# 基于串口的调试工具

## 修订历史

* V0.1.0：完成初版工具的实现。
* V0.1.1：改进了画图功能，~~修正部分BUG~~。
* V0.1.2：加入LUA脚本初始化设定功能，修改数据处理程序希望能提高计算速度，加入统计功能，~~修正部分BUG~~。
* V0.1.3：加入坐标标定功能，改进绘图窗口UI，加入剪贴板功能，为协议1加入简单的求和校验，~~修正部分BUG~~。
* V0.2.0：加入DUMP功能，完善脚本接口函数。
* V0.2.1：加入CALL功能，加入串口协议，~~修正部分BUG~~。
* V0.2.2：加入对hex16等非常规数据支持，~~修正部分BUG~~。
* V0.2.3：原绘图的最后一个版本，~~修正部分BUG~~。
* V0.3.0：换用新版绘图模块，~~修正部分BUG~~。
* V0.3.1：加入dump的单独绘图功能，~~修正部分BUG~~。
* V0.3.2：加入基于lua的台架数据modbus读取，~~修正部分BUG~~。
* V0.3.3：修改帧出错处理逻辑，提高波特率选择范围，加入电阻箱modbus。
* V0.3.4：添加host判断与警告窗口，添加lua纵坐标轴位置调整函数，加入值设置函数。
* V0.3.5：添加lua加入表格项的API，按下配置按钮API，加入绘图项API。
* V0.3.6：优化绘图显示UI，摆烂加入统计功能。

## 简介

该工具的名称定义为SCD：Serial Communication Debugger。

该工具没有明确的实体。协议只是建议性的，可根据实际需要更改。从机的C语言代码只是一种参考实现，完全可以用代码生成等得到功能兼容的程序。同理，上位机也只是一种基于C++ QT5.12界面库的参考实现，完全可以根据需要修改，必要时可用其他语言、工具链重新实现。灵活自由是该工具的最大特征。

SCD旨在使用串口模拟出基于CAN的pisnoop软件的基本功能。串口相较于CAN更为简单，成本更低（不论从嵌入式端通讯硬件要求还是PC端所需的连接设备来说），其应用门槛更低。复杂的PHY和协议虽然可提供更高的自由度，但对于开发人员而言，想要全面了解并且针对特定需求进行定制的难度也大大提高。SCD并不是要取代其他调试工具，它的设计初衷是让开发人员根据个别项目需求自行定制新功能、实现低成本高通用性地调试。

目录

[基于串口的调试工具 1](#_Toc105880485)

[修订历史 1](#_Toc105880486)

[简介 2](#_Toc105880487)

[串口通讯的格式 4](#_Toc105880488)

[设计原则： 4](#_Toc105880489)

[协议概述： 4](#_Toc105880490)

[从机给上位机的帧格式： 4](#_Toc105880491)

[上位机发给从机的命令格式 5](#_Toc105880492)

[从机软件的参考实现及应用demo 7](#_Toc105880493)

[使用方法 7](#_Toc105880494)

[实现细节 10](#_Toc105880495)

[上位机软件的参考实现和使用说明 13](#_Toc105880496)

[简单介绍： 13](#_Toc105880497)

[实现细节 19](#_Toc105880498)

[写在最后 20](#_Toc105880499)

## 串口通讯的格式

### 设计原则：

1. 协议简单，易于拓展，能快速掌握，并可自行修改对应软件，以实现客制化的更多功能（不足：单串口只能连接一个对象；牺牲部分自由度）。
2. 提供友好统一的接口，可为简单泛用性牺牲性能等其他指标。
3. 软件开源，共同丰富完善。

### 协议概述：

1. 不依赖于串口硬件分包，将串口视为流式传输，单数据帧不定长，但要求是4字节对齐的。要求全双工串口，假设传输出错率很小。
2. 该协议不依赖于硬件传输校验，也不进行校验。各协议可自行实现校验。
3. 为提高协议自由度，主机和从机是简单的状态机，但没有握手等协议控制。
4. 主机给从机发的为字符串明文协议，方便临时调试发送。从机给主机的是二进制字节流，可自行修改上位机以支持更多解析功能。

### 从机给上位机的帧格式：

不定长数据包+4\*8bit帧标识符，数据用小端输出

使用浮点中NaN的部分子集作为帧间隔和帧类型标识符 （signaling NaN **IEEE 754-2008**）

统一表示为（HSB→LSB）

0111 1111 10xx xxxx yyyy yyyy yyyy yyyy

六位x连起来解码，可称之为协议号，标定使用的协议类型。6bit最多表示64个数，即64种协议，应该用不完。后面的YYYY是少有的可以接

为保证数据通信在任意时刻中断后重连仍能识别到帧间隔，要求前面的不定长数据包按浮点数解释时都是有效的（不为NaN或Inf）。

#### 协议号：0：兼容VOFA的justfloat高刷协议

（默认主协议，用来做帧定位和帧分隔）

浮点数组形式的字节流协议，纯十六进制浮点传输，节省带宽。

yyyy部分全为0，兼容VOFA

N（目前实现，最大16个）个有效浮点数+协议号为0的帧结尾符。

#### 协议号：1：所有变量值包

N个有效浮点数+协议号为1的帧结尾符。

基本和原来的相同，只在末尾帧结尾符协议号不同，方便上位机进行识别解析。

yyyy部分为包的UINT16求和校验，提高可靠性。

校验错误将在命令行提示，但仍然显示收到的值。

N最大1199；

使用该协议读取所有记录的变量的值，注意，从机发给主机的只有浮点数！因此对超大的整数可能会得到近似值！！！

#### 协议号：2：dump协议

N个有效浮点数+协议号为2的帧结尾符。

yyyy部分为包的UINT16求和校验，提高可靠性。

校验错误将在命令行提示，但仍然使用收到的值。

使用该协议读取所有记录的变量的值，注意，从机发给主机的只有浮点数！因此对超大的整数可能会得到近似值！！

浮点1：0表示这是一个新的dump，1表示续前dump

浮点2：rsvd

浮点3-N，数据部分，最大1024。

#### 协议号：3：串口协议

将yyyy部分视为2字节串口。单帧4byte，无校验。

其他协议号目前未定义。

## 上位机发给从机的命令格式

(命令名)+(对象号)+(#号)+(参数)+(4个！号)

从机的参考实现中收到两条一样的命令后才会动作，这样实现了在人类可读的情况下提高了的通信的可靠性。

其中，对象号是写死在程序中的监控变量的编号。

目前支持的命令有 set（改变对象的值）、conf（配置从机传来的对象值序列）、chg（转换协议）

多说无益，举例说明：

注意c语言中默认数数是从0数起的。

将2号对象的值改为3.1415：

set2#3.1415！！！！

set2#3.1415！！！！

发送数字时请遵循C语言格式，不要乱加符号和空格，否则导致解析错误。

配置传输过来的数的第0个为0号对象：

conf0#0!!!!

conf0#0!!!!

