

Chart系统说明文档

1.需求说明：

需要一个具有图形界面的应用程序，这个应用程序可以打开任意一个二进制文件，并且通过将二进制文件按每两个字节的形式读入成为一个short型的整数，然后将结果显示在一个具有坐标系的图表当中，并且形成一个曲线，其中图表可以显示多个数据通道，每一个通道可以通过通道旁边的操作按钮改变现在这个通道里面显示的图形，通常而言，屏幕宽度并不能够显示整个数据文件，因此通过前后拖动滚动条来定位不同时间段的数据，同时可以进行对于曲线进行压缩以及放大来检查细节波形，同时，用户可以设置曲线的颜色，用户的设置可以通过配置文件的形式储存在本地文件文件当中。

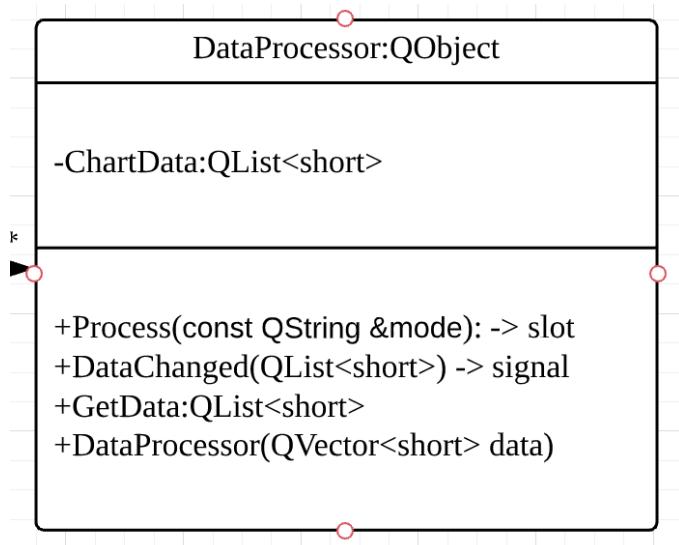
2.系统简要设计：

系统主要想要使用一种mvc的设计架构，视图层与数据层分离，然后用一个controller来传递视图层到数据层的修改与访问。同时系统的设计极大程度上考虑了课程中所讨论的开闭原则以及类的设计原则，极大程度的限制类与类之间相互访问的权限。同时，认真的考虑了每一个类的内聚性以及耦合因素，View类的设计中绝不储存model类的数据，model类也只做数据和储存的工作，其余的事物一律由controller来控制协调。

2.1 Model类的设计：

2.1.1 DataProcessor类的设计

在我们的程序当中，唯一的model就是从二进制文件读入的short数据，所以我们的程序当中只有这一个model类，除了存放我们读入的数据以外，model类还保有对于数据类的操作的方法。依照我们的类之间的设计原则，我们应该对于其他类只保持一些接口，将自己的属性私有化，让外界通过接口访问我们的属性，于是依照这个原则，我们将类中的数据设置为私有，同时，出于内聚性的原因，整个程序当中应该只有这一个类来对于数据进行修改，其他类只能行使读取的权利，所以，为了最小化其他类对于此类的访问，我们只给出getdata的接口，而不给出setdata。

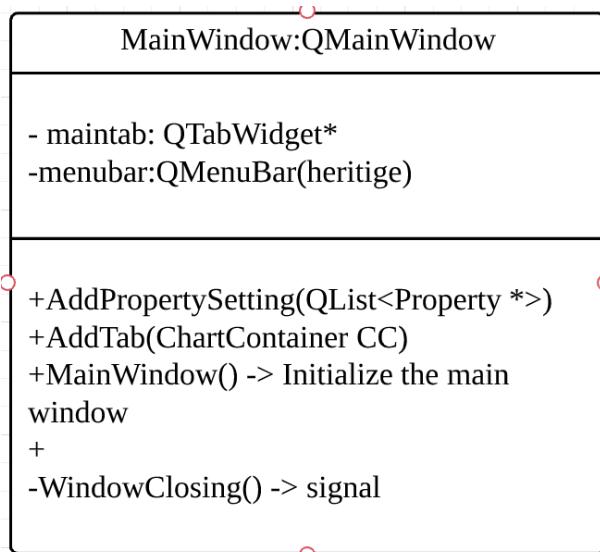


2.2 View类的设计

在我们的程序当中，主要的窗口是一个显示图像的窗口，同时，我们如果只能打开一个文件那么功能会显得稍微单一，于是我们利用qt的tab组件来实现多个文件同时打开并且能够显示的功能，为了实现这个功能，我们将每一个tabpage的内容划为一个类。同时为了实现多个通道的显示，我们每一个chart以及他的操作组件也划分为一个类，具体的设计在下面阐述。

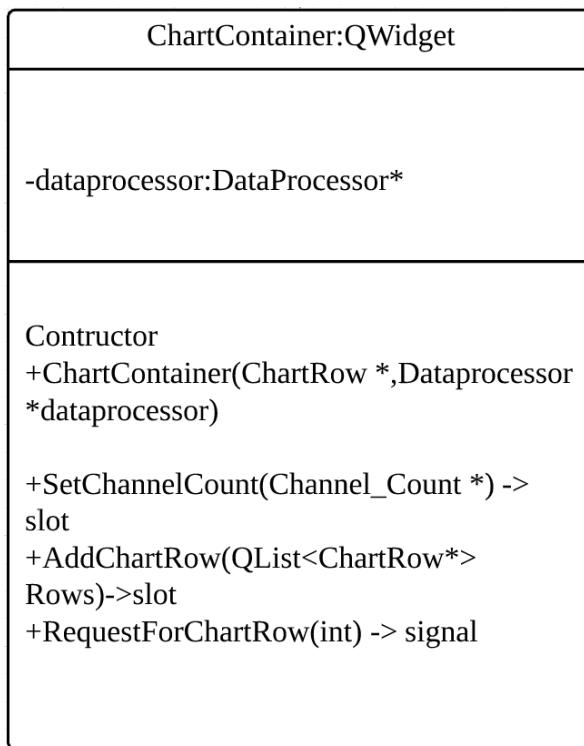
2.2.1 MainWindow类的设计

程序的主窗口需要容纳的是其中的tab，以及主菜单中对于属性的设置选项，同时，在窗口关闭时，我们需要通知其他组件关于这件事情的发生，用来实现程序关闭过后的对于配置文件的写入功能。同时，考虑到我们的其他类对于tab的访问权限应该为无，所以我们不给接口对于tab的直接读取，只有一个AddTab方法可以在tab中加入新的tabpage，以及一个AddPropertySetting将所有的类的设置对象加入到主菜单中。以下是这个类的UML类图



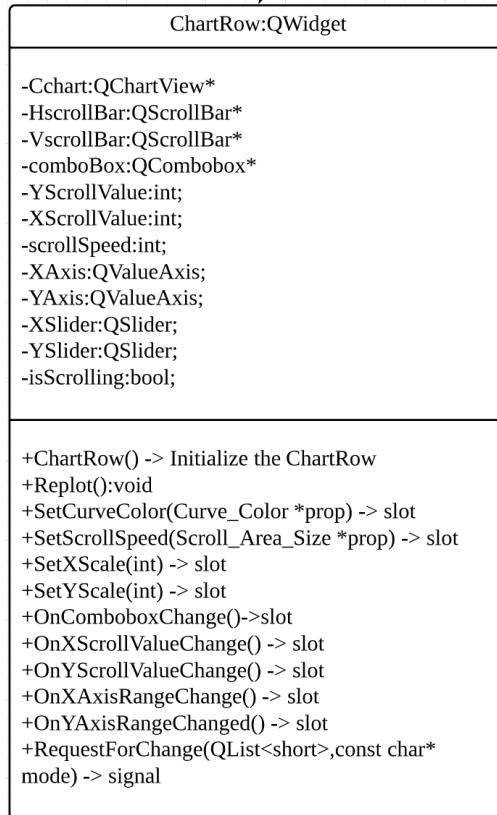
2.2.2 ChartContainer类的设计

程序的每一个tab都需要一个页面来容纳所有的通道，并且担任设置通道数的角色，所以这个类我们设置为ChartContianer，类中的属性dataprocesser是一个模版，并且通过构造函数进行唯一一次的赋值，这个唯一的dataprocessor会与chartcontiner中的每一个chartrow对象进行绑定，为每一个chartrow提供数据处理服务，同时他们两个又不直接相连，而是通过chartcontainer来关联。其中的其他两个函数AddChartrow与信号RequestForChartRow将在算法描述部分详细说明



2.2.3 ChartRow类的设计

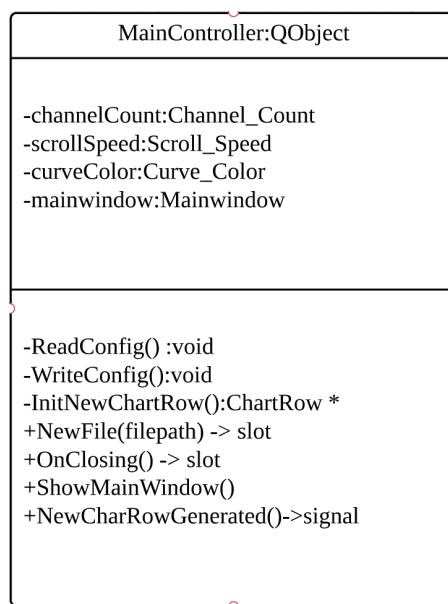
对于每一个单独的通道来说，我们需要将数据显示在图表当中，同时还有对于数据进行操作的各种组件，比如说控制xy轴方向缩放的拖动条，以及进行滚动的滚动条，同时还有进行操作的各种槽函数和信号，用于接受对于曲线颜色以及滚动速度的调整，同时，其他的类对于这个类中的属性访问都应该不能直接知道，并且其他类对于这个中的属性同时不具有读和写的权限，于是在这个类中没有给出对于属性的接口，另一方面，考虑到内聚性，用于显示的类不应该储存model类才储存的数据，于是直接采用根据数据重画chart曲线的方式来进行呈现(Replot函数)，而不是将数据存储为本地属性。具体的函数作用我们会在算法设计部分进一步的进行阐述。以下是这一个类的UML类图。



2.3 Controller 类的设计

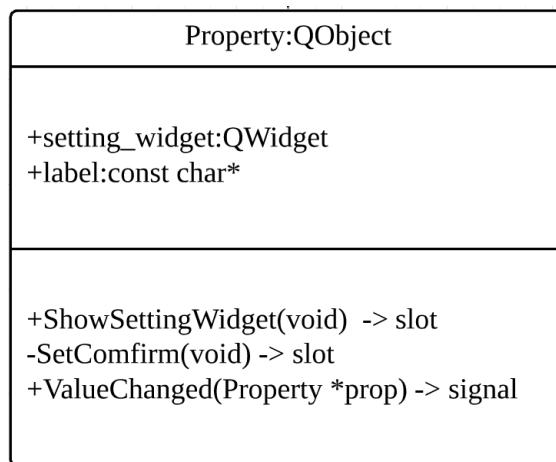
2.3.1 MainController类的设计

由于程序功能较为简单，所以我们只设计了一个Controller类来协调View类与model类，同时我们也希望Controller类来做一些初始化的工作，比如说创建初始的主窗口，以及container等，同时在初始化的时候利用qt的信号机制将他们的信号槽函数链接在一起。另一方面，读取文件以及配置文件的读取和写入也放入了主控制器的功能范围，在这里学生认为如果项目的规模足够大，这些部分的功能是应该划分为单独的类来进行实现的，并且如果要实现超大文件的分段读取提升性能，这个设计也是有必要的，但是现在我们只考虑简单一些的功能，就直接划为类中的一个函数了。

