全国高等职业教育“十三五”规划教材

LabVIEW 虚拟仪器实训教程

主　编　李江全

副主编　王玉巍　李丹阳　李树峰

机 械 工 业 出 版 社

本书从实际应用出发，系统地介绍了虚拟仪器软件 LabVIEW 的程序设计方法及其测控应用技术。全书分为两篇：LabVIEW 程序设计篇介绍了使用 LabVIEW 编程语言进行程序设计的基本知识，包括虚拟仪器的含义和组成、LabVIEW 数据操作、流程控制、节点、图形显示与变量等；LabVIEW 测控应用篇采用 LabVIEW 实现智能仪器、远程 I/O 模块和数据采集卡的串口通信及测控功能。除第 1 章外，其余各章都安排了相应实训，实训由学习目标、设计任务等部分组成。

本书内容丰富，讲解深入浅出，有较强的实用性和可操作性，可供应用型本科及高职高专测控技术、仪器仪表、工业控制、自动化、机电等专业的学生及工程技术人员学习和参考。

本书提供配套的电子课件，需要的教师可登录 www.cmpedu.com 进行免费注册，审核通过后即可下载；或者联系编辑索取（QQ：1239258369，电话：010-88379739）。

图书在版编目（CIP）数据

LabVIEW 虚拟仪器实训教程 / 李江全主编. —北京：机械工业出版社，2018.6
全国高等职业教育“十三五”规划教材
ISBN 978-7-111-62018-1

Ⅰ. ①L… Ⅱ. ①李… Ⅲ. ①软件工具－程序设计－高等职业教育－教材 Ⅳ. ①TP311.56

中国版本图书馆 CIP 数据核字（2019）第 027266 号

机械工业出版社（北京市百万庄大街 22 号 邮政编码 100037）
策划编辑：李文轶 责任编辑：李文轶
责任校对：张艳霞 责任印制：李 昂

河北宝昌佳彩印刷有限公司印刷

2019 年 3 月第 1 版 • 第 1 次印刷
184mm×260mm • 13 印张 • 318 千字
0001－3000 册
标准书号：ISBN 978-7-111-62018-1
定价：39.00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换

电话服务
服务咨询热线：（010）88379833
读者购书热线：（010）68326294

网络服务
机 工 官 网：www.cmpbook.com
机 工 官 博：weibo.com/cmp1952
教育服务网：www.cmpedu.com
金 书 网：www.golden-book.com

封面无防伪标均为盗版

前　言

虚拟仪器是现代计算机技术、通信技术和测量技术相结合的产物，是传统仪器观念的一次巨大变革。它的出现使测试技术进入一个全新的发展阶段。虚拟仪器既有传统仪器的特征，又有一般仪器不具备的特殊功能，在实际应用中表现出传统仪器无法比拟的优势。可以说，虚拟仪器是测控系统的关键组成部分，在现代测控领域有着光明的应用前景。

作为测试工程领域虚拟仪器开发软件的强有力工具，近年来，LabVIEW 得到了业界的普遍认可。

本书从实际应用出发，系统地介绍了虚拟仪器软件 LabVIEW 的程序设计方法及其测控应用技术。全书分为两篇：LabVIEW 程序设计篇介绍了使用 LabVIEW 编程语言进行程序设计的基本知识，包括虚拟仪器的含义和组成、LabVIEW 数据操作、流程控制、节点、图形显示与变量等；LabVIEW 测控应用篇采用 LabVIEW 实现智能仪器、远程 I/O 模块和数据采集卡的串口通信及测控功能。

除第 1 章外，其余各章都安排了相应实训，实训由学习目标、设计任务等部分组成。每个实训都有详细完整的操作步骤，读者只需按照给定的步骤进行操作，就可完成实训任务，从而轻松掌握虚拟仪器基本设计方法及其测控应用技术；通过实训将虚拟仪器 LabVIEW 软件的技能训练与生产实际紧密结合，从而激发学生的学习兴趣，充分体现学以致用的教学思想。

本书由石河子大学的李江全编写第 1、2 章，左静编写第 3 章，李树峰编写第 4、5 章，新疆工程学院的王玉巍编写第 6、7、8 章，空军工程大学的李丹阳编写第 9、10、11 章。北京研华科技股份有限公司等为本书提供了大量的技术支持，在此深表感谢。

由于编者水平有限，书中难免存在不妥之处，恳请广大读者批评指正。

编　者

目　录

前言

LabVIEW 程序设计篇

第 1 章　虚拟仪器概述……2
1.1　虚拟仪器的产生、概念、特点及应用……2
1.1.1　虚拟仪器的产生……2
1.1.2　虚拟仪器的概念……2
1.1.3　虚拟仪器的特点……3
1.1.4　虚拟仪器的应用……5
1.2　虚拟仪器的组成与构成方式……5
1.2.1　虚拟仪器的基本结构……5
1.2.2　虚拟仪器的构成方式……6
1.2.3　构建虚拟仪器的步骤……8
1.3　虚拟仪器的软件结构与开发平台……8
1.3.1　虚拟仪器的软件结构……8
1.3.2　虚拟仪器的开发平台……9
第 2 章　LabVIEW 程序设计基础……11
2.1　LabVIEW 的特点及应用……11
2.1.1　LabVIEW 的特点……11
2.1.2　LabVIEW 的应用……11
2.2　LabVIEW 的编程环境……12
2.2.1　启动窗口……12
2.2.2　菜单栏……14
2.2.3　工具栏……17
2.2.4　操作选板……19
2.3　LabVIEW 编程的基本概念……22
2.3.1　VI 与子 VI……22
2.3.2　前面板……22
2.3.3　程序框图……23
2.3.4　数据流驱动……24
2.4　VI 前面板设计……25
2.4.1　前面板对象的创建……25
2.4.2　前面板对象的属性配置……26
2.4.3　前面板对象的修饰……27
2.5　VI 与子 VI 设计实训……29
实训 1　体验 VI 设计……29
实训 2　子 VI 的创建与调用……33
2.6　VI 的调试方法……36
2.6.1　找出语法错误……36
2.6.2　设置断点调试……37
2.6.3　设置探针……37
2.6.4　高亮显示程序的运行……38
2.6.5　单步执行和循环运行……38
第 3 章　LabVIEW 数据操作……40
3.1　LabVIEW 数据概述……40
3.1.1　LabVIEW 数据类型……40
3.1.2　数值型数据……41
3.1.3　布尔型数据……43
3.1.4　字符串数据……44
3.1.5　数组数据……46
3.1.6　簇数据……49
3.1.7　LabVIEW 数据运算……51
3.2　LabVIEW 数据操作实训……52
实训 3　数值数据操作……52
实训 4　布尔数据操作……55
实训 5　字符串数据操作……57
实训 6　数组数据操作……60
实训 7　簇数据操作……63
第 4 章　LabVIEW 程序流程控制与节点……66
4.1　程序流程控制……66
4.1.1　条件结构……66
4.1.2　顺序结构……68
4.1.3　For 循环结构……70
4.1.4　While 循环结构……71
4.2　节点的使用……72

4.2.1 公式节点……72
4.2.2 反馈节点……73
4.2.3 表达式节点……74
4.3 流程控制与节点操作实训……74
实训 8 条件结构操作……74
实训 9 平铺式顺序结构操作……76
实训 10 层叠式顺序结构操作……77
实训 11 For 循环结构操作……80
实训 12 While 循环结构操作……82
实训 13 公式节点操作……84
实训 14 反馈节点操作……86
实训 15 表达式节点操作……87
第 5 章 LabVIEW 图形显示与变量……89
5.1 图形显示……89
5.1.1 图形显示概述……89
5.1.2 波形图表控件……90
5.1.3 波形图控件……90
5.2 局部变量与全局变量……90
5.2.1 局部变量……91
5.2.2 全局变量……91
5.3 图形显示与变量操作实训……92
实训 16 波形图表控件操作……92
实训 17 波形图控件操作……93
实训 18 局部变量操作……96
实训 19 全局变量操作……98
实训 20 菜单的设计……101

LabVIEW 测控应用篇

第 6 章 LabVIEW 串口通信……107
6.1 串口通信概述……107
6.1.1 串口通信的基本概念……107
6.1.2 串口通信标准……110
6.1.3 串口通信线路连接……112
6.1.4 PC 中的串行接口……114
6.2 LabVIEW 与串口通信……115
6.2.1 LabVIEW 中的串口通信功能模块……115
6.2.2 LabVIEW 串口通信步骤……117
实训 21 PC 与 PC 串口通信……117
第 7 章 LabVIEW 数据采集……122
7.1 数据采集系统概述……122
7.1.1 数据采集系统的含义……122
7.1.2 数据采集系统的功能……122
7.1.3 数据采集系统的输入与输出信号…123
7.2 数据采集卡……125
7.2.1 数据采集卡的类型……125
7.2.2 数据采集卡的选择……126
7.2.3 基于数据采集卡的测控系统……127
7.2.4 典型数据采集卡的安装……129
7.3 LabVIEW 与数据采集……131
7.3.1 基于 LabVIEW 的数据采集系统…131
7.3.2 DAQ 助手的使用……132
第 8 章 模拟量输入系统与 LabVIEW 实训……134
8.1 模拟量输入系统应用实例……134
8.1.1 货车自动称重……134
8.1.2 水库水位监测……135
8.1.3 发动机台架试验……136
8.1.4 发电厂锅炉监控……137
8.1.5 模拟量输入系统总结……139
8.2 模拟量输入 LabVIEW 实训……139
实训 22 智能仪表温度检测……139
实训 23 远程 I/O 模块电压采集……145
实训 24 数据采集卡电压采集……150
第 9 章 数字量输入系统与 LabVIEW 实训……154
9.1 数字量输入系统应用实例……154
9.1.1 饮料瓶计数喷码……154
9.1.2 银行防盗报警……155
9.1.3 电梯集中监控……156
9.1.4 驾考汽车压线监测……158
9.1.5 数字量输入系统总结……159
9.2 数字量输入 LabVIEW 实训……159
实训 25 远程 I/O 模块数字量输入……159
实训 26 数据采集卡数字量输入……164

第 10 章　数字量输出系统与 LabVIEW 实训……169
10.1　数字量输出系统应用实例……169
10.1.1　高速公路 ETC ……169
10.1.2　自动感应门控制 ……170
10.1.3　机械手臂定位控制……171
10.1.4　汽车充电桩控制 ……172
10.1.5　数字量输出系统总结……173
10.2　数字量输出 LabVIEW 实训 ……174
实训 27　远程 I/O 模块数字量输出 ……174
实训 28　数据采集卡数字量输出 ……178
第 11 章　模拟量输入与数字量输出系统综合实训 ……183
11.1　模拟量输入与数字量输出应用实例……183
11.1.1　温室大棚监控……183
11.1.2　轴承滚柱分级……184
11.1.3　零件磨削加工……185
11.1.4　变压器油温监控 ……185
11.1.5　模拟量输入与数字量输出系统总结 ……187
11.2　模拟量输入与数字量输出 LabVIEW 实训……188
实训 29　远程 I/O 模块温度测控……188
实训 30　数据采集卡温度测控 ……195
参考文献……201

LabVIEW 程序设计篇

第 1 章　虚拟仪器概述
第 2 章　LabVIEW 程序设计基础
第 3 章　LabVIEW 数据操作
第 4 章　LabVIEW 程序流程控制与节点
第 5 章　LabVIEW 图形显示与变量

第 1 章　虚拟仪器概述

虚拟仪器是用通用计算机硬件加上软件来仿真传统测量仪器的设备，是以测量、分析、显示为主，以控制为辅的更加先进的科学仪器。它为仪器的测量分析带来了更加辉煌的未来。虚拟仪器技术是计算机测控技术的重要分支。

1.1　虚拟仪器的产生、概念、特点及应用

1.1.1　虚拟仪器的产生

测量仪器发展至今，大体可分为四个阶段：模拟仪器、数字仪器、智能仪器和虚拟仪器。

模拟仪器：基于电磁感应原理的指针式仪器仪表。其基本结构是电磁机械式的，借助指针来显示最终结果，如指针式万用表、晶体管电压表等。

数字仪器：将对模拟信号的测量转化为对数字信号的测量，并以数字方式输出最终结果，适用于快速响应和较高准确度的测量，如数字电压表、数字频率计等。

智能仪器：内置微处理器，既能进行自动测试，又具有一定的数据处理功能。智能仪器的功能模块是以硬件和固化的软件形式存在的，对用户而言，无论是开发还是应用，都缺乏灵活性。

虚拟仪器（Virtual Instrument，VI）：是由美国国家仪器公司（National Instruments，NI）提出的。其基本思想：用计算机资源取代传统仪器中的输入、处理和输出等部分，实现仪器硬件核心部分的模块化和最小化；用计算机软件和仪器软面板实现仪器的测量和控制功能。

虚拟仪器的发展大致可分为三个阶段。

第一阶段是利用计算机来增强传统仪器的功能。通用接口总线 GPIB 标准的确立，使计算机与外部仪器通信成为可能，因此把传统的仪器通过串行接口和计算机连接起来就可以用计算机控制仪器了。

第二阶段主要在功能硬件上实现了两大技术的进步。其一是插入计算机总线槽上的数据采集卡的出现，其二是 VXI 仪器总线标准的确立。这些新技术的应用奠定了虚拟仪器硬件的基础。

第三阶段形成了虚拟仪器体系结构的基本框架。主要是采用面向对象的编程技术构筑了几种虚拟仪器的软件平台，并逐渐成为标准的软件开发工具。

虚拟仪器是现代计算机软硬件技术和测量技术相结合的产物，是传统仪器观念的一次巨大变革，是将来仪器发展的一个重要方向。

1.1.2　虚拟仪器的概念

所谓虚拟仪器，就是在以计算机为核心的硬件平台上，由用户设计和定义其功能，使其

具有虚拟面板，其测试功能由测试软件实现的一种计算机仪器系统。

虚拟仪器是一种概念仪器，迄今为止，业界对它还没有一个明确的国际标准和定义。虚拟仪器实际上就是一种基于计算机的自动化测试仪器系统。一般认为，所谓虚拟测量仪器，就是采用计算机开放体系结构取代传统的单机测量仪器，对各种各样的数据进行计算机处理、显示和存储的测量仪器。

虚拟仪器的实质是利用计算机显示器的显示功能来模拟传统仪器的控制面板，以多种形式表达输出检测结果；利用计算机强大的软件功能实现信号数据的运算、分析和处理；利用 I/O 接口设备完成信号的采集、测量与调整，从而完成各种测试功能的一种计算机仪器系统。使用者利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一样。因此，虚拟仪器的出现，使测量仪器与计算机的界限模糊了。

虚拟仪器的“虚拟”两字主要包含以下两方面的含义。

1）虚拟仪器的面板是虚拟的。

虚拟仪器面板上的各种“图标”与传统仪器面板上的各种“器件”所完成的功能是相同的。由各种开关、按钮、显示器等图标实现仪器电源的“通”“断”，被测信号的“输入通道”“放大倍数”等参数的设置，以及测量结果的“数值显示”“波形显示”等。

传统仪器面板上的器件都是“实物”，而且是通过“手动”和“触摸”进行操作的；虚拟仪器前面板是外形与实物相像的“图标”，每个图标的“通”“断”“放大”等动作通过用户操作计算机鼠标或键盘来完成。因此，设计虚拟仪器前面板就是在前面板设计窗口中摆放所需的图标，然后对图标的属性进行设置。

2）虚拟仪器测量功能是通过对图形化软件流程图的编程来实现的。

虚拟仪器是在以 PC 为核心的硬件平台支持下，通过软件编程来实现仪器的测量功能的。因为可以通过不同测试功能软件模块的组合来实现多种测试功能，所以在硬件平台确定后，就有“软件就是仪器”的说法。这也体现了测试技术与计算机深层次的结合。

虚拟仪器概念是为了适应 PC 卡式仪器而提出的。众所周知，传统仪器主要包括三个部分：数据采集与控制、数据分析和处理、数据显示。而 PC 卡式仪器由于自身不带仪器面板，有的甚至不带微处理器，因此必须借助于 PC 作为其数据分析与显示的工具，利用 PC 强大的图形环境建立图形化的虚拟仪器面板，完成对仪器的控制、数据分析与显示。这种包含实际仪器使用、操作信息的软件与 PC 结合构成的仪器，就称为虚拟仪器。或者说，虚拟仪器是指具有虚拟仪器面板的 PC 仪器，它由 PC、一系列功能化硬件模块和控制软件组成。

“Virtual”一词被通常译成“虚拟”。在测控仪器领域，“Virtual”不仅仅指用计算机屏幕来虚拟各种传统仪器的面板，还有“实质上的”“实际上的”“有效的”和“似真的”的含义，完全不同于虚拟现实中的虚拟人、虚拟太空、虚拟海底、虚拟建筑等非“实际”的概念。测控仪器强调的是“实”，而不是“虚”。因此，在研究与发展虚拟仪器技术时，要注重利用计算机的软硬件技术实现测控仪器的特点和功能，而不能仅强调虚拟的、只是视觉上的内容，要强调面向测控领域快速有效地解决实际问题。

1.1.3 虚拟仪器的特点

传统的测量仪器基本上是以硬件形式或固化的软件形式存在的，测量仪器只能由制造商来定义与设计，因而其灵活性和适应性较差。

在实验室、生产车间和户外现场，为完成某项测试和维修任务，通常需要许多仪器，如

信号源、示波器、频谱分析仪等。由众多的仪器构成的测试系统，价格昂贵，体积庞大，连接和操作复杂，测试效率低。

与传统测量仪器相比，虚拟仪器的设计理念、系统结构和功能定位方面都发生了根本性的变化。概括地说，虚拟仪器主要有以下特点。

1）软件是虚拟仪器的核心。虚拟仪器的硬件确立后，它的功能主要是通过软件来实现的，软件在虚拟仪器中具有重要的地位。借助于通用数据采集系统（或板卡），用户可以通过软件构造任意功能的仪器，软件变成了构建仪器的核心部分，因此美国国家仪器公司（NI）曾提出一个著名的口号，即“软件就是仪器”。

2）虚拟仪器的性价比高。一方面，虚拟仪器能同时对多个参数进行实时高效的测量，同时，由于信号的传送和数据的处理几乎都是靠数字信号或软件来实现的，所以还大大降低了环境干扰和系统误差的影响。另一方面，用户也可以随时根据需要调整虚拟仪器的功能，这缩短了仪器在改变测量对象时的更新周期。此外，采用虚拟仪器还可以减少测试系统的硬件环节，从而降低系统的开发成本和维护成本。因此，使用虚拟仪器比传统仪器更经济。

3）虚拟仪器的出现缩短了仪器厂商与用户之间的距离。虚拟仪器使得用户能够根据自己的需要定义仪器功能，而不像传统仪器那样，受到仪器厂商的限制，出现厂商提供的仪器功能不符合用户要求的情况。利用虚拟仪器，用户可以组建更好的测试系统，并且更容易增强系统的功能。

4）扩展性强。NI 的软硬件工具使得工程师和科学家不再局限于当前的技术。得益于 NI 软件的灵活性，只需更新用户的计算机或测量硬件，就能以最少的硬件投资，以及极少的甚至无须软件上的升级，即可改进用户的整个系统。

5）虚拟仪器具有良好的人机界面。在虚拟仪器中，测量结果是通过软件在计算机显示器上生成的，与传统仪器面板相似的图形界面由软面板来实现。因此，用户可根据自己的爱好，通过编制软件来定义所喜爱的面板形式。

6）通过软硬件的升级，可以方便地提升测试系统的能力和水平。更可贵的是，用户可以运用通用的计算机语言和软件，诸如 C++、Visual Basic、LabVIEW、LabWindows/CVI 等，扩充、编写软件，从而使虚拟仪器技术更适应、更符合用户自己测试工作的特殊需求。

7）虚拟仪器具有和其他设备互联的能力，如和 VXI 总线或现场总线等的接口能力。此外，还可以将虚拟仪器接入网络，如 Internet 等，以实现对现场生产的监控和管理。

8）虚拟仪器的软硬件都具有开放性、模块化、可重复使用及互换性等特点。因此，用户可以根据自己的需要灵活组合，大大提高了使用效率，减少了投资。

表 1-1 列出了传统仪器与虚拟仪器的主要区别。

表 1-1　传统仪器与虚拟仪器的比较

传统仪器	虚拟仪器
硬件是关键，必须由专业厂家升级	软件是关键，升级方便
基于硬件体系，开发与维护费用高	基于软件体系，开发与维护费用低
数据无法编辑	数据可编辑、存储、打印
硬件技术更新周期长	软件技术更新周期短
通用性差，价格高	价格低，并且可重用性与可配置性强
厂商定义仪器功能	用户定义仪器功能

（续）

传统仪器	虚拟仪器
系统封闭，功能固定不可更改	系统开放、灵活，功能可更改，构成多种仪器
不易与其他设备连接	容易与网络、外设及其他设备连接
图形界面小，信息量小	图形界面大，信息量大
部分具有时间记录和测试说明	完整的时间记录和测试说明
信号电缆和开关多，操作复杂	信号电缆少，采用虚拟旋钮，故障率低，有操作保护
测试部分自动化	测试过程完全自动化

1.1.4 虚拟仪器的应用

虚拟仪器由于其功能灵活，很容易构建，所以应用面极为广泛。尤其在科研、开发、测量、计量等领域更是不可多得的好工具。虚拟仪器技术先进，十分符合国际上流行的“硬件软件化”的发展趋势，因而常被称为“软件仪器”。它功能强大，可实现示波器、逻辑分析仪、频谱仪、信号发生器等多种普通仪器的全部功能。虚拟仪器系统已成为仪器领域的一个基本方案，是技术进步的必然结果。它的应用已经遍及各行各业的测量活动。

在自动控制和工业控制领域，虚拟仪器同样应用广泛。绝大部分闭环控制系统要求精确地采样，及时地处理数据和快速地传输数据。虚拟仪器系统恰恰符合上述特点，十分适合测控一体化的设计。尤其在制造业，虚拟仪器的卓越计算能力和巨大数据吞吐能力必将使其在实时监控系统、在线监测系统、电力仪表系统、流程控制系统等工控领域发挥更大的作用。

虚拟仪器的出现是仪器发展史上的一场革命，代表着仪器发展的最新方向和潮流，是信息技术的一个重要领域，对科学技术的发展和工业生产将产生不可估量的影响。

虚拟仪器可广泛应用于电子测量、振动分析、声学分析、故障诊断、航天航空、军事工程、电力工程、机械工程、建筑工程、铁路交通、地质勘探、生物医疗、教学及科研等诸多方面。

