21世纪高等学校规划教材

XVNI YIQI YUANLI JI YINGYONG

虚拟仪器原理及应用

林继鹏　茹　锋　编

中国电力出版社
http://jc.cepp.com.cn

21世纪高等学校规划教材

XVNI YIQI YUANLI JI YINGYONG

虚拟仪器原理及应用

林继鹏　茹　锋　编
巨永锋　主　审

中国电力出版社
http://jc.cepp.com.cn

内 容 提 要

本书为21世纪高等学校规划教材。

全书共分七章，主要内容包括LabVIEW概述、频谱与采样、经典滤波器的设计、变速率信号处理、最优化理论与非线性信号处理、信号的采集技术、虚拟仪器的总线技术等。书中全面系统地论述了虚拟仪器在信号处理及系统集成上的基本方法与基本理论。作为系统集成的首选开发环境，本书还详细地介绍了信号采集的基本知识和信号同步地实现方法，介绍了系统集成中的总线技术及总线技术的发展方向。本书提供了丰富的基础知识和实例，便于读者自学。

本书可作为高等学校自动化类及相关专业教材，也可供从事测试测量、系统集成的工程技术人员学习参考。

图书在版编目（CIP）数据

虚拟仪器原理及应用/林继鹏，茹锋编. —北京：中国电力出版社，2008

21世纪高等学校规划教材

ISBN 978-7-5083-8025-4

Ⅰ. 虚… Ⅱ. ①林…②茹… Ⅲ. 智能仪器—高等学校—教材 Ⅳ. TP216

中国版本图书馆CIP数据核字（2008）第157571号

中国电力出版社出版、发行

（北京三里河路6号 100044 http://jc.cepp.com.cn）

汇鑫印务有限公司印刷

各地新华书店经售

*

2009年2月第一版 2009年2月北京第一次印刷

787毫米×1092毫米 16开本 9.25印张 218千字

定价 **15.00**元

前　言

虚拟仪器是当前测控领域研究和使用的热点技术，是未来仪器发展的方向之一。

LabVIEW是美国国家仪器（National Instruments）公司推出的一款图形化编程语言（G语言），是一款通用的，面向非计算机专业人员从事测控工作的软件开发平台。于1986年10月诞生，是Laboratory Virtual Instrument Engineering Workbench的首字母缩写。

LabVIEW最大的优势是在测控系统开发方面，用户可以利用LabVIEW快速地开发出基于各种标准接口的测控工作平台，在不影响性能的前提下，开发速度可以提高10～15倍。而且LabVIEW还具有大量的信号处理分析模块，在科学计算、数据管理等方面，LabVIEW也一样可以开发出优秀的应用程序。近些年，LabVIEW在国内外发展迅速，不仅涌现出大量的成功案例，而且很多学校已经为此专门开设了虚拟仪器的课程。

本书编者多年来一直从事虚拟仪器的研究和开发工作，此期间得到了NI大量的支持和鼓励。本书是编者多年从事虚拟仪器相关工作的结果之一，其中也包含了作者的朋友、同事和学友的辛勤努力。书中附有大量的源代码，其中有些代码可能需要LabVIEW8.2以上版本的支持。

本书得到长安大学电子与控制工程学院院长巨永锋教授的认真审阅，他提出了很多宝贵的意见和建议，本书还得到汪贵平教授和刘涇教授的指点，在此一并向他们表示感谢。

虚拟仪器硬件资源多，技术发展快，应用领域广。鉴于作者的学识水平有限，书中难免存在不当之处，恳请读者批评指正。

编　者

2008. 7

目　录

第一章　LabVIEW 概述

第一节　LabVIEW 的概念

LabVIEW（Laboratory Virtual Instrument Engineering Workbench）是实验室虚拟仪器集成环境的简称，是美国国家仪器公司（National Instruments，简称 NI）的创新软件产品，也是目前应用最广、发展最快、功能最强的图形化软件集成开发环境。LabVIEW 诞生于 1986 年，目前最高版本是 LabVIEW 8.5。因为 LabVIEW 是一种图形化编程语言，所以又称作 G 语言。其编写的程序称为虚拟仪器 VI（Virtual Instrument），以 .VI 为后缀。

