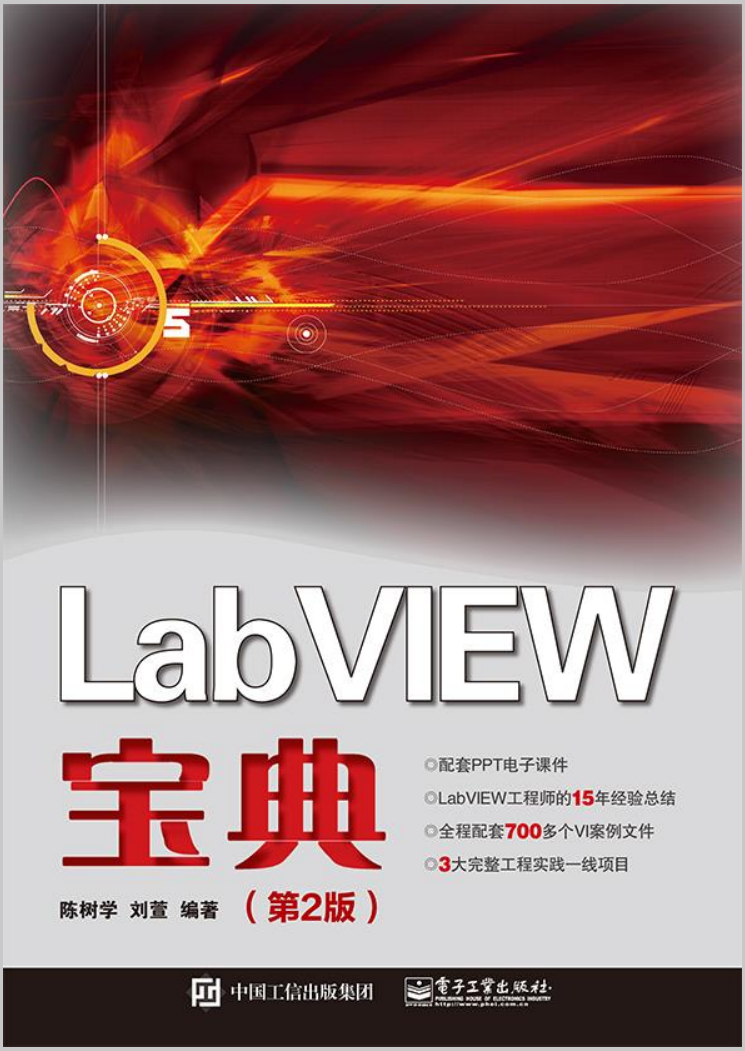

LabVIEW 宝典

- 配套PPT电子课件
- LabVIEW工程师的15年经验总结
- 全程配套700多个VI案例文件
- 3大完整工程实践一线项目

陈树学 刘萱 编著

（第2版）

中国工信出版集团 電子工業出版社

资源下载

前　言

计算机的出现，彻底改变了人们的工作和生活方式。如今计算机已经无处不在，进入了每个人的生活之中。在工程技术人员看来，计算机不仅仅是人们常见的 PC，也包括各种微处理器。从这个角度看，我们无时无刻不在使用计算机，例如，电视、洗衣机、自动提款机等都依赖计算机来实现各种便捷的功能。

相同的计算机可以完成不同的工作，在于它们使用不同的程序，而程序是由计算机编程语言创建的。短短几十年中，出现了众多的编程语言，这些编程语言的共同特点是采用文本方式创建程序。文本方式编程对编程人员要求很高，这使得计算机编程只能是少数人才可以从事的职业。

美国国家仪器公司（National Instruments， NI）的创新软件产品 LabVIEW，允许用图形方式编程，摒弃了晦涩难懂的文本代码，使得计算机编程不再是少数人的专利。LabVIEW 的最早版本诞生于 1986 年，几乎和 Windows 的最早版本同步，这注定了 LabVIEW 是多平台的编程语言，适用于不同的操作系统。

20 世纪 80 年代初，NI 公司是 GPIB 总线设备的主要供货商，丰富的硬件经验和强大的软件开发需求，促使 NI 公司的工程师们决心寻找一种代替传统编程语言的开发工具，这导致了 1986 年 LabVIEW 的横空出世。LabVIEW 是由测试工程师开发的专用编程语言，因此，LabVIEW 具有鲜明的行业特点，最早主要用于测试测量领域。NI 公司独创了虚拟仪器的理念，提出了“软件就是仪器”的理念，并逐步成为业界的标准。

随着 LabVIEW 的不断发展，几乎每隔一两年，都要推出新的版本。LabVIEW 的应用范围已经覆盖了工业自动化、测试测量、嵌入式应用、运动控制、图像处理、计算机仿真、FPGA 等众多领域。以 LabVIEW 为核心，采用不同的专用工具包和统一的图形编程方式，可以实现不同技术领域的需求。

由于 LabVIEW 版本升级过快，导致许多函数、VI 的名称与图标发生了变化。使用 LabVIEW 新版本的读者，会发现本书第 1 版中程序框图中的函数、VI 与新版 LabVIEW 对应不上，而且 LabVIEW 每次更新都增加了很多新功能。鉴于此，我们编写了本书的第 2 版。第 2 版在 2016 新版 LabVIEW 的基础上，重新编写了绝大部分程序框图，同时也针对新功能，增加了对应的内容。尤其是第 10 章，重写了大部分内容，增加了许多流行的框架结构的介绍。除了 LabVIEW 本身的框架外，也介绍了几种流行的第三方常用框架结构。

本书要点

本书作者多年使用 LabVIEW 作为主要的编程语言，因此特别关注 LabVIEW 在工业领域的具体运用以及 LabVIEW 的实用编程技术。本书共 17 章，分为入门篇、高级篇、工程应用篇。

第 1~5 章为入门篇，介绍了 LabVIEW 的基本概念、基本函数的用法和常用的运行结构，详尽地分析了 LabVIEW 的基本数据结构和文件存储方式。

第 6~11 章为高级篇，介绍了应用程序、VI 和控件的引用、属性和方法，以及各类高级控件的运用方法。第 8 章介绍了 LabVIEW 的文本编程方式及 DLL、C 语言接口，第 9 章详细介绍了基于 MATLAB 语法的 MathScript 编程技术，第 10 章介绍了 LabVIEW 基于组件的编程方法。

第 12~17 章为工程应用篇，具体而细致地讲解了在做实际工程开发时所用到的 LabVIEW 编程技术。本篇结合 LabVIEW 的常用工具包，具体分析了计算机串口、并口、网络通信方面的编程技术，以及数据记录和监控工具包、数据库连接工具包、报表生成工具包、状态图工具包、FPGA 工具包等的应用。第 13、第 14 章详细介绍了数据采集的基本原理和常用编程方法，其中重点介绍了 LabVIEW 在实时系统下的运用。第 15 章讲解了 LabVIEW 实时系统的开发案例。第 16 章讲解了 LabVIEW 数据采集系统开发案例。第 17 章讲解了 FPGA 的开发案例，以及如何利用 LabVIEW 图形编程方式，提高开发效率。

本书读者

本书可作为高等院校通信、测量技术、自动控制等相关课程的教材和教学参考书，也可作为相关工程技术人员设计开发仪器或自动测试系统的技术手册。

本书特色

本书内容非常丰富，在每个章节都安排了大量的示例，针对具体编程实践中遇到的问题，提出了多种解决方法。在兼顾基础知识的前提下，深入讨论了 LabVIEW 的高级编程方法和编程技巧。

本书的宗旨是作为实用工具书，侧重于来自工程实践的一线案例。笔者在写作过程中，与众多的 LabVIEW 爱好者进行了充分的沟通与交流，总结了编程过程中经常遇到的问题，并作为本书的重要内容。

本书使用了大量篇幅讲解 NI 公司各种专用工具包的运用，这部分内容具有一定的深度和实用价值，特别适合具有一定基础的编程人员学习运用。在入门篇中，虽然也介绍了许多 LabVIEW 的基础知识，但还是侧重于具体应用，其中大量的例程可以直接在具体项目中使用。

在高级篇中，重点介绍了 LabVIEW 常用的编程模式，以及状态图工具包的运用，同时介绍了新增的面向对象的编程方法。

补遗说明

本书写作时主要使用 8.6 版本，但是书中介绍的具体内容并不限于特定的版本，因此无版本限制。本书案例文件和相关课件在网址 http://read.zhiliaobang.com/pages/article/66 可以下载，方便读者提高学习效率，也方便教师教学。此外，对于比较重要的内容，为了让读者印象深刻，我们以"学习笔记"的体例呈现出来。

致谢与分工

本书由陈树学、刘萱两位工程师编写，我们有多年的 LabVIEW 实际开发经验，经过浓缩和总结才成此书。在编写本书的过程中得到张国强老师的大力支持，他为我们提供了开发硬件，使得写作能在真实的开发环境中进行，应该说没有他的热心帮助完成本书是难以想象的事情。本书也离不开成都道然科技有限责任公司的专业策划支持。因为本书作者为工程技术人员，对于写作并不擅长，书中错漏之处在所难免，敬请批评指正。能够为 LabVIEW 在国内的推广使用做一点力所能及的贡献，能够对广大的 LabVIEW 爱好者有所帮助，是我们最大的愿望。

目　录

第 1 部分　入门篇

第 3 部分 工程应用篇

第 1 章　打开 LabVIEW 编程之门

LabVIEW 在国内流行的时间并不长，但实际上它已经诞生 20 多年了，在国外被广泛用于教学、科研、测试和工业自动化领域。自 LabVIEW 6.1 后，LabVIEW 开始流传开来，越来越多的编程人员开始使用 LabVIEW 并把它作为首选的编程语言。LabVIEW 与常规编程语言有很大的不同，可以说它是专门为工程师开发设计的语言，专业性很强。对于从事工程应用的工程师们来说，LabVIEW 是必须掌握的编程语言。

由于 LabVIEW 的特殊性，这里对于开始学习 LabVIEW 的朋友们，提出如下建议：

- 要学会“背叛”。LabVIEW 有自己独特的编程方式，要学会 LabVIEW 的思维逻辑。
- 不要相信两三个小时就能学会 LabVIEW 之类的话，即便是一年也只是入门而已。
- 任何时候要牢记“数据流”的概念，这是 LabVIEW 编程的核心。
- LabVIEW 直接面向工程应用，因此“标准”是最重要的。
- LabVIEW 是工程师的语言，编程者首先必须自己是一位优秀的工程师。
- 学习 LabVIEW 最好的资料就是 LabVIEW 的例子程序。

学习笔记　近几年来，Lab VIEW 每年更新一个版本，新增了很多功能，同时原有功能也发生了很多变化，本书第 2 版是基于目前最新的 2016 版。

1.1　从 VI 开始

LabVIEW 同其他编程语言和软件一样，安装程序界面友好，容易使用。将安装光盘插入到光驱后，自动启动 LabVIEW 安装。只要输入正确的序列号，所有的安装过程都能自动完成。安装结束后，重新启动计算机，然后用鼠标双击 LabVIEW 的快捷方式图标，即可启动 LabVIEW。

常规编程语言，如 VB、VC 的 IDE 开发环境，都是从新建一个具体的项目开始的，每个函数必须在项目里被调用。而 LabVIEW 中的 VI 类似于常规编程语言中的函数，是可以独立于项目运行调试的，非常容易使用。对于初学者，可以从创建 VI 开始，然后逐步熟悉，直到掌握 Lab VIEW。LabVIEW 在创建复杂应用程序时，需要使用项目，项目的具体使用方法将在后续章节介绍。LabVIEW 启动窗口如图 1-1 所示，在菜单中选择“文件”→“新建 VI”项即可创建一个 VI。

新建一个 VI 后，呈现在我们面前的是两个常见的 Windows 窗口，分别为前面板窗口与程序框图窗口，如图 1-2 所示。我们在后面的讲述中将这两个窗口简称为前面板和程序框图。

学习笔记　VI 由前面板和程序框图组成。

一般常规编程语言创建的程序，由一个图形界面窗口（一般称为 GUI）和文本编辑窗口组成。LabVIEW 中的 VI，前面板相当于 GUI，程序框图则相当于文本编辑器。

显然，前面板是需要放置各种控件的，而程序框图是用来编写代码的。LabVIEW 最大的特点是，它是图形式编程语言，也就是说它的代码是完全图形化的，和常规的文本式编程语言截然不同。

通过菜单栏的“工具”菜单，可以调出“控件”选板和“函数”选板，如图 1-3 所示。其中“控件”选板用于在前面板放置控件，“函数”选板用于在程序框图中放置函数（即代码）。

图 1-1 LabVIEW 启动窗口

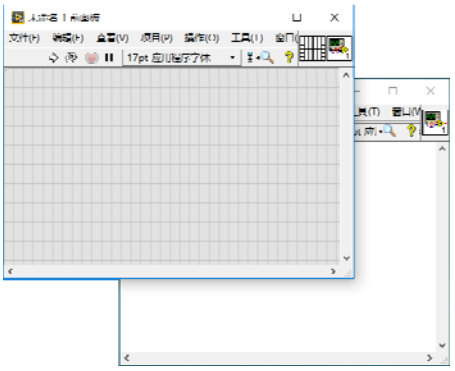

图 1-2 前面板窗口和程序框图窗口

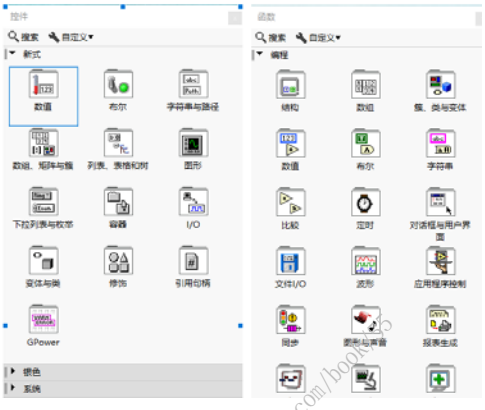

图 1-3 “控件”选板和“函数”选板

LabVIEW 给人的第一感觉，是控件的数量和种类远胜于其他编程语言。例如，在 VC 中要找到一个不同于 Windows 标准的控件是很困难的，有时候不得不采用“自画”的方式实现。另外一个明显的不同是，LabVIEW 的控件分成输入控件和输出控件两部分，输出控件又称为显示控件。

控件选板和函数选板的使用非常频繁，而使用菜单来调用它们非常不方便。最简单的调用方法是：右击前面板，弹出控件选板；右击程序框图，弹出函数选板；然后按快捷键 Ctrl+E，即可快速在前面板和程序框图之间切换。

1.1.1 创建 VI

常规语言的入门程序一般是经典的“Hello World！”，即在显示窗口放置一个显示控件，一般是文本框，然后给这个文本框赋值。这里，我们也从“Hello World”VI 的创建开始。要输出字符串“Hello World”，首先需要在前面板放置字符串显示控件。通过控件选板，选择字符串显示控件。此时出现一个带虚框的控件，将其移动到前面板合适的位置。单击前面板，字符串显示控件就会自动放置在前面板中。

在前面板放置显示控件后，在程序框图中自动出现对应的接线端子，如图 1-4 所示。接线端子是 LabVIEW 特有的概念，它与前面板控件一一对应。

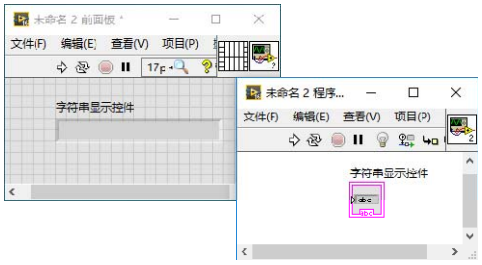

图 1-4 放置字符串显示控件

学习笔记 双击前面板中的控件或者程序框图中的接线端子，可以自动定位到对应的接线端子或控件。在快捷菜单中，通过查找控件或者接线端子，也可以实现同样目的。

现在遇到的问题是如何给这个显示控件赋值。记住，数据流是 LabVIEW 编程的核心。作为字符串显示控件，它是数据要流动到达的目标。因此，必须有一个数据流出的源。我们自然想到，字符串输入控件就是数据源。用同样的方法，在前面板放置一个字符串输入控件。接下来我们需要考虑如何在输入控件和输出控件之间建立联系。

在 LabVIEW 中创建程序框图的过程就相当于用常规语言编写代码的过程；输入控件接线端子和显示控件接线端子之间连线的过程，就相当于用常规语言编写语句的过程。

前面板和程序框图的操作都离不开工具选板，所以在连线之前首先要熟悉工具选板，如图 1-5 所示。如果工具选板未显示，通过菜单栏中的“工具”菜单可以调出工具选板。

在工具选板中，当鼠标箭头移动到工具按钮上时，会出现工具条提示。工具选板上各个按钮的名称和详细功能，如表 1-1 所示。

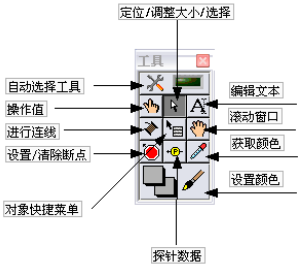

图 1-5 工具选板

表 1-1 工具条提示

图 标	名 称	功 能
	自动选择工具	选中后，根据鼠标位置自动确定工具，按Shift+Tab组合键或单击此按钮可以禁止或者启动自动选择工具
	操作值	改变控件值。对于数值型控件，可以直接控制增减量；对于字符串，可以直接输入或者更改字符串
	定位/调整大小/选择	处于箭头状态时，通过双击控件或者接线端子，可以定位到接线端子或者控件，选中对象后，可以拖动改变其大小，还可以通过矩形框选择一个或者多个对象
	编辑文本	编辑对象标签、标题或自由标签，也可以用来改变数值型控件的值
	进行连线	仅用于程序框图，用于对象之间的连线
	对象快捷菜单	与用鼠标右键选取快捷菜单的功能相同，主要用于同时修改多个控件的属性
	滚动窗口	Windows常规操作，用于平移滚动窗口
	设置/清除断点	在VI、函数、节点、连线和结构上设置或清除断点，使程序在断点处暂停
	探针数据	在连线上设置探针，可以观察流动的瞬时数据，主要用于调试
	获取颜色	取得当前窗口任意位置的颜色
	设置颜色	设置对象元素的颜色，可以和获取颜色工具配合使用

通过连线工具创建的“Hello World”VI，如图 1-6 所示。

LabVIEW 中的 VI 类似一个函数，但是与 C 语言中的函数有明显区别。用常规编程语言编写的程序都有一个明显的入口点，比如 main()函数。VI 则不同，任何一个 VI 都是可以单独运行的，不存在明显的入口点。用常规编程语言编写代码后，需要明显的编译、连接过程，VI 则不存在明显的编译过程，在我们对 VI 程序框图连线时，编译过程在后台自动发生，编译过程是动态的。

单击工具栏中的“运行”按钮，运行 VI，输入到字符串控件的值将自动显示到字符串输出控件

中。

学习笔记 输出控件经过连线，把它的值传递给显示控件。

工具栏中还提供了“连续运行”和“中止”按钮，如图 1-6 所示。这里要介绍一下“运行”和“连续运行”的区别：“运行”是程序运行一次就结束了，而“连续运行”是指 VI 连续不断循环运行。“连续运行”可以修改输入控件的值，而且显示控件能动态显示出修改的结果。当 VI 连续运行时，“连续运行”按钮上的字体变为黑色并加粗，再次单击“连续运行”按钮或者工具栏中的第 3 个按钮“终止运行”来结束连续运行。

下面在图 1-6 所示 VI 的基础上，创建计算一次函数 $y=kx+b$ 的 VI，y 作为计算结果输出，k、x、b 作为参数输入。

如图 1-7 所示，数值控件使用的是双精度数据类型，它的连线颜色和线型与字符串明显不同。从连线的颜色和线型，可以明显区分数据类型，这是 LabVIEW 图形编程方式的突出特点。

学习笔记 不同颜色、不同的线型表示不同的数据类型。

当程序框图中出现未连线或连线错误时，工具条上的“运行”按钮就变成“错误”按钮。单击它会自动弹出错误列表对话框，提示出现的错误，而常规编程语言在编译的过程中提示错误，这说明 LabVIEW 的编译过程是在后台自动完成的。

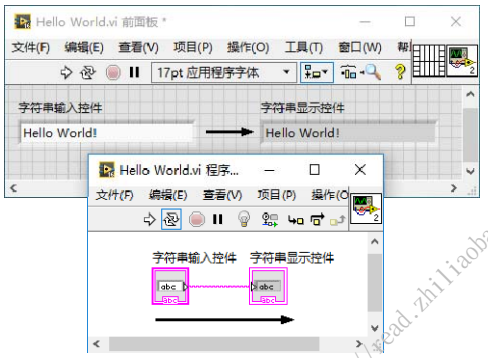

图 1-6 “Hello World”VI

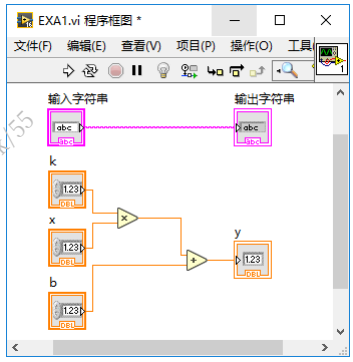

图 1-7 一次函数

学习笔记 LabVIEW 程序的编译是在连线的过程中自动进行的。

LabVIEW 被称作“G”语言。G 指的是 Graphical Programming Language，即图形化编程语言。对照其他文本方式的编程语言，G 语言中键盘的作用似乎不重要了，因为在程序框图中既看不到变量，也看不出任何语句的存在。我们完全可以理解前面板中控件的含义，这和其他编程语言一样。GUI 是由各种各样的控件集合而成的，但是它的控件有独特之处：一是数量多，二是明确区分输入控件和输出控件。

“数据流”是 LabVIEW 的核心，也是 G 语言的核心。输入控件和显示控件中间的连线就表明了数据由输入控件流动到显示控件。输入控件就是数据的“来源”，显示控件就是数据要流动到的“目的地”，而这个流动的过程是通过连线完成的。与日常的物理现象中的“流动”不同的是，流动后，输入控件的数据并没有“损耗”，依然存在，而显示控件的数据被“冲掉”了，变成了新的数据。

既然是数据流，那么一点的数据应该可以流向多点，多点的数据也应该可以汇集成一点。实际上，上面的程序已经实现多点汇集到一点了，一次函数 $y=kx+b$，k、x、b 都是输入控件，是数据源，而目的地是 y，这本身就是多个输入控件的数据汇集到一个显示控件的例子。

1.1.2 控件属性设置与快捷菜单

通过前面的学习，我们初步了解了 LabVIEW 程序是由 VI 组成的，而 VI 又是由输入控件、显示控件和数据连线组成。因此，深入了解和探讨控件是非常必要的。LabVIEW 的控件种类繁多，即使是同一类型控件，在一些细节上也有很大差别。

VB、VC 也提供了大量的控件，如命令按钮、文本框、列表框和组合框等。LabVIEW 提供的控件与众不同。首先，控件分成输入控件和显示控件，另外，控件的分组也很有特点，有数值控件、布尔控件、字符串和路径组控件、数组和矩阵组控件、列表与表格组控件、图形组控件等。这不像普通的控件分组，更像是变量的数据类型分组。

1．控件的基本属性

一般的控件具有属性、方法和事件，LabVIEW 的控件与常规控件类似，也具有属性、方法和事件。一般的控件都包括“值”属性，表示控件当前代表的数值或字符串等，也就是说控件是数据的容器，而数据的值只是控件属性之一。

LabVIEW 中不存在常规语言中变量的说法，任何数据都是依附于控件的。控件是数据的容器，数据不能离开控件而独立存在（移位寄存器和常量例外）。

LabVIEW 的控件中包含数据，但是数据是有类型区别的，比如数字可以是整型，而整型又可以分成有符号和无符号，8 位、16 位、32 位等。选定数据类型后，控件与数据类型就存在了对应的关系，不允许动态更改。

控件作为对象，由多个组成要素构成，比如标签、标题、颜色、字体等。对于一个具体的控件，通过快捷菜单或属性对话框，可以修改其属性。不同种类的控件专用属性可能不同，但是其常规属性基本相同。这里以字符串控件的属性对话框为例，介绍常见的基本属性。在控件的快捷菜单中选择“属性”，弹出属性对话框，字符串属性对话框如图 1-8 所示。

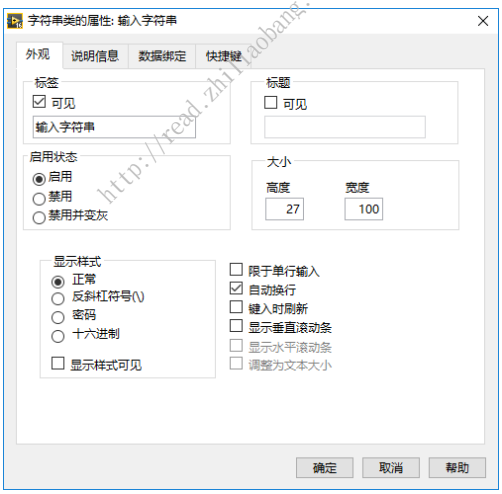

图 1-8　字符串属性对话框

字符串属性对话框采用典型的选项卡方式，其中包括“外观”、“说明信息”、“数据绑定”、“快捷键”4 个类别。按照常规的表示方法，位于上方的属性一般是通用属性，而位于下方的往往是控件的专用属性。

（1）“外观”选项卡上的属性用于控制控件显示的有关特征，控件的通用外观属性包括以下几点：

- “标签”和“标题”，标签代表的是控件的名称，它在运行过程中属于只读属性，不能在运行过程中更改，相当于常规语言中的变量名。标题是控件显示给用户的信息，属于可读写属性，在运行过程中可以随时更改，一般多语言环境的软件都采用类似的方法。
- “大小”，包括“高度”和“宽度”，表示控件的大小信息。通过高度和宽度，可以精确控制控件的大小。
- “启用状态”，包括“启用”、“禁用”和“禁用并变灰”3 个单选按钮。它们用于表示控件的状态信息，处于禁用状态的控件不接受键盘和鼠标操作。