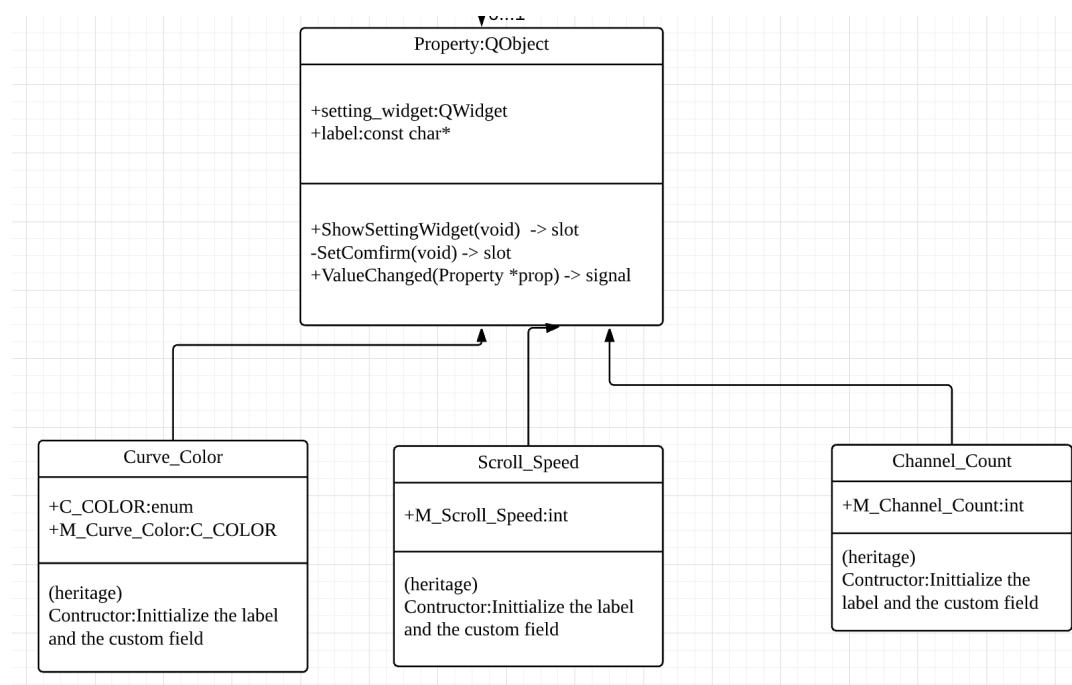


2.4 设置属性类设计

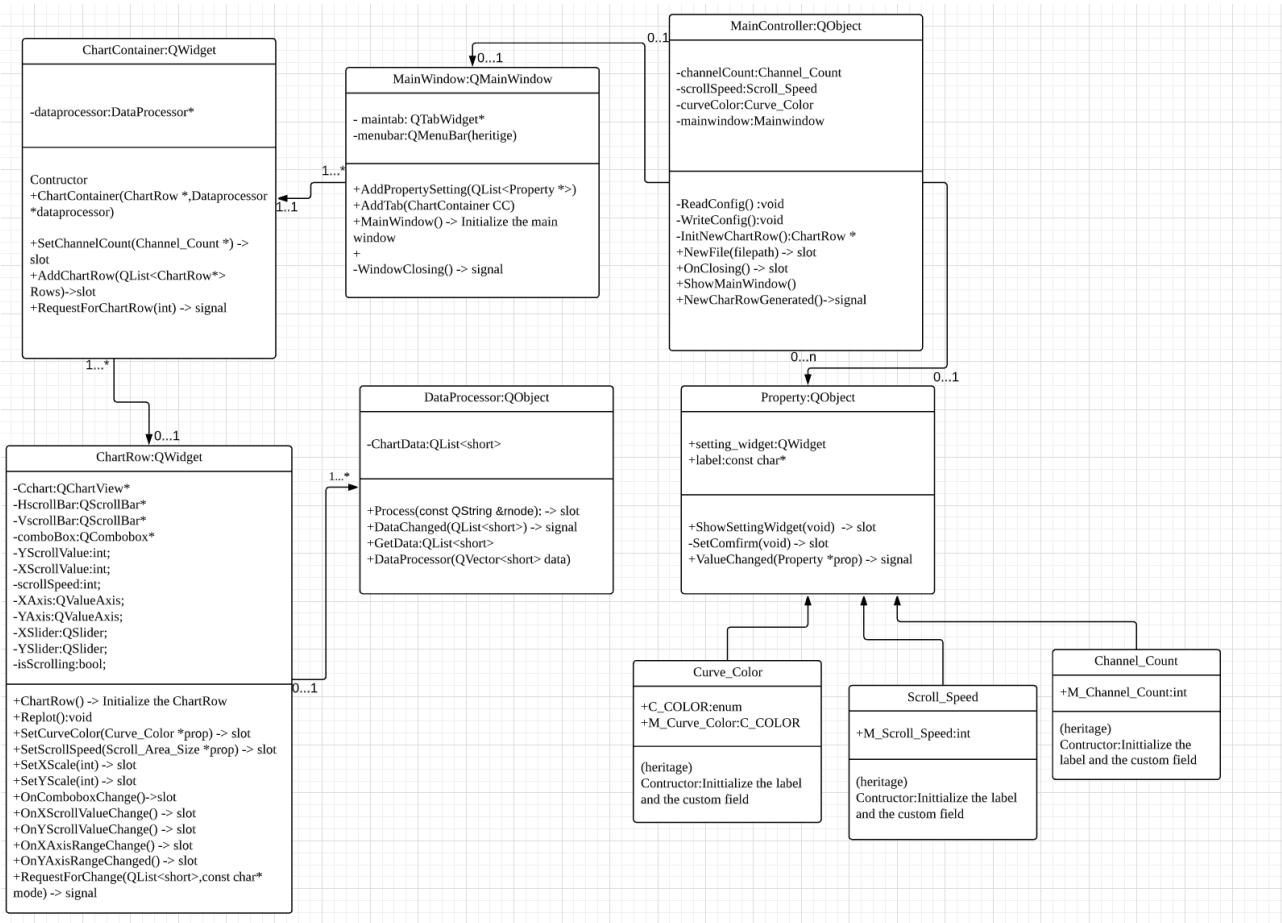
由于我们需要有几个属性：曲线颜色，通道数量，以及滚动速度需要设置，每一个设置需要的是不一样的设置窗口，但是同时，每一个设置在设置完成之后都需要发出信号来告知对应的组件关于属性值的改变，所以我们可以从这样的信息中抽象出实际上每一个属性类之间都有相似的属性，于是我们考虑到可以使用继承的关系来写这样一些类，这样还可以让类的设计符合开闭原则，首先，属性类都需要有的一个操作是显示各自的属性设置窗口，接着属性类还需要有一个槽函数来接受对自己的调用以及一个信号来通知对应的组件属性值改变。所以，这些属性的父类的设计为



同时由于每一个属性有着自己的不同类型的属性值，所以具体的属性值是保有在子类当中的，所以接着，我们的三个属性类是这样继承父类Property的：



2.5 整体的UML类图



3.核心算法阐述

3.1 组件协调基础：Qt信号槽机制

Qt的信号槽提供了一种任何继承了QObject的类之间通过一种类似函数回调的方式进行通信的方式，但是不同的是，Qt的信号槽更加的灵活。所谓信号是一个在类的头文件中用signals宏定义的函数声明，这个信号可以携带的一个参数来传递给目标的slot函数，也就是槽函数。利用信号槽的机制，组件之间的交流可以从直接的函数调用的这样的高耦合的方式变为一种信号槽之间的松耦合方式。

3.2 通过信号槽实现对于曲线的伸缩

首先，由于我们使用的是qt的chart组件，曲线可以与chart中的轴相关联，这样我们在改变轴的范围的时候，与其关联的曲线也会随之伸缩，所以现

在问题就变成了，如何控制chart中轴的范围变化。这里我们使用的是一个slider来发出一个信号，slider的值可以在我们设置的一个最大或者最小范围之内变化，当用户拖动slider的时候，slider的值会对应进行变化。并且发出一个信号，通知ChartRow类中的SetXScale/SetYScale函数。我们的轴的上界会从现在的上界变成现在的slider的值除以最大百分比乘上现在轴的下界到整个轴全长的距离，也就是：

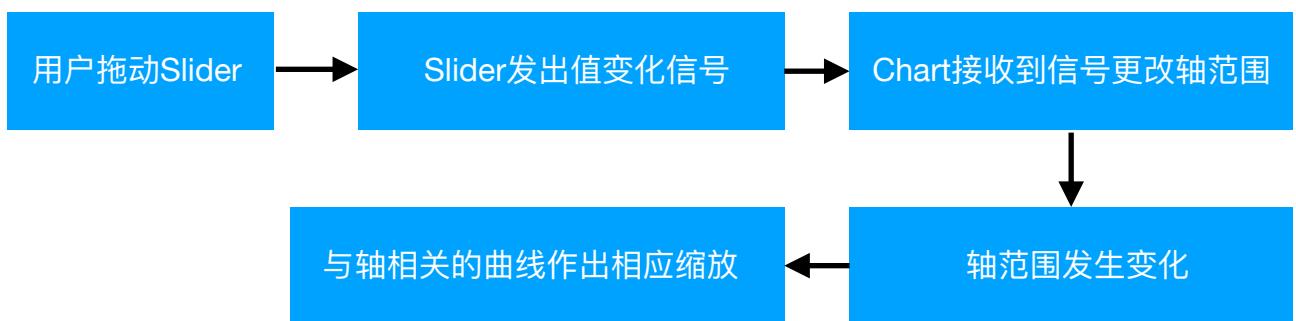
把轴的原来的范围

(min , max)

变为

(min, (value / SLIDER_MAX_PERCENT) * (length) + min)

其中value是现在slider的值，length是现在的最小值min到整个曲线的最大值的距离，总结起来，这个算法的流程图为：



3.3 通过信号槽的实现对于曲线的滚动显示

曲线的滚动可以使用chart的内置函数scroll来控制对于曲线的横向或是纵向的滚动，然后接下来的问题就是如何使用滚动条来控制这个函数的执行，并且结合曲线的放缩，正确的设置滚动条的位置和值。我们知道滚动条的位置应该是相对于现在我们可见的这一部分轴与全部的轴的长度的比例的关系的。所以轴的范围变化应该也要引起滚动条的位置以及比例变化，于是我们设计了一个槽函数OnXAxisRangeChange以及OnYAxisRangeChange来检测轴的范围变化同时设置滚动条到合适的位置，同时滚动条的值变化会发出信号，调用函数OnXScrollValueChange 以及 OnYScrollValueChange来对于图片进行合适的滚动。

在上面我们说的滚动条的位置与范围，与可见范围的轴和全部长度的轴的大小应该有这样如下的关系，首先，滚动条的范围最大值应该轴的全部长度减去可见长度，其次，滚动条的新位置的值比上现在滚动条的最大值，应该与原来的滚动条的值比上原来的滚动条最大值，所以我们得到如下的等式

