

VGUS 4.3 用户开发指南

适用于 SDW-PlusII 系列串口屏

武汉中显科技有限公司

www.viewtech.cn



修改记录

日期	修改内容
2015-6-1	初稿。
2015-8-15	增加音频播放功能。
2015-10-10	增加视频播放功能。
2015-11-28	增加二维码显示功能。
2016-4-7	增加文本滚动显示功能。
2016-8-9	1、 增加 ViewTech OS 编程指令； 2、 增加 RTC 录入后自动上传功能； 3、 文件夹名称为 “VT_SET-REPLACE” 时，只更新同名文件，不删除其它工程文件； 4、 支持大图标（大于 255 点阵）显示。
2016-9-5	1、 插入 USBMini 后支持串口接收指令； 2、 增加 85 00 指令写显存功能； 3、 上电启动亮度为屏保点亮亮度 R6，可在工程中设置启动亮度，修改亮度寄存器中亮度值最高位为 1 保存亮度； 4、 数据变量录入和数据变量显示增加数据反色显示； 5、 数据变量增加无效位补零选项； 6、 变量数据录入增加输入长度限制。
2016-11-4	增加寄存器和变量寄存器的指令操作举例。
2016-12-12	增加设置音频播放模式指令。
2017-01-13	增加寄存器 0x10-0x1D SD 卡配置寄存器映射功能。



目 录

1 VGUS 4.3 概述	5
1.1 认识中显 VGUS 屏	5
1.2 开发流程	6
1.2.1 界面及图标设计	6
1.2.2 屏参配置	8
1.2.3 分配变量存储地址并设置控件属性	8
1.2.4 生成配置文件	9
1.2.5 预览及下载	9
1.2.6 串口调试	9
1.3 集成大容量 FLASH 存储器	9
1.4 在线下载调试与 U 盘脱机拷贝	11
1.4.1 USB-mini 接口在线下载调试	11
1.4.2 U 盘脱机拷贝	11
1.4.3 用户数据加密与解密	12
1.5 固件更新	12
2 串口操作	13
2.1 指令帧结构	13
2.2 指令集	14
2.3 CRC 校验程序	15
3 寄存器	17
3.1 寄存器一览表	17
3.1.1 读取版本号	19
3.1.2 修改背光亮度	19
3.1.3 控制蜂鸣器	19
3.1.4 切换界面	19
3.2 时钟 (RTC) 读写	20
3.3 语音播放	20
3.4 视频播放	20
3.5 用户数据文件加载到变量空间	22
3.6 数据库读写	23
3.7 触摸屏校准	24
3.8 屏参配置读取	24
3.8.1 串口波特率设置 (R1/0x10)	25
3.8.2 串口指令帧头设置 (R3/0x13、RA/0x1A)	25
3.8.3 工作模式配置寄存器 1 (R2/0x12)	26
3.8.4 工作模式配置寄存器 2 (RC/0x1C)	26



3.8.5 屏保/触摸屏控制背光 (R2.5、R6、R7、R8)	27
4 输入控件	28
4.1 变量数据录入	29
4.2 弹出菜单选择	30
4.3 增量调节	31
4.4 拖动调节	31
4.5 RTC 设置.....	32
4.6 按钮键值返回	32
4.7 文本录入	33
5 显示控件	35
5.1 图标变量	37
5.1.1 变量图标显示 (0x00)	37
5.1.2 动画图标显示(0x01)	38
5.1.3 滑动刻度指示(0x02)	39
5.1.4 艺术字变量显示(0x03)	40
5.1.5 图片动画显示(0x04)	41
5.1.6 图标旋转(0x05)	41
5.1.7 位变量图标显示(0x06)	42
5.2 文本变量	43
5.2.1 数据变量显示(0x10)	43
5.2.2 文本显示(0x11)	44
5.2.3 RTC 显示(0x12).....	46
5.2.4 HEX 时间变量显示 (0x13)	47
5.2.5 文本滚屏显示 (0x14)	48
5.3 图形变量	49
5.3.1 实时曲线 (趋势图) 显示(0x20).....	49
5.3.2 基本图形显示(0x21)	50
5.3.3 列表显示 (0x22)	53
5.3.4 二维 QR 码图形显示 (0x25)	55
6 描述指针	56
6.1 描述指针介绍	56
6.2 描述指针应用举例.....	57
7 VIEWTECH_OS 用户程序设计	59
7.1 基本约定	59
7.2 VIEWTECH_OS 汇编指令集.....	60
附 1 外接键盘转接板 KAP02.....	68



1 VGUS 4.3 概述

1.1 认识中显 VGUS 屏

VGUS (Viewtech Graphical User Software) 是武汉中显科技有限公司一款组态型、用户图形界面设计软件, VGUS 屏系统架构如图 1-1 所示。

VGUS4.3 是针对 SDW-PlusII 系列串口屏开发的最新版本组态软件。

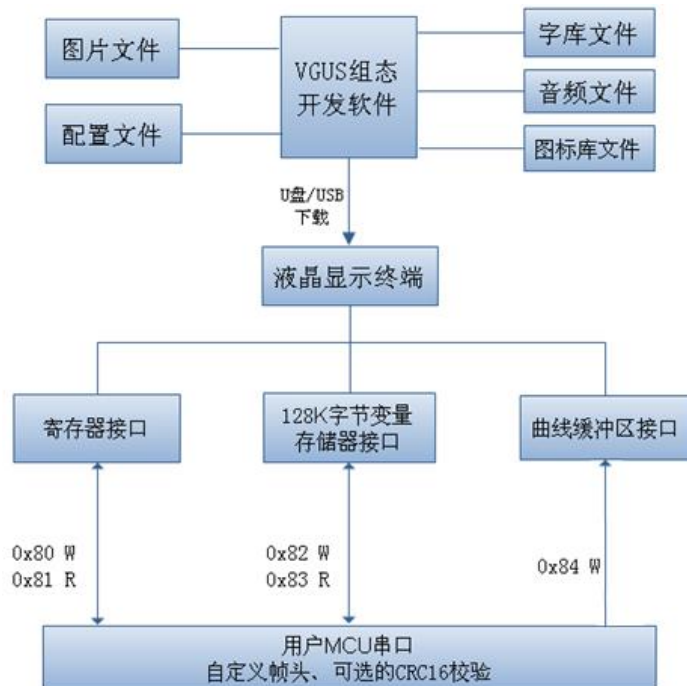


图 1-1 VGUS 屏系统架构

7 英寸 SDWe070T01T 的 VGUS 屏实物如图 1-2 所示。



图 1-2 7 英寸 VGUS 屏 SDWe070T01T 实物图

USB-mini 口用于在线调试下载数据（配置文件、图片、图标和字库），U 盘接口用于脱机批量下载数据（配置文件、图片、图标和字库），用户串口包含模块供电电源脚。

➤ VGUS 4.3主要特点

- ✧ 128K字节变量存储器空间，8通道曲线趋势图存储器，极快（80ms）的变量显示响应速度；
- ✧ 256字节寄存器空间，支持串口指令读写，用于硬件配置和控制操作；
- ✧ 128M Flash存储器空间，用于存储图片、图标、字库和用户数据库等（详见1.3节）；
- ✧ 单页最多支持128个显示变量；
- ✧ 支持USBmini在线下载调试和U盘脱机批量下载，提高研发、生产效率；
- ✧ 集成了RTC、背光亮度调节、背光自动待机、触控蜂鸣器伴音功能；
- ✧ 支持语音播放功能。

➤ VGUS屏数据格式

VGUS屏支持整数（双字节）、无符号整数（双字节）、长整数（4字节）、超长整数（8字节）等数据格式，其数值范围如表1-1所示。

表1-1 数值范围

数据格式	最小值	最大值
整数（双字节）	-32768（0x8000）	+32767（0x7FFF）
无符号整数（双字节）	0（0x0000）	65535（0xFFFF）
长整数（4字节）	-2147483648（0x80000000）	+2147483647（0x7FFFFFFF）
超长整数（8字节）	-9223372036854775808	9223372036854775807

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

➤ VGUS屏色彩定义

VGUS屏所有颜色数据均为16位，两个字节，如表1-1所示，其格式为Red5-Green6-Blue5，即红色占高5位，绿色占中间6位，蓝色占低5位。可以显示的颜色为 2^{16} 色，即65536色。

表1-2 色彩定义

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

例：红色：0xF800，绿色：0x07E0，蓝色：0x001F，白色：0xFFFF，黑色：0x0000

1.2 开发流程

VGUS屏开发流程如图1-3所示。

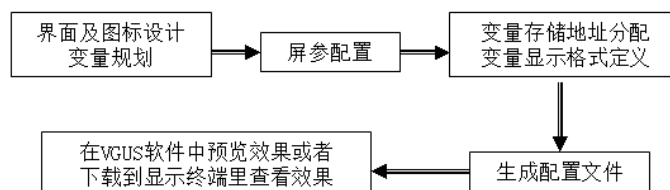


图1-3 VGUS屏开发流程

1.2.1 界面及图标设计

开始做工程前，首先设计好要使用的背景图片和图标，准备可能用到的字库文件等。



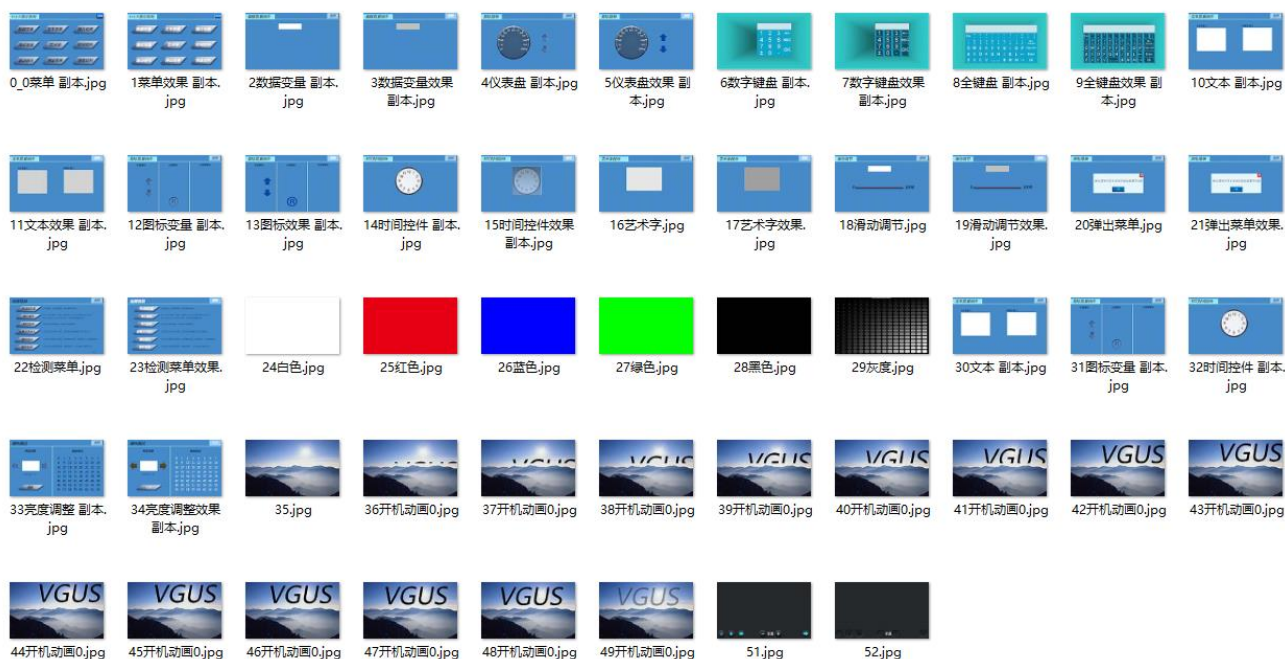


图 1-3 背景图片文件

- 图片的分辨率依据液晶屏的分辨率而定，所有型号屏的分辨率在对应的产品规格书都可以查到，如果下载与屏不匹配的分辨率的工程会导致显示异常。
- 上电默认显示 0 号图片，如果设计的工程里没有 0 号图片，则下载工程后会显示白屏。开机页面可在 VGUS4.3 开发软件的屏参配置中修改。

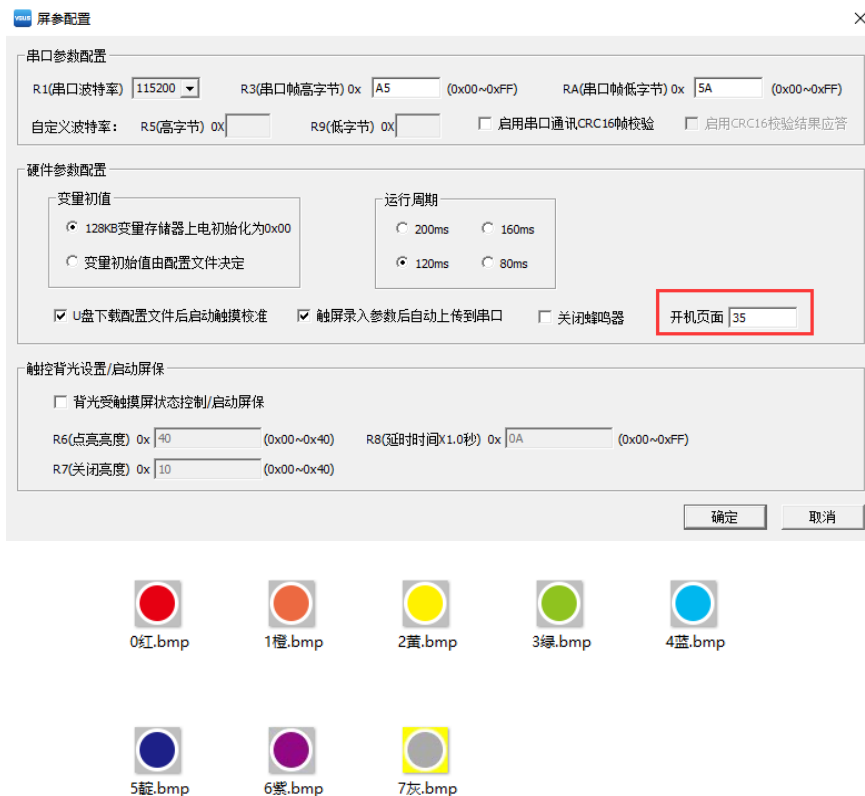


图 1-4 图标文件

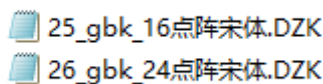


图 1-5 字库文件

1.2.2 屏参配置

屏参配置是通过 VGUS 4.3 开发工具完成,如图 1-6 所示。屏参配置中设置了串口波特率、指令帧头、CRC 校验、以及蜂鸣器、触摸屏和屏保等外设特性。用户可以用串口通过 0x10-0x1C 寄存器读取相关设置值。



图 1-6 屏参配置界面

1.2.3 分配变量存储地址并设置控件属性

通过VGUS开发工具,定义所有的变量存储地址和显示格式,生成配置文件并下载到VGUS显示终端里,用户通过串口发送变量值和对应变量存储地址给显示终端即可。VGUS组态屏具有64K字(128K字节)变量存储空间,地址范围0x0000-0xFFFF。

通过VGUS开发工具在1号界面上添加一个“数据变量”控件,设置“数据变量”的基本属性(包括变量存储地址、显示颜色、大小、显示方式等),如图1-7所示。然后生成配置文件,并下载到显示终端里。所有变量存储地址需要根据变量长度手动分配。



图1-7 数据变量基本属性设置

1.2.4 生成配置文件

VGUS软件中的变量配置完成后需在软件中生成配置文件并保存工程，如图1-8所示。



图1-8

1.2.5 预览及下载

工程可在VGUS软件中预览效果，也可通过USB或U盘下载到屏内测试功能，实际效果以下载到屏内为准。

1.2.6 串口调试

下载工程到屏内后，可以通过VGUS软件自带的串口调试助手或者第三方的串口调试工具调试变量的显示，模拟单片机发指令，在有测试主板的情况下，也可用主板直接进行串口调试。

1.3 集成大容量 Flash 存储器

VGUS屏内部集成有大容量Flash存储器，用于存储字库、图标库、图片、配置文件、音频文件和用户数据库等。

表1-3 VGUS 4.3支持的文件格式

文件格式	含义	编号限制
*.Jpg	图片文件	0-65535
*.ico	图标文件	0-255
*.dzk	用户字库文件	0-255
*.bin	用户数据文件	0-255
*.mp3	音频文件	0-4095
UserDb.bin	用户数据库文件	固定

➤ 文件命名规则

文件名必须以阿拉伯数字开头，数字代表图片的序号（从0开始编号）。例如，要把一副图片序号编为20，图片文件可以命名为“20_测试.jpg”或者“20.jpg”，但不能命名为“测试 20.jpg”。VGUS生成下载工程文件后会，文件名只保留数字，如20.jpg。

➤ 关于图片文件格式

VGUS4.3支持的图片文件格式为JPG和BMP，如果用户导入其它格式的图片文件，系统将自动转换为JPG格式，转换过程可能会导致色彩损失。

建议用户直接使用JPG格式，因为JPG格式占用空间小，显示速度快；BMP格式为原始位图文件，包含信息最全，如果使用JPG格式文件显示效果不能满足要求，用户可选择BMP格式文件，使用BMP格式会占用更多的存储空间，显示速度也会慢些；不要用画图、QQ截图等工具直接另存JPG格式文件，这样显示效果较差，因为JPG格式文件为有损压缩文件。建议首先用使用画图、QQ截图等工具保存BMP格式文件，再用 PhotoShop工具打开图片，然后另存为JPG格式，具体步骤如下：

- 1、用PhotoShop打开图片；
- 2、另存为jpg格式；



图 1-6 另存图片

- 3、选择图像效果“最佳”选项。



图1-7 另存图片品质选项

➤ 关于图标文件格式

图标源文件也支持BMP和JPG等多种格式。但是需要做图标透明显示的场合，必须使用BMP格式，因为icon生成工具提取透明图标的工作原理，是将源图片（0，0）坐标点的颜色作为背景过滤颜色。而JPG格式文件为有损压缩文件，背景颜色每个像素可能都有很小差异（可能人眼无法识别），会导致无法有效过滤背景。

icon源文件中的图片一定要以数字开头命名，不然无法生成，最好以连续的数字开头命名。

➤ 关于音频文件格式

VGUS4.3支持WAV和MP3两种音频文件格式。WAV格式占用空间大，语音立即输出无延迟，适合按键伴音等较短时间播放场合。MP3格式占用空间小，语音输出稍有延迟（0.5秒），适合语音提示等较长时间播放场合。

1.4 在线下载调试与 U 盘脱机拷贝

通过VGUS 4.3开发工具生成的配置文件（“VT_SET”文件夹下的所有内容），可以通过USBMini或者U盘两种方式下载到显示终端里。USBMini下载适合研发阶段使用；U盘下载适合研发定型后的批量生产使用，可以有效提高下载效率、降低对操作人员素质要求。

1.4.1 USB-mini接口在线下载调试

点击VGUS 4.0开发工具中的“下载配置文件”，将弹出一个窗口，如图1-6所示。将VGUS屏USB-mini接口连接到电脑并上电后，右下方将提示“USB已连接”。此时，点击右侧“下载”按钮，就可以将配置文件下载到显示终端里查看设计效果。

在修改调试过程，可能需要反复下载的是变量格式等信息，而图片、字库或者图标可能并没有修改，此时可以勾选图1-6上“不下载图片”、“不下载字库”、“不下载图标”等选项，可以有效提高下载速度。如果图片、字库或者图标有修改，此时就不能再勾选相应选项，必须将修改的文件重新下载。

