

KISS 指南

深圳卡迪电子有限公司

2017/7/7
V1.1

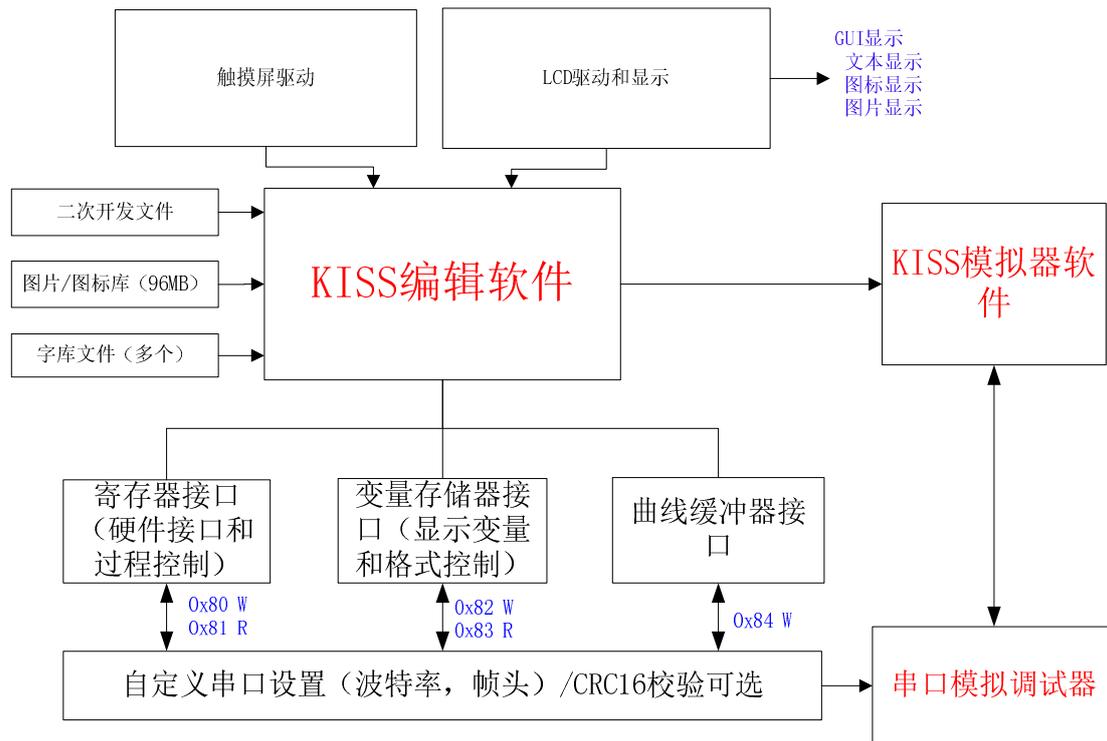
1. 目录

2. 概述.....	5
2.1 KISS 屏基本架构.....	5
2.2 KISS 屏主要特点.....	7
3. KISS 软件操作说明.....	8
3.1 工程建立及编辑.....	8
3.2 控件使用说明.....	13
3.2.1 基本触控.....	13
3.2.2 按键返回.....	15
3.2.3 弹出菜单.....	16
3.2.4 数据录入.....	17
3.2.5 增量调节.....	错误! 未定义书签。
3.2.6 拖动调节.....	19
3.2.7 滑动进度.....	错误! 未定义书签。
3.2.8 RTC 设置.....	错误! 未定义书签。
3.2.9 ASCL 录入.....	错误! 未定义书签。
3.2.10 GBK 录入.....	错误! 未定义书签。
3.2.11 变量图标.....	错误! 未定义书签。
3.2.12 动画图标.....	错误! 未定义书签。
3.2.13 艺术字.....	错误! 未定义书签。
3.2.14 图片动画显示.....	错误! 未定义书签。
3.2.15 数据变量显示.....	错误! 未定义书签。
3.2.16 文本显示.....	错误! 未定义书签。
3.2.17 位变量.....	错误! 未定义书签。
3.2.18 RTC 显示.....	错误! 未定义书签。
3.2.19 二维码.....	错误! 未定义书签。
3.3 图标库操作指南.....	32
4. 串口指令操作.....	33
4.1 帧架构.....	33
4.2 指令集.....	33
4.3 控制寄存器说明.....	34

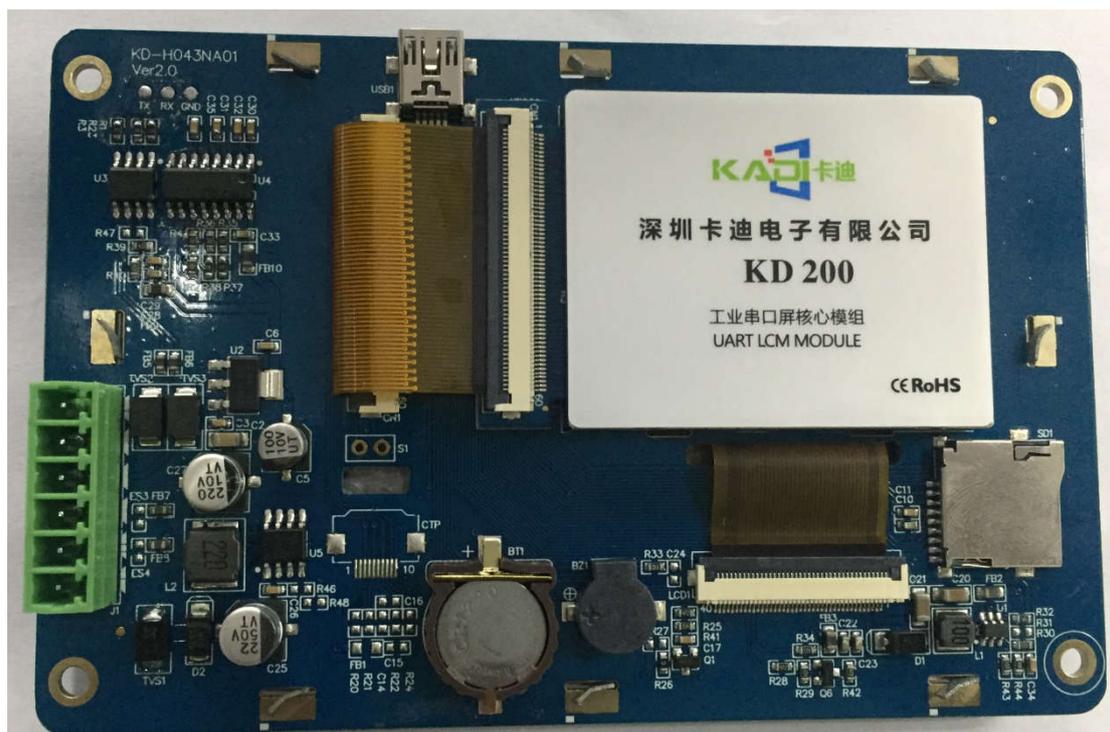
2. 概述

2.1 KISS 屏基本架构

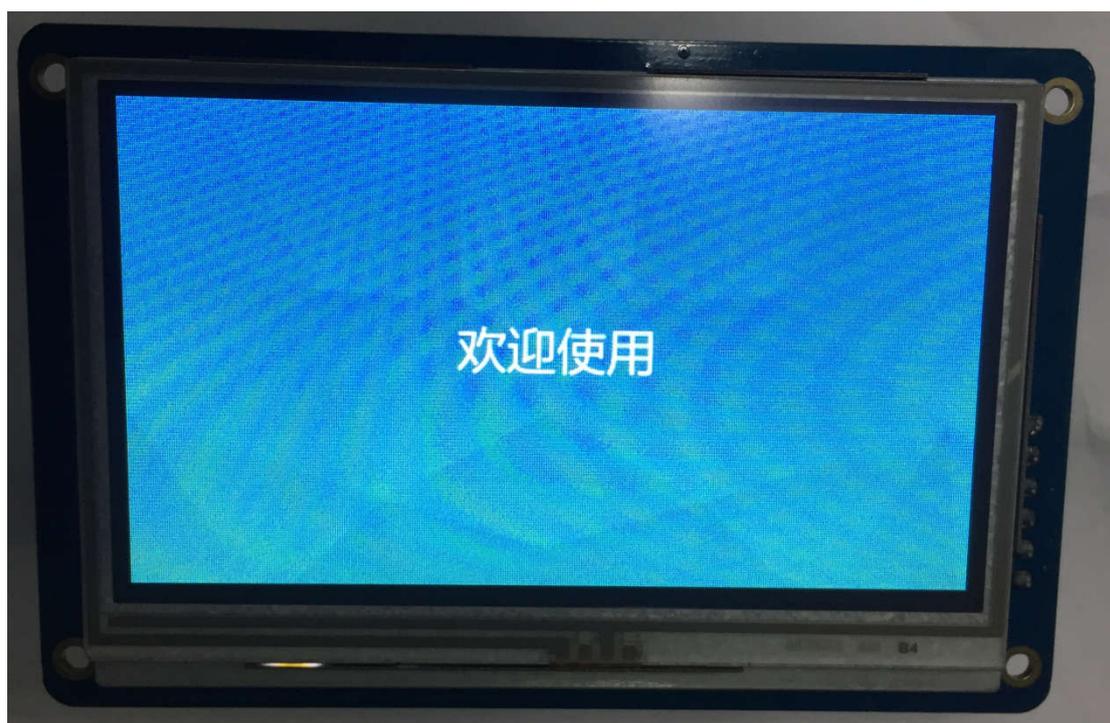
KISS (KADI Industry Support System) 是深圳卡迪电子有限公司开发的基于卡迪核心模块系列 (KD200/KD320/KD700/KD1200 等) 所设计的智慧型图形界面人际系统软件, KISS 软件主要由三部分组成: 编辑器软件, 模拟器软件和串口调试模拟器。编辑器软件用于 HMI 界面编辑并生成模块需要的文件, 模拟器软件可以在 PC 上直接模拟界面的操作, 串口模拟调试器则模拟一串口与模拟器进行通信。KISS 软件整体架构如下:



