郑州大学毕业设计（论文）

题　　目：基于 MI1000 的波形和时序生成系统的设计与实现

指导教师：郑志蕴　职称：教授

学生姓名：郭攀　学号：20102480211

专　　业：软件工程

院（系）：信息工程学院

完成时间：2014 年 5 月 4 日

2014 年 5 月 4 日

毕业设计（论文）任务书

题目来源：企业实践

<table>
<tr><td>课题名称</td><td colspan="5">基于 MI1000 的时序和波形生成系统的设计与实现</td></tr>
<tr><td>设计人姓名</td><td>郭攀</td><td>学号</td><td>20102480211</td><td>指导教师姓名、职称</td><td>郑志蕴、教授</td></tr>
<tr><td>指导时间/地点</td><td colspan="2">每周日/邮件</td><td>专业班级</td><td colspan="2">软件工程专业二班</td></tr>
</table>

一、设计（论文）内容

系统是基于 MI1000 系列硬件的配套软件。目的是方便用户产生相应的波形和时序数据，为逻辑分析仪、示波器插件分析模块提供波形和时序支持；同时将数据保存成文件，可以由 Mistudio（MI1000 系列硬件配套软件）配合 MI1000 硬件产生用户需要的电气信号。

二、设计（论文）的主要技术指标

开发工具：Qt Creator、跨平台 C++图形用户界面应用程序开发框架 Qt
语　　言：C++
开发流程：需求分析、总体设计、详细设计、编码实现
目　　标：开发一个易用的基于 MI1000 系列硬件的波形和时序生成系统

三、进度安排

2114.2.17 至 2014.2.23　收集所需资料完成开题报告
2014.2.24 至 2014.3.8　完成系统需求分析，对各个功能模块进行确定
2014.3.9　至 2014.3.23　完成设计工作
2014.3.24 至 2014.4.20　完成功能模块代码编写
2014.4.20 至 2014.4.27　对系统进行测试，找出缺陷进行完善
2014.4.28 至 2014.5.5　完成公司毕业答辩
2014.5.7 至 2014.5.20　完成学校论文编写及毕业答辩

四、毕业设计（论文）提交的文档及基本要求

1．毕业论文一本（包含封面、毕业设计任务书、开题报告、毕业设计中期检查 I、II、目录、中英文摘要、论文内容及参考文献、成绩评价意见等文件按上述顺序一块装订成册）
2．不少于 5000 汉字的科技翻译资料一份
3、毕业论文简介（A4 纸 1~2 页）（包含题目、专业、年级、姓名、指导教师、毕业论文所做的工作、解决的问题、创新之处等）
4．专业学习总结一份（不少于 5000 字）

毕业设计（论文）开题报告

课题名称	基于 MI1000 的波形和时序生成系统的设计与实现				
学生姓名	郭攀	学号	20102480211	专业班级	软件工程专业二班

一、 选题的目的意义

随着科技的迅猛发展，人民对电子技术的要求越来越高，仪器的精度要求也随之越高，容量越大，仪器能够采集更详细、存储更多的数据，但是分析能力有限，更详细、更大规模的数据需要用的 PC 机来处理，数据分析软件应运而生。

数据分析软件的研发的瓶颈就在于测试数据，连接硬件采集数据非常繁琐，而且采集到的数据量比较大，需要手工处理之后才能作为测试数据，也不够快速和及时及时，数据生成软件的重要性日益凸显。

二、国内外研究综述

近几年，国内厂商在数字示波器市场开拓及技术创新方面取得了很好的成绩，涌现出像：鼎阳科技、普源精电、优利得、绿杨、利利普、同惠电子、中策等公司且纷纷推出自己的数字示波器，竞争也基本进入白炙化状态。但与国外大公司相比，国内企业仍有很大的差距，如产品主要针对低端市场，无法满足高速测试需要，测试解决方案扩展低。

三、毕业设计（论文）所用的方法

将数据波形或协议设置、数据生成模块单独拿出来，放到动态链接库中，可以有效的提高系统的可扩展性；使用 Qt 进行界面开发，有效地规避细节问题，提高开发效率；使用线性插值算法，使得波形绘制保持高的保真度与绘制速度。

四、主要参考文献与资料获得情况

1 Jasmin Blanchette,Mark Summerfield 著.C++ GUI Qt 3.第二版.电子工业出版社.2008-8

2 蔡志明著.精通 Qt4 编程.电子工业出版社.2008-1-1

3 百度百科.MVC 框架.http://baike.baidu.com/view/5432454.htm?from_id=85990&type=search&fromtitle=mvc&fr=aladdin

4 百度百科.线性插值法. http://baike.baidu.com/link?url=0ZCQ274pJ9mSLwPjqnefpOnFvXz6aC0EjYRoIxLGI3dAWpLQMBKTXctqWOjn0aHt43yNdta7SNsVF8qkAnAPpq

五、指导教师审批意见

签字： 年 月 日

毕业设计工作中期检查Ⅰ

年　月　日

课题名称	基于 MI1000 的波形和时序生成系统的设计与实现				
姓　　名	郭攀	专业和班级	软件工程专业二班	指导教师	郑志蕴

一、毕业设计具体内容、目标和可能遇到的问题

系统是基于 MI1000 系列硬件的配套软件。目的是方便用户产生相应的波形和时序数据，为逻辑分析仪、示波器插件分析模块提供波形和时序支持；同时将数据保存成文件，可以由 Mistudio（MI1000 系列硬件配套软件）配合 MI1000 硬件产生用户需要的电气信号。

基本功能分两类：数字信号和模拟信号。其中数字信号基本功能包括 UART、CAN、Wiegand 协议的设置、生成、预览、保存功能；模拟信号基本功能包括正弦波、方波、脉冲波、梯形波、三角波、锯齿波的设置、生成、预览、保存功能。

扩展功能：SPI、1-wire、LIN、I2C 协议的设置，生成，预览，保存功能；绘图板功能（可以由用户绘制自己想要的模拟波形）。

可能遇到的问题：项目进度问题、需求变更问题。

二、采取的研究方法、技术路线、实验方案及可行性分析

项目进度问题：严格按照项目规定的时间进行，若是某阶段逾期，则在之后的时间里加班

需求变更问题：需求期间重视需求的完整性以及需求来源的多样性；设计阶段重视软件结构，设计出具有弹性的软件结构；编码阶段注重编码规范，便于需求变更引起时重新修改代码。

三、指导教师对学生出勤、文献阅读等方面的评语

签字：　　年　月　日

毕业设计工作中期检查Ⅱ

2014 年 5 月 8 日

课题名称	基于 MI1000 的波形和时序生成系统的设计与实现				
姓　　名	郭攀	专业和班级	软件工程专业二班	指导教师	郑志蕴

一、阶段性结果

项目完成了以下模块：选择协议或波形类型、配置协议或波形参数、生成协议或波形数据、预览、保存数据到文件，已经可以解决处理用户的主要需求。

二、存在的问题

1) 过低的估计了 UI 界面设计的难度,只是站在开发人员的角度设计操作习惯而没有考虑其他不同类型用户的使用习惯。
2) 波形设置对话框无法停靠到主窗体上，使得用户随时查看当前波形设置；而只能作为一个模态对话框，当需要设置的时候显示，需要绘制波形的时候隐藏。
3) 即使使用了线性插值算法算法，但在实际操作过程中，绘图板模块保存到文件的数据被重新读取后，会出现锯齿。

三、后一步工作设想

尝试解决以上存在的问题、编写绘图板模块、开始论文的编写

四、指导教师对学生出勤、论文进展方面的评语

签字：　　年　　月　　日

基于 MI1000 的波形和时序生成系统的设计与实现

摘要 基于 MI1000 的波形和时序生成系统是为给不同知识背景用户使用并生成他们想要的波形数据或时序数据的软件，该系统将生成波形或时序数据以图形化方式展示,当用户确认想要的数据后，可以通过 MI1000 系列硬件生成与数据对应的电气信号，用来满足用户的需要。

系统具有以下功能：选择协议或波形类型、配置协议或波形参数、生成协议或波形数据、预览、保存数据到文件、波形绘图板。为了扩展性，系统可选择的协议或波形以满足一定接口要求的动态链接库的形式提供，可配置参数也由动态链接库中实现的对话框规定；每份配置都将由系统生成相应的数据，这些数据将会通过线性插值算法映射到屏幕上，以供用户查看预览的波形或时序；如果数据满足用户的需要，系统将会将这些数据按照 MI1000 规定的格式保存到文件，以满足用户的需要。

关键词 MI1000、波形、时序

Abstract ***Wave form and timing generation system based on MI1000*** is a software offering wave form data and timing data for users in different knowledge background. This system will display the generated wave form and timing data in a graphic mode, the moment the users confirm data, MI1000 series hardware will conform to the data in electric signals.

Functions of the system: select the protocol or waveform types, configurate protocol or waveform parameters, generate protocol or waveform data, preview and save data to files and drawing board. The system will choose agreement or waveform to provide dynamic link library for its expansibility, the configurable parameters also are achieved by the dynamic link library dialogue box set. Each configuration will be generated by the system of corresponding data. The data will be mapped to the screen through linear interpolation algorithm in order that the users can preview waveform or timing data; if the data meet the needs of users, the system will apply these data in accordance with the provisions of MI1000 format file, to meet the needs of users.

