KGUS 屏开发指南

V6.3

目录

符号及专用名词说明

表示方法	说明	
0x 和 H	数字前加 0x 或数字后加 H 表示十六进制数，如 0x10=10H=16	
变量地址	RAM 空间中某段空间的首地址，该空间储存了变量。	
描述指针	RAM 空间中某段空间的首地址，该空间储存了描述变量属性的值。	
高/低字节	串口的所有指令及数据都是 16 进制格式，对于字型（2 字节）数据，总是采用高字节先发送（MSB）方式。比如 0x1234 传送时先传送 0x12，即 0x12 为高字节，0x34 为低字节。	
系统配置寄存器	特指对屏幕系统进行配置的寄存器，通过写入 CONFIG.txt 文件来实现配置。其中，各寄存器以大写字母 R 开头加十六进制数字来表示，如 R2，RC 等。	
变量储存器	RAM 空间，储存变量地址和描述指针指向的数据。储存的数据掉电时不保存。	
字库空间	储存配置文件、字库文件、图标库文件。	均为 FLASH 储存空间中的一部分，储存的数据掉电时不丢失。
图片空间	储存界面图片。	
数据库空间	储存用户数据库。	
寄存器空间	特指可通过通讯读写的寄存器空间，其中各寄存器地址均通过十六进制数字表示，如 0x01，0x4F 等。	

第一章 快速上手

1.1 连线与上电

首先确认屏幕电压、功耗以及对开关电源的选取。开关电源对屏幕的正常显示有十分重要的作用，选用合适的开关电源供电才可以确保正常点亮屏幕。电压过小、电流不稳、功率过低都可能导致黑屏、闪屏等不正常的显示状况。

KGUS 屏典型的接口接线方式为 8pin 接口，接口介绍如下：

8pin 接口

将软排线的一端与 KGUS 屏的端子座连接，另一端与带 USB 接口的 TTL 转 RS232 电平转接板连接，再通过 USB 线与电脑相连可实现串口通信。

8pin 接口为 2.0mm 间距。如下图所示，KGUS 屏与电脑、单片机连接所用的连接线也不尽相同。

如下图 1.1 所示，就是连线的相关说明。

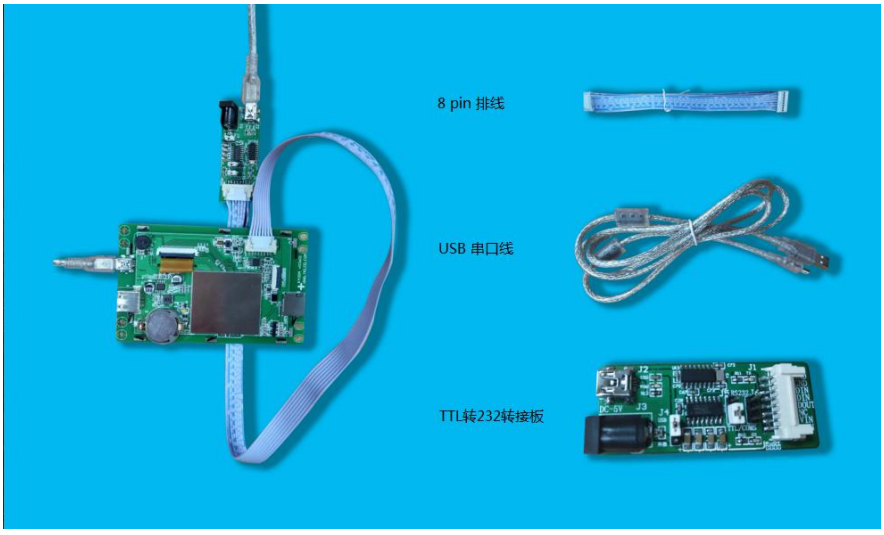


图 1.1 8pin 接口接线说明

一共有两种 USB-to-UART 芯片，MAX232 芯片和 CH340 芯片。根据芯片类型，至云利科技官网下载并安装相应驱动，以保证 KGUS 屏可与电脑正常通信。为保证通信顺畅，可执行如下步骤：

Step1：打开设备管理器，右击“USB2.0-Serial”，并选择更新驱动程序如下图 1.2 所示。

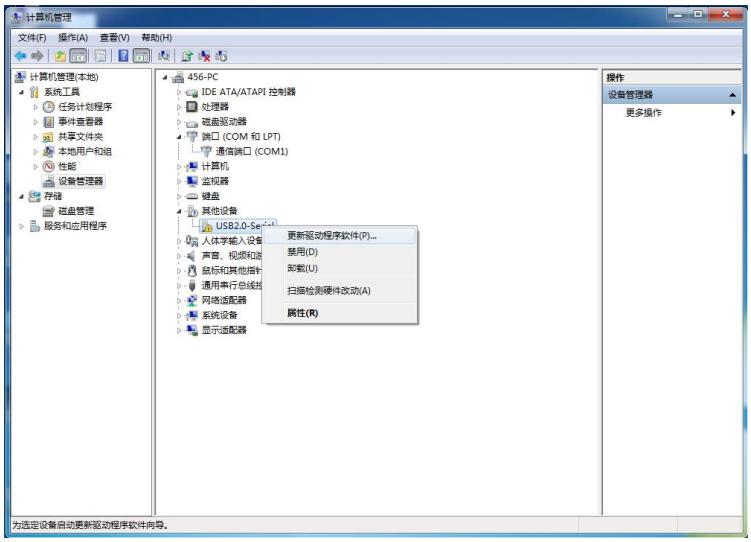


图 1.2 选择更新相关驱动设备

Step2：在弹出窗口中选择驱动软件的路径，点击下一步如图 1.3 所示。

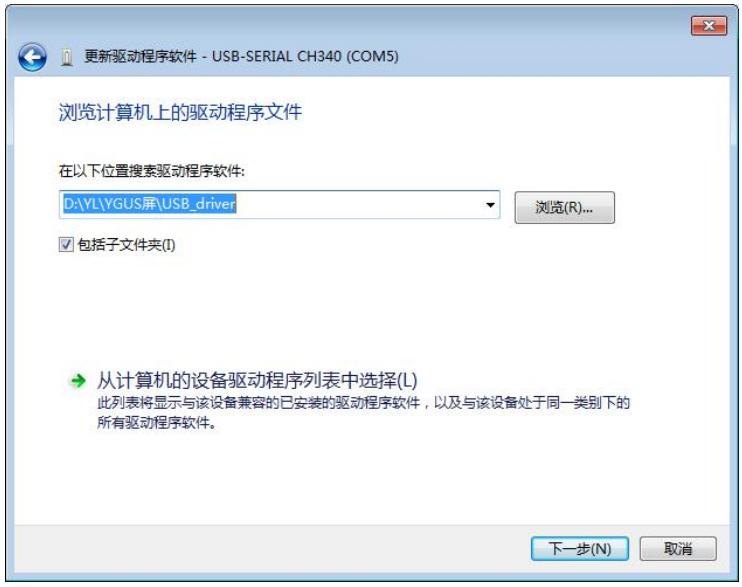


图 1.3 选择好 KGUS 驱动软件路径

Step3：根据自己选择驱动软件的路径，自动完成驱动程序的更新如图 1.4 所示。

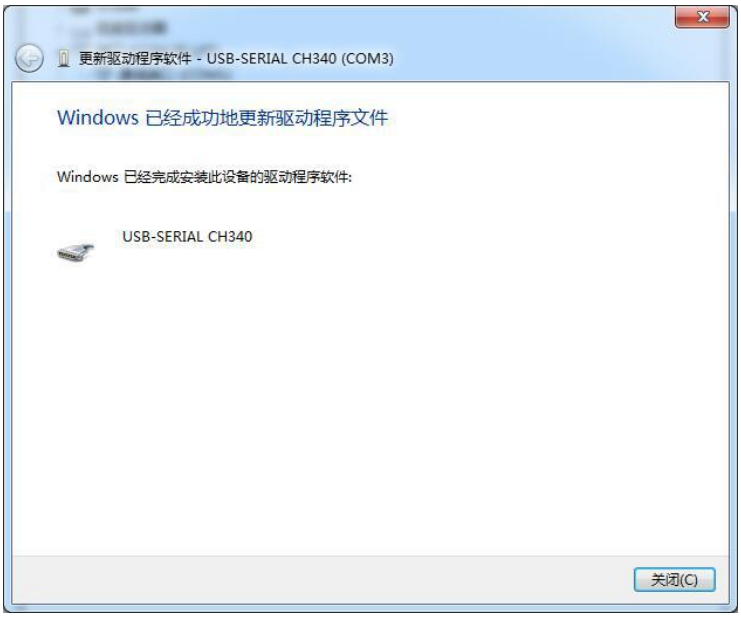


图 1.4 KGUS 软件驱动更新完成的显示

安装完成后，可在设备管理器中查看到驱动对应的端口号。

注意事项：

用户制作连接线时，需注意 DOUT 引脚是屏的发送端，DIN 引脚是屏的接收端；

当屏幕有两个电平信号可选时，如图 1.5 所示一共有三个铜柱短接上面两个铜柱时是 RS232 信号，短接下面两个铜柱时是 TTL 信号。

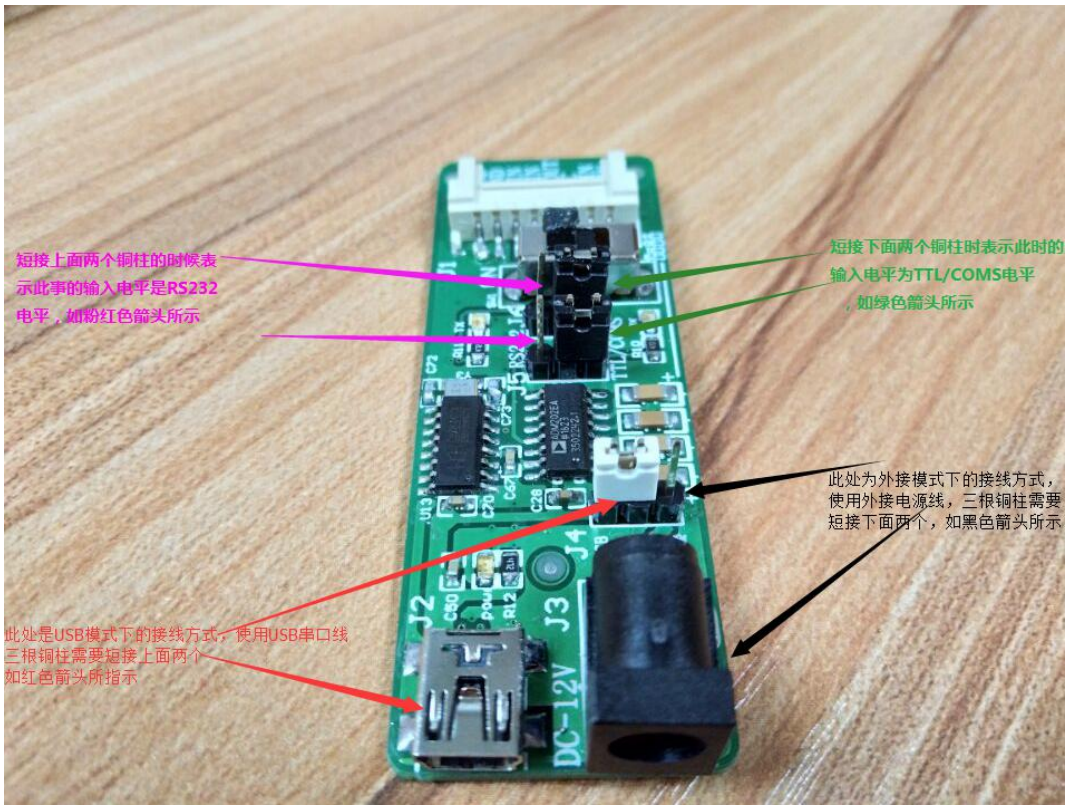


图 1.5 短接板使用说明

图 1.6 KGUS 屏背面的电平说明

1.2 KGUS 的安装与开发流程

首先从我司官网 http://www.szyunli.com 下载 KGUS 开发软件。

KGUS 屏是基于配置文件来工作的，所以整个开发过程是用户利用 PC 端 KGUS 开发软件辅助设计完成变量配置文件的过程，其基本开发流程如下步骤所示。

1.变量规划

用户需要对所需变量进行定义并对数量进行记录。比如，需要检测时间，就只要一个变量。

为了便于用户以后对工程的维护和修改，建议用户在制作工程之前先制作一个表格，将所需要的变量地址的分配进行统一规划。

2.界面设计

根据需求，由美工制作精美的图片。利用绘图软件对界面及图标、字库、按钮格式等进行制作。确保在 KGUS 屏上的显示效果与设计效果一致。

3.界面配置

通过 PC 端 KGUS 开发软件对页面上的控件的使用进行配置，然后点击生成配置按钮，在工程制作目录中会生成对应的触控配置文件和变量配置文件。

在 KGUS 中可以配置时钟显示、文本录入、动画显示、图标显示、曲线、表格等多种功能。

4.测试修改

通过 SD 卡、U 盘或者串口下载，将制作的工程下载到 KGUS 串口屏，然后根据需求，进行修改。重复 2~3 两步。

5.定版归档

定版后，需要将配置文件、图片、字库、图标库等与 KGUS 相关的文件保存到 SD 卡或 U 盘上，即可量产。

第二章 KGUS 开发体系

2.1 KGUS 开发体系简介

KGUS 开发体系是由 KGUS 屏硬件和 KGUS 开发软件构成的，如下图 2.1 所示。

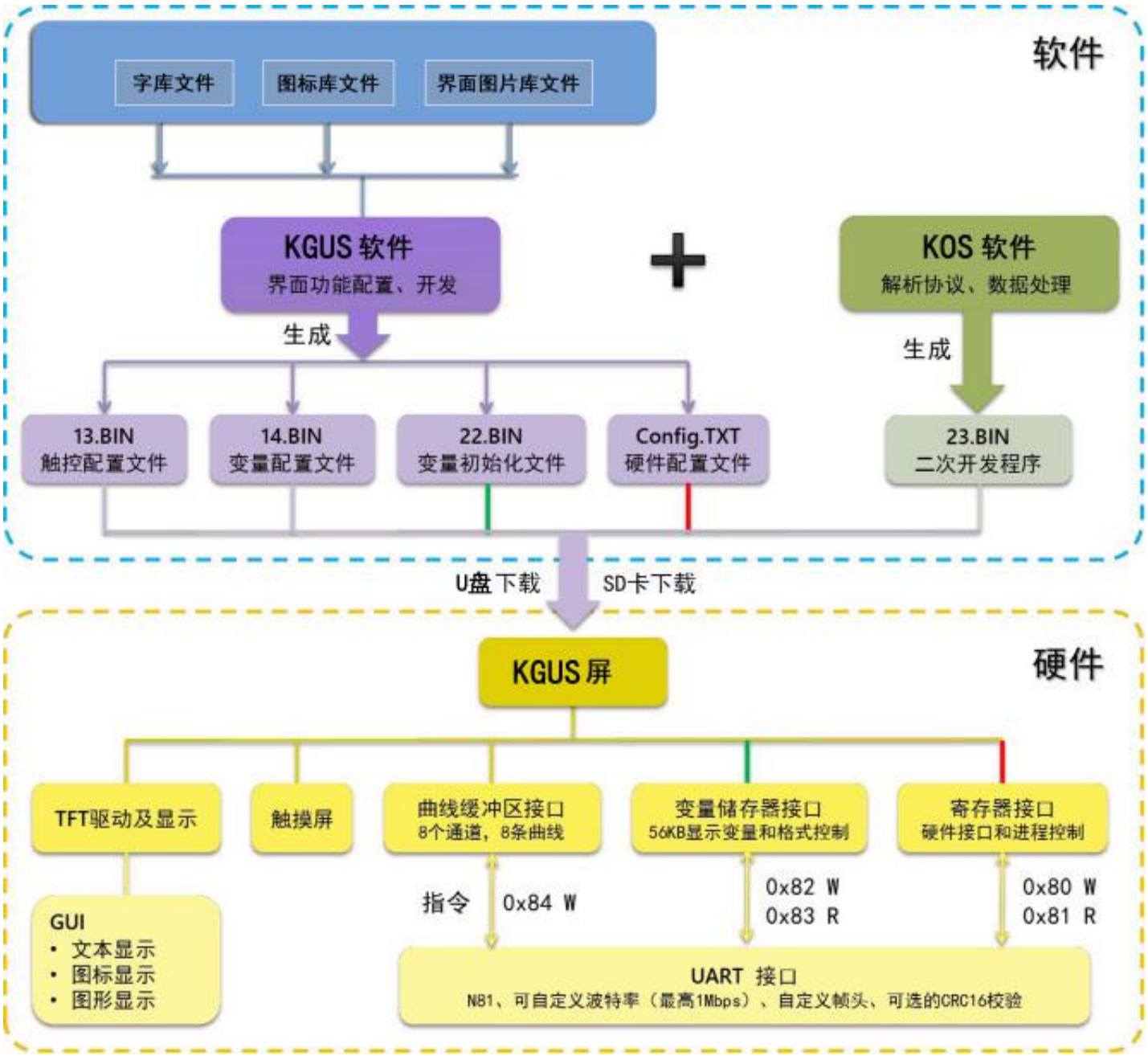


图 2.1 KGUS 开发体系图解

其中，KGUS 软件是我司自主研发的智慧型、图形界面、人机系统开发软件。该开发软件配套基于 K800 内核的 KGUS 屏使用。用户可以通过电脑端的 KGUS 开发软件对 KGUS 屏进行功能的开发与设计，它大大降低了用户的开发难度，也减少了用户的开发成本。

KGUS 屏是由我司自主研发的出厂预装 KGUS 软件并基于 K800 内核驱动的工业组态屏，具有可靠、稳定、功能性强、易用性良好等多个优点。

典型的 KGUS 屏（YL-HMI043T4827KRN-04 和 YL-HMI070T8048KRN-04）的外观如图 2.2 和图 2.3 所示。

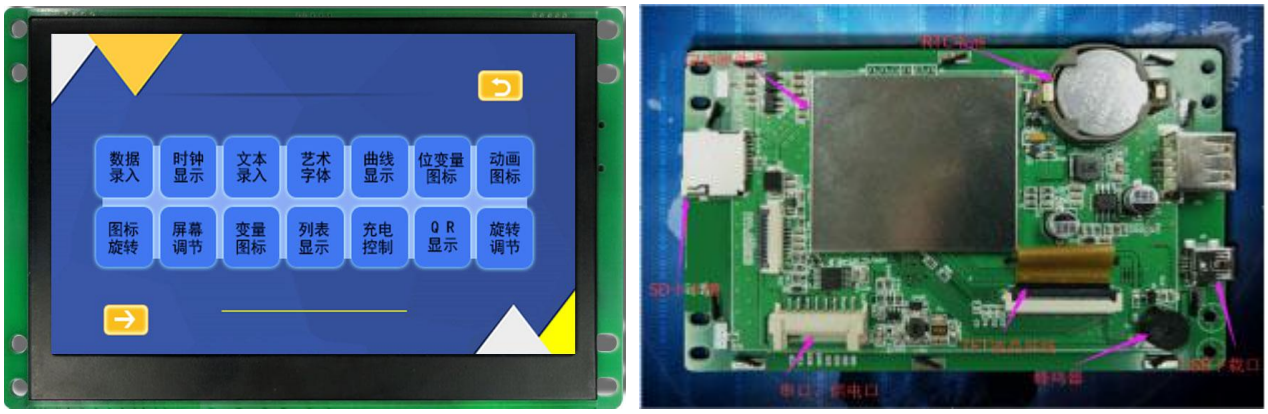


图 2.2 KGUS 屏(YL-HMI043T4827KRN-04)的外观

图 2.3 KGUS 屏(YL-HMI070T8048KRN-04)的外观

2.2 KGUS 最简单开发实例

1. 首先准备好一张. bmp 格式的 24 位色或 24 位色以上的图片（高位色的图片也可以，只是【屏幕显示的位数为 24 位】）命名以 000 开始，如 000. bmp，也可以添加文字如 000 开始. bmp，图片的命名要依次增加。如图 2. 4 所示。

名称	日期	类型	大小	标记
000.bmp	2017/9/2 13:55	Kankan BMP 图像	511 KB	
001.bmp	2017/9/2 13:56	Kankan BMP 图像	511 KB	
002.bmp	2017/8/11 9:56	Kankan BMP 图像	511 KB	
003.bmp	2017/8/11 10:27	Kankan BMP 图像	511 KB	
004.bmp	2017/8/11 10:28	Kankan BMP 图像	511 KB	
005.bmp	2017/8/11 9:59	Kankan BMP 图像	511 KB	
006.bmp	2017/8/11 10:25	Kankan BMP 图像	511 KB	
007.bmp	2017/8/11 10:27	Kankan BMP 图像	511 KB	
008.bmp	2017/8/11 10:26	Kankan BMP 图像	511 KB	
009.bmp	2017/9/2 13:57	Kankan BMP 图像	511 KB	
010功能总览.bmp	2017/9/2 13:59	Kankan BMP 图像	511 KB	
011文本滚动.bmp	2017/9/2 14:00	Kankan BMP 图像	511 KB	
012时钟显示.bmp	2017/9/2 14:01	Kankan BMP 图像	511 KB	
013文本录入.bmp	2017/9/2 14:02	Kankan BMP 图像	511 KB	

图 2.4 KGUS 图片命名方式实例

2.打开 KGUS 软件打开新建工程，并根据提示选择自己希望创建工程的文件夹，然后点击 OK 按钮。即可进入 KGUS 工程制作步骤，如图 2.5 所示。

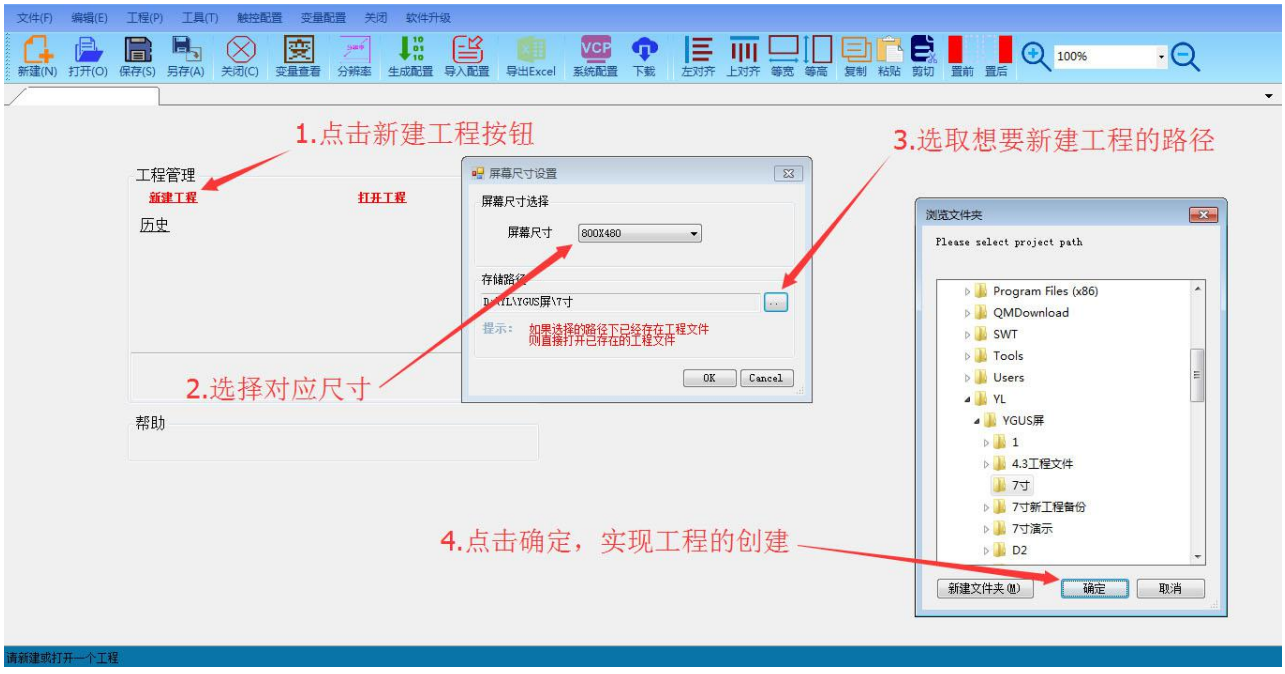


图 2.5 KGUS 软件新建工程详解

3.点击下面的添加按钮，然后找到之前准备好的图片所在位置 ，将图片选中然后点击打开，之前准备好的图片就添加进工程，如图 2.6 所示。

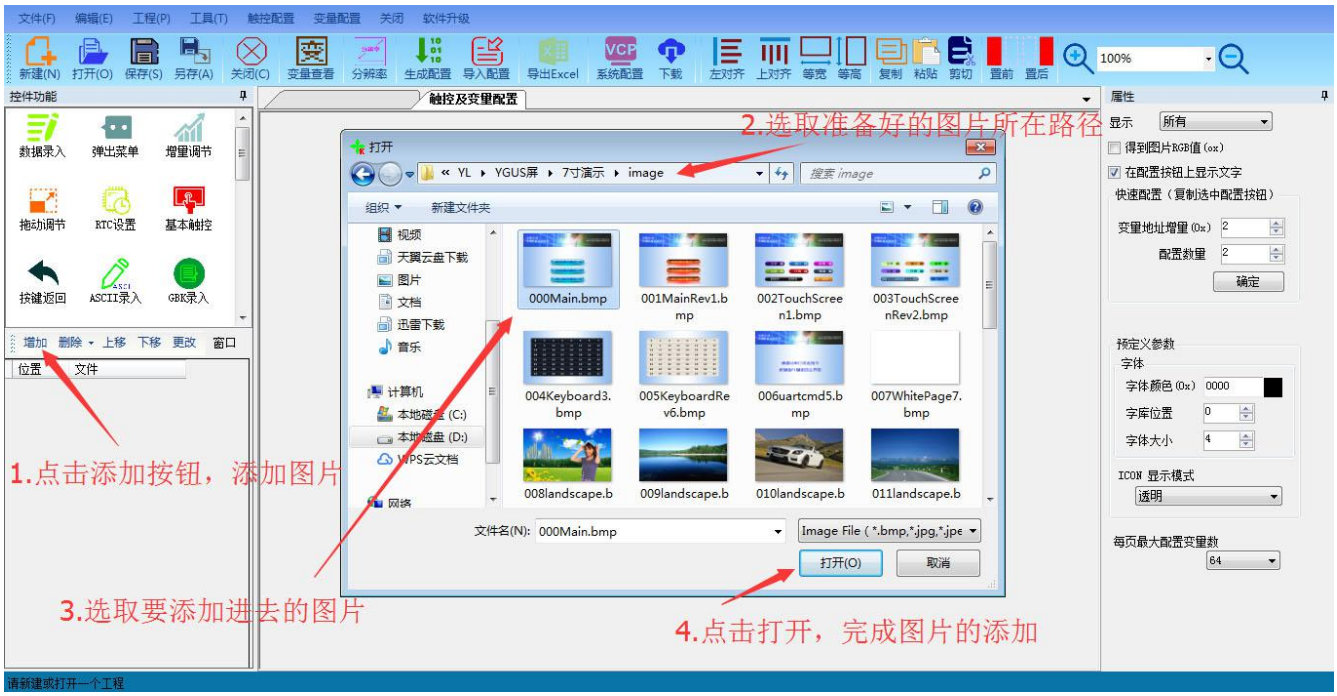


图 2.6 工程添加图片详解

4. 接着就是对图片实现功能的操作。比如想要实现页面的跳转，可以在（1）左侧的功能箱中找到，也可以在（2）触控配置下拉菜单中找到基本触控功能控件，如图 2.7 所示。

图 2.7 基本触控控件在 KGUS 软件中的位置

5.然后就是在自己想要点击实现页面跳转的地方划取一片区域，右侧会弹出对应的属性框如图 2.8 所示。

图 2.8 基本触控属性框设置

6.只需要修改页面切换处，选中想要的页面点击确定即可，如图 2.9 所示。

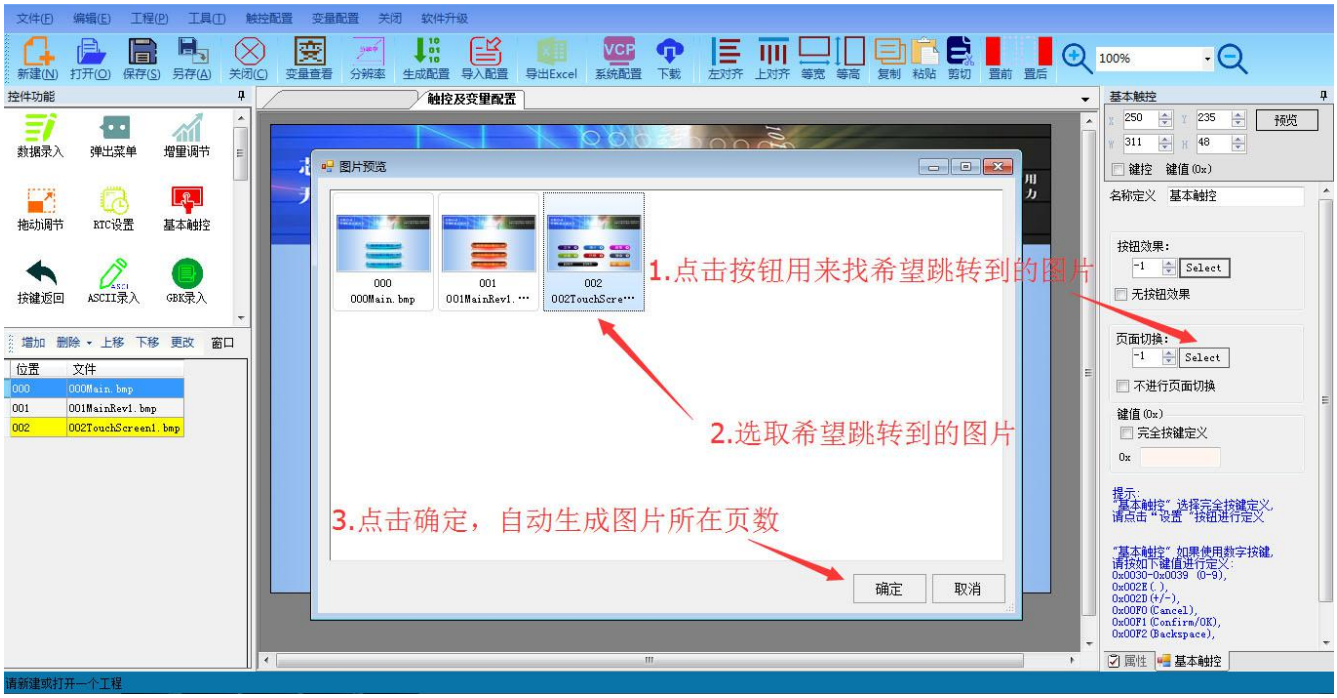


图 2.9 实现页面切换步骤

7.然后点击生成配置功能键，至此工程的制作已经完成了 。接着就是把工程下载到 KGUS 屏显示了。KGUS 屏提

供了两种下载方式，一种是通过 SD 卡下载，一种是通过 U 盘下载。只需要把 YK_SET 文件夹拷贝到 SD 卡或者 U 盘里面，直接插到 KGUS 屏里面。就可以自动下载工程到 KGUS 屏了。

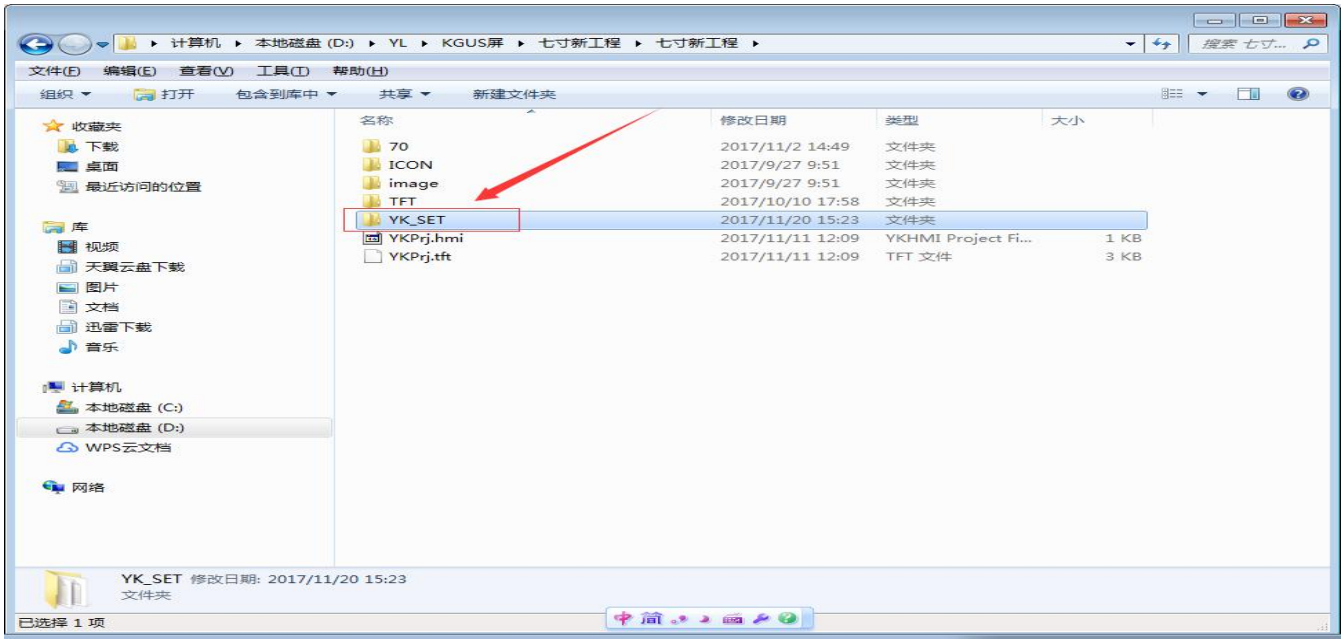


图 2.11 工程的生成与下载

2.3 KGUS 开发体系优点

与传统 LCM 通过时序或者指令控制显示不同，KGUS 屏采用直接变量驱动显示方式，所有的显示和操作都是基于预先设置好的变量配置文件来工作的，两种不同的工作方式导致用户开发应用时的软件架构和二次开发难度完全不同。

KGUS 屏的开发方式大大降低了用户的开发难度。如果只涉及少量参数和页面的简单 GUI，1 个软件工程师很快就可以搞定；如果是复杂的包含有几十个参数和页面，同时考虑动画、图标等等吸引眼球的 GUI ，只需 1 个软件工程师和 1 个美工协同工作，花费 5-7 天就能完成。

KGUS 开发体系的优势列举如下：

1. KGUS 把 GUI（用户图形界面）的每一个页面分解成多个控件，用户需要实现某个功能只需要利用 PC 端的开发软件在相应的页面上添加功能控件即可实现。
2. KGUS 有多种控件可供用户选择，能够实现丰富的功能（如显示数据、触控输入、图片动画等）。
3. 具有 56K 的 RAM 变量空间，8 通道曲线趋势图存储器，极快（最快 80ms）的变量显示响应速度。
4. 具有 256 字节的配置寄存器控件，并支持串口指令的读写，易于进行硬件控制与操作。
5. 每一页面可设置多达 128 个显示控件(支持显示控件的叠加)和任意多的触控控件。
6. 具有 USB 接口，可以使用 U 盘实现 KGUS 屏硬件参数配置、图片数据下载以及软件的升级。同时还拥有 SD 接口，在量产时极为方便，并便于生产档案的管理。
7. 集成了 RTC(公历/农历)，集成了背光亮度的调节以及触控蜂鸣器伴音功能。
8. 支持电容和电阻触摸屏，可在图片存储器空间构造高可靠性的用户数据库。

9. 集成了 K OS 平台，它允许用户把一部分的代码放在 KGUS 屏上运行，让用户的二次开发变得简单，用户可利用丰富的指令实现更为复杂的功能。这也使 KGUS 屏作为系统的主控设备成为了可能。
10. K OS 平台集成了数学运算（包括 MAC、CRC）、数据存储（包括 Flash 数据库读写）、串口通信、串口外设（如打印机）驱动、KGUS 进程控制等指令。
11. 提供了可靠的硬件平台（基于 ASIC 的 HMI 平台架构，（KGUS 软件采用汇编代码设计，总代码量约 50KB），使得 KGUS 屏不仅性能优越，运行也极为稳定可靠。
12. 产品通过 CE 和 RoHS 认证。

