上海交通大學

项目报告

学院(系): 生物医学工程

专　　业: 生物医学工程

学生姓名: 陈子龙　学号: 516021910459

学生姓名: 李润桓　学号: 516021910192

2019 年　12 月　29 日

目录

最终项目

1. 界面使用说明

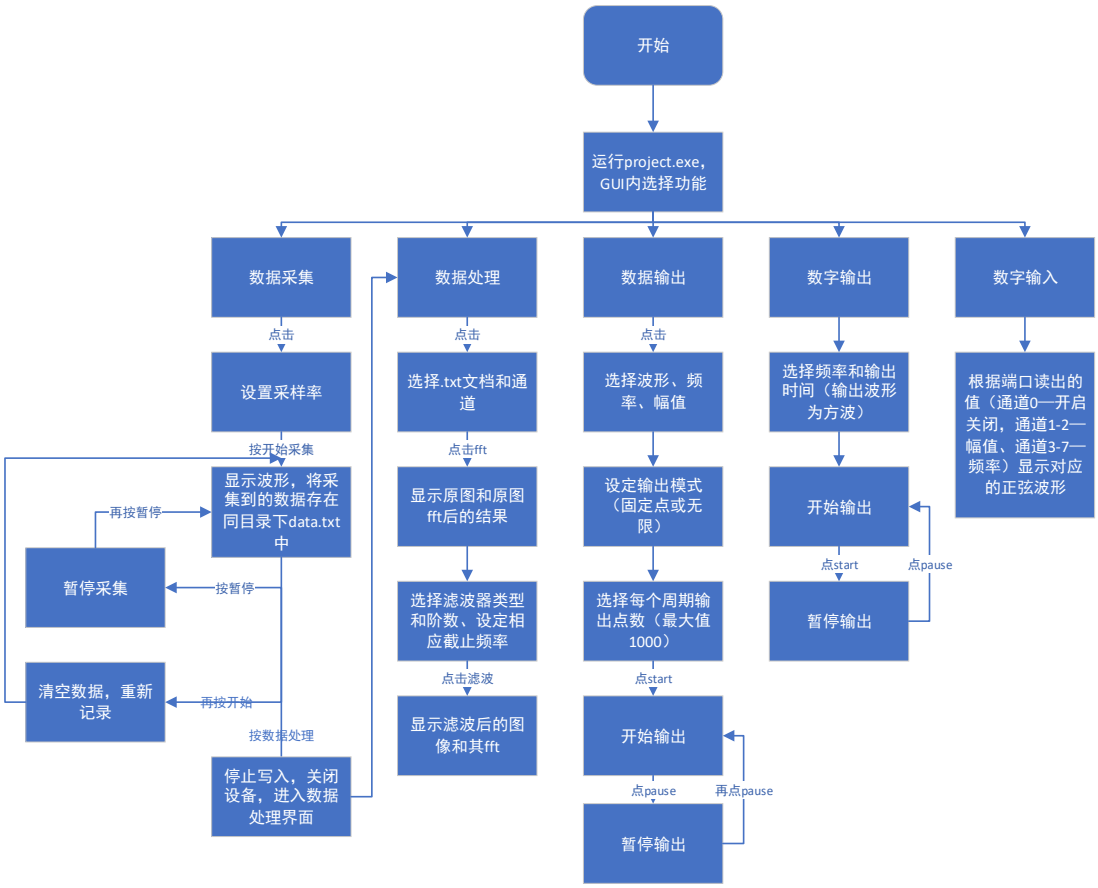

图 1. 界面使用说明图

本项目实现了一个基于 USB-4704 的简单的数据采集，输出、处理的程序。可以实现数据采集（AI）,数据处理、数据输出（AO）、数字输出、数字输入五个功能、每个功能界面的使用方法如上述界面说明图所示。详细的各功能的界面交互和其开发逻辑可参见之后的程序开发逻辑界面。

运行本项目的.exe 文件、进入功能选择界面。

图 2. 功能选择界面

点击相应的功能，打开对应的功能 GUI 界面，按照上述界面使用图的提示完成具体操作，如果仍存在问题，可参考程序开发逻辑中具体的图例。

2. 程序开发逻辑

1.1 数据采集部分

1.1.1 画图

在最开始的设计中，我们的逻辑是：当用户使用程序开始采集数据时，程序根据用户设置的采样率设置中断函数的周期。在中断函数中，程序会通过data.Get 函数获得端口的数据。程序记录当前端口接收到的数据，并和上一个数据连成直线，由于程序将axes 设置为在一次数据采集的过程中每次新增直线时不清空之前的图像，因此用户看到的是一条连续的折线。

然而使用这个方法画图的话，由于 axes 里的点会越来越多，因此每个周期的运行时间都会变长，难以实现高的采样率，因此我们使用 animatedline 和 addpoint 命令进行画图，使得每个周期的运行时间缩短至 10ms 以下，理论采样率可以达到 100Hz，并且不会出现 timer 函数运行时间随着程序运行时间增长而增加的问题。