切换协议为1号协议：

chg0#1！！！！

chg0#1！！！！

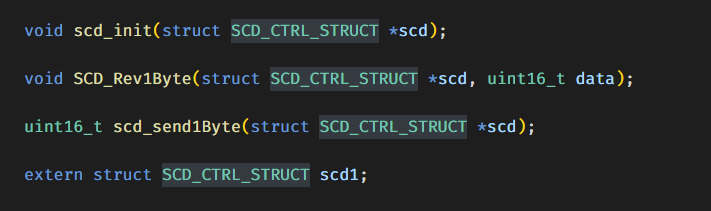
（这个命令的对象号需设为0）

## 从机软件的参考实现及应用demo

该参考实现核心就两个文件，scd.c和scd.h。

为了兼容内存位宽为16bit的DSP F28335，该参考实现的几乎没有用char、int等原生c语言类型，而使用stdint.h中提供的int16\_t等c语言标准类型。另外，还有部分地方为了实现兼容而使得写法比较奇怪，且牺牲了运行效率。

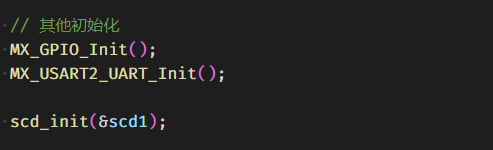
从scd.h中可以看到该参考实现接口十分简单，就3个函数。



应用demo是基于stm32l476RG NUCLEO开发板的，其方便之处在于同时集成了串口和stlink，同时可以用stm32CUBEMX方便地生成外设初始化代码。本文档随后的测试硬件都是基于此。

### 使用方法

在初始化部分：



在main主循环：



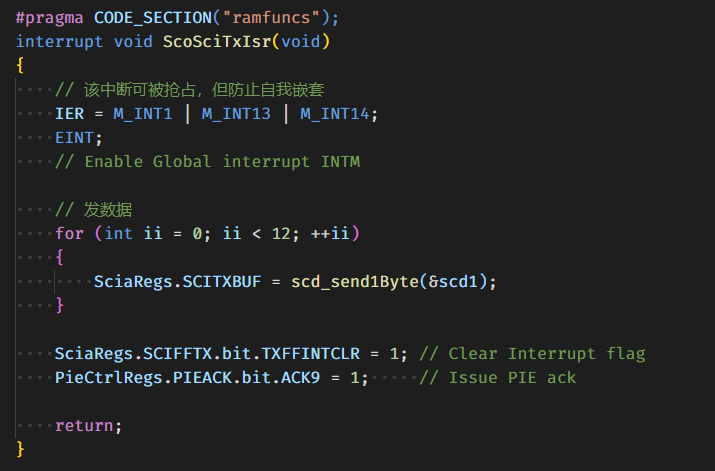
先调用初始化函数，然后再恰当的地方发送数据即可

在demo中使用了最简单的调用方法，在主循环中轮询，然后填值。

将scd\_send1Byte中返回的值填入串口，将串口收到的数据传入SCD\_Rev1Byte即可。

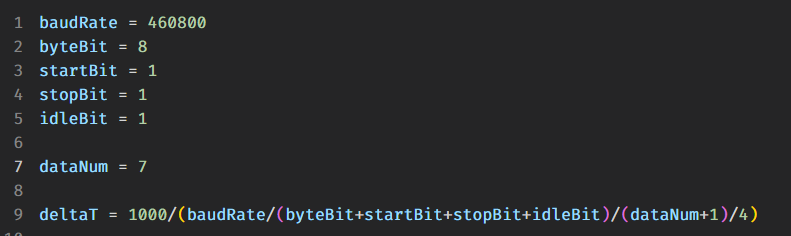
这样效率很低，而且发送的速率不仅不能确定还会随mcu计算负载波动而波动。

推荐结合FIFO、DMA、中断等硬件使用，下面是在F28335中应用的例子：

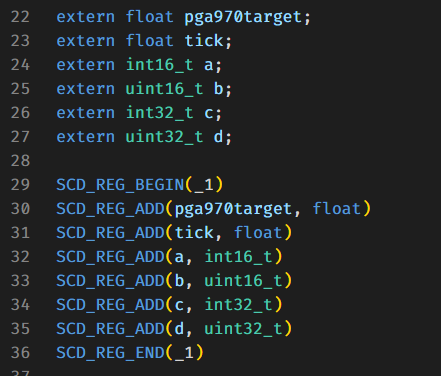


上图是发送时的情况，接收是类似的，从略。

使能串口FIFO，在串口只剩4个数据时中断。串口中断相较于PWM、ADC等主中断其优先级是很低的。在该中断中允许高优先级中断嵌套，以响应优先级更高的控制算法。每次向FIFO中填入12个数据，使得不必在频繁进入中断过程中切换上下文损失性能。这样能保证串口每时每刻都有数据发送，实现最大传输速度，也使帧间隔可以理论计算出来。如，下面的脚本可计算出：在波特率460800，停止位1位，起始位xxxx……的条件下的帧间隔。



最后需要根据具体项目内容修改scd.c文件的下面部分。



先要保证已经对要使用到的进行了声明，然后在

SCD\_REG\_BEGIN(\_1)和SCD\_REG\_END(\_1)