一个典型的 KISS 屏硬件一般有 KADI 核心板, TFT LCD, 触摸屏 (电容/电阻), 串口转换模块 (TTL/RS232/RS485), 高效电源转换电路, RTC 电路 (含电池), 蜂鸣器, USB 接口, SD 卡槽, 铁框等组成。如下图所示 (KD043NWR01B-480272):



背面



正面

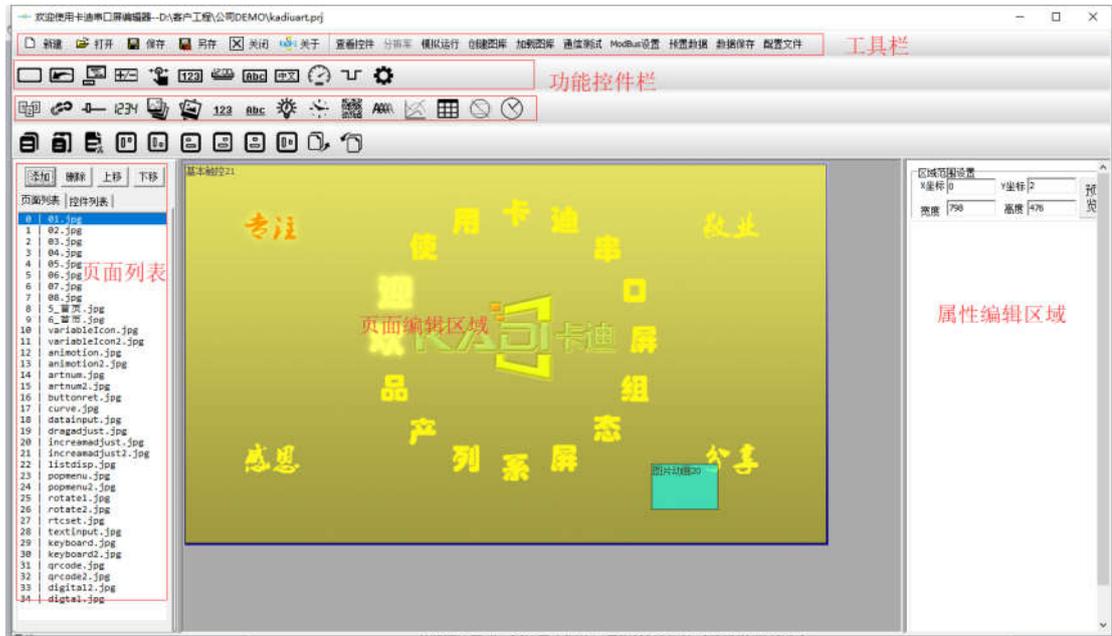
2.2 KISS 屏主要特点

- 把 GUI 分解成控件并按页面来配置，控件显示直接由变量控制。在通过 PC 软件配置好控件地址后，用户仅需要通过串口修改变量值即可实现控件的相应改变；
- 带 SD 卡接口，FAT32 格式，用户可通过 SD 卡来批量更新显示内容，便于生产管理；
- 内置 128MB 存储器，最大可扩展到 4GB，可存储海量图片，字库和图标；
- 集成 RTC 功能，自带电池保证 3 年以上供电使用，精度小于 1 秒/周，带农历功能；
- 256 字节配置寄存器空间，串口指令读写，便于硬件控制及操作；
- 可靠的高性能硬件平台，运行速度快，性能优越，质量可靠；

3. KISS 软件操作说明

3.1 工程建立及编辑

3.1.1 点击 KISS 软件，出现如下界面

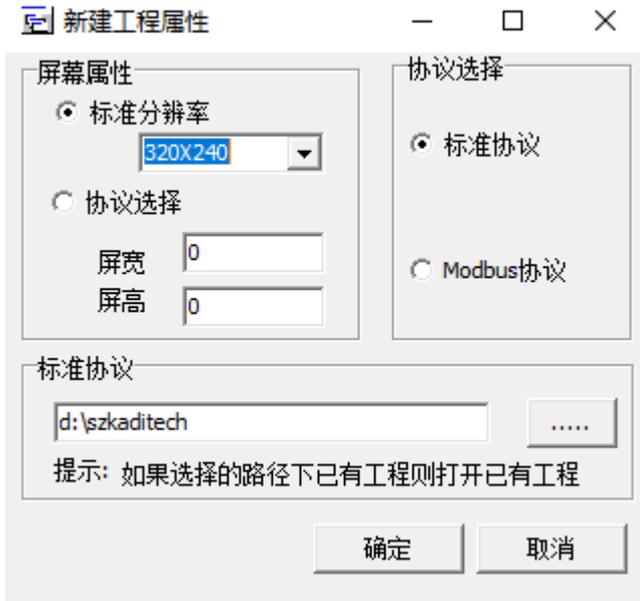


软件上半部份分为三个区域：

- 功能控件区域：文件基本操作，包括打开关闭，以及模拟器调用等
- 界面编辑控件区域：包含页面编辑需要使用的显示控件和触控控件
- 编辑控件区域：控件复制删除对齐等操作

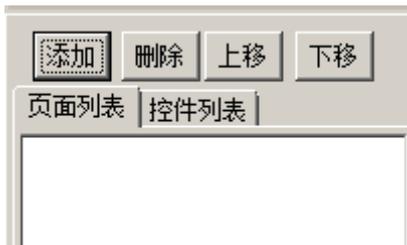
下半部份从左到右分别是页面列表，页面编辑和控件属性编辑。

3.1.2 点击“新建”按钮弹出以下界面：



选择屏幕分辨率以及使用的协议格式。分辨率支持标准分辨率和用户自定义分辨率，用户自定义分辨率最大支持 1280x800。通信协议支持我司标准协议，以及 Modbus 协议。

3.1.3 点击页面列表上方的添加，将所有设计好的页面添加到工程中：



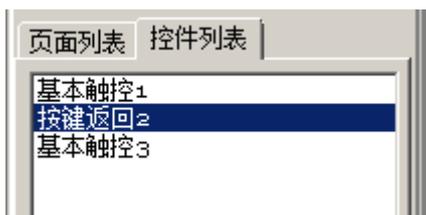
添加完成后，显示如下：

页面列表		控件列表	
位置	文件		
1	0 欢迎使用.bmp		
2	1 空闲, 请插枪.bmp		
3	2 已连接, 开始充电.		
4	3 充电方式选择.bmp		
5	4 充电模式选择.bmp		
6	5 按金额充.bmp		
7	6 按电里充.bmp		
8	7 按时间充.bmp		
9	8 刷卡启动.bmp		
10	9 输入密码.bmp		
11	10 密码输入错误.bmp		
12	11 充电启动中.bmp		
13	12 启动失败.bmp		
14	13 充电中信息.bmp		
15	14 刷卡后告警.bmp		
16	15 刷卡前告警.bmp		
17	16 结账确认.bmp		
18	17 刷卡结束充电.bmp		
19	18 充电结束.bmp		
20	19 数据输入键盘.bmp		
21	20 ASCII码输入键盘.		
22	21 屏保.bmp		
23	22 屏保.bmp		
24	23 屏保.bmp		
25	24 查看充电信息.bmp		

添加时需注意:

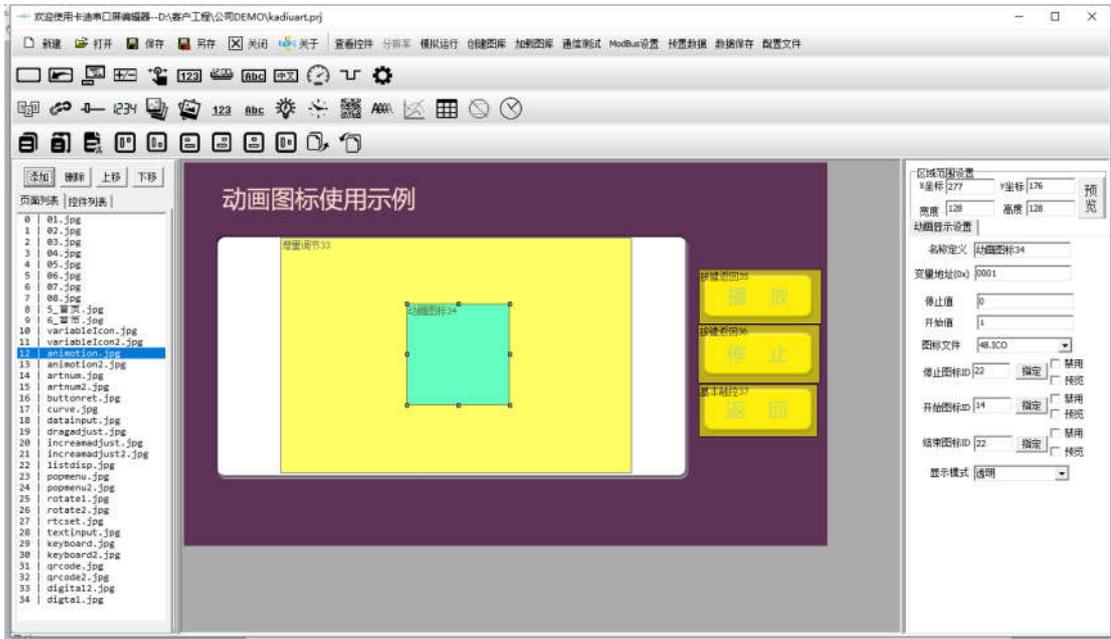
- 页面图片大小最好跟设定的分辨率一致, 否则界面显示会比较模糊;
- 支持图片格式为 jpg、 bmp 和 RGB565 格式数据, 选择一项之后, 所有图片格式必须一致;
- 图片名称按照整数顺序自动排列, 文件名称左边的数字即为页面 ID, 请按顺序编排好文件名

