



NATIONAL INSTRUMENTS™
LabVIEW™

Benutzerhandbuch

Deutschsprachige Niederlassungen

National Instruments Germany GmbH Konrad-Celtis-Straße 79 D-81369 München Tel. (089) 741 31 30 Fax (089) 714 60 35	National Instruments Ges.m.b.H. Plainbachstraße 12 A-5101 Salzburg-Bergheim Tel. (0662) 45 79 90 0 Fax (0662) 45 79 90 19	National Instruments Switzerland Sonnenbergstraße 53 CH-5408 Ennetbaden Tel. (056) 200 51 51, (022) 980 05 11 (Genf) Fax (056) 200 21 55
---	--	---

Lokaler technischer Support

Deutschland:	ni.germany@ni.com	ni.com/germany
Österreich:	ni.austria@ni.com	ni.com/austria
Schweiz:	ni.switzerland@ni.com	ni.com/switzerland

Technischer Support und Produktinformation weltweit

ni.com

National Instruments Corporate Firmensitz

11500 North Mopac Expressway Austin, Texas 78759-3504 USA Tel: 001 512 794 0100

Internationale Niederlassungen

Australien 03 9879 5166, Belgien 02 757 00 20, Brasilien 011 284 5011, China 0755 3904939, Dänemark 45 76 26 00, Finnland 09 725 725 11, Frankreich 01 48 14 24 24, Griechenland 30 1 42 96 427, Großbritannien 01635 523545, Hongkong 2645 3186, Indien 91805275406, Israel 03 6120092, Italien 02 413091, Japan 03 5472 2970, Kanada (Calgary) 403 274 9391, Kanada (Ontario) 905 785 0085, Kanada (Québec) 514 694 8521, Korea 02 596 7456, Mexiko (D.F.) 5 280 7625, Mexiko (Monterrey) 8 357 7695, Neuseeland 09 914 0488, Niederlande 0348 433466, Norwegen 32 27 73 00, Polen 0 22 528 9406, Portugal 351 1 726 9011, Schweden 08 587 895 00, Singapur 2265886, Spanien 91 640 0085, Taiwan 02 2528 7227

Weitere Informationen finden Sie im Anhang unter *Technischer Support*. Wenn Sie Vorschläge oder Kritik zur Dokumentation haben, senden Sie diese per E-Mail an: techpubs@ni.com

Wichtige Informationen

Garantie

Für die Datenträger, auf denen Sie die Software von National Instruments erhalten, wird für den Zeitraum von 90 Tagen nach Erhalt der Lieferung (nachweisbar durch Lieferschein oder andere Dokumente) garantiert, dass sie keine Material- oder Verarbeitungsfehler aufweisen, die die Ausführung der Programmieranweisungen behindern. Wird National Instruments während der Garantiezeit über bestehende Schäden informiert, so wird National Instruments nach eigener Wahl Software-Datenträger, auf denen die Ausführung der Programmieranweisungen nicht möglich ist, entweder reparieren oder ersetzen. National Instruments leistet keine Gewähr dafür, dass die Ausführung der Software zu jeder Zeit oder fehlerfrei erfolgen kann.

Einsendungen werden nur dann zur Garantiebearbeitung angenommen, wenn sie deutlich auf der Außenseite durch eine Autorisierungsnummer für die Rücksendung, eine sogenannte RMA-Nummer (Return Material Authorization), gekennzeichnet sind. National Instruments übernimmt die Versandkosten für Teile, die im Rahmen einer Garantieleistung an den Kunden zurückgesandt werden.

National Instruments geht davon aus, dass die Informationen in diesem Dokument korrekt sind. Die technischen Angaben in diesem Dokument wurden sorgfältig überprüft. Falls trotzdem technische oder typographische Fehler vorhanden sein sollten, behält sich National Instruments das Recht vor, in nachfolgenden Auflagen dieses Dokuments Änderungen ohne vorherige Mitteilung an die Benutzer dieser Auflage vorzunehmen. Leser, die der Meinung sind, dass ein Fehler vorliegt, sollten sich direkt an National Instruments wenden. National Instruments übernimmt unter keinen Umständen eine Haftung für Schäden, die aufgrund dieses Dokuments beziehungsweise der darin enthaltenen Informationen oder im Zusammenhang damit entstehen.

Soweit in dieser Garantieerklärung nicht ausdrücklich vorgesehen, übernimmt National Instruments weder ausdrücklich noch stillschweigend irgendeine Gewähr. Insbesondere wird keine Gewähr für marktgängige Qualität oder die Eignung für einen bestimmten Zweck übernommen. Schadenersatzansprüche für Schäden, die durch Verschulden oder Fahrlässigkeit von National Instruments verursacht werden, sind auf die Höhe des Kaufpreises beschränkt, den der Kunde für das Produkt bezahlt hat. National Instruments ist nicht haftbar für Schäden, die durch den Verlust von Daten, entgangenen Gewinn, durch die Einschränkung der Verwendbarkeit der Produkte oder durch mittelbare Schäden oder Folgeschäden entstehen. Dies gilt auch dann, wenn National Instruments auf die Möglichkeit solcher Schäden hingewiesen wurde. Diese Einschränkung der Haftung von National Instruments gilt für alle Arten von Schadenersatzansprüchen, sei es aufgrund Vertrags oder unerlaubter Handlung, und gilt auch bei Verschulden. Gerichtliche Schritte gegen National Instruments müssen innerhalb eines Jahres nach Entstehen des Anspruchs eingeleitet werden. National Instruments ist nicht für die Verzögerung von Leistungen haftbar, die durch Vorgänge verursacht werden, über die National Instruments bei vernünftiger Betrachtung keine Kontrolle ausüben kann. Vorliegende Garantieerklärung erstreckt sich nicht auf Schäden, Defekte, Fehlfunktionen oder Funktionsausfälle, die dadurch verursacht werden, dass der Benutzer die Anleitungen von National Instruments für die Installation, den Betrieb und die Wartung nicht einhält. Dieser Garantiausschluss gilt ebenso für Schäden, die durch Veränderungen des Produkts, durch Missbrauch oder fahrlässiges Verhalten aufseiten des Benutzers, durch Stromausfälle oder Spannungsschöße, durch Brand, Überschwemmungen, Unfälle, Handlungen Dritter oder andere Vorfälle verursacht werden, die bei vernünftiger Betrachtung nicht kontrolliert werden können.

Copyright

Diese Veröffentlichung ist urheberrechtlich geschützt. Sie darf weder teilweise noch insgesamt auf irgendeine Weise, sei es elektronisch oder mechanisch, sei es durch Fotokopieren, Aufzeichnen oder Speichern in einem Informationsabrufsystem oder im Wege der Übersetzung, ohne vorherige schriftliche Genehmigung von National Instruments Corporation vervielfältigt oder übertragen werden.

Marken

ComponentWorks™, DAQPad™, DataSocket™, HiQ™, HiQ-Script™, IVI™, LabVIEW™, National Instruments™, ni.com™, NI-DAQ™, PXI™ und SCXI™ sind Marken der National Instruments Corporation.

Produkt- und Firmenbezeichnungen die hierin genannt wurden sind Marken oder Handelsnamen der jeweiligen Gesellschaft.

Warnung bezüglich des Gebrauchs von National Instruments Produkten

- (1) Die Softwareprodukte von National Instruments wurden nicht mit Komponenten und Tests für ein Sicherheitsniveau entwickelt, welches für eine Verwendung bei oder in Zusammenhang mit chirurgischen Implantaten oder als kritische Komponenten von lebenserhaltenden Systemen, deren Fehlfunktion bei vernünftiger Betrachtungsweise zu erheblichen Verletzungen von Menschen führen kann, geeignet ist.
- (2) Bei jeder Anwendung, einschließlich der oben genannten, kann die Zuverlässigkeit der Funktion der Softwareprodukte durch entgegenwirkende Faktoren, einschließlich zum Beispiel Spannungsunterschieden bei der Stromversorgung, Fehlfunktionen der Computer-Hardware, fehlende Eignung der Software für das Computerbetriebssystem, fehlende Eignung von Übersetzungs- und Entwicklungssoftware, die zur Entwicklung einer Anwendung eingesetzt werden, Installationsfehler, Probleme bei der Software- und Hardwarekompatibilität, Funktionsstörungen oder Ausfall der elektronischen Überwachungs- oder Kontrollgeräte, vorübergehende Fehler der elektronischen Systeme (Hardware und/oder Software) unvorhergesehener Einsatz oder Missbrauch sowie Fehler des Anwenders oder des Anwendungsentwicklers (entgegenwirkende Faktoren wie diese werden nachstehend zusammenfassend "Systemfehler" genannt) beeinträchtigt werden. Jede Anwendung, bei der ein Systemfehler ein Risiko für Sachwerte oder Personen darstellt (einschließlich der Gefahr körperlicher Schäden und Tod), sollte aufgrund der Gefahr von Systemfehlern nicht lediglich auf eine Form von elektronischem System gestützt werden. Um Schäden und unter Umständen tödliche Verletzungen zu vermeiden, sollte der Nutzer oder Anwendungsentwickler angemessene Sicherheitsmaßnahmen ergreifen, um Systemfehlern vorzubeugen. Hierzu gehören unter anderem Sicherungs- oder Abschaltmechanismen. Da jedes Endnutzersystem den Kundenbedürfnissen angepasst ist und sich von dem Testumfeld unterscheidet, und da ein Nutzer oder Anwendungsentwickler Softwareprodukte von National Instruments in Verbindung mit anderen Produkten in einer von National Instruments nicht getesteten oder vorhergesehenen Form einsetzen kann, trägt der Nutzer beziehungsweise der Anwendungsentwickler die letztendliche Verantwortung für die Überprüfung und Bewertung der Eignung von National Instruments Produkten, wenn Produkte von National Instruments in ein System oder eine Anwendung integriert werden. Dies erfordert unter anderem die entsprechende Entwicklung und Verwendung sowie Einhaltung einer entsprechenden Sicherheitsstufe bei einem solchen System oder einer solchen Anwendung.

Inhaltsverzeichnis

Über dieses Handbuch

Gliederung dieses Handbuchs.....	xviii
Konventionen.....	xviii

TEIL I LabVIEW-Konzepte

Kapitel 1

Einführung in LabVIEW

LabVIEW-Dokumentationsressourcen.....	1-1
LabVIEW Beispiel-VIs und Werkzeuge.....	1-4
LabVIEW Beispiel-VIs.....	1-4
LabVIEW-Werkzeuge.....	1-4

Kapitel 2

Einführung in Virtuelle Instrumente

Frontpanel.....	2-2
Blockdiagramm.....	2-3
Anschlüsse.....	2-3
Knoten.....	2-4
Verbindungen.....	2-4
Strukturen.....	2-4
Symbol- und Anschlussfeld.....	2-5
Verwenden und Anpassen von VIs und Sub-VIs.....	2-6

Kapitel 3

LabVIEW-Umgebung

Elementpalette.....	3-1
Funktionspalette.....	3-1
Navigieren in den Paletten Elemente und Funktionen.....	3-2
Werkzeugpalette.....	3-2
Menüs und Symbolleiste.....	3-3
Menüs.....	3-3
Kontextmenüs.....	3-3
Kontextmenüs im Ausführungsmodus.....	3-3
Symbolleiste.....	3-4

Anpassen der Arbeitsumgebung.....	3-4
Anpassen der Paletten Elemente und Funktionen.....	3-4
Hinzufügen von VIs und Bedienelementen zur Benutzerbibliothek und zur Instrumentenbibliothek	3-4
Erstellen und Bearbeiten von Palettenansichten.....	3-5
Wie Ansichten von LabVIEW gespeichert werden	3-5
Erstellen von ActiveX-Unterpaletten	3-6
Darstellen von Werkzeugsätzen in den Paletten.....	3-6
Festlegen von Optionen für die Arbeitsumgebung	3-6
Wie Optionen von LabVIEW gespeichert werden	3-7
Windows.....	3-7
Macintosh	3-7
UNIX.....	3-7

Kapitel 4

Erstellen des Frontpanels

Konfigurieren von Objekten auf dem Frontpanel	4-1
Ein- und Ausblenden von optionalen Elementen.....	4-2
Umwandeln von Bedienelementen in Anzeigeelemente und umgekehrt	4-2
Ersetzen von Frontpanel-Objekten.....	4-2
Festlegen von Tastenkombinationen für Bedienelemente	4-3
Steuern des Schaltflächenverhaltens per Tastenbelegung	4-3
Festlegen der Navigationsreihenfolge von Frontpanel-Objekten	4-4
Zuweisen von Farben zu Objekten.....	4-5
Verwenden von importierten Grafiken	4-5
Gruppieren und Sperren von Objekten	4-6
Ändern der Größe von Objekten	4-6
Skalieren von Frontpanel-Objekten	4-7
Hinzufügen von Leerraum auf dem Frontpanel ohne Größenänderung des Fensters	4-9
Frontpanel-Bedien- und Anzeigeelemente.....	4-9
3D- und klassische Bedien- und Anzeigeelemente.....	4-9
Schieberegler, Drehknöpfe, Drehregler und numerische Anzeigen	4-10
Schieberegler und Anzeigen.....	4-10
Rotatorische Bedienelemente und Anzeigen	4-10
Numerische Bedienelemente und Anzeigen.....	4-11
Farbboxen	4-11
Farbrampen.....	4-11
Tasten, Schalter und Leuchten	4-12
Texteingabefelder, Beschriftungen und Pfadanzeigen.....	4-12
String-Bedien- und Anzeigeelemente.....	4-12
Dateipfad-Bedien- und Anzeigeelemente.....	4-13
Ungültige Pfade.....	4-13

Leere Pfade	4-13
Array- und Cluster-Bedien- und Anzeigeelemente	4-14
Register-Bedien- und Anzeigeelemente.....	4-14
Listenfelder.....	4-15
Ring- und Enum-Bedien- und Anzeigeelemente.....	4-15
Ring-Bedienelemente.....	4-16
Bedienelemente vom Enum-Typ	4-16
Erweiterte Bedien- und Anzeigeelemente	
vom Typ Enum	4-17
I/O-Namen-Bedien- und Anzeigeelemente	4-17
Referenzen auf Objekte oder Applikationen	4-18
Dialogelemente.....	4-19
Beschriftungen	4-19
Untertitel.....	4-20
Texteigenschaften	4-20
Gestalten von Benutzeroberflächen	4-22
Verwenden von Frontpanel-Bedien- und Anzeigeelementen.....	4-22
Gestalten von Dialogfeldern.....	4-23
Auswählen der Bildschirmgröße	4-23

Kapitel 5

Erstellen des Blockdiagramms

Beziehung zwischen Frontpanel-Objekten und Blockdiagrammanschlüssen	5-1
Blockdiagrammobjekte	5-2
Blockdiagrammanschlüsse	5-2
Datentypen für Bedien- und Anzeigeelemente	5-2
Konstanten	5-4
Konstanten	5-5
Benutzerdefinierte Konstanten	5-5
Blockdiagrammknoten	5-6
Überblick über Funktionen	5-7
Numerische Funktionen.....	5-7
Boolesche Funktionen	5-8
String-Funktionen.....	5-8
Array-Funktionen	5-8
Cluster-Funktionen	5-9
Vergleichsfunktionen	5-9
Zeit- und Dialogfunktionen	5-9
Datei-I/O-Funktionen	5-9
Signalverlaufsfunktionen.....	5-10
Applikationsteuerungsfunktionen.....	5-10
Fortgeschrittene Funktionen.....	5-10
Hinzufügen von Anschlüssen zu Blockdiagrammfunktionen	5-11

