摘要

随着军用武器装备高可靠性和小型化的需要，VXI 总线测试平台在各种武器装备的测试与维护系统中得到了广泛的应用。以任意波发生器为代表的通用信号源，是现代测试领域内应用最为广泛的通用仪器之一。由于它可产生包括各种理想及非理想的波形信号，不但可以产生纯净的正弦波、方波、三角波等常规波形，而且可以产生线性调频、调相、调幅信号以及程控占空比的低占空比方波，更为重要的是，能够根据用户测试需求生成任意波形。因此广泛用于通信、雷达、导航、宇航等领域。

本课题的主要任务是 VXI 任意波形发生器应用软件的开发，包括软面板和波形编辑器。虚拟仪器的最大特点即是抛弃了传统台式仪器的操作面板，取而代之的是计算及显示能力强大的计算机，使得人机交互界面更加人性化。本软件使用的开发工具是当前流行的虚拟仪器编程语言 LabWindows/CVI。通过软面板用户可以设置各种波形参数，进而控制硬件模块产生相应的波形信号。波形编辑器是设计过程中的难点及重点，集中体现了任意波形发生器的“任意”性，它具有强大的波形生成、编辑及处理功能。波形生成方式有包括数学方程式产生波形、波形库产生波形、导入数据文件产生波形等多种灵活多样的方式；手动绘制波形、波形段序列组合、算术处理、加窗、滤波、平滑等编辑处理功能大大丰富了软件的内容，使得仪器能够根据需要产生各种常规波形及非常规波形。

型号为 ES14V21 的 VXI 任意波形发生器已于 2003 年 1 月通过了设计定型鉴定，总体上相当于国际九十年代后期的水平。实验证明，该课题设计的 VXI 任意波形发生器应用软件操作简单、功能齐全，能满足多种波形的生成及编辑处理要求。

关键词：任意波形发生器　　软面板　　任意波形编辑器
回调函数　　LabWindows/CVI

Abstract

According to the high security and small volume in military weapon equip request, VXIbus flat roof have been applied to all kinds of weapon equip test and maintenance. The general signal sources represented by Arbitrary Waveform Generator(AWG) have become the most popular instruments in modern testing domains. AWG can not only generate conventional waves such as sine wave, triangle wave, square wave, etc, but also generate unconventional waves such as linear frequency modulation wave, phrase modulation wave, amplitude modulation wave etc, the most important, it can create arbitrary wave. So we can find AWG in communication, radar, navigation, space navigation etc fields.

The software exploitation of VXI arbitrary waveform generator is the main aim of this task. It includes Soft Panel and Arbitrary Waveform Editor(AWE). The most important property in Virtual Instrument(VI) is that the traditional operate interface has been replaced by PC, we can use PC's powerful calculate and display capability to make intercommunicate between user and instrument more easy. I made LabWindows/CVI as the exploitation tool. Soft Panel is used to set wave parameters and control instrument generate corresponding waveforms. And AWE is the emphases and difficulty in the whole software design. It embodies the "arbitrary" character of AWG. It has all kinds of generating wave tools such as through mathematic formula, through waveform storage, input special points and insert other points. In other hand, it has many edit and process methods, for example, user can draw wave by hand, make arithmetic process, and can add window, filter wave, smooth wave etc. So it help us create all sorts of waveforms.

The AWG model as ES14V21 has passed authentication in Beijing in Apl.2003, it is equal to the world level in later of 90' ages. Based on experiment, we can can prove the Arbitrary Waveform Editor has some characters ,it has powerful function in creating、editing、processing waveforms, and can be manipulated easily. In a word, it is satisfied to generate all kinds of arbitrary waveforms.

Key wods: Arbitrary Waveform Generator Soft Panel
Arbitrary Waveform Editor callback function
LabWindows/CVI

独 创 性 声 明

本人声明所呈交的学位论文是本人在导师指导下进行的研究工作及取得的研究成果。据我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得电子科技大学或其它教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示谢意。

签名：李俊义　　日期：2003 年 3月 3 日

关于论文使用授权的说明

本学位论文作者完全了解电子科技大学有关保留、使用学位论文的规定，有权保留并向国家有关部门或机构送交论文的复印件和磁盘，允许论文被查阅和借阅。本人授权电子科技大学可以将学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存、汇编学位论文。

（保密的学位论文在解密后应遵守此规定）

签名：李俊义　　导师签名：_____

日期：2003 年 3月 3日

第一章　引言

本章简介：虚拟仪器(Virtual Instrument，简称 VI)是现代化计算机技术和仪器技术深层次结合的产物，是当今计算机辅助测试（CAT）领域的一项重要技术。而其中的 VXI 总线技术更是虚拟仪器中具有极大影响力的典型代表。

基于 VXI 总线的任意波形发生器是 VXI 系统中的一个典型的和基础的激励源模块，在设计上它不仅保留了传统台式任意波形发生器所具有的各种功能，而且较传统波形发生器产生波形的种类多、频率高，可由计算机直接控制，具有广泛的应用和发展前景。

1.1　任意波形发生器简介

在现代电子测量仪器中，任意波形发生器(Arbitrary Waveform Generator)，简称 AWG 作为当代最新的一类信号源，正日益引起人们的重视。

作为一种基于 VXI 总线的卡式仪器，它可以很方便地与 VXI 总线测试系统集成，最大限度的发挥计算机和微电子技术在当今测试领域中的应用，是自动测试系统中重要的组成单元之一。它不仅能产生传统函数发生器所有的正弦、余弦、方波、三角波、斜波等常见波形以及衰减振荡正弦、指数形脉冲等复杂波形，这些信号具有所有毛刺、漂移、噪声及在被测产品离开实验室或生产车间将遇到的其他异常信号。目前，任意波形发生器已经广泛应用于磁盘驱动器测试、串行数据通信、汽车防报死、发动机控制、生物医学模拟等等领域。

任意波形发生器技术发展至今，引导技术潮流的仍是国外的几大仪器公司，如 Tektronix、Agilent，其任意波形发生器产品已经形成系列，从台式机到 VXI 模块都有不同档次的产品。

表征 AWG 的主要技术指标有以下三项：

(1) 最高取样速率：它决定了波形中的最高频率成份。由奈奎斯特定理可知：可还原的最高频率不大于二分之一的取样速率。具体说即 $f_m <= 1/2Fs$(f_m：模拟信号频宽，Fs：取样速率)，由此可见，最高取样速率越高，输出模拟信号的带宽就越大，频率特性就越好。

(2) 垂置分辨率：也称输出幅度分辨率，主要取决于 D/A 变换器，以位数表示，位数越高，离散的电压等级就越多，波形就越细、越逼真。

(3) 波形存储器容量：以每通道的字节或字长表示，容量越大，存储的波形就越复杂或可同时存储几个不同的波形。

1.2 虚拟仪器简介

所谓虚拟仪器，就是在以通用计算机为核心的硬件平台上，由用户设计定义，具有虚拟面板，测试功能由测试软件实现的一种计算机仪器系统。使用者用鼠标或键盘操作虚拟面板，就如同使用一台专用测量仪器。

随着虚拟仪器的出现，对于传统的操作面板带来了一次革命性的冲击，虚拟仪器的最大特点就是抛弃了传统的操作面板，仪器的操作与显示需借助计算机强大的计算与显示能力来实现，用户与仪器的交互界面变为了由计算机软件实现，也即软面板。

1.2.1 “虚拟”包含的两方面含义

(1) 虚拟的仪器面板

虚拟仪器面板上的各种“控件”与传统仪器面板上的各种“器件”所完成的功能是相同的。传统仪器面板上的器件都是实物，而且是用手动进行操作的，而虚拟仪器面板控件是外形与实物相象的图标，通、断、放大等对应着相应的软件程序。因此，设计虚拟面板的过程就是在面板设计窗口中摆放所需的控件，然后编写相应的程序。

(2) 由软件编程来实现的虚拟仪器测量功能

在以 PC 为核心组成的硬件平台支持下，虚拟仪器不仅可以通过软件编程设计来实现仪器的测试功能，而且可以通过不同测试功能的软件模块的组合来实现多种测试功能，因此在硬件平台确定后有“软件就是仪器”的说法。这也体现了测试技术与计算机技术深层次的结合。

1.2.2 虚拟仪器的构成

虚拟仪器由通用仪器硬件平台和应用软件两大部分组成。

(1) 通用仪器硬件平台

虚拟仪器的硬件平台由两部分构成——“计算机”和“I/O 接口设备”。计算机是硬件平台的核心；I/O 接口设备主要完成被测输入信号的采集、A/D 转换等，根据采用的不同总线及其响应的 I/O 接口硬件设备，如利用 PC 机总线的数据采集卡、GPIB 总线仪器、VXI 总线仪器模块、串口总线仪器等。

(2) 软件结构

虚拟仪器软件由两大部分构成——“应用程序”和“I/O 接口仪器驱动程序”。

应用软件实现虚拟仪器的前面板功能，达到控制仪器工作的作用。I/O 接口仪器驱动程序实现与特定外部硬件设备的扩展、驱动和通信。

1.2.3 虚拟仪器的特点

虚拟仪器的技术实质是充分利用最新的计算机软硬件技术来实现和扩展传统仪器的功能。具有如下性能特点：

- 灵活性：有面向总线的接口，功能由用户自己定义，可方便的与网络、外部设备等连接。
- 功能强大：基于计算机技术的功能模块可构成多种仪器，可以很容易的组建用户所需的测试系统。
- 使用方便：传统的仪器面板被计算机软件在屏幕上显示的软面板代替。利用计算机强大的图形显示功能，用户可以从图库里调出不同的图形控件，如开关、按钮、旋钮、波形显示区等，用户自己定义设计各种仪器，借助鼠标、键盘对软面板进行操作，就象操作传统仪器一样。

1.3 虚拟仪器编程语言——LabWindows/CVI 概述

虚拟仪器编程语言 LabWindows/CVI 是美国 NI(National Instrument)公司利用虚拟仪器技术开发的 32 位面向计算机测控领域虚拟仪器的软件开发平台，可以在多操作系统(如 Windows 98/2000/NT，Unix 等)下运行，它以 ANSI C 为核心，将功能强大、使用灵活的 C 语言平台与用于数据采集、分析和表达的测控专业工具有机的结合起来。它的集成化开发平台、交互式编程方法、丰富的功能面板和库函数大大增强了 C 语言的功能，由此可见软件在虚拟仪器中的重要作用。

和其他虚拟仪器开发工具相比，LabWindows/CVI 具有如下特点：

- 由于 LabWindows/CVI 的编程技术主要采用事件驱动方式与回调函数方式，编程方法简单易学。
- 运用 LabWindows/CVI 进行系统软件设计，以工程文件为主体框架，它包含了 C 源代码文件(*.c)、头文件(*.h)、用户界面文件(*.uir)等 3 个部分。全部软件调试好后，可将工程文件生成应用文件(*.exe)。
- 提供大量与外部代码或软件进行连接的机制，诸如 DLL(动态连接库)、DDE(共享库)、ActiveX 等。

同时，LabWindows/CVI 还有以下的模块：

- 用于仪器控制、数据采集和分析的交互式 ANSI C 编译软件包。
- 用于构成 GUI 的拖拉用户界面编程器。
- 用于快速样机开发的代码产生工具和内部编译器。
- 包含 GPIB、VXI、PXI、RS-232/485 等在内的各种仪器通讯总线标准的所有功能函数，使得不懂得总线标准的开发者也能够驱动不同总线标准的接口设备与仪器。

1.4 VXI 总线系统测试系统简介

VXI(VME Extensions for Instrumentation)原意是 VME 总线为组建仪器系统的扩充。VXI 实际上意味着体积更小、更快速的测试系统方案。它的先进设计思想体现在它组作为一个彻底的开放标准，使用户能以最短的时间组建系统。

VXI 总线是虚拟仪器的一个重要发展方向，由于其采用开放性和标准化的设计思想，得到众多仪器厂商的支持，获得了飞速的发展，现已应用于测试和自动化领域。

1.4.1 VXI 虚拟仪器的工作原理

虚拟仪器最大的特点就是在用户而不是生产厂家定义仪器的功能,其灵活性与高性能直接得益于日益发展的数字化技术和计算机技术。实际上大量测试使用的激励信号都可以先经过各种算法的软件编程与处理,通过计算机产生数字信号,经 DAC 变换成模拟信号,再经过滤波、调制、变频等处理成所需要的激励信号;而大量的测试功能也可以通过对被测信号的采集和预处理后,进行 ADC 转换变成数字量,由相应测量软件得到测试结果来完成。

1.4.2 VXI 总线系统的配置形式

VXI 总线测试系统有四种基本配置形式，可归为三大类。下面分别介绍这三类的配置形式的特点。

- GPIB-VXI 控制方式

该系统的硬件包括插入通用计算机的 GPIB 接口卡，位于 VXI 机箱的零槽的 GPIB-VXI 模块。该方式的优点是测试系统配置灵活，既可控制传统的有 GPIB 接口的器件，又可象控制 GPIB 仪器一样控制 VXI 器件。不足之处是 GPIB 总线数据传输速率只有 1Mbyte/s，远远低于 VXI 机箱背板总线的 40Mbyte/s 的传输速率。限制了 VXI 总线性能的发挥。

- 内嵌式计算机控制方式

这种方式是将一台计算机嵌入 VXI 机箱的零槽中，使用时接上显示器、键盘、鼠标等外设即可工作。这种配置所占体积小，计算机直接与 VXI 机箱背板总线连接，数据传输速率较高。但这种方式配置的是专用计算机，硬件和软件的升级比较困难，而且价格较高。

- 外部计算机控制方式

这种方案是通过某种计算机总线连接 VXI 器件和计算机，可用于这种连接的计算机总线包括 MXI 总线和 IEEE1394 总线。MXI 的配置形式硬件包括在外部计算机上和 VXI 机箱零槽的 MXI 接口卡以及连接两接口卡的 MXI 电缆。相当于将 VXI 背板总线直接与外部计算机相连，功能上与内嵌式计算机的方式是相同的，但可灵活的选择各类通用计算机的工作站。

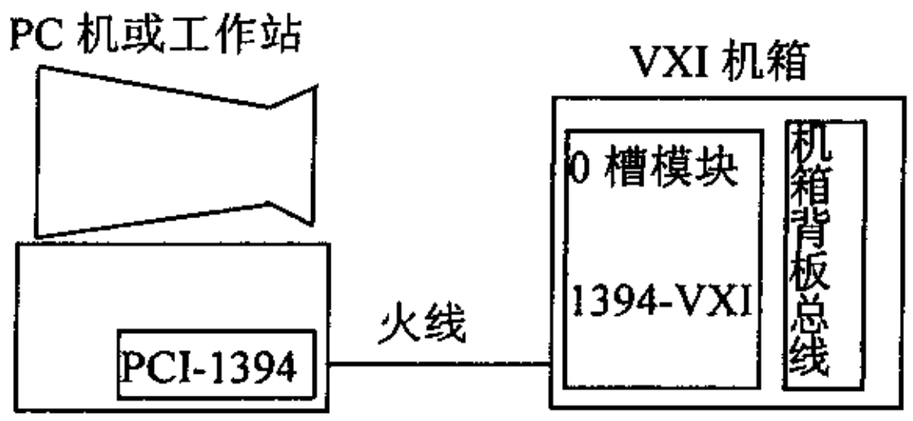

图 1-1

本课题研究的 VXI 测试系统采用的是外部计算机控制方式中的 IEEE1394 总线配置方式。IEEE1394 总线是为 PC 机与消费类电子设备之间的数据传输而定义的一种新型高速串行总线（也称火线），可支持 100/200/400Mbps 的数据传输速率。由于 1394 总线的高性能、灵活性和易用性，因而可以实现外部 PC 机与 VXI 主机箱之间的直接连接，1394-VXI 就是实现这种连接的 I/O 接口设备。如图 1-1 所示，来自 PC 机的数据经 PCI-1394 接口卡串行化处理，产生 IEEE1394 数据包，这些数据包经 1394 电缆传输到 1394-VXI 解包，在送到 VXI 机箱背板的其他模块。

1.5 本课题研究的主要内容

本课题的主要任务是设计 VXI 任意波形发生器的应用软件。该应用软件由两大部分组成，除了虚拟仪器本身应该具备的软面板之外，针对任意波形发生器的特殊要求，还包括功能强大的任意波形编辑器。

软面板主要完成控制模块产生各种标准或非标准波形的作用，用户可以通过鼠标和键盘对软面板进行操作，如选取波形类型、设置波形参数，然后发出命令，控制模块产生相应的波形信号。

任意波形编辑器是任意波形发生器的关键技术之一，是整个软件设计过程中的难点和重点。它主要完成各种任意波形的产生、编辑、处理等功能。该软件的

设计集中体现了VXI任意波形发生器的特色——能够产生“任意波形”，并且可以对产生的波形进行各种编辑处理，使得任意波形发生器成为敌我识别系统及雷达测试系统不可缺少的仿真信号发生器。

在第二章里从软件设计的总体框架介绍了设计思想及各模块之间的接口和关系，包括最上层的波形编辑器与软面板、软面板与仪器驱动器、仪器驱动器与底层监控软件各部分之间是如何组成一个整体的。任意波形编辑器的设计过程中工作量较大，并且涉及到了许多算法方面的技术及难点，因此在接下来的第三章到第五章用较大篇幅着重叙述波形编辑器的各个功能，以及遇到并解决的技术难点，包括波形产生、波形编辑、波形处理、辅助功能等，向您展示了一个功能强大、灵活多样的波形编辑器。

第二章　VXI 任意波形发生器应用软件的总体设计

本章简介：通过第一章的叙述，已经知道作为虚拟仪器的 VXI 任意波形发生器，用户对仪器的操作是通过软面板来实现的，因此软面板的设计应当简洁易用、功能完善。另外，针对任意波形发生器的特殊要求，任意波形的编辑软件应功能强大、灵活多样，以便满足各种任意波形的编辑。本章从总体构思上阐述虚拟仪器软件设计的总体思路及各软件模块之间的协调工作。

2.1　应用软件总体设计

根据 VPP(VXI Plug&Play，简称 VPP)系统规范的定义，虚拟仪器系统的软件结构由三大部分组成：最高层为虚拟仪器的应用软件；中间为仪器驱动器软件；底层是硬件监控软件。如图 2-1 所示为任意波形发生器的软件结构框图。我所做的工作主要是设计最上层的应用软件，包括波形编辑器和软面板。