1.1.2 图像显示范围与缩放

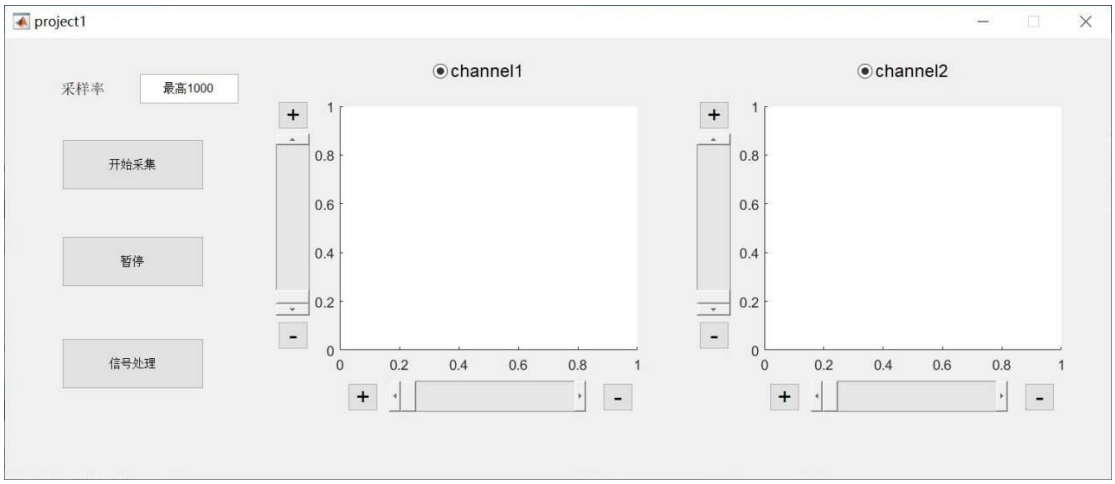

图 3.AI（数据输入）的 GUI

Axes 在 X（时间轴）、Y（电压轴）两个坐标轴旁边各有一个 slider 和两个控制缩放的按钮，控制图像显示区域的变量主要有六个：X（当前点的横坐标/时间）、Y（当前点的纵坐标/电压）、XP（当前显示区域中心点的横坐标）、YP（当前显示区域中心点的纵坐标）、regx（X 轴显示范围）、regy（Y 轴显示范围）。X 轴显示的区域是[XP−0.5*regx, XP+0.5*regx]，其中 XP 由 slider 控制，当 slider 在最左边时，XP=0，当 slider 在最右边时，XP=X，当 XP<0.5*regx 时，XP=0.5*regx。Slider 两边的“+”、“−”按钮控制 regx 的大小。Y 轴显示的区域是[YP−0.5*regy, YP+0.5*regy]，与 X 轴的控制方式同理，YP 的范围是[−0.5*regy, 0.5*regy]，Slider 两边的“+”、“−”按钮控制 regy 的大小。

1.1.3 开始、停止与继续采集

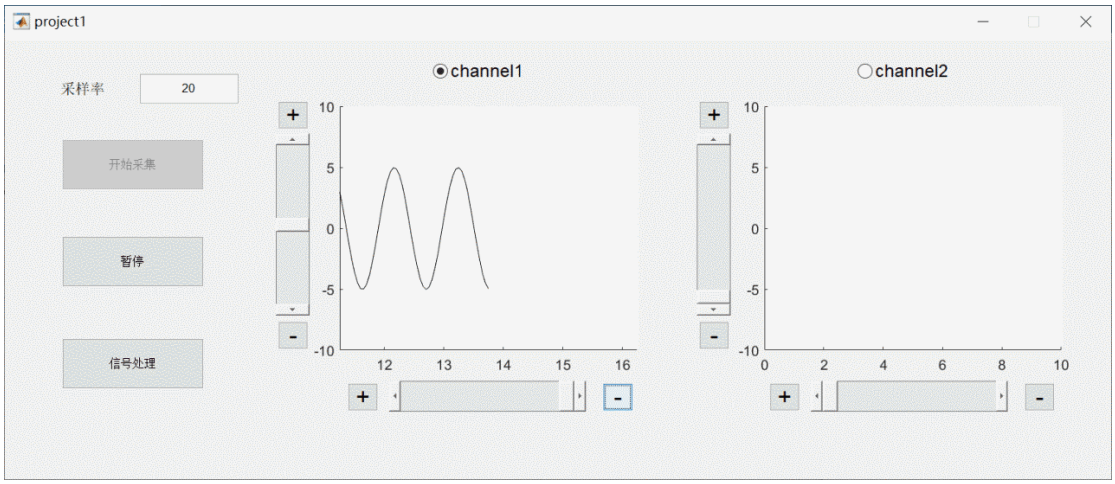

图 4.采集到的正弦信号

当开始按键被摁下，程序会清空当前的图像，并开始画图。当暂停按键被按下，程序会将 flag 变量取反，如果 flag 为 1，则开始计时，中断函数正常运行，如果 flag 为 0，则停止计时，中断函数停止运行，因此当用户第一次摁下暂停时，程序停止，再次摁下暂停，程序会从停止的地方继续运行。