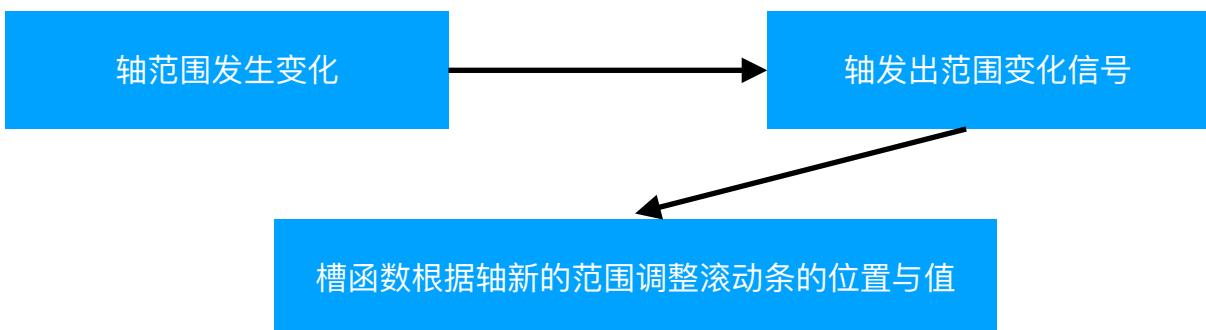
$$NewScroll_{max} = AllRange - NewRange$$

$$\frac{Scroll_{min}}{Scroll_{max}} = \frac{NewScroll_{min}}{NewScroll_{max}}$$

于是，根据这两个等式，我们可以写出在轴的范围变化时，我们需要对于滚动条作出的改变代码：

```
int ori_max = XScrollBar -> maximum();
XScrollBar -> setMaximum((int)(all_width - visible_width));
XScrollValue = (((min)/ori_max) *(all_width - visible_width));
XScrollBar -> setValue(XScrollValue);
```

用流程图表示这个过程为：



其中，可以发现的是，其实这个流程图与第一个流程图是相结合的，当我们的用户利用slider放缩曲线时引起的轴范围变化也会引起滚动条的范围以及位置好变化，客观来说这是合理的。

接着，我们需要做的就是当用户在拖动滚动条的时候去调用Scroll函数来对于chart进行滚动的显示，但是原生的Scroll函数只能指定向某个方向滚动指定的长度，仍与我们的需求存在一定的差距，解决的方法是指定一个全局变量来记录上一次我们的滚动条的值，然后每次用户拖动滚动条之后，将新的值与原来的值做比较，来获得我们需要进行滚动距离，同时考虑到我们需要将滚动速度可以由用户设置，于是我们可以写出以下的代码，以X轴方向的滚动为例

```
double scroll_distance = scrollSpeed * ((double)value - XScrollValue);
Cchart -> scroll(scroll_distance, 0);
XScrollValue = value;
```