LabVIEW 具有多个图形化的操作模板，用于创建和运行程序。这些操作模板可以随意在屏幕上移动，并且可以放置在屏幕的任意位置。操作模板共有三类，为工具（Tools）模板、控制（Controls）模板和功能（Functions）模板。

一、工具模板（Tools Palette）

工具模板（见图 1-1）提供了各种用于创建、修改和调试 VI 程序的工具。如果该模板没有出现，则可以在 Windows 菜单下选择 Show Tools Palette 命令以显示该模板。当从模板内选择了任意一种工具后，鼠标箭头就会变成该工具相应的形状。当从 Windows 菜单下选择了 Show Help Window 功能后，把工具模板内选定的任一种工具光标放在框图程序的子程序（Sub VI）或图标上，就会显示相应的帮助信息。工具图标有如下几种：

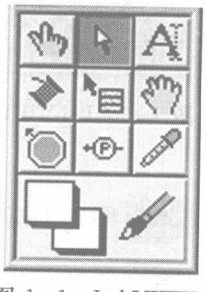


图 1-1　LabVIEW 工具模板

操作工具：使用该工具来操作前面板的控制和显示。使用它向数字或字符串中键入值时，工具会变成标签工具的形状。

选择工具：用于选择、移动或改变对象的大小。当它用于改变对象的边框大小时，会变成相应形状。

标签工具：用于输入标签文本或者创建自由标签。当创建自由标签时它会变成相应形状。

连线工具：用于在框图程序上连接对象。当联机帮助的窗口被打开时，把该工具放在任一条连线上，就会显示相应的数据类型。

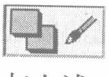
对象弹出菜单工具：用鼠标左键可以弹出对象的弹出式菜单。

漫游工具：使用该工具就可以不用使用滚动条而在窗口中漫游。

断点工具：使用该工具在 VI 的框图对象上设置断点。

探针工具：可以在框图程序内的数据流线上设置探针。程序调试员可以通过探针窗口来观察该数据流线上的数据变化状况。

颜色提取工具：使用该工具来提取颜色，用于编辑其他的对象。



颜色工具：用来给对象定义颜色。它也显示对象的前景色和背景色。

与上述工具模板不同，控制和功能模板只显示顶层子模板的图标。在这些顶层子模

板中，包含许多不同的控制或功能子模板。通过这些控制或功能子模板，可以找到创建程序所需的面板对象和框图对象。用鼠标点击顶层子模板图标就可以展开对应的控制或功能子模板，只需按下控制或功能子模板左上角的大头针就可以把对子模板变成浮动板留在屏幕上。

二、控制模板（Controls Palette）

用 LabVIEW 控制模板（见图 1-2）可以给前面板添加输入控制和输出显示。每个图标代表一个子模板。如果控制模板不显示，可以用 Windows 菜单的 Show Controls Palette 功能打开它，也可以在前面板的空白处，点击鼠标右键，以弹出控制模板。控制模板包括如图 1-2 所示的几个子模板。

