

LabVIEW™

LabVIEW 基础

全球技术支持及产品信息

ni.com

National Instruments Corporate 总部

11500 North Mopac Expressway Austin, Texas 78759-3504 USA 电话: 512 683 0100

全球办事处

澳大利亚 1800 300 800, 奥地利 43 0 662 45 79 90 0, 比利时 32 0 2 757 00 20,
巴西 55 11 3262 3599, 加拿大 800 433 3488, 中国 86 21 6555 7838, 捷克共和国 420 224 235 774,
丹麦 45 45 76 26 00, 芬兰 385 0 9 725 725 11, 法国 33 0 1 48 14 24 24, 德国 49 0 89 741 31 30,
印度 91 80 41190000, 以色列 972 0 3 6393737, 意大利 39 02 413091, 日本 81 3 5472 2970,
韩国 82 02 3451 3400, 黎巴嫩 961 0 1 33 28 28, 马来西亚 1800 887710, 墨西哥 01 800 010 0793,
荷兰 31 0 348 433 466, 新西兰 0800 553 322, 挪威 47 0 66 90 76 60, 波兰 48 22 3390150,
葡萄牙 351 210 311 210, 俄罗斯 7 095 783 68 51, 新加坡 1800 226 5886, 斯洛文尼
亚 386 3 425 4200, 南非 27 0 11 805 8197, 西班牙 34 91 640 0085, 瑞典 46 0 8 587 895 00,
瑞士 41 56 200 51 51, 台湾 866 02 2377 2222, 泰国 662 992 7519, 英国 44 0 1635 523545

如需更多关于技术支持的信息, 请查阅 “[技术支持和专业服务](#)” 附录。如需对 National Instruments 文档提出任何意见或建议, 请登录 National Instruments 网站 ni.com/info 并输入代码 feedback。

重要信息

保证书

发货日起 90 天内，National Instruments 保证其软件载体不会因材料或制作方面的问题导致无法执行编程指令。发货日以发票或其它有关证明文件为准。在此期间内，如 National Instruments 收到有关该问题的通知，将选择进行维修或更换无法执行编程指令的软件载体。National Instruments 不保证软件的运行不中断或完全无误。

任何设备获取保证服务前，必须在外包装上明确标注有从厂家获取的商品返修授权（RMA）编号。对于保证书担保的货物，National Instruments 将承担货物返还的运费。

National Instruments 确保本文件中信息的准确性。本文件已经严格审阅以确保其技术方面的准确性。如出现技术或印刷错误，National Instruments 保留对本文件后续版本的修改权，而毋须事先通知本版本的持有人。如发现错误，用户应垂询 National Instruments。National Instruments 在任何情况下均无须对由本文件或本文件中信息所引起或与之相关的任何损害承担责任。

除本文另有明确规定，National Instruments 不作其它任何明示或暗示的保证并明确拒绝适销性或针对特定目的适用性的任何保证。因 National Instruments 的过错或疏忽而导致的赔偿应限于客户所支付的金额范围之内。即使已被告知相关可能性，National Instruments 也不对数据丢失、利润损失、使用产品导致的损害，偶然或间接损害承担责任。National Instruments 的此项有限责任条款适用于任何形式的法律程序，无论是违反合同、侵权行为（包括疏忽）或其它。任何针对 National Instruments 的诉讼必须在诉讼事由发生起一年内提起。National Instruments 对其有效控制外的原因引起的任何行事延误不承担责任。本文中规定的保证不包含由以下原因引起的损害、缺陷、故障或服务方面的问题：用户未能遵守 National Instruments 有关安装、操作或维护方面的指示；用户对产品进行修改；用户对产品的滥用、误用或疏忽行为、停电或功率骤增、火灾、洪灾、事故、第三方行为，或有效控制以外的其它事件。

版权

根据版权法，未经 National Instruments Corporation 事先书面同意，本发行物不得以任何形式（包括电子或机械形式）进行全部或部分复制或传播，包括影印、录制、储存于任何信息检索系统中，或翻译。

National Instruments 公司尊重他方的知识产权，也恳请我们的用户能给予同样的尊重。NI 软件受版权和其他知识产权法律的保护。当 NI 软件被用来生产复制属于他方的软件或其他资料时，请确保您仅可在符合任何有效许可证条款或其他法律限制的前提下，以 NI 软件生产复制该资料。

USI (Xerxes C++、ICU 和 HDF5) 中使用的组件适用以下版权。关于使用条件和免责条款，见 USICopyrights.chm。

本产品包括由 Apache Software Foundation (www.apache.org) 开发的软件。

Copyright © 1999 The Apache Software Foundation. 版权所有。

Copyright © 1995–2003 International Business Machines Corporation and others. 版权所有。

NCSA HDF5 (Hierarchical Data Format 5) 软件库和工具。

Copyright 1998, 1999, 2000, 2001, 2003 by the Board of Trustees of the University of Illinois. 版权所有。

商标

National Instruments、NI、ni.com 和 LabVIEW 为 National Instruments Corporation 的商标。有关 National Instruments 商标的详细信息见 ni.com/legal 上的 *Terms of Use* 部分。

此处所提及的其它产品和公司名称为其各自公司的商标或商业名称。

FireWire® 为 Apple Computer, Inc. 的注册商标。此处所提及的其它产品和公司名称为其各自公司的商标或商业名称。

National Instruments Alliance Partner Program 的成员为独立于 National Instruments 的商业实体，与 National Instruments 无代理、合伙或合资关系。

专利权

关于 National Instruments 产品的专利权，见软件中 **帮助»专利信息**，CD 中 patents.txt 文档，或登录 ni.com/patents。

使用 NATIONAL INSTRUMENTS 产品注意事项

(1) 对某些外科移植手术设备或关键救生系统而言，运行故障可能导致严重的人身伤害。National Instruments 产品设计中未涵盖适用于上述外科移植手术设备或任何关键救生系统的组件，也未经与此相关的可靠性测试。

(2) 在包括上述情况在内的任何实际应用中，软件产品运行的可靠性可能受到不利因素影响，包括但不限于以下因素：供电不稳定、计算机硬件故障、计算机操作系统与软件的兼容性、编码器与应用软件开发工具的兼容性、安装错误、软硬件兼容性问题、电子监控或控制设备故障或失灵、电子设备的短暂时故障（硬件和/或软件）、意外使用或误用、用户或应用设计师操作失误（这些不利因素以下统称“系统故障”）。在任何应用中，如系统故障将可能对财产或人身安全造成伤害（包括人身伤害和死亡），考虑

到其可能存在的系统故障风险，不应仅依赖于某一种电子系统。为避免受损、伤害或死亡，用户或应用设计师必须采取合理谨慎的措施对系统故障采取保护措施，包括备份或关闭机制等。由于每套最终用户的系统均为定制并与 National Instruments 的测试平台有差异，且由于用户或应用设计师可能将 National Instruments 产品与其它产品一起使用，而 National Instruments 之前未对此进行测试或预计，因此当 National Instruments 产品与其它系统或程序共同使用时，用户或应用设计师应对测试和验证 National Instruments 产品的适用性承担最终责任，包括但不限于该系统和程序的合理设计、流程和安全等级。

目录

关于本用户手册

行文规范	xiii
------------	------

第 1 章

LabVIEW 简介

LabVIEW 文档资源	1-1
LabVIEW 帮助	1-1
印刷文档	1-2
自述文件	1-2
LabVIEW VI 模板、VI 范例和工具	1-3
LabVIEW VI 模板	1-3
LabVIEW VI 范例	1-3
用于 DAQ 配置的 LabVIEW 工具 (Windows)	1-3

第 2 章

虚拟仪器简介

前面板	2-1
程序框图	2-2
接线端	2-2
节点	2-3
连线	2-3
结构	2-3
图标和连线板	2-4
VI 和子 VI 的应用和自定义	2-4

第 3 章

LabVIEW 编程环境

启动窗口	3-1
控件选板	3-1
函数选板	3-2
浏览控件和函数选板	3-2
工具选板	3-2
菜单和工具栏	3-3
菜单	3-3
快捷菜单	3-3
自定义快捷方式	3-3
VI 工具栏	3-4
项目浏览器窗口工具栏	3-4
即时帮助窗口	3-4
项目浏览器窗口	3-5

导航窗口	3-5
自定义工作环境	3-5
自定义控件和函数选板	3-5
工作环境设置	3-6

第 4 章 创建前面板

前面板控件	4-1
控件样式	4-1
新式及经典控件	4-1
系统控件	4-1
数值显示框、滑动杆、滚动条、旋钮、转盘和时间标识	4-2
数值控件	4-2
滑动杆控件	4-2
滚动条控件	4-2
旋转型控件	4-3
时间标识控件	4-3
图形和图表	4-3
按钮、开关和指示灯	4-3
单选按钮控件	4-4
文本输入框、标签和路径显示框	4-4
字符串控件	4-4
组合框控件	4-4
路径控件	4-4
数组、矩阵及簇控件	4-5
列表框、树形控件和表格	4-5
列表框	4-5
树形控件	4-5
表格	4-5
下拉列表和枚举控件	4-6
下拉列表控件	4-6
枚举控件	4-6
容器控件	4-6
选项卡控件	4-6
子面板控件	4-6
I/O 名称控件	4-7
波形控件	4-7
数字波形控件	4-7
数字数据控件	4-7
对象或应用程序的引用	4-8
.NET 与 ActiveX 控件 (Windows)	4-8

配置前面板对象	4-8
显示和隐藏可选部件	4-9
输入控件和显示控件的相互转换	4-9
替换前面板对象	4-9
配置前面板	4-9
为对象上色	4-9
对齐和分布对象	4-10
组合和锁定对象	4-10
调整对象大小	4-10
在不改变窗口大小的情况下增加前面板空间	4-11
添加标签	4-11
文本特性	4-11
设计用户界面	4-12
使用前面板控件	4-12
设计对话框	4-12

第 5 章

创建程序框图

程序框图对象	5-1
程序框图接线端	5-1
输入控件和显示控件的数据类型	5-2
常量	5-2
程序框图节点	5-3
多态 VI 和函数	5-3
函数概述	5-3
向函数添加接线端	5-4
内置 VI 和函数	5-4
Express VI	5-4
使用连线连接程序框图各对象	5-4
连线的外观和结构	5-4
连接对象	5-5
转折连线	5-6
撤消连线	5-6
自动连接对象	5-6
选择连线	5-6
纠正断线	5-6
强制转换点	5-7
程序框图数据流	5-7
数据依赖关系和人工数据依赖关系	5-8
数据依赖关系不存在	5-9
数据流参数	5-9
数据流和内存管理	5-10
设计程序框图	5-10

第 6 章

运行和调试 VI

运行 VI.....	6-1
纠正断开的 VI	6-2
查找 VI 断开的原因为.....	6-2
VI 断开的常见原因.....	6-2
调试技术.....	6-3
高亮显示执行过程.....	6-3
单步执行	6-3
探针工具	6-4
断点	6-4
错误处理.....	6-4
错误簇.....	6-5
使用 While 循环处理错误	6-6
用条件结构进行错误处理.....	6-6

第 7 章

创建 VI 和子 VI

查找范例	7-1
使用内置 VI 和函数	7-1
创建子 VI.....	7-1
创建图标	7-2
设置连线板	7-2
选中部分程序框图创建子 VI	7-3
设计子 VI 的前面板.....	7-3
查看 VI 的层次结构.....	7-3
多态 VI.....	7-4
保存 VI.....	7-5
VI 命名	7-5
保存为前期版本	7-5
自定义 VI	7-5

第 8 章

循环和结构

For 循环和 While 循环结构.....	8-2
For 循环.....	8-2
While 循环	8-3
控制定时时间.....	8-4
自动索引循环.....	8-4
使用自动索引设置 For 循环总数值	8-5
While 循环的自动索引	8-5
使用循环创建数组.....	8-5

循环中的移位寄存器和反馈节点	8-6
移位寄存器	8-6
反馈节点	8-8
循环结构的默认数据	8-9
条件、顺序和事件结构	8-9
条件结构	8-9
分支选择器值和数据类型	8-10
输入和输出隧道	8-10
用条件结构进行错误处理	8-11
顺序结构	8-11
事件结构	8-12

第 9 章

用字符串、数组和簇将数据分组

用字符串将数据分组	9-1
前面板上的字符串	9-1
字符串显示类型	9-1
表格	9-2
字符串的编辑、格式化和解析	9-2
字符串的格式化和解析	9-2
用数组和簇将数据分组	9-3
数组	9-3
限制	9-3
索引	9-3
数组举例	9-4
创建数组输入控件、显示控件和常量	9-6
创建多维数组	9-6
数组函数	9-7
数组的默认数据	9-8
簇	9-8
簇元素顺序	9-9
簇函数	9-9
创建簇输入控件、显示控件和常量	9-9

第 10 章

图形和图表

图形和图表的类型	10-1
波形图和图表	10-1
波形图	10-1
波形图表	10-2
波形数据类型	10-3
XY 图	10-3

强度图和图表.....	10-3
强度图表.....	10-4
强度图	10-5
数字波形图	10-6
数字波形数据类型	10-8
混合信号图	10-8
三维图形	10-9
自定义图形和图表	10-11
多个 X 标尺和 Y 标尺	10-11
自动调整标尺.....	10-11
格式化 X 标尺和 Y 标尺	10-11
图形工具选板.....	10-12
自定义图形和图表的外观.....	10-12
自定义图形	10-12
图形游标.....	10-13
图形注释.....	10-14
在图形绘图区域内绘图	10-15
自定义三维图形.....	10-16
自定义数字波形图	10-16
自定义图表	10-17
配置图表历史长度	10-17
配置图表更新模式	10-17
曲线的层叠显示和分格显示	10-18

第 11 章 文件 I/O

文件 I/O 基础.....	11-1
选择文件 I/O 格式	11-2
用于常用文件 I/O 操作的 VI 和函数	11-2
使用存储 VI.....	11-4
创建文本文件和电子表格文件	11-5
格式化文件以及将数据写入文件	11-6
从文件中扫描数据.....	11-6
创建二进制文件	11-6
创建数据记录文件	11-6
写入波形至文件	11-7
从文件中读取波形	11-7

第 12 章

编制 VI 说明信息和打印 VI

 编制 VI 说明信息 12-1

 打印 VI 12-2

附录 A

技术支持和专业服务

词汇表

索引

关于本用户手册

在阅读本手册之前，请先参考《LabVIEW 入门》(Getting Started with LabVIEW)，这将有助于您初步了解 LabVIEW 图形化编程环境，掌握 LabVIEW 中创建数据采集和仪器控制程序的各项功能。

本文档包括 LabVIEW 的编程理论、技巧和功能，介绍了用于创建测试测量、数据采集、仪器控制、数据记录、测量分析和报表生成等各类应用程序的 VI 和函数。

LabVIEW Help 中包含本手册的全部内容。阅读本手册时如需查找某个内容更为详细的介绍，请参阅 LabVIEW Help。

本手册不会对每个选板、工具、菜单、对话框、输入控件、显示控件、内置 VI 和函数作详尽的描述。如需详细内容，或需了解 LabVIEW 各项功能的分步操作指导以及如何创建具体应用程序，请参阅 LabVIEW Help。关于 LabVIEW Help 的更多信息见第 1 章 LabVIEW 简介中的 LabVIEW 文档资源 (LabVIEW Documentation Resources) 一节。

行文规范

本手册中的行文规范：

»

» 表示通过嵌套菜单和对话框选项进行选择。**文件 » 页面设置 » 选项**，表示先下拉**文件**菜单，再选择**页面设置**，然后从对话框中选择**选项**。



该提示符号提醒您注意参考信息。



该提示符号提醒您注意重要信息。



该警告符号表示提醒采取预防措施以防受伤、避免数据丢失或系统崩溃。

粗体

粗体文本表示软件中的必选项，如菜单和对话框选项。粗体文本也表示参数名称、前面板上的输入控件和显示控件、对话框、对话框的一部分、菜单名称和选板名称。

斜体

斜体表示变量、强调、交叉引用或重要概念介绍。同时它也可以为占位符，表示须由用户填写的文字或数值。

等宽字体

等宽字体表示用户必须从键盘输入的文字、部分代码、程序范例和语法范例。该字体也用于对磁盘驱动器名称、路径、目录、程序、子程序、设备名、运算、变量、文件名和扩展名的命名。

等宽粗体

等宽粗体表示由计算机在屏幕上自动生成的消息和响应。同时用于强调不同于其他范例的代码行。

等宽斜体

等宽斜体为占位符，表示须由用户填写文字或数值。

平台

平台字体表示特定的平台，下文所提的内容只应用于该平台。

单击右键

(Mac OS) 按 <Command> 键并单击，相当于单击右键。

LabVIEW 简介

LabVIEW (Laboratory Virtual Instrument Engineering Workbench) 是一种用图标代替文本行创建应用程序的图形化编程语言。传统文本编程语言根据语句和指令的先后顺序决定程序执行顺序，而 LabVIEW 则采用数据流编程方式，程序框图中节点之间的数据流向决定了 VI 及函数的执行顺序。VI 指虚拟仪器，是 LabVIEW 的程序模块。

LabVIEW 提供很多外观与传统仪器（如示波器、万用表）类似的控件，可用来方便地创建用户界面。用户界面在 LabVIEW 中被称为前面板。使用图标和连线，可以通过编程对前面板上的对象进行控制。这就是图形化源代码，又称 G 代码。LabVIEW 的图形化源代码在某种程度上类似于流程图，因此又被称作程序框图代码。

如需开发特定程序，可购买各类附加软件工具包。所有工具包都可与 LabVIEW 无缝集成。关于工具包的详细信息，请访问 NI 网站 [ni.com\toolkits](http://ni.com/toolkits)。

LabVIEW 文档资源

LabVIEW 附带全面的参考文档，均有网页和印刷品两种版本，供 LabVIEW 初级或高级用户使用。

LabVIEW 帮助

LabVIEW 帮助 包含 LabVIEW 编程理论、编程分步指导以及 VI、函数、选板、菜单和工具的参考信息。

LabVIEW 帮助 中详细列出 National Instruments 网站中技术支持资源的链接，如 NI 开发者园地 (NI Developer Zone)、知识库 (KnowledgeBase)、产品手册文库等。

选择 **帮助** » **搜索 LabVIEW 帮助** 可打开 *LabVIEW 帮助*。同时还可以在 *LabVIEW 帮助* 中选择打印所需的帮助主题。

关于打印帮助的详细信息见 *LabVIEW 帮助*。



注

(Mac OS) 建议使用 Safari 1.0 或更高版本、Firefox 1.0.2 或更高版本浏览 *LabVIEW 帮助*。**(Linux)** 建议使用 Mozilla 1.2 或更高版本、Firefox 1.0.2 或更高版本浏览 *LabVIEW 帮助*。

安装 LabVIEW 附加软件（如工具包、模块、驱动程序）后，附加软件的相关文档将出现在 *LabVIEW 帮助* 或一个独立的帮助系统中，选择 **帮助 » 附加软件帮助**，上述 **附加软件帮助** 即是附加软件的独立帮助系统。

印刷文档

使用 LabVIEW 时，可参考以下印刷文档：

- *LabVIEW 入门指南*—帮助您熟悉 LabVIEW 图形化编程环境，掌握一些创建数据采集和仪器控制应用程序的 LabVIEW 功能。
- *LabVIEW 快速参考指南*—提供帮助文档资源、快捷键、数据类型及编辑、执行、调试工具的相关信息。
- *LabVIEW 基础*—包括 LabVIEW 的编程理论、技巧、功能、VI 和函数，用于创建测试测量、数据采集、仪器控制、数据记录、测量分析和报表生成等各类程序。*LabVIEW 帮助* 包含本手册的所有内容。
- *LabVIEW 发行说明*—介绍如何安装和卸载 LabVIEW，以及 LabVIEW 软件（包括 LabVIEW 应用程序生成器）对系统的要求。
- *LabVIEW 升级说明*—说明如何在 Windows、Mac OS 和 Linux 上将 LabVIEW 升级到最新版本。升级说明还介绍了升级后的新功能和升级时可能出现的问题。

上述文档均有 PDF 版本，存放在 labview\manuals 目录下。正常显示该 PDF 文档需 Adobe Reader 5.0.5 或更高版本。如需在所有 LabVIEW 用户手册的 PDF 文档进行搜索操作，请安装带有 Search and Accessibility 6.x 或更高版本的 Adobe Reader。**(Mac OS)** 需安装带 Search and Accessibility 6.x 或更高版本的 Adobe Reader 才能正常显示该 PDF 文档。

请登录 Adobe Systems Incorporated 网站 www.adobe.com 下载 Acrobat Reader。关于文档资源更新的详细信息，请登录 ni.com/manuals 查阅 National Instruments 产品手册文库。

自述文件

使用 LabVIEW 时，可参考以下自述文件：

- *LabVIEW 自述文件(Readme)*—介绍 LabVIEW 的最新信息，包括安装和升级、兼容信息、升级改动以及现存问题的记录。请选择 **开始 » 程序 » National Instruments » LabVIEW 8.2 » Readme**，打开 readme.html，或在 labview\readme 目录下直接打开 readme.html 文件，查看 *LabVIEW 自述文件*。
- *LabVIEW 应用程序生成器自述文件 (LabVIEW Application Builder Readme)*—说明如何安装 LabVIEW 应用程序生成器；生成器包含在 LabVIEW 专业版开发系统中，也可单独购买。请选择 **开始 » 程序 » National Instruments » LabVIEW 8.2 » Readme**，打开

readme.html，或者在 labview\readme 目录下直接打开 readme.html 文件，查看 *LabVIEW 自述文件*。

LabVIEW VI 模板、VI 范例和工具

初学者可借助 LabVIEW VI 模板、VI 范例和工具设计和创建 VI。

LabVIEW VI 模板

LabVIEW 内置 VI 模板中包括用来创建一般测量应用程序所必需的子 VI、函数、结构和前面板对象。VI 模板打开时为“未命名 VI”，应重新保存。请选择 **文件** » **新建** 打开 **新建** 对话框，对话框中列出了所有的内置 VI 模板。或在 **启动** 窗口中单击 **新建**，打开 **新建** 对话框。

LabVIEW VI 范例

LabVIEW 在数百个 VI 范例中搜索出需用到的 VI，并将这些 VI 整合到您创建的 VI 中。用户可修改范例使其适合某种应用，或将一个或多个范例复制并粘贴到创建的 VI 中。请选择 **帮助** » **查找范例** 查看或搜索 VI 范例。

关于其它 VI 范例，请访问 NI 开发者园地 ni.com/zone。

在 *LabVIEW 帮助* 的某些 VI 和函数介绍主题页面的下方单击 **打开范例** 和 **浏览相关范例** 按钮，也可访问 VI 范例。单击 **打开范例** 可打开与该主题相关的 VI 范例。单击 **浏览相关范例** 可打开 NI 范例搜索器，显示相关 VI 范例。

也可在程序框图或锁定选板中右键单击 VI 或函数，在快捷菜单中选择 **范例**，打开帮助主题，其中包含了该 VI 或函数的范例链接。

用于 DAQ 配置的 LabVIEW 工具 (Windows)

使用 LabVIEW 中的 Measurement & Automation Explorer (MAX) 配置测量设备。请选择 **工具** » **Measurement & Automation Explorer** 打开 MAX，配置 NI 软硬件。

关于管理其它类型仪器的相关信息，请打开 *LabVIEW 帮助*，在 **目录** 栏中查阅 **仪器控制**。

使用 DAQ 助手 (DAQ Assistant) 通过图形界面配置通道或测量任务。只有在安装 NI-DAQmx 后，DAQ 助手才会在 **函数** 选板上显示。关于安装 NI-DAQmx 的详细信息见 *DAQ Getting Started Guide*。可通过以下几种方式打开 DAQ 助手：

- 将 DAQ 助手 Express VI 置于程序框图中。
- 右键单击 DAQmx 全局通道控件，在快捷菜单中选择 **新建通道 (DAQ 助手)**。右键单击 DAQmx 任务控件，在快捷菜单中选择 **新建任务**。

(**DAQ 助手**)。右键单击 DAQmx 测量刻度控件，在快捷菜单中选择 **新建刻度 (DAQ 助手)**。

- 打开 Measurement & Automation Explorer，在 **Configuration** 目录树下选择 **Data Neighborhood** 或 **Scales**。单击 **新建** 按钮。配置 NI-DAQmx 通道、任务和标尺。

虚拟仪器简介

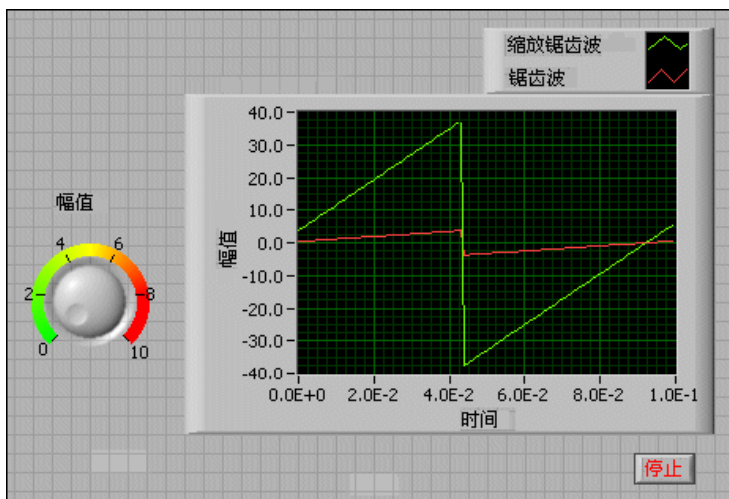
LabVIEW 程序又称虚拟仪器，即 VI，其外观和操作均模仿现实仪器，如示波器和万用表。每个 VI 都使用函数从用户界面或其它渠道获取信息输入，然后将信息显示或传输至其它文件或计算机。

VI 由以下三部分构成：

- **前面板**—即用户界面。
- **程序框图**—包含用于定义 VI 功能的图形化源代码。
- **图标和连线板**—用以识别 VI 的接口，以便在创建 VI 时调用另一个 VI。当一个 VI 应用在其它 VI 中，则称为子 VI。子 VI 相当于文本编程语言中的子程序。

前面板

前面板是 VI 的用户界面。前面板示例如下：



前面板由输入控件和显示控件组成。这些控件是 VI 的输入输出端口。输入控件是指旋钮、按钮、转盘等输入装置。显示控件是指图表、指示灯等显示装置。输入控件模拟仪器的输入装置，为 VI 的程序框图提供数据。显示控件模拟仪器的输出装置，用以显示程序框图获取或生成的数据。

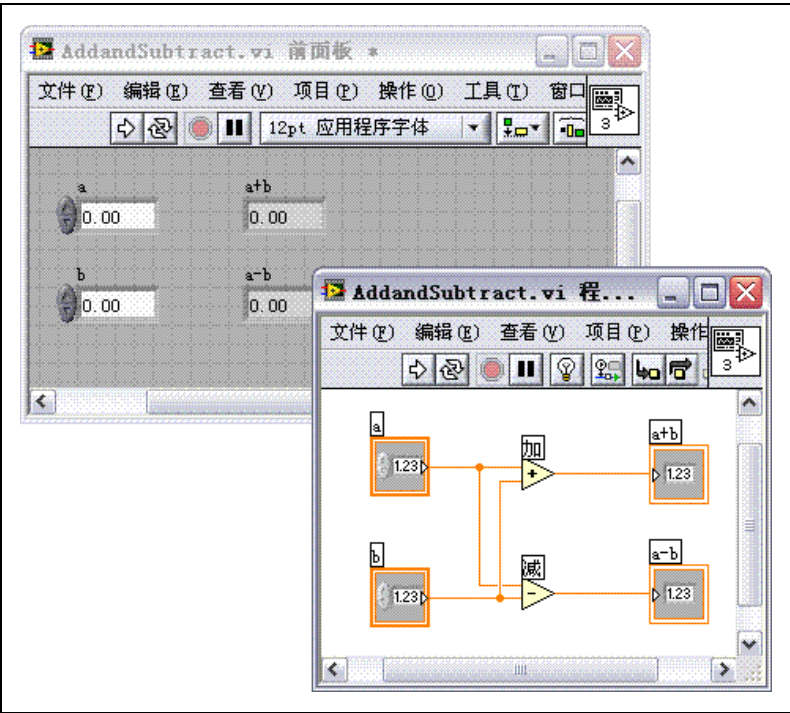
关于前面板的详细信息见第 4 章 *创建前面板*。

程序框图

前面板创建完毕后，便可使用图形化的函数添加源代码来控制前面板上的对象。程序框图是图形化源代码的集合，图形化源代码又称 G 代码或程序框图代码。前面板上的对象在程序框图中显示为接线端。

关于程序框图的详细信息见第 5 章，*创建程序框图*。

下列 VI 中含有接线端、函数和连线等程序框图对象。



接线端

接线端用以表示输入控件或显示控件的数据类型。在程序框图中可将前面板的输入控件或显示控件显示为图标或数据类型接线端。默认状态下，前面板对象显示为图标接线端。如：旋钮接线端代表前面板上的一个旋钮，如下所示。



接线端底部 DBL 代表的是双精度浮点数数据类型。如下所示的 DBL 接线端代表一个双精度浮点数输入控件。



关于数据类型的详细信息见第 5 章 *创建程序框图的输入控件和显示控件的数据类型* 一节。

接线端是在前面板和程序框图之间交换信息的输入输出端口。在前面板输入控件中输入的数据（如上图中的 **a** 和 **b**）通过输入控件接线端进入程序框图。然后，数据进入加和减函数。加减运算结束后，输出新的数据值。数据将传输至显示控件接线端，更新前面板显示控件中的数据（如上图中的 **a + b** 和 **a - b**）。

节点

节点是程序框图上的对象，具有输入输出端，在 VI 运行时进行运算。节点相当于文本编程语言中的语句、运算符、函数和子程序。上图中的加、减函数即是节点。

关于节点的详细信息见第 5 章 *创建程序框图的程序框图节点* 一节。

连线

程序框图中对象的数据传输通过连线实现。在上图中，输入控件和显示控件接线端通过连线实现加减运算。每根连线都只有一个数据源，但可以与多个读取该数据的 VI 和函数连接。不同数据类型的连线有不同的颜色、粗细和样式。断开的连线显示为黑色的虚线，中间有个红色的 x。出现断线的原因有很多，如试图连接数据类型不兼容的两个对象时就会产生断线。

关于连线的详细信息见第 5 章 *创建程序框图中的使用连线连接程序框图各对象* 一节。

结构

结构是文本编程语言中的循环和条件语句的图形化表示。使用程序框图中的结构可对代码块进行重复操作，有条件执行或按特定顺序执行代码。

关于使用结构的详细信息见第 8 章 *循环和结构*。

图标和连线板

创建 VI 的前面板和程序框图后，请创建图标和连线板，以便将该 VI 作为子 VI 调用。图标和连线板相当于文本编程语言中的函数原型。每个 VI 都显示为一个图标，位于前面板和程序框图窗口的右上角，如下图所示。



图标是 VI 的图形化表示，可包含文字、图形或图文组合。如果将一个 VI 当作子 VI 使用，程序框图上将显示代表该子 VI 的图标，可双击图标进行修改或编辑。

关于图标的详细信息见第 7 章 *创建 VI 和子 VI* 中的 *创建图标* 一节。

如需将 VI 当作子 VI 使用，还需创建连线板，如下所示。



连线板用于显示 VI 中所有输入控件和显示控件接线端，类似于文本编程语言中调用函数时使用的参数列表。连线板标明了可与该 VI 连接的输入和输出端，以便将该 VI 作为子 VI 调用。连线板在其输入端接收数据，然后通过前面板的输入控件传输至程序框图的代码中，并从前面板的显示控件中接收运算结果传输至其输出端。

关于连线板的设定见第 7 章 *创建 VI 和子 VI* 中的 *设置连线板* 一节。



注

一个 VI 的接线端应尽量控制在 16 个以内。接线端太多将影响 VI 的可读性和可用性。

VI 和子 VI 的应用和自定义

创建一个 VI，设定图标和连线板，该 VI 即可作为子 VI 调用。

关于子 VI 的详细信息见第 7 章 *创建 VI 和子 VI* 中的 *创建子 VI* 一节。

用户可自定义 VI 的外观和运行方式。

关于自定义 VI 的详细信息见第 7 章 *创建 VI 和子 VI* 中的 *自定义 VI* 一节。

LabVIEW 编程环境

LabVIEW 选板、工具和菜单可用于创建 VI 的前面板和程序框图。

LabVIEW 包含三种选板：**控件**选板、**函数**选板和**工具**选板。LabVIEW 中还有**启动**窗口、**即时帮助**窗口、**项目浏览器**和**导航**窗口。**控件**和**函数**选板可以自定义，同时还可以设置多种工作环境选项。

启动窗口

启动 LabVIEW 时将显示**启动**窗口。在这个窗口中可创建新 VI、选择最近打开的 LabVIEW 文件、查找范例以及打开 *LabVIEW 帮助*。同时还可查看各种信息和资源，如用户手册、帮助主题以及 National Instruments 网站 ni.com 上的各种资源等。

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。可通过选择**查看 » 启动窗口**显示该窗口。

控件选板

控件选板仅位于前面板。**控件**选板包括创建前面板所需的输入控件和显示控件。根据不同输入控件和显示控件的类型，将控件归入不同的子选板中。

关于输入控件和显示控件的详细信息见第 4 章 *创建前面板* 中的 *前面板控件* 一节。

如需显示**控件**选板，请选择**查看 » 控件选板**或在前面板活动窗口单击右键。LabVIEW 将记住**控件**选板的位置和大小，因此当 LabVIEW 重启时选板的位置和大小保持不变。在**控件**选板中可以进行内容修改。