4. 测试用例以及测试记录

用例序号：1

功能点：打开文件并显示在图表中

用例名称：正常打开文件

操作步骤：

1. 打开应用

2. 点击菜单栏file下的open按钮

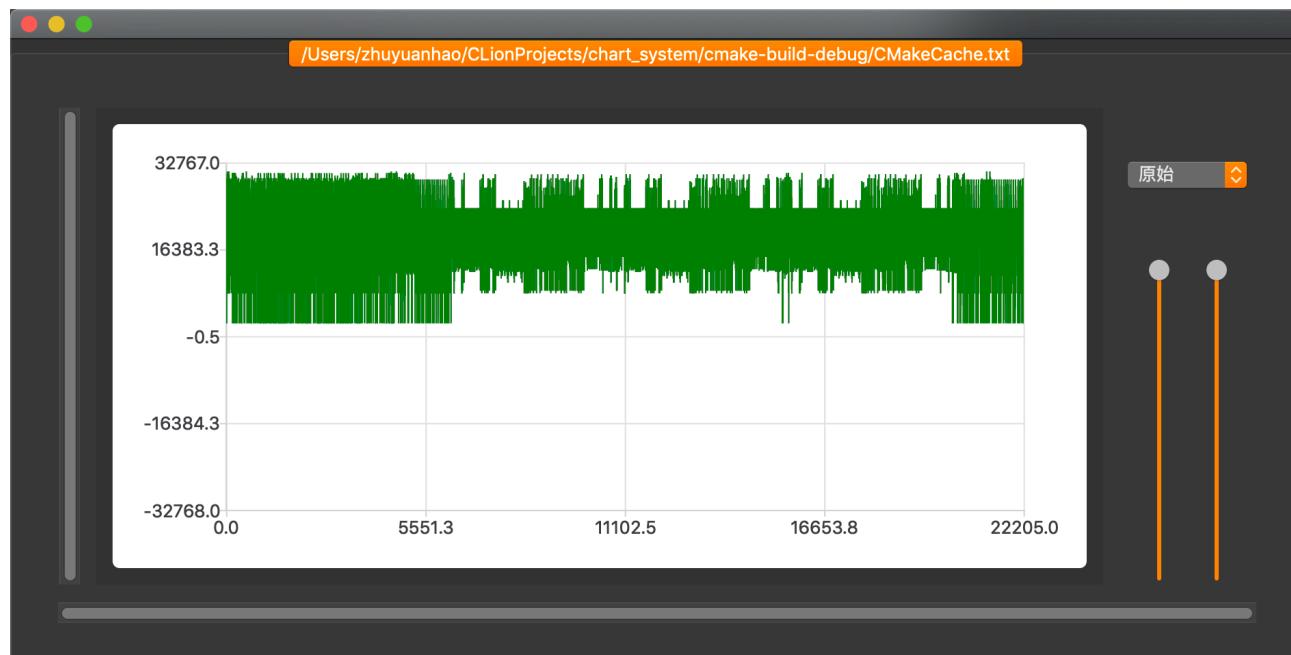
3. 选择文件

4. 点击确认

前置条件：配置文件正常，文件有足够权限

期待输出：正确显示文件波形

测试记录：



用例序号：2

功能点：打开文件并显示在图表中

用例名称：当配置文件错误时打开文件

操作步骤：1. 打开应用

2. 点击菜单栏file下的open按钮

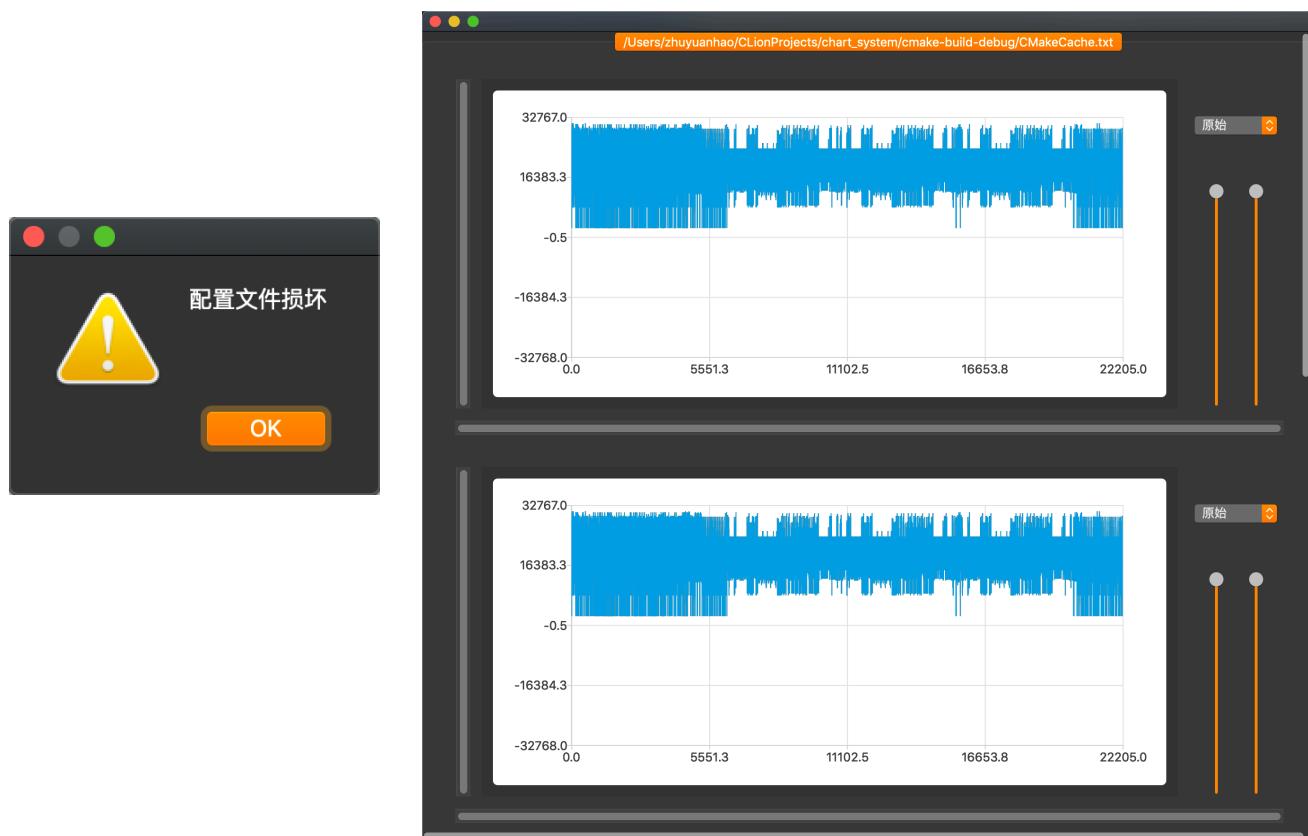
3. 选择文件

4. 点击确认

前置条件：配置文件错误，文件有足够权限

期待输出：打开应用时提示配置文件错误。然后继续执行，各属性为默认属性

测试记录：



注：和用例1相比，这次打开通道数变为了默认的三个，曲线颜色也变成了默认的蓝色

用例序号：3

功能点：打开文件并显示在图表中

用例名称：指定文件无法打开

操作步骤：1. 打开应用

2. 点击菜单栏file下的open按钮

3. 选择文件

4. 点击确认

前置条件：配置文件正常，文件无法打开

期待输出：应用提示无法打开文件

测试记录：



用例序号：4

功能点：对于图表进行横向压缩以及放大

用例名称：图表x轴缩放

操作步骤：1.打开应用

2.点击菜单栏file下的open按钮

3.选择文件

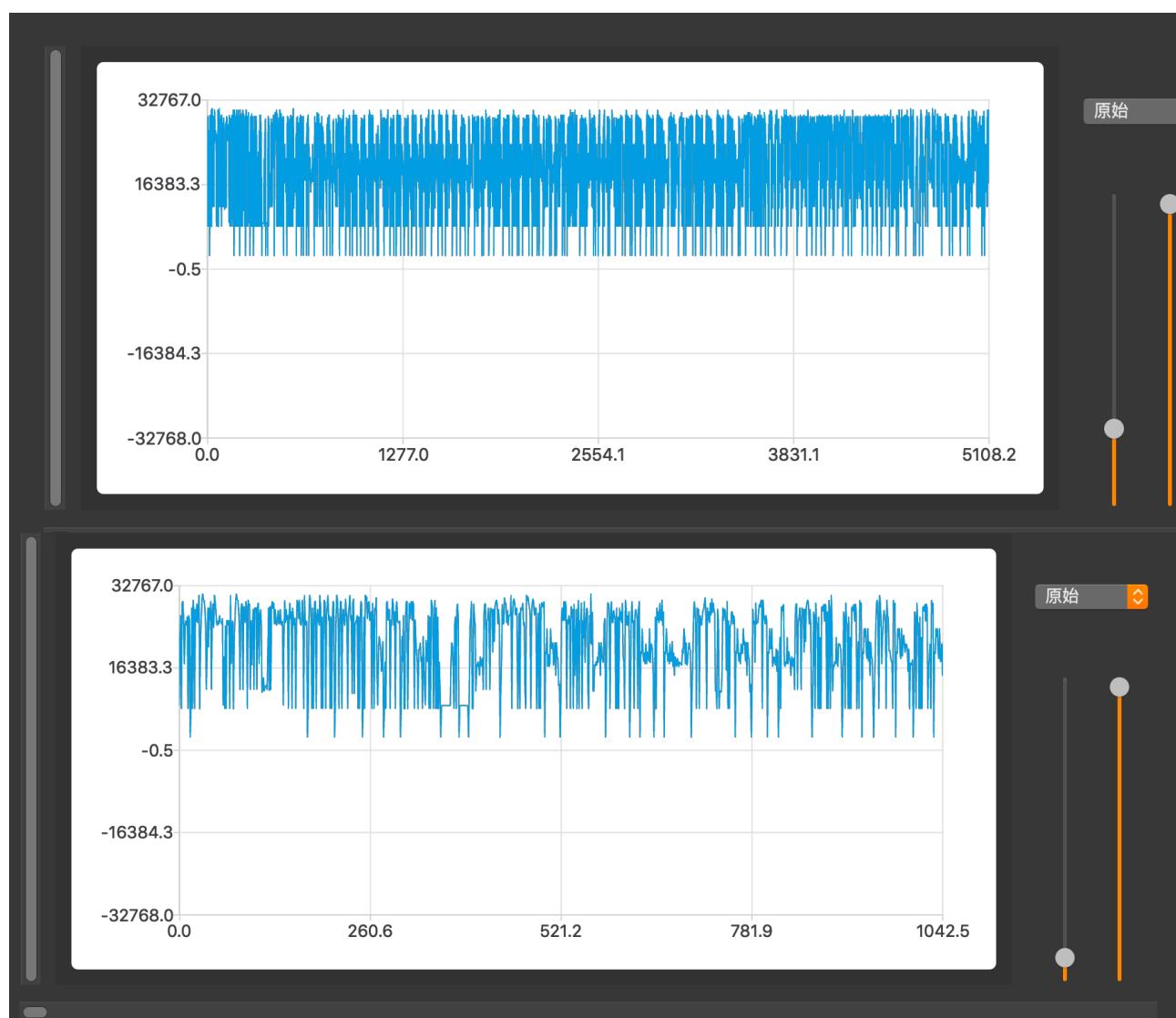
4.点击确认

5.使用滑动块进行缩放

前置条件：配置文件正常，文件正常

期待输出：图表x轴方向随滑动块正常压缩放大

测试记录：



用例序号：5

功能点：对于图表进行纵向压缩以及放大

用例名称：图表y轴缩放

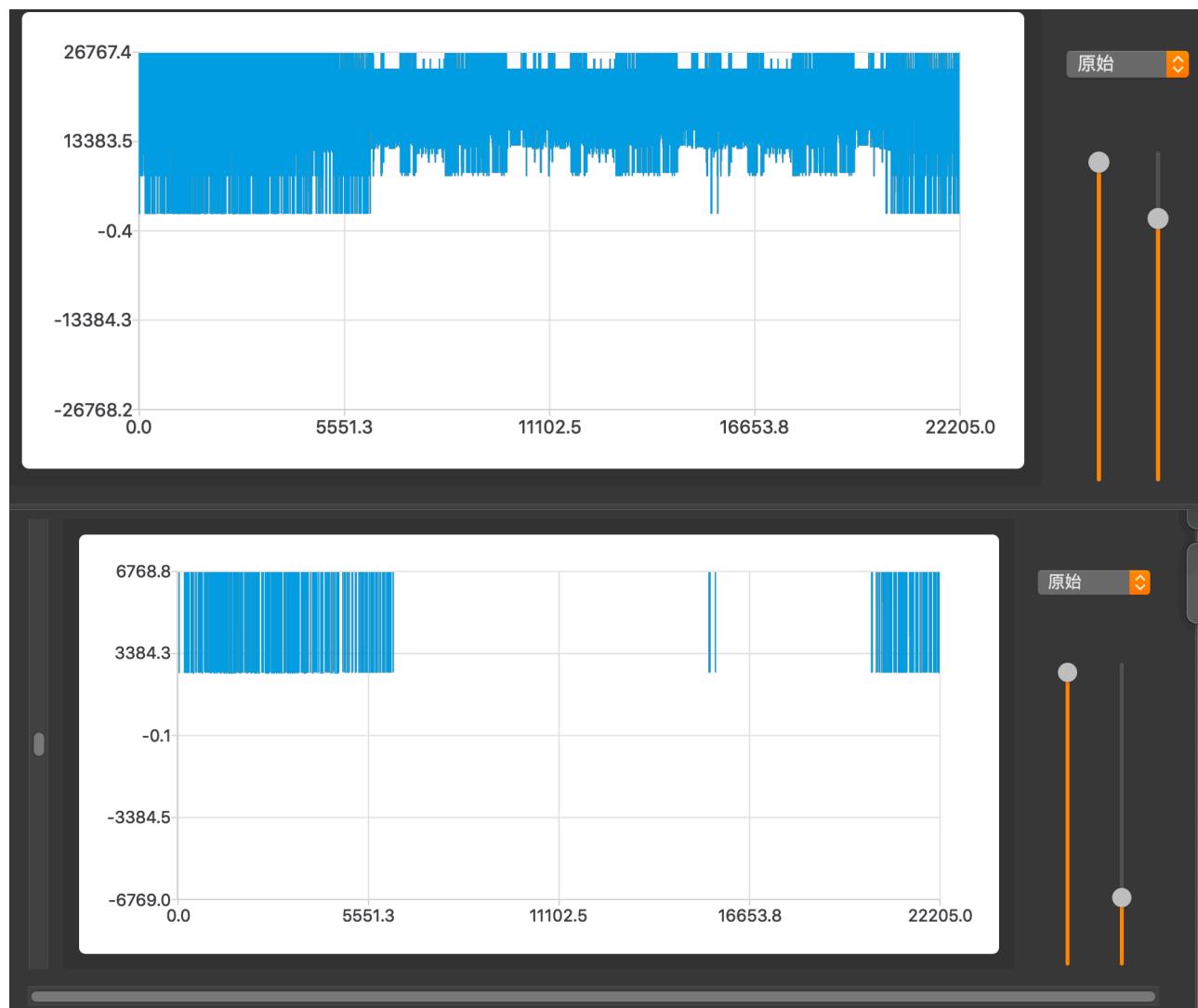
操作步骤：1.打开应用

- 2.点击菜单栏file下的open按钮
- 3.选择文件