1.2 虚拟仪器的组成与构成方式

1.2.1 虚拟仪器的基本结构

虚拟仪器的基本结构包括计算机硬件、仪器硬件和虚拟仪器软件三部分，如图 1-1 所示。

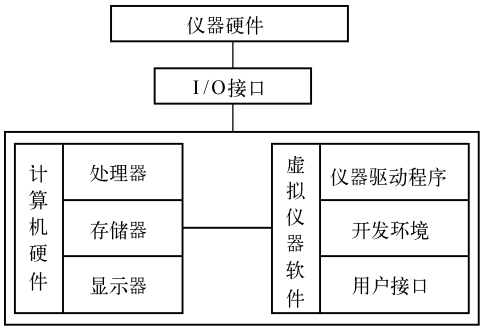


图 1-1 虚拟仪器的基本结构

1．计算机硬件

计算机硬件平台可以是各种类型的计算机，如普通台式计算机、便携式计算机、工作站、嵌入式计算机等。计算机管理着虚拟仪器的硬件、软件资源，是虚拟仪器的硬件基础。

2．仪器硬件

仪器硬件根据不同的标准接口总线转换输入或输出信号，供其他系统使用。

仪器硬件部分可由数据采集卡、GPIB 接口、串并行接口、VIX 接口、LAN 接口、现场总线接口等构成，它们的主要功能是完成被测信号的采集、传输和显示测量的结果。

3．虚拟仪器软件

虚拟仪器的软件是核心、关键部分，用于实现对仪器硬件通信和控制，对信号进行分析处理，对结果表达和输出。

虚拟仪器实质上是“软硬结合”“虚实结合”的产物，它充分利用最新的计算机技术来实现和扩展传统仪器的功能。它强调软件的作用，提出“软件就是仪器”的概念。

在虚拟仪器系统中，硬件仅仅解决信号的输入、输出，以及软件赖以生存、运行的物理环境，软件才是整个仪器系统的关键。用户可根据自己的需要通过编制不同的测试软件来构成各种功能的测试系统，其中许多硬件功能可直接由软件实现，系统具有极强的通用性和多功能性。任何使用者都能通过调整或修改仪器的软件，方便地改变仪器的功能和规模，甚至仪器的性质。

虚拟仪器软件的开发又有着自身的特殊性。这种特殊性主要体现在虚拟仪器软件在某种程度上是传统硬件的“仿真”，其设计目的之一就是用软件来实现硬件的功能。

1.2.2 虚拟仪器的构成方式

虚拟仪器的硬件平台由计算机及其 I/O 接口设备两部分组成。I/O 接口设备主要执行信号的输入、数据采集、放大、模-数转换等任务。

根据 I/O 接口设备总线类型的不同，虚拟仪器的构成方式主要有：基于 PC 的插卡式（PC-DAQ）、GPIB 总线、VXI 总线、PXI 总线、串行接口总线、现场总线等六种标准硬件体系结构，如图 1-2 所示。

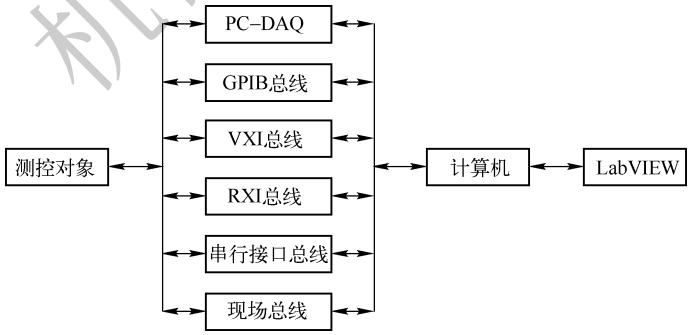


图 1-2　虚拟仪器的构成方式

1．基于 PC 的插卡式（PC-DAQ）虚拟仪器

通过在 PC 内直接插入一块内插式多功能数据采集卡，将前端仪器（如传感器等）送来的模拟信号经 A-D 转换送到计算机，直接经过 PCI 总线，由 CPU 进行分析、处理，再通过

显示器显示，外接打印机后打印等。它充分地利用计算机的资源，大大增加了测试系统的灵活性和扩展性。

这种方式受 PC 的机箱和总线的限制，存在电源功率不足、机箱内噪声电平较高、无屏蔽、插槽数目不多、尺寸较小等缺点。但因 PC 数量非常庞大，插卡式仪器虚拟价格便宜，因此其用途广泛，特别适合于工业测控部门、各种实验室和教学部门使用。

2．基于 GPIB 总线的虚拟仪器

GPIB（IEEE 488 标准）是计算机和仪器间的标准通信协议，也是最早的仪器总线。一个典型的 GPIB 测试系统由 1 台 PC、1 块 GPIB 接口卡和若干台 GPIB 仪器通过 GPIB 电缆连接而成。每台 GPIB 仪器有单独的地址，由计算机控制其操作。将 GPIB 接口板插入计算机的插槽中，建立起计算机与具有 GPIB 接口的仪器之间的通信桥梁。

利用 GPIB 技术，可以用计算机实现对仪器的操作和控制，替代传统的人工操作方式，可以方便地将多台仪器组合起来，形成较大的自动测试系统。若增加、减少或更换系统中的仪器，只需对计算机的控制软件进行相应改动，就可高效、灵活地完成各种不同规模的测试任务。

GPIB 测试系统的结构和命令简单，造价较低，主要作为台式仪器，适用于精确度要求高，但对计算机速率和总线控制实时性要求不高的传输场合。

3．基于 VXI 总线的虚拟仪器

VXI 总线是一种高速计算机总线在仪器领域的扩展。这种虚拟仪器由 VXI 标准机箱、零槽控制器、具有多种功能的模块化仪器和驱动软件、系统应用软件等组成。

VXI 总线标准具有标准开放、即插即用、结构紧凑、数据吞吐能力强、定时与同步精确、模块可重复利用、众多仪器生产厂商支持等优点，应用越来越广。VXI 总线规范使得用户在组建这种虚拟仪器时不必局限于某一厂商的产品，可以根据自己的要求自由选购各仪器厂商的仪器模块，从而使系统达到最优。

尤其在组建大中规模自动测量控制系统时，以及在对速度、精度要求非常高的场合，基于 VXI 总线的虚拟仪器有其他仪器无法比拟的优点。另外，基于 VXI 总线的组建方案功能最强大、组建的系统最稳定，但价格十分昂贵。

4．基于 PXI 总线的虚拟仪器

基于 PXI 总线的虚拟仪器是一种新型模块化仪器系统，是在 PCI 总线内核技术上增加了成熟的技术规范和要求形成的，包括多板同步触发总线技术，增加了用于相邻模块的高速通信的局部总线，并具有高度的可扩展性等优点，适用于大型高精度集成系统。

5．基于串行接口总线的虚拟仪器

通过串行接口可实现仪器与计算机、仪器与仪器之间的相互通信，从而组成由多台仪器构成的自动测试系统。RS-232 总线是早期采用的 PC 通用的串行总线，适合于单台仪器与计算机的连接，但控制性能较差。当今 PC 中已更多采用 USB 总线，基于 USB 总线的虚拟仪器开发已经受到重视。但是，USB 总线目前只用于较简单的测试系统。在用虚拟仪器组建自动测试系统时，目前最有发展前景的是采用 IEEE 1394 的高速串行接口总线。

6．基于现场总线的虚拟仪器

现场总线是一种全数字化、串行、双向、多站的通信网络，基于现场总线的虚拟仪器是以现场总线（FieldBus）为纽带，把多个分散的智能仪表、控制设备（包括智能传感器）连接成可以相互沟通信息、共同完成自控任务的网络与控制系统。用于现场总线系统的智能传

感器、变送器、仪表等统称为现场总线仪表。各种现场总线仪表采用标准化的、开放式通信协议，这样可以将不同厂商的产品方便地挂接在现场总线上，使系统具有可操作性。

1.2.3 构建虚拟仪器的步骤

在实验室里，有各种各样的仪器与设备。如何提高它们的综合使用效率？如何对它们进行更有效的管理？最有效的方法是采用“虚拟仪器”技术，即充分利用计算机强大的管理与处理能力，并以此为基础，将实验室相关设备搭配起来，构成一种全新的实验环境。实验室中的仪器与设备一般都是具有特定功能的单台设备，如果它们具有某种总线接口，就有可能进行虚拟仪器的构建。

构建虚拟仪器系统的步骤如下。

1．确定所用仪器或设备的接口形式

如果仪器设备具有 RS-232 串行总线接口，则不用进行处理，直接用连线将仪器设备与计算机的 RS-232 串行接口连接即可。

如果是 GPIB 接口，则需要额外配备一块 GPIB 接口板卡，将接口板卡插入计算机的 ISA 插槽，建立起计算机与仪器设备之间的通信渠道。

2．确定所选择的接口卡是否具有设备驱动程序

接口卡的设备驱动程序是控制各种硬件接口的驱动程序，是连接主控计算机与仪器设备的纽带。如果有设备驱动程序，则看它适合于何种操作系统；如果没有，或者所带的设备驱动程序不符合用户所用的操作系统，那么用户就有必要针对所用接口卡编写设备驱动程序。

3．确定应用程序的编程语言

如果用户有专业的图形化编程软件，如 LabVIEW、LabWindows/CVI 等，那么就可以采用它们进行编程了。若没有此类软件，则可以采用通用编程语言，如 Microsoft 公司的 Visual Basic。

4．编写用户的应用程序

在硬件连接无误的情况下，编写用户的应用程序。可根据仪器的功能，确定应用程序所采用的算法、处理分析方法和显示方式。

同其他应用程序一样，虚拟仪器软件的设计也要经历需求分析、总体设计、模块设计、代码编写、总体测试的过程。

5．调试运行应用程序

用数据或仿真的方法验证仪器功能的正确性，调试并运行仪器。

1.3 虚拟仪器的软件结构与开发平台

虚拟仪器的核心就是仪器功能的软件化，就是利用计算机的软件和硬件资源，使本来需要硬件或电路实现的技术软件化和虚拟化，最大限度地降低系统成本，增强系统的功能与灵活性。

1.3.1 虚拟仪器的软件结构

虚拟仪器的软件结构如图 1-3 所示。

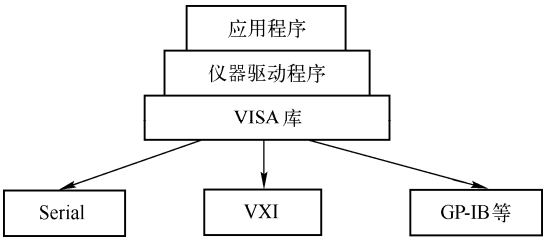


图 1-3　虚拟仪器的软件结构

从底层到顶层，虚拟仪器的软件结构包括三个部分：VISA 库、仪器驱动程序、应用程序。

1．VISA 库

虚拟仪器软件结构，其实质就是标准的 I/O 函数库及其相关规范的总称。一般称这个 I/O 函数库为 VISA 库。它驻留于计算机系统之中，执行仪器总线的特殊功能，是计算机与仪器之间的软件层连接，以实现对仪器的程控。对于仪器驱动程序开发者来说，VISA 库是一个可调用的操作函数集。

2．仪器驱动程序

设备驱动程序是完成对某一特定设备的控制与通信的软件程序集合，是应用程序实现设备控制的桥梁。每个设备都有自己的驱动程序，由仪器厂商以源码的形式提供给用户，用户在应用程序中调用仪器驱动程序。仪器驱动程序主要用来初始化虚拟仪器，设置特定的参数和工作方式，使虚拟仪器保持正常的工作状态。用户在设计应用程序时需调用仪器驱动程序。

对于市场上的大多数计算机内置插卡，厂家都配备了相应的仪器驱动程序。用户在编制应用程序时，可以像调用系统函数那样，直接调用仪器驱动程序，进行仪器操作。如果所用计算机内置插卡和外设插卡没有仪器驱动程序，用户也可以采用高级语言自己编写。

3．应用程序

应用程序建立在仪器驱动程序之上，直接面对操作用户，并提供直观、友好的操作界面，丰富的数据分析与处理功能来完成自动测试任务。

应用程序包含两个方面的程序。

1）实现虚拟面板功能的前面板软件程序。对于每个虚拟仪器模块来说，必须提供一个虚拟仪器面板。在系统集成初始化时，软面板既可用于实现仪器功能，又能帮助用户理解和熟悉仪器特性。软面板是一个可独立运行的 Windows 应用程序。

2）定义测试功能的流程图软件程序。应用软件直接面对操作用户，通过提供直观、友好的操作界面，丰富的数据分析与处理功能，来完成虚拟仪器的测试功能。它体现了虚拟仪器的优点和本质。用户可方便、直观地对应用程序进行后期开发。

1.3.2　虚拟仪器的开发平台

虚拟仪器的软件开发平台目前主要有两类。

第一类是基于传统语言（如 C、Visual Basic、Visual C++等）的通用的软件开发平台。这类语言具有适应面广、开发灵活的特点。但这种开发方式对测试人员的要求很高，需要自己将各种数据处理方法用计算机语言实现，还要求测试人员对用于数据通信的各种连接总线

（如 RS-232、GPIB、USB 等）非常熟悉。对于这些要求，绝大多数测试工程人员难以做到，或者需要花费大量的时间来研究，而懂得这些编程方法的人员又不一定懂得测试。因此，用这种平台开发测试工程软件难度大，周期长，费用高，可扩展性差。

从实现虚拟仪器功能的角度出发，开发虚拟仪器软件的平台应提供以下功能。

1）直观、丰富的仪器图形控件。由于虚拟仪器是用图形化的界面来模拟传统仪器的控制面板等交互部件，因此开发平台必须预置种类丰富的图形化控件，供软件开发者使用。

2）强大的数据处理功能。虚拟仪器的优点之一就是能利用 PC 强大的处理能力对被测信号进行数据处理、频谱分析等。因此，开发虚拟仪器的软件平台应提供大量的数据处理功能模块供开发者调用。

3）友好的人机界面。虚拟仪器的测试结果应具备按照用户的要求，以直观、友好的图形化方式显示、输出的能力，相应的开发平台也应该提供便捷的方式来实现这一目标。

从以上的分析可以看出，通用的软件开发平台无法满足虚拟仪器开发的全部要求。

因此，虚拟仪器的主导公司纷纷推出了专为虚拟仪器开发而设计的第二类虚拟仪器软件开发平台，即图形化的编程软件。这类软件通过建立和连接图标来构成虚拟仪器工作程序并定义其功能，而不是用传统的文本编辑形式。它们具有编程效率高，通用性强，交叉平台互换性好的特点，适用于大批量多品种仪器的生产。

作为测试工程领域的强有力工具，近年来，美国国家仪器公司（NI）开发的虚拟仪器软件 LabVIEW 得到了业界的普遍认可，在测试系统分析、设计和研究方面得到广泛应用。

LabVIEW 的全称是实验室虚拟仪器工程平台（Laboratory Virtual Instrument Engineering Workbench），是一种基于 G 语言（Graphics Language，图形化编程语言）的测试系统软件开发平台。它采用了工程人员熟悉的术语、图标等图形化符号来代替常规基于文字的语言程序。它把复杂、烦琐、费时的语言编程简化成完成某些功能的图标，并用线条把各种功能图标连接起来的简单图形编程方式。利用 LabVIEW，用户可通过定义和连接代表各种功能模块的图标，方便迅速地创建虚拟仪器。

第 2 章　LabVIEW 程序设计基础

本章作为 LabVIEW 程序设计的入门，介绍了 LabVIEW 的特点及应用、LabVIEW 的编程环境，LabVIEW 编程的基本概念及 LabVIEW 程序的前面板设计，然后通过实训帮助读者掌握 LabVIEW 设计虚拟仪器（VI）的方法和步骤，最后介绍了 LabVIEW 程序的调试方法。

2.1　LabVIEW 的特点及应用

2.1.1　LabVIEW 的特点

LabVIEW 包括控制与仿真、高级数字信号处理、统计过程控制、模糊控制和 PID 控制等众多附加软件包，是运行于 Windows、Linux、Macintosh 等多种平台的工业标准软件开发环境。

LabVIEW 程序又称为虚拟仪器，它的表现形式和功能类似于实际的仪器，但 LabVIEW 程序的设置和功能很容易改变。因此，LabVIEW 特别适用于实验室、多品种小批量的生产线等需要经常改变仪器参数和功能以及对信号进行分析、研究、传输等场合。

与传统的编程语言比较，LabVIEW 图形编程方式能够节省程序开发时间，其运行速度却几乎不受影响，体现出了极高的效率。

由于采用了图形化编程语言，LabVIEW 产生的程序是框图的形式，易学易用，特别适合硬件工程师、实验室技术人员、生产线工艺技术人员的学习和使用。人们可以在很短的时间内掌握并应用到实际中去。

总之，LabVIEW 能够为用户提供简明、直观、易用的图形编程方式，十分省时简便，深受用户青睐。

2.1.2　LabVIEW 的应用

LabVIEW 在包括航空、航天、通信、汽车、半导体和生物医学等世界范围的众多领域内得到了广泛应用，从简单的仪器控制、数据采集到尖端的测试和工业自动化，从大学实验室到工厂，从探索研究到技术集成，都有 LabVIEW 应用的成果。

1．LabVIEW 应用于测量与试验

LabVIEW 已成为测试与测量领域的工业标准，通过 GPIB、VXI、串行设备和数据采集卡可以构成实际的数据采集系统。它提供了工业界最大的仪器驱动程序库以及众多的开发工具，使复杂的测量与试验任务变得简单易行。

2．LabVIEW 应用于过程控制与工业自动化

LabVIEW 强大的硬件驱动能力、图形显示能力和便捷的快速程序设计为过程控制和工业自动化应用提供了优秀的解决方案。

3．LabVIEW 应用于实验室研究与计算分析

LabVIEW 为科学家和工程师提供了功能强大的高级数学分析库，涉及统计、估计、回归分析、线性代数、信号生成算法、时域和频域算法等众多科学领域，可满足各种计算和分析需要。

因此，许多工科大学已将 LabVIEW 作为课堂或实验室教学内容，作为工程师素质培养的一个方面。不同领域的科学家和工程师都借助这个易用的软件包来解决工作中的各种应用课题。

2.2 LabVIEW 的编程环境

2.2.1 启动窗口

安装 LabVIEW 2015 后，在 Windows 开始菜单中便会自动生成启动 LabVIEW 2015 的快捷方式—National Instruments LabVIEW 2015。单击该快捷方式启动 LabVIEW，启动后的窗口如图 2-1 所示。

图 2-1　LabVIEW 2015 的启动窗口

在图 2-1 的启动窗口中可选择创建项目、打开现有文件、查找驱动程序和附加文件、社区和支持，同时还可查看 LabVIEW 新闻、使用搜索功能等。

同时，利用启动窗口的菜单和按钮等功能可以创建新 VI、选择最近打开的 LabVIEW 文件、查找范例以及打开 LabVIEW 帮助。还可查看各种信息和资源，如用户手册、帮助主题以及公司网站的各种资源。

打开现有文件或创建新文件后启动窗口就会消失。关闭所有已打开的前面板和程序框图后，启动窗口会再次出现。也可在前面板或程序框图中选择“查看”→“启动窗口”，来显示启动窗口。

在启动窗口单击“创建项目”按钮，弹出“创建项目”对话框，如图 2-2 所示。在“创建项目”对话框中用户可以选择新建空白 VI、新建空白项目、简单状态机等，并且可以打

开已有的程序。同时用户可以从这个界面获得帮助支持。

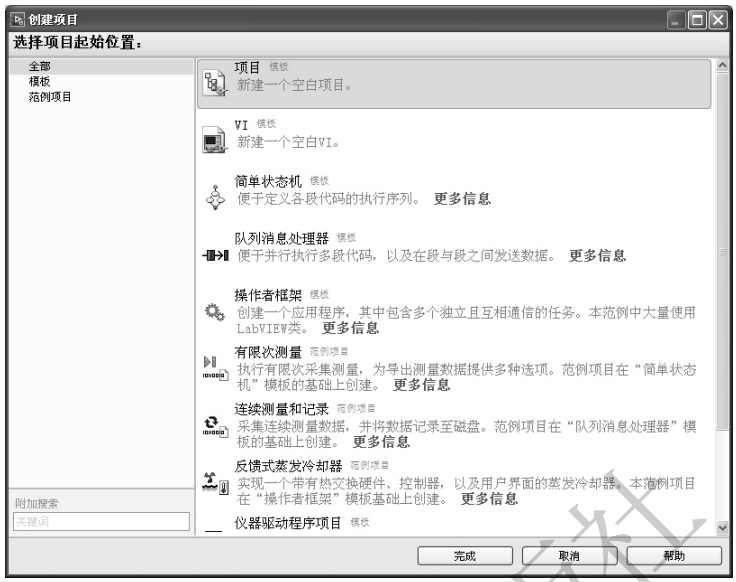


图 2-2 “创建项目”对话框

单击启动窗口中“文件”菜单下的“新建”命令，将打开如图 2-3 所示的“新建”对话框，在这里可以选择多种方式来建立文件。

图 2-3 “新建”对话框

利用“新建”对话框，可以创建三种类型的文件，分别是 VI、项目和其他文件。

其中，新建 VI 是经常使用的功能，包括新建空白 VI、创建多态 VI 以及基于模板创建 VI。如果选择新建空白 VI 方式，将创建一个空白 VI，VI 中的所有控件都需要用户自行添加；如果选择基于模板创建 VI 方式，有很多种程序模板供用户选择。

用户根据需要可以选择相应的模板进行程序设计，在各种模板中 LabVIEW 已经预先设

置了一些组件构成了应用程序的框架，用户只需对程序进行一定程度的修改和功能上的增减就可以在模板基础上构建自己的应用程序。

新建项目包括空白项目和基于向导的项目。

其他文件则包括库、类、全局变量、运行时菜单以及自定义控件等。

2.2.2 菜单栏

当用户新建一个空白 VI 后就进入 LabVIEW 的编程环境，这时将出现两个无标题窗口。一个是前面板窗口（如图 2-4 所示），用于编辑和显示前面板对象；另一个是程序框图窗口（如图 2-5 所示），用于编辑和显示流程图（程序代码）。