Verbindung von Blockdiagrammobjekten	5-11
Automatisches Verbinden von Objekten	5-13
Manuelles Verbinden von Objekten	5-14
Markieren von Verbindungen	5-14
Entfernen von unterbrochenen Verbindungen	5-14
Typumwandlungspunkte	5-15
Polymorphe VIs und Funktionen.....	5-16
Polymorphe VIs	5-16
Erstellen von polymorphen VIs	5-16
Polymorphe Funktionen	5-18
Verarbeiten von Variant-Daten	5-19
Numerische Einheiten und strikte Typenprüfung.....	5-20
Einheiten und strikte Typenüberprüfung	5-21
Blockdiagramm-Datenfluss	5-23
Datenabhängigkeit und künstliche Datenabhängigkeit.....	5-23
Fehlende Datenabhängigkeit	5-24
Datenfluss und Speicherverwaltung.....	5-25
Entwerfen des Blockdiagramms.....	5-26

Kapitel 6

Ausführen von und Fehlersuche in VIs

Ausführen von VIs	6-1
Ausführungskonfiguration eines VIs	6-2
Korrigieren von nicht ausführbaren VIs.....	6-2
Herausfinden der Ursachen für nicht ausführbare VIs.....	6-2
Häufige Ursachen für nicht ausführbare VIs	6-3
Techniken für die Fehlersuche	6-4
Highlight-Funktion.....	6-4
Einzelschrittausführung	6-5
Probe-Daten-Werkzeug.....	6-5
Haltepunkte	6-6
Aussetzen der Ausführung	6-6
Ermitteln der aktuellen Instanz eines Sub-VIs	6-7
Vorübergehendes Deaktivieren von Abschnitten des Blockdiagramms.....	6-7
Deaktivieren der Fehlersuche-Werkzeuge	6-8
Undefinierte oder unerwartete Daten	6-8
Unerwartete und Standarddaten in Schleifen.....	6-9
For-Schleifen	6-9
While-Schleifen	6-9
Standarddaten in Arrays.....	6-9
Verhindern von undefinierten Daten.....	6-10

Fehlerprüfung und Fehlerbehandlung.....	6-10
Prüfen auf Fehler	6-10
Fehlerbehandlung	6-11
Fehler-Cluster.....	6-12
Verwenden von While-Schleifen für die Fehlerbehandlung.....	6-12
Verwenden von Case-Strukturen für die Fehlerbehandlung.....	6-12

Kapitel 7

Erstellen von VIs und Sub-VIs

Planen und Entwerfen eines Projekts.....	7-1
Entwerfen von Projekten mit mehreren Entwicklern	7-2
Verwenden der integrierten VIs und Funktionen	7-3
Erstellen von Instrumentensteuerungs- und Datenerfassungs-VIs und -Funktionen.....	7-3
Erstellen von VIs, die auf andere VIs zugreifen.....	7-4
Erstellen von VIs, die mit anderen Applikationen kommunizieren	7-4
Sub-VIs	7-5
Identifizieren von häufig wiederholten Operationen.....	7-5
Einrichten des Anschlussfeldes	7-7
Festlegen von erforderlichen, empfohlenen und optionalen Ein- und Ausgängen.....	7-8
Erstellen eines Symbols.....	7-9
Erstellen von Sub-VIs aus Teilen eines VIs	7-10
Entwerfen von Sub-VIs	7-10
Anzeigen der VI-Hierarchie	7-11
Speichern von VIs.....	7-11
Vorteile des Speicherns von VIs als Einzeldateien	7-12
Vorteile des Speicherns von VIs als Bibliotheken	7-12
Verwalten von VIs in Bibliotheken.....	7-13
Benennen von VIs	7-13
Speichern für eine Vorläuferversion	7-14
Bereitstellen von VIs	7-14
Erstellen von eigenständigen Applikationen und gemeinsamen Bibliotheken.....	7-15

TEIL II

Erstellen und Bearbeiten von VIs

Kapitel 8

Schleifen und Case-Strukturen

For-Schleifen- und While-Schleifenstrukturen	8-2
For-Schleifen.....	8-2
While-Schleifen	8-3
Vermeiden von endlosen While-Schleifen	8-4
Auto-Indizierung von Schleifen.....	8-4
Verwenden der Auto-Indizierung zum Einstellen der For-Schleifenanzahl	8-5
Auto-Indizierung mit While-Schleife	8-6
Schieberegister in Schleifen.....	8-6
Steuern des Timings.....	8-7
Case- und Sequenzstrukturen	8-7
Case-Strukturen.....	8-8
Case-Selektorerwerte und Datentypen.....	8-8
Eingabe- und Ausgabebetunnel.....	8-9
Verwenden von Case-Strukturen für die Fehlerbehandlung	8-10
Sequenzstrukturen.....	8-10
Vermeiden, dass Sequenzstrukturen zu häufig verwendet werden	8-11

Kapitel 9

Gruppieren von Daten mit Hilfe von Strings, Arrays und Clustern

Strings.....	9-1
Strings auf dem Frontpanel	9-2
String-Anzeigearten.....	9-2
Tabellen.....	9-3
Programmgesteuertes Bearbeiten von Strings	9-3
Formatieren von Strings.....	9-4
Format-Bezeichner	9-5
Zahlen und Strings	9-5
Gruppieren von Daten mit Arrays und Clustern.....	9-6
Arrays.....	9-6
Indizes.....	9-7
Beispiele für Arrays.....	9-7
Beschränkungen für Arrays	9-9
Erstellen von Array-Bedien- und Anzeigeelementen und -Konstanten.....	9-9

Array-Indexanzeige.....	9-10
Array-Funktionen.....	9-11
Automatisches Ändern der Größe von Array-Funktionen	9-11
Cluster.....	9-12

Kapitel 10

Lokale und globale Variablen

Lokale Variablen.....	10-1
Erstellen von lokalen Variablen	10-2
Globale Variablen	10-2
Erstellen von globalen Variablen	10-3
Lesen und Schreiben von Variablen	10-4
Umsichtiges Verwenden von lokalen und globalen Variablen.....	10-5
Initialisieren von lokalen und globalen Variablen.....	10-5
Ablaufunterschiede.....	10-5
Den Speicher betreffende Überlegungen beim Einsatz von lokalen Variablen.....	10-6
Den Speicher betreffende Überlegungen beim Einsatz von globalen Variablen.....	10-7

Kapitel 11

Graphen und Diagramme

Grafik- und Diagrammtypen.....	11-1
Optionen für Graphen und Diagramme	11-2
Mehrere X- und Y-Achsen in Graphen und Diagrammen	11-2
Kantengeglättete Liniendarstellungen für Graphen und Diagramme.....	11-2
Anpassen der Darstellung von Graphen und Diagrammen	11-3
Anpassen von Graphen.....	11-3
Graphen-Cursor.....	11-4
Achsenoptionen.....	11-5
Achsenlegende für Kurvengraphen	11-5
Formatieren der Graphen-Achsen	11-5
Glätten von Grafiken.....	11-6
Anpassen von Diagrammen.....	11-6
Historienlänge	11-6
Aktualisierungsmodi von Diagrammen	11-6
Überlagerte Kurven und Stapelplots	11-7
Kurven- und XY-Graphen	11-8
Datentypen für Kurvengraphen mit einer Kurve	11-8
Kurvengraph mit mehreren Kurven	11-9
Datentypen für XY-Graphen mit einer Kurve	11-10
Datentypen für XY-Graphen mit mehreren Kurven	11-10

Kurvendiagramme	11-11
Intensitätsgraphen und -diagramme	11-12
Farbzuordnung	11-13
Optionen für Intensitätsdiagramme.....	11-14
Optionen für Intensitätsgraphen.....	11-14
Digitale Graphen	11-15
Maskieren von Daten	11-17
3D-Graphen	11-17
Der Datentyp „Signalverlauf“	11-18

Kapitel 12

VIs für Grafiken und Sound

Verwenden des Bildanzeigeelements	12-1
VIs für Bildplots	12-3
Verwenden von „Polar Plot.vi“ als Sub-VI	12-3
Verwenden von „Plot Waveform.vi“ als Sub-VI.....	12-4
Verwenden von „Smith Plot.vi“ als Sub-VI	12-4
Bildfunktionen-VIs.....	12-5
Erstellen und Ändern von Farben mit den Bildfunktionen-VIs	12-6
Grafikformate-VIs	12-7
Sound-VIs.....	12-8

Kapitel 13

Datei-I/O

Grundlagen der Datei-I/O.....	13-1
Auswählen eines Datei-I/O-Formats	13-2
Einsatz von Textdateien	13-2
Einsatz von Binärdateien	13-3
Einsatz von Datenprotokolldateien	13-4
Verwenden von High-Level-Datei-I/O-VIs	13-5
Verwenden von Low-Level- und fortgeschrittenen Datei-I/O-VIs und -Funktionen	13-6
Datenträger-Streaming	13-8
Erstellen von Text- und Spreadsheet-Dateien	13-9
Formatieren und Schreiben von Daten in Dateien	13-10
Einlesen von Daten aus Dateien.....	13-10
Erstellen von Binärdateien	13-11
Erstellen von Datenprotokolldateien	13-11
Schreiben von Signalverläufen in Dateien	13-12
Lesen von Signalverläufen aus Dateien	13-13
Durchfluss-Parameter	13-14

Erstellen von Konfigurationsdateien	13-14
Verwenden von Konfigurationseinstellungsdateien.....	13-15
Windows-Dateiformat für Konfigurationseinstellungen	13-15
Protokollieren von Frontpanel-Daten	13-17
Automatische und interaktive Frontpanel-Datenprotokollierung.....	13-18
Interaktive Anzeige der protokollierten Frontpanel-Daten	13-18
Löschen eines Datensatzes.....	13-19
Löschen der Protokolldateibindung	13-19
Ändern der Protokolldateibindung.....	13-20
Programmatischer Abruf von Frontpanel-Daten	13-20
Abrufen von Frontpanel-Daten mit Hilfe eines Sub-VIs.....	13-20
Festlegen von Datensätzen	13-21
Abrufen von Frontpanel-Daten mit Datei-I/O-Funktionen.....	13-22

Kapitel 14

Dokumentieren und Drucken von VIs

Dokumentieren von VIs.....	14-2
Erstellen von VI- und Objektbeschreibungen	14-2
Einrichten der VI-Revisions-Historie.....	14-2
Versionsnummern	14-3
Drucken der Dokumentation	14-4
Speichern in HTML- oder RTF-Dateien.....	14-4
Auswählen des Grafikformats für HTML-Dateien.....	14-4
Namenskonventionen für Grafikdateien.....	14-5
Erstellen eigener Hilfedateien.....	14-5
Windows-Hilfedateien	14-6
Drucken von VIs.....	14-6
Drucken des aktiven Fensters.....	14-6
Drucken von Berichten.....	14-7
Programmatisches Drucken.....	14-7
Drucken nach Ausführung	14-8
Verwenden eines Sub-VIs für einen selektiven Ausdruck bei Beendigung	14-8
Weitere Druckverfahren	14-8

Kapitel 15

Anpassen von VIs

Konfigurieren von Erscheinungsbild und Verhalten von VIs	15-1
Anpassen von Menüs	15-2
Erstellen von Menüs.....	15-3
Verarbeitung der Menüauswahl	15-3

Kapitel 16

Programmatische Steuerung von VIs

Fähigkeiten des VI-Servers	16-1
Erstellen von VI-Server-Applikationen.....	16-2
Applikations- und VI-Referenzen.....	16-3
Bearbeiten von Applikations- und VI-Einstellungen	16-4
Eigenschaftsknoten	16-4
Implizit verknüpfte Eigenschaftsknoten.....	16-4
Methodenknoten.....	16-5
Manipulieren von Eigenschaften und Methoden der Applikationsklasse.....	16-5
Manipulieren von Eigenschaften und Methoden der VI-Klasse	16-6
Manipulieren von Eigenschaften und Methoden der Applikationsklasse und VI-Klasse	16-7
Dynamisches Laden und Aufrufen von VIs	16-7
Der Knoten „Aufruf über Referenz“ und strikt typisierte VI-RefNums	16-8
Bearbeiten und Ausführen von VIs auf Netzwerkrechnern	16-9
Objekt-RefNums.....	16-9
Strikt typisierte und schwach typisierte Objekt-RefNums.....	16-9

Kapitel 17

Arbeiten mit LabVIEW im Netzwerk

Auswahl zwischen Datei-I/O, VI-Server, ActiveX und Arbeiten im Netzwerk	17-1
LabVIEW als Netzwerk-Client und -Server.....	17-2
Verwenden der DataSocket-Technologie.....	17-3
Angaben einer URL	17-4
Unterstützte DataSocket-Datenformate	17-5
Verwenden von DataSocket im Frontpanel	17-6
Lesen und Schreiben von Direktübertragungsdaten über das Blockdiagramm.....	17-8
DataSocket und Variant-Daten	17-9
Veröffentlichen von VIs im Web	17-10
Webserver-Optionen	17-10
Erstellen von HTML-Dokumenten	17-11
Veröffentlichen von Frontpanel-Bildern.....	17-11
Frontpanel-Bildformate	17-12
Low-Level-Kommunikationsanwendungen	17-12
TCP und UDP	17-12
DDE (Windows)	17-13
Apple-Ereignisse und PPC-Toolbox (Macintosh)	17-13
Pipe-VIs (UNIX).....	17-13
Ausführen von Befehlen auf Systemebene (Windows und UNIX)	17-13

Kapitel 18

ActiveX

ActiveX-Objekte, -Eigenschaften, -Methoden und -Ereignisse	18-1
ActiveX-VIs, -Funktionen, -Bedienelemente und -Anzeigeelemente.....	18-2
LabVIEW als ActiveX-Client.....	18-3
Zugriff auf eine ActiveX-fähige Applikation.....	18-4
Einfügen eines ActiveX-Objekts oder -Dokuments im Frontpanel	18-4
LabVIEW als ActiveX-Server	18-5
Verwenden von Konstanten zum Festlegen von Parametern in ActiveX-VIs.....	18-5

Kapitel 19

Aufrufen von Code aus textbasierten Programmiersprachen

Funktion „Aufruf externer Bibliotheken“	19-1
Code-Interface-Knoten	19-1

Kapitel 20

Formeln und Gleichungen

Methoden zur Nutzung von Gleichungen in LabVIEW	20-1
Formelknoten	20-2
Verwenden des Formelknotens	20-2
Variablen im Formelknoten.....	20-3
Ausdrucks-knoten	20-4
Polymorphismus in Ausdrucks-knoten.....	20-4
Verwenden von HiQ mit LabVIEW	20-5
HiQ- und MATLAB-Skript-Knoten	20-5
Programmiervorschläge für HiQ- und MATLAB-Skripts	20-7
Erforderliche HiQ-Supportdateien für eine LabVIEW-Applikation	20-7

Anhang A

Die Organisation von LabVIEW

Organisation der LabVIEW-Verzeichnisstruktur	A-1
Bibliotheken	A-1
Struktur und Support	A-2
Lernen und Anleitung.....	A-2
Dokumentation	A-2
Datei für Sonstiges	A-2
Macintosh	A-2
Vorgeschlagenes Verzeichnis zum Speichern von Dateien.....	A-3

Anhang B

Polymorphe Funktionen

Numerische Konvertierung	B-1
Polymorphismus von numerischen Funktionen	B-2
Polymorphismus von booleschen Funktionen	B-4
Polymorphismus von Array-Funktionen	B-5
Polymorphismus von String-Funktionen	B-5
Polymorphismus von String-Konvertierungsfunktionen	B-5
Polymorphismus von zusätzlichen String_in_Zahl-Funktionen	B-6
Polymorphismus von Cluster-Funktionen	B-6
Polymorphismus von Vergleichsfunktionen	B-6
Polymorphismus von logarithmischen Funktionen	B-7

Anhang C

Vergleichsfunktionen

Vergleichen von booleschen Werten	C-1
Vergleichen von Strings	C-1
Vergleichen von Zahlen	C-2
Vergleichen von Arrays und Clustern	C-2
Arrays	C-2
Modus „Vergleichen der Elemente“	C-2
Modus „Vergleichen mehrerer Elemente“	C-3
Cluster	C-3
Modus „Vergleichen der Elemente“	C-3
Modus „Vergleichen mehrerer Elemente“	C-4

Anhang D

Maskieren von digitalen Daten

Anhang E

Technischer Support

Glossar

Stichwortverzeichnis

Über dieses Handbuch

In diesem Handbuch wird die grafische Programmierungsumgebung von LabVIEW erklärt und die Techniken für das Erstellen von Applikationen in LabVIEW, wie beispielsweise Prüf- und Messapplikationen sowie Applikationen für die Datenerfassung, die Gerätesteuerung, die Datenprotokollierung, die Messungsanalyse und die Berichterzeugung erläutert.