- 4.点击确认
- 5.使用滑动块进行缩放

前置条件：配置文件正常，文件正常

期待输出：图表y轴方向随滑动块正常压缩放大

测试记录：



用例序号：6

功能点：对于图表进行横向滚动显示

用例名称：图表x轴滚动

操作步骤：1. 打开应用

2. 点击菜单栏file下的open按钮

3. 选择文件

4. 点击确认

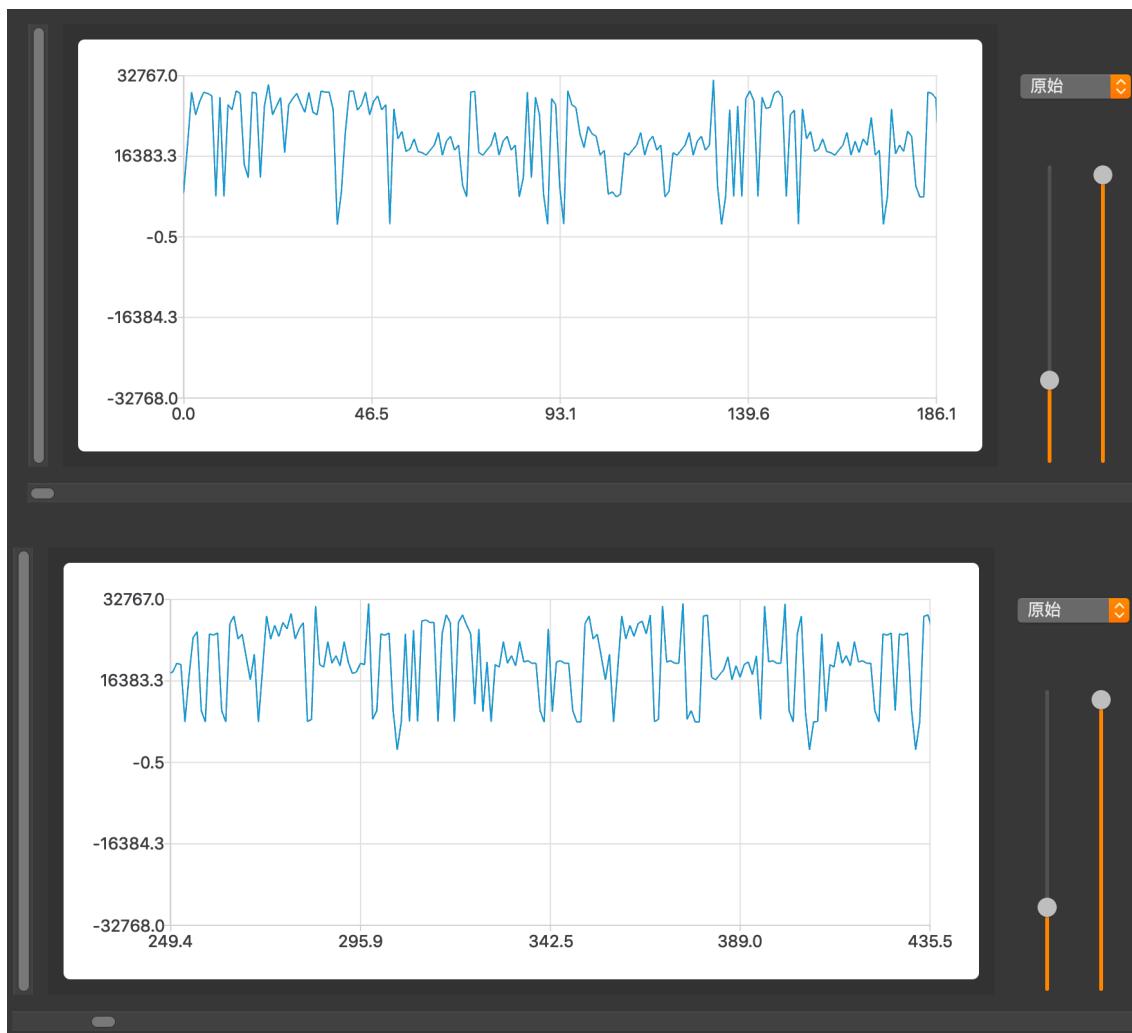
5. 使用滑动块进行缩放

6. 使用滚动条进行滚动显示

前置条件：配置文件正常，文件正常

期待输出：图表横向随滚动条正常滚动显示

测试记录：



用例序号：7

功能点：对于图表进行纵向滚动显示

用例名称：图表y轴滚动

操作步骤：1. 打开应用

2. 点击菜单栏file下的open按钮

3. 选择文件

4. 点击确认

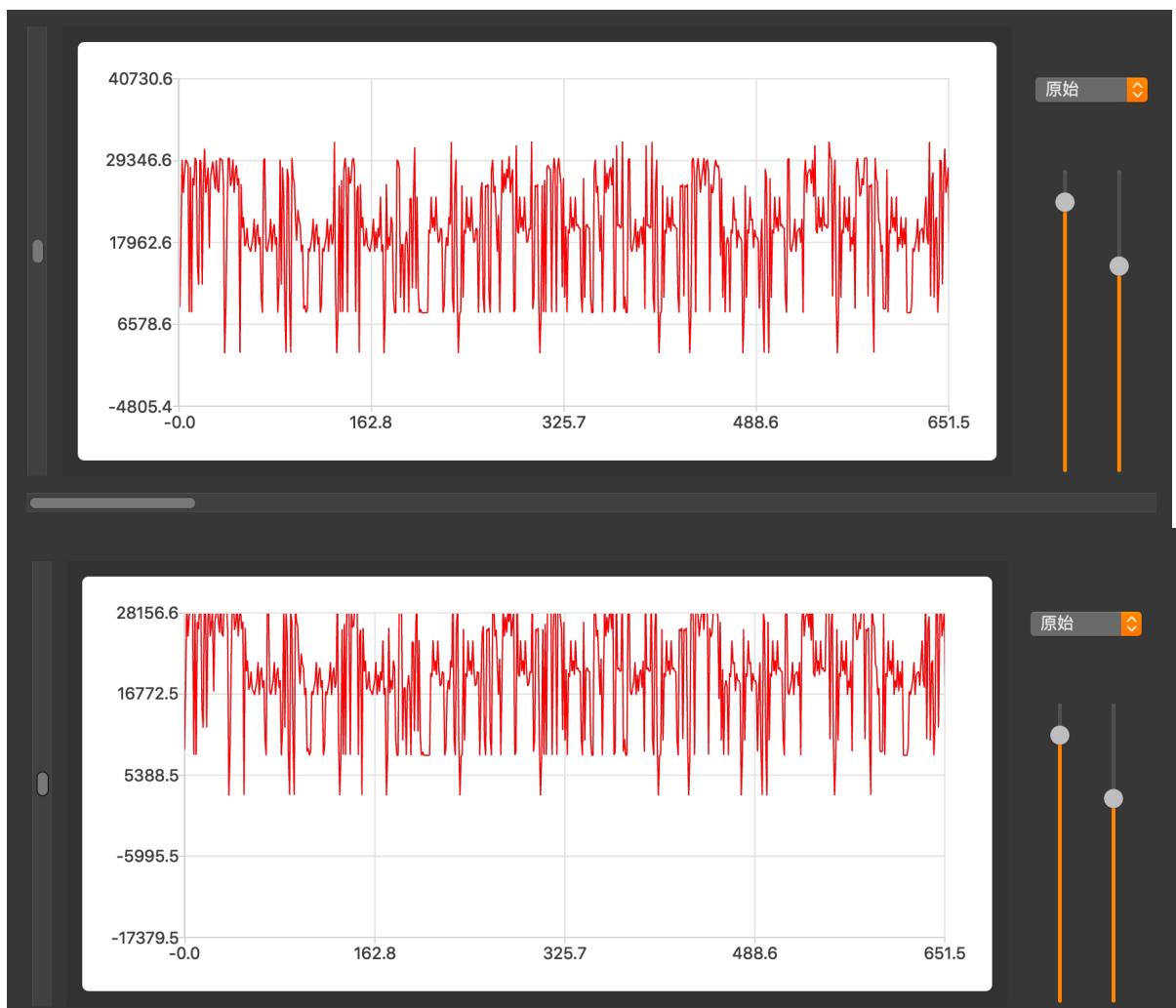
5. 使用滑动块进行缩放

6. 使用滚动条进行滚动显示

前置条件：配置文件正常，文件正常

期待输出：图表纵向随滚动条正常滚动显示

测试记录：



用例序号：8

功能点：打开多个文件，使用tab的方式显示

用例名称：以tab显示多个文件

操作步骤：1. 打开应用

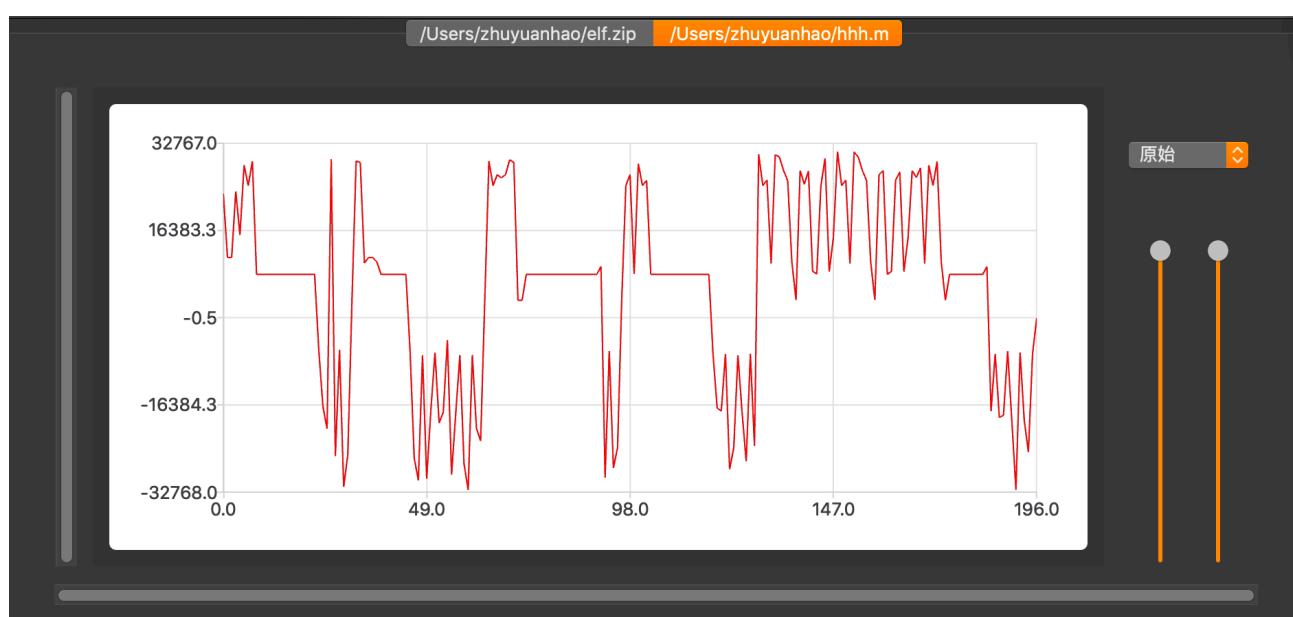
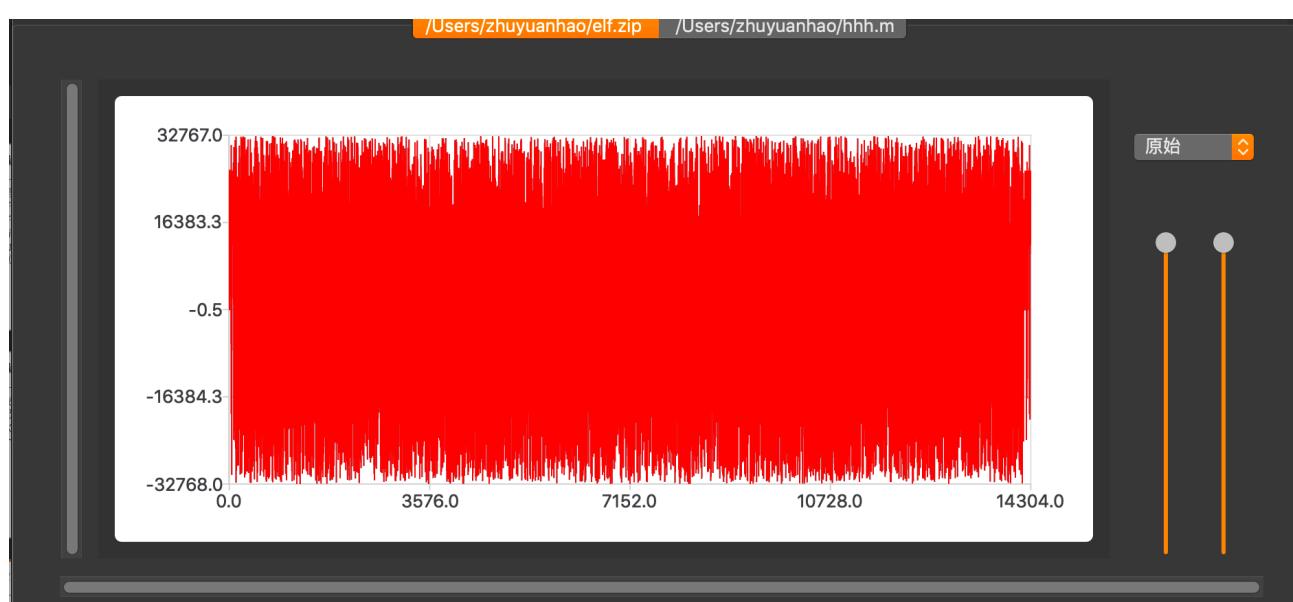
2. 点击菜单栏file下的open按钮

3. 选择文件

前置条件：配置文件正常，文件正常

期待输出：打开的多个文件展示在tab中，并且每一个文件的其他功能正常

测试记录：



用例序号：9

功能点：设置通道数

用例名称：用户正常设置通道数

操作步骤：1.点击setting菜单

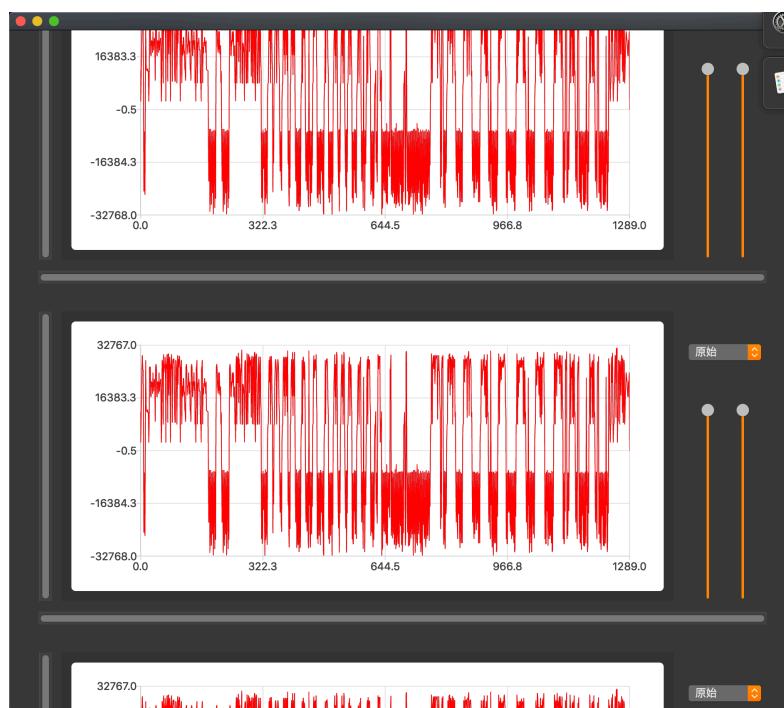
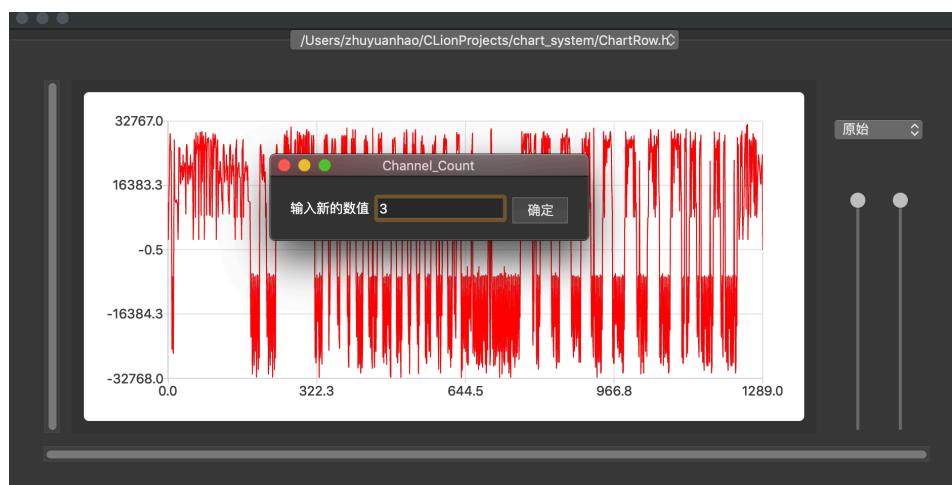
2.点击菜单中的Channel_Count项目