学习笔记 控件的标签是内部名称，用于区别控件，而标题是用于人机交互的。

（2）属性对话框的第二个选项卡为“说明信息”，由“说明”和“提示框”两部分组成。在“说明”和“提示框”中输入的是文本信息。

- 在提示框内输入的内容就是通常所说的“工具条提示”，当移动鼠标箭头到控件上时，不执行任何操作，几秒后即出现“提示框”里的说明文字。
- “说明”是控件的详细信息，可以对控件做详细解释。程序运行时，用鼠标选择控件，然后在右键快捷菜单中选择“说明”，就会弹出一个对话框，显示的就是我们输入的“说明”信息。

学习笔记 在 LabVIEW 中文本输入的手动换行是通过快捷键 Shift+Enter 实现的。

（3）其他选项卡，比如数据绑定，将在后续章节介绍。

2．输入控件和显示控件的区别

LabVIEW 控件明显分成了输入控件和显示控件两大类别。LabVIEW 作为一种用于工程的编程语言，是与测试测量和自动化控制密切相关的。从硬件的角度看，控件分成这两大类非常合适。以自动化控制为例，它的输入和输出是有明显区别的。比如继电器是输出类型的，而检测开关则是输入类型的。广泛使用的数据采集卡也分成模拟量输入和模拟量输出、数字量输入和数字量输出。

从 LabVIEW 本身的编程特点来说，数据流是 LabVIEW 的核心概念，任何数据必须是有“源”的，这个“源”就是输入控件或者常量，常量可以理解成特殊的输入控件，而数据最终流向地就是显示控件。

从控件本身的角度来看， LabVIEW 其实只是推荐了控件的使用方法，并没有绝对地区分输入控件和显示控件。比如数值控件中有“旋钮”和“量表”，旋钮是输入控件，而量表是显示控件。这与现实情况是非常符合的。对于收音机的音量控制，确实有个物理的旋钮存在，比较高级的收音机也确实有个简单的电子仪表来显示音量。

LabVIEW 的输入控件和显示控件是可以自由转换的。在控件的快捷菜单中就可以实现将输入控件转换成显示控件，或者将显示控件转换成输入控件。但这只是在程序编辑情况下才可以，在程序运行情况下是不允许转换的。如果允许的话，数据流向就会发生变化，这将导致数据流无法实现。

3．控件快捷菜单

LabVIEW 对控件的许多操作都是通过快捷菜单实现的，所以有必要深入研究一下快捷菜单。对于不同的控件，快捷菜单既有相同的部分，也有不同的部分。相同的部分适合所有的控件，而不同的部分对应着控件的特殊属性。以一个旋钮控件为例看一下控件的快捷菜单，如图 1-9 所示。

显示项中包括“标签”和“标题”，这是控件的通用基本属性。通过这个菜单可以选择控件标签及标题是否可见，前面我们通过属性对话框同样可以设置这两个属性。如果只想设置控件的部分

属性，直接用快捷菜单比较方便。

图 1-9　旋钮控件的快捷菜单

显示项中还有“单位标签”、“数字显示”和“梯度”三个选择项，图 1-10 演示了它的不同效果。

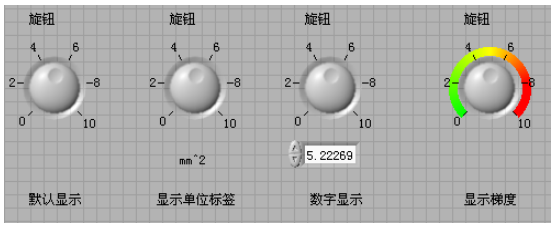

图 1-10　旋钮控件的不同显示效果

通过上面的效果图我们可以看到，一个 LabVIEW 控件由不同的可显示部件组成，比如上面的旋钮，就是由单位标签、数字显示和梯度等几个部件组合而成，而其中每一个独立的部件都可以分别设置它的属性，比如颜色、字体等。

1.1.3　创建控件、常量、局部变量、引用、属性节点和方法节点

创建控件、常量、局部变量、引用、属性节点和方法节点，是 LabVIEW 中极为常见的操作。创建方法很多，分类介绍如下。

1．创建控件

LabVIEW 的控件数量庞大，种类繁多，这极大地方便了 GUI 设计。控件从基本显示风格来区分，可分成银色、现代、古典与“系统”4 大类。

- 银色控件是 LabVIEW 提供的最新型控件，推荐使用银色控件。
- 现代控件是三维显示风格，银色控件推出之前，多使用此类控件。
- 古典控件是平面显示风格，早期 LabVIEW 使用此类控件。
- 系统显示风格，适用于编写 Windows 风格的应用程序。

任何一种编程语言都具备处理复杂数据类型的能力，LabVIEW 作为成熟的编程语言，自然也不例外。在各种编程语言中最常见的复杂数据类型为数组和记录（VB）或结构（C 语言），LabVIEW 中类似记录或结构的数据类型，称为“簇”。这只是一个称谓的区别，本质并无区别。

在控件选板中包含数组和簇控件，但它们是作为控件形式出现的，因此，它们的创建方式与数值控件相同。在控件选板选择数组和簇之后，前面板会出现数组和簇的容器，之后我们需要在数组或者簇中添加元素。

对于数组，数组中的元素具有相同的属性，因此只需选择需要的元素控件类型。建立数组容

器后，右击容器内空白处，弹出控件选板，选择合适的控件。也可以在前面板放置控件，然后拖动到数组容器中。新版本中可以通过简单控件的快捷菜单，把简单控件直接转化为数组控件，这样更为方便。簇的建立方法与数组类似，不过需要选择多种类型的控件。

控件隶属于特定的前面板，前面板往往包括多种不同类型的控件，创建控件有多种方法。

- 在控件选板中选择相应的控件，然后拖动到前面板，程序框图中就会出现相应的接线端子。
- 在程序框图函数接线端子上打开快捷菜单，选择"创建"→"输入控件"或"创建"→"显示控件"项。
- 通过剪贴板，复制一个或者多个控件。同常规的 Windows 程序一样，LabVIEW 提供了剪贴板功能，可以通过"编辑"菜单来完成，不过更方便的方法是使用快捷键。使用快捷键 Ctrl+C，可以复制控件或控件集合。使用快捷键 Ctrl+X，可以剪切控件或控件集合。使用快捷键 Ctrl+V，可以粘贴控件或控件集合。
- 用鼠标左键选择控件或控件集合，按下 Ctrl 键后拖动鼠标，可以直接生成新的控件或控件集合，这种方法称为克隆。
- 在程序框图中，选择接线端子或者接线端子组，然后按下 Ctrl 键后拖动鼠标，可以直接在前面板上创建新的控件。
- 从已经打开的别的 VI 前面板上，直接拖动控件或者控件集合到前面板。
- 从已经打开的别的 VI 程序框图中，直接拖动接线端子或者接线端子组，可以包括连线，这样可以直接复制程序，并自动创建相应控件。

2．创建常量

常量隶属于程序框图，在程序框图中创建常量有以下几种方法。

- 在函数面板上找到相应的函数，比如数值、字符串、文件路径、数组或簇，其中都包括常量。
- 在接线端子的右键快捷菜单中，选择"创建"→"创建常量"项。
- 拖动前面板上的控件或控件集合到程序框图，直接创建与控件类型对应的常量。
- 在类似的常量存在的情况下，通过剪贴板复制、剪切和粘贴来创建常量。
- 在类似的常量存在的情况下，按下 Ctrl 键后用鼠标拖动该常量，通过克隆的方式创建新的常量。
- 在已经打开的其他 VI 中，直接拖动控件或者常量到程序框图。
- 在局部变量的右键快捷菜单中，选择"创建"→"创建常量"项。
- 在属性节点的右键快捷菜单中，选择"创建"→"创建常量"项。

其中有些常量（比如数值常量）可以通过快捷菜单进一步选择其相应的数据类型（比如 U8、I16、DBL 等）。

3．创建局部变量

局部变量的具体含义将在后续章节介绍，创建局部变量的几种常用方法如下。

- 最基本的方法是在控件或者接线端子的右键快捷菜单上，选择"创建"→"局部变量"项。
- 对于已经存在的局部变量，通过剪贴板复制局部变量是不可行的，需要特别注意。LabVIEW 虽然支持局部变量的复制、剪切和粘贴操作，但是粘贴后会创建一个新的控件及其局部变量。

◆ 对于已经存在的局部变量，可以按下 Ctrl 键后用鼠标拖动局部变量来克隆。

4．创建属性节点

属性节点具体内容将在后续章节介绍。创建属性节点有如下方法。

◆ 最基本的方法是使用前面板控件或程序框图中的接线端子的右键快捷菜单，选择“创建”→“属性节点”项。

◆ 对于已经存在的属性节点，通过剪贴板复制、粘贴的并非原来控件的属性节点，而是和原来数据类型对应的通用属性节点，暂时未指向任何控件。

◆ 最简单的方法是按下 Ctrl 键后用鼠标拖动属性节点，创建一个新的属性节点。

学习笔记 选择现有的属性节点，通过复制粘贴可以创建严格类型的通用属性节点。按下 Ctrl 键后用鼠标拖动该节点即可对其克隆，克隆操作不产生新的控件。

5．创建控件的引用

创建控件的引用有如下方法。

◆ 最基本的方法是在控件或者接线端子的右键快捷菜单上，选择“创建”→“引用”项。

◆ 对于已经存在的控件的引用，按下 Ctrl 键的同时用鼠标拖动控件引用来克隆。

◆ 对于已经存在的引用，通过剪贴板复制、粘贴的并非原来控件的引用，而是和原来控件类型一致的新的控件和控件的引用。

6．创建控件的方法（中文版的帮助文件中称作“调用节点”，早期文档中译为“方法”）

LabVIEW 的控件与其他编程语言的控件一样具有属性和方法。创建控件的方法有以下几种方式。

◆ 最基本的方法是在控件或者接线端子的右键快捷菜单中，选择“创建”→“方法”项。

◆ 对于已经存在的控件方法，按下 Ctrl 键后拖动该方法来克隆。

◆ 对于已经存在的控件方法，通过剪贴板复制、粘贴的并非原来控件的方法，而是和原来控件类型一致的新的通用控件方法。

学习笔记 选择现有的方法，通过复制、粘贴可以创建严格类型的通用方法。

以上介绍了控件、常量、局部变量、属性节点、引用和方法的创建方法，其中都包括了按下 Ctrl 键拖动鼠标的方法。这种方式称作“克隆”，克隆与 Windows 的复制、粘贴是不同的。Windows 里的复制和粘贴往往创建新的控件，而克隆操作往往创建的是同一控件的局部变量或者属性节点等。

可以通过子 VI 的接线端子的快捷菜单，选择“创建”，然后选择“输入控件”、“显示控件”或者“常量”，来创建所需的输入控件、显示控件。利用这种方法，可以保证控件和子 VI 端子需要的类型一致，这是一个非常重要的功能。

学习笔记 选择子 VI 的端子，通过快捷菜单选择“创建”项，来创建控件和常量。

1.1.4 创建自定义控件

在选择控件之后，可以通过快捷菜单中的“表示法”项或者属性对话框来选择控件所代表的数据类型，一般都是一类相近的数据类型。比如数值类型中，U8、I8、U16 和 I16 都是相近的数据类型。这样，在设计过程中想更改数据类型是很容易的。如果设计的数据类型和实际需要的数据类型有很大区别，这时可以直接选择快捷菜单中的“替换”项，把原来的控件替换成想要的新形式。

在构造一个比较复杂的程序时，通常要定义一个复杂的数据结构来描述外部现象，而这个复杂的数据结构有可能贯穿整个程序的始终。一旦这个数据结构发生变化，将导致程序多处发生变化，这会给程序设计者造成极大困难，甚至导致整个程序设计的失败。

最好的方法是定义一个统一的复杂数据类型（通常是用构造簇或者类的方法来实现，让簇或者类作为一种统一的数据类型，贯穿程序设计的始终）。通过这种方法，对数据结构的修改就能引起程序中所有引用簇的地方自动更新。LabVIEW 通过把簇控件作为自定义控件，实现统一的数据类型并贯彻始终。

一个控件包括外观、数据类型、控件默认值等内容，可以自定义的是控件的哪些部分呢？如果自定义的仅仅是控件的外观，那意义不大，无非是为控件新增了特殊的显示效果。LabVIEW 在这方面做得已经足够多了，我们很容易找到所需的控件。

选择一个控件，打开快捷菜单并选择“自定义”项，弹出自定义控件编辑器，看看到底能自定义哪些部分。如图 1-11 所示，这是一个普通的数值控件自定义前面板，它与普通的前面板是有区别的。一般 VI 的前面板对应一个程序框图，而自定义控件前面板没有对应程序框图，不允许编程。

此外还可以看出，一个控件是由一些基本对象元素组合而成的，每一个元素都可以被独立修改，如颜色、大小等。重新定义这些基本元素，就可以构造出新的符合自己特殊要求的控件。

自定义控件有三种形式：输入控件、自定义类型和严格自定义类型，如图 1-11 所示。自定义控件的三种不同形式存在重要区别。图 1-12 分别调用了创建后的输入控件、自定义类型、严格自定义类型的属性对话框，我们可以发现，它们的属性对话框存在明显区别。

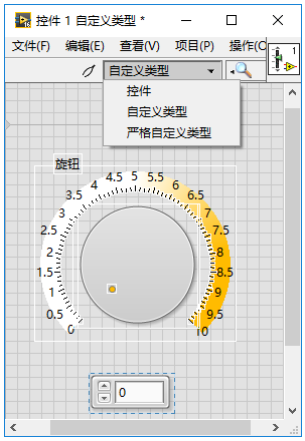

图 1-11 自定义控件编辑器

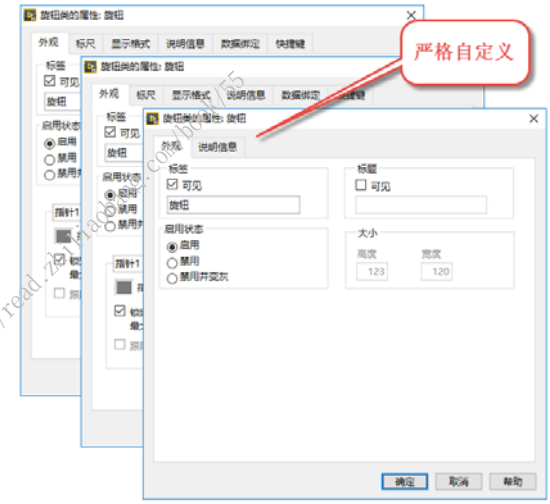

图 1-12 输入控件、自定义类型、严格自定义类型属性对话框

输入控件、自定义类型、严格自定义类型的区别如下所述。

- 输入控件保存在一个单独文件中，对于保存为输入控件的自定义控件，一旦在一个 VI 中调用它，则这个新生成的控件与原来的控件没有任何关系，可以自由地修改这个新控件的属性，如图 1-12 中左侧图所示。
- 自定义类型和严格自定义类型的自定义控件与输入控件不同，在一个 VI 中调用这两类自定义控件后，新生成的控件保持和文件中存储的自定义控件的链接关系。任何对文件中存储的自定义控件的修改，都会引起所有调用这个自定义控件的 VI 更新。这样就保证了一个精心设计的复杂数据类型在所有调用 VI 中保持同步更新。
- 自定义数据类型和严格自定义数据类型的区别在于控件数据类型保持一致的程度。对于

严格定义的数据类型，在调用它的 VI 中，除了可以修改是否可见、是否启用之外，无法对控件进行任何其他修改，完全保证和存储在文件中的自定义控件的高度一致，如外观、代表的数据类型、数据类型的精度、数据类型的输入范围等。而对于自定义数据类型，除了外观和代表的数据类型保持一致外，其他属性可以自由设置。图 1-12 中，中间的图为自定义数据类型，右侧的图为严格自定义类型。

学习笔记 可以使用自定义类型或者严格自定义类型构造通用或者复杂的数据类型。

1.2 编辑前面板和程序框图

LabVIEW 的前面板从基本用途上可以分为两类：GUI 人机交互界面和程序员交互界面（GPI）。GUI 是直接提供给操作者使用的，对编程者来说比较重要。GUI 针对不同领域的具体要求，有不同的设计标准。

GPI 和 GUI 则完全不同，GPI 是展现给程序员看的。对于多个程序员互相协作的项目，程序员交互界面也是非常重要的，也要遵循一定的标准。不过与 GUI 的标准不同，GUI 需要满足的是行业的标准规范，而 GPI 是 LabVIEW 程序员的“潜规则”，并不是必需的。因此也是仁者见仁，智者见智，不同的人有不同的理解。但是无论如何，清晰、整洁是最基本的要求。

同 VC、VB 等流行的常规编程语言一样，LabVIEW 也提供了有关控件布局方面的功能。由于 LabVIEW 是图形式编程语言，这方面的功能更加强大。

1.2.1 选择、移动和删除对象

前面板、前面板上的控件、程序框图上的接线端子、函数、图标、连线等统称为对象。所谓编辑前面板和程序框图，就是编辑这些具体的对象。

我们从前面板上的控件对象开始介绍，首先需要了解的是如何选取和移动对象，前面板控件的选择和移动的方法对程序框图同样适用。选择对象有多种方法，简单分类如下。

1．选择单个对象

通过工具选板的“定位/调整大小/选择”工具按钮选择。单击某个对象，则该对象被选中。对象被选中时，周围出现虚框。在任何情况下，采用矩形框选方式，可以直接选中对象。

2．选择多个对象

通过工具选板的“定位/调整大小/选择”按钮选择。单击选中某个对象，然后按住 Shift 键，选择其他对象，形成对象集合。在任何情况下，采用矩形框选方式，可以直接选中多个对象。

3．筛选对象

筛选对象是在选择多个对象的基础上，在按下 Shift 键的同时，单击对象，将对象添入或剔出对象选择集。如果原来对象是选中的，则剔除该对象；如果原来对象处于未选中状态，则添加到选择集中。通过虚框很容易判断对象是否被选中。

4．选择全部对象

选择全部对象，当然也可以采用矩形框选的方法，不过利用快捷键 Ctrl+A 更简单。

学习笔记 使用快捷键 Ctrl+A，可以选择前面板或程序框图中的全部对象。

5．移动对象

移动对象属于常用的编辑操作。在单选或多选对象后，直接拖动其中任何一个对象，则出现一个新的虚框，随鼠标的运动而移动，虚框的位置表明当前位置。将虚框移动到合适位置释放鼠标后，原来位置上的对象将消失，而被移动到鼠标指定位置。

如果移动之前按住 Shift 键，则可以保证移动沿水平方向或者垂直方向进行，方向取决于最开始的移动方向。采用键盘的方向键也可以移动选中的对象，而按住 Shift 键，则可以快速移动对象。

6．取消移动操作

克隆对象和移动对象都涉及中间取消的问题，取消克隆和移动操作有以下几种方法。

- 执行完毕后，在“编辑”菜单中选择“取消”项，快捷键是 Ctrl+Z。
- 克隆和移动过程中，按下 Esc 键，取消操作。
- 直接拖动到前面板窗口或者程序窗口外，取消操作。

学习笔记 在克隆和移动对象的过程中，按下 Esc 键可以取消操作。按下 Shift 键，可以沿水平或者垂直方向移动对象或者对象集合。

7．精确移动对象

移动对象时既可以用鼠标拖动 ,也可以用键盘操作。利用键盘方向键可以做比较精确的调整 ,按下 Shift 键可以快速移动对象或者对象集合。选择和移动对象，如图 1-13 所示。

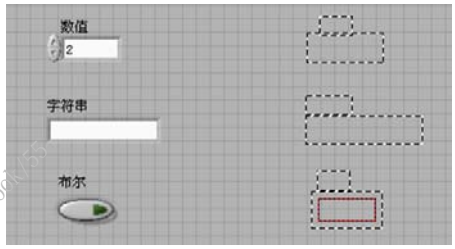

图 1-13 选择和移动对象

学习笔记 通过键盘移动对象时，按下 Shift 键可以快速移动。

8．删除对象

除了对象的创建、选择、移动、复制等操作，对象的删除也是经常用到的操作。选择要删除的对象或者对象集合，然后通过以下几种方法来删除。

- 在菜单栏的“编辑”菜单中，选择“从项目中删除”项。
- 利用 Windows 的剪切命令，快捷键为 Ctrl+X。
- 选择对象或者对象集合后，按下 Del、Delete、Backspace 三者中的任何一个键。
- 对象中包含的子元素（如控件的标签、标题等），是不能单独删除的，只能选择“显示”或者“隐藏”。

1.2.2 使用布局工具

前面板和程序框图有关布局的工具条是相同的，共有三个分类：对齐对象、分布对象和重新排序。其中每个分类中又有很多不同的子分类，下面我们分别介绍。

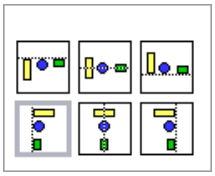
图 1-14 对齐对象

1．对齐对象

顾名思义，对齐对象是指将一组被选择的对象按照一定要求对齐排列，有上边缘、下边缘、左边缘、右边缘、垂直中心和水平居中几种对齐方式，如图 1-14 所示。

一般的控件对象默认都是显示标签的，各种对齐方式也是包括标签的对齐。标签默认位于对象的上方，如果我们移动了某个控件标签的位置，而对齐对象有可能是以标签为基准的，此时要特别注意。不过，标签是可以单独选择的，首先要对齐控件本身，然后再对齐标签或者标题。

前面板和程序框图都具有网格对齐功能。默认情况下，前面板是显示网格线的，而程序框图不显示网格线，可以通过“编辑”菜单或者快捷键 Ctrl+#切换是否网格对齐。在创建控件时，一般采用网格对齐方式。

学习笔记 在对齐时，先对齐对象，后对齐标签或者标题。快捷键 Ctrl+#用于在前面板或程序框图中切换是否网格对齐。

2．分布对象

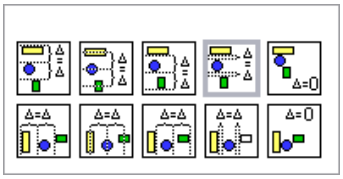

图 1-15　分布对象

各种分布对象的工具如图 1-15 所示。当我们将鼠标定位到某一个图标时，上方会出现说明文字，比如图 1-15 左上角第一个图标显示的文字是“上边缘”。结合它的图标很容易看出，它是以对象的上边缘为基准，沿垂直方向均匀分布的。

需要注意的是，图 1-15 最右边上下两个图标分别是垂直压缩和水平压缩。选择这两个选项，可以使所有选择的对象沿垂直方向或水平方向紧密排列。紧密排列功能在程序框图设计中使用得非常广泛。

这里以程序框图为例，简单地说明一下对齐和分布的使用方法。首先选择要对齐的对象集合，用矩形框选方式，选择“对齐对象”中的“左边缘”。然后选择“分布对象”中的“垂直压缩”，这样输入控件对象的排列就完成了，具体操作如图 1-16 所示。

对于前面板中的对象采用同样方法，进行对齐和垂直压缩后，会极大地节省前面板和程序框图的空间，同时使连线变得更加简单整洁。对齐、分布是 LabVIEW 程序员必须掌握的基本技能，整理后的效果如图 1-17 所示。

3．调整大小

创建一个对象后，经常需要修改其大小，尤其是前面板对象，比如输入控件、显示控件和装饰等。移动光标到某个控件上，如果光标指针是常见的箭头方式，表示目前的工具是定位/调整大小/选择状态。如果控件的大小允许调整，对象的四周和中间会出现方框标记，表示这个方向可以调整大小。移动光标到方框标记上，光标箭头变成缩放状后，按住鼠标左键沿某一方向拖动，就可以更改对象的大小了。这里的对象不仅仅指控件对象，也包括标签、标题等其他元素。但是采用鼠标拖动的方法，无法精确控制对象的大小尺寸。在开启网格对齐的情况下，鼠标定位的最小单位是一个网格。一个网格到底是多少呢？首先我们需要确定的是网格大小的基本单位是什么，无论是前面板还是程序框图，或者前面板和程序框图中的对象，都是以像素点作为基本单位的。

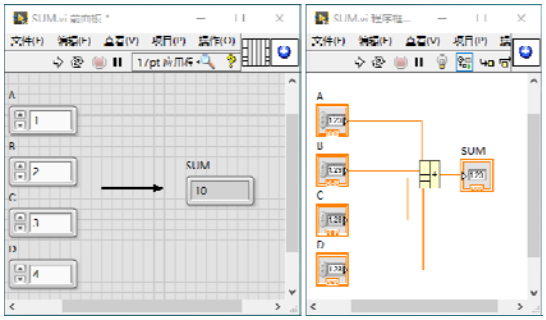

图 1-16 左边缘对齐，垂直压缩分布　　图 1-17 左边缘对齐，垂直压缩分布效果图

学习笔记 LabVIEW 中对象的大小是以像素点作为基准单位的。

LabVIEW 前面板和程序框图中的网格大小是可以设置的。默认情况下，前面板的每个网格为 12 个像素点，程序框图网格为 16 个像素点。对于新创建的 VI，可以在菜单栏中选择“工具”→“选项”，打开“选项”对话框，如图 1-18 所示。在这里，可以查看和设置网格大小。