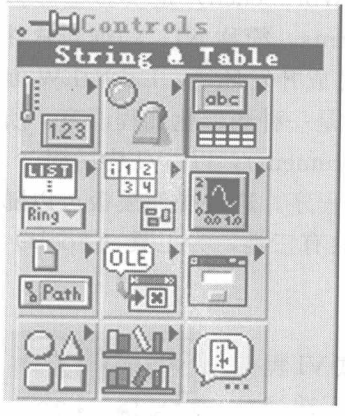


图 1-2 LabVIEW 控制模板

数值子模板：包含数值的控制和显示。


布尔值子模块：逻辑数值的控制和显示。

字符串子模板：字符串和表格的控制和显示。

列表和环（Ring）子模板：菜单环和列表栏的控制和显示。


数组和群子模板：复合型数据类型的控制和显示。


图形子模板：显示数据结果的趋势图和曲线图，包括 3D 图形。

路径和参考名（Refnum）子模板：文件路径和各种标识的控制和显示。

控件容器库子模板：用于操作 OLE、ActiveX 等功能。


对话框子模板：用于输入对话框的显示控制。


修饰子模板：用于给前面板进行美化的各种图形对象。


用户自定义的控制和显示。


调用存储在文件中的控制和显示的接口。

三、基本功能模板（Functions Palette）

功能模板是创建框图程序的工具。该模板上的每一个顶层图标都表示一个子模板。功能模板如图 1-3 所示。若功能模板不出现，则可以用 Windows 菜单下的 Show Functions Palette 功能打开它，也可以在框图程序窗口的空白处点击鼠标右键以弹出功能模板。


结构子模板：包括程序控制结构命令，例如循环控制、全局变量和局部变量等。

数值运算子模板：包括各种常用的数值运算符，如＋、－等；各种常见的数值运算式，如＋1 运算；还包括数制转换、三角函数、对数、复数等运算，以及各种数值常数。

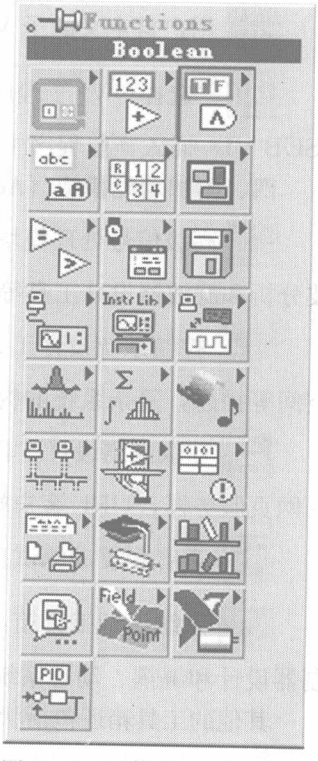


图 1-3　LabVIEW 功能模板

布尔逻辑子模板：包括各种逻辑运算符以及布尔常数。

字符串运算子模板：包含各种字符串操作函数、数值与字符串之间的转换函数，以及字符（串）常数等。

数组子模板：包括数组运算函数、数组转换函数，常数数组等。

群子模板：包括群的处理函数以及群常数等。这里的群相当于 C 语言中的结构变量。

比较子模板：包括各种比较运算函数，如大于、小于、等于等。

时间和对话框子模板：包括对话框窗口、时间和出错处理函数等。

文件输入/输出子模板：包括处理文件输入/输出的程序和函数。

仪器控制子模板：包括 GPIB（IEEE 488、IEEE 488.2）、串行、VXI 仪器控制的程序和函数，VISA 的操作功能函数。

仪器驱动程序库：用于加载各种仪器驱动程序。

数据采集子模板：包括数据采集硬件的驱动程序以及信号调理所需的各种功能模块。

信号处理子模板：包括信号发生、时域及频域分析等功能模块。

数学模型子模块：包括统计、曲线拟合、公式框节点等功能模块，数值微分、积分等数值计算工具模块。

图形与声音子模块：包括 3D、OpenGL、声音播放等功能模块。

通信子模板：包括 TCP、DDE、ActiveX 和 OLE 等功能的处理模块。

应用程序控制子模块：包括动态调用 VI、标准可执行程序的功能函数。

底层接口子模块：包括调用动态连接库和 CIN 节点等功能的处理模块。

文档生成子模板。

示教课程子模板：包括 LabVIEW 示教程序。

 用户自定义的子 VI 模板。

 “选择…VI 子程序”子模板：包括一个对话框，可以选择一个 VI 程序作为子程序（SUB VI）插入当前程序中。

四、附加功能模板（Addons）

 高级信号处理模块：包括联合时频分析（JTFA）、高分辨率谱分析（SRSA）、小波分析和滤波器设计工具等。

 控制系统设计工具箱：包括 PID 控制、模糊控制设计器以及系统传递函数、状态空间等描述，复杂系统的构建与仿真等。

 互联网络设计工具箱：包括 XML 文档的创建、编辑和访问，CGI 的建立和管理，G 网页服务器，URL 客户端，Email 收发客户端程序，FTP 和 Telnet 功能等。