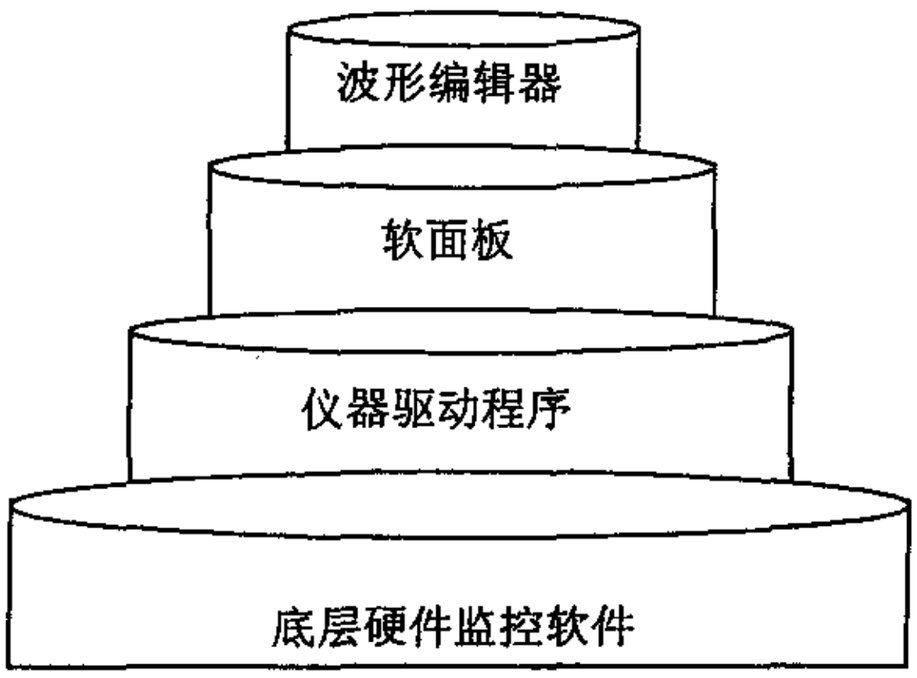

图 2-1　虚拟仪器的软件体系结构

(1) 任意波形编辑器

鉴于任意波形发生器的特殊性，仪器的应用软件由两部分构成——任意波形编辑和软面板。应用软件开发环境的选择可有两类：文本式编程语言（如Visual C++,Visual BASIC, LabWindows/CVI等）；图形化编程语言（LabVIEW, HPVEE等）。本次设计波形编辑器和软面板均用的是LabWindows/CVI。

任意波形编辑器是为了使用户能够编辑产生任意形状的特殊波形，从而大大扩大波形发生器的功能。

(2)　软面板

软面板既是仿真台式仪器的操作面板，同时它兼有Windows应用软件的界面特色。软面板是用户控制仪器产生波形的中介。用户在计算机上操作软面板就如

同操作传统仪器的各种按键、旋钮、数值键等。通过它可以修改波形的幅度、频率、偏移等参数，以此来控制仪器产生用户需要的波形信号。它接收用户输入的各种命令、波形参数后，调用下一层的仪器驱动程序，控制仪器进行工作。

(3) 仪器驱动程序

仪器驱动程序是连接上层应用软件与低层输入/输出 (I/O) 软件的纽带和桥梁。每个仪器均有自己的仪器驱动程序，为用户提供了用于仪器操作的较抽象的操作函数集。对于应用程序来说，它对仪器的操作是通过仪器驱动程序来实现的；仪器驱动程序对于仪器的操作与管理又是通过 I/O 软件所提供的统一基础与格式的函数库(VISA库)的调用来实现的。

(4) 硬件监控软件

它存在于仪器与仪器驱动程序之间，完成对仪器内部寄存器单元进行直接存取操作，为仪器与仪器驱动程序提供信息传递的底层软件层。

2.2 软面板的设计框架

ES14V21型号的VXI任意波形发生器能够产生七种标准波形——正弦、三角、斜波、方波、脉冲、直流、噪声；还能产生各种调制波形——调幅、调频、脉冲调制、频移键控、扫频；最后还能产生用户编辑的各种任意波形。作为任意波形发生器最为特殊的一点是可以将用户编辑的任意波形作为调制波形中的载波或调制波，这样就大大增强了任意波形发生器的功能。

表2-1所示为VXI任意波形发生器所能产生的波形种类列表，其中每一个"X"表示一种允许的组合。(注：如果把输出函数转换到当前所选的调制方式不允许的类型时，则会自动将载波设置为正弦波。)

	正弦波	方波	三角波	锯齿波	噪声	任意波
AM载波	X	X	X	X		X
AM调制波	X	X	X	X	X	X
FM载波	X	X	X	X		X
FM调制波	X	X	X	X	X	X
FSK调制	X	X	X	X		X
脉冲调制	X	X	X	X		X
扫频	X	X	X	X		X

表2-1 ES14V21型号的VXI任意波形发生器能产生的波形类型

由上表我们可以看得出，VXI任意波形发生器能产生的波形信号种类是非常丰富和齐全的。所有这些波形的产生都通过软面板的操作来实现。因此软面板的设计就要符合这样的要求。图2-2所示为软面板的设计总体框架。

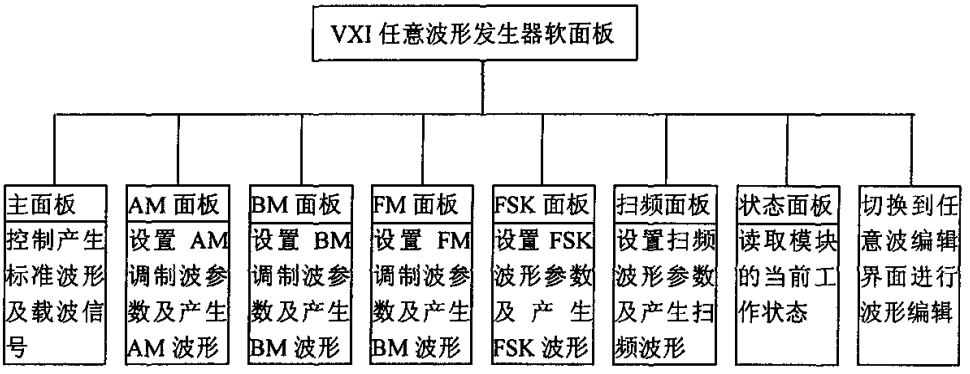

图 2-2　VXI 任意波形发生器的软面板功能框图

七种标准波形、任意波形，以及各种调制波形的产生都直接在软面板上控制，先设置波形的各种参数，然后发出命令控制模块产生相应波形。

图 2-3 所示为 VXI 任意波形发生器软面板的界面。即相当于传统仪器的操作面板，图中所示为控制硬件模块产生各种标准波形或任意波形的主界面，从该面板可以切换到其它面板产生诸如 AM、BM、FM、FSK、SWEEP 等各种波形。

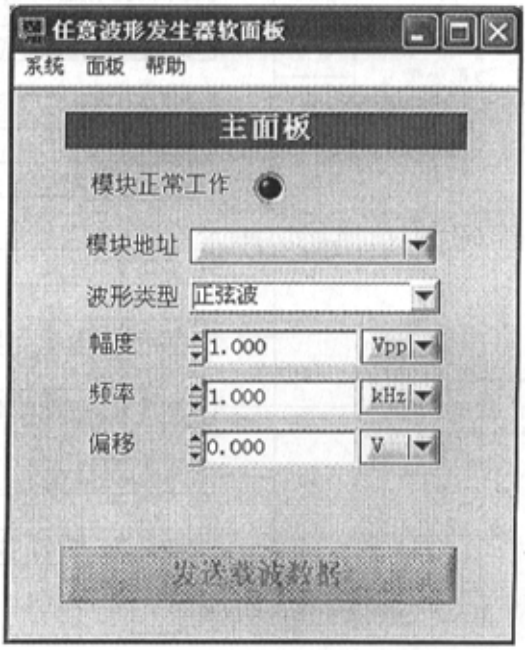

图 2-3　软面板界面

2.3 波形编辑器的设计框架

图 2-4 所示为任意波形编辑器的主要功能。该软件主要分为波形产生、波形编辑、波形处理、辅助功能等。另外，波形编辑器将波形编辑出来，还要将波形数据发送到硬件存储区，硬件支持一次存储四个不同的波形信号，然后再到软面板上控制模块调用存储区里的波形数据产生波形。

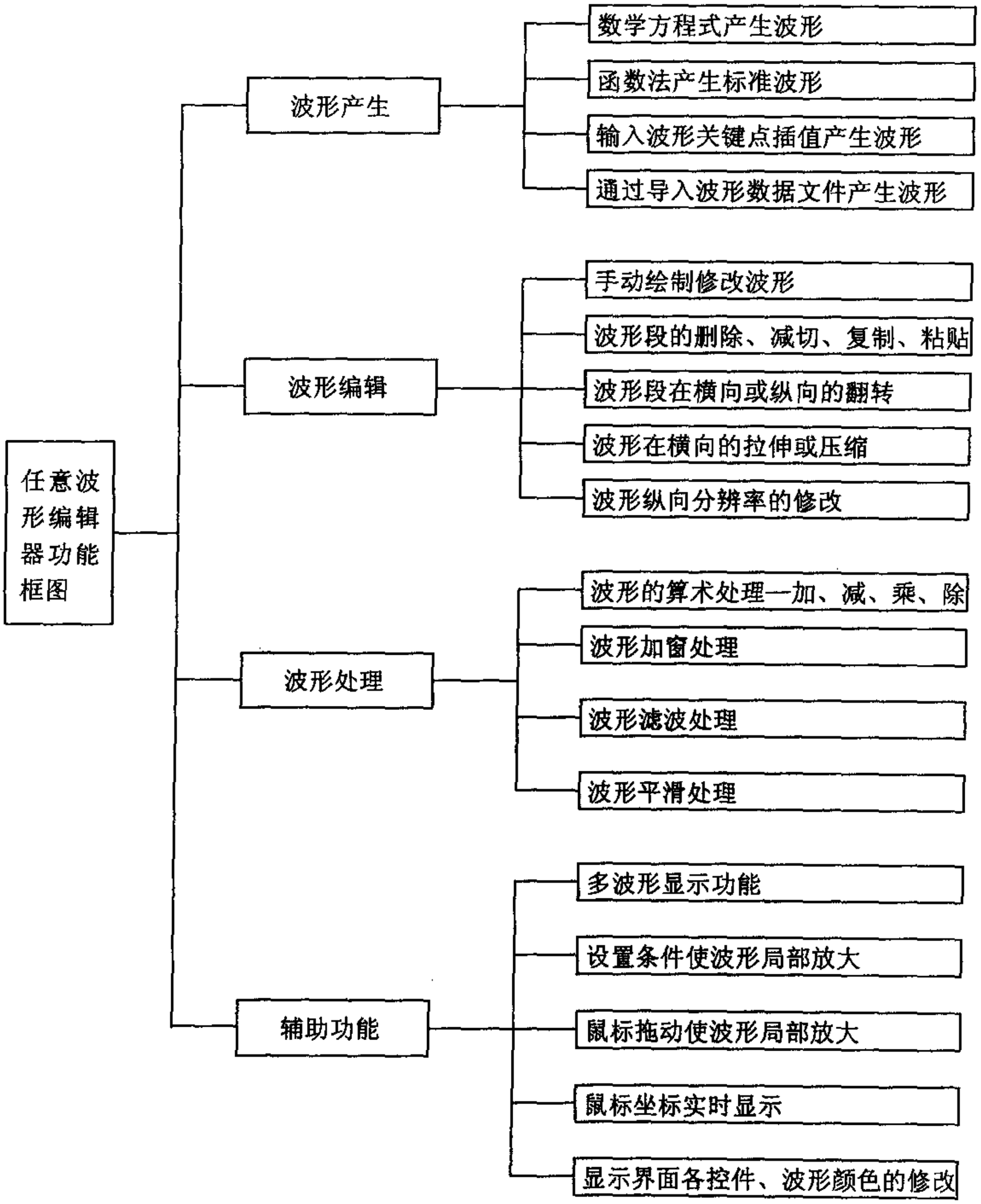

图 2-4 波形编辑器功能框图

从图 2-4 可看出，为了适应各种方面的要求，任意波形发生器的波形编辑器具有强大的功能。除了常见波形外，它还可以模拟任何一种特殊波形，如产生从高频至超低频的连续彼和脉冲波，并且可迭加上随机噪声；可以通过数学方程式产生相应的波形；可以把数字示波器采集的波形信号重新用信号发生器产生出来；可以手动绘制波形；可以把各种波形任意组合或进行算术处理等。因此，AWG 除了传统函数发生器的应用领域外，还可用于声学、电视雷达、通信广播、航天航空、电子测量、自动控制等诸多需要特殊波形的科研生产领域。任意波形编辑器是 VXI 任意波形发生器的关键技术及特色之一。为了能够编辑处理各种特殊波形，使任意波形发生器成为功能强大的仿真实验信号发生器，波形编辑器需要具备诸多产生波形及编辑波形的功能。

(1) 波形产生

当要对一个波形进行编辑的时候，必须先在显示区构造一个波形，然后对此波形进行修改，最后将符合用户要求的波形数据发送给模块。这时就涉及到波形产生的诸多方式。根据需求分析，该软件设计了四种波形产生的方式。

☆ 波形库产生标准波形

☆ 输入数学方程式产生波形

☆ 输入特征点插值产生波形

☆ 数据文件导入生成波形

(2) 波形编辑

当用以上各种方法产生了波形信号，就要对波形进行修改编辑，以获得符合用户需求的波形。

☆ 手动绘制波形

☆ 波形段的编辑——对某一波形复制、减切、粘贴、删除等

☆ 波形段拉伸或压缩

(3) 波形处理

主要是对波形进行算术、加窗、滤波、平滑等处理。

☆ 算术处理

☆ 加窗处理

☆ 滤波处理

☆ 平滑处理

(4) 辅助功能

除以上提到的三类主要功能，该软件还有许多辅助功能，这些辅助功能方便波形观察。执行这些操作的时，并不改变波形数据本身，只是显示效果有所改变。

☆ 多波形显示功能

☆ 波形局部放大

☆ 鼠标拖动放大波形

☆ 鼠标横纵坐标实时显示

☆ 显示界面上各控件颜色的改变

(5) 发送波形数据到硬件存储区

硬件存储区的大小是固定的 64k*12bit。而对于编辑好的波形，它的最大长度可能大于 64k 个点，也可能小于 64k 个点；另外，波形数据是浮点数类型，而硬件要求传送的是 12bit 的整数形式，即 0-4095 之间的整数。因此，根据硬件的要求，在波形数据发送之前，需要进行一系列的处理，处理为 64k*12bit 的格式，之后才能进行发送，程序按以下步骤进行。

① 将波形数据处理为 64k 个数。软件先检测当前波形数据是否是 64k 个数据，如果不是，用插值法先将之拉伸或压缩为 64k 个数据。

② 将处理好的 64k 个浮点型数据处理为 0-4095 之间的整数。此时长度为 64k 的数组中存储的每一个数据都是浮点型的，为了使数据转化为 12bit 的整数以满足传送需要，将每个数根据按照下述公式进行计算：

$$Y_2[i]=(unint)(4095-(Y_1[i]-Y_{min})*4095.0/(Y_{max}-Y_{min}))$$

$$i=0,1,...65535$$

其中，Y_1 表示处理前的数组，Y_2 表示处理后的数组，Y_{max} 是 Y_1 中的最大值，Y_{min} 为 Y_1 中的最小值。

③ 将处理好的 64k*12bit 个数据分块发送到硬件存储区。程序设计的是将之分为 128 块，每块 512 个数。即双重循环，外层循环 128 次，内层循环 512 次。

2.4 软面板与波形编辑器的接口

VXI 任意波形发生器的软面板与波形编辑器本来是两个独立的软件，为了用户的操作方便，现将两个软件合在一起。从软面板的菜单可以进入到波形编辑界面，由于此处的波形编辑已不是独立的一个软件，而是作为软面板的一部分，因此不再将之称为“波形编辑器”，而称之为“波形编辑界面”。图 2-5 所示为软面板操作以及进入波形编辑界面的流程图，从中也可看出二者的接口实现。

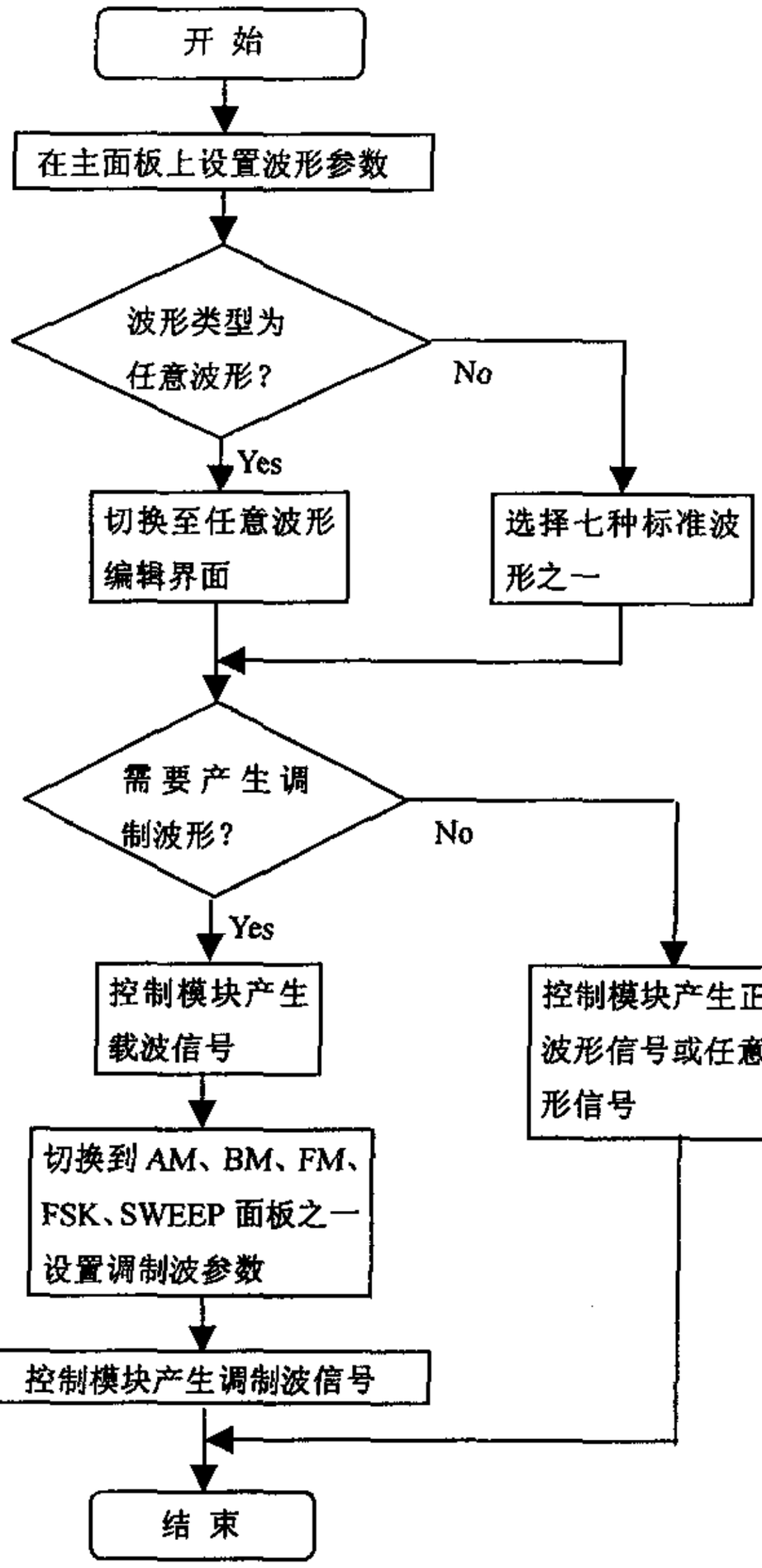

图 2-5 软面板的操作流程

从软面板的菜单可以进入产生各种波形的各个面板，主面板用来控制产生载波信号或标准波形信号，在主面板上有一个控件是输入波形类型的，当选择的波形类型是“用户波 1、2、3、4”，即是需要用户先编辑一个用户波形，将波形数据存储之后再从主面板上控制模块产生。