Dieses Handbuch vermittelt Kenntnisse über die LabVIEW-Programmierfunktionen einschließlich der LabVIEW-Benutzeroberfläche und der Programmierarbeitsbereiche, und Sie lernen die LabVIEW-Paletten, -Werkzeuge und -Dialogfelder kennen. Dieses Handbuch enthält jedoch keine spezifischen Informationen über jede Palette, jedes Werkzeug, Menü, Dialogfeld, Steuerelement oder integriertes VI oder integrierte Funktion. Weitere Informationen zu diesen Elementen sowie detaillierte schrittweise Anleitungen für den Einsatz der LabVIEW-Funktionen und das Entwickeln bestimmter Applikationen finden Sie in der *LabVIEW-Hilfe*. Weitere Informationen zur *LabVIEW-Hilfe* und den Zugriff hierauf finden Sie in dem Abschnitt [LabVIEW-Dokumentationsressourcen](#) des Kapitels 1, [Einführung in LabVIEW](#).

Das *LabVIEW-Benutzerhandbuch* steht auch im PDF-Format (Portable Document Format) zur Verfügung. Sie können die PDF-Version installieren oder von der Installations-CD aus darauf zugreifen. Wenn Sie die Installationsoption **Vollständig** wählen, installiert LabVIEW PDF-Versionen von allen LabVIEW-Handbüchern. Zum Anzeigen der PDF-Dateien muss Adobe Acrobat Reader 3.0 oder höher installiert sein. Sie können den Acrobat Reader von der Internetseite von Adobe Systems Incorporated unter www.adobe.com herunterladen.

Sie können auch von der *LabVIEW-Hilfe* auf die PDF-Dateien zugreifen, die Sie dafür vorher installieren müssen. Diese Funktion steht nur unter Windows zur Verfügung. Weitere Informationen zum Installieren der LabVIEW-Handbücher im PDF-Format finden Sie in den *LabVIEW Versionshinweisen* oder den *LabVIEW Aktualisierungshinweisen*. Weitere Informationen über den Zugriff auf die PDF-Dateien in der *LabVIEW-Library PDF* finden Sie in dem Abschnitt [LabVIEW-Dokumentationsressourcen](#) des Kapitels 1, [Einführung in LabVIEW](#).

Gliederung dieses Handbuchs

Das *LabVIEW-Benutzerhandbuch* besteht aus zwei Teilen. Teil I, *LabVIEW-Konzepte*, beschreibt die Programmierkonzepte für das Erstellen von Applikationen in LabVIEW. Die Kapitel in diesem Teil geben eine Einführung in die LabVIEW-Programmierumgebung und helfen bei der Planung einer Applikation.

Teil II, *Erstellen und Bearbeiten von VIs*, beschreibt LabVIEW-Eigenschaften, -VIs und -Funktionen, mit denen Sie die jeweilige Funktionsweisen Ihrer Applikation festlegen können. Die Kapitel in diesem Teil beschreiben die Einsatzmöglichkeiten jedes einzelnen LabVIEW-Leistungsmerkmals sowie die einzelnen Klassen von VIs und Funktionen.

Konventionen

In diesem Handbuch werden die folgenden Schreibkonventionen verwendet:

»

Das Symbol » kennzeichnet die Reihenfolge, in der Sie Menübefehle und Dialogfeldoptionen ausführen müssen. Mit der Abfolge **Datei» Seiteneinstellungen» Optionen** werden Sie angewiesen, das Menü **Datei** zu öffnen, hieraus den Befehl **Seiteneinstellungen** und anschließend im letzten Dialogfeld den Befehl **Optionen** zu wählen.



Dieses Symbol steht für einen Tipp und macht Sie auf hilfreiche Ratschläge aufmerksam.



Dieses Symbol steht für einen Hinweis und macht Sie auf wichtige Informationen aufmerksam.



Dieses Symbol steht für einen Warnhinweis, der Vorsichtsmaßnahmen beschreibt, durch die Verletzungen, Datenverluste oder Systemabstürze vermieden werden können.

fett

Fettgedruckter Text weist auf Elemente hin, die Sie auswählen oder auf die Sie in der Software klicken müssen, wie beispielsweise Menüelemente oder Dialogfeldoptionen. Bei fettgedrucktem Text kann es sich auch um Parameternamen, Bedienelemente und Tasten auf dem Frontpanel, Dialogfelder, Bereiche in Dialogfeldern, Menünamen und Palettennamen handeln.

fett gesperrt	Text in fetter, gesperrter Schrift kennzeichnet Meldungen und Antworten, die vom Computer automatisch auf dem Bildschirm ausgegeben werden. Mit diesem Schriftstil werden auch Codezeilen hervorgehoben, die sich von anderen Beispielen unterscheiden.
gesperrt	Durch diesen Schriftstil werden sowohl Codeabschnitte, Programmierbeispiele und Syntaxbeispiele als auch Texte oder Zeichen gekennzeichnet, die Sie über die Tastatur eingeben müssen. Dieser Schriftstil wird auch für die Namen von Diskettenlaufwerken, für Pfade, Verzeichnisse, Programme, Unterprogramme, Subroutinen, Gerätenamen, Funktionen, Operationen, Variablen, Dateinamen und -erweiterungen sowie für Codeauszüge verwendet.
<i>gesperrt kursiv</i>	Text in gesperrter, kursiver Schrift kennzeichnet Text, der als Platzhalter für ein Wort oder einen Wert fungiert, das oder den Sie eingeben müssen.
Klicken mit der rechten Maustaste	(Macintosh) Das Drücken der <Befehls->-Taste bei gleichzeitigem Klicken bewirkt die gleiche Aktion wie ein Rechtsklick.
<i>kursiv</i>	Text in kursiver Schrift kennzeichnet Variablen, Hervorhebungen, Querverweise oder Einführungen in wichtige Konzepte. Dieser Schriftstil weist auch auf Text hin, der als Platzhalter für ein Wort oder einen Wert fungiert, das oder den Sie angeben müssen.
Plattform	Text in diesem Schriftstil steht für eine bestimmte Plattform und weist darauf hin, dass sich der folgende Text nur auf diese Plattform bezieht.

LabVIEW-Konzepte

Dieser Teil beschreibt die Programmierkonzepte, um Applikationen in LabVIEW zu erstellen. Die Kapitel in diesem Abschnitt geben eine Einführung in die LabVIEW-Programmierumgebung und helfen bei der Planung der Applikation.

Teil I, *LabVIEW-Konzepte*, enthält die folgenden Kapitel:

- Kapitel 1, *Einführung in LabVIEW*, beschreibt LabVIEW, die dazugehörige umfassende Dokumentation sowie die Werkzeuge, mit denen Sie VIs entwerfen und erstellen können.
- Kapitel 2, *Einführung in Virtuelle Instrumente*, beschreibt die Komponenten Virtueller Instrumente oder VIs.
- Kapitel 3, *LabVIEW-Umgebung*, beschreibt die LabVIEW-Paletten, -Werkzeuge und -Menüs, mit denen Sie die Frontpanels und Blockdiagramme von VIs erstellen. Dieses Kapitel beschreibt weiterhin die Anpassung der LabVIEW-Paletten und das Einstellen verschiedener Optionen für die Arbeitsumgebung.
- Kapitel 4, *Erstellen des Frontpanels*, beschreibt das Erstellen des Frontpanels eines VIs.
- Kapitel 5, *Erstellen des Blockdiagramms*, beschreibt das Erstellen des Blockdiagramms eines VIs.
- Kapitel 6, *Ausführen von und Fehlersuche in VIs*, beschreibt das Konfigurieren der Ausführung eines VIs und das Erkennen von Problemen bei der Blockdiagrammanordnung beziehungsweise bei den durch das Blockdiagramm laufenden Daten.
- Kapitel 7, *Erstellen von VIs und Sub-VIs*, beschreibt das Erstellen eigener VIs und Sub-VIs, das Verteilen von VIs und das Erstellen unabhängiger Applikationen und Shared Libraries (DLLs).

Einführung in LabVIEW

LabVIEW ist eine grafische Programmiersprache, die Symbole anstelle von Text verwendet, um Applikationen zu erstellen. Im Gegensatz zu textbasierten Programmiersprachen, bei denen die Programmausführung von Anweisungen gesteuert wird, verwendet LabVIEW die Datenflussprogrammierung, bei welcher der Fluss der Daten die Ausführung steuert.

In LabVIEW erstellen Sie eine Benutzeroberfläche mit Hilfe einer Sammlung von Werkzeugen und Objekten. Die Benutzeroberfläche wird als Frontpanel bezeichnet. Sie fügen Code hinzu, indem Sie grafische Darstellungen von Funktionen zum Steuern der Frontpanel-Objekte verwenden. Dieser Code ist im Blockdiagramm enthalten. In mancher Hinsicht ähnelt dieses Blockdiagramm einem Flussdiagramm.

Für die Entwicklung spezieller Applikationen ist eine Reihe von Software-Werkzeugsätzen (Add-Ons) zur Erweiterung von LabVIEW erhältlich. All diese Werkzeugsätze lassen sich nahtlos in LabVIEW integrieren. Weitere Informationen zu diesen Werkzeugsätzen finden Sie auf der National Instruments-Internetseite ni.com.

LabVIEW-Dokumentationsressourcen

Zum Lieferumfang von LabVIEW gehört eine umfangreiche Dokumentation für Einsteiger und fortgeschrittene LabVIEW-Benutzer. Alle LabVIEW-Handbücher und Applikationsinformationen stehen auch als PDF-Dateien zur Verfügung. Ressourcen für aktualisierte Dokumentationen finden Sie auf der Internetseite von National Instruments ni.com.

- **Erste Schritte mit LabVIEW:** Anhand dieses Handbuchs können Sie sich mit der grafischen Programmierumgebung von LabVIEW und den grundlegenden LabVIEW-Funktionen vertraut machen, die Sie zum Erstellen von Datenerfassungs- und Gerätesteuersapplikationen verwenden.
- **LabVIEW-Tutorium:** Anhand dieses Tutoriums lernen Sie die grundlegenden LabVIEW-Konzepte kennen. Das Tutorium führt Sie durch eine Reihe von Übungen, damit Sie sich mit der grafischen Programmierung vertraut machen können. Sie greifen auf das *LabVIEW-Tutorium* zu, indem Sie **Hilfe»Inhalt und Index** auswählen

oder indem Sie im Dialogfeld **LabVIEW** auf die Schaltfläche **LabVIEW-Tutorium** klicken.

- **LabVIEW-Referenzkarte:** Verwenden Sie diese Karte, um einen schnellen Einstieg in LabVIEW zu finden. Auf der Karte werden die allgemeinen Techniken für die Bearbeitung, die Verbindung und die Fehlersuche sowie die LabVIEW-Paletten erläutert.
- **LabVIEW-Benutzerhandbuch:** Anhand dieses Handbuchs lernen Sie LabVIEW-Programmierkonzepte, Techniken, Eigenschaften, VIs und Funktionen kennen, die Sie verwenden können, um Test- und Messapplikationen sowie Applikationen für die Datenerfassung, die Gerätesteuerung, die Datenprotokollierung, die Messanalyse und die Berichterzeugung zu entwickeln.
- **LabVIEW-Hilfe:** Diese Hilfedatei soll als Referenz für Informationen zu LabVIEW-Paletten, -Menüs, -Werkzeugen, -VIs und -Funktionen dienen. Die *LabVIEW-Hilfe* enthält auch schrittweise Anleitungen für den Einsatz der LabVIEW-Leistungsmerkmale. Sie können die *LabVIEW-Hilfe* anzeigen, indem Sie **Hilfe»Inhalt und Index** auswählen.

Die *LabVIEW-Hilfe* umfasst Links zu den folgenden Ressourcen:

- *LabVIEW-Tutorium*
- Beispiel-VIs
- *LabVIEW Library PDF*, welche die PDF-Versionen aller LabVIEW-Handbücher und Applikationsinformationen enthält



Hinweis (Macintosh und UNIX) Die *LabVIEW-Hilfe* enthält keine Links zur *LabVIEW Library PDF*. Sie können jedoch in LabVIEW auf die *LabVIEW Library PDF* zugreifen, indem Sie **Hilfe»Zeige gedruckte Handbücher** wählen.

- Ressourcen für technische Unterstützung auf der Internetseite von National Instruments, wie beispielsweise die Developer Zone, die KnowledgeBase und die Bibliothek der Produkthandbücher
- **LabVIEW Measurement-Handbuch:** Anhand dieses Handbuchs erfahren Sie mehr über das Erstellen von Datenerfassungs- und Gerätesteuerungsapplikationen in LabVIEW. Wenn Sie mit LabVIEW noch nicht vertraut sind, lesen Sie das Handbuch *Erste Schritte mit LabVIEW* und das *LabVIEW-Benutzerhandbuch*, bevor Sie sich mit diesem Handbuch befassen.

- **LabVIEW Library PDF** : Mithilfe dieser PDF-Datei können Sie die PDF-Versionen aller LabVIEW-Handbücher und Applikationsinformationen finden. Sie können auf die *LabVIEW Library PDF* zugreifen, indem Sie **Hilfe»Zeige gedruckte Handbücher** auswählen.
- **LabVIEW Development Guidelines** : Anhand dieses Handbuchs lernen Sie, wie VIs erstellt werden, die einfach zu verstehen, anzuwenden und zu überarbeiten sind. In diesem Handbuch wird auch die Projektverfolgung erläutert und Entwurfs- und Dokumentations-techniken beschrieben.



Hinweis Das Handbuch *LabVIEW Development Guidelines* steht nur im LabVIEW Professional Development System zur Verfügung.

- **Using External Code in LabVIEW**: Anhand dieses Handbuchs erfahren Sie, wie Code-Interface-Knoten und externe Subroutinen verwendet werden, um Code zu importieren, der in einer textbasierten Programmiersprache geschrieben wurde. Das Handbuch enthält Informationen zu gemeinsam genutzten externen Subroutinen, zu Funktionsbibliotheken, zu Routinen für das Bearbeiten von Speicher und Dateien und zu Diagnoseroutinen. Das Handbuch befasst sich darüber hinaus auch mit dem Aufrufen von DLLs.



Hinweis Das Handbuch *Using External Code in LabVIEW* steht nur als PDF-Datei zur Verfügung.

- **LabVIEW-Applikationsinformationen**: Die LabVIEW-Applikationsinformationen befassen sich mit den fortgeschrittenen LabVIEW-Konzepten und -Applikationen. Neue und aktualisierte Applikationsinformationen finden Sie auf der Internetseite von National Instruments ni.com.
- **LabVIEW VXI VI Reference Manual**: Mit diesem Handbuch lernen Sie die VXI-VIs für LabVIEW kennen. Dieses Handbuch ist ein begleitendes Handbuch zum *NI-VXI Programmer Reference Manual*, das zum Lieferumfang der VXI-Hardware gehört.



Hinweis Das Handbuch *LabVIEW VXI VI Reference Manual* steht nur als PDF-Datei zur Verfügung.

LabVIEW Beispiel-VIs und Werkzeuge

Die LabVIEW Beispiel-VIs und -Werkzeuge helfen Ihnen beim Gestalten und Entwickeln von VIs.