 声音和震动测试工具箱：包括声音和震动信号的产生、测量、处理等。

 系统识别工具箱：包括给予模型的控制系统设计、数据分析和建立动态系统、控制器设计和解耦、动态系统仿真等功能。

其他的工具箱还包括状态机设计工具箱、OAT（Order Analyzing Toolkit）工具箱、Vision 工具箱、Database Connectivity 工具箱、SPC（Statistic Processing Toolkit）工具箱等。

第二节 LabVIEW 基本语法

一、数据类型

基本数据类型：数字型（Numeric）、布尔型（Boolean）、字符串型（String）。

构造数据类型：数组（Array）、簇（Cluster）。

其他数据类型：枚举（RefNum）、空类型。

二、数组（Array）

索引号从 0 开始，包括：一维数组（1D，列或行向量），二维数组（2D，矩阵）。

1. 组成

数据类型、数据索引（Index）、数据。

2. 创建

（1）控制模板—>Array & Cluster 子模板；

（2）根据需要将相应数据类型的前面板对象放入数组框架中。

3. 使用

（1）Array Size 返回输入数组的长度；

（2）Index Array 返回输入数组由输入索引指定的元素；

（3）Replace Array Element 替换输入数组的一个元素；

（4）Array Subset 从输入数组取出指定的元素子集；

（5）Reshape Array 改变输入数组的维数；

（6）Initialize Array 初始化数组；

（7）Build Array 建立一个新数组；

（8）Rotate 1D Array 将输入数组的最后 n 个元素移至数组的最前面；

（9）Sort 1D Array 将数组按升序排列；

（10）Reverse 1D Array 将输入的 1D 数组前后颠倒，输入数组可以是任何类型的数组；

（11）Transpose 2D Array 转置输入的二维数组，也叫矩阵转置；

（12）Search 1D Array 搜索指定元素在一维数组中的位置；

（13）Array Max & Min 返回输入数组中的最大值和最小值；

（14）Split 1D Array 将输入的一维数组在指定的元素处截断，分成 2 个一维数组；

（15）Interpolate 1D Array 对一维数组进行线性插值；

（16）Threshold 1D Array 一维数组阈值，是线性插值的逆过程；

（17）Interleave 1D Arrays 将从输入端口输入的一维数组插入到输出的一维数组中；

（18）Decimate 1D Array 将输入的一维数组分成若干个一维数组，是 Interleave 1D Arrays 的逆过程。

三、簇（Cluster）

簇类似于 Pascal 语言的 record 或 C 语言的 struct 结构变量。

1. 组成

不同的数据类型。

2. 创建

控制面板—>Array & Cluster 子面板，向框架添加所需的元素，根据需要更改簇和簇中元素的名称。

3. 使用

（1）Unbundle 解包，获得簇中元素的值；

（2）Bundle 打包，将相互关联的不同数据类型的数据组成一个簇，或给簇中的某个元素赋值；

（3）Unbundle By Name 按名称解包，获得由元素名称指定簇中相应元素的值；

（4）Bundle By Name 按名称打包，将相互关联的不同数据类型的数据组成一个簇，或给簇中的某个元素赋值；

（5）Build Cluster Array 建立簇的数组；

（6）Index & Bundle Cluster Array 将输入数组的元素按照索引组成簇，然后将这些簇组成一个数组；

（7）Cluster To Array 将簇转化为数组；

（8）Array To Cluster 将数组转化为簇。

四、For 循环结构

```
for(i=0;i<N;i++)
{
  }
```

功能模板—>Structure 子模板

1. 组成

循环框架（Loop Frame）：内有节点

重复端（Iteration Terminal）：N

计数端（Count Terminal）：i，初值为 0，递增步长为 1

2. 移位寄存器（Shift Register）

右侧移位寄存器（第 i−1 次）—>左侧移位寄存器（第 i 次）

3. 框架通道（Channel）

循环开始前，循环外节点—>循环内节点

循环结束时，循环内节点—>循环外节点

索引（Enable Indexing）—>数组

无索引（Disable Indexing）—> 最后一个数

自动索引（Auto Indexing）：循环执行时自动检测数组长度，并在每次循环时将数组中的元素按顺序一一取出