Keyword MI1000、protocal、timing generation

1 背景

科学技术的发展推动了社会的发展与变化，方便并丰富了人们的生活。现代通信技术的进步，使得古人幻想的千里眼和顺风耳变成了现实。藉此，我们可以很快了解到自己周围、祖国各地、世界各国发生的重要事情，使得地球真正的变成了一个地球村。

通信技术占据如此重要的地位，那么如何保证通信过程中数据传输的正确性？致远电子推出的 MI1000 系列硬件揉合了日常通信电路调试过程中所需的信号产生、测量等多种功能，构建了一个功能强大的闭环测试系统，为信号调试提供了极佳的仪器解决方案。

但是，通信协议多种多样，Mistudio（MI1000 系列硬件配套的软件）只包含最简单通信协议的信号产生、测量功能，在实际工作中不能满足开发人员的需要，因此需要一个可以扩展的时序和波形生成工具，辅以 MI1000 系列硬件以及 Mistudio 来生成开发人员需要的通信信号。

1.1 系统介绍

系统是基于 MI1000 系列硬件的配套软件。目的是方便用户产生相应的波形和时序数据，为逻辑分析仪、示波器插件分析模块提供波形和时序支持；同时将数据保存成文件，可以由 Mistudio（MI1000 系列硬件配套软件）配合 MI1000 硬件产生用户需要的电气信号。

1.2 国内外研究现状

数字示波器自上个世纪七十年代诞生以来，其应用越来越广泛，已成为测试工程师必备的工具之一。随着近几年来电子技术取得突破性的发展，全世界数字示波器市场进一步扩大，而作为在世界经济发展中扮演重要角色的中国，飞速发展的电子产业也催生了更庞大的数字示波器需求市场。

近几年，国内厂商在数字示波器市场开拓及技术创新方面取得了很好的成绩，涌现出像：鼎阳科技、普源精电、优利得、绿杨、利利普、同惠电子、中策等公

司且纷纷推出自己的数字示波器，竞争也基本进入白炙化状态。但与国外大公司相比，国内企业仍有很大的差距，如产品主要针对低端市场，无法满足高速测试需要，测试解决方案扩展低。

高端数字示波器市场主要由三大生产厂家主导，安捷伦科技、泰克、力科。国内早期则以普源精仪最为壮大，但这里不得不说国内示波器的黑马“鼎阳科技”（Siglent），这是一家专业的数字示波器生产厂商，专注于高性能数字示波器的开发研究。2002 年鼎阳的创始人就开始对数字示波器进行深入研究，随后推出实时采样率高达 1G 且具有多项原创专利技术的示波器，推出的第一台示波器就比同级别产品触发精度高 2—4 倍，令业界震撼！

2 系统相关技术和工具

2.1 MI1000 系列硬件

MI1000 系列硬件是致远电子推出的多功能虚拟仪器。它揉合了日常电路调试过程中所需的信号产生、测量等多种功能，构建了一个功能强大的闭环测试系统，既有数字、模拟信号的激励，亦有数字、模拟信号的检测，且逻辑分析仪还嵌入了常用信号的协议分析功能。

MI1000 系列硬件功能模块间可协同工作，控制简单，观测方便，在有效地节约工作台面积的同时，也为信号调试提供了极具性价比的仪器解决方案。作为一款虚拟仪器，其充分利用了上位机强大的数据处理能力，为用户预留二次开发的程序接口。

它集六种仪器于一身：

1) 数字示波器：双通道，60MHz 带宽，4M 最大储存深度。

2) 逻辑分析仪：32 通道输入。

3) 信号发生器：单通道，最高 40MHz 正弦波输出，支持多种信号调制。

4) 码型发生器：16 通道输出，多种跳转指令，用于数字信号激励。

5) 协议分析：支持 A/D 总线、UART、I2C、SPI 等。

6) 扫频仪：测量器件或设备传递函数的幅频特性。

2.2 线性插值算法

线性插值是数学、计算机图形学等领域广泛使用的一种简单插值方法。它经常用来补充函数表中不存在的表项。以下是对线性插值算法的介绍：

假设知坐标(x0,y0)与(x1,y1),要得到[x0,x1]区间内某一位置 x 在(x0,y0)与(x1,y1)构成的直线上的 y 值。

首先得到（y-y0）(x1-x0)=(y1-y0)(x-x0)。

然后假设方程两边的值为 a，那么这个值就是插值系数—从 x0 到 x 的距离与从 x0 到 x1 距离的比值。由于 x 值已知，所以可以从公式得到 a 的值：a=(x-x0)/(x1-x0)。同样，a=(y-y0)/(y1-y0)

最后，在代数上就可以表示成为：y = y0 + a(y1 - y0)。

2.3 MVC 框架模式

MVC 是一个框架模式，它强制性的使应用程序的输入、处理和输出分开。使用 MVC 应用程序被分成三个核心部件：模型（Model）、视图（View）、控制器（Ccontrol）。它们各自处理自己的任务。MVC 结构提供了一种按功能对各种对象进行分割的方法（这些对象是用来维护和表现数据的），其目的是为了将各对象间的耦合程度减至最小。MVC 被广泛应用于网站设计，但是实际上它适用于任何类型的需要处理较为复杂数据并显示相关信息的应用程序。

2.3.1 MVC 框架模式规定的操作流程

图 2-1 为 MVC 框架一般结构。

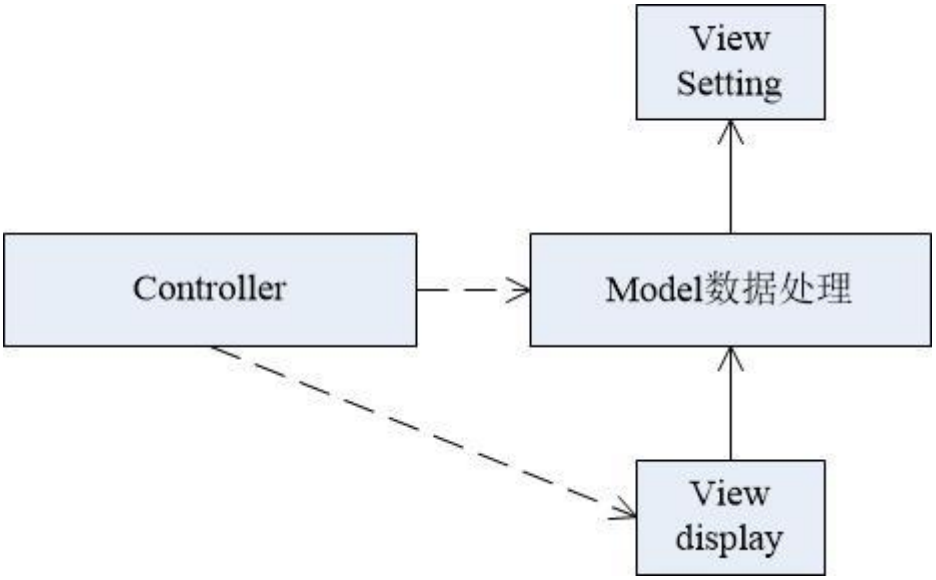

图 2-1 MVC 框架模式一般结构

1) 用户发出请求

2) Controller 获取请求，并通知 Model 数据处理模块向 ViewSetting 请求数据

3) ViewSetting 收集当前显示层的数据，并交付 Model 数据处理模块

4) Model 数据处理模块获取数据，生成相应的数据

5) Controller 通知 ViewDisplay 从 Model 数据处理请求数据

6) Model 数据处理模块将波形数据交付给 ViewDisplay

7) ViewDisplay 进行显示

2.3.2 MVC 框架模式的特性

MVC 框架模式有以下特性：

1) 多个视图可以对应一个模型。按 MVC 设计模式，一个模型对应多个视图，可以减少代码的复制及代码的维护量，一旦模型发生改变，也易于维护。

2) 应用被分隔为三层，降低了各层之间的耦合，提供了应用的可扩展性。

3) MVC 更符合软件工程化管理的精神。不同的层各司其职，每一层的组件具有相同的特征，有利于通过工程化和工具化产生管理程序代码。

2.3.3 使用 MVC 框架模式开发应用程序的优势

MVC 分层有助于管理复杂的应用程序，可以强制性地使得开发人员在一个时间内只关注一个方面，如设计试图的开发人员不需要考虑业务实现。同时它简化了分组开发，不同的开发人员可同时开发视图、控制器逻辑和业务逻辑。

2.4 图形用户界面应用程序框架 Qt

Qt 是 1991 年奇趣科技开发的一个跨平台的 C++图形用户界面应用程序框架。它提供给应用程序开发者建立艺术级的图形用户界面所需的所有功能。Qt 很容易扩展，并且允许真正地组件编程。基本上，Qt 同 X Window 上的 Motif，Openwin，GTK 等图形界面库和 Windows 平台上的 MFC，OWL，VCL，ATL 是同类型的软件产品。

使用 Qt 开发应用程序具有以下优势：

1) 优良的跨平台特性

Qt 支持下列操作系统:MS/Windows－95、98、NT4.0、ME、2000、XP、Vista、Win7、win8、win2008、Unix/X11－Linux、SunSolaris、HP-UX、CompaqTru64 UNIX、IBMAIX、SGI IRIX、FreeBSD、BSD/OS 和其它很多 X11 平台、Macintosh -Mac OS X、Embedded - 有帧缓冲(framebuffer)支持的嵌入式 Linux 平台、Windows CE

2) 面向对象

Qt 的良好封装机制使得 Qt 的模块化程度非常高，可重用性较好，对于用户开发来说是非常 方便的。Qt 提供了一种称为 signals/slots 的安全类型来替代 callback，这使得各个元件之间的协同工作变得十分简单。

3) 丰富的 API

Qt 包括多达 250 个以上的 C++类，还提供基于模板的 collections，serialization，file，I/O device，directory management，date/time 类。甚至还包括正则表达式的处理功能。

4) 支持 2D/3D 图形渲染，支持 OpenGL

5) 大量的开发文档

6) XML 支持

3 系统分析与设计

3.1 需求分析

3.1.1 功能性需求

《波形和时序生成系统》是基于 MI1000 系列硬件配套软件。它的主要功能包括：

基本功能分两类：数字信号和模拟信号。其中数字信号基本功能包括 UART、CAN、Wiegand 协议的设置、生成、预览、保存功能；模拟信号基本功能包括正弦波、方波、脉冲波、梯形波、三角波、锯齿波的设置、生成、预览、保存功能。