LabVIEW Beispiel-VIs

LabVIEW enthält Hunderte von Beispiel-VIs, die Sie verwenden und in eigene VIs integrieren können. Sie können ein Beispiel Ihrer Applikation entsprechend anpassen, oder Sie können eines oder mehrere Beispiele kopieren und in ein eigenes VI einfügen. Sie greifen auf die Beispiel-VIs zu, indem Sie **Hilfe»Beispiele** wählen. Weitere Beispiel-VIs finden Sie in der Developer Zone auf der Internetseite von National Instruments ni.com/zone oder in der Example Programs Database ni.com/support/epd.

LabVIEW-Werkzeuge

LabVIEW enthält eine Vielzahl von Werkzeugen, mit denen Sie Messgeräte schnell konfigurieren können, wie beispielsweise die Folgenden. Auf diese Werkzeuge greifen Sie über das Menü **Werkzeuge** zu.

- **(Windows)** Der Measurement & Automation Explorer hilft Ihnen beim Konfigurieren der Hardware und Software von National Instruments.
- **(Macintosh)** Das NI-DAQ-Konfigurationsprogramm hilft Ihnen beim Konfigurieren der DAQ-Hardware von National Instruments.
- **(Macintosh)** Der DAQ Channel Wizard hilft Ihnen zu definieren, welcher Gerätetyp mit den DAQ-Hardwarekanälen verbunden ist. Nachdem Sie einen Kanal definiert haben, speichert der DAQ Channel Wizard die Einstellungen.
- **(Windows und Macintosh)** Der DAQ-Kanalmonitor enthält eine Liste der konfigurierten DAQ-Kanäle.
- **(Windows und Macintosh)** Der DAQ-Lösungsassistent hilft Ihnen dabei, Lösungen für allgemeine DAQ-Applikationen zu finden. Sie können aus Beispiel-VIs wählen oder benutzerdefinierte VIs erstellen.

Einführung in Virtuelle Instrumente

Die LabVIEW-Programme werden als virtuelle Instrumente oder VIs bezeichnet, da mit Erscheinungsbild und Funktion physische Instrumente wie beispielsweise Oszilloskope und Multimeter nachgebildet werden. Jedes VI arbeitet mit Funktionen, durch die Eingaben von der Benutzeroberfläche oder aus anderen Quellen verarbeitet werden. Diese Informationen werden dann angezeigt oder in andere Dateien oder auf andere Computer verschoben.

Ein VI enthält die folgenden drei Komponenten:

- **Frontpanel:** Dient als Benutzeroberfläche.
- **Blockdiagramm:** Enthält den grafischen Quellcode, mit dem die Funktion des VIs definiert wird.
- **Symbol und Anschlussfeld:** Kennzeichnet das VI, sodass Sie das VI in einem anderen VI verwenden können. Ein VI, das einem anderen VI untergeordnet ist, wird als Sub-VI bezeichnet. Ein Sub-VI entspricht den Subroutinen in textbasierten Programmiersprachen.

Weitere Informationen ...

Weitere Informationen zum Erstellen von VIs und Sub-VIs finden Sie in der *LabVIEW-Hilfe*.

Frontpanel

Das Frontpanel ist die Benutzeroberfläche des VIs. In Abbildung 2-1 finden Sie ein Beispiel für ein Frontpanel.



Abbildung 2-1. Beispiel für ein Frontpanel

Das Frontpanel erstellen Sie mit Bedien- und Anzeigeelementen, welche die interaktiven Eingabe- beziehungsweise Ausgabeanschlüsse des VIs darstellen. Bedienelemente sind Knöpfe, Drucktasten, Drehregler und andere Eingabegeräte. Anzeigeelemente sind Graphen, LEDs und andere Anzeigen. Mit Bedienelementen werden also die Eingabegeräte von Instrumenten simuliert und Daten an das Blockdiagramm des VIs übergeben. Mit Anzeigeelementen werden die Ausgabegeräte von Instrumenten simuliert und die Daten angezeigt, die das Blockdiagramm aufnimmt oder erzeugt. Weitere Informationen zum Frontpanel finden Sie in Kapitel 4, [Erstellen des Frontpanels](#).

Blockdiagramm

Nachdem Sie das Frontpanel erstellt haben, können Sie mit Hilfe der grafischen Darstellungen von Funktionen Code hinzufügen, um die Frontpanel-Objekte zu steuern. Dieser grafische Quellcode ist im Blockdiagramm enthalten. Frontpanel-Objekte werden im Blockdiagramm als Anschlüsse dargestellt. Sie können einen Anschluss nicht aus dem Blockdiagramm löschen. Der Anschluss kann nur entfernt werden, wenn Sie das entsprechende Objekt auf dem Frontpanel löschen. Weitere Informationen über das Blockdiagramm finden Sie in Kapitel 5, [Erstellen des Blockdiagramms](#).

Das VI in Abbildung 2-2 umfasst verschiedene primäre Blockdiagrammobjekte – Anschlüsse, Funktionen und Verbindungen.

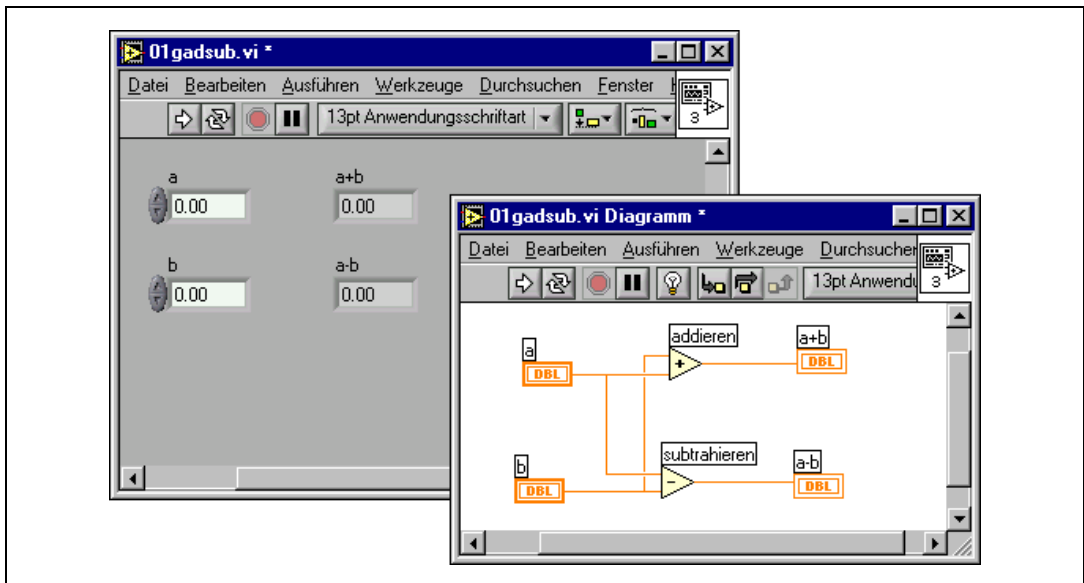


Abbildung 2-2. Beispiel für ein Blockdiagramm und das entsprechende Frontpanel

Anschlüsse



Die Anschlusssymbole verweisen auf den Datentyp des Bedien- oder Anzeigeelements. Beispielsweise stellt ein DBL-Anschluss wie links dargestellt ein numerisches Bedien- oder Anzeigeelement mit doppelter Genauigkeit und Gleitkommalfunktion dar. Weitere Informationen zu den Datentypen in LabVIEW und deren grafische Darstellung finden Sie in

dem Abschnitt *Datentypen für Bedien- und Anzeigeelemente* des Kapitels 5, *Erstellen des Blockdiagramms*.

Anschlüsse sind Eingangs- und Ausgangsports, über die Informationen zwischen dem Frontpanel und dem Blockdiagramm ausgetauscht werden. Daten, die Sie über die Frontpanel-Bedienelemente (**a** und **b** in Abbildung 2-2) eingeben, werden über die Bedienelementanschlüsse an das Blockdiagramm übergeben. Anschließend passieren die Daten die Additions- und Subtraktionsfunktionen. Wenn die Additions- und Subtraktionsfunktionen die internen Berechnungen abgeschlossen haben, werden neue Datenwerte erstellt. Die Daten fließen zu den Anzeigeelementanschlüssen, wo sie das Blockdiagramm verlassen, um erneut an das Frontpanel übergeben und dann mit Hilfe der Frontpanel-Anzeigeelemente angezeigt zu werden.

Knoten

Knoten sind Objekte im Blockdiagramm, die über Eingänge und/oder Ausgänge verfügen und Funktionen ausführen, wenn ein VI ausgeführt wird. Knoten entsprechen Anweisungen, Operatoren, Funktionen und Subroutinen in textbasierten Programmiersprachen. Die Additions- und Subtraktionsfunktionen in Abbildung 2-2 sind Knoten.

Weitere Informationen zu Knoten finden Sie in dem Abschnitt *Blockdiagrammknoten* des Kapitels 5, *Erstellen des Blockdiagramms*.

Verbindungen

Sie übertragen die Daten über Verbindungen zwischen den Blockdiagrammobjekten. In Abbildung 2-2 werden die als Bedien- und Anzeigeelemente fungierenden DBL-Anschlüsse über Verbindungen mit den Additions- und Subtraktionsfunktionen gekoppelt. Jede Verbindung verfügt über eine einzige Datenquelle, die sie jedoch mit vielen Daten lesenden VIs und Funktionen verbinden können. Verbindungen weisen in Abhängigkeit ihres Datentyps unterschiedliche Farben, Stile und Stärken auf. Eine unterbrochene Verbindung wird als eine gestrichelte schwarze Linie dargestellt. Weitere Informationen zu Verbindungen finden Sie in dem Abschnitt *Verbindung von Blockdiagrammobjekten* des Kapitels 5, *Erstellen des Blockdiagramms*.

Strukturen

Strukturen sind grafische Darstellungen der Schleifen und Case-Anweisungen in textbasierten Programmiersprachen. Verwenden Sie Strukturen im Blockdiagramm, um Codeblöcke zu wiederholen und Code bedingungsabhängig oder in einer bestimmten Reihenfolge auszuführen. Beispiele und

weitere Informationen zu Strukturen finden Sie in dem Kapitel 8, *Schleifen und Case-Strukturen*.

Symbol- und Anschlussfeld



Nachdem Sie ein VI-Frontpanel und Blockdiagramm erstellt haben, erstellen Sie das Symbol- und Anschlussfeld, damit das VI als ein Sub-VI verwendet werden kann. Jedes VI verfügt über ein Symbol, das links in der oberen rechten Ecke der Frontpanel- und Blockdiagrammfenster angezeigt wird. Das Symbol ist eine grafische Darstellung eines VIs. Es kann Text, Grafiken oder eine Kombination hieraus enthalten. Wenn Sie ein VI als Sub-VI verwenden, kennzeichnet das Symbol das Sub-VI im Blockdiagramm des VIs. Weitere Informationen zu Symbolen finden Sie in dem Abschnitt *Erstellen eines Symbols* des Kapitels 7, *Erstellen von VIs und Sub-VIs*.



Um das VI als Sub-VI verwenden zu können, müssen Sie ein Anschlussfeld wie links gezeigt erstellen. Das Anschlussfeld ist eine Sammlung von Anschlüssen, die den Bedien- und Anzeigeelementen dieses VIs entsprechen, ähnlich der Parameterliste eines Funktionsaufrufs in textbasierten Programmiersprachen. Mit dem Anschlussfeld werden die Eingänge und Ausgänge definiert, die Sie mit dem VI verbinden möchten, damit Sie es als Sub-VI einsetzen können. Ein Anschlussfeld empfängt Daten an seinen Eingabeanschlüssen, übergibt die Daten über die Frontpanel-Bedienelemente an den Code des Blockdiagramms und empfängt die Ergebnisse über die Frontpanel-Anzeigeelemente an seinen Ausgabeanschlüssen.

Wenn Sie das Anschlussfeld zum ersten Mal anzeigen, sehen Sie ein Anschlussmuster, das Sie bei Bedarf ändern können. Das Anschlussfeld verfügt im Allgemeinen über einen Anschluss für jedes auf dem Frontpanel befindliche Bedien- oder Anzeigeelement. Sie können einem Anschlussfeld bis zu 28 Anschlüsse zuweisen. Wenn Sie mit Änderungen an einem VI rechnen, die neue Eingänge oder Ausgänge fordern, belassen Sie einige Anschlüsse ohne Zuweisung. Weitere Informationen zum Einrichten von Anschlussfeldern finden Sie in dem Abschnitt *Einrichten des Anschlussfeldes* des Kapitels 7, *Erstellen von VIs und Sub-VIs*.



Hinweis Versuchen Sie nicht, einem VI mehr als 16 Anschlüsse zuzuweisen. Die Übersichtlichkeit und Brauchbarkeit des VIs wird durch zu viele Anschlüsse beeinträchtigt.

Verwenden und Anpassen von VIs und Sub-VIs

Nachdem Sie ein VI erstellt und dessen Symbol und Anschlussfeld definiert haben, können Sie es als Sub-VI verwenden. Weitere Informationen zu Sub-VIs finden Sie in dem Abschnitt *Sub-VIs* des Kapitels 7, *Erstellen von VIs und Sub-VIs*.

Sie können VIs als einzelne Dateien speichern, oder Sie können mehrere VIs gruppieren und diese in einer VI-Bibliothek speichern. Weitere Informationen zum Speichern von VIs in Bibliotheken finden Sie in dem Abschnitt *Speichern von VIs* des Kapitels 7, *Erstellen von VIs und Sub-VIs*.

Sie können auch das Erscheinungsbild und das Verhalten eines VIs anpassen. Darüber hinaus können Sie benutzerdefinierte Menüs für jedes von Ihnen erstellte VI anlegen und VIs so konfigurieren, dass die Menüleisten ein- oder ausgeblendet werden. Weitere Informationen zum Anpassen eines VIs finden Sie in dem Kapitel 15, *Anpassen von VIs*.

LabVIEW-Umgebung

Verwenden Sie die LabVIEW-Paletten, -Werkzeuge und -Menüs, um die Frontpanels und Blockdiagramme für VIs zu erstellen. Sie können die **Elemente-** und **Funktions-**Paletten anpassen und verschiedene Optionen für die Arbeitsumgebung festlegen.

Weitere Informationen ...

Weitere Informationen zum Einsatz von Paletten, Menüs und Symbolleisten und das Anpassen der Arbeitsumgebung finden Sie in der *LabVIEW-Hilfe*.

Elementepalette

Die Palette **Elemente** steht nur auf dem Frontpanel zur Verfügung. Wählen Sie **Fenster»Elementpalette**, oder klicken Sie mit der rechten Maustaste auf den Frontpanel-Arbeitsbereich, um die Palette **Elemente** anzuzeigen. Sie können die Palette **Elemente** an einer beliebigen Position im Bildschirm platzieren.

Sie können die Art und Weise ändern, wie die Palette **Elemente** angezeigt wird. Weitere Informationen zum Anpassen der Palette **Elemente** finden Sie in Abschnitt *Anpassen der Paletten Elemente und Funktionen* dieses Kapitels.

Funktionspalette

Die Palette **Funktionen** steht nur im Blockdiagramm zur Verfügung. Wählen Sie **Fenster»Funktionspalette**, oder klicken Sie mit der rechten Maustaste in den Blockdiagramm-Arbeitsbereich, um die Palette **Funktionen** anzuzeigen. Sie können die Palette **Funktionen** an einer beliebigen Position im Bildschirm platzieren.

Sie können die Art und Weise ändern, wie die Palette **Funktionen** angezeigt wird. Weitere Informationen zum Anpassen der Palette **Funktionen** finden Sie in Abschnitt *Anpassen der Paletten Elemente und Funktionen* dieses Kapitels.

Navigieren in den Paletten Elemente und Funktionen

Verwenden Sie die Navigationsschaltflächen in den Paletten **Elemente** und **Funktionen**, um nach Bedienelementen, VIs und Funktionen zu suchen und um hierin zu navigieren. Wenn Sie auf das Symbol einer Unterpalette klicken, ändert sich die gesamte Palette in die von Ihnen gewählte Unterpalette. Sie können auch mit der rechten Maustaste auf ein VI-Symbol in der Palette klicken und den Befehl **VI öffnen** aus dem Kontextmenü wählen, um das VI zu öffnen.