2.5 仪器驱动器接口

作为 VXI 总线虚拟仪器的核心，仪器驱动器是完成对仪器硬件控制的纽带和桥梁。它主要包含以下 5 个部分：

① I/O接口：提供仪器驱动器与仪器的通信能力；

② 操作接口：即虚拟仪器面板，测试人员通过对该面板的控制即可完成对仪器的操作；

③ 编程接口：可将仪器虚拟面板的操作转化为相应仪器代码；

④ 功能库：描述仪器所能完成的测试功能；

⑤ 子程序接口：使仪器驱动器在运行时能调用它所需要的软件模块。

根据 VPP 的有关规定，VXI 总线仪器模块开发商应配套模块提供相应的虚拟面板,即操作接口,以便测试人员对仪器功能有直观了解。虚拟面板建立之后，通过编程环境 LabWindows/CVI 调用通用的仪器驱动程序库，即动态连接库*.DLL，于是就将测试人员在虚拟面板上的一系列直观操作转化为仪器底层监控程序能够识别的相应的程控代码，这样就达到了控制仪器工作的效果。动态连接库(DLL)实际上是一个函数库，只有在应用程序运行期间，DLL 中的函数才被随时调用和连接，和静态连接库相比，动态连接库可以和其它的应用程序共享库中函数和资源，减少了因重复拷贝而造成的应用程序的冗长以及计算机资源的占用。

当启动软面板程序时，首先执行的是检测 VXI 模块是否与计算机连接正常，是否工作正常，只有当模块连接正确，并且工作正常，软面板才能发出一系列的控制命令。本软件在设计为模块工作正常，则软面板上的指示灯亮，否则为灭。

下面的函数是仪器驱动器库中用来控制模块产生相应波形信号的库函数，它们主要完成控制产生载波和各种调制波信号的功能。

(1) 控制产生载波信号

```
ViStatus _VI_FUNC es1403_conf (ViSession vi, ViInt16 func, ViReal32
                               freq, ViReal32 ampl, ViReal32 offs,
                               ViReal32 duty)
```

(2) 控制产生AM调制波信号

```
ViStatus _VI_FUNC es1403_amConf  (ViSession vi, ViInt16 Sour, ViReal32
                                  Dept, ViInt16 Func, ViReal32 Freq,
                                  ViInt16 Stat)
```

(3) 控制产生BM调制波信号

ViStatus _VI_FUNC es1403_bmConf (ViSession vi, ViInt16 Source, ViInt32 Ncycle, ViReal32Phase, ViReal32 Rate, ViInt16 State, ViInt16 trigSour)

(4) 控制产生FM调制波信号

ViStatus _VI_FUNC es1403_fmConf (ViSession vi, ViReal32 Dev, ViInt16 Func, ViReal32 Freq, ViInt16 Stat)

(5) 控制产生扫频波形信号

ViStatus _VI_FUNC es1403_freqSweep (ViSession vi,ViInt16 Source, ViReal32 swStart, ViReal32 swStop, ViReal32 Time, ViInt16 Space, ViInt16 State, ViInt16 trigSour)

(6) 控制产生频移键控波形信号

ViStatus _VI_FUNC es1403_fskConf (ViSession vi, ViInt16 Source, ViReal32 hopFreq, ViReal32 Rate, ViInt16 State, ViInt16 trigSour)

2.5 本章小节

本章从软件的总体功能概括描述了型号为 ES4V21 的 VXI 任意波形发生器应用软件的设计构思。阐述了模块的配套软件由哪几部分组成，各部分完成的是什么功能，软面板是如何实现对虚拟仪器的操作，即它是如何模拟传统仪器的界面，它与传统仪器相比较有那些优点及使用上的方便。

接下来在第三、四、五章里分别对任意波形编辑器的波形产生、波形编辑及处理、辅助功能等内容进行详细介绍。

第三章　波形的产生方式

本章简介：本章首先介绍了虚拟仪器开发工具 LabWindows/CVI 的事件驱动及使用回调函数编程的特点。接下来详细介绍了四种波形产生的方式 —— 数学方程式生成波形；输入特征点插值产生波形；波形数据文件的导入；手动绘制波形。用户在具体的软件使用工程中可以根据自己的需要和习惯选择合适的波形产生方式，每种方式有它的优缺点，适用于不同的场合。

3.1　LabWindows/CVI 的事件驱动编程

LabWindows/CVI 将源代码编辑、32 位 ANSI C 编译、连接、调试以及标准 C 库集成在一个交互式开发环境中。因此，用户可以快速方便的编写、调试和修改应用程序，最后生成可执行文件。使用 LabWindows/CVI 设计的应用程序可脱离 LabWindows/CVI 开发环境，用户最终看见的是和实际物理仪器相似的虚拟仪器软面板。

在应用程序开发环境 LabWindows/CVI 中设计一个用户接口，实际上是在用户计算机屏幕上定义一个相当于物理仪器面板的软面板文件，它由各种控件（如命令按钮、开关、指示灯等）构成。用户选中这些控件就可以产生一系列用户接口事件。例如：当一个用户单击一个按钮，就会触发一个用户接口事件，并传递给开发者编写的 C 语言驱动程序中对应的函数。这是运用了 Windows 编程的事件驱动机制。Windows 下的事件很多，但大多数都很少用。在 LabWindows/CVI 中，也只有几种事件需要识别处理。事实上，在 LabWindows/CVI 的图形用户界面（GUI）上的一次操作就会同时产生多种接口事件，如用鼠标单击命令按钮将触发下列事件，驱动应用程序去做相应的处理。

EVENT_GOT_FOCUS　事件：表示当命令按钮处于非激活状态，单击后将激活该控件；而当按钮本身处于激活状态，则获取输入并产生一个 GOT_FOCUS 事件。

EVENT_LEFT_CLICK　事件：在命令按钮或其它控件上单击鼠标左键就会产生 LEFT_CLICK 事件，LabWindows/CVI 用户接口控制项能识别鼠标的左击、右击、左双击、右双击。

EVENT_COMMIT 事件：当用户松开鼠标键时，就会产生一个 COMMIT 事件，表明用户对某控件完成了一个确定的事件。

还有其它许多事件类型，这里就不再一一介绍。在下面的叙述中对重要的事

件处理将做特殊的说明。

LabWindows/CVI 中可用的各种类型的控件项，在软面板编辑器中将显示不同类型的信息，并产生不同的接口事件，程序开发人员编写 C 程序来处理这些事件，在 LabWindows/CVI 中提供了两种基本的事件驱动编辑方法：回调函数法和事件循环处理法。

(1) 回调函数法

回调函数法是开发者为软面板上的控件编写一个独立的函数，当触发某个事件，就调用该函数进行相应的事件处理。例如：在编写触发波形产生的函数，它对应 GUI 上的“触发”按钮，当该按钮被鼠标点击，所有信息都送至回调函数，由它进行波形触发的一系列操作。

(2) 事件循环处理法

即是使用一个事件循环来处理用户接口的 COMMIT 事件。在循环处理方法中只处理 GUI 控制项产生的 COMMIT 事件。通过 GetUserEvent 函数过滤，将所有的 COMMIT 事件区分开，识别出是哪个控件产生的事件，并执行相应的处理。图 3-1 为事件循环处理机制示意图。

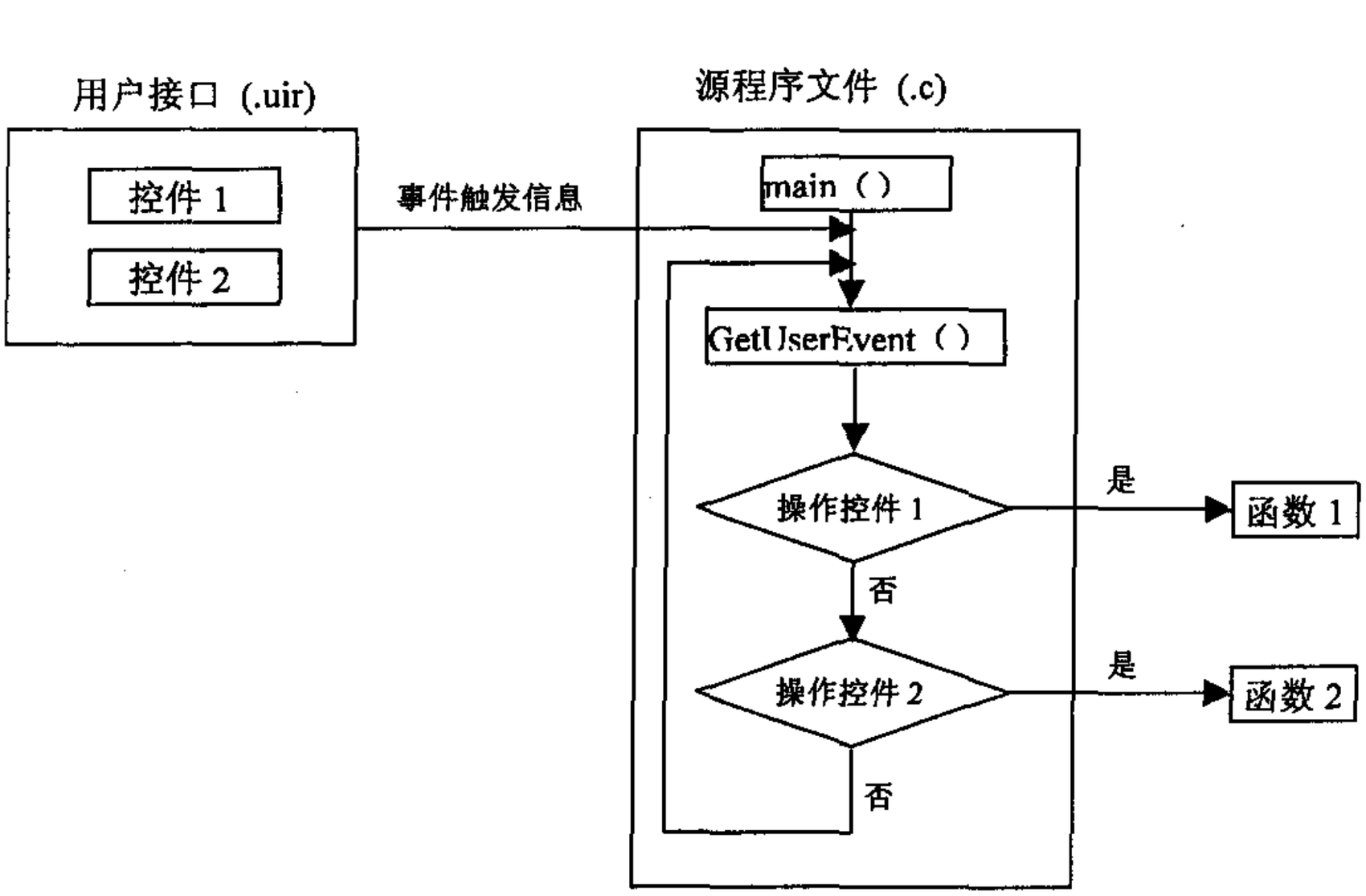

图 3-1 事件循环处理机制示意图

本软件使用的是第一种方法——回调函数法。

3.2 数学方程式生成波形

本软件设计的波形产生的方式主要有以下几种：从波形库中提取波形；数学方程式生成波形；手动绘制产生波形；输入特征点插值产生波形；波形数据导入生成波形等。下面各小节将对上述各种方法的实现详细阐述。

下图所示为数学方程式生成波形的对话框。

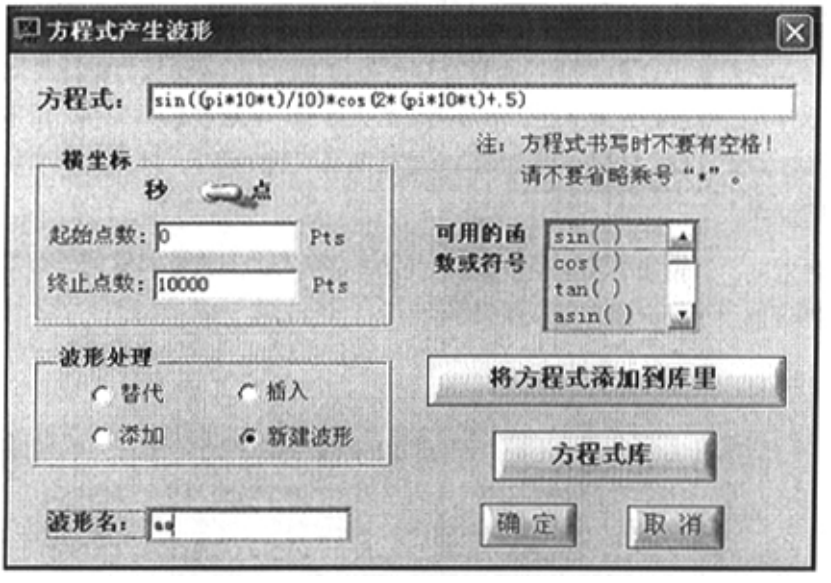

图 3-2　数学方程式生成波形的对话框

在图 3-2 所示的对话框的“方程式”控件中输入数学方程式，并设置好其它参数项，按下“确定”键，即可产生一个对应于数学方程式的波形。

数学方程式的形式为y = f(t)，自变量是离散的时间值，每一个自变量都唯一的对应一个y值，不同的表达式得出一组不同的波形数据。在用方程式计算每一个离散点的y值时，t值是如何确定的呢？软件设计中设置了一个系统时钟，时钟周期对应的是波形显示区X轴每个点的时间量。例如在图3-2上设置各参数，如设置起始点为0，终止点为9999，那么这个波形长度为10000个点。而这10000个点具体代表多少时间量，还需设置时钟频率来确定，设时钟频率为1MHz，即X轴上每一个点对应1uS的时间量，这时可以确定10000个点的波形段代表10mS（= 10000 * 1 / 1000000）的时间量。展开计算时，循环10000次，依次将自变量 t 替换为第0个点0.000001S，第1个点0.000002S，……，第9999个点0.01S，这样就可以计算出10000个点各自的Y坐标值，也即波形的幅度值数组。

图3-2所示对话框中的“可用的函数或符号”控件列出了在方程式书写中可以使用的函数名和符号种类。它们是+，-，*，/，^；cos()， sin/sine()，tan()，asin()，acos()，atan()，sinh()，cosh()，tanh()， abs()，sqrt()，exp()，log10()，ln()，floor()，ceil()，pow()，rand()。（注：其中floor(x)求不大于 x 的最大整数；ceil(x) 求不小于 x 的最小整数。符号pi替代数值π）

实现数学方程式生成波形的算法采用的是数据结构中堆栈的原理，堆栈的一个经典应用就是由数学方程式求最后结果的算法。

栈——它是限定仅在栈顶一端进行压入(push)或弹出(pop)操作的线性数据结构。栈的主要特点是先进先出，即后进栈的先处理，先进栈的后处理。通常栈可以用顺序方式存储，分配一块连续的存储区域存放栈中的表，并用一个变量指向当前的栈顶。

栈的定义：假设栈的元素个数最大不超过n，所有的元素都具有同一数据类型int，则可以用下列方式来定义栈类型stack，

```
typedef struck
    {
      int s[n];
      int *p;
    }stack;
```

其中变量 top 指向栈顶，称为栈顶指针。n 是一个正整数，表示栈中可容纳的最多元素个数。

输入算术表达式后，就要对它进行计算，计算的整个过程按以下步骤进行：

① 检查表达式的语法是否正确

这时的表达式作为一个字符串存放在一字符型的数组中，对这个数组进行从左到右的扫描，检查表达式中各元素的语法是否正确，即查看是否满足该软件所支持的数据类型、函数类型、以及符号类型等；

② 将运算数和运算符存放到各自的数组中

当确定语法正确之后，再进行一次从左到右的扫描，将整个表达式的字符分为运算数和运算符，分别存放到运算数数组和运算符数组中。

③ 将运算符和运算数按优先级别出栈进行运算

如果识别字符 C 是数字，则将 C 压入运算数的栈中。如果 C 是运算符，

则先对运算符的优先级别进行判断，当 C 的运算符级别比运算符栈顶的运算符的级别高时，直接将 C 压入运算符栈中，此时它位于栈顶；而如果 C 的级别低于运算符栈顶的运算符的级别时，从运算符栈中弹出该级别高的运算符，再从运算数栈中弹出两个运算数参与运算，运算的结果压入运算数栈中。

运算符的优先级别为+、-最低，其次*、/，最高是左括号“(”。即扫描到“(”时，不论运算符栈顶为何运算符，都将“(”入栈，入栈后的左括号优先级别降低到低于+、-、*、/的级别。当扫描到右括号“)”时，运算符栈将弹出运算符进行运算，直到与该右括号对应的左括号弹出为止。

表3-1所示为运算符入栈优先数和栈顶优先数

同一个运算符在进栈之前和在栈顶的优先级别是不同的。例如“(”，在入栈之前，它的优先级别最高，不论运算符的栈顶为何运算符，它都进栈；而进栈之后，除了运算符“)”的优先级别低于它之外，其它的运算符优先级别都高于它，不论再扫描到何运算符，该运算符都将进栈。这样就需要对运算符优先级别划分为进栈优先数和栈顶优先数。

运算符	进栈优先数	栈顶优先数
(	5	1
*	3	3
/	3	3
+	2	2
-	2	2
数学函数	4	4

表 3-1 运算符的进栈优先数和栈顶优先数

下面所述为各种运算符、数值或函数等在运算中的具体处理方法。

① 负号的处理

负号与减号同为一字符“-”，但在表达式中的优先级不同，例如 –2*5 与 3-2*5，可看出负号的优先级高于减号的。这就需要对字符“-”进行判断，根据表达式的语法规则，负数必须用括号括起来，除了写在表达式开头处可以省略括号，例如 –2*8，8*(-2)。表达式的第一个字符为“-”或“-”的前一个字符为“(”