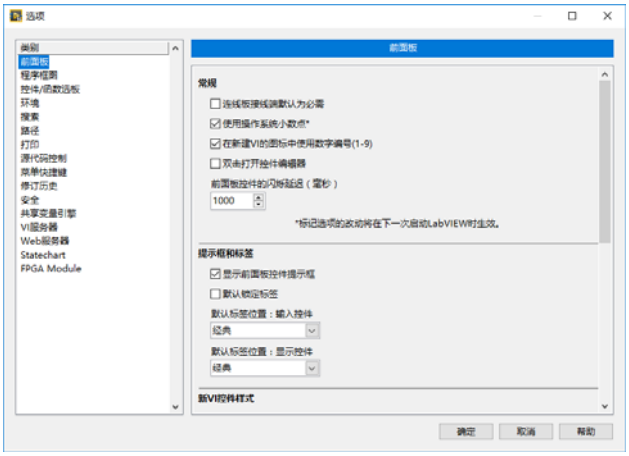

图 1-18 “选项”对话框

在这个对话框中，我们不但可以设置网格的大小，还可以设置网格相对于背景的对比度。当我们新创建一个对象时，新对象是有默认大小的。而这个默认大小可能不是网格对齐的，开启“缩放新对象以匹配网格大小”功能，可以强制新创建的对象调整大小尺寸，自动适应网格大小。另外，有的对象无法在水平和垂直两个方向同时对齐，这时 LabVIEW 会自动选择一个合适的方向单方向对齐。

学习笔记 前面板默认网格为 12 个像素，程序框图默认网格为 16 个像素。

LabVIEW 的前面板和程序框图并没有直接显示每个对象的大小，不过可以通过以下方法查看控件对象的大小。

- 在快捷菜单上，选择“属性”，然后在打开的“属性”对话框中查看当前控件对象的大小。
- 单击工具栏中的“调整对象大小”按钮，弹出“调整对象大小”对话框，如图 1-19 所示。

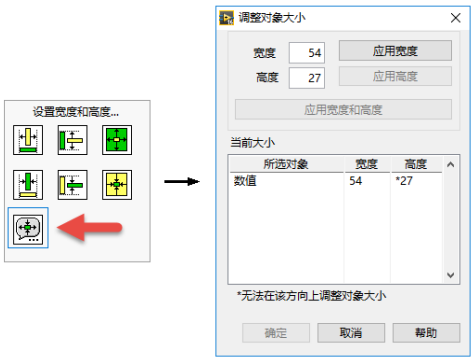

图 1-19 “调整对象大小”工具按钮和“调整对象大小”对话框

在上述对话框中不仅可以查看到控件对象的大小，更重要的是可以按照像素点修改控件对象

的大小，其中有的对象只允许单方向调整。图 1-19 所示的例子就只允许调整对象的宽度。

“调整对象大小”对话框是个较常用的对话框，在其中不仅可以查看和调整单个对象的大小，更还可以同时调整对象集合，使对象集合中的对象具有同样的宽度或者高度。

“调整对象大小”工具栏中其他几个按钮非常容易理解，都是作用于所选取的对象集合的，其功能分别为设置最大宽度、最小宽度、最大高度、最小高度。

4．重新排序

工具栏中有关对象布局的最后一项是重新排序，它包括 3 项基本功能。

（1）组合

功能相关的对象可以组合成一个组，作为一个单独的对象，统一进行复制、移动、删除等编辑操作，组中包含的对象的相对位置保持不变。

在一个包含大量对象的界面中，采用对象组合是非常必要的。对于已经对齐、大小调整好的对象，难免会出现误操作，比如移动了不应该移动的控件，而后续又执行很多的操作。这时如果想恢复到移动前的状态，不得不执行“编辑”菜单中的撤销操作，而撤销操作是一步步进行的，这样出现错误后所有有用的编辑操作都被撤销了。

在一个复杂的界面中重新分布对象也是非常困难的，因此非常有必要在一些对象元素完成编辑后，将它们组合成一个组。通过组合成组的方式，可以非常方便地实现 GUI 界面模块化。

图 1-20 中，四个方向键分别控制上、下、左、右四个方向。这几个控件对象的大小相同，功能相似，将它们组合在一起后，可以统一地移动，而相对位置保持不变。

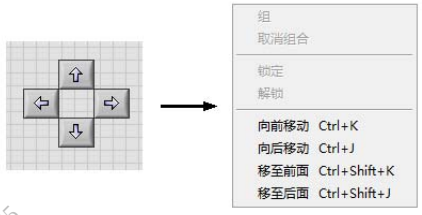

图 1-20　对象组合

（2）锁定

锁定与组合不同，组合是针对对象集合的，一个单一的对象是不可能组合的，而锁定功能既可以针对单一对象，也可以作用于整个对象组合。

锁定后，不允许对被锁定对象进行任何编辑操作，包括移动、删除、克隆等，甚至无法调用属性对话框修改属性。如果要编辑对象，必须通过“解锁”操作解除锁定，然后才能自由编辑锁定的对象。

学习笔记　对象编辑完成后，组合和锁定对象或者对象集合，可以有效防止误操作。

（3）重新排序

LabVIEW 并没有像某些软件一样在窗口显示对象的坐标，原因在于前面板的坐标原点是相对的、可移动的，并不能真正体现对象相对于前面板窗口的绝对位置。

仔细观察 LabVIEW 的前面板，有两条网格线比较特殊，它们是加深显示的，并且显示一个交叉黑点。这个交叉点就是前面板窗口客户区的坐标原点。之所以强调前面板窗口客户区，是因为前面板也是一个窗口，它也具有宽度和长度属性，它本身的位置属性也是由一个点的坐标簇构成的，不过这个坐标是相对于计算机桌面的。

LabVIEW 的对象对于垂直于屏幕的方向是有次序的，它按照对象的创建次序自动分配。越往后创建的对象相对于操作者“越近”，或者说它的次序越高。如果发生对象重叠的情况，次序高的对象将全部覆盖或者部分覆盖次序低的对象。

在 LabVIEW 中，可以通过工具栏中的“重新排序”按钮来调整相对于操作者的对象的显示次序，效果如图 1-21 所示。

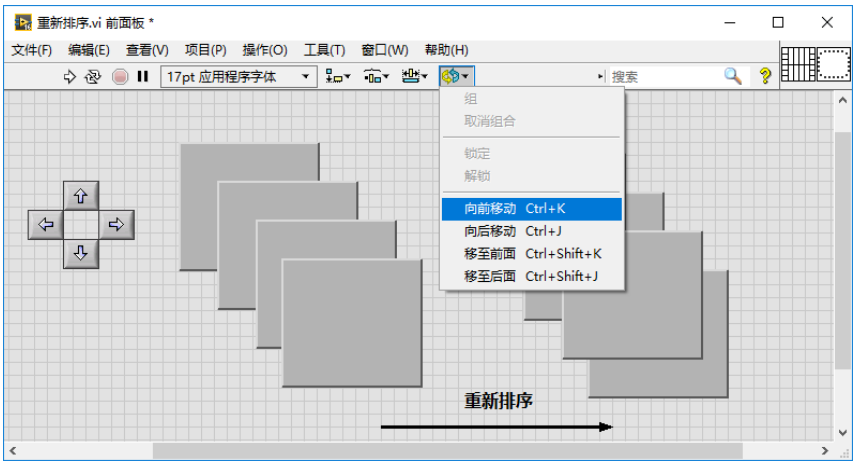

图 1-21 对象组合、锁定及改变次序

1.3 VI 及其属性对话框

在 C 语言中，函数是程序的基本单元，一个函数包括输入参数、输出参数和返回值。LabVIEW 中 VI 的概念和函数非常相似，它的输入参数就是输入控件，它的输出参数就是显示控件，C 语言的函数只能有一个返回值，而 LabVIEW 的 VI 可以有多个返回值，LabVIEW 的数据流有点类似于 C 语言的值传递过程。

1.3.1 VI 的层次结构

VI 是 Virtual Instruments 的缩写，它类似于 C 语言中的函数。在 C 语言中函数可以完成独立的、特殊的功能。函数可以被上一级的函数调用，被调用的函数称为子函数。LabVIEW 也是类似的，如果一个 VI 被上一级 VI 调用，被调用的 VI 称为子 VI，这是基本的模块式编程方法。LabVIEW 的“查看”菜单中提供了一个非常有用的功能——VI 层次结构。

VI 层次结构有自顶向下和自左至右两种不同的显示方式。图 1-22 所示的是典型的树形结构。根部称作应用程序实例，与之紧密连接的是顶层 VI。这是显示给用户的交互 GUI，顶层 VI 调用了 6 个子 VI，双击任何一个 VI 图标，可以直接打开子 VI。

顶层 VI 类似于 C 语言的 main 函数，这是应用程序的入口点。从 VI 名称上看，顶层 VI 和一般的 VI 命名无任何区别。这是一个不同于其他编程语言的一个显著特点。也就是说，任何一个 VI，既可以作为顶层 VI，又可以作为子 VI。

通过前面的介绍，我们已经知道 VI 是由前面板和程序框图组成的，而且可以单独运行。但是如果把它作为子 VI，则仅有前面板和程序框图是不够的，因为一般上一级函数都需要向子函数传递参数，然后由子函数返回处理结果，这种传递参数与返回结果的功能在 VI 中是通过连线板来实现的。

VI 还有一个重要的组成部分——图标。在上一级的程序框图中，子 VI 是以图标的方式显示的。在图 1-22 中，Testo、Test1、Test2、Test3 就是图标代表的 VI。

学习笔记 完整的 VI 由前面板、程序框图、连线板和图标组成。

下面通过创建一个计算二次函数的子 VI，来介绍建立 VI 的完整过程。

已知二次函数 $Y=AX^2+BX+C$，其中 Y 作为计算结果应该是输出，而 A、B、C、X 是作为参数

输入的。因此建立的 VI 应该包括 X、A、B、C 4 个输入控件，一个显示结果 Y 的显示控件。前面板与程序框图如图 1-23 所示。

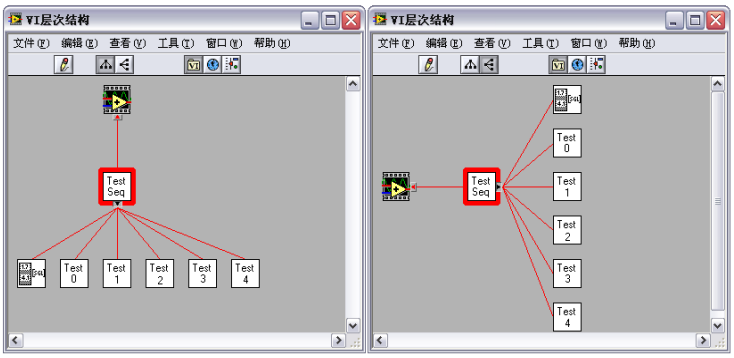

图 1-22　VI 的层次结构

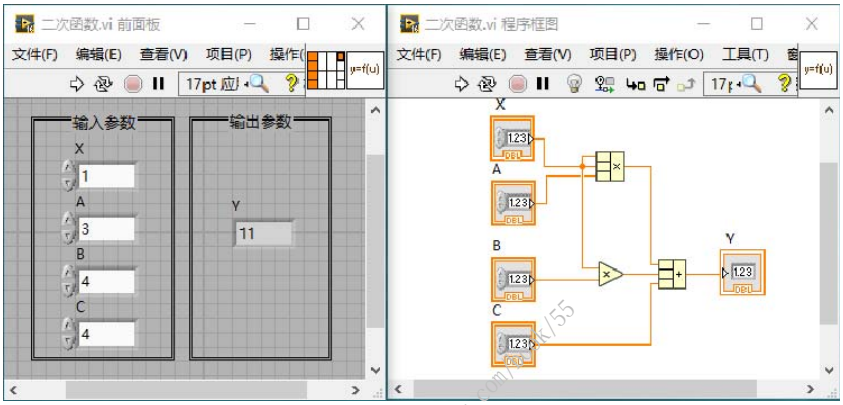

图 1-23　二次函数子 VI

创建前面板和程序框图后，右击前面板中右上角的图标，在快捷菜单中选择“显示连线板”项。利用工具选板中的连线工具，单击连线板中的端子和前面板中的控件，这样就确立了前面板控件与连线板的一一对应关系，相当于 C 语言中建立形式参数的过程。

双击图标，即可启动图标编辑器。在图标编辑器中，修改 VI 图标。这里的编辑与常规的图标编辑基本相同，也可以直接选择外部图标文件。

我们曾经创建过计算 *Y=KX+B* 的 VI，严格地说它还不能算是子 VI，原因是我们并没有分配连接板端子和制作相应的图标。而图 1-23 已经是一个完整的子 VI 了，它包括了前面板、程序框图、连接板和图标 4 个要素。

LabVIEW 的子 VI 可以单独测试，这与常规编程语言相比，具有极大的优势。常规语言的函数测试是非常麻烦的，需要专门做一个显示界面来显示测试结果，或者用打印语句输出中间运行结果和运行过程。而 LabVIEW 的 VI 是一个非常强大的模块式结构，每个 VI 模块都可以单独运行，模块之间不需要紧密的数据连接，非常有利于数据结构的封装。

VI 是通过连线板实现参数的输入和输出的。连线板位于前面板和程序框图的右上角。根据 LabVIEW 数据流动的特点，左侧作为输入，右侧作为输出，有利于连线。连线板和 VI 图标可以通过快捷菜单相互切换。显示连线板后，通过连线工具，分别单击连线板中的端子和控件，就建立了控件和连线板的一一对应关系。

VI 图标的制作比较烦琐，在图标中直接用文字来说明则相对简单。另外，也可以直接拖动一个外部图标到 VI 图标窗口，将它作为模板并做简单的修改，以满足自己特殊的需要。

编辑完 VI，保存，文件的扩展名自动命名为 VI。

1.3.2 调用子 VI

在上一节中，我们创建了一个完整的子 VI，它包括前面板、程序框图、连接板和图标 4 个组成要素，可以完成一个二次函数的计算。

LabVIEW 程序具有典型的层次结构，VI 之间的相互调用形成一个完整的程序。在一个 VI 中调用另一个 VI，有如下几种方法。

- 在函数选板中选择“VI”选项，弹出 VI 选择对话框。该对话框类似于通用文件对话框，用于选择合适的 VI。
- 找到相应的 VI 文件，直接将其拖动到程序框图窗口。
- 如果需要调用的 VI 处于打开状态，则直接将子 VI 在前面板或程序框图的图标拖动到程序框图，如图 1-24 所示。
- 如果建立了项目文件，则直接拖动项目文件中的 VI，如图 1-25 所示。

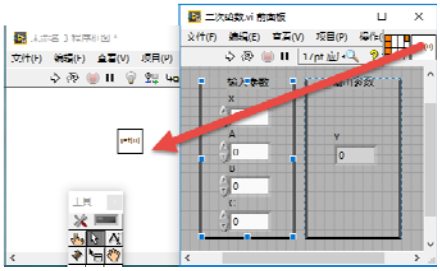

图 1-24 直接拖动打开的子 VI 的前面板或程序框图图标

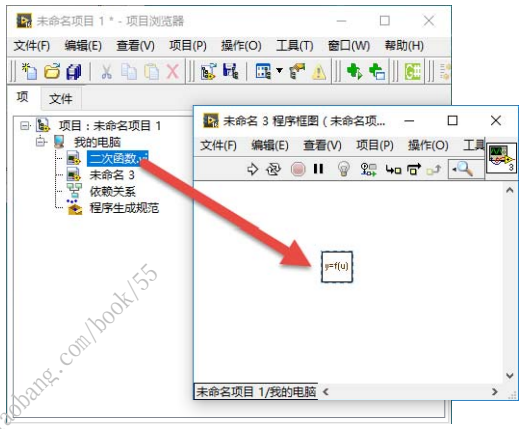

图 1-25 拖动项目文件中子 VI 的文件图标到程序框图

1.3.3 VI 的属性设置

对象是个虚拟的、综合的概念。前面板和程序框图本身就可以称为前面板对象和程序框图对象，输入控件、显示控件和装饰控件也是对象，包括接线端子和连线板也都可以称作对象。

对于输入控件和显示控件，可以通过快捷菜单弹出属性对话框，然后在对话框上设置控件对象的各种属性。VI 也是对象，同样具有各种属性，在菜单栏中，选择“文件”→“VI 属性”，或者使用快捷键 Ctrl+I，可以弹出“VI 属性”对话框，如图 1-26 所示。

学习笔记 使用快捷键 Ctrl+I 可以打开“VI 属性”对话框，在这里可以设置 VI 的各种属性。

“VI 属性”对话框中包含 VI 的大量信息。有些属于查询信息，处于只读状态，不能更改。有些属于可设置的属性，比如 VI 的外观、位置等。如图 1-26 所示，VI 属性包括以下几个分页。

1．常规

“常规”页面提供了几个重要的信息，包括 VI 文件名、文件存储的实际位置、当前修订版本号和 VI 图标信息。在“常规”页面中，还可以更改 VI 图标。当然如果 VI 已经打开，则可以在其中直接修改。

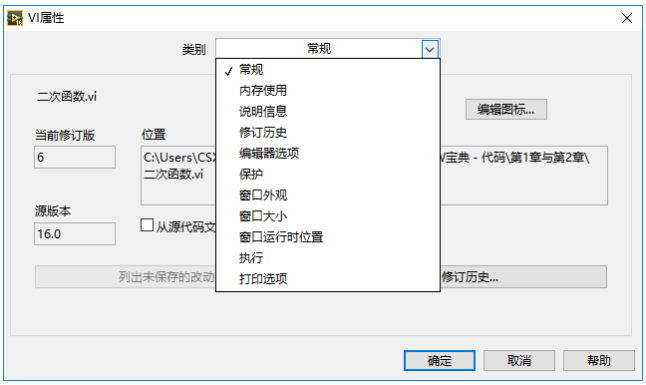

图 1-26 “VI 属性”对话框

“常规”页面一个重要的功能是，可以在此设置版本修改信息。每次对 VI 进行重大修改，都可以添加说明信息，比如修改原因、增加的功能等。单击“重置”按钮，可以将版本号重置为 0。

学习笔记 通过“VI 属性”对话框可以查看 VI 的实际存储位置。

2．内存使用

程序的优劣在很大程度上取决于内存的使用情况，在“VI 属性”对话框的“内存使用”页面上，可以查看 VI 当前占用内存的情况和 VI 占用硬盘空间的大小。

VI 占用的内存空间分为前面板对象、程序框图对象、代码空间、数据空间 4 部分。

学习笔记 在“VI 属性”对话框上，可以查看内存使用情况和 VI 占用的硬盘空间大小。

3．说明信息

类似于控件对象的说明。其他 VI 调用这个 VI 时，在即时帮助窗口显示该说明。说明信息既可以存储于 VI 本身，也可以存储于帮助文件中。

4．修订历史

设置提示输入修订信息的触发条件，包括每次保存 VI 时添加注释，关闭 VI 时提示添加注释，保存 VI 时提示输入注释，记录由 LabVIEW 生成的注释。

5．编辑器选项

在这个页面上可以设置 VI 的前面板和程序框图的网格线的大小。在菜单栏，选择“工具”→“选项”，在打开的对话框上也有网格线的设置选项。不同的是这个对话框中设置的是 LabVIEW 的基本工作环境，对所有后来创建的 VI 都起作用，而在“编辑器选项”中的修改只是对该 VI 起作用。

在“编辑器选项”页面中还可以设置自动创建控件时控件的样式和创建方式，比如通过函数接线端子自动创建。控件样式可以选择新式、经典、系统三种。

6．保护

控件对象可以通过工具栏锁定，防止用户对其非法编辑。在“保护”页面中也可以设置锁定选项，以防止未经授权而更改 VI，不过此时锁定的是整个前面板和隐藏的程序框图。如果想查看程序框图或者更改前面板，必须通过“保护”属性页解除锁定。

更严格的锁定方式是用密码锁定。我们可以设置密码，没有密码的用户是无法打开 VI 程序框图的。这样既实现了前面板的锁定，又保护了源代码。不过需要注意的是，必须精心设计密码，一旦自己忘记了密码，是没有任何方法解锁的。

学习笔记 设置 VI 密码，可以防止其他人员查看程序框图或者修改前面板。

7．窗口外观

在这里可以选择 VI 的几种外观方式。当然通过属性节点也可以设置外观方式。窗口外观有顶层 VI、对话框、默认、自定义 4 种方式，它们的区别在于是否显示主菜单，是否显示工具栏，以及是否显示窗口最大、最小、关闭按钮等。

8．窗口大小

“窗口大小”页面用来设置前面板的最小尺寸，包括宽度和高度，单位是像素点。VI 的前面板中没有直接显示出面板的大小尺寸，可以通过“窗口大小”属性页间接查看前面板的尺寸。

当单击“设置为当前前面板大小”按钮后，“宽度”框和“高度”框显示当前前面板的宽度和高度，并把当前宽度和高度作为最小宽度和高度。设置最小宽度和高度后，如果缩小前面板，最小只能达到设定的最小尺寸，扩大则不受影响。

另外，也可以直接通过输入宽度和高度数值的方法，定义 VI 前面板的最小尺寸。如果前面板的尺寸小于设定尺寸，LabVIEW 将自动调整前面板的尺寸为设定的最小尺寸。利用这个方法可以精确设置前面板的尺寸，如图 1-27 所示。

图 1-27 “窗口大小”属性页

学习笔记 利用设定前面板最小尺寸的方法，可以间接设定前面板的精确尺寸。在 VI“窗口大小”属性页上，设定“使用不同分辨率显示器时保持窗口比例”选项，可以使前面板中的对象按比例适应各种显示器。在该属性页上，还可以设定前面板上的对象与前面板成比例缩放。

9．窗口运行时位置

用来设置 VI 运行时前面板相对于桌面的位置和大小。如果设置为不变，就可以保持 VI 窗口原来的位置。也可以居中显示、最大化显示、最小化显示，或者采用自定义方式。若采用自定义方式，可以根据需要，自由设定运行时前面板的位置和大小。

10．执行

“执行”页面如图 1-28 所示，其中的“标准优先级”和“首选执行系统”设置比较复杂，一般不需要设置。下面分别介绍其他的选项。

（1）允许调试

在这个属性页上，“允许调试”复选框默认是勾选的。在允许调试的情况下，允许进行单步跟踪、设置断点、调用某个子 VI 时暂停程序、高亮显示程序运行过程等操作。

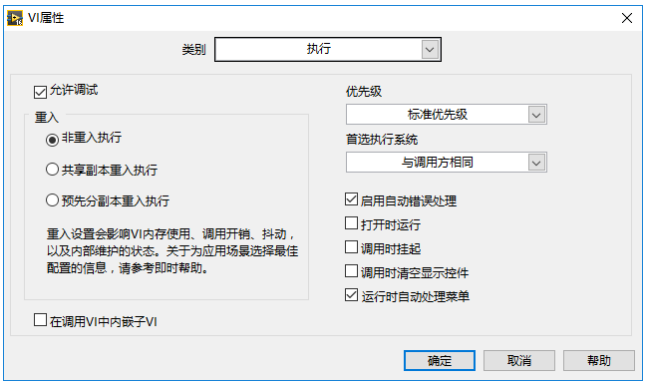

图 1-28 “VI 属性”对话框之“执行”属性页

（2）重入设置

“非重入执行”为默认选项。由于 LabVIEW 是支持多线程的，因此两个线程同时调用同一子 VI 时，需要让先满足数据流条件的子 VI 首先得到调用权，另外一个线程必须暂时等待，等到上一个调用线程结束调用时，才能得到调用的控制权。简单地说，在不允许重入执行的情况下，任意时刻只能有一个线程可以运行子 VI。通常我们创建的功能 VI 被设置为可重入执行，但是有些场合，比如针对硬件操作时，必须选择“非重入执行”方式。这保证了某一时刻，只有该 VI 在操作硬件，其他线程调用必须等待该操作完成。

在新版本中，重入执行分为了“共享副本重入执行”与“预先分副本重入执行”两项，其中“预先分副本重入执行”，就是早期版本的“重入执行”方式。

选择“预先分副本重入执行”功能，则每个线程运行的是这个子 VI 的备份，具有单独的前面板、程序框图空间和单独的数据空间、代码空间。LabVIEW 的很多内部函数节点都是可重入的。加、减、乘、除等基本运算函数，如果不允许重入，程序的运行效率会非常低。

若选择“共享副本重入执行”方式，则未初始化的移位寄存器会共享统一的数据空间。在某些特殊场合可能需要这种方式，在多个副本之间共享数据。

（3）启动自动错误处理

“启动自动错误处理”复选框，默认是勾选的。这里所要处理的是程序运行过程中发生的错误，而不是指 VI 本身的错误。当 VI 本身存在类似于常规语言的语法错误（比如函数节点未连线）时，编辑的过程中 LabVIEW 会提示错误，指出 VI 无法运行，然后弹出对话框指明错误所在。

另外，在运行过程中可能发生一些不是很重要的运行错误，比如打开一个根本不存在的文件。如果选中“启动自动错误处理”复选框，那么发生运行错误时，就会自动弹出错误对话框。这在实际应用中会带来一些不必要的麻烦。比如一个无人值守的监控程序，如果发生非特别重大错误，弹出对话框后会停止程序的运行，直到有人取消错误对话框，这显然是不合理的。这种情况下，可以取消“启动自动错误处理”复选框的选择，由程序本身设置错误捕捉陷阱，然后根据错误的类型、性质和严重程度，采取相应的处理措施。

（4）打开时运行

“打开时运行”复选框，默认不勾选。这里的打开是指在 LabVIEW 菜单栏中，选择“文件”→“打开”项来打开，或者是在计算机中直接双击 VI 文件名打开。不勾选“打开时运行”复选框，则以编辑方式打开文件；勾选这个复选框，则打开后直接运行文件。

（5）调用时挂起

“调用时挂起”复选框，默认是不勾选的。这个选项主要是在程序调试时使用。当勾选后，程序调用到这个 VI 时，暂时停止程序。这时可以通过探针等调试工具观察 VI 的运行情况。

（6）调用时清空显示控件

“调用时清空显示控件”复选框，默认情况下是不勾选的。显示控件当前显示的值完全取决于它的接线端子当前数据的流动情况。在某些情况下，数据根据条件可能不会流入到显示控件，这时可以选择“调用时清空显示控件”复选框。另外，这里的所谓“清空”，并非不显示任何值，而是显示显示控件的默认值。

