width Hight】FF FC FF FF

参数说明： Enable(1 个字节)：打开/关闭文本框限制使能
Width (2 字节): 文本框宽度；
Hight (2 字节): 文本框的高度。

设置文本框后，文字将在限定框内自动换行显示。

6.8 设置图片过滤色

指令格式：EE【44 FillColor】FF FC FF FF

参数说明： FillColor (2 字节): 过滤色 RGB 值；

设定过滤色后，当图片某一像素值正好与过滤色值相同，该点就会被屏蔽，无法在屏幕上显示。设置过滤色前后对比如图 6.2 所示。

参考程序：

```
{
```

```
    DisArea_Image(0, 0, 0, 0);          // (0,0)处显示草原背景图
    DisArea_Image(61,130, 1, 0);        // (61,130)处显示未过滤的蝴蝶图案
    SetFilterColor(65535);              //设置白色为过滤色，RGB 值为 65535
    DisArea_Image(258,68, 2, 1);        // (258,68)处显示过滤后的蝴蝶图案
}
```

图 6.2 设置过滤色前后对比图

6.9 文本显示

指令格式：EE【20 X Y Back Font Strings】FF FC FF FF

参数说明：X (2 字节) :以点为单位的 X 轴坐标值；

Y (2 字节) :以点为单位的 Y 轴坐标值；

Back (1 个字节)：背景色使能

0x01:背景色显示 0x00：背景色不显示

Font (字库编码,1 个字节)

0x00: 8x12 点阵 (ASCII) 0x01: 8x16 点阵(ASCII)

0x02: 12x24 点阵(ASCII) 0x03: 16x32 点阵(ASCII)

0x04: 12 x 12 点阵(GBK) 0x05: 16 x 16 点阵（GBK)

0x06: 24 x 24 点阵(GBK)

0x07: 32 x 32 点阵(GB2312)

0x08: 32 x 64 点阵(ASCII)

0x09: 64 x 64 点阵(GB2312)

Strings: 用户写入的字符串，高字节在前。

该指令用于实现在屏幕任意位置显示指定的文本。GBK 包含了汉字及日韩等常用字库；GB2312 只包含汉字和字符；ASCII 不能显示汉字。在实际操作中，用户确定文本的前景色、背景色、字库编码后，可以连续写入汉字或字符串，设备将会自动换行及中英文匹配显示。文本显示效果图 6.3 所示。

参考程序：

```
{
  SetFcolor(65504);       //设置文字前景色，黄色
  DisText(46, 21, 0, 7, "工业串口屏 LCM 32*32");  //坐标(46,21)处显示文字，字库为 7 号字体
```

图 6.3 文本显示效果图

6.10 光标显示

指令格式： EE【21 Enable X Y Width Hight】FF FC FF FF

参数说明： Enable(1 个字节) ： 光标使能信号

0x00：光标关闭 ，0x01：光标开启

X (2 字节) :以点为单位的 X 轴坐标值

Y (2 字节) :以点为单位的 Y 轴坐标值

Width (1 个字节) ： 光标宽度

Hight (1 个字节) ： 光标高度

该指令主要用来控制光标的闪烁和关闭。例如用户在 24*24 的汉字尾缀显示宽度 16、高度 8 的光标，效果图如 图 6.4 所示。

程序参考代码：

```
{
  SetBcolor(31);              // 设置蓝色背景色
  GUI_CleanScreen( );         // 背景清屏蓝色
  DisCursor(1,359,40,16,8);   //光标闪烁使能，在(359,40)处显示宽度 16 高度 8 的光标
}
```

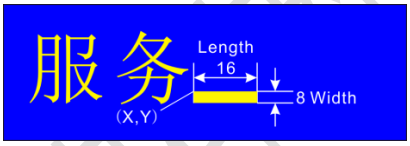


图 6.4 光标参数说明

6.11 全屏图片显示

指令格式：EE【31 Image_ID MaskEn】FF FC FF FF

参数说明：Image_ID (2 个字节)：图片编号

MaskEn (1 个字节)：过滤色使能

0x00:颜色不过滤； 0x01 执行颜色过滤

该指令主要显示全屏图片，起始坐标固定（0, 0）处，被过滤的颜色取决于过滤色设置。若图片尺寸小于全屏大小，执行该指令时候，只显示实际大小的图片。

6.12 区域图片显示

指令格式： EE【32 X Y Image_ID MaskEn】FF FC FF FF

参数说明： X (2 字节)：以点为单位的 X 轴坐标值

Y (2 字节)：以点为单位的 Y 轴坐标值

Image _ID(2 个字节)：图片编号

MaskEn (1 个字节)：过滤色使能

0x00:颜色不过滤； 0x01 执行颜色过滤

该指令用于实现任意位置图片显示。用户需要注意起始坐标和图片长宽，防止显示画面超出屏幕范围。例如用户要在屏幕（100,50）处显示一张 ID 为 2 的图片，如图 6.5 所示， 参考程序如下所示。