时，可判断该“-”是负号，将其压入运算符栈，同时在运算数栈中压入数值0，扫描到运算数时，运算数进运算数栈，扫描到下一个运算符时，由于负号的优先级高，从运算数栈中弹出两个运算数，运算符栈中弹出负号进行运算，即0减另一个运算数，结果压入运算数栈中。负号的运算就是减号的运算，唯一的区别就是负号的优先级大于减号的优先级。

② 运算数的处理

扫描到运算数时，其实为字符0-9，在C语言中，字符变量是有值的，但并非实际想表达的数值。例如字符“2”，ASIIC码值为50，与实际要表示的数值2相差48，十个字符0-9的ASIIC码值均与实际数值相差48。根据这个规则就可以判断，只要检测到ASIIC码值在48-57之间的字符即为运算数。把运算数进栈之前，将每个字符变量减去48，达到与实际要表示的数值一致。

③ 多位整数与浮点数

扫描到运算数为多位整数时，例如123，先将1压入运算数栈，再向后扫描到2，由于1和2之间没有被运算符隔开，可判断其为多位数，此时从运算数栈中弹出1，进行运算1*10+2，将结果12压入栈中，再向后扫描到3，同上，12和3之间没有运算符隔开，于是从栈中弹出12，进行运算12*10+3，结果123压入栈中，再向后扫描并确认后面的字符不再是运算数，至此，多位数123检测完毕。

扫描到运算数为浮点数时，例如1.23，先将1压入栈中，向后扫描到小数点“.”，可知该运算数为浮点数，先不进行处理，接着扫描小数点后的第一个字符为2，此时弹出1，进行运算1+2/10，结果1.2进栈，接着向后扫描，如果还是运算数，则将1.2出栈，进行运算1.2+1/100，结果1.23进栈，再向后扫描并确认后面的字符不再是运算数，至此，浮点数1.23检测完毕。

④ 字母的处理

对字母的处理与对+、-、*、/字符的处理不一样。+、-、*、/字符可以归入运算符中。而扫描到字符为 A-Z 或 a-z 时，可能有三种情况——常量、变量、函数名。下面分这三种情况具体叙述。

常量

该软件的算术式处理中涉及到的常量有 PI 或 pi，用来代替 π，即数值3.14当依次扫描检测到该字符串书写正确，则把数值3.14压入运算数栈中；否则返回出错信息。

变量

该软件的算术式处理中涉及到的变量是 t，代表离散时间量。例如 4+t，在扫描到字符“t”时，将 t 用它此时表示的时间量代替，将此数值压入运算数栈中，（具体的时间量计算在前面已有叙述）。

函数名

根据该软件支持的可以运算的函数种类——cos( )， sin/sine( )，tan( )，asin( )，acos( )，atan( )，sinh( )，cosh( )，tanh( )， abs( )，sqrt( )，exp( )，log10( )，ln( )，floor( )，ceil( )，pow( )，rand( )，在检测过程中，扫描到函数名时，将扫描到的函数与上述函数名进行搜索配对，配对成功的将之压入运算符栈。函数的优先级别高于+、-、*、/四则运算。配对不成功的返回错误信息。

根据以上的叙述，结合一个例子，对算术方程式的解析过程给予详细的叙述。例如表达式 exp(0.5*log(t))+15，设此时 t=1。

<1> 程序运行之后将该字符串读入字符型数组，再从左至右进行扫描，连续扫描到三个字符“exp”可判断为函数名，将其压入运算符栈。

<2> 检测到“(”，它的优先级高于栈顶“exp”的优先级，将“(”入栈，其栈顶优先数降到1。

<3> 扫描到字符“0”，可判断其为运算数，将0入运算数栈；接着扫描到小数点“.”，可判断该运算数为浮点数，于是将0出栈，进行运算0+5/10，结果0.5入运算数栈。

<4> 接着扫描到字符“log”，并配对检测到它是函数名，其优先级高于栈顶“*”的优先级，于是函数 log 入栈。

<5> 扫描到 log 后面的“(”的优先级高于 log 的优先级，“(”进栈。

<6> 扫描到字符“t”，前后只有一个字母，可判断其为变量，变量值已设置为1，将1压入运算数栈。

<7> 扫描到第一个“)”，将最后一个入栈的“(”出栈与之配对。第二个“(”

和第一个“)”之间没有运算符，“(”出栈后运算数栈不变。

<8> 扫描到第二个“)”时，在与之配对的第一个“(”之间有“*”和“log”两个运算符，运算符栈顶“log”出栈，因为是函数出栈，运算数栈只需弹出一个运算数，运算数栈顶1出栈进行log1运算，结果0进运算数栈。

<9> 运算符“*”出栈，数值0和0.5出栈，进行运算0.5*0，结果0入运算数栈。这时第一个“(”出栈，与第二个“)”的配对结束。

<10> 扫描到字符“+”，可判断其为运算符加号，其优先级低于运算符栈顶exp的优先级，运算数栈顶0出栈，进行运算exp(0)，结果1入运算数栈。

<11> 扫描到运算符“+”，将之入运算符栈。

<12> 向后扫描到字符“1”，将之入运算数栈，再扫描到运算数5，可知是多位数，运算数栈顶1出栈，进行运算1*10+5，结果15入运算数栈。

<13> 再继续扫描到字符串的结束符“\0”，这时运算符栈中的运算符“+”和运算数栈中的运算数“1”、“15”都要出栈，进行运算1+15，结果为16。判断运算符栈和运算数栈都为空，则程序运行结束，最后结果为16。

图 3-3 是由方程式 *sin((pi*10*t)/10)*cos(2*(pi*10*t)+.5)* 的波形。

图 3-4 是用数学方程式生成波形的流程图。

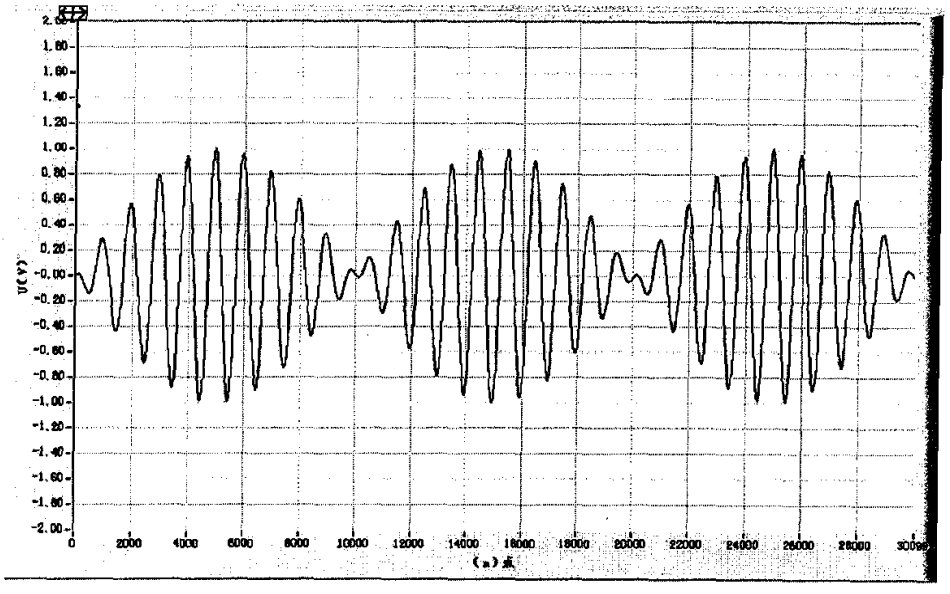

图 3-3 方程式 sin((pi*10*t)/10)*cos(2*(pi*10*t)+0.5) 的波形

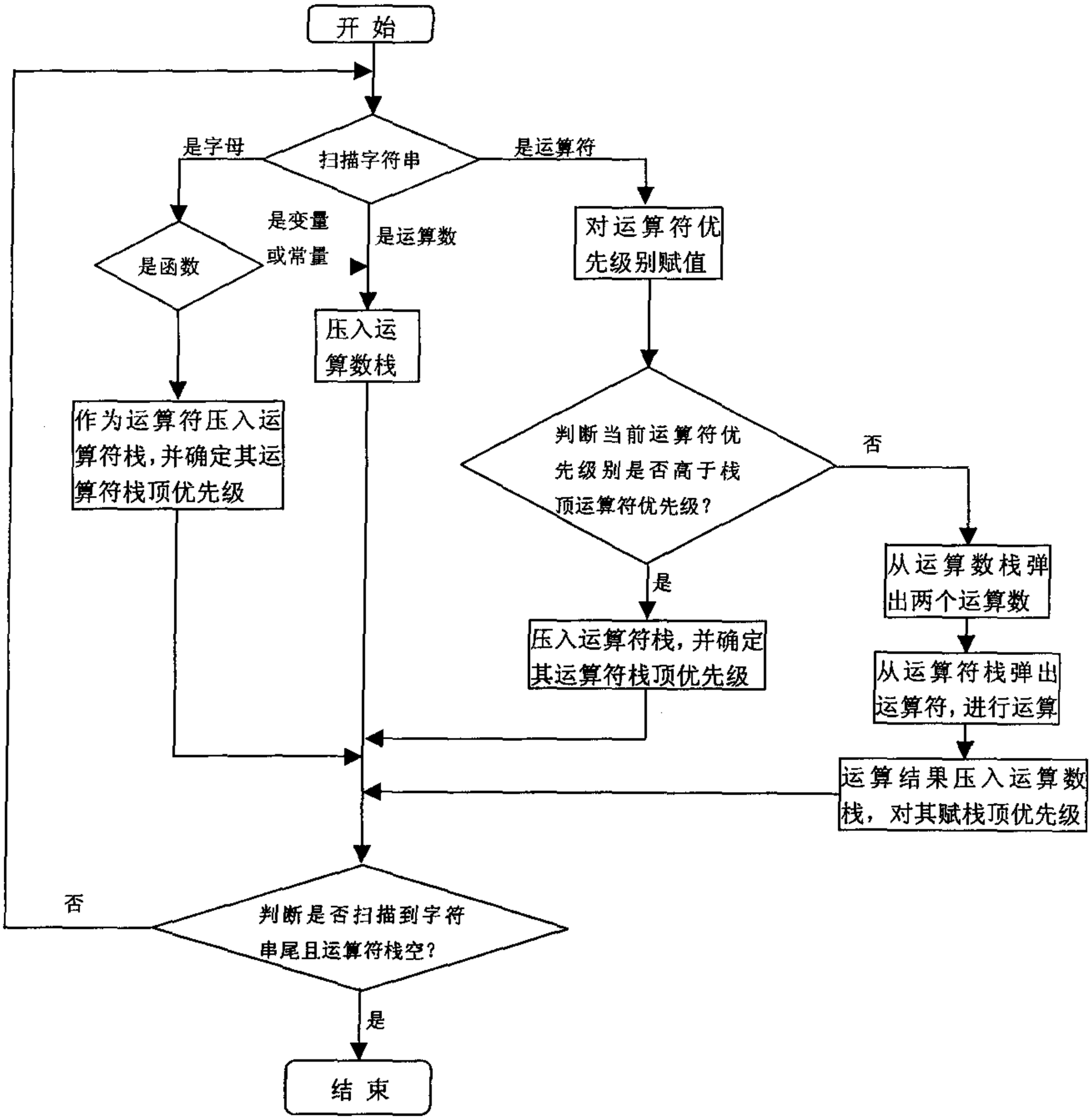

图 3-4　由数学方程式生成波形的流程图

3.3　手动绘制产生波形

手动绘制产生波形是波形生成方法中最直观、最方便的方法，它同时也充分体现了任意波形编辑过程中的“任意性”。

具体操作时，选中“手动绘制”菜单项，将鼠标移动到需要的起始位置，然后按下鼠标左键，不放，拖动鼠标在波形显示区移动，当到了合适的位置时，放开左键，则一次绘制过程完成。鼠标可以沿任意方向移动，可以多次绘制，直到

满意为止，这时按下界面上的“绘制结束”按钮，则一个新波形产生。接下来对该波形取波形名，然后可以对之进行其它编辑操作，或直接将该波形数据发送给硬件模块以产生一个与用户手动绘制相同的任意波形。

图3-5是手动绘制的一个任意波形。

图3-6即是手动绘制波形的程序流程图。

用鼠标绘制波形时，将波形的坐标区作为绘图区，实现此功能最重要的是使用了 LabWindows/CVI 中的定时控件 Timer，此控件在编辑面板时可见，当程序运行时不可见。编程时先设定一具体时间间隔进行调用此函数，该软件设定了 2mS 作为时间间隔。手动绘制过程中，按下鼠标左键不放，在坐标区沿任意方向拖动鼠标，与此同时，该定时函数每 2mS 被触发一次，每执行一次该函数采集一个离散点，即获取了当前光标处点的横纵坐标，这样就得到一系列的离散点，相临离散点之间的间隔是不等的，这与用户拖动鼠标的速度有关。于是还有许多波形点没有赋值，这样就要在相临的离散点之间插值计算出那些空缺点的幅值。考虑到每两个相临点间的距离比较小（因为设定的 Timer 函数间隔较短），于是可以采用线形插值，插值计算出来的波形是比较光滑的。

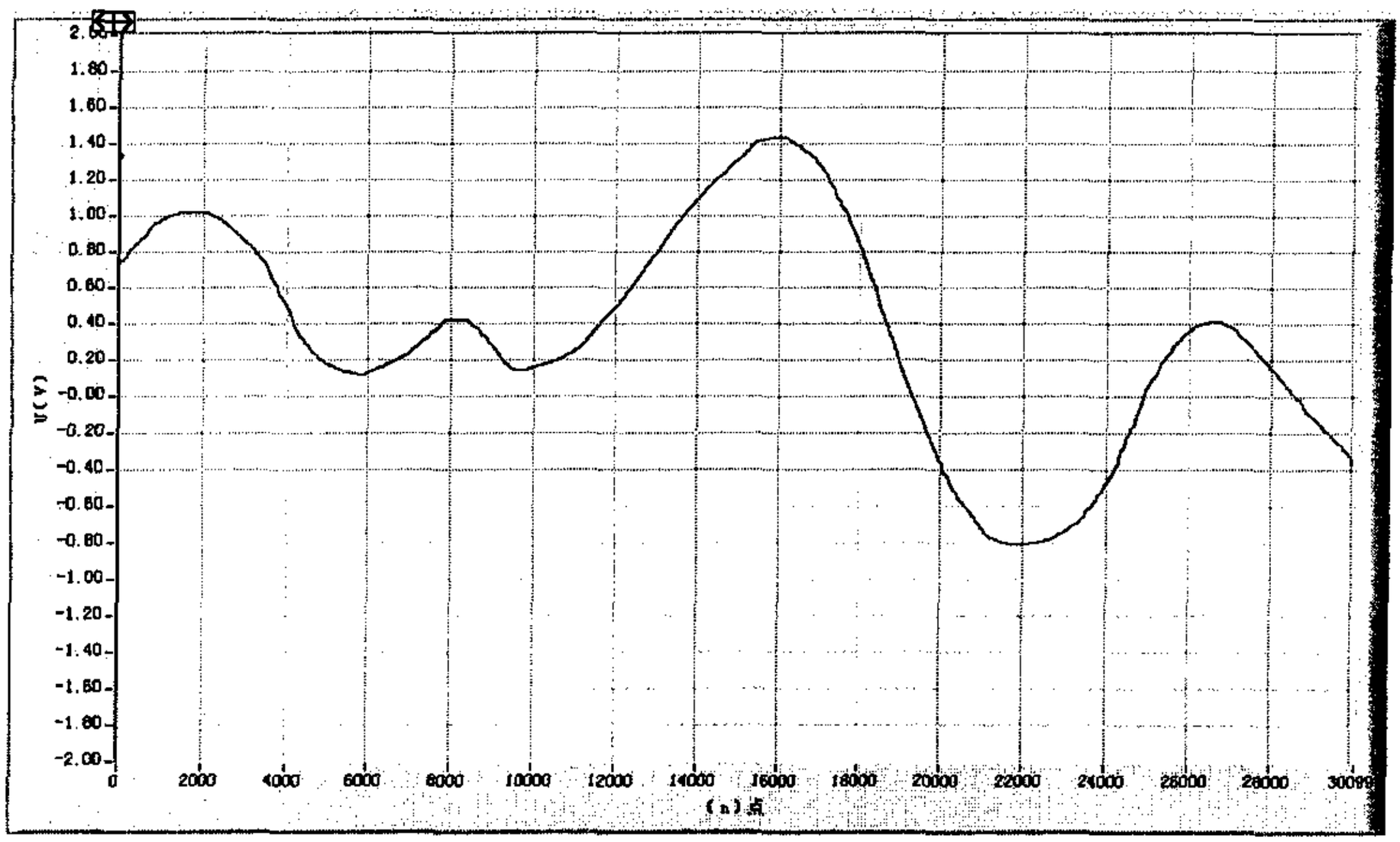

图 3-5　手动绘制的一个任意波形

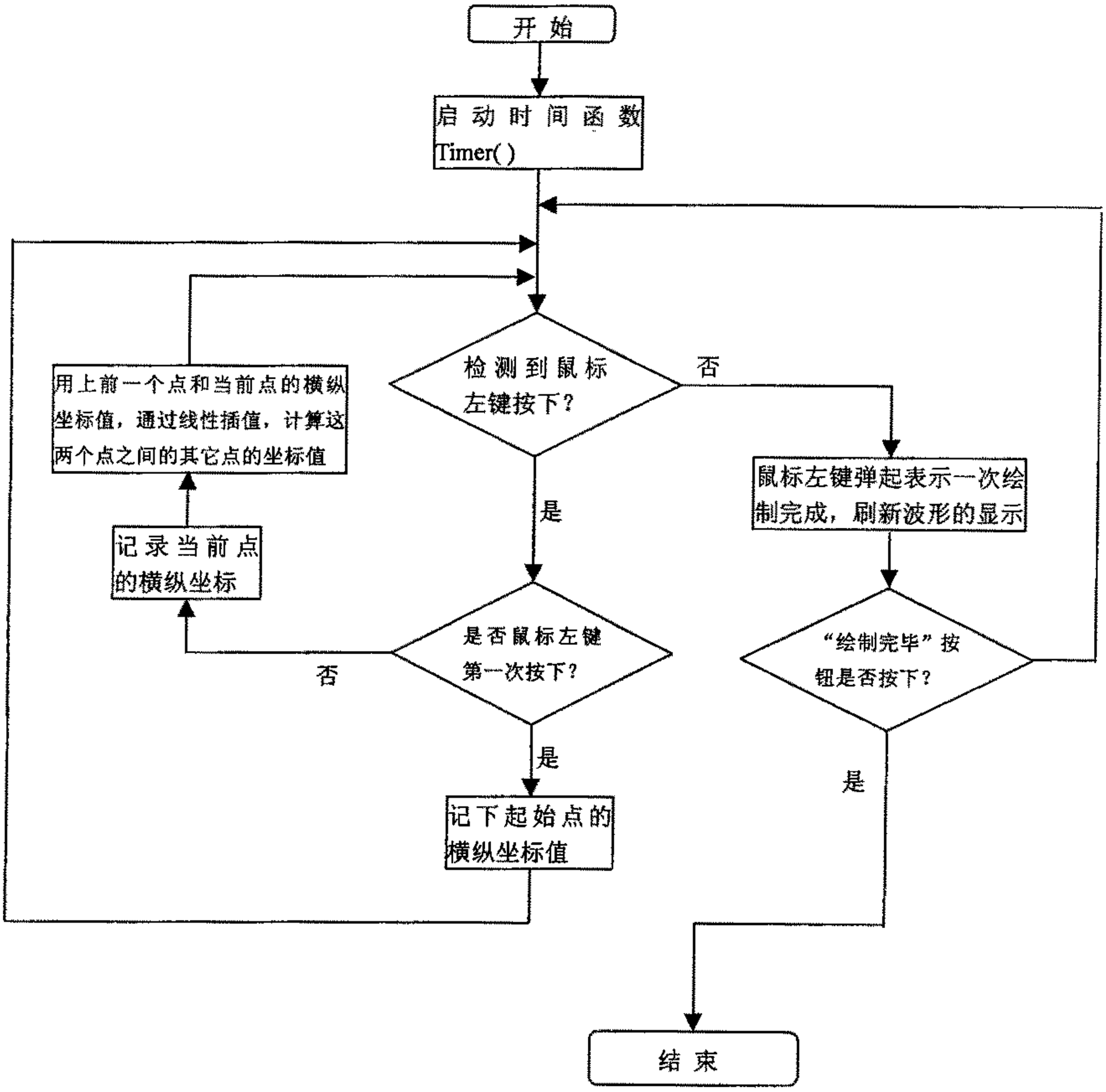

图 3-6　手动绘制波形流程图

从流程图可以看出，采用Timer控件和线性插值是实现这一功能的两个关键技术。即通过定时的循环执行该控件的回调函数以获取许多离散点的坐标值，再通过线性插值计算每两个离散点中间的各点的坐标值。