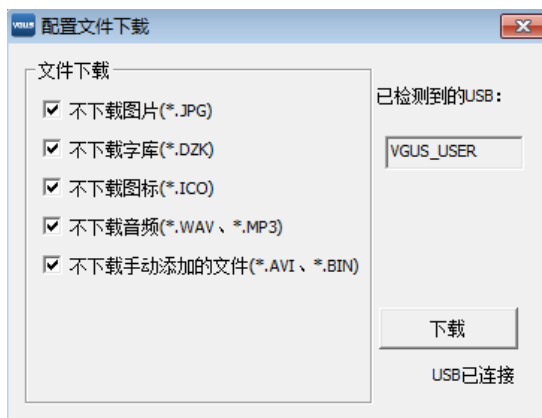


图 1-6 下载配置文件窗口

温馨提示：若出现配置文件下载失败的问题，请在电脑“我的电脑”中把屏对应的 U 盘格式化成 FAT32 的格式即可。

1.4.2 U盘脱机拷贝

✧ 将U盘格式化为FAT32文件格式，并且不要分区；

第一次使用U盘下载前，推荐将U盘格式化一次。VGUS屏文件下载仅支持FAT32文件格式。

✧ 把“VT_SET”文件夹拷到U盘根目录下；

在VGUS 4.3开发工具中项目设计完毕后，点击“生成配置文件”后，会在工程文件下生成“VT_SET”文件夹，如图1-7所示。

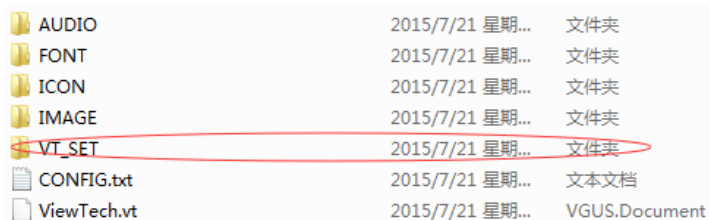


图1-7 项目文件夹

“VT_SET”文件夹下包含了所有的配置文件、字库、图片、音频文件和图标库等信息，用户不能对该

文件夹内文件做任何修改。

说明：当文件夹名称为“VT_SET-REPLACE”（中划线、下划线均可）时，只更新同名文件，不删除其它工程文件。

✧ 把U盘插入VGUS屏；

系统将自动启动数据拷贝，把U盘里面的数据逐一拷贝到VGUS屏里。

✧ 拷贝完成后VGUS屏蜂鸣器会提示下载完成，同时显示屏左上角提示“Please plug out the U disk.”；

✧ 拔出U盘，结束拷贝。

1.4.3 用户数据加密与解密

由于VGUS屏通过USB-mini接口连接到电脑后，实际是被虚拟成硬盘，通过电脑可以直接拷贝VGUS屏内部数据。因此，当用户设计完成以后，必须对VGUS屏内的数据进行加密保护，防止设计被拷贝抄袭。

用户首先自行制作一个加密文件“password.txt”，该文件包含一个不超过128字节的任意中英文字符串作为密码，用户需要牢记该密码；然后将密码文件“password.txt”复制到U盘根目录下，当将U盘插到VGUS屏上后，密码将自动拷贝到VGUS屏里，实现对VGUS屏内部数据的加密保护。

VGUS屏一旦加密后，用户将无法再对VGUS屏进行写入和读出操作，除非用解密文件“de_password.txt”对VGUS屏进行解密。

解密过程与加密相似，首先是将解密文件“de_password.txt”（密码必须与加密文件“password.txt”相同）复制到U盘根目录下，当将U盘插到VGUS屏上后，系统将自动对解密密码进行核对，如果与加密密码一致，VGUS屏将自动解密。经过解密后，用户可以重新对VGUS屏进行写入和读出操作。

加密文件“password.txt”与解密文件“de_password.txt”内的密码相同，仅仅是文件名不同。密码是一个不超过128字节的任意中英文字符串，必须妥善保存。

1.5 固件更新

✧ 将固件程序“VGUS-PlusII.bin”文件复制到U盘根目录下；

✧ 将U盘插入VGUS屏，系统立即启动固件更新



2 串口操作

VGUS屏采用异步、全双工串口（UART），串口工作模式是：8 n 1格式，即每字节数据传送采用10个位：1位起始位、8位数据位、1位停止位，无校验位。

串口波特率可通过VGUS 4.3开发工具中“屏参配置”设置。

2.1 指令帧结构

VGUS屏串口指令帧由6个数据块组成，如表2-1、2-2所示。所有的串口指令或者数据采用16进制（HEX）格式，数据采用高字节先传送（MSB）的方式。例如0x1234发送时先发送0x12，再发送0x34。

表2-1 串口发送指令帧结构 寄存器：80、81；变量存储器：82、83；绘图：84

定义	帧头	数据长度	指令	地址	数据内容	CRC 校验码
长度（字节）	2	1	1	1/2	N	2
说明	R3:RA 定义	包括指令、地址、数据和校验	0x80-0x84	寄存器地址/变量存储地址	-	R2.4 决定是否启用
举例	0xA5,0x5A	0x06	0x80	0x03	0x00, 0x01	0x18,0x24

表2-2 串口接收指令帧结构

定义	帧头	数据长度	指令	地址	数据内容长度	数据内容	CRC 校验码
长度（字节）	2	1	1	1/2	1	N	2
说明	R3:RA 定义	包括指令、数据和校验	0x81, 0x83	寄存器地址/变量存储地址	寄存器字节长/变量字长	-	R2.4 决定是否启用
举例	0xA5,0x5A	0x07	0x81	0x03	0x02	0x00,0x01	0xB9,0x9A

指令帧头为双字节，其内容用户可以自定义，通过VGUS 4.3开发工具中“屏参配置”设置。用户可以通过设定不同的帧头，实现多台VGUS屏的并联应用。

CRC校验码也是双字节，用户可以自定义是否启用CRC检验，通过VGUS 4.3开发工具中“屏参配置”设置。CRC校验不包括帧头和数据长度，仅包括指令帧中的指令和数据，采用ANSI-16（ $X16+X15+X2+1$ ）格式。

当同时启用CRC校验和CRC校验结果应答后（通过VGUS 4.3开发工具中“屏参配置”设置），VGUS屏会在CRC校验后自动通过串口外发校验结果。CRC检验结果应答帧结构如表2-2所示。

表2-2 CRC校验结果应答帧结构

定义	帧头	数据长度	指令	数据	CRC 校验码
长度（字节）	2	1	1	1	2
说明	用户自定义	固定为 0x02	接收到的指令	0xff 表示 CRC 校验正确 0x00 表示 CRC 校验错误	
举例	0xA5,0x5A	0x02	0x81	0xff	0x21,0xA0

VGUS屏串口指令缓冲区有4K字节，一个VGUS刷新周期（80/120/160/200ms）内可以传送至少4KB数据（约等于230400~691200bps波特率连续发送）。



2.2 指令集

VGUS 屏采用变量驱动模式工作，屏的工作模式和GUI状态完全由变量和寄存器来控制。相应的，串口指令也只需要对变量和寄存器进行读、写即可，一共5条指令。

表2-3 VGUS屏指令集

功能	指令	数据	说明
访问寄存器接口	0x80	下发：寄存器地址（0x00-0xFF）+写入数据	指定地址写寄存器数据
	0x81	下发：寄存器地址（0x00-0xFF）+读取字节长度（0x00-0xFF）	指定地址开始读指定字节长度的寄存器数据
		应答：寄存器地址（0x00-0xFF）+字节数据长度+读取的寄存器数据	读寄存器的 VGUS 屏应答
	VGUS 屏有 256Byte 的寄存器，主要用于相关硬件控制操作，按照字节（Byte）寻址。		
访问变量存储器接口	0x82	下发：变量地址（0x0000-0xFFFF）+写入的变量数据	指定变量地址开始写入数据（字数据）到变量存储区
	0x83	下发：变量地址（0x0000-0xFFFF）+读取变量数据字长度（0x00-0x7F）	从变量存储区指定地址开始读入指定字长度的数据
		应答：变量存储器地址+变量数据长度+读取的变量数据	读数据存储器的 VGUS 应答
	<p>VGUS 屏采用变量驱动方式，将变量数值和变量显示格式分开。变量显示格式是预先通过配置文件形式下载在显示终端里。而变量数值是通过串口实时传送给显示终端的，变量存储器就是用来存储接收到的变量数值。</p> <p>VGUS 屏有 64K word（128K Byte）的变量存储器，按照字（word）寻址操作，地址为 0x0000-0xFFFF。用户在规划变量时，要根据变量长度手工分配变量存储器地址。</p>		
写曲线缓冲区接口	0x84	CH_Mode（Byte）+DATA0（Word）+...+DATA _n	<p>写曲线缓冲区数据。</p> <p>CH_Mode定义了后续数据的通道排列顺序： CH_Mode 的每个位（bit）对应 1 个通道； CH_Mode.0 对应 0 通道,.7 对应 7 通道； 对应位置 1 表示对应的通道数据存在； 对应位置 0 表示对应的通道数据不在。</p> <p>数据按照低通道数据在前排列。</p> <p>比如 CH_Mode=0x83（10000011B），表示后续数据格式为：（通道0+通道1+通道7）+...+（通道 0+通道1+通道 7）。</p>
	<p>VGUS 屏有一个 8K Word、可以存储 8 条曲线的缓冲区，用于用户简单、快速显示曲线。</p> <p>曲线缓冲区的数据都是 16 位无符号数。</p>		
写显存接口	0x85	00+X（Word）+ Y（Word）+DATA0（Word）+...+DATA _n	<p>写显存数据。</p> <p>00 表示写显存指令； X 表示起始位置的 X 坐标，Y 表示起始位置的 Y 坐标； 数据内容为每一个像素点需要写入的颜色值。</p> <p>写入的数据超过一行的最大像素点后自动换行，显示内容不满一行时需要自行计算每行的起点位置及数据长度。</p>



		01 5a a5+filesize(4bytes)+filename(ASCII)	<p>串口下载指定文件。</p> <p>01 5a a5表示串口下载文件指令；</p> <p>filesize表示文件字节大小；</p> <p>filename表示下载文件的文件名（完整的文件名，含扩展名，如文件名为“1.jpg”，发送的ASCII码应为31 2e 6a 70 67）。</p> <p>发送指令正确后屏返回提示“Please Tx file !”</p> <p>保存完成后屏返回提示“One file Saved OK !”</p> <p>注：连接USB线时用串口下载文件会导致文件出错。</p>
顺序播放音频文件接口	0x85	03+Mode+NUM0+...+NUMn	<p>顺序播放音频文件。</p> <p>03表示顺序播放音频文件指令；</p> <p>Mode定义了播放音频文件的模式：</p> <p>0表示循环播放，1表示顺序播放，其他表示停止播放；</p> <p>NUM表示音频播放设定值（2字节，取值范围0x0000-0xFFFE），如果设为不存在的音频文件设定值将直接略过该音频文件</p>
	<p>为了提高播放效果，指令中音频文件格式必须一致，必须同时为WAV或MP3格式，WAV文件建议选用双声道数据，列表文件总大小不得超过2M，MP3格式总大小没有限制。</p>		

2.3 CRC 校验程序

```
uint8_t uchCRCHi;    //CRC 高字节
```

```
uint8_t uchCRCLo;    //CRC 低字节
```

```
static uint8_t auchCRCHi[] = { /* CRC 高位字节值表*/
```

[illegible]

```
static uint8_t auchCRCLo[] = { /* CRC 低位字节值表*/
```

0x00, 0xC0, 0xC1, 0x01, 0xC3, 0x03, 0x02, 0xC2, 0xC6, 0x06, 0x07, 0xC7, 0x05, 0xC5, 0xC4, 0x04, 0xCC, 0x0C, 0x0D, 0xCD, 0x0F, 0xCF, 0xCE, 0x0E, 0x0A, 0xCA, 0xCB, 0x0B, 0xC9, 0x09, 0x08, 0xC8, 0xD8, 0x18, 0x19, 0xD9, 0x1B, 0xDB, 0xDA, 0x1A, 0x1E, 0xDE, 0xDF, 0x1F, 0xDD, 0x1D, 0x1C, 0xDC, 0x14, 0xD4, 0xD5, 0x15, 0xD7, 0x17, 0x16, 0xD6, 0xD2, 0x12, 0x13, 0xD3,



0x11, 0xD1, 0xD0, 0x10, 0xF0, 0x30, 0x31, 0xF1, 0x33, 0xF3, 0xF2, 0x32, 0x36, 0xF6, 0xF7, 0x37, 0xF5, 0x35, 0x34, 0xF4, 0x3C, 0xFC, 0xFD, 0x3D, 0xFF, 0x3F, 0x3E, 0xFE, 0xFA, 0x3A, 0x3B, 0xFB, 0x39, 0xF9, 0xF8, 0x38, 0x28, 0xE8, 0xE9, 0x29, 0xEB, 0x2B, 0x2A, 0xEA, 0xEE, 0x2E, 0x2F, 0xEF, 0x2D, 0xED, 0xEC, 0x2C, 0xE4, 0x24, 0x25, 0xE5, 0x27, 0xE7, 0xE6, 0x26, 0x22, 0xE2, 0xE3, 0x23, 0xE1, 0x21, 0x20, 0xE0, 0xA0, 0x60, 0x61, 0xA1, 0x63, 0xA3, 0xA2, 0x62, 0x66, 0xA6, 0xA7, 0x67, 0xA5, 0x65, 0x64, 0xA4, 0x6C, 0xAC, 0xAD, 0x6D, 0xAF, 0x6F, 0x6E, 0xAE, 0xAA, 0x6A, 0x6B, 0xAB, 0x69, 0xA9, 0xA8, 0x68, 0x78, 0xB8, 0xB9, 0x79, 0xBB, 0x7B, 0x7A, 0xBA, 0xBE, 0x7E, 0x7F, 0xBF, 0x7D, 0xBD, 0xBC, 0x7C, 0xB4, 0x74, 0x75, 0xB5, 0x77, 0xB7, 0xB6, 0x76, 0x72, 0xB2, 0xB3, 0x73, 0xB1, 0x71, 0x70, 0xB0, 0x50, 0x90, 0x91, 0x51, 0x93, 0x53, 0x52, 0x92, 0x96, 0x56, 0x57, 0x97, 0x55, 0x95, 0x94, 0x54, 0x9C, 0x5C, 0x5D, 0x9D, 0x5F, 0x9F, 0x9E, 0x5E, 0x5A, 0x9A, 0x9B, 0x5B, 0x99, 0x59, 0x58, 0x98, 0x88, 0x48, 0x49, 0x89, 0x4B, 0x8B, 0x8A, 0x4A, 0x4E, 0x8E, 0x8F, 0x4F, 0x8D, 0x4D, 0x4C, 0x8C, 0x44, 0x84, 0x85, 0x45, 0x87, 0x47, 0x46, 0x86, 0x82, 0x42, 0x43, 0x83, 0x41, 0x81, 0x80, 0x40} ;

```
uint16_t CRC16(uint8_t * pMsg, uint16_t Len)
```

```
{
    uint8_t i;
    uchCRCHi = 0xFF;
    uchCRCLo = 0xFF;
    while (Len--)
    {
        i = uchCRCHi ^ (*pMsg++);
        uchCRCHi = uchCRCLo ^ uchCRCHi[i];
        uchCRCLo = uchCRCLo[i];
    }
    return (uchCRCHi << 8 | uchCRCLo);
}
```



3 寄存器

VGUS 屏内部都设计有 256 字节的寄存器，用于实现硬件相关操作，如背光调整、时钟读写、语音播放等。

3.1 寄存器一览表

表 3-1 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	VGUS 版本号，BCD 码表示，0x40 表示 V4.0	
0x01	LED_NOW	R/W	1	LED 亮度控制寄存器，0x00-0x40 最高位设置 1 时表示亮度掉电保存	
0x02	BZ_TIME	W	1	蜂鸣器鸣叫控制寄存器，单位 10ms	
0x03-0x04	PIC_ID	R/W	2	读：当前显示页面 ID；写：切换到指定页面	
0x05	TP_Flag	R/W	1	0x5A 表示触摸屏坐标有更新；其他表示触摸屏坐标未更新	
0x06	TP_Status	R	1	0x01 表示第一次按下；0x03 表示一直按下中；0x02 表示抬起；其他表示无效	
0x07-0x0A	TP_Position	R	4	触摸屏按压坐标位置 X_H:L Y_H:L	
0x0B	TPC_Enable	R/W	1	0x00 表示触控不启用；其他表示触控启用（上电默认 0xFF）	
0x0C-0x0F	RUN_TIME	R	4	上电后运行时间，BCD 码时分秒，其中小时为两个字节，最大 9999:59:59	
0x10-0x1C	RO-RC	R/W	13	屏参配置寄存器映射，当 1D 寄存器配置时，11-1C 寄存器可改写 R1-RC 的数据	
0x1D	CONFIG_EN	W	1	写入 0x5A，R1-RC 重新设置并保存；写入 0xA5，只重新设置，不保存	
0x1F-0x26	RTC_COM_ADJ	W	1	0x5A 表示用户串口申请改写 RTC 数据，VGUS 修改 RTC 后清零	
	RTC_NOW	R/W	7	YY: MM:DD:WW:HH:MM:SS, 年:月:日:星期:时:分:秒，格式为 BCD 码	
0x27-0x3F	保留		16	未定义	
0x40-0x49	En_Lib_OP	R/W	1	0x5A 表示申请用户数据文件加载到变量空间，VGUS 操作完后清零。每个 VGUS 周期执行一次读操作	
	Lib_OP_Mode	W	1	固定为 0xA0	
	Lib_ID	W	1	指定用户数据文件名 0x00-0xff	
	Lib_Address	W	3	指定数据操作首（字）地址，0x00:00:00-0x01:FF:FF	
	VP	W	2	指定变量空间的数据操作（首）地址 0x0000-0xFFFF	
	OP_Length	W	2	操作的（字）数据长度，0x0001-0xFFFF	
0x4A-0x4b	Timer0	R/W	2	16bit 软件定时器, 单位 4ms, 自减到零停止	设置值和实际运行值之间 有 +/-4ms 误差，比如设置 位 2, 实际运行值在 4-12ms 之间
0x4C	Timer1	R/W	1	8bit 软件定时器, 单位 4ms, 自减到零停止	
0x4D	Timer2	R/W	1	8bit 软件定时器, 单位 4ms, 自减到零停止	
0x4E	Timer3	R/W	1	8bit 软件定时器, 单位 4ms, 自减到零停止	
0x4F	Key_code	W	1	用户键码，用于触发配置文件，0x01-0xFF，0x00 表示无效 VGUS 处理键码后会自动清零键码寄存器	