扩展功能：SPI、1-wire、LIN、I2C 协议的设置，生成，预览，保存功能；绘图板功能（可以由用户绘制自己想要的模拟波形）。

注：协议对应数字信号，波形对应模拟信号。

3.1.2 非功能性需求

必须满足的非功能性指标：

1) 系统必须具有可扩展性：系统需要很方便的扩展可以处理的波形或时序的集合。

2) 系统必须具有可移植性：系统有可能在将来移植到其他的系统平台上。

3) 波形或时序预览时，绘制的图形不能出现锯齿。

4) 系统至少可以在连续 10 个小时，并且伴随不定时操作的情况下正确处理用户的操作。

5) 当用户设置波形或时序参数后，到要被绘制的图形出现之间的时间不能超过 1s。

6) 系统具有易用性，可以满足种知识背景的用户的使用。

3.1.3 外部接口

用户通过设置并预览得到自己需要的数据之后，需要将数据按照规定的格式

保存到文件，数据遵循下面格式：

1）模拟信号数据文件：数据文件为文本文件；一个模拟信号波形周期有 2048 个点，对模拟函数进行采样，将采样数据进行处理，最小值为 0，最大值为 1023。

2）数字信号数据文件：数据文件为二进制文件；共包括 4096 个数据，第 0~2048 个数据表示数据,第 2049~4096 数据表示命令;每个数据使用 32 位存储，每一位代表协议的一条总线。

3.2 总体设计

3.2.1 设计思路

1）系统定位

明确系统的定位：

① 它是基于 MI1000 系列硬件的配套工具。

② 它是为逻辑分析仪、示波器插件分析模块提供时序和波形支持。

2）核心问题

根据用户的需求生成特定格式的文件。

① 输入：用户的操作，也就是用户对波形或协议的设置

② 处理：根据用户输入生成数据

③ 输出：根据生成的数据绘制相应的波形或时序、根据生成的数据保存成特定格式的文件

3）细化处理流程

根据以上的处理流程以及实际情况，将处理流程进行细化为五个部分：选择协议或波形、协议或波形配置、数据生成、波形或协议预览、保存文件。

3.2.2 总体框架

为了降低耦合度，用了 MVC 的架构，系统整体结构设计如图 3-1。

。

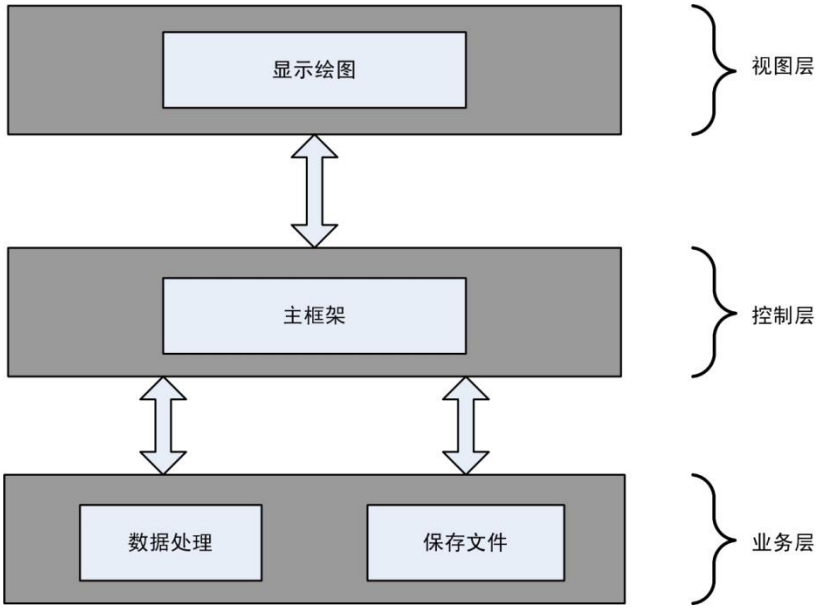

图 3-1 总体框架图

3.3 详细设计

3.3.1 操作流程

图 3-2 描述了软件的执行流程。

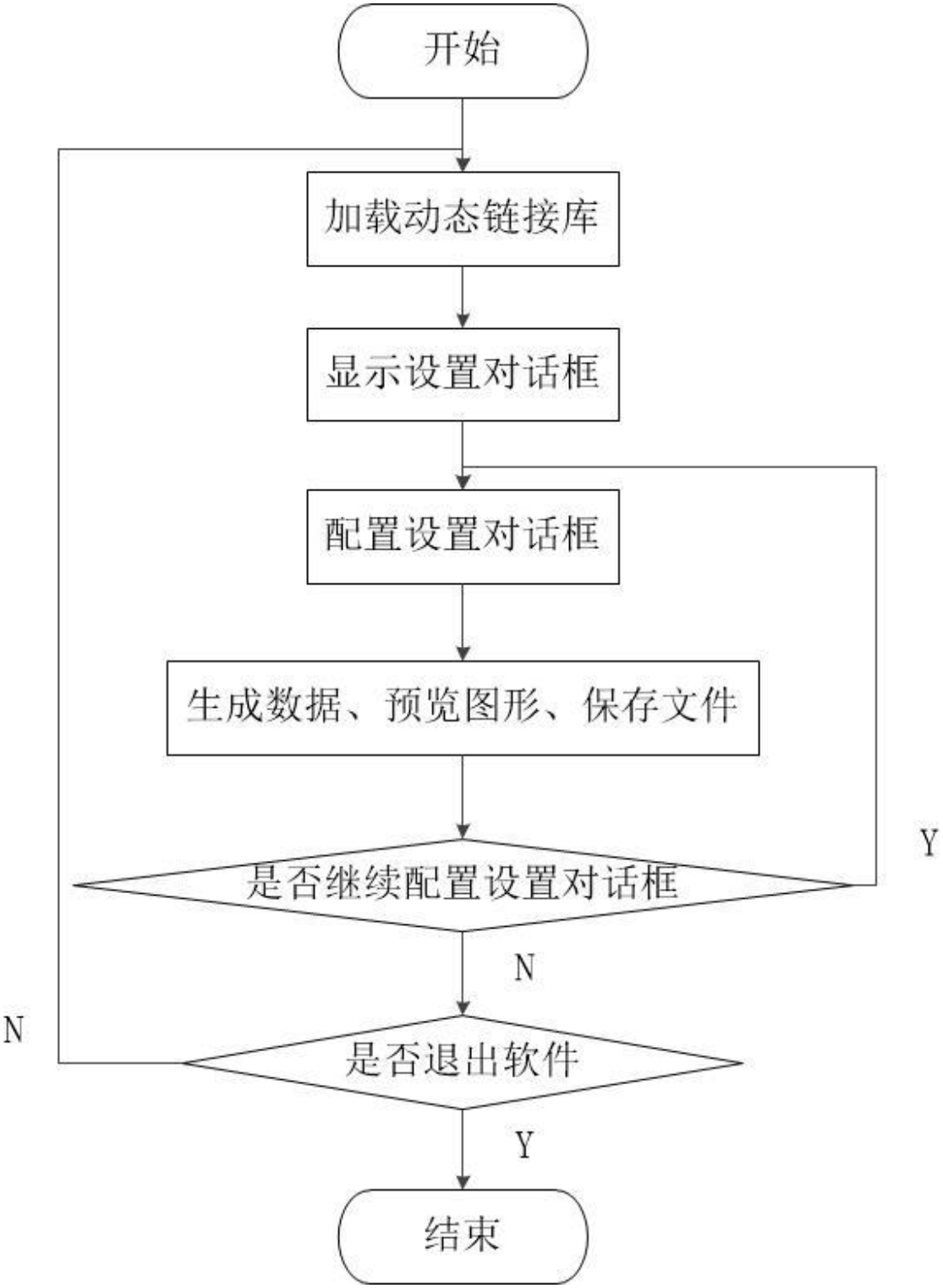
开始
加载动态链接库
显示设置对话框
配置设置对话框
生成数据、预览图形、保存文件
是否继续配置设置对话框
Y
N
是否退出软件
N
Y
结束

图 3-2 操作流程图

3.3.2 模块划分

通过分析，获取到五个模块：选择协议或波形模块、协议或波形配置模块、形并按照规定格式保存数据的绘图板模块。图 3-3 为系统模块结构图。数据生成模块、预览模块、保存文件模块；再加上一个可以支持用户自己绘制波