3.1.4 点击页面旁边的控件列表, 可以看到每个页面所使用到的控件, 方便用户选择。



3.1.5 点击不同控件然后拖到页面上, 通过设置控件属性, 即可编辑页面, 一般一个编

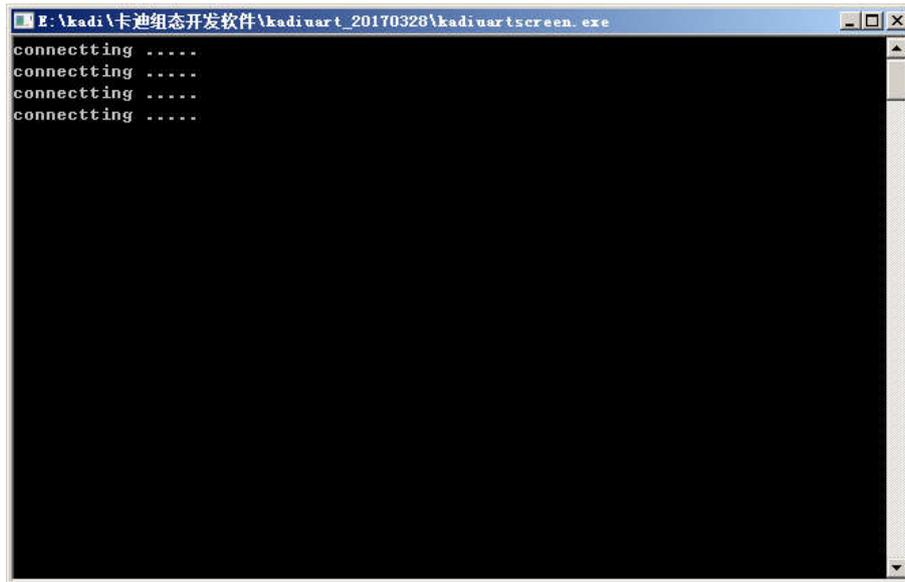
辑好的页面如下图：



3.1.6 页面编辑完成后，点击保存。然后点击菜单栏上的“模拟运行”，即可调用模拟器进行仿真调试，然后根据仿真的结果修改页面。



仿真的同时，软件也会输出一些信息，方便开发人员调试。



3.1.7 点击菜单栏上的“通信测试”，则可以调用串口模拟仿真器，配合模拟器一起，基本上可以完成用户 UI 的全部设计。

3.1.8 当仿真调试最终完成后，将生成的文件全部拷到 SD 卡，然后插到模块上，模块即将文件自动升级到内存中。生成的文件如下图所示：

名称	修改日期	类型	大小
Kadi_Pic	2017/4/5 18:23	文件夹	
Kadi_Res	2017/4/5 18:08	文件夹	
kadiuart.prj	2017/4/5 18:23	DxDesigner Pro...	44 KB



3.2 控件使用说明

控件的操作主要通过配置控件属性来实现。属性一般包括三个部分：

- 基本配置：位置，大小，颜色等；
- 显示及切换操作；
- 地址键值设置；

具体到每种控件，以下作详细介绍。

3.2.1 基本触控

区域范围设置

X坐标 31 Y坐标 14 预览

宽度 423 高度 250

基本触控

名称定义: 基本触控

按钮效果:

-1 指定

无按钮效果

页面切换:

2 指定

不进行页面

键值(0x)

0000 指定

按钮效果：点击“指定”弹出图库选择栏，选择按下后显示的效果，如果勾选了“无按钮效果”，则控件按下后无效果反馈。默认值为“-1”，代表无按钮效果。

页面切换：点击“指定”弹出图库选择栏，然后选择跳转的页面，如果勾选了“不进行页面切换”，则控件按下后无效果反馈。默认值为“-1”，代表不进行页面切换。

键值：控件 ID，可以根据通信协议通过串口来操作控件

注：当设计键盘输入时，所有的按键都采用基本触控实现，键盘按钮的有效键值为：0x0030~0x0039 (0~9)，0x002E (.)，0x002D (±)，0x00F0 (取消)，0x00F1 (确认)，0x00F2 (退格)。键值可以点击指定，然后通过跳出的键值编辑界面选择对应的按键，也可以自己编辑数值。界面如下：

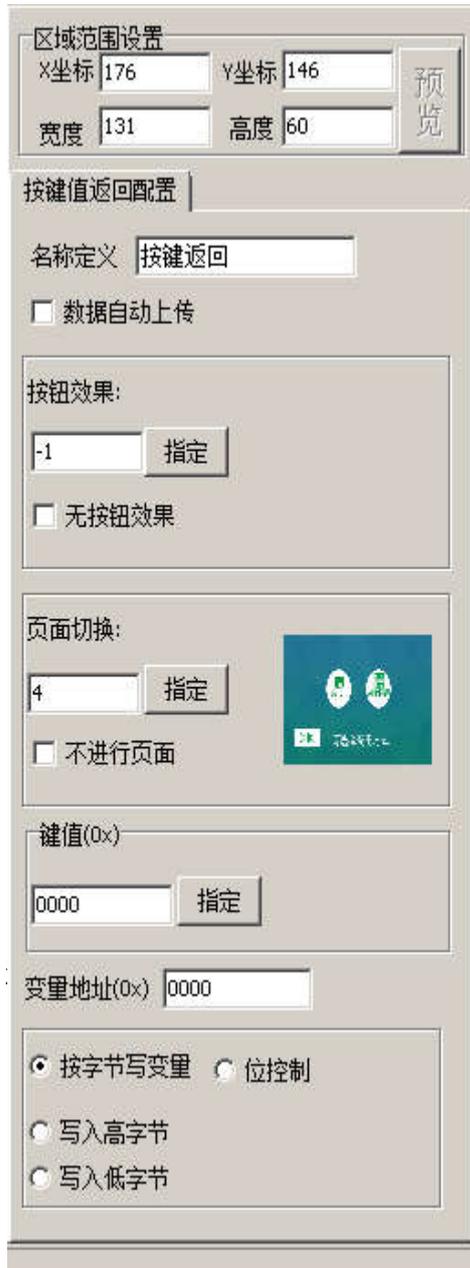


当选择指定的按键后，会自动返回一对应的键值，一般键值使用 16 位数据，当使用字母键盘输入时，字母键高字节对应大写字母 ASCII 值，低字节对应小写字母 ASCII 值，如果键盘中有定义 CapsLock 键，按下该按键后会自动切换大小写显示。比如选择键码 ‘d’，则返回键值 0x4464，使用 ASCII 录入控件时，点击 ‘d’ 显示 ‘d’，如果先按下 CapsLock 键，则点击 ‘d’ 显示 ‘D’。

如果右边键码定义不能满足需求，也可以使用自定义方式。在命令中输入自定义键值，添加描述，然后保存，选择后点击 ok 即可。如下即为自定义一个 F1F1 键值示例。



3.2.2 按键返回



区域范围设置：设置控件的位置和大小；也可以通过鼠标进行操作当选择数据自动上传后，自动将键值和地址通过串口发送出去

按钮效果：点击“指定”跳出图库选择栏，然后选择按下后显示的效果，如果勾选了“无按钮效果”。则控件按下后无效果反馈。默认值为“-1”，代表无按钮效果。

页面切换：点击“指定”跳出图库选择栏。然后选择跳转的页面，也可以直接输入页面 ID；如果勾选了“不进行页面切换”，则控件按下后无效果反馈。默认值为“-1”，代表不进行页面切换。