图 6.5 区域图片显示

程序参考代码：

```
{
  DisFull_Image(0,0);                 // 整屏图片显示
  DisArea_Image(100,50, 2, 0);        // 区域图片显示，坐标（100,50）处显示 ID=2 的图片
 }
```

6.13 图片剪切

指令格式：EE【33 X Y Image_ID Image_X Image_Y Image_W Image_H MaskEn】FF FC FF FF

参数说明： X (2 字节) :以点为单位的 X 轴坐标值
Y (2 字节) :以点为单位的 Y 轴坐标值
Image_ID (2 字节): 要剪切的图片编号
Image_X (2 字节): 被剪切的图片起点 X 坐标
Image_Y (2 字节): 被剪切的图片起点 Y 坐标
Image_W (2 字节) : 剪切的宽度
Image_H (2 字节) : 剪切的高度
MaskEn (1 个字节) 0x00:颜色不过滤 ;0x01 执行颜色过滤
备注：被过滤的颜色取决于过滤色的设置

该指令用于实现在屏幕任意坐标处显示被剪切过来的图片。用户可以对储存在 Flash 中的任意一张图片进行局部裁剪，剪切效果如图 6.6 所示。

6.14 动画显示

指令格式：EE【80 X Y FlashImage_ID Enable PlayNum 】FF FC FF FF

参数说明： X (2 字节) : 以点为单位的 X 轴坐标值
Y (2 字节) : 以点为单位的 Y 轴坐标值
FlashImage_ID (2 字节) : 动画编号
Enable(1 个字节) ： 使能信号
0x00：关闭动画播放； 0x01：开启动画播放
PlayNum(1 个字节)
0x00：重复播放； 0x01~0xFF：播放指定次数

播放停止后，设备返回 EE 02 FF FC FF FF 表示动画播放结束。

该指令用于实现任意位置 gif 动画的显示。动画只支持 gif 格式, 不支持同一画面两个以上的 gif 动画同时播放。如果期望一个画面同时播放多个 gif 动画，以及动画暂停、停止、播放上/下帧等功能，请使用动画控件指令，详情见组态控制指令表。图片动画显示效果如

图 6.6 所示。

程序参考代码：

```
{
DisFull_Image(0 , 0);                          // 全屏显示小孩图片，编号为 0
DisFlashImage(330,5, 1,1,0);                   // 坐标(330,5)处插入奶牛 Flash 动画,重复播放
DisCut_Image(343,137,0,95,30,92,116,0);        // 从图片 0 坐标(95,30)处剪切大小 92 x116
                                               // 的图片，放置在屏幕(343,137)处显示
}
```

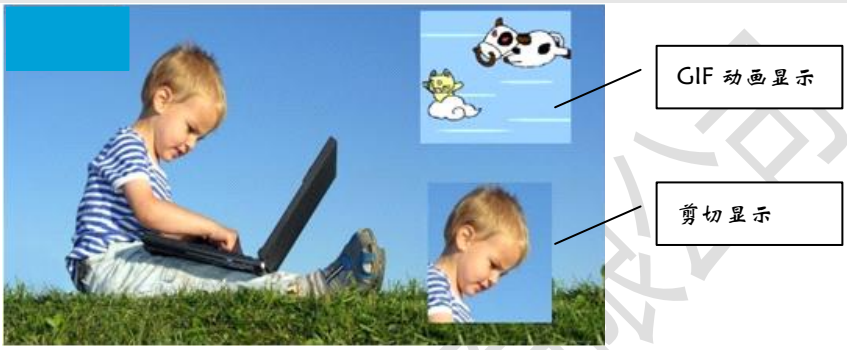


图 6.6 图片动画显示效果图

6.15 前景色画点

指令格式： EE【 50 X Y 】FF FC FF FF

参数说明： X (2 字节)：以点为单位的 X 轴坐标值

Y (2 字节)：以点为单位的 Y 轴坐标值

该指令主要实现在屏幕的任意位置画点，点的颜色值取决前景色的设置。

6.16 背景色画点（删除点）

指令格式： EE【 58 X Y 】FF FC FF FF

参数说明： X (2 字节)：以点为单位的 X 轴坐标值

Y (2 字节)：以点为单位的 Y 轴坐标值

该指令主要实现在屏幕的任意位置画点，点的颜色值取决背景色的设置。背景色画点通常配合前景色画点使用，可以用删除前景色画的点。

如图 6.7 所示，用户可以通过前景色画点在黑色背景图上显示一个黄色五角星图案，若要修改或删除此图案，可以通过调用背景色画点指令去覆盖同一坐标处前景色写入的点。当然，若用户对数据更新速度要求不太高，也可以先区域清屏，然后重新再绘制。