Die Paletten **Elemente** und **Funktionen** enthalten die folgenden Navigationsschaltflächen:



- **Nach oben:** Hiermit gelangen Sie in der Palettenhierarchie eine Ebene nach oben.
- **Suchen:** Die Palette wechselt in den Suchmodus. Im Suchmodus können Sie textbasierte Suchläufe durchführen, um Bedienelemente, VIs oder Funktionen in Paletten zu suchen.
- **Optionen:** Öffnet das Dialogfeld **Optionen: Funktionen durchsuchen**, in dem Sie das Erscheinungsbild von Paletten konfigurieren können.

Werkzeugpalette

Die Palette **Werkzeuge** steht auf dem Frontpanel und im Blockdiagramm zur Verfügung. Je nach ausgewähltem Werkzeug ändert sich die Funktion des Cursors. Wenn Sie ein Werkzeug auswählen, ändert sich das Cursorsymbol in das Werkzeugsymbol. Verwenden Sie Werkzeuge, um Frontpanel- und Blockdiagrammobjekte auszuführen und zu ändern.

Wählen Sie **Fenster»Werkzeugpalette**, um die Palette **Werkzeuge** anzuzeigen. Sie können die Palette **Werkzeuge** an einer beliebigen Position auf dem Bildschirm platzieren.



Tipp Drücken Sie die <Umschalttaste>, und klicken Sie mit der rechten Maustaste, um eine temporäre Version der Palette **Werkzeuge** an der Cursorposition anzuzeigen.

Menüs und Symbolleiste

Verwenden Sie Elemente aus den Menüs und der Symbolleiste, um Frontpanel- und Blockdiagrammobjekte auszuführen und zu ändern. VIs werden mit Hilfe der Symbolleistenschaltflächen ausgeführt.

Menüs

Die Menüs im oberen Bereich eines VI-Fensters enthalten die Elemente, die auch in anderen Applikationen auftreten, wie beispielsweise die Befehle **Öffnen**, **Speichern**, **Kopieren** und **Einfügen** sowie andere LabVIEW-spezifische Elemente. Bei einigen Menüelementen werden auch Tastenkombinationen angegeben.

(Macintosh) Die Menüs werden am oberen Bildschirmrand angezeigt.



Hinweis Wenn sich ein VI im Ausführungsmodus befindet, stehen einige Menüelemente nicht zur Verfügung.

Kontextmenüs

Das am häufigsten verwendete Menü ist das Objekt „Kontextmenü“. Allen LabVIEW-Objekten und den Leerräumen auf dem Frontpanel und im Blockdiagramm sind Kontextmenüs zugeordnet. Verwenden Sie die Elemente des Kontextmenüs, um das Aussehen oder das Verhalten von Frontpanel- oder Blockdiagrammobjekten zu ändern. Für den Zugriff auf das Kontextmenü klicken Sie mit der rechten Maustaste auf das Objekt, das Frontpanel oder das Blockdiagramm.

(Macintosh) Drücken Sie die <Befehlstaste>, und klicken Sie dann auf das Objekt, das Frontpanel oder das Blockdiagramm.

Kontextmenüs im Ausführungsmodus

Wenn ein VI ausgeführt wird oder sich im Ausführungsmodus befindet, verfügen alle Frontpanel-Objekte über eine gekürzte Sammlung an Kontextmenüelementen. Verwenden Sie die Elemente des gekürzten Kontextmenüs, um den Inhalt des Objekts auszuschneiden, zu kopieren oder einzufügen, um den Standardwert des Objekts festzulegen oder um die Beschreibung des Objekts zu lesen.

Einige der komplexeren Bedienelemente verfügen über zusätzliche Optionen. Beispielsweise enthält das Array-Kontextmenü Befehle, um

einen Wertebereich zu kopieren oder um zum letzten Element des Arrays zu wechseln.

Symbolleiste

Verwenden Sie die Schaltflächen der Symbolleiste, um ein VI auszuführen oder zu bearbeiten. Beim Ausführen eines VIs werden Schaltflächen auf der Symbolleiste angezeigt, die Sie zum Debuggen des VIs verwenden können.

Anpassen der Arbeitsumgebung

Sie können die Art und Weise ändern, wie die Paletten **Elemente** und **Funktionen** angezeigt werden, und Sie können das Dialogfeld **Optionen** verwenden, um andere Optionen für die Arbeitsumgebung einzustellen.

Anpassen der Paletten Elemente und Funktionen

Sie können die Paletten **Elemente** und **Funktionen** auf folgende Weisen anpassen:

- Sie können VIs und Bedienelemente zu den Paletten hinzufügen.
- Sie können unterschiedliche Ansichten für unterschiedliche Benutzer einrichten und einige VIs und Funktionen ausblenden, damit LabVIEW für den einen Benutzer einfacher einzusetzen ist, während für den anderen Benutzer die vollständigen Paletten bereitstehen.
- Sie können die integrierten Paletten neu ordnen, um die häufig verwendeten VIs und Funktionen leichter zugänglich zu machen.
- Sie können eine Reihe ActiveX-Steuerelemente in benutzerdefinierte Bedienelemente umwandeln und diese dann den Paletten hinzufügen.
- Sie können den Paletten Werkzeugsätze hinzufügen.

Hinzufügen von VIs und Bedienelementen zur Benutzerbibliothek und zur Instrumentenbibliothek

Die einfachste Methode zum Hinzufügen von VIs und Bedienelementen zu den Paletten **Elemente** und **Funktionen** besteht darin, die VIs und Bedienelemente im Verzeichnis `user.lib` zu speichern. Beim anschließenden Neustart von LabVIEW enthalten die Paletten **Funktionen»Eigene Bibliothek** und **Elemente»Benutzerdef. Elemente** Unterpaletten für jedes Verzeichnis, jede VI-Bibliotheks- (`.lib`) oder jede Menü- (`.menu`) Datei in `user.lib`, sowie Symbole für jede Datei in `user.lib`. LabVIEW aktua-

lisiert die Paletten automatisch, wenn Sie bestimmten Verzeichnissen Dateien hinzufügen oder Dateien hieraus entfernen.

Die Palette **Funktionen»Instrumenten-I/O** entspricht dem Verzeichnis `instr.lib`. Speichern Sie Instrumententreiber in diesem Verzeichnis, um sie über die Palette **Funktionen** einfach zugänglich zu machen.

Wenn Sie den Paletten **Elemente** und **Funktionen** mit dieser Methode VIs oder Bedienelemente hinzufügen, können Sie die exakte Position der VIs oder Bedienelemente auf den Paletten nicht bestimmen.

Erstellen und Bearbeiten von Palettenansichten

Wenn Sie die genaue Position der von Ihnen den Paletten **Elemente** und **Funktionen** hinzugefügten VIs und Bedienelemente bestimmen möchten, müssen Sie eine Palettenansicht erstellen. LabVIEW speichert die Informationen zu den Paletten **Elemente** und **Funktionen** im Verzeichnis `labview/menus`. Das Verzeichnis `menus` enthält wiederum Verzeichnisse für jede Ansicht, die Sie erstellen oder installieren. Wenn Sie LabVIEW in einem Netzwerk ausführen, können Sie individuelle `menus`-Verzeichnisse für jeden Benutzer definieren, womit es einfacher wird, Ansichten auf andere Personen zu übertragen.

Wenn Sie eine neue Ansicht einer Palette erstellen, kopiert LabVIEW das Original der integrierten Palette, das sich im Verzeichnis `labview\menus` befindet, bevor irgendwelche Änderungen vorgenommen werden. Mit dem Schutz der integrierten Paletten wird sichergestellt, dass Sie mit den Paletten experimentieren können, ohne die Originalansicht zu beschädigen.

Wie Ansichten von LabVIEW gespeichert werden

Die Dateien mit den Erweiterungen `.mnu` und `.lib` enthalten jeweils eine Palette **Elemente** und eine Palette **Funktionen**. Darüber hinaus enthält jede Datei ein Symbol für die Paletten **Elemente** und **Funktionen**. Sie müssen jede Unterpalette, die Sie erstellen, in einer separaten `.mnu`-Datei speichern.

Wenn Sie eine Ansicht auswählen, prüft LabVIEW das Verzeichnis `menus` auf Vorhandensein eines Verzeichnisses, das dieser Ansicht entspricht. Die Hauptpaletten **Elemente** und **Funktionen** und die Unterpaletten werden aus der Datei `root.mnu` in dem Verzeichnis erstellt, das LabVIEW jedes Mal automatisch erstellt, wenn Sie eine Ansicht erstellen.

Für jedes VI oder Bedienelement erstellt LabVIEW ein Symbol auf der Palette. Für jedes Unterverzeichnis, für jede `.mnu`-Datei oder jede `.lib`-Datei erstellt LabVIEW eine Unterpalette auf der Palette.

Erstellen von ActiveX-Unterpaletten

Wenn Sie im Frontpanel ActiveX-Steuerelemente verwenden, wählen Sie **Werkzeuge»Fortgeschritten»Importier ActiveX-Elemente**, um eine Reihe ActiveX-Steuerelemente in benutzerdefinierte Bedienelemente umzuwandeln und diese der Palette **Elemente** hinzuzufügen. LabVIEW speichert die Bedienelemente standardmäßig im Verzeichnis `user.lib`, da alle Dateien und Verzeichnisse in `user.lib` automatisch in den Paletten angezeigt werden.

Darstellen von Werkzeugsätzen in den Paletten

Werkzeugsätze, die Sie im Verzeichnis `vi.lib\addons` installieren, werden automatisch auf der obersten Ebene der Paletten **Elemente** und **Funktionen** angezeigt, nachdem LabVIEW neu gestartet wurde. Wenn Sie Werkzeugsätze an anderer Stelle installiert haben, können Sie diese Werkzeugsätze für einen einfacheren Zugriff in das Verzeichnis `addons` verschieben.



Vorsicht! Speichern Sie *keine eigenen* VIs und Bedienelemente im Verzeichnis `vi.lib`, da LabVIEW diese Dateien beim Aktualisieren überschreibt. Speichern Sie eigene VIs und Bedienelemente im Verzeichnis `user.lib`, damit sie den Paletten **Elemente** und **Funktionen** hinzugefügt werden.

Festlegen von Optionen für die Arbeitsumgebung

Wählen Sie **Werkzeuge»Optionen**, um LabVIEW anzupassen. Verwenden Sie das Dialogfeld **Optionen**, um Optionen für Pfade, Leistungs- und Speicher Aspekte, Frontpanels, Blockdiagramme, Rückgängigmachen, Fehlersuchprogramme, Farben, Schriftarten, Drucken, das Fenster **Revisions-Historie**, Datums- und Uhrzeitformate und andere LabVIEW-Funktionen festzulegen.

Verwenden Sie das oberste Pulldown-Menü im Dialogfeld **Optionen**, um in den unterschiedlichen Kategorien mit Optionen Ihre Auswahl zu treffen.

Wie Optionen von LabVIEW gespeichert werden

Sie müssen Optionen nicht manuell bearbeiten und auch deren exaktes Format nicht kennen, denn diese Aufgaben erledigt das Dialogfeld **Optionen** für Sie. LabVIEW speichert Optionen auf jeder Plattform auf andere Weise.

Windows

LabVIEW speichert Optionen in der Datei `labview.ini` im LabVIEW-Verzeichnis. Das Dateiformat gleicht dem anderer `.ini`-Dateien. Es beginnt mit einem LabVIEW-Abschnittsmarker, gefolgt vom Optionsnamen und den Werten, wie beispielsweise `offscreenUpdates=True`.

Wenn Sie eine andere Optionsdatei verwenden möchten, geben Sie die Datei in der Verknüpfung an, die Sie zum Starten von LabVIEW verwenden. Um beispielsweise eine auf dem Computer gespeicherte Optionsdatei mit Namen `lvrc` anstelle der Datei `labview.ini` zu verwenden, klicken Sie mit der rechten Maustaste auf dem Desktop auf das Symbol LabVIEW, und wählen Sie dann **Eigenschaften** aus dem Kontextmenü. Klicken Sie auf die Registerkarte **Verknüpfung**, und geben Sie `labview -pref lvrc` in das Textfeld **Ziel** ein.

Macintosh

LabVIEW speichert Optionen in der Textdatei `LabVIEW Preferences` im Ordner **System»Preferences**.

Wenn Sie eine andere Optionsdatei verwenden möchten, kopieren Sie die Datei `LabVIEW Preferences` in den Ordner **LabVIEW**, und nehmen Sie im Dialogfeld **Optionen** die gewünschten Änderungen an den Optionen vor. Beim Start sucht LabVIEW zunächst im Ordner **LabVIEW** nach einer Optionsdatei. Kann hier keine Datei gefunden werden, wird im Ordner **System** nach der Datei gesucht. Kann hier ebenfalls keine Datei gefunden werden, wird im Ordner **System** eine neue Datei erstellt. LabVIEW schreibt alle Änderungen, die Sie im Dialogfeld **Optionen** vornehmen, in die erste `LabVIEW Preferences`-Datei, die gefunden werden kann.

UNIX

LabVIEW speichert Optionen in der Datei `.labviewrc` im Stammverzeichnis. Wenn Sie im Dialogfeld **Optionen** eine Option ändern, schreibt LabVIEW die Änderung in die Datei `.labviewrc`. Sie können eine Datei `labviewrc` im Programmverzeichnis erstellen, um Optionen zu speichern, die für alle Benutzer die gleichen sind, wie beispielsweise der VI-Such-

pfad. Verwenden Sie die Datei `.labviewrc` zum Speichern von Optionen, die für jeden Benutzer unterschiedlich sind, wie beispielsweise Schriftart- und Farbeinstellungen, da Einträge in der Datei `.labviewrc` im Stammverzeichnis Konflikte verursachende Einträge im Programmverzeichnis außer Kraft setzen.

Wenn Sie die LabVIEW-Dateien beispielsweise in `/opt/labview` gespeichert haben, liest LabVIEW zunächst die Optionen aus `/opt/labview/labviewrc`. Wenn Sie im Dialogfeld **Optionen** eine Option ändern, wie beispielsweise die Schriftart einer Applikation, schreibt LabVIEW diese Änderung in die Datei `.labviewrc`. Beim nächsten Start von LabVIEW wird dann die Option für die Applikationsschriftart in der Datei `.labviewrc` anstelle der standardmäßigen Applikationsschriftart, wie definiert in `/opt/labview/labviewrc`, verwendet.

Optionseinträge bestehen aus einem Optionsnamen gefolgt von einem Doppelpunkt und dem Wert. Der Optionsname ist der Programmcode gefolgt von einem Punkt (.) und einer Option. Bei der Suche nach Optionsnamen wird die Groß-/Kleinschreibung berücksichtigt. Sie können den Optionswert in doppelte oder einfache Anführungszeichen einschließen. Um beispielsweise als Standardgenauigkeit Double zu verwenden, fügen Sie den folgenden Eintrag in die Datei `.labviewrc` im Stammverzeichnis ein.

```
labview.defPrecision : double
```

Wenn Sie eine andere Optionsdatei verwenden möchten, geben Sie die Datei beim Start von LabVIEW auf der Befehlszeile an. Wenn Sie beispielsweise eine Datei mit Namen `lvrc` im Verzeichnis `test` anstelle der Datei `.labviewrc` verwenden möchten, geben Sie `labview -pref /test/lvrc` ein. LabVIEW schreibt alle Änderungen, die Sie im Dialogfeld **Optionen** vornehmen, in die Optionsdatei `lvrc`. Wenn Sie in der Befehlszeile eine Optionsdatei angeben, liest LabVIEW zwar die Datei `labviewrc` im Programmverzeichnis, jedoch setzt die in der Befehlszeile angegebene Optionsdatei alle Konflikte verursachenden Einträge in der Datei im Programmverzeichnis außer Kraft.