3.输入想要设置的通道数

前置条件：配置文件正常，文件正常，用户输入的通道数在1-5之间

期待输出：系统将通道数设置为用户指定的数量

测试记录：



用例序号：10

功能点：设置通道数

用例名称：用户设置通道数时输入错误的数字

操作步骤：1.点击setting菜单

2.点击菜单中的Channel_Count项目

3.输入想要设置的通道数

前置条件：配置文件正常，文件正常，用户输入的通道数在为小于0或是大于5

期待输出：系统提示输入非法

测试记录：



用例序号：11

功能点：设置曲线颜色

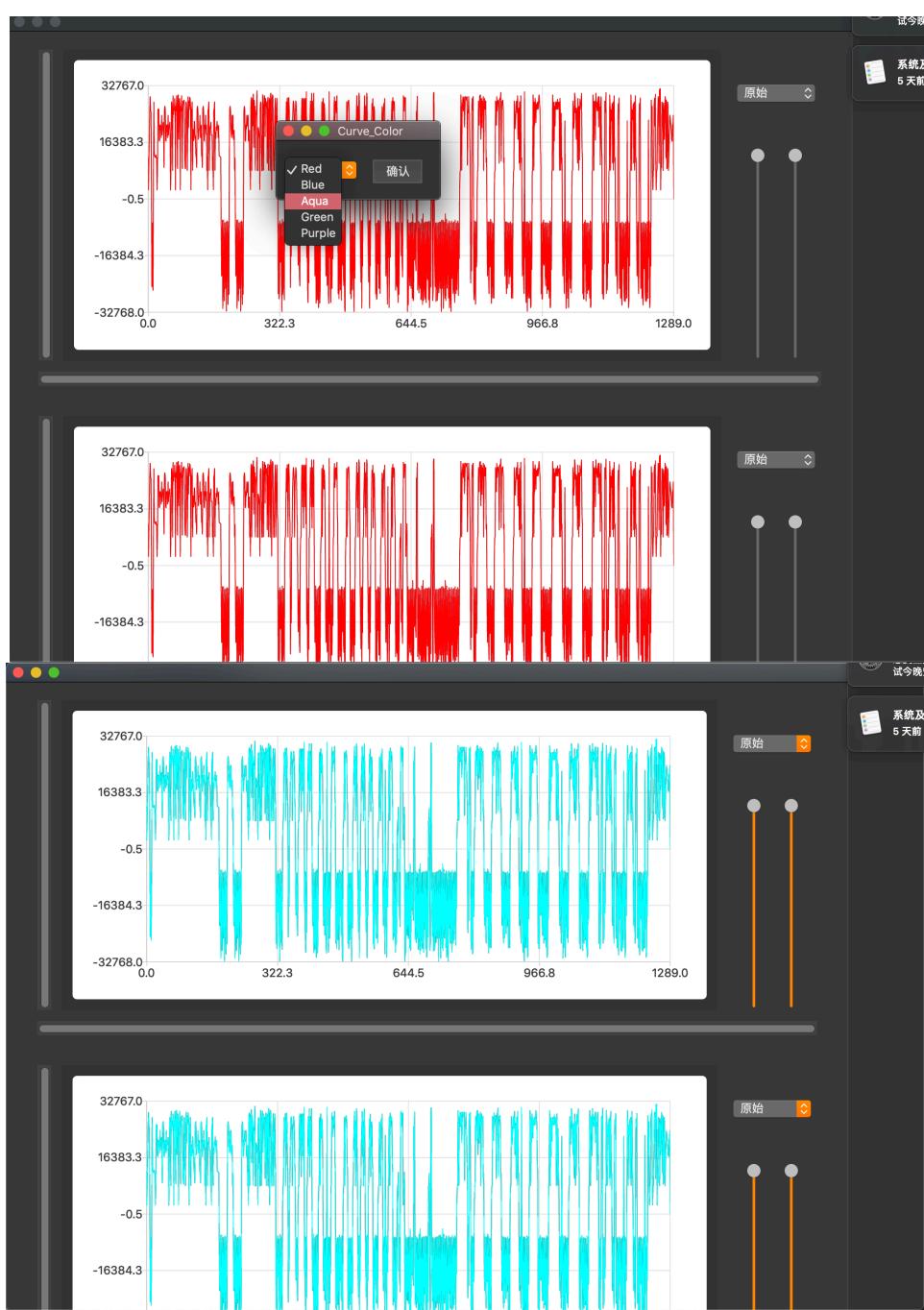
用例名称：用户设置曲线颜色

操作步骤：1.点击setting菜单
2.点击菜单中的Curve_Color项目
3.选择需要的曲线颜色

前置条件：配置文件正常，文件正常。

期待输出：系统变更所有通道的曲线颜色

测试记录：



用例序号：12

功能点：设置滚动速度

用例名称：用户正常设置滚动速度

操作步骤：1.点击setting菜单

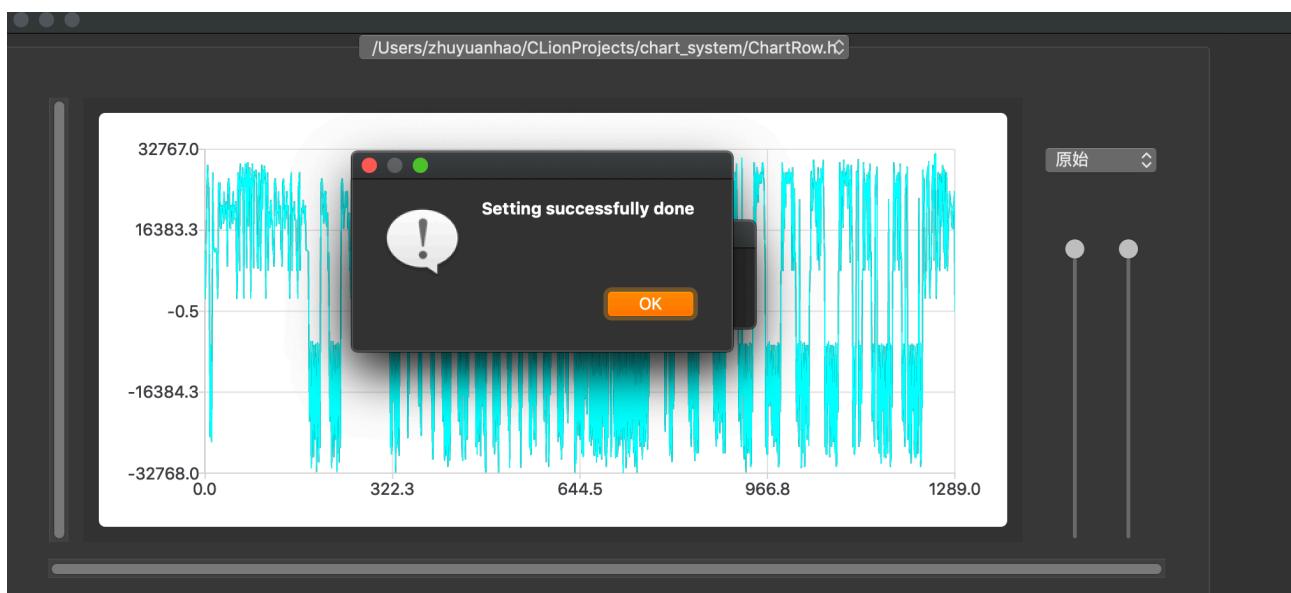
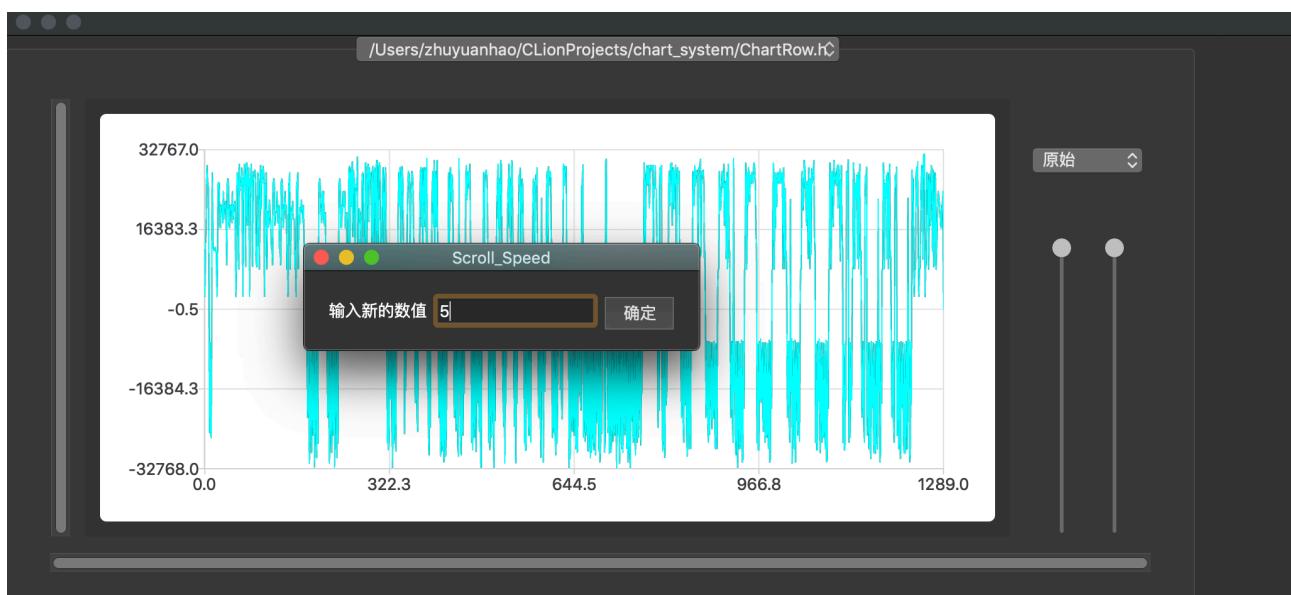
2.点击菜单中的Scroll_Speed项目