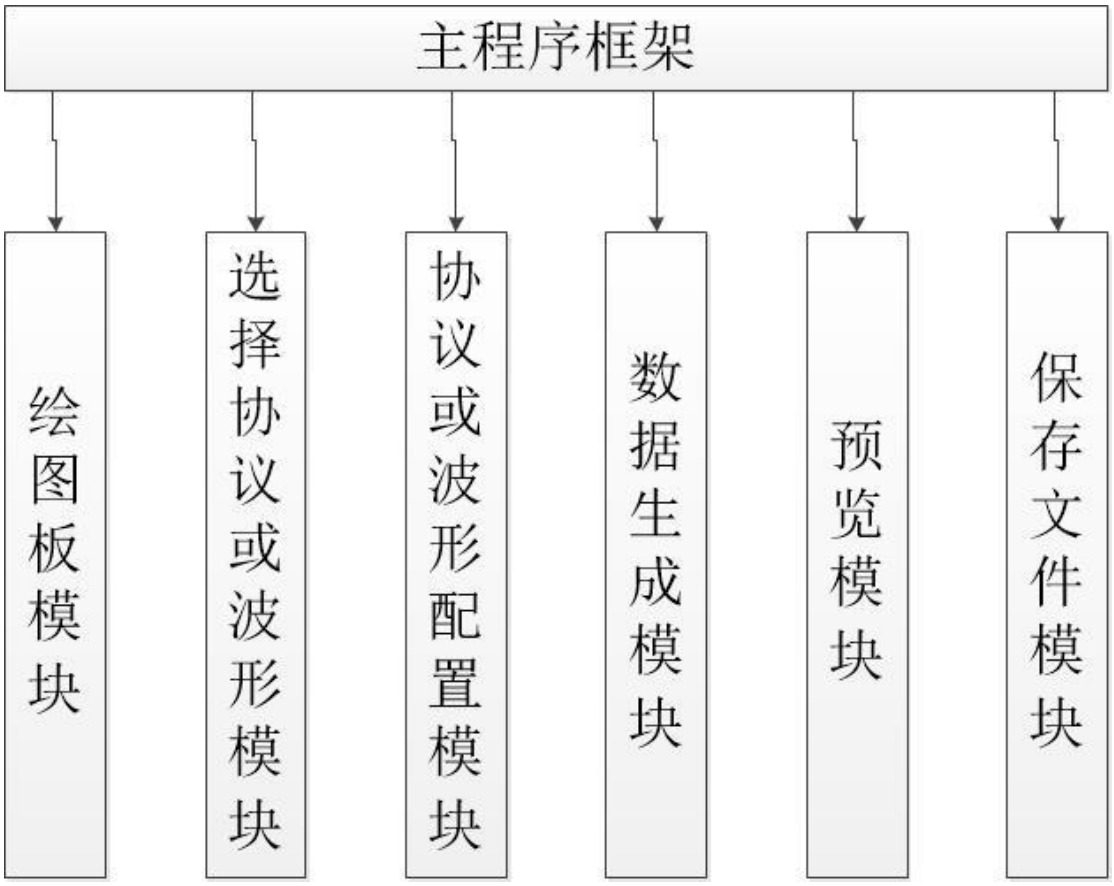

图 3-3 系统模块图

除了绘图板模块较为独立外，选择协议或波形模块、协议或波形配置模块、数据生成模块、预览模块、保存文件模块共同协作完成系统的主要任务。

3.3.3 选择协议或波形模块设计

选择协议或波形模块的功能是查看指定（plugin）文件夹下的动态链接库文件，将动态链接库的自省信息列出来，供用户选择。基本流程如图 3-4。

图 3-4 选择协议或波形模块

3.3.4 预览模块设计

预览模块是界面部分的重点部分。因为模拟信号与数字信号的巨大差异，它们需要分开处理。

1) 模拟信号预览

模拟信号的预览，需要描绘出波形的起伏，也就是在一个周期内波形的形态。

文件格式要求，使用 2048 个整数来表示一个波形周期，而且每个整数要求必须在 0-1023 之间；如何将这 2048 个点绘制到屏幕上一个指定的矩形区域而不会使波形失真便是模拟波形预览要解决的问题。

系统采用的是线性插值算法：将 2048 个点插值到将要显示区域中，这样就可以满足显示波形与信号数据一致性的要求。

实际工作中，用户经常希望同时看到模拟信号的波形以及模拟信号经过调制之后的波形，那么为了满足用户的需求，系统中将模拟信号显示的区域分为上下两部分：上面一部分用来显示模拟信号的波形，下面一部分用来显示模拟信号经过调制之后的波形。同时在右侧显示波形的设置项，左侧的界面可以随着右侧的设置项的变化而变化。

图 3-5 是模拟信号预览窗体的设计图。

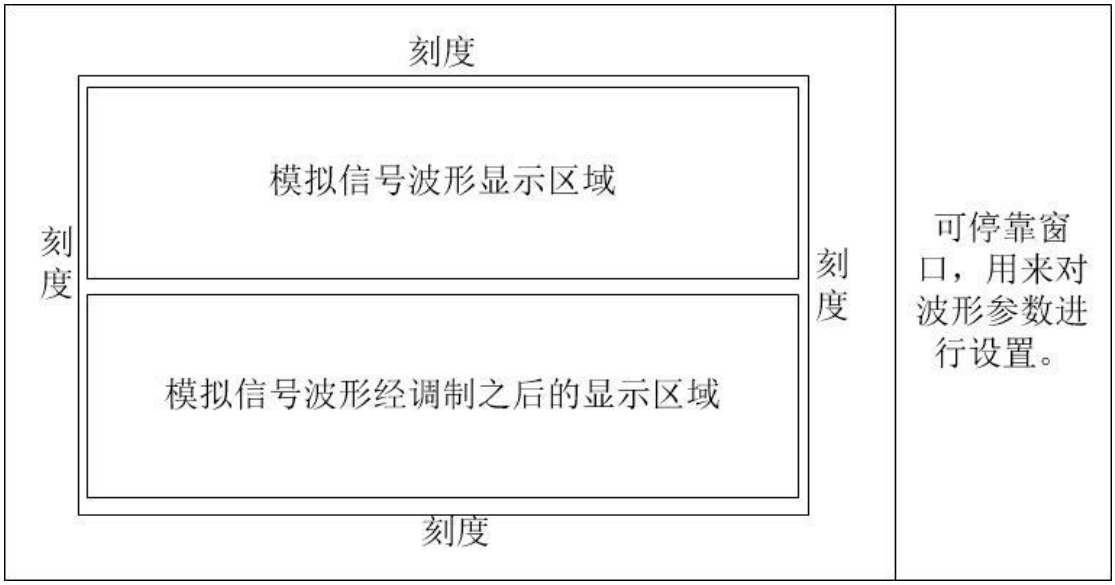

图 3-5 模拟信号显示窗体设计图

另外，为了实现易用性，满足不同用户的需求，用户可以通过双击左键或选择上下文菜单两种不同的方式来显示设置对话框。

2）数字信号预览

数字信号是经过处理后的模拟信号，它只有两种表现形式：0（低电平）、1（高电平）。仅仅绘制数字信号的走势是毫无意义的，因为用户并不能通过数字信号的走势来确认这段数据表达的含义，因此系统中使用菱形框来描述在某个协议下一段数据流表达的含义。

与模拟信号不同，数字信号数据流的长度不固定，有些协议下数据流的长度会很长，因此就需要一个滚动条机制来帮助用户确认显示哪一段数据流。

图 3-6 是数字信号显示窗体的设计图。

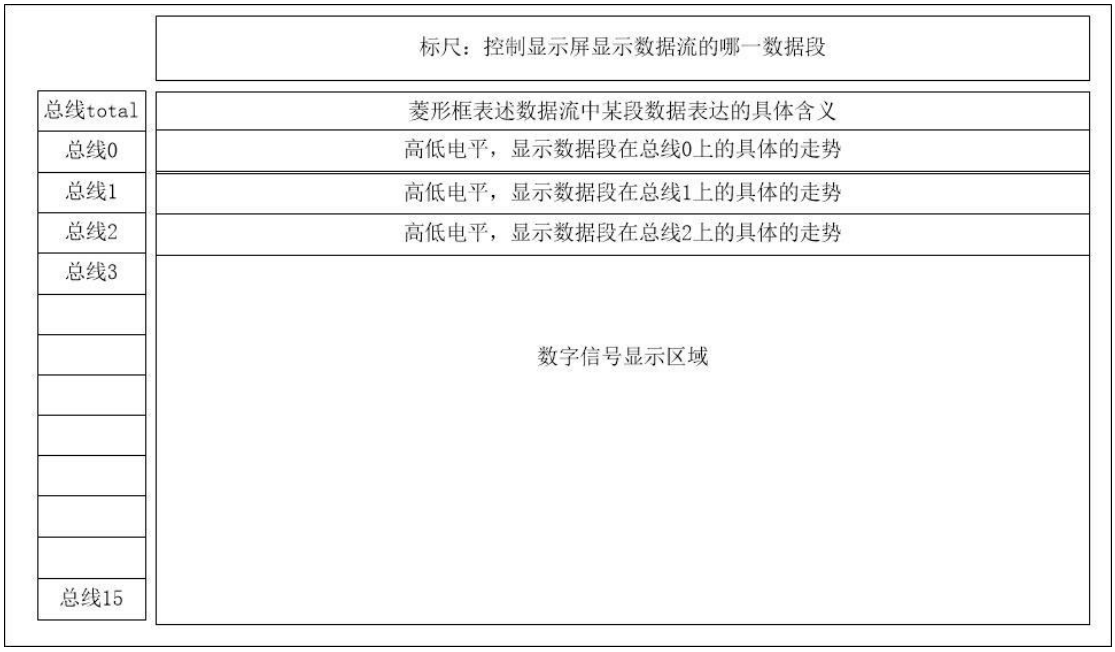

图 3-6 数字信号显示窗体设计图

相比于模拟信号侧重于表现一个周期内波形的起伏，数字信号则侧重于表现在特定的协议下数据流需要遵循的规范。相同的一串数据流经过不同的数字协议解析后将会表达不同的含义，而数字信号预览模块不可能提前知道每个协议的规范，因此就需要出现一种与协议无关的数据表现方式。

因为用户最关心的是某一段数据流表现的含义，那么可以约定从协议处理模块读取数据时，由协议处理模块额外提供一些信息。以下是从协议处理模块获取的信息：

① 数据块 ID：表明这个有含义的数据块在数据流中的位置

② 数据块长度（时间长度）

③ 数据块表示的含义（数据、空闲、开始、结束等）

④ 用户设置的数据块显示的颜色：仅仅用来显示，帮助用户区分不同数据块

根据实际考察，用户经常需要进行以下操作：跳到数据流开始处、跳到数据流结束处、缩小显示、放大显示。为了满足不同知识背景用户的操作习惯，系统将实现两种操作方式：第一种，通过点击鼠标右键调出上下文菜单，选择用户想要的操作；第二种，通过鼠标与键盘的组合，使用户快速的进行操作。