2.4 KGUS 软件处理流程

当用户用 PC 端 KGUS 软件生成配置文件并利用 USB 串口或 SD 卡下载到 KGUS 屏中时，KGUS 屏的软件处理流程如图 2.12 所示。

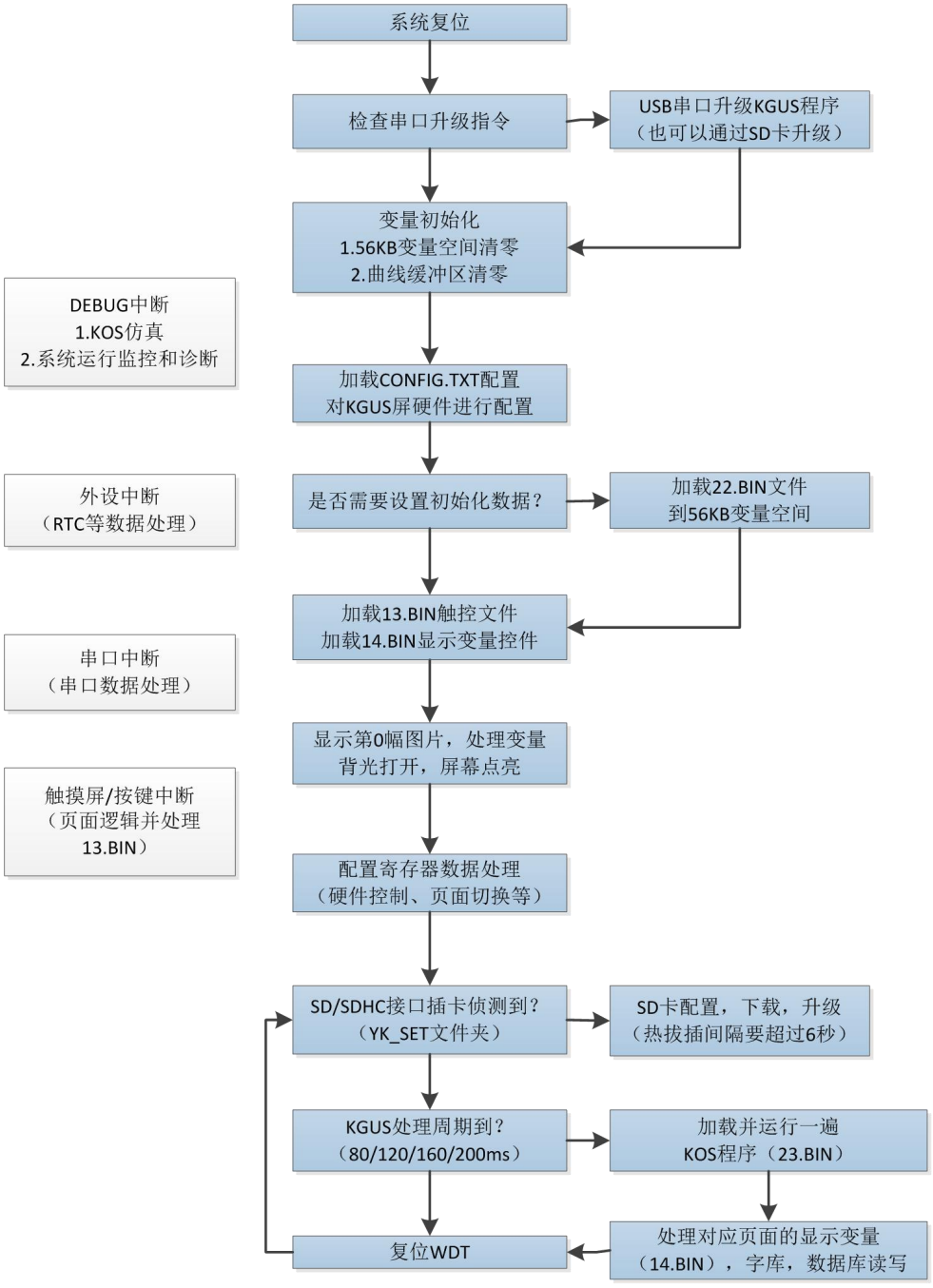


图 2.12 KGUS 软件处理流程图

注意：K OS 程序在每个 KGUS 周期（80/120/160/200ms）都完整的运行一遍，所以 K OS 程序中不能出现死循环或者通过指令循环的延时。

第三章 KGUS 屏的配置

3.1 素材文件的格式要求

从 KGUS 开发体系的介绍中我们了解到，通过 KGUS 屏开发工程时，需要准备的素材文件有图片、图标库、字库等。

KGUS 在调用素材文件时是通过文件编号来进行的。因此，**素材文件命名时，均应当用阿拉伯数字开头**，通过阿拉伯数字对文件实现编号。例如，0_总览.bmp，004.bmp，45 文本录入.ico，60 宋体 GBK16.HZK 等。阿拉伯数字后的文字可以不写。需要注意的是，同类素材中，不允许有相同编号的文件出现，否则将导致调用文件出错。

1. 图片文件

图片文件可以用来作为用户界面，存储在专用的图片存储空间中。为了实现良好的显示效果对图片文件的格式有严格的要求，图片文件必须是与 KGUS 屏具有相同分辨率的 24 位色或 32 位色 BMP 格式图片。

注意：图片不仅是用来做用户界面的背景，同时也要包含所要出现在这个界面的按钮、菜单选项的图标等等。

在 KGUS 开发软件中只能够定义该图片的某个区域的功能，而不能在某个区域添加新的按钮或菜单图标，因此请在制作图片时，规划好当前页面的所有按钮和菜单选项的位置及图标。

2. 图标库文件

通过绘图软件制作的图标文件为图片格式，用阿拉伯数字对图标的图片进行编号后，需通过 KGUS 的图标生成软件生成图标库文件。每一个图标像素大小应小于 255*255，超出该范围时，图标生成软件将自动将图标缩小至 255。

注意：如果一个图标库文件大于 256KB 的，那么它占用的空间就不止一个。因此，图标库的编号不一定非要为连续的数字。例如，当每个图标库文件占用两个空间时，文件编号应间隔 1：2 提醒.ico，4 信号灯.ico。若每个图标库文件占用三个空间时，文件编号应间隔 2：3 故障.ico，6 提示语.ico。以此类推。

3.字库文件

KGUS 支持国际通用的多种字库编码：ASCII，GBK，GB2312。KGUS 中已经预装了 ASCII 编码的字库，其中包含了点阵大小为 4*8~64*128 的全部 ASCII 字符。该字库编号为 0，可直接调用来实现数字、字母等的显示。当需要使用其他编码的字库时，需通过字库生成器生成。KGUS 支持 BIN、DZK、HZK 这三种字库文件。

KGUS 屏的数据格式和颜色系统如下：

1. KGUS 屏的数据格式

由于 KGUS 屏主要面向 MCU 等嵌入式系统应用，为了用户处理的方便，KGUS 屏所使用的数据采用整数（字）、无符号整数（字）、长整数（双字）、超长整数（4 个字）表示，相关表示的范围如下表 3.1 所示。

表 3.1 数字类型的取值范围

整数	-32768(0x8000)到+32767(0x7FFF)
无符号整数	0(0x0000)到 65535(0xFFFF)
长整数	-2147483648(0x80000000)到+2147483647(0x7FFFFFFF)
超长整数	-9223372036854775808 到+9223372036854775807

小数采用定点小数的表示，用户自定义小数位数。如 0x4D2(1234)，规定小数为 2 位时，表示 12.34。

2. KGUS 屏的颜色系统

KGUS 屏使用 65K 色的颜色系统，调色板的定义如表 3.2 所示。

表 3.2 KGUS 屏调色板的定义

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Define	R4	R3	R2	R1	R0	G5	G4	G3	G2	G1	G0	B4	B3	B2	B1	B0
	红色 0xF800					绿色 0x07E0						蓝色 0x001F				

3.2 配置文件的构成

在新建一个工程时，会自动生成一系列文件，如下图所示。其中，YKPrj.hmi 是 KGUS 软件唯一可识别的编辑程序，该文件不可重命名，不可删除。YK_SET 文件夹中包含最终将下载至 KGUS 屏中的所有文件。

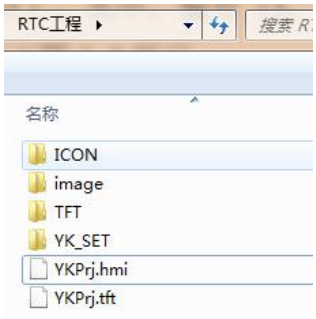


图 3.1 KGUS 工程文件根目录

如下图所示，YK_SET 文件夹包含了界面图片，触控配置文件，变量配置文件，变量初始化文件（22_Config.bin），图标库（70.ico），屏幕参数配置文件（CONFIG.txt）。其中，编号 13、14、22 的 bin 文件及 CONFIG.txt 文件是由 KGUS 软件生成的所有配置文件，如下图 3.2 所示。

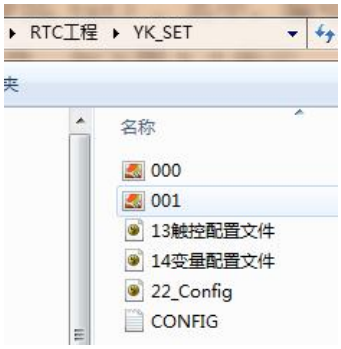


图 3.2 YK_SET 文件夹根目录

3.3 素材文件、配置文件与数据的储存

KGUS 屏提供 FLASH（256MB/1GB/2GB）储存空间、RAM（56KB）储存空间、配置寄存器（256Byte）空间，以及曲线缓冲区（16KB）。

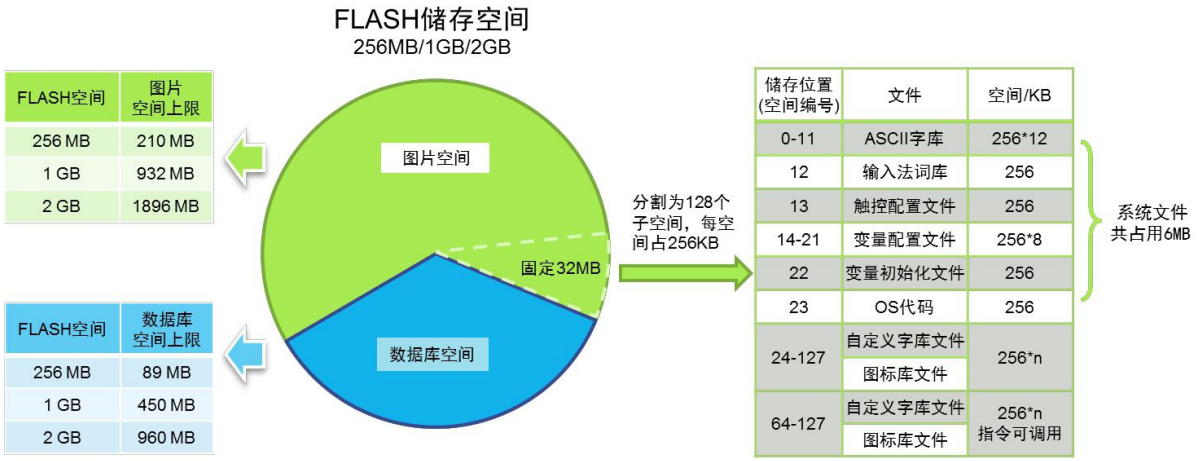
本章讲解素材文件和配置文件的储存方式，后续章节会讲解如何分配空间来储存各类文件。KGUS 屏中的文件都有数字编号。储存在同一空间中的文件的编号不可重复，例如，不能有两个 000 号图片。但是，不同空间中的文件编号互相不干扰，例如，013.bmp 和 13 触控配置文件.bin 是可以同时存在的。

3.3.1 FLASH 储存空间

FLASH 储存空间主要用于储存参数配置文件（CONFIG.txt）、图片文件、字库文件、图标库、触控配置文件、变量配置文件、用户数据等。储存于 FLASH 的数据和文件掉电时不会丢失。不论是多大容量的 FLASH 空间，都固定分出 32 MB 的空间（以下称作字库空间）来储存 KGUS 预存文件、部分配置文件，以及用户自定义的字库和图标库文件。

FLASH 储存空间按照储存内容分为图片空间和数据库空间。其中，图片空间中，固定分出 32 MB 的字库空间。字库空间被分割为 128 份，每个空间的大小为 256 KB。各项配置文件根据文件编号储存至相应的空间位置中，如下图所示。0-11 号空间储存了 KGUS 预装的编号为 0 的 ASCII 字库文件，该文件较大，故占用了从 0 号空间开始的 12 个空间。13 号空间储存了通过 KGUS 软件生成的编号 13 的触控配置文件。从图中可以看出，32MB 的字库空间中，0-23 号空间（共占用 6MB）储存的文件类型是固定的，不能用于储存其他文件。

故用户自定义的字库文件编号、图标库文件编号应当在 24-127 中取数（含 24 和 127），其中，仅 64-127 号空间可通过指令调用。



FLASH空间	图片空间上限
256 MB	210 MB
1 GB	932 MB
2 GB	1896 MB

FLASH空间	数据库空间上限
256 MB	89 MB
1 GB	450 MB
2 GB	960 MB

储存位置(空间编号)	文件	空间/KB
0-11	ASCII字库	256*12
12	输入法词库	256
13	触控配置文件	256
14-21	变量配置文件	256*8
22	变量初始化文件	256
23	OS代码	256
24-127	自定义字库文件	256*n
	图标库文件	
64-127	自定义字库文件	256*n 指令可调用
	图标库文件	

图 3.3 FLASH 储存空间

储存在字库空间中的文件应当以表示字库存储位置（0-127）的阿拉伯数字开头来命名。假如用户需要将字库文件存储到 KGUS 屏的第 50 个字库空间中，字库文件的命名格式为“50*.HKZ”（其中*可以任意输入也可以不输入），比如“50 12x12.HKZ”、“50.HKZ”或“050 12 点阵.HKZ”等，不能命名为“字库 50.HKZ”。

表 3.3 字库空间的储存内容

存储位置	文件类型	命名规格	举例	说明
0-11	专用 ASCII 字库	0*.HZK	0_ASC 库.HZK	预装
12	输入法词库	12*.BIN	12_PY 库.BIN	
13	触控配置	13*.BIN	13 触控文件.BIN	KGUS 开发软件生成
14-21	变量配置	14*.BIN	14 变量文件. BIN	
22	变量初始化	22*.BIN	22 初始.BIN	
23	OS 代码	23*.BIN	23 水处理.BIN	K OS 生成
24-127	字库文件	字库存储位置*.bin/HKZ/DZK	35 汉字库.DZK	TS3 字库提取软件生成
	图标库	字库存储位置*.ICO	51 图标库.ICO	图标工具箱生成

除去 32MB 字库空间外，剩下的空间用于储存界面图片以及用户数据等。对于不同的 FLASH 空间容量，图片和数据库的容量均有上限，如下表 3. 4 所示。

表 3.4 图片空间及数据库空间的上限值

Flash 空间	最大图片空间	最大数据库空间
256 MB	210 MB	89 MB
1 GB	932 MB	450 MB
2 GB	1896 MB	960 MB

在最大图片空间的情况下，可储存的不同分辨率的图片数量如表 3.5 中所示。

表 3.5 图片储存数量与空间大小的关系

FLASH 空间	不同分辨率 KGUS 屏的最大存储图片数量				
	320*240	480*272	800*480	800*600	1024*600
256MB	836	836	278	209	167
1GB	3728	3728	1242	932	745
2GB	7584	7584	2528	1896	1516

3.3.2 RAM 储存空间

RAM 空间固定为 56KB，分割为地址 0x0000~0x6FFF 的子空间。每一个地址对应的空间占 2 字节。在 KGUS 中使用变量地址或描述指针时，设置的地址为数据储存空间的首地址，即数据从设置的地址（首地址）开始按序依次储存。每个变量地址（首地址）都指向的空间大小是不固定的，因此在 KGUS 软件中给各个变量分配变量地址时，应计算好需储存的数据量，否则将可能出现分配空间的重叠而导致显示错误。

一般推荐描述指针设置在 0x4000~0x6F00 之间，变量地址设置在 0x0000~0x4000，这样就不会产生冲突。

注意：6F00 到 6FFF 之间的变量地址是硬件参数部分用的，应避免使用。

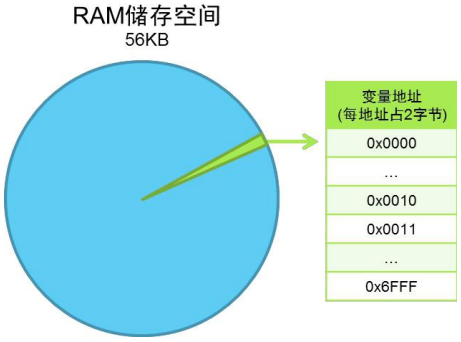


图 3.4 RAM 储存空间

A. 变量地址

变量地址是 RAM 空间中储存某一个或多个变量的子空间的首地址，在该地址指向的空间中储存了显示变量的编码或状态变量的值，下面将举例说明。例如，将一个文本显示控件的变量地址设置为 0x2000，控件中显

示的文本内容为“深圳云利科技”，那么在 RAM 空间中的储存方式如下图所示。可以看出，每个地址中可储存 2 Byte 的内容。

图 3.5 变量地址中数据的储存方式

如果需要改变这个文本显示控件中显示的字符，只需要改变相应变量地址中储存的字符编码即可。通过发送指令和触摸屏录入都可修改变量地址中储存的数据。例如，修改文本显示控件可通过文本录入控件实现，只需将两个控件设置相同的变量地址即可；详情可以参考后面章节说明。

同时，通过发送指令的方式也可修改变量地址中储存的值。

【例】发送指令：**5A A5 05 82 2001 D6D0**

2001：汉字字符“圳”储存位置的变量地址

D6D0：汉字字符“中”的编码

指令效果：字符“圳”的显示位置被替换为字符“中”

类似的，通过修改变量地址中储存的值还可修改显示数据、启用/停止动画、切换图标等多种效果。如下表 3.6 所示：

表 3.6 指令解释

功能	发送指令	效果
图标显示	**5A A5 05 82 0001 0003**	在地址为 0x0001 的控件处显示 3 号图标
实时数据	**5A A5 05 82 000A 0013**	在 0x000A 地址里写入 19

B. 描述指针

描述指针是 RAM 空间中储存描述某一变量的属性的子空间的首地址，在该地址指向的空间中储存了显示变量的各项属性值，如显示坐标、颜色、字体大小等。需要指出的是，描述指针与变量指针共用 RAM 空间，分配变量地址时应避免空间的重合。

下面继续结合文本显示控件来讲解。如下图所示，根据文本显示控件指令储存格式表格可知，文本控件描述指针指向的空间中按序依次储存了变量地址、字符显示位置的左上角坐标、字符颜色、文本框左上角和右下角坐标等数据。

图 3.6 描述指针中数据的储存方式

通常，本例们通过发送指令的方式改变描述指针中储存的值来改变描述变量的属性。

【例】发送指令：**5A A5 05 82 5003 F800** （效果：字符颜色由土黄色变为红色。）

5003：文本颜色的储存地址。

F800：红色的代码。

其他描述指针应用举例如下表 3.7 所示。

表 3.7 指针应用指令说明

功能	发送指令	效果
改变数据显示位置	5A A5 07 82 5001 0000 0000	字符显示位置的左上角坐标变为（0,0）。
改变 ASCII 字符点阵大小	5A A5 05 82 500A 10 20	将字符点阵改为 16*32，注：X 和 Y 方向的点阵值均需修改。
隐藏字符	5A A5 05 82 5008 0000	将字符长度值变为 0，以实现字符的隐藏。
更换字库文件	5A A5 07 82 5009 0042 18 18	调用 66 号字库，字库点阵大小为 24*24。即更换字库文件后字符大小也同时变更。

3.3.3 配置寄存器空间

KGUS 寄存器空间容量为 256 Byte。寄存器与存储器不同，它是用来存放寄存器状态的，比如 RTC（实时时钟）、背光亮度等实时的状态。KGUS 可以通过串口指令改变各寄存器的值，来实现上位机与 KGUS 屏信息传输及控制。

3.3.4 曲线缓冲区

KGUS 提供 16 KB 的曲线缓冲区，缓冲区中可存储 8 条曲线趋势图。将数据按照 KGUS 指令格式发送至缓冲区中，即可快速实现曲线的显示。曲线缓冲区的数据都是 16 位无符号数。

3.4 配置文件

KGUS 屏配置文件 主要包括：屏幕参数配置文件 CONFIG.txt 和变量初始化文件 22.bin，触控配置文件 13.bin 和变量配置文件 14.bin 等。

3.4.1 参数配置文件 CONFIG.txt

KGUS 屏的参数配置是通过在 CONFIG.txt 配置文件中写好寄存器参数，并用 SD 卡或者 U 盘将该文件下载到 KGUS 屏中实现的。CONFIG.txt 文件采用 windows 的文本文档格式，用类似脚本语言的方式来描述参数寄存器，每一行描述一个参数，不用修改的参数可以不写。该文件也可以通过 KGUS 开发软件来生成。在 KGUS 开发软件中，点击“配置”按钮，在弹出的设置菜单中对 KGUS 屏的串口波特率、帧头、背光控制等进行设置后，点击“输出配置文件”即可生成 CONFIG.txt 配置文件。系统配置窗口如下图所示。该窗口中每一项的设置都可以通过编辑 CONFIG.txt 文件来完成，部分功能只能通过编辑 CONFIG.txt 文件来实现，详情可查询下文中各寄存器的表格。

图 3.7 KGUS 软件的屏参数配置窗口

CONFIG.txt 文件格式必须为 R?=0X，其中?是寄存器的序号，0X 是对该寄存器配置的 16 进制(HEX)值，字母必须大写。下图 3.8 中展示了一个典型 CONFIG.txt 文件及其意义。

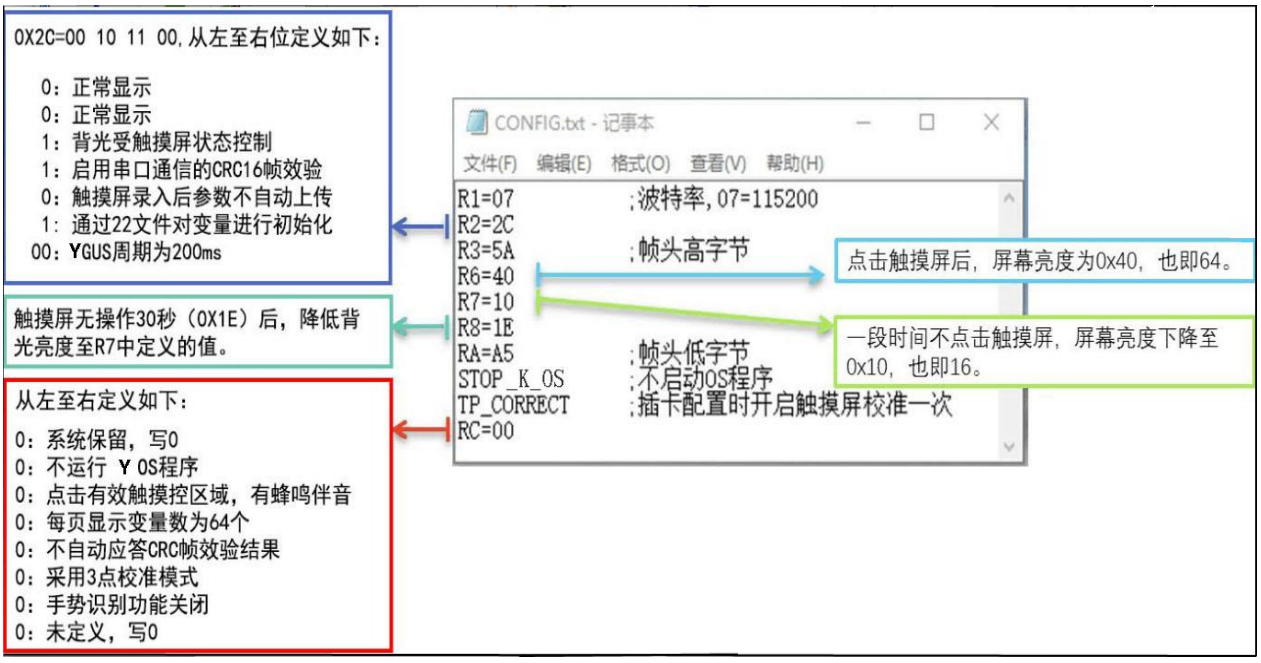


图 3.8 CONFIG.txt 标准配置举例

通常，通过 KGUS 软件生成 CONFIG.txt 文件时，文件中仅显示 R1 和其他不同于出厂设置的项目。从上图中可以看出，通过系统配置寄存器可实现很多的控制功能。需要注意的是，系统配置寄存器中，有的不能修改（如分辨率），有的直接单独使用即可，有的需要配合其他寄存器共同使用，如下所示。

1. 设置屏幕物理分辨率（R0）

显示屏的物理分辨率由 R0 寄存器设置，如表 3.8 所示。

【注】该项出厂时已经设置好，用户无需配置。

表 3.8 显示屏的物理分辨率

R0 的设置值	分辨率设置	典型的 KGUS 屏型号
01	640*480	YL-HMI056T6448TNN-04
02	800*480	YL-HMI070T8048KRN-04
03	800*600	YL-HMI080T8060KRN-04
08	1024*768	YL-HMI150T1076KRN-04
09	1024*768	YL-HMI097T1076KRN-04
0B	1024*600	YL-HMI101T1060KRN-04
20	320*240	YL-HMI035T3224KRN-04
21	480*272	YL-HMI043T482KRN-04
22	480*272	YL-HMI050T4827KRN-04

2. 设置串口波特率（R1、R5、R9）

KGUS 屏串口通信时的波特率由 R1、R5、R9 寄存器设置。**KGUS 屏出厂时 R1=07，即波特率为 115.2Kbps。**当R1 的取值为00-10 之间时，R5 和R9 无效，可在 17 档波特率中选择一档，每档对应值如下表所示。

表 3.9 串口波特率设置

R1 设置值	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10
波特率/Kbps	1.2	2.4	4.8	9.6	19.2	38.4	57.6	115.2	28.8	76.8	62.5	125	250	230.4	345.6	6912	921.6

当 R1 取值为 11 时，此时的波特率由 R5 与 R9 决定，计算公式如下：

R5：R9=6250000/波特率，其中 R5：R9 表示双字节的参数，R5 为高字节，R9 为低字节。

【例】欲设定波特率为 10Kbps，R5：R9=6250000/10000=625=0x0271，则设置 R5=02，R9=71

3.设置软件工作模式（R2、RC）

R2、RC 寄存器按二进制的位来定义，用于配置 KGUS 屏的软件工作模式。R2 寄存器的设置如下图 3.9 所示（出厂预设值均为 0）。

图 3.9 R2 寄存器的相关设置

【例】若要实现录入参数自动上传（**权重 0x08**）、KGUS 周期 120ms（**R2.1 权重 0x02；R2.0 位为 0，不计算权重**），其他参数仍为出厂值。对非 0 的项目做权重加和计算 R2，则 R2= 0x08 + 0x02=0A (**=0000 1010**)。在生成配置文件的时候，就会输出“R2=0A ；”

4. 设置串口通信帧头（R3、RA）

通信帧头有两个作用：一是用于串口数据帧的识别与同步，二是在多台 KGUS 屏并联工作时，把帧头作为设备的地址来加以区分。KGUS 屏的串口通信帧头通过 R3、RA 寄存器设置。

【例】设置 R3=66，RA=88，则发送指令必须以 0x66 0x88 开头才能被 KGUS 屏识别与接收。

注意：KGUS 屏出厂预设通信帧头为 R3=5A，RA=A5，即帧头为 0x5AA5。

5. 设置显示位时钟相位（R4）

KGUS 使用的液晶屏由于 TCON 不同，其显示数据和显示位时钟的相位关系有两种，由 R4 寄存器来设置，具体如下：

R4=00　显示数据在显示位时钟下降沿锁存
R4=FF　显示数据在显示位时钟上升沿锁存

注意：该项出厂时已设置好，用户无需配置，若配置错误将导致显示画面抖动或出现毛边。

6. 设置背光待机及唤醒（R2、R6、R7、R8）

当 R2 中 TP_LED 选用背光受触摸状态控制时，可实现无操作熄屏待机以及触摸唤醒屏幕的效果。此时需结合 R6，R7，R8 这三个寄存器共同来完成。R6 配置唤醒屏幕时的背光亮度（背光亮度范围为 0~64，0x00~0x40）；R7 配置熄屏待机时的背光亮度（背光亮度范围同上）；R8 配置启动熄屏待机的最长不操作时间，即多长时间不操作屏幕，屏幕会熄屏。如下图 3.10 所示。

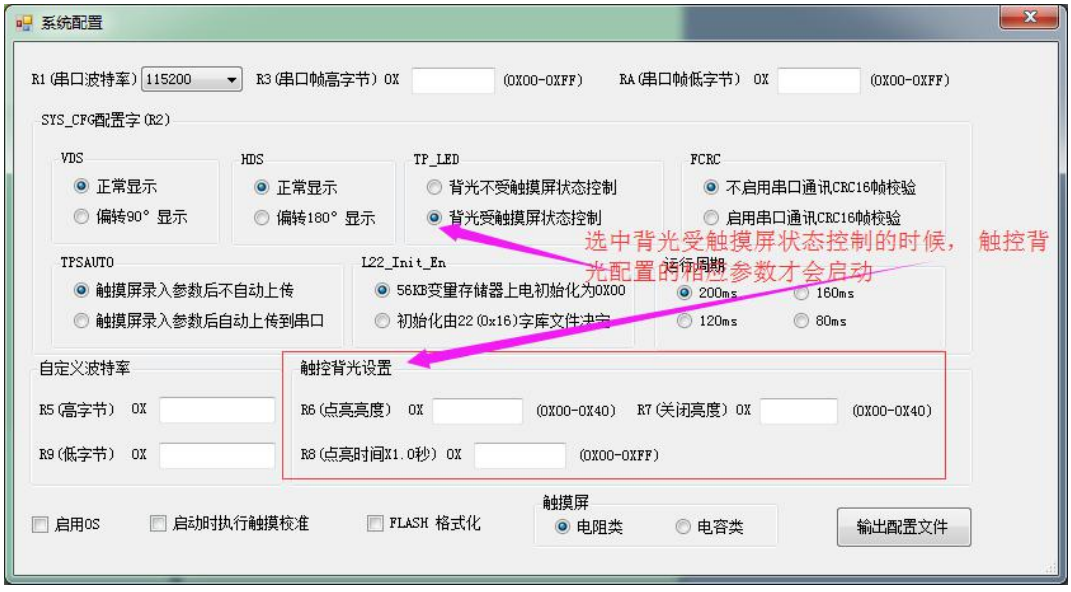


图 3.10 屏幕背光相关设置

配置方式如下表 3.9 所示。【注】背光待机后，第一次点击只是使触摸屏点亮，不会触发触控控件。

表 3.9 关于背光亮度修改的配置

寄存器	取值范围	说明
R6	0x00-0x40	唤醒触摸屏后背光的亮度。
R7	0x00-0x40	一段时间不点击触摸屏，熄屏待机时背光的亮度。
R8	0x01-0xFF	触摸屏背光熄灭的时间，单位为秒。

3.4.2 变量初始化文件 22.bin

22. bin 文件可用于对变量的显示进行初始化。当启用 22.bin 文件时，变量显示控件在上电时将显示 22.bin 文件中已定义的值。22.bin 文件可通过 KGUS 软件生成，也可通过编程软件新建，只需文件最终命名为 22*.bin（*代表任意非数字的字符，可以不写）。该文件的编辑方式如下图所示。KGUS 中每个变量地址对应两字节的空间。在图 3.11 中，变量地址 0x000A 中储存的值被修改为 “云”，变量地址 0x000B 中储存的值被修改为 “利”，即上电后所有变量地址为 0x000A 的控件均显示“云”，所有变量地址为 0x000B 的控件均显示“利”。如下图 3. 11 所示。

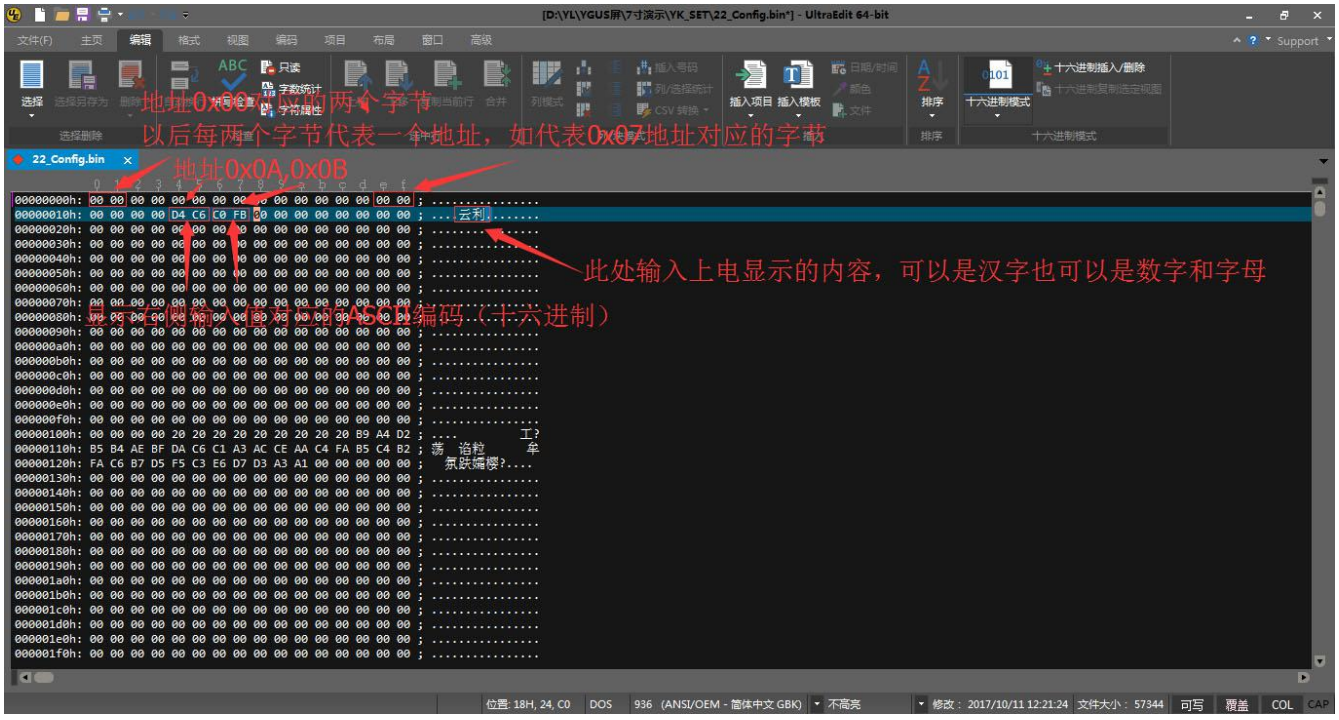


图 3.11 22.bin 文件编辑方法图示

需要注意的是，KGUS 软件默认是未启用 22.bin 文件初始化的，需要手动启用该初始化功能。22.bin 文件的启用是包含在 KGUS 参数配置中的，也就是上一节讲解的 CONFIG.txt 文件。根据 CONFIG 文件的修改方式，可通过两种方法启用 22 文件初始化。

第一种，手动在 CONFIG.txt 添加一行 R2=04 即可。（寄存器 R2 的定义详见上一节。）

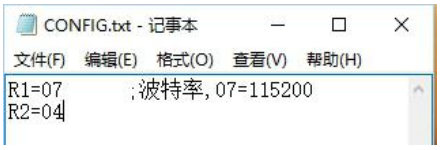


图 3.12 启用 22 文件初始化的操作方式

第二种，在 KGUS 软件中点击“配置”按钮，在弹出的界面中选择“初始化由 22 字库文件决定”，输出配置文件来更新 CONFIG 文件即可。如图 3. 13 所示。