Erstellen des Frontpanels

Das Frontpanel ist die Benutzeroberfläche eines VIs. Im Allgemeinen entwerfen Sie zuerst das Frontpanel und dann das Blockdiagramm für die Aufgaben, die an den Ein- und Ausgaben durchgeführt werden, die Sie auf dem Frontpanel zusammengestellt haben. Weitere Informationen über das Blockdiagramm finden Sie in Kapitel 5, [Erstellen des Blockdiagramms](#).

Das Frontpanel erstellen Sie mit Bedien- und Anzeigeelementen, welche die interaktiven Eingabe- beziehungsweise Ausgabeanschlüsse des VIs darstellen. Bedienelemente sind Knöpfe, Drucktasten, Drehregler und andere Eingabegeräte. Anzeigeelemente sind Graphen, LEDs und andere Anzeigen. Mit den Bedienelementen werden die Eingänge von Instrumenten simuliert und Daten an das Blockdiagramm des VIs übergeben. Mit Anzeigeelementen werden die Ausgänge von Instrumenten simuliert und die Daten angezeigt, die das Blockdiagramm sammelt oder erzeugt.

Wählen Sie **Fenster»Elementpalette**, um die Palette **Elemente** anzuzeigen, wählen Sie dann die Bedien- und Anzeigeelemente auf der Palette **Elemente** aus, und platzieren Sie diese auf dem Frontpanel.

Weitere Informationen ...

Weitere Informationen zum Entwerfen und Konfigurieren des Frontpanels finden Sie in der *LabVIEW-Hilfe*.

Konfigurieren von Objekten auf dem Frontpanel

Sie können das Frontpanel über die Kontextmenüs der Bedien- und Anzeigeelemente anpassen, indem Sie die Reihenfolge der Frontpanel-Objekte festlegen und indem Sie importierte Grafiken einbinden. Sie können Frontpanel-Objekte auch manuell in der Größe ändern und so einstellen, dass die Größe automatisch angepasst wird, wenn sich die Größe des Fensters ändert.

Ein- und Ausblenden von optionalen Elementen

Bedien- und Anzeigeelemente auf dem Frontpanel verfügen über optionale Elemente, die Sie ein- und ausblenden können. Sie können eine Liste der verfügbaren Elemente anzeigen, indem Sie mit der rechten Maustaste auf ein Objekt klicken und aus dem Kontextmenü **Sichtbare Objekte** wählen. Die meisten Objekte verfügen über eine Beschriftung und über einen Untertitel. Weitere Informationen zu Beschriftungen und Untertiteln finden Sie in Abschnitt *Beschriftungen* dieses Kapitels.

Umwandeln von Bedienelementen in Anzeigeelemente und umgekehrt

LabVIEW konfiguriert Objekte auf der Palette **Elemente** als Bedienelemente oder als Anzeigeelemente basierend auf dem jeweils typischen Einsatzspektrum. Wenn Sie beispielsweise von der Palette **Elemente»Boolesch** einen Umschalter wählen, erscheint dieser auf dem Frontpanel als Bedienelement, da ein Umschalter normalerweise ein Eingabegerät ist. Wenn Sie jedoch eine LED wählen, erscheint diese auf dem Frontpanel als Anzeigeelement, da eine LED normalerweise als Ausgabegerät dient.

Einige Paletten enthalten ein Bedien- und ein Anzeigeelement für das gleiche Objekt. Beispielsweise enthält die Palette **Elemente»Numerisch** ein digitales Bedienelement und ein digitales Anzeigeelement.

Sie können Bedienelemente in Anzeigeelemente umwandeln und umgekehrt, indem Sie mit der rechten Maustaste auf das Objekt klicken und aus dem Kontextmenü **In Bedienelement umwandeln** oder **In Anzeigeelement umwandeln** wählen.

Ersetzen von Frontpanel-Objekten

Sie können ein Frontpanel-Objekt durch ein anderes Bedien- oder Anzeigeelement ersetzen. Wenn Sie mit der rechten Maustaste auf ein Objekt klicken und aus dem Kontextmenü **Ersetzen** wählen, wird eine temporäre Palette **Elemente** angezeigt, auch wenn die Palette **Elemente** bereits geöffnet ist. Wählen Sie ein Bedien- oder ein Anzeigeelement aus der temporären Palette **Elemente**, um das aktuell markierte Objekt auf dem Frontpanel zu ersetzen.

Mit der Auswahl von **Ersetzen** aus dem Kontextmenü werden so viele Informationen wie möglich über das ursprüngliche Objekt beibehalten, wie beispielsweise der Name, die Beschreibung, die Standarddaten, Datenflussrichtung (Bedien- oder Anzeigeelement), Farbe, Größe und so weiter. Das neue Objekt behält jedoch seinen eigenen Datentyp bei. Verbindungen

vom Anschluss des Objekts oder lokale Variablen verbleiben im Blockdiagramm, können jedoch unterbrochen werden. Wenn Sie beispielsweise einen numerischen Anschluss durch einen String-Anschluss ersetzen, bleibt die ursprüngliche Verbindung im Blockdiagramm erhalten, wird jedoch unterbrochen.

Je mehr das neue Objekt dem zu ersetzenden Objekt ähnelt, desto mehr der ursprünglichen Eigenschaften können beibehalten werden. Wenn Sie beispielsweise einen Schieber durch einen anders gearteten Schieber ersetzen, weist der neue Schieber die gleiche Höhe und Skala, den gleichen Wert und Namen, die gleiche Beschreibung und so weiter auf. Wenn Sie den Schieber jedoch durch ein String-Bedienelement ersetzen, behält LabVIEW nur den Namen, die Beschreibung und die Datenflussrichtung bei, da ein Schieber nicht viel mit einem String-Bedienelement gemein hat.

Sie können auch Objekte aus der Zwischenablage einfügen, um vorhandene Frontpanel-Bedien- und Anzeigeelemente zu ersetzen. Bei dieser Methode werden keine Eigenschaften des alten Objekts beibehalten, jedoch bleibt die Verbindungen zum Objekt bestehen.

Festlegen von Tastenkombinationen für Bedienelemente

Sie können Bedienelementen Tastenkombination zuweisen, damit die Benutzer auch ohne Maus auf dem Frontpanel navigieren können. Klicken Sie mit der rechten Maustaste auf das Bedienelement, und wählen Sie aus dem Kontextmenü **Fortgeschritten»Tastenbelegung**, um das Dialogfeld **Tastenbelegung** anzuzeigen.

Wenn ein Benutzer die Tastenkombination eingibt, während das VI ausgeführt wird, erhält das hiermit verbundene Bedienelement den Fokus. Wenn es sich bei dem Bedienelement um ein Text-Bedienelement handelt, hebt LabVIEW den Text hervor, damit Sie ihn bearbeiten können. Wenn es sich um ein boolesches Bedienelement handelt, ändert sich der Status der Schaltfläche.

Der Kontextmenübefehl **Fortgeschritten»Tastenbelegung** ist für Anzeigeelemente ausgeblendet, da Sie keine Daten in ein Anzeigeelement eingeben können.

Steuern des Schaltflächenverhaltens per Tastenbelegung

Sie können verschiedenen Schaltflächen, die das Verhalten eines Frontpanels steuern, Funktionstasten zuweisen. Sie können eine Schaltfläche in einem VI so definieren, dass sie sich wie ein Dialogfeld

verhält, so dass das Drücken der <Eingabe>-Taste gleichbedeutend mit dem Klicken auf die Standardschaltfläche ist.

(Macintosh und Sun) Das Drücken der <Return>-Taste ist gleichbedeutend mit dem Klicken auf die Standardschaltfläche.

Wenn Sie die <Eingabe>- oder die <Return>-Taste mit einer Dialogfeldschaltfläche verbinden, zieht LabVIEW um diese Schaltfläche automatisch einen dickeren Rahmen.

Wenn Sie die <Eingabe>- oder die <Return>-Taste mit einem Bedienelement verbinden, kann kein String-Bedienelement auf diesem Frontpanel einen CR-Befehl (Carriage Return, Absatzschaltung, Eingabetaste) empfangen. Dementsprechend sind alle Strings auf diesem Frontpanel auf eine einzige Zeile beschränkt. Sie können Bildlaufleisten verwenden, um in längeren Strings zu navigieren.

Wenn Sie zu einem booleschen Bedienelement wechseln und die <Eingabe>- oder <Return>-Taste drücken, ändert sich das boolesche Bedienelement auch dann, wenn die <Eingabe>- oder <Return>-Taste bei einem anderen Bedienelement als Tastenkombination verwendet wird. Die zugewiesene <Eingabe>- oder <Return>-Tastenkombination ist nur gültig, wenn kein boolesches Bedienelement den Fokus hat.

Festlegen der Navigationsreihenfolge von Frontpanel-Objekten

Sie können die Navigationsreihenfolge von Frontpanel-Objekten festlegen, indem Sie **Bearbeiten»Tab-Reihenfolge setzen** wählen. Anschließend können Sie mit der <Tabulator>-Taste zwischen den einzelnen Objekten wechseln, während Sie ein VI ausführen.

Für das Aktivieren der Bedien- und Anzeigeelemente auf einem Frontpanel gibt es eine Reihenfolge, die als Panel-Anordnung bezeichnet wird und die von deren Position auf dem Frontpanel unabhängig ist. Das erste Bedien- oder Anzeigeelement, das Sie auf dem Frontpanel erstellen, ist Element 0, das zweite ist 1 und so weiter. Wenn Sie ein Bedien- oder Anzeigeelement löschen, wird die Panel-Anordnung automatisch angepasst.

Mit der Panel-Anordnung wird die Navigationsreihenfolge beim Ausführen eines VIs festgelegt. Die Panel-Anordnung bestimmt darüber hinaus die Reihenfolge, in der Bedien- und Anzeigeelemente in den Datensätzen der Datenprotokolldateien erscheinen, wenn Sie ein Protokoll der Frontpanel-Daten erstellen. Weitere Informationen zum Protokollieren

von Daten finden Sie in Abschnitt *Protokollieren von Frontpanel-Daten* des Kapitels 13, *Datei-I/O*.

Um zu verhindern, dass Benutzer mit Hilfe der <Tabulator>-Taste auf ein Bedienelement zugreifen, während das VI ausgeführt wird, aktivieren Sie das Kontrollkästchen **Dieses Bedienelement bei Auswahl übergehen** im Dialogfeld **Tastenbelegung**.

Zuweisen von Farben zu Objekten

Sie können die Farbe der meisten, jedoch nicht aller Objekte ändern. Beispielsweise werden für Blockdiagrammanschlüsse von Frontpanel-Objekten und für Verbindungen bestimmte Farben für den Typ und die Darstellung der Daten verwendet, die hierüber übertragen werden, daher können Sie diese Farben nicht ändern.

Aktivieren Sie das Farbwerkzeug, und klicken Sie mit der rechten Maustaste auf ein Objekt oder auf den Arbeitsbereich, um Frontpanel-Objekten oder dem Frontpanel- oder Blockdiagramm-Arbeitsbereich Farbe zuzuweisen oder diese zu ändern. Sie können auch die Standardfarbe der meisten Objekte ändern, indem Sie **Werkzeuge»Optionen** und anschließend aus dem obersten Pulldown-Menü den Befehl **Farben** wählen.

Verwenden von importierten Grafiken

Sie können Grafiken aus anderen Applikationen importieren, um sie als Frontpanel-Hintergrund, als Elemente in Ring-Bedienelementen und als Teile anderer Bedien- und Anzeigeelemente zu verwenden. Lesen Sie die Applikationsinformationen *LabVIEW Custom Controls, Indicators, and Type Definitions* für weitere Informationen zum Verwenden von Grafiken in Bedienelementen.

Zum Importieren einer Grafik kopieren Sie diese in die Zwischenablage und fügen sie dann auf dem Frontpanel ein. Sie können auch **Bearbeiten»Bild aus Datei importieren** wählen.

Beispiele für Bedienelemente mit importierten Grafiken finden Sie in der Bibliothek `examples\general\controls\custom.llb`.

Gruppieren und Sperren von Objekten

Verwenden Sie das Positionierwerkzeug, um die Frontpanel-Objekte auszuwählen, die Sie gruppieren und sperren möchten. Klicken Sie auf der Symbolleiste auf die Schaltfläche **Neuordnen**, und wählen Sie aus dem Kontextmenü **Gruppe** oder **Sperre**. Gruppierte Objekte behalten ihre relative Anordnung bei, wohingegen gesperrte Objekte ihre Position auf dem Frontpanel beibehalten und nicht gelöscht werden können. Sie können Objekte gleichzeitig gruppieren und sperren. Andere Werkzeuge als das Positionierwerkzeug funktionieren mit gruppierten oder gesperrten Objekten normal.

Ändern der Größe von Objekten

Sie können die Größe der meisten Frontpanel-Objekte ändern. Wenn Sie das Positionierwerkzeug auf ein Objekt bewegen, dessen Größe Sie ändern können, erscheinen an den Ecken eines rechteckigen Objekts Größenziehpunkte, und bei einem kreisförmigen Objekt werden Größenänderungskreise angezeigt. Wenn Sie die Größe eines Objekts ändern, bleibt die Schriftgröße die gleiche. Wenn die Größe einer Objektgruppe geändert wird, wird die Größe aller Objekte innerhalb der Gruppe geändert.

Einige Objekte lassen sich nur horizontal oder vertikal in der Größe ändern, wie beispielsweise digitale numerische Bedien- und Anzeigeelemente. Andere behalten ihre Proportionen bei, wenn sie in der Größe geändert werden, wie beispielsweise Drehknöpfe. Das Positionierwerkzeug bleibt gleich, jedoch lässt sich die gestrichelte Linie um das Objekt nur in eine Richtung bewegen.

Beim Ändern der Größe eines Objekts können Sie die Vergrößerungsrichtung manuell begrenzen. Um die Vergrößerung vertikal oder horizontal zu begrenzen oder um die aktuellen Proportionen eines Objekts beizubehalten, halten Sie die <Umschalt>-Taste gedrückt, während Sie auf das Objekt klicken und es ziehen. Um ein Objekt um seinen Mittelpunkt in der Größe zu ändern, halten Sie <STRG-Umschalt> gedrückt, und klicken mit dem Positionierwerkzeug auf die Ziehpunkte.

(Macintosh) Drücken Sie <Option-Umschalt>. **(Sun)** Drücken Sie <Meta-Umschalt>. **(HP-UX und Linux)** Drücken Sie die <Alt-Umschalt>.

Skalieren von Frontpanel-Objekten

Sie können Frontpanel-Objekte so einstellen, dass sie skaliert oder in Relation zur Fenstergröße automatisch in der Größe angepasst werden, wenn Sie die Größe des Frontpanel-Fensters ändern. Sie können nur ein Objekt auf dem Frontpanel skalieren, oder Sie können festlegen, dass alle Objekte auf dem Frontpanel skaliert werden. Mehrere Objekte auf dem Frontpanel können Sie jedoch erst dann gemeinsam skalieren, wenn Sie für alle die Skalierung aktiviert haben oder wenn Sie die Objekte zuerst gruppieren. Zum Skalieren eines Objekts markieren Sie das Objekt und wählen dann **Bearbeiten»Skaliere Objekt mit Frontpanel**.

Wenn Sie für ein einzelnes Objekt die Skalierung aktiviert haben, ändert das Objekt automatisch seine Größe in Relation zu jeder beliebigen Änderung der Größe des Frontpanel-Fensters. Die anderen Objekte auf dem Frontpanel positionieren sich selbst neu in Übereinstimmung mit ihrer vorherigen Platzierung auf dem Frontpanel, werden jedoch nicht skaliert und somit nicht an die neue Fenstergröße angepasst.