1.1.4 通道选择

通道的选通状态记录在 channels 数组中，当 channels 对应的为 0 时，通道未被选择，中断函数中不运行对应通道的画图程序。

1.1.5 使用鼠标取点

通过开启 datacursormode，使得程序允许用户随时使用鼠标获取图像上点的值。

1.1.6 数据存储

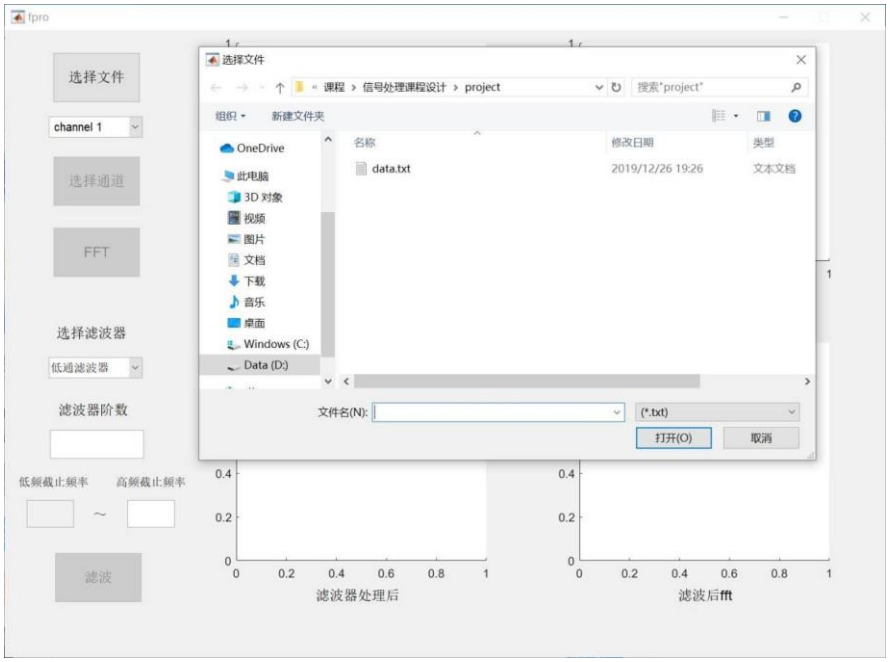

图 5.选择要处理的数据

数据的写入主要依靠 Matlab 的 fopen，fprintf， fclose 等函数。通过 fopen 打开同目录下的 data. txt 文件通过 fprintf 写入，每次读入一个数据存储一个数据。数据的格式为：采样率放在第一行，之后每一列数据（除第一行外）代表一个通道的记录。全部完成后 fclose 关闭文件。

1.2 数据处理部分

本程序实现的是所采集数据被选择的单通道的处理。数据处理可细分为以下三个功能的实现：数据的读入（包括文件及通道的选择）、FFT、滤波。我们编写了相应的函数来实现这些功能。

数据的读入：在之前记录中，我们已经将数据按我们所需的格式存好（采样率放在开头，之后每一列数据代表一个通道），给入地址，通过 fopen，fscanf，fclose 等文件相应操作，读出数据，通过所给通道选择相应数据，将所需数据读出，并显示。

FFT 的实现：为了更好使用 FFT，我们通过补零，将数据长度扩展至最近的 2 的阶次，通过 Matlab 自带的 fft 函数对其进行 FFT 计算，得到其频谱图。频谱图的 X 轴通过所记录的采样率计算得出。整个函数包括四个输入的变量：Signal（数据），Fs（采样率），handles（GUI 所需），type(用于区分数据来源于滤波前还是滤波后，与显示相关)。

滤波函数:我们使用了 Matlab 自带的巴特沃斯滤波器函数 Butter()，编写了相应的滤波函数，AI_My_Filter。整个函数所需输入的变量为:Signal(数据)，Fs(采样率)，n(滤波

器阶数),type(滤波器类型)，FH(高频截止频率)，FL（低频截至频率）。

整个数据处理界面的使用流程如示意图所示，在选择好数据位置及通道后，将显示相应通道数据，按 FFT 键进行原图的 FFT，得到其频谱图。滤波器部分，选择其类型，输入阶数，相应的截至频率，按下滤波按键，调用滤波函数和 FFT，得到原图滤波后图像及其 FFT 后的频谱图，完成整个数据处理及显示过程。