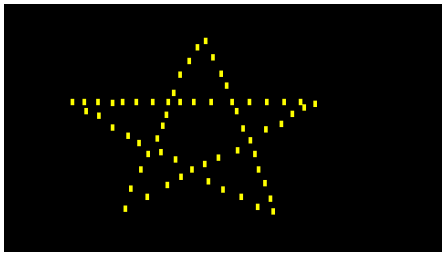


图 6.7 前景/背景色画点

6.17 画线

指令格式： EE【 51 X_0 Y_0 X_1 Y_1 】FF FC FF FF

参数说明： X_0 (2 字节) :以点为单位的直线 X 轴起点坐标值

Y_0 (2 字节) :以点为单位的直线 Y 轴起点坐标值

X_1 (2 字节) :以点为单位的直线 X 轴终点坐标值

Y_1 (2 字节) :以点为单位的直线 Y 轴终点坐标值

该指令主要实现在屏幕的任意两点之间画线，线的颜色值取决前景色的设置。参数说明如图 6.8 所示。例如通过调用画线指定实现一个简易表格，实际显示效果如图 6.8 所示。

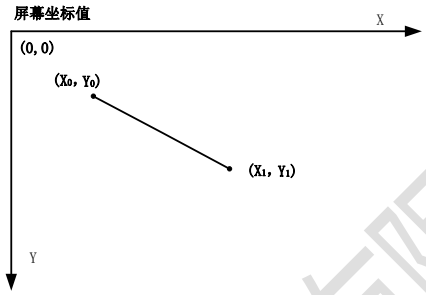


图 6.8 画线参数说明

6.18 将等间隔 X 坐标用前景色连接

指令格式： EE【 59 X_0 Xspace Y_0 Y_1 Y_2 … Y_n 】FF FC FF FF

参数说明： X0 (2 个字节) ： 以点为单位的 X 轴坐标值

Xspace(2 个字节)：以点为单位的 X 轴相邻两点固定的间隔值

Yn (2 个字节) ： 以点为单位的 Y 轴坐标值

该指令主要实现快速绘制折线。由于 X 轴前后两点的距离都是固定的 Xspace，所以从第 2 个点开始指令参数都不需要 X 坐标值。对比前景/背景色绘制折线，速度提高了一倍。

6.19 按照坐标偏移量用前景色连线

指令格式： EE【 75 X_0 Y_0 X_{1o} Y_{1o} X_{2o} Y_{2o} … X_{no} Y_{no} 】FF FC FF FF

参数说明： X0 (2 个字节)：以点为单位的 X 轴坐标值

Y0 (2 个字节)：以点为单位的 Y 轴坐标值

Xno(1 个字节) ： 以点为单位的 X 轴偏移量

Yno(1 个字节) ： 以点为单位的 Y 轴偏移量

(X,Y)为第一点的绝对坐标，后面的每一个点分别由前一个点的绝对坐标加上当前偏移量组成。偏移量的最高位为符号位，0 代表正偏移量，1 代表负偏移，最大偏移量值为正负 127 个点。