3.输入需要的新滚动速度

前置条件：配置文件正常，文件正常。用户输入值在正确范围

期待输出：系统变更曲线滚动速度

测试记录：



用例序号：13

功能点：设置滚动速度

用例名称：用户错误设置滚动速度

操作步骤：1.点击setting菜单

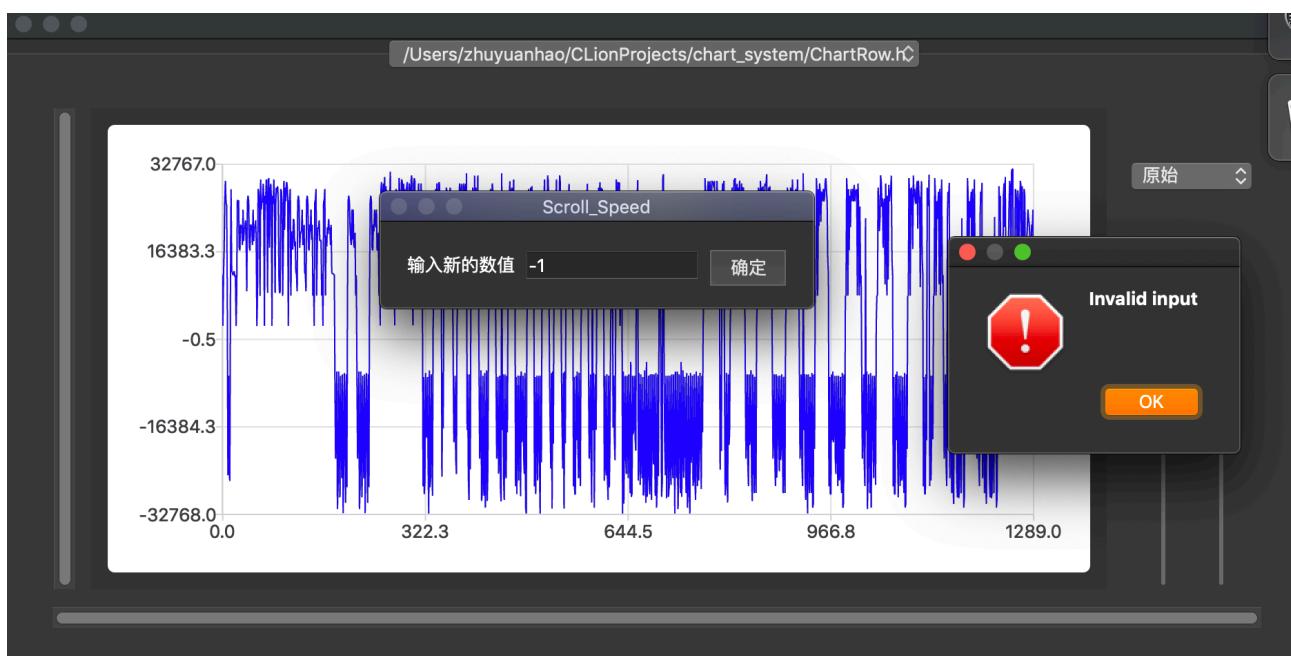
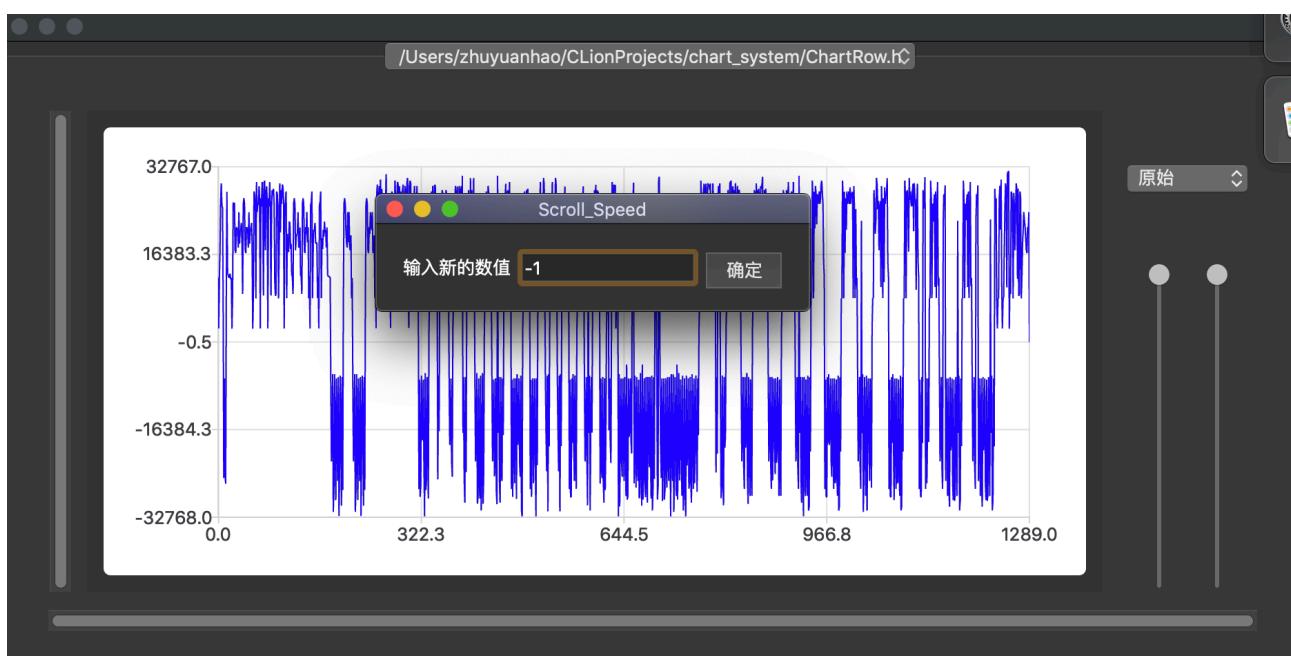
2.点击菜单中的Scroll_Speed项目

3.输入需要的新滚动速度

前置条件：配置文件正常，文件正常。用户输入值在错误范围

期待输出：系统提示输入非法

测试记录：



用例序号：14

功能点：通过选择对数据进行变换

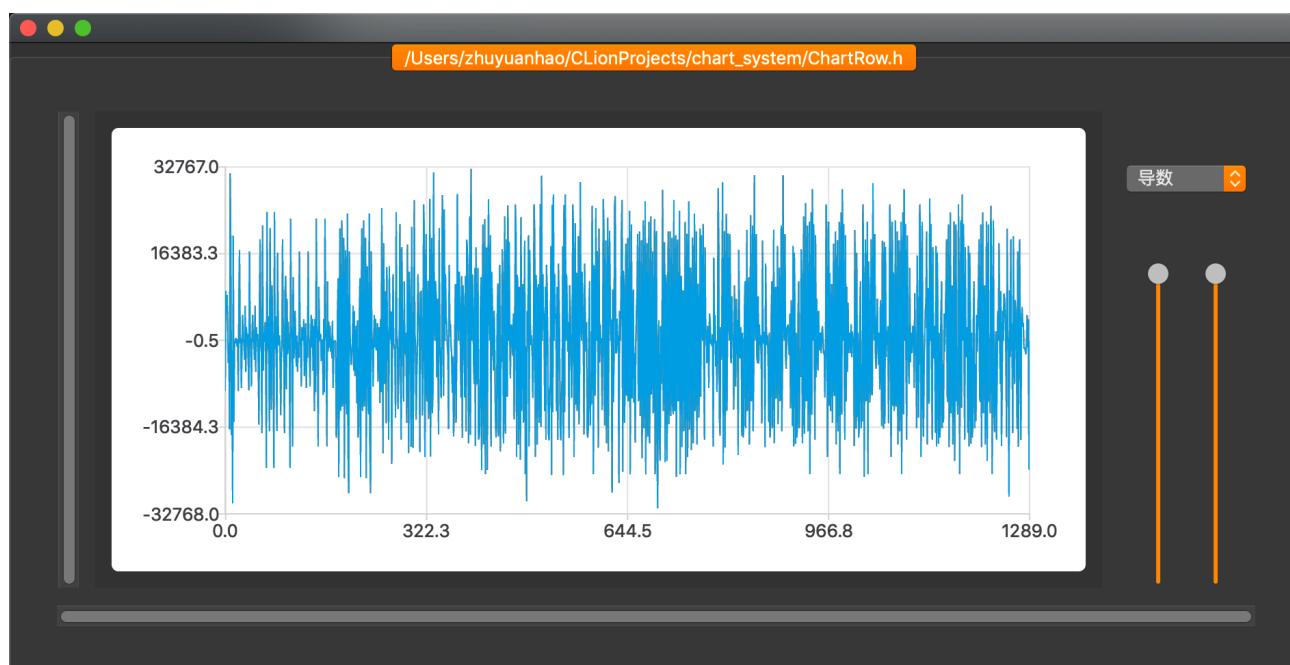
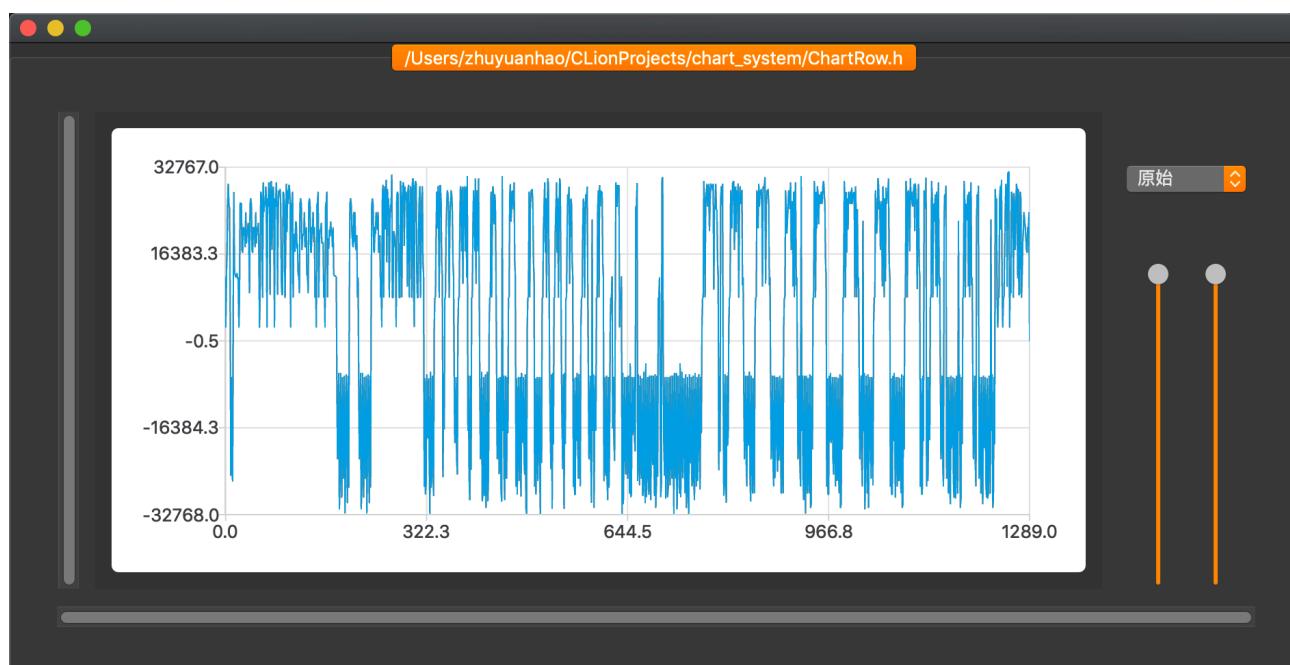
用例名称：变换通道数据，求导