关于自定义**控件**选板的详细信息见本章 *自定义控件和函数选板* 一节。

函数选板

函数选板仅位于程序框图。**函数**选板中包含创建程序框图所需的 VI 和函数。按照 VI 和函数的类型，将 VI 和函数归入不同子选板中。

如需显示**函数**选板，请选择**查看 » 函数选板**或在程序框图活动窗口单击右键。LabVIEW 将记住**函数**选板的位置和大小，因此当 LabVIEW 重启时选板的位置和大小不变。在**函数**选板中可以进行内容修改。





关于自定义**函数**选板的详细信息见本章 *自定义控件和函数选板* 一节。

浏览控件和函数选板

单击选板上的某个对象，然后将其拖放到前面板或程序框图上。也可在选板的 VI 图标上单击右键，从快捷菜单中选择**打开 VI**。

单击**控件**或**函数**选板左边的黑色箭头可展开或折叠选板类别。只有设置选板模式为**类别（标准）**或**类别（图标和文本）**时，才会显示上述黑色箭头。

使用**控件**和**函数**选板工具栏上的下列按钮，可查看、配置选板，搜索控件、VI 和函数。

	返回所属选板 —转到选板的上级目录。单击该按钮并保持光标位置不动，将显示一个快捷菜单，列出当前子选板路径中包含的各个子选板。单击快捷菜单上的子选板名称进入子选板。只有当选板模式设为 图标 、 图标和文本 或 文本 时，才会显示该按钮。
	搜索 —用于将选板转换至搜索模式，通过文本搜索来查找选板上的控件、VI 或函数。选板处于搜索模式时，可单击 返回 按钮，将退出搜索模式，显示选板。
	查看 —用于选择当前选板的视图模式，显示或隐藏所有选板目录，在 文本 和 树形 模式下按字母顺序对各项排序。在快捷菜单中选择 选项 ，可打开 选项 对话框中的 控件 / 函数选板 页，为所有选板选择显示模式。只有当点击选板左上方的图钉标识将选板锁定时，才会显示该按钮。
	恢复选板大小 —将选板恢复至默认大小。只有点击选板左上方的图钉标识锁定选板，并调整 控件 或 函数 选板的大小后，才会出现该按钮。

工具选板

在前面板和程序框图中都可看到**工具**选板。工具选板上的每一个工具都对应于鼠标的的一个操作模式。光标对应于选板上所选择的工具图标。可选择合适的工具对前面板和程序框图上的对象进行操作和修改。

如果自动工具选择已打开，当光标移到前面板或程序框图的对象上时，LabVIEW 将自动从**工具**选板中选择相应的工具。

请选择**查看 » 工具选板**打开**工具**选板。LabVIEW 将记住**工具**选板的位置和大小，因此当 LabVIEW 重启时选板的位置和大小保持不变。



提示 按 <Shift> 键并单击右键，光标处将显示**工具**选板。

菜单和工具栏

菜单和工具栏用于操作和修改前面板和程序框图上的对象。

菜单

VI 窗口顶部的菜单为通用菜单，同样适用于其它程序，如**打开**、**保存**、**复制**和**粘贴**，以及其它 LabVIEW 的特殊操作。某些菜单选项有快捷键。

(Mac OS) 菜单在屏幕最上方。



注 VI 运行时，有些菜单项不可用。

快捷菜单

所有 LabVIEW 对象均有相关的快捷菜单。创建 VI 时，可使用快捷菜单上的选项改变前面板和程序框图上对象的外观或运行方式。右键单击对象可打开快捷菜单。

(Mac OS) 按 <Command> 键并单击，相当于单击右键。

运行模式下的快捷菜单

VI 运行时或处于运行模式下，所有前面板对象都有一套精简的默认快捷菜单。可使用常用快捷菜单剪切、复制、粘贴对象的内容、将对象的值恢复为默认值或查看该对象的说明。

一些复杂的控件具有附加的菜单项。例如，旋钮的快捷菜单中包含添加指针，修改刻度显示等菜单项。

自定义快捷方式

可改变和自定义 LabVIEW 的默认快捷方式。选择**工具 » 选项**，打开**选项**对话框。从**类别**列表中选择**菜单快捷方式**，可设置 VI 菜单项的快捷方式。自定义快捷方式仅对程序框图和前面板窗口有效。选项对话框的菜单快捷方式页还列出了 LabVIEW 菜单项的默认快捷方式。

VI 工具栏

工具栏按钮用于运行、中断、终止、调试 VI、修改字体、对齐、组合、分布对象。

关于工具栏按钮的详细信息见第 6 章 *运行和调试 VI*，或参阅 *LabVIEW 帮助* 中完整的工具栏按钮列表和说明。

项目浏览器窗口工具栏

标准、项目、生成和源代码控制 工具栏中的各个按钮可用于执行 LabVIEW 项目的各种操作。工具栏位于 **项目浏览器** 窗口顶端。有时需展开 **项目浏览器** 窗口才能查看所有工具栏。

关于 LabVIEW 项目的详细信息见本章 *项目浏览器窗口* 一节。

即时帮助窗口

将光标移至一个对象上，**即时帮助** 窗口将显示该 LabVIEW 对象的基本信息。VI、函数、常数、结构、选板、属性、方式、事件、对话框和 **项目浏览器** 中的项均有即时帮助信息。**即时帮助** 窗口还可帮助确定 VI 或函数的连线位置。

关于使用 **即时帮助** 进行连线的详细信息见第 5 章 *创建程序框图* 中的 *使用连线连接程序框图各对象* 一节。

选择 **帮助 » 显示即时帮助** 显示 **即时帮助** 窗口。在工具栏中选择 **显示即时帮助窗口**，也可打开 **即时帮助**，如下所示。



(Windows) 按 <Ctrl-H> 键显示该窗口。

(Mac OS) 按 <Command-Shift-H> 键。 **(Linux)** 按 <Alt-H> 键。

即时帮助 窗口可根据内容的多少自动调整大小。也可调整 **即时帮助** 窗口的大小使之最大化。LabVIEW 将记住 **即时帮助** 窗口的位置和大小，因此当 LabVIEW 重启时该窗口的位置和最大尺寸不变。如调整 **即时帮助** 窗口的大小，LabVIEW 将对 **即时帮助** 窗口中的文本自动换行，缩短连线板中的连线的长度，如果窗口太小不能显示全部内容则将输入和输出端在表格中列出。

如 **即时帮助** 窗口中的对象在 *LabVIEW 帮助* 中也有描述，则 **即时帮助** 窗口中会出现一个蓝色的 **详细帮助信息** 链接。也可单击 **即时帮助** 中的 **详细帮助信息** 图标，如下所示。单击该链接或图标可获取更多关于对象的信息。



项目浏览器窗口

项目浏览器窗口用于创建和编辑 LabVIEW 项目。项目用于对 LabVIEW 文件和非 LabVIEW 文件进行归类、创建程序生成规范以及在目标硬件上部署或下载文件。选择**文件»新建项目**，即可打开**项目浏览器窗口**。

导航窗口

导航窗口显示编辑模式下活动前面板或程序框图的全局概况。**导航窗口**用于浏览较大的前面板或程序框图。单击**导航窗口**的某一图像区域，在前面板和程序框图上将显示该区域的相应位置。同时也可单击并拖动**导航窗口**上的图像，这时前面板和程序框图也随之移动。前面板和程序框图中不可见的部分在**导航窗口**中显示为阴影。

选择**查看»导航窗口**打开**导航窗口**。**(Windows)** 按 <Ctrl-Shift-N> 键也可显示该窗口。**(Mac OS)** 按 <Command-Shift-N> 键。**(Linux)** 按 <Alt-Shift-N> 键。



注 只有 LabVIEW 完整版和专业版开发系统才有**导航窗口**。

导航窗口的大小可调节。LabVIEW 将记住**导航窗口**的位置和大小，因此当 LabVIEW 重启时，该窗口的位置和大小不变。

自定义工作环境

可自定义**控件**和**函数**选板，也可用**选项**对话框选择选板模式并设置其它工作环境选项。

自定义控件和函数选板

可用以下方式自定义**控件**和**函数**选板。

- 配置**编辑控件和函数选板**对话框，重新排列内置选板、创建或移动子选板。选择**工具»高级»编辑选板**，显示**编辑控件和函数选板**对话框。右键单击需修改的选板，从快捷菜单中进行选择。
- 函数**选板中的项可以添加到“收藏类别”中。在锁定的**函数**选板上，右键单击对象并从快捷菜单中选择**添加项至收藏**。在**类别（标准）**和**类别（图标和文本）**模式下，也可展开一个选板以显示其子选板，右键单击该子选板的标题并从快捷菜单中选择**添加子选板至收藏**。

工作环境设置

如需自定义 LabVIEW，请选择**工具 » 选项**。**选项**对话框中可设置前面板、程序框图、路径、性能和磁盘相关选项、对齐网格、选板、撤消操作、调试工具、颜色、字体、打印、**修订历史**等 LabVIEW 属性。

选项对话框左窗格的**类别**列表中列出了可以进行设置的各类选项。

创建前面板

前面板是 VI 的人机界面。创建 VI 时，通常应先设计前面板，然后设计程序框图执行在前面板上创建的输入输出任务。

关于程序框图的详细信息见第 5 章，*创建程序框图*。

输入控件和显示控件用于创建前面板，它们分别是 VI 的交互式输入和输出端口。输入控件指旋钮、按钮、转盘等输入装置。显示控件指图形、指示灯等输出装置。输入控件模拟仪器的输入装置，为 VI 的程序框图提供数据。显示控件模拟仪器的输出装置，显示程序框图获取或生成的数据。

选择**查看 » 控件选板**，显示**控件**选板，从中选取输入控件和显示控件放置在前面板上。

前面板控件

位于前面板**控件**选板上的输入控件和显示控件可用于创建前面板。控件的种类有：数值控件（如滑动杆和旋钮）、图形、图表、布尔控件（如按钮和开关）、字符串、路径、数组、簇、列表框、树形控件、表格、下拉列表控件、枚举控件和容器控件等等。

控件样式

前面板控件有新式、经典和系统三种样式。

新式及经典控件

许多前面板对象具有高彩外观。为了获取对象的最佳外观，显示器最低应设置为 16 色位。

位于**新式**面板上的控件也有相应的低彩对象。**经典**选板上的控件适于创建在 256 色和 16 色显示器上显示的 VI。

系统控件

位于**系统**选板上的系统控件可用在用户创建的对话框中。系统控件专为在对话框中使用而特别设计，包括下拉列表和旋转控件、数值滑动杆、进度条、滚动条、列表框、表格、字符串和路径控件、选项卡控件、树形控件、按钮、复选框、单选按钮和自动匹配父对象背景色的不透明标签。这些控件仅在外观上与前面板控件不同，颜色与系统设置的颜色一致。

系统控件的外观取决于 VI 运行的平台，因此在 VI 中创建的控件外观应与所有 LabVIEW 平台兼容。在不同的平台上运行 VI 时，系统控件将改变其颜色和外观，与该平台的标准对话框控件相匹配。

关于设计对话框的详细信息见本章 *设计对话框* 一节。

数值显示框、滑动杆、滚动条、旋钮、转盘和时间标识

位于**数值**和**经典数值**选板上的数值对象可用于创建滑动杆、滚动条、旋钮、转盘和数值显示框。该选板上还有颜色盒和颜色梯度，用于设置颜色值；以及时间标识，用于设置时间和日期值。数值对象用于输入和显示数值。

数值控件

数值控件是输入和显示数值数据的最简单方式。这些前面板对象可在水平方向上调整大小，以显示更多位数。使用下列方法改变数值控件的值：

- 用操作工具或标签工具单击数字显示框，然后通过键盘输入数字。
- 用操作工具单击数值控件的递增或递减箭头。
- 使用操作工具或标签工具将光标放置于需改变的数字右边，然后在键盘上按向上或向下箭头键。

默认状态下，LabVIEW 的数字显示和存储与计算器类似。数值控件一般最多显示 6 位数字，超过 6 位自动转换为以科学计数法表示。右键单击数值对象并从快捷菜单中选择**格式与精度**，打开**数值属性**对话框的**格式与精度**选项卡，从中配置 LabVIEW 在切换到科学计数法之前所显示的数字位数。

滑动杆控件

滑动杆控件是带有刻度的数值对象。滑动杆控件包括垂直和水平滑动杆、液罐和温度计。可使用下列方法改变滑动杆控件的值：

- 使用操作工具单击或拖曳滑块至新的位置。
- 与数值控件中的操作类似，在数字显示框中输入新数据。

滑动杆控件可以显示多个值。右键单击该对象，在快捷菜单中选择**添加滑块**，可添加更多滑块。带有多个滑块的控件的数据类型为包含各个数值的簇。

关于簇的更多信息见第 9 章*用字符串、数组和簇将数据分组*中的**簇**一节。

滚动条控件

与滑动杆控件相似，滚动条控件是用于滚动数据的数值对象。滚动条控件有水平和垂直滚动条两种。使用操作工具单击或拖曳滑块至一个新的位置，单击递增和递减箭头，或单击滑块和箭头之间的空间都可以改变滚动条的值。

旋转型控件

旋转型控件包括旋钮、转盘、量表和仪表。旋转型对象的操作与滑动杆控件相似，都是带有刻度的数值对象。可使用下列方法改变旋转型控件的值：

- 用操作工具单击或拖曳指针至一个新的位置。
- 与数值控件中的操作类似，在数字显示框中输入新数据。

旋转型控件可显示多个值。右键单击该对象，选择**添加指针**，可添加新指针。带有多个指针的控件的数据类型为包含各个数值的簇。

关于簇的更多信息见第 9 章 *用字符串、数组和簇将数据分组* 中的 *簇* 一节。

时间标识控件

时间标识控件用于向程序框图发送或从程序框图获取时间和日期值。可使用下列方法改变时间标识控件的值：

- 右键单击控件并从快捷菜单中选择**格式与精度**。
- 单击**时间 / 日期浏览**按钮，显示**设置时间和日期**对话框，如下所示。



- 右键单击该控件并从快捷菜单中选择**数据操作 » 设置时间和日期**，显示**设置时间和日期**对话框。
- 右键单击该控件，从快捷菜单中选择**数据操作 » 设置为当前时间**。

图形和图表

位于**图形**和**经典图形**选板上的图形控件可用于以图形和图表的形式绘制数值数据。

关于在 LabVIEW 中使用图形和图表的更多信息见第 10 章，*图形和图表*。

按钮、开关和指示灯

位于**布尔**和**经典布尔**选板上的布尔控件可用于创建按钮、开关和指示灯。布尔控件用于输入并显示布尔值 (TRUE/FALSE)。例如，监控一个实验的温度时，可在前面板上放置一个布尔警告灯，当温度超过一定水平时，即发出警告。

布尔控件有六种机械动作。自定义布尔对象，可创建运行方式与现实仪器类似的前面板。快捷菜单可用来自定义布尔对象的外观，以及单击这些对象时它们的运行方式。

单选按钮控件

单选按钮控件向用户提供一个列表，每次只能从中选择一项。如允许不选任何项，右键单击该控件然后在快捷菜单中选择**允许不选**，该菜单项旁边将出现一个勾选标志。

单选按钮控件为枚举型，所以可用单选按钮控件选择条件结构中的条件分支。

关于枚举控件的详细信息见本章**枚举控件**一节。关于条件结构的详细信息见第 8 章**循环和结构**中的**条件结构**一节。

参考下列使用单选按钮控件的 VI 范例：

文本输入框、标签和路径显示框

位于**字符串和路径**及**经典字符串和路径**选板上的字符串和路径控件可用于创建文本输入框和标签、输入或返回文件或目录的地址。

字符串控件

操作工具或标签工具可用于输入或编辑前面板上字符串控件中的文本。默认状态下，新文本或经改动的文本在编辑操作结束之前不会被传至程序框图。运行时，单击面板的其它位置，切换到另一窗口，单击工具栏上的**确定输入**按钮，或按数字键区的 <Enter> 键，都可结束编辑状态。在主键区按 <Enter> 键将输入回车符。

右键单击字符串控件为其文本选择显示类型，如以密码形式显示或十六进制数显示。

关于字符串显示类型的详细信息见第 9 章**用字符串、数组和簇将数据分组**中的**前面板上的字符串**一节。

组合框控件

组合框控件可用来创建一个字符串列表，在前面板上可循环浏览该列表。组合框控件类似于文本型或菜单型下拉列表控件。但是，组合框控件是字符串型数据，而下拉列表控件是数值型数据。

关于下拉列表控件的详细信息见本章**下拉列表控件**一节。

关于条件结构的详细信息见第 8 章**循环和结构**中的**条件结构**一节。

路径控件

路径控件用于输入或返回文件或目录的地址。**(Windows 和 Mac OS)** 如允许运行时拖放，则可从 Windows 浏览器中拖曳一个路径、文件夹或文件放置在路径控件中。

路径控件与字符串控件的工作原理类似，但 LabVIEW 会根据用户使用操作平台的标准句法将路径按一定格式处理。

数组、矩阵及簇控件

位于**数组、矩阵和簇**及**经典数组、矩阵和簇**选板上的数组、矩阵和簇控件可用于创建数组、矩阵和簇。数组是同一类型数据元素的集合。簇将不同类型的数据元素归为一组。矩阵是若干行列实数或复数数据的集合，用于线性代数等数学操作。

关于数组和簇的详细信息见第 9 章*用字符串、数组和簇将数据分组中的用数组和簇将数据分组*一节。

列表框、树形控件和表格

位于**列表和表格**及**经典列表和表格**选板上的列表框控件用于向用户提供一个可供选择的项列表。

列表框

列表框可配置为单选或多选。多列列表可显示更多条目信息，如大小和创建日期等。

树形控件

树形控件用于向用户提供一个可供选择的层次化列表。用户将输入树形控件的项组织为若干组项或若干组节点。单击节点旁边的展开符号可展开节点，显示节点中的所有项。单击节点旁的符号还可折叠节点。



注

只有在 LabVIEW 完整版和专业版开发系统中才可创建和编辑树形控件。所有 LabVIEW 软件包均可运行含有树形控件的 VI，但不能在基础软件包中配置树形控件。

参考下列使用树形控件的 VI 范例：Directory Hierarchy in Tree
Control VI labview\examples\general\controls\Tree Control
Directory.llb

表格

表格控件可用于在前面板上创建表格。

关于使用表格控件的详细信息见第 9 章*用字符串、数组和簇将数据分组中的表格*一节。

下拉列表和枚举控件

位于**下拉列表和枚举**及**经典下拉列表和枚举**选板上的下拉列表和枚举控件可用来创建可循环浏览的字符串列表。

下拉列表控件

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。下拉列表控件以下拉菜单的形式出现，用户可在循环浏览的过程中作出选择。

下拉列表控件可用于选择互斥项，如触发模式。例如，用户可在下拉列表控件中从连续、单次和外部触发中选择一种模式。

枚举控件

枚举控件用于向用户提供一个可供选择的项列表。枚举控件类似于文本或菜单下拉列表控件，但是，枚举控件的数据类型包括控件中所有项的数值和字符串标签的相关信息，下拉列表控件则为数值型控件。

容器控件

位于**容器**和**经典容器**选板上的容器控件可用于组合控件，或在当前 VI 的前面板上显示另一个 VI 的前面板。**(Windows)** 容器控件还可用于在前面板上显示 .NET 和 ActiveX 对象。

关于 .NET 和 ActiveX 控件的详细信息见本章 *.NET 与 ActiveX 控件 (Windows)* 一节。

选项卡控件

选项卡控件用于将前面板的输入控件和显示控件重叠放置在一个较小的区域内。选项卡控件由选项卡和选项卡标签组成。可将前面板对象放置在选项卡控件的每一个选项卡中，并将选项卡标签作为显示不同页的选择器。

可使用选项卡控件组合在操作某一阶段需用到的前面板对象。例如，某 VI 在测试开始前可能要求用户先设置几个选项，然后在测试过程中允许用户修改测试的某些方面，最后允许用户显示和存储相关数据。

在程序框图上，选项卡控件默认为枚举控件。选项卡控件中的控件接线端与程序框图上的其它控件接线端在外观上是一致的。

关于枚举控件的详细信息见本章 *枚举控件* 一节。

子面板控件

子面板控件用于在当前 VI 的前面板上显示另一个 VI 的前面板。例如，子面板控件可用于设计一个类似向导的用户界面。在顶层 VI 的前面板上放置**上一步**和**下一步**按钮，并用子面板控件加载向导中每一步的前面板。

**注**

只有 LabVIEW 完整版和专业版系统才具有创建和编辑子面板控件的功能。所有 LabVIEW 软件包均可运行含有子面板控件的 VI，但不能在基础软件包中配置子面板控件。

参考下列使用子面板控件的范例：`labview\examples\general\controls\subpanel.llb`。

I/O 名称控件

位于 **I/O** 和 **经典 I/O** 选板上的 I/O 名称控件可将所配置的 DAQ 通道名称、VISA 资源名称和 IVI 逻辑名称传递至 I/O VI，与仪器或 DAQ 设备进行通信。

I/O 名称常量位于 **函数** 选板上。常量是在程序框图上向程序框图提供固定值的接线端。

**注**

所有 I/O 名称控件或常量可在任何平台上使用。这使用户可在任何平台上开发与特定平台设备进行通信的 I/O VI。但是，如果在一个不支持该设备的平台上运行带有特定平台 I/O 控件的 VI，系统将会出错。

(Windows) 工具菜单中的 Measurement & Automation Explorer 可用于配置 DAQ 通道名称，VISA 资源名称和 IVI 逻辑名称。

(Mac OS 和 Linux) 使用与仪器相关的配置程序，配置 VISA 资源名称和 IVI 逻辑名称。关于配置应用程序的详细信息见与仪器相关的文档。

波形控件

波形控件可用于对波形中的单个数据元素进行操作。波形数据类型包括波形的数据、起始时间和时间间隔 (Δt)。

关于波形数据类型的详细信息见第 10 章 *图形和图表* 中的 *波形数据类型* 一节。

数字波形控件

数字波形控件可用于对数字波形中的单个数据元素进行操作。

关于数字波形数据类型的详细信息见第 10 章 *图形和图表* 中的 *数字波形数据类型* 一节。

数字数据控件

数字数据控件显示行列排列的数字数据。数字数据控件可用于创建数字波形或显示从数字波形中提取的数字数据。将数字波形数据输入控件连接至数字数据显示控件，可查看数字波形的采样和信号。

对象或应用程序的引用

位于**引用句柄**和**经典引用句柄**选板上的引用句柄控件可用于对文件、目录、设备和网络连接进行操作。控件引用句柄用于将前面板对象信息传送给子 VI。

引用句柄是对象的唯一标识符，这些对象包括文件、设备或网络连接等。打开一个文件、设备或网络连接时，LabVIEW 会生成一个指向该文件、设备或网络连接的引用句柄。对打开的文件、设备或网络连接进行的所有操作均使用引用句柄来识别每个对象。引用句柄控件用于将一个引用句柄传进或传出 VI。例如，引用句柄控件可在不关闭或不重新打开文件的情况下修改其指向的文件内容。

由于引用句柄是一个打开对象的临时指针，因此它仅在对象打开期间有效。如关闭对象，LabVIEW 会将引用句柄与对象分开，引用句柄即失效。如再次打开对象，LabVIEW 将创建一个与第一个引用句柄不同的新引用句柄。LabVIEW 将为引用句柄所指的對象分配内存空间。关闭引用句柄，该对象就会从内存中释放出来。

由于 LabVIEW 可以记住每个引用句柄所指的信息，如读取或写入的对象的当前地址和用户访问情况，因此可以对单一对象执行并行但相互独立的操作。如一个 VI 多次打开同一个对象，那么每次的打开操作都将返回一个不同的引用句柄。VI 结束运行时 LabVIEW 会自动关闭引用句柄，但如果用户在结束使用引用句柄时就将其关闭将可以最有效地利用内存空间和其它资源，这是一个良好的编程习惯。关闭引用句柄的顺序与打开时相反。例如，如对象 A 获得了一个引用句柄，然后在对象 A 上调用方法以获得一个指向对象 B 的引用句柄，在关闭时应先关闭对象 B 的引用句柄然后再关闭对象 A 的引用句柄。

.NET 与 ActiveX 控件 (Windows)

位于 **.NET 与 ActiveX** 选板上的 .NET 和 ActiveX 控件用于对常用的 .NET 或 ActiveX 控件进行操作。可添加更多 .NET 或 ActiveX 控件至该选板，供日后使用。选择**工具 » 导入 » .NET 控件至选板**或**工具 » 导入 » ActiveX 控件至选板**，可分别转换 .NET 或 ActiveX 控件集，自定义控件并将这些控件添加至 **.NET 与 ActiveX** 选板。



注

创建 .NET 对象并与之通信需安装 .NET Framework 1.1 Service Pack 1 或更高版本。建议只在 LabVIEW 项目中使用 .NET 对象。

配置前面板对象

属性对话框或快捷菜单可用来配置控件在前面板上的外观和动作。**属性**对话框还可用来配置带即时帮助的前面板控件，在对话框中可一次设置对象的多个属性。使用快捷菜单可快速配置控件的一般属性，不同前面板对象的**属性**

对话框和快捷菜单选项会有所不同。**属性**对话框中包含大多数可通过快捷菜单设置的选项，快捷菜单也包括大多数可用**属性**对话框设置的选项。

右键单击前面板上的控件，然后在快捷菜单中选择**属性**，可打开该对象的**属性**对话框。VI 运行时，不能使用控件的**属性**对话框。

也可创建自定义控件扩展前面板对象。右键单击控件并从快捷菜单中选择**高级 » 自定义**，可自定义控件。将这种自定义输入控件或显示控件保存在某个目录或 LLB 中，就可以在其它前面板上使用该自定义控件。

显示和隐藏可选部件

前面板控件的某些部件可显示或隐藏，如标签、标题和数字显示框。前面板对象**属性**对话框中的**外观**选项卡可用来设置前面板控件的显示部件，或右键单击对象并从快捷菜单中选择**显示项**，然后选择需显示的部件。

输入控件和显示控件的相互转换

LabVIEW 根据**控件**选板中对象的典型用途将对象配置为输入控件或显示控件。例如，将翘板开关放置于前面板上，它会显示为输入控件，因为翘板开关通常是一种输入设备。将指示灯 (LED) 放置在前面板上，它会显示为显示控件，因为指示灯通常是一种输出设备。

有些选板包含同一类型或类对象的输入控件和显示控件。例如，**数值**选板既包含数值输入控件和又包含数值显示控件，因为既可以输入数值，又可以输出数值。

右键单击对象，并在快捷菜单中选择**转换为显示控件**，可将输入控件转换为显示控件。右键单击对象，并在快捷菜单中选择**转换为输入控件**，可以将显示控件转换为输入控件。

替换前面板对象

将一个前面板对象替换为其它输入控件或显示控件。右键单击对象并从快捷菜单中选择**替换**，会出现一个临时**控件**选板。从该临时**控件**选板中选择一个控件，替换前面板上的当前对象。

配置前面板

通过更改前面板对象的颜色、对齐和分布前面板对象等，自定义前面板。

为对象上色

用户可改变许多 LabVIEW 对象的颜色，也可改变大多数前面板对象、前面板窗格和程序框图工作区的颜色。但不能改变系统控件的颜色，因为这些对象的颜色与系统的颜色设置一致。

用上色工具右键单击对象或工作区，可改变前面板对象、前面板窗格和程序框图工作区的颜色。选择**工具 » 选项**，并从**类别**列表中选择**颜色**，可改变一些对象的默认颜色。

颜色会分散注意力，使用户错过重要的信息，所以应尽量合理地上色、用少量颜色并保持颜色的一致性。

对齐和分布对象

选择**编辑 » 启用前面板网格对齐**，在放置对象时通过网格自动对齐对象。选择**编辑 » 禁用前面板网格对齐**，通过可视网格手动对齐对象。按 <Ctrl-#> 键可启用或禁用网格对齐功能。在法语键盘上，按 <Ctrl-~> 键。

(Mac OS) 按 <Command-*> 键。**(Linux)** 按 <Alt-#> 键。

在程序框图上也可使用对齐网格。

打开**工具 » 选项**，然后从**类别**列表中选择**对齐网格**隐藏或自定义网格。

放置对象后如需对齐对象，先选中该对象然后选择工具栏上的**对齐对象**下拉菜单或选择**编辑 » 对齐所选项**。如需均匀排列对象，先选中该对象然后选择工具栏上的**分布对象**下拉菜单或选择**编辑 » 分布所选项**。

组合和锁定对象

定位工具可用来选择需组合和锁定的前面板对象。单击工具栏上的**重新排序**按钮，从下拉菜单中选择**组合**或**锁定**。使用定位工具移动或改变组合对象时，对象的相对位置和相对尺寸保持不变。锁定的对象在前面板上的位置保持不变，只有解锁后才能删除这些对象。可同时组合并锁定对象。除定位工具以外，其它工具都可对组合或锁定的对象进行正常操作。

调整对象大小

大多数前面板对象的大小可调整。将定位工具移到某个大小可变的对象上时，对象周围会出现调节柄或调节圈。调整对象大小时，字体大小不会变化。调整组合内某个对象的大小将同时改变组合内所有对象的大小。

某些对象，如数值控件，其大小只会在水平或垂直方向上发生变化。而在调整其它对象（如旋钮）的大小时，其比例保持不变。调整这些对象的大小时，定位光标在外观上没有任何不同，但对象周围的虚边框只能朝一个方向移动。

调整对象大小时，可手动规定对象尺寸改变的方向。如需限定对象的大小只能在垂直或水平方向发生变化，或者要保持对象的当前比例，需在选中并拖曳调节柄或调节圈时同时按住 <Shift> 键。如需以对象中心为参考点来改变大小，在选中并拖曳调节柄或调节圈的同时按住 <Ctrl> 键。

(Mac OS) 按 <Option> 键。(Linux) 按 <Alt> 键。

如需将多个对象调整为同样大小，选定这些对象然后选择工具栏上的**调整对象大小**下拉菜单。可将所有选中对象调整为最大或最小对象的宽度或高度，也可将所有选中对象调整为以像素为单位的特定大小。如果在前面板上添加分隔栏并创建窗格，按下 <Shift> 键可选择不同窗格中的对象。

在不改变窗口大小的情况下增加前面板空间

无需改变窗口大小就可为前面板增加空间。如需在空间拥挤的对象或组合对象间增加空间，按住 <Ctrl> 键然后用定位工具单击前面板工作区。按住组合键的同时，用鼠标拖曳出需插入的区域大小。

(Mac OS) 按 <Option> 键。(Linux) 按 <Alt> 键。

一个虚线矩形框就是插入空间所在的位置。释放鼠标按钮和组合键，就可在相应位置加入空间。

添加标签

标签是前面板和程序框图对象的标识。

LabVIEW 有两种标签：自带标签和自由标签。自带标签属于某一特定对象，并随对象移动，仅用于注释该对象。自带标签可单独移动，但移动该标签的对象时，标签将随对象移动。自带标签可隐藏，但无法独立于自带标签所属的对象复制或删除自带标签。右键单击数值控件，从快捷菜单中选择**显示项 » 单位标签**，可显示数值控件的独立自带标签，即单位标签。

自由标签不附属于任何对象，用户可独立创建、移动、旋转或删除自由标签。自由标签可用于对前面板和程序框图添加注释。

自由标签也可用于注释程序框图上的代码以及在前面板上列出用户指令。双击空白区域或使用标注工具可创建自由标签，或编辑任何类型的标签。

文本特性

LabVIEW 使用已在计算机上安装的字体。工具栏上的**文本设置**下拉菜单可用于改变文本属性。

文本设置下拉菜单包含以下内置字体：

- **应用程序字体**—用于**控件**选板、**函数**选板及新控件中的文本的默认字体
- **系统字体**—用于菜单的字体
- **对话框字体**—用于对话框文本的字体

如在**文本设置**下拉菜单中作出选择前就已经选择了对象或文本，则选中的所有对象或文本都将更改。如果未选任何对象或文本，则只有默认字体改变。改变默认字体并不会改变现有标签的字体，但只会影响此后创建的标签。

将具有内置字体的 VI 转移到另一个平台上时，内置字体将变为与新平台上最相近的字体。

文本设置下拉菜单具有**大小**、**样式**、**对齐**和**颜色**子菜单项。

设计用户界面

如需将一个 VI 作为用户界面或对话框，前面板的外观和布局非常重要。合理设计前面板，使用户对进行中的操作一目了然。前面板设计应类似于仪器或其它设备。

使用前面板控件

控件是前面板的重要组成部分。在设计前面板时，需考虑用户与 VI 进行交互的方式，合理组合控件。如若干控件是相互关联的，可在其周围加上一个修饰边框或放入一个簇中。位于**修饰**选板上的修饰控件包括方框、线条、箭头等对象，用于组合或分隔前面板上的对象。这些对象仅用于修饰对象，不能显示数据。

设计对话框

选择**文件 » VI 属性**，然后从**类别**下拉菜单中选择**窗口外观**，隐藏菜单栏和滚动条，创建在各个平台上外观和运行均与标准对话框类似的 VI。

如 VI 在屏幕的同一位置连续地出现对话框，应重新组织对话框，使第一个对话框中的按钮与后续对话框中的按钮不在同一直线上。因为用户双击第一个对话框中的按钮时，会无意点击了下一个对话框中的按钮。

在创建的对话框中使用**系统**选板上的系统控件。

创建程序框图

创建前面板后，可通过图形化的函数添加源代码，从而对前面板对象进行控制。程序框图是图形化源代码的集合，图形化源代码又称 G 代码或程序框图代码。

程序框图对象

程序框图对象包括接线端和节点。将各个对象用连线连接便创建了程序框图。接线端的颜色和符号表明了相应输入控件或显示控件的数据类型。常量是程序框图上向程序框图提供固定数据值的接线端。