该指令主要利用坐标偏移量将原来每个坐标的 4 个字节变为 2 个字节，减少一半的指令参数，达到绘制折线速度提高一倍。线的颜色由前景色设置决定。

6.20 将指定的坐标点用前景色连接

指令格式： EE【68 X_0 Y_0 X_1 Y_1 … X_n Y_n】FF FC FF FF

参数说明： X_n (2 字节) :以点为单位的直线 X 轴起点坐标值

Y_n (2 字节) :以点为单位的直线 Y 轴起点坐标值

该指令主要实现用前景色将指定的多个坐标点连接起来。例如要实现图 6.9 所示的折线和六边形形状，程序如下所示。

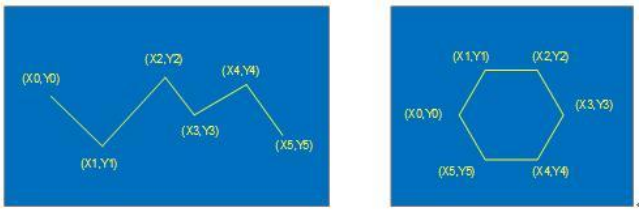


图 6.9 指定坐标点用前景色连接效果图

程序参考代码：

```
{
  SetFcolor(65523);                    //设置线的前景色为黄色
  GUI_FcolorConDots ( x0,y0,x1,y1,x2,y2,x3,y3,x4,y4,x5,y5); // 绘制(x0,y0)到(x5,y5)的折线
  GUI_FcolorConDots (x0,y0,x1,y1,x2,y2,x3,y3,x4,y4,x5,y5,x0,y0); // 绘制六边形，首尾相连
}
```

6.21 将指定的坐标点用背景色连接

指令格式： EE【69 X_0 Y_0 X_1 Y_1 … X_n Y_n】FF FC FF FF

参数说明： X_n (2 字节) :以点为单位的直线 X 轴起点坐标值

Y_n (2 字节) :以点为单位的直线 Y 轴起点坐标值

该指令主要实现用背景色将指定的多个坐标点连接起来。背景色绘制折线通常配合前景色绘制折线使用，可以用删除前景色绘制的折线。

6.22 按照坐标偏移量用背景色连线

指令格式： EE【 76 X_0 Y_0 X_{1o} Y_{1o} X_{2o} Y_{2o} … X_{no} Y_{no} 】FF FC FF FF

参数说明： X0 (2 个字节)：以点为单位的 X 轴坐标值

Y0 (2 个字节)：以点为单位的 Y 轴坐标值

Xno(1 个字节) ：以点为单位的 X 轴偏移量

Yno(1 个字节) ：以点为单位的 Y 轴偏移量

(X,Y)为第一点的绝对坐标，后面的每一个点分别由前一个点的绝对坐标加上当前偏移量组成。偏移量的最高位为符号位，0 代表正偏移量，1 代表负偏移，最大偏移量值为正负 127 个点。

该指令主要利用坐标偏移量将原来每个坐标的 4 个字节变为 2 个字节，减少一半的指令参数，达到绘制折线速度提高一倍。线的颜色由背景色设置决定。

6.23 画空心圆

指令格式： EE【 52 X_0 Y_0 R 】FF FC FF FF

参数说明： X_0 (2 个字节) : 以点为单位的圆心 X 坐标值

Y_0 (2 个字节) : 以点为单位的圆心 Y 坐标值

R (2 个字节) : 空心圆的半径

该指令用于实现在指定的坐标处画一个半径 R 空心圆，圆的线条颜色取决前景色的设置。参数说明如图 6.10 所示。

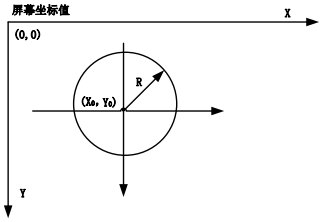


图 6.10 画空心圆参数说明

6.24 画实心圆

指令格式： EE【 53 X_0 Y_0 R 】FF FC FF FF

参数说明： X_0 (2 字节)： 以点为单位的圆心 X 坐标值

Y_0 (2 字节)： 以点为单位的圆心 Y 坐标值

R (2 字节)： 实心圆的半径

该指令用于实现在指定的坐标处画一个半径 R 实心圆，圆内填充色取决前景色的设置。参数说明与图 6.10 相同。

6.25 画圆弧

指令格式： EE【 67 X_0 Y_0 R SA EA】FF FC FF FF

参数说明： X_0 (2 字节)： 以点为单位的圆心 X 坐标值

Y_0 (2 字节)： 以点为单位的圆心 Y 坐标值

R (2 字节)： 圆的半径

SA (2 字节)： 起始角度

EA (2 字节)： 结束角度

该指令用于实现在指定的坐标处画一个半径 R 的圆弧，弧线颜色取决前景色的设置。钟表 3 点方向为起始角度 0 度，顺时针方向角度依次增加，参考坐标如图 6.11 所示。