3.3.5 绘图板模块设计

绘图板模块相对于其他模块较为独立。它自己形成一个可执行文件，可以在主程序中作为一个工具打开，也可以单独打开。

绘图板模块的目的是为了生成用户需要的波形。它的输入是用户通过鼠标的“胡乱”滑动，因此可以通过获取用户鼠标按下左键并滑动时的轨迹，来取得输入；然后将这些屏幕上的点通过线性插值算法映射成 2048 个整数，就得到符合规定的点集合。

3.3.6 内部接口设计

为了实现程序的扩展性，系统中决定将波形或协议设置模块、波形或协议数据生成模块的实现放到动态链接库中，这样其他开发人员只需要按照一定的接口规定，就可以为程序添加波形或协议处理插件。

接口主要定义了主窗体与数据生成模块之间数据的传递：

1) 获取基本信息：指明 DLL 是用来处理数字信号的还是模拟信号的
2) 弹出设置窗体：用户可以通过设置窗体来对波形或协议进行参数设置
3) 获取设置参数：得到用户设置的显示参数
4) 获取生成的数据：得到通过波形或协议参数产生的数据
5) 保存数据：将产生的数据保存到文件

4 系统实现

4.1 内部接口的定义

由于数字信号与模拟信号差异较大，因此用于数据信号和用于模拟信号的接口函数不完全一样。图 4-1 为软件内部接口定义

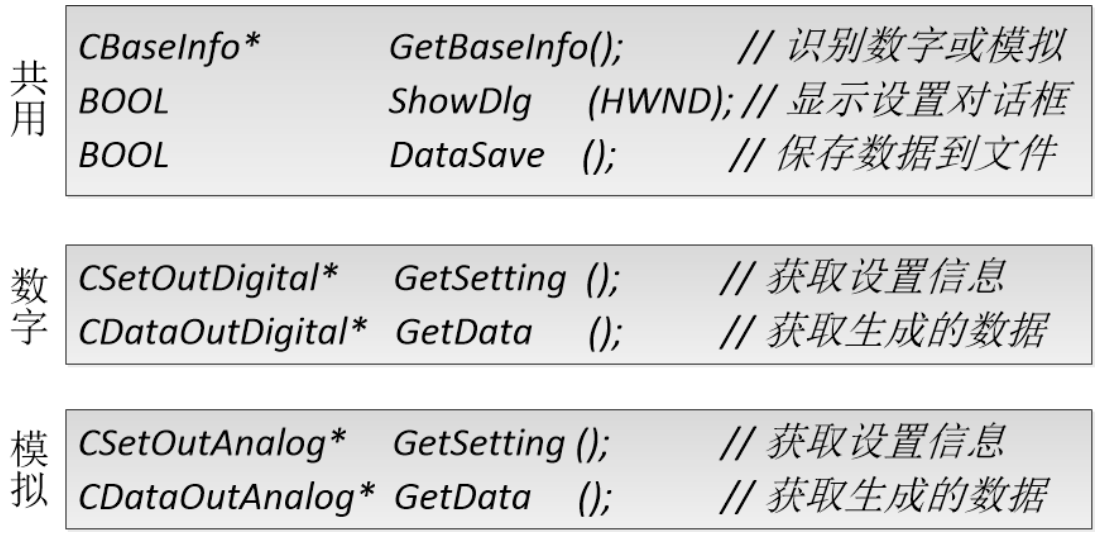

图 4-1 内部接口定义

其中 CBaseInfo、CSetOutDigital、CDataOutDigital、CSetOutAnalog、CdataOutAnalog 五个类都是数据类，专门用来存储数据。

4.2 选择协议或波形模块的实现

选择协议或波形模块由类 ListPluginsDialog 进行管理。类的定义如代码段 4-1。

代码段 4-1 类 ListPluginsDialog 的定义

```
class ListPluginsDialog : public QDialog, private Ui::ListPluginsDialog
{
public:
    void findPlugins();          // 获取plugins文件夹下符合规定的动态链接库文件
private:
    QStandardItemModel *model;  // 用来插入数据到对话框以及从对话框获取数据
signals:
    // 该对话框为模态对话框，当选中某个项目并确认时，发射这个信号
    void selectedFile( QString,QString,QString );
```

```
private slots:
    void onOkClicked();          // 当确认按钮被按下时，调用这个函数
};
```

类在构造的时候，设置对话框的显示属性，并连接相应的 signal 和 slot；当该类被告知需要显示选择对话框的时候，它将会在 plugin 文件夹下查找所有的动态链接库文件（Dll），然后逐一获取该动态链接库文件的自省信息，最后将这些信息插入到对话框中；当用户选择了对话框中的某一项，并点击确认按钮时，该类的 onOkClicked 函数将会被调用，它负责提取出用户点击项的详细信息，并将这些信息通过 selectedFile 信号发送出去。代码段 4-2 是关键代码段。

代码段 4-2 类 ListPluginsDialog 中 findDialog 方法的实现

```
// 获取 plugins 目录下的所有 dll 文件信息
QDir dir(QString("plugin\\"));
QStringList filters;
filters << "*.dll";
dir.setNameFilters(filters);
QFileInfoList allFiles = dir.entryInfoList ();
PluginInfo * plugin;            // 使用下面的类来管理动态链接库文件
// 遍历获取到的 dll 文件，它们的自省信息显示出来
for(int i = 0;i<allFiles.size ();i++)
{   QFileInfo fileInfo = allFiles.at (i);     // 初始化 PluginInfo
    plugin = new PluginInfo(UKNOWN_TYPE);
    if(plugin->setName(fileInfo.filePath()))
    {
        if(plugin->loadBaseInfo())
        {
            CBaseInfo * bi = plugin->getBaseInfo(); //获取自省信息
            model->insertRow(0);                    // 在对话框中增加一行空白行
            // 在新增的空白行中插入自省信息
            model->setData(
                        model->index(0,0),
                        QString(gbk_codec->toUnicode(bi->strName))
             );
            model->setData(
                        model->index(0, 1),
                        (bi->isDigital == false)?tr("模"):tr("数")
             );
            model->setData(model->index(0, 2), fileInfo.filePath ());
        }
    }
    delete plugin;
}
```

该模块最终实现的界面如图 4-2

。

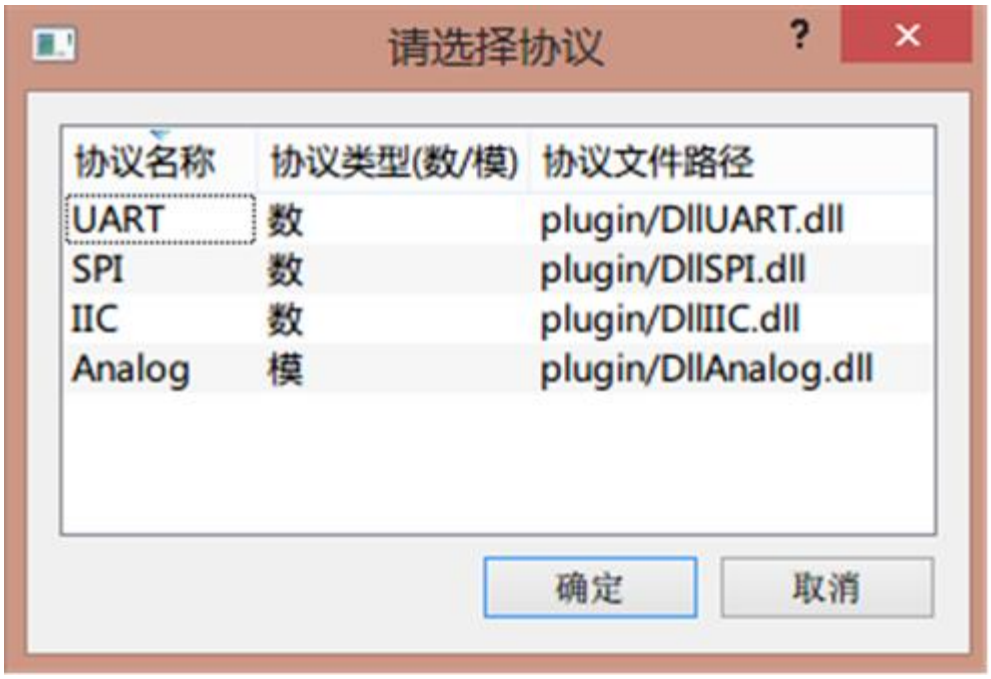

图 4-2 选择协议或波形模块的窗体实现

4.3 模拟信号预览模块的实现

模拟信号的显示使用了多个类的协同工作来实现。首先，使用类 AxisOfCoordinates 负责绘制模拟波形以及模拟波形经过调制之后的波形，代码段 4-3 是类 AxisOfCoordinates 的定义。

代码段 4-3 类 AxisOfCoordinates 的定义

```
class AxisOfCoordinates : public QWidget
{
public:
    PluginInfo * pi;
protected:
    void contextMenuEvent (QContextMenuEvent *);  // 重新实现上下文菜单事件
    void mouseDoubleClickEvent(QMouseEvent *);    // 重新实现鼠标双击事件
    void paintEvent (QPaintEvent *);              // 重新实现绘图事件
private:
    const int upDownSpace;            // 坐标轴上方距离屏幕上方的距离
    const int leftRightSpace;         // 坐标轴左边距离屏幕左边的距离
    const int circle;                 // 坐标轴距离绘制波形区域的距离
    QVector<QString> xCoor;           // 用来保存 x 坐标轴刻度值的数组
    QVector<QString> yCoor;           // 用来保存 y 坐标轴刻度值的数组
    typedef QVector<int> WaveVector;
    WaveVector waveVector;            // 用来保存波形数据的数组
```

```
    CDataOutAnalog * data;              // 从波形处理模块获取的波形数据
    CSetOutAnalog * set;                // 从波形处理模块获取的波形设置参数
    void paintMesh(QPainter * pt);      // 绘制网格：坐标轴线、背景网格、刻度值
    void drawWave( QPainter *pt,QRect,QColor );  //绘制波形
public slots:
    void readData();                    // 获取模拟信号的数据和设置
};
```