程序框图接线端

前面板对象在程序框图中显示为接线端。双击程序框图上的一个接线端，则前面板上相应的输入控件或显示控件将高亮显示。

接线端是前面板和程序框图之间交换信息的输入输出端口。输入到前面板输入控件的数据值经由输入控件接线端进入程序框图。运行时，输出数据值经由显示控件接线端流出程序框图而重新进入前面板，最终在前面板显示控件中显示。

LabVIEW 中使用的接线端包括输入控件和显示控件接线端、节点接线端、常量及用于各种结构的接线端。连线则把接线端连接起来，使数据在接线端间传递。右键单击一个程序框图对象，从快捷菜单中选择**显示项 » 接线端**可令接线端显示。再次右键单击该对象，从快捷菜单中选择**显示项 » 接线端**则令接线端隐藏。该快捷菜单项对于可扩展 VI 和函数不可用。

前面板输入控件或显示控件在程序框图上可显示为图标接线端或数据类型接线端。默认状态下，前面板对象显示为图标接线端。例如，旋钮图标接线端代表前面板上的一个旋钮输入控件。如下图所示。



接线端底部的 DBL 表明其数据类型为双精度浮点型。下图所示的 DBL 接线端代表一个双精度浮点型输入控件。



右键单击接线端，取消勾选快捷菜单中的**显示为图标**则仅显示该接线端的数据类型。使用图标接线端不仅可显示前面板对象的数据类型，还可显示前面板对象在程序框图上的类型。使用数据类型接线端则较节省程序框图的空间。



注

由于图标接线端比数据类型接线端大，因此将一个数据类型接线端转换为图标接线端后，可能会无意中覆盖其它程序框图对象。

输入控件接线端的边框比显示控件接线端的边框粗。同时，箭头在前面板接线端上的位置也表明了该接线端是输入控件还是显示控件。输入控件接线端的箭头在右边，显示控件接线端的箭头在左边。

输入控件和显示控件的数据类型

常用的输入控件和显示控件的数据类型包括浮点型、整型、时间标识、枚举型、布尔、字符串、数组、簇、路径、动态、波形、引用句柄和 I/O 名称。关于输入控件和显示控件的数据类型符号和用途，见 *LabVIEW 帮助* 中的完整列表。

接线端的颜色和符号表明了相应输入控件或显示控件的数据类型。许多数据类型有其相应进行数据操作的函数，如位于**字符串**选板的字符串函数，其对应的数据类型为字符串。

符号数值

未定义数据或非预期数据将使后续的所有操作无效。浮点数据操作返回以下两种符号值用以表明错误的计算或无意义的结果：

- NaN（非法数字）表示无效操作所产生的浮点值，如对负数取平方根。
- Inf（无穷）表示超出某数据类型值域的浮点数值。例如，1 被 0 除时产生 Inf。

LabVIEW 可返回 +Inf 或 -Inf。+Inf 表示数据类型的最大值，-Inf 表示数据类型的最小值。

LabVIEW 不检查整数的上溢或下溢条件。

常量

常量是程序框图上向程序框图提供固定数据的接线端。通用常量即有固定值的常量，如 pi (π) 和 Inf (∞)。用户定义常量是在 VI 运行之前被定义和编辑的常量。

常量多位于所在选板的底部或顶部。

右键单击 VI 或函数的输入端，从快捷菜单中选择**创建 » 常量**可创建一个用户定义常量。

用操作工具或标签工具单击常量可编辑常量的值。如自动选择工具已启用，则双击该常量可切换到标签工具，对常量的值进行编辑。

程序框图节点

节点是程序框图上的对象，带有输入输出端，在 VI 运行时进行运算。节点类似于文本编程语言中的语句、运算符、函数和子程序。LabVIEW 有以下类型的节点：

- **函数**—内置的执行元素，相当于操作符、函数或语句。
- **子 VI**—用于另一个 VI 程序框图上的 VI，相当于子程序。
关于在程序框图中使用子 VI 的更多信息，见第 7 章 *创建 VI 和子 VI* 中的 *创建子 VI* 一节。
- **Express VI**—协助常规测量任务的子 VI。Express VI 是在配置对话框中配置的。
关于使用 Express VI 的更多信息见第本章 *Express VI* 一节。
- **结构**—执行控制元素，如 For 循环、While 循环、条件结构、平铺式和层叠式顺序结构、定时结构和事件结构。
关于结构的更多信息见第 8 章，*循环和结构*。

关于程序框图节点的完整列表请参见 *LabVIEW 帮助*。

多态 VI 和函数

多态 VI 和函数会根据输入数据类型不同而自动调整数据类型。与某些 VI 和函数一样，绝大多数的 LabVIEW 结构为多态。

函数多态的程度各不相同：可以是全部或部分输入多态，也可以是完全没有多态输入。有一些函数输入可接收数值或布尔值。有一些函数输入可接收数值或字符串。一些函数的输入不仅接收标量数值，还接收数值数组、数值簇或数值簇数组等。还有一些函数输入仅接收一维数组，即使其数组元素可以是任意数据类型。另外一些函数输入则接收所有数据类型，包括复数值。

关于数组和簇的详细信息见第 9 章 *用字符串、数组和簇将数据分组* 中的 *用数组和簇将数据分组* 一节。

函数概述

函数是 LabVIEW 中最基本的操作元素。**函数**选板上的函数图标使用淡黄色背景色和黑色前景色。函数没有前面板或程序框图，但有连线板。用户不能打开或编辑函数。

向函数添加接线端

某些函数接线端的数目可以改变。例如，要创建一个含有十个元素的数组，就必须向“创建数组”函数添加十个接线端。

使用定位工具分别向上或向下拖动函数的顶部或底部可以为函数添加接线端。也可用定位工具删除函数的接线端，但不能删除已经连好线的接线端。要删除这些接线端必须先删除已存在的连线。

关于用连线连接对象的更多信息见本章 *使用连线连接程序框图各对象* 一节。

内置 VI 和函数

函数选板中还包含 LabVIEW 自带的 VI。在应用程序中将内置 VI 和函数作为子 VI 使用有助于缩短程序的开发时间。单击**函数**选板中的**查看**按钮并从快捷菜单中选择**始终显示类别 » 显示所有类别**，可显示**控件**选板中所有的目录。

关于在程序框图中使用子 VI 的更多信息，见第 7 章 *创建 VI 和子 VI* 中的 *使用内置 VI 和函数* 一节。

关于所有内置 VI 和函数的详细信息见 *LabVIEW 帮助*。

Express VI

Express VI 用于完成常规测量任务。Express VI 的配置通过对话框完成，因此是需要连线数最少的节点。Express VI 的输入输出取决于它的配置。Express VI 在程序框图上以可扩展节点的形式出现，其图标底色为蓝色。

关于使用 Express VI 的更多信息见 *LabVIEW 入门指南*。

使用连线连接程序框图各对象

连线用于在程序框图各对象间传递数据。每根连线都只有一个数据源，但可与多个读取数据的 VI 和函数连接，这与在文本编程语言中传递必需参数相似。必须连接所有需要连接的程序框图接线端。否则 VI 将处于断开状态而无法运行。打开**即时帮助**窗口可获知程序框图节点的哪些接线端需要连接。必需接线端的标签在**即时帮助**窗口中以粗体字显示。

关于断开的 VI 的更多信息见第 6 章，*运行和调试 VI*，*纠正断开的 VI* 部分。

连线的外观和结构

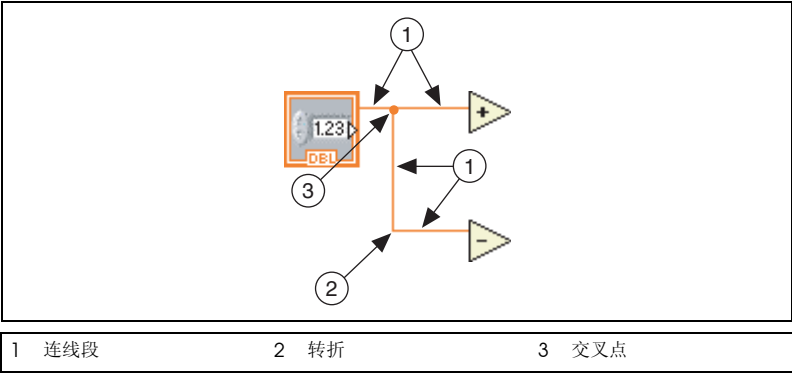
连线的颜色、样式和粗细视其数据类型而异，这与接线端以不同颜色和符号来表示相应输入控件或显示控件的数据类型相似。断开的连线显示为黑色的

虚线，中间有个红色的 x。出现断线的原因有很多，如试图连接数据类型不兼容的两个对象时就会产生断线。断线中间红色 x 任意一边的箭头表明了数据流的方向，而箭头的颜色表明了流过连线数据的数据类型。

关于数据类型的更多信息见本章 *输入控件和显示控件的数据类型* 一节。关于数据流的更多信息见本章 *程序框图数据流* 一节。

当连线工具移到 VI 或函数节点上时，未连线的接线端将出现接线头。接线头表明了每个接线端的数据类型。同时将出现一个提示框，显示接线端的名称。一旦接线端被连接，当连线工具移到其节点时，接线端的接线头便不再出现。

连线段是一条水平或垂直的连线。连线中的转折是两段连线交叉的地方。两段或多段连线的相交点称为交叉点。一个连线分支包含了从交叉点到交叉点、接线端到交叉点或中间没有交叉点的接线端到接线端的所有连线段。下图显示了连线段、转折和交叉点。



连接对象

连线工具可以手动方式为程序框图上不同节点的接线端连线。连线工具的光标点到处即为连线开始的位置。连线工具移到某个接线端上时，接线端将不断闪烁。连线工具移到某个 VI 或函数接线端时，将出现一个提示框，显示接线端的名称。为接线端连线时可能会产生断线。在运行 VI 前必须纠正这些断线。

关于纠正断线的更多信息，见本章 *纠正断线* 一节。

借助**即时帮助**窗口可确定准确的连线位置。将光标移到某个 VI 或函数接线端时，**即时帮助**窗口会列出该 VI 或函数的每一个接线端。**即时帮助**窗口不会显示可扩展 VI 和函数的接线端，如创建数组函数。点击**即时帮助**窗口中的**显示可选接线端和完整路径**按钮可显示连线板的可选接线端。

连线交叉时，第一根连线处会出现一小段空白，表示第一根连线位于第二根连线下。

转折连线

用连线连接接线端时，在垂直或水平方向移动光标可将连线 90 度转折。如需在多个方向转折连线，可先点击鼠标按钮一次以定位连线，再向新的方向移动光标。这样可不断定位连线并将连线接往新方向。

撤消连线

如需取消最后的连线定位点，按 <Shift> 键并点击程序框图上的任意位置。如需中止整个连线操作，右键单击程序框图中的任意位置。

(Mac OS) 按 <Option> 键并单击。(Linux) 单击鼠标中间按钮。

自动连接对象

将选中的对象移到程序框图上其它对象旁时，LabVIEW 将以暂时连线来显示有效的连线方式。将对象放置在程序框图上时，放开鼠标后 LabVIEW 将自动连线。也可对程序框图上已有对象进行自动连线。LabVIEW 将为最匹配的接线端连线，对不匹配的接线端不予连线。

使用定位工具来移动对象时，按空格键则切换到自动连线模式。

选择连线

使用定位工具单击、双击或连续三次点击连线可以选择相应的连线。单击连线选中的是连线的一个直线段。双击连线选中的是连线的一个连线分支。连续三次点击连线选中的是整条连线。

纠正断线

断开的连线显示为中间有红色 x 的黑色虚线。出现断线的原因有很多，如试图连接数据类型不兼容的两个对象时就会产生断线。将连线工具移到断线上将显示一个描述产生断线原因的提示框。此时**即时帮助**窗口中也会出现同样的信息。右键单击该连线，从快捷菜单中选择**错误列表**可打开**错误列表**窗口。如需显示关于连线断开原因的更多信息，请单击**帮助**按钮。

用定位工具连续三次点击连线并按 <Delete> 键可以删除断线。还可右键单击连线，从快捷菜单中选择**删除连线分支**、**创建连线分支**、**删除松终端**、**整理连线**、**转换为输入控件**、**转换为显示控件**、**在源处启用索引**和**在源处禁用索引**等选项。这些选项因断线原因而异。

选择**编辑 » 删除断线**或按 <Ctrl-B> 键，可以清除所有断线。(Mac OS) 按 <Command-B> 键。(Linux) 按 <Meta-B> 键。

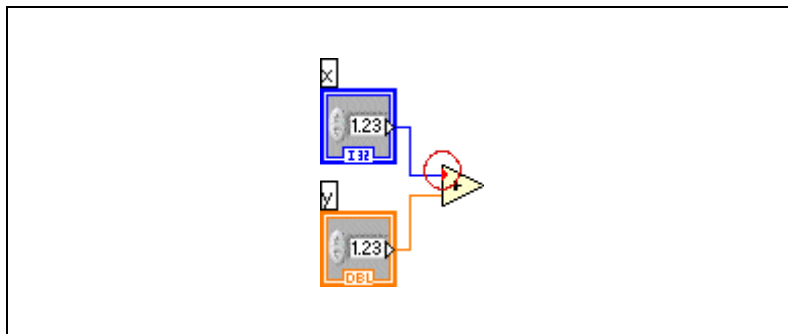


注意

清除所有断线时应谨慎。有时程序框图连线尚未全部完成时也会出现断线。

强制转换点

将两个不同的数值数据类型连接在一起时，程序框图节点上会出现强制转换点以示警告。强制转换点表示 LabVIEW 已经将传递给节点的数值转换成了不同的数据类型。例如，加函数需要两个双精度浮点数输入。如需其中一个输入为整数，“加”函数上就会出现一个强制转换点。如下图所示。



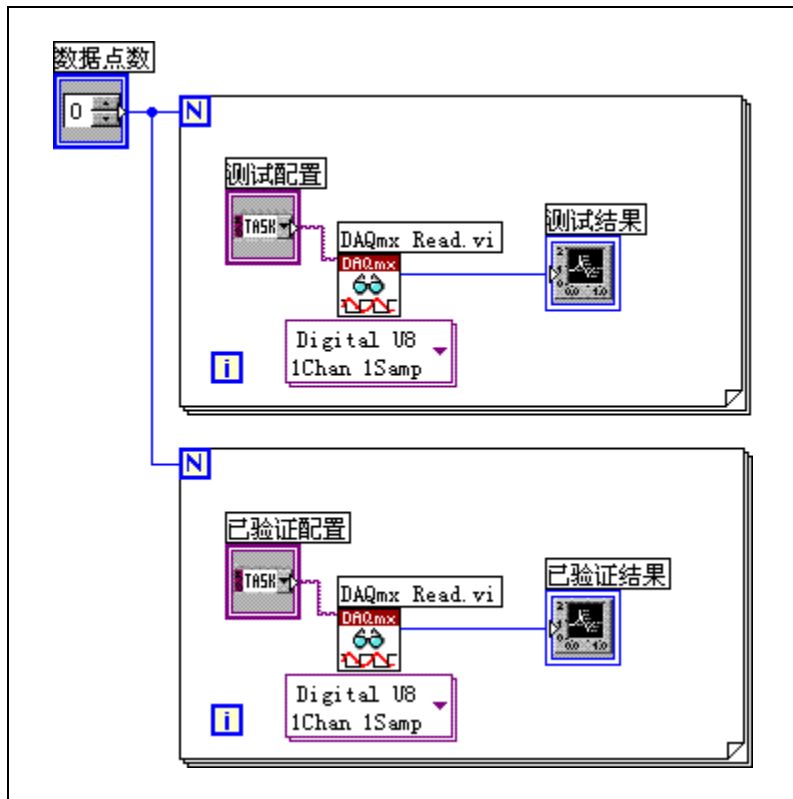
强制转换点表明了 VI 在哪里占用了更多的内存且运行时间增加。因此，创建 VI 时应尽量保持数据类型一致。

程序框图数据流

LabVIEW 按照数据流模式运行 VI。当具备了所有必需的输入时，程序框图节点将运行。节点在运行时产生输出数据并将该数据传送给数据流路径中的下一个节点。数据流经节点的动作决定了程序框图上 VI 和函数的执行顺序。

Visual Basic、C++、JAVA 以及绝大多数其它文本编程语言都遵循程序执行的控制流模式。在控制流中，程序元素的先后顺序决定了程序的执行顺序。

LabVIEW 是以数据流而不是命令的先后顺序决定程序框图元素的执行顺序。因此可创建具有并行操作的程序框图。例如，可同时运行两个 For 循环并在前面板上显示其结果。如以下程序框图所示。



数据依赖关系和人工数据依赖关系

控制流执行模式由指令驱动。数据流执行模式则由数据驱动, 又称为数据依赖。即从其它节点接收数据的节点总是在其它节点执行完毕后再执行。

没有连线的程序框图节点可以任意顺序执行。当自然的数据依赖关系不存在时，可用流经参数控制执行顺序。当数据流参数不可用时，可用顺序结构控制执行顺序。

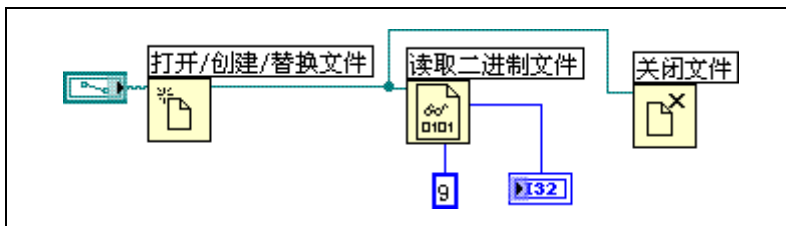
关于流经参数的更多信息见本章数据流参数一节。关于顺序结构的更多信息见第 8 章循环和结构中的顺序结构一节。

在人工数据依赖关系中接收节点并不真正使用接收到的数据。相反，接收节点根据数据是否到达来触发节点的执行。关于使用人工数据依赖关系的范例见 `labview\examples\general\structs.llb` 中的 **Timing Template (data dep)** VI。

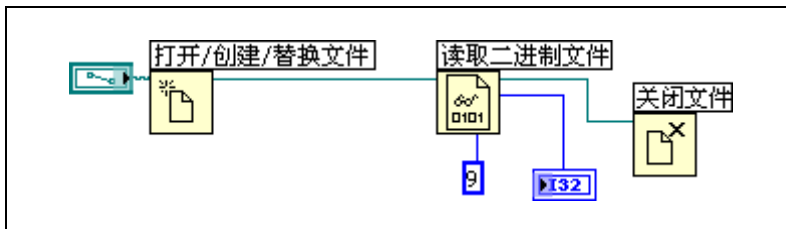
数据依赖关系不存在

数据依赖关系不存在时，不要想当然地认为程序的执行顺序是从左到右，自顶向下的。应确保数据流的连线，从而对事件顺序进行明确定义。

在下面的程序框图中，读取二进制文件函数和关闭文件函数之间不存在数据依赖关系，因为二者没有相连。该范例将由于不能确定哪个函数先执行而无法按所期望的顺序执行。如“关闭文件”函数先运行，“读取二进制文件”函数将不执行。



在下面的程序框图中，“读取二进制文件”函数的输出连接到“关闭文件”函数，二者建立了数据依赖关系。“关闭文件”函数只有在接收到“读取二进制文件”函数的输出后才能执行。



数据流参数

数据流参数通常为引用句柄或错误簇，它返回的是与相应的输入参数相同的值。当自然的数据依赖关系不存在时，可使用数据流参数来控制执行顺序。把要执行的第一个节点的数据流输出连接到要执行的下一个节点的相应输入，便创建了人工数据依赖关系。如没有这些数据流参数，则必须使用顺序结构来确保数据操作按期望的顺序执行。

关于错误 I/O 的更多信息，见第 6 章 *运行和调试 VI* 中的 *错误处理* 一节。关于顺序结构的更多信息，见第 8 章 *循环和结构* 中的 *顺序结构* 一节。

数据流和内存管理

相较于控制流执行模式，数据流执行模式使内存管理更为简单。在 LabVIEW 中，无需为变量分配内存或为变量赋值。只需创建带有连线的程序框图以表示数据的传输。

生成数据的 VI 和函数会自动为该数据分配内存。当该 VI 或函数不再使用该数据时，LabVIEW 将释放相关内存。向数组或字符串添加新数据时，LabVIEW 将分配足够的内存来管理这些新数据。

设计程序框图

设计程序框图时应遵循以下原则：

- 使用从左到右，自上而下的布局。尽管程序框图中各个元素的位置并不决定执行顺序，但应避免从右向左的连线方式，以使程序框图显得有结构，有条理，且易于理解。只有连线和结构才能决定执行顺序。
- 不要创建占用多于一个或两个屏幕的程序框图。太过庞大或复杂的程序框图将为理解和调试带来困难。
- 观察程序框图中的某些组件可否在其它 VI 中重复使用，或程序框图中某一部分可否组合成一个逻辑组件。如符合条件，将该程序框图分成几个执行特定任务的子 VI。使用子 VI 有利于对程序框图的修改进行管理和程序框图的快速调试。

关于子 VI 的更多信息见第 7 章 *创建 VI 和子 VI* 中的 *创建子 VI* 一节。

- 使用错误处理 VI、函数和参数在程序框图中管理错误。

关于处理错误的更多信息见第 6 章 *运行和调试 VI* 中的 *错误处理* 一节。

- 避免在结构边框下或重叠的对象之间进行连线，因为 LabVIEW 可能会隐藏这些连线的部分线段。
- 不要将对象放置在连线上方。将接线端或图标放置在连线上方易引起存在连接的错觉，而实际上连接并不存在。
- 自由标签可对程序框图上的代码进行注释。

关于使用自由标签的更多信息见第 4 章 *创建前面板* 中的 *添加标签* 一节。

运行和调试 VI

要使 VI 运行，须为 VI 的所有子 VI、函数和结构的接线端连接正确的数据类型。有时 VI 并不按预期的方式产生数据或运行。LabVIEW 可找出在程序框图的组织或流程序框图的数据中存在的问题。

运行 VI

运行 VI 将执行为该 VI 所设计的操作。工具栏上的**运行**按钮为白色实心箭头时表示 VI 可以运行。如下图所示。



白色实心箭头也表示为该 VI 创建连线板后可将其作为子 VI 使用。

关于创建连线板的更多信息见第 7 章 *创建 VI 和子 VI* 中的 *设置连线板* 一节。

单击**运行**或**连续运行**按钮或程序框图工具栏上的单步执行按钮，VI 便开始运行。VI 运行时，**运行**按钮变为黑色箭头，表明该 VI 正在运行。如下图所示。



VI 在运行时无法对其进行编辑。

单击**运行**按钮，VI 只运行一次，并在完成其数据流后停止。单击**连续运行**按钮，VI 将连续运行直到手动停止 VI 的运行为止。如下图所示。



单击单步执行按钮，VI 将以步进方式运行。

关于用单步执行按钮调试 VI 的更多信息见本章 *单步执行* 一节。

纠正断开的 VI

如一个 VI 无法执行，则表示该 VI 是断开的或不可执行的。创建或编辑 VI 时，如 VI 存在错误，**运行**按钮显示为断开。如下图所示。



如已完成程序框图的连线而该按钮仍显示为断开，则表示 VI 是断开的且不能运行。

查找 VI 断开的原因

警告并不妨碍 VI 运行。警告仅帮助用户避免 VI 中的潜在问题。而错误会使 VI 断开。VI 在运行前须排除任何错误。

单击断开的**运行**按钮或选择**查看 » 错误列表**可查找 VI 断开的原因。**错误列表**列出了所有的错误。**错误项**列出了内存中所有含有错误的项的名称，如 VI 和项目库。如两个或多个项具有相同的名称，则错误项部分会显示每一项的特定应用程序实例。**错误和警告**列出了在**错误项**中选中的 VI 错误和警告信息。**详细信息**描述了错误信息，有时还会建议如何纠正错误。单击**帮助**按钮，可显示 *LabVIEW 帮助*中对错误的详细描述和纠正错误步骤的相关主题。

单击**显示错误**按钮或双击错误描述，可高亮显示程序框图或前面板中包含错误的区域。

如 VI 中含有警告且**错误列表**窗口中的**显示警告**复选框被选中，工具栏将包含**警告**按钮。如下图所示。



VI 断开的常见原因

下表列出了编辑 VI 时导致 VI 断开的常见原因：

- 数据类型不匹配或存在未连接的接线端，会导致程序框图含有断线。
关于纠正断线的信息见第 5 章 *创建程序框图*中的 *纠正断线*一节。
- 必需连接的程序框图接线端没有连线。
关于设置必需的输入端和输出端的信息见第 5 章 *创建程序框图*中的 *使用连线连接程序框图各对象*一节。
- 子 VI 处于断开状态或在程序框图上放置子 VI 图标后编辑了该子 VI 的连线板。
关于子 VI 的信息见第 7 章 *创建 VI 和子 VI*中的 *创建子 VI*一节。

调试技术

如在 VI 未断开状态下得到了非预期数据，可利用调试技术发现和纠正 VI 或程序框图数据流的问题。

高亮显示执行过程

单击**高亮显示执行过程**按钮可查看程序框图的动态执行过程。如下图所示。



高亮显示执行过程通过沿连线移动的圆点显示数据在程序框图上从一个节点移动到另一个节点的过程。使用高亮显示执行的同时，结合单步执行，可查看 VI 中的数据从一个节点移动到另一个节点的全过程。



注

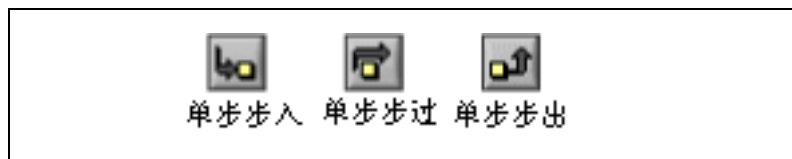
高亮显示执行过程会导致 VI 的运行速度大幅降低。

如**错误输出**簇报告错误，则在**错误输出**输出端旁将出现错误值，且错误值外围有一个红色边框。如没有错误发生，则**错误输出**输出端旁将出现**确定**按钮，其边框为绿色。

关于错误簇的更多信息见本章**错误簇**一节。

单步执行

单步执行 VI 可查看 VI 运行时程序框图上 VI 的每个执行步骤。单步执行按钮仅在单步执行模式下影响 VI 或子 VI 的运行。如下图所示。



单击程序框图工具栏上的**单步步过**或**单步步入**按钮可进入单步执行模式。将鼠标移动到**单步步过**、**单步步入**或**单步步出**按钮上可看到一个提示框，该提示框描述了单击该按钮后的下一步执行情况。可单步执行子 VI，也可正常运行子 VI。

如单步执行 VI 同时高亮显示执行过程，则执行符号将出现在当前运行的子 VI 的图标上。如下图所示。



探针工具

通用探针可查看流经连线的数据。右键单击连线，从快捷菜单中选择**自定义探针**可使用通用探针。

断点

断点工具可在程序框图上的 VI、节点或连线上放置一个断点，程序运行到该处时暂停执行。如下图所示。



在连线上设置断点后，数据流经该连线后程序将暂停执行。在程序框图上放置一个断点，使程序框图在所有节点执行后暂停执行。

VI 暂停于某个断点时，程序框图将出现在最前方，同时一个选取框将高亮显示含有断点的节点或连线。光标移动到断点上时，“断点”工具光标的黑色区域变为白色。

程序执行到一个断点时，VI 将暂停执行，同时**暂停**按钮显示为红色。可采取以下措施：

- 用单步执行按钮单步执行程序。
- 在连线上添加探针查看中间数据。
- 改变前面板控件的值。
- 单击**暂停**按钮可继续运行到下一个断点处或直到 VI 运行结束。

LabVIEW 将断点与 VI 一起保存，但断点只在 VI 运行时有效。选择**操作 » 断点**然后单击**查找**按钮可查看所有断点。

可逐个或在 VI 层次结构中删除断点。

错误处理

无论 VI 有多完美，也很难预见到用户可能遇到的每一个问题。如没有一个检查错误的机制，则可确定的仅是 VI 存在错误。通过错误检查则可确认发生错误的原因和错误出现的位置。

在执行任何形式的输入 / 输出 (I/O) 操作时，都应考虑到发生错误的可能性。几乎所有的 I/O 函数都会返回错误信息。应在 VI 中包括错误检查，尤其对于文件、串口、仪器测量、数据采集和通讯等 I/O 操作更应如此，并提供一个恰当的错误处理机制。

默认状态下，LabVIEW 将通过挂起执行、高亮显示出现错误的子 VI 或函数并且显示错误对话框的方式来自动错误每个错误。

选择**文件»VI 属性**，并从**类别**下拉菜单中选择**执行**可禁用当前 VI 的自动处理错误功能。选择**工具»选项**并从**类别**列表中选择**程序框图**，可禁用任何新创建的空白 VI 的自动处理错误功能。如需禁用一个 VI 中的子 VI 或函数的自动处理错误功能，可将其**错误输出**参数与另一个子 VI 或函数的**错误输入**参数连接，或连接到一个**错误输出**显示控件。

另有几种处理错误的方法。例如，程序框图上有一个 I/O VI 超时，但并不希望整个应用程序都停止运行，同时也不希望错误对话框出现。也可能需要在一段时间内重新运行该 VI。在 LabVIEW 中，可在 VI 的程序框图上进行这些错误处理的设置。

位于**对话框与用户界面**选板上的 LabVIEW 错误处理 VI 和函数，以及大多数 VI 和函数的**错误输入**和**错误输出**参数可管理错误。如 LabVIEW 遇到了错误，可在不同类型的对话框中显示错误信息。将错误处理和调试工具结合使用可发现并处理错误。

VI 和函数通过数值错误代码或错误簇返回错误。通常，函数以数值错误代码返回错误，而 VI 以错误簇，即错误输入和错误输出来返回错误。

LabVIEW 中的错误处理遵循数据流模式。错误信息就像数据值一样流经 VI。错误信息从 VI 的起点一直连接到终点。错误处理 VI 与一个 VI 连接后可确定该 VI 的运行是否未出差错。**错误输入**和**错误输出**簇可在创建或使用时的每一个 VI 中传递错误信息。错误簇为流经参数。

关于流经参数的详细信息见第 5 章 *创建程序框图的数据流参数*一节。

VI 运行时，LabVIEW 会在每个执行节点检测错误。如 LabVIEW 没有发现任何错误，则该节点将正常执行。如 LabVIEW 检测到错误，则该节点会将错误传递到下一个节点且不执行那一部分代码。后面的节点也照此处理，直到最后一个节点。执行流结束时，LabVIEW 报告错误。

错误簇

错误输入和**错误输出**簇包括以下信息：

- **状态**是一个布尔值，错误产生时报告 TRUE。
- **错误代码**是一个 32 位有符号整数，通过数值表示错误。一个非零错误代码和 FALSE **状态**相结合可表示警告但不是错误。
- **错误源**是用于识别错误发生位置的字符串。

一些支持布尔数据的 VI、函数和结构也可识别错误簇。例如，将一个错误簇连接到“选择”、“退出 LabVIEW”或“停止”函数的布尔输入端。如发生错误，错误簇将把 TRUE 值传递给该函数。

关于簇的更多信息见第 9 章 *用字符串、数组和簇将数据分组*中的簇一节。

使用 While 循环处理错误

可将错误簇连接到 While 循环的条件接线端以停止 While 循环的运行。错误簇连接到条件接线端上时，只有错误簇**状态**参数的 TRUE 或 FALSE 值会传递到接线端。当错误发生时，While 循环停止执行。

将一个错误簇连接到条件接线端上时，快捷菜单项**真 (T) 时停止**和**真 (T) 时继续**将变为**错误时停止**和**错误时继续**。

关于使用 While 循环的更多信息见第 8 章 *循环和结构* 中的 *While 循环* 一节。

用条件结构进行错误处理

将错误簇连接到条件结构的条件选择器接线端时，条件选择器标签将显示两个选项：错误和无错误。同时条件结构的边框的颜色将改变：错误时为红色，无错误时为绿色。发生错误时，条件结构将执行错误子程序框图。

关于使用条件结构的更多信息见第 8 章 *循环和结构* 中的 *条件结构* 一节。

子 VI 和错误处理模板 VI 相结合可创建带有错误处理条件结构的 VI。

关于模板 VI 的更多信息见第 1 章 *LabVIEW 简介* 中的 *LabVIEW VI 模板* 一节。

创建 VI 和子 VI

VI 可作为用户界面，也可以是程序中一项常用操作。了解如何创建前面板和程序框图后，即可开始创建并自定义 VI 和子 VI。

查找范例

创建新 VI 之前，可考虑选择**帮助 » 查找范例**，打开 NI 范例搜索器，搜索满足要求的 VI 范例。如未找到合适的 VI 范例，可在**新建**对话框中打开 VI 模板，模板中包含**函数**选板中一些内置 VI 和函数。

关于 VI 范例和 VI 模板的详细信息见第 1 章 *LabVIEW 简介* 中的 *LabVIEW VI 模板、VI 范例和工具* 一节。

使用内置 VI 和函数

LabVIEW 包含多个用于创建特定应用程序的内置 VI 和函数，例如，数据采集 VI 和函数、访问其它 VI 的 VI、以及与其它应用程序通信的 VI。将这些 VI 作为子 VI 在应用程序中使用，可缩短开发时间。在创建新 VI 之前，可考虑在**函数**选板中查找类似的 VI 和函数，在现有 VI 的基础上创建 VI。

创建子 VI

可将新创建的 VI 用于另一个 VI。一个 VI 被其它 VI 在程序框图中调用，则称该 VI 为子 VI。子 VI 可重复调用。要创建一个子 VI，需先为子 VI 创建连线板和图标。