学习笔记 通过“VI 属性”对话框的“执行”属性页设定“调用时清空显示控件”，显示控件将显示默认值。

11．打印选项

常规编程语言都提供了代码打印功能，而 LabVIEW 的代码实际就是程序框图，它能打印出来吗？一个条件选择结构可能包括很多条件分支，而同一时刻屏幕上只能显示其中一种，能打印出全部吗？

LabVIEW 不但可以打印，还可以对不同的 VI 进行单独的设置，单独的打印设置随着 VI 一起存储。如图 1-29 所示，在“打印选项”页面上，可以选择是否打印页眉，是否对前面板加边框，是否缩放前面板匹配打印页面，是否缩放程序框图匹配打印页面，还可以自定义上、下、左、右的页边距。

图 1-29 “打印选项”页面

“每次 VI 执行结束时自动打印前面板”复选框默认不勾选。这项设置在特定情况下非常有用。例如，我们要制作一个票据打印系统，首先可以制作一个标准的前面板，其中包括必须输入的数据和相关计算的结果显示，然后勾选这个选项，那么当 VI 调用结束，系统就会自动调用打印机打印结果。在工业控制中，可以自动打印报警信息或者程序中间运行结果。

VI 属性非常多，这里无法一一介绍。要了解这些属性的作用，需要在编程时仔细体会。通过以上的讨论，我们已经学会了如何创建、编辑、调用 VI，以及如何设置 VI 的属性。

我们知道，VI 的前面板是由各种控件组成的，所以必须充分了解各种控件的使用方法，这是 LabVIEW 编程的基础。

1.4 基本控件及其使用方法

我们必须首先了解的基本控件包括基本数值控件、布尔控件、数组控件、簇控件、波形图表控件和波形图控件。之所以首先要了解这些基本控件，是因为它们是最常用的，是构成一个 VI 的基本控件对象。VI 就是程序，程序是离不开数据和运行结构的。

1.4.1 基本数值控件

LabVIEW 是通过控件选板选择控件对象的。控件选板包含了大量的 LabVIEW 控件，按照控件能代表的数据类型（比如数值型数据、布尔型数据、字符串和路径数据等）分成不同的类别。每个类别中的控件所包含的数据类型是相同的或者近似的，同类别控件又分为不同外观形式的输入控件和显示控件。

1．打开和固定控件选板

打开控件选板的最基本方式当然是使用菜单，控件选板、函数选板的打开命令都位于“查看”菜单中。不过最方便的方法是右键单击前面板任意位置，弹出控件选板。选择完成后，弹出的控件选板会自动关闭。

单击控件选板左上角的图钉按钮可以使选中的控件类别固定显示，始终处于打开状态。这种情况适用于创建多个类型相似但是外观不同的控件。

用图钉按钮固定显示，是 LabVIEW 常用的基本操作，函数面板中也有作用相同的图钉按钮。如图 1-30 所示，这里演示了图钉按钮的使用方法。

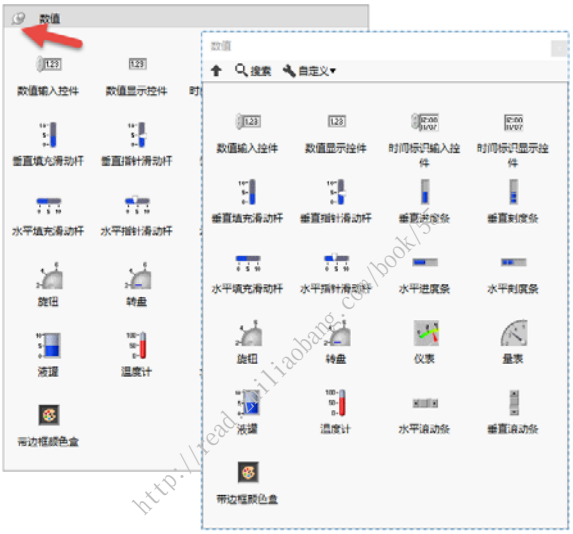

图 1-30　图钉按钮的使用

数值控件有多种显示方式，但是它们都有一个共同的特点——包含的数据类型都是数值型的。数值输入控件是最基本的数值控件，其他不同外观的数值型控件都是以它为基础的。LabVIEW 的控件是典型的对象继承结构，数值输入控件是其他数值型控件的基类。

2．数值控件的组成和显示方式

数值输入控件对象由一些基本对象元素组成，这些元素包括增量按钮、减量按钮、数字文本框、标签、标题、单位标签和基数等。基数指的是进制形式，可以是十进制、十六进制、八进制、二进制。基数不同，不过是数值的表现形式不同，它所代表的值是相同的。

图 1-31 说明了数值输入控件的基本对象元素以及进制、单位的不同表现形式。数值输入控件的基数默认是不显示的，可以在它的快捷菜单上选择“显示”→“基数”项来确定是否显示。常规语言中对一个数用不同的进制显示，实现起来非常复杂，而在 LabVIEW 中只需选择相应的进制。这充分说明了 LabVIEW 的确是工程师的语言，直观、方便又快捷。

如果数值控件的基数处于显示状态，则单击基数标记（数值左侧），可以自由选择十进制、

十六进制、八进制、二进制和 SI 符号。如果是整型数，那么这些数值都可以选择；如果是双精度数等，则只能选择十进制和 SI 符号。另外，LabVIEW 可以自动判断哪些基数是可以显示其他进制的，哪些是不能显示的。选择 SI 符号，会以字母的形式显示比较大或者非常小的数值。例如，10^3 用 K 表示，10^6 用 M 表示，10^9 用 G 表示，10^{-3} 用 m 表示，10^{-6} 用 u 表示，10^{-9} 用 n 表示等。

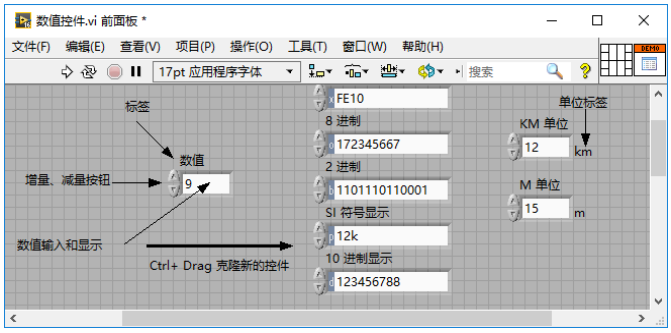

图 1-31 数值控件的组成和进制转换

学习笔记 在数值控件的快捷菜单上，可以选择是否显示基数。通过单击控件上的基数，可以选择十进制、二进制、八进制、十六进制和 SI 显示。很大和很小的数值适合用 SI 符号表示。

数值控件不仅可以实现数制的自由转换，还可以携带物理单位。在图 1-31 中，右面的两个数据是包括单位的。它们采用克隆方式复制，所以数值相同。如果把 km 改成 m，数值将自动由 1 变成 1000，自动实现不同物理单位的转换。

LabVIEW 数值输入控件的单位标签默认是不显示的，但是可以在它的快捷菜单上选择“显示”→“单位标签”项，来确定是否显示。

学习笔记 适当地运用数值控件的单位标签，可以自动进行单位转换。

单位转换的功能是非常重要的。因为在实际应用过程中，经常会遇到单位转换的问题。只要适当选取单位标签，LabVIEW 会自动地为我们完成单位转换的工作。另外，LabVIEW 不仅可以对相同单位类型进行转换（比如长度单位），还可以通过运算自动处理组合单位，如图 1-32 所示，长度相乘，自动生成面积单位。

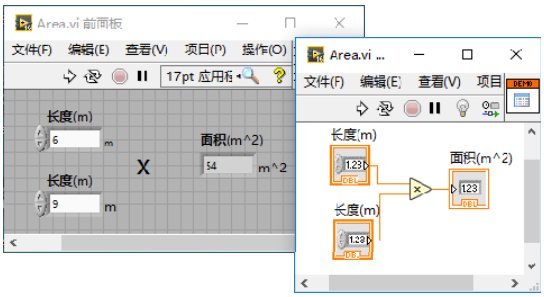

图 1-32 长度的乘积是面积，LabVIEW 自动处理单位

3．数值控件的属性

在快捷菜单上，选择“属性”项，即可打开属性对话框。属性对话框由许多属性页组成，涵盖了大量的控件属性设置信息，该属性对话框使用极其频繁，必须详细了解。不同的控件属性对话框的内容是不同的，其中数值控件的属性对话框最为复杂。如图 1-33 所示，这里以量表控件的属性对话框为例，说明一下数值型控件属性的用法。

如图 1-33 所示，数值控件的属性对话框由多个页面组成。标签、标题等通用属性我们已经介绍过，下面重点关注数值控件的专用属性。

（1）外观

“外观”属性页的下半部分是专用属性选项，不同类型的控件这部分选项有很大不同。对于量表控件，可以通过“添加”按钮，增加指针。

量表、旋钮、滑动杆控件是继承于基本数值控件的，它们内部本身就包含一个基本的数值控件。默认情况下基本数值控件是隐藏的，可选择显示数值控件来打开它，并选择“显示基数”，修改后效果如图 1-34 所示。

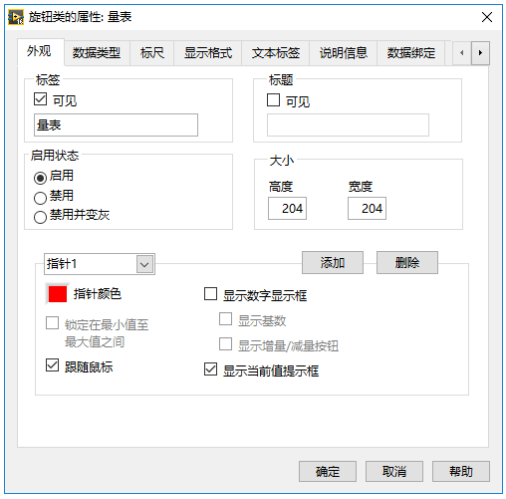

图 1-33　量表控件的属性对话框

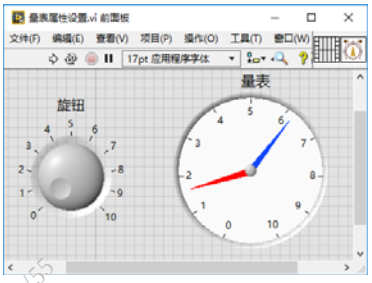

图 1-34　修改后量表控件的显示效果

对于可以用鼠标拖动的数值控件，还有三个重要的选择项。

- “显示当前值提示框”。默认情况下，运行时用鼠标拖动旋钮或者滑动杆，会自动出现一个黄色的数字框，提示当前位置的值。
- “锁定在最小值至最大值之间”。如果取消该复选框的选择，则指针或者旋钮随着鼠标的移动自由跟随移动。当达到最大值继续移动时，马上会回到最小值。这种情况在实际应用中有时会非常危险。例如，通过旋钮控制重物的高度，如果没有该指定，当鼠标拖动超过最大值时，则会从最高点突然降落到最低点，造成事故。如果勾选了该复选框，则当鼠标拖动数值到最大和最小区域之外时，拖动操作将无效，从而避免了上述情况的发生。
- “跟随鼠标”。对于旋钮类型控件，还可以选择是否开启鼠标跟随功能。未选择“跟随鼠标”复选框时，只能通过拖动改变控件的值；选择“跟随鼠标”复选框后，单击旋钮或量表的任意位置，则旋钮或者指针自动指向到鼠标位置，控件的值随之自动改变。

学习笔记　使用旋转型数值控件要特别注意是否选择“锁定在最小值至最大值之间”复选框。

（2）数据类型

对“数据类型”属性页的设置非常简单，主要是选择控件所代表的数据类型，比如各种整型数、浮点数等。对于定点数，“数据类型”属性页提供了比较详细的设置。改变数据类型一般通过快捷菜单中“表示法”项来完成，这比使用“数据类型”属性页来选择要方便得多。

（3）数据输入

“数据输入”属性页如图 1-35 所示。默认情况下“使用默认界限”复选框是勾选的。LabVIEW 是面向工程应用的软件，因此，对输入控件要求很高，必须保证用户的输入是有效的和合理的。“数据输入”属性页用来设置输入的有效范围以及超出界限后的处理方法。

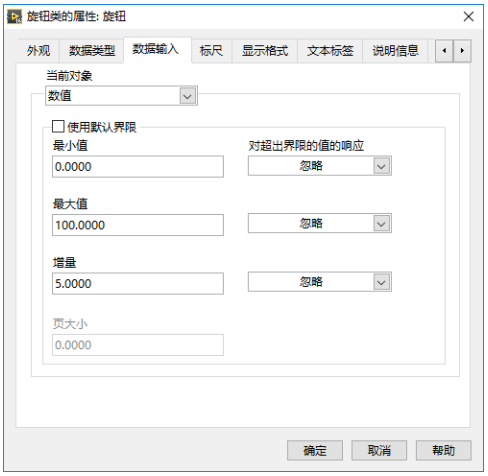

图 1-35 “数据输入”属性页

如图 1-35 所示，“数据输入”属性页主要用于设置数据的输入范围，包括如下内容。

- “最小值”和“最大值”参数，用于设置数值的输入范围。对于图形用户界面来说，对数值设定输入范围是非常重要的。根据实际需要，程序设计者有必要限定输入的具体范围。例如 ,如果要设计一个输入年龄的输入控件 ,首先要通过“数据类型”属性页或控件快捷菜单，选择合适的数据类型。比较合理的选择是 U8 数据类型，它只占用 1 字节，默认最大值是 255，最小值是 0。然后限定合适的输入范围，比如最小值是 0，最大值是 150。无论如何，我们不能依赖用户来输入合理数据。在程序设计时就应该充分考虑各种情况，要由程序保证数据的输入是合理的、有效的。

学习笔记 需要适当地设置数据的合理输入范围，保证用户输入的是有效数据。输入范围的设置对 GUI 起作用，对不显示前面板的子 VI 调用无效。

- “增量”参数用于设置，在数值型控件中单击“增量”、“减量”按钮时，数值变化的最小量，一般使用默认设置。
- 在“对超出界限的值的响应”区域可以对于超出范围的数据，设置“忽略”或者“强制”的处理方式。采用“忽略”方式，超出设定范围的数据依然有效。选择“强制”方式，则自动把大于最大值的输入强制转化成最大值，自动把小于最小值的输入值转化成最小值。

对于数值控件的多位数输入，用光标定位，比如光标定位于百位，单击增、减量按钮，则数值整百增加或者减少，光标定位于千位，单击增、减量按钮，则数值整千增加或者减少。

学习笔记 将光标定位在数字的某位上，则单击增、减量按钮数值可以按照该位所代表的数值快速增加或者减少。

（4）显示格式

如图 1-36 所示，“显示格式”属性页包含了丰富的格式信息，包括选择计数法、选择进制、时间格式等，还有很多特殊的用法，包括是否隐藏无效 0，是否使用最小域宽等。选择最小域宽，

比如设置为 6 位，当数值不满 6 位时，不足位用空格或者 0 填充。选择空格时，可以选择左侧填充空格或者右侧填充空格。

图 1-36 “显示格式”属性页

图 1-37 显示格式之高级编辑

如图 1-37 所示，在属性页左下方，选中“高级编辑模式”选项，即可自定义显示格式，包括多行显示、不同进制显示等，功能非常强大。例如，选择“数值格式代码”项，然后在下方选择“浮点表示”项，则格式字符串如图 1-38 左图所示。最后得到的效果如图 1-38 右图所示。

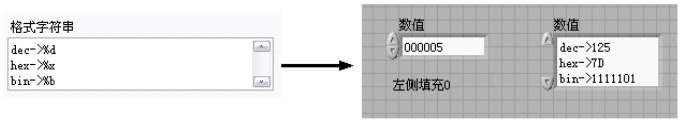

图 1-38 高级编辑后的效果

在数值控件的高级显示格式中，格式字符串的运用非常关键，具体的数字格式字符串如表 1-2 所示。

表 1-2 整数和浮点数格式字符串

整　　数	浮 点 数
x：十六进制整数（例如，B8）	f：带小数格式的浮点数（例如，12.345）
o：八进制整数（例如，701）	e：科学计数法表示的浮点数（例如，1.234E1）
b：二进制整数（例如，1011）	g：根据数字的指数，LabVIEW使用f或e。若指数大于–4或小于指定的精度，则LabVIEW使用f。若指数小于–4或大于指定的精度，LabVIEW使用e
d：带符号的十进制整数	p：以SI符号表示的浮点数
u：不带符号的十进制整数	

学习笔记 格式字符串以“%”开始，按快捷键 Shift+Enter 可以在编辑字符串的时候换行。

（5）说明信息

“说明信息”属性页包括“说明”和“提示框”。控件的说明是出现在即时帮助窗口中的，使用快捷键 Ctrl+H 可以随时打开和关闭帮助窗口。当光标在控件上停留时，则显示提示框，如图 1-39 所示。

在“说明信息”属性页的说明框和提示框中，可以编辑和添加控件的详细说明和简要提示。通过标注“<B>”和“</B>”，还可以使这段文字加粗显示。“说明”中的信息可以写详细些，但是“提示框”中的文字要简明、直观。说明和提示框的具体用法如图 1-40 所示，特别要注意加粗文字的方法。

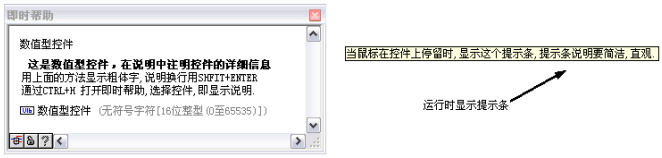

图 1-39 即时帮助和运行时控件提示条

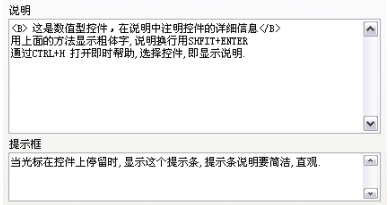

图 1-40 控件“说明”和“提示框”

控件在编辑状态下是不显示提示框的，只有运行时才能显示。

即时帮助窗口下方有“显示完整连线端及路径”、“锁定帮助”和“详细帮助”3 个按钮。其中第一个非常实用，用来在即时帮助窗口显示 VI 在硬盘中的绝对路径。

（6）快捷键

在“快捷键”属性页上，可以为显示给用户的 GUI 上的控件设置快捷键。不显示给用户的子 VI，则不需要配置快捷键。不同类型的控件，“快捷键”属性页上的选项是不同的。

LabVIEW 会自动显示控件可以配置快捷键的部分，比如数值型控件，可以配置增量按钮或减量按钮，其中增量按钮和减量按钮还可以进一步配置是否选中。如果配置选中，则使用快捷键后，不但控件的值发生变换，同时数值控件还处于选中状态，具有焦点。配置快捷键时既可以配置单独的键，也可以用 Ctrl 或者 Shift 配置成组合键。

学习笔记 如果控件处于隐藏、禁用或者禁用并发灰状态，则快捷键不起作用。

有的程序（比如一个监控程序），不希望无关人员停止它，此时可以将“停止”按钮设置为禁止、禁止并发灰或隐藏的方式。但是隐藏了“停止”按钮，有授权的人也将无法停止程序。这种情况下，可以将“停止”按钮移动到显示界面之外，并为它配置一个切换的快捷键。然后就可以通过快捷键停止程序运行，而其他不知道快捷键的人员将无法停止程序。

另外，在“快捷键”属性页中，还可以设置控件是否忽略 Tab 键。通常情况下，可以用 Tab 键切换控件，被切换的控件具有焦点。如果为控件设置“忽略 Tab 键”功能，则按 Tab 键时，将越过这个控件选择下一个。

（7）文本标签

“文本标签”属性页如图 1-41 所示，设置好文本标签的效果图如图 1-42 所示。很多数值控件的快捷菜单中，都有“文本标签”选项，比如滑动杆、旋钮、转盘和量表等。它们不仅可以显示连续数字，也可以通过文本标签功能变成档位开关。

图 1-41 “文本标签”属性页

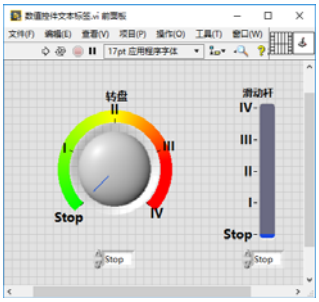

图 1-42 文本标签效果图

1.4.2 基本布尔控件

布尔控件属于常用控件，使用极其频繁。与常规语言的布尔型控件不同，LabVIEW 提供了大量的、功能各异的布尔型控件，极大地方便了用户。不仅如此，LabVIEW 在 DSC 组件中也提供了大量的布尔型控件，比如管路、阀门等。

开关型控件在工业领域是非常重要的，比如各类开关、按钮、继电器等，从物理描述上来看都是布尔型，只有“开”和“关”两种状态。在编程语言中一般使用真和假描述，这样更具有普遍性。

1．布尔型控件

LabVIEW 的布尔数据类型占用 1 字节，而不是 1 位。1 字节从二进制的角度上看是由 8 位组成的，1 字节实际上可以表示 8 个真假状态。

目前，几乎所有的编程语言都采用整数来表示布尔量。虽然字节相对于位来说，占用的空间比较大，但它是各种编程语言支持的基本数据类型，运算速度很快。只有在单片机编程中，由于 RAM 空间极其有限，才采用位表示布尔型数据。

学习笔记 LabVIEW 的布尔型数据占用 1 字节。

LabVIEW 的布尔型控件从外观上分成三大类——现代型控件、古典型控件和系统控件。现代型控件具有立体外观，也称作三维控件。LabVIEW 的古典型布尔控件的外观某些时候更类似于真实的开关、按钮等。古典布尔控件主要用在早期版本的 LabVIEW 中，不过现在仍然有很多人继续使用。系统型控件与操作系统本身用的控件是类似的，在涉及软件系统配置时，经常使用系统型控件。

LabVIEW 布尔型控件从名称上看分成两类，按钮布尔控件和开关布尔控件。按钮控件和开关控件虽然都是布尔型控件，但是它们的物理意义是有区别的。

真实的按钮按下时原来的状态改变，释放后自动恢复到原来的状态。原来的状态是接通还是断开，取决于接线方式，因此有常开按钮和常闭按钮。开关则不同，改变状态后，开关自己保持在一个稳定状态，直到下一次改变为止。比如，计算机机箱上的是启动按钮而不是启动开关，因为按钮内部有个弹簧，当手离开后，弹簧使按钮自动复位，而灯的开关则完全不同，当打开开关后，它会自动保持在打开的状态。

学习笔记 LabVIEW 布尔型控件分成按钮型和开关型，应根据需要选择按钮或者开关。

虽然 LabVIEW 的布尔型控件分成按钮型和开关型两种，但是 LabVIEW 内部并没有区分按钮型和开关型。从编程的角度看它们是完全相同的，只是默认的操作方式不同。编辑 GUI 时还是要根据需要，选择合适的按钮或者开关，以免造成用户误解。

2．LabVIEW 布尔型控件的机械动作属性

所有的控件属性都是通过快捷菜单和属性对话框进行设置的。布尔控件的标签、标题、可见性、开启与否、说明、快捷键配置等通用属性，与数值型控件非常类似。

除了上述的通用属性外，LabVIEW 的布尔型控件还有一个特别的属性——机械动作属性。机械动作属性是布尔型控件特有的，也是常规编程语言中不存在的属性。

布尔控件“值改变”的瞬间是非常重要的，在现实世界里也存在这种现象。比如我们有一个手持的计数器，每按一下按钮，需要增加一个计数。这时我们就要考虑机械动作的问题。如果按钮一旦按下就开始计数，由于仪器内部反应非常快，在我们按下到释放之前，内部可能产生多次计数，这显然是不合理的。正确的做法是在按钮抬起时计数，这样就可以按一下，产生一次计数。

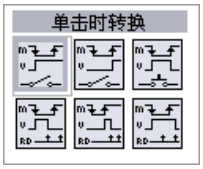

图 1-43 布尔控件的机械动作

在各类机械动作中，该类动作称作"释放时转换"。LabVIEW 布尔控件的机械动作共分成 6 种，根本区别在于转换生效的瞬间和 LabVIEW 读取控件的时刻，如图 1-43 所示。

在属性对话框的"机械动作"属性页中，不但可以选择 6 种不同的机械动作，还可以直接预览实际效果。如果对机械动作本身非常熟悉，直接在快捷菜单中选择机械动作比较快捷。

如图 1-43 所示，总计 6 种机械动作，它们的图标非常形象，最上边的 M（mouse）表示操作控件时鼠标的动作，V（value）表示控件输出值，RD（read）表示 VI 读取控件的时刻。下面按照图 1-43 中从左至右的顺序，介绍这 6 种机械动作。

（1）单击时转换

这种机械动作相当于机械开关。鼠标单击后，立即改变状态，并保持改变的状态，改变的时刻是鼠标单击的时刻。再次单击后，恢复原来状态，与 VI 是否读取控件无关。

（2）释放时转换

当鼠标按键释放后，立即改变状态。改变的时刻是鼠标按键释放的时刻。再次单击并释放鼠标按键时，恢复原来状态，与 VI 是否读取控件无关。

（3）单击时转换保持到鼠标释放

这种机械动作相当于机械按钮。鼠标单击时控件状态立即改变，鼠标按键释放后立即恢复，保持时间取决于单击和释放之间的时间间隔。

（4）单击时触发

这种机械动作，鼠标单击控件后，立即改变状态。何时恢复原来状态，取决于 VI 何时在单击后读取控件，与鼠标按键何时释放无关。如果在鼠标按键释放之前读取控件，则按下的鼠标不再继续起作用，控件的值已经恢复到原来状态。如果在 VI 读取控件之前释放鼠标按键，则改变的状态保持不变，直至 VI 读取。简而言之，改变的时刻等于鼠标按下的时刻，保持的时间取决于 VI 何时读取。

（5）释放时触发

这种机械动作与"单击时触发"类似，差别在于改变的时刻是鼠标按键释放的时刻，何时恢复取决于 VI 何时读取控件。

（6）保持触发直至鼠标释放

这种机械动作，鼠标按键按下时立即触发，改变控件值。鼠标按键释放或者 VI 读取，这两个条件中任何一个满足，立即恢复原来状态。到底是鼠标释放还是 VI 读取触发的，取决于它们发生的先后次序。