键值：写入该控件地址的数据。

变量地址：数据存储地址。

数据写入方式：

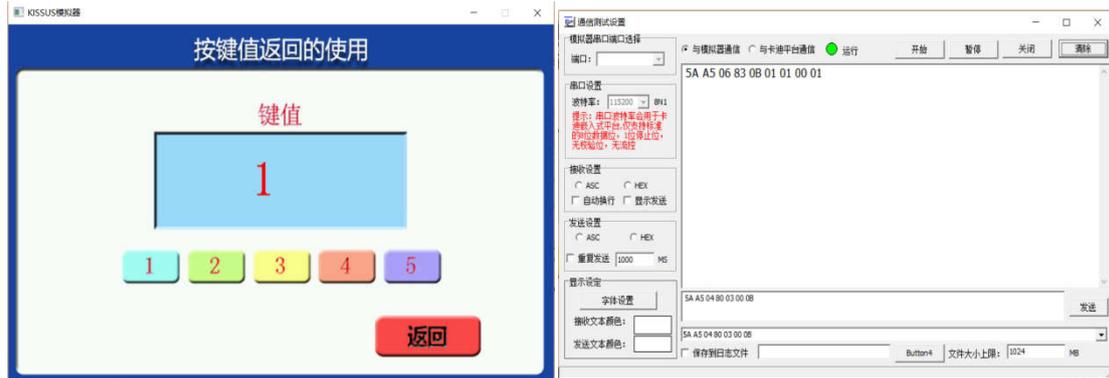
按字节写入：把键值写入变量地址。

写入高字节：把键值的低位写入变量的高位。

写入低字节：把键值从低位写入变量的低位。

位控制：将键值的最低位写入变量地址的指定位。

如触摸的左图 1 键，返回的信息如下



3.2.3 弹出菜单

弹出菜单

名称定义

数据自动上传

按钮效果

无按钮效果

使用图库

变量地址

按字节写变量 位控制

写入高字节

写入低字节

弹出菜单设置

所在页面:

菜单区域

--

显示位置

执行函数

函数名称 使用

变量地址: 定于数据存储地址, 以及数据存储

弹出菜单设置: 选择点击控件后需要显示的内容, 可以在图片库中选择, 也可以通过输入页面和区域位置参数来设置。

弹出菜单主要用于实现一些按键按下后需要 确认取消动作的弹出框, 下面为弹出菜单 的实现示例:

变量地址: 数据存储地址。

数据写入方式:

按字节写入: 把键值写入变量地址。

写入高字节: 把键值的低位写入变量的高位。

写入低字节: 把键值从低位写入变量的低位。

位控制: 将键值的最低位写入地址的指定位。

所在页面: 弹出菜单所在页面 ID。

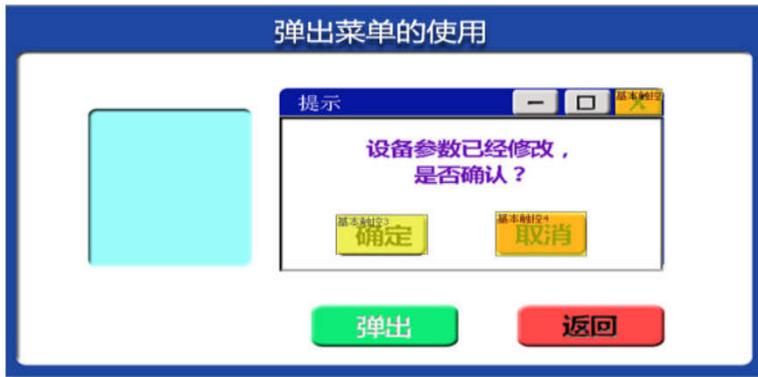
菜单区域: 弹出菜单在所在页面的区域范围。

显示位置: 菜单在当前页面显示的位置。

函数名称: 二次开发的伪指令跳转到的功能函数

注释: 弹出菜单上只能做基本控件。

将作图配置完出现的效果图如下。



图弹出菜单效果图

3.2.4 数据录入

数据录入

名称定义:

数据自动上传

按钮效果:

无按钮效果

使用图库

页面切换

不进行页面切换

变量地址(0x)

变量类型

整数位数 小数位数

显示位置

显示颜色

字库位置 范围23~127

字体大小 范围4~255

光标颜色

输出显示方式

键盘位置

键盘所在页面

键盘区域 ~

显示位置

启用范围设置

最小值 最大值

自带显示属性

执行函数

函数名称 使用

按钮效果: 按钮按下效果图选择, -1 为默认值, 表示无动画

页面切换: 按钮按下后切换页面选择, -1 为默认值, 表示不进行页面切换

变量地址: 数据在存储器中的位置

显示颜色: 数据显示的字体颜色

字库位置: 选择所使用的字库

字体大小: 设置使用的字体大小

对齐方式: 设置数据在显示区域中的方式, 分左对齐和右对齐两种

变量类型:

0x00 = 整数(字)

0x01 = 长整数(双字)

0x02 = 无符号字节参数(变量指针高字节)

0x03 = 无符号字节参数(变量指针低字节)

整数/小数位数: 设置数据整数/小数部分的长度

变量单位: 设置长度及单位字符串

显示方式: 直接显示或者显示*号

键盘位置: 当前页或是其他页

键盘所在页面: 选择键盘所在页面和键盘区域

键盘区域: 键盘所在页键盘的区域

显示位置: 键盘在当前页的显示位置

启用范围设置: 是否启用范围设置

自带显示属性: 从左到右输入, 从右到左输入

函数名称: 二次开发的伪指令跳转到的功能函数

注: 变量录入调用的键盘要做基本触控, 如 0-9 对应的返回值为 0x0030-0x0039, 0x00F1(确定), 0x00F0(取消), 0x00F2(退格)

录入数据后通过”数据变量显示”将数据显示

将左图配置完成后, 即可实现如下图功能



数据录入显示效果图

3.2.5 增量调节

增量调节

名称定义

数据自动上传

按钮效果:



无按钮效果

使用图库

变量地址(0x)

按字节写变量 位控制

写入高字节

写入低字节

调节方式

逾限处理方式

调节步长

最小值:

最大值:

按键效果:

二次开发功能函数

函数名称: 使用

数据自动上传: 勾选后, 数据通过串口自动上传
按钮效果: 按钮按下效果图选择, -1 为默认值, 表示无动画

页面切换: 按钮按下后切换页面选择, -1 为默认值, 表示不进行页面切换

变量地址: 数据存储地址。

数据写入方式:

按字节写入: 把键值写入变量地址。

写入高字节: 把键值的低位写入变量的高位。

写入低字节: 把键值从低位写入变量的低位。

位控制: 将键值的最低位写入地址的指定位。

调节方式: 自加和自减

处理方式: 循环/停止

调节步长: 增加或者减少的步长 (0-32767)

最小值: 增量调节的最小值

最大值: 增量调节的最打字

按键效果: 按住连续调节/按住只调节一次

函数名称: 二次开发的伪指令跳转到的功能函数

调节后可通过数据变量显示将数据显示出来
 将左图配置完成后, 即可实现如下图的功能



如图增量调节效果图

3.2.5 拖动调节

拖动调节

名称定义

数据自动上传

变量地址(0x)

拖动方式

数据返回格式

拖动方式

起始位置对应的返回值

终止位置对应的返回值

执行函数

函数名称: 使用

数据自动上传: 勾选后数据通过串口自动上传

变量地址: 数据存储地址。

数据返回格式: 调节字地址/调节高字节/调节低字节。

拖动和方式: 横向/纵向

函数名称: 二次开发的伪指令跳转到的功能函数

注: 拖动调节主要用于滑动操作显示, 跟“滑动进度”配合使用, 可实现横向和纵向的滑动操作
下面介绍滑动进度来显示效果图

滑度刻度指示设置

名称定义

变量地址(0x)

起始变量刻度值

终止变量刻度值

刻度模式

起始与终止刻度坐标, 系统自动默认为配置按钮的区域范围

图标文件

滑动图标 禁用 预览

显示模式

图标显示(Y)坐标值

图标前移偏移量 (0~255)

VP模式

3.2.6 滑度刻度

变量地址: 指定数据存储的位置

起始/终止刻度值: 设置滑动起点和终点对应的数值

刻度模式: 选择横向/纵向刻度

图标文件: 选择要用到的图标库

滑动图标: 从图库中选择滑动块使用的图标

显示模式: 滑动时显示背景/透明

坐标偏移量: 滑动一步前进的数值, X 对应横向滑动, Y 对应纵向滑动

VP 模式:

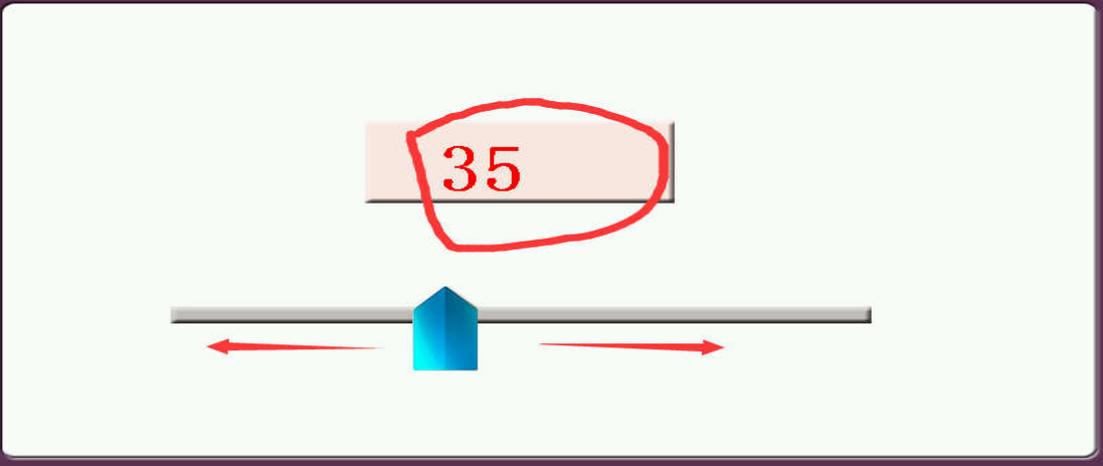
指向一个整型变量

指向一个整型变量的高字节

指向一个整型变量的低字节

将左图配置完成后, 即可实现如下图的功能

拖动调节使用示例



拖动和滑度使用的效果图

RTC设置

名称定义

数据自动上传

按钮效果:

无按钮效果

使用图库

显示位置:

显示颜色

字库位置 范围:0 , 23~127

字体大小 范围:4~255

光标颜色

键盘设置

所在页面 

键盘区域 --

显示位置

二次开发功能函数

函数名称: 使用

3.2.7 RTC 设置

数据自动上传: 勾选后, 数据通过串口自动上传

按钮效果: 按钮按下效果图选择, -1 为默认值, 表示无动画

显示位置: 设置时间键盘现在在当前的页面区域

显示颜色: 数据显示的字体颜色

字库位置: 选择所使用的字库

字体大小: 设置使用的字体大小

光标颜色: 黑色/白色

变量单位: 设置长度及单位字符串

显示方式: 直接显示或者显示*号

所在页面: 选择键盘所在页面和键盘区域

键盘区域: 键盘所在页键盘的区域

显示位置: 键盘在当前页的显示位置

函数名称: 二次开发的伪指令跳转到的功能函数

将左图配置完成后, 即可实现如下图的功能



RTC 设置效果图

ASCII录入

名称定义

数据自动上传

按钮效果：

无按钮效果

使用图库

页面切换：

不进行页面切换

变量地址(0x)

文本长度

录入模式

字库位置 范围:0, 23~127

X方向点阵数 范围:4~255

Y方向点阵数 范围:4~255

光标颜色

文本颜色

输入状态返回

录入显示区域
 --

键盘位置

键盘设置

所在页面

键盘区域 --

显示位置

输入过程显示方式

二次开发功能函数

函数名称: 使用

3.2.8 ASCLL 录入

按钮效果：选择按钮按下时的效果图，-1 为默认值，表示按下无效果

页面切换：选择按钮按下时切换到的页面，-1 为默认值，表示不切换页面

变量地址：数据在存储器中的位置

文本长度：设置数据的长度

录入模式：修改文本/重新录入

字库位置：选择所使用的字库

X/Y 方向点阵数：设置字符点阵大小

光标颜色：黑色/白色

文本颜色：设置字体颜色

输入状态返回：勾选时，在 (VP-1) 位置保存输入结束标记和有效数据长度

录入显示区域：文本录入过程中显示区域

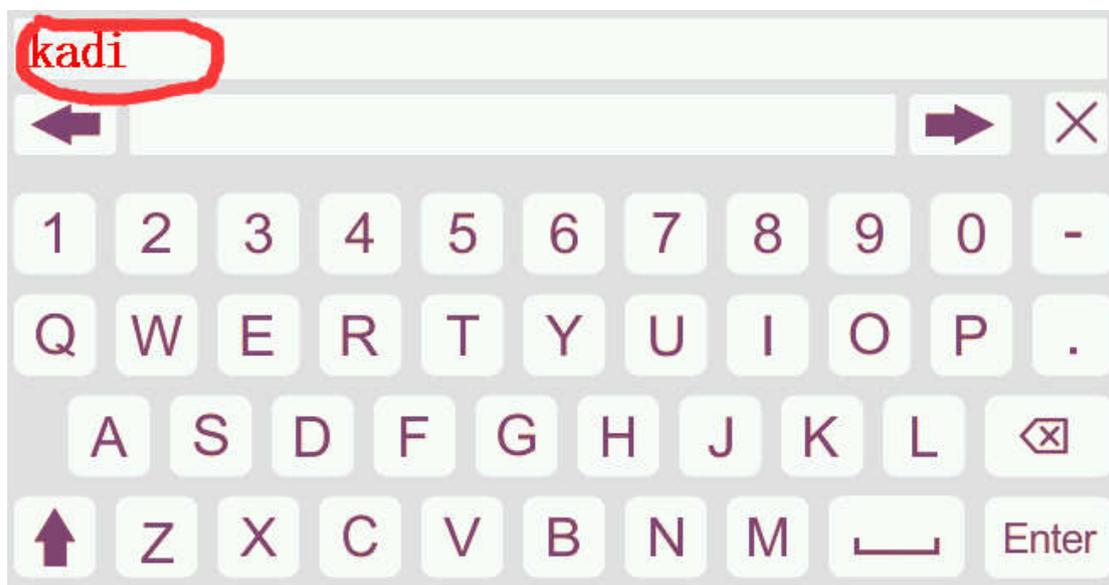
键盘位置：当前页/其他页 (**切换到其他页面选择**)

键盘设置：选择键盘所在的页面，区域，及显示位置

输入过程显示方式：正常显示/ '*' 替代显示

函数名称：二次开发的伪指令跳转到的功能函数

将左图配置完成后，即可实现如下图的功能



图为 ASCLL 录入效果图

3.2.9 GBK 录入

GBK录入

名称定义 GBK录入 56

数据自动上传

按钮效果:

-1 指定

无按钮效果

使用图库 48.ICO

页面切换:

-1 指定

不进行页面切换

变量地址(0x) 0100

文本长度 100

录入模式 修改文本

显示字库 24 范围:0, 23~127

点阵大小 24 范围:4~255

文本颜色 F800

录入工程显示字库 24 范围:0, 23~127

点阵大小 24 范围:4~255

文本颜色 F800

显示方式 拼音提示显示在上边

光标颜色 黑色

输入状态返回

录入显示区域

44 129 -- 761 197 指定

拼音显示位置 532 147 设置

显示间距 0 键盘位置 其他页

键盘设置

所在页面 23 

键盘区域 32 178 -- 765 463

显示位置 35 121 设置

执行函数

函数名称: 使用

按钮效果: 选择按钮按下时的效果图, -1 表示无效果, 如果选择“使用图库”, 则按下会显示图库中选择的图片

页面切换: 选择是否需要切换页面, -1 表示不切换。

变量地址: 数据在存储器中的位置

文本长度: 设置数据的长度

录入模式: 修改文本/重新录入

显示字库: 选择所使用的字库

点阵大小: 设置字符点阵大小

文本颜色: 字体显示颜色

录入工程显示字库: 录入过程中显示所使用的字库及点阵大小, 文本颜色

显示方式: 设置拼音是显示在上边还是左边

录入显示区域: 输入过程中拼音和汉字显示区域及起始位置

键盘设置: 选择所使用的键盘及显示位置

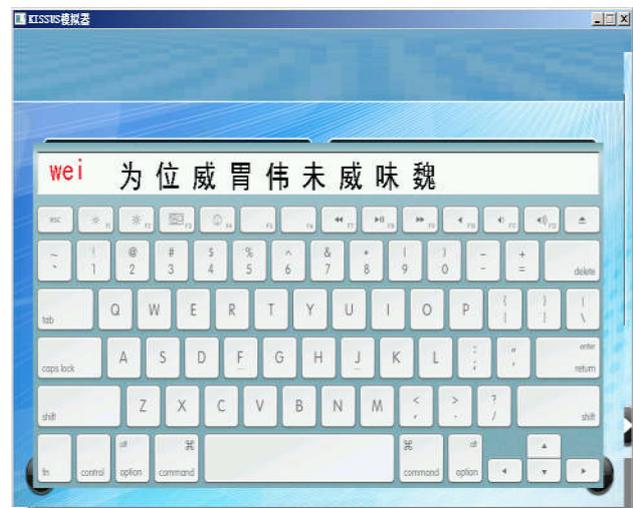
录入显示区域: 文本录入过程中显示区域

键盘位置: 当前页/其他页 (**切换到其他页面选择**)

键盘设置: 选择键盘所在的页面, 区域, 及显示位置

函数名称: 二次开发的伪指令跳转到的功能函数

将左图配置完成后, 即可实现如下图的功能



3.2.9 变量图标

变量图标

名称定义

描述指针(0x)

变量地址(0x)

图标文件

最小值:

对应的图标 禁用 预览

最大值:

对应的图标 禁用 预览

显示模式

初始值

区域设置: 图标显示的位置和大小

变量地址: 数据存储的位置

图标文件: 选择使用的图标库

变量上限/下限: 定义变量显示的范围, 越界不显示

对应的图标: 变量上下限对应的图标 ID

显示模式: 显示背景/透明

注: 当最小值为 0 时显示的图标为 24

当最大值为 1 时显示的图标为 25

将左图配置完成后, 即可实现如下图的功能



变量图标效果图, 通过点击切换图标可以实现按钮图标的切换

3.2.10 动画图标

动画显示设置

名称定义

变量地址(0x)

停止值

开始值

图标文件

停止图标ID 禁用
 预览

开始图标ID 禁用
 预览

结束图标ID 禁用
 预览

显示模式

变量地址：数据存储的位置
停止值：变量为该值时显示固定图标
开始值：变量为该值时开始动画显示
图标文件：选择使用的图标库
停止图标：变量为停止值时显示的图标
开始/结束图标：循环播放动画所使用的图标范围（对应图标库 ID）
显示模式：显示背景/透明

注：起始图标的位置必须小于终止图标位置，在终止图标上也设置图标动画变量，可实现图标不间断播放，实现动画效果。发送串口指令或者其他控件设置寄存器停止值可停止播放动画。