间加入要用到的值，注意使用stdint.h 中定义的名字。

上面的程序将pga970target定为0号对象，tick定为1号对象，以此类推。

SCD\_REG\*\*\*\* 一连串之间不能有空行，其结尾不能有标点。

目前支持：

float

int16\_t

uint16\_t

int32\_t

uint32\_t

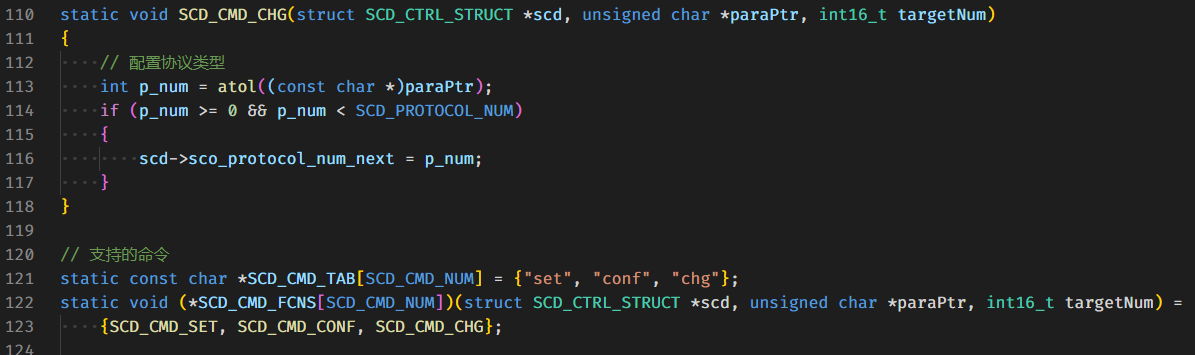
5种类型。

以上为移植的所有过程。

### 实现细节

倘若需要修改代码添加功能，就需要了解其大概的实现方式。

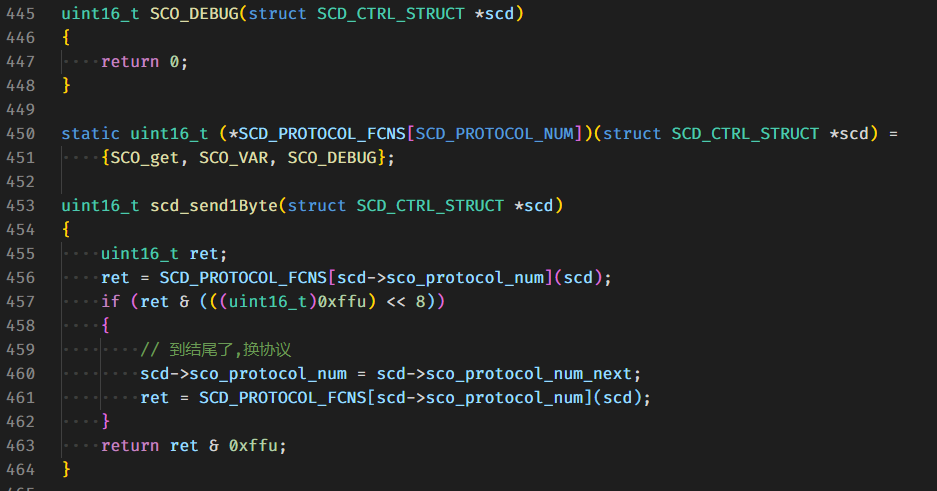
一项项说明工程量太大，细节逻辑直接参考源代码即可。



若要增加新的命令，先写出具体实现函数，在注册到上面的数组中即可，上面

void SCD\_CMD\_CHG(struct SCD\_CTRL\_STRUCT \*scd, unsigned char \*paraPtr, int16\_t targetNum)函数即是 协议切换命令“chg“ 的执行函数。

该框架会再解析字符串后调用相应的函数处理，其中scd为指向整个scd参数结构体的指针，方便实现高级功能， paraPtr指向参数字符串数组的地址，targetNum为获取到的对象号。



若要发送新的帧，只需将新的发送函数填入SCD\_PROTOCOL\_FCNS函数指针数组即可。

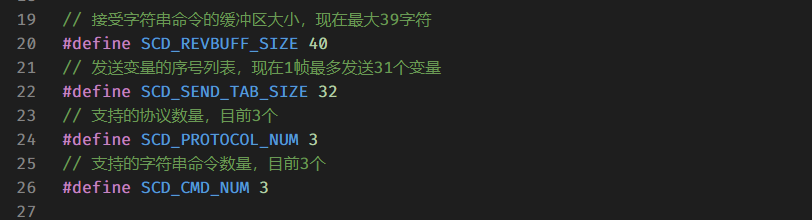
上图的SCO\_DEBUG就是一个最简单的只会使串口发0x00的函数实现。

要求在每帧结尾发送0x100，以便框架能适时地切换协议，因此上面的SCO\_DEBUG协议是不符合规范的，只做调试用。

实际上scd\_send1Byte函数就是根据内部状态跳转到实际的协议函数执行。

另外，值得说明的是，目前的协议号0、1实现都没有buffer，也就是传来的值不一定取自同一step。要保证每一帧的值都来自于同一个step是很难的，首先要开辟等于要传的变量的总大小的buffer，耗费宝贵的内存空间，其次要保证该函数不会被有影响的中断抢占。因此目前的实现不保证这一点。

最后提醒这几个宏的意义，要注意使用时不要超过范围。



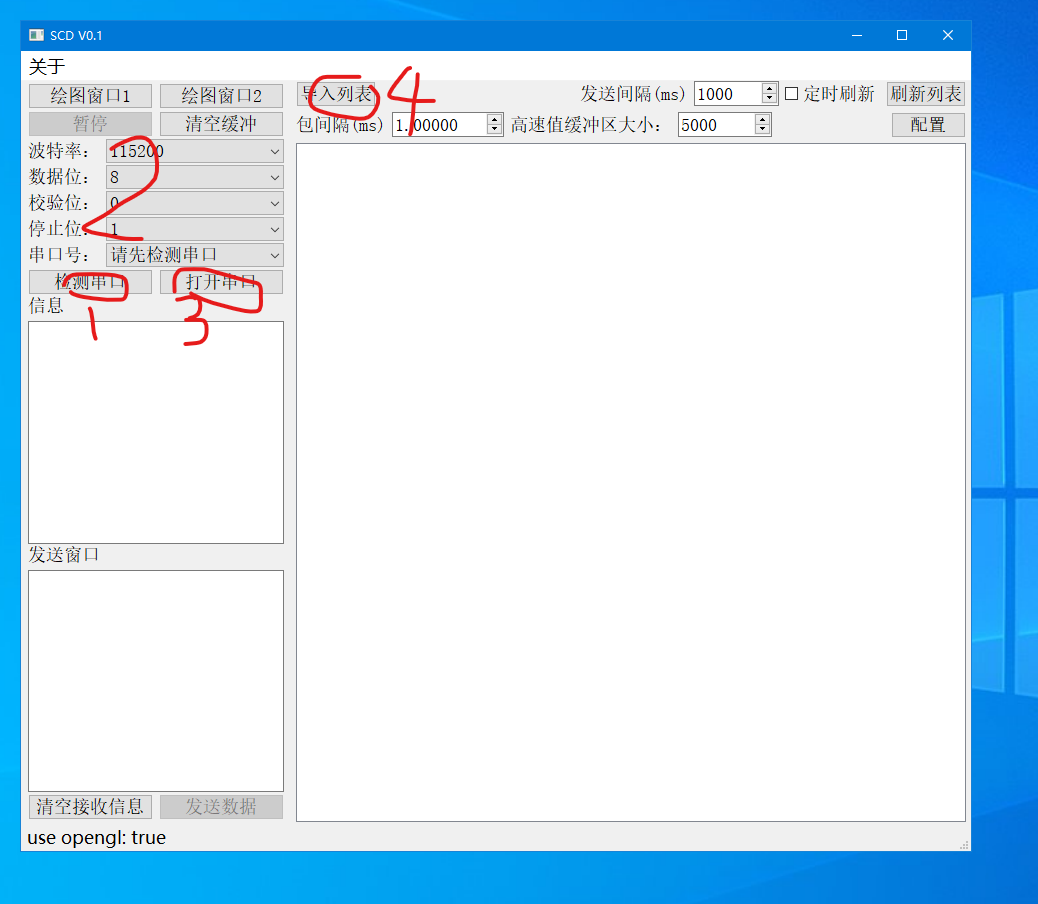
用SCD\_REG\_\*\*\*\*定义的目标对象的信息存放于FLASH，理论上只要FLASH够大，可以存32767个。其他变量存放于RAM，若RAM紧张可修改宏值节省内存。