手动绘制波形不但可以产生一个新的波形，还可以编辑修改已存在的波形。例如，可以手动在原波形上拉出一个尖峰；手动将原波形中不需要的陡峭处抹平；手动绘制一个形状特殊，无法用数学方程式表达的波形。总之，手动绘制的方法简单、直观，大大丰富了波形编辑的处理能力。

以下截取的程序段 Timer 控件的回调函数 Timer_Draw( )，通过获取鼠标左键按下过程中得到的各个离散点的横纵坐标值，然后在每两个点之间进行线性插值，最终得到整个波形所有点的坐标值。在每次进入函数 Timer_Draw( ) 时，先得到当前坐标值nowX，nowY，然后根据前一次进入函数获取的 oldX，oldY，先调用库函数 PlotRectangle( ) 将两点之间的波形清除，然后在此两点之间连线，需要说明的是此时的连线只是给用户一个视觉上的现象，即每两个相邻离散点之间是线性插值，而实际的值需要在随后的程序中通过计算来获得。最后将最新的纵坐标值 nowY 放入数组_waveY_hand[]中来更新原来的波形数据。

```
if (MouseIsInCtrl (mainPanel, PANEL_GRAPH, mouseX, mouseY))
  {
    ......
    GetGraphCursor (mainPanel,  PANEL_GRAPH,  3,  &nowX,  &nowY);
    ......
    PlotRectangle (mainPanel, PANEL_GRAPH, oldX, _yMax,  nowX+1, _yMin,
                _interface.ground_color,  _interface.ground_color);
    PlotLine (mainPanel, PANEL_GRAPH, oldX, oldY, nowX, nowY,
            _interface.wave_color);
    _waveY_hand[i] = nowY;
    interval_num  = abs(nowX - oldX)  - 1
    if (nowX >= oldX)
      {
        for (k = 0; k < interval_num; k ++)
            {
             _waveY_hand[oldX + k + 1] =
            (_waveY_hand[i] - _waveY_hand[j]) / (nowX - oldX) * (k + 1)
           + _waveY_hand[j];
          }
      }
    else
      {
        for (k = 0; k < interval_num; k ++)
          {
             _waveY_hand[abs(oldX - k - 1)] =
                (_waveY_hand[j]_waveY_hand[i]) / (oldX - nowX) *
       (interval_num
                -k- 1) + _waveY_hand[i];
            }
       }
     ......
}
```

3.4 从波形库中提取各种波形

波形库中的信号有十几类，其中标准信号占大多数，也有部分CVI软件中已有的波形函数，如噪声。这些信号的幅值、周期数可以通过各参数控件进行修改，而一些信号的特殊参数，如矩形的占空比、脉冲的脉冲宽度等，只针对特定波形才有用，即只有当这类波形被选中的时候，这些特殊参数才处于激活状态。以下是波形库中各类信号的编程模型。

- 正弦波

正弦信号在测试领域的应用十分广泛，例如，电子放大器增益的测量、相位差的测量、以及系统频域的测量等都要用到正弦信号。还有其它信号都可以看作是基频的正弦信号与高次谐波的叠加。正弦信号是测试领域中最基本的信号。

设 T 为信号的周期数，每个周期波形的点数为 n，载入该信号的任意波形发生器硬件缓冲区的波形点数是固定的65536个，即 64k 个点。

65536 = n * T，　其中 n 为整数

x(i) = A*sin(2*pi*i/n)　　i = 0,1,2,3……n

对于正弦波的产生可以直接代公式计算，也可以使用CVI提供的库函数来实现，函数代码为 SineWave ( )。

- 三角波

三角波也是测试领域里非常有用的一种波形。它的模型如下：

x(i) = A*(2*i/n)　　i = 0,1,2,3,……n/2

x(i) = 2*A*(1-i/n)　　i = (n/2)+1,(n/2)+2,……,n

同正弦波一样，它的产生也可以使用库函数 TriangleWave() 来实现。

- 方波

方波很多时候用作逻辑电路里的时钟信号。模型如下：

x(i) = A　　i = 0,1,2,3,……,n/2

x(i) = -A　i = (n/2)+1,(n/2)+2,……,n

方波的产生也可以使用库函数 SquareWave( ) 来实现。

- 锯齿波

在示波器、电视、雷达等电子设备中，显示屏上出现的各种图象是由电子运动形成的。为了让电子按照一定规律运动，常用到锯齿波作为时基信号。例如，要在示波器的显示屏上不失真的观察到被测波形，就要在水平偏转板上加锯齿波信号，使电子束沿着水平方向均匀扫过显示屏。锯齿波的模型如下：

$x(i) = A*i/n \qquad i = 0,1,2,3,\ldots\ldots,n$

调用的库函数为 SawtoothWave( )。

- 直流信号

$x(i) = A \qquad i = 0,1,2,3,\ldots\ldots,n$

- 指数波形（上升）

在RC电路中，开关闭合或打开的瞬间，电容电阻两端的电压，或电路的电流都是呈指数形式变化的。

$x(i) = A*\exp((i-n)/T)) \quad i = 0,1,2,3,\ldots\ldots,n$ T为时间常数

- 指数波形（下降）

$x(i) = A*\exp(-i/T) \qquad i = 0,1,2,3,\ldots\ldots,n$

利用指数波形可在计算机上模拟RC电路、RL电路的状态响应实验。利用指数波形和正弦波形的组合可模拟RLC电路的状态响应实验。利用计算机模拟，减少了对实际实验设备的需求。例如，在实物实验时需要用示波器观察电压的变化，而在模拟实验中，电压的变化显示在计算机显示屏饿波形显示区中，并可做到与实物示波器显示出相同的效果。

- 白噪声

随机过程按它的功率谱密度函数的形状来进行分类，可分为白噪声和有色噪声两大类。具有均匀功率谱的白噪声是一种普通存在的最为重要的噪声。一个均值为0，功率谱密度在整个频率轴上有非零常数，即 $\Phi(f) = N_0/2$，N_0为噪声的单边谱密度。

在CVI的 Advance Analysis 库中有生成白噪声数据的函数，可直接调用。函数代码为：WhiteNoise(n, Amplitude, seed, data)。其中 n 为生成波形的点数；Amplitude 为波形幅值； seed 取值小于 0 时，若是第二次调用该函数则产生的波形数据与第一次调用的不同，若为大于等于 0 时，则产生的波形数据与第一次调用的相同；data 为存放波形数据数组的首地址。

- 统一噪声

可调用CVI中的函数 Uniform(n, seed, data)。其中的参数类似白噪声信号函数的参数定义。

- 高斯噪声

可调用CVI中的库函数 GaussNoise(n, deviation, seed, data)。其中 deviation 是高斯噪声的标准差；其它参数定义同上。

- 阶梯波

阶梯波在测量和控制设备中应用较多，可作为时序控制信号和多级电位基准信号。例如，在晶体管的特性测量中，就需要这种信号，半导体三极管输入阶梯波基极电流，就可测出在不同基极电流的输出特性。

设阶数为T，则每一阶的波形点数为 $n = 65536/T$。

第一阶：x(i) = A/T　　i = 1,2,3,……,n

第二阶：x(i) = 2*A/T　　i = n+1, n+2, n+3,……,2n

……

第T阶：　x(i) = A　　i = Tn+1-n, Tn+2-n, Tn+3-n,……,65535

- 梯形波

设 T 为该波形的周期数，每个周期波形的点数为 n，梯形信号为等腰梯形，上底为下底的1/3。第一个周期波形的数学模型为:

上升段 x(i) = A*i*3/n　　i = 1,2,3,……,n

平直段 x(i) = A　　i = [n/3]+1,……,[2*n/3]

下降段 x(i) = 3A*(1-i/n)　　i = [2*n/3]+1,……,n

- sin(x)/x

x = 0 时，作为被除数，可考虑其为一较小的数。

x(i) = A*sin(2*pi*i/n)/j　　当i = 0时，　j=0.000001；其余 i = 1,2,…,n时，i=j

3.5 输入特征点插值产生波形信号

输入特征点，然后插值产生波形的方法是手动绘制波形方法的一种有利补充。使用鼠标直接绘制波形的功能操作简便、迅速，可以在大体上绘制所需的波形形状，但很难对波形的幅值精度等保持较高的水平。而采用输入特征点再插值产生波形的方法就可比较准确的控制一些用户所需的特征波形的部分。即该方法产生波形分两个步骤进行。

步骤一：输入特征点坐标值

即输入每个点的横纵坐标值，有两种方法可以采用：一是用鼠标打点；二是在数值框中输入坐标值。下面对这两种输入特征点的方法分别叙述。

(1) 鼠标打点确定特征点的坐标值

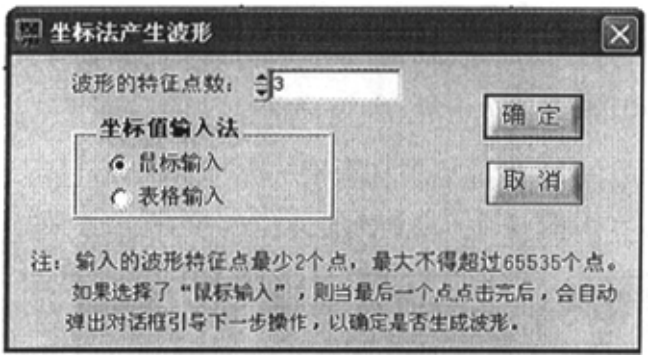

图 3-7 特征点输入法的选择对话框

(2) 表格法输入特征点坐标值

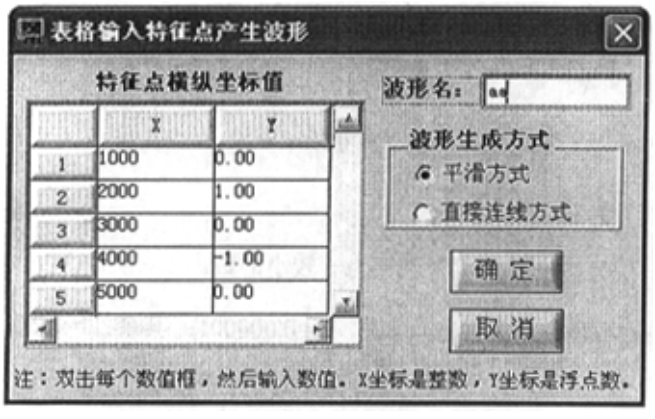

图 3-8 表格法输入坐标值对话框

在图3-7所示对话框的“坐标值输入法”的组合控件，若选择是“鼠标输入”，

则接下来可以用鼠标左键直接打点，程序将记录每个点的横纵坐标值；若选择的是“表格输入”，则会弹出如图3-8所示的对话框，在给出的表格里填上各个点的坐标值，这些坐标值也将分别记录在横坐标和纵坐标数组中。

步骤二：插值计算其余点的横纵坐标值

插值的算法有两种：一是采用线性插值，这种方法生成的波形就是一段段的直线段相连起来；二是采用曲线拟合插值，这种方法生成的波形比较光滑。下面对这两种插值方法分别叙述。

(1) 线性插值

采用线性插值，即是用特征点数组中每两个相邻点的横纵坐标值，根据线性比例的关系计算每两个特征点之间的其它点的坐标值。例如，设两个相邻特征点的坐标分别为(X_1，Y_1)、(X_2，Y_2)，要求的点的坐标为(X_i，Y_i)

根据 $(Y_i - Y_1)/(X_i - X_1) = (Y_2 - Y_1)/(X_2 - X_1)$ 可以求得 Y_i 的值。

其中 X_i 是介于 X_1 和 X_2 之间的数值。

图 3-9(a) 是用线性插值得到的波形。

a。线性插值法　　b。曲线拟合插值法

图 3-9　输入特征点插值产生的波形

(2) 曲线拟合插值

采用曲线拟合插值可以参考调用CVI已有的函数 SpInterp( )，该函数利用已有点的坐标值，根据算法计算其余点的坐标值。SpInterp( ) 的数学模型为：Y = A*Y[i] + B*Y[i] + C*Y"[i] + D*Y"[i]，其中 Y" 是波形数据点的二次导数；

A = (X[i+1]-X)/(X[i+1]-X[i])　　B = 1-A

C = $(A^3-A)*(X[i+1]-X[i])^2/6$　　D = $(B^3-B)*(X[i+1]-X[i])^2/6$

由以上公式可知，要计算出最后结果，只有特征点的横纵坐标值是不够的，还要有纵坐标的二次导数值。于是调用 Advance Analysis库中的 Spline( ) 函数来获取特征点的二次导数数组。由此可见，通过特征点插值产生波形的实现最关键是使用了 Spline( ) 和 SpInterp( ) 两个库函数。

3.6 波形数据文件导入生成波形

图3-10所示为波形数据文件的导入流程图。

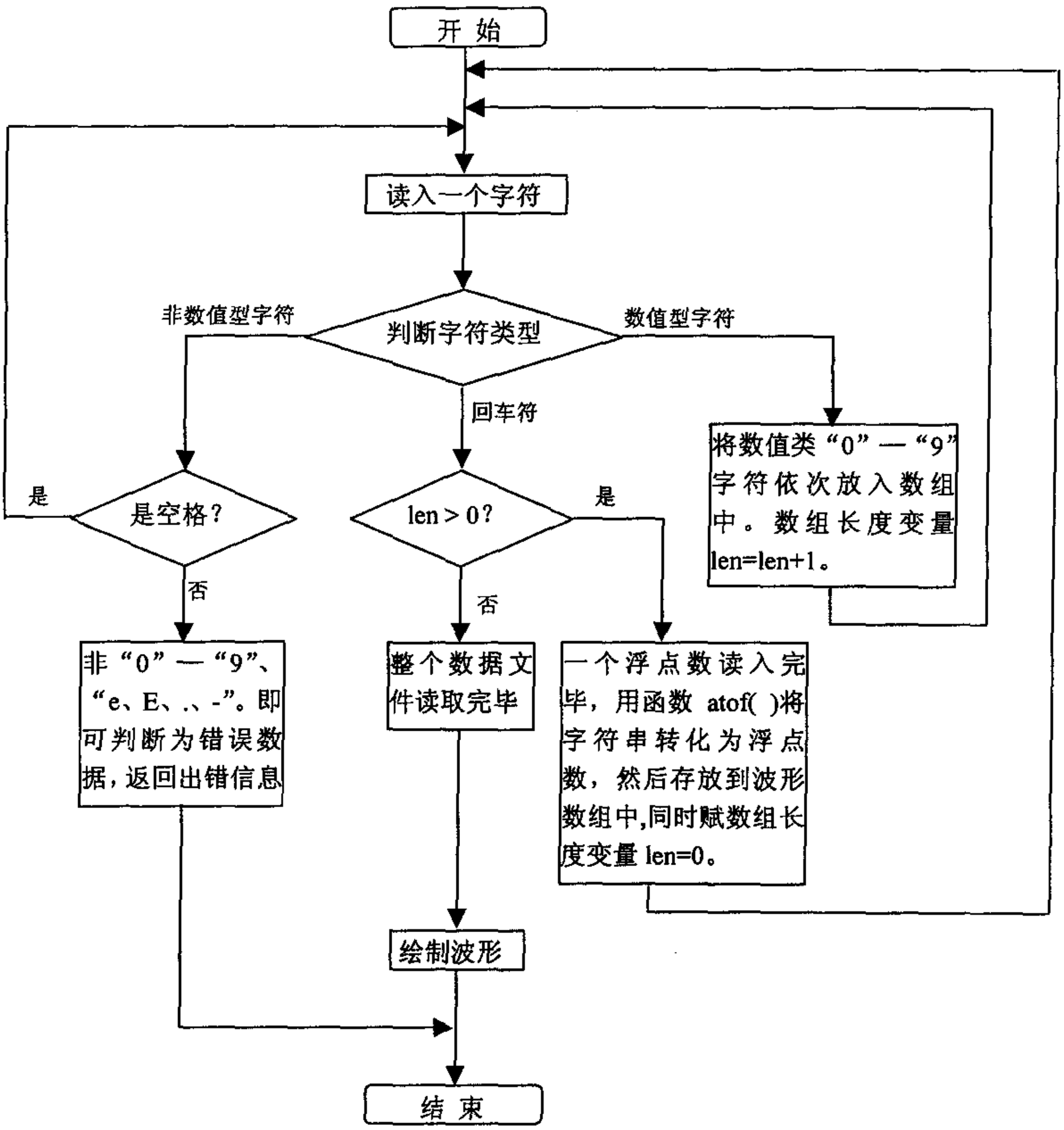

图 3-10 导入波形数据文件生成波形的流程图

这个功能的使用即是将用数字示波器获取的波形信号存成的数据文件，以浮点数的形式导入到该波形编辑器中来。或者由该软件创建的波形信号也可保存成文件，下次使用时直接导入，就不必再进行编辑了。波形的导入类似于文件的打开。