五、While 循环结构

当循环次数不能确定时，用 While 循环。

```
while(条件)
{
    }
```

或

```
do
{
    }while(条件)
```

组成：

循环框架（Loop Frame）

重复端口（Iteration Terminal）

条件端口（Conditional Terminal）每次迭代结束时，条件端口检测数据连线输入的布尔值，若为 TRUE（默认值），停止循环；若为 FALSE，继续循环。如果不赋值，则只执行一次。在最新的 LabVIEW 中，条件端口的停止条件可以自由设置。

移位寄存器（Shift Register）：同 for 循环。

框架通道（Channel）：同 for 循环。

六、顺序结构（Sequence Structure）

传统编程语言：控制流程（Control Flow）。

LabVIEW：数据流程（Data Flow）。

在 LabVIEW 中只有当某个节点的所有输入均有效时，LabVIEW 才能执行该节点—>数据从属性（Data Dependency）。

1. 组成

顺序框架（Sequence Frame）；

框图标识符（Diagram Identifier）；

递增/递减按钮（Increment/Decrement Buttons）。

2. 本地结果（Sequence Local）

在顺序框架中向后传递数据。

3. 框架通道（Frame Channel）

无 Enable Indexing 和 Disable Indexing 两种属性。

4. 公共连线（Common Threads）

建立流程控制权（Flow Control Right）。

Error Cluster 也是一种很好的公共连线，这种技术称为 ERROR I/O。

七、选择结构（Case Structure）

```
switch(表达式)
{case   常量表达式 1:语句 1;
case   常量表达式 2:语句 2;
              ⋮
case   常量表达式 n:语句 n;
default              :语句 n+1;
}

if(条件判断表达式)
{
  }
else
{
  }
```

组成如下：

选择框架（Case Frame）；

选择端口（Selection Terminal）：布尔型、数字整型、字符串型；

框图标识符（Diagram Identifier）；

递增/递减按钮（Increment/Decrement Buttons）。

八、公式节点（Formula Node）

创建如下：

（1）功能模板—>Structures 子模板—>Formula Node。

（2）添加输入输出端口。

（3）按照 C 语言的语法规则在公式节点的框架中加入程序代码，可进行简单的数学公式运算。在最新的 LabVIEW 8 中，这一功能可由 MathScript 替代，MathScript 采用 Matlab 语法，具备 Matlab 的基本功能。

利用公式节点的属性（Attribute Node）可以获得满意的人机交互界面，其包括：

（1）Visible Attribute；

（2）Disabled Attribute；

（3）Key Focus Attribute；

（4）Blinking Attribute；

(5) Position Attribute;

(6) Bounds Attribute (Read Only)。

第三节 模块化设计

模块化程序设计是相对于简单问题提出的，所谓“简单问题”是指求解问题的规模不大，即能用不超过 20 行的语句就可以完成问题的求解任务。这些问题求解，用 N－S 流程图（Nassit 与 Shneiderman 于 1973 年提出）表示比较直观和容易理解。但是，在实际应用中，求解问题的规模是相当大、相当复杂的。只用前面学过的编程技术还不够，还需要一种能把复杂问题分解成规模更小、更简单、更易理解的若干子问题的技术。这就导出另一种程序设计的技术——模块化的程序设计技术。

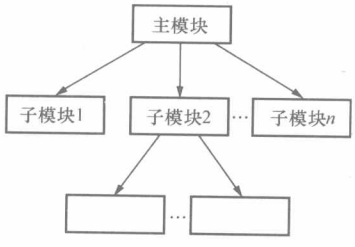


图 1-4 程序的分层结构

模块化的程序设计技术是指一个程序由一个主模块和若干子模块构成。如果需要，子模块还可以有其子模块。它们之间形成如图 1-4 所示的程序分层结构。

用这种分层管理来完成问题的求解，需要有数据传递的机制。当主模块调用子模块时，需要传给子模块一些数据，子模块完成一定的功能并返回主模块所需要的一些数据。主模块再把这些数据进行另外的加工处理，以完成整个问题的求解。很显然，由于模块之间需要数据传递，那么调用与被调用之间必须有一个接口（或称约定）。例如输入参数的个数、数据类型、返回值的个数及数据类型等。