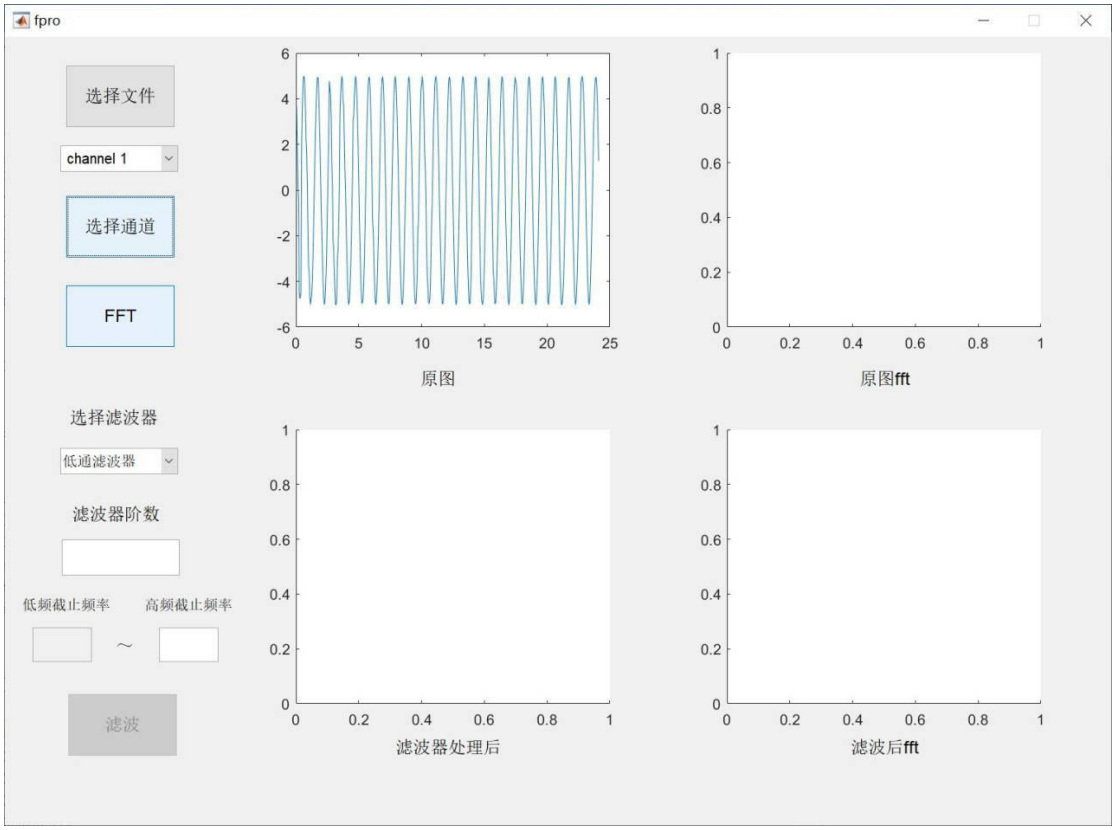

图 6. 数据处理的 GUI

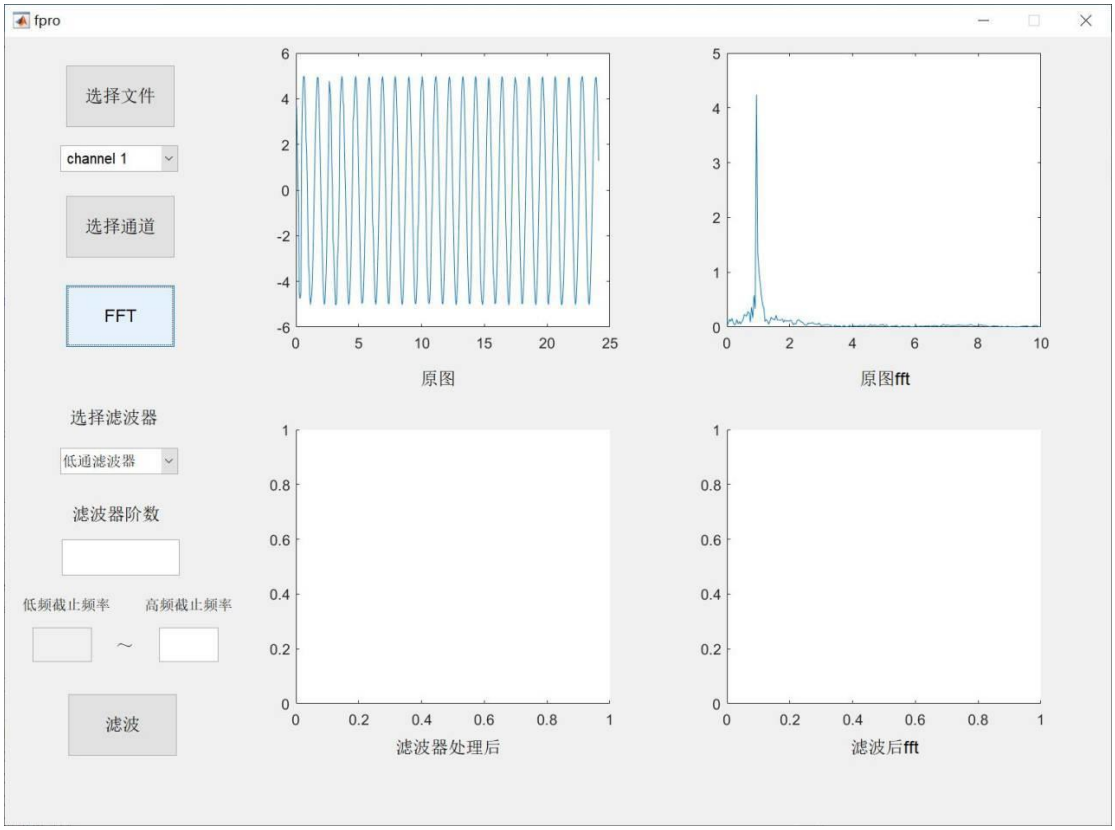

图 7. 对正弦函数进行 FFT

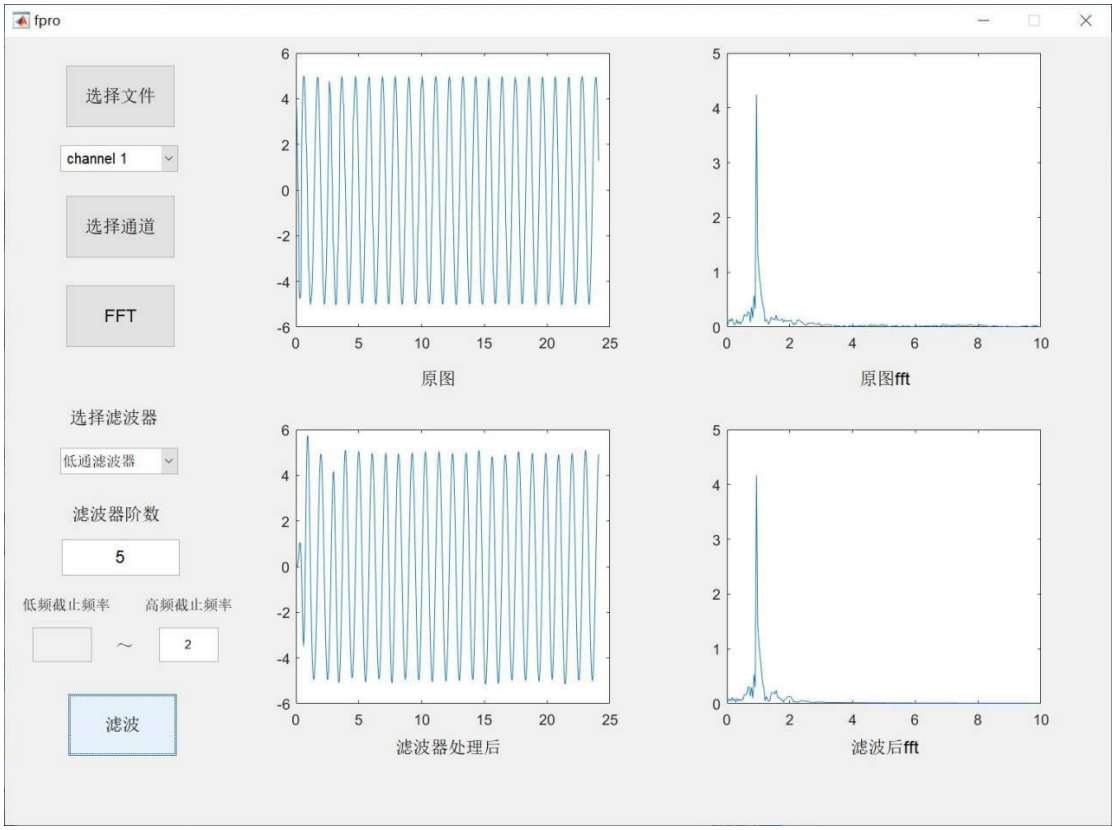

图 8. 对正弦函数进行滤波

1.3 数据输出

1.3.1 生成数据

根据用户选择的波形、每个周期的点数，程序会生成一个周期对应的数据。在后面的程序里会重复输出这些数据。

1.3.2 输出数据

每次触发timer函数时，程序会将之前生成的波形数据乘以幅值后依次输出。Timer函数的触发周期是由用户设置的信号频率和周期点数共同决定的，即：Timer的周期=1/（信号频率*每个周期的点数）。如果用户选择输出固定点数的信号，则程序里还会对输出的信号数进行统计，达到用户设置值后就会停止输出，这样就实现了输出符合用户设置的数据。

1.3.3 画图

画图同样是采用 animatedline 和 addpoint 命令结合的方式，这样可以高效地完成动态显示。每次输出数据后会将输出的值增加到图像上，这样通过画图就可以监测端口的输出值。