子 VI 的节点类似于文本编程语言中的子程序调用。一个程序中的子程序调用指令并不是子程序本身。同理，节点也不是子 VI。一个程序框图含有相同子 VI 节点的数目与该子 VI 被调用的次数相等。

子 VI 的控件和函数从调用该 VI 的程序框图中接收数据，并将数据返回至该程序框图。如需创建一个被调用的子 VI，单击**函数**选板上的**选择 VI**图标或文本，找到目标 VI 并双击，即可将该 VI 放置在程序框图上。

用操作或定位工具双击程序框图上的子 VI，即可编辑该子 VI。保存子 VI 时，子 VI 的改动将影响到所有调用该子 VI 的程序，而不只是当前程序。

创建图标

每个 VI 都在前面板和程序框图窗口的右上角有一个图标，如图下所示。



图标是 VI 的图形化表示，可包含文字、图形或图文组合。如将 VI 当作子 VI 调用，程序框图上将显示该子 VI 的图标。

默认图标中有一个数字，表明 LabVIEW 启动后打开新 VI 的个数。右键单击前面板或程序框图右上角的图标并从快捷菜单中选择**编辑图标**，或双击前面板右上角的图标可将默认图标替换为创建的自定义图标。

也可从文件系统中拖动一个图形放置在前面板或程序框图的右上角。LabVIEW 会将该图形转换为 32 × 32 像素的图标。

关于 VI 图标的标准图形的详细信息，登陆 National Instruments 网站 ni.com/info 并输入信息代码 `expnr7` 进行查询。

设置连线板

要将一个 VI 当作子 VI 使用，需创建连线板，如下图所示。



连线板集合了 VI 各个接线端，与 VI 中的控件相互呼应，类似文本编程语言中函数调用的参数列表。连线板标明了可与该 VI 连接的输入和输出端，以便将该 VI 作为子 VI 调用。连线板在其输入端接收数据，然后通过前面板控件将数据传输至程序框图的代码中，从前面板的显示控件中接收运算结果并传递至其输出端。

将前面板上的输入控件和显示控件分配至连线板的每个接线端，从而定义连接。如需定义连线板，右键单击前面板右上角的图标，并从快捷菜单中选择**显示连线板**。图标的位置上将出现连线板。第一次打开连线板时，可看到连线板的模式。右键单击连线板，从快捷菜单中选择**模式**可为 VI 选择不同的接线端模式。

连线板上的每个单元格代表一个接线端，使用各个单元格分配输入和输出。默认的连线板模式为 4 × 2 × 2 × 4。使用默认模式可保留多余的接线端，当需要为 VI 添加新的输入或输出端时再进行连接。

连线板中最多可设置 28 个接线端。如前面板上的控件不止 28 个，可将其其中的一些对象组合为一个簇，然后将该簇分配至连线板上的一个接线端。

关于使用簇组合数据的详细信息见第 9 章 *用字符串、数组和簇将数据分组* 中的 *簇* 一节。

右键单击连线板并从快捷菜单中选择 **模式** 可为 VI 选择不同的接线端模式。例如，可选择带有附加接线端的连线板模式。空置出附加接线端，需要时再进行连接。接线板应有一定的灵活性，使改变对 VI 层次结构的影响减小到最小。

选中部分程序框图创建子 VI

用定位工具选择需重复使用的部分程序框图，选择 **编辑 » 创建子 VI**，可将部分 VI 转换成子 VI。选中的程序框图将被替换为新子 VI 的图标。LabVIEW 可为新的子 VI 创建输入控件和显示控件，并根据所选控件的数目自动配置连线板，将子 VI 与现有的连线对接。

通过选中部分程序框图创建子 VI 方便快捷，但需要仔细规划，创建符合逻辑的 VI 层次结构。考虑需选中的对象，从而避免改变最后生成 VI 的功能。

设计子 VI 的前面板

根据在连线板中的位置，将控件一一放置在前面板上。输入控件放在前面板左边，显示控件放在前面板右边。**错误输入**簇通常在前面板左下角，**错误输出**簇在前面板右下角。

关于连线板的设定的详细信息见本章 *设置连线板* 一节。

查看 VI 的层次结构

VI 层次结构 窗口以图形化的方式显示所有打开的 LabVIEW 项目和终端，以及内存中所有 VI 的调用结构，包括自定义类型和全局变量。选择 **查看 » VI 层次结构**，打开 **VI 层次结构** 窗口。该窗口用于查看内存中该 VI 的子 VI 和其它节点以及搜索 VI 层次结构。

关于项目的详细信息见第 3 章 *LabVIEW 编程环境中的项目浏览器窗口* 一节。

VI 层次结构 窗口显示顶层图标，代表 LabVIEW 主应用程序实例，其下显示的是所有未包括在该项目或项目应用程序实例中的展开的 VI。如在 LabVIEW 中添加一个项目，**VI 层次结构** 窗口中将显示代表该项目的顶层 VI 图标。所有添加的对象均位于项目之下。

将光标移至 **VI 层次结构** 窗口的对象上，下方将显示该 VI 的名称。使用定位工具拖动 **VI 层次结构** 窗口中的 VI，可将其放到另一个 VI 的程序框图中作为子 VI 使用。也可选择并复制一个或多个节点，把它们粘贴到其它程序框图上。在 **VI 层次结构** 窗口中双击某个 VI 可显示该 VI 的前面板。

如一个 VI 含有子 VI，则该 VI 的底边上有黑色箭头。单击该箭头按钮可显示或隐藏子 VI。如所有子 VI 均为隐藏状态则箭头为红色。如所有子 VI 都已显示则箭头为黑色。

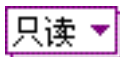
多态 VI

多态 VI 可在一个输入或输出端接收不同的数据类型。多态 VI 是具有相同模式连线板的子 VI 的集合。该集合中的每个 VI 均为多态 VI 的一个程序实例。

例如，读取键值 VI 就是一个多态 VI。其**默认值**接线端可以接收的数据类型有布尔、双精度浮点数、32 位有符号整型、路径、字符串或 32 位无符号整型。

对于绝大多数多态 VI，连接到 VI 输入端的数据类型决定使用何种实例。如果多态 VI 中没有任何实例与其连接的数据类型兼容，则会出现断线。如连到多态 VI 的数据类型不能决定使用哪个实例，则必须手动选择实例。如果在多态 VI 中手动选择实例，该 VI 将不再是多态 VI，因为它将只接收和返回选中的数据类型。

如需手动选择实例，右键单击多态 VI，在快捷菜单中选择**选择类型**，然后选择所需实例。同时也可使用操作工具单击多态 VI 选择器，然后从快捷菜单中选择实例，如下图所示。



右键单击程序框图上的多态 VI 并从快捷菜单中选择**显示项 » 多态 VI 选择器**，显示选择器。要使多态 VI 重新接收所有可处理的数据类型，用右键单击多态 VI 并从快捷菜单中选择**选择类型 » 自动**或使用操作工具单击多态 VI 选择器并从快捷菜单中选择**自动**。

如需对不同的数据类型执行同样的操作，这时可考虑创建多态 VI。



注 只有在 LabVIEW 专业版开发系统中才可以创建和编辑多态 VI。

例如，如需对单精度浮点数、数值数组或波形执行同样的数学运算，需要创建三个独立的 VI—数值运算、数组运算和波形运算。当需要执行该操作时，根据不同的输入数据类型选择其中的一个 VI 放置在程序框图上。

创建和使用一个多态 VI 代替手动选择 VI。

保存 VI

选择**文件»保存**，保存 VI。为了便于识别，保存 VI 时应使用描述性的名称。可将 VI 保存为 LabVIEW 的前期版本，便于日后升级 LabVIEW，以及必要时在两个不同的 LabVIEW 版本中维护这些 VI。

VI 命名

保存 VI 时，应使用描述性的名称。描述性的名称便于识别 VI 并了解该如何使用 VI，例如，Temperature Monitor.vi 和 Serial Write & Read.vi。含义模糊的文件名会造成文件混淆。保存了多个 VI 后，更是难以识别。例如，VI#1.vi。

命名时同时要考虑用户是否可能在其它平台上使用该 VI，因此不要使用一些平台上具有特殊用途的符号，例如，\:/?*<> 和 #。



注

如一台计算机上有多个同名 VI，将这些 VI 分别放在相应目录或 LLB 下，避免 LabVIEW 在打开顶层 VI 时错误调用子 VI。

保存为前期版本

将 VI 保存为 LabVIEW 的前期版本，可方便日后升级 LabVIEW，以及必要时在两个不同的 LabVIEW 版本中维护这些 VI。选择**文件»保存为前期版本**，将文件保存为 LabVIEW 前期版本。

将 VI 保存为前期版本时，LabVIEW 不是仅转换该 VI，而是转换层次结构中的所有 VI（但不包括 labview\vi.lib 目录下的 VI）。

VI 可能经常使用 LabVIEW 前期版本中没有的功能。在这种情况下，LabVIEW 将尽量保存该 VI 的现有功能，并报告说明哪些功能不可转换。该报告会立即出现在**警告**对话框中。单击**确定**按钮，确认已收到警告，并关闭对话框。单击**保存至文件**按钮可将这些警告保存为文本文件以备日后查看。

自定义 VI

根据应用程序的要求可对 VI 和子 VI 进行配置。例如，如需将一个 VI 作为子 VI 使用，该子 VI 要求用户输入，可将该子 VI 设置为每次调用时都显示前面板。

选择**文件»VI 属性**，配置 VI 的外观和动作。**VI 属性**对话框顶部的**类别**下拉菜单中列出各种 VI 选项设置。

VI 属性对话框包括下列选项：

- **常规**—显示 VI 保存的当前路径、修订号、修订记录，以及自该 VI 上次保存以来所作的任何修改信息，还可在该页上编辑 VI 图标。
- **说明信息**—该选项用于添加说明，并链接至相关帮助文件主题。
关于说明信息选项的详细信息见第 12 章 *编制 VI 说明信息和打印 VI 中的编制 VI 说明信息* 一节。
- **安全**—用于锁定 VI 或通过密码保护 VI。
- **窗口外观**—用于自定义 VI 的窗口外观，如窗口标题和样式。
- **窗口大小**—用于设置窗口的大小。
- **执行**—用于定义如何运行 VI。例如，将 VI 置为打开时立即运行，或配置为作为子 VI 被调用时暂停运行。
- **编辑器选项**—该选项用于设置当前 VI 对齐网格的大小、改变控件样式。这里的控件是指右键单击接线端并从快捷菜单中选择 **创建 » 输入控件** 或 **创建 » 显示控件** 所显示的控件。

关于对齐网格的详细信息见第 4 章 *创建前面板中的对齐和分布对象* 一节。

循环和结构

结构是传统文本编程语言中的循环和条件语句的图形化表示。使用程序框图中的结构可对代码块进行重复操作，根据条件或特定顺序执行代码。

与其它节点类似，结构也具有可与其它程序框图节点进行连线的接线端。输入数据存在时结构会自动执行，执行结束后将数据提供给输出线路。

每种结构都含有一个可调整大小的清晰边框，用于包围根据结构规则执行的程序框图部分。结构边框中的程序框图部分被称为子程序框图。从结构外接收数据和将数据输出结构的接线端称为隧道。隧道是结构边框上的连接点。

结构选板中的以下结构可用于控制程序框图的执行方式：

- **For 循环**—按设定的次数执行子程序框图。
- **While 循环**—执行子程序框图直至满足某个条件。
- **条件结构**—包含多个子程序框图，根据传递至该结构的输入值，每次只执行其中一个子程序框图。
- **顺序结构**—包含一个或多个按顺序执行的子程序框图。
- **事件结构**—包括一个或多个子程序框图，在用户交互产生某个事件时执行。
- **定时结构**—执行一个或多个包括限时和延时的子程序框图。
- **条件禁用结构**—包含一个或多个子程序框图，每个子程序框图均在运行时编译和执行。
- **程序框图禁用结构**—包含一个或多个子程序框图，运行时只编译和执行其中一个子程序框图。

右键单击结构边框可显示快捷菜单。

For 循环和 While 循环结构

For 循环和 While 循环可用来控制重复性操作。

For 循环

如下图所示，For 循环将按设定的次数执行子程序框图。



总数接线端（输入端）的值表示重复执行该子程序框图的次数，如下图所示。



将循环外部的数值连接到总数接线端的左边或顶部，可手动设定循环次数，或者使用自动索引自动设定循环总数。

关于手动设定循环总数的信息详见本章的 *使用自动索引设置 For 循环总数值* 一节。

如下图所示，计数接线端（输出端）表示已完成循环的次数。



计数器总是从零开始计数。第一次循环时，计数接线端返回 0。

总数和计数接线端都是 32 位有符号整数。如将一个浮点数连接到总数接线端，LabVIEW 将对其进行取整，并将其强制转换到 32 位有符号整数的范围内。如果将 0 或负数连接到总数接线端，该循环无法执行并在输出中显示该数据类型的默认值。

在 For 循环中添加移位寄存器可将当前循环中的数据传递至下一次循环。

关于在循环中添加移位寄存器的详细信息见本章的 *移位寄存器* 一节。

While 循环

如下图所示，类似于文本编程语言中的 Do 循环或 Repeat-Until 循环，While 循环执行子程序框图直到满足某个条件。



While 循环执行子程序框图直到条件接线端（输入端）接收到某一特定的布尔值。如下图所示，条件接线端的默认动作和外观为**真 (T) 时停止**。

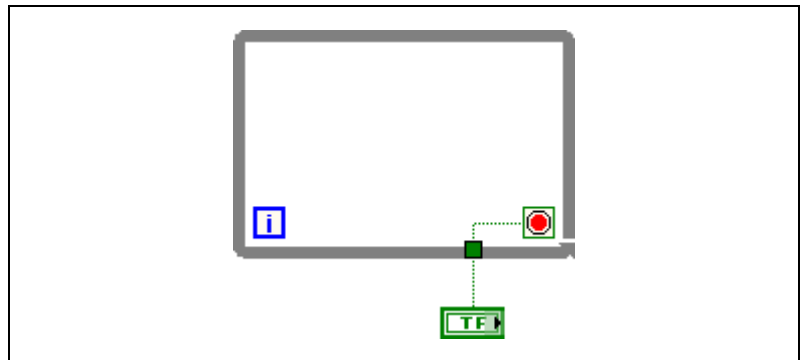


当条件接线端为**真 (T) 时停止**时，While 循环将执行其子程序框图直到条件接线端接收到一个 TRUE 值。右键单击该接线端或 While 循环的边框，并选择如下图所示的**真 (T) 时继续**，可改变条件接线端的动作和外观。



当条件接线端为**真 (T) 时继续**时，While 循环将执行其子程序框图直到条件接线端接收到一个 FALSE 值。通过使用操作工具单击条件接线端也可改变条件。

如下图所示，如果将布尔控件的接线端放置在 While 循环的外部并且该控件被设置为 FALSE，当循环执行时，如果条件接线端为**真 (T) 时停止**，则会导致无限循环。如果将循环外部的控件设置为 TRUE，且条件接线端为**真 (T) 时继续**，也会导致无限循环。



由于输入控件的值只在循环开始前被读取一次，因此改变控件的值并不能停止无限循环。要停止一个无限循环，必须单击工具栏上的**中止执行**按钮中止整个 VI。

使用 While 循环的条件接线端也可进行基本的错误处理。将错误簇连接到条件接线端时，仅有错误簇中**状态**参数的 TRUE 或 FALSE 值被传递到该接线端，并且**真 (T) 时停止**和**真 (T) 时继续**快捷菜单选项也相应地分别变为**错误时停止**和**错误时继续**。

关于错误簇和错误处理的详细信息见第 6 章 *运行和调试 VI 的错误处理* 一节。

如下图所示，计数接线端（输出端）表示已完成的循环次数。



计数器总是从零开始计数。第一次循环时，计数接线端返回 0。

在 While 循环中添加移位寄存器可将当前循环中的数据传递到下一次循环。

关于在循环中添加移位寄存器的详细信息见本章的 *移位寄存器* 一节。

控制定时时间

您可能需要控制进程的执行速度，如将数据绘制到图表上的速度。此时，在循环中使用等待 (Wait) 函数可以使循环在重新执行之前等待一定的时间，时间的单位为毫秒。

自动索引循环

如果将一个数组连接到 For 循环或 While 循环的输入隧道，启用自动索引可读取和处理数组中的各个元素。

关于数组的详细信息见第 9 章 *用字符串、数组和簇将数据分组的数组* 一节。

将数组连接到循环边框的输入隧道并且启用输入隧道的自动索引时，从第一个元素开始每次均有一个数组元素进入循环。当禁用自动索引时，整个数组将一次性全部传递到循环中。启用数组输出隧道的自动索引功能时，该输出数组从每次循环中接收一个新元素。因此，自动索引的输出数组的大小与重复的次数相等。例如，如循环执行了 10 次，那么输出数组就含有 10 个元素。如果禁用输出隧道上的自动索引，仅有最后一次循环的元素被传递到程序框图上的下一个节点。

右键单击循环边框上的隧道，并从快捷菜单中选择**启用索引**或**禁用索引**可以启用或禁用自动索引。默认情况下，While 循环禁用自动索引。

循环边框上的方括号表示已启用自动索引。输出隧道和下一个节点间连线的粗细也表示循环是否正在使用自动索引。使用自动索引时，连线较粗，因为此时连线上包含一个数组而不是一个标量。

循环在一维数组中提取标量建立索引，在二维数组中提取一维数组建立索引，依此类推。输出隧道的情况正好相反。标量元素按顺序累积形成一维数组，一维数组累积形成二维数组，以此类推。

使用自动索引设置 For 循环总数值

如果将连接到 For 循环输入接线端的数组启用自动索引，LabVIEW 会将总数接线端设置成与数组大小一致，因此用户无需为总数接线端连接数值。由于 For 循环每次可处理数组中的一个元素，因此默认情况下，LabVIEW 对连接到 For 循环的每个数组均启用自动索引。如不需要一次处理数组中的一个元素，可以禁用自动索引。

如果多个隧道启用自动索引，或对计数接线端进行连线，则计数值将取其中的较小值。例如，如果两个启用自动索引的数组进入循环，分别含有 10 个和 20 个元素，同时将值 15 连接到总数接线端，这时该循环执行 10 次，并且只对第二个数组中的前 10 个元素建立索引。再如，在一个图形上绘制两个数据源，并只需绘制前 100 个元素，这时可将值 100 连接到总数接线端。如果其中一个数据源只含有 50 个元素，那么循环将执行 50 次，并且只对前 50 个元素建立索引。数组大小函数可用来确定数组的大小。

While 循环的自动索引

如果为一个进入 While 循环的数组启用自动索引，则 While 循环将以与 For 循环同样的方式对该数组建立索引。但是，While 循环只有在满足特定条件时才会停止执行，因此 While 循环的执行次数不受该数组大小的限制。当 While 循环索引超过输入数组的大小时，LabVIEW 会将该数组元素类型的默认值输入循环。通过使用“数组大小”函数可以防止将数组默认值传递到 While 循环中。“数组大小”函数显示数组中元素的个数。将 While 循环设置为当循环次数与数组大小相同时停止执行。



注意

由于不能提前确定输出数组的大小，因此启用 For 循环的自动索引比启用 While 循环的自动索引更有效。循环次数过多可能会引起系统内存溢出。

使用循环创建数组

不仅可以使循环读取和处理数组中的元素，还可以通过 For 循环和 While 循环创建数组。将循环中的 VI 或函数的输出连接到循环边框上。在 While 循环中，右键单击隧道并从快捷菜单中选择**启用索引**。使用 For 循环时，默认情况下已启用索引。隧道输出的是一个数组，数组中的每个元素都是每次循环结束后 VI 或函数返回的值。

关于数组的详细信息见第 9 章*用字符串、数组和簇将数据分组的数组*一节。

参考以下创建数组的范例：`labview\examples\general\arrays.llb`。

循环中的移位寄存器和反馈节点

在 For 循环或 While 循环中，移位寄存器或反馈节点可将某次循环的值传递到下一次循环中。

移位寄存器

移位寄存器可用于将上一次循环的值传递至下一次循环。如下图所示，移位寄存器以一对接线端的形式出现，分别位于循环两侧的边框上，位置相对。

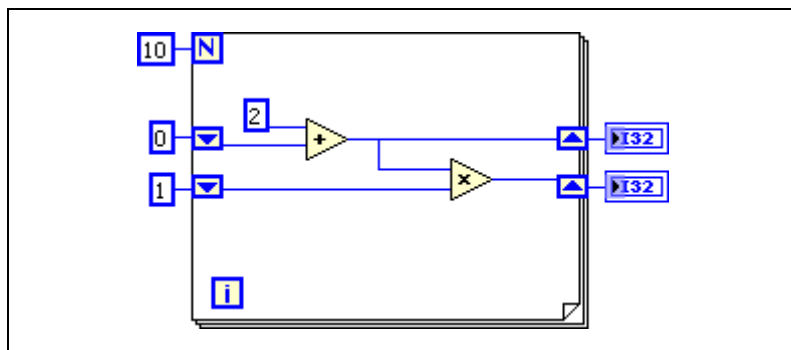


右侧接线端含有一个向上的箭头，用于存储每次循环结束时的数据。LabVIEW 可将连接到右侧寄存器的数据传递到下一次循环中。循环执行后，右侧接线端将返回移位寄存器保存的值。

右键单击循环的左侧或右侧边框，并从快捷菜单中选择**添加移位寄存器**可以创建一个移位寄存器。

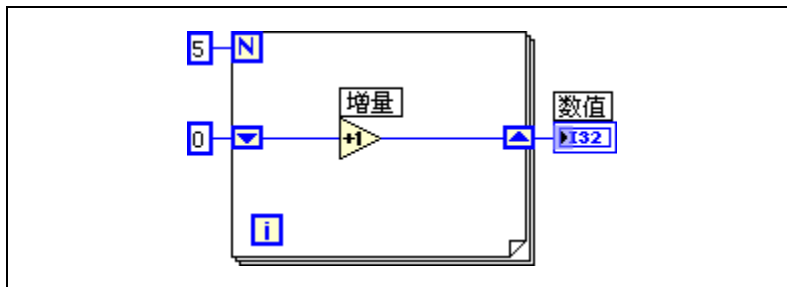
移位寄存器可以传递任何数据类型，并和与其连接的第一个对象的数据类型自动保持一致。连接到各个移位寄存器接线端的数据必须属于同一种数据类型。

循环中可添加多个移位寄存器。如下图所示，如循环中的多个操作都需使用之前循环的值，可以通过多个移位寄存器保存结构中不同操作的数据值。



初始化移位寄存器

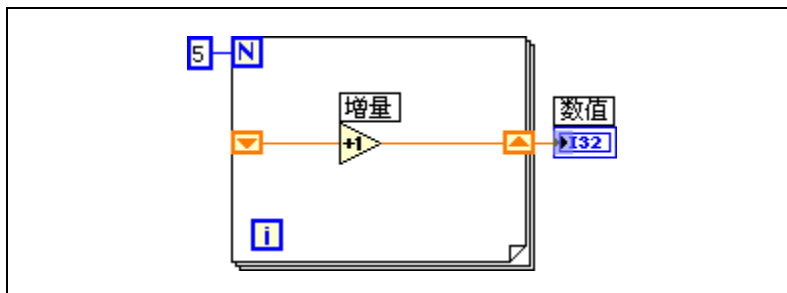
初始化移位寄存器，即重设 VI 运行时移位寄存器传递给第一次循环的值。如下图所示，将输入控件或常量连接到循环左侧的移位寄存器接线端，即可初始化移位寄存器。



上图中的 For 循环将执行 5 次，每次循环后，移位寄存器的值都增加 1。For 循环完成 5 次循环后，移位寄存器会将最终值 (5) 传递给显示控件并结束 VI 运行。每次执行该 VI，移位寄存器的初始值均为 0。

如未初始化移位寄存器，循环将使用最后一次执行时写入该寄存器的值，在循环未执行过的情况下使用该数据类型的默认值。

使用未初始化的移位寄存器还可以保留 VI 多次执行之间的状态信息。下图即是未初始化的移位寄存器。

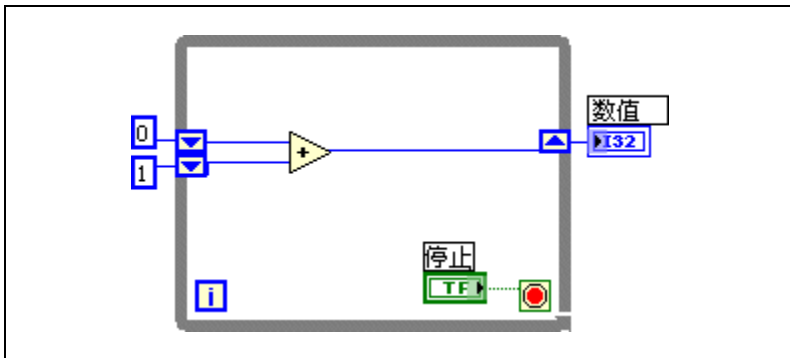


上图中的 For 循环将执行 5 次，每次循环后，移位寄存器的值都增加 1。第一次运行 VI 时，移位寄存器的初始值为 0（即 32 位整型数据的默认值）。For 循环完成 5 次循环后，移位寄存器会将最终值 (5) 传递给显示控件并结束 VI 运行。而第二次运行该 VI 时，移位寄存器的初始值是上一次循环所保存的最终值 5。For 循环执行 5 次后，移位寄存器会将最终值 (10) 传递给显示控件。如果再次执行该 VI，移位寄存器的初始值是 10，依此类推。关闭 VI 之前，未初始化的移位寄存器将保留上一次循环的值。

层叠移位寄存器

层叠移位寄存器可访问以前多次循环的数据。层叠移位寄存器可以保存以前多次循环的值，并将值传递到下一次循环中。如需创建层叠移位寄存器，右键单击左侧的接线端并从快捷菜单中选择**添加元素**。

如下图所示，层叠移位寄存器只位于循环左侧，因为右侧的接线端仅用于把当前循环的数据传递给下一次循环。



如在上图中给左侧接线端添加另一个元素，上两次循环的值将传递至下一次循环中，其中最近一次循环的值保存在上面的寄存器中，而上一次循环传递给寄存器的值则保存在下面的接线端中。

反馈节点

如下图所示，在 For 循环或 While 循环中，将一个或一组节点的输出连接到这个（些）节点的输入时会自动出现反馈节点。



也可在**函数**选板上选择反馈节点，然后将其放在 For 循环或 While 循环中。使用反馈节点可以避免在循环中出现太长太多连线。

右键单击反馈节点，在快捷菜单中勾选**初始化接线端**，在循环边框上添加初始化接线端，初始化循环。在**函数**选板中选择反馈节点或将已初始化的移位寄存器转换为反馈节点，循环会自动生成初始化接线端。VI 运行时，初始化反馈节点将重置传递给第一次循环的初始值。如未初始化反馈节点，那么该反馈节点将传递最后一次写入该节点的值。如循环从未执行，则传递其数据类型的默认值。如初始化接线端的输入端没有连接数值，则每次 VI 运行时，反馈节点的初始输入都将是上一次执行的最终值。

右键单击移位寄存器，从快捷菜单中选择**替换为反馈节点**，可将移位寄存器替换为反馈节点。右键单击反馈节点并从快捷菜单中选择**替换为移位寄存器**，可将反馈节点替换为移位寄存器。

循环结构的默认数据

移位寄存器未初始化时，While 循环将产生默认数据。

如果将 0 连接到 For 循环的总数接线端，或者将空数组作为输入连接到 For 循环并且启用自动索引，For 循环将产生默认数据。该循环将不会执行，任何禁用自动索引的输出隧道将返回该隧道数据类型的默认值。无论循环执行与否，都可使用移位寄存器在循环之间可以传输数据。

关于数据类型默认值的更多信息参见《LabVIEW 快速参考指南》。

条件、顺序和事件结构

条件结构、层叠式顺序结构、平铺式顺序结构以及事件结构都包含多个子程序框图。条件结构根据传递给该结构的输入值执行相应的子程序框图。层叠式顺序结构和平铺式顺序结构按特定顺序执行所有子程序框图。而事件结构则根据用户与 VI 的交互情况执行其中的某一个子程序框图。

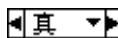
条件结构

如下图所示，条件结构包括两个或两个以上子程序框图（也称“条件分支”）。



每次只能显示一个子程序框图，并且每次只执行一个条件分支。输入值将决定执行的子程序框图。条件结构类似于文本编程语言中的 switch 语句或 if...then...else 语句。

如下图所示，条件结构顶部的条件选择器标签是由结构中各个条件分支对应的选择器值的名称以及两边的递减和递增箭头组成。



单击递减和递增箭头可以滚动浏览已有条件分支。也可以单击条件分支名称旁边的向下箭头，并在下拉菜单中选择一个条件分支。

将一个输入值或选择器连接到如下图所示的选择器接线端即可以选择需执行的条件分支。



选择器接线端可以连接的数据类型有整型、布尔值型、字符串型和枚举型。条件选择器接线端可置于条件结构左边框的任意位置。如果选择器接线端的数据类型是布尔值型，则该结构包括真和假分支。如果选择器接线端的数据类型为整型、字符串型或枚举型，该结构可以使用任意个分支。

指定条件结构的默认条件分支以处理超出范围的数值。否则应明确列出所有可能的输入值。例如，如果选择器的数据类型是整型，并且已指定 1、2 和 3 分支，则必须指定一个默认选框以便在输入数据为 4 或任何其它有效的整数时执行。

分支选择器值和数据类型

在条件选择器的标签中输入单个值或数值列表和范围。如使用列表，数值之间用逗号隔开。如使用数值范围，指定一个类似 10..20 的范围可用于表示 10 到 20 之间的所有数字（包括 10 和 20）。也可以使用开集范围。例如，..100 表示所有小于等于 100 的数，100.. 表示所有大于等于 100 的数。同时也可以将列表和范围结合起来使用，如 ..5, 6, 7..10, 12, 13, 14。如在同一个条件选择器标签中输入的数值范围有重叠，条件结构会以更紧凑的形式重新显示该标签。例如，上例将显示为 **..10, 12..14**。如使用字符串范围，范围 a..c 包括 a 和 b，但不包括 c。而 a..c,c 则同时包括结束值 c。

如果输入选择器的值与选择器接线端所连接的对象不是同一数据类型，则该值将变为红色，在结构执行之前必须删除或编辑该值，否则 VI 不能运行。同样由于浮点算术运算可能存在四舍五入误差，因此浮点数不能作为条件选择器值。如果将一个浮点数连接到条件分支，LabVIEW 将对其进行舍入到最近的偶数值。如果在条件选择器标签中输入浮点值，则该值将变成红色，在执行结构前必须对该数值进行删除或修改。

输入和输出隧道

可为条件结构创建多个输入输出隧道。所有输入都可供条件分支选用，但条件分支不需使用每个输入。但是，必须为每个条件分支定义各自的输出隧道。在某一个条件分支中创建一个输出隧道时，所有其它条件分支边框的同一位置上也会出现类似隧道。只要有一个输出隧道没有连线，该结构上的所有输出隧道都显示为白色正方形。每个条件分支的同一输出隧道可以定义不同的数据源，但各个条件必须兼容这些数据类型。右键单击输出隧道并从快捷菜单中选择**未连线时使用默认**，所有未连线的隧道将使用隧道数据类型的默认值。

用条件结构进行错误处理

将错误簇连接到条件结构的条件选择器接线端时，条件选择器标签将显示两个选项：错误和无错误。错误时边框为红色，无错误时边框为绿色。发生错误时，条件结构将执行错误子程序框图。

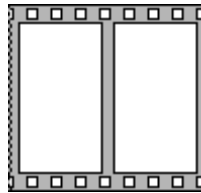
关于错误处理的详细信息见第 6 章 *运行和调试 VI 的错误处理* 一节。

顺序结构

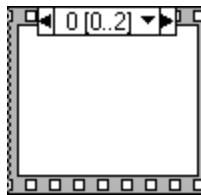
顺序结构包含一个或多个按顺序执行的子程序框图或帧。跟程序框图其它部分一样，在顺序结构的每一帧中，数据依赖性决定了节点的执行顺序。在 LabVIEW 中并不常用顺序结构。

顺序结构有两种类型：平铺式顺序结构和层叠式顺序结构。

如下图所示，平铺式顺序结构可以一次显示所有帧。当所连接的数据都传递至该帧时，将按照从左到右的顺序执行所有帧，直到执行完最后一帧。每帧执行完毕后将数据传递至下一帧。



如下图所示，层叠式顺序结构将所有的帧依次层叠，因此每次只能看到其中的一帧，并且按照帧 0、帧 1、直至最后一帧的顺序执行。



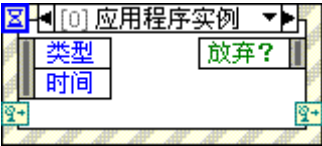
充分利用 LabVIEW 固有的并行机制，避免使用太多顺序结构。顺序结构虽然可以保证执行顺序但同时也阻止了并行操作。例如，如果不使用顺序结构，使用 PXI、GPIB、串口、DAQ 等 I/O 设备的异步任务就可以与其它操作并发运行。

需控制执行顺序时，可以考虑建立节点间的数据依赖性。例如，数据流参数（如错误 I/O）可用于控制执行顺序。

关于错误 I/O 的详细信息见第 6 章 *运行和调试 VI 的错误处理* 一节；关于流经参数的详细信息见第 5 章 *创建程序框图的数据流参数* 一节。

事件结构

如下图所示，事件结构包括一个或多个子程序框图或事件条件分支，运行结构时将执行其中某一条件分支或子程序框图。



事件结构将等待直至某一事件发生，并执行相应条件分支从而处理该事件。事件可以来自用户界面、外部 I/O 或应用程序的其它部分。用户界面事件包括鼠标点击、键盘按键等动作。外部 I/O 事件诸如硬件定时器或触发器在完成数据采集或发生错误情况时产生信号。通过编程可以产生其它类型的事件，并实现与应用程序的不同部分进行通讯。LabVIEW 支持用户界面事件和通过编程产生的事件，但不支持外部 I/O 事件。



注 只有 LabVIEW 完整版和专业版开发系统中才包括事件结构。在 LabVIEW 基础版中可以运行带有这些功能的 VI，但无法重新配置事件处理组件。

用字符串、数组和簇将数据分组

字符串、数组和簇对可将数据分组。字符串将 ASCII 字符序列归为一组。数组将相同类型的数据元素归为一组。簇将不同类型的数据元素归为一组。

用字符串将数据分组

字符串是可显示的或不可显示的 ASCII 字符序列。字符串提供了一个独立于操作平台的信息和数据格式。常用的字符串操作包括：

- 创建简单的文本信息。
- 将数值数据以字符串形式传送到仪器，再将字符串转换为数值。
- 将数值数据存储到磁盘。如需将数值数据保存到 ASCII 文件中，须在数值数据写入磁盘文件前将其转换为字符串。
- 用对话框指示或提示用户。

在前面板上，字符串以表格、文本输入框和标签的形式出现。LabVIEW 提供了用于对字符串进行操作的内置 VI 和函数，可对字符串进行格式化、解析字符串等编辑操作。

前面板上的字符串

字符串输入控件和显示控件可模拟文本输入框和标签。

关于字符串输入控件和显示控件的更多信息见第 4 章 *创建前面板* 中的 *字符串控件* 一节。

字符串显示类型

右键单击前面板上的字符串输入控件或显示控件，从下表所示的显示类型中选择。该表还提供了每个显示类型的范例。

显示类型	说明	消息
正常显示	可打印字符以控件字体显示。不可显示字符通常显示为一个小方框。	有四种显示类型。\\ 是反斜杠符号。
診“\\”代码显示	所有不可显示字符显示为反斜杠。	There\\sare\\sfour\\sdisplay\\stypes.\\n\\\\sis\\sa\\sbackslash.