0x50-0x52	Music_Set	W	1	0x5b:播放语音 0x5c:停止播放
	Music_Num	W	2	音频播放设定值（0-4095）
0x53-0x54	Vol_Adj_En	W	1	0x5A 表示申请调整播放音乐的音量
	Vol	W	1	音量值，范围 0x00-0x40，上电默认值是 0x40
0x55	Vol_Status	R	1	音频播放状态 0x00：播放停止；0x01：播放中
0x56-0x5F	En_DBL_OP	R/W	1	0x5A 表示用户申请进行数据库操作，VGUS 操作完后清零 每个 VGUS 周期执行一次数据库读或写操作
	OP_Mode	W	1	0x50 表示把变量存储器空间数据写入数据库空间 0xA0 表示把数据库空间的数据读入变量存储器空间
	DBL_Address	W	4	数据库首（字）地址（数据库和用户的“VT_SET”文件夹均存放在“VGUS_USER”盘符下）
	VP	W	2	变量空间的首（字）地址，0x0000-0xFFFF
	OP_Length	W	2	数据库操作的（字）长度，0x0001-0xFFFF
0x60-0x67	Play_Avi_Set	R/W	1	0x5a:申请播放 avi 视频，VGUS 播放完后清零
	Avi_Type	W	1	0x00：单曲播放 VGUS 屏内视屏（默认模式） 0x01：单曲播放 U 盘内视频 0x02：顺序播放 VGUS 屏内视屏 0x03：顺序播放 U 盘内视频 注：视频文件扩展名必须是*.avi； 单曲播放时文件名必须为阿拉伯数字, 如“123.avi”； 顺序播放时文件名可以为字母+数字, 如“wuhan123.avi”。
	Play_Position	W	4	视频窗口左上角坐标位置（X_H, X_L, Y_H, Y_L） 注：(0,0)表示居中显示。
	Play_Avi_Num	W	2	通过视频文件名选择播放视频曲目，最多允许 65536 个视频； 仅用于单曲播放，顺序播放时无效。
0x68-0x69	Vol_Adj_En	W	1	0x5A 表示申请调整播放视频音量
	Vol	W	1	播放视频音量值，范围 0x00-0x3F，上电默认值是 0x3F。
0x6a	Play_Control	W	1	0x5A：播放/暂停
0x6b	Play_Stop	W	1	0x5A：停止
0x6c	Play_Next	W	1	0x5a:播放下一首 (VGUS 处理后清零，单曲播放时结束)
0x6d	Play_Status	R	1	0x00=空闲 0x01=播放中 0x02=暂停
0x6e-0xE8	保留		137	未定义
0xE9	Scan_Status	R/W	1	读：0x00 表示触摸屏未处于录入状态 0x01 表示触摸屏录入状态； 写：如果触摸屏在录入状态下写 0x00，强制退出录入状态。
0xEA	TPCal_Triger	W	1	写入 0x5A 启动一次触摸屏校准，校准完成后会被 VGUS 清零。



0xEB	Trendline_Clear	W	1	0x55 表示清除全部 8 条曲线缓冲区数据； 0x56-0x5D 表示分别清除 CH0-CH7 通道的曲线缓冲区数据。 曲线缓冲区数据清除后，本寄存器会被 VGUS 清零。
0xEC-0xED	保留		2	保留
0xEE-0xEF	Reset_Triger	W	2	写入 0x5AA5 导致 VGUS 屏软件复位一次
0xF0-0xFF	保留		16	保留

3.1.1 读取版本号

寄存器地址	定义	R/W	字节长度	说明
0x00	Version	R	1	VGUS 版本号，BCD 码表示，0x40 表示 V4.0

读取版本信息，串口下发指令 A5 5A 03 81 00 01

返回 A5 5A 04 81 00 01 43

说明：A5 5A：帧头

04：指令的字节长度，81 00 01 43 共4字节（不含帧头）

81：读寄存器指令

00：寄存器地址

01：返回数据的字节长度，43为1字节

43：返回的数据信息

3.1.2 修改背光亮度

寄存器地址	定义	R/W	字节长度	说明
0x01	LED_NOW	R/W	1	LED 亮度控制寄存器，0x00-0x40 最高位设置 1 时表示亮度掉电保存

例如，关闭背光

A5 5A 03 80 01 00

建议：1、设备对功耗要求较高时，可以通过修改屏的背光亮度达到降低功耗的目的；

2、如有需要单片机控制达到屏保效果的情况，可用该指令控制背光亮灭从而达到屏保的效果。

3.1.3 控制蜂鸣器

寄存器地址	定义	R/W	字节长度	说明
0x02	BZ_TIME	W	1	蜂鸣器鸣叫控制寄存器，单位 10ms

例如，控制蜂鸣器鸣叫2s，发送指令

A5 5A 03 80 02 C8

3.1.4 切换界面

寄存器地址	定义	R/W	字节长度	说明
0x03-0x04	PIC_ID	R/W	2	读：当前显示页面 ID；写：切换到指定页面



例如，切换到2号界面的指令

A5 5A 04 80 03 00 02

3.2 时钟（RTC）读写

表 3-2 时钟读写寄存器

寄存器地址	定义	R/W	字节长度	说明
0x1F-0x26	RTC_COM_ADJ	W	1	0x5A 表示用户串口申请改写 RTC 数据，VGUS 修改 RTC 后清零
	RTC_NOW	R/W	7	YY: MM:DD:WW:HH:MM:SS, 年:月:日:星期:时:分:秒, 格式为 BCD 码

➤ 串口读取时钟RTC

0x20 寄存器开始保存了当前 RTC值，使用 0x81 指令读取。

读取日历（YY:MM:DD:WW:HH:MM:SS）：A5 5A 03 81 20 07

读取时间（HH:MM:SS）：A5 5A 03 81 24 03

➤ 串口修改（写/设置）时钟RTC

用 0x80 指令改写 0x1F寄存器为 0x5A，并给 0x20 开始的寄存器写入需要修订的时间，即改写了 RTC。

例如把RTC设置为2015-06-01星期一18:56:00，串口发送以下指令：

A5 5A 0A 80 1F 5A 15 06 01 00 18 56 00 VGUS屏会自动换算星期，改写时间时星期可以写任意值。

3.3 语音播放

表 3-3 语音播放寄存器

寄存器地址	定义	R/W	字节长度	说明
0x50-0x52	Music_Set	W	1	0x5b: 播放语音 0x5c: 停止播放
	Music_Num	W	2	音乐播放设定值（0-4095）
0x53-0x54	Vol_Adj_En	W	1	0x5A 表示申请调整播放音乐的音量
	Vol	W	1	音量值，范围 0x00-0x40，上电默认值是 0x40
0x55	Vol_Status	R	1	音频播放状态 0x00：播放停止；0x01：播放中

语音文件的扩展名为*.wav或*.mp3，WAV文件建议选用双声道数据，用户可通过U盘将语音文件下载到VGUS屏，用户可以通过0x80指令写相关寄存器控制语音播放和进行音量调节。

举例，一段语音（比如“欢迎光临武汉中显”）保存为6.wav，要以100%音量播放，串口下发：

A5 5A 07 80 50 5b 00 06 5A 40

要停止语音播放，串口下发：

A5 5A 05 80 50 5c 00 06

3.4 视频播放

SDW-PlusII 系列产品在不增加硬件成本，创新实现了串口屏播放视频的设计，满足开机动画、广告视



频、娱乐视频、设备维护视频等场合对播放视频功能的需求。

1. 开机动画：是展现设备档次、企业形象的有力手段。目前同行产品都是通过连续显示图片来实现动画，一方面因为受到存储图片张数限制，播放时长都有限，同时也不可能实现声音的同步播放。

2. 广告视频：如零售设备、自助售卖设备，在设备工作间隙可以兼备播放广告视频。

3. 娱乐视频：如健身器材、美容设备、医疗器械等，在设备工作同时可以播放娱乐影像视频。

4. 设备维护视频：如设备的日常保养、常见故障处理等，可以将视频嵌入到设备中，有力提高日常售后服务效率、降低售后服务成本。

视频文件的扩展名为*.avi，分辨率可小于或等于屏幕分辨率，允许最多 65536 个视频文件，视频编码为 mjpeg 格式，音频编码 mp3(采样频率 16kHz)；播放视频功能时显示屏上变量不再刷新、上一节中语音播放不可用。视频文件必须手动复制到“VT_SET”文件夹下。

表3-4 视频播放寄存器

寄存器地址	定义	R/W	字节长度	说明
0x60-0x67	Play_Avi_Set	R/W	1	0x5A: 申请设置播放器参数
	Avi_Type	W	1	0x00: 单曲播放 VGUS 屏内视屏（默认模式） 0x01: 单曲循环播放 VGUS 屏内视屏 0x02: 顺序循环播放 VGUS 屏内视屏 0x03: 单曲播放 U 盘内视频 0x04: 单曲循环播放 U 盘内视频 0x05: 顺序循环播放 U 盘内视频 注：视频文件扩展名必须是*.avi； 单曲播放时文件名必须为阿拉伯数字，如“123.avi”； 顺序播放时文件名可以为字母+数字，如“wuhan123.avi”。
	Play_Position	W	4	视频窗口左上角坐标位置（XH, XL, YH, YL） 注：(0, 0, 0, 0)表示居中显示。
0x68-0x69	Play_Avi_Num	W	2	通过视频文件名选择播放视频曲目，最多允许 65536 个视频； 仅用于单曲播放，顺序播放时无效。
	Vol_Adj_En	W	1	0x5A: 申请调整播放视频音量
0x6a	Vol	W	1	播放视频音量值，范围 0x00-0x3F，上电默认值是 0x3F。
	Play_Control	W	1	0x5A: 播放/暂停 对于单曲播放方式，当播放完当前视频后，系统自动跳回到当前图片界面。
0x6b	Play_Stop	W	1	0x5A: 停止 执行停止播放视频后，系统自动跳回到当前图片界面，也可以按照按钮跳转。
0x6c	Play_Next	W	1	0x5A: 播放下一首
0x6d	Play-Prev	W	1	0x5A: 播放前一首
0x6e	Play_Status	R	1	0x00=空闲； 0x01=播放中； 0x02=暂停。



用户寄存器 0x61-0x6e 映射于用户变量存储器 0xff01-0xff0e,即操作用户变量存储器 0xff01-0xff0e 可实现相同的功能（主要是用来触控实现播放 avi 视频，无须用户 MCU 指令干预）。

可以通过用户指令控制和触控两种方式，来控制视频播放。

方式一：用户指令控制方式。用户 MCU 通过 0x80 指令写寄存器 0x60-0x6d，实现 avi 视频的播放、暂停、继续、停止等功能；

方式二：触控方式。在播放器界面，用户制作播放/暂停、停止、下一曲、音量调节等按钮，通过“按钮键值返回”功能修改变量存储器 0xff01-0xff0e，实现无需用户 MCU 干预下播放 avi 视屏；

小技巧，实现开机默认播放开机动画。通过 VGUS4.3 组态软件将 0xff0a 单元初值设置为 0x5a，将开机动画视频文件名设置为 0.avi，串口屏上电后将自动播放开机动画 0.avi。

具体方法为设置一个数据变量，地址设为 0xff0a，初始值设为 90。

视频文件在使用前，请先按照 avi 文件格式要求，采用“格式工厂”软件转换，然后再拷贝到“VT_SET”文件夹下。

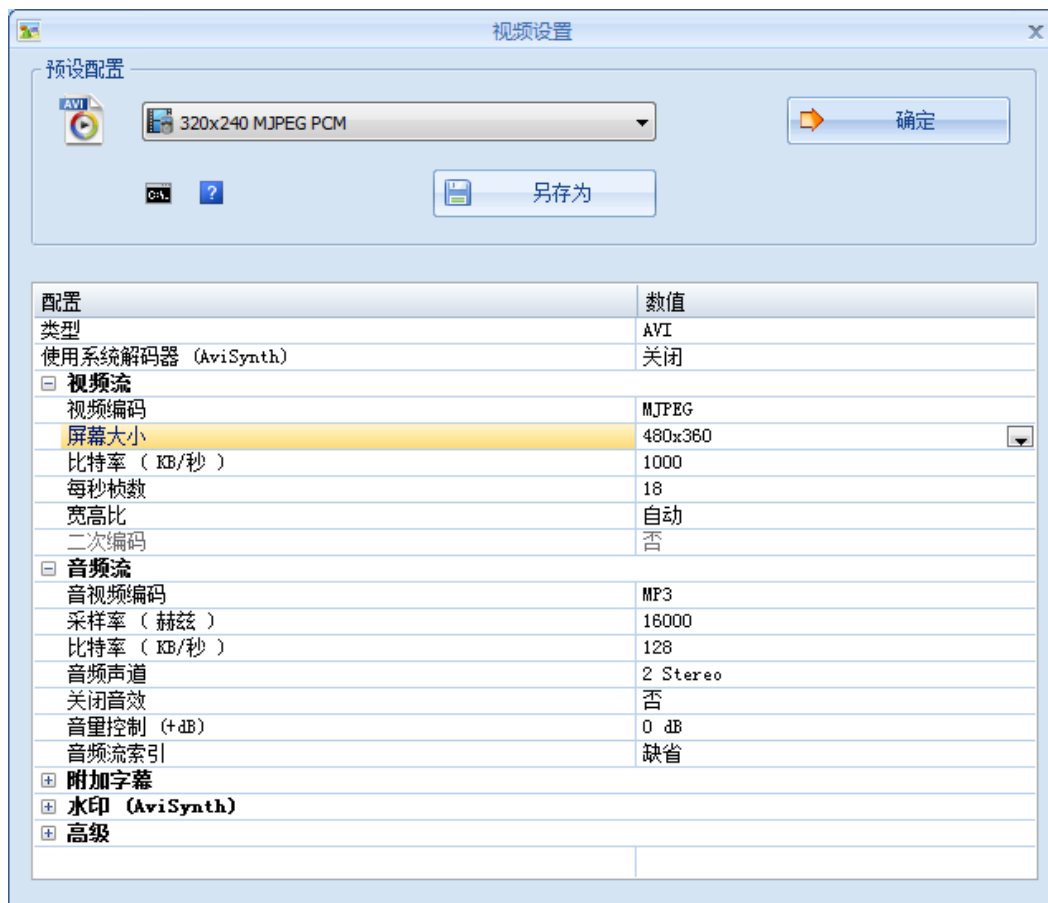


图3-1 音视频文件格式编辑软件“格式工厂”

举例，在显示屏（10，10）位置单曲播放 U 盘内 01 号视频：

首先修改播放参数 A5 5A 0A 80 60 5A 03 00 0A 00 0A 00 01，然后发送播发视频的指令 A5 5A 03 80 6A

5A

3.5 用户数据文件加载到变量空间

表 3-5 用户数据文件加载到变量空间寄存器

寄存器地址	定义	R/W	字节长度	说明
0x40-0x49	En_Lib_OP	R/W	1	0x5A 表示申请用户数据文件加载到变量空间, VGUS 操作完后清零。每个 VGUS 周期执行一次读操作
	Lib_OP_Mode	W	1	固定为 0xA0
	Lib_ID	W	1	指定用户数据文件名 0x00-0xff,
	Lib_Address	W	3	指定数据操作首(字)地址, 0x00:00:00-0x01:FF:FF
	VP	W	2	指定变量空间的数据操作(首)地址 0x0000-0xFFFF
	OP_Length	W	2	操作的(字)数据长度, 0x0001-0xFFFF

用户数据文件扩展名为*.bin, 编号为0x00-0xff。可以通过串口指令操作, 把用户数据文件加载到变量存储器中(如果用户系统需要使用, 可以使用0x82指令再从变量存储器中读取)。用户数据文件必须手动复制到“VT_SET”文件夹下。

例如从“80.bin”用户数据文件的 0x 00 00 00 地址开始读取4KW (0x10 00) 数据到变量存储器 0x1000开始的位置, 串口下发指令:

A5 5A 0C 80 40 5A A0 50 00 00 00 10 00 10 00

3.6 数据库读写

用户数据库的文件名为“UserDb.bin”, 可随机读写, 且具有掉电存储用户数据的功能,

“UserDb.bin”和“VT_SET”文件夹均存放在“VGUS_USER”盘符下, 数据库文件的大小与用户工程文件有关, 可查看“VGUS_USER”的剩余空间, 确定可用数据库文件的大小。

表3-6 数据库读写寄存器

寄存器地址	定义	R/W	字节长度	说明
0x56-0x5F	En_DBL_OP	R/W	1	0x5A 表示用户申请进行数据库操作, VGUS 操作完后清零 每个 VGUS 周期执行一次数据库读或写操作
	OP_Mode	W	1	0x50 表示把变量存储器空间数据写入数据库空间 0xA0 表示把数据库空间的数据读入变量存储器空间
	DBL_Address	W	4	数据库首(字)地址(数据库和用户的“VT_SET”文件夹均存放在“VGUS_USER”盘符下)
	VP	W	2	变量空间的首(字)地址, 0x0000-0xFFFF
	OP_Length	W	2	数据库操作的(字)长度, 0x0001-0xFFFF

例如把变量地址 0000-0100 的数据写入数据库地址 0000 0000-0000 0100, 数据长度为 256W(0x01 00), 串口下发指令:

A5 5A 0C 80 56 5A 50 00 00 00 00 00 01 00

把数据库地址 0000 0000-0000 0100 的数据写入变量地址 0000-0100, 数据长度为 256W (0x01 00), 串口下发指令:

A5 5A 0C 80 56 5A A0 00 00 00 00 00 01 00



➤ 数据库空间数据导出

数据库内存储的数据可以通过串口或者USB口导出。

✧ 使用U盘导出数据

VGUS屏工作状态下插入U盘（Fat32格式），VGUS屏会自动将数据库文件“UserDb.bin”拷贝到U盘根目录下。

✧ 使用串口导出数据

首先通过寄存器0x56-0x5F将数据读入变量存储器空间，然后再使用0x83指令从变量存储器中读取。

3.7 触摸屏校准

所有 VGUS 屏出厂前均已经校准，用户使用前无需逐个再次校准。VGUS 屏提供了两种启动触摸屏校准方法，分别适用于不同应用场合。

➤ 方法1——产品集成

用户 MCU 通过串口向 VGUS 屏 0xEA 寄存器写入 0x5A 将启动一次触摸屏校准。通过该方法，可以将触摸屏校准功能作为最终设备的一个选项功能，提供给设备操作人员。该方法适用于用户 MCU 发串口指令使用。

表 3-7 触摸屏校准寄存器

寄存器地址	定义	R/W	字节长度	说明
0xEA	TPCal_Triger	W	1	写入 0x5A 启动一次触摸屏校准，校准完成后会被 VGUS 清零

触摸屏校准发送指令A5 5A 03 80 EA 5A

➤ 方法2——批量生产

在 VGUS 4.3 开发工具“屏参配置”中，勾选“U 盘下载配置文件后启动触摸校准”，则在每次下载配置文件完毕后，VGUS 屏将立即自动启动一次触摸屏校准。如果客户希望在生产环节，每块屏都再次校准一次，可以使用该方法，该方法适用于批量生产阶段使用。所有 VGUS 屏出厂都已经进行校准，不建议客户再次校准。

无论使用上述哪种方法，一旦触摸屏校准启动后，需要按照屏幕提示操作，依次点击屏幕“左上角”、“左下角”、“右下角”、“右上角”和“中心点”十字交叉点提示的触摸位置；当校准完成时，显示终端会自动进入触摸测试状态，点击触摸屏可观察触摸位置准确度。

3.8 屏参配置读取

对于简单应用场合，用户无需了解该节知识。

屏参配置是通过 VGUS 4.3 开发工具设置实现，如图 1-4 所示。屏参配置中设置了串口波特率、指令帧头、CRC 校验、及蜂鸣器、触摸屏和屏保等外设特性。用户可以用串口通过 0x10-0x1C 寄存器读取相关设置值。

表 3-8 屏参配置寄存器



寄存器地址	定义	R/W	字节长度	说明
0x10-0x1C	R0-RC	R	13	VGUS 开发工具“屏参配置”的映射寄存器，串口只读，写无效。

3.8.1 串口波特率设置 (R1/0x10)

当 R1 取值在 00-10 时, R5、R9 无效, 可以选择 17 档固定波特率之一, 如下表(波特率单位为 bps):