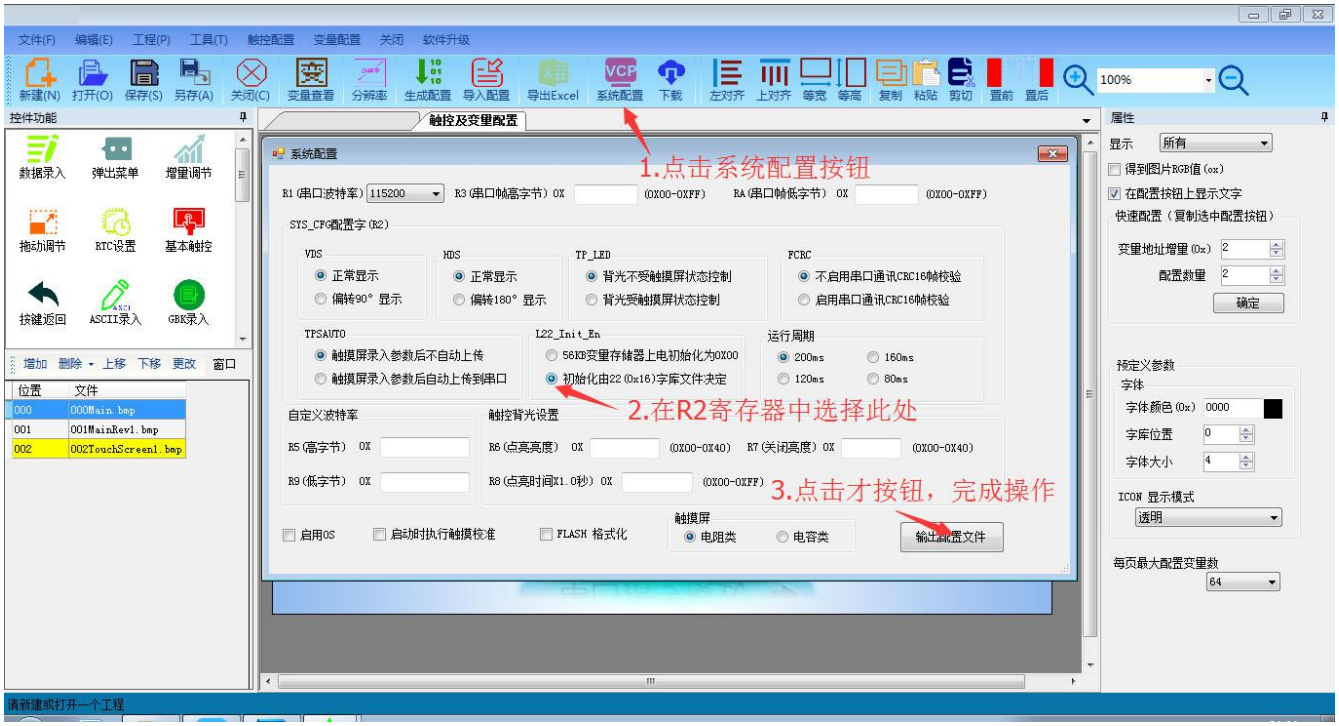


图 3.13 在 KGUS 软件中启用 22 文件

最后，将新的 22.bin 文件和 CONFIG.txt 文件放置在 YK_SET 文件夹中，通过 U 盘或 SD 卡下载至 KGUS 屏中即可。

3.5 KGUS 屏的调试

3.5.1 屏幕校准

共有四种校准方式：

1) 校准方式 1

开机状态下，在 4 秒内快速连续点击触摸屏的非触控区域（即未设定触控功能的区域）超过 20 次，则进入触摸屏的校准模式，其步骤如下：

Step1：4 秒内快速点击触摸屏非触控区域超过 20 次。

Step2：蜂鸣器长鸣 1 秒，听到蜂鸣器鸣叫时停止点击屏幕。

Step3：进入校准模式，按十字交叉线的提示点击触摸屏的指定位置来校准触摸屏。

Step4：校准结束，将自动返回校准前的画面。

2) 校准方式 2

在 CONFIG.txt 文件中，写入一行特殊的文本“TP_CORRECT”将启动一次触摸屏校准功能。

3) 校准方式 3

通过串口向 0xEA 寄存器写入 0x5A 指令将启动一次触摸屏的校准过程。

4）校准方式 4

在给 KGUS 屏上电的时候手放在屏幕上任意位置，便会启动校准功能。

3.5.2 下载工具 SD 卡的使用

KGUS 屏的所有硬件参数设置、程序下载以及 KGUS 软件升级都可以通过屏上的 SD/SDHC 接口来完成。第一次使用 SD 卡时推荐用户先对 SD 卡进行格式化，并将其文件系统设置为 FAT32 格式。

1. 通过 SD 卡下载工程

将包含图片文件、字库文件、13.bin、14.bin、CONFIG.txt 等素材文件和配置文件的 YK_SET 文件夹复制到 SD 卡中；然后，把 SD 卡插到 KGUS 屏的 SD 卡接口，KGUS 屏检测到 SD 卡后，会显示蓝屏提示用户检测到了 SD 卡，然后将根据 SD 卡中的文件自动进行屏幕参数配置或将数据下载到 KGUS 屏的 FLASH 中。SD 卡下载完成后，KGUS 屏会自动复位一次，用户拔出 SD 卡，下载结束。

注意：两次 SD 卡热插拔之间至少间隔 6 秒，否则 KGUS 屏会认为是同一张 SD 卡而不进行数据下载操作。

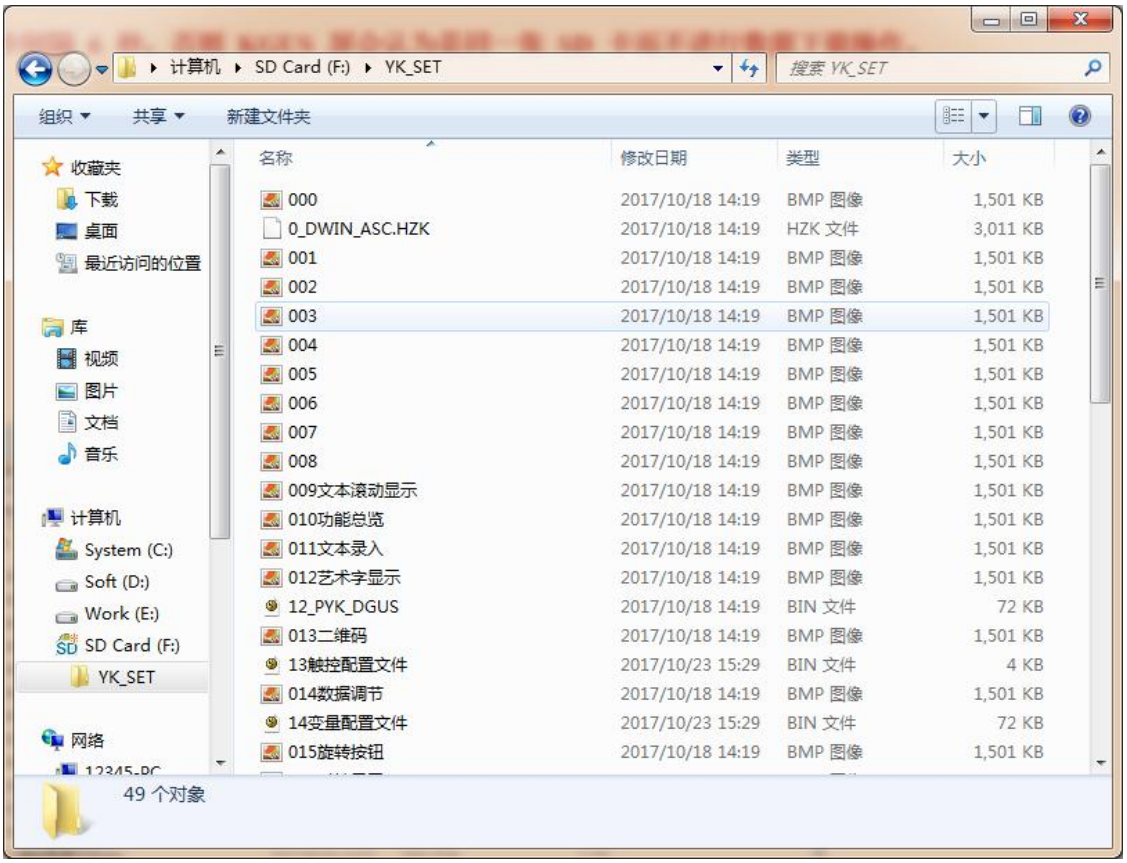


图 3.14 YK_SET 文件夹

2. 通过 SD 卡升级 KGUS 版本

KGUS 屏可通过 SD 卡对其内置的 KGUS 软件进行升（降）级。把 KGUS 更新程序（KGUS_V*.BIN）放到 SD 卡 YK_SET 目录下即可。

3. SD/SDHC 接口禁止和解锁

■ **SD/SDHC 接口禁止**

在客户测试完成正式量产后，为了防止用户通过 SD 卡进行错误的升级或下载操作从而导致工作异常，可以通过在 CONFIG.txt 文件中，增加一行特殊的文本来禁止 SD 卡接口，方式如表 3.10 所示。

表 3.10 CONFIG.txt 文档中禁止 SD 卡接口文本的说明

第 1 部分	SD_LOCK_	固定
第 2 部分	5000	用来重新启动 SD 接口的密码保存在变量存储器空间的地址，范围是 0000-6FF8
第 3 部分	66666666	重新启动 SD 卡接口的 8 位密码

注意：SD 卡被禁止后，KGUS 屏将不能更新数据、资料和校准触摸屏。请用户务必保护好 SD 卡的开锁密码，因为一旦 SD 接口卡锁死且没有开锁密码，只能通过返厂更换内核 CPU 才能再次启用。

【例】假设禁止 SD/SDHC 接口后的重新启用密码为 66666666，密码保存在变量存储空间的 0x5000 位置，则禁止 SD/SDHC 接口的步骤如下：

Step1：在 CONFIG.txt 文档中增加指令 SD_LOCK_5000_66666666

Step2：把 CONFIG.txt 用 SD 卡下载到 KGUS 屏中

Step3：KGUS 将自动禁用 SD/SDHC 接口

■ SD/SDHC 接口解锁

有三种方式解锁 SD/SDHC 接口，仍以上例进行说明：

方法 1：通过串口发送正确的密码到存储空间位置，SD 卡将被激活一次。

发送指令如下：5A A5 0B 82 60 00 **36 36 36 36 36 36 36 36**（其中后 8 组数字是密码所对应的 ASCII 码，如 6 所对应的 16 进制 ASCII 码为 36）

方法 2：使用触摸屏 ASCII 文本录入功能来设置一个“解锁“操作菜单，也可以激活一次 SD 卡。

方法 3：CONFIG.txt 文档中写入取消 SD 卡禁止的文本命令“SD_UNLOCK_密码”，存入 SD 卡去重新激活 SD/SDHC 接口。在上例中，写入的文本命令为 SD_UNLOCK_66666666。

第四章 KGUS 屏的串口通信

串口通信工具推荐（SSCOM3.2，友善串口调试助手等），本文档以使用 SSCOM3.2 为例。

KGUS 屏采用异步、全双工串口（UART），串口模式为 8n1，每次传送采用十个比特位的数据，包括 1 个起始位，8 个数据位，1 个停止位。

串口通信可采用 RS232 或 TTL 两种通信方式(具体连接方式参考第一章)。串口波特率可通过 CONFIG.txt 文件来配置。

串口的所有指令或数据都是 16 进制（HEX）格式；对于字型（2 字节）数据，总是采用高字节先传送（MSB）方式，如 0x3132 先传送 0x31。

KGUS 屏的串口接收 FIFO 为 4KB，即在 80/120/160/200ms（1 个 KGUS 周期）内可以传送至少 4KB 数据（约等于 230400-691200bps 波特率连续发送）；一个 KGUS 周期能够传送的最大数据长度取决于用户界面的复杂程度；推荐客户在一个 KGUS 周期内不要发送超过 4KB 的数据给 KGUS 屏。

4.1 检测串口通信状况

当用户确保串口接线无误后，可通过 SSCOM3.2 来进行测试以验证串口的通信是否正常。

测试方法如下：

Step1：按照上述方法正确接线，打开 SSCOM3.2 串口调试助手，设置波特率等参数如图 4.1 所示（其中串口号请根据实际连接电脑的端口来选择）。

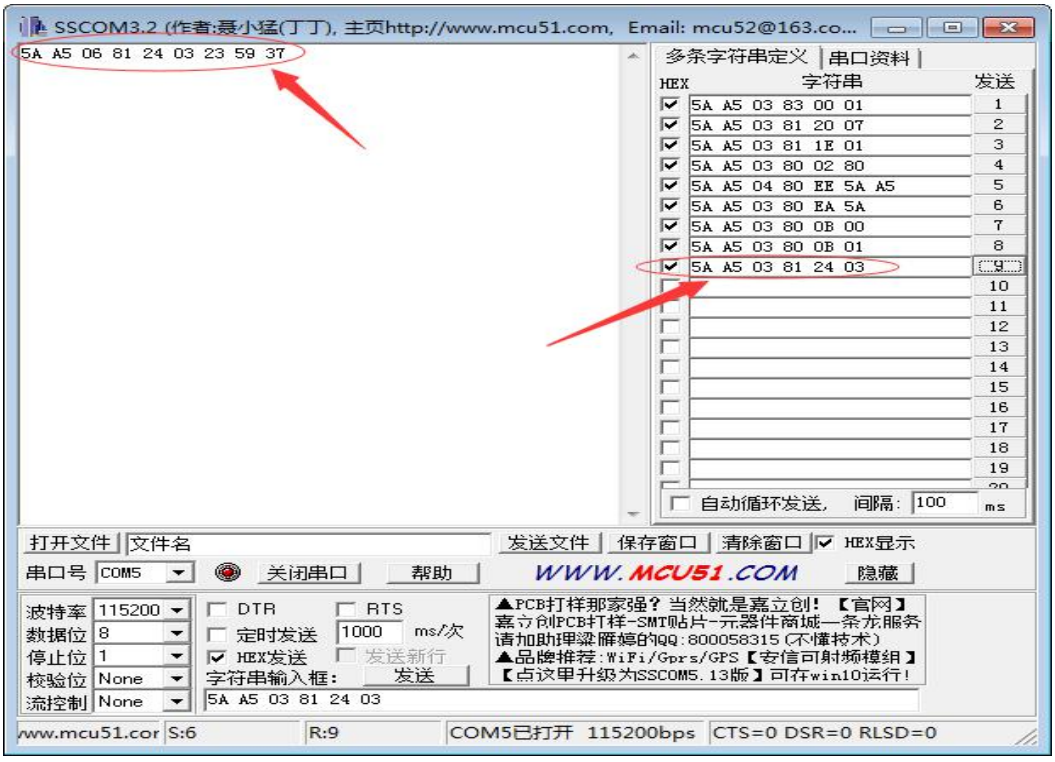


图 4.1 串口通信检测

Step2：在字符串输入框发送指令 5A A5 03 81 24 03（该指令用于查询 KGUS 屏显示的当前时间），若在上面的接收窗口中接收到大致如 5A A5 06 81 24 03 23 59 37 的应答指令，则通信正常。

4.2 通信指令说明

在前面的章节中介绍了 KGUS 把 GUI（用户图形界面）的每一个页面分解成多个控件变量，即 KGUS 屏采用变量驱动模式工作，屏的工作模式和 GUI 的状态完全由数据变量来控制。因此，串口指令也只需要对变量进行读、写即可，指令集非常简单，一共只有 5 条指令。读写指令集如表 4.1 所示。

功能	指令	数据	说明
访问寄存器	0x80	下发：寄存器地址（0x00-0xFF)+写入数据	向指定的寄存器地址写入数据
	0x81	下发：寄存器地址（0x00-0xFF)+读取字节长度（0x00-0xFF)	从指定的寄存器地址开始读取指定字节长度的数据
		应答：寄存器地址（0x00-0xFF)+数据字节长度+读取的寄存器数据	向 KGUS 屏发送读寄存器指令后，KGUS 屏的应答
	KGUS 屏有 256Byte 的寄存器，主要用于硬件操作的软件接口，按照字节（Byte）寻址操作		
访问变量存储器(RAM)	0x82	下发：变量存储器地址（0x0000-0x6FFF)+写入的变量数据	从指定的变量存储器地址开始写入数据串（字数据）到变量存储区
	0x83	下发：变量存储器地址（0x0000-0x6FFF)+读取变量数据字长度（0x00-0x7F）	从变量存储区指定地址开始读取指定字长度的字数据
		应答：变量存储器地址+变量数据字长度+读取的变量数据	向 KGUS 屏发送读变量存储器的指令后，KGUS 屏的应答
	KGUS 屏有 56KB 的变量存储器，主要用于变量数据存储，按照字（Word）寻址操作		
写曲线缓冲区	0x84 CH_Mode(Byte)+DATA0(Word)+…+DATAn		写曲线缓冲区数据： ◇ CH_Mode 的每一位（bit）对应 1 个通道； CH _Mode.0 对应 0 通道，.7 对应 7 通道； 该通道对应位置为 1 表示该通道数据存在；为 0 表示该通道数据不存在。 ◇ 数据按照低通道数据在前排列。 如：CH_Mode=0x66(01100110B)，表示后续数据格式为：（通道 1 + 通道 2+通道 5+ 通道 6）+…+（通道 1 +通道 2 + 通道 5+通道 6）。
	KGUS 屏有一个 16KB、可存储 8 条曲线趋势图的曲线缓冲区，用于用户简单、快速显示曲线。曲线缓冲区的数据都是 16 位无符号数。		

表 4.1 KGUS 指令说明

一条完整的串口指令结构如下 4.1 图所示。

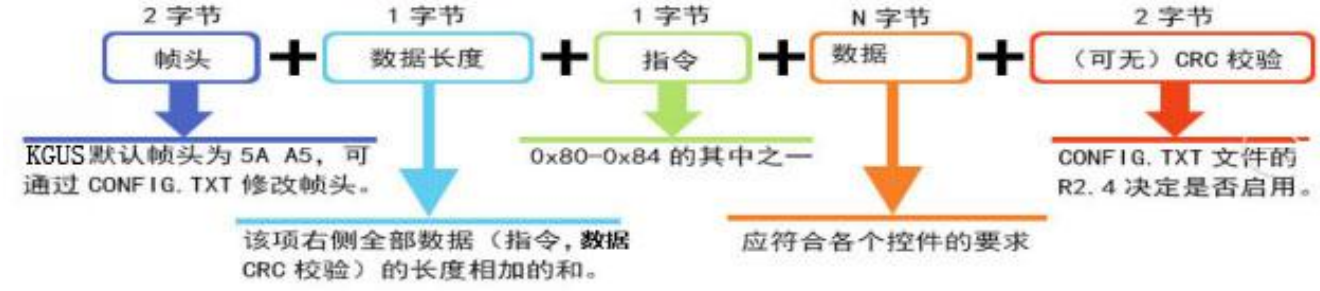


图 4.1 串口指令结构

其中，CRC 校验不包括帧头和数据长度，仅针对指令和数据进行校验。CRC 校验采 ANSI CRC-16（X16+X15+X2+1）格式。当启用 CRC 帧校验并开启自动应答功能后（R2.4=1，RC.3=1），KGUS 屏会在 CRC 校验完成后自动应答校验情况，

返回指令结构如下：

帧头+02+（KGUS 屏接收的）指令+数据（0xFF 表示 CRC 校验正确，0x00 表示 CRC 校验错误）+CRC。

【例】上位机通过串口向 KGUS 屏下发 5A A5 03 81 20 07 指令，其中 5A A5 为帧头，03 表示指令数据的字节长度，81 是读取寄存器数据的指令，20 是读取寄存器的首地址，07 为读数据的字节长度。该指令的功能是读取 0x20 寄存器中所保存的当前 RTC 值（年、月、日、星期、时、分、秒）。串口返回的数据为：5A A5 0A 81 20 07+当前 RTC 值。

4.3 常见串口通信故障排除

通信故障时，应先判断接线是否正确，尤其是使用 RS232 模块进行串口通信时，请确认模块的 TXD 口与 KGUS 屏的 232_RX 口连接，模块的 RXD 口与 KGUS 屏的 232_TX 口连接，切不可接反。

4.3.1 KGUS 屏与电脑通信故障

故障一：对于同时提供 TTL 和 232 串口的屏，如何切换串口模式？

短接下图 4.2 中的 J6 位置即可。

图 4.2 KGUS 触摸屏背面串口切换标识

故障二：已正确连接屏和电脑，调试终端设置也正确，发送指令没有返回值？

这可能是由于波特率不一致导致的。通过 CONFIG.TXT 文件将波特率和帧头重新配置成出厂默认值：

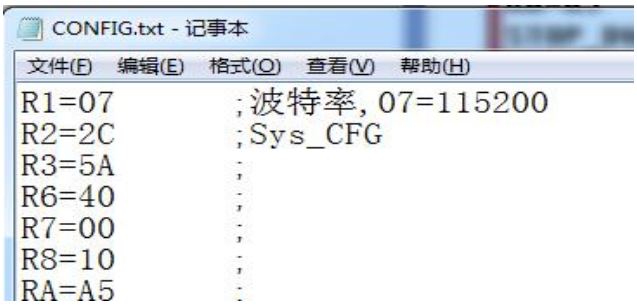


图 4.3 设置 CONFIG 文件使屏恢复出厂设置

将 YK_SET 文件夹放进 SD 卡根目录下，对屏进行“掉电–插卡–上电”操作，屏幕蓝屏 1 秒左右，读取 SD 卡里的 CONFIG.txt 文件的配置。然后通过设置调试终端并发送指令检查通讯情况。

第五章 KGUS 屏的配置寄存器

配置寄存器空间是用于存放指令状态的，比如 RTC（实时时钟）、背光亮度等实时的状态。了解寄存器的地址以及各寄存器的功能，就可以通过串口指令来实现上位机与 KGUS 屏信息传输及控制。

KGUS 提供 256 Byte 配置寄存器空间；80 指令用于向寄存器写入数据，81 指令用于读取寄存器数据。

5.1 配置寄存器功能汇总

首先给出 KGUS 屏寄存器的地址定义表和说明，用户可参考例子尝试读写操作。

表 5.1 寄存器一览表

寄存器地址	定义	R/W	字节长度	说明
0x00	Version	R	1	KGUS 版本号，以 BCD 码表示，0x12 表示 V1.2
0x01	LED_SET	W	1	背光亮度控制存储器，范围 0x00-0x40
0x02	BZ_TIME	W	1	蜂鸣器鸣叫控制寄存器，单位为 10ms
0x03	PIC_ID	R/W	2	读：当前显示的页面 ID 写：切换到指定 ID 的页面
0x05	TP_Flag	R/W	1	0x5A=触摸屏坐标更新其他=触摸屏坐标未更新 当用户读取数据后未清零这个标记，则触摸屏数据不再更新。
0x06	TP_Status	R	1	0x01=第一次按下 0x03=长按 0x02=抬起其他=无效
0x07	TP_Posion	R	4	触摸屏按压的坐标位置：X_H:L Y_H:L
0x0B	TPC_Enable	R/W	1	0x00=触控不启动 其他=触控启动上电时默认为 0xFF
0x0C-0x0F	RUN_TIME	R	4	上电运行时间，BCD 码表示时分秒，其中小时为两个字节，最大 9999：59：59
0x10-0x1C	RO-RC	R	13	SD 卡配置寄存器的映射，串口只读，串口写无效
0x1D	保留		1	未定义
0x1E	LED_STA	R	1	背光亮度返回值
0x1F	RTC_COM_ADJ	W	1	0x5A 表示用户申请通过串口改写了 RTC 数据，KGUS 修改 RTC 完成后清零。
0x20	RTC_NOW	R/W	16	读写 RTC：YY:MM:DD:WW:HH:MM:SS+农历 YY:MM:DD+天干地支+生肖
0x30-0x3F	保留		16	未定义
0x40	En_Lib_OP	R/W	1	0x5A 表示用户申请进行读字库存储器操作，当 KGUS 操作完成后清零。 每个 KGUS 周期执行一次读操作。
0x41	Lib_OP_Mode	W	1	0xA0 表示把指定字库空间的数据读入变量存储器空间。
0x42	Lib_ID	W	1	指定字库空间，范围 0x40-0x7F,每个字库 256KB，对应最大 Flash 空间 16MB.
0x43	Lib_Address	W	3	指定字库空间的数据操作首（字）地址，范围 0x00:00:00-0x01:FF:FF
0x46	VP	W	2	指定变量存储器空间的数据操作首（字）地址，范围 0x00:00-0x6F:FF
0x48	OP_Length	W	2	数据操作的（字）长度，范围 0x00:01-0x6F:FF
0x4A	Timer0	R/W	2	16bit 软件定时器，单位为4ms，自然减到0 停止。
0x4C	Timer1	R/W	1	8bit 软件定时器，单位为 4ms，自然减到 0 停止。
0x4D	Timer2	R/W	1	8bit 软件定时器，单位为 4ms，自然减到 0 停止。
0x4E	Timer3	R/W	1	8bit 软件定时器，单位为 4ms，自然减到 0 停止。
				设计值和实际运行值之间有+/-4ms 的误差，不如设置为 2，实际运行在 4-12ms 之间。
0x4F	Key_code	W	1	用户键码，用于触发 0x13 触控文件：范围 0x01-0xFF，其中 0x00 表示无效； KGUS 处理键码后会自动清零键码寄存器。
0x50	Play_Music_Set	W	3	格式为 0x5A:Play_Start:Play_Num，播放指定音乐。 Play_Start 是播放起始段，Play_Num 是连续播放段数（0x00 将停止播放）。
0x53	Volume_Adj	W	2	格式为 0x5A:VOL，调整播放音量，音量=VOL/64，上电默认值为 0x40。
0x55	保留		1	未定义

0x56	En_DBL_OP	R/W	1	0x5A 表示用户申请进行数据库存储器操作，KGUS 操作完成后清零。每个 KGUS 周期执行一次读或写操作。
0x57	OP_Mode	W	1	0x50:把变量存储器空间数据写入数据库空间。 0xA0:把数据库空间的数据读入变量存储器空间。
0x58	DBL_Address	W	4	数据库空间字地址，范围 0x00:00:00:00-1D:FF:FF:FF,最大 480MW(960MB)数据空间。数据库从物理空间存储空间的第 64MB 开始存储，和图片存储器空间有重合，每 1Byte 数据库存储器占据 2Byte 物理存储器。使用 SD 卡导出数据库时， 每个字库大小为 128KB，编号从 236 开始，960MB 读写时，KGUS 会自动处理跨字库情况。
0x5C	VP	W	2	指定变量存储器空间的数据操作首（字）地址，范围 0x00:00-0x6F:FF
0x5E	OP_Length	W	2	数据库操作的（字）长度，范围 0x00:01-0x6F:FF
0x60-0xE8	保留		137	未定义
0xE9	Scan_Status	R	1	0x01=触摸屏处于录入状态 0x00=触摸屏未处于录入状态
0xEA	TPCal_Triger	W	1	写入 0x5A 启动一次触摸屏校准，校准完成后会被 KGUS 清零。
0xEB	Trendline_Clear	W	1	写入特殊定义的数值以清除对应的曲线缓冲区数据。 0x55:清除全部 8 条曲线缓冲区数据。 0x56-0x5d:分别清除 CH0-CH7 通道的曲线缓冲区数据。 曲线缓冲区数据清除完成后，KGUS 会将本寄存器清零。
0xEC-0xED	保留		2	保留
0xEE-0xEF	Reset_Triger	W	2	写入 0x5AA5 会使 KGUS 屏软件复位一次
0xF0-0xFF	保留		16	保留

【例 1】背光亮度值查询及调节。

背光亮度存储在地址为寄存器 0x01 空间中。通过发送指令读取背光亮度值，如下图 5.1 所示。

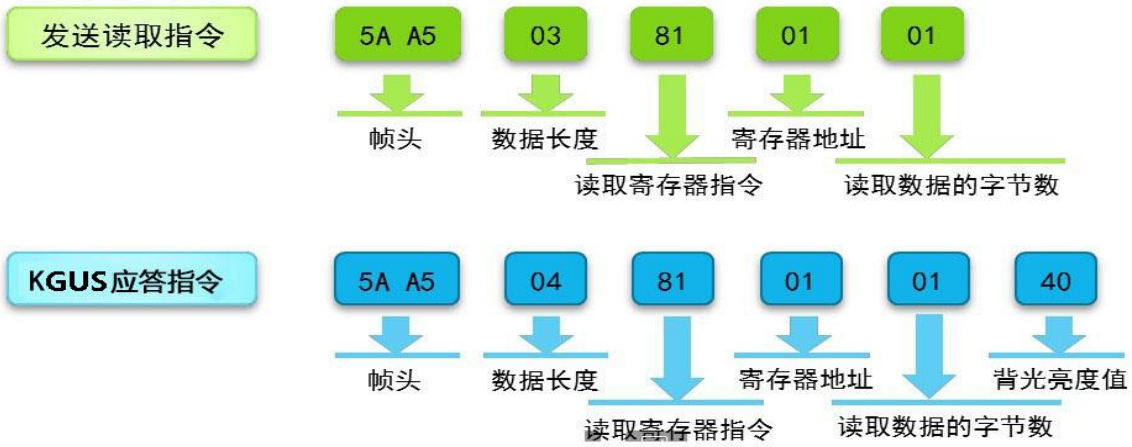


图 5.1 发送指令读取屏幕背光亮度

发送指令查询背光亮度，收到 KGUS 屏的应答为 5A A5 04 81 01 01 40，最右的 0x40 为读取的数据，即背光亮度。KGUS 屏亮度范围是 0x00-0x40，可知当前屏幕处于最亮的状态。需调节背光亮度时，可发送写寄存器指令，如下图所示。指令最右端 0x10 为调节后的背光亮度，发送指令可观察到屏幕变暗。可再次发送查询指令，屏幕应答 5A A5 04 81 01 01 10，进一步证明亮度修改成功。如下图 5.2 所示。

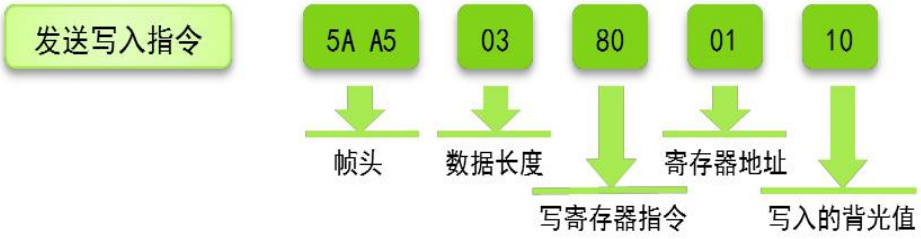


图 5.2 发送指令调节背光亮度

注意：若系统配置寄存器 R2.5=1（开启熄屏待机和触摸唤醒屏幕功能），则在通过指令调节背光时需 5A A5 03 80 01 10，5A A5 03 80 01 3F 两条指令一起发送。

【例 2】读取页面 ID 和切换页面。

页面 ID 存储在 0x03 寄存器空间。通过发送指令读取当前页面编号，如下图 5.3 所示，当前显示页面 ID 为 0。

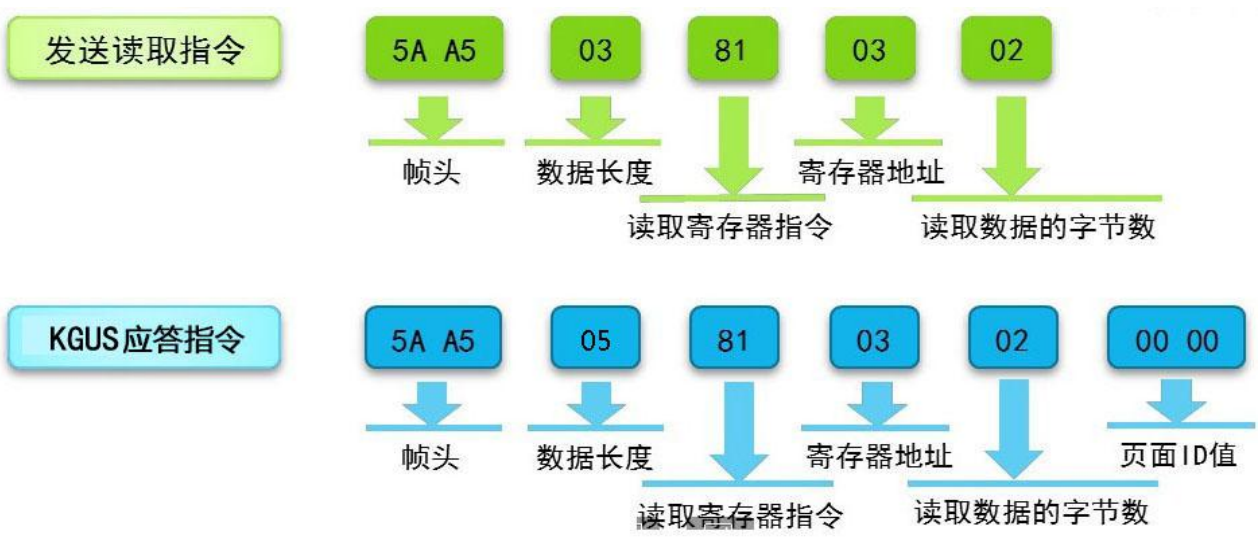


图 5.3 发送指令读取页面 ID

通过发送指令还可实现页面切换，如下图 5.4 所示，指令效果为切换至 ID 为 0x10 的页面，即编号为 16 的页面。若并未下载 ID 为 16 的页面至 KGUS 屏中，则会显示黑屏。

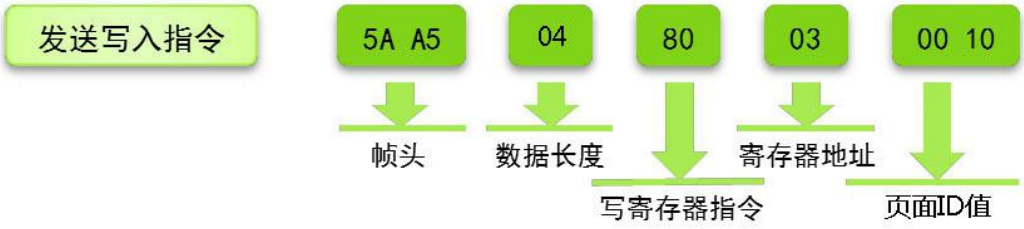


图 5.4 发送指令切换页面

通过读写配置寄存器，还可实现多种功能：

启动蜂鸣	5A A5 03 80 02 20	单位 10ms，最长鸣响时间为 FF
复位	5A A5 04 80 EE 5A A5	相当于掉电/上电操作
触摸屏校准	5A A5 03 80 EA 5A	
关闭触摸功能	5A A5 03 80 0B 00	
开启触摸功能	5A A5 03 80 0B 01	

5.2 寄存器应用举例

5.2.1 RTC 的读写

1. 读 RTC

RTC 相关寄存器如表 5.1 所示。

表 5.1 RTC 相关寄存器

0x1F	RTC_COM_ADJ	W	1	0x5A 表示用户申请通过串口改写了 RTC 数据，KGUS 修改 RTC 完成后清零。
0x20	RTC_NOW	R/W	16	读写 RTC：YY:MM:DD:WW:HH:MM:SS+农历 YY:MM:DD+天干地支+生肖

从表中可知，以 0x20 为首地址的 16 个字节的寄存器空间存储了当前的时间，其公历时间储存在前 7 个字节中。若要读取公历（年月日星期时分秒），可通过串口助手发送指令：5A A5 03 81 20 07，如图 5.5 所示。屏幕应答为 5A A5 0A 81 20 07 17 11 08 03 00 03 04，即现在的时间是 17 年 11 月 8 日星期三 00:03:04。

若只要读取时间（时分秒），从 RTC 存储格式得知从 24 到 26 的三个字节存储的 HH:MM:SS，因此发送指令 5A A5 03 81 24 03 即可。