## 上位机软件的参考实现和使用说明

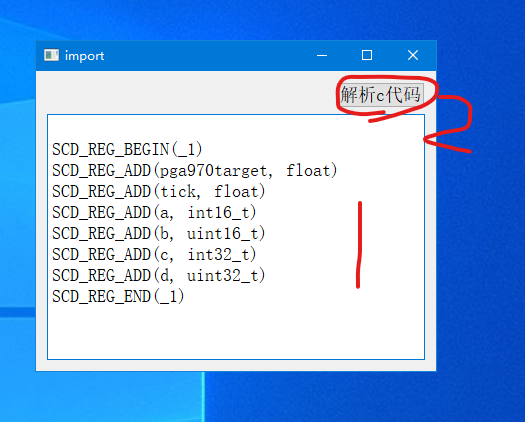
### 简单介绍：

程序是64位的，即scd文件夹的scd.exe，双击即可运行。不支持32位系统，部分老win7系统可能提示缺少vc++运行库，补安装即可。

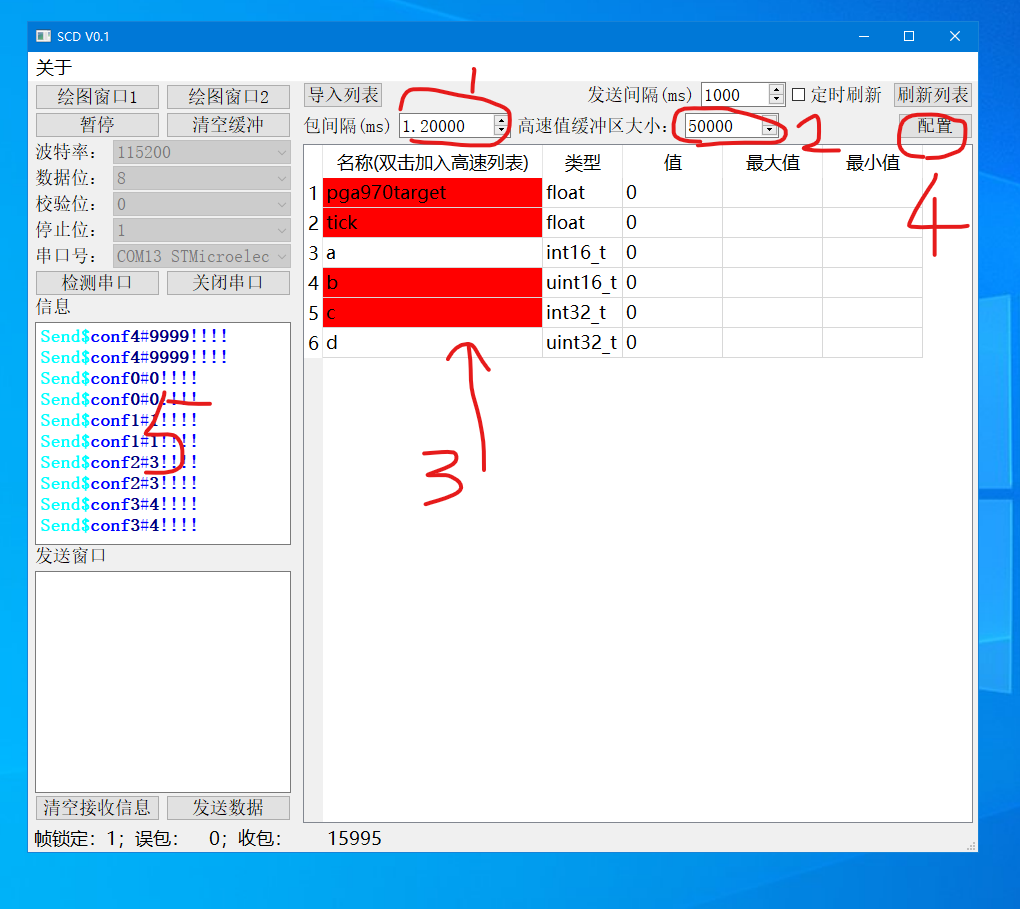
软件界面如图，使用应该很简单，试试就会。只做简单说明