一个模块有一个模块名。它由接口部分和功能实现部分组成。模块的接口标明模块名、输入参数的个数及类型，同时也标明返回参数的个数及类型。模块的实现部分是一段程序代码。模块化的程序设计有利于分工合作，有利于程序的调试和维护。G 语言用函数或过程来实现模块。

子 VI（SubVI）相当于普通编程语言中的子程序，即被其他 VI 调用的 VI。可以将任何一个定义了图标和连接器的 VI 作为另一个 VI 的子程序。在框图中打开 Functions→Select a VI…，就可以选择要调用的子 VI 。构造一个子 VI 主要的工作就是定义它的图标和连接器。

每个 VI 在前面板和流程图窗口的右上角都显示了一个默认的图标。启动图标编辑器的方法是，用鼠标右键单击面板窗口的右上角的默认图标，在弹出菜单中选择 Edit Icon。

图 1-5 显示了图标编辑器的窗口。可以用窗口左边的各种工具设计像素编辑区中的图标形状。编辑区右侧的方框中显示实际大小的图标。图标编辑器的具体使用细节请参阅有关资料。

连接器是 VI 数据的输入输出接口。如果用面板控制对象或者显示对象从子 VI 中输出或者输入数据，那么这些对象都需要在连接器面板中有一个连线端子。可以通过选择 VI 的端子数并为每个端子指定对应的前面板对象以定义连接器。

定义连接器的方法是，用鼠标右键单击面板窗口中的图标窗口，在快捷菜单中选择 Show Connector。

连接器图标会取代面板窗口右上角的图标。LabVIEW 自动选择的端子连接模式使控制对象的端子位于连接器窗口的左边，显示对象的端子位于连接器窗口右边。选择的端子数取决于前面板中控制对象和显示对象的个数。连接器中的各个矩形表示各个端子所在的区域，可以用