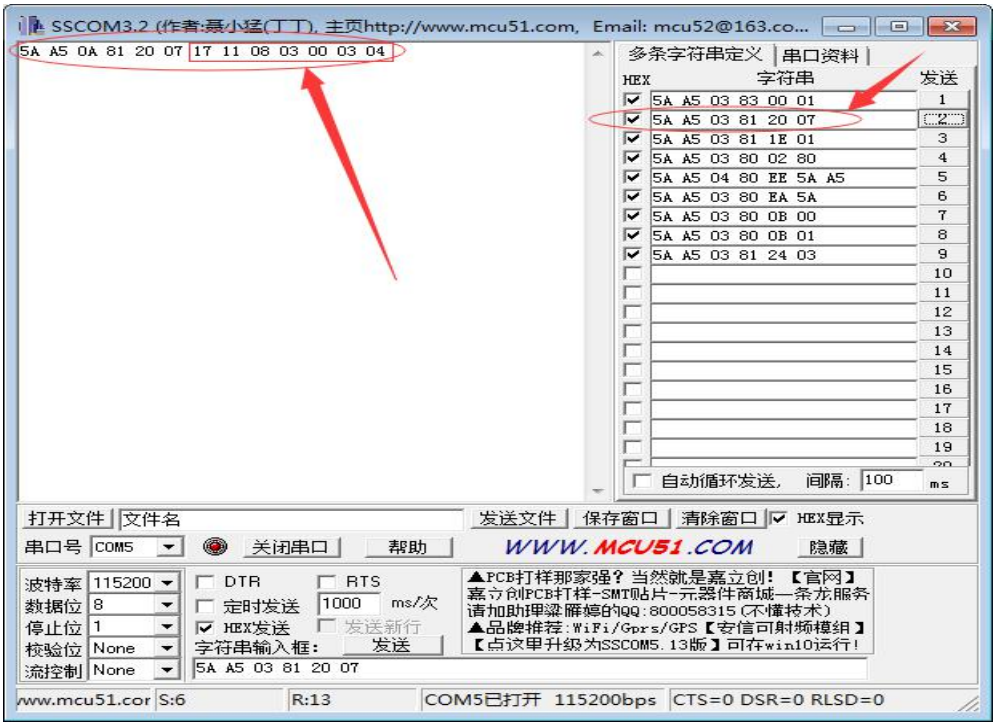


图 5.5 发送指令读取当前时间

2. 写 RTC

如果发现 KGUS 屏的时钟不准需要校准时，可向 RTC 寄存器写入时间数据。

Step1：需要发送改写 RTC 的申请，即向 0x1F 寄存器写入数据 0x5A，指令为 5A A5 03 80 1F 5A。

Step2：向以 0x20 为首地址的 RTC 寄存器中写入想要修改的数据即可。例如，将时间设定为 2016 年 1 月 25 日 17：08：20，则发送 5A A5 09 80 20 16 01 25 **00** 17 08 20。

还可将上面两个指令合并以简化操作，即直接从 1F 地址处写入 5A+欲修改的时间，发送的指令 5A A5 0A 80 1F 5A 16 01 25 **00** 17 08 20。

【注】写 RTC 时，只需要改写公历的年月日时分秒即可，KGUS 屏会自动换算星期和农历。在上面的例子中，改写星期位置的数据就随便写为了 00。

5.2.2 字库的读取

前面介绍了字库空间，第 24-127 号字库空间可储存用户自定义的字库文件和图标库文件，其中**仅第 64~127 字库可通过串口指令读取到变量存储器（RAM）中**。读取到变量储存器中的数据可通过 0x82 指令再次读取。

表 5.2 字库相关寄存器

0x40	En_Lib_OP	R/W	1	写入 0x5A 表示启动读字库数据操作。 当 KGUS 操作完成后清零。每个 KGUS 周期执行一次读操作。
0x41	Lib_OP_Mode	W	1	写入 0xA0 表示把指定字库空间的数据读入变量存储器空间。
0x42	Lib_ID	W	1	指定字库空间，范围 0x40-0x7F,每个字库 256KB，对应最大 Flash 空间 16MB.
0x43	Lib_Address	W	3	指定字库空间的数据操作首（字）地址，范围 0x00:00:00-0x01:FF:FF
0x46	VP	W	2	指定变量存储器空间的数据操作首（字）地址，范围 0x00:00-0x6F:FF
0x48	OP_Length	W	2	数据操作的（字）长度，范围 0x00:01-0x6F:FF

【例】从 82 号（0x52）字库的 0x000010 地址开始读取 8 KB（8*1024/2=4096 Word，即 0x1000 Word）数据到首地址为 0x1000 的 RAM 空间中。发送指令时，按序将要写入寄存器中的数据放入指令中发送即可。**读取数据不能超过字库空间，即 Lib_Address+OP_Length<256KB 即 0x02 00 00）。**

发送指令：**5A A5 0C 80 40 5A A0 52 00 00 10 10 00 10 00**

40：寄存器首地址

5A：0x40 中写入 5A 表示启动读字库操作

A0：0x41 中写入 A0 表示将字库数据读入 RAM 空间

52：字库文件编号

00 00 10：从字库文件的 0x000010 地址开始读数据

10 00：将读取的数据写入首地址为 0x1000 的 RAM 空间中

10 00：要读取的数据总长度（单位：字）

5.2.3 数据库的读写

表 5.3 数据库相关寄存器

0x56	En_DBL_OP	R/W	1	0x5A 表示启动数据库读写操作，KGUS 操作完成后清零。每个 KGUS 周期执行一次读或写操作。
0x57	OP_Mode	W	1	0x50：把变量存储器空间数据写入数据库空间。 0xA0：把数据库空间的数据读入变量存储器空间。
0x58	DBL_Address	W	4	数据库空间起始（字）地址，范围 0x0000 0000~0x1DFF FFFF，最大 960MB 数据空间。每 1Byte 数据库存储器占据 2Byte FLASH。
0x5C	VP	W	2	指定变量存储器空间的数据操作首（字）地址，范围 0x0000-0x6FFF
0x5E	OP_Length	W	2	数据库操作的（字）长度，范围 0x0001-0x6FFF

读写数据库时，首先需要知道数据库在 FLASH 空间中的储存位置，也即数据库的首地址。从讲解 FLASH 储存空间的章节中，我们了解到可以分配给数据库的空间是有上限的（详见 3.3 节）。给数据库分配最大储存空间时，对应了最小的图片空间，表 5.4 中给出了计算数据库空间首地址所需的参数对照表。

表 5.4 图片 ID 及存储系数 K1 对照表

分辨率	320*240	480*272	640*480	800*480	800*600	1024*600	1024*768
K1	1	1	3	3	4	5	6
PIC_ID	128	128	42,43	42,43	32	25,26	21,22

PIC_ID：给数据库空间分配最大值时，最多可储存的图片数量，同时，也是此时可使用的最大图片编号。表格中部分分辨率有两个 PIC_ID，如 640*480 分辨率中 PIC_ID 为 42 和 43，意味着最大图片 ID 为 42，但由于一张图片占用了多个空间，数据库应当从原 44 号图片储存位置开始储存。

计算数据库空间起始地址

设有 N 幅图片需要存储，则数据库的最小首地址=((N*K1)-128)*64*1024

注：N 大于 PIC_ID；128，64，1024 均为常数，可直接带入计算。

【例】480*272 分辨率下，预留 200 幅图片的空间，那么数据库的最小起始地址 Adr_Min 为：

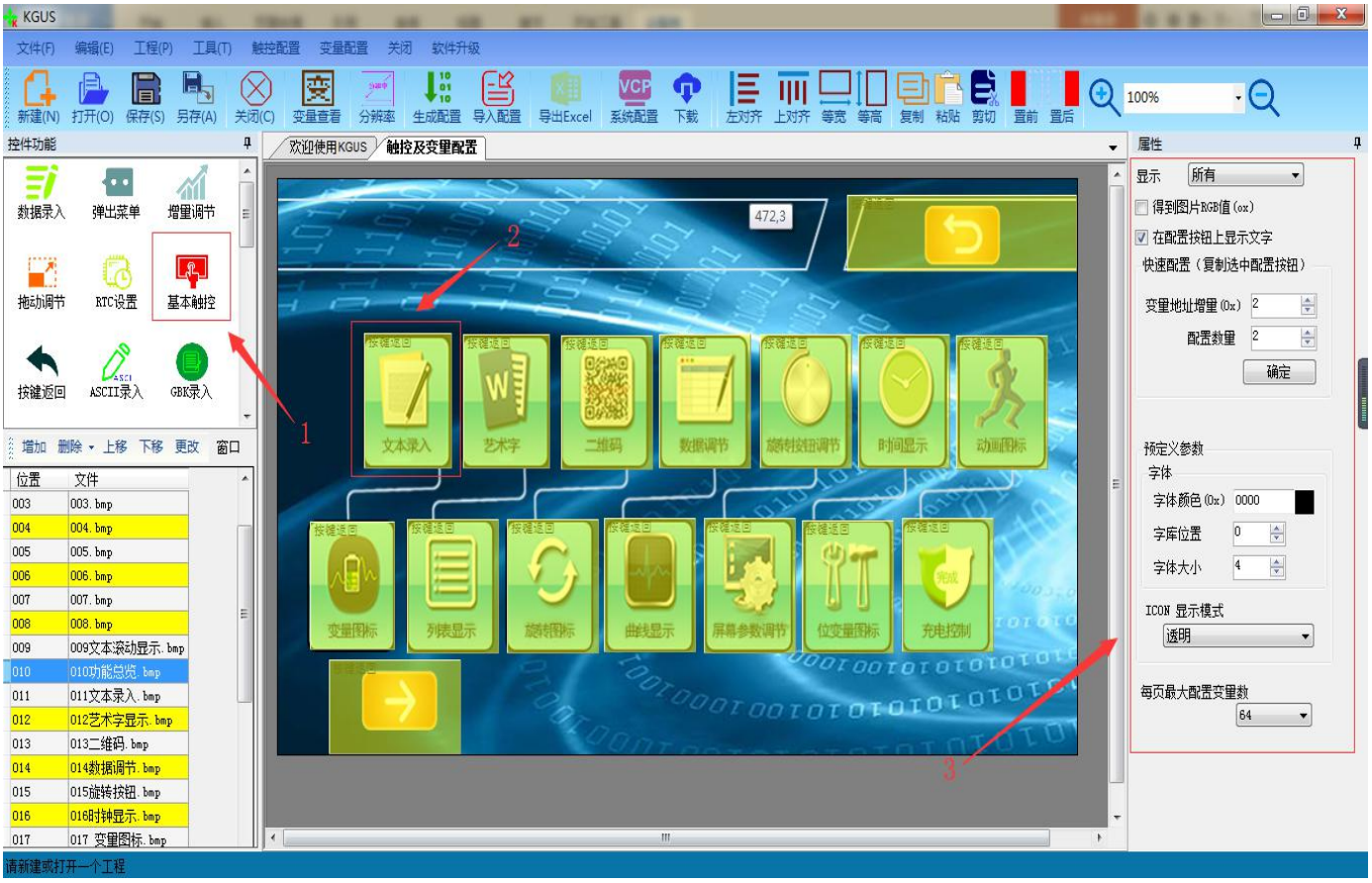
Adr_Min=((200*1)-128)*64*1024 =4718592= 0x0048 0000 **【每 1Byte 数据库存储器占据 2Byte FLASH】**

使用 SD 卡导出数据库时，每个字库大小为 128KB，编号从 236 开始。KGUS 会自动处理跨字库情况。用户读写数据库的过程中，KGUS 会对数据进行加密和纠错操作，以确保数据存储的可靠性。用户数据库在 FLASH 中是由若干个大小为 128KB 的数据库构成，每个页面写寿命是 10 万次（启动 1 次写操作则减少 1 次写寿命），但是读写操作中的地址是连续的，不会受到分页的影响，KGUS 会自动地处理分页的问题。

第六章 触控/键控配置文件说明

在 KGUS 屏的开发设计中，触控/键控配置与显示变量配置是设计的关键。本章将对触控/键控配置文件（13.BIN）进行详细的介绍与讲解。

触控/键控配置文件可以利用 PC 端的 KGUS 开发软件非常便捷的设置与生成。比如要在页面的某个位置添加一个基本触控功能，只需要点击基本触控图标（如图 6.1 中 1 所示），并在图片的指定区域划取一个触发范围（如图 6.1 中 2 所示），之后在界面右侧弹出的基础触控设置选单中对触控功能进行简单的设置就能够完成（如图 6.1 中 3 所示）。

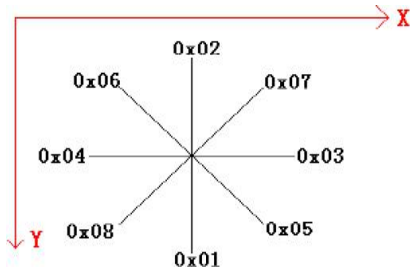


（详细的 KGUS 软件使用方法可以参考公司官网上的 KGUS 软件视频教程或者找销售工程师索取）

图 6.1 基础触控设置

为了便于用户更深入地理解，本章将以指令的思维方式对触控/键控配置文件进行更深入地讲解。触控配置文件存储在字库空间中，它是由 N 条按照页面配置的触控指令组成，每条触控指令固定占用 16、32 或者 48 字节存储空间。一条触控指令由 6 部分组成，如表 6.1 所示。

表 6.1　一条触控指令的组成

序号	定义	数据字节长度	说 明
1	Pic_ID	2	页面 ID
2	TP_Area	8	 触控按钮区域可由其左上角坐标（Xs,Ys）和右下角坐标（Xe,Ye）来确定。当 Xs=0xFFFF 时，表示触发控制器由 0x4F 寄存器的键码值触发，此时 Ys_H（即 Ys 地址的高字节）为设定的触发键码值（Ys_L，Xe，Ye 的值可任意写）；由键码值触发时，请把按钮按压效果设置为无效。 触摸屏手势动作识别键码固定为 0x01-0x08（方向和键码对应关系如上图所示），识别出手势后自动写入 0x4F 寄存器，是否启用由寄存器 RC.1 来控制。（RC.1=1 时开启触摸屏手势识别功能）。
3	Pic_Next	2	按钮按压操作后的目标切换页面，0xFF**表示不进行页面切换。
4	Pic_On	2	按钮按压时的效果图所在的页面，0xFF**表示没有按钮按压效果。
5	TP_Code	2	触控键码： 0xFF**表示无效键码。 0xFE**或 0xFD**表示触控功能按键，比如 0xFE00 表示启动触摸屏数字录入。其中 00 为触控键码值，可参考触控/键控功能一览表。 0xFE**的功能按键可以由 R2.3 设置为变量改变后是否自动上传（R2.3=0 时触摸屏录入参数后不自动上传，R2.3=1 时是否自动上传由相应触控变量的配置决定），0xFD 的功能按键始终禁止变量改变后自动上传。 其他表示触控键码，用 ASCII 表示；比如 0x0031 表示按键“1”。
6	TP_FUN	32	当 TP_Code=0xFE**时，用来对触控功能按键进行描述。

下面将为用户介绍触控/键控的各种功能，并对一些常用的功能分进行详细地讲解。

6.1 触控/键控功能一览表

触控/键控配置文件能够让 KGUS 屏实现多个功能，功能如表 6.2 所示。

表 6.2 KGUS 屏实现的功能

序号	触控键码	功能	说 明
01	00	数据录入	录入整数、定点小数等各种数据到指定变量存储空间。
02	01	弹出菜单选择	点击触发一个弹出菜单，返回菜单项的键码。
03	02	增量调节	点击按钮，对指定变量进行++/--操作，可设置步长和上下限。 设置 0-1 范围循环调节，配合图标变量显示可以实现栏目复选框功能， 如点击一下“选中”，再次点击“取消”。
04	03	拖动调节	拖动滑块实现变量数据录入，可设置刻度范围。
05	04	RTC 设置	KGUS 屏通过触摸键盘设置 RTC，需要完整录入公历（年、月、日、时、分、秒）
06	05	按键值返回	点击按键，直接返回按键值到变量，支持位变量返回。
07	06	文本录入	使用文本方式录入各种字符，录入过程支持光标移动和编辑。 支持 ASCII 字符、GBK 中文及繁体注音输入法录入； 修改字库和 0#字库可以支持所有类似 ASCII 字符的 8bit 编码文字录入； 配合 K OS 可以实现 Unicode 或多语种混合录入。
08	07_00	寄存器写到变量空间	提供了通过触摸屏改写寄存器空间的方法，间接控制硬件。
09	07_01	变量空间写到寄存器	比如把背光寄存器读到变量，调节变量后再回写来调节背光亮度。
10	07_02	图像转成单色位图（纵向）	把指定区域的彩色位图转换成单色位图并保存在 VP 指定的变量区域。
11	07_05	图像转成单色位图（横向）	主要用于当前屏幕显示内容的打印输出。
12	07_03	发送数据到 COM1	点击触摸屏，把指定 VP 区域的数据发送到用户串口（COM1）.
13	07_04	发送数据到 COM2	点击触摸屏，把指定 VP 区域的数据发送到用户串口（COM2）。 COM2 用于 KGUS 屏功能扩展，并没有引出。.
14	07_06	发送触摸屏坐标到 COM2	点击触摸屏，将把点击位置坐标发送到扩展串口（COM2）。
15	08	触摸屏按压状态数据返回	点击触摸屏，按照规定返回数据到变量或串口。
16	09	转动调节	转动旋钮实现变量数据录入，可设置刻度范围。

6.2 触摸屏数据录入

触摸屏的数据录入即录入数据到指定的变量存储空间中。录入的数据包含数字、字符及汉字等。这些数据显然无法直接存储到变量存储空间（比如字符 e 需要转换为对应的 ASCII 码 0x45），因而需要用触摸键盘将输入的内容映射为对应的键值。

【例】将一个数字键盘的每个数字键都映射为对应的 ASCII 码，首先用 PC 端 KGUS 开发软件在键盘页面的键值区域添加基本触控，即点击基本触控按钮并框选中数字 7 键的区域，在右侧的菜单中键值(0x)中输入对应的值 0037（0037：将数字 7 定义为 ASCII 码 0x37），具体对应的值下面蓝色字体已经给出讲解。如图 6.2 所示。

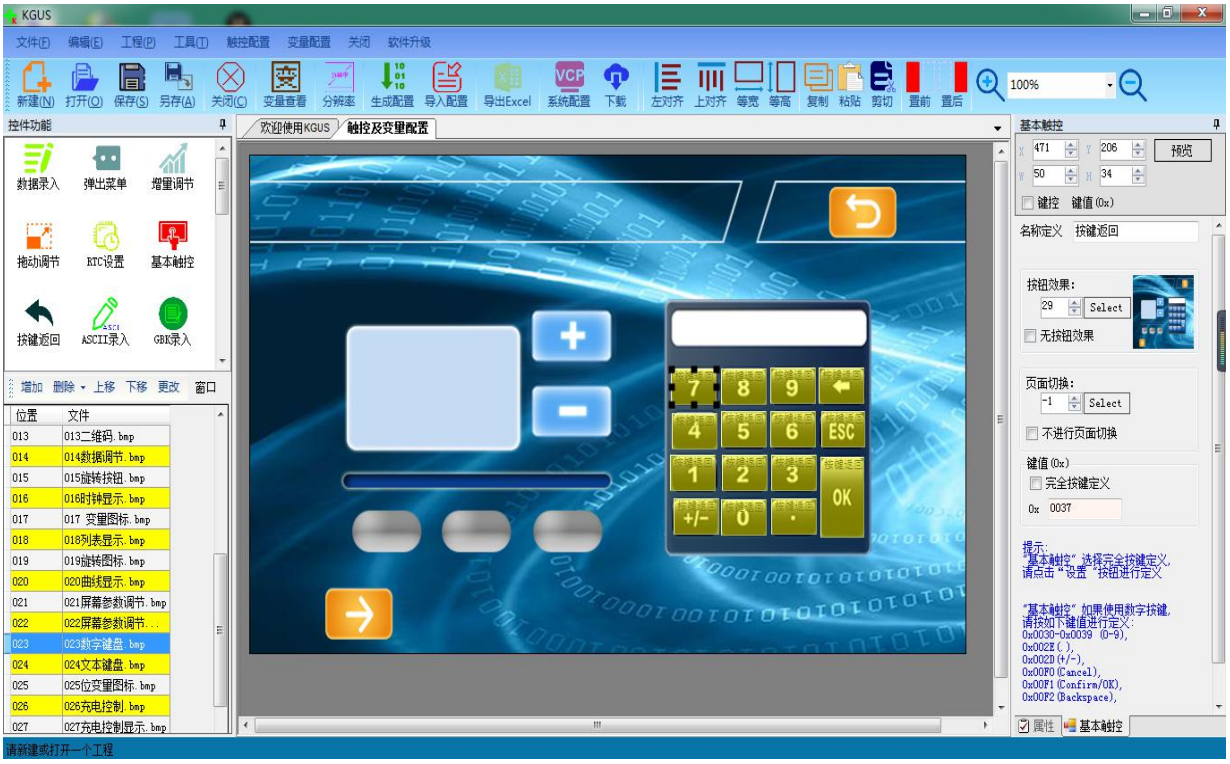


图 6.2 基础触控键码的设置

其余按键的定义与此类似，即可完成数字键盘的定义。

6.2.1 数据录入

数字录入即录入整数、定点小数等各种数据到指定变量存储空间。其指令存储格式如表 6.3 所示。

表 6.3 数字录入指令存储格式

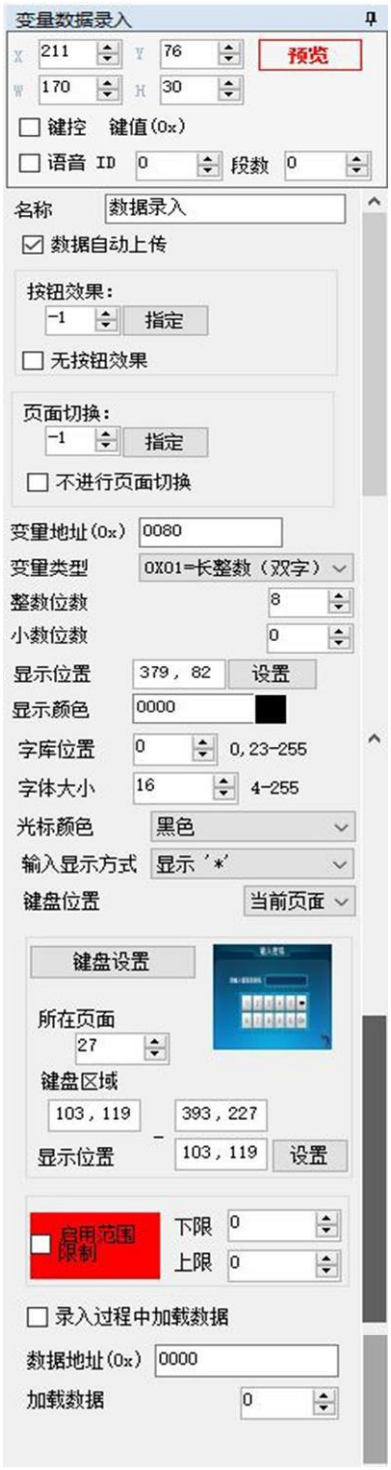
地址	定义	数据长度	说 明
0x00	Pic_ID	2	页面 ID
0x02	TP_Area	8	触控按钮区域：左上角坐标（Xs,Ys），右下角坐标（Xe,Ye）。
0x0A	Pic_Next	2	目标切换页面，0xFF**表示不进行页面切换。
0x0C	Pic_On	2	按钮按压效果图所在的页面，0xFF**表示没有按钮按压效果。
0x0E	TP_Code	2	触控键码的值为 0xFE00，表示启动触摸屏数字录入。
0x10	0xFE	1	固定值 0xFE
0x11	*VP	2	录入数据对应的变量地址指针
0x13	V_Type	1	返回变量类型： 0x00 表示 2 字节变量： 整数：-32768 到 32767； 无符号整数：0 到 65536 0x01 表示 4 字节变量：

			长整数：-2147483648 到 2147483647 无符号长整数：0 到 4294967295 0x02 表示*VP 高字节，无符号数：0 到 255 0x03 表示*VP 低字节，无符号数：0 到 255 0x04 表示 8 字节超长整数：-9223372036854775808 到 9223372036854775807
0x14	N_Int	1	录入的整数位数。如录入 1234.56，则 N_Int=0x04。
0x15	N_Dot	1	录入的小数位数。如录入 1234.56，则 N_Int=0x02。
0x16	（x，y）	4	输入过程显示位置：右对齐方式，（x，y）是字符串输入光标的右上角坐标。
0x1A	Color	2	输入字体的显示颜色。
0x1C	Lib_ID	1	显示使用的 ASCII 字库位置，0x00 为默认字库。
0x1D	Font_Hor	1	字体大小，x 方向点阵数
0x1E	Cusor_Color	1	光标颜色，0x00 表示黑色，否则为白色。
0x1F	Hide_En	1	0x00 表示录入过程中的文字不直接显示，显示为"*"；为其他值时则直接显示输入过程的内容。
0x20	0xFE	1	固定值 0xFE
0x21	KB_Source	1	0x00 表示键盘在当前页面；其他值表示键盘不再当前页面。
0x22	PIC_KB	2	键盘所在页面 ID，仅当 KB_Source 不等于 0x00 时有效。
0x24	AREA_KB	8	键盘区域坐标：左上角坐标（Xs,Ys），右下角坐标（Xe,Ye) 仅在键盘不在当前页时有效，即 KB_Source 不等于 0x00。
0x2C	AREA_KB_Position	4	键盘在当前页面显示位置的左上角坐标，仅在键盘不在当前页时有效。
0x30	0XFE	1	固定值 0XFE
0x31	Limite_En	1	0xFF：表示启用输入范围限制，输入越界无效（等同取消）； 为其它值时表示输入无范围限制。
0x32	V_min	4	输入下限，4 字节（长整数或无符号长整数）。
0x36	V_max	4	输入上限，4 字节（长整数或无符号长整数）。
0x3A	Return_Set	1	0x5A：录入过程中向 Return_VP 地址（0x3B）加载 Return_Data（地址 0x3D 的内容），结束后自动恢复。 0x00:录入过程中不加载数据。 加载数据功能：主要用于和变量显示的 SP（描述指针）修改结合，实现对多参数录入过程自动标示，比如修改字体颜色、大小、启动一个（位）变量图标或者区域反色。也可以作为录入过程的标记位，配合 DWIN_OS 开发实现特殊需求。
0x3B	Return_VP	2	录入过程中加载数据的 VP 地址。
0x3D	Return_Data	2	录入过程中加载到 Return_VP 的数据。
0x3F	保留	1	写 0x00

【注】数字录入的有效键码为 0x0030-0x0039，0x002E(.)，0x002D(+/-)， 0x00F0(取消)，0x00F1 (确定)，0x00F2(退格)

数字录入功能可以简单地通过 PC 端 KGUS 开发软件来实现。打开 KGUS 软件，点选触控配置菜单下的“数据录入”按钮，接下来用鼠标框选一个区域，就可在右侧的菜单中对该功能进行设置。如图 6.3 所示。

首先在触控配置中找到数据录入控件然后在需要录入数据的图片的划取区域，此时右边会弹出属性框，对属性框进行操作，具体操作如下



区域范围设置：设置触控按钮区域

预览：查看触控按钮的效果；

名称定义：为按钮设置一个名称，在“变量查看”中方便查询；

数据自动上传：勾选后，数据将上传至串口；

按钮效果：按钮按压效果图所在页面（-1 时默认为没有按压效果）。

页面切换：指定到切换目标图片；

变量地址：定义数据存储地址；
变量类型：0x00=整数（字）；0x01=长整数（双子）；0x02=无符号字节参数（变量地址高字节）；0x03=无符号字节参数（变量地址低字节）；0x04=超长整数（8 字节）；
整数位数：录入数据整数位数；
小数位数：录入数据小数位数；
显示位置：输入过程中数据的显示位置；
显示颜色：输入过程中字体显示的颜色，可以手动填写；
字库位置：显示的 ASCII 字库位置，“0”表示 0 字库。
字体大小：X 方向的点阵数目；
光标颜色：黑色/白色；
输入显示方式：直接显示/显示为“*”；

键盘位置：其他页/当前页。

键盘设置：设置键盘所在页面及键盘区域；

所在页面：选择键盘所在页面；

键盘区域：键盘所在页的键盘区域；

显示位置：键盘在当前页的位置（当键盘不在当前页时）；

启用范围限制：勾选后规定输入数字的上下限（越界不能录入）。（**范围限制的限值是整数位加上小数位的取值范围，例如设了 3 个整数位，2 个小数位，则取值上限为 10000，而不是 100。**）

【注】录入数据后亦可通过“数据变量、艺术字变量”等将该数据显示出来。

图 6.3 数字录入功能设置

如果想要通过串口屏录入数据，就需要将键盘的所有键进行赋值使能，注意：赋值的区域切不可出现交叠或者覆盖现象，否则将会导致工程不能正确生成。

6.2.2 文本录入

文本录入包含 ASCII 码和 GBK 汉字的录入，字符与键码按照标准的 ASCII 码进行定义，ASCII 码列表如表 6.4 所示。

表 6.4 ASCII 码列表

键码	普通	大写	键码	普通	大写	键码	普通	大写	键码	普通	大写
0x7E60	`	~	0x5171	q	Q	0x4161	a	A	0x5A7A	z	Z
0x2131	1	!	0x5777	w	W	0x5373	s	S	0x5878	x	X
0x4032	2	@	0x4565	e	E	0x4464	d	D	0x4363	c	C
0x2333	3	#	0x5272	r	R	0x4666	f	F	0x5676	v	V
0x2434	4	$	0x5474	t	T	0x4767	g	G	0x4262	b	B
0x2535	5	%	0x5979	y	Y	0x4868	h	H	0x4E6E	n	N
0x5E36	6	^	0x5575	u	U	0x4A6A	j	J	0x4D6D	m	M
0x2637	7	&	0x4969	i	I	0x4B6B	k	K	0x3C2C	,	<
0x2A38	8	*	0x4F6F	o	O	0x4C6C	l	L	0x3E2E	.	>
0x2839	9	(	0x5070	p	P	0x3A3B	;	:	0x3F2F	/	?
0x2930	0	)	0x7B5B	[	{	0x2227	‘	“	0x2020	SP	SP
0x5F2D	-	_	0x7D5D	]	}	0x0D0D	Enter	Enter			
0x2B30	=	+	0x7C5C	\	|						

【注】表格中低字节表示小写键码，高字节表示大写键码。如 0x61 对应 a，0x41 对应 A。又如 0x31 对应 1，0x21 对应!。

文本键盘的键码定义须小于 0x80(ASCII 码)。0x0D 键码录入会自动转换成 0x0D 0x0A；0x00 和 0xFF 键码禁用。此外，键盘还有一些功能按键，键码定义如表 6.5 所示。

表 6.5 功能按键，键码定义

键码	定义	说明
0x00F0	Cancel	取消录入返回，不影响变量数据。
0x00F1	Return	确认录入返回，录入文本保存到指定的变量位置。
0x00F2	Backspace	向前（退格）删除一个字符。
0x00F3	Delete	向后删除一个字符。
0x00F4	CapsLock	大写锁定。如果启用，对应按钮必须定义按钮按下的效果。
0x00F7	Left	光标前移一个字符；GBK 汉字录入中用于翻页。
0x00F8	Right	光标后移一个字符；GBK 汉字录入中用于翻页。

现在我们分别介绍 ASCII 码录入和 GBK 汉字录入

（1）ASCII 码录入的指令存储格式如表 6.6 所示。

表 6.6 ASCII 码录入的指令存储格式

地址	定义	数据长度	说 明
0x00	Pic_ID	2	页面 ID
0x02	TP_Area	8	触控按钮区域：（Xs,Ys），(Xe,Ye)
0x0A	Pic_Next	2	目标切换页面，0xFF**表示没有按钮按压效果。
0X0C	Pic_On	2	按钮按压效果图所在页面，0xFF**表示没有按压效果。
0x0E	TP_Code	2	0xFE06（即文本录入的触控键码）。
0x10	0xFE	1	0xFE
0x11	*VP	2	变量地址指针

0x13	VP_Len_Max	1	文本变量最大长度，字（Word）数目，范围为 0x01-0x7B; 文本保存到指定地址时，将自动在文本结束处加上 0xFFFF 作为结束符； 录入的文本变量实际可能占用的最大变量空间=VP_Len_Max+1。
0x14	Scan_Mode	1	录入模式控制：0x00=重新录入，0x01=打开原来的文本再修改。
0x15	Lib_ID	1	显示所要使用的 ASCII 字库位置，0x00=默认字库。
0x16	Font_Hor	1	字体大小，X 方向的点阵数目
0x17	Font_Ver	1	字体大小，Y 方向的点阵数目（使用默认字库时，Y 方向点阵数必须为 X 方向点阵数的两倍）。
0x18	Cusor_Color	1	光标颜色，0x00=黑色，其他=白色
0x19	Color	2	文本显示颜色
0x1B	Scan_Area_Start	4	录入文本显示区域左上角坐标（Xs,Ys）
0x1F	Scan_Return_Mode	1	
0x20	0xFE	1	0xFE
0x21	Scan_Area_End	4	录入文本区域右下角坐标（Xe,Ye)
0x25	KB_Source	1	键盘页面位置选择：0x00=键盘在当前页面；其他=键盘不在当前页面。
0x26	PIC_KB	2	键盘所在页面（当键盘不在当前页面时有效）
0x28	Area_KB	8	键盘所在页面时的键盘区域坐标：（Xs,Ys）,（Xe,Ye） (键盘不在当前页时有效）
0x30	0xFE	1	0xFE
0x31	AREA_KB_Position	4	当键盘不在当前页面时，键盘显示位置的左上角坐标。
0x35	Display_EN	1	0x00=输入过程正常显示； 0x01=输入过程显示为"*"，用于密码输入。
0x36	NULL	10	写 0x00

与数字录入类似，ASCII 码文本录入同样可以简单地通过 PC 端 KGUS 开发软件来实现。打开 KGUS 软件，并点击触控配置菜单中的 ASCII 录入按钮。接下来用鼠标框选一个区域，就可在右侧的菜单中对该功能进行设置。**录入文本后通过“文本显示”功能将该数据显示出来。**

【注】KGUS 预装的 0#字库包含了 4*8---64*128 点阵的所有 ASCII 码字符。

（2）GBK 汉字文本录入

GBK 汉字录入的指令存储格式如表 6.7 所示。文本录入后需通过变量配置中的“文本显示”功能将该数据显示出来。

【注】

- **拼音“bd”对应所有 GBK 编码的全角标点符号录入**
- **注音输入法的键码（低字节）按照表 6.8 定义（注音输入法主要用于台湾地区）。**

表 6.7 GBK 汉字录入的指令存储格式

地址	定义	数据长度	说明
0x00	Pic_ID	2	页面 ID
0x02	TP_Area	8	触控按钮区域：（Xs,Ys）,(Xe,Ye)
0x0A	Pix_Next	2	目标切换页面，0xFF**表示不进行页面切换。
0x0C	Pic_On	2	按钮按压效果图所处的页面，0xFF**表示没有按钮按压效果。
0x0E	TP_Code	2	0xFE06（即文本录入的触控键码）。
0x10	0xFE	1	0xFE
0x11	*VP	2	变量地址指针
0x13	VP_Len_Max	1	文本变量最大长度，字（word）数目，0x01-0x7B; 文本保存到指针地址时，将自动在文本结束处加上 0xFFFF 作为结束符；
			录入的文本变量实际可能占用的最大变量空间为：VP_Len_Max+1。
0x14	Scan_Mode	1	录入模式控制：0x00=重新录入；0x01=打开已有文本再修改。
0x15	Lib_GBK1	1	汉字字符显示使用的 GBK 字库 ID，ASCII 字符默认使用 0x00 字库。
0x16	Lib_GBK2	1	录入过程中汉字字符显示所使用的 GBK 字库 ID。
0x17	Font_Scale1	1	Lib_GBK1 字体大小，点阵数目
0x18	Font_Scale2	1	Lib_GBK2 字体大小，点阵数目
0x19	Cusor_Color	1	光标颜色，0x00=黑色，其他=白色

0x1A	Color0	2	录入文本的显示颜色。
0x1C	Color1	2	录入过程中文本的显示颜色。
0x1E	PY_Disp_Mode	1	录入过程中，拼音提示和对应汉字的显示方式： * 0x00=拼音提示显示在上边，对应的汉字显示另起一行显示在下面； 拼音提示和汉字显示左对齐，行间距为 Scan_Dis。 * 0x01=拼音提示显示在左边，对应的汉字显示在右边； 汉字提示起始显示 x 位置为：Scan1_Area_Start+3 x Font_Scale2+Scan_Dis。
0x1F	Scan_Return_Mode	1	0xAA=在*（VP-1）位置保存输入结束标记和有效数据长度。 *（VP-1）高字节，输入结束标记：0x5A 表示输入结束，0x00 表示还在输入中。 *（VP-1）低字节，有效输入数据长度，字节单位。 0xFF=不返回输入结束标记和数据长度。
0x20	0xFE	1	0xFE
0x21	Scan0_Area_Start	4	录入文本显示区域左上角坐标（Xs,Ys）。
0x25	Scan0_Area_End	4	录入文本显示区域右下角坐标（Xe,Ye）。
0x29	Scan1_Area_Start	4	录入过程中拼音提示文本显示区域的左上角坐标。
0x2D	Scan_Dis	1	录入过程中，每个汉字显示的间距。每行最多显示 8 个汉字。
0x2E	0x00	1	0x00
0x2F	KB_Source	1	键盘页面位置选择：0x00=键盘在当前页面；其他=键盘不在当前页面。
0x30	0xFE	1	0xFE
0x31	PIC_KB	2	键盘所在页面 ID。（仅当键盘不在当前页面时有效）
0x33	Area_KB	8	键盘所在页面的键盘区域坐标：（Xs,Ys）；（Xe,Ye）。
0x3B	Area_KB_Position	4	键盘不在当前页面时，键盘在当前页面显示的左上角坐标。
0x3F	Scan_Mode	1	0x02=拼音输入法 0x03=注音输入法（台湾地区繁体录入）。