图 2-4 LabVIEW 的前面板窗口

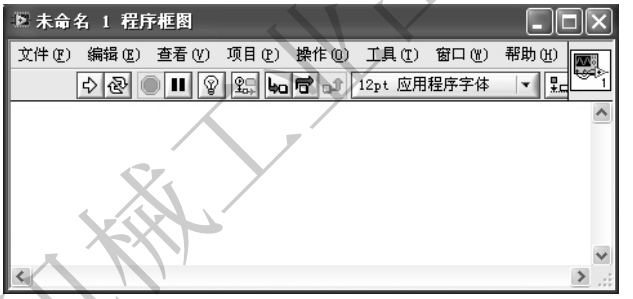


图 2-5 LabVIEW 的程序框图窗口

两个窗口拥有相同的菜单栏：包括文件、编辑、查看、项目、操作、工具、窗口、帮助 8 大项。

1.“文件”菜单

“文件”菜单包括了对程序（即 VI）操作的所有命令。

1）“新建 VI”：用于新建一个空白的 VI。

2）“新建”：用于打开“创建项目”对话框，新建空白的 VI、根据选板创建 VI 或者创建其他类型的 VI。

3）“打开”：用于打开一个已存在的 VI。

4）“关闭”：用于关闭当前 VI。

5）“关闭全部”：关闭打开的所有 VI。

6）“保存”：保存当前编辑过的 VI。

7）“另存为”：另存为其他 VI。

8）“保存全部”：保存所有修改过的 VI，包括子 VI。

9）“保存为前期版本”：为了能在前期版本中打开现在所编写的程序，可以保存为前期版本的 VI。

10）“创建项目”：新建工程文件。

11）“打开项目”：打开工程文件。

12）“页面设置”：用于设置当前 VI 的一些打印参数。

13）“打印”：打印当前 VI。

14）“VI 属性”：用于查看和设置当前 VI 的一些属性。

15）“近期项目”：用于快速打开曾经打开过的工程。

16）“近期文件”：用于快速打开曾经打开过的 VI。

17）“退出”：用于退出 LabVIEW 编程环境。

2．“编辑”菜单

“编辑”菜单中列出了所有对 VI 及其组件进行编辑的命令。

1）“撤销”：用于撤销上一步操作，回到上一次编辑之前的状态。

2）“重做”：执行与撤销相反的操作，执行该命令时，可以恢复最近“撤销”所做的修改。

3）“剪切”：删除选定的文本、控件或者其他对象，并将其放到剪贴板中。

4）“复制”：用于将选定的文本、控件或者其他对象复制到剪贴板中。

5）“粘贴”：用于将剪贴板中的文本、控件或者其他对象从剪贴板中放到当前光标位置。

6）“删除”：用于删除当前选定的文本、控件或者其他对象，与剪切不同的是，删除时不会把这些对象放入剪贴板中。

7）“选择全部”：选择全部对象。

8）“当前值设置默认值”：将当前前面板上对象的取值设为该对象的默认值，这样当下一次打开该 VI 时，该对象将被赋予该默认值。

9）“重新初始化为默认值”：将前面板上对象的取值初始化为原来的默认值。

10）“自定义控件”：用于定制前面板中的控件。

11）“导入图片至剪贴板”：将文件中图片导入至剪贴板。

12）“设置 Tab 键顺序”：当用 Tab 键切换前面板上对象的顺序时，可以用该命令进行设置。

13）“删除断线”：用于除去 VI 程序框图中由于连线不当造成的断线。

14）“创建子 VI”：用于创建一个子 VI。

15）“VI 修订历史”：用于记录 VI 的修订历史。

16）“运行时菜单”：用于设置程序运行时的菜单项。

17）“查找和替换”：搜索和替换对象。

3．“查看”菜单

“查看”菜单包括了程序中所有与显示操作有关的命令。

1）“控件选板”：用于显示 LabVIEW 的控件选板。

2）“函数选板”：用于显示 LabVIEW 的函数选板。

3）“工具选板”：用于显示 LabVIEW 的工具选板。

4）“快速放置”：用于显示快速放置对话框，可依据名称指定选板对象，并将对象置于程序框图或前面板。

5）“断点管理器”：用于显示断点管理器窗口，该窗口用于在 VI 的层次结构中启用、禁用或清除全部断点。

6）“探针检测窗口”：用于打开探针检测窗口。右击程序框图中的连线，在快捷菜单中选择“探针”或“使用探针工具”，可显示该窗口。

7）“错误列表”：用于显示 VI 的错误。

8）“加载并保存警告列表”：显示“加载并保存警告”对话框，通过该对话框可查看要加载或保存项目警告的详细信息。

9）“VI 层次结构”：显示 VI 的层次结构，用于显示该 VI 与其调用的子 VI 之间的层次关系。

10）“浏览关系”：用于浏览程序中所使用的所有 VI 之间的相对关系。

11）“启动窗口”：打开 LabVIEW 的启动窗口。

12）“导航窗口”：用于显示 VI 的导航窗口。

13）“工具栏”：用于显示工具栏选项。

4.“项目”菜单

“项目”菜单包含了 LabVIEW 中所有与项目操作相关的命令。

1）“创建项目”：用于新建一个项目文件。

2）“打开项目”：用于打开一个已有的项目文件。

3）“保存项目”：用于保存一个项目文件。

4）“关闭项目”：用于关闭项目文件。

5）“添加至项目”：将 VI 或者其他文件添加到现有的项目文件中。

6）“文件信息”：显示当前项目的信息。

7）“解决冲突”：打开“解决项目冲突”对话框，可通过“重命名冲突项”，或“使冲突项从正确的路径重新调用依赖项”解决冲突。

8）“属性”：显示当前项目属性。

5.“操作”菜单

“操作”菜单包括了对 VI 操作的基本命令。

1）“运行”：用于运行 VI。

2）“停止”：用于中止 VI 的运行。

3）“单步步入”：单步执行以进入程序单元。

4）“单步步过”：单步执行以完成程序单元。

5）“单步步出”：单步执行以跳出程序单元。

6）“调用时挂起”：当 VI 被调用时，挂起程序。

7）“结束时打印”：在 VI 运行结束后打印该 VI。

8）“结束时记录”：在 VI 运行结束后将运行结果记录到记录文件。

9）“数据记录”：用于打开它的下级菜单以设置记录文件的路径等。

10）“切换至运行模式”：用于将 LabVIEW 在运行模式和编辑模式间切换。

11）“连接远程前面板”：用于在“远程面板”对话框中设置远程的 VI 连接、通信。

12）“调试应用程序或共享库”：对应用程序或共享库进行调试。

6.“工具”菜单

“工具”菜单包括编写程序时所用到的工具，包括一些主要工具和辅助工具。

1）“Measurement & Automation Explorer”：用于打开 MAX 程序。

2）“仪器”：在“仪器”的子菜单中可以选择“连接 NI 的仪器驱动网络”或者“导入 CVI 仪器驱动程序”。

3）“性能分析”：对 VI 的性能即占用资源的情况进行比较。

4）“安全”：对用户所编写的程序进行保护，如设置密码等。

5）“用户名”：用于设置用户的姓名。

6）“通过 VI 生成应用程序”：用于在“通过 VI 生成应用程序”对话框中通过所打开的 VI 生成独立的应用程序。

7）“LLB 管理器”：打开库文件管理器。

8）“导入”：用来向当前程序导入.net 控件、ActiveX 控件、共享库等。

9）“共享变量”：包含共享变量函数。

10）“在磁盘上查找 VI”：用来搜索磁盘上指定路径下的 VI。

11）“NI 范例管理器”：用于查找 NI 为用户提供的各种范例。

12）“远程前面板管理器”：用于管理远程 VI 的远程连接。

13）“Web 发布工具”：用于打开网络发布工具管理器窗口，对通过网络访问用户的 VI 进行设置。

14）“高级”：它的子菜单里是一些对 VI 操作的高级工具。

15）“选项”：用于设置 LabVIEW 以及 VI 的一些属性和参数。

7．“窗口”菜单

利用“窗口”菜单可以打开 LabVIEW 程序的各种窗口，例如前面板窗口、程序框图窗口以及导航窗口。

1）“显示前面板/显示程序框图”：用于在程序框图和前面板间切换。

2）“左右两栏显示”：用于将 VI 的前面板和程序框图进行左右（即横向）排布。

3）“上下两栏显示”：用于将 VI 的前面板、程序框图进行上下（即纵向）排布。

另外，在“窗口”菜单的最下方显示了当前打开的所有 VI 的前面板和程序框图，因而可以从这里直接进入那些 VI 的前面板或程序框图。

8．“帮助”菜单

LabVIEW 提供了功能强大的帮助功能，这集中体现在它的帮助菜单上。

1）“显示即时帮助”：选择是否显示即时帮助窗口以获得即时帮助。

2）“锁定即时帮助”：用于锁定即时帮助窗口。

3）“查找范例”：用于查找 LabVIEW 中自带的所有例程。

4）“网络资源”：打开 NI 公司的官方网站，在网络上查找 LabVIEW 程序的帮助信息。

5）“专利信息”：显示 NI 公司的所有相关专利。

6）“关于 LabVIEW”：显示 LabVIEW 的相关信息。

2.2.3 工具栏

工具栏按钮包括运行、中断、终止、调试 VI、修改字体、对齐、组合、分布对象等。

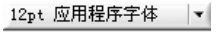
1．前面板工具栏

前面板窗口和程序框图窗口都有各自的工具栏，工具栏包括用于控制 VI 的命令按钮。图 2-6 是前面板窗口中的工具栏。

图 2-6　前面板工具栏

表 2-1 为前面板窗口的工具栏中各按钮的作用。

表 2-1　前面板窗口的工具栏中各按钮的功能简介

图　标	名　称	功　能
	“运行”按钮	单击它可以运行 VI。注意“运行”按钮的图案变化，按钮不同的形状表示了 VI 的运行属性（正常运行、警告错误）
	“连续运行”按钮	单击该按钮，可连续地重复执行 VI，再次单击该按钮可以停止程序的连续运行
	“终止执行”按钮	用于立即停止程序的运行。注意：使用该按钮来停止 VI 的运行，是强制性的停止，可能会错过一些有用的信息
	“暂停/继续”按钮	单击该按钮可使 VI 暂停执行，再次单击可使 VI 继续执行
	“文本设置”按钮	单击该按钮将弹出一个下拉列表，从中可以设置字体的格式，如字体类型、大小、形状和颜色等
	“对齐对象”按钮	首先选定需要对齐的对象，然后单击该按钮，将弹出一个下拉列表，该列表可以设置选定对象的对齐方式，如竖直对齐、上边对齐、左边对齐等
	“分布对象”按钮	首先选定需要排列的对象，然后单击该按钮，将弹出一个下拉列表，从中可以设置选定对象的排列方式，如间距、紧缩等
	“调整对象大小”按钮	首先选定需要调整大小的对象，然后单击该按钮，将弹出一个下拉列表，从中可以设置对象的最大、最小宽度、高度等
	“重新排序”按钮	当几个对象重叠时，可以重新排列每个对象的叠放次序，如前移、后移等

2．程序框图工具栏

程序框图窗口的工具栏按钮大多数与前面板工具栏中的按钮相同，另外还增加了 4 个调试按钮。程序框图窗口的工具栏如图 2-7 所示。

图 2-7　程序框图工具栏

表 2-2 为 4 个调试按钮的作用。

表 2-2　程序框图窗口的工具栏中调试按钮的功能简介

图标	名　　称	功　　能
	“高亮显示程序执行过程”按钮	单击该按钮，VI 以一种缓慢的节奏一步一步地执行，所执行到的节点都以高亮方式显示，并显示 VI 运行时的数据流动。这样用户可以清楚地了解到程序的运行过程，也可以很方便地查找错误。再次单击该按钮，可以停止 VI 的这种执行方式，恢复到原来的执行方式
	“开始单步（入）执行”按钮	单击此按钮，程序将以单步方式运行，如果节点为一个子程序或结构，则进入子程序或结构内部执行单步运行方式
	“开始单步（跳）执行”按钮	也是一种单步执行的按钮，它是以一个节点为执行单位，即单击一次按钮以执行一个节点。如果节点为一个子程序或结构，可作为一个执行单位，一次执行完该节点，然后转到下一个节点，而不会进入节点内部执行。闪烁的节点表示该节点处于等待执行状态
	“单步步出”按钮	当在一个节点（如子程序或结构）内部执行单步运行方式时，单击该按钮可一次执行完该节点，并直接跳出该节点转到下一个节点

2.2.4 操作选板

LabVIEW 中的操作选板分为工具选板、控件选板和函数选板，LabVIEW 程序的创建主要依靠这三个选板完成。

工具选板提供了用于创建、修改和调试程序的基本工具；控件选板涵盖了各种输入控件和显示控件，主要用于创建前面板中的对象，构建程序的界面；函数选板包含了程序编写过程中用到的函数和 VI，主要用于构建程序框图中的对象。控件选板和函数选板中的对象被分门别类地安排在不同的子选板中。

一般在启动 LabVIEW 的时候，三个选板会出现在屏幕上，由于控件选板只对前面板有效，所以只有在激活前面板的时候才会显示。同样，只有在激活程序框图的时候才会显示函数选板。如果选板没有被显示出来，可以通过选择菜单“查看→工具选板”来显示工具选板，通过菜单“查看→控件选板”显示控件选板，通过菜单“查看→函数选板”显示函数选板。也可以在窗口的空白处右击以弹出控件选板或函数选板。

图 2-8　工具选板

1．编辑工具——工具选板

在前面板和程序框图中都可看到工具选板，LabVIEW 的工具选板如图 2-8 所示。利用工具选板可以创建、修改 LabVIEW 中的对象，并对程序进行调试。工具选板是 LabVIEW 中对对象进行编辑的工具。

工具选板中各种工具的图标、名称及其相应的功能如表 2-3 所示。

表 2-3　工具选板各工具功能简介

图　标	名　称	功　能
	“自动选择工具”按钮	按下该按钮，将光标移到前面板或程序框图的对象上时，系统会自动选择工具选板中相应的工具，方便用户操作。当用户选择手动时，需要手动选择工具选板中的相应工具
	“操作值工具”按钮	用于改变控件值
	“定位/调整大小/选择工具”按钮	用于选取对象，改变对象的位置和大小
	“编辑文本工具”按钮	用于输入标签文本或者创建标签
	“进行连线工具”按钮	用于在程序框图中连接两个对象的数据端口，当使用“进行连线工具”方式接近对象时，会显示出其数据端口以供连线之用。如果打开了帮助窗口，那么把进行连线工具置于某连线上时，会在帮助窗口显示其数据类型
	“对象快捷菜单工具”按钮	当用该工具单击某对象时，会弹出该对象的快捷菜单
	“滚动窗口工具”按钮	使用该工具时，不需要滚动条就可以自由滚动整个图形
	“设置/清除断点工具”按钮	用于在调试程序过程中设置断点
	“探针数据工具”按钮	用于在代码中加入探针，以便调试程序过程中监视数据的变化
	“获取颜色工具”按钮	用于从当前窗口中提取颜色
	“设置颜色工具”按钮	用于设置窗口中对象的前景色和背景色

2．前面板设计工具——控件选板

控件选板仅位于前面板，包括了用于创建前面板对象所需的输入控件和显示控件，是用户设计前面板的工具。输入控件是指按钮、旋钮、转盘等输入装置，用来模拟仪器的输入，为 VI 的程序框图提供数据；显示控件是指图表、指示灯等显示装置，用来模拟仪器的输出，显示程序框图获取或生成的数据。

LabVIEW 2015 中的控件选板如图 2-9 所示。

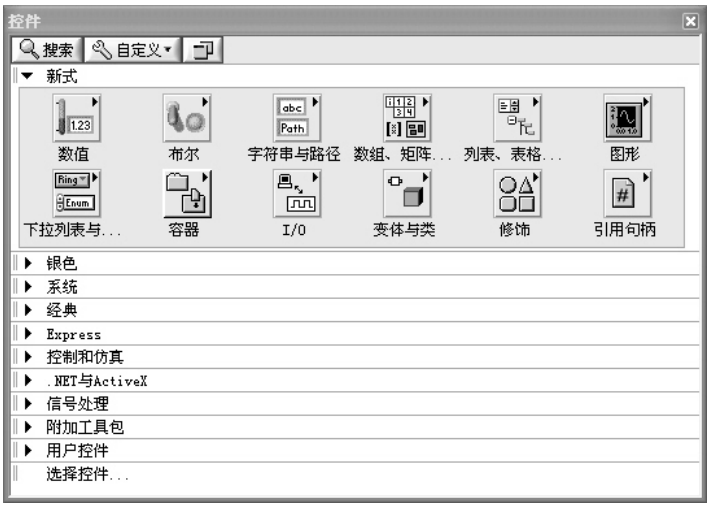


图 2-9　控件选板

在控件选板中，按照所属类别，各种输入控件和显示控件被分门别类地安排在不同的子选板中。应用控件选板中的这些子选板，用户可以创建出界面美观且功能强大的 VI 前面板。

控件选板中常用子选板的图标、名称和功能如表 2-4 所示。

表 2-4　控件选板中常用子选板的功能简介

图　　标	子选板名称	功　　能
数值	“数值”	用于输入和显示数值，用于设计具有数值数据类型属性的输入控件和显示控件，如滑动杆、旋钮、滚动条、转盘和数值显示框等
布尔	“布尔”	用于输入并显示布尔值，用于设计具有布尔数据类型属性的输入控件和显示控件，如按钮、开关、指示灯等
字符串与路径	“字符串与路径”	其中的字符串控件用于输入或编辑前面板上的文本或标签，路径控件用于输入或返回文件或目录的地址
数组、矩阵...	“数组、矩阵和簇”	用于作为数组、矩阵和簇类型数据的输入和显示
列表、表格...	“列表、表格和树”	用于表格形式数据的输入和显示，如列表框、多列列表框、树型列表框、表格等
图形	“图形”	用于将数据以图形方式显示，如波形图、曲线图、密度图以及各种三维曲面及曲线等显示对象
下拉列表与...	“下拉列表与枚举”	其中的下拉列表控件是将数值与字符串或图片建立关联的数值对象，以下拉列表的形式出现，供用户在循环浏览的过程中作出选择； 枚举控件用于向用户提供一个可供选择的列表
容器	“容器”	用于组合控件，或在当前 VI 的前面板上显示另一个 VI 的前面板；还可用于在前面板上显示.NET 和 ActiveX 对象
I/O	“I/O”	可将所配置的 DAQ 通道名称、VISA 资源名称和 IVI 逻辑名称传递至 I/O 的 VI，使其与仪器或 DAQ 设备通信
修饰	“修饰”	用于前面板界面的设计和装饰，如产生用于装饰界面的框和线条等
引用句柄	“引用句柄”	可用于对文件、目录、设备和网络连接进行操作

3．程序框图设计工具——函数选板

函数选板仅位于程序框图，包含了程序编写过程中用到的函数和 VI，主要用于构建程

序框图中的节点，对 VI 程序框图进行设计。LabVIEW 2015 的函数选板如图 2-10 所示。按照其功能类型将各种函数、VIs 和 Express VIs 放入不同的子选板中。

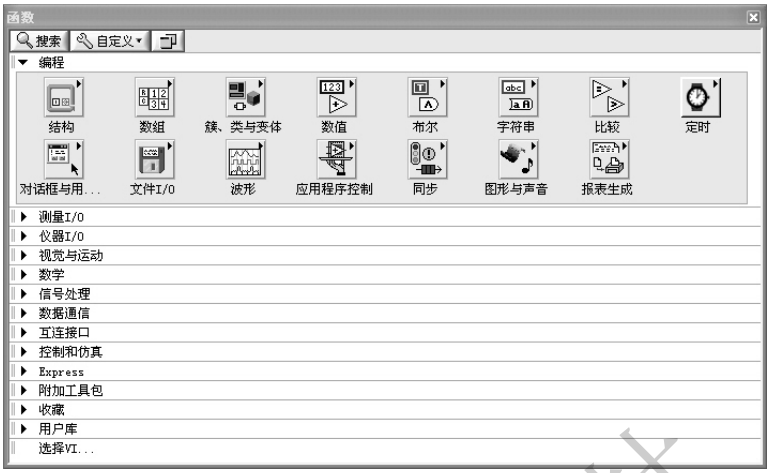


图 2-10　函数选板

函数选板中常用子选板的图标、名称和功能如表 2-5 所示。

表 2-5　函数选板中常用子选板的功能简介

图　标	子选板名称	功　能
结构	“结构”	用于设计程序的顺序、分支和循环等结构（如顺序结构、条件结构、While 循环、For 循环、事件结构）、公式节点、全局变量、局部变量、反馈节点等
数组	“数组”	用于创建数组和对数组进行操作，如设置数组大小、将元素插入数组、从数组中删除元素、初始化数组等
簇与变体	“簇与变体”	用于创建簇和对簇进行操作，如捆绑和创建簇数组；将簇转换为数组和簇常量等
数值	“数值”	包括数值类型转换函数、三角函数、对数函数、复数函数、算术运算、数值常数、表达式节点、数值分析等
布尔	“布尔”	用于进行布尔型数据的运算，如逻辑运算；包含布尔型常数、布尔量与数值的转换函数等
字符串	“字符串”	用于对字符串型数据的操作，包含对字符串操作的各种函数，字符串与数值、数组和路径的转换函数，字符串常量，构建串的快速 VI 字符等
比较	“比较”	用于比较布尔型、数值型、字符串型以及簇和数组型数据，包含各种比较运算函数等
定时	“定时”	用于控制程序执行的速度，包含各种定时函数和时间转换函数
对话框与用...	“对话框与用户界面”	用于创建各种按钮对话框、提示对话框、显示对话框以及建立菜单、帮助、事件等
文件I/O	“文件 I/O”	用于创建、打开、读取及写入文件等，包括各种操作函数，对路径进行操作的各种函数
波形	“波形”	用于进行和波形有关的操作，如各种波形显示函数和快速 VI 等
应用程序控制	“应用程序控制”	打开与关闭应用程序，包含程序的停止和退出等程序控制函数
图形与声音	“图形与声音”	用于创建图形，从图形文件获取数据，对声音信息进行处理等； 包含各种图形图像显示、声音播放等函数。
报表生成	“报表生成”	用于创建和控制应用程序报表，包含简易选板，文本报表、新建报表、打印报表等。注：进入报表生成函数可以看到简易文本报表等函数。

函数选板是编写 VI 时使用最为频繁的工具，因而熟悉它的各个子选板的功能对编写程序是十分有用的，在使用 LabVIEW 编写程序的过程中，读者可以逐步了解它的每个子选板，包括每个函数、VIs 以及 Express VIs 的功能，熟练使用这些工具是编写好 LabVIEW 应用程序的保证。