艺术字变量显示

名称定义

变量地址(0x)

图标文件

ICON0 禁用
 预览

对应的ICON0_ID,排列顺序为0123456789

显示模式

变量类型

整数位数

小数位数

对齐方式

3.2.11 艺术字

变量地址：数据存储的位置
图标文件：选择使用的图标库
 ICON0:0 对应 ICON_ID 排列的顺序为 0123456789-1. :
显示模式：显示背景/透明
位数：设置整数位数和小数位数
对齐方式：左对齐/右对齐
 将左图配置完成后，即可实现如下图的功能



3.2.12 图片动画显示

图片动画显示

名称定义

起始位置

终止位置

显示时间 (8ms)

起始位置：设置动画开始的页面
终止位置：设置动画结束的页面
显示时间：页面显示间隔时间=设置值×8ms

注：设置完成后，则模块会从指定的页面开始，按设定的时间间隔，一帧一帧播放，直到终止页面。页面动画主要用于设计开机动画，一般在分辨率 480×272 时速度可以达到 20 帧。

3.2.13 数据变量显示

数据变量显示

名称定义

变量地址(0x)

显示颜色 

字库位置 范围:0 , 23~127

字体大小 范围:4~255

对齐方式

变量类型

整数位数

小数位数

变量单位长度

显示单位

单位字符串,ASCII编码

变量地址：数据在存储器中的位置
文本颜色：字体显示颜色
字体大小：显示数据的大小范围为 4~255
对齐方式：左对齐/右对齐
整数位数：显示整数的位数；
小数位数：显示小数的位数；
变量类型：整数(2 字节)
 长整数(4 字节)
 超长整数(8 字节)
 无符号整数(2 字节)
 无符号长整数(4 字节)
变量单位长度：变量单位（固定字符串）显示长度，0x00 表示没有单位显示。
显示单位：单位字符串，ASCII 编码

3.2.14 文本显示

文本显示

名称定义

变量地址(0x)

显示颜色

编码方式 0x00 = 8bit编码

字符间距不自动调整

文本长度

FONT0

编码方式为0x01~0x04时ASCII字库位置

FONT1

编码方式为0x00,0x05,以及0x01~0x04的非ASCII字符使用的字库

X方向点阵数 范围:4~255

Y方向点阵数 范围:4~255

水平间隔

垂直间隔

变量地址：文本变量数据存储的位置

显示颜色：文本字体颜色

编码方式：显色文本字符编码

0x00: ASCII 编码

0x01: GB2312 内码

0x02: GBK

0x03: BIG5

0x04: SJIS

0x05: UNICODE

文本长度：显示字符串长度

FONT0：编码方式 1~4 时 ASCII 字库位置

FONT1：编码方式为 0, 5 以及 1~4 的非 ASCII 字符使用的字库

X/Y 方向点阵数：设置字符的大小

水平间隔：设置字符间距离

垂直间隔：设置两行字符间距离

文本显示一般可用两种方式显示：通过 ASCII 录入或者数据录入，调用键盘输入显示；或者通过串口发送指令实现，以下示例为通过串口发送指令显示：

新建卡号窗口为文本显示，地址为 0x0701，然后运行模拟器，在串口输出窗口发送 5A A5 0F 82 07 01 31 32 33 34 35 36 37 38 39 30 41 42，这时可以在模拟器上看到卡号显示为 1234567890AB。同时在模拟器信息窗口上也可以看到接收到了数据。

3.2.15 位图标

区域范围设置

X坐标 Y坐标 预览

宽度 高度

位变量图标显示

名称定义

变量地址(0x)

Act_Bit_Set 1111111111111111

设置

显示模式

移动模式

移动间隔

图标文件

ICON05 指定

ICON0E 指定

ICON15 指定

ICON1E 指定

ICON Mode

变量地址：位变量指针，字变量

Act_Bit_Set：点击设置弹出“位控制设置”菜单，为 1 的 bit 说明变量地址对应位置需要显示

显示模式：0x01-0x07，见下表 1

移动模式：0x00-0x03，位图图标排列方式，见下表 2

移动间隔：下一个图标坐标移动间隔

图标文件：选择使用的图标库

ICONOS：不显示动画模式，bit0 图标 ID

显示动画模式，bit0 图标动画起始 ID

ICONOE：显示动画模式，bit0 图标动画结束 ID

ICON1S：不显示动画模式，bit1 图标 ID

显示动画模式，bit1 图标动画起始 ID

ICON1E：显示动画模式，bit1 图标动画结束 ID

ICON Mode：透明显示/显示背景



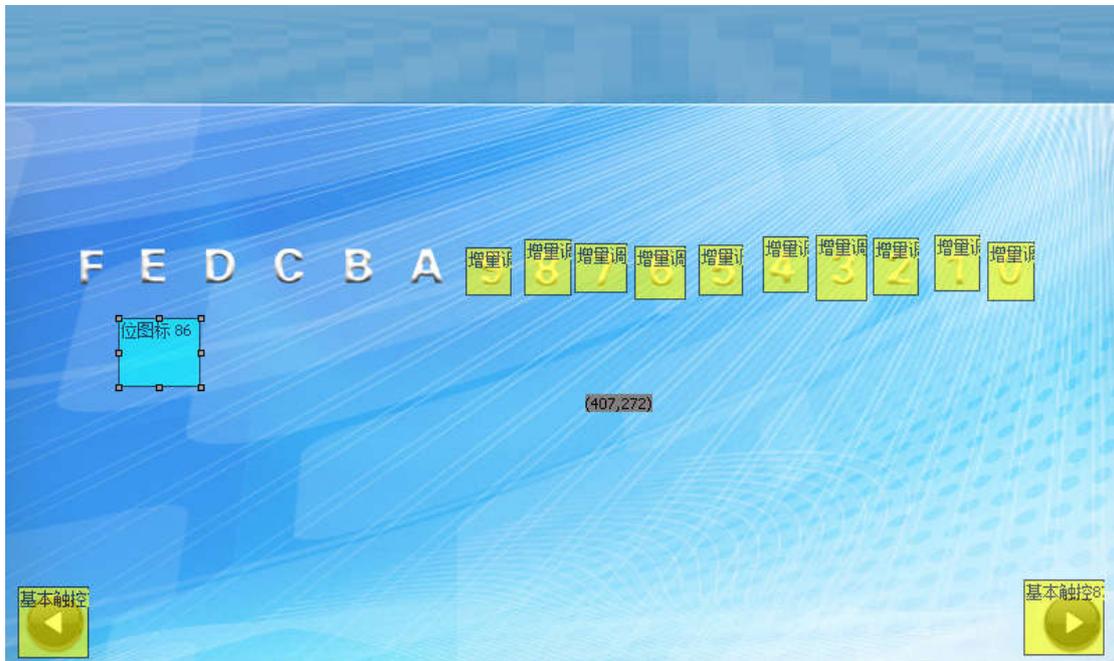
显示模式	Bit 值	
	0	1
0x00	ICONOS	ICON1S
0x01	ICONOS	不显示
0x02	ICONOS	ICON1S- ICON1E 动画
0x03	不显示	ICON1S
0x04	不显示	ICON1S- ICON1E 动画
0x05	ICONOS- ICONOE 动画	ICON1S
0x06	ICONOS- ICONOE 动画	不显示
0x07	ICONOS- ICONOE 动画	ICON1S- ICON1E 动画

表 1 显示模式值

移动模式	移动方向	位处理方式
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 保留移动间隔位置
0x03	Y++	Act_Bit_Set 指定的不处理 bit 保留移动间隔位置

表 2 移动模式值

如下图，建立一个位变量控件，设置使用 bit0~9，另外 0~9 数字分别建立一个增量调节控件，所有控件统一地址设置为 0x0220。增量调节控件设为位控制方式，分别控制 bit0~9，按下时对应位置一，再按一次清零。位控制属性按照上图设定。



模拟运行后，按下对应数字，可以看到对应的图标切换。调整显示模式和移动模式，可以看到不同的位显示方式。

3.2.16RTC 显示

RTC显示

名称定义:

字体颜色: ■

字库位置: 范围:0, 23~127

X方向点阵数: 范围:4~255

日期格式:

编码字符串:
使用RTC编码表和ASCII字符构成

说明	编码
公历_年	Y
公历_月	M
公历_日	D
公历_小时	H
公历_分钟	Q
公历_秒	S
公历_星期	W

假设当时时间是:
2012-05-02 12:00:00 星期三
Y-M-D H:Q将显示为 05-02 WED 12:00

- 字体颜色:** RTC 显示的字体颜色
- 字库位置:** RTC 显示所使用的字库
- 点阵数:** RTC 显示所使用的字体大小
- 日期格式:** RTC 显示格式，按公历显示

将左图配置完成后，即可实现如下图的功能



3.2.17 二维码显示

二维码

名称定义

变量地址(0x)