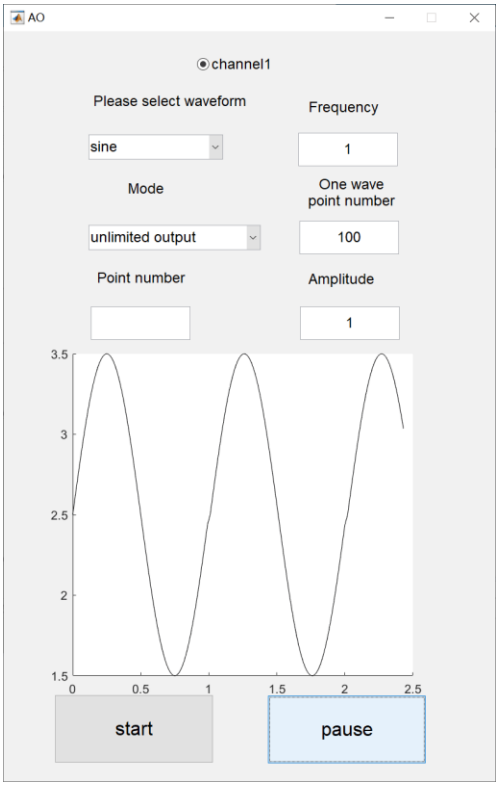

图 9.数据输出（AO）GUI（输出正弦波）

1.3.4 开始、停止与继续输出

每次按下开始按钮时程序会判断是否为首次运行程序，如果不是首次运行，则会停止并删除 Timer，以免 Timer 函数输出数据造成错误。按下停止按钮后程序会判断当前 Timer 是否已经停止，如果没有停止，则停止 Timer，如果已经停止，则重新启动 Timer。

1.4 数字输出

1.4.1 特定频率与时间的方波输出

特定频率（<50Hz）与时间的方波的输出主要通过时钟中断 timer 完成，整个功能需要用户输入 frequency(频率)和 time（持续输出时间，可不填，不填默认为 inf）。其中将 timer 函数的 period 设为 1/frequency，在每次时钟中断改变输出端口的值 Dout，来完成特定频率的方波输出。时间通过 timer 的 TasksToExecute 得到（通过 timer 的运行次数模拟执行时间）。端口状态的改变通过 bufferForWriting.Set（port、value）、instantDoCtrl.Write,instantDoCtrl.Read 来对端口输出进行修改，再通过 bufferForReading.Get 指令对端口的输出值进行读入检验输出是否正确。

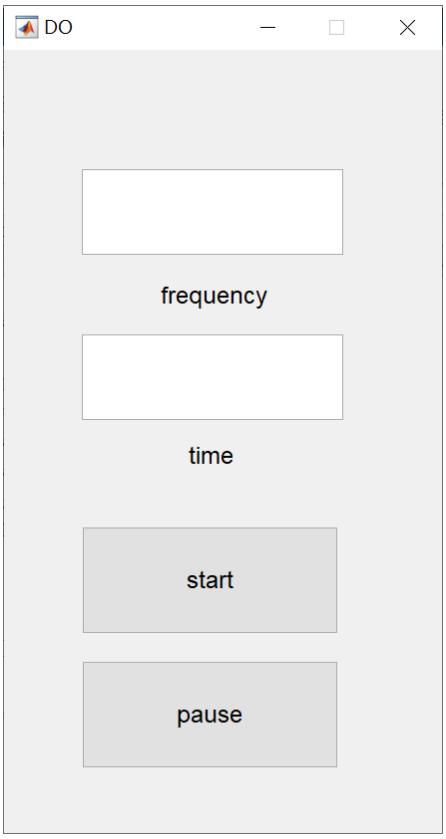

图 10.DO 的 GUI

1.4.2 开始、停止、继续输出

通过 button 控制时钟来达到对输出的控制。按开始时会重新运行整个 DO 程序，通过 Isempty 函数来判断有无时钟，若程序原先没有时钟，则按用户输入的 frequency 和 time 生成对应的时钟，按 1.1.1 所述内容进行输出，若程序原先存在时钟，则删除原时钟，生成新的时钟中断，借此实现输出的实时更新。

暂停功能通过 pause 按钮实现。若时钟正在运行，则通过 stop 语句进行时钟的关闭，反之，若时钟被暂时关闭，则通过 start 语句进行开启。时钟运行状态通过 flag 变量进行记录。