在类 AxisOfCoordinates 的构造函数中，进行了必要的变量初始化以及 signal 与 slot 的连接；初始情况下，它没有与任何波形类型相关联，因此在绘图函数 paintEvent 中只是绘制了必要的坐标轴以及背景；当它与具体的波形相关联时，它会被告知调用 readData 函数，readData 函数将会从管理动态链接库的类 PluginInfo 中获取模拟信号的数据和设置参数，进行必要的处理，然后通知绘图函数 paintEvent 绘制坐标轴以及波形。

其中，从动态链接库中获取的数据，是 2048 个 int 值，每个 int 值都介于 0~1023 之间，如何将这 2048 个数据映射到屏幕上而不使波形失真是技术难点。

系统中采用的是线性插值算法。假设屏幕上显示波形的区域宽度为 x 个像素点（x 的值一般为 100~1000），那么这 2048 个点落到屏幕上显示波形的区域后，很多点都落到了屏幕上相邻两个像素点的中间（即落点像素点不为整数）。如图图 4-3，假设 J0-J10 是 2048 个点中的 11 个点，I0-I3 是 x 个点中的 4 个点；其中 J0 经过映射之后落到了 J0 上，J1、J2、J3 经过映射之后落到了 J0 与 J1 之间，J4、J5、J6 落到了 I1、I2 之间…；现在的问题变成了如何通过已知的 Jn（n=0…10）的值求取 Im（m=0…3）的值。要求取 Im 的值，首先需要找出一个 k，使得 Jk 与 Jk+1 落到 Im 的两侧；设 Im 对应的像素点位置为 X0（整数），对应的值为 Y0，Jk 对应的像素点位置为 X1（非整数），对应的值为 Y1，Jk+1 对应的像素点位置为 X2（非整数），对应的值为 Y2,则 Im 对应的值为：

Y0 = Y2 - （Y2-Y1）*（X2-X0）/(X2-X1)。

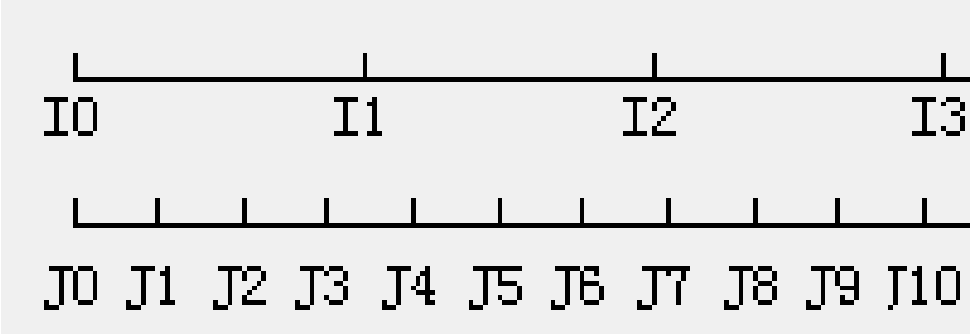

图 4-3 逻辑点与像素点的对应图

代码段 4-4 是实现该算法的代码段。

代码段 4-4 类 AxisOfCoordinates 中线性插值算法的实现

```
void AxisOfCoordinates::drawWave ( QPainter *pt,QRect rect,QColor color)
{
    // 局部变量初始化
    double x = 0.0;
    double y = 0.0;
    int oldx = -10000;
    int oldy = -10000;
    QPainterPath path;      // 绘图路径
    pt->save();             // 保存之前画笔，用来在程序结束的时候进行恢复
    QPen pen(QBrush(QColor(color)),2);
    pt->setPen(pen);
    // 使用线性插值算法将 2048 个点映射到屏幕上
    for( int i = 0 ; i < waveVector.count() - 1 ; i ++ )
    {
        // 找到 2048 个点中临近屏幕上点的两个连续点
        if( static_cast<int>(
                  static_cast<double>(i) * rect.width() / waveVector.count()
            ) < static_cast<int>(
                   static_cast<double>(i+1)* rect.width() / waveVector.count() )
            )
        {
            // 求出屏幕上该点的插值后的值
            x = static_cast<double>(i+1) * rect.width() / waveVector.count();
            x = static_cast<double>( static_cast<int>(x) );
            y = static_cast<double>(waveVector.at(i)) +
                ( x - static_cast<double>(i) * rect.width() / waveVector.count() )*
                ( waveVector.at(i+1) - waveVector.at(i) ) /
                static_cast<double>( rect.width() ) / waveVector.count();
            // 在绘图路径上绘制点
            if(!( oldx == -10000 && oldy == -10000 ))
            {
```

```
                path.moveTo(oldx,oldy);
                path.lineTo(
                        static_cast<int>(x)+rect.x(),
                        rect.y()+rect.height()
                              - static_cast<int>(y*rect.height()/1023));
            }
            oldx=static_cast<int>(x)+rect.x();
            oldy=rect.y()+rect.height()-static_cast<int>(y*rect.height()/1023);
        }
    }
    pt->drawPath(path);       // 将点绘制到屏幕上
    pt->restore();
}
```

之后，通过类 AxisOfCoordinates 和 DockWidget 组合成类 SimulatePreview，类 SimulatePreview 直接与程序主体框架类 MainWindow 进行交互，然后控制类 AxisOfCoordinates 对用户的操作进行响应；类 AxisOfCoordinates 不会与类 MainWindow 直接进行交互，实现了类的封装。

注：因为技术原因，系统中没有实现停靠窗口这一设计，而由于时间的问题，我们只能将设置对话框显示成一个模态对话框，只在修改参数的时候进行显示。

该模块最终实现的界面如图 4-4

图 4-4 模拟信号显示模块的窗体实现

注：由于设置对话框并非我实现的，因此这里不做过多描述。

4.4 数字信号预览模块的定义

数字信号的显示也使用了多个类的协同工作来实现。首先，使用类 Staff 来控制显示波形的哪一段（数字信号数据流将要被显示的起点以及宽度），类 ButtonTree 用来绘制总线，类 DigitPreviewArea 类负责根据 Staff 提供的位置信息来绘制波形。代码段 4-5 是这三个类的定义。

代码段 4-5 类 Staff、ButtonTree、DigitPreviewArea 的定义（简）

```
class Staff : public QWidget
{
private:
    quint64 startTime;              //标尺标注的开始时间，单位为 1ns
    int time;                       //标尺中每一小格表示的时间，单位为 1ns
    int maxDisNum;                  //标尺显示的小格个数
    int SmallLength;                //每小段标尺的像素距离
    int NumerSmallLength;           //每大段标尺的小段个数
};
class ButtonTree : public QWidget
{
signals:
    void buttonSelected(QRect);        // 当某个 bus 按钮被选中后时，发射这个信号
    void buttonPosChanged(QVector<QRect>);  // 当按钮位置变化时，发送这个信号
};
class DigitPreviewArea : public QWidget
{
public:
    PluginInfo *pi;
private:
    QRect currentSelectedButtonPos;          // 标识当前被选中的 bus 位置
    QVector<QRect> buttonPos;                // 标识所有 bus 的位置
    // 通过以下变量绘制波形
    quint64 startTime;
    int timeLength;
    int disLength;
    int disNum;
    CDataOutDigital * data;                  // 数字信号的数据
    CSetOutDigital * set;                    // 数字信号的设置
    //负责绘制 bus 的 0 号总线，也就是 16 根总线数据的混合体，菱形框
    void drawBusTotal( QPainter * );
    void drawRhomb( QPainter *,quint64,int,QString,QColor );  // 绘制菱形
    void drawBus( QPainter * );              //负责绘制 bus 的 1-16 号总线，折线
    void drawReferrenceLine( QPainter * );   //负责绘制折线的参考线
```

```
};
```

类 Staff 中变量：startTime、time、maxDisNum、SmallLength 来控制数字信号显示区域的一些属性，startTime 控制显示区域最左侧显示的数据的时间（单位为 ns），time 表示标尺中一个最小的刻度显示的时间长度（单位为 ns），maxDisNum 表示显示区域中最多能显示的刻度数，SmallLength 表示每个刻度在屏幕上显示的像素长度；通过这四个变量，可以控制数字信号数据流中哪一段在屏幕上显示、以及显示到屏幕上的那个位置（x 轴的位置）。

类 ButtonTree 用来绘制总线按钮，它用来界定每条总线的高度。

类 DigitPreviewArea 用来绘制数字信号将要被显示的数据段的走势。与类 AxisOfCoordinates 的实现类似，类 DigitPreviewArea 在构造函数中对一些变量进行初始化，以及建立 signal 与 slot 的连接；在未与具体协议绑定之前，绘图函数只是绘制出背景；当类 DigitPreviewArea 被告知与某一个具体协议绑定之后，类 DigitPreviewArea 将会通过 PluginInfo 从动态链接库中获取出数字信号数据流以及数据信号显示参数，然后绘图函数根据这些数据，调用 drawBusTotal（负责绘制 bus 的 0 号总线，也就是 16 根总线数据的混合体，菱形框）、drawBus(负责绘制 bus 的 1-16 号总线，折线)、drawReferrenceLine(负责绘制折线的参考线，每两根参考线之间的距离代表一个二进制位的时间)来绘制用户想要观察的数据流走势图。