表 6.8 注音输入法的键码

注音	ㄅ	ㄆ	ㄇ	ㄈ	ㄉ	ㄊ	ㄋ	ㄌ	ㄍ	ㄎ	ㄏ	ㄐ
键码	0xc5	0xc6	0xc7	0xc8	0xc9	0xca	0xcb	0xcc	0xcd	0xce	0xcf	0xd0
注音	ㄑ	ㄒ	ㄓ	ㄔ	ㄕ	ㄖ	ㄗ	ㄘ	ㄙ	ㄧ	ㄨ	ㄩ
键码	0xd1	0xd2	0xd3	0xd4	0xd5	0xd6	0xd7	0xd8	0xd9	0xe7	0xe8	0xe9
注音	ㄚ	ㄛ	ㄜ	ㄝ	ㄞ	ㄟ	ㄠ	ㄡ	ㄢ	ㄣ	ㄤ	ㄥ
键码	0xda	0xdb	0xdc	0xdd	0xde	0xdf	0xe0	0xe1	0xe2	0xe3	0xe4	0xe5
注音	ㄦ	ㄧ	ˊ	ˇ	ˋ	˙						
键码	0xe6	0x99	0x40	0x98	0x41	0x42						

与数字录入类似，GBK 汉字文本录入同样可以简单地通过 PC 端 KGUS 开发软件来实现。打开 KGUS 软件，并点击触控配置菜单下的 GBK 录入按钮。接下来用鼠标框选一个区域，就可在右侧的菜单中对该功能进行设置。设置的方法与数字录入相似，在此不再复述。

6.3 弹出菜单选择

弹出菜单选择功能即点击触发一个弹出菜单并返回菜单项的键码。其指令储存格式如表 6.9 所示。

表6.9　弹出菜单选择功能指令储存格式

地址	定义	数据长度	说 明
0x00	Pic_ID	2	页面 ID
0x02	TP_Area	8	触控按钮区域：（Xs,Ys），（Xe,Ye）
0x0A	Pic_Next	2	目标切换页面，0xFF**表示不进行页面切换。
0x0C	Pic_On	2	按钮按压效果图所处的页面，0xFF**表示没有按钮按压效果。
0x0E	TP_Code	2	0xFE01(点击触发一个弹出菜单，返回菜单项的键码。)
0x10	0xFE	1	0xFE
ox11	*VP	2	变量地址指针，返回的数据有 VP_Mode 决定。
0x13	VP_Mode	1	0x00=把 0x00**键码写入 VP 字地址（整数型）； 0x01=把 0x**键码写入 VP 字地址的高字节地址（VP_H）； 0x02=把 0x**键码写入 VP 自地址的低字节地址（VP_L）； 0x10-0x1F：把**键码最低位（1bit）变量并写入 VP 字地址的指定位（0x10 修改 VP.0，0x1F 修改 VP.F）
0x14	Pic_Menu	2	弹出菜单的图片位置。
0x16	Area_Menu	8	弹出菜单区域：左上角坐标(Xs,Ys)，右下坐标(Xe,Ye)。
0x1E	Menu_Position_X	2	菜单在当前页面显示位置的左上角 X 坐标。
0x20	0xFE	1	0xFE
0x21	Menu_Position_Y	2	菜单在当前页面显示位置的左上角 Y 坐标。
0x23	NULL	13	写 0x00

【注】输入过程中有效键码：0x0000-0x00FF，其中 0x00FF 为取消（即不选择参数直接返回）。

弹出菜单控件需要配合动画图标控件和基本触控控件使用。

首先在触控配置中找到弹出菜单按钮，然后将需要点击按钮弹出菜单的区域划取出来，接着就是对弹出菜单控件属性框的操作。如下图 6.4 所示

图 6.4 弹出菜单属性配置

区域范围设置：设置触控按钮区域；

预览：查看触控按钮的效果；

名称定义：为按钮设置一个名称，在“变量查看”中方便查询；

数据自动上传：勾选后，数据将上传至串口；

按钮效果：按钮按压效果图所在页面（-1 时默认为无动画）。

变量地址：定义数据存储地址；

变量类型：0x00=键码写入 VP 字地址；0x01=键码写入 VP 地址的高字节(VP_H）

0x02=键码写入 VP 地址的低字节(VP_L)；0x10-0x1F=键码最低位(1bit)变量写入 VP 字地址的指定位（0x01 修改 VP.0，0x1F 修改 VP.F）；

弹出菜单设置：设置弹出菜单所在的页面；

所在页面：所要弹出菜单在页面的区域；

显示位置：弹出的菜单在当前页面要显示的位置。

【注】弹出的菜单上只能做基本触控和按键值返回。

我们结合实例来讲解具体操作如下：

变量地址：由于是和动画图标结合使用，所以两个地址必须一致（本例设置的都是 0666）；如图 6.8.1 和图 6.8.2 所示。

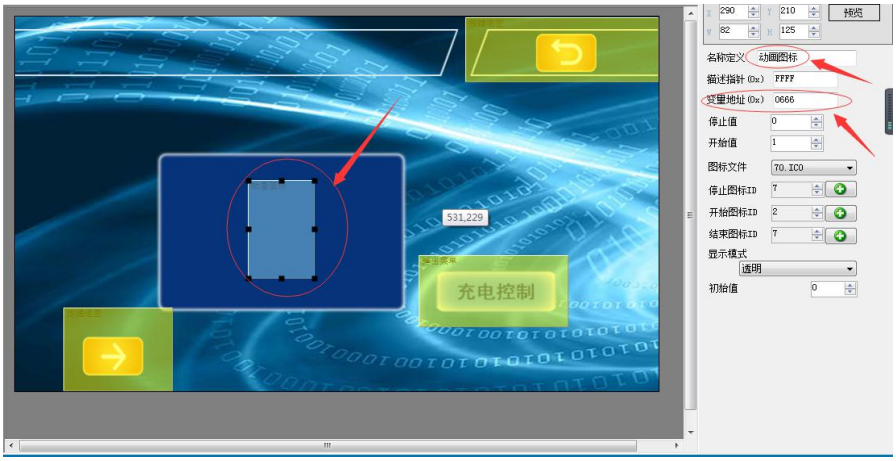


图 6.8.1

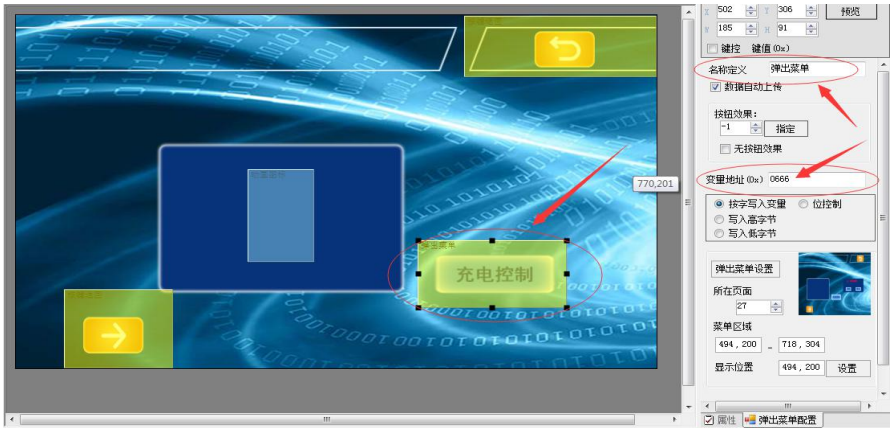


图 6.8.2

弹出菜单设置：选中带有弹出菜单所在页，然后在里面划取弹出菜单的区域；如下图 6.9 所示

图 6.9 弹出菜单区域的选取

所在页面：根据自己所选中的页数，自动生成；

菜单区域：根据自己划取的区域，自动生成；

显示位置：可与菜单区域的坐标相同；

配置完成后，切记要去弹出菜单所在页将菜单上的按钮分别通过基本触控控件进行赋值使能。

弹出菜单的有效输入键码是 0x0000~0x00ff，其中 0x00ff 表示为取消（即不选择参数直接返回），所以图标上的 X 号应赋值为 0x00ff（见图 6.10.1），“开始”按钮根据自己在动画显示控件中设置的开始值赋值，（本例设置的开始值是 1，所以此刻本例的开始按钮就赋值为 0001）（见图 6.10.2 和图 6.10.3），同样停止按钮根据自己在动画显示控件中设置的停止值赋值，（本例设置的停止值是 0，所以此刻本例的停止按钮就赋值为 0000）（见图 6.10.2 和 6.10.4）。

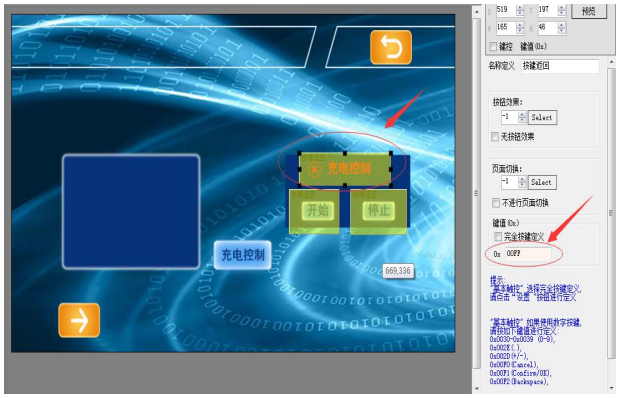


图 6.10.1

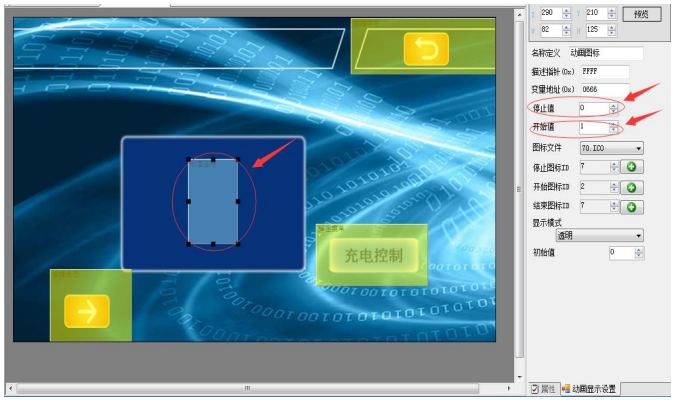


图 6.10.2

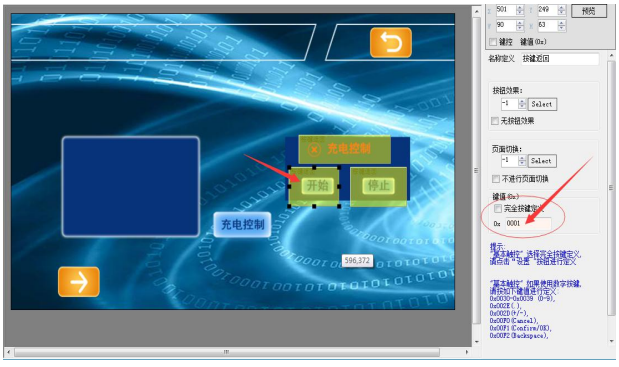


图 6.10.3

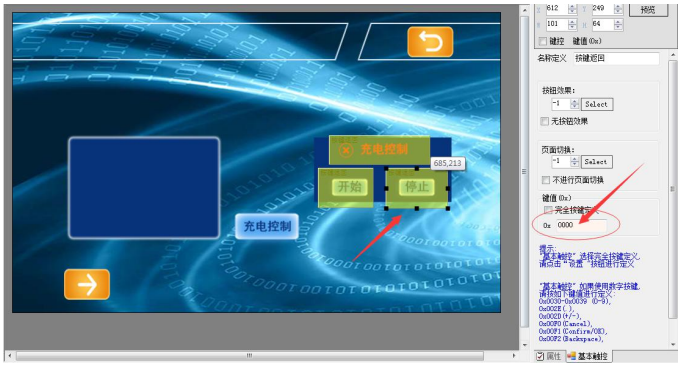


图 6.10.4

6.4 增量调节

增量调节功能的指令存储格式如表 6.10 所示。

表 6.10　增量调节功能的指令存储格式

地址	定义	数据长度	说　明
0x00	Pic_ID	2	页面 ID
0x02	TP_Area	8	触控按钮区域：（Xs,Ys），(Xe,Ye)
0x0A	Pic_Next	2	目标切换页面，必须为 0xFF**，表示不进行页面切换。
0x0C	Pic_On	2	按钮按压效果图所处的页面，0xFF**表示没有按钮按压效果。
0x0E	TP_Code	2	0xFE02
0x10	0xFE	1	0xFE
0x11	*VP	2	变量地址指针，返回数据由 VP_Mode 决定。

0x13	VP_Mode	1	0x00=调节 VP 字地址（整型数）； 0x01=调节 VP 字地址的高字节地址（1 字节无符号数，VP_H）； 0x02=调节 VP 字地址的低字节地址（1 字节无符号数，VP_L）； 0x10-0x1F：对 VP 字地址的指定位（0x10 对应 VP.0，0x1F 对应 VP.F)进行调节，调节范围必须设置为 0-1。
0x14	Adj_Mode	1	调节方式：0x00= -- 其它= ++
0x15	Return_Mode	1	逾限处理方式：0x00=停止（等于门限）其它=循环调节
0x16	Adj_Step	2	调节步长，0x0000-0x7FFF
0x18	V_Min	2	下限：2 字节整数（VP_Mode=0x01 或 0x02 时，仅低字节有效）
0x1A	V_Max	2	上限：2 字节整数（VP_Mode=0x01 或 0x02 时，仅低字节有效）
0x1C	Key_Mode	1	0x00：按住按键时连续调节； 0x01：按住按键时只调节 1 次。
0x1D	NULL	3	写 0x00

【注】调节后的变量可通过“数据变量、图标、艺术字变量”等将数据显示出来。

可用 PC 端 KGUS 开发软件实现增量调节功能。在 KGUS 软件中，点击触控配置中的增量调节 按钮，接下来用鼠标框选一个区域，并在右侧的菜单中对该功能进行设置，如调节步长和上下限及选择对应的调节方式（++/--）等。

增量调节和数据变量控件一起配合使用。

首先在触控配置中找到增量调节，然后选中并划取增量调节的区域，然后对右侧弹出的属性框进行操作，具体操作如下：

变量地址：需要和数据变量控件中的变量地址保持一致，本例为 0X0300；（见图 6.10.5 和图 6.10.6）

调节方式、逾限处理方式、调节步长、上限、下限等都可根据需求选择。

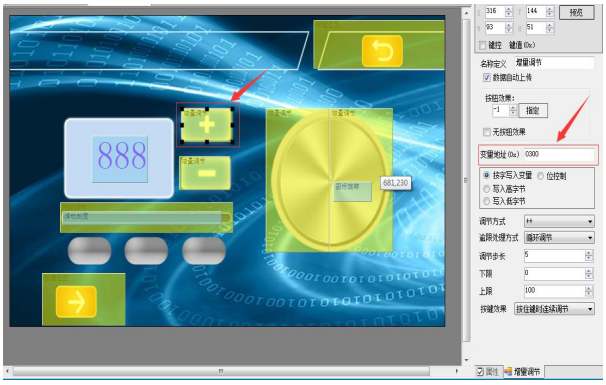


图 6.10.5

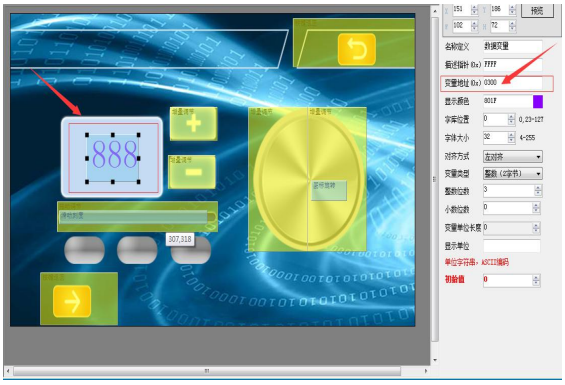


图 6.10.6

6.5 拖动调节

拖动调节功能的指令储存格式如表 6.11 所示。

表 6.11　拖动调节功能的指令储存格式

地址	定义	数据长度	说 明
0x00	Pic_ID	2	页面 ID
0x02	TP_Area	8	触控按钮区域：（Xs,Ys），（Xe,Ye）
0x0A	Pic_Next	2	目标切换页面，必须为 0xFF**，表示不进行页面切换。
0x0C	Pic_On	2	按钮按压效果图所处的页面，必须为 0xFF**，表示没有按钮按压效果。
0x0E	TP_Code	2	0xFE03
0x10	0xFE	1	0xFE
0x11	*VP	2	变量地址指针。

0x13	Adj_Mode	1	☆高 4bit 定义了数据返回格式： 0x0*=调节 VP 字地址（整型数）； 0x1*=调节 VP 字地址的高字节地址（1 字节无符号数，VP_H）； 0x2*=调节 VP 字地址的低字节地址（1 字节无符号数，VP_L）。 ☆低 4bit 定义了拖动方式： 0x*0=横向拖动；0x*1=纵向拖动。
0x14	Area_Adj	8	有效调节区域：（Xs,Ys）（Xe,Ye），必须和触控区域一致。
0x1C	V_Begain	2	起始位置对应的返回值，整数。
0x1E	V_End	2	终止位置对应的返回值，整数。

【注】拖动调节需要配合“滑动刻度指示”来实现，有关滑动刻度指示的相关内容请参考第 8 章。拖动调节不支持按键（即 0x4F 寄存器保存的键码）控制。调节滑块的数据可以通过“数据变量、变量图标”等功能显示出来。

可以用 PC 端 KGUS 开发软件实现拖动调节功能。在 KGUS 软件中，点击触控配置菜单下的拖动调节 按钮，接下来用鼠标框选一个区域，并在右侧的菜单中对该功能进行设置，如拖动方式以及起始终止位置对应的返回值等。

拖动调节控件需要和数据变量控件配合使用。拖动调节的优点是直观、快捷，而且参数不会越界。当需要更精确的拖动录入时，可以把拖动调节所指向的变量数据用“数据变量”功能来显示出来。

下面通过实例说明：

首先在触控配置中找到拖动调节控件，然后在需要使用拖动调节的区域划取一块区域，然后对弹出的属性框进项如下操作：

变量地址：需要和数据变量的地址保持一致 0300；（见图 6.10.6 和图 6.11.1）

数据返回格式：调节字地址；

拖动方式：根据自己需求选取（此处本例选取的是横向拖动）；

起始位置对应的返回值：可自行定义（本例定义的是 0）；

终止位置对应的返回值：可自行定义（本例定义的是 100）；

操作完成后的属性框大致如下图 6.12 所示

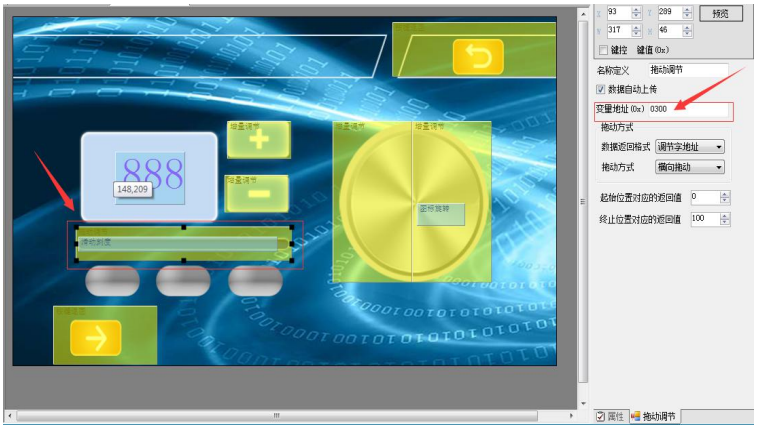


图 6.11.1

图 6.12 拖动调节属性框设置

6.6 RTC 设置

在工程项目中，时间显示与设置有非常广泛的应用。为了便于用户对时间进行直接的设置，加入了 RTC 设置功能，即 KGUS 屏通过触摸键盘直接设置 RTC，需要完整录入公历（年、月、日、时、分、秒）。该功能指令存储格式如表 6.12 所示。

表 6.12　RTC 设置功能存储格式

地址	定义	数据长度	说　明
0x00	Pic_ID	2	页面 ID
0x02	TP_Area	8	触控按钮区域：（Xs,Ys）（Xe,Ye）
0x0A	Pic_Next	2	目标切换页面，0xFF**表示不进行页面切换。
0x0C	Pic_On	2	按钮按压效果图所处的页面，0xFF**表示没有按钮按压效果。
0x0E	TP_Code	2	0xFE04（即进行 RTC 设置）
0x10	0xFE	1	0xFE
0x11	0x00 00 00	3	0x00 00 00
0x14	(x,y)	4	输入过程显示位置，右对齐方式，(x,y)是字符串右上角坐标。
0x18	Color	2	输入字体的显示颜色。
0x1A	Lib_ID	1	显示使用的 ASCII 字库位置，0x00=默认字库。
0x1B	Font_Hor	1	字体大小，X 方向点阵数目。
0x1C	Cusor_Color	1	光标颜色，0x00=黑色其他=白色
0x1D	KB_Source	1	0x00=键盘在当前页面其他=键盘不在当前页面
0x1E	PIC_KB	2	键盘所在页面 ID，仅当键盘不在当前页面时有效。
0x20	0xFE	1	0xFE
0x21	Area_KB	8	键盘区域：（Xs,Ys）（Xe,Ye）。仅当键盘不在当前页面时有效。
0x29	Area_KB_Position	4	键盘在当前页面显示位置的左上角坐标；仅当键盘不在当前页面时有效。
0x2D	NULL	3	写 0x00

【注】RTC 配置所用到的键盘制作方法与变量录入的相同，都按照 ASCII 码进行键值定义。时间的显示由“RTC 显示、时钟显示”功能来实现，有关显示的部分将在第 7 章进行讲解。

RTC 设置控件主要用于通过触摸屏修改时间，需要和 RTC 显示控件、基本触控控件配合使用。

在 KGUS 软件的触控配置中找到 RTC 设置按钮，接下来用鼠标框选一个区域，并在右侧的菜单中对该功能进行设置，

显示位置：可以找到一个键盘所在页键盘显示区域的右上角的坐标，手动输入即可；

如下图 6.13 所示

图 6.13 时间录入位置显示

显示颜色：根据自己喜好（此处选用默认颜色：黑色）；

字库位置：0 号字库；

字体大小：自主选择（此处本例选用的是 16 号）；

光标颜色：自主定义；

键盘设置：找到键盘所在页，将键盘所在区域划取即可；

如下图 6.14 所示

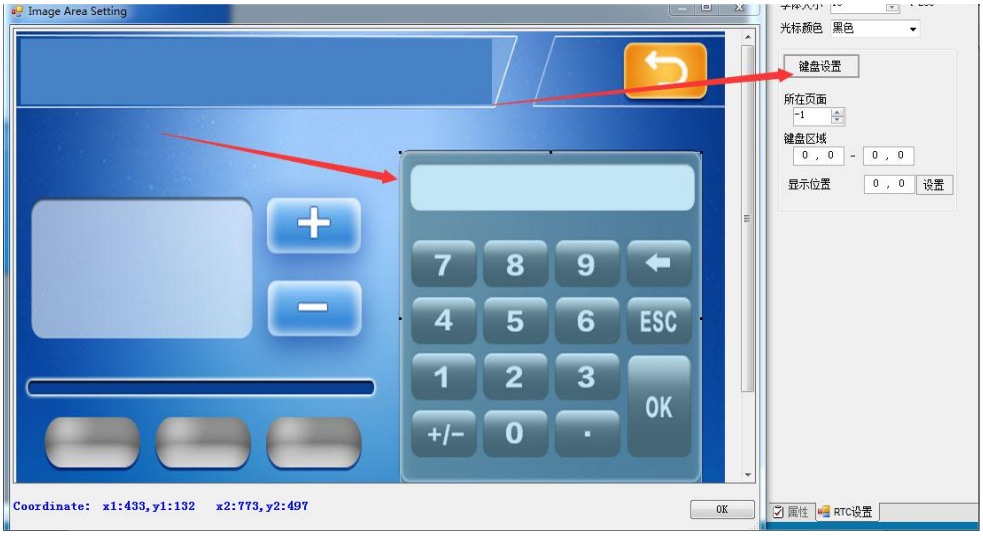


图 6.14 键盘显示区域划取

所在页面：根据自己在键盘设置中的选择自动生成；

键盘区域：根据自己在键盘设置中的选择自动生成；

显示位置：点击设置，在想要显示键盘输入的区域的左上角点击一下即可；

如下图 6.15 所示

图 6.15 键盘在操作页显示位置

操作完成后，切记要将键盘上的所有键通过基本触控控件进行使能。

程序配置完成后，当需要修改时间时，只需点击定义的按钮并在弹出的键盘中完整地输入公历（年、月、日、时、分、秒）即可方便的修改时间。

操作完成后的属性框大致如下图 6.16 所示

图 6.16 时间设置属性框设置

6.7 按键返回值

按键返回值功能即点击按钮，直接返回按键的值到指定的变量（支持位变量返回）。按键返回控件配合变量图标控件来使用。

表 6.13 按键的返回值指令存储格式

地址	定义	数据长度	说 明
0x00	Pic_ID	2	页面 ID
0x02	TP_Area	8	触控按钮区域：(Xs,Ys)（Xe,Ye）
0x0A	Pic_Next	2	目标切换页面，0xFF**表示不进行页面切换
0x0C	pic_On	2	按钮按压效果图所处的页面，0xFF**表示没有按钮按压效果。
0x0E	TP_Code	2	0xFE05
0x10	0xFE	1	0xFE
0x11	*VP	2	变量地址指针
0x13	TP_Mode	1	0x00=返回键值保存在 VP 字地址（整型数）; 0x01=返回键值低字节保存在 VP 字地址的高字节地址（VP_H); 0x02=返回键值低字节保存在 VP 字地址的低字节地址（VP_L); 0x10-0x1F：把返回键值的最低位（1bit）写入 VP 字地址的指定位（0x10 修改 VP.0，0x1F 修改 VP.F）。
0x14	Key_Code	2	返回键值。
0x16	NULL	10	写 0x00。

首先在触控配置中找到按键返回按钮，划取按钮所在区域，右侧弹出属性框，对属性框进行操作。具体操作如下：

变量地址：必须和在变量图标控件中的变量地址一致（先前本例在变量图表控件中定义的变量地址是 0020，所以此时本例的变量地址还是 0020）；

键值：根据自己在变量图标控件中定义的变量下限的值或者变量上限的值进行填写（本例的开始按钮是根据下限的值进行填写值为 0001；结束按钮是根据上限的值进行填写值为 0000）；

其中开始按钮的配置如下图 6.17 左侧所示，停止按钮的配置如下图 6.17 右侧所示

图 6.17 按键返回值属性框设置

6.8 参数配置

“参数配置”能够实现多个功能，包括寄存器与变量之间的数据传输、图像转为单色位图、发送数据到串口等。参数配置控件可以和拖动调节，滑块刻度，数据录入等控件配合使用。

其指令存储格式如表 6. 14 所示。

表 6.14　参数配置指令存储格式

地址	定义	数据长度	说 明
0x00	Pic_ID	2	页面 ID
0x02	TP_Area	8	触控按钮区域：（Xs,Ys）（Xe,Ye）
0x0A	Pic_Next	2	目标切换页面，0xFF**表示不进行页面切换。
0x0C	Pic_On	2	按钮按压效果图所处的页面，0xFF**表示没有按钮按压效果。
0x0E	TP_Code	2	0xFE07
0x10	0xFE	1	0xFE
0x11	Mode	1	操作模式选择，见"操作模式表"说明。
0x12	Data_Pack	14	操作模式数据包，见"操作模式表"说明。

参数配置的操作模式如表 6.15 所示：

表 6.15 参数配置的操作模式

<table>
<tr><th>Mode</th><th>Data_Pack</th><th>Data_Pack 说明</th><th>功 能</th></tr>
<tr><td>0x00</td><td>无</td><td>无</td><td>加载寄存器变量区数据到 0x6F00-0x6FFF 变量存储器空间（占据低字节）。</td></tr>
<tr><td>0x01</td><td>无</td><td>无</td><td>加载 0x6F00-0x6FFF 变量存储器空间（占据低字节）数据到寄存器变量区；同时改写对应的 R1-R3，R5-RA SD/SDHC 接口配置变量。</td></tr>
<tr><td rowspan="3">0x02</td><td>Tran_Area</td><td>将要转换的区域坐标：
(Xs,Ys)(Xe,Ye)</td><td rowspan="3">将指定区域的内容转换成单色位图（纵向取模打印位图格式），并保存到 VP 指针指向的数据存储器。
1、区域宽度（Xe-Xs+1）必须是偶数；
2、区域高度（Ye-Ys+1）必须是 8 的倍数；
3、*VP 指针保存数据格式如下：
*VP：状态位，处理完后设置成 0x5555；
*VP+1：横向字长度=(Xe-Xs+1)&0xFFFE/2；
*VP+2：数据段个数=(Ye-Ys+1)&0xFFF8/8；
*VP+3：位图数据开始，MSB 方式。
如果启用了参数自动上传功能（R2.3=1），那么转换完成后，会按照*VP 内容被修改成 0x5555 而自动上传一条提示信息。
本指令主要用于屏幕内容的打印输出。</td></tr>
<tr><td>*VP</td><td>保存转换位图数据的缓冲区首地址</td></tr>
<tr><td colspan="2"><table>
<tr><td></td><td>X=0</td><td>X=1</td><td>X=2</td><td>X=3</td><td>…</td><td>X=126</td><td>X=127</td></tr>
<tr><td>Y=0</td><td>D0.15</td><td>D0.7</td><td>D1.15</td><td>D1.7</td><td></td><td>D63.15</td><td>D63.7</td></tr>
<tr><td>…</td><td>…</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>Y=7</td><td>D0.8</td><td>D0.0</td><td>D1.8</td><td>D1.0</td><td></td><td>D63.8</td><td>D63.0</td></tr>
<tr><td>Y=8</td><td>D64.15</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>…</td><td>…</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>Y=15</td><td>D64.8</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table></td></tr>
<tr><td rowspan="2">0x03</td><td>*VP</td><td>数据指针</td><td rowspan="2">把*VP 指针位置开始的，T x_Len 字节长度的数据发送到用户串口。
Tx_Len 是一个字变量，长度从 0x0001-0xFFFF。</td></tr>
<tr><td>Tx_Len</td><td>需要发送的数据长度</td></tr>
<tr><td>0x04</td><td colspan="3">功能同 0x03，区别在于数据发送到 COM 2（系统保留串口）。</td></tr>
<tr><td rowspan="2">0x05</td><td>Tran_Area</td><td>将要转换的区域坐标：
(Xs,Ys)(Xe,Ye)</td><td rowspan="2">将指定区域的内容转换成单色位图（横向取模打印位图格式），并保存到 VP 指针指向的数据存储器。
1、区域宽度（Xe-Xs+1）必须是 16 的倍数；
2、*VP 指针保存数据格式如下：
*VP：状态位，处理完后设置成 0x5555；
*VP+1：横向字长度=(Xe-Xs+1）&0xFFF0/16；
*VP+2：数据段个数=(Ye-Ys+1)；
*VP+3：位图数据开始，MSB 方式。
如果启用了参数自动上传功能（R2.3=1），那么转换完成后，会按照*VP 内容被修改成 0x5555 而自动上传一条提示信息。
本指令主要用于屏幕内容的打印输出。</td></tr>
<tr><td>*VP</td><td>保存转换位图数据的缓冲区首地址</td></tr>
<tr><td rowspan="2">0x06</td><td>Frame_Head</td><td>帧头（2 字节）</td><td rowspan="2">把当前点击位置的触摸屏坐标发送到 COM 2（系统保留串口），格式如下：
Frame_Head+X+Y+Check(X,Y 的 1 字节累加和)+Frame_end.</td></tr>
<tr><td>Frame_End</td><td>帧尾（2 字节）</td></tr>
</table>

首先在触控配置中选取参数配置按钮，然后右侧会弹出属性框。

按钮效果：可根据自己需求选择是否需要；

页面切换：可根据自己需求选择是否需要；

Mode：KGUS 软件常用的为 00 和 01 两种模式；（00 模式表示加载寄存器变量区数据到 0x6F00~0X6FFF 变量存储区空间低字节处，01 模式表示加载 0x6F00~0x6FFF 低字节变量存储区空间数据到寄存器变量区；同时改写对应的 R1-R3，R5-RA SD/SDCH 接口配置变量），操作完成的属性框具体如图 6.18 所示：

图 6.18 参数配置

DataPack：在 00 和 01 模式下　不需要填写任何东西；

第七章 显示变量配置文件说明

第六章介绍了触控/键控配置文件的功能及配置方法。其中很多功能如拖动调节、转动调节等均需要配合显示变量来实现。本章将对显示变量配置文件（14.BIN）进行详细的介绍与讲解。

同触控/键控配置文件一样，显示变量配置文件也可以利用 PC 端的 KGUS 开发软件非常便捷的生成与设置。为了便于用户更深入地理解，本章将以指令的思维方式对显示变量配置文件进行更深入地讲解。

显示变量配置文件存储在字库空间中，它是由 N 条按照页面配置的变量指令组成，每条变量指令固定占用 32 字节存储空间。每个页面固定分配 2KB 或 4KB（0x0800 或 0x1000）变量存储空间，故每个页面最多可以设置 64 或 128 个变量。显示变量配置文件最大为 2MB，故可以配置最多 1024 个页面（128 变量模式下为 512 个页面）。对于相同类型的变量，存储位置越靠后，其显示优先级就越高。

一条触控指令由 6 部分组成，如表 7.1 所示。

表 7.1 一条触控指令 6 部分组成

序号	定义	数据字节长度	说明
1	0x5A	1	固定
2	Type	1	变量类型
3	*SP	2	变量描述文件从 Flash 加载后存储到数据存储区的地址指针，0xFFFF 表示不转存到数据存储区。
4	Len_Dsc	2	变量描述内容的字长度
5	*VP	2	变量地址，0x0000-0xFFFF，有些无需指定地址的变量写 0x0000 即可 当变量地址的高字节为 0xFF 时，本条指令将被取消
6	Description	N	变量描述内容

下面将为用户介绍显示变量的各种功能并对一些常用的功能分进行详细地讲解。

7.1 显示变量功能一览表

显示变量配置文件能够让 KGUS 屏实现多个功能，功能如表 7.2 所示。

表 7.2 显示变量配置文件功能

序号	功能代码	功能	说 明
01	00	变量图标显示	将一个数据变量的变化范围线性对应一组 ICON 图标显示；当变量变化时，图标也自动相应切换。多用于精细的仪表盘、进度条显示。
02	01	动画图标显示	将一个定值数据变量对应了 3 种不同的图标指示状态：不显示、显示固定图标、显示动画图标。多用于变量的报警提示。
03	02	滑块刻度指示	将一个数据变量的变化范围对应一个图标（滑块）的显示位置变化。多用于液位、刻度盘、进度表的指示。
04	03	艺术字变量显示	用 ICON 图标取代字库来显示变量数据。
05	04	图片动画显示	将一组全屏图片按照指定速度播放。多用于开机界面或屏保。
06	05	图标旋转指示	把一个数据变量的变化范围线性对应角度数据，然后把一个 ICON 图标按照对应的角度数据旋转后显示出来。多用于指针仪表板显示。
07	06	位变量图标显示	把一个数据变量的每个位（bit）的 0/1 状态对应 8 种不同显示方案中的两种，用 ICON 图标（或图标动画）来对应显示。
08	10	数据变量显示	把一个数据变量按照指定格式（整数、小数、是否带单位）用指定字体和大小的阿拉伯数字显示出来。
09	11	文本显示	把字符串按照指定的格式（选择字库决定）在指定的文本框显示区域显示。
10	12_00	文本格式 RTC 显示	按照用户编辑的格式把公历 RTC 用文本显示出来。
11	12_01	表盘格式 RTC 显示	采用 ICON 图标旋转，用指针表盘方式把公历 RTC 显示出来。
12	13	HEX 数据显示	把变量数据按照字节 HEX 方式间隔用户指定的 ASCII 字符显示出来。多用于计时显示，比如把 1234 显示为 12:34
13	14	文本滚屏显示	把变量数据在指定区域内按指定方向滚动显示。
14	20	实时曲线（趋势图）	结合 0x84 串口写曲线缓冲区数据来自动匹配显示实时曲线（趋势图）。可以指定显示区域，中心轴坐标、显示比例（放大/缩小）可控。
15	21_01	绘图_置点	置点(x,y,color)
16	21_02	绘图_端点连线	端点连线(color,(x0,y0),…(xn,yn))
17	21_03	绘图_矩形	显示矩形，颜色、位置和大小可控。
18	21_04	绘图_矩形填充	填充指定的矩形区域，填充颜色、位置和大小可控。
19	21_05	绘图_画圆	显示整圆弧，颜色、位置和大小可控。
20	21_06	绘图_图片剪切粘贴	从指定图片上剪切一个区域粘贴到当前显示页面上。
22	21_08	绘图_封闭区域填充	封闭区域填充，种子点坐标、填充颜色可控。
23	21_09	绘图_频谱显示	根据变量数据显示频谱（垂直线条），线条颜色、位置可控。
24	21_0A	绘图_线段显示	根据变量数据连接线段，端点、颜色可控。
25	21_0B	绘图_圆弧显示	显示圆弧，半径、颜色和起止角度可控。
26	21_0C	绘图_字符显示	根据变量数据进行单个字符显示。
27	21_0D	绘图_矩形域 XOR	对指定的矩形域位图数据用指定颜色进行 XOR 操作，多用于高亮显示。
28	21_0E	绘图_双色位图显示	变量存储器看成双色位图数据，0/1 对应颜色可以指定，多用于自定义光标。
29	21_0F	绘图_位图显示	变量存储器数据位为 5K 色位图数据，多用于实时图标（照片）下载显示。
30	21_10	绘图_区域放大粘贴	把指定区域放大 1 倍粘贴到指定位置，多用于配合 21_0F 指令实现照片的实时显示。
31	22	列表显示	把按照二维数组定义的数据用表格分栏显示出来。
32	25	二维码显示	根据指定内容在屏幕显示指定的二维码图形。