需要注明的是，只有存在的文件才能被导入，文件的后缀名为“.txt”，即文本文件的格式，并且数据文件的保存格式是一行一个数值，数值结束是回车符，下一行接着写下一个数值。

从上图所示的流程图可以看出，导入波形文件生成波形的过程中有几点是必不可少的。第一，文件格式要符合本软件规定的格式，即文本文件格式，每行一个数据，回车换行写下一个数据。第二，读到数值型字符将之放到字符串缓冲区，同时字符串长度变量加一，待读到回车符即表示该字符串结束，随后将其转化为浮点数。第三，读到非数值型字符时，若是空格则忽略，接着读下一个字符；若不是空格则表示数据文件有误，停止读取，返回错误信息。第四，读到回车符，若字符串长度变量非零，即是一个数据字符串的结束；若字符串长度变量为零，即是整个数据文件的结束。

下面是波形数据导入的一部分流程图。

……

```
do{
    if (ReadFile (_readFileHandle,  &buffer, 1) == -1)
       {
         read_ERROR = TRUE;
         break;
       }
    i ++;
    if (data_num > MAX_POINT)
         break;
    if ((buffer >= 48 && buffer <= 57) || buffer == '.' || buffer
        == '-' || buffer == 'e' || buffer == 'E')
      {
        num_buffer[len] = buffer;
        len ++;
      }
    else if (buffer == '\n')
      {
        if (len == 0)
          {
            read_OVER = TRUE;
            break;
```

```
        }
        num_buffer[len] = '\0';
        Y[data_num] = atof(num_buffer);
        data_num ++;
        len = 0;
      }
   else
     continue;
}while(i < fileSize);
.....
```

3.7 本章小节

本章介绍的几种波形产生的方式——数学方程式产生、手动绘制、波形库提取、输入特征点插值产生、数据文件导入产生，您可以根据需要或习惯任意选择其中的某种或某几种来获得您希望的波形，各种方法都有它自己的特点及优缺点。数学方程式产生方式可以获得许多与数学直接相关的波形，波形种类繁多，缺点是波形数据在计算过程中速度比其它方式要慢一些；手动绘制方式是最直观的一种生成波形的方式，用户直接操作鼠标进行绘制，缺点是绘制的某些点不够精确；输入特征点插值产生波形可以弥补手动绘制的不精确性，将比较关键的某些点确定好，由计算机自动插值计算出其它点的坐标值，这种方法快捷、简单；数据文件导入产生波形实现的是与其它诸如数字存储示波器的结合应用，由这些仪器采集得到的波形数据直接导入计算机，然后由任意波形发生器复原产生出相同的波形信号，以供以后的实验中使用。

第四章　波形的编辑与处理

本章简介：软件设计的波形编辑的功能丰富多样，集中体现了波形编辑器的特色——“任意”性。有波形序列的组合，将不同的信号组合连接起来；有对整个波形或某波形段进行横向的拉伸、压缩；有对波形显示效果的变化，如对波形某一部分进行放大、缩小等。下面就对以上提到的各种编辑处理方式详细介绍。

4.1　波形序列的组合

以上介绍过，有些测试中，可能需要测试的信号是几种不同信号连接而成的。例如，正弦波后连着方波、其后又是三角波，最后的波形为调幅波等。图 4-1 即是用波形序列组合处理后的任意波形，从图中可看出，该波形由好多中波形信号连接而成。

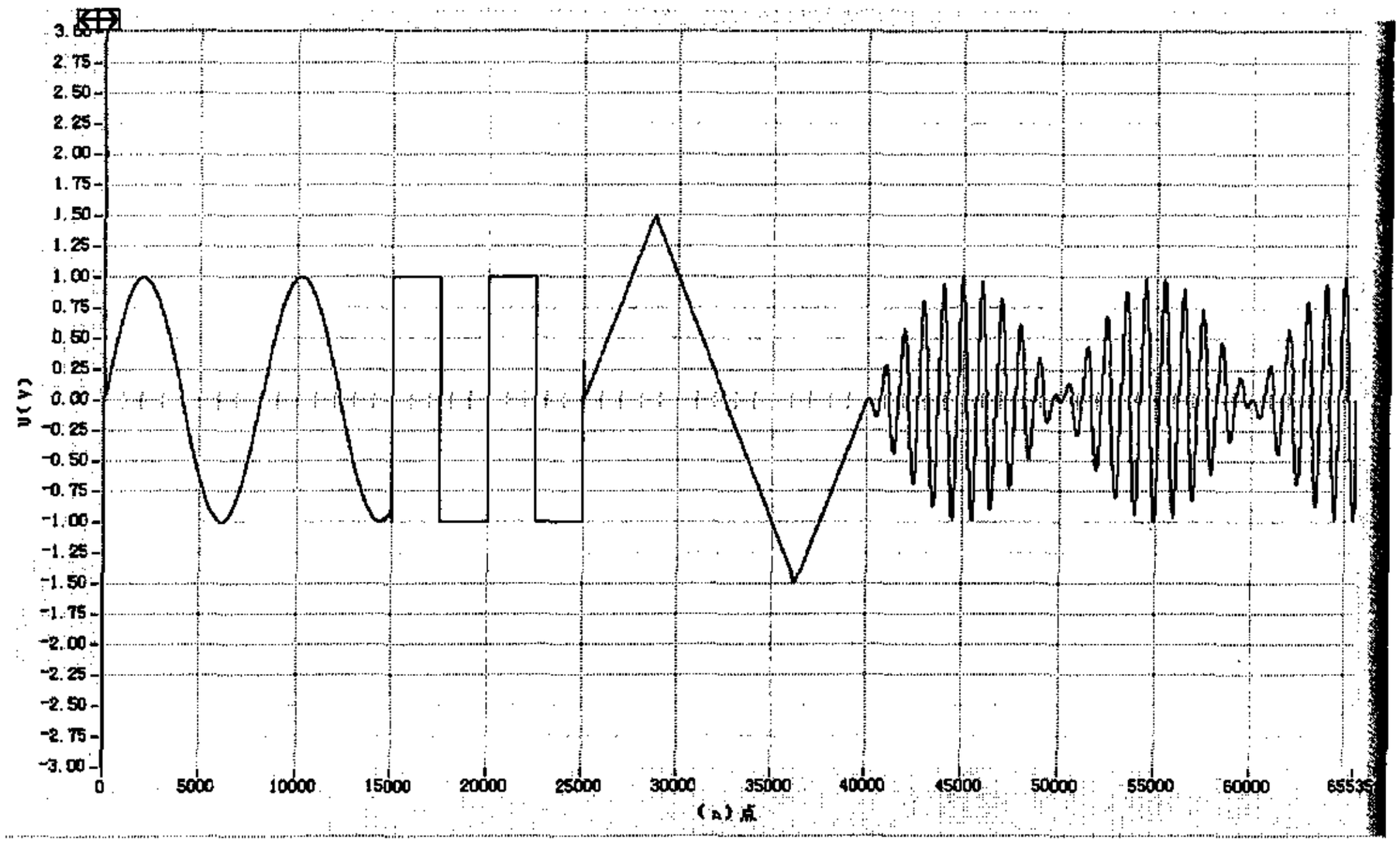

图 4-1　波形序列组合生成的任意波形

要实现波形序列组合生成任意波形，必不可少的几个编辑功能是：波形的复制、减切、粘贴、替代、删除。即可以先准备几个波形，如正弦波、三角波、方波、锯齿波等，然后复制三角波的某一段波形，替代到正弦波的某处，再复制或减切方波的某一段波形，粘贴到正弦波的末尾，再复制一段锯齿波，粘贴到刚才粘贴的方波段后面。

对于复制或减切操作，需给出一个缓冲区，用来存放被复制或者被减切的波形段数据；对于粘贴或替代操作，直接从该缓冲区读取数据，添加到当前被编辑的波形指定区域。波形的指定区域由波形显示区的两根左滑标和右滑标来限定。

4.2 波形的拉伸或压缩（插值处理）

第二章里，在讲述把波形编辑器编辑的波形数据发送给硬件模块时，提到当编辑的波形数据大于或小于 64k 个数据时，由于硬件提供的是固定的 64k*12bit 存储空间，于是发送之前必须将波形数据处理为 64k 个数据。这样硬件存储区里存放的才是一个完整周期的波形数据。

波形拉伸或压缩的设计思想是一样的，关键使用了 CVI 的 Advance Analysis 库中的 Spline( )、SpInterp( )两个函数，以及使用了求最小公被数的思想，即将原始波形先插值拉伸到最小公被数大小的长度，然后再等间隔抽取所需要的最终点数。下面举例，并对照流程图说明波形拉伸或压缩的设计思路。

设现在有一个起始点为 0，终止点为 99 的正弦波，现想将波形拉伸至终止点为 65535，即将一个原来长度为 N_1=100 个点的波形扩展到长度为 N_2=512。设数组 Y_1[100] 保存原始波形各点的纵坐标值，Y_2[512] 保存拉伸后的波形各点纵坐标值，数组 X[100] 保存将原始波形展开后的各点新的横坐标值，Y"[100] 保存原始波形幅值的二次导数。程序按以下步骤执行。

<i> 对于还没有进行处理的波形，每个点的横坐标依次是 X[0]=0，X[1]=1，X[2]=2，…，X[99]=99，每个点的纵坐标存放在 Y_1[100] 中。

<ii> 求原始点数 N_1 和最终点数 N_2 的积 $N_3=N_1*N_2$=51200。先将原始波形的各点按等间隔 511 拉伸。这时可以比喻为在 51200 整个范围内，每隔 511 个点有一个已知点，根据这些已知点的坐标值可以插值求其它点。拉伸后各点的横坐标值为 X[0]=0，X[1]=512，X[2]=1024，…，X[99]=50688。

<iii> 将以上各数组带入库函数 Spline(X, Y_1, N_1, 0.0, 0.0, Y")，其中 X、Y_1、N_1 是输入参数，Y"是输出。

<iv> 插值算法本来是，用已知的 100 个点的横纵坐标值插值计算，每两个点之间要插值计算 511 个点，最后一个点后面还要插值计算 511 个点。但这样的思路将浪费许多不必要的计算，因为整个波形插值计算出来将有 51200 个点，而实际只需要 51200 个点中的 512 个点，其它的 50688 个点的计算就是白费的。因此，在插值计算时，就只调需要的点进行计算，不需要的点则略过。于是调用函数 SpInterp( )，每隔 99 个点，插值计算一个点的 Y 值，这样，循环 512 次，即求出了 512 个点的 Y 值。