布尔控件还具有一个独特的属性——布尔文本属性。布尔文本用文字的方式表示出当前布尔控件的状态，即真或假。很多布尔控件从颜色上可以区分真、假状态，有些则不然。布尔输入型控件向用户明确表明当前状态非常重要。通过显示布尔文本，用户可以准确理解控件的当前状态，从而选择对应的操作。

设置布尔文本的目的是显示"真/假"两种状态，但是在实际应用中，可以有多种描述方法，比如 ON/OFF、开/关、抬起/落下、升/降等。可以自由选择符合实际意义的文本描述，不过要特别注意和真/假的对应关系。

在布尔控件选板中，还存在确定、取消和停止三个常用的按钮。这三个按钮的属性基本相同。因为极其常用，所以单独列出，免掉了修改按钮外观和说明的麻烦。

3．个性化布尔控件

我们前面已经提到过控件的自定义问题。控件自定义功能的使用极其广泛，尤其是布尔型控件。在 GUI 制作中，经常需要独具特色的按钮或者开关，来模拟外部真实开关和按钮。这就需要我们根据要求自定义布尔控件。LabVIEW 提供了强大的控件自定义功能，从而使我们可以非常方便地制作特色控件。

先看一些特色按钮和指示灯的效果图，如图 1-44 所示。图 1-44 中所示的自定义按钮都是通过控件自定义功能创建的。自定义布尔控件最耗时的操作是图片的制作，一般可以通过专业图片处理软件来完成。有了数码相机之后，图片的制作相对容易多了，可以把真实的按钮和开关拍摄下来作为素材，再通过专业图片处理软件，做简单的处理就可以使用了。

图 1-44　自定义按钮效果图

一般布尔控件需要 4 张图片，分别为：① 布尔控件为 FALSE 的图片；② 布尔控件为 TRUE 的图片；③布尔控件从 TRUE 转换成 FALSE 触发瞬间的图片；④ 布尔控件从 FALSE 转换成 TRUE 的图片。通常情况下，只需准备表示 TRUE 和 FALSE 的两幅图片，①、③ 使用相同图片，②、④ 使用相同图片。下面以自定义按钮为例说明自定义布尔控件的详细过程。

step 1　准备图片素材。

step 2　启动控件编辑器。可以通过两种方式启动控件编辑器，一种方法是在菜单中选择“文件”→“新建”→“其他”→“自定义控件”项；另一种方法是在前面板上布尔控件的快捷菜单中，选择“高级”→“自定义”项。

step 3　在工具栏的控件类型下拉列表中选择输入控件，其他两项分别是自定义类型和严格自定义类型，通过工具条中的切换按钮切换到编辑模式，删除三维边框。现代控件，为体现三维效果，一般都包括一个边框对象作为装饰，不需要时可以删除这个装饰对象。然后导入图片，可以通过剪贴板导入图片，也可以直接从文件中导入。右击控件，通过快捷菜单，选择“从文件中导入图片”项，此时导入的是 FALSE 状态图片，然后依次导入 4 幅图片，其中，1、3 为代表 FALSE 的图片，2、4 为代表 TRUE 的图片。

step 4　建立图标，存储文件。类似于一般 VI 的制作，自定义控件存储在一个单独的文件中，具有自己的文件名和图标。使用自定义控件时，通过控件选板中的“选择控件”项，在文件对话框中找到对应的后缀为 CTL 的文件，打开后，即在 VI 的前面板中创建了这个控件。也可以通过“我的电脑”，找到对应的文件，直接将其拖动到 VI 的前面板。

4．单选按钮

单选按钮为常见控件，LabVIEW 把单选按钮归类在布尔控件的类别中。单选按钮本质上是枚举型控件，属于“多选一”方式，而布尔控件本身只有真/假两种状态。单选按钮是由多个布尔控件组合而成的，通过替换操作可以选择各种 LabVIEW 的布尔控件，如图 1-45 所示。

默认的单选按钮只包含两个，在快捷菜单上选择“添加单选按钮”项，可以增加按钮的数量。单选按钮允许使用不同外观，但是通常情况下使用的都是相同类型的按钮。首先用替换操作选择具有所需外观的按钮，隐藏不需要显示的项目，比如标题、标签、布尔文本等。然后拉大它的边框，选择要复制的按钮，采用按住 Ctrl 键+拖动的方式克隆按钮。

在单选按钮的快捷菜单中，还有一个重要的选项——允许不选。我们不需要选择单选按钮中的任何按钮的时候，就可以使用“允许不选”功能。

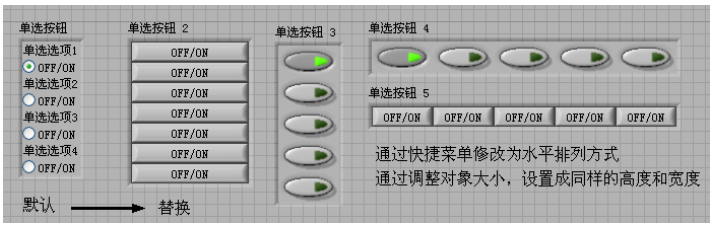

图 1-45　通过替换操作，选择单选按钮布尔控件

1.4.3 控件的通用编辑方法

控件的大部分属性都是通过快捷菜单设置的。控件由基本对象元素构成，比如标签和标题属于文本对象，各类装饰属于装饰对象。LabVIEW 允许改变文本的字体、字号、粗细、前景颜色和背景颜色等。

上述所有操作属于通用编辑操作，前面板上的控件和程序框图上的接线端子等都可以使用这些操作。控件的创建、复制、改变大小、对齐、排序等，前面已经提到了，也属于通用编辑操作。

1．文本的编辑方法

工具选板中专门设置有一个编辑文本的按钮，如果工具选板未出现，可以在菜单栏上选择“查看”→“工具选板”项，显示工具选板。按下 Shift 键，然后右击前面板或程序框图，显示临时工具选板。

学习笔记　按下 Shift 键，然后右击前面板或程序框图，可以显示临时工具选板。

选择工具选板中的“编辑文本”按钮，单击要编辑的文本，比如控件标签，则文本自动被选中，文字将反白显示。光标定位在单击处，这时可以添加或者替换文字。双击选取整个文本，此时若输入文本，则原文本整体被替换。

当工具选板处于自动状态时，双击文字所在区域，将自动选择整个文本。然后单击文字，光标自动定位到单击处。

最常见的方式是拖动选择文本，使用光标在文本上画框，可以选取部分或者全部文本。

学习笔记　工具选板处于自动状态时，双击文本，即可选取整个文本区域。

对于数字控件的数值编辑，既可以用“编辑文本”按钮，也可以用“操作值”按钮。使用光标在文本上画框选中一个或者几个数字。

学习笔记　对数值控件“值”的修改可以通过“操作值”按钮或“编辑文本”按钮来完成。

2．文本的外观设置

文本的外观包括字体、大小、样式和颜色，它们都可以在工具栏上进行设置，如图 1-46 所示。

图 1-46　工具栏的文本设置按钮

选中文本后，工具栏上将自动显示选中文本的字号和字体。非顶层 VI 的标签或者标题等，其文本外观可以保持默认设置。为了使 GUI 窗口的显示醒目，可以选择一些特殊的字体、字号和颜色。含有物理单位的标签，则可以用粗体显示名称。单位只要使用无格式字体即可。

学习笔记　选中文字后，按组合键“Ctrl - ”(即 Ctrl 键与减号键)可缩小字体，按组合键“Ctrl+”

可放大字体。

文字样式包括无格式、加粗、加下画线、斜体、加轮廓线等。文本的颜色也可以在工具栏上设置。如图 1-46 所示，在下拉列表中选择“颜色配置”项，即可打开颜色对话框，从而为文本选择合适的颜色。文字的背景颜色，则可以通过工具选板中的“设置颜色”工具按钮配置。

3．自由标签

LabVIEW 的自由标签类似于 VB 中的 Label 控件。它显示只读文字信息，通常用于装饰和注释说明，在前面板和程序框图中都可以使用。在系统控件中也有自由标签，它和现代控件选板上的自由标签有所不同。系统控件标签的背景色和窗体客户区是相同的，而自由标签在前面板中是透明的，在程序框图中是黄色背景。系统标签与自由标签的不同效果，如图 1-47 所示。

创建自由标签同文本编辑一样，都是通过工具选板中的“编辑文本”工具按钮进行的。在工具选板处于自动状态时，在前面板或程序框图的空白处双击，可以创建自由标签。

标签通常用于在前面板或程序框图中显示说明信息，相当于常规编程语言中的注释。新版本针对标签提供了一个非常有用的新功能，标签可以用箭头标记要说明的节点，比如函数、子 VI、等。老版本在整理程序框图时，标签与要说明的节点无法一一对应。

当鼠标移动到标签时，其右下角会出现箭头标志，拖曳箭头到需要标记处，即建立了对应关系。当移动所标记节点时，箭头自动移动跟踪标记，使用非常方便。在标签的菜单中，也新增了“标签关联至对象”选项，用于启动上述新功能。

在新版本中，针对标签提供了超级链接功能。如果在标签的快捷菜单中启用了“超级链接”功能，则标签中的有效网址自动变成蓝色并加下画线。运行时双击，即可在默认浏览器中打开该网页。图 1-47 演示了标签的超级链接功能，运行时双击可打开搜狐网。

学习笔记 工具选板处于自动状态时，双击前面板或程序框图，将自动创建自由标签。

4．修改颜色

LabVIEW 中对对象颜色的修改都是在颜色对话框中进行的，颜色对话框如图 1-48 所示。

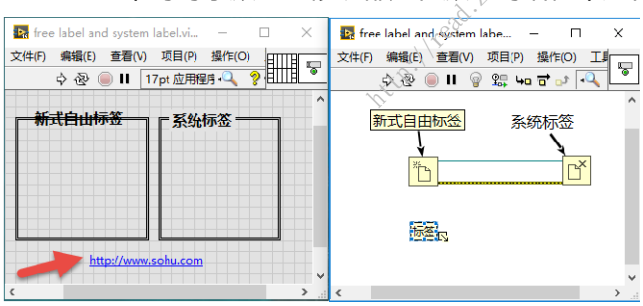

图 1-47　系统标签和自由标签

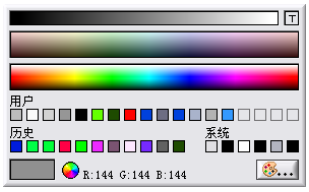

图 1-48　颜色对话框

颜色对话框从上到下分为多个部分。

- 顶层的颜色条由白色、黑色和灰色组成，主要用于区域较大部分的调色，比如前面板、装饰。单击右上角的按钮，当按钮显示文字“ T ”时，表示选择透明色；再次单击该按钮，按钮显示文字“ X ”时，表示不使用透明色。
- 第二个颜色条是亚暗色调，适用于中等大小的区域，比如控件等。
- 第三个颜色条是深色调，适用于比较小的区域，比如指示灯、曲线等。深色调是高亮的，只能用在特别需要引起注意的地方，不可滥用。否则，用户的注意力可能被吸引到无关

区域。

- 第四行为用户组颜色，设置的是 LabVIEW 本身对象常用的颜色。使用用户组颜色有利于界面设计的统一和协调。当光标移动到每个小的颜色框按钮上时，最下方 R、G、B 处提示该颜色适用的控件，比如指示灯、进度条等。在菜单栏上，选择“工具”→“选项”→“颜色设置”项，在属性页中可以设置用户组的颜色。
- 第五行为历史颜色，记录的是你最近使用的颜色。
- 左下方的是长方形颜色显示效果框。

学习笔记 光标在颜色选择框上移动时，选中控件颜色自动跟随其变化。单击右下方的按钮，可以打开系统颜色对话框。

控件的很多对象元素都是可以更改颜色的，如果希望采用和界面上已有对象同样的颜色，那么可以使用工具选板上的“提取颜色”按钮。该按钮外形似吸管状，单击它，然后在现有对象上提取颜色，即可为其他控件配置相同的颜色。

学习笔记 选择工具选板中“配置颜色”工具按钮后，按下 Ctrl 键，该按钮可以暂时变成提取颜色按钮，获取颜色后，再继续配置颜色。

1.4.4 字符串和路径控件

LabVIEW 以字符串输入控件和字符串显示控件的方式，提供了对字符串的支持。在常规语言中，很少有专门的路径类型，路径不过是特殊格式的字符串而已。在 LabVIEW 中，路径是一种专门的数据类型，同时和字符串存在密切的关系，二者之间可以自由转换。在 LabVIEW 的字符串和路径选板中，还包括组合框控件。组合框提供了预先定义的一组字符串，以供用户选择。

1．字符串控件

字符串控件是字符串数据的容器，字符串控件的值属性是字符串。如同其他类型控件一样，LabVIEW 的字符串控件也分为输入控件和显示控件。输入控件的值可以由用户通过鼠标或者键盘来改变，而显示控件则不允许用户直接输入，其只能通过数据流的方式，显示字符串信息。

LabVIEW 的字符串控件颇有特色，具有 4 种不同的显示方式，可以通过快捷菜单或者属性对话框设置，如图 1-49 所示。

在新版中，可以在字符串控件中显示样式。位于字符串左侧，如箭头所示，默认不显示该标志，从快捷菜单中可以选择是否显示。显示该项时，可以在运行时随意切换显示样式。

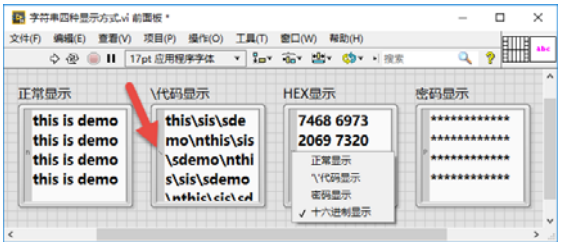

图 1-49 不同显示方式的效果展示

（1）正常显示

以字符的方式显示字符串数据，这是字符串默认的显示方式。对于不可显示的字符，则显示乱码。可显示字符也可称作可打印字符。

（2）使用“\”转义显示

不可显示的字符以反斜杠加 ASCII 十六进制的方式显示。对于回车、换行、空格等特殊字符，则采用反斜杠加特殊字符的方式显示。LabVIEW 支持的特殊字符如表 1-3 所示。

表 1-3　特殊字符的反斜杠表示

代　码	十六进制	十 进 制	含　义
\b	0x08	8	退格符号
\n	0x0A	10	换行符号
\r	0X0D	13	回车符号
\t	0x09	9	制表符号
\s	0x20	32	空格符号
\\	0x5C	92	“\”符号
\f	0x0C	12	进格符号
\oo：\FF			8位字符的十六进制值

在代码中，反斜杠“\”后的特殊字符必须是小写的，而对于“\ + ASCII HEX”，其中十六进制中的 A、B、C、D、E、F 必须是大写的。比如，“\02”表示输入 STX，“\1B”表示 ESC。

（3）密码显示

选择密码显示方式时，用户输入的字符在输入字符串控件中显示为星号，一般常用于登录对话框。此时输入的真实内容是字符，类似于正常模式，只是显示为星号而已。字符串控件支持复制、粘贴命令，如果在密码显示状态下，选择复制，则复制的是星号，而不是星号代表的字符。

（4）十六进制显示

以十六进制数值方式显示字符串，这种方式在通信和文件操作中，经常会遇到。图 1-49 展示了不同显示方式下字符串的不同效果。

在“\”转义显示方式中，空格和换行是特殊字符，分别为\s 和\n。

学习笔记 字符串控件通过回车换行，但是字符串内部未包含回车，只有换行符“\n”。

LabVIEW 在字符串控件快捷菜单和属性对话框中，还提供了几个非常重要的属性选项，如图 1-50 所示。

图 1-50　属性页和快捷菜单

1）限于单行输入

选中该项，将只允许输入一行文本，不响应回车换行操作。输入的回车被忽略，因此无法换行。

2）键入时刷新

选中该项，控件的值在输入每个字符时将同步刷新，不用等待回车换行。该项默认情况下未选中。未选中时，必须在结束输入时才产生字符串值改变事件。

学习笔记 通过单击前面板使字符串控件失去焦点，则 LabVIEW 将自动确认输入完成。

3）启用自动换行

默认情况下，自动换行功能是启用的。这样，当输入到字符串输入控件的行末尾时，将自动转到下一行。需要说明的是，这种换行只是显示上的，实际字符串中并没有真正换行。

如果不启用自动换行，则输入的字符始终单行显示。在快捷菜单上，选择“显示”→“水平滚动条”项，可以查看不在显示区域的行文本。启用自动换行时，“水平滚动条”项是不允许选择的。

结束输入有两种方法。

- 当执行文本输入操作时，工具条最左侧显示“确定输入”按钮，图标为对号。单击“确定输入”按钮后，按钮消失且文本输入被确认。
- 另外一个更方便的方法是单击前面板或前面板上其他控件，使字符串控件失去焦点，则 LabVIEW 将自动确认文本输入。

2．组合框控件

组合框控件可用来创建一个字符串列表。在前面板上可按次序浏览该列表。组合框控件类似于文本型或菜单型下拉列表控件。但是，组合框控件的值属性包含的是字符串型数据，而下拉列表控件是数值型数据。

在组合框控件上打开快捷菜单，选择“编辑项”选项，即可向列表中添加字符串供用户选择。组合框属性对话框的“编辑项”中的字符串顺序，决定了控件中的字符串顺序。默认状态下，组合框控件允许用户输入未在该控件字符串列表中定义的字符串。在组合框控件上打开快捷菜单，取消“允许未定义字符串项”的勾选，即可禁止用户输入未定义字符串。

如果在运行时向组合框控件输入字符串，LabVIEW 将即时显示以所输入字母开头的第一个最短的匹配字符串。如果没有匹配的字符串，则也不允许输入未定义的字符串，LabVIEW 将不会接收或显示用户输入的字符。

在配置组合框控件的字符串列表时，可为每个字符串指定一个自定义值，使前面板组合框控件中显示的字符串，与程序框图中组合框控件接线端返回的字符串不同。具体方法如下。

step 1 在组合框控件上打开快捷菜单，选择“编辑项”选项，打开组合框属性对话框。

step 2 在对话框的“编辑项”属性页中，取消“值与项值匹配”复选框的勾选。然后在该对话框表格的“值”列中，修改与控件中每个字符串对应的值即可。

3．路径控件

路径控件是 LabVIEW 提供的独特的数据类型，专门用来表示文件或者目录的路径。常规语言一般都是用字符串控件，附加一些特殊的格式来表示路径的。LabVIEW 的路径控件极大地方便了文件和目录的选择操作。

与字符串控件不同的是，路径控件包括一个“浏览”按钮。单击“浏览”按钮，将弹出文件选择对话框。在这里，可以选择相应的文件或者目录的绝对路径。

路径控件支持拖动操作，在计算机上找到文件或者文件所在文件夹后，直接拖动文件或文件所在的文件夹到路径控件，则路径控件会显示被拖动文件或者目录的绝对路径。

学习笔记 拖动文件或者文件所在文件夹到路径控件，路径控件将显示它们的绝对路径。

路径控件本身比较简单，但是 LabVIEW 对路径控件的“浏览”按钮专门提供了属性设置，如图 1-51 所示。

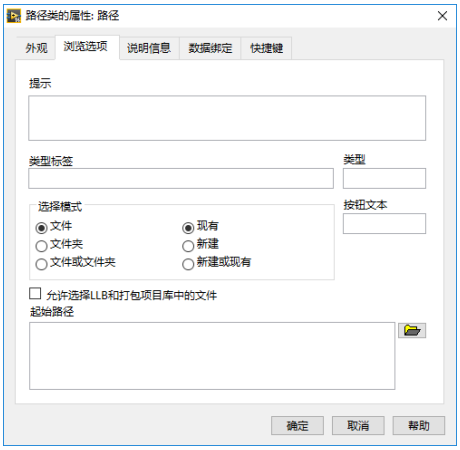

图 1-51 “浏览”按钮的属性设置

“浏览选项”属性页上包括很多可设置的选项，它们的具体作用如下。

（1）提示

在这里可以输入文件对话框的标题条。如果为空，则文件对话框标题栏显示为“打开”。

（2）类型标签

在这里可以设置文件类型匹配符，比如要选择 Word 文档，可以设置“类型标签”为“Word”，在右边设置“类型”为“*.doc”，则在对话框文件类型中将用 Word（*.doc）显示所有 doc 型文档。

（3）类型

在这里可以设置要选择的文件类型。全部文件（*.*）是内部存在的，不需要设置。若选择多种类型，则用分号隔开，注意不能有空格。例如，“*.doc；*.txt”，表示显示所有 doc 和 txt 型文件。

（4）选择模式

选择其中的某个单选框，可以设置打开文件或者文件夹，打开现有文件、文件夹，新建文件或文件夹。

（5）允许选择 LLB 和打包项目库中的文件

LLB 是 LabVIEW 特有的文件格式，可以把多个 VI 或自定义控件压缩存储在一个 LLB 类型的文件中。选择该复选框，将 LLB 作为文件夹后，可以选择其中包括的文件。不勾选该复选框，则只能选择 LLB 文件，而不能选择其中包括的文件。

（6）起始路径

在这里可以指定初始路径。如未指定初始路径，将默认使用最近打开的文件路径。指定该项后将显示指定初始路径下的文件和文件夹。

1.4.5 下拉列表与枚举控件

下拉列表与枚举控件从它包含的数据类型来说，属于数值控件。它们都是用文本的方式表示数值。下拉列表有多种表现形式，包括文本下拉列表、菜单下拉列表、图片下拉列表，以及文本与图片下拉列表。

在菜单下拉列表和文本下拉列表中，文字的输入可以通过快捷菜单中的“编辑”项进行。更简单的方法则是调用属性对话框，然后在“编辑项”属性页中设置。

图片下拉列表和文本下拉列表只能通过快捷菜单编辑。选择合适的项目后，可以从剪贴板导入图片，也可以从文件夹中直接拖动图片到图片下拉列表。文本下拉列表与图片下拉列表中的文字，则是通过工具按钮中的“编辑文本”按钮添加的。

下拉列表用文字或者图片的方式表示数字。数字可以是整型数，也可以是浮点数。既可以是有序的，比如从 0 开始递增的整型数；也可以是无序的，由用户自定义它代表的数字。

下拉列表上的各项，可以设置为启用或者禁用。如果设置为禁用，则该选择项目用灰色显示，不允许选择。下拉列表另外一个特有属性是“是否允许运行时有未定义值”，默认是未勾选的。在未勾选的情况下，只能选择设计好的条目。勾选时，将自动增添一个其他项。勾选该项，列表框

边上将出现一个数字框。在框中修改数字并回车，列表框将采用用户输入的新数值。

枚举控件与下拉列表控件非常相似。枚举控件只能代表整数，而且是有序、自动分配的。其中自定义枚举控件非常重要，广泛用于状态机中。

1.4.6 数组控件及其属性设置

数值型控件、布尔型控件和字符串型控件，它们的共同点是都包括一种基本的数据类型，如整数、浮点数和字符串等。基本数据类型在 LabVIEW 中也称作标量。

右击数组框架，可出现常见的控件选板。选择合适的输入控件或者输出控件，就建立了数组。也可以在前面板中先创建合适的控件，然后将其拖动到数组框架中。此时建立的数组只包括控件的类型，未包含任何实际数据。同时控件发灰显示，数组包含的元素长度为 0。

在新版本中，提供了创建数组的功能。先在前面板中创建一个标量，比如数值控件，在其快捷菜单中，选择“转换数组”项，则自动创建一个指定类型的一维数组。

沿着水平或者垂直方向拖动数组，同时显示多个数组元素。单击其中一个元素，则此时数组就包含了实际元素，包括被单击的元素与其前面的所有元素。实际元素不再处于发灰的状态，变成有效状态。

数组框架的左侧是数组的索引框。LabVIEW 的数组以索引号 0 表示数组中的首个数据，临近索引号的控件就是索引代表的数据。在图 1-52 中，索引号 99 所代表的数据就是离它最近的那个布尔控件，当前值为 TRUE。可以看出，该数组有 100 个元素，索引号为 0~99。

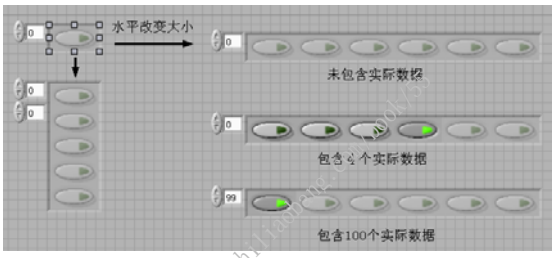

图 1-52 数组建立过程

学习笔记 数组中索引号代表的是与索引距离最近的元素。

通过上面的方式建立的数组是一维数组，而 LabVIEW 支持多维数组。增加数组的维数可以采用以下 3 种方法。

- 在索引框的快捷菜单中，选择“增加维度”项。
- 直接向下拖动索引框。
- 使用属性对话框增加维数。

相应的，删除维数，可以在索引框快捷菜单中，选择“减少维度”项；也可以直接向上拖动索引框。

学习笔记 通过拖动索引框，可以增加或者减少数组的维度。

数组控件的属性设置比较简单，主要是设置是否显示索引框，增加或者减少数组间隙。在数组元素较多的时候，可以显示水平或者垂直滚动条。滚动条是水平的还是垂直的取决于数组控件是水平显示还是垂直显示。多维数组可以同时显示水平和垂直滚动条。

数组中的元素可以是各种类型的控件，但是不能是数组的数组，也就是说，数组包含的元素不能是数组。数组中的元素除了它所代表的值以外，其他是完全相同的，具有同样的标签、标题，