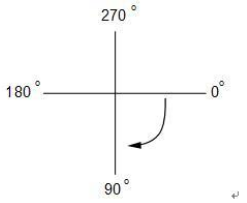


图 6.11 圆弧起始角度参考图

6.26 画空心矩形

指令格式： EE【 54 X_0 Y_0 X_1 Y_1 】FF FC FF FF

参数说明： X_0 (2 字节) :以点为单位的空心矩形左上角 X 坐标值

Y_0 (2 字节) :以点为单位的空心矩形左上角 Y 坐标值

X_1 (2 字节) :以点为单位的空心矩形右下角 X 坐标值

Y_1 (2 字节) :以点为单位的空心矩形右下角 Y 坐标值

该指令用于实现在屏幕任意位置画一个空心矩形，矩形边框颜色取决前景色的设置。参数定义如图 6.12 所示。

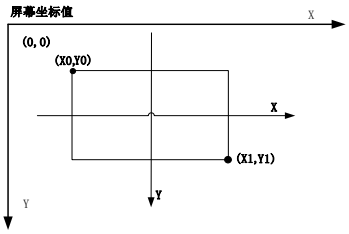


图 6.12 画空心矩形参数说明

6.27 画实心矩形/局部清屏

指令格式： EE【 55 X_0 Y_0 X_1 Y_1 】FF FC FF FF

参数说明： X_0 (2 字节) :以点为单位的实心矩形左上角 X 坐标值

Y_0 (2 字节) :以点为单位的实心矩形左上角 Y 坐标值

X_1 (2 字节) :以点为单位的实心矩形右下角 X 坐标值

Y_1 (2 字节) :以点为单位的实心矩形右下角 Y 坐标值

该指令用于实现在屏幕任意位置画一个实心矩形，矩形填充色取决前景色的设置。参数定义与图 6.12 相同。该功能还可以作为局部清屏使用。

6.28 画空心椭圆

指令格式：EE【 56 X_0 Y_0 X_1 Y_1 】FF FC FF FF

参数说明： X_0 (2 字节) :以点为单位的空心椭圆最左端 X 坐标值

Y_0 (2 字节) :以点为单位的空心椭圆最上端 Y 坐标值

X_1 (2 字节) :以点为单位的空心椭圆最右端 X 坐标值

Y_1 (2 字节) :以点为单位的空心椭圆最下端 Y 坐标值

该指令用于实现在屏幕任意位置画一个空心椭圆，椭圆边框颜色取决前景色的设置。参数定义说明如图 6.13 所示。

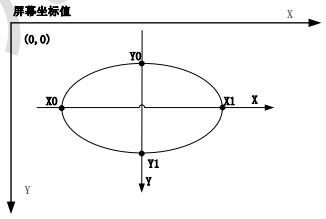


图 6.13 画空心椭圆参数说明

6.29 画实心椭圆

指令格式：EE【 57 X_0 Y_0 X_1 Y_1 】FF FC FF FF

参数说明： X_0 (2 字节) :以点为单位的实心椭圆最左端 X 坐标值

Y_0 (2 字节) :以点为单位的实心椭圆最上端 Y 坐标值

X_1 (2 字节) :以点为单位的实心椭圆最右端 X 坐标值

Y_1 (2 字节) :以点为单位的实心椭圆最下端 Y 坐标值

该指令用于实现在屏幕任意位置画一个实心椭圆，椭圆填充色取决前景色的设置。

6.30 背光调节

指令格式： EE【 60 Light_level 】FF FC FF FF

参数说明： Light_level (1 个字节) ： 背光亮度值

该指令主要用于液晶背光亮度的调节，取值范围 00H～FFH。00H 表示背光最亮，FFH 表示背光最暗，共有 255 级亮度调节。

若屏幕一定时间内无操作动作，建议用户降低背光亮度至 30%左右，以提高背光寿命。

6.31 自动屏保模式

指令格式： EE【77 Enable BL_ON BL_OFF BL_ON_Time 】FF FC FF FF

参数说明： Enable(1 个字节)：使能信号

0x00：关闭省电模式 0x01：开启省电模式

BL_ON(1 个字节)：触摸激活后背光的亮度值

BL_OFF(1 个字节)：进入屏保模式后背光的亮度值

BL_ON_Time(2 个字节)：无触摸动作时，进入屏保模式的时间(单位: 1 秒)