图 4-2 是用插值法拉伸波形的示意图。图 4-3 和 4-4 分别是某波形拉伸前和拉伸后的波形图。从图中可看出，使用该算法拉伸的波形形状不会失真。

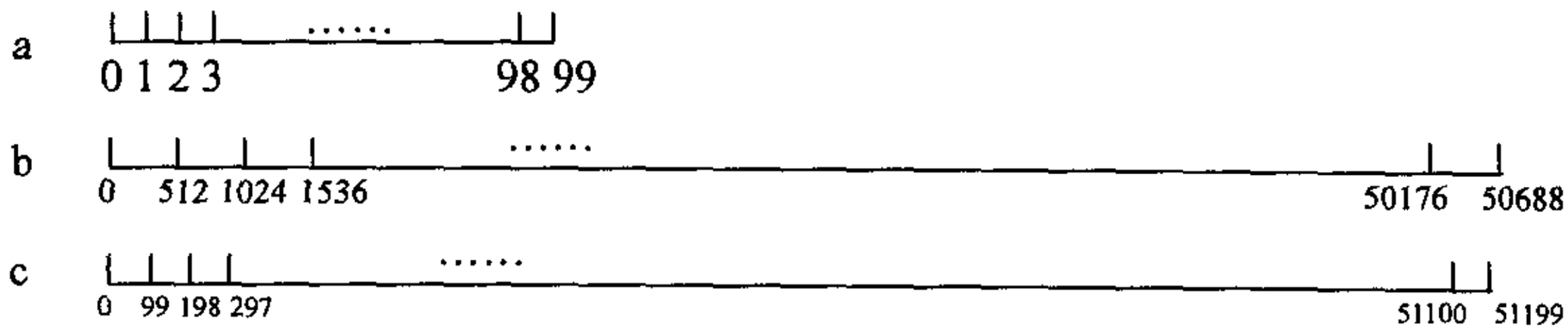

图 4-2　波形拉伸或压缩设计示意图

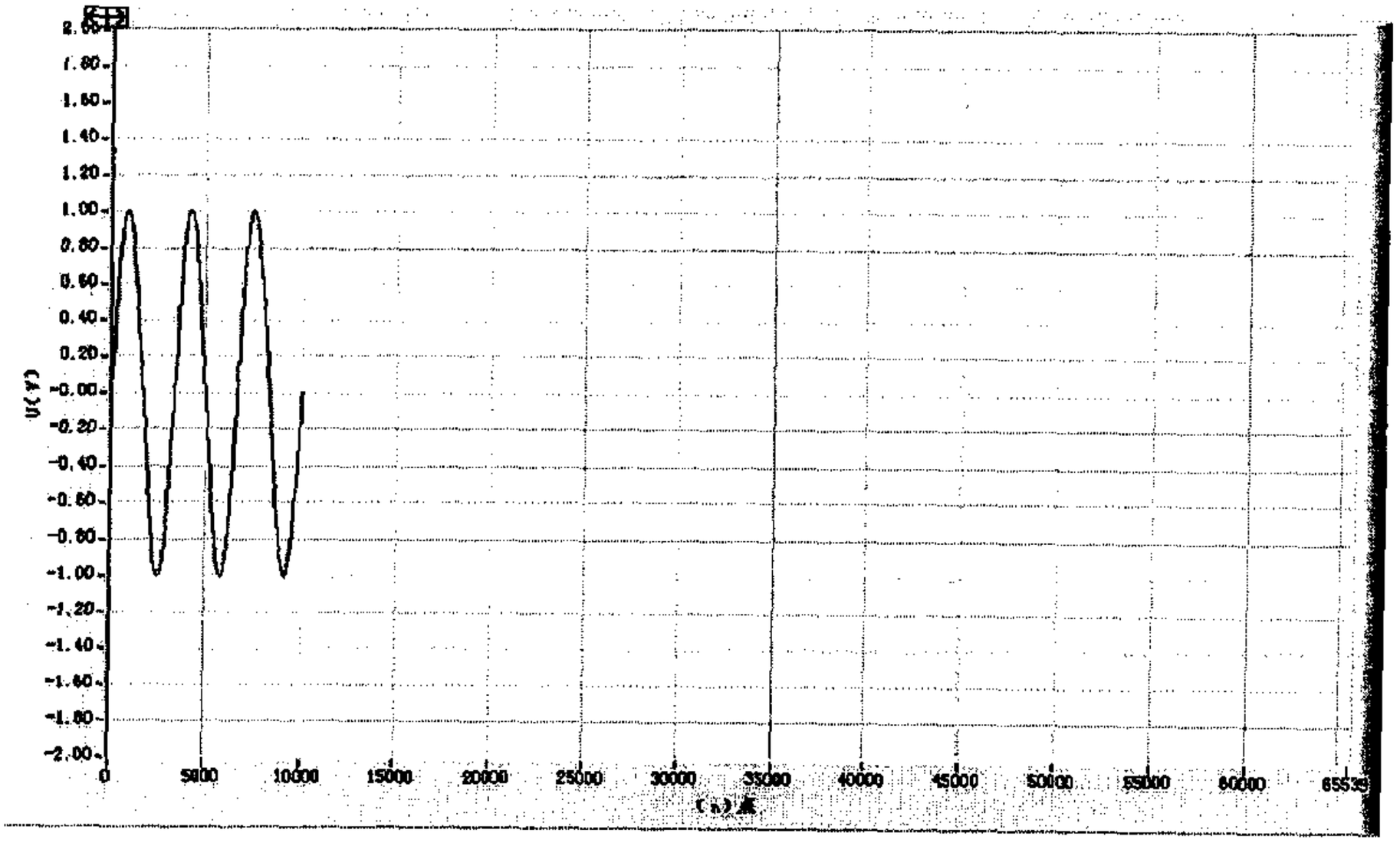

图 4-3 长度为 10000 个点的任意波形

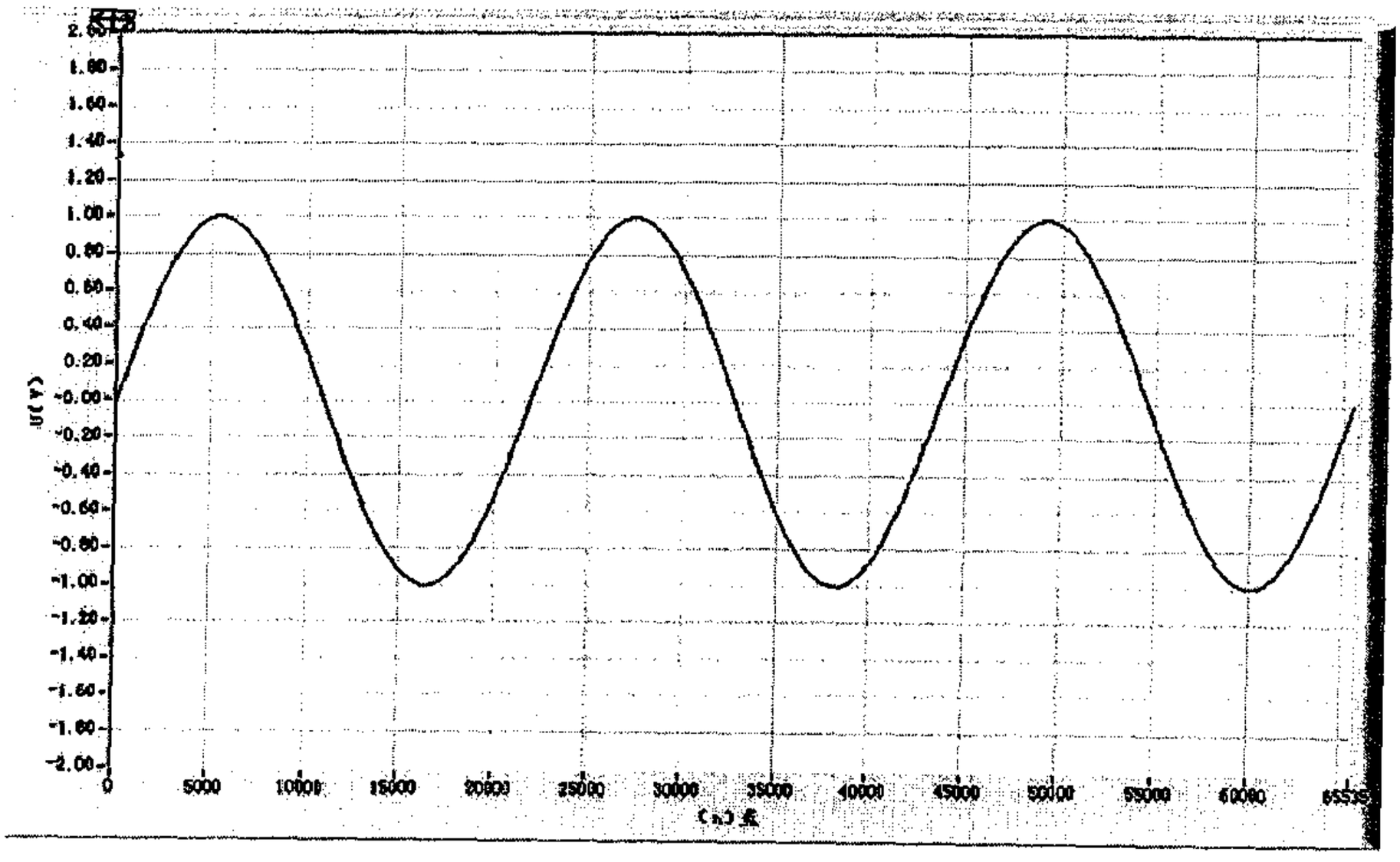

图 4-4　将上图所示波形拉伸为长度 64k 的波形

下面是波形进行拉伸的程序段。old_len 和 new_len 分别是原始波形 A 和即将扩展的波形 B 的长度值。确定好原始波形和目标波形之后，现在来构造一个虚拟波形 C，该虚拟波形的长度为 old_len*new_len ，即原始波形 A 的长度 old_len 拉伸到了它本身的 new_len 倍。（之所以称 C 为虚拟波形，是因为在后面程序并不计算此波形每一个点的坐标值，而只是以它为基础，计算它中间符合要求的一部分点的坐标值即可。）old_X[]、old_Y[] 分别保存原始波形 A 变到长度为虚拟波形 C 时原波形各点现在的坐标值。 dao_Y[] 保存的是波形各点的二阶导数值，通过函数 Spline( ) 获得，即该函数的最后一个参数 dao_Y 为函数的输出。然后，循环 new_len-1 次，每次先计算目标波形 B 的各点现在在虚拟波形 C 中的各点的横坐标值，然后将横坐标值代入函数 SpInterp( )，即可计算目标波形各点的纵坐标值 new_Y[]。最后一条语句是将波形 A 的终止点的纵坐标值复制为波形 B 的终止点的坐标值，因为终止点和起始点一样，是不需要插值获取的，原始波形和目标波形起始终止点的纵坐标值是相同的。至此，整个插值过程结束，数组 new_Y[] 中保存的即是我们需要的拉伸后的波形 B 的各个点的纵坐标值。

```
.....
old_len = wave[_nowWaveNum].samples;
old_X = malloc(old_len * sizeof(double));
old_Y = malloc(old_len * sizeof(double));
dao_Y = malloc(old_len * sizeof(double));
new_Y = malloc(new_len * sizeof(double));
for(i = 0; i < old_len; i++)
  {
    j = i * (new_len - 1);
    old_X[i] = j;
    old_Y[i] = wave[_nowWaveNum].Y[i];
  }
Spline (old_X, old_Y, old_len, 0.0, 0.0,  dao_Y);
for(i = 0; i < (new_len-1); i++)
  {
    x_interval = i * (old_len - 1);
    SpInterp(old_X, old_Y, dao_Y, old_len, x_interval, &new_Y[i]);
  }
new_Y[new_len-1] = old_Y[old_len-1];
.....
```

4.3 算术处理

该软件提供了许多种波形编辑的功能。包括波形的算术处理、加窗处理、滤波处理、平滑处理。下面将详细介绍各种波形处理方法的实现及理论。

波形的算术处理即是指将某波形加、减、乘、除某常量，或者将两个波形进行相加、相减、相乘、相除的操作。

需要注意的就是，某波形除以一个常量时，除数不能是 0。两波形相除时，例如波形 A 除以波形 B，即是波形 A 的每一点的 Y_A 值除以波形 B 的每一点的 Y_B 值，数组 Y_B 中可能会有 0 值，碰到 0 时就将 0 修改为一个很小的数值，如修改为 Y_B 最大值 1/1000，这样就可以避免计算过程中的语法错误。

4.4 加窗处理

信号分析中，几乎都要求信号是有限长度的，因而，对于连续信号就要进行截取处理，这就要用到窗函数。窗函数作用于信号的过程可以用下面的式子表示：

y(t) = x(t) * w(t)，其中 y(t)是信号 x(t) 加窗之后的波形，是个有限长度的信号，w(t)是窗函数。

无论信号是时限的还是非时限的，加窗对信号离散后的分析必然是有影响的。加窗在时域内相当于原信号乘以窗函数，对应的在频域内则是原信号的频谱与窗函数的频谱卷积，这必然使原信号的频谱失真。

理论研究总结出改善窗口函数形状的标准是以下两点：

(1) 尽量减少窗口函数频谱中的旁瓣，也即，使能量尽量集中在主瓣中，这样就可以减少肩峰和余振，提高阻带的衰减。

(2) 主瓣的宽度尽量窄，以获得较陡的过渡带。

然而，主瓣过宽会影响信号频谱的分辨率，旁瓣太大，会有严重的频谱泄露，但往往主瓣窄的窗函数的旁瓣大，两种要求相互矛盾，因而要权衡利弊，选择合适的窗函数。

LabWindows/CVI 中提供了多种窗函数，最常用的几种窗函数，如海宁窗(Hanning)、海明窗(Hamming)、布拉克曼窗(Blackman)、凯塞窗(Kaiser)等，都是直接作用于信号，得到处理后的结果。

本软件提供了海明窗和布拉克曼窗两种窗函数对信号进行加窗处理。

(1) 海明窗

函数定义为：int status = HamWin(double x[], int n)

其中，在输入时，x[]用来存储输入信号的各点幅值；输出时，x[]用来存储加窗后信号幅值。n 表示数组元素个数。

函数中，海明窗的定义是：w[i] = 0.54-0.46*cos(2*pi*i/n)　　i = 0,1,2,…,n-1

函数 HamWin( ) 的输出是输入信号和海明窗共同计算的结果，计算公式如下：x[i] = x[i] * w[i]　　i = 0,1,2,…n-1

(2) 布拉克曼窗

函数定义为：int status = BkmanWin(double x[], int n)，参数定义如上。

布拉克曼窗的定义为：w[i] = 0.42-0.5*cos(2*pi*i/n)+0.08*cos(4*pi*i/n)

i = 0,1,2,…,n-1

同上，函数 BkmanWin( )的输出是输入信号和布拉克曼窗共同计算的结果，计算公式同上。

图 4-5 所示为信号加窗处理的对话框。

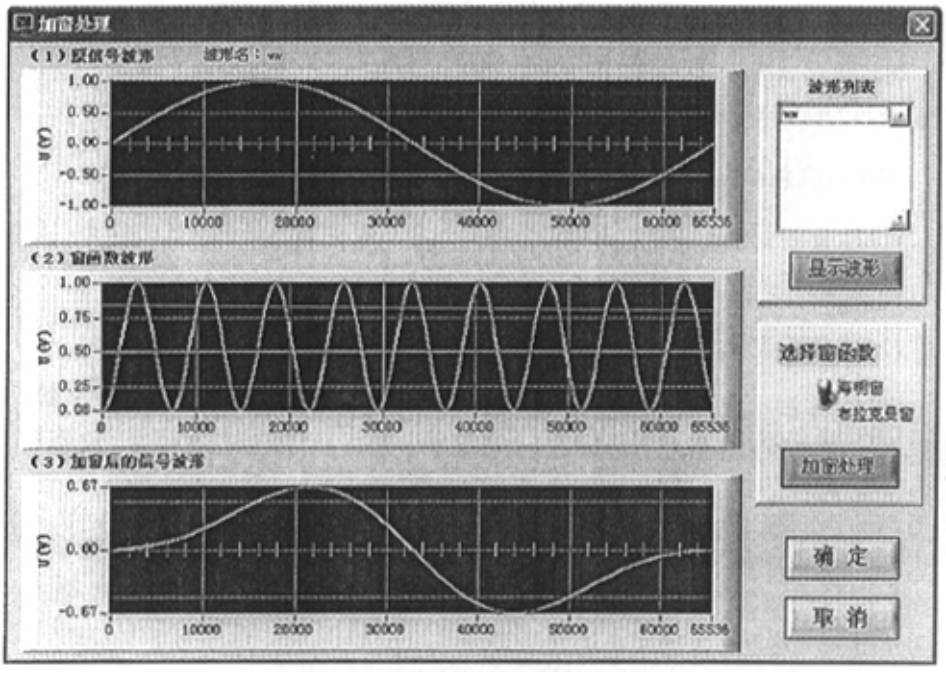

图 4-5 加窗处理的对话框

图中的第一个波形是原始波形；第二个波形是海明窗或布拉克曼窗的波形；第三个波形是原始波形加窗处理后的波形。

下面是加窗处理的一部分程序。

```
if (event == EVENT_COMMIT)
{
  //选择需要进行加窗处理的某一个波形 select_wave
  GetCtrlIndex(addWindow_panel, WINDOW_PAN_WFM_LISTBOX, &select_wave);
  //选择加窗的类型：海明窗或布拉克曼窗
  GetCtrlVal (addWindow_panel, WINDOW_PAN_BINARYSWITCH, &window_type);
  num = wave[select_wave].samples;
  final_Y = malloc(num * sizeof(double));
  win_Y   = malloc(num * sizeof(double));
  free(add_win_Y);
  add_win_Y = malloc(num * sizeof(double));
  CopyID(wave[select_wave].Y, num, final_Y);
  switch(window_type)
        {
         case 1://海明窗
                HamWin (final_Y, num);
                w[i]=0.54-0.46*cos(2*PAI*i/n)
                for(i = 0; i < num; i++)
                    win_Y[i] = 0.54 - 0.46 * cos(2.0 * PAI * i / num);
                final_min = final_Y[0];
                final_max = final_Y[0];
                win_min = win_Y[0];
                win_max = win_Y[0];
                break;
         case 0://布拉克曼窗
                BkmanWin (final_Y, num);
                for(i = 0; i < num; i++)
                win_Y[i] = 0.42 - 0.5 * cos(2.0 * PAI * i / num) + 0.08
                           * cos(4 * PAI * i / num);
                break;
      }
   ......
}
```

4.5 滤波处理

使用滤波器对信号进行滤波，可以得到想要的频率分量。滤波器分为模拟滤波器和数字滤波器，可以用软件实现的只有数字滤波器。

数字滤波器的设计有两大类：无限长单位脉冲响应滤波器（IIR 滤波器）和有限长单位脉冲响应滤波器（FIR 滤波器）。它们各有自己的特点，IIR 可以较好的保留幅值频率特性；FIR 可以实现相位的不失真。在实际使用中，若注重幅值特性，对相位要求不敏感的场合，如语言通讯等，适合选择 IIR 滤波器；而对相位信息有很高的要求，如图象信号处理、数据传输等场合则选择 FIR 滤波器。

LabWindows/CVI 提供了大量的滤波器设计函数，IIR 滤波器函数类中包含了常用的巴特沃斯滤波器、切比雪夫滤波器和椭圆滤波器等。FIR 滤波器函数类主要是以窗口法构造的 FIR 滤波器。

图 4-6 所示为进行滤波处理的对话框。

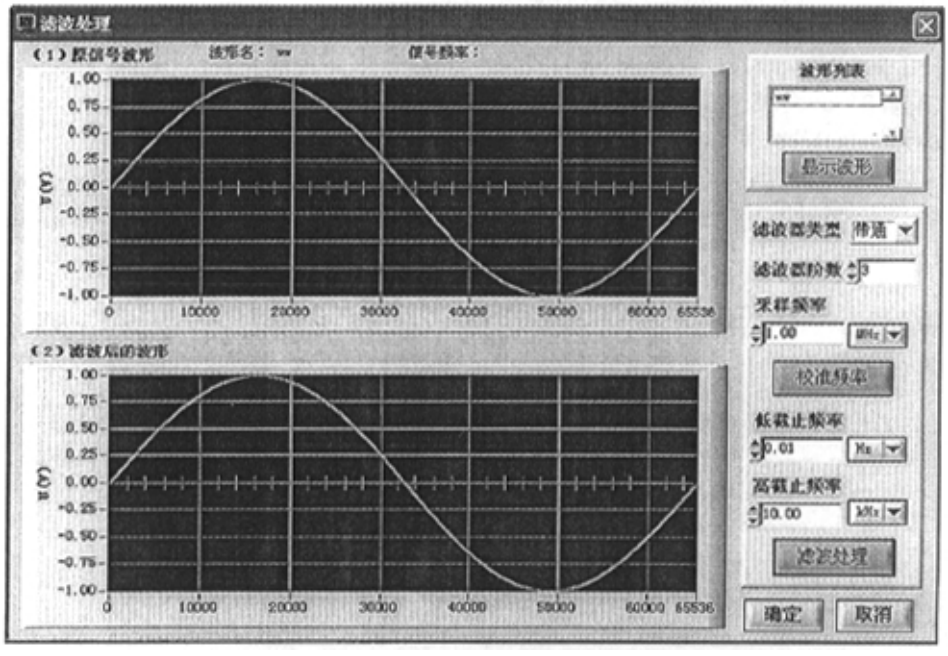

图 4-6　滤波处理对话框

在本软件的设计中，使用的是逐步滤波法。步骤如下：

(1)　选定滤波器类型和阶次，用函数 AllocIIRFilterPtr( )。

函数定义：用函数 AllocIIRFilterPtr(int type, int order)。

其中，type 是滤波器类型（低通、高通、带通、带租）；order 是滤波器的阶数，默认值为 3。

(2) 选择滤波器的种类。用 Bw_CasadeCoef( )表示选用巴特沃斯滤波器。

函数定义：Bw_CasadeCoef(double fs, double fl, double fh, IIRFilterPtr filterInformation)。

其中，s 是采样频率；fl 是低截止频率；fh 是高截止频率；filterInformation 是滤波器结构变量指针，返回滤波器的系数值和内部状态参量。

(3) 实现对信号的滤波，用函数 IIRCascadeFiltering( ) 。

函数定义：IIRCascadeFiltering(const double x[], int n, IIRFilterPtr filterInformation, double y[])。

其中，x[]是原始信号采样值；n 是信号采样值个数，也即数组 x 和 y 的元素个数；filterInformation 是滤波器结构变量指针，给定滤波器系数。

4.6 平滑处理

许多仪器的软件设计中都设计到对采样波形的平滑处理。即削减波形中多余的毛刺、尖峰，使波形的连接更加光滑。

平滑处理的方法是求平均值。先选定一个要进行平滑处理的波形点，然后将它左面几个点和右面几个点的幅度值相加，再除以相加的总的波形点数。例如，波形的起始点为 0，终止点为 999，则从第 8 点开始进行平滑处理，程序循环到第 991 点。计算公式为：

$$y[i] = \left(\sum_{m=0}^{7} y[i-1-m] + \sum_{m=0}^{7} y[i+1+m] \right) / 16$$

这样平滑处理后的波形将有一定程度的失真，因每个点的幅值是其左边和右边各 8 个点的幅值的平均值，因此更新后的幅值将有所减小。但平滑效果较好，对于比较不规则的波形，连续进行多次平滑处理之后，波形将变得较为光滑平坦。

下面是进行平滑处理的程序段。

```
……
samp = wave[_nowWaveNum].samples;
for(i = 16; i < samp-16; i++)
  {
    total_Y = 0;
    for(j=16; j>0; j--)
       total_Y = total_Y + wave[_nowWaveNum].Y[i-j];
    for(j=1; j<=16; j++)
       total_Y = total_Y + wave[_nowWaveNum].Y[i+j];
```

```
    wave[_nowWaveNum].Y[i] = total_Y / 32.0;
  }
```

4.7 本章小节

本章介绍了波形编辑器又一类主要的功能，即对波形的编辑及处理，只有前一章叙述的波形产生方式是不可能实现产生任意波形的功能的。对于用户需要的各种波形，只有通过灵活的叠加、减切、粘贴、加、减、乘、除，拉伸压缩等操作，才能编辑出式样繁多、符合各种场合需要的任意波形。

波形的处理包括加窗、滤波、平滑等，充分利用计算机强大的数据处理能力对数字信号进行处理，虽然这些功能不是波形编辑过程中的必须功能，但它们丰富了软件的内容，使得对数字信号的很多算术处理也能够一并进行。

第五章 波形编辑器的辅助功能

本章简介：软件提供的各种辅助功能加强了波形显示效果，方便用户编辑过程中的观察。例如可以将波形的某一部分或整个波形的显示进行放大或缩小，使得观察波形的局部细节或整体形状更为方便；其次界面上的各种控件颜色等可以由用户根据自己的喜爱进行修改，使得软件的使用更加人性化；多波形同时显示的功能使得波形编辑更加方便，例如要在两个波形之间进行某波形段的减切、粘贴等操作，就可以将这两个波形同时在界面上显示，这样在两波形间的切换操作也比较直观。下面，将就以上提到的几点软件设计的辅助功能进行详细的叙述。

5.1 波形局部区域或整个波形的放大、缩小

有时为了更好的观察波形的局部细节，需要将指定的某部分区域进行放大，观察结束后还可以还原刚才的显示大小；又有的时候需要观察整个波形的整体形状或要将某波形段粘贴到该波形的末尾，这时就需要缩小波形的显示，使得一屏中有较多的波形能够显示。

对于波形显示效果的放大或缩小，在软件的设计中提供了三种方法可以实现。第一，通过直接修改波形显示区的横纵坐标轴的最大最小值来改变显示范围；第二，可以用鼠标在波形显示区里拖动，拖出一个矩形状的区域，则包含在此区域中的波形将被放大为整屏显示，之后还可以恢复刚才的显示大小；第三，通过设置某一条件的基准值，然后以这个基准值为中心进行显示区域的放大，之后也可以恢复刚才的显示大小。下面将一一叙述这三种方法。