7.2 图标库文件的制作

欲使用图标相关的功能，首先要将图标库文件加载到 PC 端 KGUS 开发软件中。方法如下：
首先打开 KGUS 软件下拉菜单中的 ICO 文件生成器，如图 7.1 所示。

图 7.1 ICO 文件生成工具

在 ICO 文件生成器中选择想要转换成 ICO 图标的图片文件（切记生成的图标库名字必须在 23~127 之间），并点击生成新 ICO 文件即可生成 ICO 图标库文件（例如 70.ICO），如果之前就有想要往里面添加新的图标就需要点击附加到 ICO 文件按钮；如图 7.2 所示

图 7.2 ICO 文件生成步骤

注意：ICO 文件生成后需要将其拷贝到工程目录下的 YK_TFT 文件夹中，并下载到 KGUS 屏里，这样才可以调用。

7.2.1 变量图标显示

变量图标显示功能是将一个数据变量的变化范围线性对应一组 ICON 图标显示；当变量变化时，图标也自动相应切换。多用于精细的仪表盘、进度条显示。

其指令存储格式如表 7.3 所示。

表 7.3　变量图标显示指令存储格式

地址		定义	数据长度	说　明
0x00		0x5A00	2	固定值，其中后两位 00 为功能代码，详参第一节中的一览表
0x02		*SP	2	变量指针描述，0xFFFF 表示由配置文件加载
0x04		0x0008	2	固定值 0x0008
0x06	0x00	*VP	2	变量指针，变量为整数格式
0x08	0x01	(x,y)	4	变量显示位置，图标左上角坐标位置
0x0C	0x03	V_Min	2	变量下限，越界不显示
0x0E	0x04	V_Max	2	变量上限，越界不显示
0x10	0x05	Icon_Min	2	V_Min 对应的图标 ID
0x12	0x06	Icon_Max	2	V_Max 对应的图标 ID
0x14	0x07_H	Icon_Lib	1	图标库存储位置
0x15	0x07_L	Mode	1	ICON 显示模式，0x00=透明其他=显示图标背景

【注】两个地址存储格式分别对应了 64/128 变量模式。下面的指令与此相同，不再赘述。

变量图标控件通常配合数字录入，增量调节和按键值返回等控件共同使用。使用到的图标的图标库 ICO 文件需要提前做好。可以简单地通过 PC 端 KGUS 开发软件来实现。打开 KGUS 软件，并点击变量配置中的变量图标按钮。接着用鼠标框选一个区域，就可在右侧的菜单中对该功能进行设置。如图 7.3 所示。

图 7.3 变量图标功能设置菜单

区域范围设置： (X,Y)为 ICON 图标左上角显示在当前页位置坐标。

名称定义：为按钮设置一个名称，在“变量查看”中方便查询；

描述指针：变量指针描述，0xFFFF 表示由配置文件加载

变量地址：定义变量数据的存储地址；

图标文件：要调用的图标库文件（**需要先将图标库文件存入 KGUS 屏中才能正确读取，参考本章开始的图标库文件添加方法**）

变量上限、变量下限：规定变量显示的范围，越界不显示。

对应的图标：变量上下限对应的图标；

显示模式：透明显示（滤除背景色显示）/显示背景。

通过实例说明一下。

描述指针：FFFF 表示由配置文件加载；

变量地址：可以自行定义（此处本例定义为 0500）；

图标文件：找到自己制定的 ICO 文件（本例的 ICO 文件名为 70.ICO）；

变量下限：可以根据具体操作来自行定义（本例制作的是红绿灯操作，所以本例定义为 0）；

对应的图标：选取变量为变量下限的值所对应的图标（本例定义为红灯图片）；

变量上限：可以根据具体操作来自行定义（本例制作的是红绿灯操作，所以本例定义为 1）；

对应的图标：选取变量为变量上限的值所对应的图标（本例定义为绿灯图片）；

显示模式：透明；

操作完成后属性框大致如下图 7.4 所示

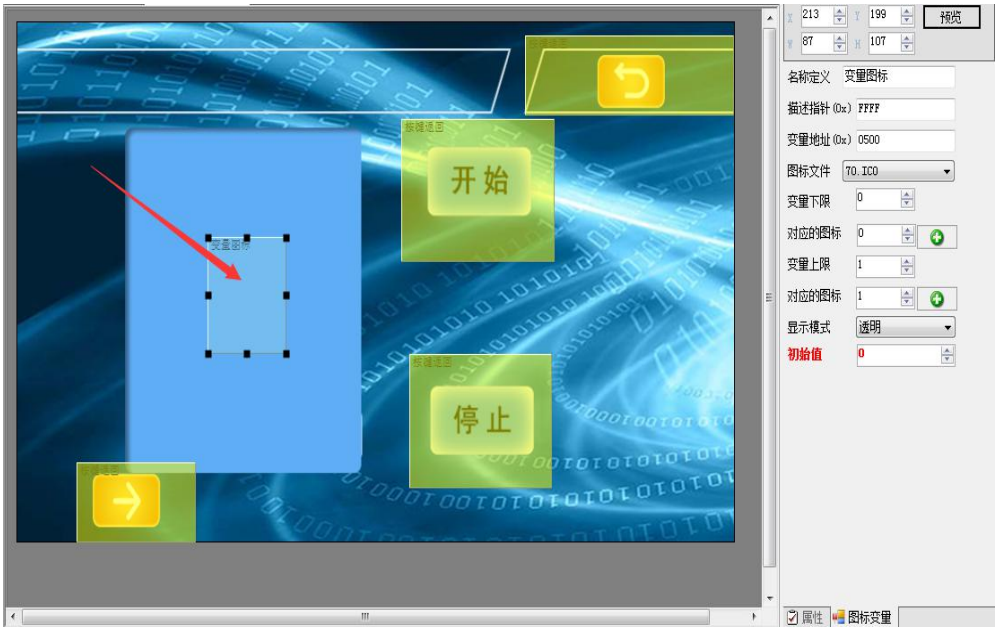


图 7.4 变量图标控件的设置

7.2.2 动画图标显示

动画图标显示功能是将一个定值数据变量对应了 3 种不同的图标指示状态：不显示、显示固定图标、显示动画图标。多用于变量的报警提示。其指令存储格式如表 7.4 所示。

表 7.4　动画图标显示指令存储格式

地址		定义	数据长度	说 明
0x00		0x5A01	2	固定值 0x5A01
0x02		*SP	2	变量描述指针，0*FFFF 表示由配置文件加载。
0x04		0x000A	2	固定值 0x000A
0x06	0x00	*VP	2	初始图标变量指针， 变量为双字， 低位字保留，高位字无符号数（0x0000-0xFFFF)用户数据控制动画图标显示。
0x08	0x01	（x,y）	4	变量显示位置，图标左上角坐标位置。

0x0C	0x03	Reset_ICON_En	2	0x0000:停止时，不复位动画图标起始值（动画图标显示从 ICON_Start 到 ICON_End 间的一个任意值开始显示）。 0x0001:停止时，复位动画图标起始值（动画图标显示将固定从 ICON_Start 开始显示）。
0x0E	0x04	V_Stop	2	变量为该值时显示固定图标。
0x10	0x05	V_Start	2	变量为该值时自动显示动画图标。
0x12	0x06	ICON_Stop	2	变量为 V_Stop 时固定显示该图标。
0x14	0x07	ICON_Start	2	变量为 V_Start 值时，自动从 ICON_Start 到 ICON_End 显示图标，形成动画效果。
0x16	0x08	ICON_End	2	
0x18	0x09:H	ICON_Lib	1	图标库存储位置
0x19	0x09:L	Mode	1	ICON 显示模式，0x00=透明

首先在变量配置中找到动画图标控件，然后将需要呈现动画图标的区域划取出来。

接下来就是对属性框的操作，具体操作如下：

描述指针：FFFF 表示由配置文件加载；

变量地址：可自主填写（此处本例选择的是 0666）；

停止值：可自己设定（本例设定的是 0）；

开始值：可自己设定（本例设定的是 1）；

图标文件：这个需要提前设定，具体操作可查看滑动刻度控件中 ICO 生成方式，此处命名为 70.ICO；

停止图标 ID：可自主定义，要勾选是否过滤背景色（本例定义的是满电量图标 7 号图标）；

开始图标 ID：低电量的图标，要勾选是否过滤背景色（本例的低电量图标是 0 号图标）；

结束图标 ID：满电量的图标，要勾选是否过滤背景色（本例的满电量图标是 7 号图标）；

具体如下图 7.4.1，图 7.4.2 所示

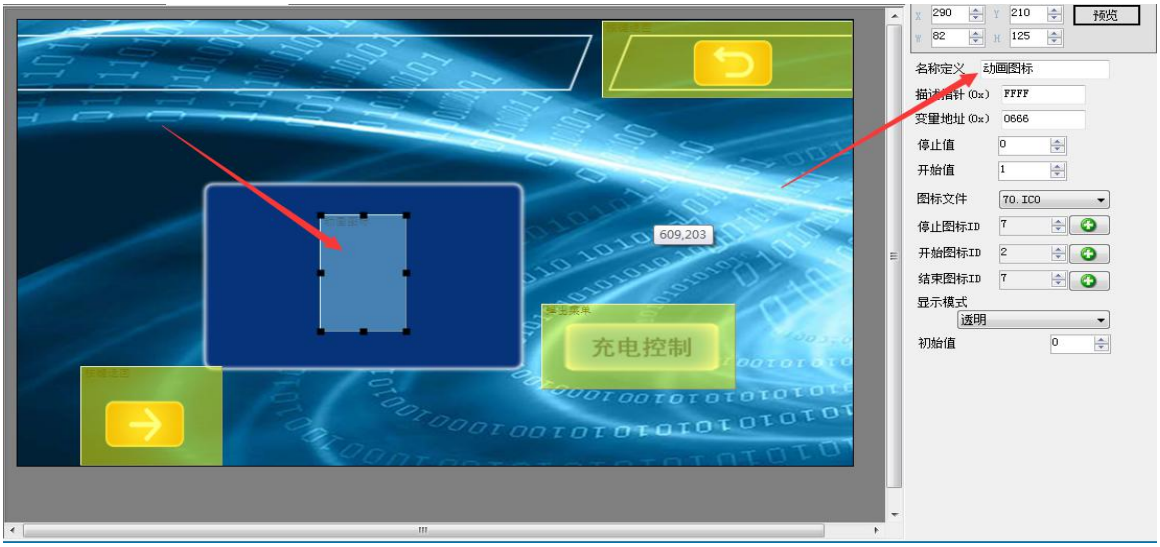


图 7.4.1 动画图标控件的设置

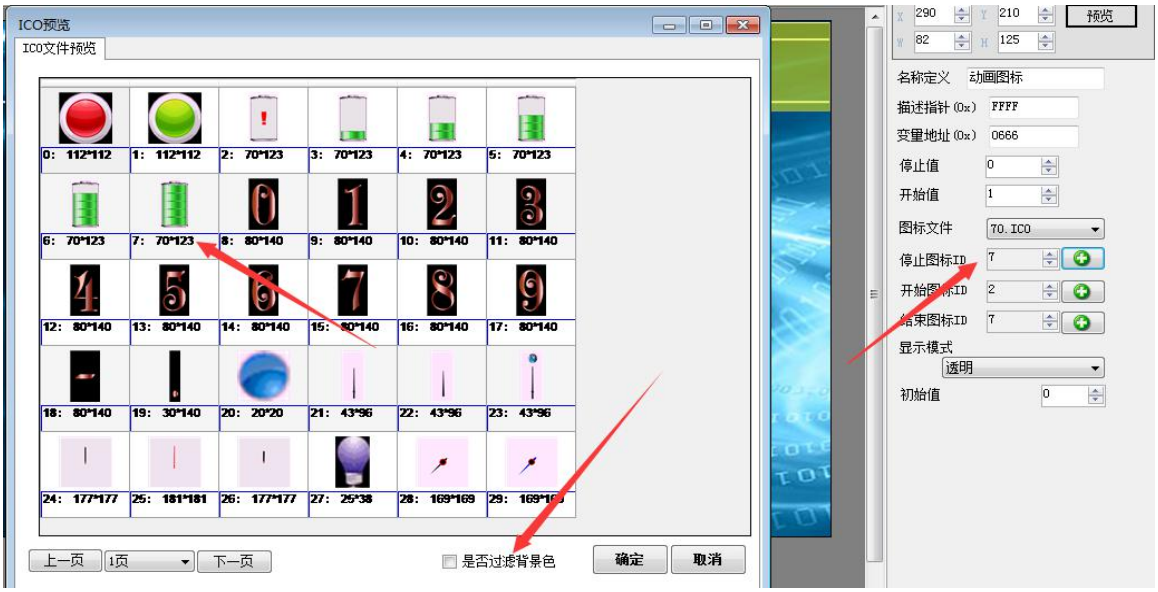


图 7.4 .2 动画图标控件的设置

7.2.3 滑块刻度指示

在第 6 章 6.5 节中介绍了拖动调节，本节将介绍配合其使用的滑块刻度指示。滑块刻度指示功能是将一个数据变量的变化范围对应一个图标（滑块）的显示位置变化。多用于液位、刻度盘、进度表的指示。滑块刻度指示是显示功能，拖动调节是控制功能，因而两者配合在一起能够实现拖动滑块图标改变变量数值的功能。此外，该功能也可以单独用于进度条的显示，此时无需拖动调节功能的配合。

其指令存储格式如表 7.5 所示。

表7.5 滑块刻度指示指令存储格式

地址		定义	数据长度	说明
0x00		0x5A02	2	固定值 0x5A02
0x02		*SP	2	变量描述指针，0xFFFF 表示由配置文件加载
0x04		0x000A	2	固定值 0x000A
0x06	0x00	*VP	2	变量指针，变量格式由 VP_Data_Mode 决定。
0x08	0x01	V_Begain	2	对应起始刻度的变量值
0x0A	0x02	V_End	2	对应终止刻度的变量值。
0x0C	0x03	X_Begain	2	起始刻度坐标（纵向为 Y 坐标）
0x0E	0x04	X_End	2	终止刻度坐标（纵向为 Y 坐标）
0x10	0x05	ICON_ID	2	刻度滑动块的图标 ID
0x12	0x06	Y	2	刻度指示图标显示的 Y 坐标位置（纵向为 X 坐标）
0x14	0x07:H	X_adj	1	刻度指示图标显示的 X 坐标前移偏移量（纵向为 Y 坐标），0x00-0xFF
0x15	0x07:L	Mode	1	刻度模式：0x00=横向刻度条 0x01=纵向刻度条
0x16	0x08:H	ICON_Lib	1	图标库存储位置
0x17	0x08:L	ICON_Mode	1	ICON 显示模式，0x00=透明（不显示背景），其它=显示图标背景
0x18	0x09:H	VP_Data_Mode	1	0x00：*VP 指向一个整型变量 0x01：*VP 指向一个整型变量的高字节地址 0x02：*VP 指向一个整型变量的低字节地址

注：配合拖动调节使用时，拖动调节框选范围要与滑块刻度指示的范围一致，这样才能实现滑块随手指拖动的效果。

首先在变量配置中找到滑动刻度按钮，点击。然后在需要显示滑动刻度的区域划取。然后对右侧弹出的属性框进行如下操作：

描述指针：FFFF 表示由配置文件加载；

变量地址：和增量调节等控件的变量地址保持一致 6F01；

图标文件：如果之前已经完成 ICO 文件的生成和拷贝工作，此刻直接在右侧的图标文件中选取需要使用到的图标库文件即可；

滑动图标：点击  按钮在选中的图标库文件中找到自己想要使用的图标，勾选过滤背景颜色，添加进去即可；

显示模式：透明模式；

（x）坐标前移偏移量：自主定义，不可为 0（0 没有显示效果）；

变量类型：指向一个整形变量；

初始值：根据需求自主定义；

操作完成后的属性框大致如下图 7.5 所示

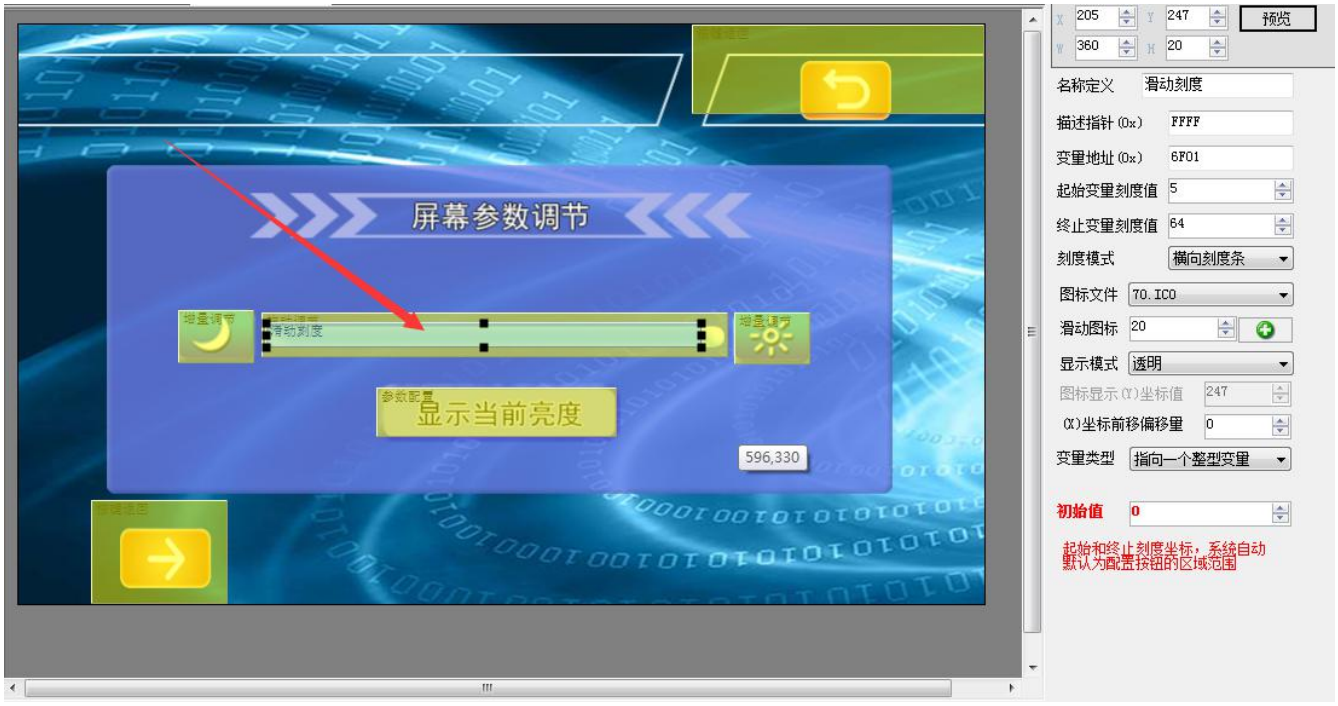


图 7.5 滑动刻度控件的设置

7.2.4 艺术字变量显示

艺术字变量显示是用 ICON 图标取代字库来显示变量数据。需要配合基本控件以及数据录入等控件一起使用，和变量图标显示功能用法类似。其指令存储格式如表 7.6 所示。

表 7.6　艺术字变量显示指令存储格式

地址	定义	数据长度	说明
0x00	0x5A03	2	固定值：0x5A03

0x02		*SP	2	变量描述指针，0xFFFF 表示由配置文件加载
0x04		0x0007	2	固定值：0x0007
0x06	0x00	*VP	2	变量指针
0x08	0x01	（X，Y）	4	起始显示位置： 左对齐模式，该坐标为显示字符串的左上角坐标； 右对齐模式，该坐标为显示字符串的右上角坐标。
0x0C	0x03	ICON0	2	0 对应的 ICON_ID，排列顺序为 0123456789…。
0x0E	0x04:H	ICON_Lib	1	ICON 库位置
0x0F	0x04:L	ICON_Mode	1	ICON 显示模式，0x00=透明其他=显示背景
0x10	0x05:H	整数位数	1	显示的整数位数
0x11	0x05:L	小数位数	1	显示的小数位数
0x12	0x06:H	变量数据类型	1	0x00=整数（2 字节），范围-32768 到 32767 0x01=长整数（4 字节），范围-2147483648 到 2147483647 0x02=*VP 高字节，无符号数，范围 0 到 255 0x03=*VP 低字节，无符号数，范围 0 到 255 0x04= 超长整数（ 8 字 节 ）， 范 围 -9223372036854775808 到 9223372036854775807 0x05=无符号整数（2 字节），范围 0 到 65535 0x06=无符号长整数（4 字节），范围 0 到 4294967295
0x13	0x06:L	对齐模式	1	0x00=左对齐 0x01=右对齐

首先在变量配置菜单中找到艺术字变量控件，在需要显示艺术字的区域划取。然后对右侧弹出的属性框进行如下操作：

描述指针：FFFF 表示由配置文件加载；

变量地址：可自行定义（本例为 0060）；

图标文件：根据之前的图标库文件制作方法制作（本例为 70.ICO），切记必须要将制作好的图标库文件放到 YK_SET 文件夹中才可以正常使用；

起始图标：自主定义；

显示模式：透明；

变量类型：整数；

整数位数：自主定义；

小数位数：根据需求自主定义一般为 0；

对齐方式：自主定义；

初始值：自主定义；

操作完成后的属性框大致如下图 7.6 所示

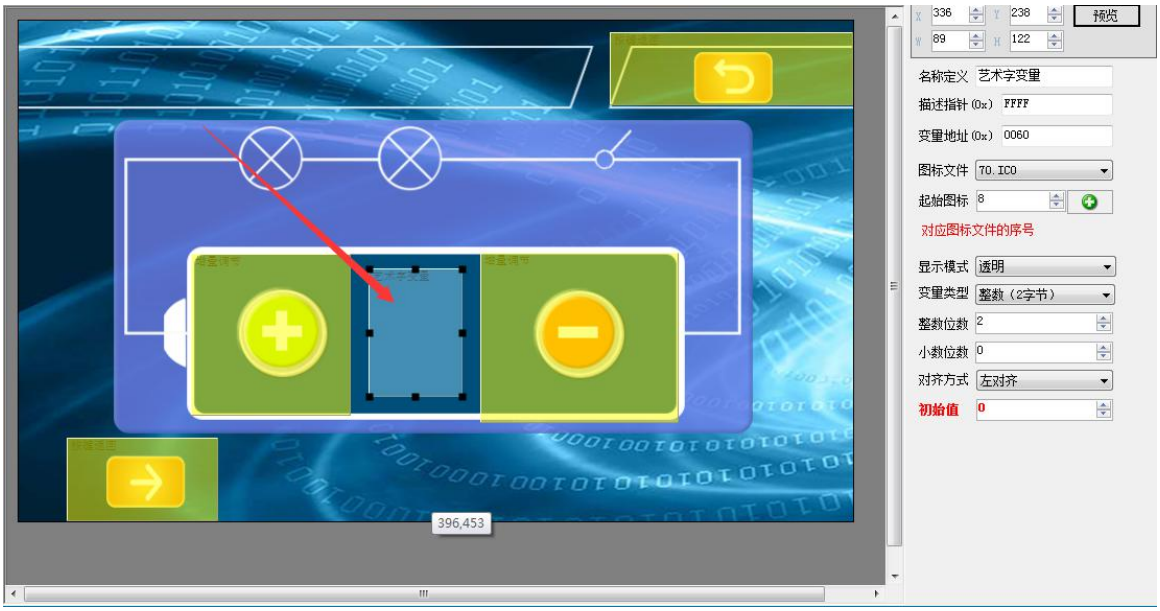


图 7.6 艺术字变量控件的属性框设置

7.2.5 图片动画显示

图片动画显示功能是将一组全屏图片按照指定速度播放，该功能多用于开机界面或屏保。其指令存储格式如表 7.7 所示。

表 7.7　图片动画显示指令存储格式

地址		定义	数据长度	说明
0x00		0x5A04	2	固定值 0x5A04
0x02		*SP	2	变量指针描述，0xFFFF 表示由配置文件加载。
0x04		0x0004	2	固定值 0x0004
0x06	0x00	0x0000	2	固定值 0x0000
0x08	0x01	Pic_Begain	2	起始图标位置
0x0A	0x02	Pic_End	2	终止图标位置
0x0C	0x03:H	Frame_Time	1	一帧（一幅图片）显示的时间，单位为 8ms。

【注】起始图片位置必须小于终止图片位置。如果在终止页面中也设置了图片动画变量，用户可实现重播。

首先在变量配置菜单中找到图片动画控件，然后在以 000 命名的图片中随意划取一块区域，该功能无需按钮触发，因此区域可任意框选，只要保证在指定页面即可。然后操作右面弹出的属性框

描述指针：FFFF 表示由配置文件加载；

起始图片位置：第一张图片动画的位置；

终止图片位置：最后一张图片动画的位置；

显示时间设置：根据自己需求自己定义，具体显示时间是设置的时间*8 ms（设置的范围 0~256）；

操作完成后大致如下图 7.7 所示

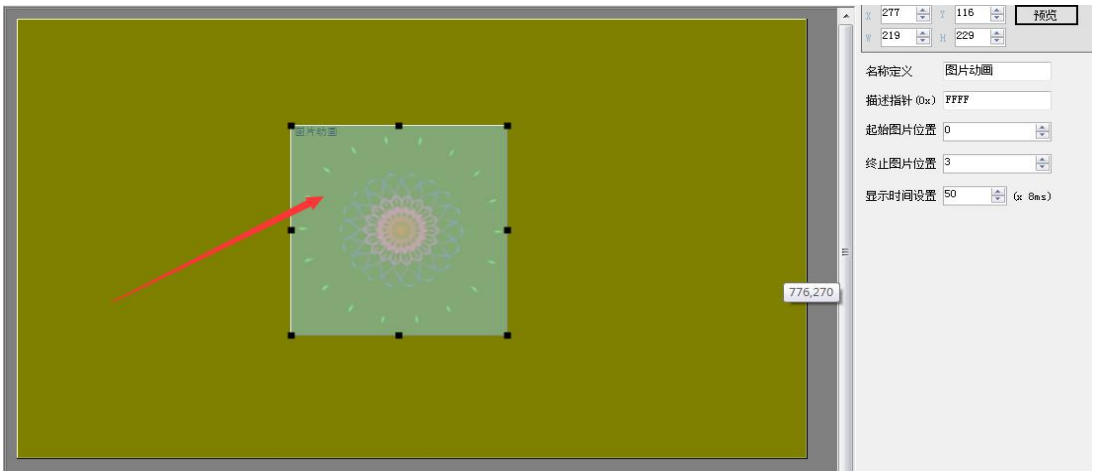


图 7.7 图片动画控件属性框的设置

7.2.6 图标旋转

图标旋转指示功能是把一个数据变量的变化范围线性对应角度数据，然后把一个 ICON 图标按照对应的角度数据旋转后显示出来。该功能多用于指针仪表板显示。其指令存储格式如表 7.8 所示。

表 7.8　图标旋转指示指令存储格式

地址		定义	数据长度	说明
0x00		0x5A05	2	固定值 0x5A05
0x02		*SP	2	变量描述指针，0xFFFF 表示由配置文件加载
0x04		0x000C	2	固定值 0x000C
0x06	0x00	*VP	2	变量指针，变量模式由 VP_Mode 决定。
0x08	0x01	ICON_ID	2	指定的图标 ID。
0x0A	0x02	ICON_Xc	2	ICON 图标上的旋转中心位置：X 坐标
0x0C	0x03	ICON_Yc	2	ICON 图标上的旋转中心位置：Y 坐标
0x0E	0x04	Xc	2	ICON 显示到当前屏幕的旋转中心位置：X 坐标
0x10	0x05	Yc	2	ICON 显示到当前屏幕的旋转中心位置：Y 坐标
0x12	0x06	V_Begain	2	对应起始旋转角度的变量值，整型数，越界不显示
0x14	0x07	V_End	2	对应终止旋转角度的变量值，整型数，越界不显示
0x16	0x08	AL_Begain	2	起始旋转角度，0-720（0x000-0x2D0），单位 0.5°。
0x18	0x09	AL_End	2	终止旋转角度，0-720（0x000-0x2D0），单位 0.5°。
0x1A	0x0A:H	VP_Mode	1	0x00：*VP 指向一个整型变量 0x01：*VP 指向一个整型变量的高字节数据 0x02：*VP 指向一个整型变量的低字节数据
0x1B	0x0A:L	Lib_ID	1	ICON 图标库 ID。

0x1C	0x0B	Mode	1	ICON 显示模式，0x00=透明（不显示图表背景）其它=显示图标背景

【注】旋转始终假定为"顺时针"，即AL_End 必须大于AL_Begin，若AL_End 小于AL_Begin，系统会自动加上360°

在KGUS开发软件中，可以在变量配置菜单中选择图标旋转，之后框选区域并完成该功能的配置即可。

该功能的角度定义如图 7.8 所示。

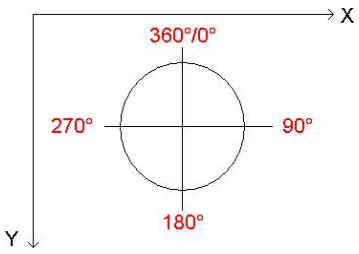


图 7.8　图标旋转功能的角度定义

图标旋转控件要和数据录入、增量调节、滑动刻度、拖动调节等控件配合使用。

需要准备的就是制作好需要用到的图标库文件，并把该文件放入工程目录下的 YK_SET 文件夹中。准备好的图片如下图 7.9 所示

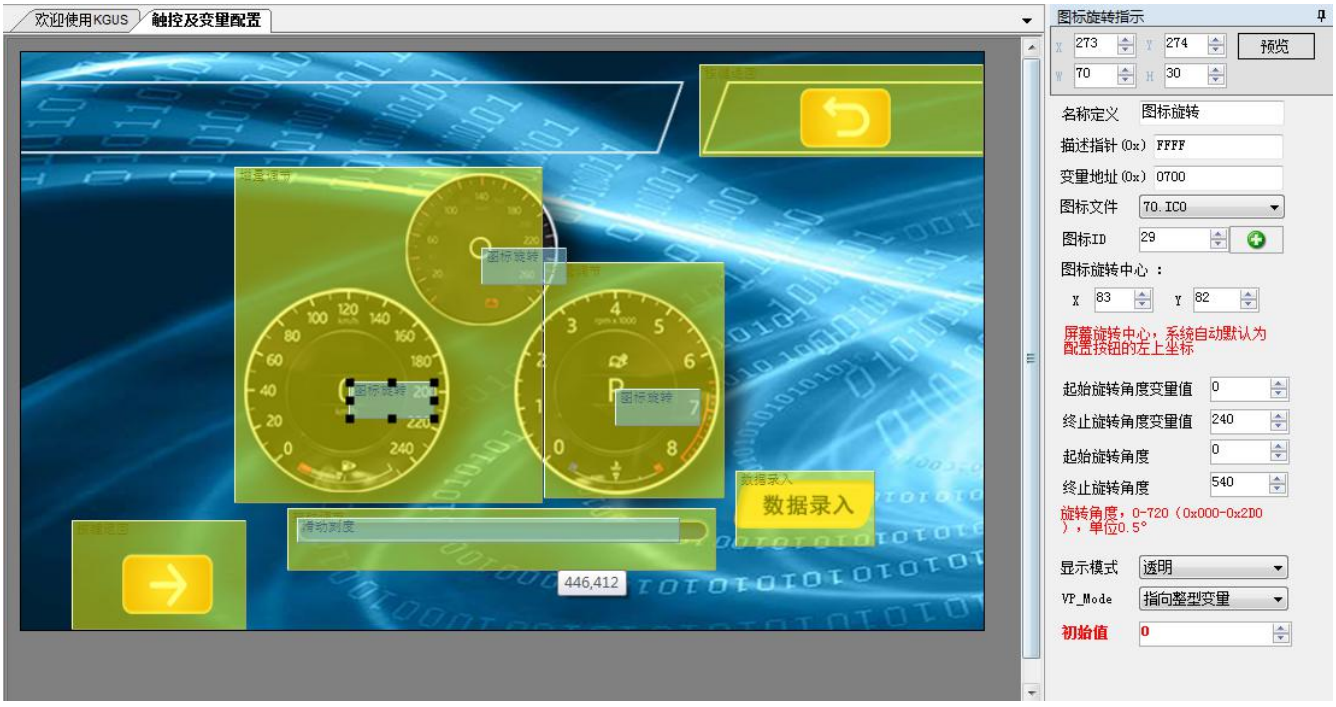


图 7.9 转动调节图片及控件选取

描述指针：采用默认的（0xFFFF）；

变量地址：自己定义（本例是 0700）；

图标文件：制作好的图标库文件（本例是 70.ICO）；

图标 ID：选取自己希望旋转的图标，在选取的时候注意要勾选过滤背景色，然后对弹出的图标设置旋转中心，就是在希望成为旋转中心的地方点击；

图标旋转中心：根据自己之前的点击，自动生成；

起始旋转角度变量值：指针旋转的起始值（如上图所示的话就是 0 作为起始值）；

终止旋转角度变量值：指针旋转的终止值（如上图所示的话就是 240 作为终止值）；

起始旋转角度：在制作图标的时候，选取图标的指示方向指向指针旋转的起始值，这样此处就可以填写 0；

终止旋转角度：根据起始旋转角度计算出对应的终止旋转角度，此处的旋转角度为 0~720，单位为 0.5°，可以根据旋转的角度范围占圆形的百分比然后乘以 720 得到（如上图，本例要让指针旋转的变量值为 0~240，也就是占圆形的 3/4，就用 3/4 x 720 得到 的结果 540 就是终止旋转角度）；

显示模式：选用默认；

VP_Mode：选用默认；

拖动调节和滑块刻度等控件的操作参见相应的说明，只需修改其中的变量刻度值即可。

7.2.7 位变量图标显示

位变量图标显示功能是把一个数据变量的每个位（bit）的 0/1 状态对应 8 种不同显示方案中的两种，用 ICON 图标（或图标动画）来对应显示。其指令存储格式如表 7.9 所示。