表 3-9 串口波特率设置寄存器

R1 (0x)	00	01	02	03	04	05	06	07	
波特率 (bps)	1200	2400	4800	9600	19200	38400	57600	115200	
R1 (0x)	08	09	0a	0b	0c	0d	0e	0f	10
波特率 (bps)	28800	76800	62500	125000	250000	230400	345600	691200	921600

当 R1 取值为 11 时, 此时波特率由 R5、R9 决定, 并由下式计算:

$R5:R9 = 6250000 / \text{波特率}$ R5:R9 表示一个双字节参数, R5 为高字节, R9 为低字节。

比如, 设定 10000bps 波特率, $R5:R9 = 6250000 / 10000 = 625 = 0x0271$ R5=02 R9=71

VGUS 屏出厂波特率为 115200bps。

3.8.2 串口指令帧头设置 (R3/0x13、RA/0x1A)

关于指令帧头的详细信息见表2-1, 指令帧头的设置主要达到以下两个目的:

- (1) 用于串口指令帧的识别和同步;
- (2) 多台VGUS屏并联工作时, 把帧头作为设备地址加以区分。

VGUS 屏出厂帧头为 A5 5A。

3.8.3 工作模式配置寄存器1（R2/0x12）

R2寄存器按位（bit）定义，用于配置VGUS屏软件工作模式，如下表所示（阴影部分表示出厂设置值）。

表3-10 寄存器R2位功能定义

位	权重	定义	说明
.7	0x80	VDS	未定义
.6	0x40	HDS	未定义
.5	0x20	TP_LED	0=禁止屏保功能，背光不受触摸屏状态控制 1=启用屏保功能，背光受触摸屏状态控制，控制参数由R6、R7、R8寄存器设定 R6为上电启动时的显示亮度，用户可在工程中设置启动亮度
.4	0x10	FCRC	0=禁止串口通信的CRC16帧校验 1=启用串口通信的CRC16帧校验
.3	0x08	TPSAUTO	0=触摸屏录入参数后不自动上传（用户查询） 1=触摸屏录入参数后是否自动上传到串口由相应触控变量的配置决定
.2	0x04	变量初值	0=128KB变量存储器上电初始化为0x00 1=128KB变量存储器上电初值由配置文件决定
.1	0x02	FRS1	设置VGUS周期，VGUS周期越小则变量响应越灵敏，但处理变量的能力越低。
.0	0x01	FRS0	对于1024*768分辨率，建议VGUS周期设置成120ms以上。 VGUS周期会影响动画图标显示的动画速度。

VGUS周期	80ms	120ms	160ms	200ms
FRS1	1	1	0	0
FRS0	1	0	1	0

3.8.4 工作模式配置寄存器2（RC/0x1C）

RC寄存器（AUX_CFG配置字）按位（bit）定义，用于配置VGUS屏软件工作模式，如下表所示（阴影部分表示出厂设置值）。

表3-11 寄存器RC位功能定义

位	权重	定义	说明
.7	0x80	系统保留	必须写0

. 6	0x40	未定义	写0
. 5	0x20	TP_BUZZ_EN	0=点击触摸屏有效区域时有蜂鸣器提示音 1=点击触摸屏有效区域时无蜂鸣器提示音，但可以通过向0x02寄存器写入数据控制蜂鸣器鸣叫
. 4	0x10	PAGE128_EN	0=每页显示变量数目为64个 1= 未定义
. 3	0x08	CRC_ACK_EN	0=启动CRC帧校验后，不应答帧校验结果 1=启动CRC帧校验后，应答帧校验结果
. 2	0x04	TP_CAL_MOD	未定义
. 1	0x02	未定义	写0
. 0	0x00	未定义	写0

3.8.5 屏保/触摸屏控制背光（R2.5、R6、R7、R8）

当设置R2.5=1时，背光亮度将受触摸屏状态控制（背光待机后，第一次点击触摸屏不会触发动作）

表3-12 屏保功能寄存器

R#	取值范围	说明
R6	0x00~0x40	触摸屏控制背光启动后，点击触摸屏后背光点亮的亮度。（屏保工作亮度）
R7	0x00~0x40	触摸屏控制背光启动后，一段时间不点击触摸屏，背光关闭的亮度。（屏保保护亮度）
R8	0x01~0xFF	触摸屏控制背光启动后，触摸屏背光点亮时间，单位为 1.0 秒。（屏保延时时间）

举例，设置 R2.5=1，R6=0x40，R7=0x10，R8=0x1E，30 秒（0x1E）不点击触摸屏，背光亮度将自动降低到 0x10（25%亮度）；点击触摸屏后，背光亮度将自动调节到 0x40（100%亮度）。

屏参配置

×

串口参数配置

R1(串口波特率) 115200
R3(串口帧高字节) 0x A5 (0x00~0xFF)
RA(串口帧低字节) 0x 5A (0x00~0xFF)

自定义波特率:
R5(高字节) 0x
R9(低字节) 0x
☐ 启用串口通讯CRC16帧校验
☐ 启用CRC16校验结果应答

硬件参数配置

变量初值

☒ 128KB变量存储器上电初始化为0x00
☐ 变量初始值由配置文件决定

运行周期

☐ 200ms
☐ 160ms
☒ 120ms
☐ 80ms

☒ U盘下载配置文件后启动触摸校准
☒ 触屏录入参数后自动上传到串口
☐ 关闭蜂鸣器
开机页面 35

触控背光设置/启动屏保

☐ 背光受触摸屏状态控制/启动屏保

R6(点亮亮度) 0x 40 (0x00~0x40)
R7(屏保亮度) 0x 10 (0x00~0x40)
R8(延时时间X1.0秒) 0x 0A (0x00~0xFF)

确定

取消

4 输入控件

表 4-1 输入控件一览表

序号	功能	说明
00	变量数据录入	录入整数、定点小数等各种数据到指定变量存储空间。
01	弹出菜单选择	点击触发一个弹出菜单，返回菜单项的键值。
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 范围循环调节可以实现栏目复选框功能
03	拖动调节	拖拉滑块实现变量数据录入，可设置刻度范围。
04	RTC设置	VGUS 屏触摸键盘设置 RTC 组件，需要完整录入公历年月日时分秒
05	按钮键值返回	点击按钮，直接返回键值到变量，支持位变量返回
06	文本录入	文本方式录入各种字符，录入过程支持光标移动、编辑。 直接支持 ASCII 字符、GBK 中文、繁体注音输入法录入； 修改字库和 0号字库可以支持所有类似 ASCII 字符的 8bit 编码文本录入；
07	硬件参数配置	提供了触摸屏改写寄存器空间的方法，来间接控制硬件。 比如把背光寄存器内容读取到变量，调节变量后再回写来调节背光亮度
		点击触摸屏，把指定VP区域的数据发送到用户串口（COM1）。
08	按压数据同步返回	点击触摸屏，按照规定返回到变量或串口。
09	转动调节	转动旋钮实现变量数据录入，可设置刻度范围。

4.1 变量数据录入

按钮属性	
名称定义	数据录入
数据自动上传	<input checked="" type="checkbox"/>
按钮效果	3
页面切换	无
音频文件	无
变量反色显示	<input type="checkbox"/>
变量属性	
变量存储地址(0x)	0020
变量类型	整数(字)
整数位数	4
小数位数	0
显示位置	497,172
文本颜色	0; 0; 0
字库位置	0
字体大小	16
光标颜色	黑色
输入显示方式	正常显示
启用范围限制	<input type="checkbox"/>
键盘属性	
键盘在当前页面	<input type="checkbox"/>
键盘设置	Click to set
所在页面	6
键盘区域	(345,89) (638,370)
显示位置	221,164



例如：设置录入数据整数位数为4位，点击“数据录入”按钮输入“1234”，点击“OK”，串口返回

指令：

A5 5A 06 83 00 20 01 04 D2

说明：A5 5A：帧头

06：指令字节长度，83 00 20 01 04 D2 共6字节（不含帧头）

83：读变量寄存器指令

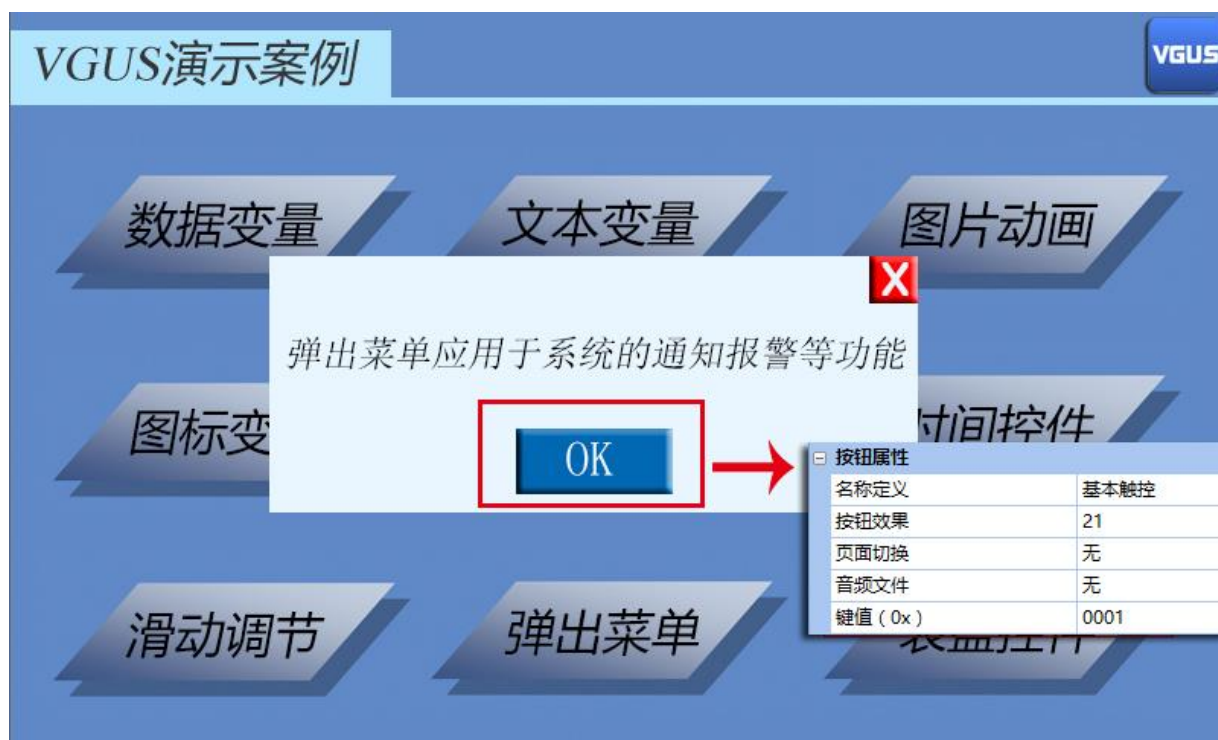
00 20：变量地址，在控件属性中设置

01：数据字长度，04 D2 共1字长

04 D2：数据内容，1234 的十六进制数据

4.2 弹出菜单选择

按钮属性	
名称定义	弹出菜单
数据自动上传	<input checked="" type="checkbox"/>
按钮效果	1
音频文件	无
变量属性	
变量存储地址(0x)	0200
变量模式	按字调节变量
菜单属性	
弹出菜单设置	Click to set
所在页面	20
菜单区域	(196,157) (622,326)
显示位置	170,160



例如：设置“弹出菜单”按钮变量地址为 0200，在弹出菜单页面上设置“OK”按钮的键值为 0001。点击“弹出菜单”按钮会弹出设置好的菜单，点击菜单上的“OK”键，会将按钮的键值 0001 返回到地址 0200，串口返回指令：

A5 5A 06 83 02 00 01 00 01

说明：A5 5A：帧头

06：指令字节长度，83 02 00 01 00 01 共 6 字节（不含帧头）

83：读变量寄存器指令

02 00：变量地址，在弹出菜单控件属性中设置

01：数据字长度，00 01 共 1 字长

00 01：数据内容，设置的按钮键值 00 01

4.3 增量调节

按钮属性	
名称定义	增量调节
数据自动上传	<input checked="" type="checkbox"/>
按钮效果	13
音频文件	无
变量属性	
变量存储地址(0x)	0120
调节模式	按字调节变量
调节方式	++
逾限处理方式	停止
调节步长	1
下限	0
上限	9
按键模式	按键连续调节

例如：设置下限为 0，上限为 9，连续点击“增量调节”按钮，串口返回指令：

A5 5A 06 83 01 20 01 00 01

A5 5A 06 83 01 20 01 00 02

.....

A5 5A 06 83 01 20 01 00 08

A5 5A 06 83 01 20 01 00 09

说明：A5 5A：帧头

06：指令字节长度，83 01 20 01 00 01 共 6 字节（不含帧头）

83：读变量寄存器指令

01 20：变量地址，在控件属性中设置

01：数据字长度，00 01 共 1 字长

00 01：数据内容，从设置的下限变化到上限，即从 00 01 连续变化到 00 09

4.4 拖动调节

按钮属性	
名称定义	拖动调节
数据自动上传	<input checked="" type="checkbox"/>
音频文件	无
拖动调节属性	
变量存储地址(0x)	0180
数据返回格式	调节字地址
拖动方向	横向拖动
起始值	0
终止值	1000

例如：设置起始值为 0，终止值为 1000，拖动“拖动调节”按钮，串口返回指令：

A5 5A 06 83 01 80 01 00 00

.....



A5 5A 06 83 01 80 01 03 E8

说明：A5 5A：帧头

06：指令字节长度，83 01 80 01 00 00 共 6 字节（不含帧头）

83：读变量寄存器指令

01 80：变量地址，在控件属性中设置

01：数据字长度，00 00 共 1 字长

00 01：数据内容，从设置的起始值变化到终止值，即从 00 00 连续变化到 03 E8

4.5 RTC 设置

按钮属性	
名称定义	RTC
数据自动上传	<input checked="" type="checkbox"/>
按钮效果	15
音频文件	无
RTC变量属性	
显示位置	776,193
文本颜色	0; 0; 0
字库位置	0
字体大小	14
光标颜色	黑色
键盘属性	
键盘在当前页面	<input type="checkbox"/>
键盘设置	Click to set
所在页面	6
键盘区域	(344,89) (637,370)
显示位置	491,187

例如：点击“RTC 设置”按钮输入时间 16-10-26 16: 09: 00，串口返回指令：

A5 5A 0A 81 20 07 16 10 26 03 16 09 00

说明：A5 5A：帧头

0A：指令字节长度，81 20 07 16 10 26 03 16 09 00 共 10 字节（不含帧头）

81：读寄存器指令

20：控制 RTC 的寄存器地址

07：数据字节长度，16 10 26 03 16 09 00 共 7 字节长

16 10 26 03 16 09 00：时间变量的 BCD 码

4.6 按钮键值返回

按钮属性	
名称定义	按键返回
数据自动上传	<input checked="" type="checkbox"/>
按钮效果	23
页面切换	无
音频文件	无
键值 (0x)	0222
变量属性	
变量存储地址(0x)	0000
调节模式	按字调节变量



例如：点击“按钮键值返回”按钮，串口返回指令：

A5 5A 06 83 00 00 01 02 22

说明：A5 5A：帧头

06：指令字节长度，83 00 00 01 02 22 共 6 字节（不含帧头）

83：读变量寄存器指令

00 00：变量寄存器地址

01：数据字长度，02 22 共 1 字长

02 22：数据内容，设置的键值

4.7 文本录入

名称定义	GBK 录入
数据自动上传	<input checked="" type="checkbox"/>
按钮效果	11
页面切换	无
音频文件	无
变量属性	
变量存储地址(0x)	0140
文本长度	40
录入模式	重新录入
显示字库位置	66
文本点阵大小	24
文本颜色	0; 0; 0
拼音字库位置	66
拼音点阵大小	24
拼音文本颜色	255; 0; 0
拼音显示方式	上边
光标颜色	黑色
输入状态返回	<input checked="" type="checkbox"/>
录入显示区域	(342,154) (669,212)
拼音显示位置	71,154
显示间距	1
键盘属性	
键盘在当前页面	<input type="checkbox"/>
键盘设置	Click to set
所在页面	8
键盘区域	(109,103) (723,429)
显示位置	63,148



例如：点击“GBK 录入”按钮输入“武汉中显”，点击“Enter”按钮，串口返回指令：

A5 5A 0E 83 01 40 05 CE E4 BA BA D6 D0 CF D4 FF FF

说明：A5 5A：帧头

0E：指令字节长度，83 01 40 06 CE E4 BA BA D6 D0 CF D4 FF FF 共 14 字节（不含帧头）

83：读变量寄存器指令

01 40：变量地址，在控件属性中设置

05：数据字长度，CE E4 BA BA D6 D0 CF D4 FF FF 共 5 字长

CE E4 BA BA D6 D0 CF D4 FF FF：数据内容，“武汉中显”的汉字内码 CE E4 BA BA D6 D0 CF D4，最后 2 字节 FF FF 是录入结束的标志位，文本录入及 ASCII 录入时有结束标志位。



5 显示控件

表 5-1 描述指针控件一览表

序号	功能	说明
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。多用于精细的仪表板、进度条显示。
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。
02	滑块刻度显示	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。
03	艺术字变量显示	用 ICON 图标取代字库来显示变量数据。
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。
05	图标旋转显示	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，多用于指针仪表盘显示。
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。多用于开关状态显示，比如风机的运转（动画）、停止（静止图标）。
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。
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。
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
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。
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。
14	文本滚动显示	从右向左滚动文本显示
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。
21_01	绘图_置点	置点（x, y, color）
21_02	绘图_端点连线	端点连线（color, (x0, y0), ..., (xn, yn)）
21_03	绘图_矩形	显示矩形，颜色和位置、大小可控
21_04	绘图_矩形填充	填充指定的矩形区域，填充颜色和位置、大小可控
21_05	绘图_画圆	显示整圆弧，颜色和位置、大小可控
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
21_07	绘图_ICON图标显示	ICON 图标显示，图标库可以选择
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
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
21_0A	绘图_线段显示	根据变量数据连接线段，端点、颜色可控
21_0B	绘图_圆弧显示	显示圆弧，半径、颜色、起止角度可控
21_0C	绘图_字符显示	根据变量数据进行单个字符显示
21_0D	绘图_矩形区域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
21_0E	绘图_双色位图显示	变量存储器数据看成双色位图数据，0/1 对应颜色可指定，多用于自定义光标
21_0F	绘图_位图显示	变量存储器数据位 65K 色位图数据，多用于实时图标（照片）下载显示。

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 OF 指令实现照片实时显示
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。
23	二维码显示	根据指定内容在屏上显示指定的二维码图形信息。



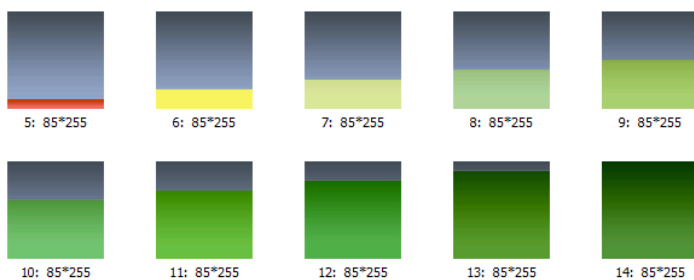
5.1 图标变量

5.1.1 变量图标显示 (0x00)

地址偏移量 (双字节)	定义	数据长度 (字节)	说明
0x00	*VP	2	变量指针，变量为整数格式。
0x01	X, Y	4	变量显示位置，图标左上角坐标位置。
0x03	V_Min	2	变量下限，越界不显示。
0x04	V_Max	2	变量上限，越界不显示。
0x05	Icon_Min	2	V_Min对应的图标ID。
0x06	Icon_Max	1	V_Max对应的图标ID。
0x07:H	Icon_Lib	1	图标库存储位置。
0x07:L	Mode	1	ICON显示模式。0x00=透明（不显示背景），其他=显示图标背景。



例如：让上图中左边刻度条显示以下 10 种不同状态，串口发送指令：



变量属性	
名称定义	变量图标
描述指针(0x)	FFFF
变量存储地址(0x)	0120
图标文件	24.ICO
变量下限	0
下限对应图标	5
变量上限	9
上限对应的图标	14
ICON显示模式	显示背景
初始值	0

A5 5A 05 82 01 20 00 00

A5 5A 05 82 01 20 00 01

.....