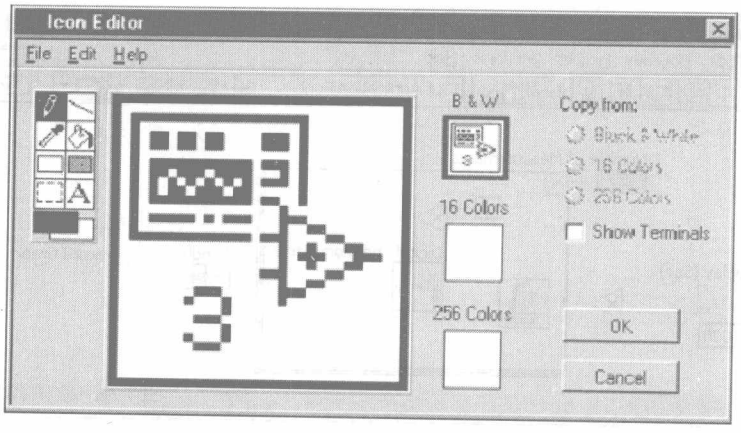


图 1-5　图标编辑器窗口

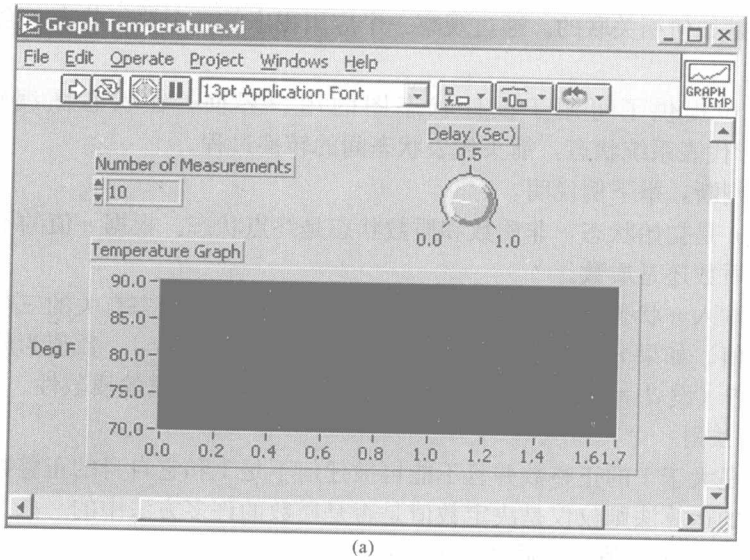
它们从 VI 中输入或者输出数据。如果必要，也可以选择另外一种端子连接模式。方法是在图标上单击鼠标右键弹出快捷菜单，选择 Show Connector，再次弹出快捷菜单，选择合适的 Patterns。

假设在某项设计中需要使用一个等间隔采样的温度测试仪器，可以作如下设计：

（1）从菜单“File”下选择“New VI”命令；

（2）在前面板中依次放置“Num Control”、“Dial”和“Graph”控件，并分别将其“Label”属性修改为“Number of Measurements”、“Delay (Sec)”和“Temperature Graph”。如图 1-6（a）所示；

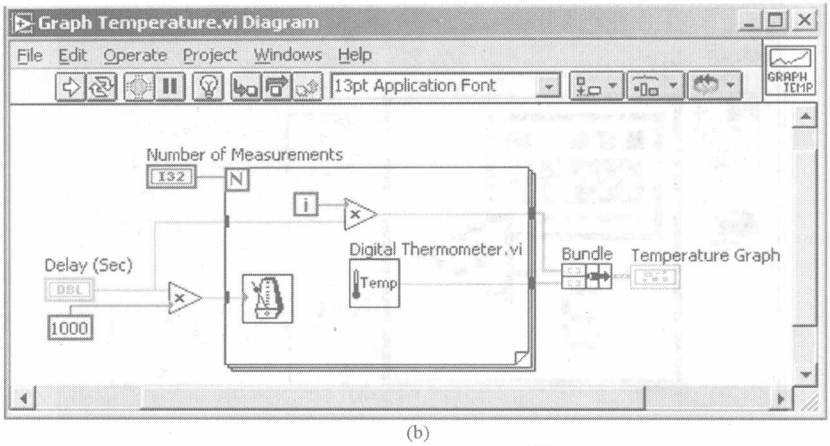
（3）打开后面板（快捷方式为 Ctrl+E），在“Tool Palette”中选择连线工具设计程序，程序设计结果如图 1-6（b）所示。



(a)

图 1-6　虚拟温度记录仪（一）

（a）前面板



(b)

图 1-6 虚拟温度记录仪（二）

（b）流程图

第四节 基于状态机的程序设计

状态机是一个具有独立的状态和转换过程的系统，转换过程管理系统的执行流程。在一个状态图里，每个状态可以通往一个或多个状态或是结束该程序流程。每个状态图根据用户的输入或是内部状态来决定下一个状态的执行。所有的应用程序都需要一个初始化状态，随之而来的转换状态会执行不同的任务。终点状态就是执行的最终状态，此时结束所有动作。

状态图可以有效地简化应用程序的复杂设计流程。除了可以对复杂流程进行观察外，状态图还具有编写应用程序的功能。要生成一个有效的状态图，必须知道应用程序的各个状态和这些状态之间是如何相关联的。通过观察一个应用程序的各个执行状态，会提高对应用程序的总设计能力。

状态图工具包提供了可用于设计状态图的用户界面。在状态图编辑窗口下（如图 1-7），椭圆形代表系统状态，箭头代表状态间的转换过程。

关于质数的判断，举下例说明。

问题：输入 n 是初始状态，非质数和质数状态是终点状态。根据 n 值的不同，用状态图来决定数值是非质数还是质数。

为了决定在输入 n 状态后哪个状态执行，状态图用与输入状态有关的三个转换箭头的说明条件来估计 n 值。如果 $n \leqslant 1$，数值是非质数；如果 $n \leqslant 3$ 且 $n \neq 1$，数值为质数；如果数值大于 3，三个状态（设 $d = 2$，n/d，和 $d = d + 1$）决定了数据的质数性。