Unit_Pixels

变量地址：二维码所使用传输端口地址

单位像素：每个二维码单元像素所使用的物理点阵大小（0-7），比如设置为2，则每个单元占用2x2点阵大小。

属性设置好后，通过串口接收数据，然后生成二维码显示。

显示效果如下图



3.3 图标库操作指南

图标库主要是将页面之外所用到的图标，比如时钟指针，艺术字等通过打包处理后生成

lib 文件，供系统调用。在菜单栏点击创建图库 **创建图库**，出现以下画面：



点击添加将图标加载进来，通过上移下移功能调整图标位置（比如艺术字需按 0~9 顺序排列），然后点击创建，即自动在 PIC_RES 文件夹下生成对应名称的图标库。

图标支持 jpg, png, bmp 格式图片。

加载完成后一般显示如下：



4. 串口指令操作

4.1 帧架构

KISS 串口指令数据帧由 4 个数据块组成，如下表所述：

数据块	1	2	3	4	5
定义	帧头	数据长度	指令	数据	指令和数据的 CRC 校验
数据长度	2	1	1	N	2
说明	一般由两个字节组成，默认 5A A5	数据长度包括指令，数据和校验	0x80-0x84		可选择是否需要校验

4.2 指令集

功能	指令	数据	说明
访问控制寄存器	0x80	ADR (0x00-0xFF) +Data_Pack	指定地址写寄存器数据
	0x81	ADR (0x00-0xFF) +RD_LEN (0x00-0xFF)	指定地址度 RD_LEN 字节寄存器数据
		ADR (0x00-0xFF) +RD_LEN+Data_Pack	读寄存器的应答
访问数据寄存器	0x82	ADRH:L (0x0000-0x6FFF) +Data0~DataN	指定地址开始写入数据串（字数据）到变量存储区
		ADRH:L (0x0000-0x6FFF) +RD_LEN (0x00-0xFF)	从变量存储区指定地址开始读入 RD_LEN 长度字数据
	ADRH:L (0x0000-0x6FFF) +RD_LEN + Data0~DataN	读数据存储器的应答	
写曲线缓冲区	0x84	CH_Mode (Byte) +DATA0 (Word) +... DataN	写曲线缓冲区数据 CH_Mode 定义了后续数据的通道排列顺序： <ul style="list-style-type: none"> ➤ CH_Mode 的每个位 (nit) 对应 1 个通道； ➤ CH_Mode Bit0 对应 0 通道, bit7 对应 7 通道； ➤ Bit 置 1 表示通道数据存在； ➤ 数据按照低通道数据在前排列比如 CH_Mode=0x83 (1000011B), 表示后续格式为：(通道 0+通道 1+通道 7) ... (通道 0+通道 1+通道 7)

在前面的控件说明中，基本上每个控件都有一个变量地址，设置该地址后，即可用访问数据寄存器指令（0x82/0x83）来进行读写操作。

在 2.2.7 文本显示 控件说明中，我们有使用到串口发送指令来显示字符串，其指令为：

5A A5 0F 82 07 01 31 32 33 34 35 36 37 38 39 30 41 42

帧头 数据长度 15 指令 0x82 存储地址 0x0701 数据

PS: 该指令未使用 CRC 校验

4.3 控制寄存器说明

寄存器地址	定义	字节长度	说明
0x00	Version	1	版本号, BCD 码表示, 0x10 表示 1.0
0x02	BZ_TIME	1	蜂鸣器鸣叫控制寄存器, 单位 10ms
0x03	PIC_ID	2	读: 当前显示页面 ID 写: 切换到指定页面 (进程控制)
0x05	TP_Flag	1	0x5A=触摸屏坐标有更新 其他: 触摸屏坐标未更新
0x06	TP_Status	1	0x01: 第一次按下 0x02: 抬起 0x03: 一直按下 其他: 无效
0x07	TP_Position	4	触摸按下坐标位置, X_H:L, Y_H:L
0x0B	TPC_Enable	1	0x00: 触控不启用 其他: 触控启用 (上电默认 0xFF)
0x20	RTC_NOW	16	时间设置 YY:MM:DD:WW:HH:MM:SS + 天干地支+生肖
0x40	En_Lib_OP	1	0x5A 表示用户申请进行字库存储器操作, 操作完后清零
0x41	Lib_OP_Mode	1	0x50: 把指定的变量存储空间数据写入到指定字库空间 0xA0: 把指定字库空间的数据读入变量存储器空间
0x42	Lib_ID	1	指定的字库空间, 0x40-0x7F, 每个字库 128KW, 对应 Flash 空间为 8MW (16MB)
0x43	Lib_Address	3	指定字库空间的数据操作首地址: 0x0000-0x6FFF
0x46	VP	2	指定变量存储器空间的数据操作首地址 0x0000-0x6FFF
0x48	OP_Length	2	数据操作的字长度, 0x0001-0x6FFF
0x4A	Timer0	2	16bit 软件定时器自减到零停止, 单位为 4ms
0x4C	Timer1	1	8bit 软件定时器自减到零停止, 单位为 4ms
0x4D	Timer2	1	8bit 软件定时器自减到零停止, 单位为 4ms
0x4E	Timer3	1	8bit 软件定时器自减到零停止, 单位为 4ms
0x4F	Key_code	1	用户键码, 用于触发 0x13 触控文件
0x50	Play_Music	3	音乐播放器设定值, Play_Start 为播放起始段, Play_Num 为连续播放段数
0x53	Volume_Adj	2	写入 0x5A:VOL 将调整播放器音乐的音量 音量= VOL/64,

0x56	En_DBL_OP	1	0x5A 表示用户申请进行数据库存储器操作
0x57	OP_Mode	1	0x50: 把变量存储器控件数据写入数据库的控件 0xA0: 把数据库空间的数据读入变量存储器控件
0x58	DBL_Address	4	数据库空间地址 0x00:00:00:00-1D:FF:FF:FF 最大 480MW 数据库空间, 数据从物理存储空间的第 64MB 开始存储, 每 1Bite 数据库存储器占据 2Byte 物理存储器, 使用 SD 卡导出数据时, 每个字库大小为 64KW (128KB)
0x5c	VP	2	指定变量存储器空间的数据操作首地址 范围: 0x0000--0x6FFF
0x5E	Op_Length	2	数据库操作的长度范围 0x0001-0x6FFF
0xE0	Tp_Set_En	1	0x5A 表示进行一次触摸屏配置操作, 操作完后清零配置数据在 0xE1-0xE5 寄存器, 配置错误将导致触摸屏工作不正常
0xE1	Save_En	1	掉电保存
0xE2	Tp_Sens	1	置为 0x0C 电容触摸屏出厂设置为 0x00 过高的灵敏度设置可能会导致要延误动作
0xE3	Key2_Delay	1	一直按住触摸屏时, 第二次按键在第一次按压后的延时时间, 0x05-0xff, 为 10ms, 设置越小点动模式越不好控制, 设置越大, 增量调节滑动调节响应越迟钝,
0xE4	TP_OFF_Delay	1	触摸屏松开后, 在重新开始检测触摸屏的延时时间, 0x01-0x32 单位为 10ms. 设置越小触摸屏配合越流畅, 但是容易连续。
0xE5	Tp_Speed	1	一直按住触摸屏后 触摸屏的响应间隔, 0x05-0x32 单位为 10ms, 设置越小增量调节速度越快
0xE6-0xE8	保留	3	
0xE9	Scan_Status	1	0x01=触摸屏录入状态 0x00=触摸屏未处于录入状态
0xEA	TPCal_Trigger	1	写入 0x5A 启动一次触摸屏校正, 校正后自动清零, 写入特殊定义的数值一清除对应的曲线缓冲区数据
0xEB	TrendLine	1	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
0xEE-0xEf	Reset_Trigger	2	写入 0x5AA5 是屏软件复位一次

举例: 读取页面操作, 发送 5A A5 05 81 03 02 00 00

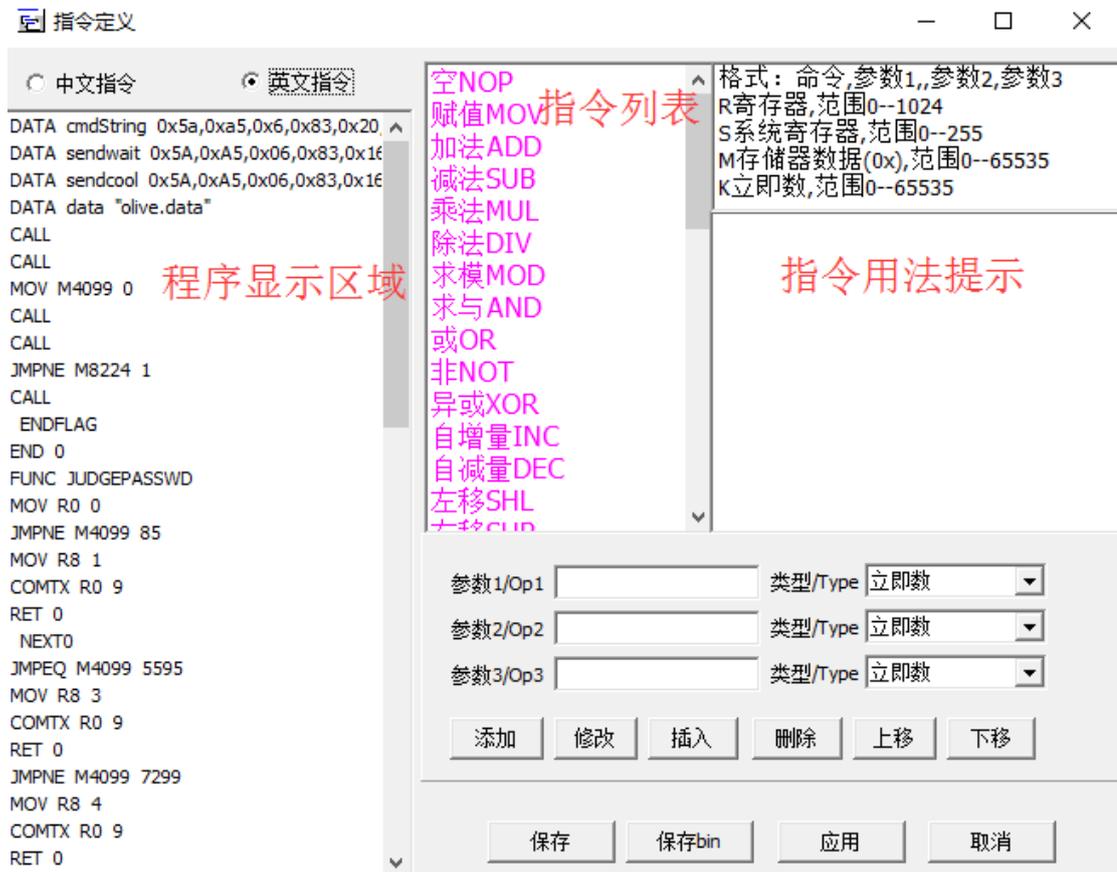
模块返回 5A A5 05 01 03 02 00 04, 表示当前页面 ID 为 4