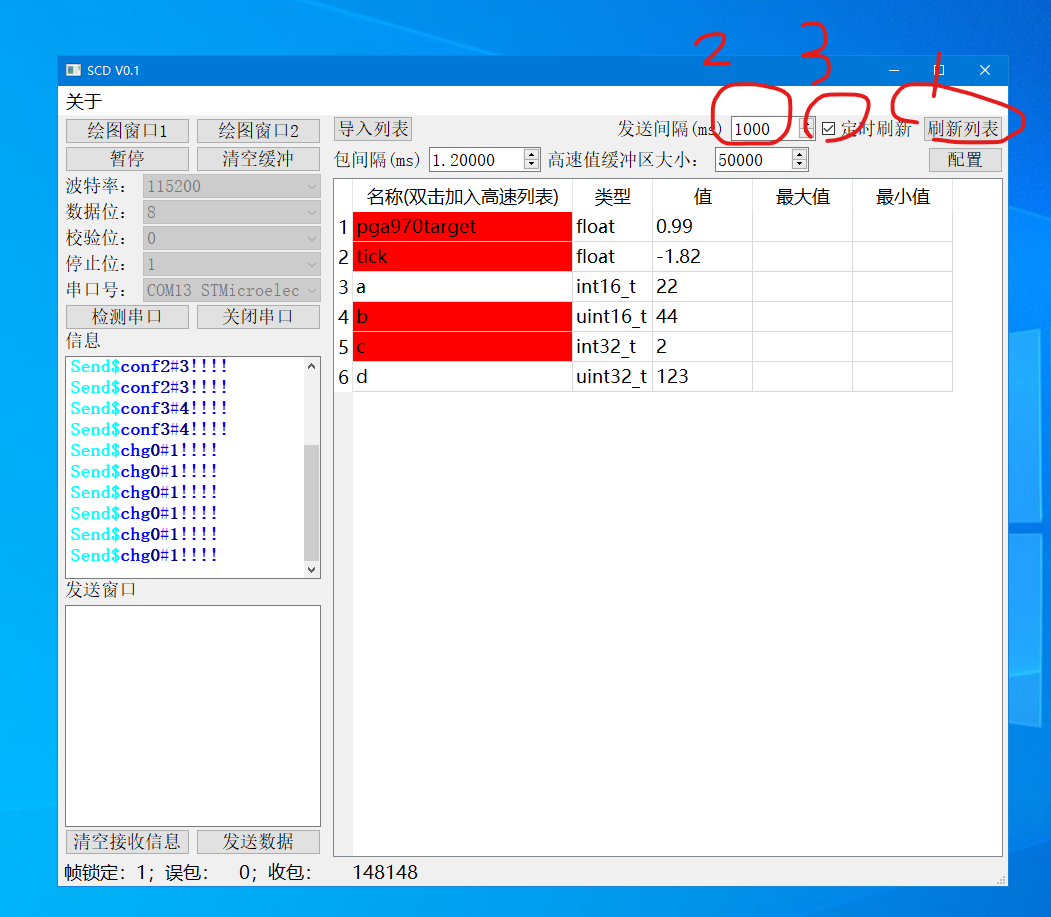
1. 配置串口模式并打开串口，这时变灰的按钮会重新使能。
2. 点击导入列表，粘贴上原c程序部分，点击解析按钮，即可实现自动解析，完成后可关闭该窗口。



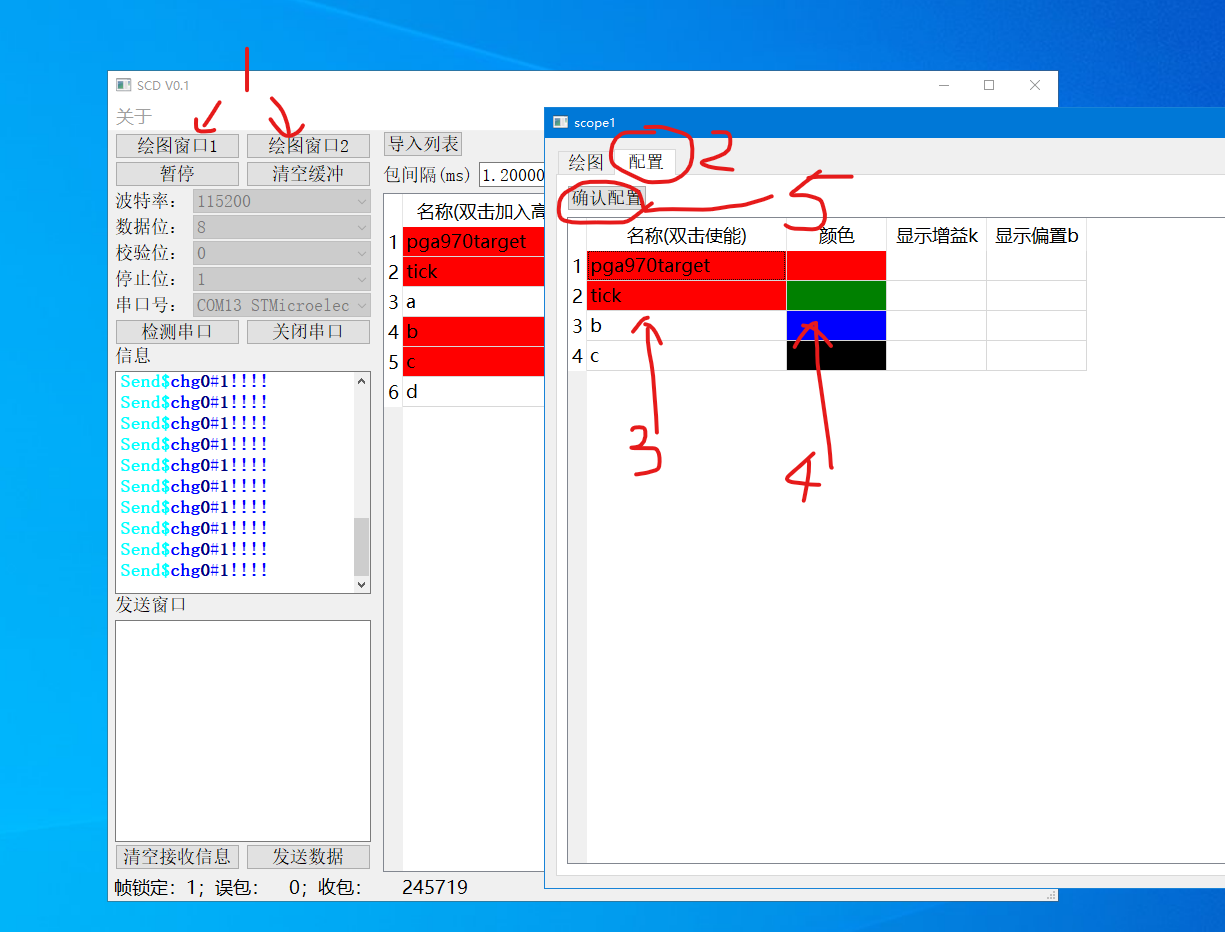
1. 根据实际情况配置包间隔和高速缓冲区的大小，双击变量名选择要加入高速列表的对象，点击配置完成设定，程序会自动完成对应命令的发送。