操作步骤：1.在右侧combobox中选择求导

前置条件：配置文件正常，文件正常。

期待输出：系统给出对应的波形图

测试记录：



用例序号：15

功能点：通过选择对数据进行变换

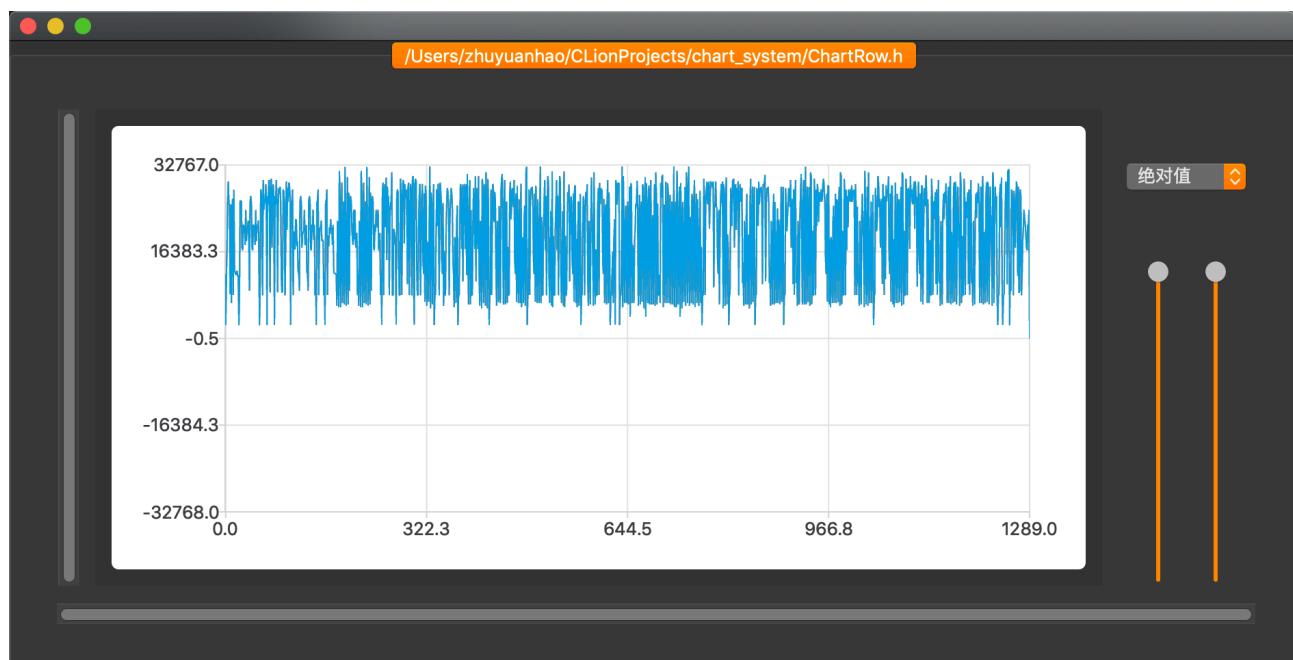
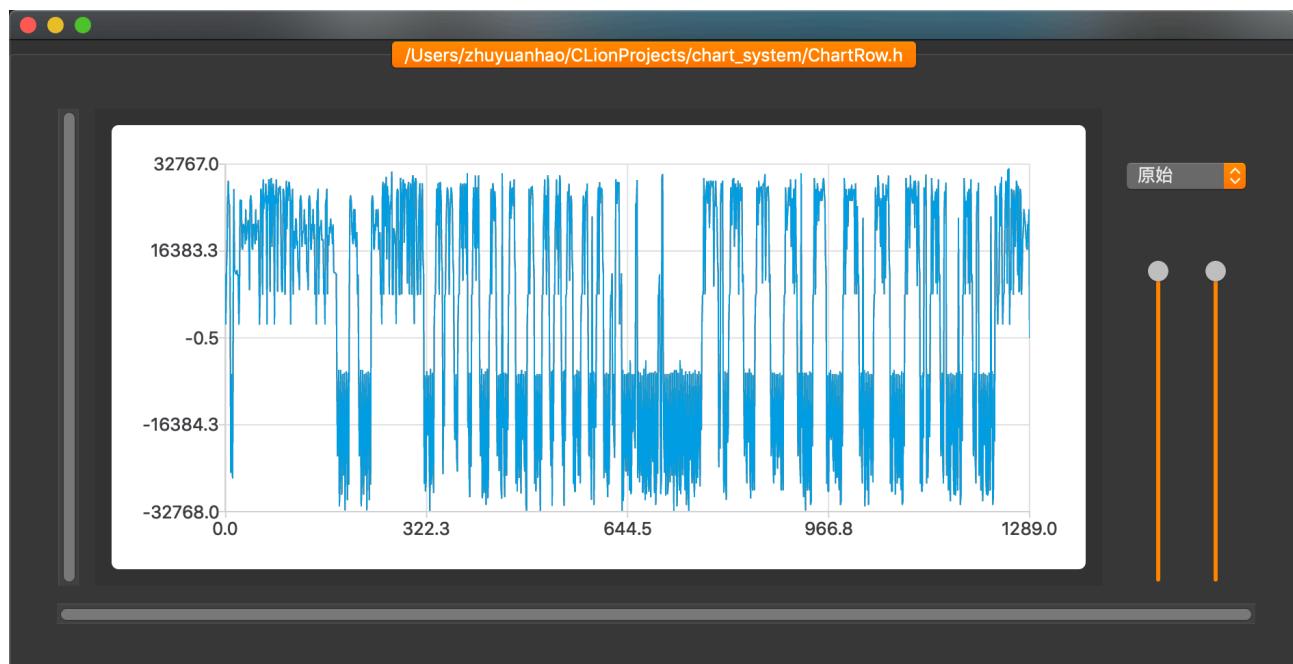
用例名称：变换通道数据，绝对值

操作步骤：1.在右侧combobox中选择绝对值

前置条件：配置文件正常，文件正常。

期待输出：系统给出对应的波形图

测试记录：



用例序号：15

功能点：通过选择对数据进行变换

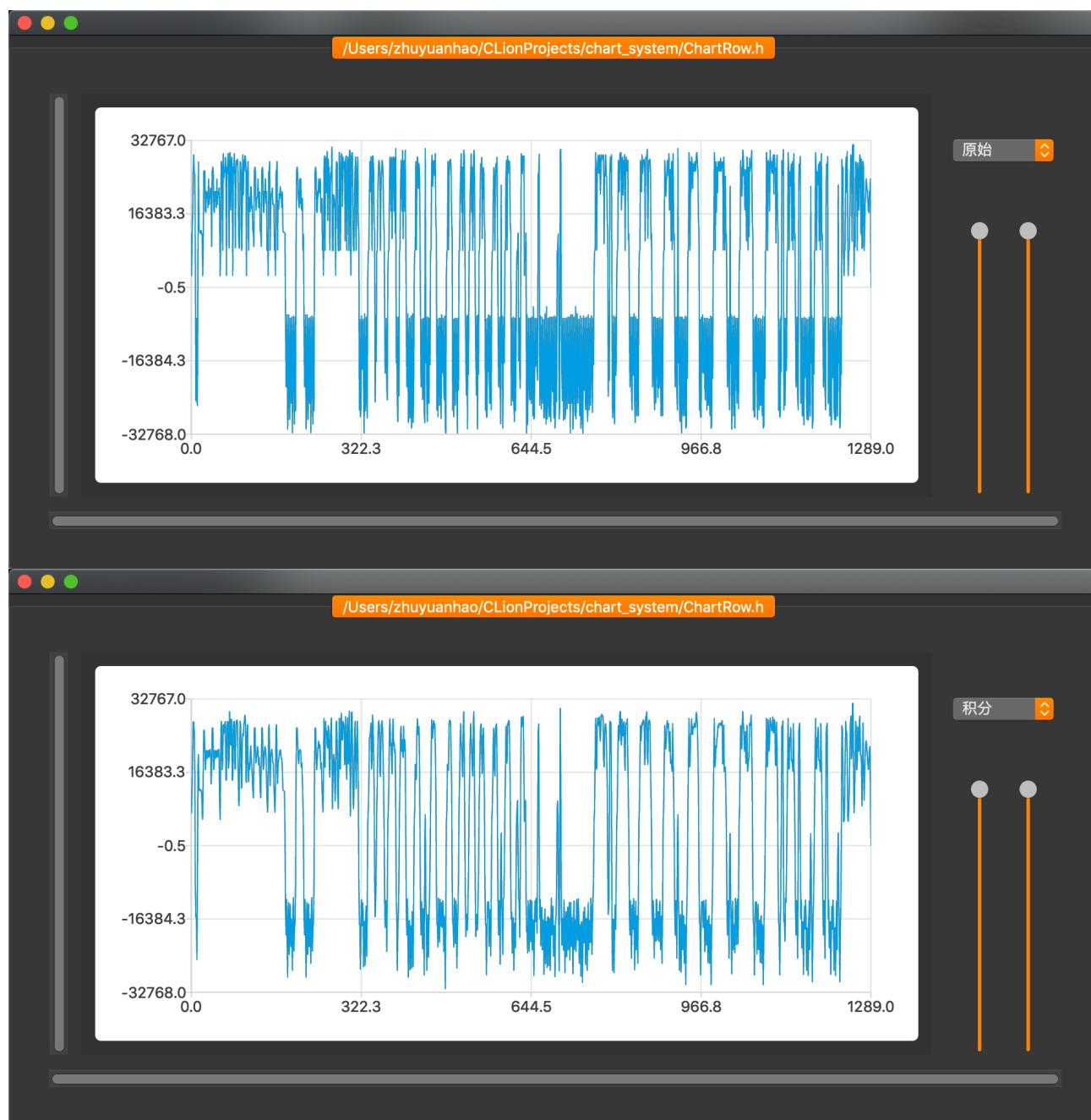
用例名称：变换通道数据，积分

操作步骤：1.在右侧combobox中选择积分

前置条件：配置文件正常，文件正常。

期待输出：系统给出对应的波形图

测试记录



5.系统总结

这一次的项目让我明白了如何将软件构造课程中的知识运用到实践当中，我明白了前期的设计以及需求分析的重要性，一开始对于软件做细节上的详细设计是不可能的，我们需要有对于需求的仔细分析，以及对于架构的设计，根据架构画出UML图，然后经过反复的迭代来达到我们的需求。

在设计这个项目的过程中，我非常刻意的想要实践我们在课程中所说到的对于高质量的子程序以及类的要求。经过实践，我才深刻的意识到，原来这两件事情在实际的实践中是非常不容易的，如何尽量的显示其他类对自己的访问，以及类与类之间如何有最小的可知性，是非常值得思考的一件事情。同时如何恰当的划分每一个类的功能，也是需要反复思考斟酌的事情，比如说，一开始的时候我把增加通道数这个职责也给到中央的主控制器，但是后来我就发现，这样以来中央控制器的内聚性好像就变得很差，什么事情都需要它来做，我们在设计类的时候不应该这么做，所以最后决定将添加通道数的职责划分到chartcontainer这个类，另一方面，对于数据的存放，到底是在有了一个dataprocessor类的情况下依然将数据在chartrow这个类中保存一份，还是说chartrow这个view类中直接不保存数据，而通过qt的信号槽机制来最大化view类的内聚性，在一开始的时候，我采取的是保存数据在charrow中的方式，但是最后，为了最大化内聚，chartrow中没有保存数据，这就是为什么每一次在初始化chartrow对象之后要调用其对应的dataprocessor对象来出发一次处理，然后因此通知chartrow来进行绘制。对于这样的斟酌与迭代改进，在项目的过过程中还有许多，而正是这样的反复斟酌和推翻之前想法的过程，让我逐渐了解了在构建一个软件的过程当中，设计部分的重要性，当我们完成一个系统的概念完整性之后，代码编程就真的只是将想法实践而已，同时预先设计好测试用例对于软件开发的前期准备来说也是极其重要的。

总的来说，我认为软件构造这门课是非常有用的一门课，这门课让我深刻的理解了在实践中一些非常好的措施，同时也通过实践获取了这一部分的知识，我从中收获了非常多的知识！

项目github地址：