A5 5A 05 82 01 20 00 08

A5 5A 05 82 01 20 00 09

说明：A5 5A：帧头

05：指令字节长度，82 01 20 00 01 共 5 字节

82：写变量寄存器指令

01 20: 设置的变量地址，如上图变量属性所示

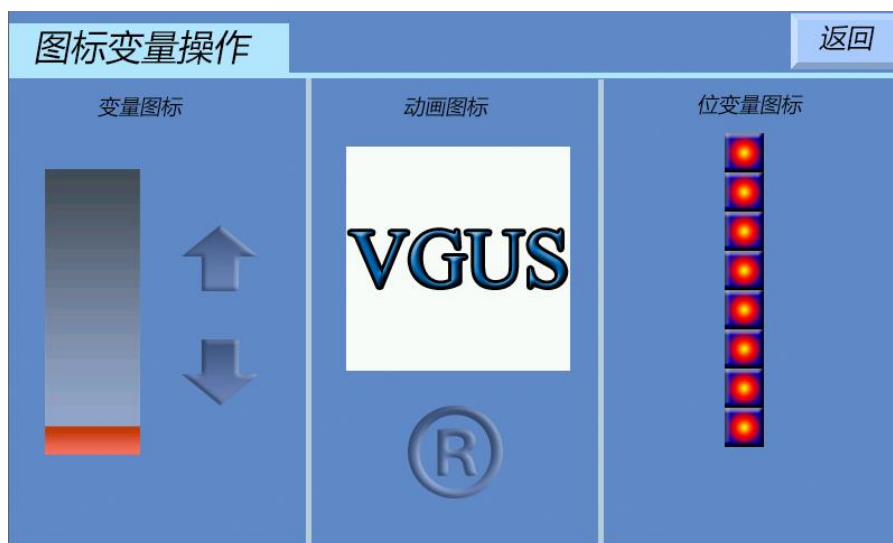
00 00: 设置的变量值，变量的上下限 0-9 与编号为 5-14 的图标一一对应，即发送 00 00 显示 5 号图标，00 01 显示 6 号图标，以此类推

图标 ICON 文件的生成详细步骤请参考 VGUS 开发软件使用说明。

5.1.2 动画图标显示(0x01)

地址偏移量 (双字节)	定义	数据长度 (字节)	说明
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
0x01	x, y	4	变量显示位置，图标左上角坐标位置。
0x03	0x0000	2	固定
0x04	V_Stop	2	变量为该值时固定。
0x05	V_Start	2	变量为该值时自动显示动画图标。
0x06	Icon_Stop	2	变量为V_Stop时固定显示的图标。
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。
0x08	Icon_End	2	
0x09:H	Icon_Lib	1	图标库存储位置。
0x09:L	Mode	1	ICON显示模式。0x00=透明

当变量不等于 V_Stop 或者 V_Start 时，不显示图标或者动画。



例如：让上图中中间动画图标显示下列动画效果，发送的指令：



变量属性	
名称定义	动画图标
描述指针(0x)	FFFF
变量存储地址(0x)	0121
停止值	0
开始值	1
图标文件	24.ICO
停止图标	15
开始图标	15
结束图标	28
ICON显示模式	显示背景
初始值	0

开始动画指令 A5 5A 05 82 01 21 00 01

停止动画指令 **A5 5A 05 82 01 21 00 00**

说明：**A5 5A**：帧头

05：指令字节长度，**82 01 21 00 01** 共 5 字节

82：写变量寄存器指令

01 21：设置的变量地址，如上图变量属性所示

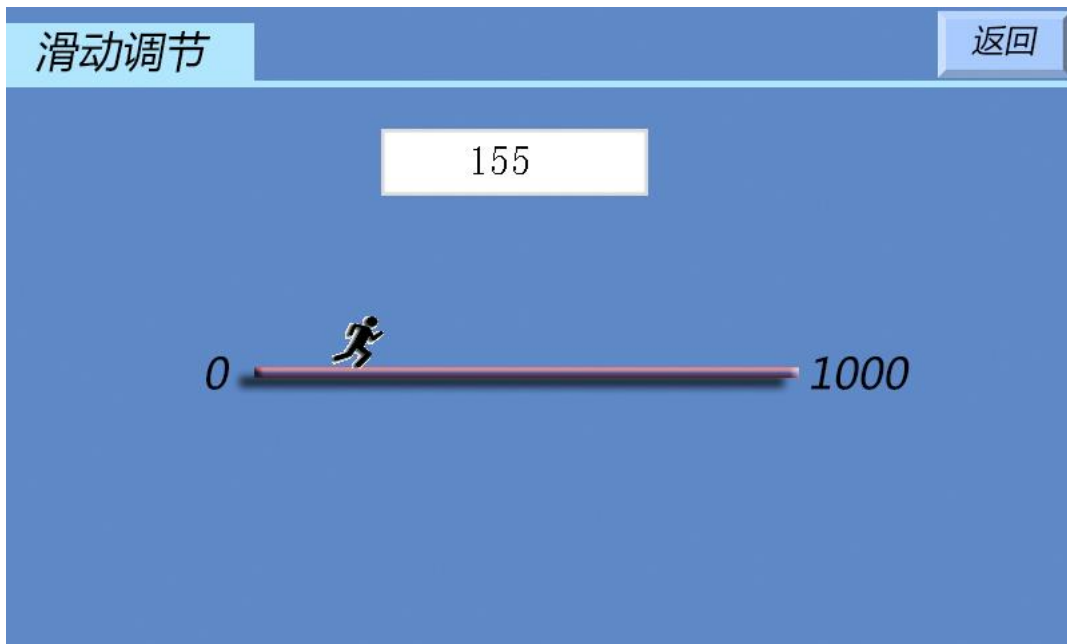
00 00：设置的变量开始值，发送开始值后循环播放开始图标到结束图标，发送停止值后显示停止图标

图标 **ICON** 文件的生成详细步骤请参考 **VGUS** 开发软件使用说明。

5.1.3 滑动刻度指示(0x02)

地址偏移量 (双字节)	定义	数据长度 (字节)	说明
0x00	*VP	2	变量指针，变量格式由VP_DATA_Mode来决定。
0x01	V_begain	2	对应起始变量刻度值。
0x02	V_end	2	对应终止变量刻度值。
0x03	X_begain	2	起始刻度坐标（纵向为Y坐标）。
0x04	X_end	2	终止刻度坐标（纵向为Y坐标）。
0x05	Icon_ID	2	滑动刻度块的图标ID。
0x06	Y	2	刻度指示图标显示的Y坐标值。
0x07:H	X_adj	1	刻度指示图标显示的X坐标前移偏移量。
0x07:L	Mode	1	刻度模式。0x00=横向刻度条，其他=纵向刻度条。
0x08:H	Icon_Lib	1	图标库存储位置。
0x08:L	Icon_mode	1	ICON显示模式。0x00=透明（不显示背景），其他=显示图标背景。
0x09:H	VP_DATA_Mode	1	0x00=*VP指向一个整型变量； 0x01=*VP指向一个整型变量高字节数据； 0x02=*VP指向一个整型变量低字节数据。

变量属性	
名称定义	滑块刻度指示
描述指针(0x)	FFFF
变量存储地址(0x)	0180
起始刻度值	0
终止刻度值	1000
刻度模式	横向刻度条
图标文件	24.ICO
滑动图标	0
ICON显示模式	透明
Y坐标前移偏移量	230
X坐标前移偏移量	0
变量模式	指向一个整型变量
初始值	0



例如：将上图中滑动刻度显示在进度条的任意位置，串口发送指令：

A5 5A 05 82 01 80 00 00

A5 5A 05 82 01 80 00 01

.....

A5 5A 05 82 01 80 03 E7

A5 5A 05 82 01 80 03 E8

说明：A5 5A：帧头

05：指令字节长度，82 01 80 00 01 共 5 字节

82：写变量寄存器指令

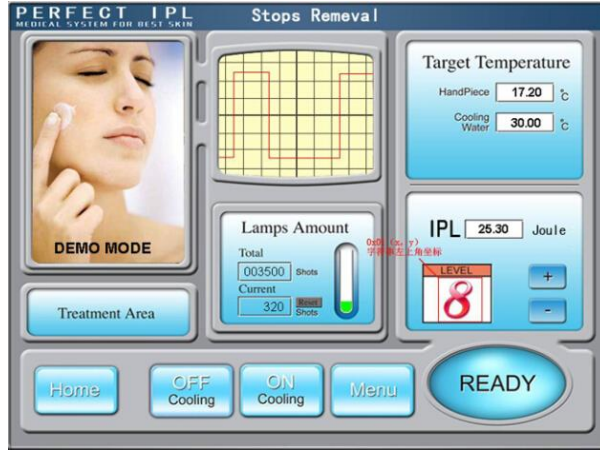
01 80：设置的变量地址，如上图变量属性所示

00 00：设置的变量刻度值，即从 00 00 变化到 03 E8

5.1.4 艺术字变量显示(0x03)

地址偏移量 (双字节)	定义	数据长度 (字节)	说明
0x00	*VP	2	变量指针
0x01	X, Y	4	起始显示位置 左对齐模式，坐标为显示字符串左上角坐标 右对齐模式，坐标为显示字符串的右上角坐标
0x03	Icon0	2	0对应的ICON_ID，排列顺序为0123456789-.
0x04:H	Icon_Lib	1	图标库存储位置。
0x04:L	Icon_Mode	1	ICON显示模式：0x00=透明（不显示背景） 其他=显示图标背景
0x05:H	整数位数	1	显示的整数位数
0x05:L	小数位数	1	显示的小数位数
0x06:H	数据变量类型	1	0x00=整数(2 字节)，-32768 到 32767 0x01=长整数(4 字节)-2147483648 到 2147483647 0x02=*VP 高字节，无符号数0 到 255

			0x03=*VP 低字节, 无符号数0 到 255 0x04= 超 长 整 数 (8 字 节)-9223372036854775808 到 9223372036854775807 0x05=无符号整数(2 字节)0 到 65535 0x06=无符号长整数(4 字节)0 到 4294967295
0x06:L	对齐模式	1	0x00=左对齐, 0x01=右对齐



可理解为艺术字的图标显示，显示方法与图标显示基本一致。

5.1.5 图片动画显示(0x04)

地址偏移量 (双字节)	定义	数据长度 (字节)	说明
0x00	0x0000	2	固定
0x01	Pic_begain	2	起始图标位置
0x02	Pic_end	2	终止图标位置
0x03:H	Frame_Time	1	一幅图片显示的时间, 8ms

如果在 Pic_End 页面也设置“图片动画”控件，将可以实现不断重播。

串口发送指令切换图片或者触控按钮切换图片可以结束重播。

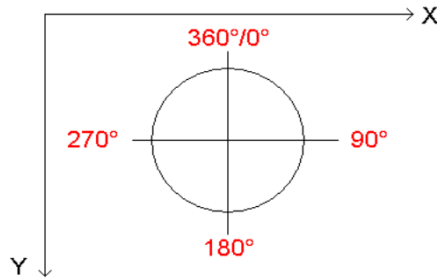
5.1.6 图标旋转(0x05)

地址偏移量 (双字节)	定义	数据长度 (字节)	说明
0x00	*VP	2	变量指针，变量格式由Mode决定。
0x01	Icon_ID	2	指定图标ID。
0x02	Icon_Xc	2	Icon图标上的指定旋转中心位置X坐标。
0x03	Icon_Yc	2	Icon图标上的指定旋转中心位置Y坐标。
0x04	Xc	2	Icon图标显示到当前屏幕的旋转中心位置X坐标。
0x05	Yc	2	Icon图标显示到当前屏幕的旋转中心位置Y坐标。
0x06	V_Begain	2	对应起始旋转角度的变量值，整型，越界不显示。
0x07	V_End	2	对应终止旋转角度的变量值，整型，越界不显示。
0x08	AL_Begain	2	起始旋转角度0-720 (0x0000-0x02D0),单位0.5度。
0x09	AL_End	2	转动角度= (终止角度-起始角度) / (终止值-起始值) *输入值;

			终止旋转角度0-720 (0x0000-0x02D0)，单位0.5度。
0x0A: H	VP_Mode	1	0x00=*VP指向一个整型变量； 0x01=*VP指向一个整型变量高字节数据； 0x02=*VP指向一个整型变量低字节数据。
0x0A: L	Lib_ID	1	图标库存储位置。
0x0B	Mode	1	ICON显示模式。0x00=透明（不显示背景），其他=显示图标背景。

本指令主要用于仪表刻度盘的指针指示。

旋转始终假定为“顺时针”转动，即 AL_End 必须大于 AL_Begain(如果 AL_End 小于 AL_Begain，系统处理时会自动加上 360°)。



5.1.7 位变量图标显示(0x06)

地址偏移量 (双字节)	定义	数据长度 (字节)	说明																													
0x00	*VP	2	位变量指针，字变量																													
0x01	*VP_AUX		辅助变量指针，双字，用户禁用																													
0x02	Act_Bit_Set	2	为1的bit值说明*VP对应位置需要显示																													
0x03: H	Display_Mode	1	显示模式定义： <table><tr><th rowspan="2">Display_Mode</th><th colspan="2">Bit值</th></tr><tr><th>0</th><th>1</th></tr><tr><td>0x00</td><td>ICONOS</td><td>ICON1S</td></tr><tr><td>0x01</td><td>ICONOS</td><td>不显示</td></tr><tr><td>0x02</td><td>ICONOS</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>ICONOS-ICON0E动画</td><td>ICON1S</td></tr><tr><td>0x06</td><td>ICONOS-ICON0E动画</td><td>不显示</td></tr><tr><td>0x07</td><td>ICONOS-ICON0E动画</td><td>ICON1S-ICON1E动画</td></tr></table>	Display_Mode	Bit值		0	1	0x00	ICONOS	ICON1S	0x01	ICONOS	不显示	0x02	ICONOS	ICON1S-ICON1E动画	0x03	不显示	ICON1S	0x04	不显示	ICON1S-ICON1E动画	0x05	ICONOS-ICON0E动画	ICON1S	0x06	ICONOS-ICON0E动画	不显示	0x07	ICONOS-ICON0E动画	ICON1S-ICON1E动画
			Display_Mode		Bit值																											
				0	1																											
			0x00	ICONOS	ICON1S																											
			0x01	ICONOS	不显示																											
			0x02	ICONOS	ICON1S-ICON1E动画																											
			0x03	不显示	ICON1S																											
			0x04	不显示	ICON1S-ICON1E动画																											
			0x05	ICONOS-ICON0E动画	ICON1S																											
			0x06	ICONOS-ICON0E动画	不显示																											
0x07	ICONOS-ICON0E动画	ICON1S-ICON1E动画																														
0x03: L	Move_mode	1	位图图标排列方式： 0x00=X++, Act_Bit_Set指定的不处理bit不保留位置 0x01=Y++, Act_Bit_Set指定的不处理bit不保留位置 0x02=X++, Act_Bit_Set指定的不处理bit保留Dis_MOV位置 0x03=Y++, Act_Bit_Set指定的不处理bit保留Dis_MOV位置																													

0x04:H	Icon_Mode	1	ICON显示模式: 0x00=透明 (不显示背景) 其他=显示图标背景
0x04:L	Icon_Lib	1	图标库存储位置
0x05	ICONOS	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
0x06	ICONOE	2	显示动画模式, bit 0 图标动画结束 ID 位置
0x07	ICONIS	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
0x08	ICONIE	2	显示动画模式, bit 1 图标动画结束 ID 位置
0x09	(x, y)	4	起始变量显示位置, 图标左上角坐标
0x0B	Dis_MOV	2	下一个图标坐标移动的间隔

5.2 文本变量

5.2.1 数据变量显示(0x10)

地址偏移量 (双字节)	定义	数据长度 (字节)	说明
0x00	*VP	2	变量指针
0x01	X, Y	4	起始显示位置, 显示字符串左上角坐标
0x03	Color	2	显示颜色
0x04:H	Lib_ID	1	ASCII字库位置
0x04:L	字体大小	1	字符X方向的点阵数
0x05:H	对齐方式	1	0x00=左对齐, 0x01=右对齐, 0x02=居中
0x05:L	整数位数	1	显示的整数位数 (整数位数和小数位数之和不能超过20)
0x06:H	小数位数	1	显示的小数位数 (整数位数和小数位数之和不能超过20)
0x06:L	变量数据类型	1	0x00=整数(两字节): -32768到32767 0x01=长整数(4字节): -2147483648到2147483647 0x02=VP*高字节: 0到255 0x03=VP*低字节: 0到255 0x04=超长整数(8字节): -9223372036854775808 到 9223372036854775807 0x05=无符号整数(2字节): 0到65536 0x06=无符号长整数(4字节): 0到4294967295
0x07:H	Len_unit	1	变量单位显示长度, 0x00表示不显示单位
0x07:L	String_unit	9	单位字符串, ASCII编码
0x0C	Bclr	2	背景色



变量属性	
名称定义	数据变量显示
描述指针(0x)	FFFF
变量存储地址(0x)	0020
文本颜色	0; 0; 0
字库位置	0
字体大小	16
对齐方式	左对齐
变量类型	int (2Byte)
整数位数	5
小数位数	0
显示单位	
初始值	0
无效位补零	<input type="checkbox"/>



例如：显示整数“1234”，串口返回指令：

A5 5A 05 83 00 20 04 D2

说明：A5 5A：帧头

05：指令字节长度，83 00 20 04 D2 共 5 字节（不含帧头）

83：写变量寄存器指令

00 20：变量地址，在控件属性中设置

04 D2：数据内容，1234 的十六进制数据

5.2.2 文本显示(0x11)