该指令主要用于设置屏保模式被激活和进入屏保模式的背光亮度值。省电模式不仅可以延长液晶屏的背光寿命，还可以降低液晶发光管对外散发的热量。

6.32 蜂鸣器控制

指令格式： EE【 61 Time 】FF FC FF FF

参数说明： Time (1 个字节): 蜂鸣器讯响的时间，单位 10ms

该指令用于蜂鸣器的控制，通过设定 Time 参数实现不同频率的讯响。一般触摸讯响时间 Time 设置为 100ms。

6.33 配置触摸屏

指令格式： EE【 70 Cmd 】FF FC FF FF

参数说明： Cmd(1个字节)：配置参数

BIT0: 1表示触摸屏打开，0表示触摸屏关闭；

BIT1: 1表示触摸时蜂鸣器自动响， 0表示不响；

BIT4~BIT2：触摸坐标值上传方式

000: 表示按下触摸屏时才上传1次坐标

001: 表示触摸屏被按下直至释放后上传1次坐标

010: 触摸一直被按下时，每100ms上传1次坐标，释放时也上传1次坐标

011: 表示触摸屏被按下和释放时均上传1次坐标

BIT5: 1表示在4秒内连续点击某个非触控区域20下，屏幕进入触摸校准模式，0表示禁止此功能；

BIT7-BIT6：保留

触摸坐标值上传格式：

按下时上传格式：EE 01 X Y FF FC FF FF

释放时上传格式：EE 03 X Y FF FC FF FF，X、Y均为2个字节，高字节在前

该指令包含了触摸使能、开闭蜂鸣器和坐标值上传方式。如下图 6.14 所示，若触摸上传格式配置为“000”，用户按下屏幕(50,100)的位置后，设备上传数据：EE 01【00 32 00 64】FF FC FF FF。用户主机通过判断接收到的坐标(X,Y)是否在有效触摸区域内即可确定当前触摸是否有效。设备自身对触摸压力值进行了多次采样和运算，用户无需再进行二次运算。

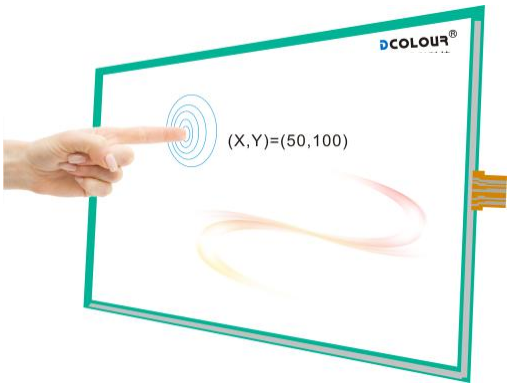


图 6.14 触摸屏工作介绍

6.34 触摸屏校准

指令格式： EE【 72 】FF FC FF FF

参数说明： 无

该指令用于触摸屏的校准。设备出厂前均进行了校准，用户无需再次校准。发送校准命令后，根据屏幕的提示点击对应的光标，如图 6.15 所示。点击完毕后设备将会提示是否校准成功，否则需要重新校准。用户也可以通过上位机软件发送指令进行校准。

除此之外，用户在非触控区域某一点 4 秒内快速点击 20 下，系统将自动进入触摸校模式，校准完成后自动返回当前显示画面。该功能比较适合现场进行触摸校准。

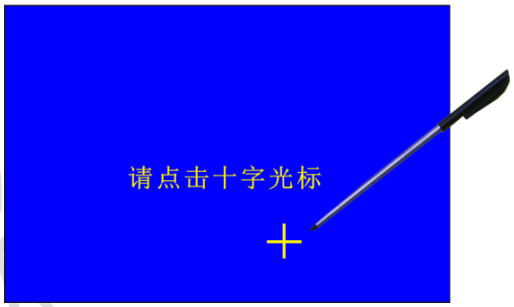


图 6.15 触摸屏校准示意图

6.35 触摸屏体验

指令格式： EE【 73 】FF FC FF FF

参数说明： 无

该指令属于测试命令。如图 6.16 所示，用户按下触摸后将在对应坐标处显示一个红色的实心圆，方便用户直观地测试触摸屏的好坏及体验触摸值的精准。设备与 PC 连接成功后，用户可点击 VisualTFT 软件工具栏的“体验触摸”来体验触摸的灵敏度和准确度。

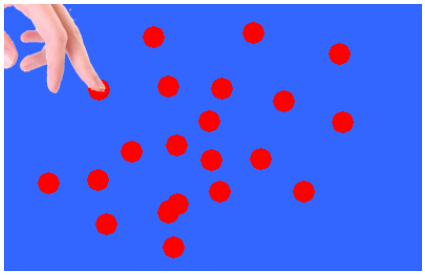


图 6.16 触摸体验效果图

6.36 设置波特率

指令格式： EE【A0 Baudset】FF FC FF FF

参数说明： Baudset(1 个字节)：波特率编序，单位 bps

0x00: 1200 0x01: 2400 0x02: 4800
0x03: 9600 0x04: 19200 0x05: 38400
0x06: 57600 0x07: 115200 0x08:1M
0x09:2M 0x0A : 218750 0x0B : 437500
0x0C : 875000 0x0D : 921800