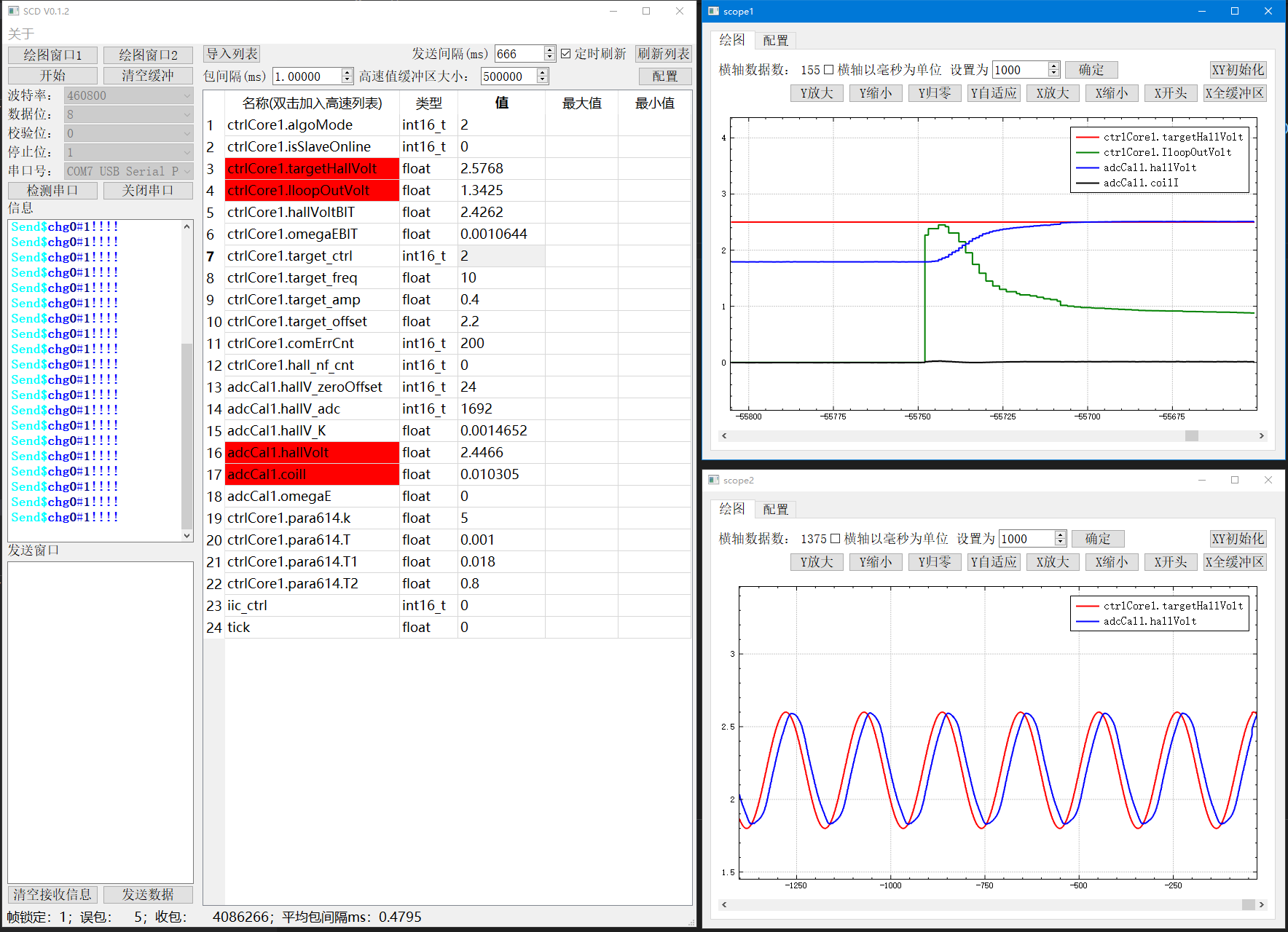
1. 点击刷新列表可用协议1得到表中变量的值。设定好间隔后勾选定时刷新可实现自动定时更新。



1. 打开绘图窗口，有两个，可以分别运行，方便多屏电脑使用，在配置选项卡中可设置要绘制的参数及其颜色，点击确认配置完成修改。



1. 回到绘图窗口即可看到波形。单击鼠标左键可拖动波形，双击鼠标左键可看特定点的值，再双击还原，使用滚轮可实现图像缩放，使用鼠标右键可框选放大特定部分波形。



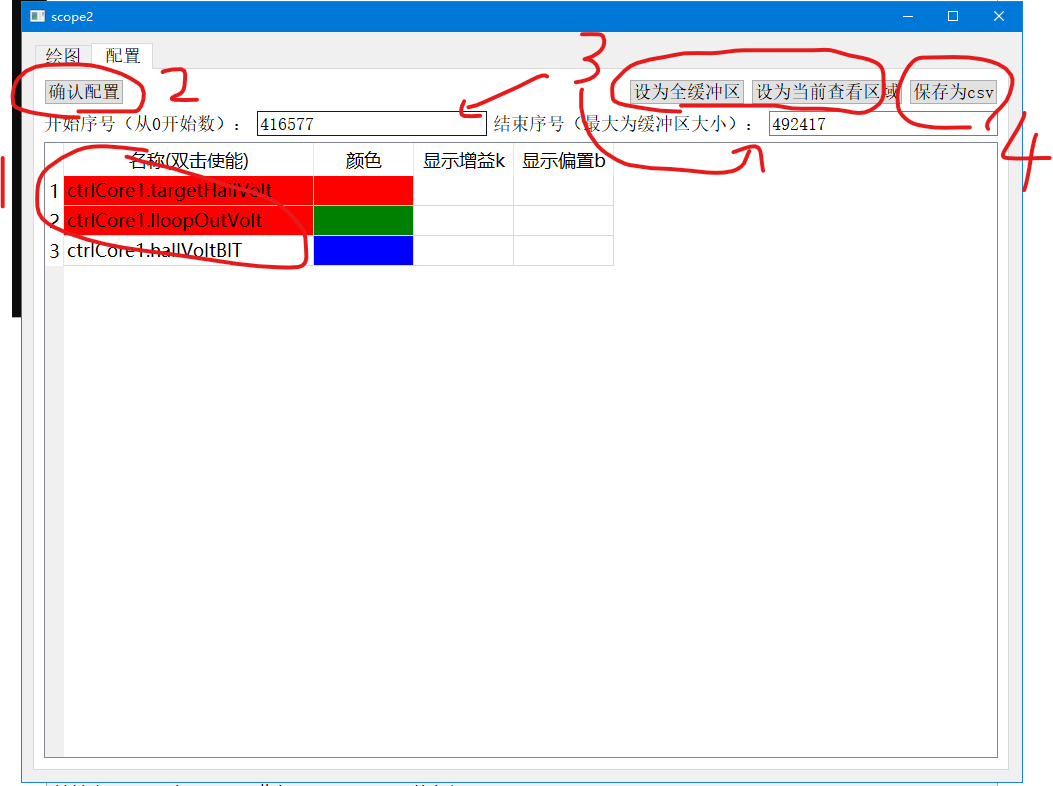
其他说明：

点击绘图框归位可回到图像原位。有X/Y放大（以中心位置为轴）、X/Y缩小（以中心位置为轴）、Y归零按钮。Y自适应按钮可以自动设定Y轴使得当前X范围内的点都可以显示。X开头按钮会在保持X轴刻度不变的情况下显示最右端（最新）波形。X全缓冲区会显示整个缓冲区的波形。