地址偏移量 (双字节)	定义	数据长度 (字节)	说明
0x00	*VP	2	文本指针

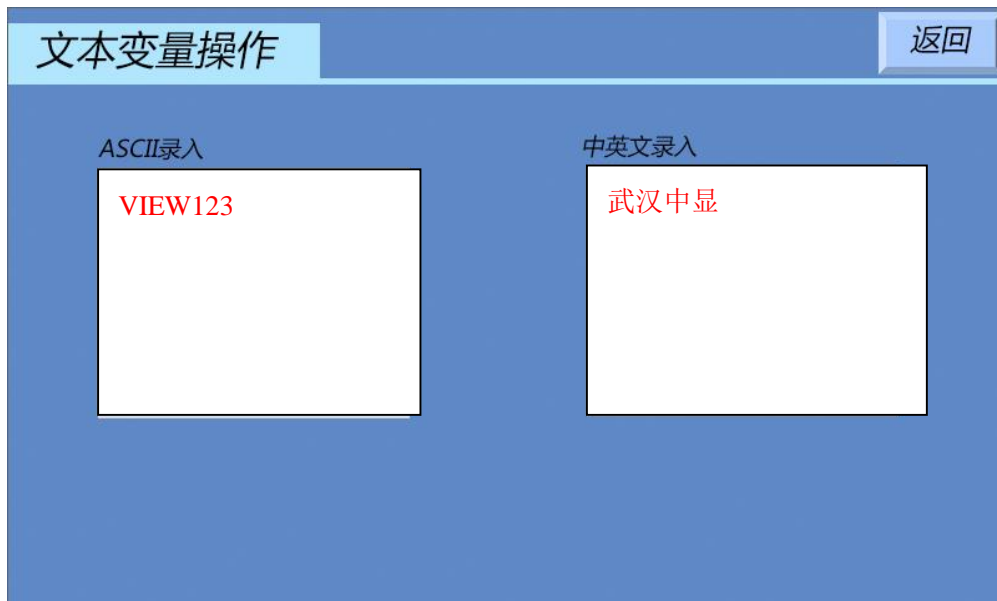


0x01	X, Y	4	起始显示位置，显示字符串左上角坐标
0x03	Color	2	显示文本颜色
0x04	Xs, Ys, Xe, Ye	8	文本框
0x08	Text_Length	2	显示字节数量，遇到0xFFFF数据或者显示到文本框尾则不再显示
0x09:H	Font0_ID	1	编码方式为0x01-0x04时ASCII字库位置
0x09:L	Font1_ID	1	编码方式为0x00、0x05，以及0x01-0x04的非ASCII字符使用的字库位置
0x0A:H	Font_X_Dots	1	字体X方向点阵数，0x01-0x04模式时ASCII字符X按照Y/2计算
0x0A:L	Font_Y_Dots	1	字体Y方向点阵数，字库中Y方向的点阵数必须为偶数
0x0B:H	Encode_Mode	1	. 7定义了文本显示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，字符宽度为固定设定的点阵数 . 6-. 0定义了文本的编码方式： 0x00=8bit编码，0x01=GB2312内码，0x02=GBK，0x03=BIG5，0x04=SJIS， 0x05=UNICODE
0x0B:L	HOR_Dis	1	字符水平间距
0x0C:H	VER_Dis	1	字符垂直间距
0x0C:L	未定义	1	写0x00

注意，文本显示时，字库中字体的Y方向点阵数目必须为偶数。

VGUS 屏预装的 0#字库，包含 4*8—64*128 点阵的所有 ASCII 字符。

变量属性	
名称定义	文本显示
描述指针(0x)	FFFF
变量存储地址(0x)	0140
文本颜色	 255; 0; 0
编码方式	0x01=GB2312内码
字符间距自动调整	<input checked="" type="checkbox"/>
文本长度	40
FONT0 ID	0
FONT1 ID	34
X方向点阵数	24
Y方向点阵数	24
水平间隔	0
垂直间隔	0
初始值	



例如：显示英文加数字“VIEW123”，串口发送指令：

A5 5A 0A 82 01 40 56 49 45 57 31 32 33

说明：**A5 5A**：帧头

0A：指令字节长度，**82 01 40 56 49 45 57 31 32 33** 共 10 字节（不含帧头）

82：写变量寄存器指令

01 40：变量地址，在控件属性中设置

56 49 45 57 31 32 33：数据内容，“VIEW123”的 ASCII 码

例如：显示中文“武汉中显”，串口发送指令：

A5 5A 0B 82 01 40 CE E4 BA BA D6 D0 CF D4

说明：**A5 5A**：帧头

0B：指令字节长度，**82 01 40 CE E4 BA BA D6 D0 CF D4** 共 11 字节（不含帧头）

82：写变量寄存器指令

01 40：变量地址，在控件属性中设置

CE E4 BA BA D6 D0 CF D4：数据内容，“武汉中显”的汉字内码

5.2.3 RTC显示(0x12)

► 文本RTC显示

地址偏移量 (双字节)	定义	数据长度 (字节)	说明
0x00	0x0000	2	
0x01	X, Y	4	起始显示位置，显示字符串左上角坐标
0x03	Color	2	字体颜色
0x04:H	Lib_ID	1	字库位置
0x04:L	字体大小	1	X方向的点阵数
0x05	String_Code	Max14	编码字符串，使用RTC编码表和ASCII字符组成



			假设当前时间2014-05-01 12:00:00 星期三 Y_M_D H:Q:S 0x00将显示为 2014-05-01 12:00:00 M_D W H:Q 0x00将显示为 05-01 WEN 12:00
--	--	--	---

RTC 编码表

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 THU FRI SAT
编码结束	0x00	

➤ 表盘时钟显示

地址偏移量 (双字节)	定义	数据长度 (字节)	说明
0x00	0x0001	2	
0x01	X,Y	4	时钟表盘的指针中心
0x03	Icon_Hour	2	时针 ICON 的 ID, 0xFFFF 表示时针不显示。
0x04	Icon_Hour_central	4	时针 ICON 的旋转中心位置。
0x06	Icon_Minute	2	分针 ICON 的 ID, 0xFFFF 表示分针不显示
0x07	Icon_Minute_central	4	分针 ICON 的旋转中心位置。
0x09	Icon_Second	2	秒针 ICON 的 ID, 0xFFFF 表示秒钟指针不显示
0x0A	Icon_Second_central	4	秒针 ICON 的旋转中心位置。
0x0C:H	ICON_Lib	1	指针图标所在的 ICON 库文件 ID
	未定义	1	写 0x00

5.2.4 HEX时间变量显示 (0x13)

地址偏移量 (双字节)	定义	数据长度 (字节)	说明
0x00	*VP	2	变量指针数据串首地址, 变量为BCD编码
0x01	X, Y	4	起始显示位置, 显示字符串左上角坐标
0x03	Color	2	字体颜色
0x04:H	Byte_Num	1	*VP指针高字节开始显示的数目, 0x01-0x0F
0x04:L	Lib_ID	1	字库位置, 必须是半角字库, 如果Lib_ID不为0, 则必须用8bit编码



0x05:H	Font_X	1	X方向点阵数目
0x05:L	String_Code	Max13	<p>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编码后，会从编码字符串中顺序的取出一个ASCII字符来间隔显示</p> <p>编码字符串中特殊编码定义：</p> <p>0x00：无效，不显示字符，两个BCD编码连在一起</p> <p>0x0D：换行显示。即 $X=Xs, Y=Y+Font_X*2$</p>

变量属性	
名称定义	时间变量
描述指针(0x)	FFFF
变量存储地址(0x)	6F00
文本颜色	0; 0; 255
字库位置	0
字体大小	10
Byte_Num	2
编码字符串	:
16进制录入	3A

例如：显示时间变量“08: 30”，串口发送指令：

A5 5A 05 82 6F 00 08 30

说明：A5 5A：帧头

05：指令字节长度，82 6F 00 08 30 共 5 字节（不含帧头）

82：写变量寄存器指令

6F 00：变量地址，在控件属性中设置

08 30：数据内容，08 30 的 BCD 码

5.2.5 文本滚屏显示（0x14）

地址偏移量 (双字节)	定义	数据长度 (字节)	说明
0x00	*VP	2	<p>文本指针</p> <p>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</p>
0x01:H	Rolling_Mode	1	滚屏模式：0x00表示从右向左滚屏
0x01:L	Rolling_Dis	1	滚屏间距，每个VGUS周期文本滚动的像素点阵数
0x02:H	Adjust_Mode	1	<p>对齐方式：0x00=左对齐，0x01=右对齐，0x02=居中</p> <p>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</p>
0x02:L	未定义	1	写 0x00
0x03	Color	2	显示文本颜色
0x04	Xs Ys Xe Ye	8	文本框
0x08:H	Font0_ID	1	<p>编码方式为 0x01-0x04 时：ASCII 字符显示的字库位置</p> <p>编码方式为 0x00、0x05 时：本参数不用设置，写 0x00 即可</p>



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
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 Y/2计算）
0x09:L	Font_Y_Dots	1	字体Y方向点阵数目
0x0A: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
0x0A:L	Text_Dis	1	字符间隔

注意，文本显示时，字库中字体的Y方向点阵数目必须为偶数。

VGUS 屏预装的 0 号字库，包含 4*8—64*128 点阵的所有 ASCII 字符。

发送指令参考文本显示。

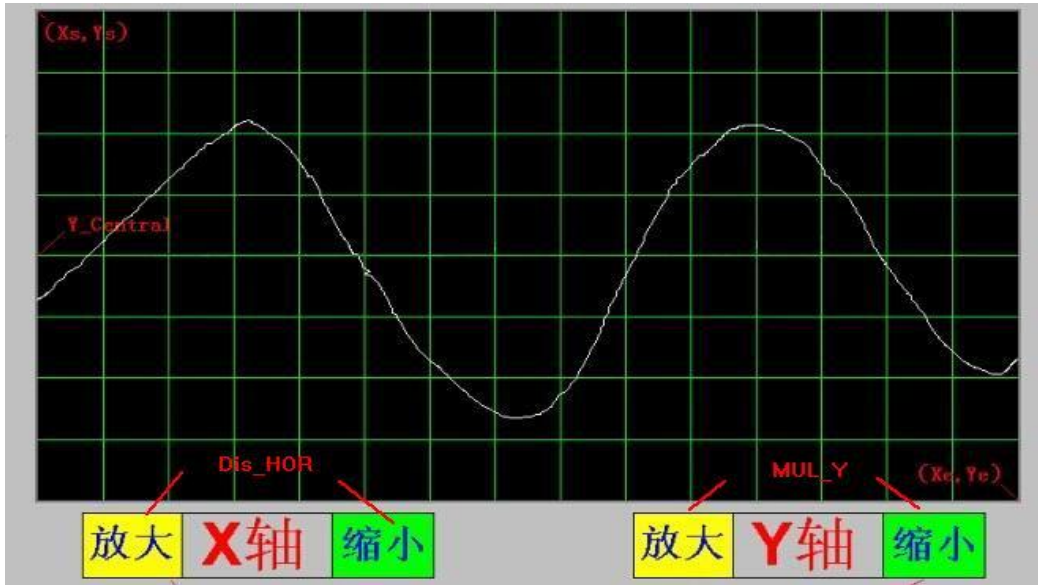
5.3 图形变量

5.3.1 实时曲线（趋势图）显示(0x20)

地址偏移量 (双字节)	定义	数据长度 (字节)	说明
0x00	0x0000	2	无定义
0x01	Xs:Ys:Xe:Ye	8	曲线窗口：左上角坐标（Xs,Ys），右下角坐标（Xe,Ye）
0x05	Y_Central	2	曲线中心位置
0x06	VD_Central	2	中心曲线对应的曲线数据值
0x07	Color	2	曲线颜色
0x08	NUL_Y	2	曲线放大倍数，单位是1/256，0x0000-0xFFFF
0x09:H	CHANEL	1	数据源通道：0x01-0x07
0x09:L	Dis_HOR	1	横轴间隔：0x01-0xFF

曲线数据用 0x84 指令发送，请参考 2.2 指令集说明





如果把变量描述内容存储在数据存储空间（*SP 指定存储位置），那么：

- 结合 0x02 增量触控指令，可以实现无需要用户代码干预的曲线自动缩放
- 结合 0x03 拖动触控指令修改 Y_Central 值，可以实现无需要用户代码干预的曲线上下移动

满量程曲线的纵轴放大倍数计算：

$MUL_Y = (Ye - Ys) * 256 / (Vmax - Vmin)$ 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 A/D 采集数据（Vmax=4095 Vmin=0）要对应应在 Ys=50, Ye=430 的屏幕区域满量程显示，那么：

$MUL_Y = (430 - 50) * 256 / (4095 - 0) = 23.7$ 向下舍入取 23。

5.3.2 基本图形显示(0x21)

地址偏移量 (双字节)	定义	数据长度 (字节)	说明
0x00	*VP	2	变量数据指针
0x01	Area	8	绘图显示区域定义：指定区域的左上角、右下角坐标 绘图越界将不显示，仅对 0x0001-0x0005、0x0009、0x000A、0x000B 指令有效
0x05:H	Dashed_Line_ Ent	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 其它：使用线段的绘图指令使用实线显示线段
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 1 段虚线点阵数、第 2 段实线点阵数、第 2 段虚线点阵数 例如，设置 0x10 0x04 0x10 0x04 将显示虚线，设置 0x10 0x04 0x02 0x04 将显示点划线
		13	保留，写0x00

基本图形显示先在配置文件中定义一个“绘图板”功能，而具体的绘图操作则由*VP 指向的变量存储器内容决定。用户通过改变变量存储器功能来实现不同的绘图功能。

（变量存储空间的）变量数据格式说明

地址	定义	说明
VP	CMD	绘图指令

VP+1	Data_Pack_Nu m_Max	最大数据包数目：连线指令（0x0002） 定义为连线线条数目，即顶点数-1
VP+2	DATA_Pack	数据

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	连线顶点0坐标，X坐标高字节为判断条件
		0x03	2	(x, y) 1	连线顶点1坐标，X坐标高字节为判断条件
		0x01+2*n	2	(x, y) n	连线顶点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 填充	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	圆颜色
0x0006	图片 区域 剪切、 粘贴	0x00	2	Pic_ID	剪切图片区域所在页面ID，高字节为判断条件
		0x02	2	(x, y) s	剪切图片区域左上角
		0x04	2	(x, y) e	剪切图片区域右下角
		0x06	2	(x, y)	剪切图片区域粘贴到当前页面的坐标位置，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0x**07的高字节指定 图标固定为不显示背景色
0x008	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
		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°，0-720



		0x05	1	DEG_E0	终止角度，单位 0.5°，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，8bit 编码，对仅高字节有效 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	矩形区域XOR	0x00	2	(x, y) s	矩形区域左上角坐标，x坐标高字节为判断条件
		0x02	2	(x, y) e	矩形区域右下角坐标
		0x04	1	Color	矩形区域XOR的颜色，0xFFFF将进行反色操作
0x000E	双色位图显示	0x00	2	(x, y) s	位图显示矩形区域左上角坐标，x坐标高字节为判断条件
		0x02	1	X_Dots	位图X方向点阵数
		0x03	1	Y_Dots	位图Y方向点阵数
		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，即下一行的数据总是从一个新数据字（word）开始
0x000F	位图显示	0x00	2	(x, y) s	位图显示矩形区域左上角坐标，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，565RGB 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
		0x04	2	(x, y) e	待放大矩形域右下角坐标

判断条件：0xFF 绘图操作结束，0xFE 本次操作跳过（忽略）

基本图形显示举例（以 0x0006 图片区域剪切、粘贴指令为例）

Step1 在界面上定义一个基本图形显示，变量*VP 指向 0x1000 地址；

Step2 用 USB 把配置文件下载到 VGUS 屏；

Step3 串口向 0x1000 地址(*VP)写入 0x0006 指令相关内容（把第三幅页面的(100,100)(512,256)区域剪切粘贴到当前界面的(0,0)位置）。





只要 VP 内容不变 VGUS 屏将在绘图板指令所在页面执行指令，显示剪切、粘贴内容。

区域范围设置	
X坐标	0
Y坐标	0
宽度	800
长度	480
变量属性	
名称定义	基本图形
描述指针(0x)	FFFF
变量存储地址(0x)	1000
虚线/点划线格式	<input type="checkbox"/>

坐标（340，237）与坐标（585，237）端点连线

A5 5A 11 82 10 00 00 02 00 01 00 00 01 54 00 ED 02 49 00 ED

坐标（340，237）置点

A5 5A 11 82 10 00 00 01 00 01 01 54 00 ED 00 00

左上角坐标（340，237）右下角坐标（585，340）矩形

A5 5A 11 82 10 00 00 03 00 01 01 54 00 ED 02 49 01 54 00 00

圆心坐标（470，325），半径为 50 整圆

A5 5A 0F 82 10 00 00 05 00 01 01 D6 01 45 00 32 00 00

1 号界面左上角坐标（265，64）右下角坐标（785，256）剪切粘贴到当前界面坐标（265，64）

A5 5A 15 82 10 00 00 06 00 01 00 01 01 09 00 40 03 11 01 00 01 09 00 40

端点坐标（80，80）（128，128）线段显示

A5 5A 11 82 10 00 00 0A 00 01 00 00 00 50 00 50 00 80 00 80

说明：上述所有指令中 00 01 指最大数据包数目，即执行画图指令的次数，如画一条线段、一个矩形框、一个圆设置为 00 01，画两条线段、两个矩形框、两个圆设置为 00 02。

5.3.3 列表显示 (0x22)

地址偏移量 (双字节)	定义	数据长度 (字节)	说明
0x00	*VP	2	表格内容指针，即TAB_X_Num与TAB_Y_Num数组的首地址
0x01:H	TAB_X_Num	1	列数目，0x01-0xFF
0x01:L	TAB_Y_Num	1	行数目，0x01-0xFF

0x02:H	TAB_X_Start	1	表格起始显示的列位置, 0x00-0xFF
0x02:L	TAB_Y_Start	1	表格起始显示的行位置, 0x00-0xFF
0x03:H	Unit_Data_Num	1	<p>➤ 0x01-0xFF所有单元格存储的数据长度相同,</p> <p>➤ 0x00由*VP变量地址指针指向变量存储空间定义不同列单元格的数据长度。(word, 字长度)</p> <p>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</p> <p>例如, *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</p>
0x03:L	Encode_Mode	1	<p>.7定义了文本显示字符间距是否自动调整:</p> <p>.7=0字符间距自动调整</p> <p>.7=1字符间距不自动调整, 字符宽度为固定设定的点阵数</p> <p>.6定义了表格内容格式:</p> <p>.6=0 表格内容为文本格式</p> <p>.6=1. 未定义</p> <p>.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</p> <p>.4 未定义, 写 0。</p> <p>.3-.0定义了文本的编码方式: 0x00=8bit编码, 0x01=GB2312内码, 0x02=GBK, 0x03=BIG5, 0x04=SJIS, 0x05=UNICODE</p>
0x04	Xs, Ys, Xe, Ye	8	表格显示区域定义, 左上角坐标, 右下角坐标, 越界不显示
0x08	Color_line	2	表格边框颜色
0x09	Color_text	2	表格文本颜色
0x0A:H	Font0_ID	1	编码方式为0x01-0x04时ASCII字库位置
0x0A:L	Font1_ID	1	编码方式为0x00、0x05, 以及0x01-0x04的非ASCII字符使用的字库位置
0x0B:H	Font_X_Dots	1	字体X方向点阵数, 0x01-0x04模式时ASCII字符X按照X/2计算
0x0B:L	Font_Y_Dots	1	字体Y方向点阵数, 字库中Y方向的点阵数必须为偶数
0x0C:H	TAB_X_Adj_Mod	1	设置TAB_X_Start不为0时, 进行表头显示控制, 0x00=首列不显示, 0x01=首列显示
0x0C:L	TAB_Y_Adj_Mod	1	设置TAB_Y_Start不为0时, 进行表头显示控制, 0x00=首行不显示, 0x01=首行显示

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

第 1 个字高字节: Mode 选择数据类型

0x00=整数(2 字节), -32768 到 32767

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

0x02=*VP 高字节, 无符号数0 到 255

0x03=*VP 低字节, 无符号数0 到 255

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

0x05=无符号整数(2 字节)0 到 65535

0x06=无符号长整数(4 字节)0 到 4294967295

0x10=时间格式 1, 12:34:56BCD 码串



0x11=时间格式 2, 12-34-56BCD 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=其它无定义

第 2 个字: 定义单元格文本颜色

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

对于特别大的表格, 通过触摸屏修改 TAB_X_Start、TAB_Y_Start 值可以很方便的实现表格的定位和拖动

5.3.4 二维QR码图形显示 (0x25)

地址偏移量 (双字节)	定义	数据长度 (字节)	说明
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。
0x01	(x, y)	4	二维码显示的坐标位置。 (x, y) 为二维码左上角在屏幕的坐标位置。 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 459 字节) 两种。
0x03	Unit_Pixels	2	每个二维码单元像素所占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。