PS: 该指令未使用 CRC 校验

5. Kadi OS 用户程序设计

Kadi os 平台采用类似汇编程序编写规范，方便用户针对自己的特殊需求快速可靠的进行二次开发

Kadi os 的 PC 软件编译界面如下图所示



基于 Kadi 编辑器平台,用户程序设计主要以文件的方式程序，可以实现多元化的设计,用户程序的执行是以文件的形式运行，从而大大的提升了系统的可靠性。

Kadi 平台上的使用 OS 来解析用户自定义数据协议和进行数据，代替了 HMI 和工控机的繁琐，不仅降低了成本，也使系统的存储大大的降低了。

5.1 指令约定

每条指令分成 4 段：指令命令，Op1 参数 1， Op2 参数 2， Op3 参数 3
每个参数，根据情况的不同可以使用以下 4 种类型的数据：

R 寄存器数据，范围为 0-1024(单个字节)

S 系统寄存器，范围为 0-255

M 存储器数据，范围为 0-65535(两个字节)

T 文本数据， 为变量名称，指令会自行匹配相关数据,长度

D 数据数组，为变量名称，指令会自行匹配相关数据

K 立即数。范围为 0-65535；（单个存储器地址只能存放一个双字节数据）

所有操作指令中，对于运算操作对于数据字节数量都按取大优先的原则。

说明如下：

ADD R0, M0, 1, 则自动扩展 R 寄存器为连续两字节。结果为 R(0:1)+=M0;

ADD M0, R0, 1, 则自动扩展 R 寄存器为连续两字节。结果为 M0 += R(0:1);

ADD R0, R10 则表示单字节地址操作： R0+= R10;

ADD R0, 200 因为 200 为小于 255 则直接用单字节操作。

ADD R0,1024 则表示 R(0:1)+=1024，用两字节数据参考运算。

下面分别对各指令进行详细说明：

指令功能	操作码	参数	说明
空操作	NOP		不执行任何操作
变量之间的赋值操作	MOV	Op1: 目标操作数 Op2: 源操作数 Op3: 操作数量	将 Op2 的数据赋值到 Op1。如果不写操作数 3，则默认只操作一次。且 Op1 不能为立即数，当 OP1 和 OP2 类型不等时按单字节地址自动加 1。例如： MOV R0,M10 则会把 M10 的高字节给 R0,低字节给 R1。 如果 Op2 为立即数和 T，D 时，且 OP3 为有效值时，Op2 不会执行数值加 1，而为 R，S，M 则会地址自动加 1。
加法	ADD	Op1: 目标操作数 Op2: 源操作数	将 Op2 数据加到 Op1 上 OP1 只能为 R 和 M 寄存器 OP2 只能为 RMK 例如 ADD R0 ,M0001 将 M 寄存器的 0001 地址开始的两个字节数据加到 R0 寄存器，超出一个字节自动存储到 R1 寄存器。
减法	SUB	Op1: 目标操作数 Op2: 源操作数	说明参考 ADD 操作 Op1 = Op1-Op2
乘法	MUL	Op1: 目标操作数 Op2: 源操作数	说明参考 ADD 操作 Op1 = Op1*Op2
除法	DIV	Op1: 目标操作数	说明参考 ADD 操作

		Op2: 源操作数	Op1 = Op1/Op2
求余数	MOD	Op1: 目标操作数 Op2: 源操作数	说明参考 ADD 操作 Op1 = Op1%Op2
与运算	AND	Op1: 目标操作数 Op2: 源操作数	说明参考 ADD 操作 Op1 = Op1&Op2
或运算	OR	Op1: 目标操作数 Op2: 源操作数	OP1 只能为 R 和 M 寄存器 OP2 只能为 RMK 例子 OR R0, M0001 将 M 寄存器的 0001 地址开始的两个字节数据与 R0 数据值或运算，或运算完后 将数值高字节存起在 R0 低字节存储在 R1 Op1 = Op1 Op2
非运算	NOT	Op1: 目标操作数 Op2: 源操作数	OP1 只能为 R 寄存器 Op1 = ! Op1;
异或	XOR	Op1: 目标操作数 Op2: 源操作数	OP1 只能为 R 和 M 寄存器 OP2 只能为 RMK 例子 XOR R0, M0001 将 M 寄存器的 0001 地址开始的两个字节数据与 R0 数据值异或运算，运算完后 将数值高字节存起在 R0 低字节存储在 R1 Op1 = Op1^Op2
自加一	INC	Op1: 目标操作数	OP1 只能为 R 和 M 寄存器 Op1 = Op1 +1;
自减一	DEC	Op1: 目标操作数	OP1 只能为 R 和 M 寄存器 Op1 = Op1 -1;
左移位	SHL	Op1: 目标操作数 Op2: 源操作数<16	OP1 只能为 R 和 M 寄存器 Op1 = Op1<<Op2; SHL R12,3
右移位	SHR	Op1: 目标操作数 Op2: 源操作数<16	OP1 只能为 R 和 M 寄存器 Op1 = Op1>>Op2;
置位	SETB	Op1: 目标操作数 Op2: 源操作数	OP1 只能为 R 和 M 寄存器 将其中的某一位置 1 SETB Op1 ,11
清除位	CLRB	Op1: 目标操作数 Op2: 源操作数	OP1 只能为 R 和 M 寄存器 将其中的某一位置 0 CLRB Op1 ,11

相等跳转执行 Op3	JMPEQ	Op1: 目标操作数 Op2: 源操作数 Op3: 跳转标签	Op1 可以为 RMS 寄存器 Op2 可以为 RSMK Op3 执行跳转的功能 JMPNE M1003,0x0055,NEXT0 如果 M1003 该地址的数据等于 0x0055 就跳转到 NEXT0 标签处
不相等跳转执行 Op3	JMPNE	Op1: 目标操作数 Op2: 源操作数 Op3: 跳转标签	Op1 可以为 RMS 寄存器 Op2 可以为 RSMK Op3 执行跳转的功能
大于于相等跳转执行 Op3	JMPBE	Op1: 目标操作数 Op2: 源操作数 Op3: 跳转标签	Op1 可以为 RMS 寄存器 Op2 可以为 RSMK Op3 执行跳转的功能
小于等于跳转	JMPLE	Op1: 目标操作数 Op2: 源操作数 Op3: 跳转标签	Op1 可以为 RMS 寄存器 Op2 可以为 RSMK Op3 执行跳转的功能
无条件跳转	JMP	Op1: 跳转标签	JMP NEXT0
调用	CALL	Op1:调用函数的名称	CALL Test(函数名称)
返回	RET	无	
结束	END	无	
函数	FUNC	Op1:函数名称	
文本变量	TEXT	Op1: 文本名字 Op2: 文本数据	TEXT hello "hello word!" Op1 位 hello 名称 Op2 数据
数据变量	DATA	Op1: 文本名字 Op2: 数据数组	DATA cmdString 0x5a,0xa5,0x6,0x83
标签	LABEL	Op1: 文本标签	
发送	COMTX	Op1: 数据地址 Op2: 发送长度	COMTX R0,9
接收	COMRX	Op1: 数据地址 Op2: 接受长度	COMRX R0,9
文件打开	FOPEN	Op1: 新建或打开名称 Op2: 文件操作标志 Op3: 文件大小	FOPEN data,1,R100 Op3 为 R100 用于存取文件的大小 Op2 为 1 表示写数据 为 0 表示读取数据
文件定位	FSEEK	Op1: 文件偏移位置 Op2: 搜索类型	
文件读取	FREAD	Op1: 数据内存 Op2: 数据数量 Op3: 成功失败标记	FWRITE R0,6,1 从 R0 地址数据开始读取 6 个数据写入文件中

文件写入	FWRITE	Op1: 数据内存 Op2: 数据数量 Op3: 成功失败标记	FREAD R0,6, 1 从文件中读取 6 个数据放到 R0 开始处
文件关闭	FCLOSE	无	
文件删除	FDEL	Op1: 文件名 Op2: 删除情况返回	
文件拷卡	FCOPY	Op1: 文件名 Op2: 拷贝完成情况	