该指令主要用于波特率的配置,范围为 1200-2Mbps。新的波特率值断电保存。用户可以直接通过上位机 VisualTFT 的“调试助手”配置新的波特率，如图 6.17 所示。

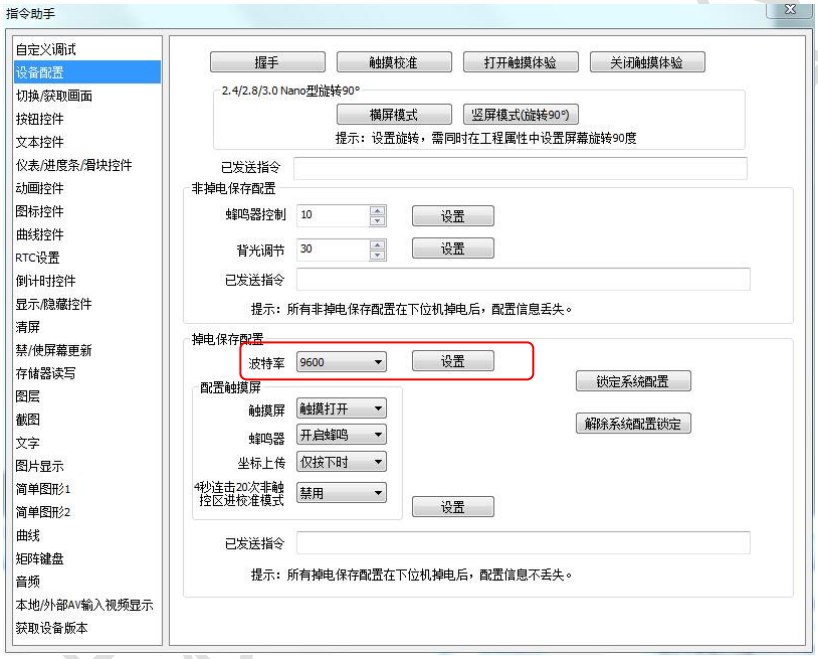


图 6.17 波特率设置

6.37 矩阵键盘控制

指令格式： EE【79 cmd】FF FC FF FF

参数说明：Cmd(1个字节)：配置参数

BIT0: 1表示矩阵键盘使能，0表示矩阵键盘关闭；

BIT1: 1表示按下键时蜂鸣器自动响， 0表示不响；

BIT4~BIT2：矩阵键盘上传方式

000: 表示按下键盘时上传1次坐标

001: 表示键盘被按下直至释放后上传1次坐标

010: 键盘长久被按下时，每100ms上传1次坐标，释放时再上传1次坐标

011: 表示键盘被按下和释放时均上传1次坐标

BIT7-BIT5：保留

设备支持外接 4*4 矩阵键盘输入。按下键盘后，键值通过串口上传给主机。

键值编码的上传格式：

按下时上传格式： EE 12 K FF FC FF FF；

释放时上传格式： EE 13 K FF FC FF FF，K 为 1 个字节，代表上传的键值

键盘为 4*4 矩阵键盘,键值编码范围为 0 到 15，其对应关系如下表 6.1 键码查询表所示。

表 6.1 键码查询表

列 / 行	K6	K7	K8	K9
K1	0	1	2	3
K2	4	5	6	7
K3	8	9	10	11
K4	12	13	14	15

6.38 写数据到 FLASH

指令格式： EE【87 Addr Data0 Data1 Data2…Datan】FF FC FF FF

参数说明： Addr (4 个字节)：数据写入的起始地址

Datan(1 个字节)：写入的数据

存储空间为 128K 字节，地址范围是 0~0x1FFFF。

写入成功后，设备返回： EE 0C FF FC FF FF。

该指令主要将用户部分的一些数据保存在设备内的 Flash 中，当做 EEPROM 使用。

6.39 读取保存在 FLASH 中的数据

指令格式：EE【88 Addr Length】FF FC FF FF

参数说明：Addr (4 个字节)：数据读起始地址

Length (2 个字节)：读取数据的长度，单位为字节

返回的数据格式为：EE 0B Data0 Data1 Data2…Datan FF FC FF FF。

该指令主要用于将写入存储器中的数据读出。

6.40 清除图层

指令格式：EE【05 Layer】FF FC FF FF

参数说明：Layer(1 个字节)：清除的图层

该指令主要用于清除指定的图层。

6.41 切换画面时自动清除当前图层

指令格式：EE【06 Enable】FF FC FF FF

参数说明：Enable (1 个字节)：使能信号

0x01：自动清除用户图层 0x00：禁止清除图层

该指令主要用于设置切换画面时是否需要自动清除当前用户图层。

6.42 截取当前屏幕并保存在 FLASH 中

指令格式： EE【0x46 Image_ID】FF FC FF FF

参数说明： Image_ID (1 个字节) ：用户自定义保存在存储器中的画面编号

该指令主要用于实现将当前屏幕显示内容保存在 Flash 中。Image_ID 值可以任意设置，与工程图片 ID 值不冲突。