变量属性	
名称定义	二维码显示
描述指针(0x)	FFFF
变量存储地址(0x)	0000
Unit_Pixels	4

例如: 显示网站 www.viewtech.cn 生成的二维码, 串口发送指令:

A5 5A 12 82 00 00 77 77 77 2E 76 69 65 77 74 65 63 68 2E 63 6E

说明: A5 5A: 帧头

12: 指令字节长度, 82 00 00 77 77 77 2E 76 69 65 77 74 65 63 68 2E 63 6E 共 18 字节 (不含帧头)

82: 写变量寄存器指令

00 00: 变量地址, 在控件属性中设置

77 77 77 2E 76 69 65 77 74 65 63 68 2E 63 6E: 数据内容, www.viewtech.cn 的 ASCII 码

6 描述指针

对于简单应用场合，用户无需了解本章知识。

6.1 描述指针介绍

VGUS 屏采用变量驱动，所有显示的字符、图标等都定义为一个变量，并分配变量存储地址、定义显示格式，然后生成配置文件并下载到显示终端上。在需要刷新显示时，用户仅需将变量内容和变量存储地址通过串口对应发送到显示终端，显示终端会自动按照定义好的显示格式显示。

对于每个变量来讲，其显示格式是固定的，是由下载到 VGUS 屏中的配置文件定义的。当用户需要临时修改变量显示格式时，可以通过本章介绍的描述指针实现。

用户在定义每个变量的时候，需要设置是否启用描述指针，如图 5-1 所示。



属性设置	
区域范围设置	
X坐标	473
Y坐标	462
宽度	6
高度	6
变量属性	
名称定义	数据变量显示
描述指针(0x)	FFFF
变量存储地址(0x)	0000
文本颜色	255; 0; 0
字库位置	0
字体大小	16
对齐方式	左对齐
变量类型	int (2Byte)
整数位数	8
小数位数	0
显示单位	
初始值	0

图 6-1 设置数据变量显示属性

如图描述指针一栏填写 FFFF 表示当前变量禁止描述指针功能。如果填写的是其它数据（0000-FFFE 中任意一个）表示当前变量启用描述指针，并且填写的数据用来指定变量存储器地址，以该地址为起始单元，在变量存储器内连续开辟了一块空间（不同的变量类型，块长度以及数据格式也都不同，每种变量本章都有详细表格定义了对应数据格式），用于存储变量的显示格式。用户可以通过变量存储器地址，利用 0x82 指令去动态修改变量显示格式。

每一个显示变量的属性都可以通过描述指针来读写。描述指针地址范围 0x0000-0xFFFF，也保存在用户变量数据存储区，因此用户定义的变量地址不能与描述指针的地址重叠。

6.2 描述指针应用举例

本节以改变一个数据变量的显示颜色为例，介绍描述指针的简单应用。

- 1、新建工程添加图片后，在界面上添加一个数据变量显示控件，设置描述指针为 4000，变量存储地址为 0000。如图 6-2-1 所示：


名称定义	数据变量显示
描述指针(0x)	4000
变量存储地址(0x)	0000
文本颜色	 119; 119; 119
字库位置	0
字体大小	16
对齐方式	左对齐
变量类型	int (2Byte)
整数位数	4
小数位数	0
显示单位	
初始值	1234

图 6-2-1 添加数据变量

- 2、查看 5.2 节关于数据变量的说明，如图 6-2-2 所示：

地址偏移量 (双字节)	定义	数据长度 (字节)	说明
0x00	*VP	2	变量指针
0x01	X, Y	4	起始显示位置，显示字符串左上角坐标
0x03	Color	2	显示颜色
0x04:H	Lib_ID	1	ASCII字库位置
0x04:L	字体大小	1	字符X方向的点阵数
0x05:H	对齐方式	1	0x00=左对齐，0x01=右对齐，0x02=居中
0x05:L	整数位数	1	显示的整数位数（整数位数和小数位数之和不能超过20）
0x06:H	小数位数	1	显示的小数位数（整数位数和小数位数之和不能超过20）
0x06:L	变量数据类型	1	0x00=整数(两字节)：-32768到32767 0x01=长整数(4字节)：-2147483648到2147483647 0x02=VP*高字节：0到255 0x03=VP*低字节：0到255 0x04=超长整数(8字节)： -9223372036854775808 到 9223372036854775807 0x05=无符号整数(2字节)：0到65536 0x06=无符号长整数(4字节)：0到4294967295
0x07:H	Len_unit	1	变量单位显示长度，0x00表示不显示单位
0x07:L	String_unit	9	单位字符串，ASCII编码
0x0C	Bclr	2	背景色

图 6-2-2 数据变量描述指针数据格式

从图中可以看到,颜色对应的偏移量为03,那么颜色属性对应的变量存储器地址为 $0x4000+03=0x4003$ 。通过 0x82 指令修改 0x4003 单元内的数据即改变了数据变量的显示颜色。

例如串口发送: A5 5A 05 82 40 03 F8 00

那么就设置数据变量显示为红色 (0xF800 为红色)。

本例中使用按钮键值返回控件,按下按钮时,直接将键值作为对应颜色值写入变量存储器 0x4003 单元。按钮返回控件属性设置如图 6-2-3 所示。

按钮属性	
名称定义	按键返回
数据自动上传	<input checked="" type="checkbox"/>
按钮效果	1
无按钮效果	<input type="checkbox"/>
页面切换	-1
无页面切换	<input checked="" type="checkbox"/>
语音	<input type="checkbox"/>
键值 (0x)	F800

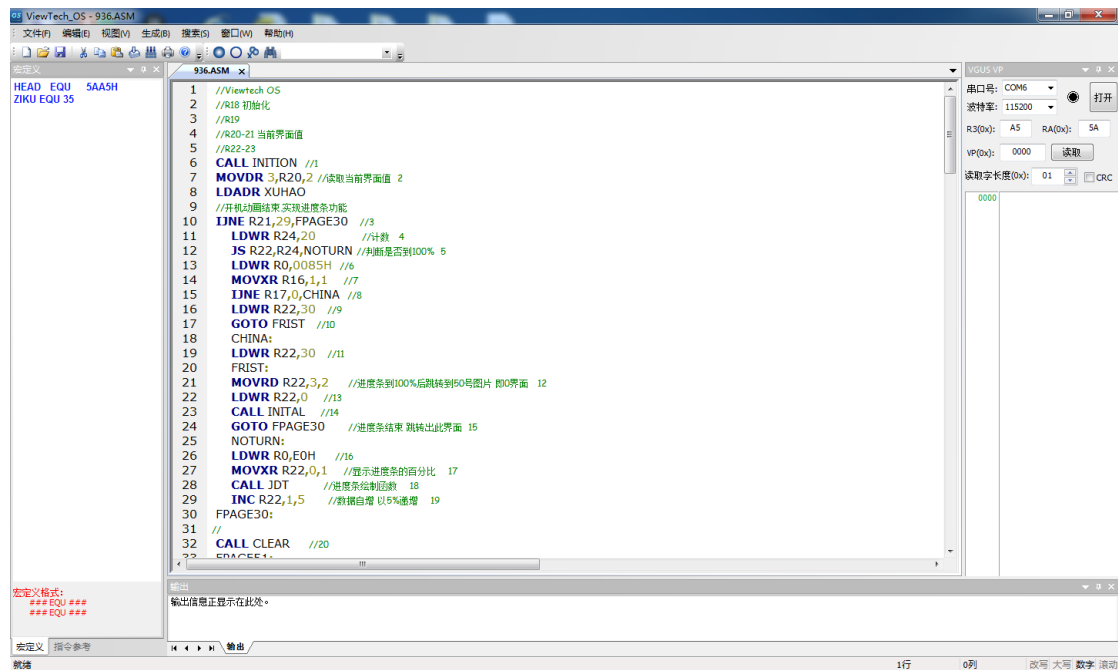
变量属性	
变量存储地址(0x)	4003
调节模式	按字写入变量

图 6-2-3 按钮返回控件属性设置

7 ViewTech_OS 用户程序设计

ViewTech_OS 平台采用类似汇编程序编写规范，在 VGUS 稳定的 GUI 平台下，方便用户针对自己的特殊需求快速、可靠的进行二次开发。

ViewTech_OS 的 PC 软件编译界面如下图所示。



基于 VGUS 的 ViweTech_OS 平台，用户最大代码空间是 256KB（32764 行代码）。ViewTech_OS 程序在每个 VGUS 周期（80/120/160/200mS）都完整运行一遍，所以 ViewTech_OS 程序中不能出现死循环。

VGUS 平台上 ViewTech_OS 的常见应用是使用 ViewTech_OS 来解析用户自定义数据协议和进行数据处理，代替标准 HMI 或工控机，不仅降低成本（VGUS 屏价格只有标准 HMI 或工控机的几分之一）也大大提升了系统可靠性，（标准 HMI 多是基于 PC 或者工控机架构，软件在 Windows CE 之类的通用操作系统平台下开发）。

7.1 基本约定

ViewTech_OS 寄存器变量：R0-R255，256 Byte；

VGUS 寄存器：对应 VGUS 0x80/0x81 指令访问的寄存器变量空间（0x00-0xFF）；

VGUS 变量：对应 VGUS 0x82/0x83 指令访问的变量存储器空间（0x0000-0x6FFF）；

字库空间：对应 32-127（0x20-0x7F）号汉字库，24MB。

伪汇编指令

➤ EQU 替换，编译时直接替换

比如，

PICID EQU 3

WORD EQU 2

MOVDR PICID,R10,WORD; 等效成 MOVDR 3,R10,2



- **DB** 定义 1 个字节或字（定义数据小于 255 将自动定义为字节）的 ROM 数据
- LDADR TAB1;** 把 TAB1 的 24bit 地址保存到 R5:R6:R7 地址指针寄存器

TAB1: DB 1,2,3,4

DB 1000,2000,3000,4000,-100

DB "武汉中显 VGUS"

- 注释用的无效标记，使用 **;**。

7.2 ViewTech_OS 汇编指令集

R#表示 ViewTech_OS 的 256 个寄存器之任意一个或一组，R0-R255；

<>表示立即数，汇编代码中，100，0x64，64H，064H 都是表示 10 进制数据 100。

COM2 串口仅 VGUS+内核硬件平台支持。

指令功能	操作码	操作数	说明
空操作	NOP		不执行任何操作。 NOP
VGUS 变量寄存器数据交换	MOVXR	R#,<MOD> <NUM>	R#: 寄存器或寄存器组； <MOD>: 0=寄存器到变量 1=变量到寄存器； <NUM>: 交换的数据字（Word）长0x00-0x80； 当<NUM>为 0x00 时，数据长度由 R9 决定。 VGUS 变量指针由 R0:R1 寄存器定义。 MOVXR R20,0,2
装载 N 个 8bit 立即数到寄存器组	LDBR	R#,<DATA>,<NUM>	R#: 寄存器或寄存器组。 <DATA>: 要装载的数据。 <NUM>: 要装载的寄存器个数，0x00表示256个。 LDBR R8,0x82,3
装载 1 个 16bit 立即数到寄存器组	LDWR	R#,<DATA>	R#: 寄存器组。 <DATA>: 要装载的数据。 LDWR R8,1000 LDWR R8,-300
程序空间查表（程序空间数据到寄存器）	MOVC	R#,<NUM>	R#: 寄存器或寄存器组。 <NUM>: 查表返回的字节数据长度。 表地址指针由R5:R6:R7寄存器定义。 MOVC R20,10
寄存器和寄存器数据交换	MOV	R#S,R#T,<NUM>	R#S: 源寄存器或寄存器组。 R#T: 目标寄存器或寄存器组。 <NUM>: 交换的字节数据长度，0x00表示长度由R9寄存器定义。 MOV R8,R20,3
寄存器到 VGUS 寄存器	MOV RD	R#,<D#>,<NUM>	R#: 寄存器或寄存器组； D#: VGUS的寄存器或寄存器组； <NUM>: 交换的字节数据长度。 MOV RD R10,3,2
VGUS 寄存器到寄存器	MOV DR	D#,<R#>,<NUM>	R#: 寄存器或寄存器组； D#: VGUS的寄存器或寄存器组； <NUM>: 交换的字节数据长度。 MOV DR 3R10,2



VGUS 变量和字库、数据库数据交换	MOVXL	<MOD>,<NUM>	<p><MOD>: 0=字库数据读到VGUS变量; 1=VGUS变量数据写到字库; 2=用户数据库数据读到VGUS变量; 3=VGUS变量数据写到用户数据库。 <NUM>: 交换数据(字, Word)长度; NUM=0, 则由 R9:R10 定义。</p> <p>VGUS 变量地址由R0:R1寄存器定义。 字库读写模式 (MOD=0、1): 字库由 R4(0x20-0x7F)寄存器指定, R5:R6:R7为字库内的数据操作首地址, 越界取消。 用户数据库读写模式 (MOD=2、3): 用户数据库首地址由R4:R5:R6:R7寄存器指定。 MOVXL 0,300</p>
VGUS变量之间交换数据	MOVXX	<NUM>	<p><NUM>: 交换 (字, Word) 数据长度。 <NUM>为0表示长度由R8:R9寄存器定义。 VGUS源变量地址由R0:R1寄存器定义。 VGUS目标变量地址由R2:R3寄存器定义。 MOVX 100</p>
寄存器变址寻址	MOVA		<p>R2规定了源寄存器 (组) 地址; R3规定了目标寄存器 (组) 地址; R9规定了交换的数据长度, 字节数。 MOVA</p>
32bit 整形数加法	ADD	R#A,R#B,R#C	<p>$C=A+B$, A、B为32bit整数; C为64bit整数。 ADD R10,R20,R30</p>
32bit 整形数减法	SUB	R#A,R#B,R#C	<p>$C=A-B$, A、B为32bit整数; C为64bit整数。 SUB R10,R20,R30</p>
64bit 长整数 MAC	MAC	R#A,R#B,R#C	<p>$C=(A*B+C)$, A、B是32bit整数, C是64bit整数。 MAC R10,R20,R30</p>
64bit 整形数除法	DIV	R#A,R#B,<MOD>	<p>A/B, 商是A, 余数是B。 A和B都是64bit寄存器。 <MOD>: 0=商不进行四舍五入, 1=商进行四舍五入。 DIV R10,R20,1</p>
变量扩展成 32bit	EXP	R#S,R#T,<MOD>	<p>把 R#S指向的数据转成32bit整数保存到R#T。 R#S: 源寄存器或寄存器数。 R#T: 32bit目标寄存器。 <MOD>: R#S数据类型, 0=8bit无符号1=8bit带符号2=16bit无符号数3=16bit整数。 EXP R10, R20,2</p>
32bit 无符号 MAC	SMAC	R#A,R#B,R#C	<p>$C=A*B+C$。 A、B是16bit无符号数, C是32bit无符号数。 SMAC R10, R20,R30</p>
寄存器自增量	INC	R#,<MOD>,<NUM>	<p>$R\#=R\#+NUM$, 无符号数自增计算, <NUM>为0x00-0xFF。 <MOD>: R#数据类型, 0=8bit 1=16bit。 INC R10,1,5</p>
寄存器自减量	DEC	R#,<MOD>,<NUM>	<p>$R\#=R\#-NUM$, 无符号数自减计算, <NUM>为 0x00-0xFF。 <MOD>: R#数据类型, 0=8bit 1=16bit。 DEC R10,0,1</p>



加载地址	LDADR	<Address>	把<Address>加载到R5:R6:R7 LDADR TAB LDADR 0x123456
与逻辑运算	AND	R#A,R#B,<NUM>	A=A AND B, 序列与逻辑运算。 <NUM>: R#A、R#B变量字节数目。 AND R10, R20,1
或逻辑运算	OR	R#A,R#B,<NUM>	A=A OR B, 序列或逻辑运算。 <NUM>: R#A、R#B变量字节数目。 OR R10, R20,1
异或逻辑运算	XOR	R#A,R#B,<NUM>	A=A XOR B, 序列异或逻辑运算。 <NUM>: R#A、R#B变量字节数目。 XOR R10, R20,1
解整数线性方程	ROOTLE		由16bit 整数 (X0,Y0)、(X1,Y1) 两点确定的直线上的X, 求解对应的Y值。 输入: X=R16 X0=R20 Y0=R22 X1=R24 Y1=R26。 输出: Y=R18。 ROOTLE
ANSI CRC-16 计算	CRCA	R#S,R#T,R#N	对序列数据计算 ANSI CRC-16(X16+X15+X2+1)。 R#S: 输入的寄存器组。 R#T: 输出 CRC 结果, 16bit, LSB 模式保存。 R#N: 保存计算 CRC 数据字节长度寄存器, 8bit。 CRCA R10, R80,R9
CCITT CRC-16 计算	CRCC	R#S,R#T,R#N	对序列数据计算 CCITT CRC-16(X16+X12+X5+1)。 R#S: 输入的寄存器组。 R#T: 输出 CRC 结果, 16bit, MSB 模式保存。 R#N: 保存计算 CRC 数据字节长度寄存器, 8bit。 CRCC R10, R80,R9
到 COM1_Rx_FIFO 读取MODBUS 数据帧	RMODBUS	R#A,R#T,R#C	检查 COM1 接收 FIFO 是否有满足要求的MODBUS 数据帧, 如果有则读取数据到寄存器并清空接收 FIFO。 R#A: 保存 MODBUS 接收数据包前 3 个字节 (地址:指令:长度) 的寄存器组。 如果长度为 0x00, 那么表示不进行长度匹配, 紧随其后的数据 (第 4 个字节) 表示了地址、指令、校验和之外的数据长度。 R#C: 返回状态的寄存器, 寄存器保存了返回数据: 0x00 表示未接收到匹配的 MODBUS 数据帧; 0xFF 表示接收到匹配的 MODBUS 数据帧并把数据读取到 R#T 寄存器组。 R#T: 匹配后, 保存MODBUS 数据的寄存器组。 RMODBUS R10, R20,R13
位分解	BITS	R#,<VP>	把 R#的 8 个比特分解到 VP 指向的 8 个 VGUS 字变量, MSB 方式, bit 1 分解为 0x0001, bit 0 分解为 0x0000。 R#: 需要进行位分解的寄存器, 8bit。 <VP>: VGUS 变量地址。 BITS R10,0x2000
位组合	BITI	R#,<VP>	把VP指向的8个VGUS字变量组合成1个字节位变量, 方式MSB 0x0000 为 bit 0, 非 0x0000 数据为 bit 1。 R#: 存储位组合数据的寄存器, 8bit。 <VP>: VGUS 变量地址。