显示类型	说明	消息
密码显示	每一个字符（包括空格在内）显示为星号(*)。	***** *****
十六进制显示	每个字符显示为其十六进制的 ASCII 值，字符本身并不显示。	5468 6572 6520 6172 6520 666F 7572 2064 6973 706C 6179 2074 7970 6573 2E0A 5C20 6973 2061 2062 6163 6B73 6C61 7368 2E

表格

表格控件用于在前面板上创建表格。表格的每一个单元格都是一个字符串，一个单元即某一行和某一列的交叉处。因此，表格表示一个二维字符串数组。

关于数组的更多信息见本章 *数组* 一节。

字符串的编辑、格式化和解析

字符串函数可通过以下方式编辑字符串：

- 查找、提取和替换字符串中的字符或子字符串。
- 将字符串中的所有文本转换为大写或小写。
- 在字符串中查找和提取匹配模式。
- 从字符串中提取一行。
- 将字符串中的文本移位和反序。
- 连接两个或多个字符串。
- 删除字符串中的字符。

关于以编程方式编辑字符串时如何最少地占用内存的更多信息见 *LabVIEW 帮助* 中的 *LabVIEW Style Checklist*。关于使用字符串函数编辑字符串的范例见 `labview\examples\general\strings.llb`。

字符串的格式化和解析

如需在另一个 VI、函数或应用程序中使用数据，通常须先将数据转换为字符串，再将字符串格式化为 VI、函数或应用程序能够读取的格式。例如，Microsoft Excel 要求字符串含有分隔符，如制表符、逗号与空格。Excel 用分隔符分隔数字或单词，并存入单元格。

例如，如需写入二进制文件函数将一维数值数组写入电子表格，须将该数组格式化成字符串并用制表符等分隔符将各个数字分开。如需写入电子表格文件 VI 将一个数值数组写入电子表格，必须用数组至电子表格字符串转换函数将数组格式化并指定格式和分隔符。

字符串函数可执行以下任务：

- 从一个字符串中提取字符串子集。
- 将数据转换为字符串。
- 格式化字符串用于文字处理或电子表格应用程序。

文件 I/O VI 和函数可将字符串保存到文本和电子表格文件中。

格式说明符

许多情况下，须在“字符串”函数的**格式字符串**参数中输入一个或多个格式说明符以格式化一个字符串。格式说明符是一个指明数值数据与字符串间如何相互转换的代码。LabVIEW 用转换代码确定参数的文本格式。例如，格式说明符 `%x` 可将十六进制整数与字符串相互转换。

用数组和簇将数据分组

数组和簇控件及函数可将数据分组。数组将相同类型的数据元素归为一组。簇将不同类型的数据元素归为一组。

数组

数组由元素和维度组成。元素是组成数组的数据。维度是数组的长度、高度或深度。数组可以是一维或多维的，在内存允许的情况下每一维度可有多达 $(2^{31}) - 1$ 个元素。

可以创建数值、布尔、路径、字符串、波形和簇等数据类型的数组。对一组相似的数据进行操作并重复计算时，可考虑使用数组。数组最适于存储从波形采集而来的数据或循环中生成的数据（每次循环生成数组中的一个元素）。

限制

数组中不能再创建数组。允许创建多维数组或创建每个簇中含有一个或多个数组的簇数组。不能创建元素为子面板控件、选项卡控件、.NET 控件、ActiveX 控件、图表或多曲线 XY 图的数组。

关于簇的更多信息见本章簇一节。

索引

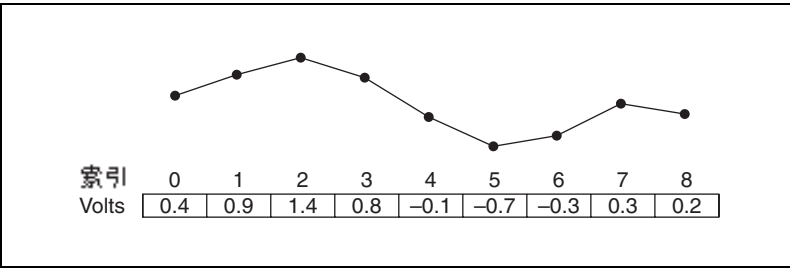
定位数组中的某个特定元素需为每一维度建一个索引。在 LabVIEW 中，通过索引可浏览整个数组，也可从程序框图数组中提取元素、行、列和页。

数组举例

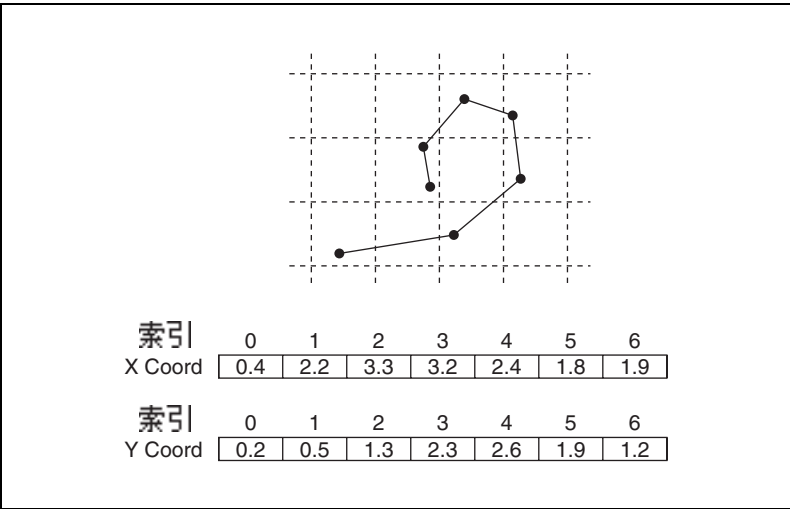
例如，太阳系中的九颗行星可以用一个简单的文本数组表示。在 LabVIEW 中可用含有九个元素的一维字符串数组表示。

数组元素是有序的。数组通过索引访问数组中任意一个特定的元素。索引以零开始，即索引的范围是 0 到 $n - 1$ ，其中 n 是数组中元素的个数。例如，对于九颗行星而言， $n = 9$ ，因此索引的范围是 0 到 8。地球是第三个行星，因此其索引为 2。

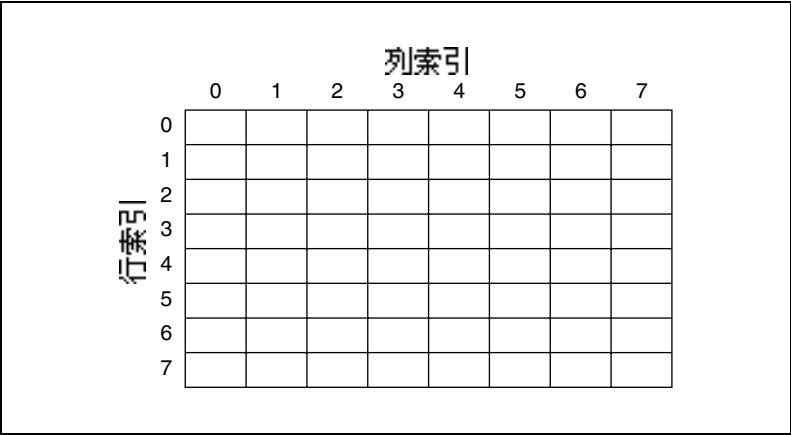
下图为一个数组的例子：以数值数组表示波形，数组的每个元素是具有相继时间间隔的电压值。



下图所示的例子更为复杂：以点数组表示的图形，其中每个点是包含一对表示 X 坐标和 Y 坐标的数值簇。

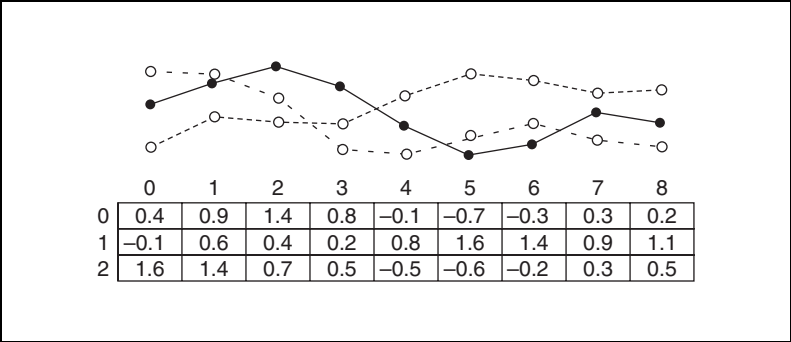


前两个范例都使用了一维数组。二维数组元素存储在网格中。需要一个行索引和一个列索引来定位数组中的某一个元素，并且这两个索引都从零开始。下图显示了一个 8 列 8 行的二维数组，其中包含 $8 \times 8 = 64$ 个元素。



例如，一个棋盘有八列和八行共 64 个位置。每个位置可为空或有一个棋子。二维字符串数组可表示一个棋盘。其中每个字符串是占据棋盘上相应位置的一个棋子的名称，或是空字符串（当位置为空时）。

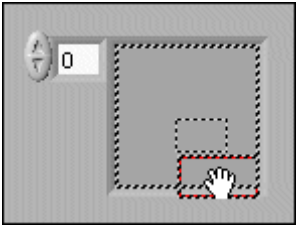
为前面例子中的一维数组添加一行可将数组推广到二维数组。下图显示了以二维数值数组表示的一组波形。其中行索引指定波形，列索引指定波形上的点。



关于使用数组的范例见 labview\examples\general\arrays.llb。

创建数组输入控件、显示控件和常量

通过以下方式可在前面板上创建一个数组输入控件或数组显示控件：在前面板上放置一个数组外框，然后将一个数据对象或元素拖曳到该数组外框中。数据对象或元素可以是数值、布尔、字符串、路径、引用句柄、簇输入控件或显示控件。如下图所示。



数组外框会自动调整大小以容纳新对象。

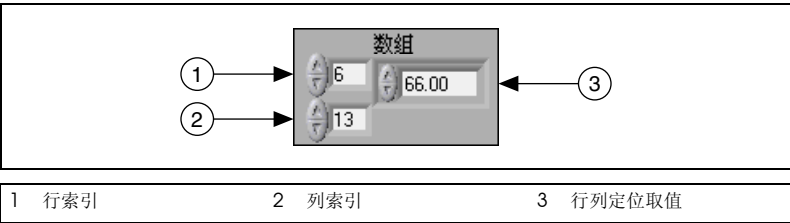
如需在程序框图中创建数组常量，则先从**函数**选板上选择数组常量，将数组外框放置于程序框图上，然后将字符串常量、数值常量、布尔常量或簇常量放入数组外框。数组常量可存储常量数据或同另一个数组进行比较。

创建多维数组

如需在前面板上创建一个多维数组控件，则右键单击索引框并从快捷菜单中选择**添加维度**。也可改变索引框的大小直到出现所需的维数。如需一次删除数组的一个维度，右键单击索引框并从快捷菜单中选择**删除维度**。也可改变索引框的大小来删除维度。

如需在前面板上显示某个特定的元素，可在索引框中输入索引数字或使用索引框上的箭头找到该数字。

例如，一个二维数组包含行和列。左边的两个方框中上面的索引为行索引，下面的索引为列索引。如下图所示。行和列右边的组合显示指定位置的数据值。位置在第六行、第十三列处的值为 **66**。如下图所示。



行和列是从零开始的，即第一列为列 0，第二列为列 1，依此类推。将下面数组中的索引显示的值改为行 1，列 2，将显示 6。

0	1	2	3
4	5	6	7
8	9	10	11

如试图显示超出数组维度范围的某一行或某一列，数组显示控件将变暗以表示该数据没有定义，同时 LabVIEW 将显示该数据类型的默认值。数据类型的默认值取决于该数组的数据类型。

定位工具可调整数组的大小并一次显示多行或多列。

数组函数

数组函数可创建数组并对其操作。例如，执行以下操作：

- 从数组中提取单个数据元素。
- 在数组中插入、删除或替换数据元素。
- 分解数组。

创建数组函数可通过编程方式创建数组。也可使用循环创建数组。

关于使用循环创建数组的信息见第 8 章 *循环和结构* 中的 *使用循环创建数组* 一节。

关于在循环中使用数组函数时如何最少地占用内存的更多信息参见 *LabVIEW 帮助* 中的 *LabVIEW Style Checklist*。

自动调整数组大小的函数

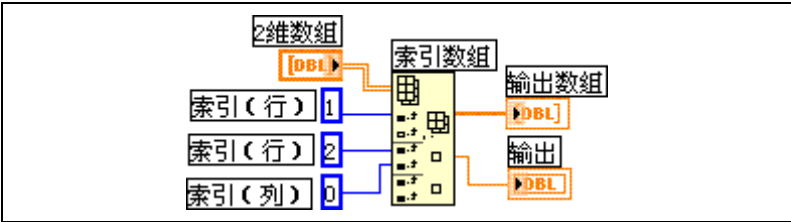
索引数组，替换数组子集，数组插入，删除数组元素和数组子集等函数可自动调整大小以匹配所连接的输入数组的维数。例如，如将一个一维数组连接到以上某一个函数，则该函数只显示单个索引输入。如将一个二维数组连接到同样的函数，则该函数显示两个索引输入，其中一个用于行索引，另一个用于列索引。

定位工具可手动调整这些函数的大小，以便通过这些函数访问多个数组元素或子数组（行、列或页）。扩展这些函数中的某个函数时，该函数将根据与之相连数组的维数的增加而增加。如将一个一维数组连接到以上某个函数，则该函数将以单个索引输入为单位扩展。如将一个二维数组连接到这个函数，该函数将以两个索引输入为单位扩展，其中一个用于行索引，另一个用于列索引。

连接的索引输入决定了要访问或修改的子数组的形状。例如，“索引数组”函数的输入为一个二维数组，但只连接了行索引输入，则提取的是该数组的

完整的一行。如只连接了列索引输入，则提取的是该数组的完整的一列。如同时连接了行索引输入和列索引输入，则提取的是该数组的单个元素。每个输入分组都是独立的，因此可访问数组中任何维度的任何部分。

该程序框图使用“索引数组”函数提取二维数组中的一行和一个元素。如下图所示。



如需访问数组中的多个连续值，则可将“索引数组”函数扩展且无需为所扩展的索引输入端赋值。例如，要提取某个二维数组的第一、二、三行，可将该索引数组函数扩展三个单位，然后将一维数组显示控件连接到每个子数组的输出端。

数组的默认数据

索引超出数组范围时，数组元素参数将会生成默认值。数组大小函数可确定数组的大小。

如使用 While 循环导致索引超过数组中最后元素，或“索引数组”函数的索引输入端赋了一个太大的数值，或将空数组赋给了“索引数组”函数，都会无意中超过数组索引的范围。

关于索引的更多信息见第 8 章 *循环和结构* 中的 *自动索引循环* 一节。关于数据类型默认值的更多信息参见 *LabVIEW 快速参考指南*。

簇

簇将不同类型的数据元素归为一组。LabVIEW 错误簇是簇的一个例子，它包含一个布尔值、一个数值和一个字符串。簇类似于文本编程语言中的记录或结构体。

关于使用错误簇的更多信息见第 6 章 *运行和调试 VI* 中的 *错误簇* 一节。

将几个数据元素捆绑成簇可消除程序框图上的混乱连线，减少子 VI 所需的连线板接线端的数目。连线板最多可有 28 个接线端。如前面板上要传送给另一个 VI 的输入控件和显示控件多于 28 个，则应将其其中的一些对象组成一个簇，然后为该簇分配一个连线板接线端。

程序框图上的绝大多数簇的连线样式和数据类型接线端为粉红色。错误簇的连线样式和数据类型终端显示为深黄色。由数值控件组成的簇，有时也称为

点，其连线样式和数据类型接线端为褐色。褐色的数值簇可连接到数值函数，如加或平方根函数，以便对簇中的所有元素同时进行同样运算。

簇元素顺序

尽管簇和数组元素都是有序的，但如需解除捆绑，必须一次解除所有簇元素的捆绑或使用按名称解除捆绑函数根据名称解除捆绑。簇不同于数组的地方还在于簇的大小是固定的。与数组一样，簇包含的不是输入控件即是显示控件。簇不能同时含有输入控件和显示控件。

簇元素有自己的逻辑顺序，与它们在簇外框中的位置无关。放入簇中的第一个对象是元素 0，第二个为元素 1，依此类推。如删除某个元素，则顺序会自动调整。簇顺序决定了簇元素在程序框图中的“捆绑”和“解除捆绑”函数上作为接线端出现的顺序。右键单击簇边框，从快捷菜单中选择**重新排序簇中控件**可查看和修改簇顺序。

如需连线两个簇，则二者必须有相同数目的元素。由簇顺序确定的相应元素的数据类型也必须兼容。例如，如一个簇中的双精度浮点数值在顺序上对应于另一个簇中的字符串，那么程序框图的连线将显示为断开且 VI 无法运行。如数值的表示不同，LabVIEW 会将它们强制转换成同一种表示法。

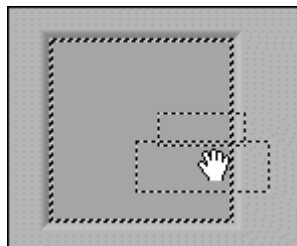
簇函数

簇函数可创建和操作簇。例如，执行以下操作：

- 从簇中提取单个数据元素。
- 向簇添加单个数据元素。
- 将簇拆分成单个数据元素。

创建簇输入控件、显示控件和常量

通过以下方式在前面板上创建一个簇输入控件或簇显示控件：在前面板上放置一个簇外框，再将一个数据对象或元素拖曳到簇外框中，数据对象或元素可以是数值、布尔、字符串、路径、引用句柄、簇输入控件或簇显示控件。



如需在程序框图中创建一个簇常量，则从**函数**选板中选择一个簇常量，将该簇外框放置于程序框图上，再将字符串常量、数值常量、布尔常量或簇常量放置到该簇外框中。簇常量用于存储常量数据或同另一个簇进行比较。

图形和图表

图形或图表用于图形化显示采集或生成的数据。

图形和图表的区别在于各自不同的数据显示和更新方式。含有图形的 VI 通常先将数据采集到数组中，再将数据绘制到图形中。该过程类似于电子表格，即先存储数据再生成数据的曲线。数据绘制到图形上时，图形不显示之前绘制的数据而只显示当前的新数据。图形一般用于连续采集数据的快速过程。

与图形相反，图表将新的数据点追加到已显示的数据点上以形成历史记录。在图表中，可结合先前采集到的数据查看当前读数或测量值。当图表中新增数据点时，图表将会滚动显示，即图表右侧出现新增的数据点，同时旧数据点在左侧消失。图表一般用于每秒只增加少量数据点的慢速过程。

图形和图表的类型

LabVIEW 包含以下类型的图形和图表：

- **波形图和图表**—显示采样率恒定的数据。
- **XY 图**—显示采样率非均匀的数据及多值函数的数据。
- **强度图和图表**—在二维图上以颜色显示第三个维度的值，从而在二维图上显示三维数据。
- **数字波形图**—以脉冲或成组的数字线的形式显示数据。
- **混合信号图**—显示波形图、XY 图和数字波形图所接受的数据类型。同时也接受包含上述数据类型的簇。
- **(Windows) 三维图形**—在前面板 ActiveX 对象的三维图上显示三维数据。

关于各种图形和图表的范例见 `labview\examples\general\graphs`。

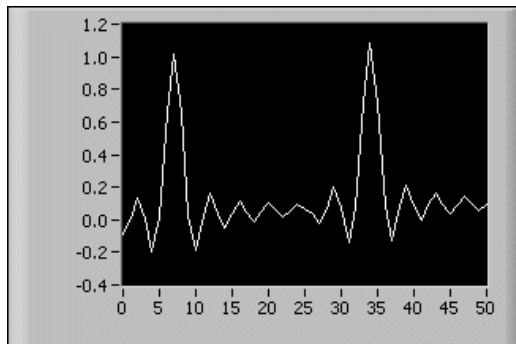
波形图和图表

LabVIEW 使用波形图和图表显示具有恒定速率的数据。

波形图

波形图用于显示测量值为均匀采集的一条或多条曲线。波形图仅绘制单值函数，即在 $y = f(x)$ 中，各点沿 x 轴均匀分布。例如一个随时间变化的波形。

下图显示了一个波形图的范例。



波形图可显示包含任意个数据点的曲线。波形图接收多种数据类型，从而最大程度地降低了数据在显示为图形前进行类型转换的工作量。



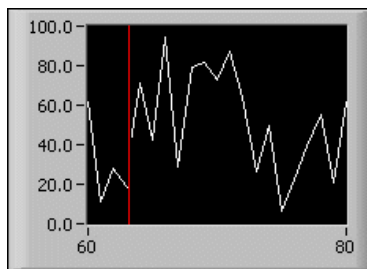
注

数字波形图用于显示数字数据。关于数字波形图及其接收的数据类型的更多信息见本章 *数字波形图* 一节。

关于波形图所接收的数据类型，见 `labview\examples\general\graphs\gengraph.11b` 中的 *Waveform Graph VI* 范例。

波形图表

波形图表是显示一条或多条曲线的特殊数值显示控件，一般用于显示以恒定速率采集到的数据。下图显示了一个波形图表的范例。



波形图表会保留来源于此前更新的历史数据，又称缓冲区。右键单击图表，从快捷菜单中选择 **图表历史长度** 可配置缓冲区大小。波形图表的默认图表历史长度为 1,024 个数据点。向图表传送数据的频率决定了图表重绘的频率。

关于波形图表的范例见 `labview\examples\general\graphs\charts.11b`。

波形数据类型

波形数据类型包含波形的数据、起始时间和时间间隔 (Δt)。可使用“创建波形”函数创建波形。默认状态下,很多用于采集或分析波形的 VI 和函数都可接收和返回波形数据类型。将波形数据连接到一个波形图或波形图表时,该波形图或波形图表将根据波形的数据、起始时间和 Δx 自动绘制波形。将一个波形数据的数组连接到波形图或波形图表时,该图形或图表会自动绘制所有波形。

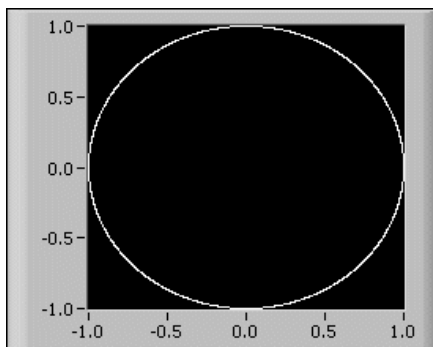
关于数字波形数据类型的更多信息见本章的 *数字波形数据类型* 一节。

XY 图

XY 图是通用的笛卡尔绘图对象,用于绘制多值函数,如圆形或具有可变时基的波形。XY 图可显示任何均匀采样或非均匀采样的点的集合。

XY 图中可显示 Nyquist 平面、Nichols 平面、S 平面和 Z 平面。上述平面的线和标签的颜色与笛卡尔线相同,且平面的标签字体无法修改。

下图显示了一个 XY 图的范例。

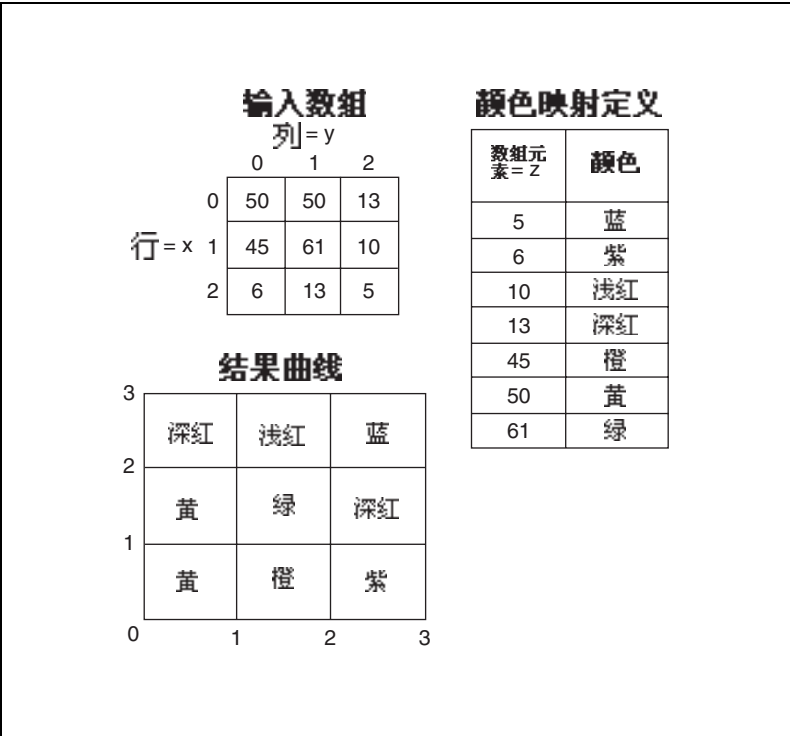


XY 图可显示包含任意个数据点的曲线。XY 图接收多种数据类型,从而将数据在显示为图形前进行类型转换的工作量减到最小。

关于 XY 图的范例见 `labview\examples\general\graphs\gengraph.llb` 中的 XY Graph VI。

强度图和图表

强度图和图表通过在笛卡尔平面上放置颜色块的方式在二维图上显示三维数据。例如,强度图和图表可显示图形数据,如温度图和地形图(以量值代表高度)。强度图和图表接收三维数字数组。数组中的每一个数字代表一个特定的颜色。在二维数组中,元素的索引可设置颜色在图形中的位置。下图显示了强度图表操作的有关概念。



数据行在图形或图表上将以新列显示。如希望以“行”的方式显示该行，则可将一个二维数组数据类型连接到强度图形或图表，右键单击该图形或图表，从快捷菜单中选择**转置数组**。

数组索引与颜色块的左下角顶点对应。颜色块有一个单位面积，即由数组索引所定义的两点间的面积。强度图或图表最多可显示 256 种不同颜色。

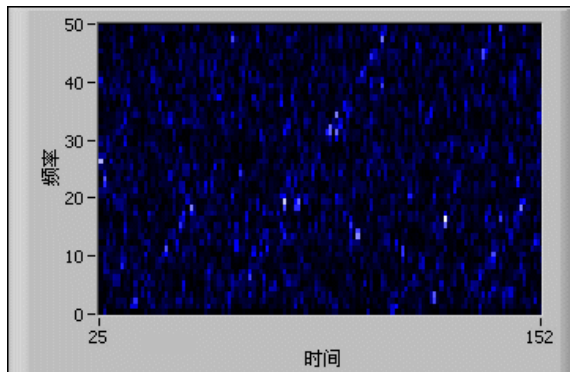
关于强度图和图表的范例见 labview\examples\general\graphs\intgraph.llb。

强度图表

在强度图表上绘制一个数据块以后，笛卡尔平面的原点将移动到最后一个数据块的右边。图表处理新数据时，新数据出现在旧数据的右边。如图表显示已满，则旧数据将从图表的左边界移出。这一点类似于带状图表。

关于带状图表的更多信息见本章的**配置图表更新模式**一节。

下图表示一个强度图表的范例。



强度图表和波形图表共享部分可选项，如标尺图例和图形工具选板，右键单击图表，从快捷菜单中选择**显示项**可显示或隐藏上述选项。此外，由于强度图表将颜色作为第三个维度，因此一个类似于颜色梯度控件的标尺可定义强度图表的范围和数值到颜色的映射。

关于颜色映射的更多信息见本章的**强度图和图表的颜色映射**一节。

与波形图表一样，强度图表也有一个来源于此前更新而产生的历史数据，又称缓冲区。右键单击图表，从快捷菜单中选择**图表历史长度**可配置缓冲区大小。强度图表缓冲区的默认大小为 128 个数据点。强度图表的显示需要占用大量的内存。

强度图

强度图类似于强度图表，但它并不保存先前的数据，也不接收更新模式。每次将新数据传送到强度图时，新数据将替换旧数据。和其它图形一样，强度图也有游标。每个游标可显示图形上指定点的 x 、 y 和 z 值。

关于游标的更多信息见本章的**图形游标**一节。

强度图和图表的颜色映射

强度图或强度图表通过颜色在二维图上显示三维数据。为强度图或强度图表设置好颜色映射后，可配置其颜色标尺。颜色标尺包括至少两个随机刻度，每个刻度均包含数值和对应的显示颜色。强度图或强度图表所显示的颜色与指定颜色的数值一一对应。颜色映射适用于数据范围的可视化显示，如曲线数据超过阈值时。

用定义颜色梯度数值控件颜色的方式可为强度图和图表设置交互式颜色映射。



注 强度图或图表显示的颜色会受到显卡所能显示的颜色和颜色数量的限制，同时还受分配给显示所用的颜色数的限制。

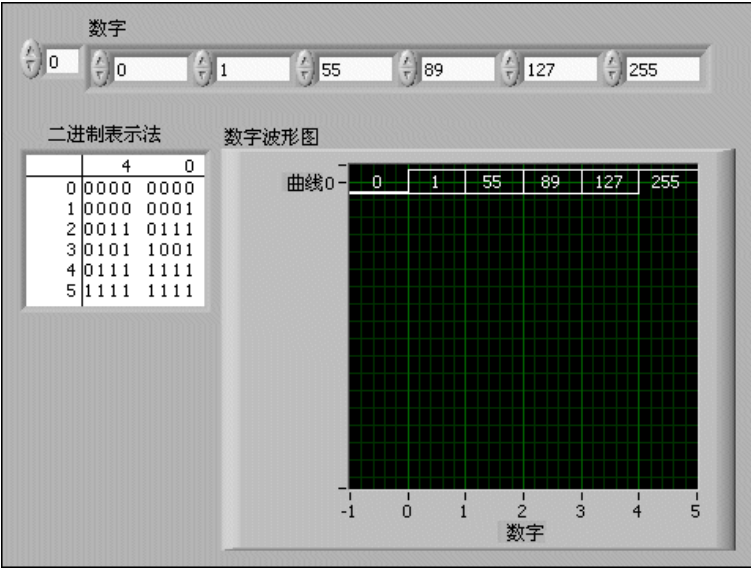
关于颜色映射的范例见 labview\examples\general\graphs\intgraph.llb 中的 Create IntGraph Color Table VI。

数字波形图

数字波形图用于显示数字数据，尤其适于用到定时框图或逻辑分析器时使用。

数字波形图接收数字波形数据类型、数字数据类型和上述数据类型的数组作为输入。默认状态下，由于数字波形图压缩数字总线，因此该图形会在单条曲线上绘制数字数据。如连接了一个数字数据数组，则数字波形图将按照数组的顺序为每个数组元素绘制不同的曲线。

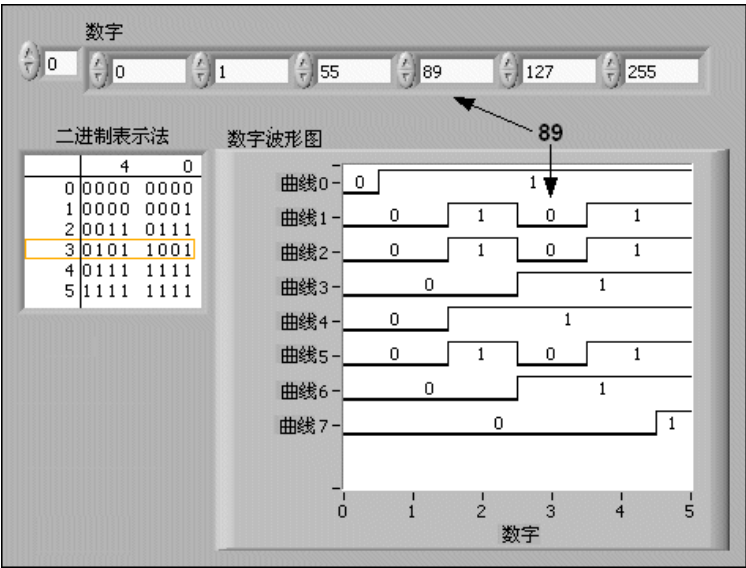
以下前面板中的数字波形图显示了在单条曲线上绘制数字数据。VI 将**数字**数组中的数字转换为数字数据，然后在**二进制表示法**数字数据显示控件中显示这些数字的二进制表示。在该数字图形中，数字 0 以无顶部直线的形式表示所有数字位的值为零。而数字 255 则以无底部直线的形式来表示所有二进制位的值为 1。