2.3 LabVIEW 编程的基本概念

LabVIEW 是一个功能完整的程序设计语言，具有区别于其他程序设计语言的一些独特结构和语法规则。

应用 LabVIEW 编程的关键是掌握 LabVIEW 的基本概念和图形化编程的基本思想。

2.3.1 VI 与子 VI

用 LabVIEW 开发的应用程序称为 VI（Virtual Instrument，虚拟仪器）。

一个最基本的 VI 是由节点、端口以及连线组成的应用程序。

VI 运行采用数据流驱动，具有顺序、循环、条件等多种程序结构控制。

在 LabVIEW 中的子程序被称做子 VI（SubVI）。在程序中使用子 VI 有以下优点。

1）将一些代码封装成为一个子 VI（即一个图标或节点），可以使程序的结构变得更加清晰、明了。

2）将整个程序划分为若干模块，每个模块用一个或者几个子 VI 实现，易于程序的编写和维护。

3）将一些常用的功能编制成为一个子 VI，在需要的时候可以直接调用，不用重新编写这部分程序，因而子 VI 有利于代码复用。

正因为子 VI 的使用对编写 LabVIEW 程序有很多益处，所以在使用 LabVIEW 编写程序的时候经常会使用子 VI。

子 VI 由 3 部分组成，除前面板对象、程序框图外，还有图标的连接端口。连接端口的功能是与调用它的 V1 交换数据。

基于 LabVIEW 图形化编程语言的特点，在 LabVIEW 环境中，子 VI 也是以图标（节点）的形式出现的。在使用子 VI 时，首先需要定义其数据输入和输出的端口，然后就可以将其当做一个普通的 VI 来使用。

因此在使用 LabVIEW 编程时，应与其他编程语言一样，尽量采用模块化编程的思想，有效地利用子 VI，简化 VI 程序框图的结构，使其更加简洁，易于理解，以提高 VI 的运行效率。

2.3.2 前面板

前面板就是图形化用户界面，用于设置输入数值和观察输出量，是人机交互的窗口。由于 VI 前面板是模拟真实仪器的前面板，所以输入量称为控制，输出量称为指示。

在前面板中，用户可以使用各种图标，如仪表、按钮、开关、波形图、实时趋势图等，这可使前面板的界面像真实的仪器面板一样。

图 2-11 是一个调压器程序的前面板。

前面板对象按照功能可以分为控制、指示和修饰三种。控制是用户设置和修改 VI 程序

中输入量的接口，如图 2-11 中的调压旋钮；指示则用于显示 VI 程序产生或输出的数据，如图 2-11 中的电压表、上限灯。

如果将一个 VI 程序比作一台仪器的话，那么控制就是仪器的数据输入端口和控制开关，而指示则是仪器的显示窗口，用于显示测量结果。

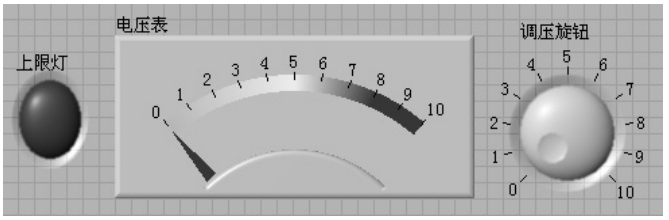


图 2-11　调压器程序的前面板

在本书中，为方便起见，将前面板中的控制和指示统称为前面板对象或控件，控制即输入控件，指示即显示控件。

修饰的作用仅是将前面板点缀得更加美观，并不能作为 VI 的输入或输出来使用。在控件选板中专门有一个修饰子选板。

2.3.3　程序框图

每一个前面板都有一个程序框图与之对应。上述调压器的程序框图如图 2-12 所示。程序的功能是通过调压旋钮产生数值，送到电压表显示，当数值大于等于 8 时，上限灯改变颜色。

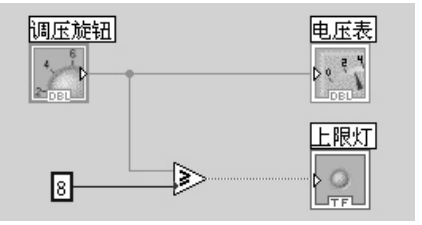


图 2-12　调压器的框图程序

程序框图用图形化编程语言编写，可以把它理解成传统编程语言程序中的源代码。用图形来进行编程，而不是用传统的代码来进行编程，这是 LabVIEW 最大的特色。

程序框图由节点、端口和连线组成。

1．节点

节点是 VI 程序中的执行元素，类似于文本编程语言程序中的语句、函数或者子程序。上述调压器的程序框图中的数值常量、比较函数就是节点。

LabVIEW 共有 4 种类型的节点，如表 2-6 所示。

表 2-6　LabVIEW 节点类型

节点类型	节点功能
功能函数	LabVIEW 内置节点，提供基本的数据与对象操作，例如，数值计算、文件 I/O 操作、字符串运算、布尔运算、比较运算等。
结构	用于控制程序执行方式的节点，包括顺序结构、条件结构、循环结构及公式节点等。
代码接口节点	LabVIEW 与 C 语言文本程序的接口。通过代码接口节点，用户可以直接调用 C 语言编写的源程序。
子 VI	将创建的 VI 以 SubVI 的形式调用，相当于传统编程语言中子程序的调用。通过功能选板中的 Select VI 子选板可以添加一个 SubVI 节点。

2．端口

节点之间、节点与前面板对象之间通过数据端口和数据连线来传递数据。

端口是数据在程序框图部分和前面板对象之间传输的通道接口，以及数据在程序框图的节点之间传输的接口。端口类似于文本程序中的参数和常数。

端口有两种类型：输入/输出端口和节点端口（即函数图标的连线端口）。输入或输出端口用于前面板，当程序运行时，从输入控件输入的数据就通过输出端口传送到程序框图。而当 VI 程序运行结束后，输出数据就通过输入端口从程序框图送回到前面板的显示控件。

当在前面板创建或删除输入控件或显示控件时，可以自动创建或删除相应的输出/输入端口。

一般情况下，LabVIEW 中的每个节点至少有一个端口，用于向其他节点或图标传递数据。

3．连线

节点之间由数据连线按照一定的逻辑关系相互连接，以确定程序框图内的数据流动方向。

连线是端口间的数据通道，类似于文本程序中的赋值语句。数据是单向流动的，从源端口向一个或多个目的端口流动。

不同的线型代表不同的数据类型，每种数据类型还以不同的颜色予以强调或区分。

接线头是连线的线头部分。接线头的作用是帮助端口的连线位置正确。当把连线工具放到端口上时，接线头就会弹出。接线头还有一个黄色小标识框，用于显示该端口的名字。

连接端口通常是隐藏在图标中的。图标和连接端口都是由用户在编制 VI 时根据实际需要创建的。

2.3.4 数据流驱动

由于程序框图中的数据是按照程序中的逻辑关系沿数据连线流动的，因此，LabVIEW 编程又称为“数据流编程”。“数据流”控制 LabVIEW 程序的运行方式。

对一个节点而言，只有当它的输入端口上的所有数据都被提供以后，它才能够执行下去。当节点程序运行完毕，它会把结果数据送到其输出端口中，这些数据很快通过数据连线送至与之相连的目的端口。

“数据流”与常规编程语言中的“控制流”类似，相当于控制程序语句一步一步地执行。

例如，两数相加程序的前面板如图 2-13 所示，与之对应的程序框图如图 2-14 所示，这个 VI 程序控制 a 和 b 中的数值相加，然后把相加之和乘以 100，将结果送至指示 c 中显示。

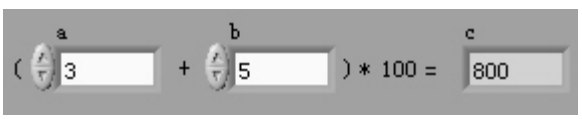


图 2-13　两数相加程序的前面板

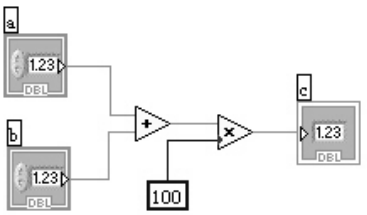


图 2-14　两数相加的程序框图

在这个程序中，程序框图从左向右执行，但这个执行次序不是由其对象的摆放位置来确定。由于相乘节点的输入量是相加节点的运算结果，因此只有当相加运算完成并把结果送到相乘运算节点的输入端口后，相乘节点才能执行下去。

2.4 VI 前面板设计

把 VI 应用程序界面称作前面板。前面板是 LabVIEW 的重要组成部分，是用 LabVIEW 编写的应用程序的界面。LabVIEW 提供非常丰富的界面控件对象，可以方便地设计出生动、直观、操作方便的用户界面。

LabVIEW 提供的专门用于前面板设计的输入和显示控件被分门别类地放置在控件选板中，当用户需要使用时，可以根据对象的类别从各个子选板中选取。前面板的对象按照其类型可以分为数值型、布尔型、字符串型、数组型、簇型、图形型等多种类型。

在用 LabVIEW 进行程序设计的过程中，对前面板的设计主要是编辑前面板控件和设置前面板控件的属性。

2.4.1 前面板对象的创建

设计应用程序界面所用到的前面板对象全部包含在控件选板中。

放置在前面板上的每一个控件都具有很多属性，其中多数与显示特征有关，在编程时就可以通过在控件上右击（即右键单击，以下同）更改其属性值。

设计前面板需要用到控件选板，用鼠标选择控件选板上的对象，然后在前面板上拖放即可。

以下举例说明前面板对象的创建过程。首先创建新的应用程序并保存为“创建对象.vi”。

切换到前面板窗口，在控件选板上单击“数值”控件子选板，选择“数值输入控件”，如图 2-15 所示，在前面板的适当位置单击，即可创建数值输入控件。修改数值输入控件的标签并输入“数字 1”。使用同样的方法可以创建数值型控件“垂直指针滑动杆”和“旋钮”，如图 2-16 所示。在程序框图窗口中会自动产生代表相应控件的图标符号，如图 2-17 所示。

图 2-15 “数值”控件子选板

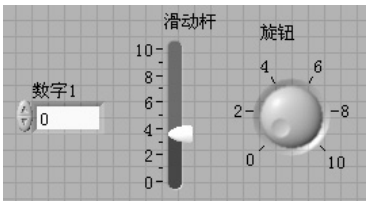


图 2-16 前面板窗口中对象的生成

图 2-17 程序框图窗口中自动生成的图标

2.4.2 前面板对象的属性配置

此处介绍的前面板对象属性的配置方法适用于输入控件和显示控件。

右击前面板对象，如滑动杆控件，弹出快捷菜单，如图 2-18 所示。这里只介绍输入控件和显示控件共有的快捷菜单部分。

1）显示项：该菜单显示一个对象可以显示/隐藏的部分，如标签、标题等。

2）查找接线端：在代码窗口中高亮显示前面板对象。当代码窗口中的对象太多时，直接寻找控件对象是非常有效的。

3）转换为显示控件/转换为输入控件：将指定的对象改变为显示控件或输入控件。

任何一个前面板对象都有输入和显示两种属性，右击前面板对象，在弹出的快捷菜单中选择“转换为显示控件”或“转换为输入控件”命令可以在输入和显示两种属性之间切换。

一般控件可以指定为显示量，也可以转化为输入量。例如右击滑动杆控件，在弹出的快捷菜单中选择“转换为显示控件”命令，该控件已经变成了显示件。该变化也同时反映到程序框图窗口中的图标上。

4）创建：针对选择的对象创建局部变量、引用和属性节点等。

5）替换：选择其他的控件来代替当前的控件。

6）数据操作：包含一个编辑数据选项的子菜单。主要包括以下选项：重新初始化为默认值和当前值设置为默认值。图 2-16 中，各个控件在设计时就已经有了默认的初始值，如果要改变这个初始值，则可在设计时给控件输入指定的数值，然后在控件上右击，在弹出的快捷菜单中选择“数据操作→当前值设置为默认值”命令，如图 2-19 所示。这样每次在程序打开时，控件就自动获得了新的默认值。

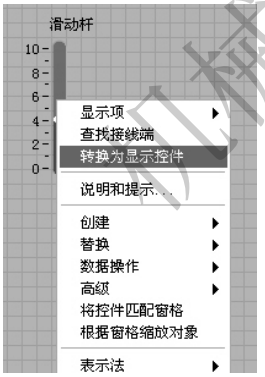


图 2-18　改变控件属性的快捷菜单

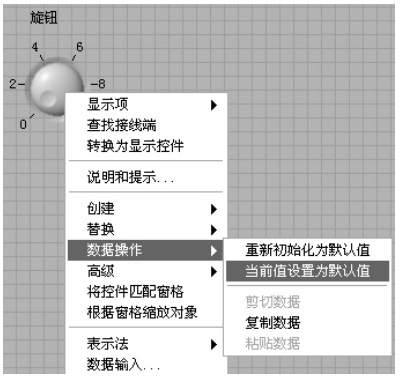


图 2-19　设置控件的默认值

7）高级：包含控件高级编辑选项的子菜单。主要包括以下选项。

快捷键：为控件分配快捷键，用户在没有鼠标的情况下仍然可以访问控件。

同步显示：控件将显示全部的更新数据，这种设置方法将影响 LabVIEW 的运行性能。

自定义：由用户定制控件，在控件编辑器中设计个性化的前面板对象。

隐藏输入控件/隐藏显示控件：在前面板中隐藏控件对象。要访问隐藏的对象，在程序

框图窗口中右击控件对象，在弹出的快捷菜单中选择“显示输入控件”或“显示显示控件”命令即可。

2.4.3 前面板对象的修饰

作为一种基于图形模式的编程语言，LabVIEW 在图形界面的设计上有着得天独厚的优势，可以设计出漂亮、大方而且方便、易用的程序界面（即程序的前面板）。为了更好地进行前面板的设计，LabVIEW 提供了丰富的修饰前面板的方法以及专门用于装饰前面板的控件，下面介绍修饰前面板的方法和技巧。

1．设置前面板对象的颜色

前景色和背景色是前面板对象的两个重要属性，合理地搭配对象的前景色和背景色会使用户的程序增色不少。一般情况下，控件选板上的对象是以默认颜色被拖放到前面板，可以通过简单的操作进行修改。

对于前面板对象的颜色的编辑，需要用到工具选板里的取色工具和颜色设置工具。

此处创建新的 VI“设置颜色.vi”。在程序的前面板创建 1 个数值量控件“液罐”，颜色等均采用默认值。

颜色设置工具为  ，图标内有前后两个调色板，分别代表前景色和背景色。分别用鼠标单击两个调色板，会出现颜色选择面板，如图 2-20 所示，以设置前景和背景的颜色。用鼠标单击颜色设置工具按钮后，再在编辑对象的适当位置上单击，则被编辑对象就被设置成指定的前景色或背景色。

另外一种简便的操作是，用鼠标单击颜色设置工具按钮  后，在被编辑对象的适当位置上右击，弹出颜色对话框，并且动态地渲染被编辑的对象，选择合适的颜色后单击，即可完成颜色的设置。

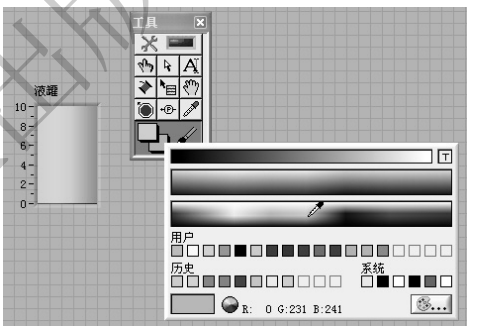


图 2-20 设置控件颜色

2．设置前面板对象的文字风格

在 LabVIEW 中，可以设置前面板文本对象的字体、颜色以及其他风格特征。这些可以通过 LabVIEW 工具栏中的字体按钮 

进行设置。单击该按钮，将弹出用于设置字体的下拉菜单，在菜单中，用户可以选择文字的字体、颜色、大小和风格。用户也可以通过字体按钮的下拉菜单打开设置字体的对话框来设置字体的常用属性。“前面板默认字体”对话框如图 2-21 所示，在这个对话框中可以设置字体的几乎所有属性。

3．调整前面板对象的位置与排列

为了提高前面板外观设计的效率，LabVIEW 提供了前面板对象编辑控制的一些工具，尤其是在界面对象比较多时，这些工具就显得尤为重要。

在 LabVIEW 程序中，设置多个对象的相对位置关系是布置和修饰前面板过程中一件非常重要的工作。在 LabVIEW 中提供了专门用于调整多个对象位置关系的工具，它们位于 LabVIEW 的工具栏上。

LabVIEW 所提供的用于修改多个对象位置关系的工具如图 2-22 所示。这两种工具分别用于调整多个对象的对齐关系以及对象之间的距离。

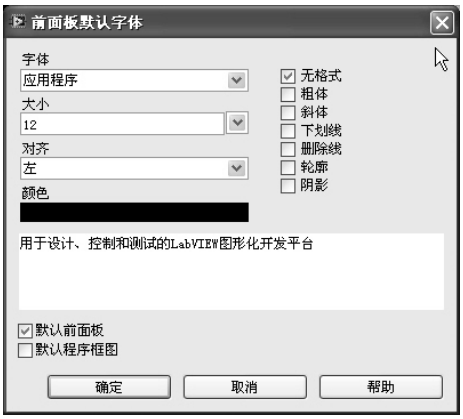


图 2-21 “前面板默认字体”对话框

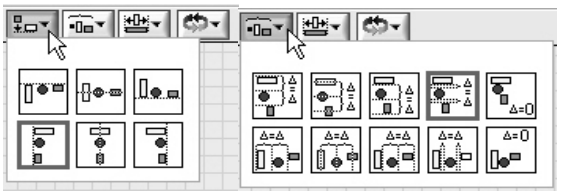


图 2-22 用于设置多个对象之间位置关系的工具

群组工具可以将一系列对象设置为一组，以固定其相对位置关系，也可以锁定对象，以免在编辑过程中对象被移动。

4．调整前面板对象的大小

一般情况下，控件选板上的对象是以默认大小被拖放到前面板的，可以通过简单的操作进行修改。

对于大小的修改，当工具选板处于自动选择状态或处于定位状态时，只需将鼠标指针移动到被编辑对象的边缘处，对象上会出现几个方框或圆框，单击并拖动方框或圆框到合适位置后松开鼠标左键，则控件对象被放大或缩小。图 2-23 所示为对数值型“液罐”控件进行缩放。

但对于特殊的控件，其编辑方式可能不尽一致，将鼠标指针切换到选择状态，然后在对象上移动，当鼠标指针的形状发生改变时，拖动即可进行缩放编辑。

在 LabVIEW 的工具栏上有设置对象大小的工具，如图 2-24 所示。

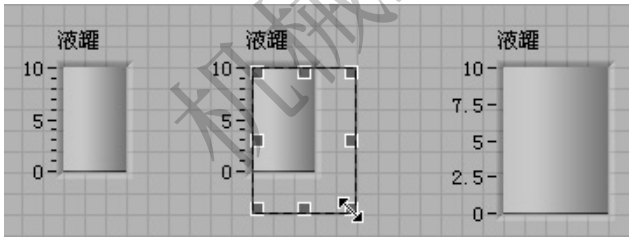


图 2-23 调整前面板对象的大小

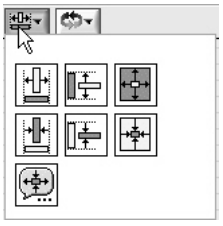


图 2-24 调整控件大小的工具

利用设置对象大小的工具，用户可以按照一定的规则调整对象的尺寸，也可以用按钮来指定控件的高度和宽度，进而设置对象的大小。

5．用修饰控件装饰前面板

LabVIEW 提供了装饰前面板上对象的设计工具，这些界面元素对程序不产生任何影响，仅仅是为了增强界面的可视化效果。它包括一系列线、箭头、方形、圆形、三角形等形状的修饰模块，这些模块如同一些搭建美观程序界面的积木，合理组织、搭配这些模块可以构造出绚丽的程序界面。

LabVIEW 中用于修饰前面板的控件位于控件选板中的“修饰”子选板中，如图 2-25 所示。

图 2-25　修饰类控件

在 LabVIEW 中，“修饰”子选板中的各种控件只有前面板图形，而没有在程序框图上与之对应的图标，这些控件的主要功能就是进行界面的修饰。

6．设置前面板对象的显示和隐藏

LabVIEW 提供的控件都具有是否可见的属性。这个属性可以在程序开发时设定，也可以在程序运行时通过代码来控制，以下举例说明。

新建应用程序，在前面板添加数值显示控件，在程序框图窗口中右击数值显示控件，在弹出的快捷菜单中选择“高级→隐藏显示控件”命令，如图 2-26 所示，数值显示控件在前面板已经不可见了。

要恢复其可见性，可切换到程序框图窗口，右击数值显示控件，在弹出的快捷菜单中选择“显示显示控件”命令，如图 2-27 所示，这时前面板窗口中出现隐藏的数值显示控件。

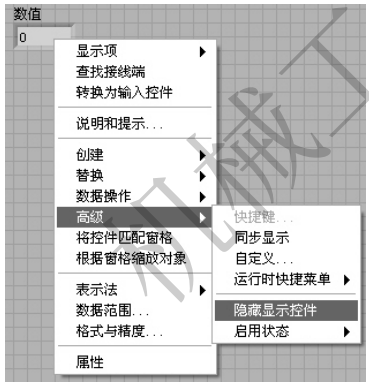


图 2-26　设计时隐藏控件

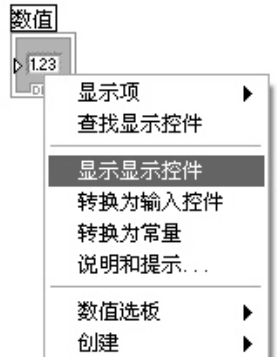


图 2-27　使隐藏的控件可见

2.5　VI 与子 VI 设计实训

实训 1　体验 VI 设计

一、学习目标

1）认识虚拟仪器软件 LabVIEW 的编程环境。

2）掌握虚拟仪器软件 LabVIEW 应用程序（VI）的设计步骤。

3）掌握虚拟仪器软件 LabVIEW 前面板和程序框图的设计方法。

二、设计任务

有一台仪器（例如电压表），需要调整其输入值（比如电压大小），当调整值（电压值）超过设定值（电压上限）时，通过指示灯颜色变化发出警告。

三、任务实现

1．建立新 VI

运行 LabVIEW2015，在启动窗口中选择“创建项目”命令，在创建项目窗口中双击“新建一个空白 VI”命令，进入 LabVIEW 的编程环境。

这时出现两个未命名窗口。一个是前面板窗口，用于编辑和显示前面板对象；另一个是程序框图窗口，用于编辑和显示流程图。

2．程序前面板设计

切换到 LabVIEW 的前面板窗口，显示控件选板，给程序前面板添加控件。

本实训中，程序前面板有一个旋钮，一个仪表，一个指示灯，共三个控件。

1）为了调整数值，向前面板添加一个旋钮控件：控件→数值→旋钮。其位置如图 2-28 所示。选择旋钮控件，将其拖动到前面板空白处单击。将标签改为“调压旋钮”。

2）为了显示数值，向前面板添加一个仪表控件：控件→数值→仪表。其位置如图 2-28 所示。选择仪表控件，将其拖动到前面板空白处后单击，将标签改为“电压表”。

图 2-28　旋钮、仪表控件位置

3）为了显示报警信息，向前面板添加一个指示灯控件：控件→布尔→圆形指示灯。其位置如图 2-29 所示。选择圆形指示灯控件，将其拖动到前面板空白处后单击，将标签改为“上限灯”。

图 2-29　圆形指示灯控件位置

控件添加完成后，可以调整控件大小和位置。设计的程序前面板如图 2-30 所示。

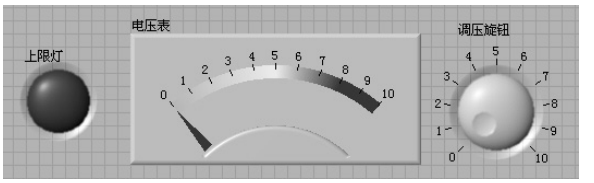


图 2-30　程序前面板

3．程序框图设计

（1）添加节点

每一个程序前面板都对应着一段程序框图。在程序框图中对 VI 进行编程，以控制和操作定义在前面板上的输入和输出对象。

切换到程序框图窗口，可以看到前面板添加的控件图标，选择这些图标，调整其位置。通过函数选板添加节点。

1）添加一个数值常量：函数→数值→数值常量。其位置如图 2-31 所示。选择“数值常量”节点，将其拖动到窗口空白处后单击，将数值设置为“8”。

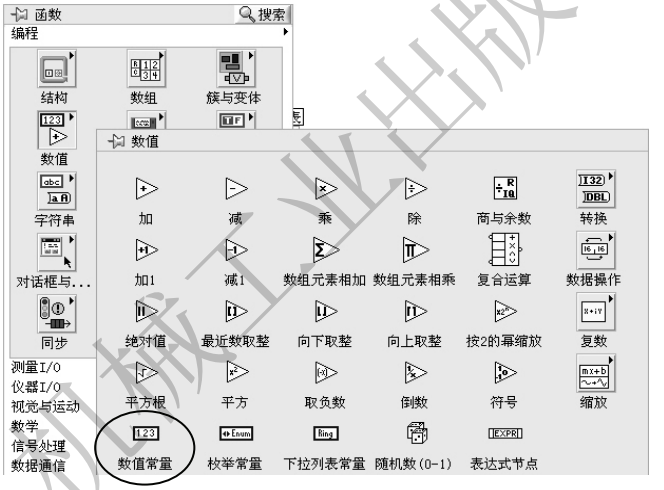


图 2-31　数值常量节点

2）添加一个比较函数“≥”：函数→比较→大于等于？。其位置如图 2-32 所示。选择“大于等于？”比较节点，将其拖动到窗口空白处后单击。右击比较节点图标，弹出快捷菜单，选择“显示项”→“标签”命令，可以看到图标上方出现标签“大于等于?”