及其他属性。

1.4.7 簇控件

簇和数组是 LabVIEW 最常见的复合数据类型，簇类似于 C 语言的结构和 VB 中的记录。在描述一个外部现象时，使用簇是必不可少的。各种编程语言都提供了类似于簇的数据结构，主要是因为这种数据结构能很好地实现数据的分类和分层。在此基础上，诞生了类的概念和面向对象的编程语言。可以说簇这种复合数据类型是 LABIVEW 的核心数据类型。

1．簇的创建

簇控件和数组控件位于同一个控件选板中，创建簇的基本方法和创建数组类似，具体操作如下。

step 1 单击控件选板中簇控件，随着鼠标的移动，出现簇的虚框；鼠标移动到合适位置后，单击前面板，簇的框架就建立起来了。

step 2 右击簇的框架，即出现控件选板。选择合适的控件作为簇的一个元素，将其加入至簇中，也可以首先在前面板创建所需的控件，然后将其拖动到簇框架中。

step 3 重复上面的过程，依次加入所需的元素。

需要注意的是，簇也分为输入控件和显示控件两种，至于到底属于哪种则取决于输入的第一个元素是输入控件还是显示控件。如果是输入控件，则整个簇变成输入控件。后续加入的元素即使是显示控件，也自动转换成输入控件，反之亦然。当然，整个簇可以通过选择快捷菜单上的选项，转换成输入控件或者输出控件。

2．簇的大小和排序

与数组不同的是，簇中的每一个元素都是相互独立的，具有自己的标签和标题。每个元素都可以通过各自的属性对话框修改相应属性。簇本身的属性非常简单，除了标签、标题、可见性等通用属性外，只具有几个特别的属性，例如快捷菜单“自动调整大小”和“重新排序簇中控件”中所包含的属性。

“自动调整大小”属性中包括 4 个选择项，分别是“无”、“自动匹配大小”、“水平排列”和“垂直排列”，如图 1-53 所示。选择“无”的情况下，簇的框架大小和控件分布完全由用户决定。若选择“自动匹配大小”，则控件的分布由用户决定，框架自动缩小，匹配控件。选择“水平排列”和“垂直排列”时，控件分布和框架的大小都是 LabVIEW 自动调整的。

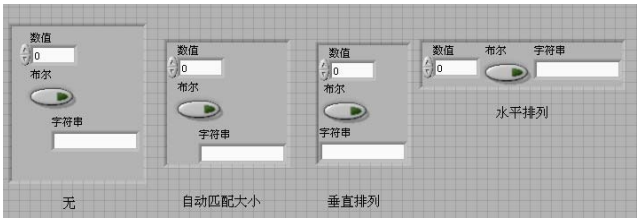

图 1-53　簇的调整方式及效果

当簇中元素比较多时，水平排列或者垂直排列时，所占前面板中的空间会比较大，此时需要多行多列排布。首先把簇中的元素适当分组，分组后再重新水平或者垂直分布，这样就构成了多行多列分布。

3．簇的逻辑次序

簇控件的一个极其重要的特性是逻辑次序。对于簇中的元素，根据添加元素的先后，最先加

入的元素的序号为 0，后面依次为 1、2 等。与簇相关的函数有些需要根据簇的序号操作，因此正确排序极其重要。簇的序号与元素控件的位置无关，只与生成次序有关。

通过簇的快捷菜单可以重新定义簇元素的内部次序。具体方法为，如图 1-54 所示，启动排序窗口，光标变成手形。按照 0、1、2……的次序分别单击簇元素，进行重排。单击工具条的“确认”按钮，将使修改生效。如果单击“取消”按钮，则放弃修改。

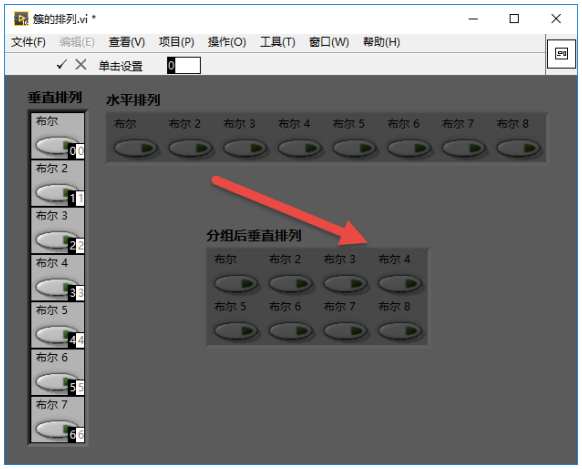

图 1-54 簇元素排序

4．簇的自定义

簇作为复合数据类型，常常用作一种数据结构。在多个 VI 中，通常需要采用同一种数据结构，然后利用数据流相互传递数据。这种情况下，簇的作用就非常明显了。我们可以在一个 VI 中创建簇结构，然后采用复制的方法，使各个 VI 采用相同的簇。

上述方法存在一个致命的缺陷：如果簇结构中需要增加一个或多个新元素，则每个子 VI 中使用的簇结构必须重新构造。

通过严格自定义簇控件就可以轻松解决这个问题。进行严格自定义簇控件后，簇的定义保存在单独的文件中，每个使用它的 VI 都和这个文件保持链接关系。因此，对这个文件的修改，将自动体现在各个使用它的 VI 中。这样就一次性地完成了整个修改。

学习笔记 程序中所有用到的簇结构，要尽可能地使用严格自定义方式。数组可以在运行中改变大小，而簇在运行中不能改变大小。

簇有着非常重要的作用，通过简单的簇结构，可以构造出复杂的簇结构，即簇的嵌套。下面通过创建雇员信息簇，说明如何构造自定义簇。

雇员信息可以大致分成两部分：与公司无关的信息，比如姓名、年龄、性别等；与公司有关的信息，比如职务、工资、部门、工作年限等。前者可以称为个人信息，后者称为基本信息。下面来建立个人信息簇和基本信息簇。

step 1 打开自定义控件编辑器。

step 2 个人信息簇的元素包括姓名、年龄、性别、家庭住址、身份证号、电话号码，输入完成后，建立图标，存储文件。

step 3 基本信息簇的元素包括职务、所属部门、工资、内部电话等，如图 1-55 所示。

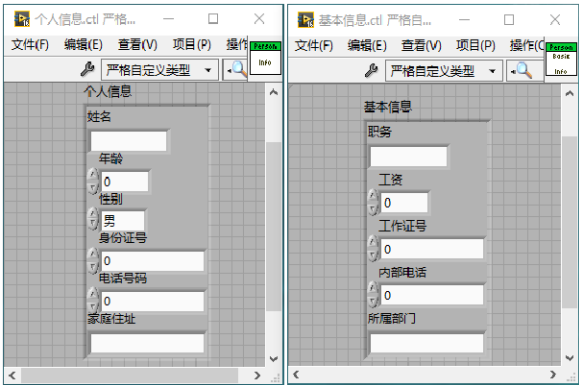

图 1-55　严格自定义类型的簇

step 4　建立雇员完整信息簇，完整信息簇中包含两个元素，分别为图 1-55 中所示的两个自定义簇。

1.4.8　时间标识控件与波形数据控件

时间标识控件和波形数据控件是 LabVIEW 中特殊的控件，这两类控件存在密切关系，波形数据控件中的开始时间元素就是时间标识控件。

1．时间标识控件

为了描述和使用系统时间，LabVIEW 专门提供了时间标识输入控件和显示控件。需要注意的是，时间标识控件与数值控件位于同一个控件选板中，这也说明时间标识控件所包含的数据本质上是数值类型。

时间标识输入控件从外观上看，非常类似于一般的数值输入控件。控件的左侧有增量和减量按钮。增减量默认情况下为 1，当光标定位在文本区域内的时、分、秒、年、月、日中的任何位置时，按照光标所处的位置，进行加 1 或者减 1 的操作。

时间标识输入控件的右侧是时间/日期浏览按钮。单击“浏览”按钮，弹出时间日期对话框。在时间日期对话框中可以设置时间标识输入控件所代表的时间。在快捷菜单或属性对话框中可以选择是否显示“浏览”按钮。

时间日期对话框使用起来非常简单。如果是新创建的时间标识输入控件，打开此对话框后自动定位到当前计算机系统日期。既可以通过文本编辑的方式修改时间，也可以通过鼠标选择年、月、日，设置完成后，单击“确认”按钮，所设置的时间和日期立即生效。

LabVIEW 内部使用双精度浮点数表示时间，单位是秒，同时根据当前的操作系统决定时区。新创建时间标识输入控件时，默认的双精度数是 0，代表的绝对时间是 1904 年 1 月 1 日上午 08:00:00。

学习笔记　数值 0 表示的中国标准时间是 1904 年 1 月 1 日早晨 8 点。

时间标识控件虽然使用起来比较简单，但是时间和日期的表现形式却非常丰富。通过时间标识控件的属性对话框，根据用户的需要，可定制时间标识的显示方式，如图 1-56 所示。

LabVIEW 包括的大量预先定义的时间格式，可以通过“时间标识”属性页直接选取。通过属性页中的“高级编辑”模式，还可以自定义时间和日期的显示方式，包括格式时间、日期字符串的自定义。

格式字符串以“%”开头，必须包括在时间容器中，具体格式如表 1-4 所示。

图 1-56 时间标识显示格式

表 1-4 时间字符串格式列表

名 称	格 式	说明与示例
绝对时间容器	%< >T	其他的格式字符串必须位于时间容器中，可以多行设置多个容器，实现时间标识的多行显示。示例：%<%H>T显示24小时制的小时数
星期名称缩写	%a	星期名称缩写，对于中文操作系统，与星期名称相同
星期名称	%A	以中文大写的方式显示星期数，比如，星期四
月份名称缩写	%b	月份名称缩写，对于中文操作系统，与月份名称相同
月份名称	%B	以中文大写的方式显示月份，比如九月、十月
第几天/日	%d	显示年、月、日中的日
默认日期格式	%c	按照计算机操作系统默认的格式显示日期
24小时制小时数	%H	在24小时制方式下显示小时数（01～23）
12小时制小时数	%I	在12小时制方式下显示小时数（01～12）
一年中的天值	%j	显示当前日期是这一年的第几天
月份	%m	显示当前日期中的月份 （01～12）
分钟数	%M	显示当前时间的分钟数（00～59）
AM/PM标志	%p	显示当前时间是上午还是下午，中文显示
秒数	%S	显示当前时间中的秒数（00～59）
小数形式的秒值	%<digit>u	小数点后显示的是毫秒数，digit表示小数点后的位数。例如，%5u
一年中的星期数	%U	显示当前日期是该年度的第几星期（0～53），星期日为每星期的开始
星期数	%w	以数值的方式显示星期几。比如，4表示星期四（0～6），0表示星期日
一年中的星期数	%W	与%U类似，区别是%W以星期一作为一星期的开始
本地日期格式	%x	本地日期显示方式。显示方式为：2008-9-11
长本地日期格式	%1x	显示方式为：2008'年'9'月'11'日'
长本地日期缩写	%2x	中文操作系统，与%1x 相同
本地时间格式	%X	本地时间显示格式为：HH：MM：SS，比如10：48：32
两位数年份	%y	以两位数方式显示年份，比如：2008 显示为08
四位数年份	%Y	以4位数的方式显示年份，比如：2008显示为2008
与通用时间时差	%z	中国标准时间与通用时间的时差为08：00：00

可对时间标识输入控件设置范围属性。例如，中国标准时间默认的输入范围是从 1994-1-1 08:00:00 开始，最大到 2038-1-19 11:14:07。

2．波形数据控件

波形数据是 LabVIEW 特有的数据类型，在数据采集卡采集外部物理量的过程中，一般按照采集卡内部设定的扫描时钟，等时间间隔逐次采集。描述这样一个采集过程，需要 3 个要素：起始时间、时间间隔和采集的数值。正是由于波形数据的特殊性，波形数据控件位于 IO 控件组中。

波形数据控件由 3 个控件组成：时间控件 to 表示开始时间，数值控件 dt 表示时间间隔，数值控件 Y 数组表示连续采集的数据。

通过波形数据控件，可以非常容易地计算出每个数据对应的时间点。第 i 个数据的时间点为 Ti=to+i*dt，i 表示数组 Y 的索引号。

1.5 小结

本章从 LabVIEW 最基本的控件开始，介绍了如何创建、编辑和调用 VI。VI 是 LabVIEW 最基本的概念，贯穿于 LabVIEW 编程的始终，尤其是其丰富的属性设置，使用起来极其灵活。

VI 的设计具有明显的层次结构，模块化编程是 LabVIEW 编程的突出特点。LabVIEW 编程的其他重要特征，比如数据流和多线程，本章只是简单提及，在后续章节将专门讨论。

在本章的后半部分，使用较大篇幅介绍了 LabVIEW 常规控件的用法，特别是数值型控件和布尔型控件，它们的使用非常广泛。除了基本控件之外，LabVIEW 还提供了大量的高级控件，这些将在后续章节介绍。

◆ ◆ ◆

第 2 章　LabVIEW 基本函数

第 1 章介绍了 LabVIEW 各类基本控件的创建和属性配置方法，本章将介绍 LabVIEW 的各类基本函数。

数据处理是 LabVIEW 编程的重要任务。LabVIEW 对数据的操作是通过各种基本函数实现的。与常规语言不同，LabVIEW 不存在专门的运算符，它的所有运算都是通过函数实现的。因此，掌握 LabVIEW 基本函数的用法，是 LabVIEW 编程者必须具备的技能。

在 LabVIEW 的帮助文件中，经常出现节点、函数、函数节点等术语。函数节点通常也称作函数，节点包括函数。LabVIEW 经常用节点的数量来统计 VI 的性能，所以了解节点的真正含义是非常有必要的。

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

- 函数：内置的执行元素，相当于文本编程语言中的运算符、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。
- 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。
- 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等。
- 公式节点和表达式节点：公式节点是可以直接向程序框图输入方程的结构，其大小可以调节。表达式节点是用于计算含有单变量表达式或方程的结构。
- 属性节点和调用节点：属性节点是用于设置或读取类属性的结构。调用节点是设置对象执行方式的结构。
- 通过引用节点调用：用于调用动态加载的 VI 的结构。
- 调用库函数：用于调用大多数标准库或 DLL 的结构。
- 代码接口节点（CIN）：用于调用以文本编程语言所编写的代码的结构。

2.1　必须了解的一些基本算术运算函数

LabVIEW 是一门独特的编程语言，它的基本概念很难与常规编程语言相对应。LabVIEW 中没有单独的运算符的概念，常规编程语言中的运算符在 LabVIEW 中等同基本算术运算函数。

2.1.1　基本运算函数

如图 2-1 所示，数值函数选板不但包含了加、减、乘、除等基本运算函数，还包含常用的高级运算函数，比如平方、随机数、常量和类型转换等。数值函数选板是最常用的函数选板，其中的很多函数都是多态的，允许多种类型的参数输入，所以必须仔细研究，灵活掌握。

学习笔记　多态函数的运用是必须掌握的基本技巧，要仔细理解体会。

数值函数选板中的函数对标量运算都是适用的，它的多态特点主要体现在对数组和簇的运算

上，使用起来极其灵活。

2.1.2 标量之间的基本运算

标量的运算包括加、减、乘、除、乘方等，运算的结果还是标量，如图 2-2 所示。

图 2-1　数值函数选板

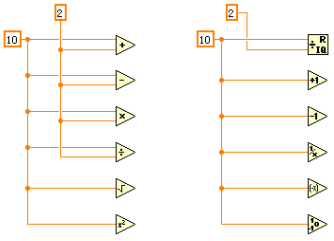

图 2-2　标量的基本运算

2.1.3 标量与数组的运算

标量与数组的运算，指的是对标量与数组中的每一个元素进行相应运算，运算结果是相同维数的数组。

图 2-3 所示的是标量常量与数组常量运算的例子，实际上是对标量常量与数组常量中的每个元素进行计算。计算的结果是新的数组。

图 2-4 所示的程序通过基本运算，对输入数组清零，然后对数组元素赋初值 5。

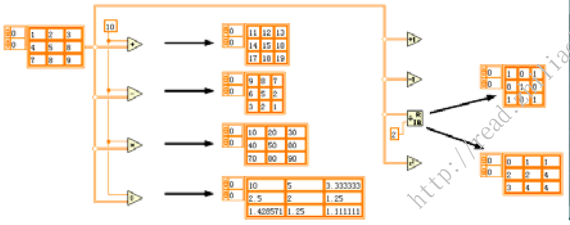

图 2-3　标量与数组的运算

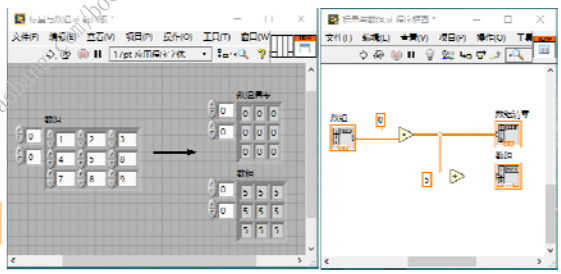

图 2-4　数组清零，并赋初值

2.1.4 数组与数组的运算

数组与数组的运算分为多种不同方式，本节将对这些方式进行详细的介绍。

1．相同维数、相同大小的数组运算

相同维数、相同大小的数组运算，就是对相同索引的数组元素进行相应运算，形成新的相同维数、相同大小的数组。运算后，数组的结构未发生变化，如图 2-5 所示。

2．相同维数、不同大小的数组运算

对于这种情况，首先根据较小的数组长度，对较大的数组进行剪裁操作，使两个数组具有相同的长度。然后对元素进行运算，形成新的数组，如图 2-6 所示。

学习笔记　在做相同维数、不同大小的数组运算时，要对较大的数组进行剪裁。

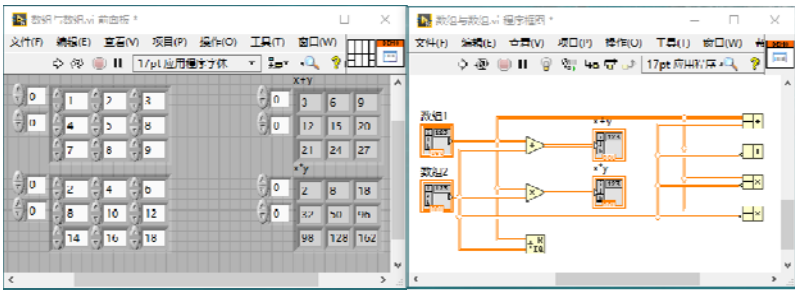

图 2-5 相同维数、相同大小的数组之间的运算

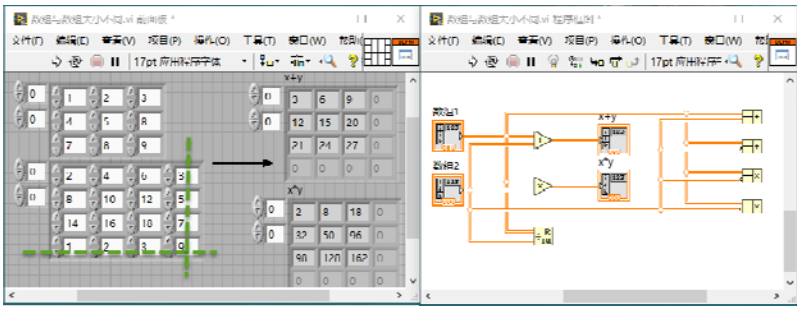

图 2-6 相同维数、不同大小的数组之间的运算

3．空数组

空数组是只有维数而长度为零的数组，不包含任何元素。建立一个新数组或者数组常量后，未用赋值工具给任何元素赋值时，该数组就是空数组。空数组的元素发灰显示，表示处于不可用状态。某些情况下空数组的运用十分重要。注意，对相同维数的数组与空数组进行运算，结果为空数组，如图 2-7 所示。不同维数的数组不允许进行运算操作。

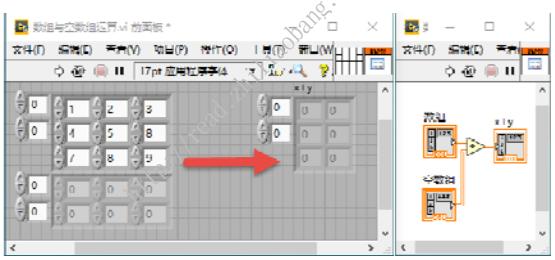

图 2-7 数组与空数组运算的结果为空数组

2.1.5 数组的函数

如图 2-8 所示，函数选板的"数组"分类中提供了大量的针对数组操作的函数。这些函数的功能十分强大，使用非常灵活，参数也有很多变化。同一问题，往往可以用多种函数解决。因此，仔细分析它们的用法非常重要。

1．"数组大小"函数

对于此函数，如果输入是一维数组，返回的则是 I32 数据，I32 的值表示一维数组的长度；如果输入是多维数组，则返回一个元素为 I32 类型的数组，数组中每一个元素表示对应维数的大小。通过计算返回的一维数组的长度，可以推算出当前数组的维数，如图 2-9 所示。

2．“索引数组”函数

数组中元素的寻址是通过索引实现的，各类编程语言都提供了数组的索引功能。LabVIEW 数组的索引是从 0 开始的。

图 2-8　数组函数选板

LabVIEW 的“索引数组”函数的使用非常灵活，既可以索引取出单个元素，也可以返回数组。比如对于二维数组，可以通过只连接行索引、禁用列的方式返回某一行；也可以通过只连接某列、禁用行的方式，返回某列，如图 2-10 所示。

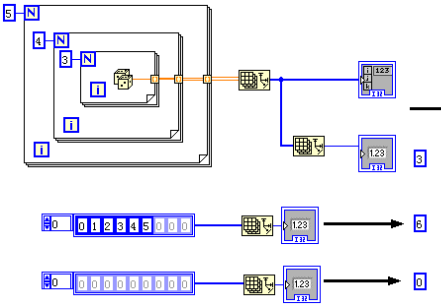

图 2-9 “数组大小”函数举例

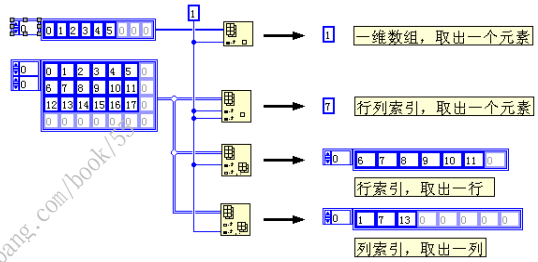

图 2-10 “索引数组”函数中索引的多种用法

如果需要索引取出几个连续或者间隔很小的数据，则可以使用顺序索引方式。以一维数组为例，连接索引值为 1 后，向下拖动索引数组，会自动增加新的输入/输出端。多维数组也支持这种顺序索引的方式，如图 2-11 所示。

对于数组只连接行，不连接列，则自动禁用列；只连接列，不连接行，则自动禁用行。

3．“替换数组子集”函数

在一维数组中，替换的可以是一个元素，也可以是一个子数组。此函数的“索引”输入端子表示开始替换的位置，如果不连接“索引”输入端子，则默认从 0 开始替换。如果索引号+子数组长度大于原数组长度，则只替换到末尾，多余部分将被忽略。

通过下拉接线端子，可以顺序执行多次替换，替换次序为从上至下，如图 2-12 所示。

在二维或多维数组中，可以进行元素替换、行替换、列替换和行列子集替换。如果替换部分超出原数组长度，超出部分将被忽略。不连接数组索引时，默认为“0”，如图 2-13 所示。

4．“数组插入”函数

将一个数组连接到此函数时，函数将自动调整大小以显示数组各个维度的索引。如未连接任何索引输入，则该函数将把新的元素或子数组添加到数组末尾。如果指定的索引超出原数组范围，则操作被忽略。

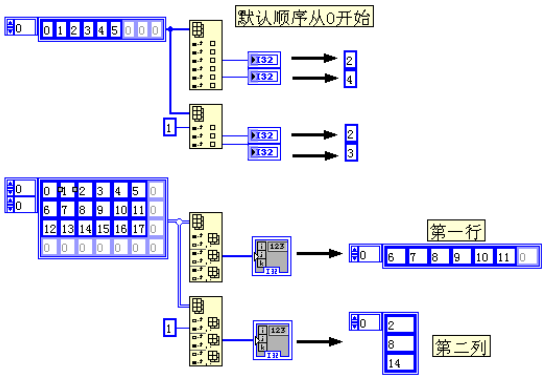

图 2-11　数组顺序索引

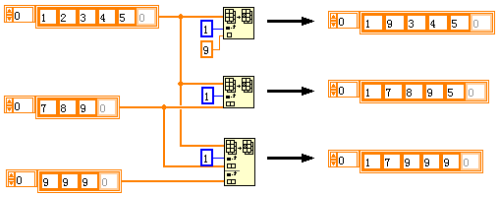

图 2-12　一维数组元素、子集顺序替换

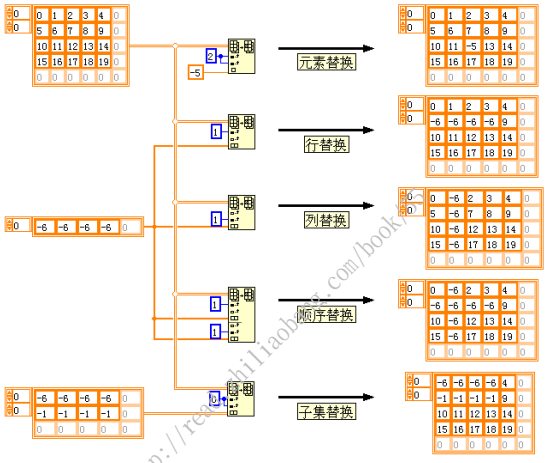

图 2-13　二维数组替换的几种方式

学习笔记　在数组插入操作中，如果未连接索引则自动将新增内容加至数组末尾。

一维数组的插入方法如图 2-14 所示。二维数组的插入方法如图 2-15 所示。

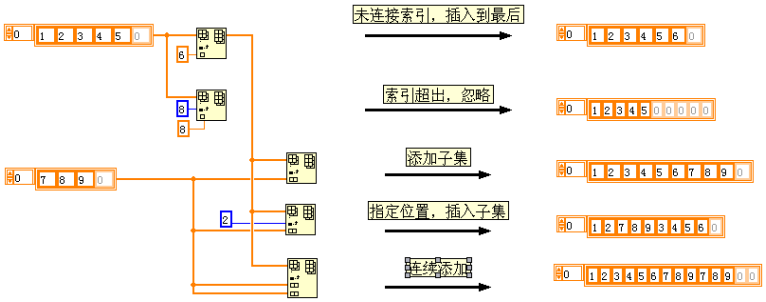

图 2-14　一维数组的插入方法

对于多维数组，每次只能改变一个方向的大小，不允许直接插入标量。如果插入的行或者列的长度加上索引位置大于原数组的行或者列的长度，则多余部分被忽略；如果小于原数组行或者列的长度，则以默认值补齐。总之，每次只能插入一行或者一列，新的行或者列的长度与原数组的行列长度相同。