右键单击 y 标尺，从快捷菜单中选择**扩展数字总线**，可绘制数字数据的每一个采样值。每条曲线表示数字图中的各个不同二进制位。可为绘制在数字波形图上的数据自定义外观。

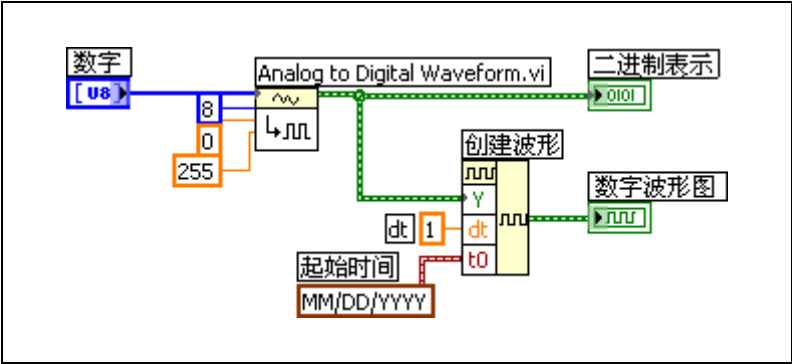
关于自定义数字波形图的更多信息见本章的 *自定义数字波形图* 一节。

以下前面板中的数字波形图显示了 **数字** 数组中的六个数字。



二进制表示法 数字显示控件显示了这些数字的二进制表示。表中的每一列代表一个二进制位。例如，数字 89 在内存中需要 7 个二进制位（第 7 列的 0 表示未使用的二进制位）。数字波形图上的点 3 绘制了表示数字 89 必需的 7 个二进制位，数值 0 表示曲线 7 上未使用的第 8 个二进制位。

以下 VI 显示了将数字数组转换为数字数据，并用创建波形函数收集在数字数据控件中输入的起始时间、时间间隔 (Δt) 以及数字以显示数字数据。



关于数字数据控件的更多信息见第 4 章 *创建前面板中的数字数据控件* 一节。

关于数字波形图的范例见 labview\examples\general\graphs\DWDT Graphs.llb。

数字波形数据类型

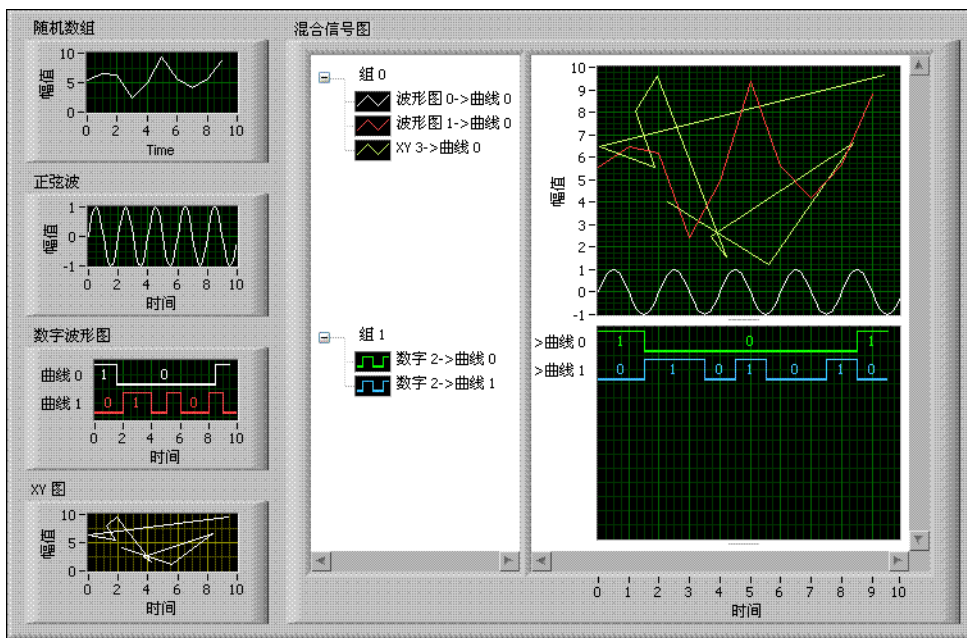
数字波形数据类型包含数字波形的起始时间、时间间隔 (Δx)、数据和属性。可使用“创建波形”函数创建数字波形。将数字波形数据连接到一个数字波形图上时, 该图形会根据时间信息和数字波形数据自动绘制波形。将数字波形数据连接到数字数据显示控件可查看数字波形的采样和信号。

关于波形数据类型的更多信息见本章的波形数据类型一节。

混合信号图

混合信号图可显示模拟数据及数字数据, 且接受所有波形图、XY 图和 数字波形图所接受的数据。

一个混合信号图中可能包含多个绘图区域。但一个绘图区域仅能显示数字曲线或者模拟曲线之一而无法兼有二者。LabVIEW 在绘图区域中绘制图像上数据。混合信号图将在必要时自动创建足以容纳所有模拟和数字数据的绘图区域。向一个混合信号图添加多个绘图区域时, 每个绘图区域都有其各自的 y 标尺。所有绘图区域共享同一个 x 标尺, 以便比较数字数据和模拟数据的多个信号。下图显示了一个混合信号图的范例。



关于混合信号图接收的数据类型的范例见 labview\examples\general\graphs\gengraph.11b 中的 Mixed Signal Graph VI。

三维图形

大量实际应用中的数据，例如某个表面的温度分布、联合时频分析、飞机的运动等，都需要在三维空间中可视化显示数据。三维图形可令三维数据可视化，修改三维图形属性可改变数据的显示方式。



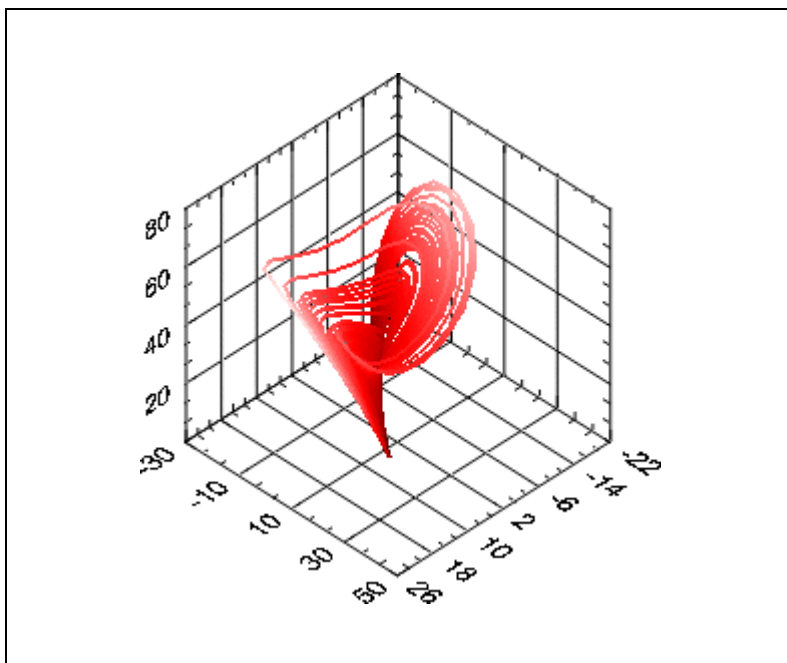
注

仅有 Windows 版的 LabVIEW 完整版和专业版开发系统中才有三维图形控件。

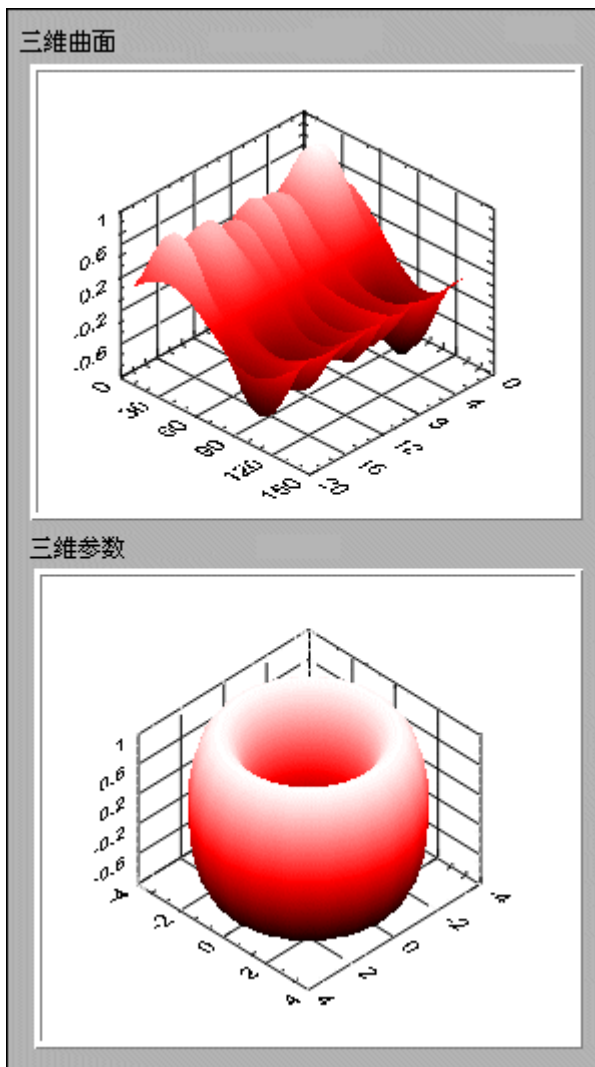
LabVIEW 中包含以下三维图形：

- **三维曲面图**—在三维空间绘制一个曲面。
- **三维参数图**—在三维空间绘制一个参数曲面。
- **三维曲线图**—在三维空间绘制一条曲线。

三维图形与三维图形 VI 连接后可用于绘制曲线和曲面。曲线包含图形上的单个点，每个点均具有 x、y 和 z 坐标，VI 用线连接这些点。曲线可理想地显示运动对象的轨迹，如飞机的飞行轨迹。下图显示了一个三维曲线图范例。



曲面图用 x 、 y 和 z 数据绘制图形上的各点，再将这些点连接，形成数据的三维曲面。例如，可用曲面图绘制地形图。下图显示了三维曲面图和三维参数图的范例。



三维图形利用了 ActiveX 技术及处理三维显示的 VI。选中一个三维图形时，LabVIEW 将在前面板上放置一个包含该三维图形控件的 ActiveX 容器。同时在程序框图上放置一个指向该三维图形控件的引用。LabVIEW 会将该引用连接至三个三维图形 VI 中的一个。

自定义图形和图表

每个图形和图表都提供了各种选项，用于自定义图形和图表的外观、提供更多显示信息及突出显示数据等。尽管图形和图表绘制数据的方式不同，但也有一些相同的快捷菜单项。有些选项仅适用于特殊类型的图形或图表。

关于图形或图表特有选项的更多信息见本章的 *自定义图形* 和 *自定义图表* 两节。

多个 X 标尺和 Y 标尺

波形图、XY 图、强度图、数字波形图和 (Windows) 平台上的三维图形均接收多个 x 标尺和 y 标尺，而所有图表只接收多个 y 标尺。混合信号图仅接收一个 X 标尺。在图形或图表上，使用多个标尺可显示 x 标尺或 y 标尺不共享的多条曲线。如需为图形或图表添加多个标尺，右键单击图形或图表的标尺，从快捷菜单中选择 **复制标尺**。

自动调整标尺

图形和图表均可自动调整水平标尺和垂直标尺，以便与连接到图形或图表上的数据相吻合。这被称为自动调整标尺。右键单击图形或图表，从快捷菜单中选择 **X 标尺 » 自动调整 X 标尺** 或 **Y 标尺 » 自动调整 Y 标尺**，即可打开或关闭自动调整标尺。默认状态下，图形和图表已启用自动调整标尺功能。不过自动调整标尺会降低系统的性能。

操作工具或标签工具可直接改变图形和图表的水平标尺或垂直标尺。

格式化 X 标尺和 Y 标尺

属性 对话框中的 **格式与精度** 选项卡可指定 x 轴和 y 轴的标尺在图形或图表上的显示方式。

默认状态下，x 标尺用浮点表示法并带有时间标签，y 标尺用自动格式表示并带有幅值标签。如需配置图形和图表的标尺，右键单击该图形或图表，从快捷菜单中选择 **属性**，然后在 **图形属性** 对话框或 **图表属性** 对话框中加以配置。

属性 对话框中的 **格式与精度** 选项卡可指定图形或图表标尺数值格式。单击 **标尺** 选项卡可重命名标尺并对其格式化。默认状态下，图形或图表标尺在自动切换到科学表示法之前最多可显示六位数字。

在 **格式与精度** 选项卡中，选择 **高级编辑模式** 可显示文本选项，直接输入格式化字符串。要自定义标尺的外观和数值精度，请输入格式化字符串。

图形工具选板

VI 运行时使用图形工具选板可与图形或图表进行交互。如下图所示。



通过图形工具选板可进行游标移动、缩放、平移显示图像等操作。右键单击图形或图表，从快捷菜单选择**显示项 » 图形工具选板**可显示或隐藏图形模板。图形工具选板包含还包括包含信号信息的各种属性下列按钮，从左到右依次为：

- **游标移动工具**（仅对图形有效）— 移动所显示图形上的游标。
- **缩放**— 放大或缩小显示图形。
- **平移工具**— 在显示区域内选中并移动曲线。

单击图形工具选板中的某个按钮，即可移动游标、缩放或平移显示图像。启用的按钮会显示绿色指示灯。

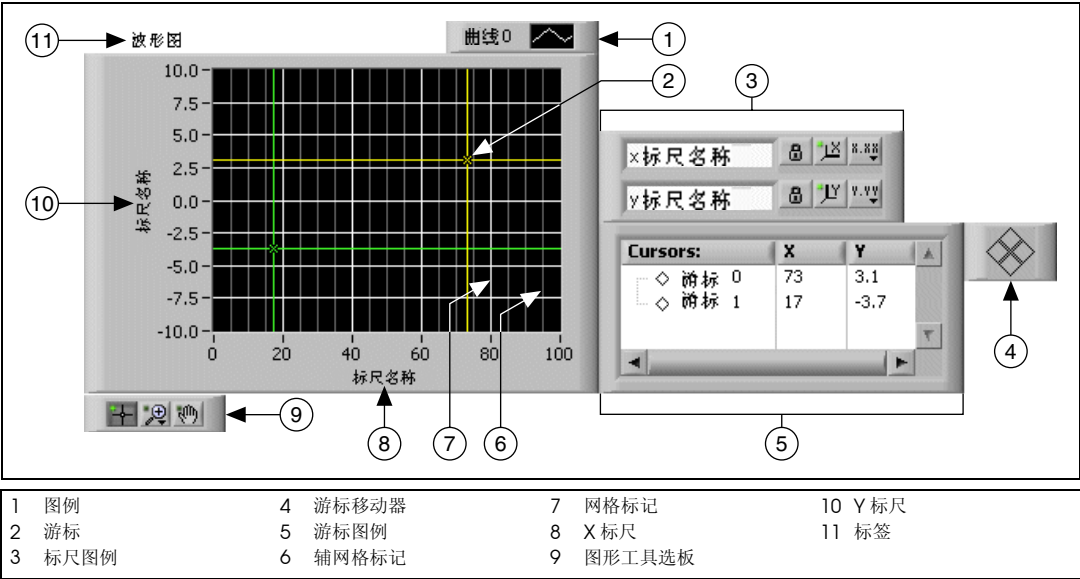
自定义图形和图表的外观

通过显示或隐藏选项可自定义图形和图表的外观。右键单击图形或图表，从快捷菜单选择**显示项**可显示或隐藏以下选项：

- **图例**— 定义曲线的颜色和式样。改变图例的大小可显示多条曲线。
- **标尺图例**— 定义标尺标签、配置标尺属性。
- **图形工具选板**— 在 VI 运行时移动游标、缩放以及平移图形或图表。
- **X 标尺和 Y 标尺**— 对 x 标尺和 y 标尺进行格式化。
关于格式化标尺的更多信息见本章的**格式化 X 标尺和 Y 标尺**一节。
- **游标图例**（仅对图形有效）— 在已定义的点坐标处显示刻度。图形上可显示多个游标。
- **X 滚动条**— 滚动显示图形或图表中的数据。滚动条可查看图形或图表当前未显示的数据。
- **数字显示**（仅对波形图表有效）— 显示图表的数值。

自定义图形

每个图形均包含各种选项，用户可自定义图形以满足数据显示的要求。例如，可修改图形游标的行为和外观或配置图形标尺。下图显示了一个图形所具有的元素。



图例中的绝大多数元素均可添加。方法是：右键单击图形，从快捷菜单中选择**显示项**选择相应的元素即可。右键单击图形，从快捷菜单中选择相应选项可配置图形。

图形游标

在图形上用游标可读取绘图区域上某个点确切的值。游标值显示在游标图例中。

右键单击该图形，从快捷菜单选择**显示项 » 游标图例**可查看游标图例。右键单击游标图例中任意区域，选择**创建游标**，从快捷菜单中选择游标模式便可在图形中添加游标。

游标模式定义了游标位置。游标包含下列模式：

- **自由**—不论曲线的位置，游标可在整绘图区域内自由移动。
- **单曲线**—仅将游标置于与其关联的曲线上。游标可在曲线上移动。右键单击游标图例，从快捷菜单中选择**关联至**，游标可与一个或所有曲线实现关联。
- **多曲线**—将游标置于绘图区域内的特定数据点上。多曲线游标可显示与游标相关的所有曲线在指定 x 值处的值。游标可置于绘图区域内的任意曲线上。右键单击游标图例，从快捷菜单中选择**关联至**，游标可与一个或所有曲线实现关联。该模式只对混合信号图形有效。

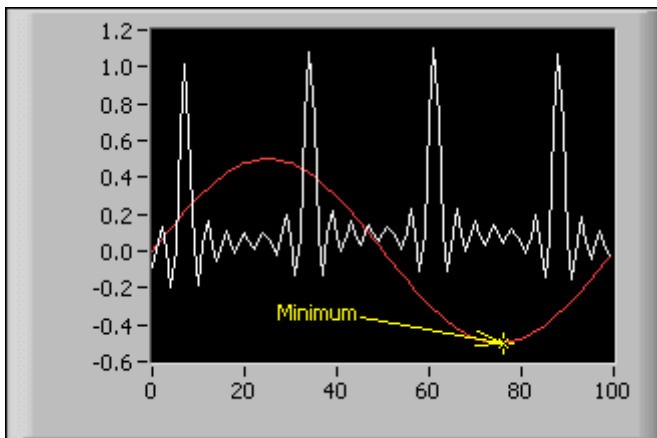


注 创建游标模式后无法对其进行修改，必须删除游标并创建另一游标。

有多种方式定义游标的外观。如在曲线上为游标添加标签、指定游标的颜色、指定线条、点和游标式样等。右键单击游标图例所在行，从快捷菜单中选择相应选项可自定义游标。

图形注释

图形注释可在绘图区域内高亮显示数据点。注释包含一个标签和用于确定注释和数据点的箭头。一个图形可有任意多个注释。下图显示了一个使用注释的图形范例。



右键单击图形，从快捷菜单中选择**数据操作 » 创建注释**可显示**创建注释**对话框。**创建注释**对话框可指定注释名称、确定注释指向绘图区域内的曲线的方式。

创建注释对话框中的**锁定风格**下拉菜单可指定注释与绘图区域内曲线关联的方式。**锁定风格**包含以下选项：

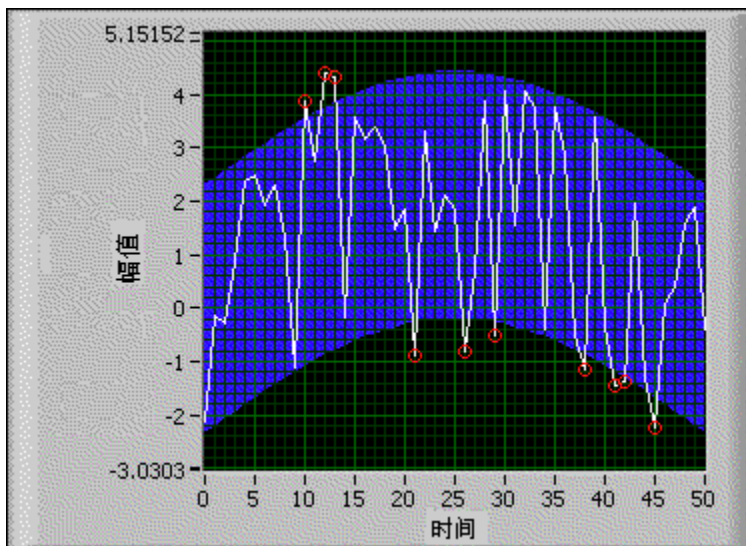
- **自由**—可在绘图区域内自由移动注释。LabVIEW 并未将注释与绘图区域内的曲线关联。
- **关联至所有曲线**—可将注释移至绘图区域内任意曲线上最近的数据点。
- **关联至一条曲线**—仅可在特定曲线上移动注释。

可通过多种方式自定义注释的行为和外观，如隐藏或显示绘图区域内的注释名称或箭头，指定注释的颜色，指定线条、点和注释的式样等。右键单击注释，从快捷菜单中选择相应选项便可自定义注释。

如需删除注释，右键单击该注释，从快捷菜单中选择**删除注释**。右键单击图形，从快捷菜单中选择**数据操作 » 删除全部注释**，则可删除绘图区域内的所有注释。

在图形绘图区域内绘图

通过在绘图区域中绘制前景、背景、或中间图像，可自定义图形绘图区域。用于绘制曲线图像的画布具有一个坐标系，其原点 (0, 0) 总是位于图形绘图区域的左上角。强度图、混合信号图和 波形图均有“曲线图像”属性。该属性用于在背景中创建自定义网格、在中间创建数据包围、或在图形绘图区域的前景中通过形状对数据点进行注释。在以下图像中，示范了通过圆周来注释数据点，同时创建一个包络从而为数据指定一个已定义的容忍度。

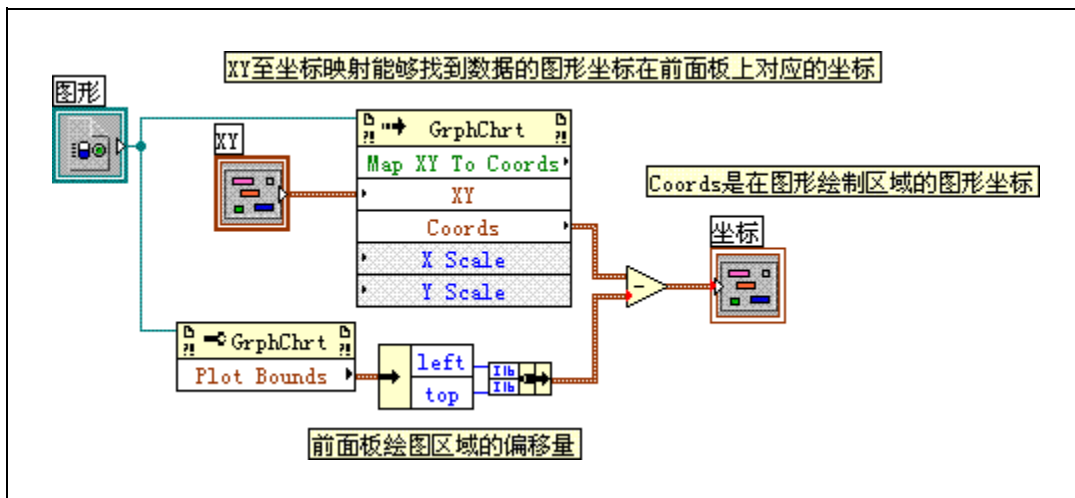


背景图像位于曲线数据和网格线的后方。通过“曲线图像：背景”属性可设置该背景图像用于混合信号图及波形图。

前景图像位于曲线数据和网格线的前方。通过“曲线图像：前景”属性可设置该前景图像用于混合信号图及波形图。

中间图像位于曲线数据和网格线之间。通过“曲线图像：中间”属性可设置该中间图像用于混合信号图及波形图。

使用“曲线图像”这一属性时，LabVIEW 将从图像绘图区域的原点开始绘制。如希望通过该属性将图形数据点而不是绘制区域原点作为绘制的参考点，则可使用 XY 至坐标映射的方法将数据点的图形坐标映射到前面板上的坐标。接着，可得出图形的绘图区域与前面板原点间的偏移量，从而找到开始绘制的正确方位。以下程序框图显示了查找绘制的正确方位的一个方法。



注

在前面板上使用分隔栏，创建若干窗格。接着，计算出图形的绘图区域与前面板原点间的偏移量，从而找到开始绘制的正确方位。

关于各图形所接收的“曲线图像”属性，见波形图属性、强度图属性和混合信号图属性。

如需删除图形，则必须将一个空图像与合适的属性相连，或者将图像设置为透明。如需对带有自定义注释或包络的图形进行大小调整或复制，则必须再次运行 VI 以重绘图像。

关于“曲线图像”属性及“XY 至坐标映射”方法的使用范例，见 labview\examples\general\graphs\Graph Pictures.llb。

自定义三维图形

三维图形包括各种自定义操作选项，如三维绘图式样、标尺格式化、网格和绘图投影。由于三维图形使用了可处理三维效果的 ActiveX 技术和 VI，因此设置三维图形的选项与设置其它图形的选项有所不同。创建应用程序时，ActiveX 属性浏览器可设置三维图形的属性。右键单击三维图形，从快捷菜单中选择**属性浏览器**可显示 ActiveX 属性浏览器。

如需允许用户在运行时改变常规属性，或通过编程设置属性，可用三维图形属性 VI。

自定义数字波形图

可为数字波形图的曲线自定义外观。右键单击图例中的曲线，可调整线条粗细、设置转换类型和位置及对图形标签进行格式化。

自定义图表

与图形显示新数据并覆盖已存储数据的方式不同，图表对数据进行周期性更新并保留先前已经存储的历史数据。

可自定义图表以符合数据的显示要求。图表选项包括滚动条、标尺图例、图形工具选板、数字显示和标尺的时间表示等。也可修改图表历史长度、更新模式和曲线显示方式。

配置图表历史长度

LabVIEW 将已添加到图表中的数据点存储于一个缓冲区，又称图表历史。图表历史缓冲区的默认大小为 1024 个数据点。右键单击该图表，从快捷菜单中选择**图表历史长度**可配置历史缓冲区大小。图表滚动条可查看先前已经采集的数据。右键单击图表，从快捷菜单中选择**显示项 » X 滚动条**可显示滚动条。



注 大型图表的历史值可能占用大量的内存。

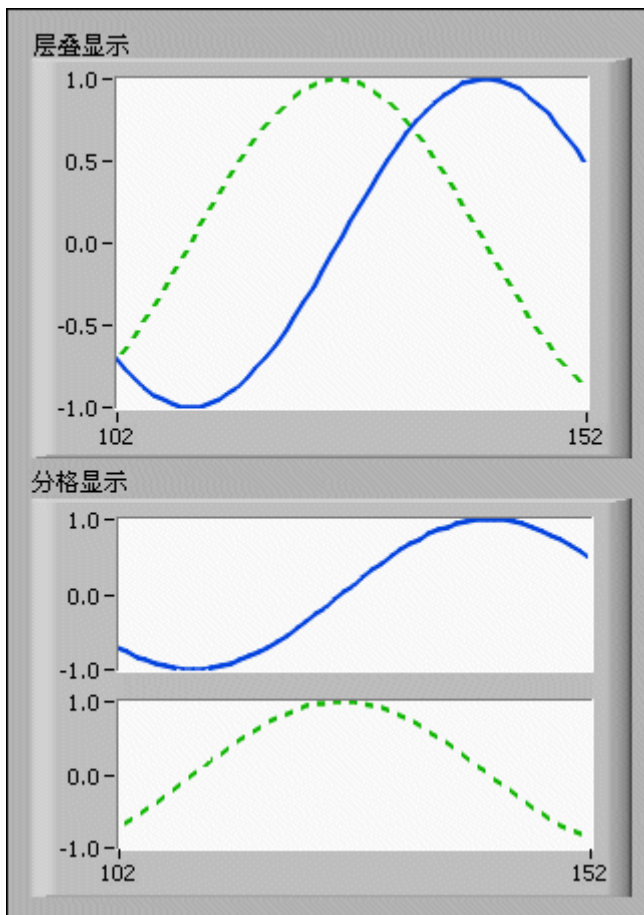
配置图表更新模式

可对图表的更新和新数据显示方式进行配置。右键单击该图表，从快捷菜单中选择**高级 » 更新模式**可配置图表刷新模式。图表的数据显示方式如下：

- **带状图表**—从左到右连续滚动地显示运行数据，旧数据在左，新数据在右。带状图表类似纸带图形记录器。图表的默认更新模式为**带状图表**。
- **示波器图表**—显示某一项数据，如脉冲或波形，并从左到右地滚读图表。图表将新数值绘制到前一个数值的右边。当曲线到达绘图区域的右边界时，LabVIEW 将擦除整条曲线并从左边界重新开始绘制。示波器图表的重新跟踪显示特性类似于示波器。
- **扫描图**—类似于示波器图表。两者的不同之处在于，扫描图表中旧数据在右新数据在左，并有一条垂直线将这两部分数据隔开；其次，当曲线到达绘图区域的右边界时，LabVIEW 并不擦除扫描图表中的曲线。扫描图的显示方式类似于心电图仪。

曲线的层叠显示和分格显示

有两种方式在一个图表上显示的多条曲线，其一为通过单个垂直标尺显示，即曲线的层叠显示；其二为通过多个垂直标尺显示，即曲线的分格显示。下图显示了层叠显示和分格显示的范例。



右键单击图表，从快捷菜单中选择**分格显示**可以多个垂直标尺的方式查看图表曲线；选择**层叠显示**则以单个垂直标尺的方式查看图表曲线。

关于不同类型图表及其所接收数据类型的范例见 labview\examples\general\graphs\charts.11b 中的 Charts VI。

文件 I/O

文件 I/O 操作可在文件中读写数据。**文件 I/O** 选板上的“文件 I/O”和函数可实现文件 I/O 的所有功能，其中包括：

- 打开和关闭数据文件。
- 读写数据文件。
- 读写电子表格格式的文件。
- 移动或重命名文件和目录。
- 修改文件特性。
- 创建、修改和读取配置文件。

使用一个 VI 或函数就可进行文件打开、读写和关闭操作，也可分别使用函数控制过程中的各个步骤。“读取测量文件”Express VI 和“写入测量文件”Express VI 可对 .lvrm、.tdm 或 .tdms 文件进行读取数据和写入数据的操作。

关于 .tdm 文件的更多信息见本章的 *使用存储 VI* 一节。

文件 I/O 基础

典型的文件 I/O 操作包括以下流程。

1. 创建或打开一个文件，文件打开后，引用句柄即代表该文件的唯一标识符。
关于引用句柄的更多信息见第 4 章 *创建前面板的对象或应用程序的引用* 一节。
2. 文件 I/O VI 或函数从文件中读取或向文件写入数据。
3. 然后关闭该文件。

文件 I/O VI 和某些文件 I/O 函数，如读取文本文件和写入文本文件可执行一般文件 I/O 操作的全部三个步骤。执行多项操作的 VI 和函数可能在效率上低于执行单项操作的函数。

选择文件 I/O 格式

采用何种**文件 I/O** 选板上的 VI 取决于文件的格式。LabVIEW 可读写的文件格式有文本文件、二进制文件和数据记录文件三种。使用何种格式的文件取决于采集和创建的数据及访问这些数据的应用程序。

根据以下标准确定使用的文件格式：

- 如需在其它应用程序（如 Microsoft Excel）中访问这些数据，使用最常见且便于存取的文本文件。
- 如需随机读写文件或读取速度及磁盘空间有限，使用二进制文件。在磁盘空间利用和读取速度方面二进制文件优于文本文件。
- 如需在 LabVIEW 中处理复杂的数据记录或不同的数据类型，使用数据记录文件。如果仅从 LabVIEW 访问数据，而且需存储复杂数据结构，数据记录文件是最好的方式。

如果数据本身不是文本格式（例如，图形或图表数据），由于数据的 ASCII 码表示通常要比数据本身大，因此文本文件要比二进制和数据记录文件占用更多内存。例如，将 -123.4567 作为单精度浮点数保存时只需 4 个字节，如使用 ASCII 码表示，需要 9 个字节，每个字符占用一个字节。

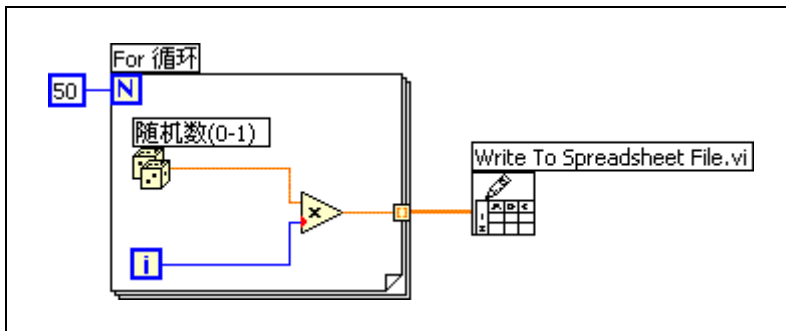
另外，很难随机访问文本文件中的数值数据。尽管字符串中的每个字符占用一个字节的空間，但是将一个数字表示为字符串所需要的空間通常是不固定的。如需查找文本文件中的第 9 个数字，LabVIEW 须先读取和转换前面 8 个数字。