Sofort nachdem Sie für ein einzelnes Objekt auf dem Frontpanel die automatische Skalierung aktiviert haben, werden verschiedene Bereiche auf dem Frontpanel mit grauen Linien hervorgehoben, wie in Abbildung 4-1 gezeigt. Diese Bereiche kennzeichnen die Positionen der anderen Frontpanel-Objekte in Relation zu dem Objekt, das Sie skalieren möchten. Wenn Sie die Größe des Frontpanel-Fensters ändern, positioniert sich das Objekt, für das Sie die Skalierung aktiviert haben, automatisch neu in Relation zu seiner ursprünglichen Position, und dessen Größe wird angepasst. Die grauen Linien verschwinden, wenn Sie das VI ausführen.



Abbildung 4-1. Frontpanel mit Objekt, für das die Skalierung aktiviert ist

Wenn LabVIEW Objekte automatisch skaliert, gelten die gleichen Konventionen wie beim manuellen Ändern der Größe von Objekten. Beispielsweise können einige Objekte nur horizontal oder vertikal in der Größe geändert werden, und der Schriftgrad bleibt der gleiche, wenn Sie die Größe eines Objekts ändern.

Nachdem LabVIEW ein Objekt automatisch skaliert hat, kann es vorkommen, dass das Objekt nicht wieder seine exakte ursprüngliche Größe annimmt, wenn Sie das Fenster wieder in die ursprüngliche Größe und Position bringen. Bevor Sie das VI speichern, wählen Sie daher **Bearbeiten»Rückgängig**, um die Originalgrößen von Frontpanel-Fenster und Objekten wiederherzustellen.

Sie können die Skalierung für ein Array aktivieren, oder Sie können die Objekte innerhalb eines Arrays skalieren. Wenn Sie die Skalierung für ein Array aktivieren, passen Sie die Anzahl der Zeilen und Spalten an, die im Array angezeigt werden. Wenn Sie die Skalierung für die Objekte im Array aktivieren, wird immer die gleiche Anzahl Zeilen und Spalten im Array angezeigt, allerdings in unterschiedlicher Größe.

Hinzufügen von Leerraum auf dem Frontpanel ohne Größenänderung des Fensters

Sie können dem Frontpanel Leerraum hinzufügen, ohne die Größe des Fensters zu ändern. Um den Leerraum zwischen gehäuften oder zu nahe gruppierten Objekten zu vergrößern, drücken Sie die <Strg>-Taste und klicken mit dem Positionierwerkzeug in den Frontpanel-Arbeitsbereich. Während Sie die Taste gedrückt halten, ziehen Sie einen Bereich in der Größe auf, wie er eingefügt werden soll.

(Macintosh) Drücken Sie die <Option>-Taste. **(Sun)** Drücken Sie die <Meta>-Taste. **(HP-UX und Linux)** Drücken Sie die <Alt>-Taste.

Ein von einem gestrichelten Rahmen umschlossenes Rechteck zeigt, wo der Leerraum eingefügt wird. Lassen Sie die Taste los, und fügen Sie den Leerraum ein.

Frontpanel-Bedien- und Anzeigeelemente

Verwenden Sie die auf der Palette **Elemente** befindlichen Bedien- und Anzeigeelemente, um das Frontpanel zu erstellen. Bedienelemente sind Knöpfe, Drucktasten, Drehregler und andere Eingabegeräte. Anzeigeelemente sind Graphen, LEDs und andere Anzeigen. Mit den Bedienelementen werden die Eingänge von Instrumenten simuliert und Daten an das Blockdiagramm des VIs übergeben. Mit Anzeigeelementen werden die Ausgänge von Instrumenten simuliert und die Daten angezeigt, die das Blockdiagramm sammelt oder erzeugt.

3D- und klassische Bedien- und Anzeigeelemente

Viele Frontpanel-Objekte verfügen über ein dreidimensionales Erscheinungsbild mit hoher Farbanzahl. Stellen Sie den Monitor auf wenigstens 16-Bit-Farben ein, damit die Objekte optimal dargestellt werden.

Für die 3D-Frontpanel-Objekte stehen auch die entsprechenden zweidimensionalen Objekte mit geringer Farbanzahl zur Verfügung. Verwenden Sie die 2D-Bedien- und Anzeigeelemente, die sich auf der Palette **Elemente»Elemente klass. Format** befinden, um VIs für Monitore mit 256 Farben oder 16-Bit-Farben zu erstellen.

Schieberegler, Drehknöpfe, Drehregler und numerische Anzeigen

Verwenden Sie die auf den Paletten **Elemente»Numerisch** und **Elemente»Elemente klass. Format»Numerisch** befindlichen numerischen Bedien- und Anzeigeelemente, um Schieberegler, Drehknöpfe, Drehregler und numerische Anzeigen nachzubilden. Die Palette umfasst auch Farbboxen und eine Farbrampe für das Festlegen von Farbwerten. Verwenden Sie die numerischen Bedien- und Anzeigeelemente, um numerische Daten einzugeben und anzuzeigen.

Schieberegler und Anzeigen

Die Schieberegler und Anzeigen beinhalten vertikale und horizontale Schieberegler, einen Tank und ein Thermometer. Sie ändern den Wert eines Schiebereglers oder einer Anzeige, indem Sie den Schieber mit Hilfe des Bedienwerkzeugs an eine neue Position schieben, indem Sie im Schiebereglerobjekt auf einen Punkt klicken oder indem Sie die optionale numerische Anzeige verwenden. Wenn Sie den Schieber an eine neue Position ziehen und das VI während der Änderung aktiv ist, übergibt das Bedienelement Zwischenwerte an das VI, und zwar abhängig davon, wie häufig das VI das Bedienelement liest.

Schieberegler oder Anzeigen können mehr als einen Wert anzeigen. Klicken Sie mit der rechten Maustaste auf das Objekt, und wählen Sie **Schieber hinzufügen** aus dem Kontextmenü, um weitere Schieber hinzuzufügen. Der Datentyp eines Bedienelements mit mehreren Schiebern ist ein Cluster, der jeden der numerischen Werte enthält. Weitere Informationen zu Clustern finden Sie in Abschnitt [Cluster](#) des Kapitels 9, [Gruppieren von Daten mit Hilfe von Strings, Arrays und Clustern](#).

Rotatorische Bedienelemente und Anzeigen

Zu den rotatorischen Bedienelementen und Anzeigen gehören Drehknöpfe, Drehregler, Runduminstrumente und Drehspulinstrumente. Die rotatorischen Bedienelemente funktionieren ähnlich wie die Schieberegler und Anzeigen. Sie ändern den Wert eines rotatorischen Bedien- oder Anzeigeelements, indem Sie die Zeiger verschieben, indem Sie auf einen Punkt im rotatorischen Objekt klicken oder indem Sie die optionale numerische Anzeige verwenden.

Rotatorische Bedien- oder Anzeigeelemente können mehr als einen Wert anzeigen. Klicken Sie mit der rechten Maustaste auf das Objekt, und wählen Sie aus dem Kontextmenü **Zeiger hinzufügen**, um neue Zeiger hinzuzufügen. Der Datentyp eines Bedienelements mit mehreren Zeigern ist ein Cluster, der jeden der numerischen Werte enthält. Weitere

Informationen zu Clustern finden Sie in Abschnitt *Cluster* des Kapitels 9, *Gruppieren von Daten mit Hilfe von Strings, Arrays und Clustern*.

Numerische Bedienelemente und Anzeigen

Numerische Bedienelemente und Anzeigen stellen die einfachste Möglichkeit zum Eingeben und Anzeigen von numerischen Daten dar. Sie können diese Frontpanel-Objekte horizontal in der Größe ändern, um weitere Stellen anzuzeigen. Sie können den Wert eines numerischen Bedienelements oder einer numerischen Anzeige ändern, indem Sie wie nachstehend beschrieben vorgehen:

- Verwenden Sie das Bedienwerkzeug oder das Beschriftungswerkzeug, klicken Sie hiermit in das Fenster der numerischen Anzeige, und geben Sie die Werte über die Tastatur ein.
- Klicken Sie mit dem Bedienwerkzeug auf die Nach-Oben- oder Nach-Unten-Pfeiltasten eines digitalen Bedienelements.
- Platzieren Sie den Cursor mit Hilfe des Bedien- oder des Beschriftungswerkzeugs rechts neben der Ziffer, die Sie ändern möchten, und drücken Sie dann die Nach-Oben- oder Nach-Unten-Taste auf der Tastatur.

Farbboxen

In einer Farbbox wird die Farbe angezeigt, die einem festgelegten Wert entspricht. So können Sie Farbboxen verwenden, um unterschiedliche Bedingungen wie beispielsweise Werte außerhalb des Bereichs anzuzeigen. Der Farbwert wird als Hexadezimalzahl im Format RRGGBB ausgedrückt. Die ersten beiden Ziffern steuern den Farbwert für Rot. Das zweite Ziffern paar steuert den Farbwert für Grün. Die letzten beiden Ziffern steuern den Farbwert für Blau.

Sie legen die Farbe der Farbbox fest, indem Sie mit dem Bedienwerkzeug oder dem Farbwerkzeug darauf klicken, um die Farbwahltablette anzuzeigen.

Farbrampen

Bei einer Farbrampe wird der numerische Wert mit Hilfe von Farbe angezeigt. Sie konfigurieren eine Farbskala, die aus wenigstens zwei beliebigen Markern besteht, von denen jeder über einen numerischen Wert und eine entsprechende Anzeigefarbe verfügt. Wenn sich der Eingangswert ändert, ändert sich die Farbe in die Farbe, die diesem neuen Wert entspricht. Farbrampen eignen sich für die visuelle Anzeige von Datenbereichen, wie beispielsweise einem Warnungsbereich für den Fall,

dass ein Runduminstrument in einen gefahrenträchtigen Wertebereich eintritt. Sie können beispielsweise auch eine Farbrampe verwenden, um die Farbskala für Intensitätsdiagramme und -graphen festzulegen. Weitere Informationen zu Intensitätsdiagrammen und -graphen finden Sie in Abschnitt *Intensitätsgraphen und -diagramme* des Kapitels 11, *Graphen und Diagramme*.

Klicken Sie mit der rechten Maustaste auf die Farbrampe, und verwenden Sie die Optionen aus dem Kontextmenü, um das Aussehen, die Größe, die Farbe und die Anzahl der Farben anzupassen.

Sie können auch jedem Drehknopf, jedem Drehregler und jedem Runduminstrument auf dem Frontpanel eine Farbrampe hinzufügen. Drehpulinstrumente verfügen standardmäßig über eine Farbrampe.

Tasten, Schalter und Leuchten

Verwenden Sie die auf den Paletten **Elemente»Boolesch** und **Elemente»Elemente klass. Format»Boolesch** befindlichen booleschen Bedien- und Anzeigeelemente, um Tasten, Schalter und Leuchten nachzubilden. Mit booleschen Bedien- und Anzeigeelementen werden boolesche Werte (TRUE/FALSE) eingegeben beziehungsweise angezeigt. Wenn Sie beispielsweise die Temperatur eines Experiments überwachen, können Sie eine boolesche Warnleuchte auf dem Frontpanel platzieren, um anzuzeigen, wann die Temperatur ein bestimmtes Niveau übersteigt.

Verwenden Sie das Kontextmenü, um das Erscheinungsbild des booleschen Objekts anzupassen und um festzulegen, wie sich das Objekt verhält, wenn darauf geklickt wird.

Texteingabefelder, Beschriftungen und Pfadanzeigen

Verwenden Sie die in den Paletten **Elemente»String & Pfad** und **Elemente»Elemente klass. Format»String & Pfad** befindlichen String- und Pfadbedienelemente, um Texteingabefelder und Beschriftungen nachzubilden und um die Position einer Datei oder eines Verzeichnisses einzugeben oder anzuzeigen.

String-Bedien- und Anzeigeelemente

Mit Hilfe des Bedienwerkzeugs oder des Beschriftungswerkzeugs geben Sie Text in ein String-Bedienelement auf dem Frontpanel ein oder ändern den vorhandenen Text. Standardmäßig wird neuer oder geänderter Text erst dann an das Blockdiagramm übergeben, wenn Sie die Bearbeitungssitzung beenden. Sie beenden die Bearbeitungssitzung, indem Sie an

beliebiger Stelle auf das Frontpanel klicken, indem Sie zu einem anderen Fenster wechseln, indem Sie auf die Schaltfläche **Eingabe** auf der Symbolleiste klicken oder indem Sie die <Eingabe>-Taste auf dem numerischen Eingabeblock drücken. Durch Drücken der <Eingabe>-Taste der Tastatur wird ein Wagenrücklauf (Carriage Return, Absatzschaltung) erzeugt.

(Macintosh und Sun) Durch Drücken der <Return>-Taste der Tastatur wird ein Wagenrücklauf (Carriage Return, Absatzschaltung) erzeugt.

Weitere Informationen zu den String-Bedien- und Anzeigeelementen finden Sie in Abschnitt *Strings auf dem Frontpanel* des Kapitels 9, *Gruppieren von Daten mit Hilfe von Strings, Arrays und Clustern*.

Dateipfad-Bedien- und Anzeigeelemente

Verwenden Sie Dateipfad-Bedien- und Anzeigeelemente, um die Position einer Datei oder eines Verzeichnisses einzugeben oder anzuzeigen. Dateipfad-Bedien- und Anzeigeelemente funktionieren ähnlich wie die String-Bedien- und Anzeigeelemente, allerdings formatiert LabVIEW den Dateipfad unter Verwendung der Standardsyntax der von Ihnen verwendeten Plattform.

Ungültige Pfade

Wenn eine Funktion, die einen Pfad zurückgibt, fehlschlägt, gibt die Funktion im Anzeigeelement einen ungültigen Wert für den Pfad, nämlich die Angabe `Kein Pfad`, zurück. Verwenden Sie den Wert `Kein Pfad` als Standardwert für ein Dateipfad-Bedienelement, damit Sie feststellen können, wann der Benutzer fälschlicherweise keinen Pfad angibt, und lassen Sie ein Dateidialogfeld mit Optionen für das Auswählen eines Pfades anzeigen. Verwenden Sie die Funktion `Dateidialog`, um ein Dateidialogfeld anzuzeigen.

Leere Pfade

Ein leerer Pfad in einem Dateipfad-Bedienelement wird unter Windows und Macintosh als leerer String und unter UNIX als Schrägstrich (/) angezeigt. Verwenden Sie leere Pfade, um den Benutzer zur Eingabe eines Pfades aufzufordern. Wenn Sie einen leeren Pfad mit einer Datei-Ein-/Ausgabe- (I/O-) Funktion verbinden, bezieht sich der leere Pfad auf die Liste der dem Computer zugeordneten Laufwerke.

(Macintosh) Der leere Pfad bezieht sich auf gemountete Datenträger.

(UNIX) Der leere Pfad bezieht sich auf das Stammverzeichnis.

Array- und Cluster-Bedien- und Anzeigeelemente

Verwenden Sie die auf den Paletten **Elemente»Array & Cluster** und **Elemente»Elemente klass. Format»Array & Cluster** befindlichen Bedien- und Anzeigeelemente, um Arrays und Cluster aus anderen Bedien- und Anzeigeelementen zu erstellen. Weitere Informationen zu Arrays und Clustern finden Sie in Abschnitt *Gruppieren von Daten mit Arrays und Clustern* des Kapitels 9, *Gruppieren von Daten mit Hilfe von Strings, Arrays und Clustern*.

Die Paletten **Array & Cluster** enthalten auch standardmäßige Fehlereingangs- und Fehlerausgangs-Cluster sowie Bedien- und Anzeigeelemente in Form von Register. Weitere Informationen zu Fehler-Clustern finden Sie in Abschnitt *Fehler-Cluster* des Kapitels 6, *Ausführen von und Fehlersuche in VIs*.

Register-Bedien- und Anzeigeelemente

Verwenden Sie die Register-Bedienelemente, um Frontpanel-Bedien- und Anzeigeelemente überlappend in einem kleineren Bereich anzuordnen. Ein Register-Bedienelement besteht aus Seiten und Registern. Setzen Sie die Frontpanel-Objekte auf die jeweiligen Seiten eines Register-Bedienelements, und verwenden Sie die Register zum Auswählen der verschiedenen Seiten. Sie können eine unbegrenzte Anzahl Frontpanel-Objekte auf einem Register-Bedienelement platzieren.