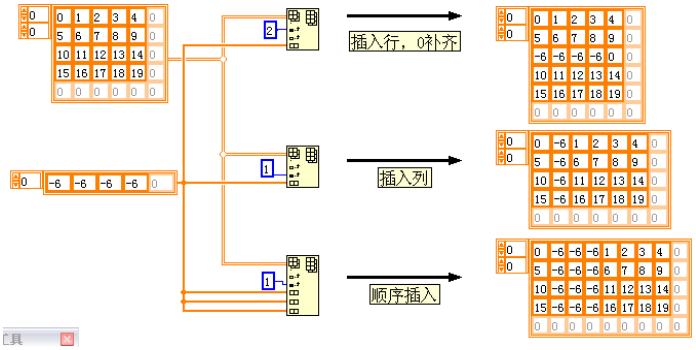

图 2-15　二维数组的插入方法

5．“删除数组元素”函数

该函数从数组中删除一个元素或子数组，输出端子将返回删除后的数组子集和已删除的元素或子集，参见图 2-16。

将一个数组连接到该函数时，函数将自动调整大小以显示数组各个维度的索引。未连接索引时，自动从数组末尾开始删除。

学习笔记　在数组的删除操作中，如未连接索引将自动从数组末尾开始删除。

在二维数组的删除操作中，只能删除指定长度的行或者列，不允许同时删除行和列，如图 2-17 所示。

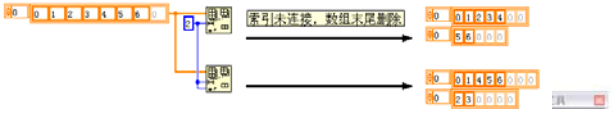

图 2-16　一维数组的删除操作

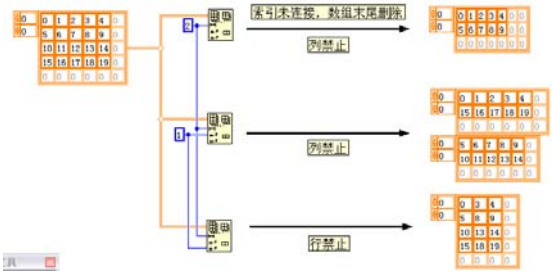

图 2-17　二维数组的删除操作

6．“初始化数组”函数

“初始化数组”函数输入端子包括“大小”和“初始值”两个，输出端子返回创建的数组。“大小”输入端子定义的是数组的长度，向下拖动“大小”输入端子可以增加维数。可以将数组的维数初始化为 0，如果维数为 0，则初始化后的数组为空数组，如图 2-18 所示。

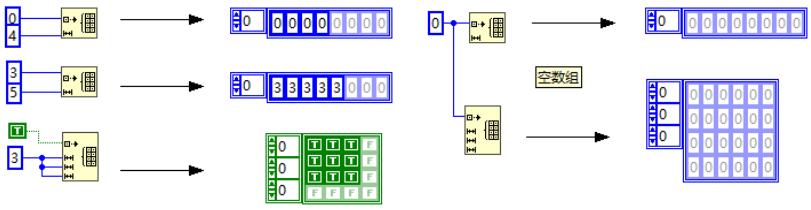

图 2-18 数组初始化

学习笔记 将数组维数设置为“0”，则生成空数组。

7．“创建数组”函数

该函数连接多个数组或者向数组中添加元素。将多个标量连接到函数的输入端子可以构建一个一维数组，如果连接到输入端子的数据为标量和数组，则实现的是数组元素添加操作。

下拉输入接线端子，可以增加输入端子的数量。对于标量和数组，自动采取添加方式，即非连接方式。对于数组和数组，可以选择连接方式和不选择连接方式，如果选择连接方式，对二维数组，实际是添加行的操作，参见图 2-19。

8．“数组子集”函数

该函数返回数组的一部分。输入端子指定开始索引号和长度，如果索引号大于数组实际长度，则返回同类型的空数组。如果长度为 0，则返回同类型的空数组，如图 2-20 所示。

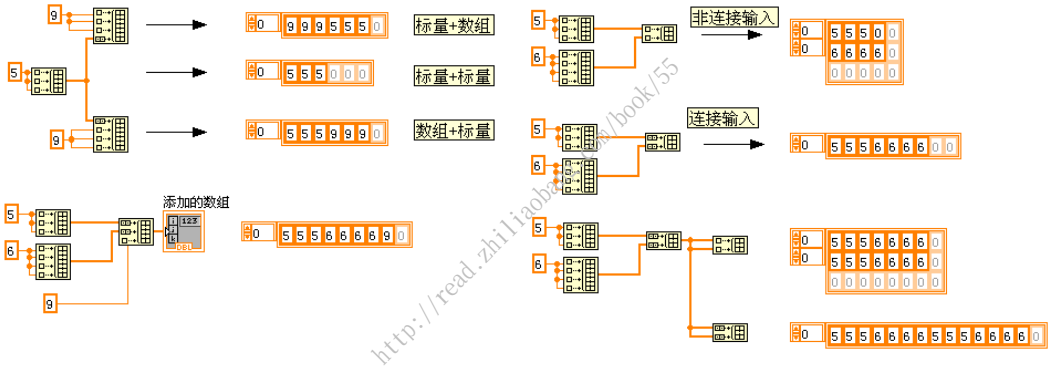

图 2-19 创建数组的几种方法

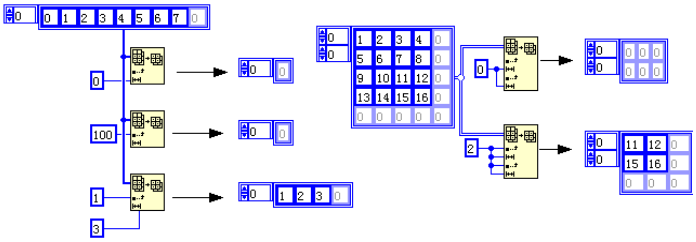

图 2-20 “数组子集”函数示例

学习笔记 在“数组子集”函数中，如果索引超出数组实际大小，或者实际长度为 0，则返回空数组。

9．“数组最大值与最小值”函数

该函数返回数组第一个最大值、最小值及其索引。对于多维数组，索引输出端子返回的为数

组，数组的元素表示对应维数的索引。

“数组最大值与最小值”函数为多态函数，可以接受多种数据类型作为输入。在图 2-21 中，分别演示了一维数组和二维数组如何求取最大值和最小值。其中二维数组索引输出端子返回的是一维数组，包括两个元素，分别是对应的行索引和列索引，如图 2-21 所示。

10．“重排数组维数”函数

该函数根据给定的维数和维数的大小，重新排列一维数组或者多维数组。如果给定数组的元素个数多于原来数组元素的数量，则用默认值补齐；反之，则原来数组多余的元素被舍弃；如果维数大小设置为 0，则返回空数组，如图 2-22 所示。

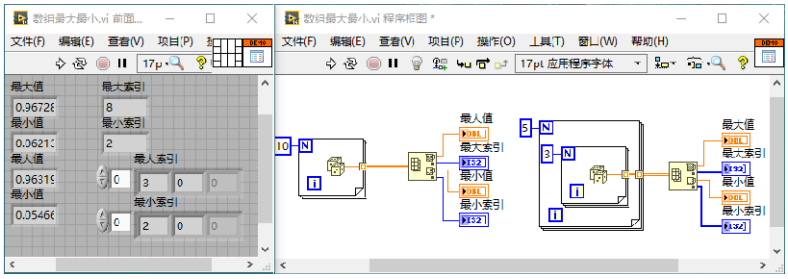

图 2-21　求数组最大值、最小值及其索引

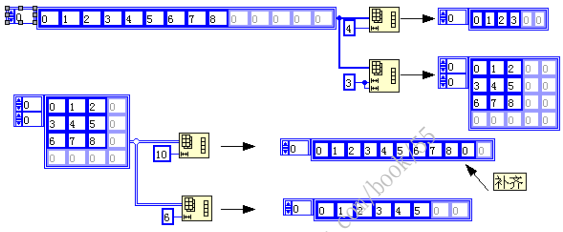

图 2-22　数组重排

11．“一维数组排序”函数

该函数返回元素按照升序排列的数组。如果数组的元素为簇，则该函数按照第一个元素的比较结果对元素进行排序。如果第一个元素匹配，函数将比较第二个和其后的元素，并进行排序。此函数的输入只能是一维数组，且只能按升序排列。如果需要降序排列，则可对升序数组反转，如图 2-23 所示。

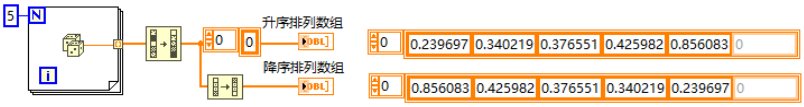

图 2-23　数组排序

12．“搜索一维数组”函数

此函数搜索一维数组中是否存在指定元素。如果存在，则返回元素的索引号，如果不存在，返回-1。通过“开始索引”输入端子指定搜索起始的位置，搜索到第一个符合条件的元素后，搜索立即停止。如果需要搜索多个或者全部符合条件的元素，则可以通过 While 循环实现，数组搜索的例子如图 2-24 所示。

“搜索一维数组”函数为多态函数，支持字符串数组。

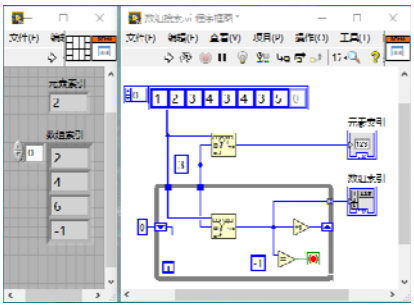

图 2-24 数组搜索示例

13．“拆分一维数组”函数

该函数以指定索引为界，把一维数组分解成两个一维数组。第一个子数组包括索引 0 到指定索引减 1 的所有元素。也就是说，指定索引的元素包含在第二个数组之中。

如果指定索引为 0，则第一个子数组是空数组，第二个子数组是原来的数组。如果指定索引大于数组最大索引，则第一个数组是原来的数组，第二个数组是空数组。此函数的用法如图 2-25 所示。

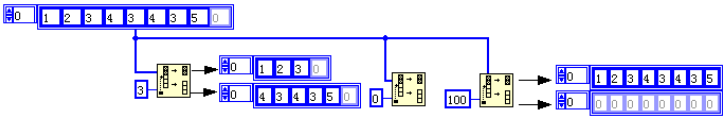

图 2-25 数组拆分示例

14．“反转一维数组”函数

“反转一维数组”函数非常简单，它反转所有元素的次序。比如数组{1，2，3，4}被反转后变为{4，3，2，1}。

15．“一维数组循环移位”函数

当输入参数 *n*>0 时，该函数将数组最后 *n* 个元素置于前端。当 *n*<0 时，该函数将数组前面 *n* 个数据置于后端，如图 2-26 所示。

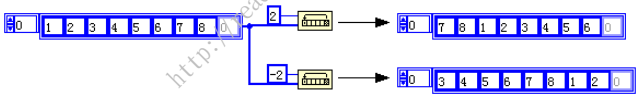

图 2-26 数组移位

下面的例子利用“一维数组循环移位”函数计算 A[i+2]–A[i]，形成一个新的数组，如图 2-27 所示。

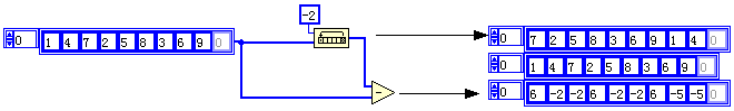

图 2-27 一维数组移位

16．“一维数组插值”函数

“一维数组插值”函数如图 2-28 所示。

一维数组插值采用线性插值的方式。当一维数组是数字时，X 端给定的是索引。当一维数组是点簇的时候，X 端给定的点是 X 坐标值，该函数的使用方法如图 2-29 所示。

图 2-28 “一维数组插值”函数

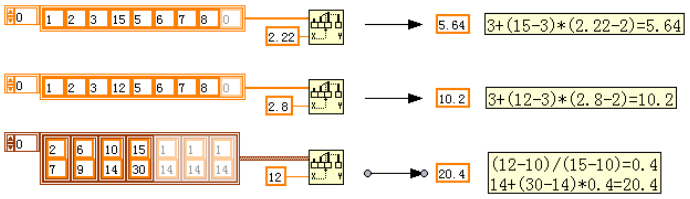

图 2-29 一维数组插值

17．“以阈值插值一维数组”函数

该函数实际是一维数组插值的逆运算，具体用法如图 2-30 所示。首先运用一维数组插值函数（这部分与图 2-29 相同），求取指定索引下的插值。然后通过阈值插值一维数组函数，进行逆运算，把插值的结果作为阈值，返回索引值。

18．“交织一维数组”函数

该函数输入端子连接的必须是一维数组。如果输入数组长度不同，则自动按最小长度截取成相同长度后，依次抽取一维数组中相同索引的元素合成一个新的数组。新的一维数组的长度等于一维数组最小长度乘以输入数组的数量，如图 2-31 所示。

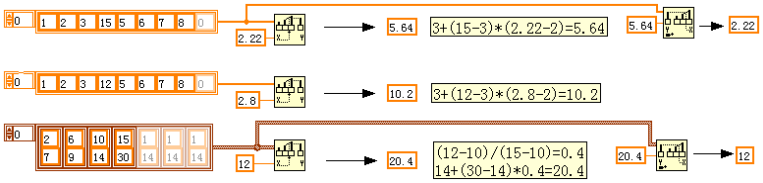

图 2-30 以阈值插值

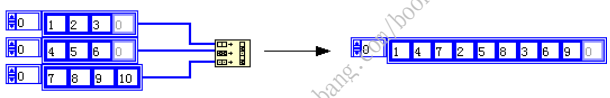

图 2-31 数组交织

19．“抽取一维数组”函数

抽取一维数组是交织一维数组的逆运算，它使数组的元素分成若干输出数组，依次输出元素，如图 2-32 所示。

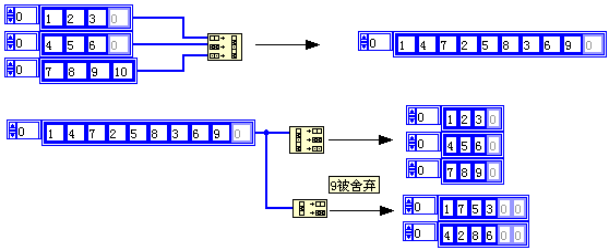

图 2-32 数组抽取

20．“二维数组转置”函数

二维数组转置就是重新排列二维数组。第一行变成第一列，第二行变成第二列，依次类推，如图 2-33 所示。

21．“数组至簇转换”函数

将一维数组转换成簇时，簇元素的名称将由数组名称和数组元素索引组合而成，簇元素的顺

序依赖数组索引的次序。

数组可以自由改变大小，而簇的大小是固定的。这就需要在转换之前，手动指定簇的大小。在"数组至簇转换"函数的快捷菜单中，可以指定簇的大小。簇的大小默认是 9，最大是 256。当数组大小小于簇大小时，用默认值填充；当数组大小大于簇大小时，多出的数组元素被忽略，如图 2-34 所示。

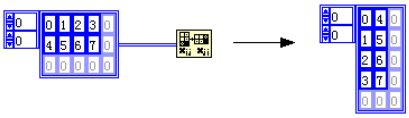

图 2-33 数组转置

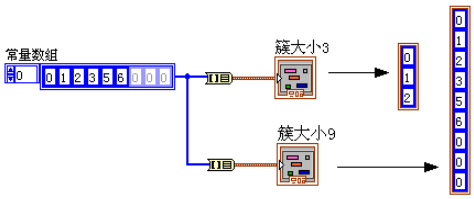

图 2-34 数组至簇的转换

学习笔记 将数组转换至簇时，必须预先在快捷菜单中设定簇的大小，默认大小为 9。

22．"簇至数组转换"函数

"簇至数组转换"函数要求被转换的簇元素类型必须相同，而簇被转换成与簇元素相同数据类型的数组。簇中的元素可以是簇，只要保证簇元素的类型相同即可。"簇至数组转换"函数的使用方法如图 2-35 所示，通过"簇至数组转换"函数，包含多个相同类型的簇元素的簇转换为对应类型的数组。

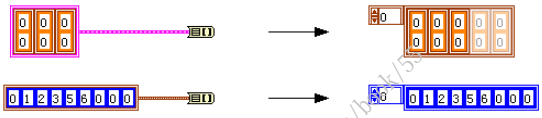

图 2-35 簇至数组转换

23．"矩阵至数组转换"函数

矩阵是 LabVIEW 8 新增的数据类型，而 MathScript 的引入极大地提高了 LabVIEW 的数据处理能力。"矩阵至数组转换"函数可以把矩阵转换成相同数据类型的数组。矩阵是多态的，可以是实数矩阵，也可以是复数矩阵。此函数的应用实例如图 2-36 所示。

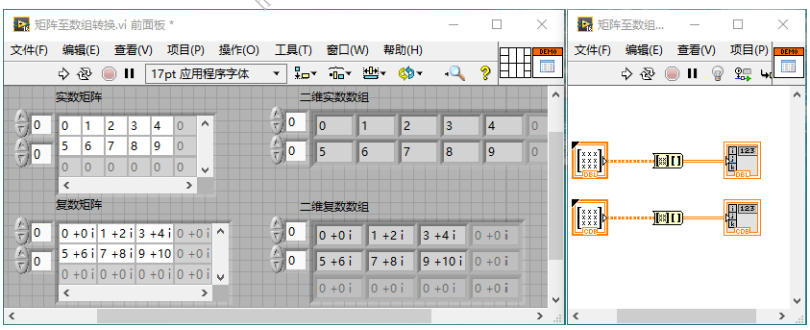

图 2-36 矩阵至数组的转换

24．"数组至矩阵转换"函数

"数组至矩阵转换"函数为"矩阵至数组转换"函数的逆操作。该函数为多态函数，如果输入的是一维实数数组，则转换结果为实数列向量。如果输入为一维复数数组，则转换结果为复数列向量。

2.1.6 标量与簇的基本运算

如果簇中的元素是数值型数据，则标量与簇的基本算术运算相当于对标量和簇的每个元素进行算术运算，并返回一个新的簇。

1．标量与簇的运算

标量与簇的运算示例如图 2-37 所示。标量与簇的运算相当于对标量与簇中每个元素进行运算，结果为一个新的簇。

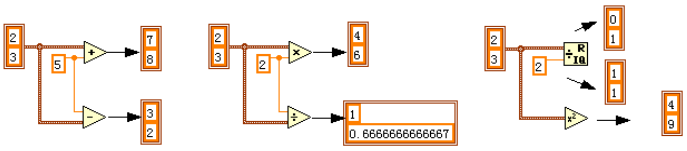

图 2-37　标量与簇的基本运算

学习笔记　标量与簇的运算，就是对标量与簇中的每一个元素进行运算。

2．标量与簇数组的运算

二维数组的元素为点簇，如果用一对标量构成的簇描述一个几何点，则点簇的数组就可以描述一条曲线。通过点簇的数组与标量运算，可以实现曲线的平移，如图 2-38 所示。

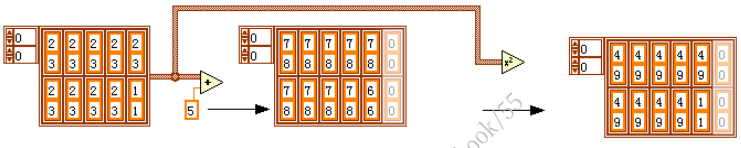

图 2-38　标量与簇数组运算示例

3．标量与嵌套簇的运算

如图 2-39 所示，簇中包括两个点簇，构成一个嵌套的簇，这个簇可以用来描述几何矩形。标量与嵌套簇的运算，是对标量与簇中所有元素进行运算。通过标量计算可以实现平移、缩放等。

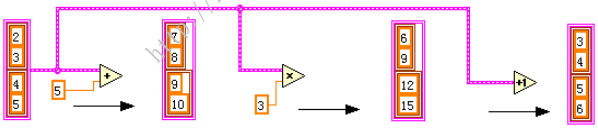

图 2-39　标量与嵌套簇运算示例

2.1.7 簇与簇的运算

相同类型的簇之间进行运算，实际上是对簇中对应元素进行相应运算，结果为与原来簇类型相同的新簇。同类型的簇与簇数组运算也是如此，如图 2-40 所示。

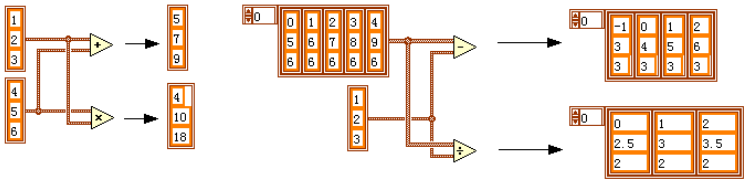

图 2-40　同类型簇的运算

不同类型的簇是不允许直接运算的。相同类型的簇是指两个簇中包含的对应元素类型相同，

而且两个簇中包含的元素个数，即簇的大小也完全相同。

2.1.8 簇的函数

标量与簇、簇与簇的基本运算是通过运算函数实现的，但是在很多场合中需要处理簇中一个或者几个特定元素。LabVIEW 在簇、类与变体函数选板中提供了有关簇的常用节点，它们可以处理簇中的特定元素。

我们知道，簇中的元素是具有独立标签的，标签代表簇中元素的名称。同时簇中的元素也是有次序的。因此，LabVIEW 提供了两种方法寻址特定的簇元素：按名称和按次序。

1．“按名称解除捆绑”函数

此函数按名称返回簇中的元素，可以同时选择多个名称，并返回多个簇元素，不需要关心次序问题。通常情况下，应尽可能地使用按名称解除捆绑，这样更直观，不容易出现错误。对于一个比较复杂，尤其是具有多种相同数据类型元素的簇，采用按次序寻址的方式很容易造成混乱。

如图 2-41 所示，这里的矩形簇用来描述一个矩形，它内部包含两个簇，分别是位置簇和边界簇。如果想直接取出嵌套簇的元素，并不需要层层解包，直接单击选取即可（如图 2-42 所示）。下拉或上拉接线端子可以增加新的元素，即可以同时处理多个簇元素。

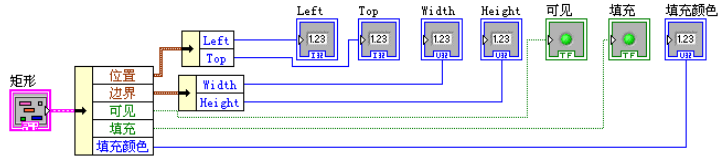

图 2-41　按名称解除捆绑，获取簇元素

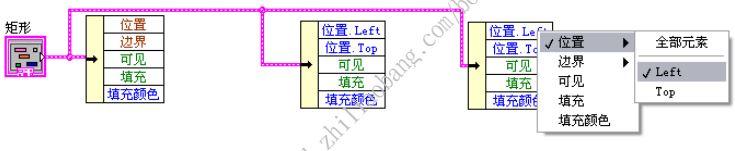

图 2-42　直接选取嵌套簇中的元素

2．“按名称捆绑”函数

该函数可以替换一个或多个簇元素。无论是捆绑还是解除捆绑操作，按名称操作簇元素都远比按次序要清晰得多，图 2-43 中使用的是“按名称捆绑”函数。后面的图 2-46 使用的是“捆绑”函数，“捆绑”函数需要按次序进行捆绑。两个程序框图实现的是相同的功能，显然使用“按名称捆绑”函数更为清晰。

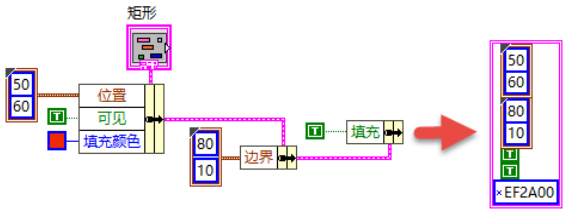

图 2-43　按名称捆绑，替换簇中的元素

捆绑操作实际是替换操作。通过函数选板生成的按名称捆绑簇，默认情况下是不含数据类型的，所以首先必须连接数据类型，这样 LabVIEW 才能正确识别这个簇，自动分析出簇中所包含的元素。

经常会遇到这种情况，我们在捆绑一个簇的时候，并没有输入控件，只有一个显示控件。显示控件是不能直接和“捆绑”函数的“类型”输入端子直接连接的，必须指定簇的类型，如图 2-44 所示。常规的方式是利用“局部变量”或者“常量连接类型”输入端子。另外，也可以用显示控件的值属性返回数据类型，不过这种方法不适合子 VI，因为属性节点将导致前面板被载入内存，效率较低。

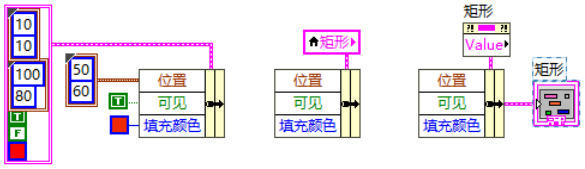

图 2-44　按名称捆绑显示控件

如果是一个非常大的簇，则它的常量会占据很大的空间。这种情况下，可以创建子 VI，由子 VI 导出簇常量。实际上，我们只关心簇的数据类型，至于其中包括的实际值则无关紧要。