图 2-32　大于等于？节点

添加的所有节点及其布置如图 2-33 所示。

（2）节点连线

使用工具箱中的连线工具，将所有节点连接起来。

当需要连接两个端口时，在第一个端口上选择连线工具，然后移动到另一个端口上，再单击即可实现连线。端口的先后次序不影响数据流动的方向。

当把连线工具放在节点端口上时，该端口区域将会闪烁，表示连线将会接通该端口。当把连线工具从一个端口接到另一个端口时，不需要按住鼠标键。当需要连线转弯时，单击一次鼠标键即可。

1）将“调压旋钮”控件的输出端口与“电压表”控件的输入端口相连。

2）将“调压旋钮”控件的输出端口与比较函数“大于等于？”的输入端口“x”相连。

3）将数值常量“8”与比较函数“大于等于？”的输入端口“y”相连。

4）将比较函数“大于等于？”的输出端口“x >= y?”与“上限灯”控件的输入端口相连。

连好线的程序框图如图 2-34 所示。

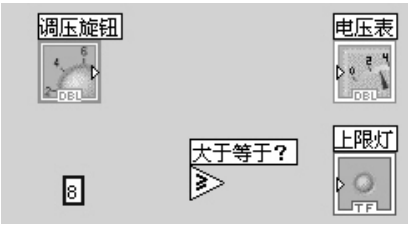


图 2-33　程序框图——节点布置图

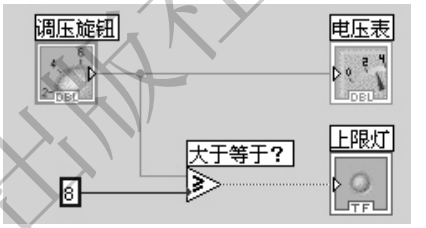


图 2-34　程序框图——节点连线图

4．运行程序

切换到前面板窗口，单击工具栏中的“连续运行”按钮，运行程序（再次单击该按钮可以停止程序的连续使用）。

程序运行时，使用鼠标左键单击转动“调压旋钮”控件，按住旋钮不放并转动，改变输入数值，可以看到“电压表”指针随着转动；当数值大于等于 8 时，“上限灯”颜色发生变化。

程序运行界面如图 2-35 所示。

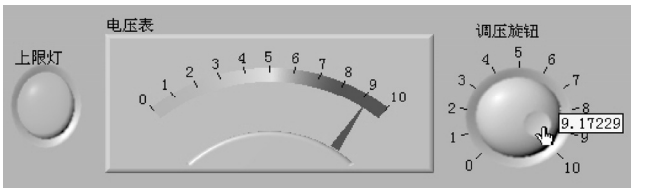


图 2-35　程序运行界面

5．保存程序

从前面板窗口中选择菜单“文件→保存”命令或者“另存为”命令，出现“命名 VI”对话框，选择文件目录，输入文件名，保存 VI。

用户既可以把 VI 作为单独的程序文件保存，也可以把一些 VI 文件同时保存在一个 VI

库中，VI 库文件的扩展名为.llb。

NI 公司推荐将程序的开发文件作为单独的程序文件保存在指定的目录下，尤其是开发小组共同开发一个项目时。

6．打开程序

从前面板窗口“文件”下拉菜单中单击“打开”子菜单可出现打开文件对话框（或在启动窗口中选择“打开”按钮）。对话框中列出了 VI 目录及库文件，每一个文件名前均带有一个图标。

打开目录或库文件后，选择想要打开的 VI 文件，单击“确定”按钮打开程序，或直接双击图标将其打开。

打开已有的 VI 还有一种较简便的方法，如果该 VI 在之前使用过，则可以在“文件→近期打开的文件”级联菜单中，找到 VI 并打开。

实训 2　子 VI 的创建与调用

一、学习目标

掌握子 VI 的创建与调用方法。

二、设计任务

1）设计一个 VI，完成两数相加（a+b=c），然后把该 VI 创建成子 VI。

2）再设计一个 VI，调用已建立的子 VI。

三、任务实现

1．子程序的创建

（1）程序前面板设计

新建 VI，切换到 LabVIEW 的前面板窗口，通过控件选板给程序前面板添加控件。

1）添加两个数值输入控件：控件→数值→数值输入控件。将标签分别改为“a”和“b”。

2）添加一个数值显示控件：控件→数值→数值显示控件。将标签改为“c”。

设计的程序前面板如图 2-36 所示。

（2）连接端口的编辑

1）右击 VI 前面板的右上角连接端口，在弹出的快捷菜单中选择“模式”命令，会出现连接端口选板，选择其中一个连接端口（本例选择的连接端口具有 2 个输入端口和 1 个输出端口），如图 2-37 所示。

图 2-36　子 VI 前面板

图 2-37　选择的连接端口

2）在工具选板中将鼠标指针切换到连线工具状态。

3）用鼠标在控件 a 上单击，选中控件 a，此时控件 a 的图形周围会出现一个虚线框。

4）将鼠标指针移动至连接端口的一个输入端口上并单击，此时这个端口就建立了与控件 a 的关联关系，端口的名称为 a，颜色变为棕色。

当其他 VI 调用这个子 VI 时，从这个连接端口输入的数据就会输入到控件 a 中，然后程序从控件 a 在程序框图中所对应的端口中将数据取出，进行相应的处理。

同样，建立数值输入控件 b 与另一个输入端口的关联关系；建立数值显示控件 c 与输出端口的关联关系，如图 2-38 所示。

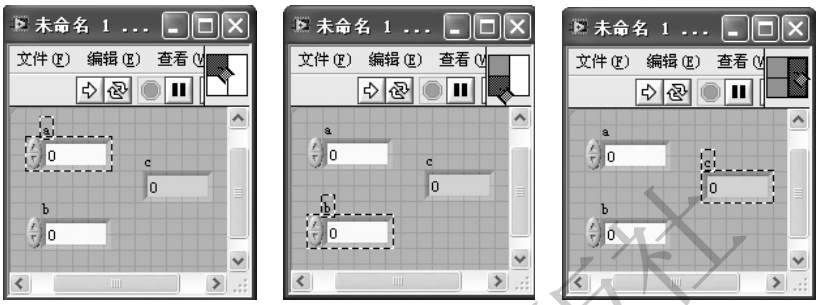


图 2-38　建立控件 a、b、c 与连接端口的关联关系

在完成了连接端口的定义之后，这个 VI 就可以当作子 Vl 来调用了。

（3）程序框图设计

切换到 LabVIEW 的程序框图窗口，调整控件位置，添加节点与连线。

1）添加一个加函数：函数→数值→加。

2）将数值输入控件 a 的输出端口与加函数的输入端口“x”相连。

3）将数值输入控件 b 的输出端口与加函数的输入端口“y”相连。

4）将加函数的输出端口“x+y”与数值显示控件 c 的输入端口相连。

5）保存程序，文件名为“addSub”。

连线后的程序框图如图 2-39 所示。

（4）运行程序

切换到前面板窗口，单击工具栏中的“连续运行”按钮，运行程序。

改变数值输入控件 a、b 的值，数值显示控件 c 显示两数相加的结果。

程序运行界面如图 2-40 所示。

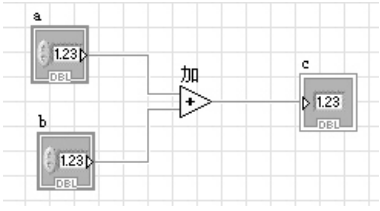


图 2-39　子 VI 程序框图

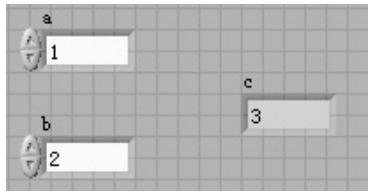


图 2-40　子 VI 运行界面

2．子程序的调用

新建一个 LabVIEW 程序。

（1）程序前面板设计

切换到 LabVIEW 的前面板窗口，通过控件选板给程序前面板添加控件。

1）添加两个数值输入控件：控件→数值→数值输入控件。将标签分别改为“a”和“b”。

2）添加一个数值显示控件：控件→数值→数值显示控件。将标签改为“c”。

设计的程序前面板如图 2-41 所示。

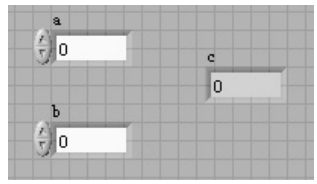


图 2-41　主 VI 前面板

（2）程序框图设计

切换到 LabVIEW 的程序框图窗口，调整控件位置，添加节点与连线。

1）添加子 VI：选择函数选板中的“选择 VI”子选板，如图 2-42 所示，弹出“选择需打开的 VI”对话框，如图 2-43 所示，在对话框中找到需要调用的子 VI，本例是选择任务 1 建立的子程序 addSub.vi，选中后单击“确定”按钮。

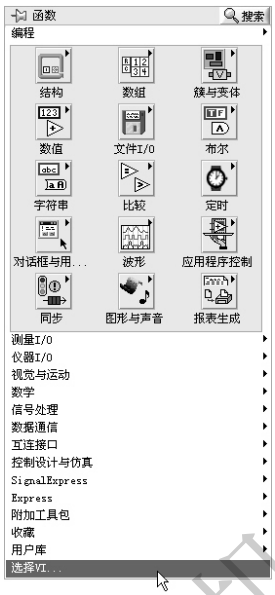


图 2-42　“选择 Vl”子选板

图 2-43　“选择需打开的 VI”对话框

2）将 addSub.vi 的图标放至程序框图窗口中。

3）将数值输入控件 a 的输出端口与 addSub.vi 图标的输入端口 a 相连。

4）将数值输入控件 b 的输出端口与 addSub.vi 图标的输入端口 b 相连。

5）将 addSub.vi 图标的输出端口 c 与数值显示控件 c 的输入端口相连。

6）保存程序，文件名为“addMain”。

连线后的程序框图如图 2-44 所示。

（3）运行程序

切换到前面板窗口，单击工具栏中的“连续运行”按钮，运行程序。

改变数值输入控件 a、b 的值，数值显示控件 c 显示两数相加的结果（可以看出，子程序 addSub.vi 相当于“加”函数）。

程序运行界面如图 2-45 所示。

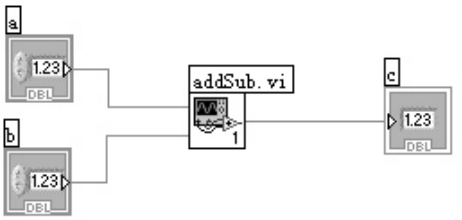


图 2-44　主 VI 程序框图

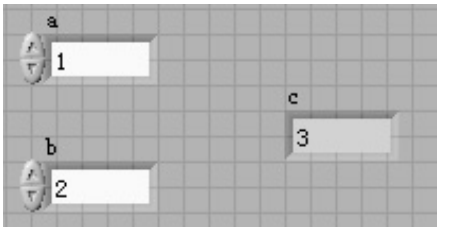


图 2-45　主 VI 运行界面

2.6　VI 的调试方法

在编写了 LabVIEW 的程序代码后，一般需要对程序进行调试。调试的目的是保证程序没有语法错误，并且能够按照用户的目的正确运行，得到正确的结果。

LabVIEW 提供了强大的容错机制和调试手段，例如设置断点调试和设置探针，这些手段可以帮助用户进行程序的调试，发现并改正错误。本节将主要介绍 LabVIEW 提供的用于调试程序的手段以及调试技巧。

2.6.1　找出语法错误

LabVIEW 程序必须在没有基本语法错误的情况下才能运行，LabVIEW 能够自动识别程序中存在的基本语法错误。如果一个 VI 存在语法错误，则程序框图窗口工具栏上的“运行”按钮将会变成一个折断的箭头，表示程序存在错误而不能被执行。单击“运行”按钮，会弹出错误列表，如图 2-46 所示。

单击错误列表中的某一错误，列表中的“详细信息”栏中会显示有关此错误的详细说明，帮助用户更改错误。单击“显示警告”复选框，可以显示程序中的所有警告。

使用 LabVIEW 的错误列表功能时，有一个非常重要的技巧，就是当双击错误列表中的某一项错误时，LabVIEW 会自动定位到发生该错误的对象上，并高亮显示该对象，如图 2-47 所示，数值输入控件与布尔显示控件连接时出现错误。这样便于用户查找并更正错误。

图 2-46　错误列表

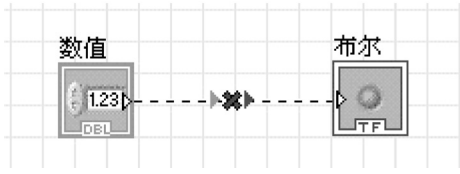


图 2-47　高亮显示程序中的错误

2.6.2 设置断点调试

为了查找程序中的逻辑错误，此时希望程序框图一个节点一个节点地执行。使用断点工具可以在程序的某一点暂时中止程序的执行，用单步方式查看数据。当不清楚程序中哪里出现错误时，设置断点是一种排除错误的手段。在 LabVIEW 中，从工具选板找到断点工具，如图 2-48 所示。在想要设置断点的位置处单击，便可以在那个位置设置一个断点。另外一种设置断点的方法是在需要设置断点的位置右击，从弹出的快捷菜单中选择“设置断点”，即可在该位置设置一个断点。如果想要清除所设定的断点，只要在设置断点的位置处单击即可。

对于节点或者图框，断点显示为红框；对于连线，断点显示为红点。实训的图 2-34 中在比较函数与指示灯控件之间的连线上设置断点后的程序框图如图 2-49 所示。

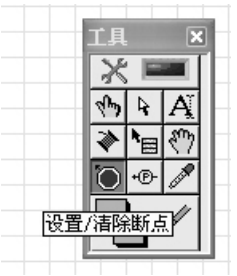


图 2-48　设置断点

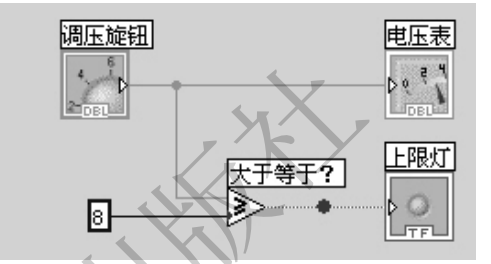


图 2-49　设置断点后的程序框图

运行程序时会发现每当运行到断点位置时程序会停下来，并高亮显示数据流到达的位置，用户可以在这个时候查看程序的运行是否正常，数据显示是否正确。

在断点位置停止运行时的程序框图如图 2-50 所示。从图中可以看出，程序停止在断点位置，并高亮显示数据流到达的对象。按下“单步执行”按钮，闪烁的节点被执行，下一个将要执行的节点变为闪烁，表示它将被执行。也可以单击“暂停”按钮，这样程序将连续执行直到下一个断点。当程序检查无误后，用户可以在断点处单击以清除断点。

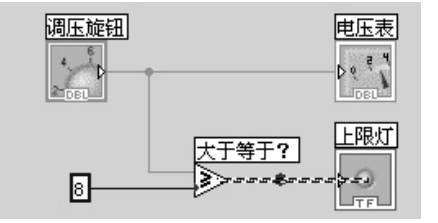


图 2-50　运行带有断点的程序

2.6.3 设置探针

在有些情况下，仅仅依靠设置断点还不能满足调试程序的需要，探针便是一种很好的辅助手段，可以在任何时刻查看任何一条连线上的数据，探针犹如一颗神奇的“针”，能够随时侦测到数据流中的数据。

在 LabVIEW 中，设置探针的方法是用工具选板中的探针工具，如图 2-51 所示。单击程序框图中程序的连线，这样可以在该连线上设置探针以侦测这条连线上的数据，同时在程序中将浮动显示探针监视窗口。要想取消探针，只需要关闭浮动的探针监视窗口即可。

实训 1 中设置好探针的程序框图如图 2-52 所示。运行程序时在探针监视窗口中将显示出探针设置处的数据。

图 2-51　设置探针

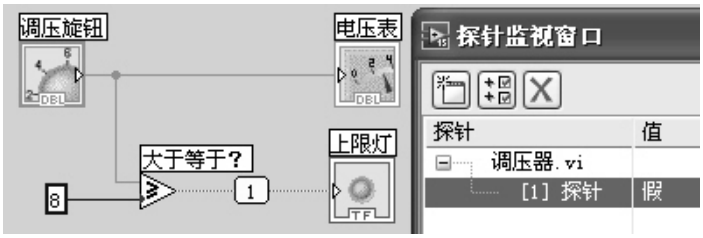


图 2-52　设置好探针的程序框图

利用探针可以检测数据的功能，可以了解程序运行过程中任何位置上的数据，即可知道数据流在空间的分布。利用断点可以将程序停止在任意位置，即可知道数据流在时间的分布。那么综合使用探针和断点，就可以知道程序数据在任何空间和时间的分布了。这一点对 LabVIEW 程序的调试非常重要。

2.6.4　高亮显示程序的运行

有时希望在程序运行过程中，能够实时显示程序的运行流程以及当数据流流过数据节点时的数值，LabVIEW 为用户提供了这一功能，这就是以“高亮显示”方式运行程序。

单击 LabVIEW 工具栏上的“高亮显示程序执行过程”按钮，程序将会以高亮显示方式运行。这时该按钮变为，如同一盏被点亮的灯泡。

下面以“高亮显示程序执行过程”的方式执行实训 1 的程序。在程序的运行过程中，其程序框图如图 2-53 所示。在这种方式下，程序以较慢的速度运行，没有被执行的程序以灰色显示，执行后的程序以高亮显示，同时还可显示数据流线上的数据值。这样，就可以根据数据的流动状态跟踪程序的执行，清楚地看到程序中数据流的流向，并且可以实时地了解每个数据节点的数值。

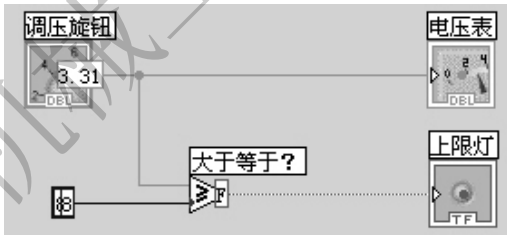


图 2-53　以“高亮显示程序执行过程”方式运行程序

在多数情况下，需要结合多种方式调试 LabVIEW 程序，例如可以在设置探针的情况下，高亮显示程序的运行，并且单步执行程序。这样程序的执行细节将会一览无余。

2.6.5　单步执行和循环运行

单步执行和循环运行是 LabVIEW 支持的两种程序运行方式，和正常运行方式不同的是，这两种运行方式主要用于程序的调试和纠错。它们是除了设置断点和探针两种方法外，另外一种行之有效的程序调试和纠错方式。

在单步执行方式下，用户可以看到程序执行的每一个细节。单步执行的控制由工具栏上

的三个按钮：“开始单步（入）执行”按钮、“开始单步（跳）执行”按钮和“单步步出”按钮完成。这三个按钮表示三种不同类型的单步执行方式。“开始单步（入）执行”按钮表示单步进入程序流程，并在下一个数据节点前停下来；“开始单步跳执行”按钮表示单步进入程序流程，并在下一个数据节点执行后停下来；“单步步出”按钮表示停止单步执行方式，即在执行完当前节点的内容后立即暂停。