由于直接控制这三个类进行数字信号的显示较为复杂，而且为了在提供服务时屏蔽掉细节问题，系统中使用类 DigitPreview 来对这三个类进行管理：DigitPreview 直接与 MainWindow 进行交互，然后通过控制 Staff、ButtonTree、DigitPreviewArea 来实现用户的操作请求，而 Staff、ButtonTree、DigitPreviewArea 对 MainWindow 来说是透明的。

该模块最终实现的界面如图 4-5。

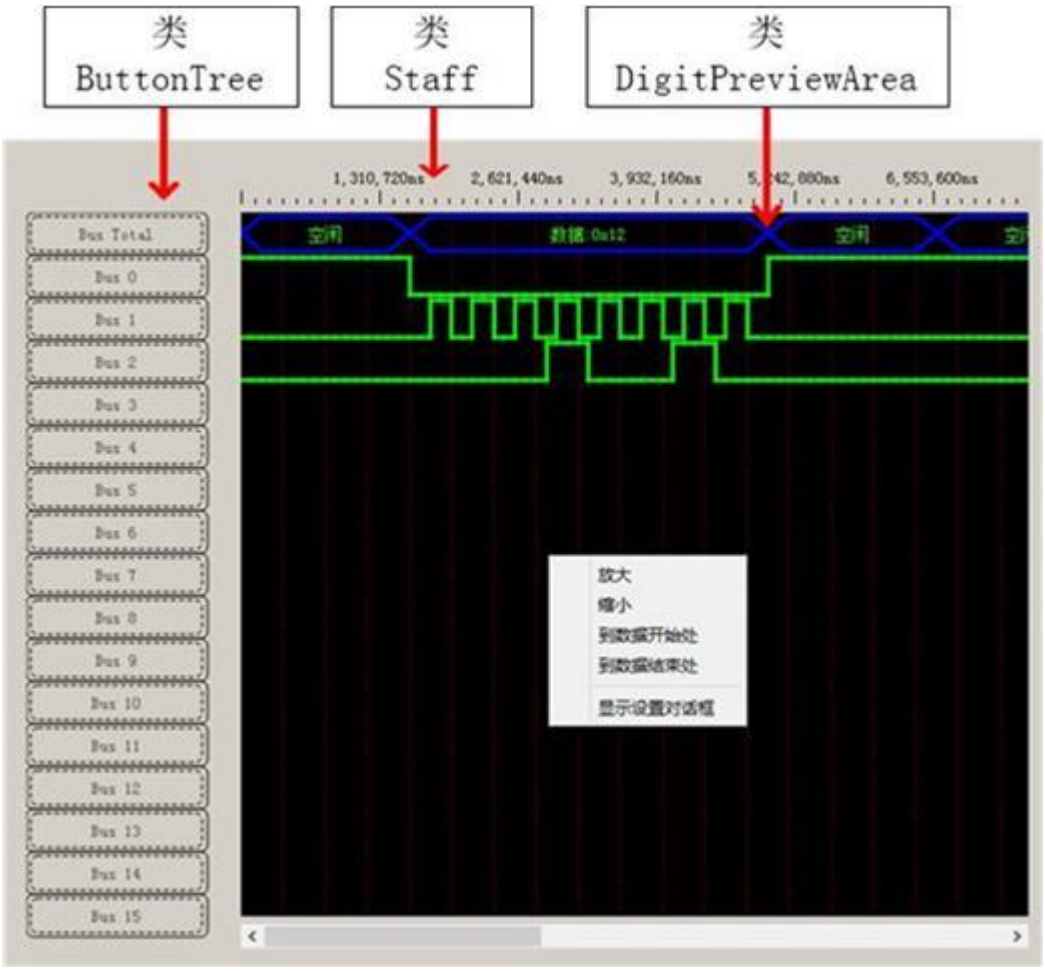

图 4-5 数字信号预览窗体的实现

注：由于设置对话框并非我实现的，因此这里不做过多描述。

4.5 绘图板模块的定义

绘图板模块较为独立，它的最终表现形式是一个单独的可执行文件，也就是说它除了在主程序中打开外，还可以单独打开。

绘图板模块的主要功能由类 Canvas 实现。代码段 4-6 是 Canvas 的定义。

代码段 4-6 类 Canvas 的定义

```
class Canvas : public QWidget
{
public:
    Canvas(QWidget *parent = 0);
    QVector<int > data;              // 文件中的数据
    QVector<QPoint> path;            // 绘图板上的点，由 data 插值获取
    // 文件中数据与屏幕上的点相互转换
    void pathToData();
```

```
    void dataToPath();
    // 插值算法
    void LinearInterpolation( QRect,QVector<QPoint>,QRect,QVector<QPoint> &  );
protected:
    void paintEvent (QPaintEvent *);
    void mousePressEvent (QMouseEvent *);
    void mouseReleaseEvent (QMouseEvent *);
    void mouseMoveEvent (QMouseEvent *);
private:
    const int leftright;      // 绘图板绘图区的左右边界
    const int updown;         // 绘图板绘图区的上下边界
    bool isPushDown;          // 识别当前鼠标左键是否被按下
};
```

Canvas 在构造函数中，初始化一些变量以及 signal 与 slot 的连接；当用户控制鼠标左键按下时，isPushDown 被置为 true；当用户鼠标移动时，若是 isPushDown 为 true，则记录鼠标移动的轨迹，并将轨迹中在界定区域内的点记录到 path 变量中；当用户鼠标左键抬起时，Canvas 将会把 path 中的数据通过插值算法映射到 data 中。

为了将数据处理与界面显示想分离，系统使用了另外一个类 DrawBoard 来控制 Canvas。代码段 4-7 是 DrawBoard 的定义。

代码段 4-7 类 DrawBoard 的定义

```
class DrawBoard : public QWidget, private Ui::DrawBoard
{
public slots:
    void readWave();                // 响应读文件的操作
    void saveWave();                // 响应保存数据到文件的操作
    void clearWaveData();           // 响应清除当前数据的操作
};
```

类 DrawBoard 用来与用户交互，控制 Canvas 完成用户的操作请求。

该模块最终实现的界面如图 4-6。

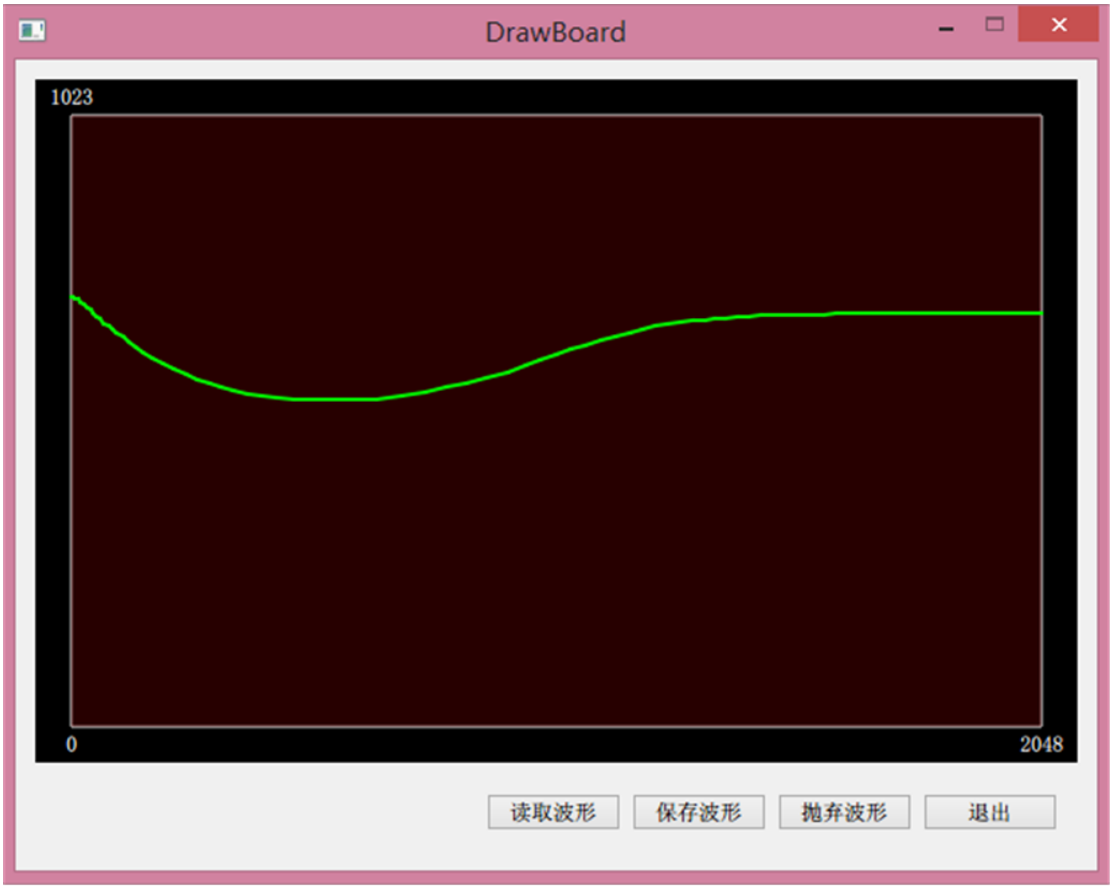

图 4-6 绘图板模块的窗体实现

4.6 程序主体框架的定义

程序的主体框架，由类 MainWindow 控制，负责控制以上的模块来帮助用户解决实际问题。主程序使用属性页的方式加载数字信号处理模块以及模拟信号处理模块；当主程序框架接收到来自用户的操作请求后，将会根据当前页面的信息来决定将这些操作请求传递至数字信号处理模块还是模拟信号处理模块；信号处理模块获取到这些操作请求之后，将会对这些请求进行响应，以实现用户的需求。