表 7.9　位变量图标显示指令存储格式

<table>
<tr><th colspan="2">地址</th><th>定义</th><th>数据长度</th><th>说明</th></tr>
<tr><td colspan="2">0x00</td><td>0x5A06</td><td>2</td><td>固定值0x5A06</td></tr>
<tr><td colspan="2">0x02</td><td>*SP</td><td>2</td><td>变量描述指针，0xFFFF 表示由配置文件加载</td></tr>
<tr><td colspan="2">0x04</td><td>0x000C</td><td>2</td><td>固定值0x000C</td></tr>
<tr><td colspan="2">0x06</td><td>*VP</td><td>2</td><td>位变量指针，字变量</td></tr>
<tr><td colspan="2">0x08</td><td>*VP_AUX</td><td>2</td><td>辅助变量指针，双字，用户软件不能访问</td></tr>
<tr><td colspan="2">0x0A</td><td>Act_Bit_Set</td><td>2</td><td>值为1 的bit 位置说明*VP 对应位置需要显示。</td></tr>
<tr><td colspan="2">0x0C</td><td>Display_Mode</td><td>1</td><td>定义显示模式：
<table>
<tr><th rowspan="2">Display_ Mode</th><th colspan="2">位变量(bit)值</th></tr>
<tr><th>0</th><th>1</th></tr>
<tr><td>0x00</td><td>ICON0S</td><td>ICON1S</td></tr>
<tr><td>0x01</td><td>ICON0S</td><td>不显示</td></tr>
<tr><td>0x02</td><td>ICON0S</td><td>ICON1S-ICON1E 动画</td></tr>
<tr><td>0x03</td><td>不显示</td><td>ICON1S</td></tr>
<tr><td>0x04</td><td>不显示</td><td>ICON1S-ICON1E 动画</td></tr>
<tr><td>0x05</td><td>ICON0S-ICON0E 动画</td><td>ICON1S</td></tr>
<tr><td>0x06</td><td>ICON0S-ICON0E 动画</td><td>不显示</td></tr>
<tr><td>0x07</td><td>ICON0S-ICON0E 动画</td><td>ICON1S-ICON1E 动画</td></tr>
</table>
比如设置 Display_Mode=2，那么：
*VP 对应的变量某个位为 0 时，显示 ICONS 图标</td></tr>
<tr><td colspan="2">0x0D</td><td>Move_Mode</td><td>1</td><td>位图图标排列方式：
0x00=X++，Act_Bit_Set 指定的不显示 bit 不保留位；
0x01=Y++，Act_Bit_Set 指定的不显示 bit 不保留位置
0x02=X++，Act_Bit_Set 指定的不显示 bit 保留DIS_MOV 位置
0x03=Y++，Act_Bit_Set 指定的不显示 bit 保留DIS_MOV 位置</td></tr>
</table>

0x0E	0x04:H	Icon_Mode	1	ICON 显示模式：0x00=透明 0x01=不透明
0x0F	0x04:L	Icon_Lib	1	图标库存储位置
0x10	0x05	ICON0S	2	不显示动画模式，bit_0 图标ID 显示动画模式，bit_0 图标动画起始 ID 位置
0x12	0x06	ICON0E	2	显示动画模式，bit_0 图标动画结束 ID 位置
0x14	0x07	ICON1S	2	不显示动画模式，bit_1 图标ID 显示动画模式，bit_1 图标动画起始 ID 位置
0x16	0x08	ICON1E	2	显示动画模式，bit_1 图标动画结束 ID 位置
0x18	0x09	(x,y)	4	起始位变量显示位置，图标左上角坐标位置
0x1C	0x0B	DIS_MOV	2	下一个图标坐标移动坐标间隔
0x1E	0x0C	保留	2	写0x00

位变量图标的使用需要配合增量调节控件使用。操作之前需要制作好图标库文件。

在变量配置功能中找到位变量图标控件，然后划取适量的区域，对右面弹出的属性框进行操作，具体操作如下：

描述指针：FFFF 表示由配置文件加载；

描述指针：采用默认即可；

变量地址：自己定义；

辅助地址：不用操作；

设置：将自己想要显示图标的位勾选上，不想要显示的位不勾选；

如下图 7.10 所示

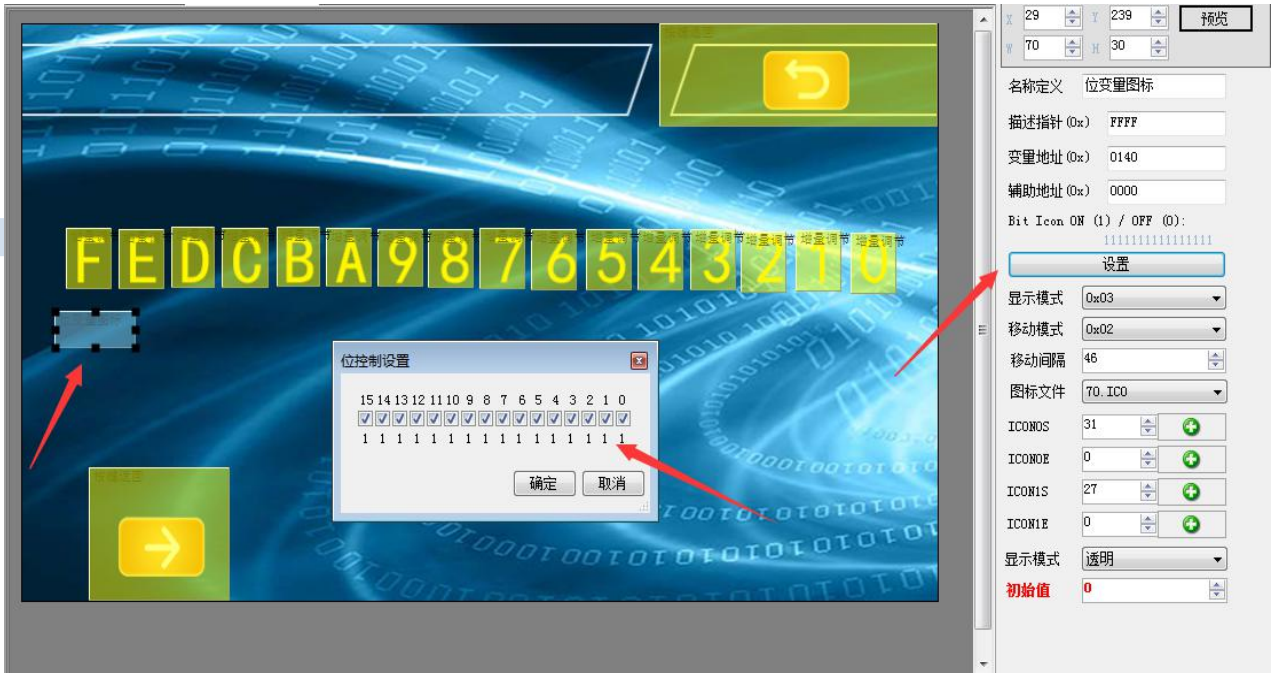


图 7.10 位变量图标控件位显示的设置

显示模式：定义了显示模式，此处选用 0x03（表示当该比特位的值为 0 时，不显示，为 1 的时候显示 ICON1S 定义的显示图标）；

图标文件：选用自己制作的图标库文件；

ICON0S：不显示动画模式，就是比特值为 0 的时候需要显示的图标（0 的时候不显示就是选取空白，直接输入值即可，比如你制作的图标一共两个然后地址就是 0 和 1，如果你不想要显示图标的话，此处只需要手动输入 2 即可）。 显示动画模式 0 比特时图标动画起始 ID 位置；

ICON0E：显示动画模式，0 比特时图标动画终止 ID 位置；

ICON1S：不显示动画模式，比特值为 1 的时候需要显示的图标，找到你先要显示的图标，勾选过滤背景色。 显示动画模式，1 比特时动画图标开始 ID 位置；

ICON1E：显示动画模式，1 比特时动画图标结束 ID 位置；

移动模式：定义了位图图标的显示方式，选用 0x02（表示是横向移动，当对应的比特位置的值是 1 的时候才显示保留上一个图标位置并做出相对位移）；

移动间隔：根据两个位之间的距离，设置一个合适的值即可；

显示模式：透明；

默认值：0；

7.3 文本的显示

7.3.1 数据变量显示

数据变量控件功能是把一个数据变量按照指定格式（整数、小数、是否带单位）用指定字体和大小的阿拉伯数字显示出来。其指令存储格式如表 7.10 所示。

表 7.10　数据变量显示指令存储格式

地址		定义	数据长度	说明
0x00		0x5A10	2	固定值 0x5A10
0x02		*SP	2	变量描述指针，0xFFFF 表示由配置文件加载
0x04		0x000D	2	固定值 0x000D
0x06	0x00	*VP	2	变量指针
0x08	0x01	X,Y	4	起始显示位置，显示字符串左上角坐标。
0x0C	0x03	COLOR	2	显示颜色
0x0E	0x04:H	Lib_ID	1	ASCII 字库位置
0x0F	0x04:L	字体大小	1	字符 X 方向点阵数
0x10	0x05:H	对齐方式	1	0x00=左对齐 0x01=右对齐 0x02=居中
0x11	0x05:L	整数位数	1	显示整数位
0x12	0x06:H	小数位数	1	显示小数位 整数位数和小数位数之和不能超过 20。
0x13	0x06:L	变量数据类型	1	0x00=整数（2 字节），范围为-32768 到 32767 0x01=长整数（4 字节），范围为-2147483648 到 2147483647 0x02=*VP 高字节，无符号数，范围 0 到 255 0x03=*VP 低字节，无符号数，范围 0 到 255 0x04=超长整数（8 字节），范围为 -9223372036854775808 到 9223372036854775807 0x05=无符号整数（2 字节），范围为 0 到 65535 0x06=无符号长整数（4 字节），范围为 0 到 4294967295
0x14	0x07:H	Len_unit	1	变量单位（固定字符串）显示长度，0x00 表示没有单位显示
0x15	0x07:L	String_Unit	Max11	单位字符串，ASCII 编码

首先找到一张 bmp 格式图片。先在新建工程中添加进去。然后在变量配置中找到数据变量按钮，并画出显示数据的区域，此时右侧会弹属性框，进行如下操作：

描述指针：FFFF 表示由配置文件加载；

变量地址：可以自行修改；本例是 2005

显示颜色：可以点击黑框根据自己的喜好进行修改；

字库位置：字库位置默认是 0 号字库不需要修改；
字体大小：可以根据自己习惯修改，比如 32；
对齐方式：一般设置为居中，也可以自己定义；
整数位数：自己定义但不宜过多，可以定义为 3；
小数位数：自己定义但不宜过多，不需要的话可以设置为 0；
显示单位：就是根据需求选填或者不填（比如此处是电压单位的话这里就可以填写 V）；
具体参数如下图 7.11 所示

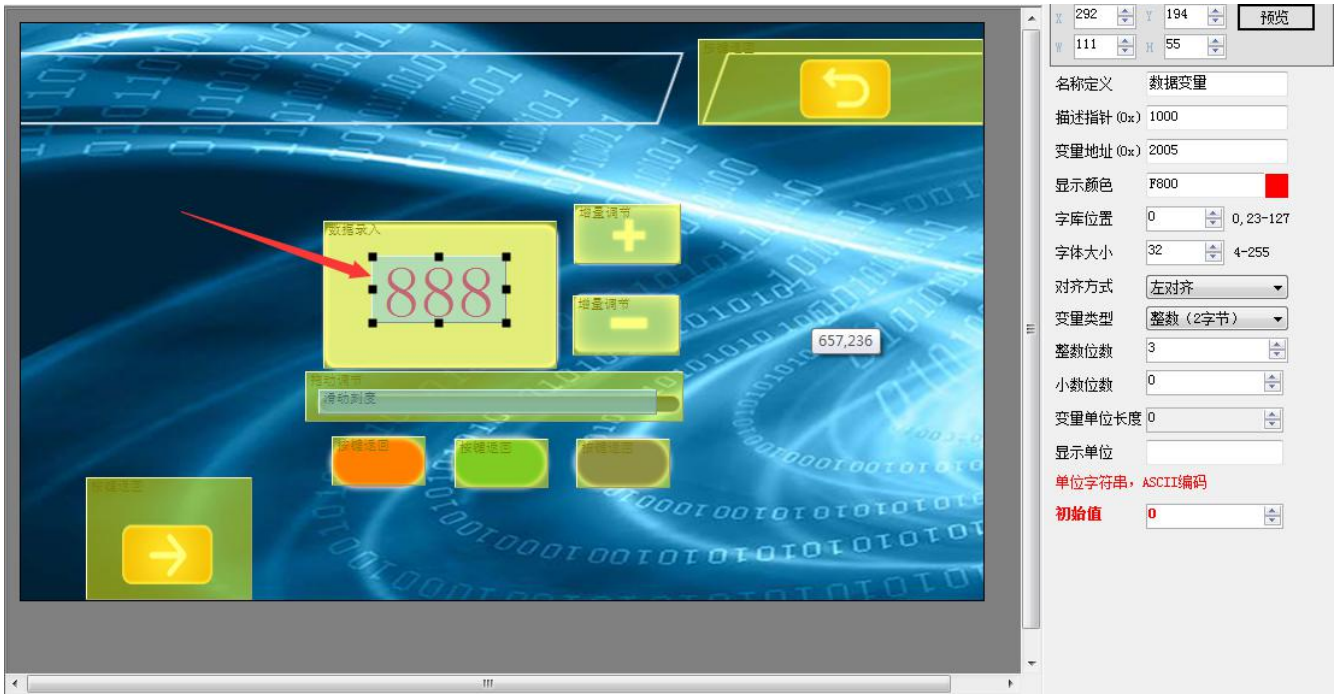


图 7.11 数据变量控件的属性框设置

7.3.2 文本显示

文本显示功能是把字符串按照指定的格式（选择字库决定）在指定的文本框显示区域显示。该功能通常配合文本录入功能使用。文本显示控件的使用需要用到汉字字库，这个需要提前设置好。本例配置的是 GBK 编码方式的 25 号字库。

其指令存储格式如表 7.11 所示。

表 7.11　文本显示指令存储格式

地址		定义	数据长度	说明
0x00		0x5A11	2	固定值 0x5A11
0x02		*SP	2	变量描述指针，0xFFFF 表示由配置文件加载
0x04		0x000D	2	固定值 0x000D
0x06	0x00	*VP	2	文本指针

0x08	0x01	（X，Y）	4	起始显示位置，显示字符串左上角坐标。
0x0C	0x03	Color	2	显示文本颜色
0x0E	0x04	(Xs,Ys)(Xe,Ye)	8	文本框
0x16	0x08	Text_Length	2	显示字节数量，当遇到 0xFFFF、0x0000 或者显示到文本框尾将不再显示。
0x18	0x09:H	Font0_ID	1	编码方式为 0x01-0x04 时的 ASII 字符使用的字库位置。
0x19	0x09:L	Font1_ID	1	编码方式为 0x00、0x05，以及 0x01-0x04 时的非 ASCII 字符使用的字库位置。
0x1A	0x0A:H	Font_X_Dots	1	字体 X 方向点阵数（0x01-0x04 模式，ASCII 字符的 X 方向点阵数按照 X/2 计算）
0x1B	0x0A:L	Font_Y_Dots	1	字体 Y 方向点阵数
0x1C	0x0B:H	Encode_Mode	1	.7 定义了文本显示的字符间距是否自动调整； .7=0 字符间距自动调整； .7=1 字符间距不自动调整，字符宽度固定为设定的点阵数。 .6-.0 定义了文本编码方式： 0=8bit 编码 1=GB2312 内码 2=GBK 3=BIG5 4=SJIS 5=UNICODE
0x1D	0x0B:L	HOR_Dis	1	字符水平间隔
0x1E	0x0C:H	VER_Dis	1	字符垂直间隔
0x1F	0x0C:L	未定义	1	写 0x00

【注】字体 Y 方向点阵数目必须为偶数。KGUS 屏预装 0#字库，包含 4*8--6*128 点阵的所有 ASCII 字符。

在变量配置中选择文本显示，并在需要显示的区域划取。然后右侧会显示属性框在属性框中进行修改实现文本的显示。

描述指针：FFFF 表示由配置文件加载；

变量地址：可选取默认值也可自己定义；

显示颜色：自主定义，本例是 0020；

编码方式：选取 0x02=GBK；

文本长度：自主定义（此处是字节数，2 个字节代表一个文字）；

（1）GBK 录入模式下的文本显示操作：

FONT0_ID：这是 ASCII 字库位置，此模式下忽略；

FONT1_ID：这是非 ASCII 字库位置，也就是我们需要使用到的汉子字库；

X 方向点阵数：根据自己制定的文字库选择；

Y 方向点阵数：根据自己制定的文字库选择；

水平间隔：自主定义；

垂直间隔：自主定义；

GBK 录入模式操作完成后的属性框大致如下图 7.12 所示

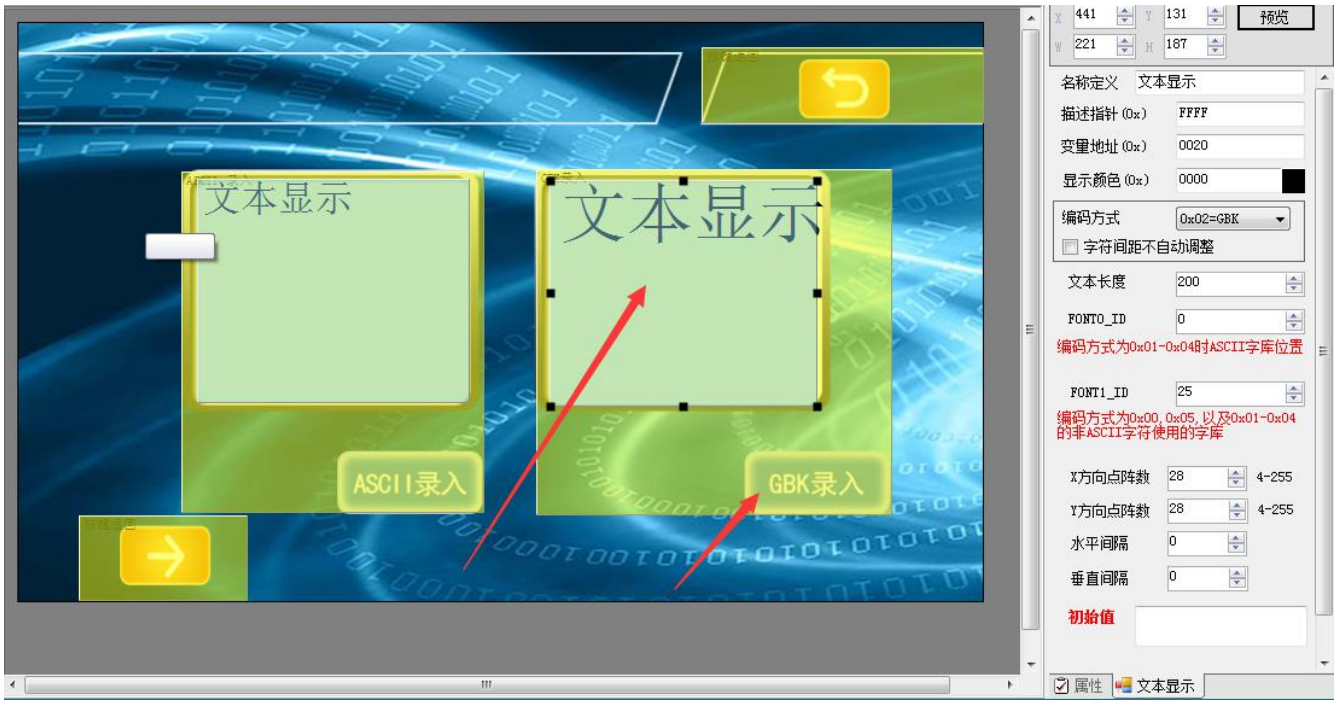


图 7.12 GBK 录入模式下的属性框设置

（2）ASCII 录入模式下的文本显示操作：

FONT0_ID：这是 ASCII 字库位置，也就是 0 号字库；

FONT1_ID：这是非 ASCII 字库位置，此模式下忽略；

X 方向点阵数：根据自己制定的文字库选择；

Y 方向点阵数：Y 方向的点阵数必须是 X 方向的点阵数的二倍（由于 X 方向的点阵数是 16 所以此时 Y 方向的点阵数就是 32）；

水平间隔：自主定义；

垂直间隔：自主定义；

ASCII 录入模式操作完成后的属性框大致如下图 7.13 所示

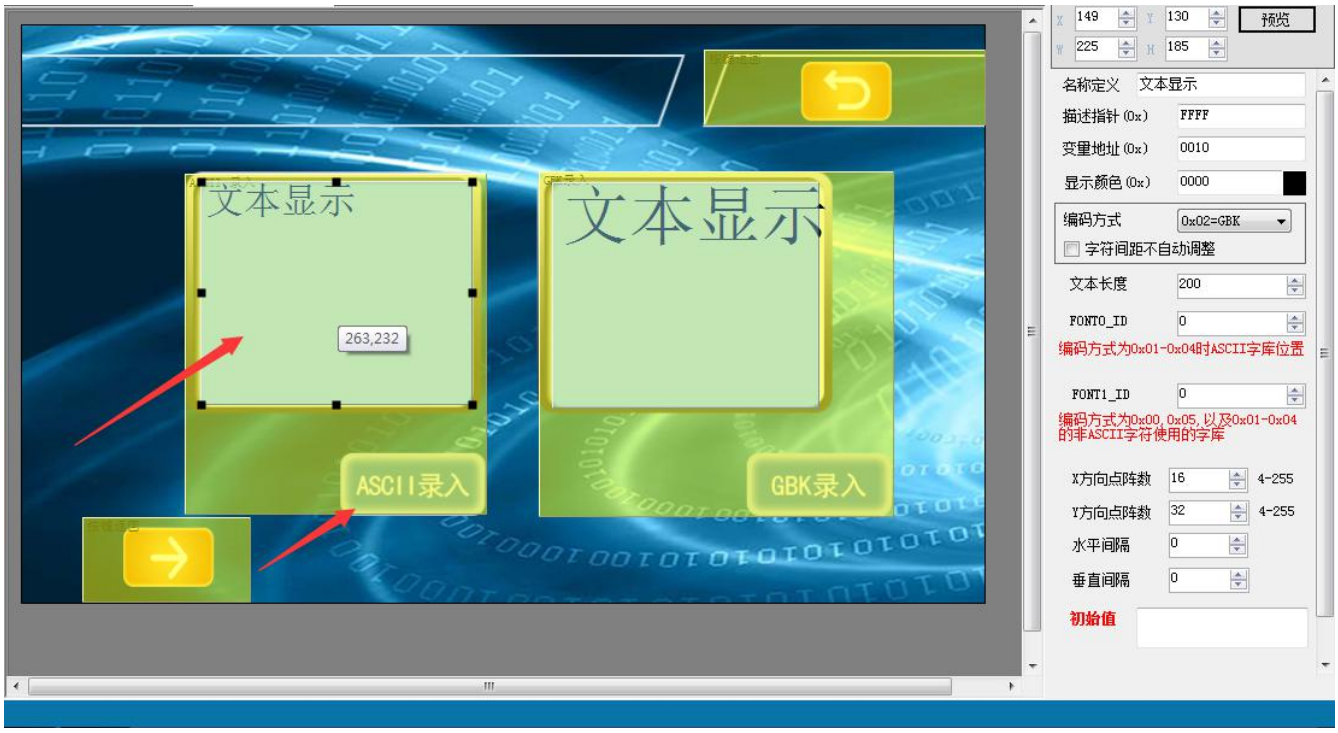


图 7.13 ASCII 录入模式下的属性框设置

7.3.3 RTC 显示

1) 文本格式 RTC 显示

文本时钟显示功能是按照用户编辑的格式把公历 RTC 用文本显示出来。指令存储格式如表 7.12 所示。

表 7.12 文本时钟显示指令存储格式

地址		定义	数据长度	说明
0x00		0x5A12	2	固定值 0x5A12
0x02		*SP	2	变量描述指针，0xFFFF 表示由配置文件加载
0x04		0x000D	2	写 0x000D
0x06	0x00	0x0000	2	写 0x0000
0x08	0x01	(X,Y)	4	显示位置，显示字符串左上角坐标。
0x0C	0x03	Color	2	字库颜色
0x0E	0x04:H	Lib_ID	1	字库位置
0x0F	0x04:L	字体大小	1	X 方向点阵数
0x10	0x05	String_Code	MAX16	编码字符串，使用 RTC 编码和 ASCII 字符构成。 假设当前时间是 2012-05-02 12:00:00 星期三，那么 Y-M-D H:Q:S0x00 将显示为 2012-05-02 12:00:00 M-D W H:Q 0x00 将显示为 05-02 WED 12:00

RTC 编码如表 7.13 所示。

表 7.13 RCT 编码

说明	编码	显示格式
公历_年	Y	2000-2099
公历_月	M	01-12
公历_日	D	01-31
公历_小时	H	00-23
公历_分钟	Q	00-59
公历_秒	S	00-59
公历_星期	W	SUN MON TUE WED THUFRI SAT
编码结束	0x00	

该控件只是将时间显示，若想通过触控进行修改，需要和 RTC 设置控件配合使用。

在变量配置中选取 RTC 显示，在右侧弹出的属性框中操作完成 RTC 的显示。

描述指针：FFFF 表示由配置文件加载；

字体颜色：可根据个人喜好更改（此处选用的是默认颜色黑色）；

字库位置：0 号字库；

X 方向点阵数：根据自己显示区域的大小可自由调试（由于显示区域小，又想要显示字幕不太小。所以此处选用 12 号，将年月日和时分秒周分为两个 RTC 显示，这样就可以完全显示）；

操作完成后的属性框具体如下图 7.14.1 和图 7.14.2 所示

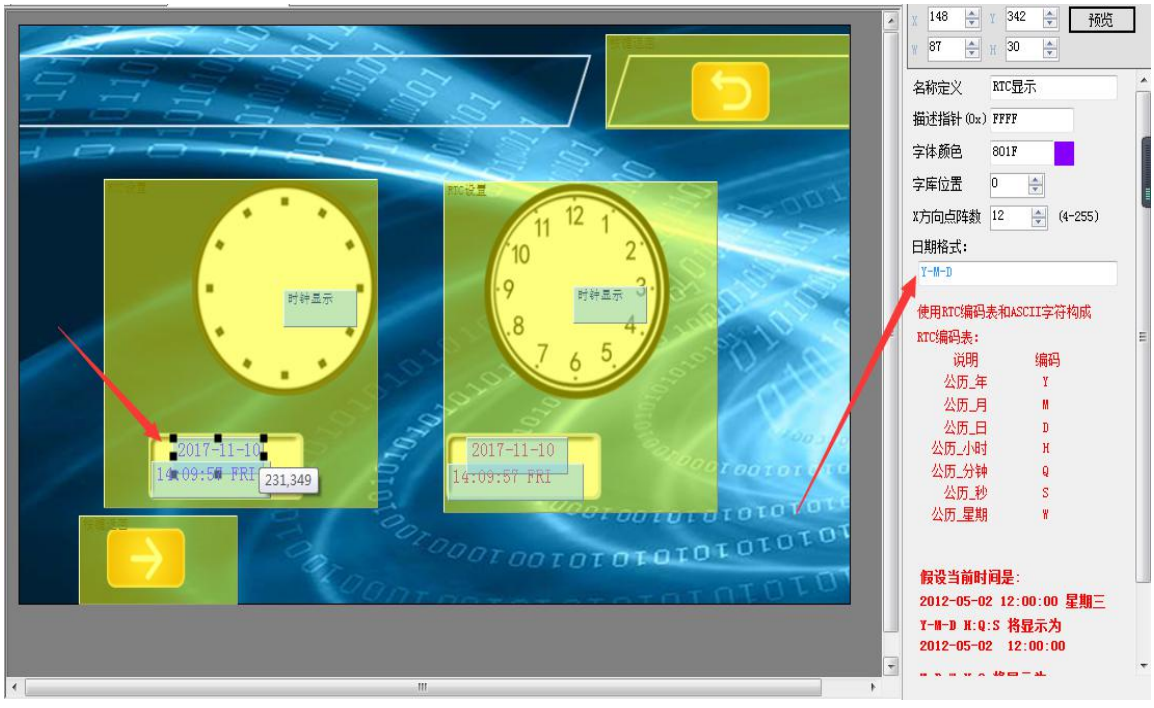


图 7.14.1 RTC 显示属性框设置

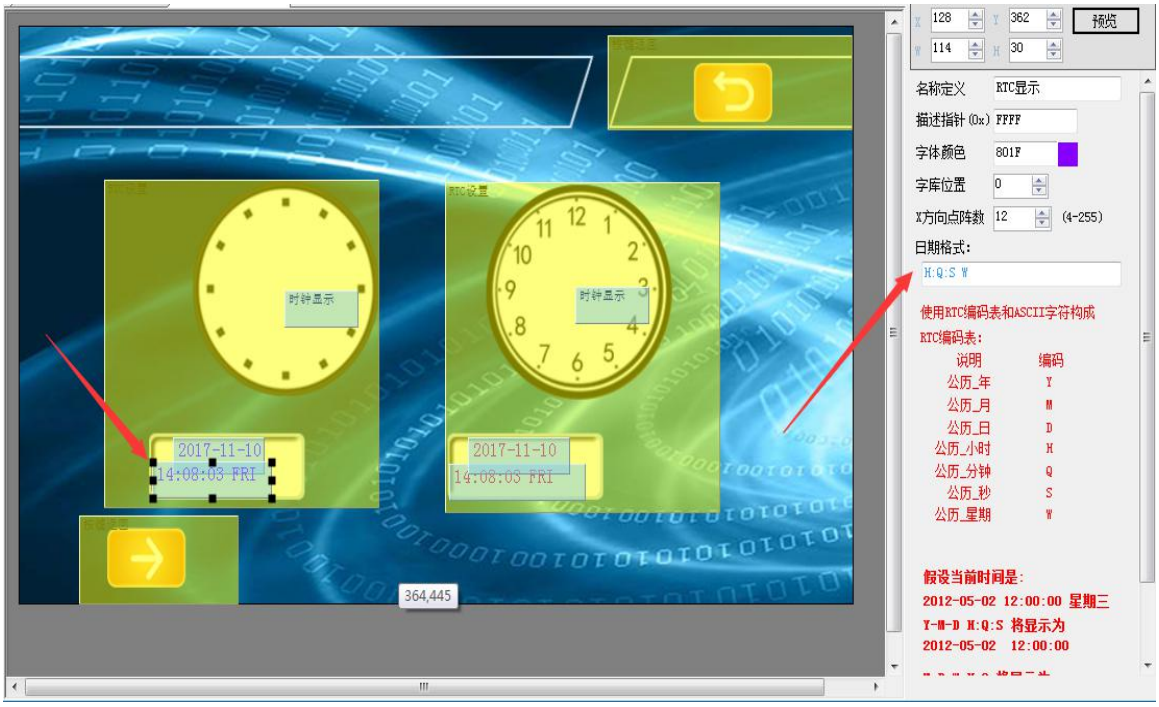


图 7.14 .2 RTC 显示属性框设置

2) 表盘时钟 显示

表盘时钟显示功能采用 ICON 图标旋转，用指针表盘方式把公历 RTC 显示出来。指令存储格式如表 7.14 所示。

表 7.14 表盘时钟显示指令存储格式

地址		定义	数据长度	说明
0x00		0x5A12	2	固定值 0x5A12
0x02		*SP	2	变量描述指针，0xFFFF 表示由配置文件加载
0x04		0x000D	2	写 0x000D
0x06	0x00	0x0001	2	写 0x0001
0x08	0x01	（X,Y）	4	时钟表盘的指针中心。
0x0C	0x03	Icon_Hour	2	指针 ICON 的 ID，0xFFFF 表示时针不显示。
0x0E	0x04	Icon_Hour_Central	4	时针 ICON 的旋转中心位置。
0x12	0x06	Icon_Minute	2	分针 ICON 的 ID，0xFFFF 表示分针不显示。
0x14	0x07	Icon_Minute_Central	4	分针 ICON 的旋转中心位置。
0x18	0x09	Icon_Second	2	秒针 ICON 的 ID，0xFFFF 表示秒针不显示。
0x1A	0x0A	Icon_Second_Central	4	秒针 ICON 的旋转中心位置。
0x1E	0x0C:H	Icon_Lib	1	指针图标所在的 ICON 库文件 ID
0x1F	0x0C:L	未定义	1	写 0x00

首先要知道该控件是要配合 RTC 显示、RTC 设置等控件共同使用的。并且在制作工程之前需要准备好含有表针的图标，使用工具菜单中的 ICO 生成工具生成图标库文件，并且将生成的图标库文件放到工程目录下的 YK_SET 文件夹中。

然后在变量配置功能中找到表盘时钟显示按钮，在需要显示的图片划取一块区域，然后对右侧弹出的属性框进行操作，具体操作如下：

描述指针：FFFF 表示由配置文件加载；

图标文件：选用自己制作好的图标库文件；

时针图标：在生成的图标库文件中选取自己希望用作时针的图标，并且选择好旋转中心，如图 7.15 所示

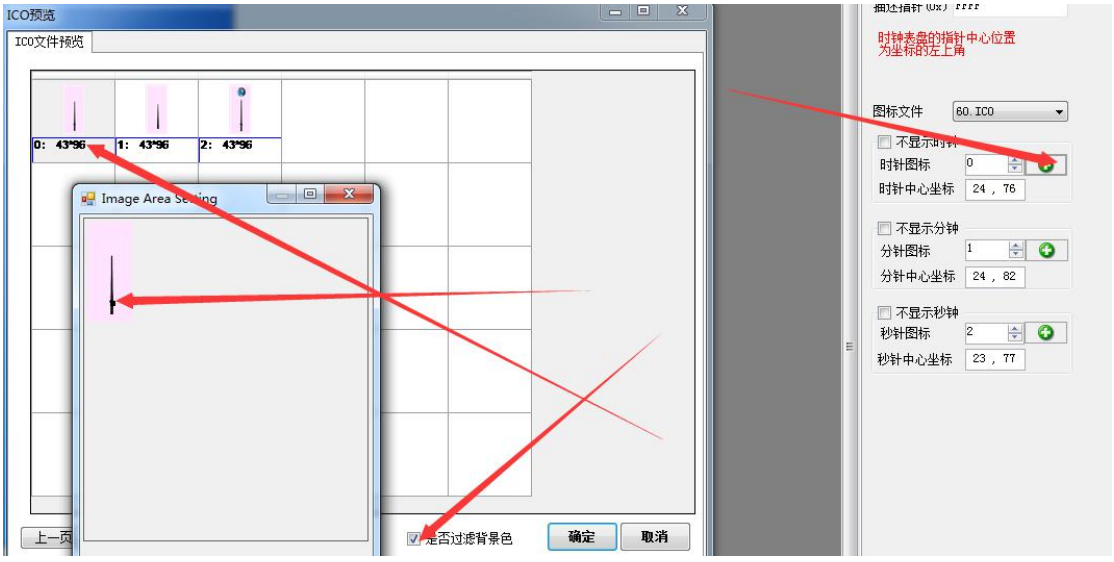


图 7.15 表盘时钟属性框设置

时针中心坐标：自动生成；

剩下的分针和秒针也进行同样的操作即可。

7.3.4 时间变量显示

时间变量显示功能是把变量数据按照字节 HEX 方式间隔用户指定的 ASCII 字符显示出来。比如把 1234 显示为 12:34 。其指令存储格式如表 7.15 所示。

表7.15 时间变量显示指令存储格式

地址		定义	数据长度	说 明
0x00		0x5A13	2	固定值 0x5A13
0x02		*SP	2	变量描述指针，0xFFFF 表示由配置文件加载
0x04		0x000D	2	写 0x000D
0x06	0x00	*VP	2	变量指针数据串首地址，变量为 BCD（HEX）编码。 比如：数据 0x32 显示为 32； 数据 0xBF 将 2 显示为 BF。
0x08	0x01	（X,Y）	4	字符串显示起始位置左上角坐标。
0x0C	0x03	Color	2	字体颜色
0x0E	0x04:H	Byte_Num	1	*VP 指针高字节开始显示的字节数目，0x01-0x0F
0x0F	0x04:L	Lib_ID	1	字库位置：字库必须是半角方式。 如果 Lib_ID 不为 0，字库必须使用 8bit 编码。
0x10	0x05:H	Font_x	1	X 方向点阵数目。
0x11	0x05:L	String_Code	MAX15	编码字符串，用来和时间变量组合出客户需要的显示格式。 每显示一个 BCD 时间码后，会到编码字符串顺序取出一个 ASCII 字符来间隔显示。 编码字符串中，特殊字符定义如下： 0x00：无效，本字符不显示，两个 BCD 时间码将连在一起显示； 0x0D：换行显示，即 X=Xs，Y=Y+Font_X*2。