学习笔记　在簇的捆绑和解除捆绑操作中，按名称方式操作簇元素是推荐使用的方式。

3．“解除捆绑”函数

该函数将一个簇分割为独立的元素。将簇连接到该函数时，函数将自动调整大小以显示簇中的各个元素并输出，簇中的元素是按照内部次序自动排列的，如图 2-45 所示。

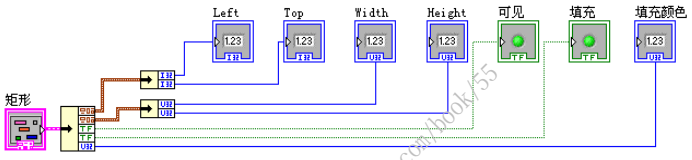

图 2-45　“解除捆绑”函数按次序获取簇中的元素

使用“解除捆绑”函数，LabVIEW 必须一次显示簇中包含的所有元素。这与按名称解除捆绑不同，按名称解除捆绑可以选择一个或者多个元素。另外，此函数中只显示元素的数据类型，如果数量比较多，则很难区分元素。

4．“捆绑”函数

“捆绑”函数将独立元素组合为簇。也可使用该函数改变现有簇中独立元素的值，而无须为所有元素指定新值，如图 2-46 所示。要实现上述操作，该函数中间的“簇类型”接线端子需要指明簇的类型。将簇连接到该函数时，函数将自动调整大小以显示簇中的各个输入元素。

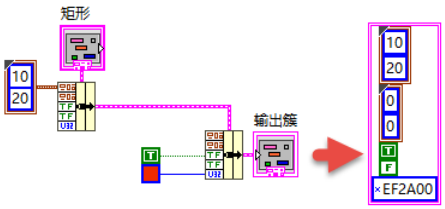

图 2-46　“捆绑”函数按次序进行捆绑

捆绑了新的元素后，则原来的元素将被替换。如果有未连接的端子，则其内部数值会保持不变。

5．“创建簇数组”函数

“创建簇数组”函数将每个输入对象捆绑为簇，然后将捆绑的簇组成簇数组。

特别需要注意，“创建簇数组”函数把输入参数捆绑成一个新的簇，并把这个簇作为元素来构建数组，而不是把这个输入参数本身作为元素。图 2-47 演示了两种方式的区别，同时演示了簇数组与普通数组的区别。

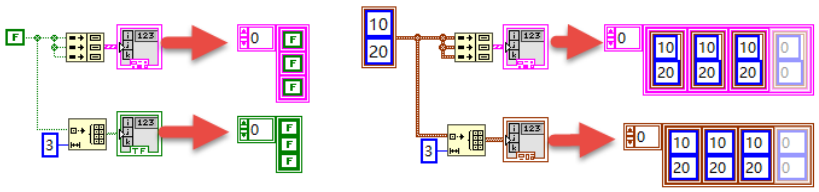

图 2-47　创建簇数组

6．“索引与捆绑簇数组”函数

“索引与捆绑簇数组”函数，用于对多个数组建立索引，并创建一个簇数组，其中第 i 个元素包含每个输入数组的第 i 个元素。

“索引与捆绑簇数组”函数的使用很广泛，特别适于绘制 *XY* 图或三维图形。三维曲线通常由三维点构成的簇数组来描述，簇包含三个元素，分别代表 *X*、*Y*、*Z* 坐标。实际测量的数据往往返回的是 *X*、*Y*、*Z* 的一维数组，通过“索引与捆绑簇数组”函数则可以将这些一维数组转换成三维点的数组，如图 2-48 所示。

图 2-48 “索引与捆绑簇数组”函数示例

2.2 必须了解的位运算函数和逻辑运算函数

所有与硬件联系比较密切的编程语言都支持位运算和逻辑运算。例如，C 语言不但有与、或、非等逻辑运算，还能进行按位与、按位或、按位非等运算。LabVIEW 也不例外，它的位运算和逻辑运算功能更加强大。

LabVIEW 采用多态函数的方式，综合了逻辑运算和位运算。至于到底要完成逻辑运算还是位运算，则完全取决于运算符的输入参数是布尔型还是整型。

2.2.1 常用逻辑运算函数

一般在包含布尔运算的软件中，把一个布尔变量设置成 True 的操作称作置位操作。反之，把一个布尔变量设置成 False 的操作称作复位操作，这里我们沿用这种说法。如图 2-49 所示，图中所示的就是使用常用逻辑运算函数，对输入的布尔数据进行置位和复位操作。

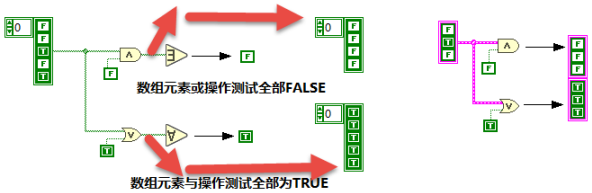

图 2-49　布尔数组、簇置位、复位操作

2.2.2 位运算

LabVIEW 的布尔函数是多态的，不仅可以进行逻辑运算，也可以进行位运算。一个 U8 型数据占据 1 字节的空间，可以表示 8 位。由低到高，通常称为 BIT0，BIT1，…，BIT7，对应二进制的每一位。通过位运算，可以对它的每一位进行置位和复位操作。

LabVIEW 中所谓的位操作与单片机中的位操作是不同的。单片机中的位，是可以单独寻址的，因此可以单独设置位。但是 LabVIEW 中的位操作，实际是通过整个字节进行操作的。根据需要，可保留其他位的原来状态，只改变需要的位的状态。

通常位操作包括置位、复位和位测试。置位是对指定的位置 1，复位是对指定的位置 0，位测试是不改变原来位的值，仅返回指定位的状态。

置位操作是使用按位或的方式实现的。例如，要对 BIT6 置 1，可以与 0x40 执行或运算。此时因为其他位是 0，或操作后保持不变；而 BIT6 与 1 或操作后，该位变成 1。

复位操作是采用按位与的方式实现的。例如，复位 BIT6，首先对 0x40 取非。这样除了 BIT6 为 0 外，其他位都为 1，然后与原来的数进行与运算即可。

位测试比较简单。例如，测试 BIT6 时，将要测试的数和 0x40 做与操作。如果结果为 0，则说明 BIT6 为 0；如果结果非 0，则说明 BIT6 为 1。

整数的位操作包括置位、复位和位测试，如图 2-50 所示。

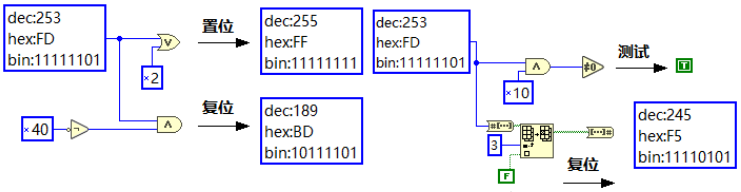

图 2-50　置位、复位和位测试

还有一种方法，不过相对比较麻烦，效率也低一些。这种方法就是把 U8 数转换成布尔数组，然后利用替换的方式，替换相应的位，再把布尔数组转换成 U8，从而实现置位、复位操作。

学习笔记　整数转换成布尔数组后，索引为 0 的元素表示最低位 BIT0。

2.2.3　深入理解复合运算函数

复合运算函数的功能非常强大，既可以进行算术运算，又可以进行逻辑和位运算。更重要的是，它支持多目运算，即可以输入多个参数。因此，在实际应用中，复合运算函数的使用较广泛。

可以使用复合运算函数进行算术运算，即加法和乘法等；也可以进行逻辑和位运算，即与、或和异或等。复合运算函数另外一个重要的功能是逆操作。通过快捷菜单选择逆操作的时候，“输入参数”端子处会出现一个黑色的圆圈，表示当前端子选择的是逆操作。通过下拉和上拉，可以增加节点的输入参数。

对于加、乘的算术运算，选择逆操作，实际就是进行减法和除法运算。对于逻辑运算和位运算，选择逆操作，实际是进行逻辑取非、对整数按位取反运算。复合运算函数是多态函数，其输入可以是标量、数组、簇等。利用复合运算函数可以实现连续的加/减或者连续的乘/除操作，如图 2-51 所示。

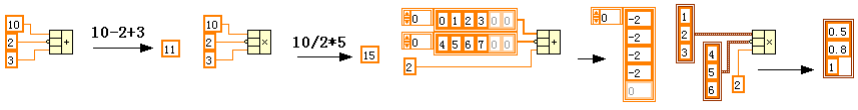

图 2-51　利用复合运算函数执行算术运算

虽然复合函数不支持字符串运算，但是字符串和 U8 数组是一一对应的，因此可以通过 U8 数组间接处理字符串。如果字符串选择正常显示方式，那么 U8 数组中保存的就是它的 ASCII 码。这样对 U8 数组进行处理后，再将 U8 数组转换成字符串，就间接地实现了字符串的位处理，比如用按位异或运算对字符串进行简单的加密处理。按位异或的特点是，当连续两次与同一个值进行异或操作时，将恢复原来的值。这在图形处理中非常常见。利用这个特点就可以对字符串或者文件进行简单的加密处理。将字符串转换成 U8 数组，可以实现字符串的位操作，如图 2-52 所示。

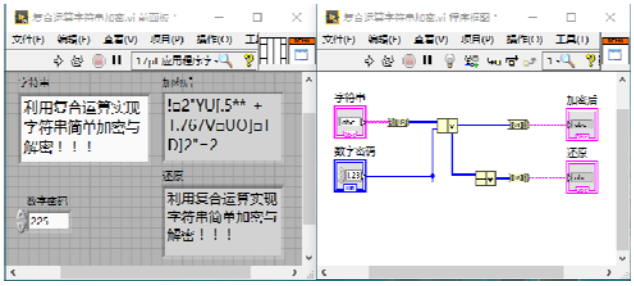

图 2-52　对字符串简单加密

138 译码器从原理上看就是由基本的逻辑门电路组合而成的，LabVIEW 这种强大的图形式编程语言也特别适合模拟逻辑电路，可以用 LabVIEW 逻辑运算简单地模拟 138 译码器的原理，如图 2-53 所示。

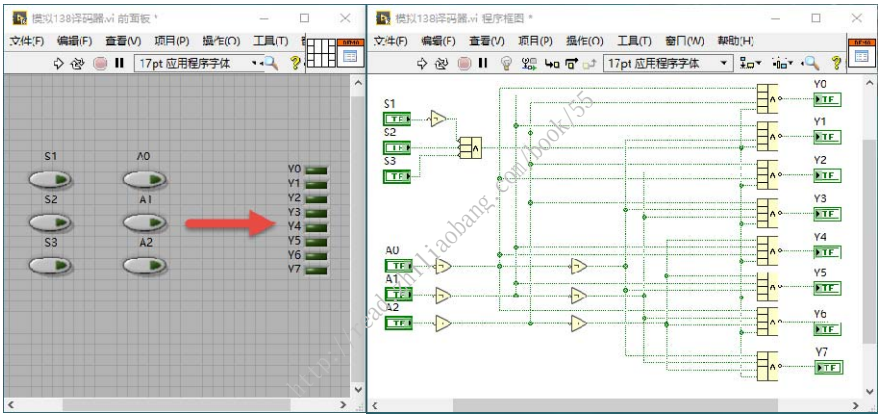

图 2-53　138 译码器原理模拟

图 2-53 所示是完全从 138 译码器内部原理来模拟。如果只想模拟其性能，利用位操作也很容易实现，如图 2-54 所示。

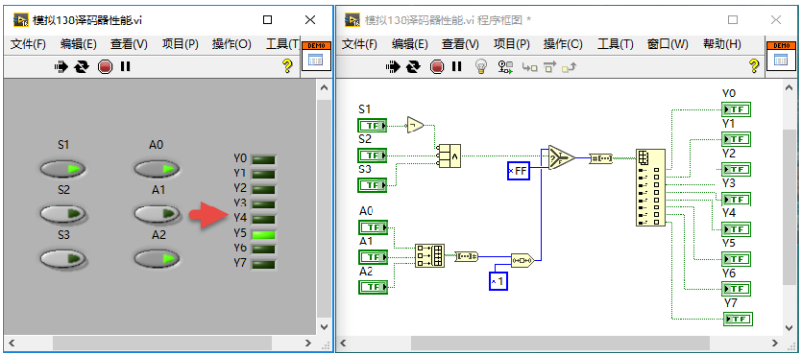

图 2-54　138 译码器性能模拟

2.3 必须了解的关系运算函数和比较函数

正如 LabVIEW 没有明确指定算术运算符一样，LabVIEW 中也没有明确的关系运算符的概念，所有的关系运算和高级比较函数都称为比较函数，它们都列在比较函数选板上。从比较函数选板不难看出，LabVIEW 对比较函数有一些基本的分类。这些类别为：基本关系运算符、o 关系运算符、字符函数和其他高级比较函数。

2.3.1 比较模式

LabVIEW 的多数比较函数都存在一个重要的选择项——比较模式。在比较函数的快捷菜单中，可以选择比较模式。比较模式分为“比较元素”和“比较集合”两种。

对于数组和簇这样的复合数据类型，如果选择“比较元素”，则最终结果是相同大小的布尔数组或者布尔簇。如果选择“比较集合”，则两个输入将作为整体比较，结果是一个布尔标量。比较集合时，要求两个输入类型完全相同。而比较元素没有这种限制，可以比较标量和数组、标量和簇等，如图 2-55 所示。

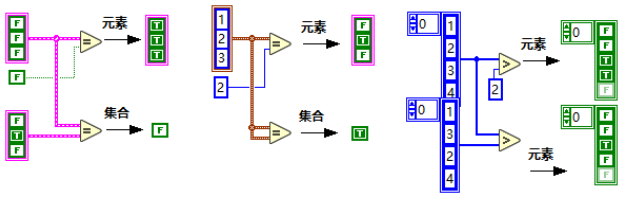

图 2-55 “比较元素”与“比较集合”

学习笔记 比较集合时要求输入类型完全相同，复合数据可以和标量进行元素比较。如果输入类型相同，但是长度不同的数组之间做比较，较长的数组将被截断为较短数组的长度。

另外，对于由元素构成的不同的簇，是不允直接比较的，直接比较会发生连线错误。

2.3.2 通用关系运算函数

通用关系运算函数包括等于、不等于、大于、小于、大于或等于和小于或等于 6 个函数。从编程语言的角度看，必须具备两个关系运算符——等于和大于，其他的运算符都是可以通过相应的逻辑运算得到的。LabVIEW 作为倾向于具体应用的编程语言，提供了如此之多的关系运算不过是为了便于编程。

这几个通用关系运算函数的用法极其相似，具有很多共同点，比如：

- 都是双目关系运算符，具有两个参数输入。
- 都允许设置比较模式，可以设置为“比较元素”或者“比较集合”。
- 比较的对象可以是标量与标量、标量与数组、标量与簇、数组与数组、簇与簇。簇与簇比较时，要求元素类型、大小必须完全相同。数组与数组比较时，较长的数组将会被截短，然后再进行比较。
- 都支持字符串的比较。比较字符串时实际是比较字符串所对应的 ASCII 数组。

使用通用关系运算符首先要考虑的是比较模式的问题。选择“比较元素”还是“比较集合”，完全取决于实际设计需要。例如，要检查一个数组是否发生了改变，可以用“相等”进行比较。如果选择“比较集合”的方式，就会从索引为 0 的元素开始比较。如果对应元素相等，则继续比较。一旦某个对应元素不同，则立即返回 False，不再对剩余元素进行比较，因此效率比较高。反之，如

果选择“比较元素”方式，则必须对对应的所有元素进行比较，然后再返回一个布尔数组，因此效率比较低。

学习笔记 选择“比较集合”的模式，运算效率比较高。

有些时候，必须选择“比较元素”的方式。例如，比较两个字符串时，我们需要知道字符串中发生变化的字符和字符所在位置，所以只能选择“比较元素”方式。再比如，如果需要取出数组中所有大于 5 的元素，首先必须确定数组中哪些是大于 5 的元素，哪些不是。类似的问题都必须采用“比较元素”的方式。

对于字符串的直接比较，LabVIEW 自动采用“比较集合”的方式，如图 2-56 所示。

对于字符串直接比较，LabVIEW 自动选择“比较集合”的方式，即使设置成“比较元素”的方式也枉然。如果必须用“比较元素”的方式，则可以先将字符串转换成 U8 数组，然后进行比较。

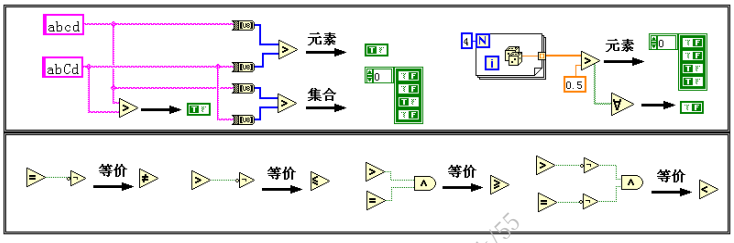

图 2-56 比较字符串

2.3.3 “比较 0”关系运算函数

“比较 0”关系运算函数是通用关系运算符的特殊形式，相当于一个参数输入为 0。“比较 0”函数自动采用“比较元素”的方式，因为标量是不允许和复合数据类型直接进行集合比较的，所以只能采用“比较元素”方式。LabVIEW 对“比较 0”函数根本没有提供比较模式的选项。

“比较 0”函数只接受数值标量、数值型数组、数值型簇和时间标识作为输入，不支持字符串，如图 2-57 所示。

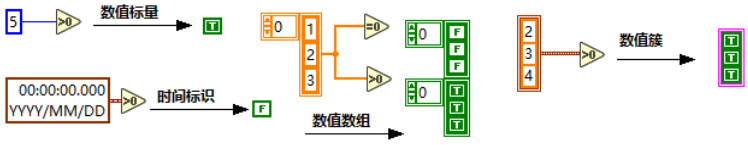

图 2-57 “比较 0”函数的参数形式

需要特别注意的是，同其他编程语言一样，LabVIEW 中的浮点数也存在精度损失的问题。浮点数只有 2 的 *N* 次方是可以精确表示的，比如 0.5、0.25、0.125 等。其他的数都是根据精度取一个邻近的尽可能精确的 2 的 *N* 次幂来表示。这样，浮点数计算的误差不可避免。在使用关系运算符判断等于或不等的情况下要特别注意，由于误差的问题，往往会得到错误的判断结果。

如图 2-58 所示，从数学的角度看，0.42−0.5 + 0.08 = 0 和 0.08 + 0.42−0.5 = 0 应该满足交换率，运算结果完全相同。但是在 LabVIEW 中，二者的计算结果却不满足交换率，前者不等于 0，而后者等于 0。

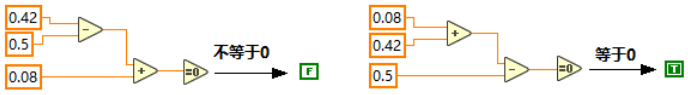

图 2-58　浮点数的比较错误

可以通过放大法和最小浮点数比较法解决上述问题，如图 2-59 所示。

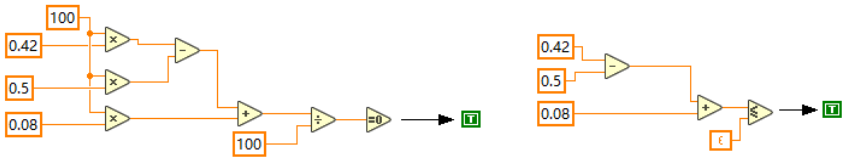

图 2-59　消除浮点数比较错误的两种方法

2.3.4 复杂关系运算函数

这一组函数可以实现比较复杂的比较功能，其并不局限于简单的比较，需要仔细研究体会。

1．“选择”函数

“选择”函数相当于 C 语言中的三目条件运算符。当输入端子 s 为 True 时，函数返回连接到真输入端子 t 的值。当 s 为 False 时，返回连接到假输入端子 f 的值。这里，t、s、f 分别是 true、select、false 的缩写。

“选择”函数是多态函数，选择端 s 只接受布尔输入，t、f 可连接各种数据类型，但是必须保证二者类型完全相同。t、f 可以连接数值、数组、簇、字符串、路径、时间标识、矩阵等。选择函数可以连接的数据类型，如图 2-60 所示。

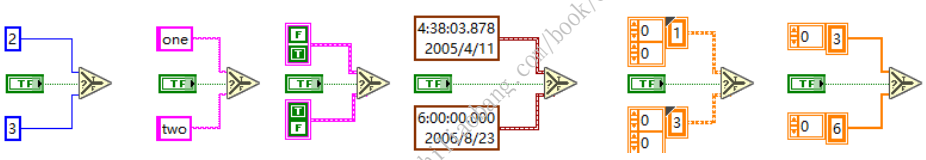

图 2-60　“选择”函数可以连接的数据类型

2．“最大值与最小值”函数

该函数返回 x、y 两个输入参数中最大和最小的值，顶部输出端返回最大值，底部输出端返回最小值。x、y 可以是各种数据类型，但必须保证二者是相同的数据类型。x、y 可以连接数值、数组、簇、字符串、路径和时间标识等。

学习笔记　此函数中有两个时间标识，以较新的时间为最大值，较早的时间为最小值。

“最大值与最小值”函数支持比较模式，默认选择“比较元素”方式，如图 2-61 所示。

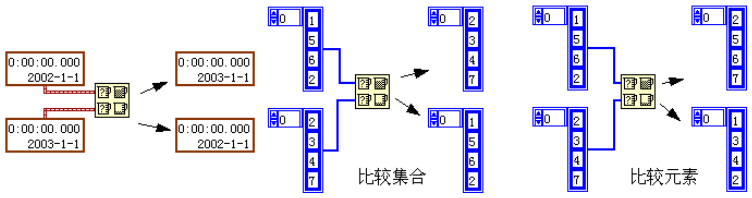

图 2-61　“最大值最小值”函数的比较模式

3．“判断范围并强制转换”函数

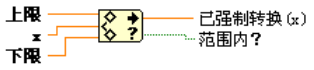

图 2-62　“判断范围并强制转换”函数

“判断范围并强制转换”函数如图 2-62 所示。

“判断范围并强制转换”函数非常复杂，它可以接受布尔、数值、字符串、路径、数组、簇、矩阵等多种类型的输入，同时又可以选择比较模式。

在该函数的快捷菜单中，可以选择是否包括上限和下限。当选择“包括”时，函数的图标中菱形框自动变成黑色菱形，表示上限或者下限被包括。

“判断范围并强制转换”函数包括两个重要的功能：测试输入是否在指定范围内，属于关系运算；强制转换，属于数据处理范畴。通常情况下，只使用其中一个功能。

该函数支持比较模式，对于复合数据类型，可以选择“比较元素”或者“比较集合”模式。与其他关系运算函数一样，在“比较集合”方式下，要求所有输入参数的上限、x、下限类型必须完全相同，即只能做数组和数组、簇和簇的比较。而对于“比较元素”方式，可以做标量和数组、标量和簇的比较，如图 2-63 所示。

学习笔记 对于标量和标量的关系运算，选择“比较集合”与选择“比较元素”是等价的。

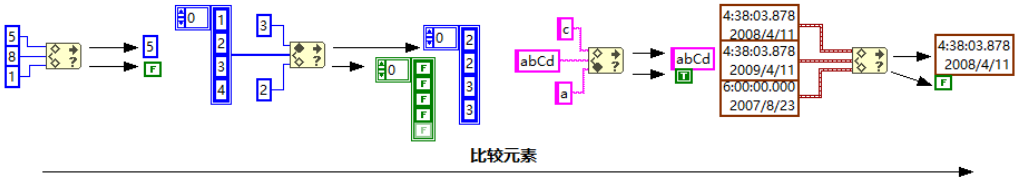

图 2-63 用“判断范围和强制转换”函数做比较

当输入的上限和下限完全相同时，可以对数组和数值型簇中的所有元素赋相同的值。这是非常有用的特性。对于数组操作，只有初始化数组时可以在完成定义数组的同时，为其中所有元素赋同样的值。对于已经创建的数组，可以采用元素替代的方法实现所有元素赋相同的值。此时使用“判断范围和强制转换”函数更为方便，不需要复杂的编程，如图 2-64 所示。

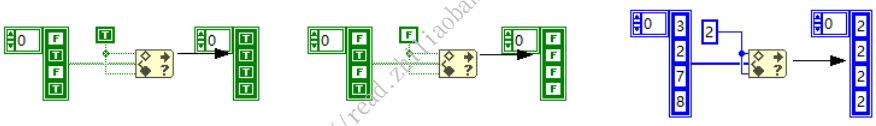

图 2-64 “判断范围与强制转换”函数的特殊用法

此函数在设计子 VI 时非常有用。在接收数据流传输过来的数据时，子 VI 的设计者必须检查数据的合理性，剔除不合理的数据。例如，一个人的年龄超过 500 显然是错误的，应该提出警告或者强制转换为合理数据。

4．“非法数字/路径/引用句柄”函数

该函数用来测试输入的数字、路径和引用句柄是否是合法数据。该函数的使用方法如图 2-65 所示。对于数值型数据，LabVIEW 定义了一个特殊的符号 NaN 来表示非法数字。NaN 的含义是数字无意义。

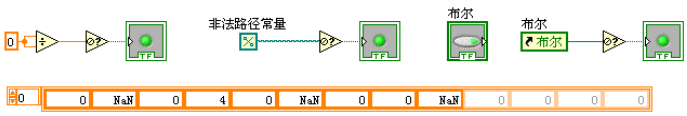

图 2-65 “非法数字/路径/引用句柄”函数示例

对于数值型控件或数组，可以直接输入字母 NaN，也可以通过计算得到。例如，计算 0/0，则 LabVIEW 返回 NaN。在绘图控件中，NaN 是不显示的。根据这个原理，可以对曲线分段显示，