可精确设定横轴显示多少刻度。

注意：缩放太小时为了提高性能将进行抽样显示，可能出现波形跳动、失真。

也可将获得的数据保存为csv。先确定要导出（绘图）的数据项（双击标红），再可回到绘图页面查看波形，然后点击设为全缓冲区或当前查看区域（设定导出范围以减小导出数据大小），也可手动进行更改。最后点击保存为csv即可。



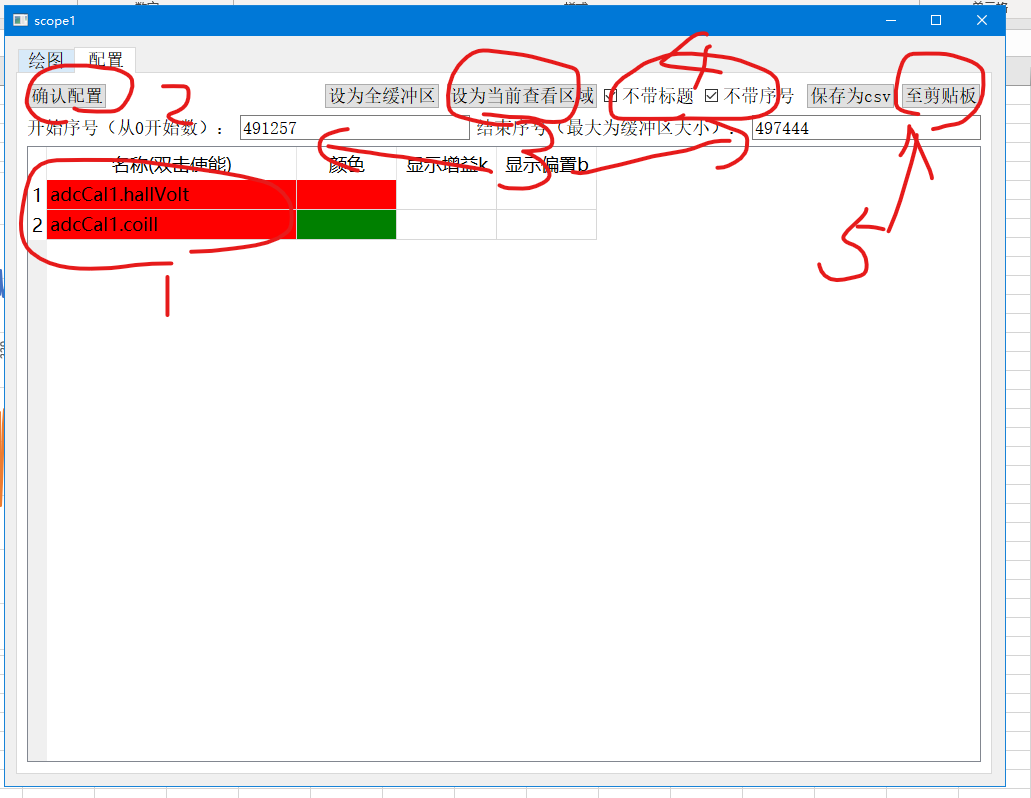
单击主程序的暂停可暂停高速缓冲的刷新，但不影响其他地方工作。

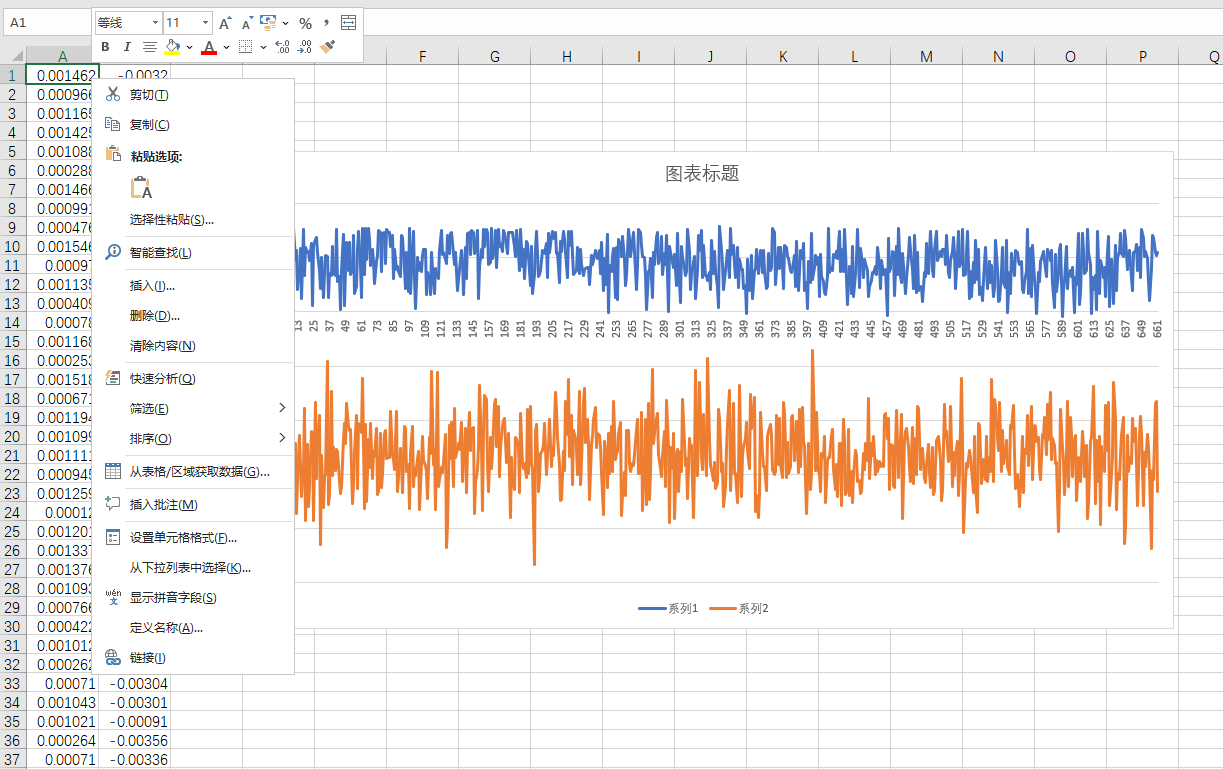
单击清空缓冲可清空缓冲，单击清空接受信息可清空窗口信息，单击发送数据可发送发送窗口里的字符串。