时间变量控件主要是用来显示时间的，通过串口通信进行数据录入。

在变量配置功能中选取时间变量控件，然后将图片中想要显示时间的区域划取出来，对右侧弹出的属性框进行操作，具体操作如下所示：

区域范围设置：(X,Y)为字符串左上角显示在当前页面位置的坐标。**名称定义**：为按钮设置一个名称，在“变量查看”中方便查询。

描述指针：定义描述文件数据存储器的地址。默认为 0xFFFF，由配置文件加载（一般不用修改）。

变量地址：变量指针数据串首地址，变量为 BCD 编码。

显示颜色：字体显示的颜色。

字库位置：调用的字库位置，字库必须是半角方式；如果字库位置不为 0，字库必须使用 8bit 编码。字体大小：字体 X 方向点阵数目。

Byte_Num： VP 指针高字节开始显示的字节数目，0x01-0x0F。

编码字符：直接输入想要最终显示的 ASCII 字符，字符转换工具。

输入 16 进制数据：点击转换，自动将编码字符串转换成 16 进制数据，此项也可以直接输入。

图7.16 时间变量显示功能配置菜单

注意：编码字符串是用来和变量数据组合出用户需要显示的格式。每显示一个 BCD 码后，会到编码字符串中按顺序取出一个 ASCII 字符来间隔显示。0x00 表示无效，本字符不显示，两个 BCD 码将连在一起。0x0D 表示换行显示。

【例】如上图所示，假设在编码字符串中输入“-- :: ”，点击转换后，16 进制数据为 2D2D203A3A20。假设变量是 0x20171020102216，则最终输出显示为 17-10-20 10:22:16 。

7.3.5 文本滚屏显示

文本滚屏显示功能是把变量数据在指定区域内按指定方向滚动显示。其指令存储格式如表 7.16 所示。

表 7.16　文本滚屏显示指令存储格式

地址		定义	数据长度	说明
0x00		0x5A14	2	固定值 0x5A14
0x02		*SP	2	变量描述指针
0x04		0x000B	2	固定值 0x000B
0x06	0x00	*VP	2	文本指针。 文本指针前 3 个字必须保留，用户显示文本内容从（VP+3）开始存放。文本必须以 0xFF 或 0x00 结尾。
0x08	0x01:H	Rolling_Mode	1	滚屏模式：0x00=从右向左滚屏。
0x09	0x01:L	Rolling_Dis	1	滚屏间距，每个 DGUS 周期文本滚动的像素点阵数。
0x0A	0x02:H	Adjust_Mode	1	0x00=左对齐 0x01=右对齐 0x02=居中 文本显示内容不足文本框时滚屏停止，此时显示对齐模式方有效。
0x0B	0x02:L	未定义	1	写 0x00
0x0C	0x03	Color	2	显示文本颜色
0x0E	0x04	Xs Ys Xe Ye	8	文本框区域
0x16	0x08:H	Font0_ID	1	编码方式为 0x01-0x04 时，ASCII 字符显示的字库位置。 编码方式为 0x00、0x05 时，该参数不要设置，写 0x00 即可。
0x17	0x08:L	Font1_ID	1	编码方式为 0x01-0x04 时，非 ASCII 字符显示的字库位置。 编码方式为 0x00、0x05 时，显示字符使用的字库位置。
0x18	0x09:H	Font_X_Dots	1	字体 X 方向点阵数（0x01-0x04 模式，ASCII 字符 X 将自动按照 X/2 计算）。
0x19	0x09:L	Font_Y_Dots	1	字体 Y 方向点阵数目。
0x1A	0x0A:H	Encode_Mode	1	.7 定义了显示的字符间距是否自动调整： .7=0 字符间距自动调整； .7=1 字符间距不自动调整，字符宽度为设定的点阵数。 .0 到.6 定义了文本编码方式： 0=8bit 编 码 1=GB2312 内 码 2=GBK 3=BIG5 4=SJIS 5=UNICODE
0x1B	0x0A:L	Text_Dis	1	字符间距
0x1C	0x0B:H	未定义	4	写 0x00

注意：滚动文本的长度必须大于划取区域的长度，否则将不会呈现滚屏的效果。

首先在变量配置中找到滚动文本控件按钮，然后对弹出的属性框进行操作，具体操作如下：

描述指针：FFFF 表示由配置文件加载；

变量地址：自行定定义（本例定义的是 0260）；

显示颜色：根据自己喜好；

编码方式：0x02=GBK，选用自己制定的编码方式也可以；

滚屏模式：根据自己习惯选择；

滚屏间距：可自行定义但是不可以为 0，代表的是文本滚屏的像素点阵数；

对齐方式：自主选择；

FONT0_ID: 0 号字库；

FONT1_ID: 自己之前制作的文本字库；

X 方向点阵数： 自己之前制作的文本字库的点阵数；

Y 方向点阵数： 自己之前制作的文本字库的点阵数；

字符间距：自主选择；

初始值：可以直接填写，也可以不直接填写。如果选用直接填写的话，记得要在之前空出三个字的距离，因为这些地址被系统占用了，必须空出来否则会出现缺字现象。如果不直接填写可以通过 UltraEdit 软件在 22_Config.bin 文件中进行赋值。注意赋值的时候一定要注意地址是十六进制的就是要将地址 0260 转变成十进制然后再乘以二，再转变回十六进制。这时候的地址才是真正显示数据的地址（0260 转变成十进制是 0608，608*2=1216,1216 转变成十六进制是 04C0，也就是说在 04C0 地址输入数据才是可以显示的真实数据，切记前面要空出六个字节的空隙）。如下图 7.21 所示

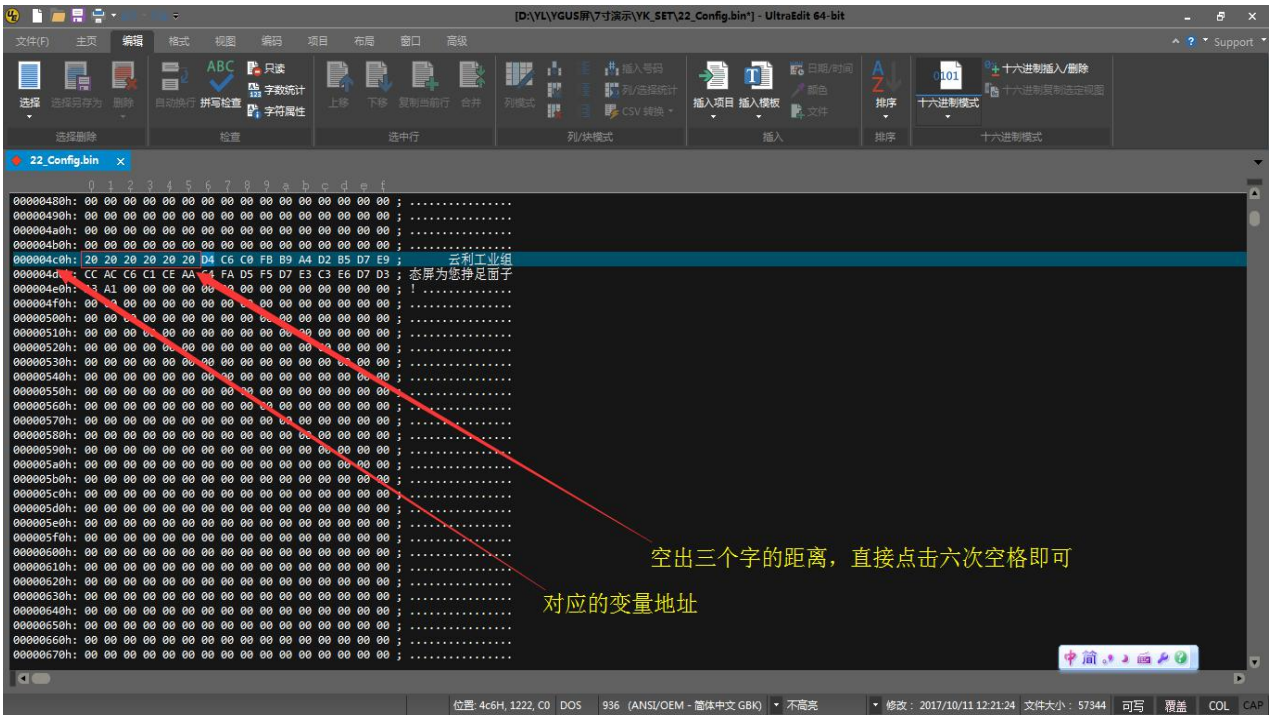


图 7.21 滚屏显示输入

设置完成后的属性框大致如下图 7.22 所示

图 7.22 滚动文本属性框设置

7.4 图形变量的显示

7.4.1 实时曲线（趋势图）显示

实时曲线（趋势图）显示功能是结合 0x84 指令写曲线缓冲区数据来自动匹配显示实时曲线（趋势图）。可以指定显示区域，中心轴坐标、显示比例（放大/缩小）可控。该功能的指令存储格式如表 7.17 所示。

表 7.17 实时曲线（趋势图）显示指令存储格式

地址		定义	数据长度	说 明
0x00		0x5A20	2	固定值 0x5A20
0x02		*SP	2	变量描述指针，0xFFFF 表示由配置文件加载
0x04		0x000A	2	固定值 0x000A
0x06	0x00	0x0000	2	无定义
0x08	0x01	Xs,Ys Xe,Ye	8	曲线窗口左上角坐标（Xs,Ys）和右下角坐标（Xe,Ye） 曲线越界将不显示。
0x10	0x05	Y_Central	2	曲线中心轴位置
0x12	0x06	VD_Central	2	中心轴对应的曲线数据值，一般取数据最大值和最小值之和的一半。
0x14	0x07	Color	2	曲线颜色
0x16	0x08	MUL_Y	2	纵轴放大倍数，单位是 1/256，0x0000-0x7FFF。
0x18	0x09:H	CHANEL	1	数据源通道，0x00-0x07
0x19	0x09:L	Dis_HOR	1	横轴间隔，0x01-0xFF。

实时曲线控件的使用需要结合以后的串口通信发送指令来使用，此处只做简单介绍。

首先在变量配置中找到实时曲线按钮，然后将需要显示曲线的区域划取出来，然后修改右边弹出的属性框，具体操作如下：

描述指针：FFFF 表示由配置文件加载；

Y_Central：就是 Y 坐标的中心值；

如下图 7.23 所示

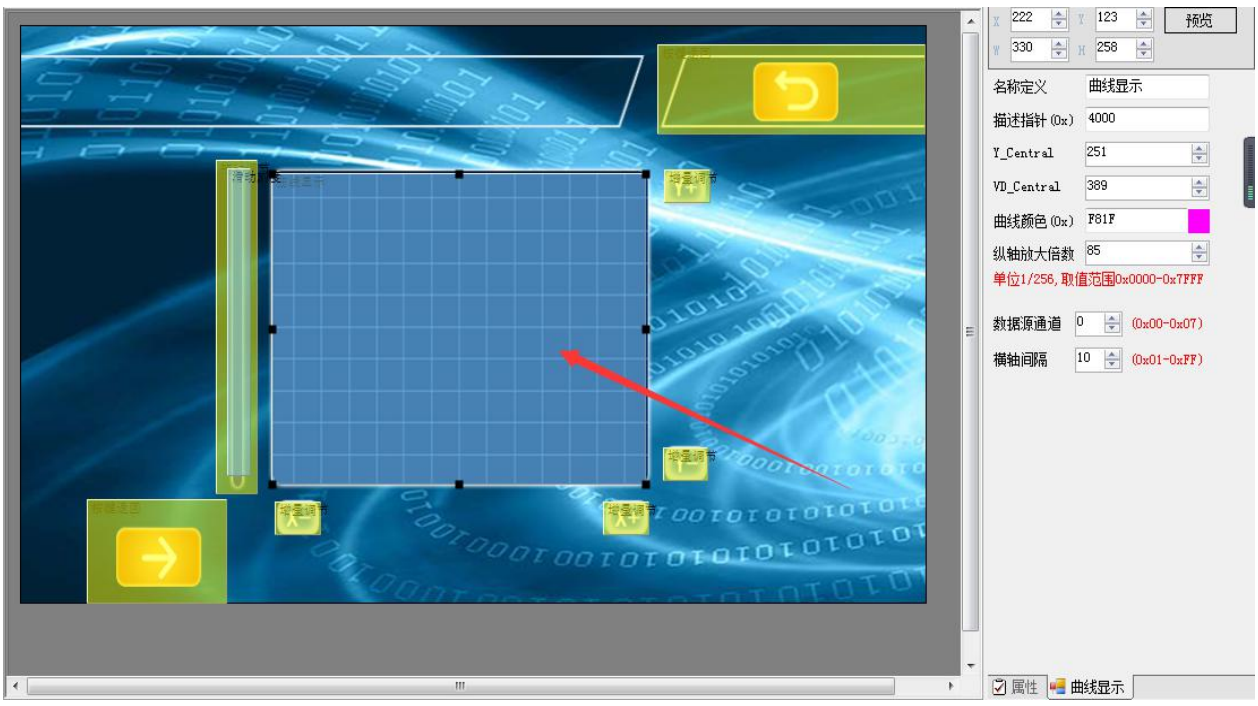


图 7.23 曲线显示属性框设置

VD_Central：中心轴对应的曲线数据值，一般取最大值和最小值和的一半；

曲线颜色：自主定义即可；

满量程曲线的纵轴放大倍数计算：MUL_Y=(Ye-Ys)*256/(Vmax-Vmin)，其中 Ye,Ys 为曲线窗口的 Y 坐标，Vmax,Vmin 为曲线数据的最大最小值。比如一个 12bit 的 A/D 采集数据(Vmax=4095,Vmin=0)要对应在 Ys=50,Ye=430 的屏幕区域满量程显示， 那么经计算 MUL_Y=23.7，向下舍入取 23。

数据源通道：定义了八个（0x00~0x07）只可以选取其中的一个通道显示曲线；
横轴间隔：横轴的间距，根据自己需求，自主定义（取值范围 0x01~0xFF）；

如果需要显示曲线线条比较粗，可在同一个位置放置多个上下（Y 轴）平移的曲线变量并引用同一个数据源通道即可实现。

注意：如果把变量描述内容存储在数据存储空间（*SP 指定的存储位置），那么结合增量触控指令，可实现无需用户代码干预的曲线自动缩放；结合拖动触控指令修改 Y_Central 值，则可实现无需用户代码干预的曲线上下移动。

7.4.2 基本图形显示

基本图形显示功能是在显示配置文件 14.BIN 中定义一个“绘图板”功能，而具体的绘图操作则由 *VP 指向的变量存储器的内容决定。用户通过改变变量存储器中的数据来实现不同的绘图功能。其指令存储格式如表 7. 18 所示。

表 7.18 基本图形显示指令存储格式

地址		定义	数据长度	说 明
0x00		0x5A21	2	固定值 0x5A21
0x02		*SP	2	变量描述指针，0xFFFF 表示由配置文件加载
0x04		0x0008	2	固定值 0x0008
0x06	0x00	*VP	2	变量数据指针
0x08	0x01	Area	8	绘图显示区域的左上角坐标、右下角坐标；绘图越界将不显示。 仅对 0x0001-0x0005、0x0009、0x000A、0x000B 指令有效。
0x10	0x05:H	Dashed_Line_En	1	0x5A：使用线段的绘图指令（0x02、0x03、0x09、0x0A 指令）将使用虚线或者点划线显示线段； 其他：使用线段的绘图指令使用实线显示线段。
0x11	0x05:L	Dash_Set	4	4 个字节依次设置了虚线（点划线）格式： 第 1 段实线点阵数、第一段虚线点阵数、第 2 段实线点阵数、第 2 段虚线点阵数。 比如，设置 0x10 0x04 0x10 0x04 将显示虚线；设置 0x10 0x04 0x02 0x04 将显示点划线。
0x15	0x07	未定义	13	保留，写 0x00

变量数据指针所指向的变量数据格式说明如表 7. 19 所示。

表 7.19 变量数据指针所指向的变量数据格式说明

地址	定义	说明
VP	CMD	绘图指令
VP+1	Data_Pack_Num_Max	最大数据包数据：连线指令（0x0002）,定义为连线线条数目（顶点数-1）;
VP+2	DATA_Pack	数据

绘图指令数据包说明如表 7.20 所示。

表 7.20 绘图指令数据包说明

指令 (CMD)	操作	绘图数据包格式说明（相对地址和长度单位均为字（word)）			
		相对地址	长度	定义	说 明
0x0001	置点	0x00	2	(x,y)	置点坐标位置，x 坐标高字节为判断条件。
		0x02	1	Color	置点颜色
0x0002	端点连线	0x00	1	Color	线条颜色
		0x01	2	(x,y)0	阵线顶点 0 坐标，x 坐标高字节为判断条件。
		0x03	2	(x,y)1	阵线顶点 1 坐标，x 坐标高字节为判断条件。
		0x01+2*n	2	(x,y)n	阵线顶点 n 坐标，x 坐标高字节为判断条件。
0x0003	矩形	0x00	2	(x,y)s	矩形框左上角坐标，x 坐标高字节为判断条件。
		0x02	2	(x,y)e	矩形框右下角坐标。
		0x04	1	Color	矩形颜色。

0x0004	矩形域填充	0x00	2	(x,y)s	矩形框左上角坐标，x 坐标高字节为判断条件。
		0x02	2	(x,y)e	矩形框右下角坐标。
		0x04	1	Color	矩形域填充颜色。
0x0005	整圆弧显示	0x00	2	(x,y)	圆心坐标，x 坐标高字节为判断条件。
		0x02	1	Rad	半径
		0x03	1	Color	圆弧颜色
0x0006	图片区域剪切、粘贴	0x00	1	Pic_ID	剪切图片区域所在页面 ID；高字节为判断条件
		0x01	2	(x,y)s	剪切图片区域左上角坐标。
		0x03	2	(x,y)e	剪切图片区域右下角坐标。
		0x05	2	(x,y)	剪切图片区域粘贴到当前页面坐标位置的左上角坐标。
0x**07	ICON 图标显示	0x00	2	(x,y)	显示坐标位置，x 坐标高字节为判断条件。
		0x02	1	ICON_ID	图标 ID，图标库位置由指令高字节指定。 图标固定为不显示背景色。
0x0008	区域填充	0x00	2	(x,y)	种子点坐标，x 坐标高字节为判断条件。
		0x02	1	Color	填充颜色。
0x0009	频谱显示(垂直线条）	0x00	1	Color0	把(X0,Y0s）(x0,Y0e）用 Color0 颜色连线，X0 高字节为判断条件。
		0x01	3	X0,Y0s,Y0e	
0x000A	线段显示	0x00	1	Color	把(Xs,Ys）(xe,Ye）用 Color 颜色连线，Xs 高字节为判断条件。
		0x01	2	(Xs,Ys)	
		0x03	2	(Xe,Ye)	
0x000B	圆弧显示	0x00	1	Color0	圆弧显示颜色
		0x01	2	(X,Y)0	圆心（X,Y）坐标，X 坐标高字节为判断条件。
		0x03	1	RAD0	半径
		0x04	1	DEG_S0	起始角度，单位 0.5°，范围 0-720
		0x05	1	DEG_E0	终止角度，单位 0.5°，范围 0-720
0x000C	字符显示	0x00	1	Color0	字符显示颜色
		0x01	2	(X,Y)0	字符显示位置，字符左上角坐标，X 坐标高字节为判断条件。
		0x03H	0.5	Lib_ID	字库位置
		0x03L	0.5	En_Mode	字符编码模式：0=8bit 1=GB2312 2=GBK 3=BIG5 4=SJIS 5=UNICODE
		0x04H	0.5	X_Dots	字符 X 方向点阵数
		0x04L	0.5	Y_Dots	字符 Y 方向点阵数
		0x05	1	Text0	字符数据，对 8bit 编码，仅高字节有效。当编码方式为 01-04 时，如果字符数据为 ASCII 字符，将自动使用 0#预装字库显示。
0x000D	矩形域 XOR	0x00	2	(x,y)s	矩形域左上角坐标，x 坐标高字节为判断条件。
		0x02	2	(x,y)e	矩形域右下角坐标。
		0x04	1	Color	矩形域做 XOR 的颜色，0xFFFF 将进行反色操作。
0x000E	双色位图显示	0x00	2	(x,y)s	位图显示矩形域左上角坐标，x 坐标高字节为判断条件。
		0x02	1	X_Dots	位图 X 方向点阵数目
		0x03	1	Y_Dots	位图 Y 方向点阵数目
		0x04	1	Color1	"1"bit 对应的显示颜色
		0x05	1	Color0	"0"bit 对应的显示颜色；如果设置 Color0 和 Color1 相同，表示"0"bit 不需要显示，直接跳过。
		0x06	N	Data_Pack	显示数据，MSB 方式：为方便用户读写数据，每行数据必须对齐到一个字，即下一行数据总是从一个新数据字（Word）开始。
0x000F	位图显示	0x00	2	(x,y)s	位图显示矩形域左上角坐标，x 坐标高字节为判断条件。
		0x02	1	X_Dots	位图 X 方向点阵数目
		0x03	1	Y_Dots	位图 Y 方向点阵数目
		0x04	N	Data_Pack	显示数据，每个像素点一个字（MSB，5R6G5B 数据格式）。
0x0010	区域放大一倍粘贴显示	0x00	2	(x,y)	放大一倍后图像粘贴在屏幕左上角坐标，x 高字节为判断条件。
		0x02	2	(x,y)s	待放大矩形域左上角坐标。

受变量空间大小限制，最大显示位图为 196*146（4:3）或 226*126（16:9）

待放大区域位于放大后图像区域内时，必须右下角对齐。嵌套放大可以得到更大的

	0x04	2	(x,y)e	待放大矩形域右下角坐标。 放大倍数。

基本图形控件的实现可以通过串口指令完成。

首先在变量配置中找到基本图形按钮，然后选中图片并划取需要显示图形的区域，右侧弹出的属性框：

描述指针：FFFF 表示由配置文件加载；

变量地址：自主定义（本例设定的变量地址是 0300）；

虚线\点划线：不用勾选；

然后生成配置。至此软件的操作就算完成了。接着就是将配置拷贝到显示屏里去，同时打开串口调试工具，输入指令即可完成基本图形控件的功能。指令具体如下图 7.24 所

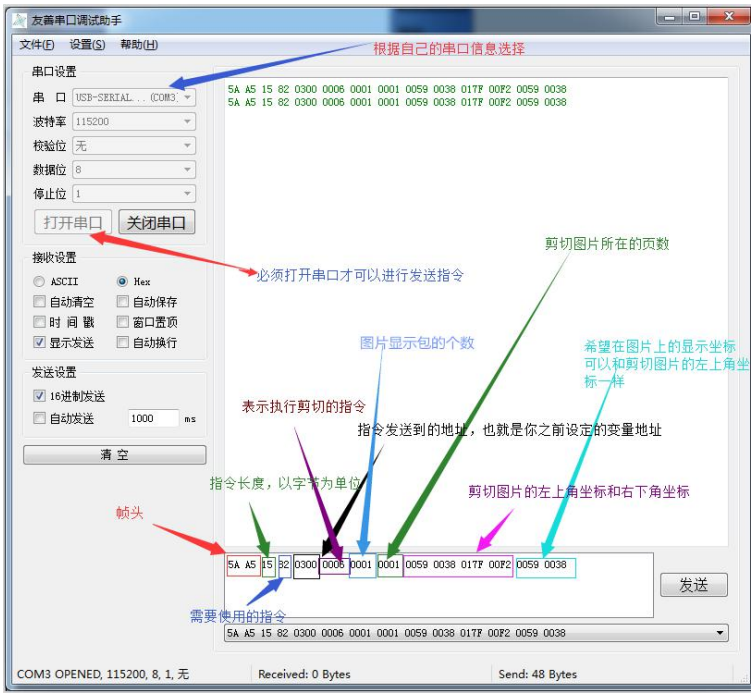


图 7.24 基本图形显示剪切粘贴指令

如果想要实现其他绘图功能可以参考使用指南里的具体指令。

【例】用上面的方法通过 KGUS 开发软件在某一页中加入一个画板，之后将该程序通过 SD 卡下载到 KGUS 屏中，并通过串口发送如表 7.21 中格式的指令。每次上电后之前绘制的图形都会清空。若要保证每次上电后都存有固定的图形，可通过 22 变量初始化文件（22*.BIN）来实现。

表 7.21 绘图指令举例说明

帧头	数据长度	指令	变量地址	绘图指令(CMD)	数据包个数	左上角/右下角坐标	颜色（红）
5A A5	11	82	03 00	00 03	00 01	00 64 00 64 02 BC 01 90	F8 00

根据上面对绘图指令包的说明，绘图指令 0x0003 为画矩形，因此该指令是通过串口向变量存储器（地址为 0x0300）中写入数据，实现绘制一个红色矩形框的功能。其他绘图实现方法与此类似，用户可自行尝试。

7.4.3 列表显示

列表显示功能是把按照二维数组定义的数据用表格分栏显示出来。其指令存储格式如表 7.22 所示。

表 7.22 列表显示指令存储格式

地址	定义	数据长度	说 明
0x00	0x5A22	2	固定值 0x5A22
0x02	*SP	2	变量描述指针，0xFFFF 表示由配置文件加载
0x04	0x000C	2	固定值 0x000C
0x06 0x00	*VP	2	表格描述指针，即 TAB[TAB_X_Num][TAB_Y_Num]数组的首地址。

0x08	0x01:H	TAB_X_Num	1	列数目，0x01-0xFF
0x09	0x01:L	TAB_Y_NUM	1	行数目，0x01-0xFF
0x0A	0x02:H	TAB_X_Start	1	表格起始显示列位置，0x00-0xFF
0x0B	0x02:L	TAB_Y_Start	1	表格起始显示行位置，0x00-0xFF
0x0C	0x03:H	Unit_Data_Num	1	0x01-0x7F 所有单元格存储数据长度相同 一个单元格所占的数据空间长度（Word，字长度）。 0x00 由*VP 指针指向的变量存储空间定义了不同列单元的数据长度（Word，字长度）。 当 Unit_Data_Num=0x00 时，表格数据内容存储位置相应后延（TAB_X_NUM/2）向上取整个字地址。 比如，*VP=0x1000,TAB_X_Num=0x07，那么： 0x1000-0x1003 依次存储了第 0-6 列的表格数据长度，其中 1003 的低字节未使用。0x1004 地址开始存储表格内容。
0x0D	0x03:L	Encode_Mode	1	.7 定义了文本显示的字符间距是否自动调整： .7=0 字符间距自动调整； .7=1 字符间距不自动调整，字符宽度固定为设定的点阵数。 .6 定义了表格内容格式 .6=0 表格内容为文本格式； .6=1 表格内容格式由单元格数据的前两个字表示，见备注[1]； .5 定义了边框线条是否显示： .5=0 显示边框； .5=1 不显示边框； .4 未定义，写 0。 .3-.0 定义了文本编码方式： 0=8bit 1=GB2312 内码 2=GBK 3=BIG5 4=SJIS 5=UNICODE
0x0E	0x04	Xs Ys Xe Ye	8	表格显示区域的左上角及右下角坐标： 表格总是从左上角位置开始显示，越界将结束显示。
0x16	0x08	Color_Line	2	表格边框线条颜色
0x18	0x09	Color_Text	2	表格文本显示颜色
0x1A	0x0A:H	Font0_ID	1	编码方式 0x01-0x04 时 ASCII 字库位置。
0x1B	0x0A:L	Font1_ID	1	编码方式 0x00、0x05，以及 0x01-0x04 的非 ASCII 字符使用的字库。
0x1C	0x0B:H	Font_X_Dots	1	字体 X 方向点阵数（0x01-0x04 模式，ASCII 字符 X 按照 X/2 计算）
0x1D	0x0B:L	Font_Y_Dots	1	字体 Y 方向点阵数目
01E	0x0C:H	TAB_X_Adj_Mod	1	当设置 TAB_X_Start 不为 0 时，进行显示表头控制： 0x00=首列不显示 0x01=首列显示
0x1F	0x0C:L	TAB_Y_Adi_Mod	1	当设置 TAB_Y_Start 不为 0 时，进行显示表头控制： 0x00=首行不显示 0x02=首行显示

备注[1]：当 Encode_Mode.6=1 时，每个单元格数据内容前两个字定义了表格的数据格式，其说明如下：

第一个字高字节：Mode 选择数据类型

0x00=整数（2 字节），范围-32768 到32767

0x01=长整数（4 字节），范围-2147483648 到2147483647

0x02=*VP 高字节，无符号数，范围 0 到255

0x03=*VP 低字节，无符号数，范围 0 到255

0x04=超长整数（8 字节），范围-9223372036854775808 到9223372036854775807

0x05=无符号整数（2 字节），范围0 到65535

0x06=无符号长整数（4 字节），范围0 到4294967295

0x10=时间格式1，12:34:56 BCD 码串

0x11=时间格式2，12-34-56 BCD 码串

0x12=时间格式 3，YYYY-MM -DD HH:MM:SS BCD 码串

0xFF=

文本格式第一个字低字节：

Mode=0x00-0x06 定义了变量数据的定点显示格式，高 4bit 表示整数位数，低 4bit 表示小数位数。

Mode=0x10-0x11 时间 BCD 码串的字节长度

Mode=其他无定义

第二个字：定义了单元格文本颜色。

【注】如果表格实际内容短于 Unit_Data_Num 规定的长度时，使用 0xFFFF 做为单元格文本结束符。对于特别大表格，通过触摸屏修改 TAB_X_Start 和TAB_Y_Start 值可很方便的实现表格的定位和拖动。

【注】表格中的数据会在掉电后丢失，若要在表格中显示固定数据，可通过 22 变量初始化文件（22*.BIN）来实现。

首先在变量配置中找到列表显示并划取想要显示列表的区域，然后操作右侧弹出的属性框，具体操作如下：

描述指针：FFFF 表示由配置文件加载；

变量地址：可自己定义（本例采用的变量地址 0600）；

列数：根据自己划取区域的大小合理设置（本例设置的是 3，取值范围为 0x01~0xFF）；

行数：根据自己划取区域的大小合理设置（本例设置的是 7，取值范围为 0x01~0xFF）；

起始显示行：自行定义（本例定义的 0，取值范围为 0x00~0xFF）；

起始显示列：自行定义（本例定义的 0，取值范围为 0x00~0xFF）；

Unit_Data_Num（0x）：表示列宽（可自主定义）；

采用默认的 0，在采用 0 的时候需要使用到 ULTRAEDIT 软件对 YK_SET 文件夹中的 22_config.bin 文件进行自定义，具体如下图 7.25 所示

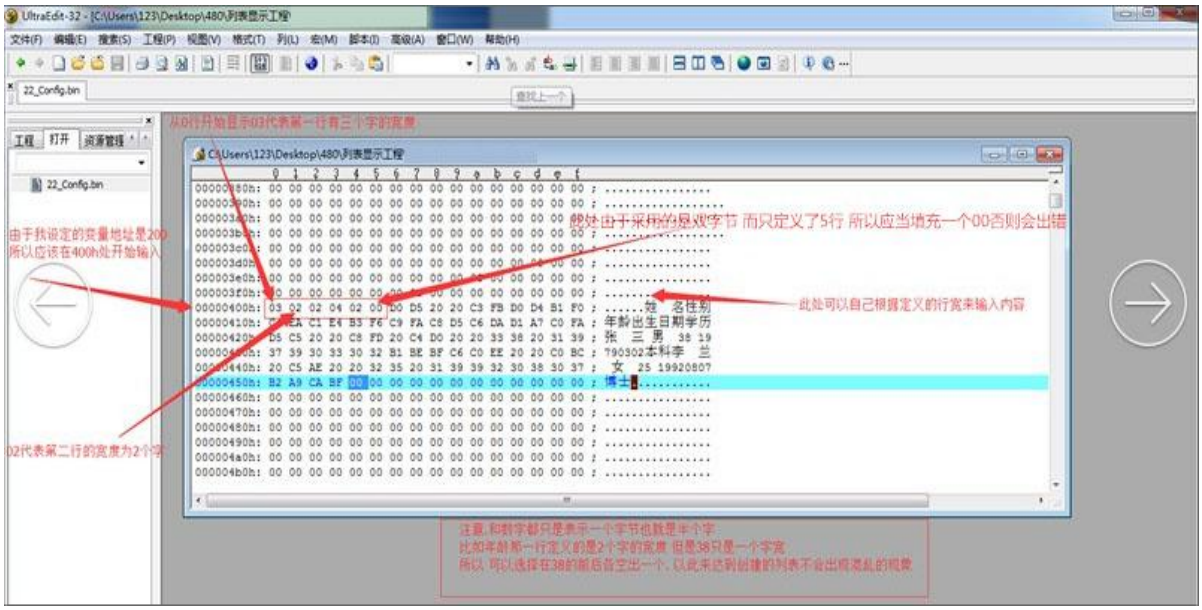


图 7.25 自主定义列宽时的操作

取值为 01~7F 之间时表示等列宽，例如 03，表示所有列宽为三个字；

然后在 YK_SET 文件夹中的 22_config.bin 中就不需要之前的制定行宽，直接可以写入；

如下图 7.26 所示

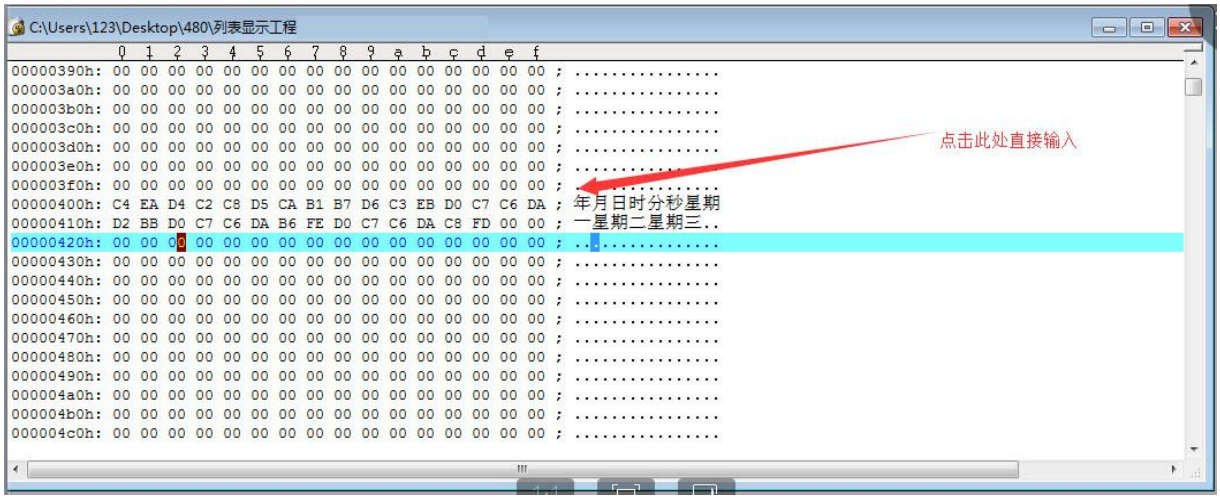


图 7.26 限定列宽时的操作

编码方式：0x02=GBK；

边框颜色：自主定义；

文本颜色：自主定义；

FONT0_ID：选用默认的 0 号字库；

FONT1_ID：选用之前定义好的字库（本例选用的是 25 号 28 点阵的字库）；

注意：制定好的字库必须放到 YK_SET 文件夹中，才可以使用。

X 方向点阵数：根据自己定义的字库填写；

Y 方向点阵数：根据自己定义的字库填写；

首列显示控制：自主定义；

操作完成后的属性框大致如图 7.27 所示

图 7.27 列表显示控件的属性框设置

7.4.4 二维码显示

二维 QR 码显示功能是根据指定内容在屏幕显示指定的二维码图形。其指令存储格式如表 7.23 所示。

表 7.23 二维 QR 码图形显示指令存储格式

地址		定义	数据长度	说 明
0x00		0x5A25	2	固定值 0x5A25
0x02		*SP	2	变量描述指针，0xFFFF 表示由配置文件加载
0x04		0x0004	2	固定值 0x0004
0x06	0x00	*VP	2	二维码显示内容指针。 二维码内容最长 458Bytes,0x0000 或 0xFFFF 为结束符。
0x08	0x01	(x,y)	4	二维码显示的左上角坐标位置。 二维码图形有 45*45 单元像素（数据少于 155 字节）和 73*73 单元像素（数据少于 459Bytes)两种。
0x0C	0x03	Unit_Pixels	2	每个二维码单元像素所占用的物理像素点阵大小，0x01-0x07。 设置 Unit_Pixels=4，那么每个单元像素将显示为 4*4 点阵大小。
0x0B-0x1F		保留	18	未定义，写 0x00。

二维码控件的使用需要结合 GBK 录入或者通过串口通信配合实现。

首先在一张图片上划取需要显示二维码的区域，然后对其右侧的属性框进行操作，

描述指针：FFFF 表示由配置文件加载；

变量地址：自主定义；

Unit_Pixels：表示每个二维码像素所占用的物理点阵大小，取值范围为 0x01~0x07；