(1) 直接修改波形显示区的横纵坐标轴的最大最小值

图 5-1 所示为修改波形显示区横纵坐标轴的最大最小值的对话框。

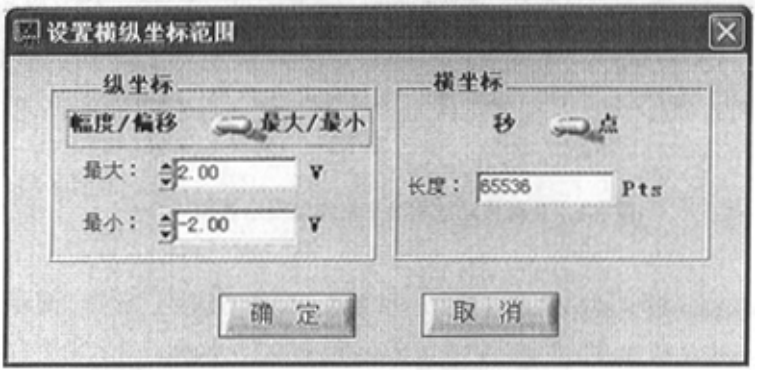

图 5-1 改变显示区横纵坐标轴范围的对话框

其中横轴的单位是“点”，纵轴的单位是“V”。实现此功能的关键是使用了CVI的库函数 **SetAxisRange( )**。

函数定义为：int SetAxisRange (int panelHandle, int controlID, int xAxisScaling, double xminORxinit, double xmaxORxinc, int yAxisScaling, double ymin, double ymax);

这是针对CVI的 Graph 控件而设定的函数。Graph 的各种属性中有四个十分重要的属性，这就是横轴的最大、最小值，纵轴的最大、最小值，它们决定了波形显示区 Graph 的显示范围。只要修改了这四个参数，就可以任意的设置显示区横纵方向的大小，使得满足用户的要求。

(2) 鼠标拖动放大波形为全屏显示

图 5-2 所示为鼠标拖动过程中的示意图，图中的矩形即是鼠标拖动过程中产生的，框在矩形中的波形段即将放大为整屏显示。

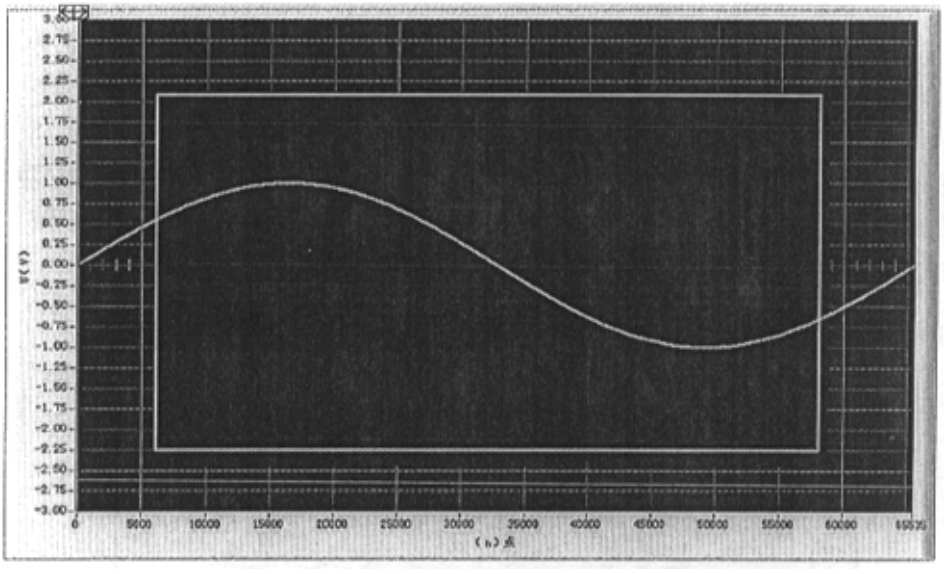

图 5-2　鼠标拖动放大波形操作过程示意图

用鼠标拖动放大波形是最直观的一种方法。操作方法是，选择“显示”菜单里的“鼠标拖动放大波形”选项。在波形显示区里移动鼠标，当放到了合适的位置（这个位置是矩形的左上角），然后按下鼠标左键，开始拖动鼠标。拖动过程中不放左键，而且拖动时会有一个矩形出现，矩形的左上角即是鼠标开始按下的位置，矩形的右下角是当前鼠标的位置，因此，矩形的大小随鼠标的移动而更新。

直到拖动到需要的位置（这个位置是矩形的右下角），放开左键，至此放大操作完成。你将看到刚才被矩形包围的波形部分，现在变成全屏显示。如果想恢复放大之前的显示效果，请选择“显示”菜单里的“恢复原始波形”选项即可实现。

在此功能的设计中，除用到刚才提到的函数 SetAxisRange( ) 来改变显示区的横纵向范围，还用到了叙述“手动绘制波形”设计中使用的 Timer 控件。于是可知，在整个软件的设计中用到了两个 Timer 控件，第一个 Timer1 在手动绘制波形时用来采集离散的波形点，另外还负担实时显示鼠标坐标的功能；第二个 Timer2 就是现在为了绘制不断更新的矩形而设的。需要注意的是：两个 Timer 控件不能同时处于激活状态，即 Timer2 初始为非激活状态(unenable)，当选中了“鼠标拖动放大波形”菜单项时，先停止 Timer1，在将 Timer2 激活(enable)。

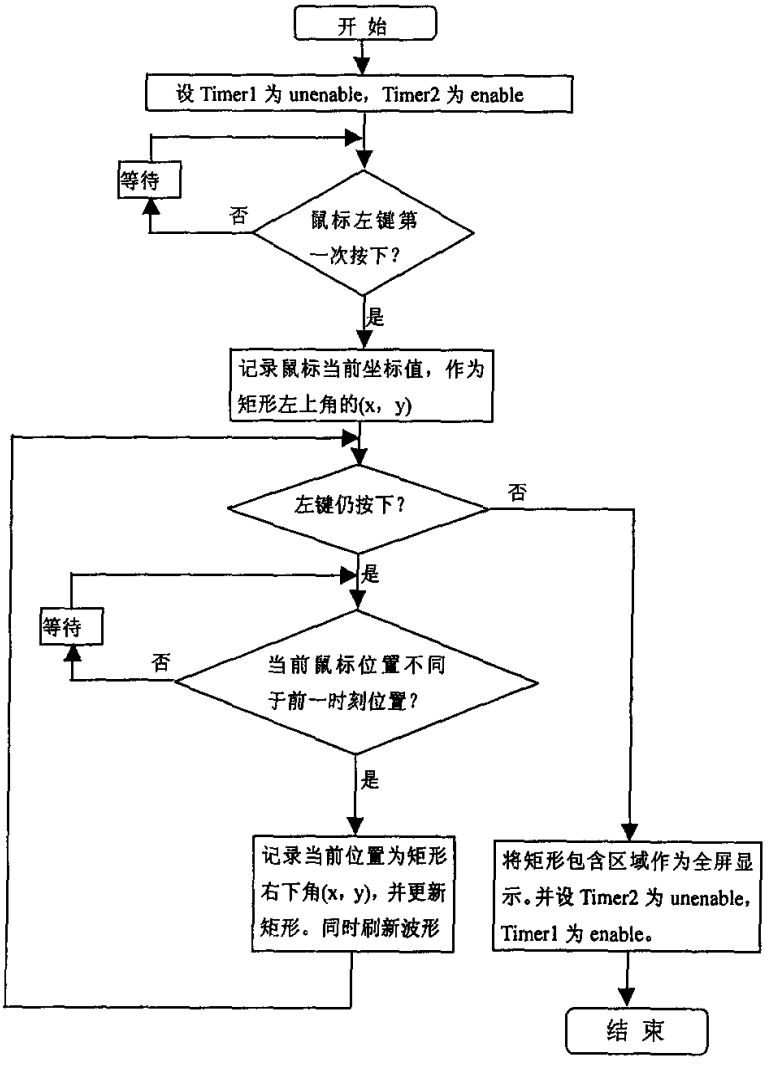

图 5-3 鼠标拖动放大波形流程图

除了控件 Timer 的使用，在鼠标拖动过程中还需执行不断刷新波形的操作。因为，当生成矩形时，该矩形有填充色，会覆盖显示区的波形，为了使矩形看起来是只有边框而没有填充色的效果，需要不断的刷新波形。因此，操作过程中会看到波形有轻微的抖动现象。图 5-3 所示为鼠标拖动放大波形的流程图。

(3) 设置纵轴的基准值为中心放大波形为全屏显示

此功能相似于示波器上的“自动”(Auto Set)，即自动将波形的最大幅值设置为显示区纵轴的最大显示范围，最小幅值设置为纵轴的最小显示范围，这种情况实际是将波形中部设置为纵轴的中间值。

除上述的“以波形中部为对称放大波形为全屏显示”，软件还设计了以 0 为对称放大波形，以及以给定一幅值为对称放大波形两种方法。此三种方法的显示效果虽然不相同，但原理一样，都是先找出波形幅度的最大、最小值，然后以设置好的纵轴中心值为基准，将整个波形在全屏显示。

5.2 多波形显示功能

多波形显示功能在前面的叙述中提到过，此功能的主要用途是将多个波形用不同颜色同时显示，这样可以便于观察比较各个波形。例如要将波形 A 的某一段减切，然后粘贴到波形 B 上，这时如果将 A、B 两波形同时显示，互相切换进行操作，观察起来就比较直观。

实现该功能关键使用了 CVI 提供的库函数 **PlotY( )** 和 **DeleteGraphPlot( )**。

函数定义为：
```
int PlotY (int panelHandle, int controlID, void *yArray, int
           numberOfPoints, int yDataType, int plotStyle, int
           pointStyle, int lineStyle, int pointFrequency,
           int color);

int DeleteGraphPlot ( int panelHandle, int controlID, int plotHandle,
                      int refresh);
```

从函数各个参数的含义可知，“多波形显示”和“单个波形显示”的区别就在于函数 DeleteGraphPlot( ) 中三个参数的使用。

“单个波形显示”时，例如波形之间进行切换，应该先清空整个显示区，即把当前显示的波形清除掉，再绘制上另一个波形，这时参数 =-1。

“多波形显示”时，例如同时显示了波形 A、B、C、D，当前的 A 是活动波形，用黄色显示，B、C、D 是背景波形，用兰色显示。现在要在波形 A、C

之间进行切换，即 A 变成兰色显示的背景波形，C 变成黄色显示的活动波形。不必把所有波形全部删除再重新绘制，只需要删除 A 和 C，再把 A 用兰色绘制，C 用黄色绘制。而程序如何知道这么多波形，谁是 A 谁是 C 呢？在每一次用函数 PlotY( ) 绘制波形时，对应每一个波形有返回值 plotHandle，将 plotHandle 代入 DeleteGraphPlot( ) 的第三个参数 plotHandle，这样从显示区删除的就是 plotHandle 对应的一个波形。在这个例子中，假设波形 A、C 的 plotHandle 分别是 plotHandle_A 和 plotHandle_B，那么连续调用两次 DeleteGraphPlot( )，将 plotHandle_A 和 plotHandle_B 分别代入，则波形 A 和 C 被删除，波形 B、D 不变，再连续调用两次 PlotY( )，绘制波形 A 时 PlotY( )的最后一个参数 color=VAL_BLUE，绘制 B 时 color=VAL_YELLOW，这样 A 为兰色波形，C 为黄色波形，C 此时成了活动波形，可以对它进行其它的编辑处理操作而不会影响另外几个波形。

5.3 显示界面控件颜色的修改

该辅助功能主要是方便用户可以随自己的喜好修改显示界面上各种控件及波形的颜色。实现该功能关键使用了 CVI 的颜色控件 Color Numeric，程序运行时，鼠标单击颜色控件，就会有一个调色框弹出，选中其中你喜欢的颜色，各种颜色在程序编写中是由 0-256 之间的数值来标识的，将该数值传入修改控件属性的库函数 **SetCtrlAttribute( )** 的相应参数，则对应的控件颜色被替换为你刚才从调色框中选取的颜色。

图 5-4 即是用来改变显示界面中各种控件和波形颜色的对话框。

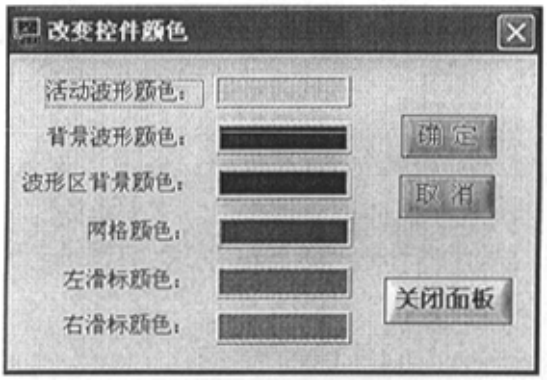

图 5-4 修改控件和波形颜色的对话框

图中提供的可以修改颜色的有：活动波形颜色；背景波形颜色；显示区背景颜色；网格颜色；左右滑标颜色。它们的初始颜色分别是：黄色、深兰色、黑色、

深灰色、浅兰色、紫红色。可以将它们当中的任意一项改为用户喜爱的颜色。

5.4 本章小节

本章介绍的是波形编辑器的辅助功能，这里介绍的只是其中的几项在软件设计过程中难度相对大的内容。辅助功能主要针对的是界面显示效果方面的修改，波形显示的放大或缩小、多波形同时显示、界面控件颜色的修改等都是为了方便用户在波形编辑过程中视觉上的观察。充分利用了计算机强大的图形显示功能，这样就能体现虚拟仪器在图形显示上比传统台式仪器的优越性。辅助功能同样也是为了丰富软件内容而设的。

结束语

本文对当今虚拟仪器的优秀测试平台 VXI 任意波形发生器以及它的软件进行了比较详细的介绍。VXI 任意波形发生器的研制开发是为了提高我国军用测试系统技术的整体水平，以任意波形发生器为代表的通用信号源，是现代测试领域内应用最为广泛的通用仪器之一。从整体上看，信号 ES14V21 的 VXI 任意波形发生器相当于国际上九十年代中期的水平，最终由中国电子科技集团公司组织会议通过了仪器鉴定。

课题研究的主要对象是任意波形发生器的应用软件，包括软面板和任意波形编辑器。本课题采用的开发工具是当前比较流行的虚拟仪器开发工具 LabWindows/CVI6.0。软面板相当于传统台式仪器的操作界面，界面设置简单易学，用户可以方便的在软面板上设置各种波形参数，控制模块产生相应的波形信号，包括 9 种常规波形、各种调制波形，以及用户需要的任意波形。波形编辑器主要负责任意波形的创建、编辑、处理，功能强大，能产生各种形状的波形信号，使得任意波形发生器成为仿真实验的最佳工具。论文从整体结构搭建及各部分的具体实现进行了详细的阐述，对虚拟仪器软件的整个设计过程及思路给予了具体的介绍。实验证明，本课题设计的应用软件实用性强，功能强大、操作过程简单易学。

测试程序的开发过程中还是有不足之处，在许多方面还需要改进。在此就个人意见提出一些软件的改进措施。

☆ 用多个时钟控件分别处理手动任意绘制波形、水平方向绘制、垂直方向绘制，以提高绘制波形的速度。

☆ 现在波形库中的波形种类比较单一，应增加波形库里的波形种类。

☆ 软件产生的波形数据在传送到硬件存储区的过程中应该设置为分辨率可选，现在设计的是固定的 12bit。

☆ 波形编辑器中的波形参数设置有和软面板中重复的地方，可将这些设置简化。

致谢

本课题是在导师王厚军、田书林、黄建国老师的悉心指导下完成的，他们为课题的开展注入了大量心血。导师忘我的工作热情、严谨的治学态度、深厚的理论知识及实际工作经验使我深受裨益，尤其是导师们在生活上、思想上对我的关怀和帮助使我非常感激。在此还要感谢教研室的师奕兵、徐建南、韩熙利老师，他们在我读研期间从许多方面给予了我关怀和帮助。

在课题研究期间，同课题组的老师和同学对我思路的开发以及软件的完善等各方面给予了很大的帮助，在此要感谢课题组负责教师周鹏老师，同课题组的同学刘科、冉廷华、邵海涛。大家在同一个实验环境下，互相帮助，在一个轻松、愉快的环境里度过紧张的课题研究阶段。另外，还要感谢陆明、蒋薇、段芙蓉同学，他们给予了我许多很必要的帮助。最后要感谢我的家人，他们默默而有力的支持是我前进的动力。

参考文献

[1] 刘君华 白鹏等 虚拟仪器编程语言 LabWindows/CVI 教程 电子工业出版社 2001.8

[2] 陈光禩 VXI 总线测试平台技术 电子科学出版社

[3] User Manual Arbitrary Waveform Editor Tektronix Apl.1999

[4] LabWindows/CVI User Manual Feb.1998 Editor

[5] 秦小麟 林钧海 用 C 语言实现的数据结构 航空工业出版社

[6] 路林吉 饶家明 面向仪器与测控过程的交互式 C/C++开发平台 —— LabWindows/CVI 电子技术 2000.4

[7] 樊世友 范梅生 LabWindows/CVI 环境下的仪器驱动器开发 机械工程学院学报 2001.12

[8] 王雁东 陈光禩 “VXI 总线测试软件平台”主框架的设计 测控技术 2001 年第 20 卷

[9] 周泓 汪乐宇 VXI 即插即用虚拟仪器软面板设计 现代科学仪器 1998.4

[10] 潘光斌 VXI 总线测试系统软件设计技术研究 计量技术 2001.6

[11] 郭伟 杨江平 VXI 总线虚拟仪器软件设计方法及其应用 计算机工程与应用 2000.8

[12] 郭勇 肖明清 基于 VXI 虚拟仪器软件中的事件 电子技术 2001.1

[13] 孙明 基于 VXI 总线的新型任意波形发生器的设计 数据采集与处理 2002 年 6 月第 17 卷第 2 期

[14] 路林吉 谢萍 VXI 总线概述 电子技术 2000 年第 6 期

[15] 陈客松 王章瑞 一种虚拟仪器概念的任意波形发生器的研制 仪表技术 2000 年第 6 期

[16] 田湛君 鞍山师 基于 VXI 总线虚拟仪器构成及其总线模块的设计 师范学院学报 2002.3

[17] 王忠民 杨大勇 基于 VXI 总线的虚拟仪器技术 机械工业自动化

[18] 胡朝晖　　计算机在测控领域的应用 —— 虚拟仪器系统　　广西工学院学报　1998 年 6 月第 9 卷第 2 期

[19] 朱正伟　　虚拟仪器技术及其应用　　江苏石油化工学院学报　　2000.9

[20] 郭恩全　赵兴奋　　虚拟仪器发展趋势及其对军用测试技术的影响　　计算机自动测量与控制　　1999.7

个人简历

出生年月：1978 年 8 月 20 日

本科专业：自动化工程学院测试电子仪器及测量技术

获学士学位时间：2000 年 7 月

研究生专业：自动化工程学院测试计量技术及仪器

获硕士学位时间：2003 年 4 月

研究生阶段参加科研课题：① VXI 任意波形发生器波形编辑软件的设计

② 标量网络分析仪的软件开发

获奖情况：2001 年和 2002 年均获研究生三等奖学金