1. 定义运算法则

如果数字 n 是大于 1 的正整数并且不能再被任何不是 1 和它自身的正整数整除时，该数就是质数。下面的运算法则仅仅是决定数值是否是质数的许多方法中的一条。

（1）如果 $n=1$，则数值不是质数，退出；

（2）如果 $n=2$ 或 3，则数值是质数，退出；

（3）如果 $n>3$，必须检验该数值是否只能被 1 和它自身整除。开始时设置除数 $d=2$，

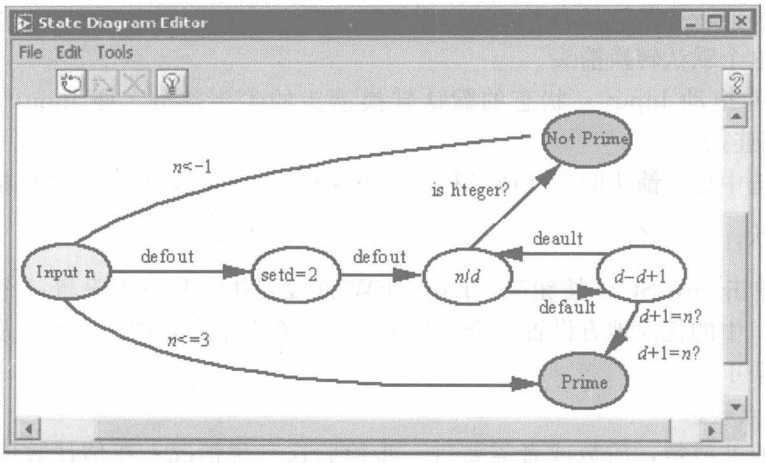


图 1-7　状态图编辑窗口

如果 n/d 是整数，则 n 是非质数，退出；如果 n/d 不是整数，通过设置 $d = d + 1$ 来增大除数，直到 $d + 1 = n$，如果 $d + 1 = n$ 才能被整除，则数值 n 为质数，退出。

2. 检验数值 1、2 和 3

（1）显示框图程序后选择状态图，并拖放至状态图编辑器模板里；出现初始状态用的绿色椭圆所示。初始状态接受状态机的输入并且是状态机运行时第一个执行的状态。

初始状态也有一个与之相连的转换箭头。转换箭头代表状态间的转换关系，并且显示出当特定条件为真时接下来执行哪个状态。

（2）在前面板上，放置一个数字型控件，数字控件的标签写为 n 与在数值运算法则里的第一个步骤为用户选择的数值 n 相对应。

（3）改变控件的数字类型为有符号的 32 位整数（I32）。

（4）将这个 VI 保存为“Prime Test. vi”。

3. 生成新状态

生成一个状态图前，第一步就是指定基本的逻辑关系。由运算法则可知数值 1 不是质数，并且数字 2、3 均是最小的质数。如果数值大于 3，必须用到质数检验运算法则继续检验该数值。这个定义表明需要四个状态：一个输入状态、一个提示需要继续测试的状态、一个非质数状态和一个质数状态。

（1）在状态图编辑器窗口，双击 Init 状态同时改变输入为 n 的标签。

注意，当你选中状态图编辑器窗口中的任何对象时，LabVIEW 都会将该对象的轮廓变为粉红色。

（2）需要一个可以指示何时需要进一步检验的状态，单击“new State”按钮。

注意，与初始状态不同，这个状态是黄色的并且代表一个转换状态。转换状态是执行过程中依赖于特定条件的中间状态。

（3）将新状态的标签改为 Keep Testing，如 default Keep Testing 。

（4）将 Keep Testing 状态移动到 Input n 状态的右边。

注意，Keep Testing 状态有一个默认转换箭头，创建的每个新状态都有一个默认转换箭