用于常用文件 I/O 操作的 VI 和函数

文件 I/O 模板上的 VI 和函数可用于常见文件 I/O 操作，如读写以下类型的数据：

- 在电子表格文本文件中读写数值
- 在文本文件中读写字符
- 从文本文件读取行
- 在二进制文件中读写数据

以下程序框图说明如何用写入电子表格文件 VI 向电子表格文件传递数字。运行该 VI 时，LabVIEW 会提示是否将数据写入现有文件，或者创建新文件。



打开、读取、写入函数需输入文件路径。如没有连接文件路径，则出现对话框提示以指定所要读写的文件。

文件 I/O 选板上的函数可控制单个文件 I/O 出操作，这些函数可创建或打开文件，向文件读写数据及关闭文件。上述 VI 可实现以下任务：

- 创建目录。
- 移动、复制或删除文件。
- 列出目录内容。
- 修改文件特性。
- 对路径进行操作。

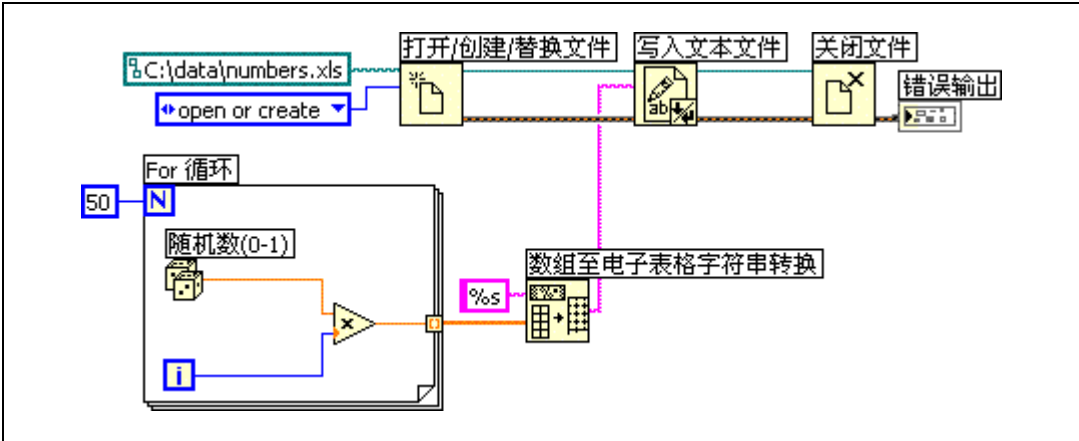
路径是一种 LabVIEW 数据类型，用来指定文件在磁盘上的位置。如下图所示。



路径包含文件所在的磁盘、文件系统根目录到文件之间的路径以及文件名。在控件中可按照平台特定的标准语法输入或显示一个路径。

关于路径控件的详细信息见第 4 章 *创建前面板的路径控件* 一节。

以下程序框图说明如何使用“文件 I/O”函数向电子表格文件传输数值数据。运行该 VI 时，打开 / 创建 / 替换文件函数会打开 `numbers.xls` 文件。写入文本文件函数将数值字符串写入文件。关闭文件函数将关闭文件。如不关闭文件，文件将继续留在内存中，且其它应用程序或用户无法访问该文件。



将上述程序框图与写入电子表格 VI 的程序框图相比较，两者完成的任务相同。程序框图用单个函数执行各个文件操作，如采用数组至电子表格字符串转换函数将数字数组转换成字符串。“写入电子表格文件”VI 执行多项文件操作，包括打开文件、将数字数组转换成字符串及关闭文件。

参考下列高级文件 I/O 操作的 VI 和函数范例：Write Datalog File
Example VI labview\examples\file\datalog.llb。

“文件 I/O”函数还可用于流盘操作，它可以减少函数因打开和关闭文件与操作系统交互的次数，从而节省内存资源。流盘是一项在进行多次写操作时保持文件打开的技术，如在循环中使用流盘。如将路径控件或常量连接至写入文本文件、写入二进制文件或写入电子表格文件函数，则函数将在每次函数或 VI 运行时打开关闭文件，增加了系统占用。避免对同一文件进行频繁的打开和关闭操作，可提高 VI 效率。

在循环之前放置打开 / 创建 / 替换文件函数，在循环内部放置读或写函数，在循环之后放置关闭文件函数，即可创建一个典型的流盘操作。此时只有写操作在循环内部进行，从而避免了重复打开关闭文件的系统占用。

对于速度要求高，时间持续长的数据采集，流盘是一种理想的方案。数据采集的同时将数据连续写入文件中。为获取更好的效果，在采集结束前应避免运行其它 VI 和函数（如分析 VI 和函数等）。

使用存储 VI

存储选板上的“存储”VI 可在二进制测量文件 (.tdm) 中读取和写入波形及波形属性。通过 .tdm 文件可在 NI 软件（如 LabVIEW 和 DIAdem）间进行数据交换。



注 “存储”VI 仅适用于 Windows 操作系统。

“存储”VI 将波形和波形属性组合，从而构成通道。通道组可管理一组通道。一个文件中可包括多个通道组。如按名称保存通道，就可从现有通道中快速添加或获取数据。除数值之外，“存储”VI 也支持字符串数组和时间标识数组。在程序框图上，引用句柄可代表文件、通道组和通道。

“存储”I 也可查询文件以获取符合条件的通道组或通道。

如开发过程中系统要求发生改动，或需要在文件中添加其它数据，则“存储”VI 可修改文件格式且不会导致文件不可用。

参考下列使用“存储”VI 的范例：`labview\examples\file\storage.llb`。

读取测量文件 Express VI 和写入测量文件 Express VI 也可对 .tdm 测量文件进行读取数据和写入数据的操作。

创建文本文件和电子表格文件

要将数据写入文本文件，必须将数据转化为字符串。要将数据写入电子表格，必须格式化字符串为包含分隔符（如制表符）的字符串。

关于格式化字符串的详细信息见第 9 章 *用字符串、数组和簇将数据分组的字符串的格式化和解析* 一节。

由于大多数文字处理应用程序读取文本时并不要求格式化的文本，因此将文本写入文本文件无需进行格式化。如需将文本字符串写入文本文件，可用写入文本文件函数自动打开和关闭文件。

写入二进制文件函数可创建独立于平台的文本文件。读取二进制文件函数可在独立于平台的文本文件中读取数据。

关于二进制文件的详细信息见 *创建二进制文件* 一节。

写入电子表格文件 VI 或数组至电子表格字符串转换函数可将来自图形、图表或采样的数据集转换为电子表格字符串。

由于文字处理应用程序采用了“文件 I/O”VI 无法处理的字体、颜色、样式和大小不同的格式化文本，因此从文字处理应用程序中读取文本可能会导致错误。

如需将数字和文本写入电子表格或文字处理应用程序，使用字符串函数和数组函数格式化数据并组合这些字符串。然后将数据写入文件。

关于使用上述函数进行格式化和组合数据的详细信息见第 9 章 *用字符串、数组和簇将数据分组的字符串的编辑、格式化和解析和数组函数* 一节。

格式化文件以及将数据写入文件

格式化写入文件函数可将字符串、数值、路径和布尔数据格式化为文本，并将格式化以后的文本写入文件。该函数可一次实现多项操作，而无需先用格式化写入文件函数格式化字符串，然后用写入文本文件函数将结果字符串写入文件。

关于格式化字符串的详细信息见第 9 章 *用字符串、数组和簇将数据分组的字符串的格式化和解析* 一节。

从文件中扫描数据

扫描文件函数可扫描文件中的文本获取字符串、数值、路径和布尔值并将该文本转换成某种数据类型。该函数可一次实现多项操作，无需先用读取二进制文件或读取文本文件函数读取数据，然后使用扫描字符串将结果扫描至文件。

创建二进制文件

写入二进制文件函数用于创建二进制文件。“写入二进制文件”函数的**数据**输入端可连接任何类型的数据。将相应类型的控件或常量连接至“读取二进制文件”函数的**数据类型**输入端，就可使用读取二进制文件函数指定要从文件中读取的数据类型。“写入二进制文件”“读取二进制文件”函数可读取由不同操作系统创建的文本文件。

参考下列使用“读取二进制文件”和“写入二进制文件”函数的范例：

创建数据记录文件

使用**数据记录**选板的“数据记录”函数采集数据并将数据写入文件，从而创建和读取数据记录文件。

无需将数据记录文件中的数据按格式处理。但是，读取或写入数据记录文件时，必须首先指定数据类型。例如，采集带有时间和日期标识的温度读数时，将这些数据写入数据记录文件需要将该数据指定为包含一个数字和两个字符串的簇。参考下列向数据记录文件写入数据的范例：Simple Temp Datalogger VI labview\examples\file\datalog.11b。

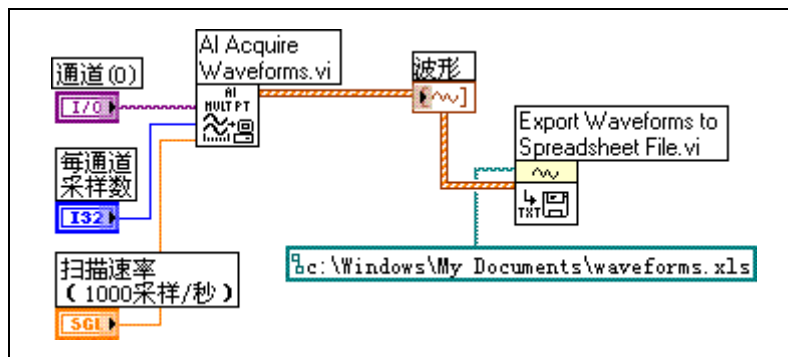
如读取一个带有时间和日期记录的温度读数文件，需将要读取的内容指定为包含一个数字和两个字符串的簇。参考读取数据记录文件的范例：Simple Temp Datalog Reader VI labview\examples\file\datalog.11b。

写入波形至文件

写入波形至文件和导出波形至电子表格文件 VI 可将波形写入文件。可将波形写入电子表格、文本文件或数据记录文件。

如只需在 VI 中使用波形，则可将该波形保存为数据记录文件 (.log)。

以下 VI 采集多个波形并在一个图形上进行显示，然后将这些波形写入电子表格文件。

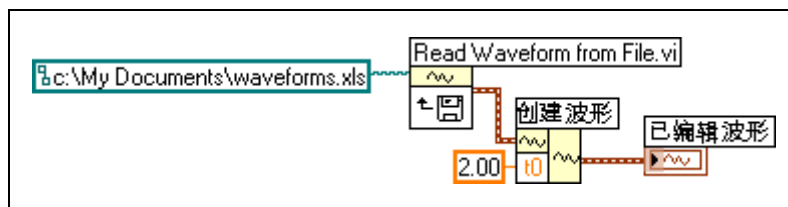


存储选板上的“存储”VI 或写入测量文件 Express VI 可将波形写入文件。

从文件中读取波形

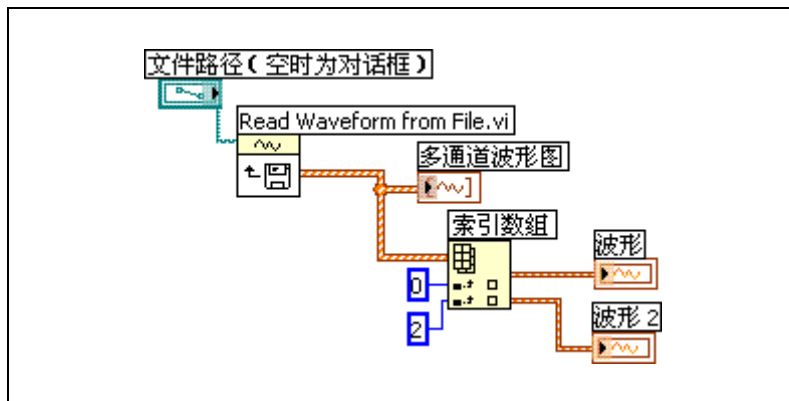
从文件读取波形 VI 可从文件中读取波形。读取单个波形后，可用创建波形函数添加或编辑波形数据元素，或用获取波形属性函数提取波形属性。

下列 VI 可从文件中读取一个波形，编辑波形的 **to** 元素，并将编辑后的波形绘制在图形上。



从文件读取波形 VI 也可从文件中读取多个波形。该 VI 返回一个波形数组，可在多曲线图形中显示。如需访问文件中的单个波形，需为波形数据类型建立数组索引。如以下程序框图所示。该 VI 可访问含有多个波形的文件。索引数组函数读取文件中的第一和第三个波形，并将它们绘制在两个独立的波形图上。

关于索引数组的详细信息见第 9 章 *用字符串、数组和簇将数据分组的数组* 一节。关于波形图的详细信息见第 10 章 *图形和图表的波形图* 一节。



存储选板上的“存储”VI或测量文件读取 Express VI 从文件中读取波形。

编制 VI 说明信息和打印 VI

LabVIEW 可编制 VI 说明信息及打印 VI。

编制 VI 说明信息的目的在于记录每个开发阶段的程序框图及前面板的信息。

打印 VI 时，根据需求选择相应打印选项。有些选项适于打印 VI 的相关信息，而另一些则适于打印 VI 生成的数据和结果报表。选择打印方式时应考虑以下因素：以手动还是通过编程打印、报表格式中需包括哪些内容选项、是否需要在独立可执行程序中设置打印功能、VI 在何种平台上运行等。

编制 VI 说明信息

LabVIEW 可为已完成的 VI 编制说明信息并为 VI 用户创建操作说明。在 LabVIEW 中可查看和打印该说明信息，并将其保存为 HTML、RTF 或文本文件。

如需创建有效的 VI 说明信息，应创建 VI 和对象的说明。

为 VI 及其对象（如输入控件和显示控件）创建说明，用于描述 VI 或对象的目的并向用户提供使用说明。在 LabVIEW 中可查看和打印说明，或将说明保存为 HTML、RTF 或文本文件。

选择**文件»VI 属性**，从**类别**下拉菜单中选择**说明信息**，便可创建、编辑和查看 VI 说明。右键单击对象并从快捷菜单中选择**说明和提示**，也可创建、编辑和查看对象说明。提示框是一种简短的说明，在 VI 运行时将鼠标移到某个对象上，提示框就会出现。如未在**说明和提示**对话框中输入提示，则不会出现提示框。将鼠标分别移到 VI 图标或对象上时，该 VI 或对象的说明会在**即时帮助**窗口中出现。



注 函数选板上的 VI 或函数无法为其输入说明。

打印 VI

以下为打印 VI 的主要方法：

- 选择**文件 » 打印窗口**将打印活动窗口的内容。
- 选择**文件 » 打印**将打印 VI 更全面的信息，包括前面板、程序框图、子 VI、控件、VI 历史等。
- 通过编程打印一个 VI 窗口或对一个含有 VI 说明信息或 VI 返回数据的报表进行打印或保存。

选择**文件 » 打印**将打印 VI 说明信息或将其保存为 HTML、RTF 或文本文件。说明信息可采用内置的说明信息格式或自定义的说明信息格式。说明信息可包含以下各项：

- 图标和连线板
- 前面板和程序框图
- 输入控件、显示控件和数据类型接线端
- 输入控件和显示控件的标签和标题
- VI 和对象的说明
- VI 层次结构
- 子 VI 列表
- 修订历史



注

但对于某些类型的 VI，其说明信息可能无法包括上述所有项。例如，多态 VI 没有前面板或程序框图，因此为多态 VI 所创建的说明信息中无法包括这两项。

LabVIEW 生成的 HTML 或 RTF 文件可创建用户已编译帮助文件。

(Windows) 可将 LabVIEW 生成的单个 HTML 文件编译为 HTML 帮助文件。**(Mac OS)** 可将 LabVIEW 生成的单个 HTML 文件用于 Apple 帮助文件。

可将 LabVIEW 生成的 RTF 文件编译为 **(Windows)** WinHelp 或 **(Linux)** HyperHelp 文件。



技术支持和专业服务

如需更多关于技术支持及专业服务的内容，请访问 National Instruments 网站 ni.com。

- **技术支持**—在线技术支持资源在 ni.com/support 上，包括以下内容：
 - **自助资源库**—请访问 National Instruments 网站，查阅有关软件驱动及更新、可搜索的知识库、产品使用手册、疑难解答向导、数千个 程序实例、产品教程、应用指南、仪器驱动等相关信息。
 - **免费技术支持**—所有注册用户将免费获得基础服务，其中包括在 ni.com/forums 的 NI 开发交流平台上与数百位应用工程师进行技术交流。National Instruments 的应用工程师将确保您所有的问题得以解答。

如需了解更多当地的技术支持服务，请登录
ni.com/services 或 ni.com/contact 与当地办事处联系。

- **培训及认证**—请访问 ni.com/training 查阅有关定制培训、电子教学虚拟课堂、互动光盘及认证项目信息。同时也可在全球各地报名参加面授课程。
- **系统集成**—NI 联盟伙伴可帮助解决项目时间限制、内部技术资源短缺或其它项目问题。详情请致电当地 NI 办事处或登录网站 ni.com/alliance。

如您在 ni.com 找不到所需信息，请客户联系 NI 当地的办事处或 NI 总部。全球办公室电话号码见本书扉页。您也可登录 ni.com/niglobal 全球办事处查找最新的办事处联系方式、技术支持电话、电子邮件地址并获取最新消息。

词汇表

符号	前缀	值
y	yocto	10^{24}
z	zepto	10^{21}
a	atto	10^{18}
f	femto	10^{15}
p	pico	10^{12}
n	纳 (nano)	10^9
μ	微 (micro)	10^6
m	毫 (milli)	10^3
c	centi	10^2
d	deci	10^1
da	deka	10^1
h	hecto	10^2
k	千 (kilo)	10^3
M	兆 (mega)	10^6
G	千兆 (giga)	10^9
T	太 (tera)	10^{12}
P	peta	10^{15}
E	exa	10^{18}
Z	zetta	10^{21}
Y	yotta	10^{24}

数值 / 符号

- ∞

无穷。
- Δ

Delta；差值。x 表示 x 从一个值到另一个值的改变量。

π Pi。

1D 一维。

2D 二维。

3D 三维。

A

A 安培。

ASCII 美国标准信息交互码。

B

编辑模式 (edit mode) 可以对 VI 进行修改的状态。

编译 (compile) 将高级代码转换为机器可执行代码的过程。在用户创建或编辑修改 VI 后，LabVIEW 在首次运行这些 VI 前将自动进行编译。

标尺 (scale) 图形、图表和数值输入控件和显示控件的组成部分，包含一连串固定间隔的标识，用于表示测量单位。

标量 (scalar) 由刻度上的某一点所表示的数值，标量是相对于数组的单个值。标量布尔值和簇均为其各自对应数据类型的单个实例。

标签 (label) 用于命名、描述前面板或程序框图中的对象或区域的文字对象。

标签工具 (Labeling tool) 创建标签并向文本窗口输入文本的工具。

表示法 (representation) 数值数据类型的子类型，其中包括 8 位、16 位和 32 位有符号和无符号整数，以及单精度、双精度、扩展精度和浮点数。

波形 (waveform) 以特定采样率采集的多个电压读数的集合。

波形图表
(waveform chart) 以特定速率绘制数据点的显示控件。

布尔控件 (Boolean controls and indicators) 前面板对象，用于操作和显示布尔数据 (TRUE 或 FALSE)。

C

采样 (sample)	单个模拟或数字输入或输出的数据点。
菜单栏 (menu bar)	用于列出程序中的主菜单名的水平条，位于标题栏下方。虽然部分菜单和命令对所有的程序都适用，但一般每个程序都有其特定的菜单栏。
操作工具 (Operating tool)	在控件中输入数据或对控件进行操作的工具。
操作员	执行程序操作和监控的人。
测量设备 (measurement device)	指某个 DAQ 设备如 E 系列多功能 I/O (MIO) 设备、SCXI 信号调理模块、开关模块等。
层叠式顺序结构 (Stacked Sequence structure)	以数字顺序执行子程序框图的程序控制结构。在无法使用数据流参数 (flow-through parameter) 时，可以通过顺序结构控制不存在数据依赖关系的节点的执行顺序。层叠式顺序结构每次仅显示其中一帧，并按照顺序执行所有帧。
层次结构 窗口 (Hierarchy window)	见 VI 层次结构窗口 (VI Hierarchy window)。
常量 (constant)	常量是程序框图中提供固定数据值的接线端。 也见 通用常量 (universal constant) 和用户自定义常量 (user-defined constant)。
程序框图 (block diagram)	程序或算法的图形化表示。程序框图由可执行图标（即节点）和在节点间传送数据的连线组成。程序框图即为 VI 的源代码，位于 VI 的程序框图窗口。
触发 (trigger)	引发或启动任何形式数据采集的事件。
簇 (cluster)	一组有序的、未索引的任何数据类型的数据元素，数据类型可以是数值、布尔、字符串、数组或簇等。类似于 C 语言中的结构体。这些元素必须同为输入控件或同为显示控件。
簇外框 (cluster shell)	包含簇元素的前面板对象。
错误簇 (error cluster)	包含一个布尔状态显示控件、一个数字代码显示控件和一个字符串的显示控件。
错误输出 (error out)	输出 VI 的错误结构。
错误输入 (error in)	输入 VI 的错误簇。
错误信息 (error message)	软件或硬件存在故障，或是输入了无法接收的数据时的说明信息。

D

DAQ	见数据采集 (data acquisition, DAQ) 或 NI-DAQ。
DAQ 设备 (DAQ device)	用于采集或生成数据的设备, 可以包含多个通道和转换设备。DAQ 设备包括插入式驱动器、PCMCIA 卡和 DAQPad 设备, 均连接到计算机的 USB 或 1394 (FireWire™) 端口。SCXI 模块也属于 DAQ 设备。
DAQ 助手 (DAQ Assistant)	配置测量任务、通道和刻度的图形化界面。
带状图表 (strip chart)	模仿纸带图表记录器的数值绘图显示控件, 并随着绘制数据而滚动显示。
当前 VI (current VI)	前面板、程序框图或图标编辑器处于活动状态的 VI。
点 (point)	以两个 16 位的整数分别代表横坐标和纵坐标的簇。
调节圈或调节柄 (resizing circles or handles)	位于对象边框上的圆圈或手柄, 表示在该位置上可调整对象大小。
顶层 VI (top-level VI)	位于 VI 层次结构中最顶层的 VI, 该术语用于区分 VI 和子 VI。
定位工具 (Positioning tool)	移动对象、改变对象大小的工具。
断点 (breakpoint)	调试时的执行中暂停。
断点工具 (Breakpoint tool)	用于在 VI、节点或连线上设置断点的工具。
断开的 VI (broken VI)	发生错误导致无法运行的 VI; 在断开的 运行 按钮中用断开的箭头表示。
对话框 (dialog box)	当一个应用程序需要更多信息来执行命令时出现的弹出窗口。
对象 (object)	前面板和程序框图上条目的统称, 包括输入控件、显示控件、节点、连线以及导入的图片。
多态 (polymorphism)	节点可根据不同的数据形式、类型、结构进行自动调节的功能。

E

Express VI	用于进行常规测量任务的子 VI。Express VI 可在配置对话框中进行配置。
二维 (two-dimensional)	包括 2 个维数, 类似于包含多个行或列的数组。

F

For 循环 (For Loop)	按规定次数执行子程序的交互式循环结构。相当于文本代码： <code>For i = 0 to n - 1, do...</code>
范围 (range)	测量、接收、传输某一量值的上下限度范围，用上限值和下限值表示。
方法 (method)	对象接收信息时的执行过程。方法通常与类相关。
符号 (glyph)	小的图片或图标。
复选框 (checkbox)	在对话框中小的方形选框，用于选中项目或取消勾选。复选框一般用于在多个选项中进行选择。可同时选中多个复选框。

G

G	LabVIEW 使用的图形化编程语言。
GPIO	见 通用接口总线 (General Purpose Interface Bus)。
工具 (tool)	特定的光标工具，可用于实现特定操作。
工具栏 (toolbar)	工具栏包含各种命令按钮，可用于运行和调试 VI。
工具 选板 (Tools palette)	程序中包含工具集合的一个选板，该选板中的工具可用于编辑和调试前面板和程序框图中的对象。

H

hex	十六进制。十六进制计数系统。
函数 (function)	内置的执行单元，相当于文本编程语言中的操作符、函数或语句。
函数 选板 (Functions palette)	包含 VI、函数、程序框图结构和常量的选板。
滑块 (slider)	输入控件或显示控件上的可移动部分，滑块的移动反映数值的改变。
缓冲区 (buffer)	存储采集或生成数据的临时存储空间。
活动窗口 (active window)	表示当前接收用户输入的窗口，通常为最前端的窗口。活动窗口的标题栏以高亮显示。单击一个窗口或从 窗口 菜单中选择一个窗口，可使该窗口成为活动窗口。

I

Inf	以浮点表示无穷的数字化表示。
I/O	输入 / 输出。数据在计算机系统的输入输出，包括通讯通道、操作输入装置、数据采集和控制接口等。
IVI	可互换虚拟仪器。为常规测试和测量仪器创建通用接口 (API) 的软件标准。

J

即时帮助窗口 (Context Help window)	当光标移动到每一个 LabVIEW 对象上时，即时帮助窗口用于显示该对象的基本信息。VI、函数、常数、结构、选板、属性、方式、事件、对话框和 项目浏览器 中的项均有即时帮助信息。
计数接线端 (iteration terminal)	For 循环或 While 循环的接线端，记录当前已完成的循环的次数。
接线端 (terminal)	用于传递数据的节点端口。
节点 (node)	程序执行单元。相当于文本编程语言中的语句、运算、函数和子程序。在程序框图中，节点包括函数、结构和子 VI。
结构 (structure)	程序控制单元，例如，平铺式和层叠式顺序结构 (Flat Sequence structure, Stacked Sequence structure)、条件结构 (Case structure)、For 循环 (For Loop) 或 While 循环 (While Loop) 等。
矩形 (rectangle)	包含 4 个 16 位整数的簇。前两个值确定左上角的横纵坐标位置。后两个值确定右下角的横纵坐标位置。
矩阵 (matrix)	由可表示线性方程中各项系数的数字或其它数学元素组成的矩形阵列。
句柄 (handle)	指向一块内存的指针的指示器，表示引用数组和字符串。一个字符串数组是指向一个内存块的句柄，这个内存块又包含了指向字符串的句柄。

K

空数组 (empty array)	数据类型已定义而元素个数为空的数组。例如，一个在数据显示窗口具有数值控件但是没有为任何元素定义值的数组为空数组。
控件 (control)	以交互方式向 VI 输入数据或以编程方式向子 VI 输入数据的前面板对象，如旋钮、按钮或转盘。
控件 (Controls) 选板 (palette)	包含前面板中输入控件、显示控件和修饰型对象的选板。
控制流 (control flow)	由指令的先后顺序决定执行顺序的编程系统。大多数文本编程语言为控制流语言。
库 (library)	见 LLB 或项目库。
快捷菜单 (shortcut menu)	右键单击某个对象时出现的下拉菜单。快捷菜单只作用于其所属的对象。
捆绑函数 (bundle function)	捆绑各类元素来创建簇的函数。

L

LabVIEW	Laboratory Virtual Instrument Engineering Workbench。LabVIEW 是一种图形化编程语言，用图标代替文本行来创建程序。
LED	发光二极管，指示灯。
LLB	包含用于特定用途相关 VI 集合的 LabVIEW 文件。
类 (class)	包含属性、方法和事件的一个类别。类被排列成一定的层次结构，每个类继承上一级别类的属性和方法。
离散 (discrete)	具有独立变量的不连续值，通常为采样时间。
连线 (wire)	节点间的数据路径。
连线板 (connector pane)	前面板或程序框图窗口右上角区域，显示 VI 接线端的模式；用于定义可连接到 VI 的输入和输出端。
连线段 (wire segment)	单条水平或垂直的连线段。
连线分支 (wire branch)	连线中从交叉点到交叉点、接线端到交叉点或中间没有交叉点的接线端到接线端的连线段。

连线工具 (Wiring tool)	用于连接接线端之间数据路径的工具。
连线接线头 (wire stubs)	当连线工具移至 VI 或函数节点上时，在未连接的接线端旁所出现的线头。
连线节点 (connector)	VI 或函数节点的一部分，包含输入和输出接线端。数据通过接线端在节点中进行传输。
列表框 (listbox)	对话框内的一个选项框，列出一个命令的所有可选项。例如，列表框中可列出磁盘上的文件名列表。

M

MAX	见 Measurement & Automation Explorer。
Measurement & Automation Explorer	Windows 平台下的标准 NI 硬件配置和分析环境。
脉冲 (pulse)	振幅瞬时从零值到非零值，该信号称为脉冲。
默认 (default)	预设值。许多 VI 输入在没有用户指定值的情况下将使用默认值。
目录 (directory)	将文件方便地分组的一种结构模式。目录以地址的形式显示文件的位置，可以包含文件或文件子目录。

N

NaN	< 非法数值 >，指在不确定的浮点运算中产生的结果。一般用于不确定的操作结果，如 $\log(1)$ 。
NI-DAQ	所有 NI-DAQ 设备和信号调理组件的驱动程序。NI-DAQ 是一个可在应用程序开发环境 (ADE) 如 LabVIEW 中调用的 VI 和 ANSI C 函数扩展程序库，用于配置所有 NI 测量设备，如 M 系列多功能输入输出 (MIO)DAQ 设备、信号调理模块和开关模块。
NI-DAQmx	最新的 NI-DAQ 驱动程序，带有控制测量设备所需的最新 VI、函数和开发工具。NI-DAQmx 在以下方面优于前期版本的 NI-DAQ：在 LabVIEW、LabWindows™/CVI™, 和 Measurement Studio 中 DAQ Assistant 对设备通道设置和测量任务的支持；更优异的性能如单点模拟 I/O 更加快速、创建 DAQ 程序的 API 更为简洁并可使用更少的函数和 VI。

P

PXI	PCI 在仪器领域的扩展 (PCI eXtensions for Instrumentation)。基于计算机的模块化仪器平台。
配置程序工具 (configuration utility)	指 Windows 上运行的 Measurement & Automation Explorer 和 Mac OS 或 Linux 上运行的仪器配置工具。
频率 (frequency)	f, 速率的基本单位, 使用频率计数器或频谱分析仪测量每秒所发生的事件或振动。频率是信号周期的倒数。
平铺式顺序结构 (Flat Sequence structure)	以数字顺序执行子程序框图的程序控制结构。在无法使用数据流参数 (flow-through parameter) 时, 可以通过顺序结构控制不存在数据依赖关系的节点的执行顺序。平铺式顺序结构一次显示所有帧, 并按照从左到右的顺序执行, 直到执行完最后一帧。

Q

前面板 (front panel)	VI 的交互式用户界面。前面板外观类似于真实的物理仪器 (如示波器和万用表)。
强度图 (intensity map/plot)	使用颜色在二维图中显示三维数据的方法。
强制转换 (coercion)	LabVIEW 执行的一种类型自动转换, 用于改变一个数据元素的数值表示法。
强制转换点 (coercion dot)	出现在程序框图节点上, 以警示两个不同数字数据类型的数据被直接连接。任何数据类型与变体数据类型直接相连时也会出现强制转换点。
曲线 / 标绘图 (plot)	数据阵列在图形或图表上的图形表示。
驱动程序 (driver)	用于控制硬件设备 (如 DAQ 设备) 的软件。
驱动器 (drive)	a-z 范围内的一个字母再加一个冒号 (:), 表示一个逻辑磁盘驱动器。

R

人工数据依赖关系 (artificial data dependency)	数据流编程语言中的条件, 即根据数据的到达 (不是数值本身) 触发一个节点的执行。
--	---

S

扫描图 (sweep chart)	以示波器操作为模型的数值显示控件。扫描图与示波器图类似，但扫描图中包括一条垂直线用于分隔新旧数据。
上色工具 (Coloring tool)	设置前景色和背景色的工具。
设备 (device)	可以作为一个实体访问的仪器或控制器，用于控制或监控现实世界的输入输出端口。设备通常通过一些通讯网络连接到主机。 也见 DAQ 设备 (DAQ device) 和测量设备 (measurement device)。
示波器图 (scope chart)	以示波器操作为模型的数值显示控件。
事件 (event)	模拟或数字信号的条件或状态。
数据采集 (data acquisition, DAQ)	<ol style="list-style-type: none"> 1. 通过传感器、采集传感器和测试探针或测试装置采集并测量模拟或数字电子信号。 2. 生成模拟或数字电子信号。
数据记录 (datalog)	采集数据并同时将数据存储于磁盘文件中。LabVIEW 文件 I/O VI 和函数可用于记录数据。
数据记录文件 (datalog file)	将数据存为一系列单一的任意数据类型记录的文件（数据类型可在创建文件时指定）。一个数据记录文件中的所有记录都必须为同一数据类型，该类型也可以是复合类型。例如，可以指定每条记录为包含一个字符串、一个数值和一个数组的簇。
数据类型 (data type)	信息格式。LabVIEW 中绝大多数 VI 和函数接收的数据类型包括数值 (numeric)、数组 (array)、字符串 (string)、布尔 (Boolean)、路径 (path)、引用句柄 (refnum)、枚举 (enumeration)、波形 (waveform) 和簇 (cluster) 等。
数据流 (data flow)	由可执行节点组成的编程系统，这些可执行节点只有在接收到所有必需的输入数据后才会开始运行。节点在执行时将自动生成输出数据。LabVIEW 就是一个数据流系统。数据流经节点的动作决定了程序框图上 VI 和函数的执行顺序。
数据依赖关系 (data dependency)	数据流编程语言的条件，指一个节点只有在从其它节点接收数据后才能执行。 也见 人工数据依赖关系 (artificial data dependency)。
数值输入控件和显示控件 (numeric controls and indicators)	用于管理和显示数值数据的前面板对象。
数组 (array)	按一定顺序排列的带索引的同类数据元素列表。