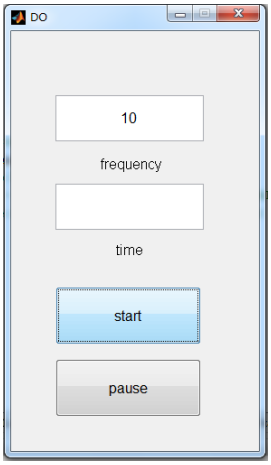

图 11.程序输出 10Hz 方波信号

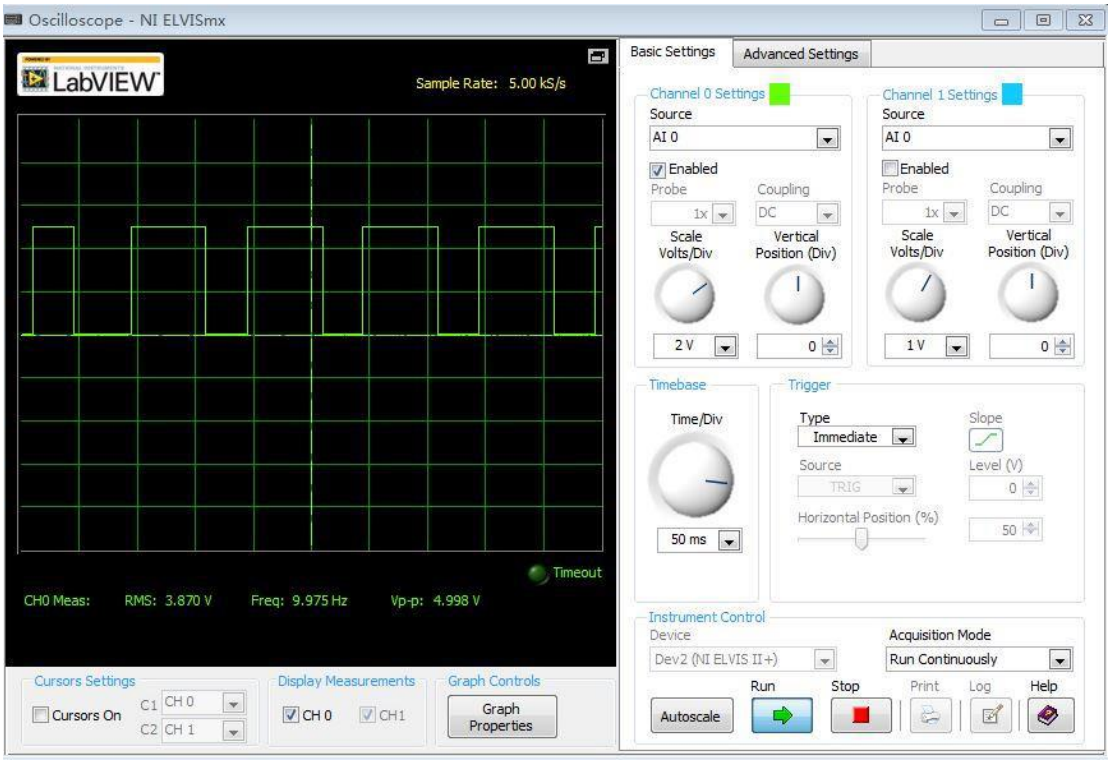

图 12.示波器观察到的信号

1.5 数字输入

数字输入我们用了 buffer.get 来读取端口值，我们将 port0 的 8 位数据中，第 0 位定为开始/截止位，1-2 位为通道表示幅度（0~3V），3-7 位代表频率（0~31Hz）。我们利用取余和取模运算将第 0 位、第 1-2 位和 3-7 位分别取出，作为函数中的参数，频率用于控制 Timer 的周期，复制用于控制输出信号的值，第 0 位的值作为 flag 控制程序是否运行，这样就能能够输出用户规定的特定频率和幅值的方波信号了。因为信号是通过 5 路管脚控制的，因此频率范围是 0~31Hz。