下面结合实训 1 介绍单步运行调试程序的方法。

单击“开始单步（入）执行”按钮，程序开始以单步方式执行，程序每执行一步，便停下来并且以高亮显示当前程序执行到的位置，如图 2-54 所示。

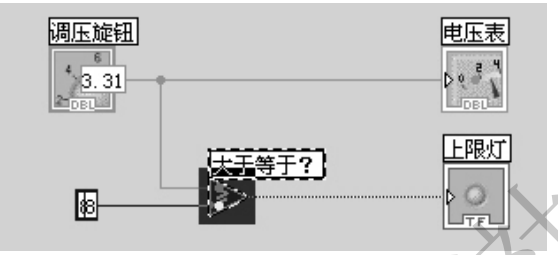


图 2-54　单步执行程序

在 LabVIEW 中支持循环运行方式，其“循环运行”按钮为，其功能是当程序中的数据流流经最后一个对象时，程序会自动重新运行，直到用户手动按下“停止”按钮为止。

第 3 章　LabVIEW 数据操作

数据是操作的对象，操作的结果会改变数据的状况。作为程序设计人员，必须认真考虑和设计数据结构及操作步骤（即算法）。

与其他基于文本模式的编程语言一样，LabVIEW 的程序设计中也涉及常量、变量、函数的概念以及各种数据类型，这些是 LabVIEW 进行程序设计的基础，也是构建 LabVIEW 应用程序的基石。

3.1　LabVIEW 数据概述

3.1.1　LabVIEW 数据类型

LabVIEW 的数据类型按其功能可以分为两类：常量和变量。按其特征又可分为两大类，即数字量类型和非数字量类型，并用不同的图标来代表不同的数据类型。原则上，数据是在相同数据类型的变量之间进行交换的，但 LabVIEW 拥有自己的数据类型转换机制，这也提供了一种程序的容错机制，使其可以在不同数据类型的变量之间交换数据。

在 LabVIEW 中，各种不同的数据类型，其变量的图标边框的颜色不同，因而，从图标边框的颜色可以分辨其数据类型。

1．常用的数据类型

LabVIEW 中常用的数据类型有以下几类。

1）数值型数据类型：又分为整型、浮点型和无符号型等。

2）布尔型数据类型：使用 8 位（一个字节）的数值来存储布尔量数据。如果数值为 0，布尔量数据为“假”，其他非 0 数值代表“真”。

3）数组数据类型：是一组相同数据类型数据的集合。

4）字符串数据类型：以单字节整数的一维数组来存储字符串数据。

5）簇数据类型：和数组不同的是，簇可以用来存储不同数据类型的数据。根据簇中成员的顺序，使用相应的数据类型来存储不同的成员。

6）波形数据类型：用来存储波形数据的一种数据类型。

7）路径数据类型：以句柄或指针来存储数据类型。

8）I/O 通道号数据类型：用来表示 DAQ 设备的 I/O 通道名称。

9）动态数据类型：这种类型的数据在应用时不必具体指定其数据类型，在程序运行过程中，根据需要，对象被动态赋予各种数据类型。

2．常量

LabVIEW 设置了以下两类常量。

1）通用常量。例如圆周率 π、自然对数的底 e 等，这些常数位于函数选板/数值子选板/

数学与科学常量子选板中，数学与科学常量子选板如图 3-1 所示。

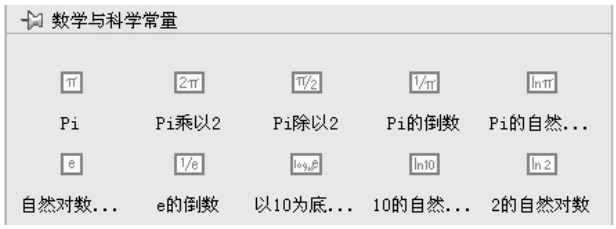


图 3-1　数学与科学常量子选板

2）用户定义常量。LabVIEW 函数选板中有各种常用数据类型的常量，用户可以在编写程序时为它们赋值。例如，数值常量位于数值子选板，它的默认值是 32 位整型数 0，用户可以给它定义任意类型的数值，程序运行时就使用这个值。

3.1.2　数值型数据

1．数值型数据的分类

在 LabVIEW 中，按照精度和数据的范围，数值型数据可以分为表 3-1 所示的几类。

表 3-1　数值型数据类型表

数据类型	标记	简要说明
单精度浮点数	SGL	内存存储格式为 32 位
双精度浮点数	DBL	内存存储格式为 64 位
扩展精度浮点数	EXT	内存存储格式为 80 位
复数单精度浮点数	CSG	实部和虚部内存存储格式均为 32 位
复数双精度浮点数	CDB	实部和虚部内存存储格式均为 64 位
复数扩展精度浮点数	CXT	实部和虚部内存存储格式均为 80 位
8 位整型数	I8	有符号字节型，取值范围为-128～127
16 位整型数	I16	有符号整型，取值范围为-32 768～32 767
32 位整型数	I32	有符号长整型，取值范围为-2 147 483 648～2 147 483 647
无符号 8 位整型数	U8	无符号字节型，取值范围为 0～255
无符号 16 位整型数	U16	无符号整型，取值范围为 0～65 535
无符号 32 位整型数	U32	无符号长整型，取值范围为 0～4 294 967 295

上面的数值型数据类型，随着精度的提高和数据类型所表示数据范围的扩大，其消耗的系统资源（内存）也随之增加。因而，在程序设计时，为了提高程序运行的效率，在满足使用要求的前提下，应该尽量选择精度低和数据范围相对小的数据类型。

当然，有些情况下，变量的取值范围是不能确定的，这时可以取较大的数据类型以保证程序的安全性。在 LabVIEW 中，数据类型是隐含在控制、指示以及常量之中的。

2．数值型数据的创建

数值类型的前面板对象包含在控件选板的数值子选板中，如图 3-2 所示。数值子选板中的前面板对象就相当于传统编程语言中的数字变量。

LabVIEW 中的数字常量是不出现在前面板窗口中的，只存在于程序框图窗口中。在函数选板的数值子选板中有一个名为数值常量的节点，这个节点就是 LabVIEW 中的数字常

量，如图 3-3 所示。

图 3-2　数值控件子选板

图 3-3　数值常量节点

前面板数值子选板包括多种形式的输入和指示，它们的外观各不相同，有数值输入/显示控件、滑动杆、滚动条、液罐、温度计、旋钮及仪表等。它们本质都是完全相同的，都是数值型，只是外观不同而已。LabVIEW 的这一特点为创建虚拟仪器的前面板提供了很大的方便。只要理解了其中一个的用法，就可以掌握其他全部数值类型前面板对象的用法。

3．设置数值型控件的属性

LabVIEW 中的数值型控件有着许多公有属性，每个控件又有自己独特的属性，这里只对控件的公有属性进行简单的介绍。

右击前面板中的数值型控件，弹出图 3-4 所示的快捷菜单，从菜单中可以通过选择“标签”“标题”等命令显示控件的这些属性，另外，通过工具选板中的文本按钮 A 来修改标签和标题的内容。

数值型控件的其他属性可以通过它的属性对话框进行设置。右击控件图标，从弹出的快捷菜单中选择“属性”命令，可以打开图 3-5 所示的属性对话框。对话框分为外观、数据类型、显示格式、说明信息和数据绑定选项卡。

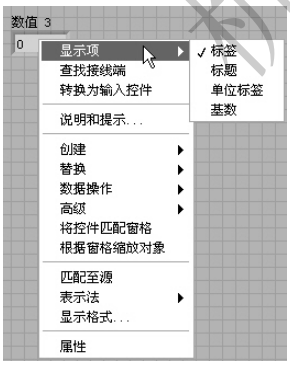


图 3-4　数值型控件的属性快捷菜单

图 3-5　数值型控件的属性对话框

在“外观”选项卡中，用户可以设置与控件外观有关的属性。用户可以修改控件的标签和标题属性以及设置其是否可见；可以设置控件的激活状态，以决定控件是否可以被程序调用。在“外观”选项卡中用户也可以设置控件的颜色和风格。

在“数据类型”选项卡中，用户可以设置数值型控件的数据范围以及默认值。

在“显示格式”选项卡中，用户可以设置控件的数据显示格式以及精度。也可以用该选项卡将数值设置为时间和日期格式。

3.1.3 布尔型数据

布尔型数据即逻辑型数据，它的值为“真”（1）或“假”（0）。LabVIEW 使用 8 位（一个字节）的数值来存储布尔型数据。

1．布尔型数据的创建

布尔型数据是一种二值数据，非“0”即“1”。在 LabVIEW 中，布尔型控件用于布尔型数据的输入和显示。作为输入控件，主要表现为一些开关和按钮，用来改变布尔型控件的状态，控制程序的运行或切换其运行状态；作为显示控件，如指示灯，用于显示程序的运行状态。

在 LabVIEW 中，布尔型数据体现在布尔型前面板对象中。布尔型前面板对象包含在控件选板的布尔子选板中，如图 3-6 所示。

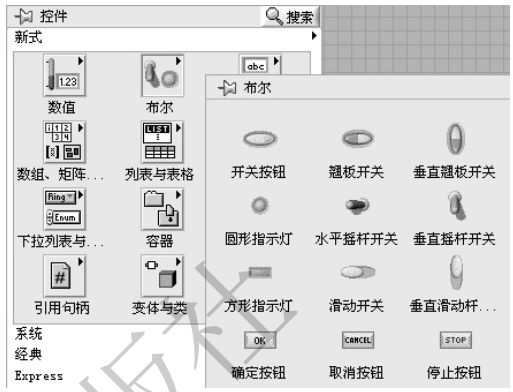


图 3-6　布尔控件子选板

可以看到，布尔子选板中有多种布尔型前面板对象，如不同形状的按钮、指示灯和开关等，这都是从实际仪器的按钮、指示灯和开关演化来的，十分形象。采用这些布尔控件，可以设计出逼真的虚拟仪器前面板。

布尔子选板中的布尔型前面板对象相当于传统编程语言中的布尔变量。在函数选板的布尔子选板中，“真常量”与“假常量”节点就是 LabVIEW 中的布尔常量，如图 3-7 所示。

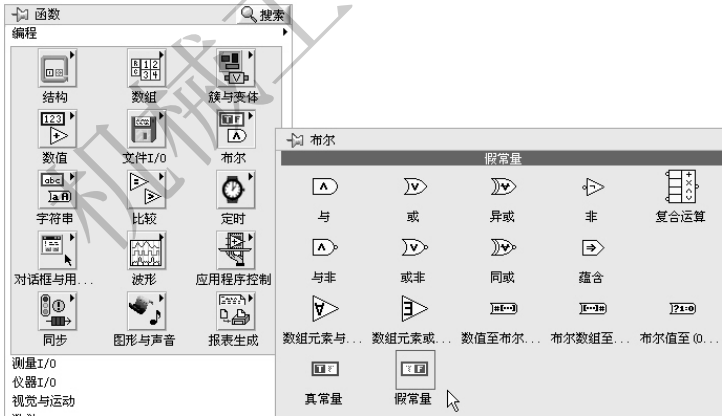


图 3-7　函数选板的布尔子选板中的节点

2．设置布尔型控件的属性

与传统编程语言中的逻辑量不同的是，这些布尔型前面板对象有一个独特的属性，叫作机械动作属性，这是模拟实际继电器开关触点开/闭特性的一种专门开关控制特性。右击一个布尔控件，从弹出的快捷菜单中选择“机械动作”命令，会出现一个图形化的下拉菜单，如图 3-8 所示。下拉菜单中有 6 种不同的机械动作属性。按照从左向右、自上而下的顺序，它们的含义分别为当按下按钮时触发、当松开按钮时触发、当按钮处于

按下状态时触发、按下按钮后以“点动”方式触发、松开按钮时以“点动”方式触发、松开按钮前结束。

机械动作属性的含义是什么？比如有一个按钮，在弹起状态时它的值为“0”，在按下状态时它的值为“1”。机械动作属性定义了用鼠标点按按钮时，按钮的值在什么时刻由“0”阶跃为“1”。这一点对于真实的仪器按钮来说非常重要。由于 LabVIEW 是用来设计虚拟仪器的，因此这一点也显得很重要。灵活使用按钮的这种属性，对于能否开发出优秀的虚拟仪器具有一定的意义。菜单中的图标很直观地显示出了鼠标的点按动作与按钮“0”“1”值的变化关系。

布尔型控件的属性对话框包括外观、操作、说明信息及数据绑定选项卡，如图 3-9 所示。在“外观”选项卡中，用户可以调整开关或按钮的颜色等外观参数；在“操作”选项卡中，用户可以设定按钮或开关的机械动作类型，对每种动作类型进行相应的说明，并可以预览开关的运动效果以及开关的状态。

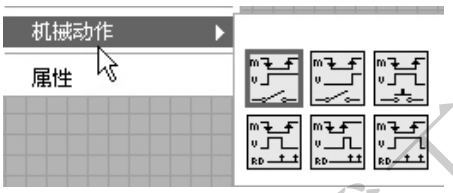


图 3-8　布尔型控件的“机械动作”命令

图 3-9　布尔型控件的属性对话框

布尔型控件可以用文字的方式在控件上显示其状态，如果要显示开关的状态，只需要在布尔型控件的“外观”选项卡中选择“显示布尔文本”复选框即可。

3.1.4　字符串数据

字符串、字符串数组和含字符串的簇是前面板设计、仪器控制和文件管理等任务中常见的数据结构，也是使用比较灵活的数据结构。

1．字符串数据的作用

在 LabVIEW 的编程中，常用到字符串控件或字符串常量，用于显示一些屏幕信息。

字符串是一系列 ASCII 码字符的集合，这些字符可能是可显示的，也可能是不可显示的，如换行符、制表位等。

程序通常在以下情况用到字符串：传递文本信息；当把数值型的数据作为 ASCII 码格式存盘时，需要先把数值转换为字符串；在传统仪器通信控制中，需要把数值型的数据转换为字符串数据后进行传递。

2．字符串数据的创建

在 LabVIEW 的前面板上，与创建字符串数据相关的控件位于控件选板的字符串与路径

子选板中，如图 3-10 所示。

图 3-10　字符串与路径控件子选板

用得最频繁的字符串控件是字符串输入控件和字符串显示控件，这两个控件分别是字符串的输入量和显示量。对于字符串输入控件，可以使用操作工具或标签工具在字符串控件中输入或修改文本。字符串显示控件则主要用于字符串的显示。如果控件中有多行文本，则可以拖动控件边框改变其大小，使文本得以全部显示。

用操作工具或标签工具单击字符串输入控件的显示区，即可在控件显示区的光标位置进行字符串的输入和修改。字符串的输入和修改操作与常见的文本编辑操作几乎完全一样。

LabVIEW 的字符串输入控件就是简单的文本编辑器。用户可以通过双击鼠标并拖动来选定一部分字符，对已选定的文字进行剪切、复制和粘贴等编辑操作，还可改变选定文字的大小、字体和颜色等属性。同样，常用的文本编辑功能键在输入字符串时同样有效，如光标键、换页键、退格键和删除键等。

字符输入完毕，可以右击控件，在弹出的快捷菜单中选择“数据操作→当前值设置为默认值”命令进行保存。下次重新启动该 VI 时，字符串的内容将保持不变。

LabVIEW 的字符串控件可同时输入或输出多行的文本，为了便于观察，可用定位工具来调整显示区大小。

在 LabVIEW 的程序框图中也可以创建字符串数据。创建的方式有两种，一种是通过创建字符串的函数，另一种是利用函数选板中的相应控件直接创建字符串常量。两种方式用到的函数以及控件位于函数选板中的字符串子选板中，如图 3-11 所示。

图 3-11　函数选板中的字符串子选板

3．设置字符串数据的属性

字符串的显示形式有以下几种。

1）正常显示：正常显示字符串。

2）“\”代码显示：控制码显示。

3）密码显示：用显示密码的方式显示字符串，主要用于输入口令。用“*”代替所有字符。

4）十六进制显示：用十六进制数显示所有字符的 ASCII 码值。

字符串显示控件可在不同的显示形式之间进行切换，可右击控件，在弹出的快捷菜单中选择相应的命令进行。例如，字符串“LabVIEW”的几种显示形式如图 3-12 所示。

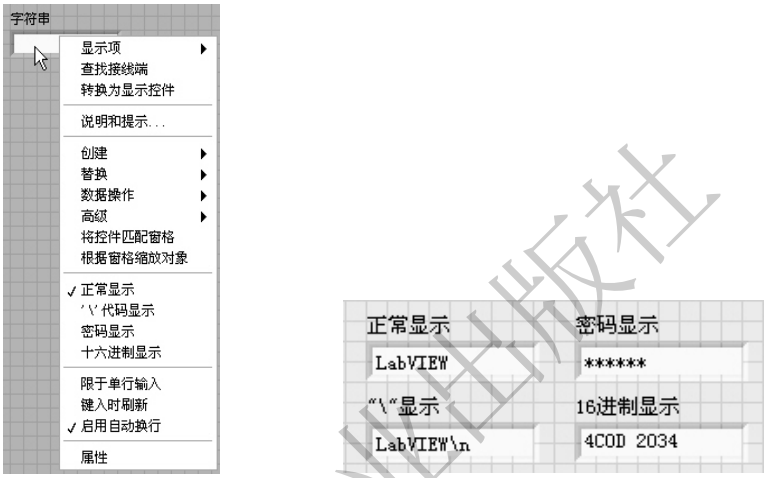


图 3-12　字符串的显示形式

字符串输入控件和显示控件的属性可以通过其属性对话框进行设置。在控件的图标上右击，从弹出的快捷菜单中选择“属性”命令，可以打开图 3-13 所示的“字符串属性：字符串”对话框。“字符串属性：字符串”对话框由外观、说明信息、数据绑定及快捷键选项卡组成。

在“外观”选项卡中，用户不仅可以设置标签和标题等属性，而且可以设置文本的显示方式。如果选择“显示垂直滚动条”复选框，则当文本框中的字符串不止一行时会显示滚动条；如果选择“限于单行输入”复选框，那么将限制用户在单行输入字符串，而不能回车换行；如果选择“键入时刷新”复选框，那么文本框的值会随用户输入的字符而实时改变，不会等到用户按〈回车〉键后才改变。

图 3-13　字符串型控件的属性对话框

3.1.5　数组数据

在程序设计语言中，“数组”是相同数据类型数据的集合，是一种存储和组织相同类型数据的良好方式。

1．数组数据的组成

LabVIEW 中的数组是由同一类型数据元素组成的大小可变的集合，这些元素可以是数值型、布尔型、字符型等类型，也可以是簇，但不能是数组。这些元素必须同时都是输入控件或同时都是显示控件。

前面板的数组对象往往由一个盛放数据的容器和数据本身构成，在程序框图中则体现为一个一维或多维矩阵。

数组可以是一维的，也可以是多维的。一维数组是一行或一列数据，可以描绘平面上的一条曲线。二维数组是由若干行和列数据组成的，可以在一个平面上描绘多条曲线。

LabVIEW 是图形化编程语言，因此，LabVIEW 中数组的表现形式与其他语言有所不同，数组由三部分组成：数据索引、数据和数据类型。其中，数据类型隐含在数据中，如图 3-14 所示。

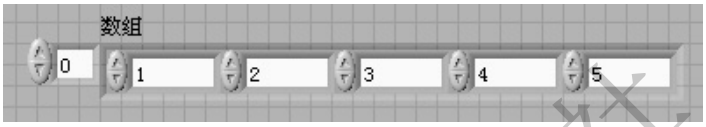


图 3-14　一维数组的组成

数组左侧为索引显示，其中的索引值是位于数组框架中最左侧或最上面元素的索引值，这样做是由于数组中能够显示的数组元素个数是有限的，用户通过索引显示可以很容易地查看数组中的任何一个元素。在数组中，数组元素位于右侧的数组框架中，按照元素索引由小到大的顺序从左至右或从上至下排列。

对数组成员的访问是通过数组索引进行的，数组中的每一个元素都有其唯一的索引数值，可以通过索引值来访问数组中的数据。索引值的范围是 0～n-1，n 是数组成员的数目。每一个数组成员有一个唯一的索引值，数组索引值从 0 开始，到 n-1 结束，例如图 3-15 中二维数组里的数值 9 的行索引值是1，列索引值是 3。

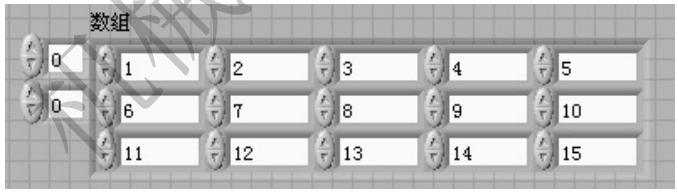


图 3-15　二维数组的组成

LabVIEW 中的数组与其他编程语言相比比较灵活。如 C 语言，在使用一个数组时，必须首先定义该数组的长度，但 LabVIEW 却不必如此，它会自动确定数组的长度。数组中元素的数据类型必须完全相同，如都是无符号 16 位整数或都是布尔型等。

2．数组数据的创建

在 LabVIEW 中，可以用多种方法来创建数组数据。其中常用的有以下两种方式：在前面板上创建数组数据；在程序框图中创建数组数据。

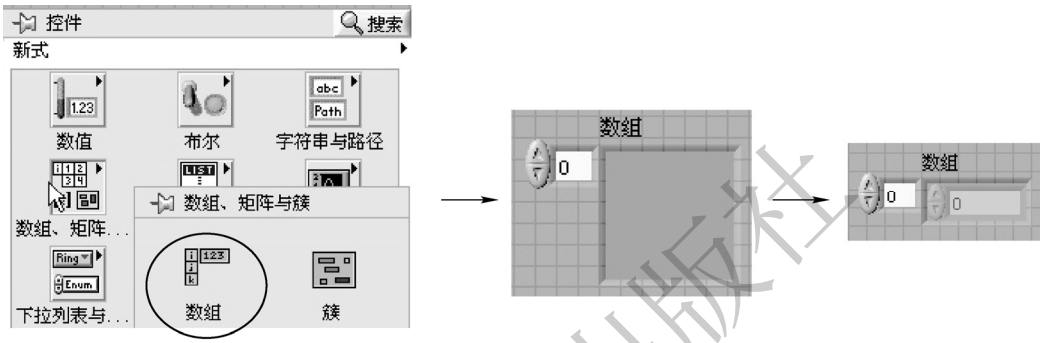
（1）在前面板上创建数组

在前面板上，数组的创建分两步进行。

1）从控件选板的数组、矩阵与簇子选板中选择数组框架，如图 3-16a 所示。注意，此时创建的只是一个数组框架，不包含任何内容，在程序框图中的对应端口只是一个黑色中空的矩形图标。

2）根据需要将相应数据类型的前面板对象放入数组框架中。用户可以直接从控件选板中选择对象放进数组框架内，也可以把前面板上已有的对象拖进数组框架内。这个数组的数据类型以及它是输入控件还是显示控件完全取决于放入的对象。