			BITI R10,0x2000
HEX 转 ASC	HEXASC	R#S,R#T,<MOD>	<p>R#S: 需要转换的 32bit 整数;</p> <p>R#T: 转换后的 ASCII 字符串寄存器组;</p> <p><MOD>: 转换模式控制, 高 4bit 为整数位长度, 低 4bit 为小数位数目。</p> <p>转换的 ASCII 串带符号, 右对齐, 空位用 0x20 填充。</p> <p>对于数据 0x12345678,</p> <p><MOD>=0x62 转换结果为+054198.96</p> <p><MOD>=0xF2 转换结果为+3054198.96</p> <p>HEXASC R20, R30,0x62</p>
序列比较	TESTS	R#A,R#B,<NUM>	<p>依次比较A、B两个寄存器序列的值:</p> <p>值不同时, 返回A序列此时的地址到R0寄存器;</p> <p>如果A、B相同则返回0x00到R0寄存器。</p> <p>R#A: A序列寄存器。</p> <p>R#B: B序列寄存器。</p> <p><NUM>: 最大比较数据字节长度。</p> <p>TESTS R10, R20,16</p>
COM 配置	COMSET	<MODE>,<BS>	<p>设置串口模式:</p> <p><MODE>:</p> <p>高 4bit 选择要配置的串口, 0=COM1 1=COM2</p> <p>低 4bit 选择模式</p> <p>0x*0=N81 模式;</p> <p>0x*1=E81 模式;</p> <p>0x*2=O81 模式;</p> <p>0x*3=N82 模式。</p> <p><BS>: 波特率设置值</p> <p>对于 COM1, 设置值=6250000/设置的波特率。</p> <p>对于 COM2, 设置值高字节选择波特率因子</p> <p>00=5.208Mbps 01=15.625Mbps 02=1.302Mbps</p> <p>设置值低字节设置波特率</p> <p>设置值低字节=波特率因子/要设置的波特率。</p> <p>每次设置会自动清空串口接收 FIFO 一次。</p> <p>COMSET 0,54</p>
位测试、跳转	JB	R#,<Bit>,<TAB>	<p>测试R#寄存器的第<Bit>位, 1跳转, 0 继续执行下一条代码, 跳转范围+/-127 条指令。</p> <p>R#: 位测试的寄存器, 16bit。</p> <p><Bit>: 位测试位置, 0x00-0x0F, MSB 方式。</p> <p><TAB>: 跳转位置。</p> <p>JB R10,15, TEST1</p> <p>NOP</p> <p>TEST1: ADD R8, R12,R16</p>
变量比较、不相等跳转	CJNE	R#A,R#B,<TAB>	<p>比较 A、B两个8bit寄存器的内容, 相等则执行下一条指令, 不等则跳转, 跳转范围+/-127 条指令。</p> <p>TEST1: NOP</p> <p>INC R10,0,1</p> <p>CJNE R10, R11,TEST1</p>
整形数比较, 小于跳转	JS	R#A,R#B,<TAB>	<p>比较 A、B两个16bit 整数的大小, BA>=B则执行下一条指令, A<B则跳转, 跳转范围+/-127 条指令。</p> <p>JS R10, R12,TEST1</p> <p>NOP</p>



			TEST1: NOP
变量和立即数比较、不相等跳转	IJNE	R#,<INST>,<TAB>	比较 8bit 寄存器和立即数<INST>的内容，相等则执行下一条指令，不等则跳转，跳转范围+/-127 条指令。 IJNE R10, 100,TEST1 NOP TEST1: NOP
强制结束当前输入法	EXIT	R#A,R#B	强制结束当前的输入法。 R#A: 控制是否切换页面，0x00=不切换，0x01=切换； R#B: 要切换回的页面 ID（16bit）。 EXIT R10, R11
子程序调用返回	RET		CALL 调用指令返回。 RET
子程序调用	CALL	<PC>	调用子程序，最多支持 32 级程序嵌套。 CALL TEST
直接跳转	GOTO	<PC>	程序跳转 GOTO TEST1 NOP TEST1: NOP
串口发送	COMTXD	<COM>,R#S,R#N	把数据发送到指定的串口。 <COM>: 选择串口 0=COM1（VGUS 用户串口） 1=COM2（VGUS 保留串口）。 R#S: 要发送的数据寄存器组。 R#N: 要发送的字节数寄存器，8bit，寄存器数据 0x00 表示发送 256 字节数据。 COMTXD 0, R10,R9
串口打印	CPRTS	<COM>,<VP>	检查 VP 指向的 VGUS 变量地址有没有打印信息，有则打印到串口。VP 为 VGUS 的 0xFE07_05 打印指令对应的变量 VP 值,打印后清除 VP 地址的打印标记。 <COM>: 选择串口 0=COM1（VGUS 用户串口） 1=COM2（VGUS 保留串口）。 CPRTS 0,0x2000
检查COM_Rx_FIFO	RDXLEN	<COM>,R#	返回 COM 接收缓冲区（FIFO）接收数据字节长度（0-253）到 R#寄存器，0x00 表示没有数据。 <COM>: 选择串口，0=COM1 1=COM2。 RDXLEN 0, R10
读取COM_Rx_FIFO	RDXDAT	<COM>,R#A,R#B	从 COM 接收缓冲区（FIFO）中读取 R#B 个字节（01-253）到 R#A寄存器组；读取后 FIFO 长度自动调整。 <COM>: 选择串口，0=COM1 1=COM2。 RDXDAT 0, R11,R10
直接串口发送	COMTXI	<COM>,R#,<NUM>	把R#指向的<NUM>个寄存器内容发送到 COM。 <COM>: 选择串口，0=COM1 1=COM2。 COMTXI 0, R20,16



读取当前输入法内容到寄存器	SCAN	R#,<NUM>	把当前输入法下已经录入的最多<NUM>个字符加载到 R#+1 开始的寄存器, R#保存数据长度, 字符个数从当前输入法光标位置往前计算。 SCAN R20,6
写指定通道动态曲线缓冲区	WRLINE	R#S,R#I,<CH>	把R#S指向的N个16bit无符号整数, 加上16bit无符号整数 V_BIAS后写到<CH> (0x00-0x07) 指定的动态曲线缓冲区, R#I寄存器指向一个3字节变量: N, V_BIAS。 WRLINE R80, R10,2
汉字库匹配搜索	LIBSCH	R#A,R#B,R#C	到指定汉字库搜索匹配字符串的数据。 R#A 的 2 个寄存器规定了匹配字符串格式: R#A: 字符串长度 (0x00-0x1F), 0x00 表示字符串由 0x00或 0xFF 结尾。 R#A+1: 字符串数据, 最多 31 个字符。 R#B 的 11 个寄存器规定了数据库 (DATA[M][N] 数组) 格式和搜索格式: R#B: 字库 ID (0x20-0x7F, 0x00 表示不用重新加载), 每个 VGUS 周期调用一次后, 只要没有对其它汉字库做过操作就不用重新加载。 R#B+1: 二维数组的行维度 M, 0x0001-0x0FFFF; R#B+3: 二维数组的列维度 N, 0x01-0x80, 字数目; R#B+4: 搜索匹配模式, 0x00=左对齐, 0x01=任意位置匹配; R#B+5: 搜索开始行, 0x0000-0xFFFF; R#B+7: 每行中的搜索开始列, 0x00-0xFF, 一列中的字节位置; R#B+8: 每行中的搜索终止列, 0x00-0xFF, 仅当搜索匹配模式是 0x01 (任意位置匹配) 时有效; R#B+9: 搜索匹配后, 返回数据在搜索列的起始位置, 0x00-0xFF; R#B+A: 搜索匹配后, 返回的数据字节长度, 0x01-0xFF。 R#C 的 4 个寄存器规定了返回变量: R#C: 搜索标记, 0x00=未匹配, 0xFF=匹配; R#C+1: 匹配时的(行维度+1)值; R#C+3: 匹配时, 回传数据保存的寄存器首地址。 LIBSCH R10,R12,R23
擦除指定的字库	ERASE	<L_ID>	<L_ID>: 要擦除的汉字库 ID, 0x20-0x7F; 如果<L_ID>为 0x00, 表示字库位置由 R9 寄存器指定。 ERASE 40
字节累加校验和计算	SUMADD	R#S,R#T,R#N	对序列数据计算字节累加和校验 R#S: 输入的寄存器组; R#T: 输出 1 字节累加和结果, 8bit。 R#N: 序列长度寄存器, 8bit。 SUMADD R10,R80,R9
带进位字节累加校验和计算	SUMADDC	R#S,R#T,R#N	对序列数据计算带进位的字节累加和校验 R#S: 输入的寄存器组; R#T: 输出 1 字节累加和结果, 8bit。 R#N: 序列长度寄存器, 8bit。 SUMADDC R10,R80,R9



异或校验和计算	SUMXOR	R#S,R#T,R#N	<p>对序列数据计算字节异或校验</p> <p>R#S: 输入的寄存器组;</p> <p>R#T: 输出 1 字节异或结果, 8bit。</p> <p>R#N: 序列长度寄存器, 8bit。</p> <p>SUMXOR R10,R80,R9</p>
HEX 转换成压缩 BCD 码	HEXBCD	R#S,R#T,<MOD>	<p>把 HEX 数据转换成压缩的 BCD 码, 比如数据 1000 将转换成0x10,0x00。</p> <p>R#S: 输入HEX数据的寄存器组首地址</p> <p>R#T: 输出压缩BCD码数据的寄存器首地址</p> <p><MOD>: 高4bit表示输入HEX字节数, 0x01-0x08 低4bit表示输出BCD码字节数, 0x01-0x0A</p> <p>HEXBCD R10,R80,0x23</p>
压缩 BCD 码转换成 HEX	BCDHEX	R#S,R#T,<MOD>	<p>把压缩的 BCD 码转换成 HEX 数据, 比如数据 0x1000 将转换成0x3E8 (1000)。</p> <p>R#S: 输入压缩 BCD 码的寄存器组首地址</p> <p>R#T: 输出 HEX 数据的寄存器首地址</p> <p><MOD>: 高 4bit 表示输入 BCD 码字节数, 0x01-0x0A 低 4bit 表示输出 HEX 字节数, 0x01-0x08</p> <p>BCDHEX R10,R80,0x32</p>
ASCII 字符串转 HEX 字符	ASCHEX	R#S,R#T,<LEN>	<p>把 ASCII 字符串转换成 64bit 带符号 HEX 数据。</p> <p>R#S: 输入 ASCII 字符串寄存器首地址;</p> <p>R#T: 输出 HEX 数据, 64bit 寄存器;</p> <p><LEN>: ASCII 字符串数据长度, 包括符号位和小数点, 0x01-0x15。</p> <p>ASCHEX R10,R80,0x05</p>
到 COM1_Rx_FIFO 读取DL/T645 数据帧	RD645	R#A,R#T,R#C	<p>检查 COM1 接收 FIFO 是否有满足要求的 DL/T645 的数据帧, 如果有则读取数据到寄存器并清空接收 FIFO。</p> <p>R#A: 保存 6 字节地址 (LSB 排列, 压缩 BCD 码) 寄存器组。</p> <p>R#C: 返回状态的寄存器, 寄存器保存了返回数据; 0x00 表示未接收到匹配的 DL/T645 数据帧; 0xFF 表示接收到匹配的 DL/T645数据帧并把数据读取到 R#T 寄存器组。</p> <p>R#T: 结果寄存器, 匹配后, 保存 DL/T645 数据的寄存器组, 数据格式如下: 控制码+数据长度+数据</p> <p>RD645 R10,R20,R16</p>
时间计算函数	TIME	R#A,R#B,<MOD>	<p>R#A、R#B: 保存 6bytes 时间变量的寄存器, 时间变量为 BCD 格式;</p> <p>MOD=0, 计算 A=A-B, 计算两个时间的之间的相对值。 A 必须大于 B, 当 A<B 时, 不计算并返回 R#A 第一个字节为 0xFF。</p> <p>MOD=1, 计算 A=B-RTC;</p> <p>MOD=2, 计算 A=RTC-B。</p> <p>TIME R0,R10,0</p>
增加显示变量	ADDL14	R#A,R#B,<MOD>	<p>R#A: 保存 1 条显示变量 (32Bytes) 的寄存器;</p> <p>R#B: 显示变量的添加位置, 0x00-0x1F, 最多添加 32 个显示变量。</p> <p><MOD>:</p> <p>0x5A=添加到指定位置</p>

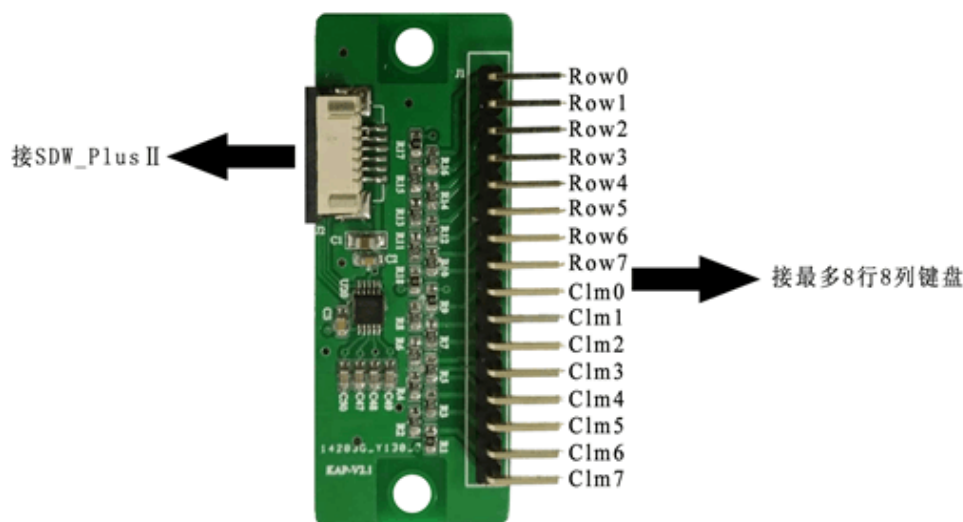


			其它=删除指定位置，此时 R#A 无定义。 ADDL14 R80,R79,0x5A
平方根计算	SQRT	R#A,R#B	计算一个 64 位无符号数 R#A 的平方根并保存到 R#B 中。 R#A: 保存了 8 Byte 无符号数; R#B: 保存了 4 Byte 无符号数结果。 SQRT R80,R90
输入法缓冲区添加	SCANADD	R#A,R#B	文本输入法状态下，把 R#A 指向、字节长度由 R#B 定义的字符串由当前光标位置添加到输入字符缓冲区，字符串中的 0xFF 将会被忽略。 SCANADD R80, R90 只在文本输入法状态下有效。
FEC 编码	FECEN	R#A,R#B,R#C	对 R#A 指向，字节（Byte）长度为 R#C 的数据串进行 FEC 编码，编码输出保存在 R#B 指针位置。 FECEN R80, R100, R10 注意，FEC 编码会把 1Byte 原始数据编码为 2Byte 编码数据。
FEC 解码	FECDE	R#A,R#B,R#C	对 R#A 指向，字（Word）长度为 R#C 的数据串进行 FEC 解码，解码输出保存在 R#B 指针位置。 FECDE R80, R100, R10
程序结束	END		ViewTech_OS 程序运行结束指令。 END



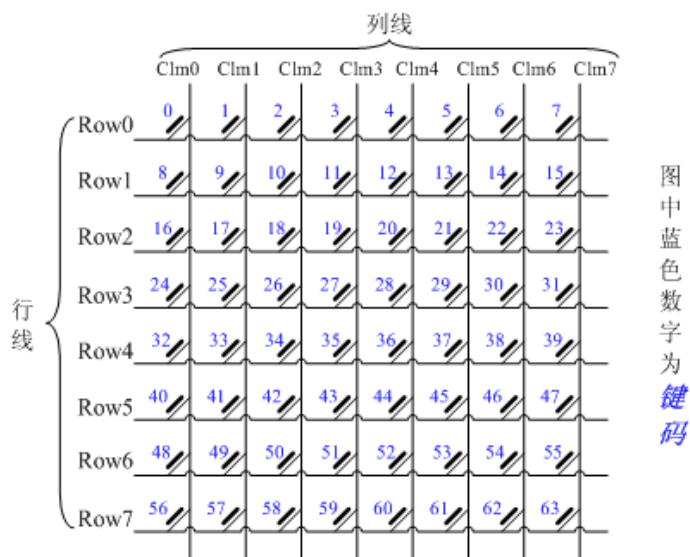
附 1 外接键盘转接板 KAP02

SDW-PlusII 系列串口屏很多型号都提供有键盘接口，如图 1-2 所示，该接口通过转接板 KAP02，可以外接最多 8 行*8 列键盘。



图附 2-1 KAP02 实物图

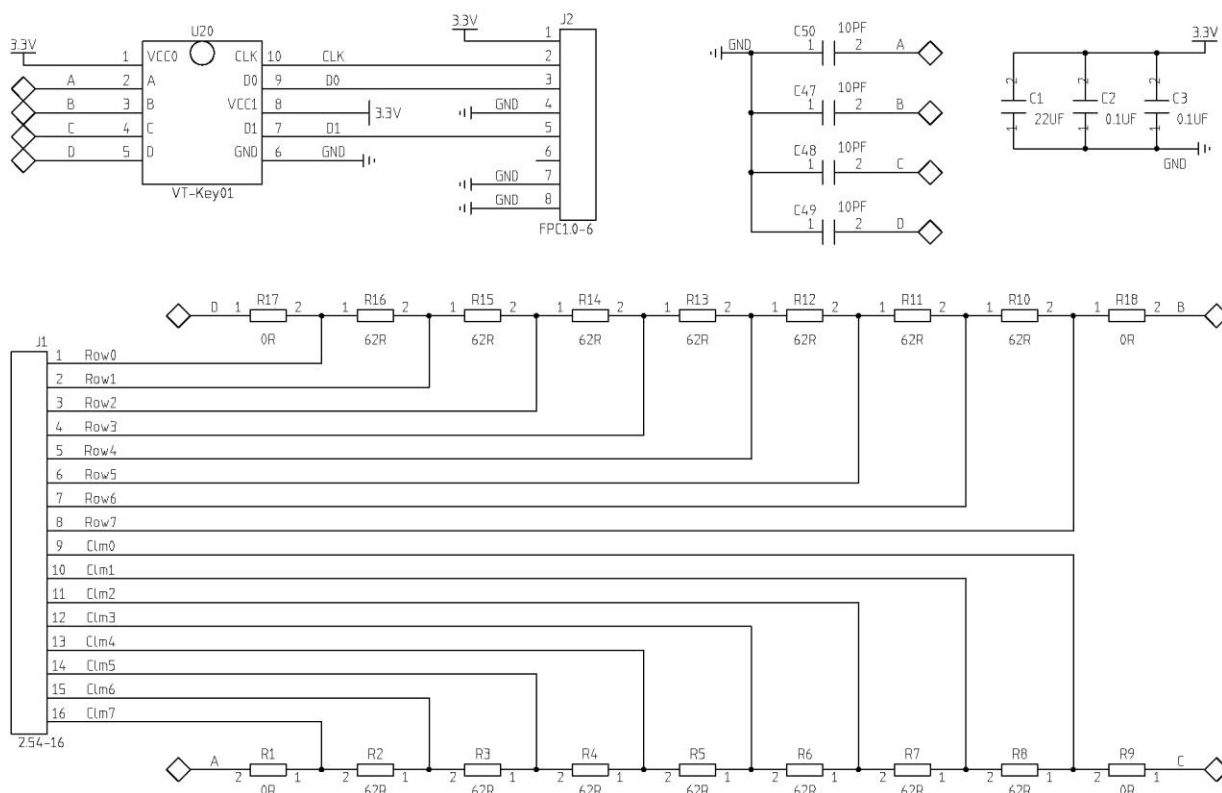
如上图所示，转接板 KAP02 提供两个插座，一个是 6PIN 的 1.0 间距 FPC 插座，用于连接到 SDW-PlusII 串口屏上，另一个是 16PIN 的 2.54 间距单排弯针，用于外接最多 8 行 8 列键盘。



图附 2-2 键盘行列线连接及键码分配

键盘行列线连接关系及键码如上图所示。每个行线和列线交叉的位置都是一个按键，最多共 64 个按键，键码分别对应为 0~63。

用户在通过 VGUS4.3 组态软件设计定义按钮的时候，通过指定键码，就可以将特定按钮与外接键盘关联起来。



图附 2-3 KAP02 电路原理图

用户需要用到外接键盘的时候，可以联系相关人员选配 KAP02 转接板，也可以根据图附 2-3 的电路原理图自行制作。