6.43 显示保存在 FLASH 中的截取画面

指令格式： EE【0x47 Image_ID】FF FC FF FF
参数说明： Image_ID (1 个字节)
该指令主要用于显示保存在设备 FLASH 中的截取画面。

6.44 RTC 模式设置

指令格式： EE【85 Cmd DisMode TextMode Color Xpoint Ypoint 】FF FC FF FF
参数说明： Cmd (1 个字节)：参数配置
BIT0：使能信号
0：RTC 关闭 1：RTC 开启
BIT7-BIT1：保留
DisMode(1 个字节)：显示模式
0x00：格式 HH:MM:SS
0x01：格式 20XX-MM-DD HH:MM:SS
Font (1 个字节) ：字体选择
0x00: 8x12 点阵 (ASCII) 0x01: 8x16 点阵(ASCII)
0x02: 12x24 点阵(ASCII) 0x03: 16x32 点阵(ASCII)
0x04 12 x 12 点阵(GBK) 0x05: 16 x 16 点阵（GBK）
0x06: 24 x 24 点阵(GBK) 0x07: 32 x 32 点阵(GB2312)
0x08: 32 x 64 点阵(ASCII) 0x09: 64 x 64 点阵(GB2312)
Color (2 个字节)：显示颜色
X (2 个字节): 以点为单位的 X 轴坐标值
Y (2 个字节): 以点为单位的 Y 轴坐标值

该指令主要用于 RTC 显示参数的设定。通过设定对应的参数实现不同时钟格式、字体和位置的显示。RTC 相关设置指令建议用户直接采用上位机 VisualTFT 进行设置，设置参考界面如图 6.18 所示。

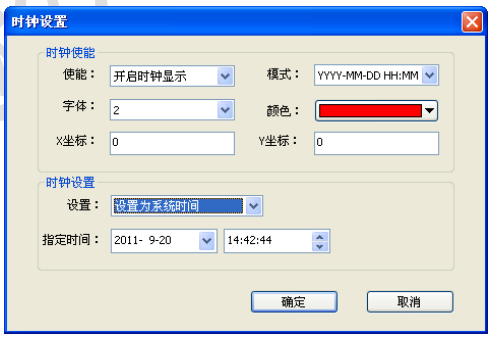


图 6.18 RTC 设置参考图

6.45 RTC 时钟设置

指令格式： EE【81 Sec Min Hour Day Week Mon Year】FF FC FF FF
参数说明： Sec: 秒设置 Min: 分设置
Hour: 小时设置 Day: 日设置
Week: 星期设置 Mon: 月设置
Year: 年设置

以上参数均为 1 个字节，以 BCD 码表示，星期天设置为 0x00

该指令主要用于当前时间参数的设定，建议用户直接通过上位机软件进行设置。建议用户使用时钟控件来显示 RTC，可以直接点击触摸弹出键盘来校准当前时间。

6.46 读取 RTC 时钟

指令格式：EE【82】FF FC FF FF

参数说明：无

该指令主要用于获取当前时间数值。数据上传格式：EE F7 Year Mon Week Day Hour Min Sec FF FC FF FF。以上参数均为 1 个字节，以 BCD 码表示。

6.47 锁定系统配置

指令格式：EE【08 A5 5A 5F F5】FF FC FF FF

参数说明：无

该指令防止在系统运行过程中，收到主机错误指令帧导致系统配置意外修改。一旦配置被锁定，设备将无法接收外部串口命令进行修改，直到锁定被解除。配置参数包括：波特率、触摸和矩阵键盘工作模式、自动背光调节参数，可以直接在 PC 的指令助手中进行配置。

6.48 解除系统配置锁定

指令格式：EE【09 DE ED 13 31】FF FC FF FF

参数说明：无

一旦解除系统配置锁定，设备可以重新接收外部串口命令来修改配置参数。用户可以直接在 PC 的指令助手中进行配置。