图 3-16b 所示的是将一个数值输入控件放入数组框架，这样就创建了一个数值类型数组（数组的属性为输入）。当数组创建完成之后，数组在程序框图中相应的端口就变为相应颜色和数据类型的图标了。

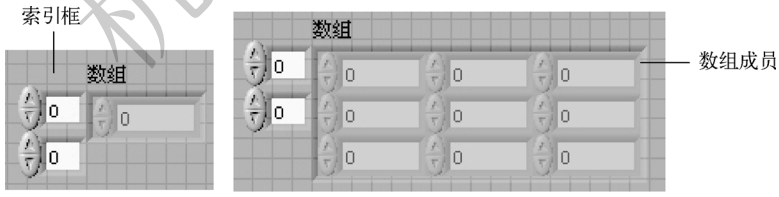


a) b)

图 3-16　在前面板上创建数组

数组在创建之初都是一维数组，如果需要创建一个多维数组，把定位工具放在数组索引框任意一角，向上或向下拖动鼠标来增加索引框数量就可以增加数组的维数。如图 3-17a 所示，两个索引框中，上一个是行索引，下一个是列索引。

刚刚创建的数组只显示一个成员，如果需要显示更多的数组成员，把定位工具放在数组数据显示区任意一角，当光标形状变成网状折角时，向任意方向拖动来增加数组成员数量就可以显示更多数据，如图 3-17b 所示。



a) b)

图 3-17　增加数组成员

a) 增加数组维数　b) 显示更多的数组成员

（2）在程序框图中创建数组常量

创建数组常量时，先从函数选板的数组子选板中选择数组常量对象并放到程序框图窗口中，然后根据需要选择一个数据常量，放到空数组中即可。

（3）数组成员赋值

用上述方法创建的数组是空的，从外观上看，数组成员都显示为灰色，可根据需要用操

作工具或定位工具为数组成员逐个赋值。若跳过前面的成员为后面的成员赋值，则前面的成员根据数据类型自动赋一个空值，例如，0、F 或空字符串。数组赋值后，赋值范围以外的成员仍然显示为灰色的。如图 3-18 所示，创建了一个数组常量，并将一个字符串常量放到空数组中，然后给它赋值“abc”。

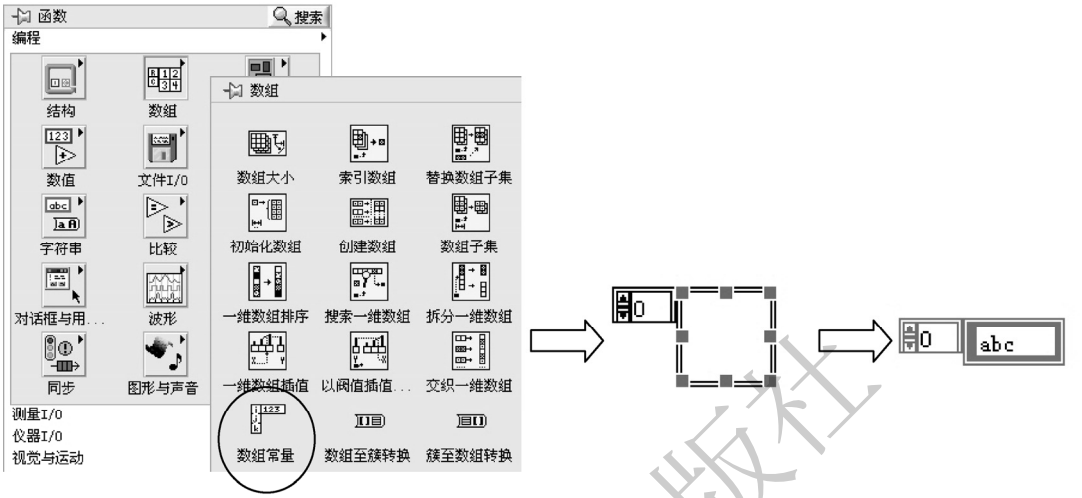


图 3-18　在程序框图中创建数组常量并赋值

3．数组数据的使用

在程序框图设计中，对一个数组进行操作，包括求数组的长度、对数据排序、取出数组中的元素、替换数组中的元素或初始化数组等各种运算。传统编程语言主要依靠各种数组函数来实现这些运算，而在 LabVIEW 中，这些函数是以功能函数节点的形式来表现的。

3.1.6　簇数据

簇是 LabVIEW 中一个比较特别的数据类型，它可以将几种不同的数据类型集中到一个单元中以形成一个整体。

1．簇数据的组成

在程序设计时，仅有整型、浮点型、布尔型、字符串型和数组型数据是不够的，有时为便于引用，还需要将不同的数据类型组合成一个有机的整体。例如，一个学生的学号、姓名、性别、年龄、成绩和家庭地址等数据项，这些数据项都与某一个学生相联系。如果将这些数据项分别定义为相互独立的简单变量，是难以反映它们之间的内在联系的。应当把它们组成一个组合项，在组合项中包含若干个类型不同（当然也可以相同）的数据项。簇就是这样一种数据结构。

簇是一种类似数组的数据结构，用于分组数据。一个簇就是一个由若干不同数据类型的成员组成的集合体，类似于 C 语言中的结构体。用户可以把簇想象成一束通信电缆线，电缆中的一根线就是簇中的一个数据元素。

使用簇可以为编程带来以下的便利。

1）簇通常可将程序框图中多个地方的相关数据元素集中到一起，这样只需一条数据连线即可把多个节点连接到一起，减少了数据连线数量。

2）子程序中有多个不同数据类型的参数输入或输出时，把它们组合成一个簇可以减少连接板上端口的数量。

3）某些控件和函数必须使用簇这种数据类型的参数。

簇的成员可以是任意的数据类型，但是必须同时都是控件或同时都是指示器。

2．簇数据的创建

（1）在前面板上创建簇

在前面板上，簇的创建类似于数组的创建。首先在控件选板数组、矩阵与簇子选板中创建簇的框架，然后向框架中添加所需的元素，最后根据编程需要更改簇和簇中各元素的名称。这个簇的数据类型以及它是输入控件还是显示控件完全取决于放入的对象。如图 3-19 所示，在前面板中创建了一个簇，簇中放入了一个数值输入控件、一个字符串输入控件、一个布尔型指示灯控件。

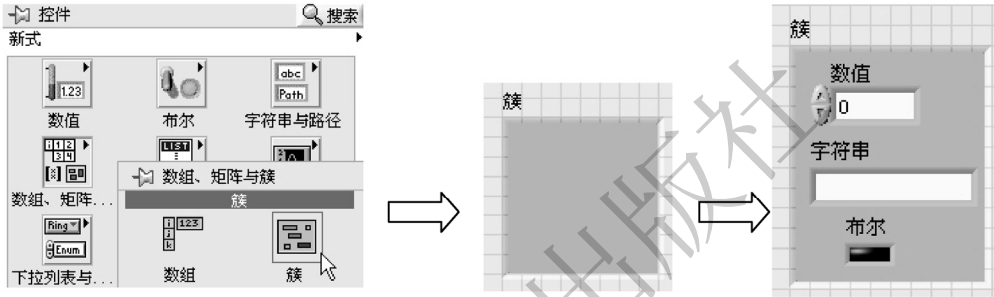


图 3-19　在前面板创建簇

在 LabVIEW 中，簇只能包含输入控制和显示控件中的一种，不能既包含输入控制又包含显示控件，但可以用修饰子选板中的图形元素将二者集中在一起，但这种集中仅是位置上的集中。

（2）在程序框图中创建簇常量

在程序框图中创建簇常量类似于在前面板上创建簇。创建时，先从函数选板的簇与变体子选板中选择簇常量的框架并放到程序框图中，然后根据需要选择一些数据常量并放到簇框架中。如图 3-20 所示，创建了一个簇常量，并将一个数值常量、一个字符串常量、一个布尔型真常量放到簇框架中。用上述方法创建的簇常量，其成员还没有有效的值，从外观上看都为灰色。用户可根据需要用操作工具或定位工具为簇成员逐个赋值。

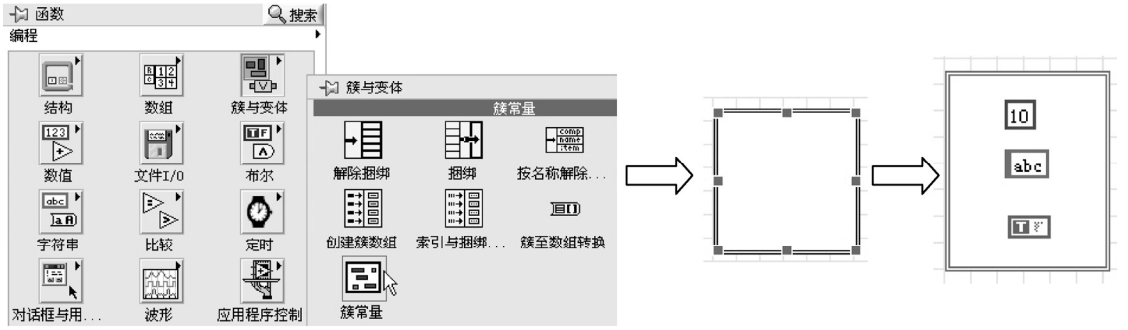


图 3-20　在程序框图中创建簇常量

用户也可以把前面板上的簇控件拖动或复制到程序框图窗口中，产生一个簇常量。只有数值型成员的簇边框是棕色的，其他为粉红色。

簇成员按照它们放入簇的先后顺序排序，将放入簇框架中的第一个对象标记为 0，将放入的第二个对象标记为 1，以此类推。如果要访问簇中的单个元素，必须记住簇顺序，因为簇中的单个元素是按顺序而不是按名字访问的。

在程序框图设计中，用户使用一个簇时，主要是访问簇中的各个元素，或由不同类型但相互关联的数据组成一个簇。

3.1.7 LabVIEW 数据运算

1．基本数学运算

LabVIEW 中的数学运算主要由函数选板的数值子选板中的节点完成，如图 3-21 所示。

数值子选板由基本数学运算节点、类型转换节点、复数节点和附加常数节点等组成。

基本数学运算节点主要实现加、减、乘、除等基本运算。基本数学运算节点支持数值量输入。与一般编程语言提供的运算符相比，LabVIEW 中的数学运算节点功能更强，使用更灵活。它不仅支持单一的数值量输入，还支持处理不同类型的复合型数值量，比如由数值量构成的数组和簇。

2．比较运算

比较运算也就是通常所说的关系运算，比较运算节点包含在函数选板的比较子选板中，如图 3-22 所示。

图 3-21　数值子选板

图 3-22　比较子选板

在 LabVIEW 中可以进行以下几种类型的比较：数字值的比较、布尔值的比较、字符串的比较以及簇的比较。

（1）数字值的比较

比较节点在比较两个数字值时，会先将其转换为同一类型的数字。当一个数字值和一个非数字相比较时，比较节点将返回一个表示二者不相等的值。

（2）布尔值的比较

两个布尔值相比较时，“真”值比“假”值大。

（3）字符串的比较

字符串的比较是按照字符在 ASCII 表中的等价数字值进行比较的。例如，字符“a”（在

ASCII 表中的值为 97）大于字符“A”（值为 65）；字符“A”大于字符“0”（值为 48）。当两个字符串进行比较时，比较节点会从这两个字符串的第一个字符开始逐个比较，直至有两个字符不相等为止，并按照这两个不相等字符的大小确定整个字符串的大小。