代码段 4-8 是 MainWindow 的类定义。

代码段 4-8 类 MainWindow 的定义

```
class MainWindow : public QMainWindow, private Ui::MainWindow
{
protected:
    void closeEvent (QCloseEvent *);
private:
```

```
    ExtensionTool * et;                          // 控制绘图板工具的显示
signals:
    void saveData();        // 保存按钮被按下时，发射这个信号
    void initWidget(QString);      // 通知被选中的 tab 页进行初始化工作
private slots:
    // 将动态链接库文件跟 tab 页绑定
    void protocalLinktoTab(QString,QString,QString);
    void addDigitWindow();      // 增加一个新的处理数字信号的 tab 页
    void addSimulateWindow();   // 增加一个新的处理模拟信号的 tab 页
    void selectProtocal();      // 当点击选择协议对话框时，调用这个函数
    void removeOneTab(int);     // 移除一个 tab 页，并释放绑定的动态链接库
    void onActionSaveData();    // 发射给当前 tab 一个保存数据到文件的信号
    void openDrawBoard();       // 打开绘图板工具
};
```

该模块最终实现的界面如图 4-7。

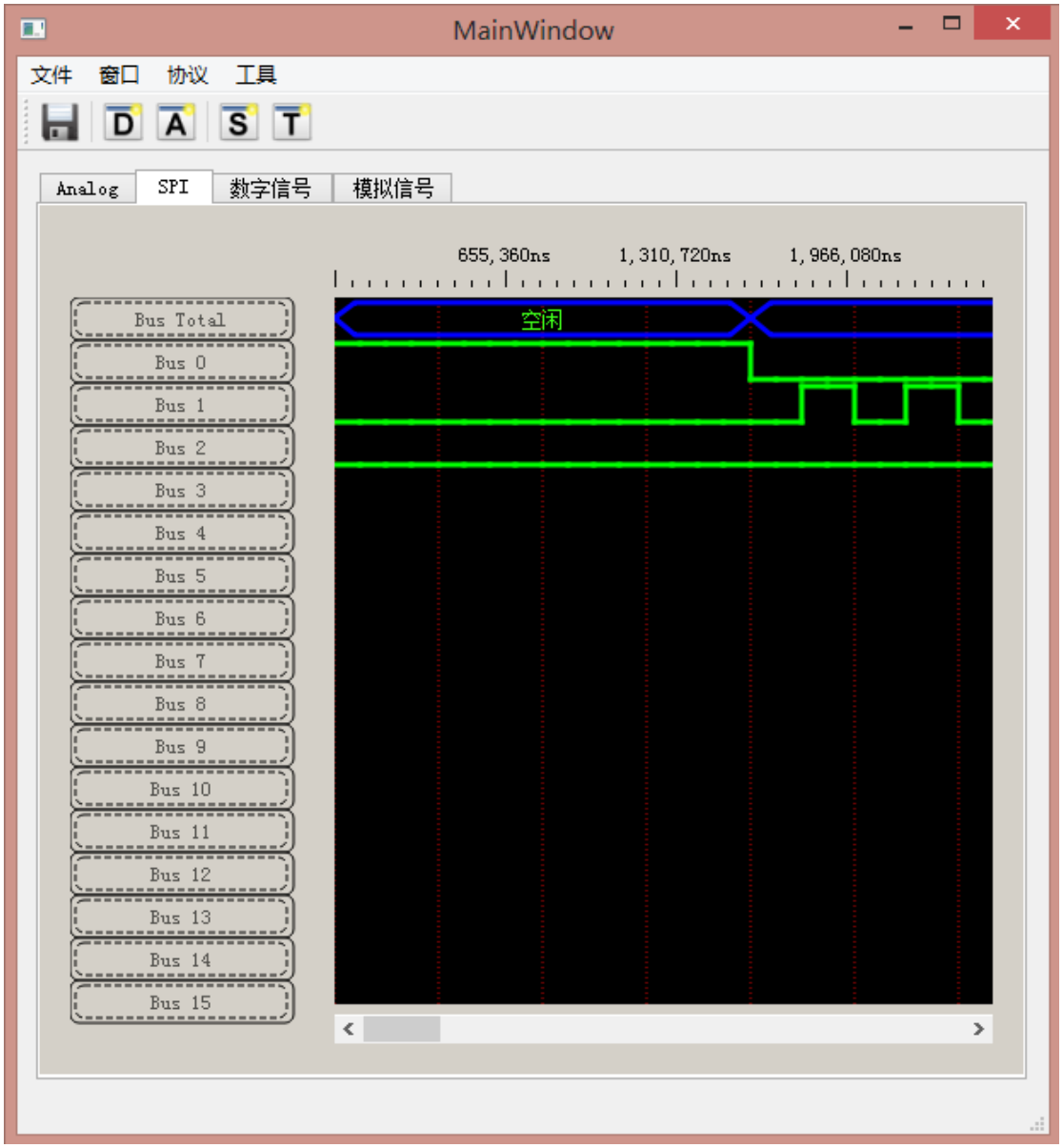

图 4-7 主体框架的窗体实现

其中菜单包括：保存数据到文件、打开数字信号处理窗口、打开模拟信号处理窗口、打开选择协议或波形对话框、打开绘图板工具。MainWindow 中的 slot 就是为了响应这些菜单操作：onActionSaveData 用来响应保存数据到文件操作，addDigitWindow 用来响应打开数字信号处理窗口操作、addSimulateWindow 用来响应打开模拟信号处理窗口操作、selectProtocal 用来响应打开选择协议或波形对话框操作、openDrawBoard 用来响应打开绘图板工具操作。

5 总结

系统已经实现了以下功能：选择协议或波形类型、配置协议或波形参数、生成协议或波形数据、预览、保存数据到文件、波形绘图板。用户可以选择自己想要的波形或时序，然后配置参数；待配置完毕，系统将生成相应的数据，并将它绘制到屏幕上；用户可以通过这些图像来确认是否能够满足他们的需要；若不能满足，则可以返回重新配置，若能满足，则可以通知系统按照规定的格式将数据保存到文件中。

但是，无论是在系统开发过程中还是在最终的系统实现上，都出现了一些问题：

1）因为管理经验欠缺而导致的时间进度安排不合理，过低的估计了学习新知识的时间代价，导致系统逾期完成，且一些问题无法解决。

2）过低的估计了 UI 界面设计的难度，只是站在开发人员的角度设计操作习惯而没有考虑其他不同类型用户的使用习惯。

3）波形设置对话框无法停靠到主窗体上，使得用户随时查看当前波形设置；而只能作为一个模态对话框，当需要设置的时候显示，需要绘制波形的时候隐藏。

4）即使使用了线性插值算法算法，但在实际操作过程中，绘图板模块保存到文件的数据被重新读取后，会出现锯齿。

本次系统的开发人员经验较为欠缺，是一次初步尝试，在以后将会有经验丰富的开发人员指导并重启系统，以真正的达到商用软件的标准。

致谢

时光飞逝，转眼间一个学期的毕业设计已经完成，心中有许多感慨，许多感动。在公司里，我的导师韦先平韦工在设计思路以及技术上给予了我许多支持，使得我明白了什么是面向对象设计、什么是 MVC、什么叫做 UI 为用户服务；而当我回到学校，郑志蕴老师为我的论文提供了许多宝贵的意见，使得我可以在完成毕设之后顺利的完成论文，进行毕业答辩。

在公司的时间里，我对论文毫不重视，认为毕设最重要的是实现，当实现完成之后随便花几天时间写论文就行了；但是郑老师与我谈话后我才知道论文的重要性：一篇论文要能够详细的描述一个系统的需求、设计、实现等各个方面，论文中每句话都要斟词酌句，以最简练的语言，清晰的表述出作者的思想；论文的编写，不仅仅是对系统的描述，还是开发人员对系统整个开发过程的总结，通过它，开发人员可以发现在系统开发中出现的一些失误，并进行反思。 然而，在与郑老师的交流中，我感觉最为受益的是一种工作的态度：认真。无论论文多么的差，郑老师都很有耐心的为我们修改，这是值得我们在今后的学习和工作中学习的。

最后，谢谢在这次毕设中帮助过我的每一个人，正是他们，才有了这个系统的诞生，谢谢你们。

参考文献

1 EEPW 论坛. 国产示波器向中高端市场迈进.http://forum.eepw.com.cn/thread/159316/1

2 致远电子官网. MI1062 六合一多功能组合仪器产品概述.http://www.zlg.cn/MI/

3 百度百科.qt. http://baike.baidu.com/view/23681.htm?fr=aladdin

4 Jasmin Blanchette,Mark Summerfield 著.C++ GUI Qt 3.第二版.电子工业出版社.2008-8

5 蔡志明著.精通 Qt4 编程.电子工业出版社.2008-1-1

6 百度百科.MVC 框架.http://baike.baidu.com/view/5432454.htm?from_id=85990&type=search&fromtitle=mvc&fr=aladdin

7 百度百科.线性插值法. http://baike.baidu.com/link?url=0ZCQ274pJ9mSLwPjqnefpOnFvXz6aC0EjYRoIxLGI3dAWpLQMBKTXctqWOjn0aHt43yNdta7SNsVF8qkAnAPpq

8 百度百科.iic.http://baike.baidu.com/view/194759.htm?fr=aladdin

9 百度百科.SPI.http://baike.baidu.com/view/245026.htm

10 百度百科.UART.http://baike.baidu.com/view/245027.htm?fr=aladdin

毕业设计（论文）成绩评价意见

论文题目	基于 MI1000 的波形和时序生成系统的设计与实现

指导教师评语：

评定成绩： 签名： 年 月 日

评阅人评语：

评定成绩： 签名： 年 月 日

答辩小组评语：

答辩小组成员签名：

答辩成绩： **组长签名：** 年 月 日

答辩委员会意见（同意给优、良、中、及格等次）

总成绩(综合)： 签名： 年 月 日