数组外框 (array shell)	用于放置数组的前面板对象。一个数组外框包括一个索引显示框、一个数据对象窗口和一个可选标签。它可接收多种数据类型。
顺序结构 (sequence structure)	见平铺式顺序结构 (Flat Sequence structure) 或层叠式顺序结构 (Stacked Sequence structure)。
隧道 (tunnel)	在结构中的数据输入或输出接线端。

T

探针 (probe)	用于检查 VI 中数值的调试功能。
探针工具 (Probe tool)	在连线上设置探针的工具。
提示框 (tip strip)	小型黄色文本框，显示接线端名称以确定需连接的接线端。
条件分支 (case)	条件结构的一个子程序框图。
条件接线端 (conditional terminal)	While 循环 (While Loop) 的接线端，包含一个决定 VI 是否再执行下一次循环的布尔值。
条件结构 (Case structure)	条件控制结构，根据输入的条件选择执行几个子程序框图中的某一个。它综合了控制流语言中的 IF、THEN、ELSE 和 CASE 语句。
通道 (channel)	<ol style="list-style-type: none"> 1. 物理通道 (Physical) 一用于测量和发生模拟信号或数字信号的接线端或管脚。一个单一的物理通道可以包括多个端口，如一个差分模拟输入通道或一个八条数字线的端口。一个计数器也可以是一个物理通道，但计数器与其对应的接线端可采用不同的命名。 2. 虚拟通道 (Virtual) 一包括名称、物理通道、输入终端连接、测量或发生信号类型及刻度信息在内的一组属性设置。定义 NI-DAQmx 的虚拟通道可以在任务外（全局）或任务内（局部）。在传统 NI-DAQ 或更早的版本中配置虚拟通道是可选的，但对于在 NI-DAQmx 中进行的每个测量却是必不可少的。传统 NI-DAQ 在 MAX 中配置虚拟通道。而在 NI-DAQmx 中，虚拟通道既可以在 MAX 中也可以在用户程序中配置，可以将通道配置作为任务的一部分也可以独立配置通道。 3. 开关通道 (Switch) 一开关通道表示开关上的任意连接点。根据开关的拓扑结构，开关通道可能由一条或多条信号线组成（通常为一条、两条或四条）。开关通道无法用来创建虚拟通道。开关通道只能在 NI-DAQmx 开关函数和 VI 中使用。
通用常量 (universal constant)	可传出特定 ASCII 字符或标准数字常量的程序框图对象，并且无法进行修改，如 π 。

通用接口总线 (General Purpose Interface Bus)	GPIB。与 HP-IB 同义。通过计算机控制电子仪器的标准总线。也称 IEEE 488 总线，因为它是根据 ANSI/IEEE 488-1978、488.1-1987 和 488.2-1992 标准进行定义的。
图标 (icon)	程序框图上节点的图形化表示。
图表 (chart)	一个或多个图形的二维显示，图形上保留了原来数据的历史记录，用户可自定义最多保留多少记录。图表接收数据并且以逐个点或逐个数组的方式更新显示，在缓冲器中保留一定数目的数据点以便于显示。 也见 示波器图 (scope chart), 带状图 (strip chart) 和扫描图 (sweep chart)。
图例 (legend)	图形或图表的示例，用于显示在该图形或图表上的名称和样式。
图片 (picture)	图片显示控件同来创建图片的一系列制图指令。
图形 (Graph)	一条或多条曲线（或标绘图）的二维显示。图形成块地接收并绘制数据。
图形控件 (graph control)	在平面直角坐标系中显示数据的前面板对象。
拖曳 (drag)	用屏幕上的光标执行选择、移动、复制或删除对象的操作。

V

VI	见 虚拟仪器 (virtual instrument)。
VISA	见 虚拟仪器软件架构 (Virtual Instrument Software Architecture)。
VI 层次结构窗口 (VI Hierarchy window)	图形化显示 VI 和子 VI 的层次结构的窗口。

W

While 循环 (While Loop)	一种循环结构，重复执行某一代码段直至条件满足时才停止。
维数 (dimension)	数组的大小和结构。
无法显示的字符 (non-displayable characters)	无法显示的 ASCII 字符，如 null、backspace、tab 等。

X

下拉菜单 (pull-down menus)	菜单栏上的菜单。下拉菜单中的项通常为某一类菜单的集合。
下拉列表控件 (ring control)	特殊的数值控件，包括 32 位整数（从 0 开始并按顺序递增）以及一连串文本标签或图形。
显示控件 (indicator)	显示输出的前面板对象，例如图形或指示灯。
向导 (wizard)	包含多个连续页面的对话框，可以翻动页面，并在相关页面中填写相关信息。
项目 (project)	包括一些 LabVIEW 文件和非 LabVIEW 文件，可用于说明如何生成程序、为终端调配项目或进行配置设置。
项目浏览器 (Project Explorer) 窗口	创建和编辑 LabVIEW 项目的窗口。
项目库 (project library)	LabVIEW 项目库包括 VI、自定义类型、共享变量、选板目录文件等文件和项目库等。
像素 (pixel)	数码图像的最小单位。
虚拟仪器 (virtual instrument)(VI)	使用 LabVIEW 编写的程序，具有真实的物理仪器的外观和功能。
虚拟仪器软件构架 (Virtual Instrument Software Architecture)	VISA。用于控制 GPIC、VXI、RS2232 和其它类型仪器的单个接口库。
选板 (palette)	包含用于创建前面板和程序框图所需的对象和工具的面板。
选取框 (marquee)	环绕所选对象的活动虚线框。

Y

一维 (one-dimensional)	只有一个维数，如数组中仅有一行元素。
仪器驱动程序 (instrument driver)	在系统中控制仪器硬件并与之通讯的一套实用函数。
移位寄存器 (shift register)	循环结构中的可选机制，可将变量值从一次循环传递至下一次循环中。移位寄存器相当于编程语言中的静态变量。

引用句柄 (refnum)	引用编号。LabVIEW 用以指代某一对象如 VI、程序、ActiveX 或 .NET 对象的编号。引用句柄作为 VI 或函数的输入参数对对象进行操作。
应用程序 (application)	由 LabVIEW 开发系统创建、在 LabVIEW 运行系统环境中执行的应用程序。
应用程序实例 (application instance)	LabVIEW 创建的程序实例，用于 LabVIEW 项目中的各个终端。在 项目浏览器 窗口打开 VI 时，该 VI 将在该终端相应的应用程序实例中打开。LabVIEW 也会创建一个主应用程序实例，其中包括不属于项目的已打开 VI 或未从项目中打开的 VI。 也见 终端 (target)。
用户 (user)	见 操作员 (operator)。
用户定义常量 (user-defined constant)	可传递用户预设值的程序框图对象。
语法 (syntax)	特定编程语言中语句必须遵从的规则集合。
原型建模 (prototype)	以简单快速的方式实施一项任务以确定设计是否可行。原型建模通常功能不全面或者设计有漏洞。正式版本中通常不使用原型，而是重新实施一项任务。
源代码控制 (source control)	用于在确保无数据丢失的前提下实现 VI 和控制权限共享的一种解决方案。采用源代码控制软件可实现文件在多个用户中共享、加强安全保护和跟踪共享项目的修改。有时也称为源码控制。
运行按钮 （断开状态） (broken Run button)	发生错误导致 VI 无法运行时， 运行 (Run) 按钮处于断开状态。
运行模式 (run mode)	VI 正在运行或已被预设为运行状态。单击前面板工具栏的 运行 或 连续运行 按钮、程序框图中的单步按钮或选择 操作 » 切换至运行模式 可进入运行模式。当 VI 处于运行模式时，所有的前面板对象都有一个简化的快捷菜单项集合。VI 在运行时无法对其进行编辑。

Z

整数 (integer)	任何正整数、负整数和零。
帧 (frame)	平铺式或层叠式顺序结构的子程序框图。
执行过程高亮显示 (execution highlighting)	动态显示 VI 执行过程以显示 VI 中数据流的一种程序调试技术。

终端 (target)	运行 VI 的设备或机器。必须通过 LabVIEW 项目在 RT、FPGA 或 PDA 终端上进行操作。
转换 (conversion)	改变数据元素的类型。
子 VI (subVI)	在其它程序框图中使用的 VI，类似于子程序。
子程序框图 (subdiagram)	位于结构边框内的程序框图。
字符串 (string)	以文本形式来表示值。
字符串输入控件和显示控件 (string controls and indicators)	用于处理和显示文本的前面板对象。
自动调整标尺 (autoscaling)	标尺根据标绘数值的范围进行自动调整。在图形标尺中，自动标尺调整将确定刻度的最大和最小值。
自动索引 (auto-indexing)	循环结构中，在结构边框上分解和集合数组的功能。当数组进入带有自动索引功能的循环时，循环会自动分解数组，即从一维数组提取标量，而从二维数组处提取一维数组，依此类推。当数据离开循环时，循环以相反的顺序将数据集成数组。
自由标签 (free label)	前面板或程序框图中不隶属于任何对象的标签。
总数接线端 (count terminal)	一个 For 循环 (For Loop) 的接线端，其值决定了该 For 循环执行子程序框图的次数。

索引

A-D

按钮、前面板, 4-3

安装、LabVIEW, 1-2

版本、将 VI 保存为前期版本, 7-5

帮助

(也见“即时帮助窗口”), 3-4

技术支持, A-1

帮助文件

创建, 12-2

HTML, 12-2

RTF, 12-2

帮助系统

词汇表, G-1

相关文档, 1-1

保存 VI、为前期版本, 7-5

本《用户手册》行文规范, xiii

编程范例, 1-3

编程实例 (NI 资源共享), A-1

编制 VI 说明信息

创建对象说明, 12-1

创建提示框, 12-1

创建 VI 说明信息, 12-1

打印, 12-2

编制 VI 文档

帮助文件, 12-2

表格, 4-5

标签、对话框, 4-1

波形

从文件读取, 11-7

数据类型, 10-3

图表, 10-2

图形, 10-1

写入文件, 11-7

补充文档, 1-1

(也见“相关文档”), 1-1

布尔控件, 4-3

步进运行 VI, 6-3

菜单, 3-3

快捷方式, 3-3

下拉列表控件, 4-6

组合框, 4-4

菜单栏、隐藏, 4-12

参数

数据类型, 5-2

数据流, 5-9

参数列表

(见“连线板”), 7-2

层叠式顺序结构

执行, 8-11

层叠显示, 10-18

查找

错误, 6-2

选板上的控件、VI 和函数, 3-2

常量, 5-2

簇, 9-9

数组, 9-6

程序框图, 2-2

标签, 4-11

常量, 5-2

对象, 5-1

函数, 5-3

节点, 5-3

结构, 8-1

强制转换点, 5-7

删除函数上的接线端, 5-4

设计, 5-10

手动连线, 5-4

数据类型, 5-2

数据流, 5-7

输入控件和显示控件接线端, 5-1

显示接线端, 5-1

向函数添加接线端, 5-4

选项, 3-6

自动连线, 5-6

字体, 4-11

程序框图上的灰色点, 5-7

创建

- 簇, 9-9
- 电子表格文件, 11-5
- 对象说明, 12-1
- 多态 VI, 7-4
- 二进制文件, 11-6
- 前面板, 4-1
- 数据记录文件, 11-6
- 数组, 9-6
- 提示框, 12-1
- 图标, 7-2
- VI 说明, 12-1
- 文本文件, 11-5
- 选中部分程序框图创建子 VI, 7-3
- 用户定义常量, 5-2
- 子 VI, 7-1

垂直滚动条, 4-2

词汇表, G-1

磁盘空间、选项, 3-6

簇

- 常量, 9-9
- 创建, 9-9
- 错误, 6-5
- 控件, 4-5
- 连线样式, 9-8
- 元素顺序, 9-9

簇元素顺序, 9-9

错误

- 查找, 6-2
- 处理, 6-4
- 处理方法, 6-5
- 窗, 6-2
- 簇, 6-5
- 代码, 6-5
- 调试技术, 6-3
- 断开的 VI, 6-2
- I/O, 6-5
- 检查, 6-4
- 列表, 6-2
- 通过 While 循环处理, 6-6
- 显示, 6-2
- 用条件结构处理, 6-6
- 自动处理, 6-4

DAQ、传递通道名称, 4-7

打开、LabVIEW, 3-1

打印

VI 说明信息, 12-2

选项, 3-6

带状图表, 10-17

单步执行、调试 VI, 6-3

弹出菜单

(见“快捷菜单”), 3-3

单选按钮控件, 4-4

导航窗口、功能, 3-5

点、强制转换, 5-7

电子表格文件

创建, 11-5

调试

错误处理, 6-4

单步执行, 6-3

断点工具, 6-4

断开的 VI, 6-2

高亮显示执行过程, 6-3

技术, 6-3

未定义数据, 5-2

选项, 3-6

循环, 8-9

自动处理错误, 6-4

调整大小

(见“调整大小”), 4-10

前面板对象, 4-10

定时、控制, 8-4

读取、文件, 11-1

断点工具

调试 VI, 6-4

断开的 VI

常见原因, 6-2

纠正, 6-2

显示错误信息, 6-2

断线, 5-6

对话框

标签, 4-1

控件, 4-1

设计, 4-12

下拉列表控件, 4-6

显示控件, 4-1

字体, 4-11

对齐对象, 4-10

对齐网格, 4-10

对象

- 程序框图, 5-1
- 创建说明, 12-1
- 创建提示框, 12-1
- 打印说明, 12-2
- 对齐, 4-10
- 发布, 4-10
- 均匀排列, 4-10
- 可选部件, 4-9
- 前面板和程序框图接线端, 5-1
- 输入控件和显示控件的相互转换, 4-9
- 添加标签, 4-11
- 显示可选部件, 4-9
- 在程序框图上手动连线, 5-4
- 在程序框图上自动连线, 5-6
- 在前面板上调整大小, 4-10
- 在前面板上色, 4-9
- 在前面板上锁定, 4-10
- 在前面板上替换, 4-9
- 在前面板上隐藏, 4-9
- 在前面板上重叠, 4-6
- 在前面板上组合, 4-10

多态

- 创建 VI, 7-4
- VI, 7-4

多态 VI 的实例

- (也见 “多态 VI”), 7-4
- 手动选择, 7-4

E-G

Express VI 和函数、概述, 5-4

二进制、创建文件, 11-6

发布、前面板对象, 4-10

发出接线端

- (见 “输入控件”), 5-1

For 循环

- 计数接线端, 8-2
- 控制定时时间, 8-4
- 默认数据, 8-9
- 移位寄存器, 8-6
- 执行, 8-2
- 自动索引, 8-5
- 总数接线端, 8-2

发行说明, 1-2

反馈节点

- 初始化, 8-8
- 替换为移位寄存器, 8-8
- 选择, 8-8

范例, 1-3

范例 (NI 资源共享), A-1

非法数字浮点值, 5-2

非预期数据, 5-2

分格显示, 10-18

浮点数、上溢和下溢, 5-2

符号数值, 5-2

GPIOB、配置, 1-3

高亮显示执行过程、调试 VI, 6-3

格式化

- 前面板文本, 4-11

- 字符串, 9-2

- 字符串中的格式说明符, 9-3

格式字符串参数, 9-3

工具

- 入门指南, 1-3

- 选板, 3-2

工具栏, 3-4

- 项目, 3-4

工作环境选项、设置, 3-6

滚动

- 图表, 10-12

- 图形, 10-12

滚动条

- 列表框, 4-5

- 隐藏, 4-12

滚动条控件, 4-2

H-K

HTML、帮助文件, 12-2

函数, 5-3

- 浏览, 3-2

- 删除接线端, 5-4

- 搜索, 3-2

- 添加接线端, 5-4

函数选板, 3-2

- 浏览, 3-2

- 搜索, 3-2

- 自定义, 3-5

滑动杆控件, 4-2

- (也见 “数值”), 4-2

- 滑块、添加, 4-2
- 绘图、图形绘图区域, 10-15
- 混合信号图, 10-8
- I/O
 - (也见“文件 I/O”), 11-1
 - 错误, 6-5
 - 名称控件, 4-7
- Inf 浮点值, 5-2
- 即时帮助窗口, 3-4
 - 创建对象说明, 12-1
 - 创建 VI 说明信息, 12-1
- 计数接线端
 - For 循环, 8-2
 - While 循环, 8-4
- IVI、传递逻辑名称, 4-7
- 基于计算机的仪器、配置, 1-3
- 技术支持, A-1
- 加标签、常量, 5-2
- 节点, 2-3
 - 程序框图, 5-3
 - 执行数据流, 5-8
- 结构, 8-1
 - 层叠式顺序, 8-11
 - For 循环, 8-2
 - 分支, 8-9
 - 平铺式顺序, 8-11
 - 事件, 8-12
 - While 循环, 8-3
 - 位于程序框图, 2-3
- 接收接线端
 - (见“显示控件”), 5-1
- 解锁、前面板对象, 4-10
- 接线端, 2-2
 - 常量, 5-2
 - 程序框图, 5-1
 - 从函数删除, 5-4
 - 打印, 12-2
 - For 循环的计数, 8-2
 - 计数, 8-2
 - 连线, 5-4
 - 模式, 7-3
 - 强制转换点, 5-7
 - 使用自动索引设置总数值, 8-5
 - 输入控件和显示控件, 5-1
 - 添加至函数库, 5-4
 - 条件, 8-3
 - While 循环中的计数, 8-4
 - 显示, 5-1
 - 选择器, 8-9
- 经典控件, 4-1
- 警告、显示, 6-2
- 纠正
 - 带有非预期数据的 VI, 6-3
 - 断开的 VI, 6-2
 - 断线, 5-6
- 矩阵、控件, 4-5
- 均匀排列对象, 4-10
- 开关、前面板, 4-3
- 空间、为前面板或程序框图增加空间, 4-11
- 控件
 - 表格、字符串, 9-2
 - 程序框图上, 5-1
 - 簇, 4-5
 - 打印, 12-2
 - 调整大小, 4-10
 - 对话框, 4-1
 - 滚动条, 4-2
 - 滑动杆, 4-2
 - I/O 名称, 4-7
 - 接线端, 5-1
 - 矩阵, 4-5
 - 可选部件, 4-9
 - 列表框, 4-5
 - 浏览, 3-2
 - 路径, 4-4
 - 枚举, 4-6
 - 前面板上使用指南, 4-12
 - 上色, 4-9
 - 数据类型, 5-2
 - 数据类型接线端, 5-1
 - 数值, 4-2
 - 数组, 4-5
 - 搜索, 3-2
 - 锁定, 4-10
 - tab, 4-6
 - 替换, 4-9
 - 图标, 5-1
 - 下拉列表, 4-6
 - 显示可选部件, 4-9

- 选板, 3-1
- 旋转型, 4-3
- 隐藏, 4-9
- 引用句柄, 4-8
- 用户界面设计, 4-12
- 转换为显示控件, 4-9
- 字符串, 4-4
- 字符串显示类型, 9-1
- 组合, 4-10
- 控件类型、枚举, 4-6
- 控件选板, 3-1
 - 浏览, 3-2
 - 搜索, 3-2
- 控制流编程模式, 5-7
- 快捷菜单、运行模式, 3-3
- 快捷方式、自定义, 3-3
- 快速参考指南, 1-2

L-O

LabVIEW

- 安装, 1-2
- 简介, 1-1
- 卸载, 1-2
- 选项, 3-6
- 自定义, 3-6

LabVIEW 简介, 1-1

历史

- 图表, 10-17
- 选项, 3-6

连接接线端, 5-4

连线, 2-3

- 断开的, 5-6
- 对象, 5-5
- 手动, 5-5
- 选择, 5-6
- 自动, 5-6

连线板, 2-4

- 创建, 7-2
- 打印, 12-2

连续运行 VI, 6-1

量表, 4-3

- (也见 “数值”), 4-2

列表框、控件, 4-5

浏览、控件和函数选板, 3-2

流盘, 11-4

路径

- 控件, 4-4
- 文件 I/O, 11-3
- 选项, 3-6

Measurement & Automation Explorer, 1-3

moving、标签, 4-11

枚举控件, 4-6

命名、VI, 7-5

模板、VI, 7-1

默认数据

- 数组, 9-8
- 循环, 8-9

默认条件分支, 8-10

默认值、数据类型, 5-2

模式、接线端, 7-3

National Instruments 技术支持和服务, A-1

NI 技术支持和服务, A-1

内存

- 强制转换点, 5-7
- 数据流编程模式管理, 5-10

P-S

培训及认证 (NI 资源共享), A-1

配置

- 前面板, 4-9
- 前面板输入控件, 4-8
- 前面板显示控件, 4-8
- 设置外观和动作, 7-5

平铺式顺序结构、执行, 8-11

前面板, 2-1

- 标签, 4-11
- 不改变窗口大小增加空间, 4-11
- 调整对象大小, 4-10
- 对齐对象, 4-10
- 分布对象, 4-10
- 将输入控件转换为显示控件, 4-9
- 将显示控件转换为输入控件, 4-9
- 接线端, 5-1
- 均匀排列对象, 4-10
- 设计, 4-12
- 输入控件, 4-1
- 锁定对象, 4-10
- 替换对象, 4-9
- 为对象上色, 4-9
- 文本特性, 4-11

- 显示对象的可选部件, 4-9
- 显示控件, 4-1
- 选项, 3-6
- 隐藏可选部件, 4-9
- 在子面板控件中加载, 4-6
- 重叠对象, 4-6
- 字体, 4-11
- 组合对象, 4-10
- 前面板下拉菜单, 4-6
- 前面板指示灯, 4-3
- 前期版本、保存 VI, 7-5
- 强度图, 10-3
 - 选项, 10-5
 - 颜色映射, 10-5
- 强度图表, 10-3
 - 选项, 10-4
 - 颜色映射, 10-5
- 强制转换点, 5-7
- 驱动 (NI 资源共享), A-1
- 曲线
 - 层叠, 10-18
 - 分格, 10-18
- 取消组合、前面板对象, 4-10
- Repeat-Until 循环
 - (见 “While 循环”), 8-3
- 人工数据依赖关系, 5-8
- 容器, 4-6
 - 选项卡控件, 4-6
 - 子面板控件, 4-6
- 入门指南, 1-2
- 软件 (NI 资源共享), A-1
- 三维图形, 10-9
- 扫描图表, 10-17
- 删除
 - 断线, 5-6
 - 函数上的接线端, 5-4
- 上色、前面板对象, 4-9
- 上溢数字, 5-2
- 设计
 - 程序框图, 5-10
 - 对话框, 4-12
 - 前面板, 4-12
 - 用户界面, 4-12
- 设置、工作环境选项, 3-6
- 升级说明, 1-2
- 升级 VI, 7-5
- 示波器图表, 10-17
- 时间标识
 - (也见 “数值”), 4-2
 - 控件, 4-3
- 首选项
 - (见 “选项”), 3-6
- 数据记录文件
 - 创建, 11-6
 - 读取, 11-6
- 数据类型, 5-2
 - 波形, 10-3
 - 打印, 12-2
 - 默认值, 5-2
 - 输入控件和显示控件, 5-2
 - 条件选择器值, 8-10
- 数据流
 - 观察, 6-3
 - (见 “数据流”), 6-3
- 数据流编程模式, 5-7
 - 内存管理, 5-10
- 数据依赖关系, 5-8
 - 不存在, 5-9
 - 人工, 5-8
- 输入控件, 4-1
 - 布尔, 4-3
 - 低彩, 4-1
 - 高彩, 4-1
 - 经典, 4-1
 - 时间标识, 4-3
 - 新式, 4-1
- 树形控件, 4-5
- 数值
 - 符号值, 5-2
 - 格式化, 4-2
 - 控件, 4-2
- 数字、上溢和下溢, 5-2
- 数字波形数据类型, 10-8
- 数字波形图、显示数字数据, 10-6
- 数字数据、数字波形数据类型, 10-8
- 数字图形, 10-6

数组

- 创建常量, 9-6
- 创建输入控件和显示控件, 9-6
- 大小, 9-8
- 多维数组的索引, 9-3, 9-6
- 二维数组举例, 9-4
- 控件, 4-5
- 默认数据, 9-8
- 使用循环创建, 8-5
- 维度, 9-3
- 限制, 9-3
- 一维数组举例, 9-4
- 自动索引循环, 8-4

水平滚动条, 4-2

顺序结构

- 比较平铺式和层叠式, 8-11
- 过度使用, 8-11
- 控制执行顺序, 5-8

搜索

- 选板上的控件、VI 和函数, 3-2

隧道, 8-1

- 输入和输出, 8-10

锁定、前面板对象, 4-10

缩放、图形, 10-11

索引、在数组中使用, 9-3

索引循环, 8-4

- For 循环, 8-5
- While 循环, 8-5

T-W

替换、前面板对象, 4-9

提示框、创建, 12-1

添加

- 接线端至函数, 5-4
- 前面板空间, 4-11

添加标签、字体, 4-11

条件结构

- 错误处理, 6-6
- 数据类型, 8-10
- 选择器接线端, 8-9
- 选择器接线端值, 8-10
- 指定一个默认条件分支, 8-10
- 执行, 8-9

条件接线端, 8-3

通信、文件 I/O, 11-1

图标, 2-4

- 编辑, 7-2
- 创建, 7-2
- 打印, 12-2

图表, 10-1

- 标尺格式化, 10-11
- 波形, 10-2
- 层叠显示, 10-18
- 多个标尺, 10-11
- 分格显示, 10-18
- 滚动, 10-12
- 类型, 10-1
- 历史长度, 10-17
- 强度, 10-3
- 刷新模式, 10-17
- 图形工具选板, 10-12
- 选项, 10-11
- 自定义动作, 10-17
- 自定义外观, 10-12

图片、下拉列表控件, 4-6

图形, 10-1

- 标尺格式化, 10-11
- 波形, 10-1
- 多个标尺, 10-11
- 滚动, 10-12
- 混合信号, 10-8
- 类型, 10-1
- 强度, 10-3
- 三维, 10-9
- 缩放, 10-11
- XY, 10-3
- 选板, 10-12
- 选项, 10-11
- 游标, 10-13
- 在绘图区域内绘图, 10-15
- 注释数据点, 10-14
- 自定义动作, 10-12
- 自定义三维, 10-16
- 自定义数字波形, 10-16
- 自定义外观, 10-12

图形工具选板, 10-12

While 循环

- 错误处理, 6-6
- 计数接线端, 8-4
- 控制定时时间, 8-4
- 默认数据, 8-9
- 条件接线端, 8-3
- 无限, 8-4
- 移位寄存器, 8-6
- 执行, 8-3
- 自动索引, 8-5

VI, 2-1

- 编制说明信息, 12-1
- 层次结构, 7-3
- 创建说明, 12-1
- 错误处理, 6-4
- 打印, 12-2
- 调试技术, 6-3
- 断开的, 6-2
- 多态, 7-4
- 范例, 1-3
- 纠正, 6-2
- 命名, 7-5
- 模板, 7-1
- 设置外观和动作, 7-5
- 升级, 7-5
- 运行, 6-1

VI 层次结构

- 打印, 12-2
- 正在查看, 7-3

VI 层次结构窗口

- 打印, 12-2
- 显示, 7-3

VISA、传递资源名称, 4-7**网格, 4-10**

- 选项, 3-6

网络资源, A-1**未定义数据, 5-2**

- 非法数字, 5-2
- 数组, 9-8
- 无穷, 5-2

维度、数组, 9-3**文本**

- 格式化, 4-11
- 输入框, 4-4
- 下拉列表控件, 4-6

文本文件

- 创建, 11-5
- 二进制格式, 11-6
- 在多个平台上创建, 11-6

文档, 1-1

- (也见“相关文档”), 1-1
- 本《用户手册》简介, *xiii*
- 本《用户手册》行文规范, *xiii*
- 参考其它资料, 1-1
- NI 资源共享, A-1
- 指南, 1-1

温度计

- (也见“数值”), 4-2
- 滑动杆控件, 4-2

文件 I/O, 11-1

- 创建电子表格文件, 11-5
- 创建二进制文件, 11-6
- 创建数据记录文件, 11-6
- 创建文本文件, 11-5
- 电子表格文件, 11-5
- 读取波形, 11-7
- 读取数据记录文件, 11-6
- 高级文件函数, 11-3
- 格式, 11-2
- 基本操作, 11-1
- 流盘, 11-4
- 路径, 11-3
- 使用存储 VI, 11-4
- 写入波形, 11-7
- 引用句柄, 11-1
- 用于常用操作的函数, 11-2
- 用于常用操作的 VI, 11-2

文件 I/O 的格式, 11-2**问题记录, 1-2****无限 While 循环, 8-4****X-Z****x 标尺、多个, 10-11****系统、控件, 4-1****系统字体, 4-11****XY 图, 10-3****下拉列表控件, 4-6****下溢数字, 5-2**

显示

- 错误, 6-2
- 接线端, 5-1
- 警告, 6-2
- 前面板对象的可选部件, 4-9

显示控件, 4-1

- 布尔, 4-3
- 程序框图上, 5-1
- 簇, 4-5
- 打印, 12-2
- 低彩, 4-1
- 调整大小, 4-10
- 对话框, 4-1
- 高彩, 4-1
- 滚动条, 4-2
- 滑动杆, 4-2
- I/O 名称, 4-7
- 接线端, 5-1
- 经典, 4-1
- 矩阵, 4-5
- 可选部件, 4-9
- 路径, 4-4
- 前面板上使用指南, 4-12
- 上色, 4-9
- 时间标识, 4-3
- 数据类型, 5-2
- 数据类型接线端, 5-1
- 数值, 4-2
- 数组, 4-5
- 锁定, 4-10
- tab, 4-6
- 替换, 4-9
- 图标, 5-1
- 显示可选部件, 4-9
- 新式, 4-1
- 旋转型, 4-3
- 隐藏, 4-9
- 用户界面设计, 4-12
- 转换为输入控件, 4-9
- 字符串, 4-4
- 字符串显示类型, 9-1
- 组合, 4-10

向导, 1-3

相关文档, 1-1

(也见“相关文档”), 1-1

写入、文件, 11-1

卸载 LabVIEW, 1-2

性能、选项, 3-6

修订历史、打印, 12-2

修正、VI, 6-2

选板

- 工具, 3-2
- 函数, 3-2
- 浏览, 3-2
- 输入控件, 3-1
- 选项, 3-6
- 自定义, 3-5
- 自定义函数, 3-5
- 自定义控件, 3-5

旋钮, 4-3

(也见“数值”), 4-2

选项、设置, 3-6

选项卡控件, 4-6

选择、连线, 5-6

选择器接线端, 8-9

值, 8-10

旋转型控件, 4-3

循环

- 创建数组, 8-5
- For (概述), 8-2
- 控制定时时间, 8-4
- 默认数据, 8-9
- While (概述), 8-3
- 无限, 8-4
- 移位寄存器, 8-6
- 自动索引, 8-4

y 标尺、多个, 10-11

颜色, 4-9

- 低彩控件, 4-1
- 高彩控件, 4-1
- 选项, 3-6
- 映射, 10-5

液罐

(也见“数值”), 4-2

滑动杆控件, 4-2

仪表, 4-3

(也见“数值”), 4-2

疑难解答

(另见“调试”), 6-3

疑难解答 (NI 资源共享), A-1

仪器、配置, 1-3

仪器驱动 (NI 资源共享), A-1

移位寄存器, 8-6

隐藏

菜单栏, 4-12

滚动条, 4-12

前面板对象的可选部件, 4-9

引用句柄

控件, 4-8

文件 I/O, 11-1

硬件、配置, 1-3

映射、强度图和图表的颜色映射, 10-5

应用程序生成器、自述文件, 1-2

应用程序字体, 4-11

用户手册, 1-2

游标、图形, 10-13

语句

(见“节点”), 5-3

源代码

(见“程序框图”), 2-2

运行时、快捷菜单, 3-3

运行 VI, 6-1

诊断工具 (NI 资源共享), A-1

整数、上溢和下溢, 5-2

正在创建、程序框图, 5-1

支持、技术, A-1

执行

调试 VI, 6-3

高亮显示, 6-3

流, 5-7

执行流, 5-7

执行顺序, 5-7

执行速度、控制, 8-4

执行 VI、调试 VI, 6-3

指针、通过快捷菜单访问, 4-3

知识库, A-1

重叠前面板对象, 4-6

重复、代码块, 8-2

注释

(也见“添加标签”), 4-11

使用, 10-14

转盘, 4-3

(也见“数值”), 4-2

子程序

(也见“子 VI”), 7-1

自带标签, 4-11

自定义

工作环境, 3-5

设置外观和动作, 7-5

选板, 3-5

自动连线, 5-6

自动索引

For 循环, 8-5

默认数据, 8-9

While 循环, 8-5

字符、格式化, 4-11

字符串, 9-1

表格, 9-2

格式化, 9-2

格式说明符, 9-3

控件, 4-4

通过编程编辑, 9-2

显示控件, 4-4

显示类型, 9-1

组合框, 4-4

子面板控件, 4-6

字体

对话框, 4-11

设置, 4-11

系统, 4-11

选项, 3-6

应用程序, 4-11

子 VI, 7-1

层次结构, 7-3

创建, 7-1

多态 VI, 7-4

选中部分程序框图, 7-3

总数接线端, 8-2

使用自动索引设置, 8-5

组合

簇数据, 9-8

前面板对象, 4-10

数组中的数据, 9-3

字符串中的数据, 9-1

组合框, 4-4

《用户手册》, 见文档。