（4）簇的比较

簇的比较与字符串的比较类似，比较时，从簇的第 0 个元素开始，直至有元素不相等为止。簇中元素的个数必须相同，元素的数据类型和顺序也必须相同。

3．逻辑运算

传统编程语言使用逻辑运算符将关系表达式或逻辑量连接起来，形成逻辑表达式。逻辑运算符包括与、或、非等。在 LabVIEW 中，这些逻辑运算符是以图标的形式出现的。

逻辑运算节点包含在函数选板的布尔子选板中，如图 3-23 所示。逻辑运算节点的图标与集成电路常用逻辑符号一致，可以使用户方便地使用这些节点，而无须重新记忆。

图 3-23　布尔子选板

3.2　LabVIEW 数据操作实训

实训 3　数值数据操作

一、学习目标

1）掌握数值型数据的各种输入与显示的创建方法。

2）掌握数值型数据的基本数学运算方法。

二、设计任务

（一）任务 1

1．任务描述

通过滑动杆、转盘、滚动条产生数值，通过量表、温度计、进度条、液罐输出显示。

2．任务实现

（1）程序前面板设计

新建 VI，切换到 LabVIEW 的前面板窗口，通过控件选板给程序前面板添加控件。

1）为了产生数值，添加一个填充滑动杆控件：控件→数值→垂直填充滑动杆。

2）为了产生数值，添加一个转盘控件：控件→数值→转盘。

3）为了产生数值，添加一个滚动条控件：控件→数值→水平滚动条。

4）为了产生数值，添加一个指针滑动杆控件：控件→数值→垂直指针滑动杆。

5）为了显示数值，添加一个量表控件：控件→数值→量表。

6）为了显示数值，添加一个温度计控件：控件→数值→温度计。

7）为了显示数值，添加一个进度条控件：控件→数值→水平进度条。

8）为了显示数值，添加一个液罐控件：控件→数值→液罐。

设计的程序前面板如图 3-24 所示。

（2）程序框图设计

切换到 LabVIEW 的程序框图窗口，调整控件位置。

1）将垂直填充滑动杆控件的输出端口与量表控件的输入端口相连。

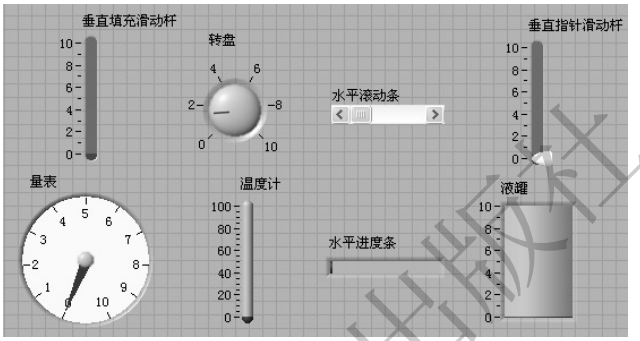


图 3-24　程序前面板

2）将转盘控件的输出端口与温度计控件的输入端口相连。

3）将水平滚动条控件的输出端口与水平进度条控件的输入端口相连。

4）将垂直指针滑动杆控件的输出端口与液罐控件的输入端口相连。

连线后的程序框图如图 3-25 所示。

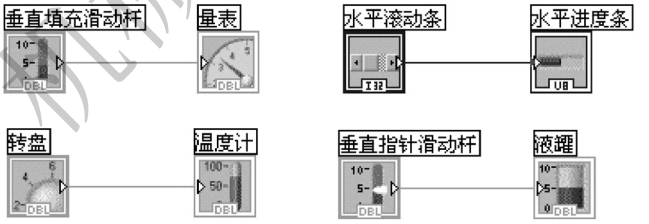


图 3-25　程序框图

（3）运行程序

切换到前面板窗口，单击工具栏中的“连续运行”按钮，运行程序。

通过鼠标推动或转动滑动杆、转盘、滚动条等改变数值，量表控件、温度计控件、水平进度条控件、液罐控件显示值发生同样变化。

用户可以使用鼠标改变各输入控件的上限刻度值，比如将转盘的上限刻度 10 改为 100。

程序运行界面如图 3-26 所示。

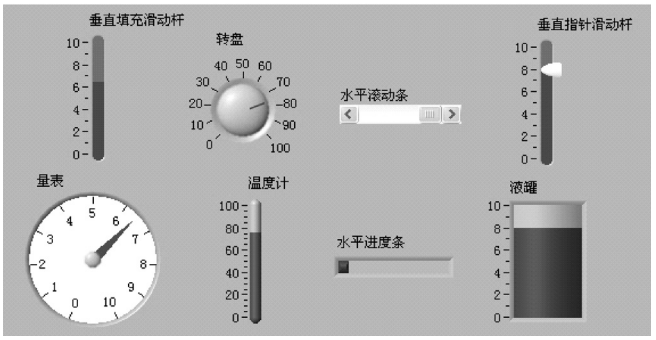


图 3-26　程序运行界面

（二）任务 2

1．任务描述

将某数值与一个数值常量相减，对结果求绝对值后输出显示。

2．任务实现

（1）程序前面板设计

新建 VI，切换到 LabVIEW 的前面板窗口，通过控件选板给程序前面板添加控件。

1）为了输入数值，添加一个数值输入控件：控件→数值→数值输入控件。将标签改为“a”。

2）为了显示数值，添加三个数值显示控件：控件→数值→数值显示控件。将标签分别改为“数值常量”“相减输出”和“绝对值输出”。

设计的程序前面板如图 3-27 所示。

（2）程序框图设计

切换到 LabVIEW 的程序框图窗口，调整控件位置，添加节点与连线。

1）添加一个减函数：函数→数值→减。

2）添加一个数值常量：函数→数值→数值常量。将值设置为“20”。

3）添加一个绝对值函数：函数→数值→绝对值。

4）将数值输入控件 a 的输出端口与减函数的输入端口“x”相连。

5）将数值常量“20”与减函数的输入端口“y”相连。

6）将数值常量“20”与“数值常量”显示控件的输入端口相连。

7）将减函数的输出端口“x-y”与“相减输出”数值显示控件的输入端口相连。

8）将减函数的输出端口“x-y”与绝对值函数的输入端口“x”相连。

9）将绝对值函数的输出端口“abs(x)”与“绝对值输出”数值显示控件的输入端口相连。

连线后的程序框图如图 3-28 所示。

图 3-27　程序前面板

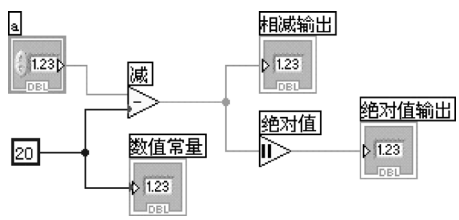


图 3-28　程序框图

（3）运行程序

切换到前面板窗口，单击工具栏中的“连续运行”按钮，运行程序。

改变数值输入控件 a 的值，与数值常量 20 相减，求绝对值后输出结果。

程序运行界面如图 3-29 所示。

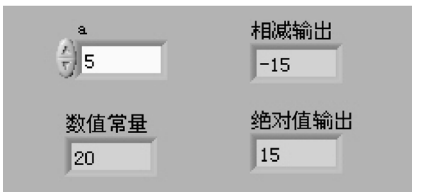


图 3-29　程序运行界面

实训 4　布尔数据操作

一、学习目标

1）掌握布尔型数据输入与显示的创建方法。

2）掌握数值型数据的比较运算和逻辑运算方法。

二、设计任务

（一）任务 1

1．任务描述

在程序前面板中，通过开关控制指示灯颜色的变化。

2．任务实现

（1）程序前面板设计

新建 VI，切换到 LabVIEW 的前面板窗口，通过控件选板向程序前面板添加控件。

1）添加两个修饰控件：控件→修饰→平面圆形。

通过鼠标改变其大小和形状，通过工具箱中的设置颜色工具改变其颜色。其中，大椭圆相当于人的脸，小椭圆相当于人的嘴巴。

2）添加两个指示灯控件：控件→布尔→圆形指示灯。

将标签分别改为“眼睛 1”和“眼睛 2”，然后分别右击两个指示灯控件，选择显示项，隐藏标签。通过鼠标改变其大小。这两个指示灯相当于人的两只眼睛。

3）添加一个开关控件：控件→布尔→垂直摇杆开关。

将标签改为“鼻子”，然后右击开关控件，选择显示项，隐藏标签。通过鼠标改变其大小。这个垂直摇杆开关相当于人的鼻子。

设计的程序前面板如图 3-30 所示。形状和布局类似于人的脸部。

（2）程序框图设计

切换到 LabVIEW 的程序框图窗口，调整控件位置。

将垂直摇杆开关控件（“鼻子”）的输出端口分别与两个指示灯控件（“眼睛 1”和“眼睛 2”）的输入端口相连。

连线后的程序框图如图 3-31 所示。

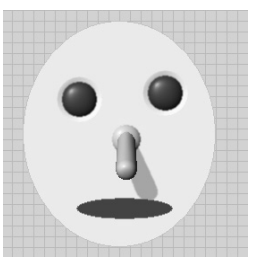


图 3-30　程序前面板

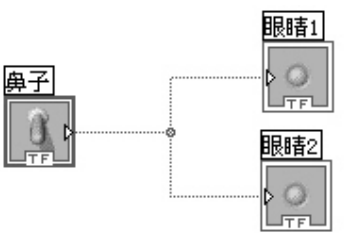


图 3-31　程序框图

（3）运行程序

切换到前面板窗口，单击工具栏中的“连续运行”按钮，运行程序。

在程序前面板中单击开关（“鼻子”），两个指示灯（“眼睛”）颜色发生变化。

程序运行界面如图 3-32 所示。

（二）任务 2

1．任务描述

当两个数值同时大于某个数值时，指示灯的颜色发生变化。

2．任务实现

（1）程序前面板设计

新建 VI，切换到 LabVIEW 的前面板窗口，通过控件选板向程序前面板添加控件。

1）添加两个数值输入控件：控件→数值→数值输入控件。将标签分别改为“a”和“b”。

2）添加一个指示灯控件：控件→布尔→圆形指示灯。将标签改为“指示灯”。

设计的程序前面板如图 3-33 所示。

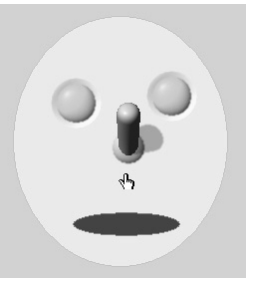


图 3-32　程序运行界面

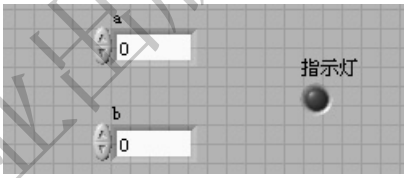


图 3-33　程序前面板

（2）程序框图设计

切换到 LabVIEW 的程序框图窗口，调整控件位置，添加节点与连线。

1）添加两个比较函数：函数→比较→大于？。

2）添加两个数值常量：函数→数值→数值常量。将数值均设置为“5”。

3）添加一个布尔“与”函数：函数→布尔→与。

4）将数值 a 控件的输出端口与比较函数“大于？”（上）的输入端口“x”相连。

5）将数值常量“5”（上）与比较函数“大于？”（上）的输入端口“y”相连。

6）将数值 b 控件的输出端口与比较函数“大于？”（下）的输入端口“x”相连。

7）将数值常量“5”（下）与比较函数“大于？”（下）的输入端口“y”相连。

8）将比较函数“大于?”（上）的输出端口“x>y?”与逻辑“与”函数的输入端口“x”相连。

9）将比较函数“大于?”（下）的输出端口“x>y?”与逻辑“与”函数的输入端口“y”相连。

10）将“与”函数的输出端口“x 与 y?”与指示灯控件的输入端口相连。

连线后的程序框图如图 3-34 所示。

（3）运行程序

切换到前面板窗口，单击工具栏中的“连续运行”按钮，运行程序。

改变数值 a 和数值 b 的大小，当数值 a 和数值 b 同时大于数值 5 时，指示灯改变颜色。

程序运行界面如图 3-35 所示。

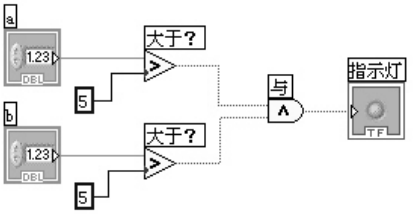


图 3-34　程序框图

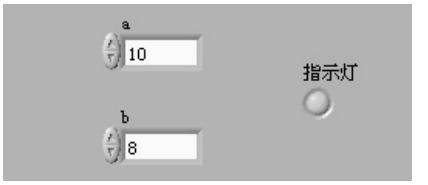


图 3-35　程序运行界面

实训 5　字符串数据操作

一、学习目标

1）掌握字符串数据的创建与属性设置方法。

2）掌握字符串数据的连接、长度计算等操作。

二、设计任务

（一）任务 1

1．任务描述

将两个字符串连接成一个新的字符串，并计算新字符串的长度。

2．任务实现

（1）程序前面板设计

新建 VI，切换到 LabVIEW 的前面板窗口，通过控件选板向程序前面板添加控件。

1）为了输入字符串，添加两个字符串输入控件：控件→字符串与路径→字符串输入控件。将标签分别改为“字符串 1”和“字符串 2”。

2）为了显示连接后的字符串，添加一个字符串显示控件：控件→字符串与路径→字符串显示控件。将标签改为“连接后的字符串”。

3）为了显示字符串的长度，添加一个数值显示控件：控件→数值→数值显示控件。将标签改为“长度”。

设计的程序前面板如图 3-36 所示。

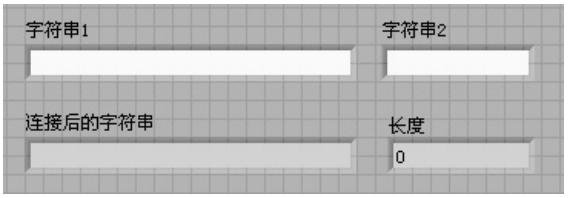


图 3-36　程序前面板

（2）程序框图设计

切换到 LabVIEW 的程序框图窗口，调整控件位置，添加节点与连线。

1）添加一个连接字符串函数：函数→字符串→连接字符串。

2）添加一个字符串长度函数：函数→字符串→字符串长度。

3）将两个字符串输入控件的输出端口分别与连接字符串函数的两个输入端口相连。

4）将连接字符串函数的输出端口与字符串显示控件的输入端口相连。

5）将连接字符串函数的输出端口与字符串长度函数的输入端口相连。

6）将字符串长度函数的输出端口与数值显示控件“长度”的输入端口相连。

连线后的程序框图如图 3-37 所示。

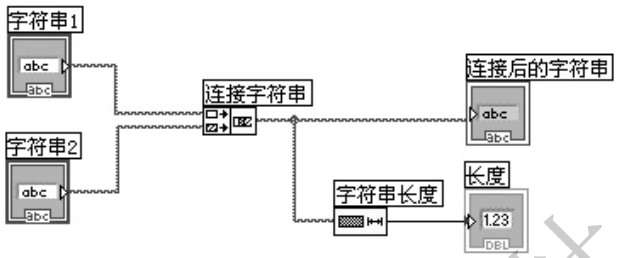


图 3-37　程序框图

（3）运行程序

切换到前面板窗口，单击工具栏中的“连续运行”按钮，运行程序。

将两个字符串，即“LabVIEW 2015 中文版”和“入门与提高”，连接成一个新的字符串，并作为结果显示。计算连接后的字符串“LabVIEW 2015 中文版入门与提高”的长度是 28。

在字符串中，一个英文字符和数字的长度是 1，一个汉字的长度是 2。

程序运行界面如图 3-38 所示。

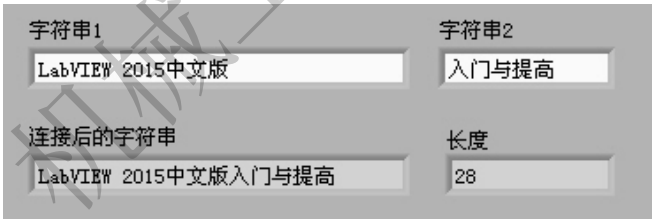


图 3-38　程序运行界面

（二）任务 2

1．任务描述

通过组合框下拉列表选择不同的字符串，并以不同的方式显示。

2．任务实现

（1）程序前面板设计

新建 VI，切换到 LabVIEW 的前面板窗口，通过控件选板向程序前面板添加控件。

1）添加一个组合框控件用于输入选择：控件→字符串与路径→组合框。标签为“组合框”。

2）添加一个字符串显示控件用于字符串正常显示：控件→字符串与路径→字符串显示

控件。将标签改为“字符串正常显示”。

3）添加一个字符串显示控件用于字符串密码形式显示：控件→字符串与路径→字符串显示控件。将标签改为“密码形式显示”。右击该字符串显示控件，在弹出的快捷菜单中选择“密码显示”命令。

设计的程序前面板如图 3-39 所示。

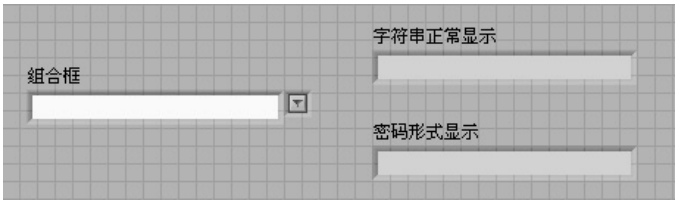


图 3-39　程序前面板

（2）组合框编辑

右击前面板中的组合框控件，在弹出的快捷菜单中选择“编辑项”命令，出现“组合框属性：组合框”对话框，如图 3-40 所示。

图 3-40 “组合框属性：组合框”对话框

单击“Insert”（“插入”）按钮，在左侧输入“LabVIEW”，再重复单击 “Insert”（“插入”）按钮两次，分别输入“2015”和“登录密码”。选择字符串，单击“上移”或“下移”按钮可调整字符串位置。下拉列表编辑完成后，单击“确定”按钮确认。

（3）程序框图设计

切换到 LabVIEW 的程序框图窗口，调整控件位置。

将组合框控件的输出端口分别与“字符串正常显示”控件、“密码形式显示”控件的输入端口相连。

连线后的程序框图如图 3-41 所示。

（4）运行程序

切换到前面板窗口，单击工具栏中的“连续运行”按钮，运行程序。

单击组合框右侧的下拉按钮出现下拉列表，选择不同的值，分别进行正常显示和密码显示。

程序运行界面如图 3-42 所示。

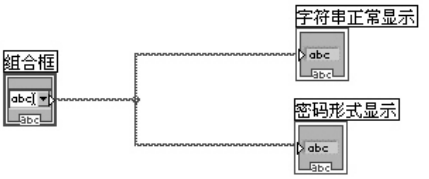


图 3-41　程序框图

图 3-42　程序运行界面

实训 6　数组数据操作

一、学习目标

掌握数组数据的创建与操作方法。

二、设计任务

（一）任务 1

1．任务描述

使用初始化数组函数建立一个所有成员全部相同的数组。

2．任务实现

（1）程序前面板设计

新建 VI，切换到 LabVIEW 的前面板窗口，通过控件选板向程序前面板添加控件。

1）添加一个数组控件：控件→数组、矩阵与簇→数组。标签为“数组”。

2）添加一个字符串显示控件：控件→字符串与路径→字符串显示控件。将字符串显示控件移到数组控件数据显示区框架中。

3）选中数组控件索引框，其周围出现方框，把鼠标指针放在下方框上，向下拖动鼠标增加索引框数量，将数组维数设置为 2；选中数组控件数据显示区框架，其周围出现方框，把鼠标指针放在下方框或右方框上，向下和向右拖动就可增加数组成员数量，显示更多数据。本例将成员数量设置为 3 行 4 列。

设计的程序前面板如图 3-43 所示。

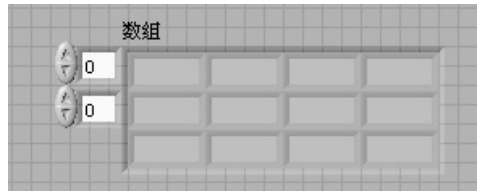


图 3-43　程序前面板

（2）程序框图设计

切换到 LabVIEW 的程序框图窗口，添加节点与连线。

1）添加一个初始化数组函数：函数→数组→初始化数组。把鼠标指针放在函数节点下方框上，向下拖动，将输入端口“维数大小”设置为两个。

2）添加一个字符串常量：函数→字符串→字符串常量。将值设置为“a”。

3）添加两个数值常量：函数→数值→数值常量。将值分别设置为“3”和“4”。

4）将字符串常量“a”与初始化数组函数的输入端口“元素”相连。

5）将数值常量“3”“4”分别与初始化数组函数的两个输入端口相连。

6）将初始化数组函数的输出端口与数组控件的输入端口相连。

连线后的程序框图如图 3-44 所示。

（3）运行程序

切换到前面板窗口，单击工具栏中的“运行”按钮，运行程序。

本例创建了一个 3 行 4 列的所有成员都是“a”的字符串常量数组。

程序运行界面如图 3-45 所示。

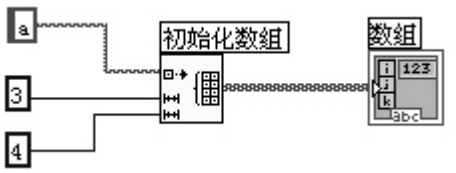


图 3-44　程序框图

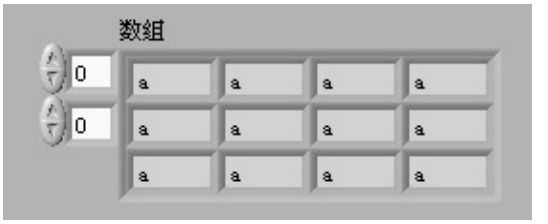


图 3-45　程序运行界面

（二）任务 2

1．任务描述

将多个数值或字符串创建为一个一维数组。

2．任务实现

（1）程序前面板设计

新建 VI，切换到 LabVIEW 的前面板窗口，通过控件选板向程序前面板添加控件。

1）添加一个数组控件：控件→数组、矩阵与簇→数组。标签为“数值数组”。

2）添加一个数值显示控件：控件→数值→数值显示控件。将数值显示控件移到“数值数组”控件数据显示区框架中。将“数值数组”成员数量设置为 3 列。

3）添加一个数组控件：控件→数组、矩阵与簇→数组。标签为“字符串数组”。

4）添加一个字符串显示控件：控件→字符串与路径→字符串显示控件。将字符串显示控件移到“字符串数组”控件数据显示区框架中。将“字符串数组”成员数量设置为 3 列。

设计的程序前面板如图 3-46 所示。

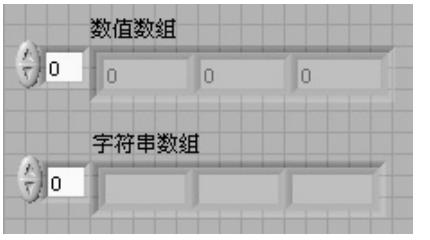


图 3-46　程序前面板

（2）程序框图设计

切换到 LabVIEW 的程序框图窗口，调整控件位置，添加节点与连线。

1）添加两个创建数组函数：函数→数组→创建数组。把鼠标指针放在函数节点下方框上，向下拖动，将输入端口“元素”设置为 3 个。

2）添加三个数值常量：函数→数值→数值常量。将值分别设置为“12”“30”和“5”。

3）添加三个字符串常量：函数→字符串→字符串常量。将值分别设置为“Study”“LabVIEW”和“2015”。

4）将数值常量“12”“30”和“5”分别与创建数组函数（左）的三个输入端口相连。

5）将创建数组函数（左）的输出端口与“数值数组”控件的输入端口相连。

6）将字符串常量“Study”“LabVIEW”和“2015”分别与创建数组函数（右）的三个输入端口相连。

7）将创建数组函数（右）的输出端口与“字符串数组”控件的输入端口相连。

连线后的程序框图如图 3-47 所示。

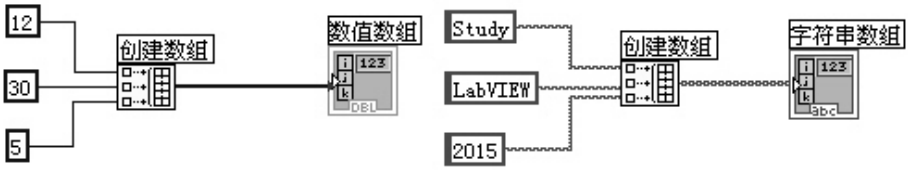


图 3-47　程序框图

（3）运行程序

切换到前面板窗口，单击工具栏中的“运行”按钮，运行程序。

本例中将三个数值创建为一个一维数值数组；将三个字符串创建为一个一维字符串数组。

程序运行界面如图 3-48 所示。

（三）任务 3

1．任务描述

将多个一维数组创建为一个二维数组。

2．任务实现

（1）程序前面板设计

新建 VI，切换到 LabVIEW 的前面板窗口，通过控件选板给程序前面板添加控件。

1）添加一个数组控件：控件→数组、矩阵与簇→数组。设置标签为“数组”。

2）添加一个数值显示控件：控件→数值→数值显示控件。将数值显示控件移到数组控件数据显示区框架中。先将数组维数设置为 2，再将成员数量设置为 2 行 3 列。

设计的程序前面板如图 3-49 所示。

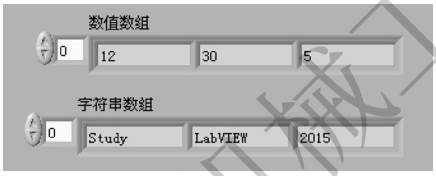


图 3-48　程序运行界面

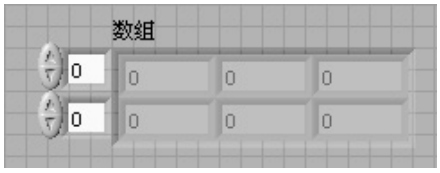


图 3-49　程序前面板

（2）程序框图设计

切换到 LabVIEW 的程序框图窗口，添加节点与连线。

1）添加一个创建数组函数：函数→数组→创建数组。把鼠标指针放在函数节点下方框上，向下拖动，将输入端口“元素”设置为两个。

2）添加一个数组常量：函数→数组→数组常量。

向数组常量数据显示框架中添加数值常量。把鼠标指针放在数组常量右侧方框上，向右拖动，将数组常量列数设置为 3，分别输入数值“1”“2”和“3”。

3）再添加一个数组常量：函数→数组→数组常量。

向数组常量数据显示框架中添加数值常量。把鼠标指针放在数组常量右侧方框上，向右拖动，将数组常量列数设置为 3，分别输入数值“4”“5”和“6”。

4）将两个数组常量分别与创建数组函数的两个输入端口相连。

5）将创建数组函数的输出端口与数组控件的输入端口相连。

连线后的程序框图如图 3-50 所示。

（3）运行程序

切换到前面板窗口，单击工具栏中的“运行”按钮，运行程序。

本例将两个一维数组合成一个二维数组。程序运行界面如图 3-51 所示。

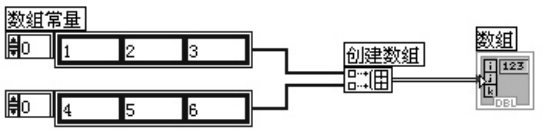


图 3-50　程序框图

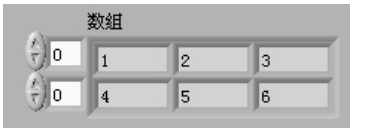


图 3-51　程序运行界面

实训 7　簇数据操作

一、学习目标

掌握簇数据的创建与操作方法。

二、设计任务

（一）任务 1

1．任务描述

将一些基本数据类型的数据元素合成为一个簇数据。

2．任务实现

（1）程序前面板设计

新建 VI，切换到 LabVIEW 的前面板窗口，通过控件选板给程序前面板添加控件。

1）添加一个旋钮控件：控件→数值→旋钮。标签为“旋钮”。

2）添加一个开关控件：控件→布尔→翘板开关。标签为“布尔”。

3）添加一个字符串输入控件：控件→字符串与路径→字符串输入控件。标签为“字符串”。

4）添加一个簇控件：控件→数组、矩阵与簇→簇。标签为“簇”。将簇控件框架放大。

5）分别将一个数值显示控件、一个圆形指示灯控件、一个字符串显示控件放入簇控件框架中。

设计的程序前面板如图 3-52 所示。

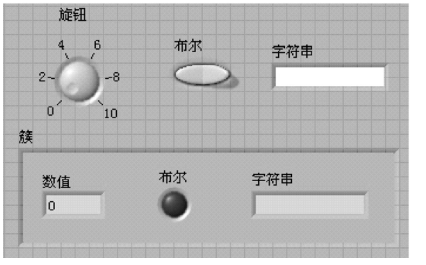


图 3-52　程序前面板

（2）程序框图设计

切换到 LabVIEW 的程序框图窗口，调整控件位置，添加节点与连线。

1）添加一个捆绑函数：函数→簇与变体→捆绑。把鼠标指针放在函数节点下方框上，向下拖动，将输入端口“元素”设置为三个。

2）将旋钮控件、开关控件、字符串输入控件分别与捆绑函数的三个输入端口相连。此时，捆绑函数的三个输入端口数据类型发生变化，自动与连接的数据类型保持一致。

3）将捆绑函数的输出端口“输出簇”与簇控件的输入端口相连。

连线后的程序框图如图 3-53 所示。

（3）运行程序

切换到前面板窗口，单击工具栏中的“连续运行”按钮，运行程序。

转动旋钮，单击布尔开关，输入字符串，单击界面空白处，在簇数据中显示变化结果。

程序运行界面如图 3-54 所示。

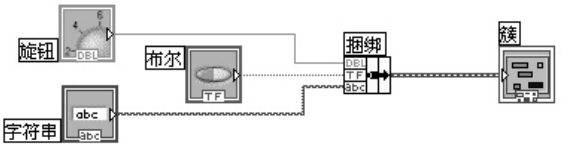


图 3-53　程序框图

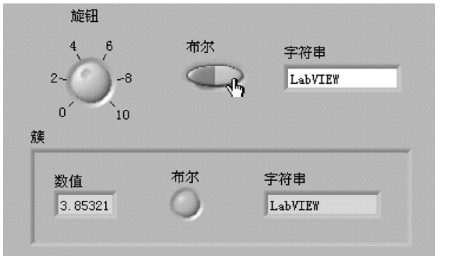


图 3-54　程序运行界面

（二）任务 2

1．任务描述

将一个簇中的每个数据成员进行分解，并将分解后的数据成员作为函数的结果输出。

2．任务实现

（1）程序前面板设计

新建 VI，切换到 LabVIEW 的前面板窗口，通过控件选板给程序前面板添加控件。

1）添加一个簇控件：控件→数组、矩阵与簇→簇。标签为“簇”。将簇控件框架放大。

2）分别将一个旋钮控件、一个数值输入控件、一个布尔开关控件、一个字符串输入控件放入簇控件框架中。

3）添加两个数值显示控件：控件→数值→数值显示控件。将标签分别改为“旋钮输出”“数值输出”。

4）添加一个指示灯控件：控件→布尔→圆形指示灯。标签改为“布尔输出”。

5）添加一个字符串显示控件：控件→字符串与路径→字符串显示控件。标签改为“字符串输出”。

设计的程序前面板如图 3-55 所示。

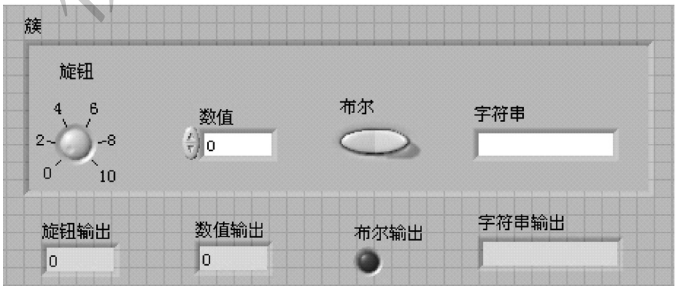


图 3-55　程序前面板

（2）程序框图设计

切换到 LabVIEW 的程序框图窗口，调整控件位置，添加节点与连线。

1）添加一个解除捆绑函数：函数→簇与变体→解除捆绑。

2）将簇控件的输出端口与解除捆绑函数的输入端口“簇”相连。

当一个簇数据与解除捆绑函数的输入端口相连时，其输出端口数量和数据类型自动与簇数据成员一一对应。

3）将解除捆绑函数的输出端口“旋钮”“数值”“布尔”“字符串”分别与“旋钮输出”控件、“数值输出”控件、“布尔输出”控件、“字符串输出”控件的输入端口相连。

连线后的程序框图如图 3-56 所示。

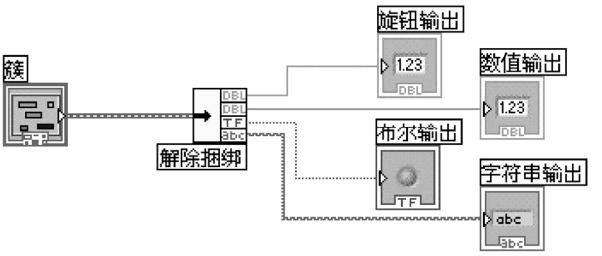


图 3-56　程序框图

（3）运行程序

切换到前面板窗口，单击工具栏中的“连续运行”按钮，运行程序。

在簇数据中转动旋钮、改变数值大小、单击布尔开关、输入字符串，单击界面空白处，旋钮输出值、数值输出值、布尔输出值、字符串输出值发生同样变化。

程序运行界面如图 3-57 所示。

图 3-57　程序运行界面