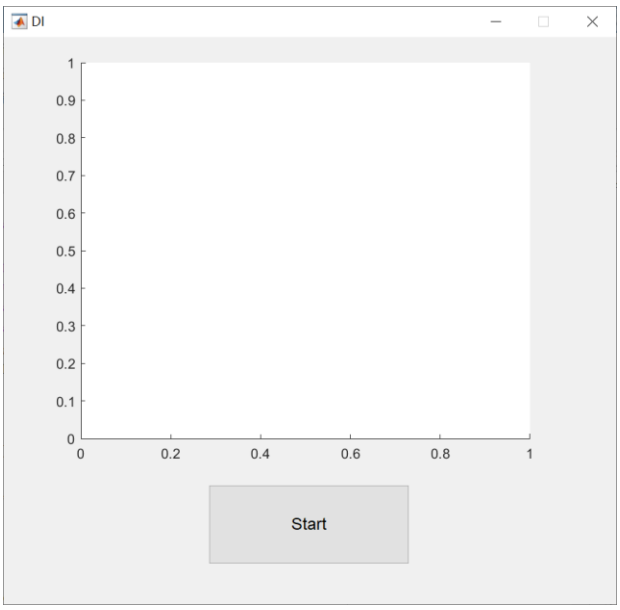

图 13.DI 的 GUI 设计

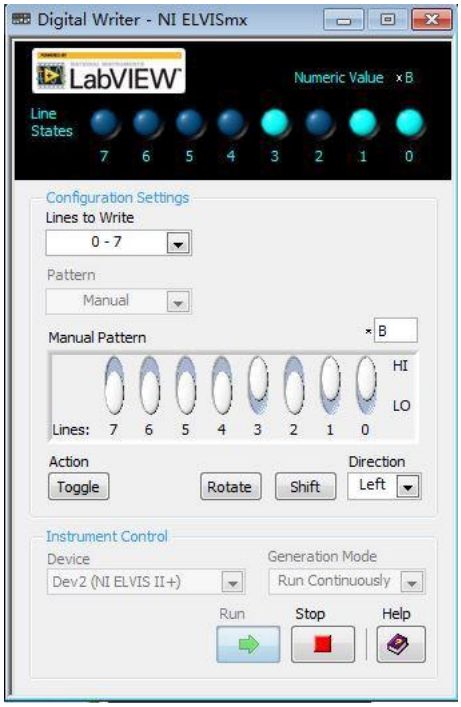

图 14. 控制 USB-4704 显示 1v、1Hz 的正弦波

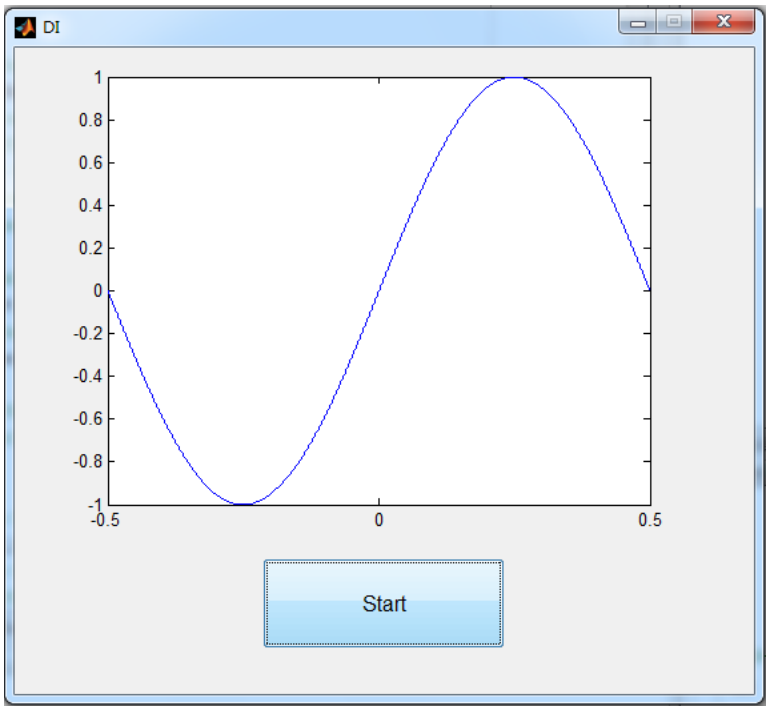

图 15. GUI 显示的波形

2. 出现的问题及解决方案

2.1 全局变量

问题描述：有时候需要在程序外部整体的改变子程序里面的变量，有时候是跨.m 文件使用，此时外部使用时会报错。

问题解决：改用 handle 句柄传递变量或者采用全局变量。

2.2 数据异常

具体描述：在 AI 的程序中，在第二次按采集数据，刷新界面并清空数据，重新进行记录时，会发现显示的数据出现乱跳的现象，而不是像第一次采集时得到索要采集的波形。

错误原因：排查后发现，出现该问题的原因是在第二次按下采集数据调用相应函数时，原来的时钟中断没有关掉，导致系统出现了两个时钟中断，使得采集的数据发生错乱。

解决方式：在第 N 次按下采集数据(N>1)时，清空原来的时钟中断。

2.3 程序运行越来越慢

具体描述：在 Aid 的程序中，随着采集数据的增多，程序的运行速度会越来越慢，导致不能很好的满足采样率的要求。

错误原因：通过断点调试，我们发现问题出在绘图函数中，我们通过 plot 函数这进行绘图（保留原图，将新的数据点与原图最后的数据点相连），该过程会随着数据量的增多运行时间变长，推测可能是保留原图后继续绘制时其函数可能会将之前的数据读入，随着数据的增加导致运行时间的变长。

解决方案：将动态显示由 plot 函数改为 animatedline 和 addpoint 函数，程序的运行时间显著减小，且不会随着运行而增长。

2.4 静态变量在 GUI 中无法使用

具体描述：静态变量在同一.m 文件下可以自由传递，但是在 GUI 使用中却无法自由传递。

错误原因：可能与 GUI 内部的原理有关，原因未找到。
解决方式：我们将所需的变量加入 handles 结构体中，通过传递 handles 达到相同的作用。

2.5 isempty 函数判断错误
具体描述：我们利用 isempty 函数判断变量是否为空来判断是否是第一次运行程序，然而 isempty 函数判断的结果经常与预期不符。
错误原因：在程序结束时为清空 global 变量，导致下次运行时该变量不为空，在析构函数中加入 clear global 命令后该问题得到解决。

2.6 deletefcn 中的命令在关闭 GUI 时无法有效执行
具体描述：在 GUI 的 deletefcn 中我们加入了关闭 GUI 后必要的处理，然而这些命令无法得到有效的执行。
解决方式：我们将放在 deletefcn 函数中的命令改为放在 closerequestfcn 中以后，命令得到有效的执行。

2.7 数据异常
具体描述：为了将端口的数据中特定的位数取出，需要进行取模和取余操作，Matlab 在计算取模操作时，会进行四舍五入，因此得到的结果与预期不符。
解决方案：我们将端口的数据转化为 double 类型后，Matlab 可以进行正常的取模运算。

2.8 数字输出信号频率偏低
具体描述：当用户设置数字输出一定频率，例如 30Hz 的方波信号时，输出的信号会略低于 30hz。
解决方案：Matlab 设置 Timer 的周期时会舍弃毫秒以下的时间，因此周期可能比实际值略高，因此输出信号的频率偏低。