发送16进制数据的功能暂不支持，可临时用VOFA软件实现。

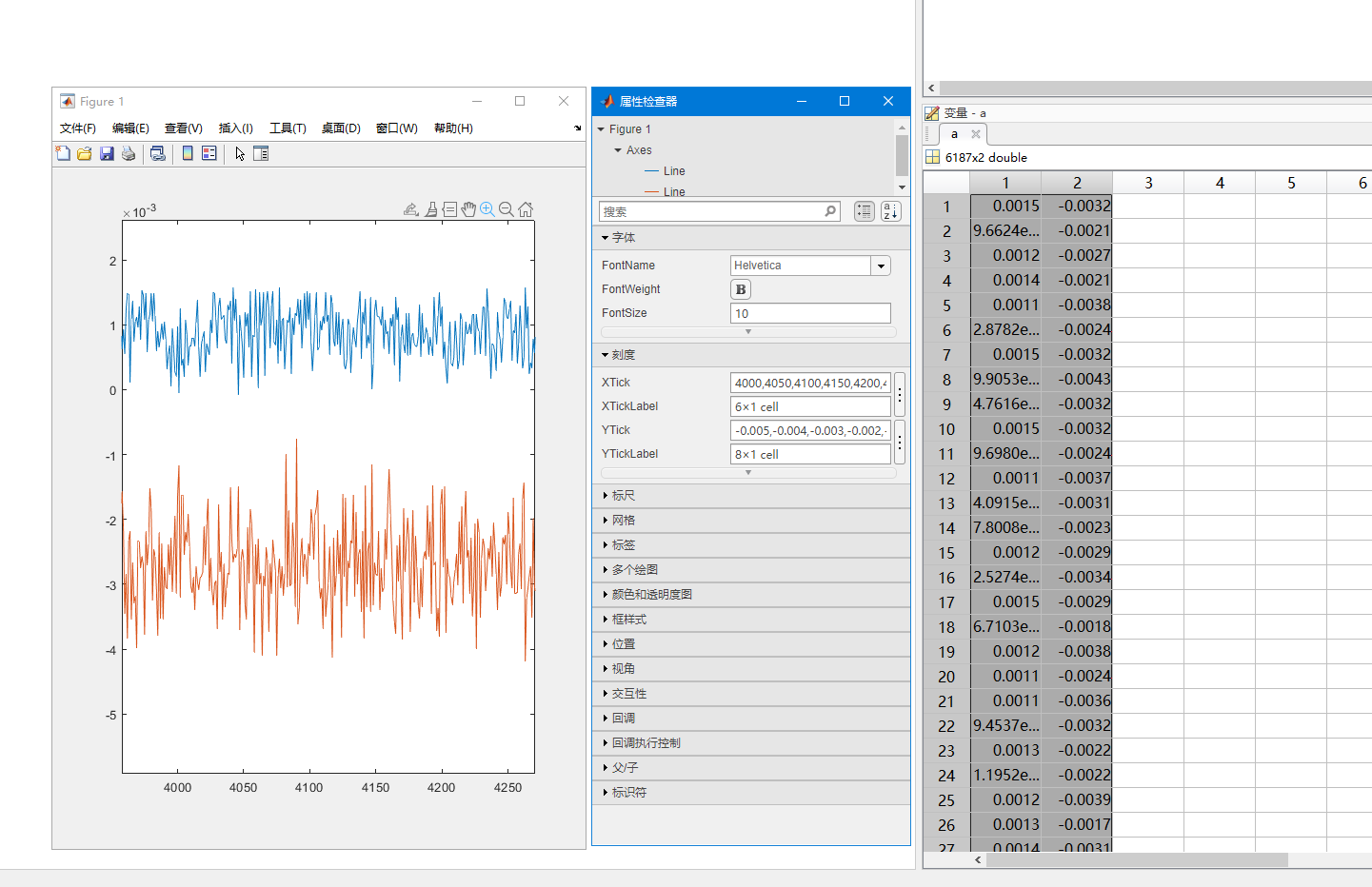
由于本协议兼容VOFA软件的justfloat协议，因此可用VOFA软件完成本软件没做的功能。。。vofa软件使用说明见：<https://www.vofa.plus/docs/learning>。

新功能：可选是否加入序号和标题，同时可与剪贴板联动，方便导入其他软件如粘贴入MATLAB和excel中。





粘贴到excel测试



粘贴到matlab测试

上位机加入了内嵌的LUA解释器以实现更加灵活的配置和基于脚本的测试需求。

配置文件即根目录下的config/scd\_cfg\_win64.lua，具体配置项含义见文件注释。

目前即加入了LUA5.4本体，并实现了一系列的lua调用函数接口。主要如下：

scd\_periodRun(elapsedMsFromStart)

自定义于初始化脚本文件，该函数会被定时运行，elapsedMsFromStart为从程序运行起的毫秒数。

scd\_dump(listNum,dumpNums);

将编号为listNum（从1开始数）的dumpNums个变量转储至计算机内存（覆盖原有的DUMP值）。

scd\_dumpOut(colCnt,method);

将内存中的dump值识别为colCnt列数据，使用method方法输出，可以为“cb”至剪贴板，“cmd”至控制台，“csv”至csv文件。

scd\_setVar(listNum,setVar);

发送将listNum号变量设置为setVar的指令，该函数不会直接更改上位机表格。

num scd\_getVar(listNum);

获得将listNum号变量的值，将从上位机收到的存储中直接读取。

scd\_refresh();

发送指令，更新表格中的值。

scd\_sendStr(strToSend);

使用串口发送字符串strToSend;

### 实现细节

若要修改本软件需学习C++基础内容和初步了解qt框架。

测试使用qt5.12版本，MSVC和MinGW都可编译通过，并正常运行。

本软件相当于一团胶水，粘合了各个部件和开源库。越写脑子越乱，整体结构还有优化空间，欢迎DL重构或补充功能。

自己看源码吧，咕咕咕了。

## 写在最后

整个压缩包包括本文档、下位机demo（含参考c语言实现），上位机参考实现及其源码。还有VOFA串口上位机软件，用来暂时顶替还没有编的功能。欢迎使用和改编！