Register-Bedienelemente sind nützlich, wenn Sie über mehrere Frontpanel-Objekte verfügen, die zusammen oder jeweils während einer bestimmten Phase des Betriebs verwendet werden. So könnten Sie beispielsweise mit einem VI arbeiten, bei dem der Benutzer zunächst eine Reihe von Einstellungen konfigurieren muss, bevor ein Testlauf erfolgen kann, und das es dem Benutzer ermöglicht, bestimmte Aspekte im Verlauf des Tests zu ändern. Schließlich hat der Benutzer dann die Möglichkeit, nur die relevanten Daten anzuzeigen und zu speichern.

Im Blockdiagramm wird das Register-Bedienelement standardmäßig als Bedienelement vom Typ Enum dargestellt. Anschlüsse für Bedien- und Anzeigeelemente, die mit dem Register-Bedienelement verbunden werden, werden im Blockdiagramm wie jeder andere Anschluss dargestellt. Weitere Informationen zu Bedienelementen vom Typ Enum finden Sie in Abschnitt *Bedienelemente vom Enum-Typ* dieses Kapitels.

Listenfelder

Verwenden Sie die Listenfeld-Bedienelemente, die sich auf den Paletten **Elemente»Liste & Tabelle** und **Elemente»Elemente klass. Format»Liste & Tabelle** befinden, um Benutzern eine Liste mit Elementen bereitzustellen, aus der sie wählen können.

Verwenden Sie den Eigenschaftsknoten des Listenfeld-Bedienelements, um Listeneinträge zu ändern und um Informationen über Listeneinträge zu sammeln, wie in den folgenden Schritten erklärt:

- Legen Sie Eintrags-Strings fest.
- Fügen Sie neben dem Listeneintrag ein Symbol hinzu, ähnlich wie im Dialogfeld **Speichern**, in dem Verzeichnisse und Dateien durch unterschiedliche Symbole gekennzeichnet werden.
- Deaktivieren Sie einzelne Einträge in der Liste.
- Fügen Sie Trennlinien zwischen Listeneinträgen ein.
- Ermitteln Sie die aktuell ausgewählten Einträge, indem Sie den Wert des Bedienelements auslesen.
- Ermitteln Sie gegebenenfalls, auf welche Einträge der Benutzer doppelgeklickt hat.

Weitere Informationen zu Eigenschaftsknoten finden Sie in Abschnitt *Eigenschaftsknoten* des Kapitels 16, *Programmatische Steuerung von Vis*.

Listenfelder unterstützen die automatische Vervollständigung, was bedeutet, dass Sie die ersten Zeichen eingeben und LabVIEW den entsprechenden Eintrag im Listenfeld findet. Drücken Sie die <Tabulator>-Taste, um zum nächsten passenden Eintrag zu wechseln. Drücken Sie <Umschalt-Tabulator>, um zum vorherigen passenden Eintrag zu wechseln.

Listenfelder verfügen automatisch über Bildlaufleisten, die jedoch nur aktiviert werden, wenn das Listenfeld mehr Einträge aufweist als angezeigt werden können.

Ring- und Enum-Bedien- und Anzeigeelemente

Verwenden Sie die Ring- und Enum-Bedien- und Anzeigeelemente, die sich in den Paletten **Elemente»Ring & Enum** und **Elemente»Elemente klass. Format»Ring & Enum** befinden, um String-Listen zu erstellen, in denen Sie zyklisch navigieren können, was bedeutet, dass nach dem letzten Eintrag wieder der erste Eintrag angezeigt wird.

Ring-Bedienelemente

Ring-Bedienelemente sind numerische Objekte, bei denen numerische Werte mit Strings oder Grafiken verbunden werden. Ring-Bedienelemente werden als Pulldown-Menüs angezeigt, in denen der Benutzer navigieren kann, um seine Auswahl zu treffen.

Ring-Bedienelemente eignen sich für die Auswahl von sich gegenseitig ausschließenden Elementen wie Triggermodi. Beispielsweise können Sie für die Benutzer ein Ring-Bedienelement bereitstellen, in dem Sie zwischen kontinuierlichen, einzelnen und externen Triggern auswählen können.

Die Reihenfolge der Einträge im Ring-Bedienelement ist abhängig von der Reihenfolge, in der die Einträge eingegeben werden. Jeder Eintrag verfügt über einen numerischen Wert im Bereich Null bis $n-1$, wobei n für die Anzahl der Einträge steht. Im Ring-Bedienelement wird der letzte Eintrag angezeigt, wenn der Benutzer einen beliebigen Wert eingibt, der größer oder gleich $n-1$ ist, und der erste Eintrag, wenn der Benutzer einen Wert eingibt, der kleiner oder gleich Null ist.

Ring-Bedienelemente unterstützen die automatische Vervollständigung der Eingabe, und die Pulldown-Menüs von Ring-Bedienelementen können über Bildlaufleisten verfügen.

Bedienelemente vom Enum-Typ

Verwenden Sie Bedienelemente vom Enum-Typ, um für die Benutzer eine Liste mit Aktionen bereitzustellen, aus der Sie auswählen können. Ein Bedienelement vom Typ Auflistung oder Enum gleicht einem Ring-Bedienelement mit Strings. Bei einem Bedienelement vom Enum-Typ ist der Wert jedoch ein String und keine Zahl wie beim Ring-Bedienelement. Beispielsweise können Sie ein Bedienelement vom Enum-Typ verwenden, um die Fälle einer Case-Struktur auszuwählen. Weitere Informationen zu Case-Strukturen finden Sie in Abschnitt [Case-Strukturen](#) des Kapitels 8, [Schleifen und Case-Strukturen](#).

Der Datentyp eines Bedienelements vom Enum-Typ ist vorzeichenloses Byte, vorzeichenloses Wort oder vorzeichenloser Long. Klicken Sie mit der rechten Maustaste auf das Bedienelement vom Enum-Typ, und wählen Sie aus dem Kontextmenü **Darstellung**, um den Datentyp des Bedienelements zu ändern.

Erweiterte Bedien- und Anzeigeelemente vom Typ Enum

Alle arithmetischen Funktionen außer „Inkrement“ und „Dekrement“ behandeln das Bedienelement vom Enum-Typ genauso wie einen vorzeichenlosen numerischen Wert. „Inkrement“ erhöht den letzten aufgelisteten Wert auf den ersten, und „Dekrement“ verringert den ersten aufgelisteten Wert auf den letzten aufgelisteten Wert. Wenn eine mit einem Vorzeichen versehene Ganzzahl zwangsweise in einen Enum-Typ umgewandelt wird, werden negative Zahlen gleich dem ersten Enum-Wert und außerhalb des Bereichs befindliche positive Zahlen gleich dem letzten Enum-Wert gesetzt. Außerhalb des Bereichs befindliche vorzeichenlose Ganzzahlen werden immer dem letzten Enum-Wert gleichgesetzt.

Wenn Sie einen Fließkommawert an ein Anzeigeelement vom Enum-Typ übergeben, wird die Zahl in das Auflistungselement aus den Werten des Enum-Anzeigeelements umgewandelt, das der Zahl am nächsten ist. LabVIEW behandelt außerhalb des Bereichs befindliche Zahlen wie bereits beschrieben. Wenn Sie ein Bedienelement vom Enum-Typ mit einem beliebigen numerischen Wert verbinden, ist der Wert der Enum-Index. Wenn Sie ein Bedienelement vom Enum-Typ mit einem Anzeigeelement vom Enum-Typ verbinden möchten, müssen die Einträge der Auflistung übereinstimmen. Das Anzeigeelement kann jedoch über Einträge verfügen, die über die des Bedienelements hinausgehen.

I/O-Namen-Bedien- und Anzeigeelemente

Verwenden Sie die I/O-Namen-Bedien- und Anzeigeelemente, die sich in den Paletten **Elemente»I/O** oder **Elemente»Elemente klass. Format»I/O** befinden, um DAQ-Kanalnamen, VISA-Ressourcennamen und logische IVI-Namen zu übergeben, die Sie für I/O-VIs konfigurieren, damit diese mit einem Instrument oder einem DAQ-Gerät kommunizieren können.

Die I/O-Namenskonstanten befinden sich auf den Paletten **Funktionen»Instrumenten-I/O** und **Funktionen»Datenerfassung**.

(Windows) Verwenden Sie den im Menü **Werkzeuge** befindlichen Measurement & Automation Explorer, um DAQ-Kanalnamen, VISA-Ressourcennamen und logische IVI-Namen zu konfigurieren.

(Macintosh) Verwenden Sie das NI-DAQ-Konfigurationsprogramm aus dem Menü **Werkzeuge**, um die DAQ-Hardware von National Instruments zu konfigurieren. Mit Hilfe des DAQ-Kanal-Assistenten aus dem Menü **Werkzeuge** können Sie DAQ-Kanalnamen konfigurieren.

(Macintosh und UNIX) Verwenden Sie die Konfigurationsprogramme Ihres Instruments, um VISA-Ressourcennamen und logische IVI-Namen zu konfigurieren. Weitere Informationen zu den Konfigurationsprogrammen finden Sie in der Dokumentation zu dem jeweiligen Instrument.

Referenzen auf Objekte oder Applikationen

Verwenden Sie die Referenznummer-Bedien- und Anzeigeelemente, die sich in den Paletten **Elemente»RefNum** und **Elemente»Elemente klass. Format»RefNum** befinden, um mit Dateien, Verzeichnissen, Geräten und Netzwerkverbindungen zu arbeiten.

Eine Referenznummer oder RefNum ist ein eindeutiger Bezeichner für ein Objekt wie beispielsweise eine Datei, ein Gerät oder eine Netzwerkverbindung. Wenn Sie eine Datei, ein Gerät oder eine Netzwerkverbindung öffnen, erstellt LabVIEW eine RefNum, die mit dieser Datei, diesem Gerät oder dieser Netzwerkverbindung verknüpft ist. Bei allen Operationen, die Sie mit geöffneten Dateien, Geräten oder Netzwerkverbindungen ausführen, werden die RefNums verwendet, um das jeweilige Objekt zu identifizieren. Sie verwenden ein RefNum-Bedien- oder -Anzeigeelement, um eine RefNum an ein VI zu übergeben oder von diesem übergeben zu lassen. Beispielsweise können Sie ein RefNum-Bedien- oder Anzeigeelement verwenden, um den Inhalt der Datei zu ändern, auf welche die RefNum verweist, ohne die Datei schließen und erneut öffnen zu müssen.

Da eine RefNum ein temporärer Verweis auf ein geöffnetes Objekt ist, gilt sie nur für den Zeitraum, in dem das Objekt geöffnet ist. Wenn Sie das Objekt schließen, hebt LabVIEW die Verknüpfung zwischen RefNum und dem Objekt auf, und die RefNum verwaist. Wenn Sie das Objekt erneut öffnen, erstellt LabVIEW eine neue RefNum, die jedoch nicht der ersten RefNum entspricht.

LabVIEW speichert die mit jeder RefNum verbundenen Informationen, wie beispielsweise die aktuelle Position für das Lesen von oder das Schreiben an das Objekt sowie den Grad des Benutzerzugriffs, sodass Sie zwar gleichlaufende, jedoch unabhängige Operationen an einem einzigen Objekt durchführen können. Wenn ein VI ein Objekt mehrfach öffnet, gibt jede geöffnete Operation eine andere RefNum zurück.

Dialogelemente

Verwenden Sie die auf der Palette **Elemente»Dialogelemente** befindlichen Dialogelemente in den Dialogfeldern, die Sie erstellen. Die Dialogelemente und -anzeigen sind speziell für die Verwendung in Dialogfeldern ausgelegt und umfassen Ring-Bedienelemente, Schaltflächen, Registerdialogfelder, Kontrollkästchen und Optionsfelder. Diese Bedienelemente unterscheiden sich von denjenigen auf dem Frontpanel lediglich im Hinblick auf die Darstellung. Diese Bedienelemente werden nämlich in den Farben dargestellt, die Sie für den Desktop definiert haben.

Da sich das Erscheinungsbild der Dialogelemente abhängig von der Plattform ändert, auf der Sie das VI ausführen, ist die Darstellungsweise von Bedienelementen in von Ihnen erstellten VIs mit allen LabVIEW-Plattformen kompatibel. Wenn Sie ein VI auf einer anderen Plattform ausführen, passen sich die Dialogelemente in Farbe und Erscheinungsbild an die standardmäßigen Dialogfeld-Steuerelemente dieser Plattform an.

Wählen Sie **Datei»VI-Einstellungen**, und wählen Sie dann **Fenstererscheinungsbild** aus dem Pulldown-Menü **Kategorie**, um die Menüleiste und die Bildlaufleisten auszublenden und um VIs zu erstellen, die wie die standardmäßigen Dialogfelder der jeweiligen Plattform aussehen und sich ebenso verhalten. Weitere Informationen zum Konfigurieren des Aussehens und des Verhaltens von VIs finden Sie in Abschnitt [Konfigurieren von Erscheinungsbild und Verhalten von VIs](#) des Kapitels 15, [Anpassen von VIs](#).

Beschriftungen

Verwenden Sie Beschriftungen, um Objekte auf dem Frontpanel und im Blockdiagramm zu kennzeichnen.

LabVIEW umfasst zwei Arten von Beschriftungen, und zwar mit Objekten verknüpfte Beschriftungen und freie Beschriftungen. Verknüpfte Beschriftungen gehören zu einem speziellen Objekt, werden mit diesem verschoben und kennzeichnen nur dieses Objekt. Sie können eine verknüpfte Beschriftung zwar unabhängig verschieben, wenn Sie jedoch das Objekt verschieben, mit dem die Beschriftung verknüpft ist, wird die Beschriftung mit dem Objekt verschoben. Sie können verknüpfte Beschriftungen ausblenden, Sie können sie jedoch nicht unabhängig vom zugehörigen Objekt kopieren oder löschen. Sie können auch eine Einheitenbeschriftung für numerische Bedienelemente anzeigen, indem Sie aus dem Kontextmenü **Sichtbare Objekte»Einheitenbeschriftung** auswählen. Weitere Informationen zu numerischen Einheiten finden Sie in

Abschnitt *Numerische Einheiten und strikte Typenprüfung* des Kapitels 5, *Erstellen des Blockdiagramms*.

Freie Beschriftungen sind nicht mit irgendeinem Objekt verbunden, und Sie können sie unabhängig erstellen, verschieben, drehen oder löschen. Verwenden Sie freie Beschriftungen, um Anmerkungen in Frontpanels und Blockdiagrammen zu erstellen.

Zum Erstellen von freien Beschriftungen oder zum Bearbeiten von jeglicher Art von Beschriftung verwenden Sie das Beschriftungswerkzeug.

Untertitel

Frontpanel-Objekte können auch über Untertitel verfügen. Klicken Sie mit der rechten Maustaste auf das Objekt, und wählen Sie aus dem Kontextmenü **Sichtbare Objekte»Untertitel**, um einen Untertitel anzuzeigen. Im Gegensatz zu einer Beschriftung hat ein Untertitel keinen Einfluss auf den Namen des Objekts, und Sie können einen Untertitel für eher beschreibende Zwecke verwenden. Der Untertitel erscheint nicht im Blockdiagramm.

Texteigenschaften

LabVIEW verwendet die Schriftarten, die bereits auf dem Computer installiert sind. Die Attribute von Text ändern Sie über das Pulldown-Menü **Texteinstellungen** der Symbolleiste. Wenn Sie Objekte oder Text auswählen, bevor Sie eine Auswahl aus dem Pulldown-Menü **Texteinstellungen** treffen, wirken sich die Änderungen auf alle ausgewählten Objekte aus. Wenn keine Objekte ausgewählt sind, bezieht sich die Änderung auf die Standardschriftart. Mit dem Ändern der Standardschriftart ändern Sie nicht die Schriftart von bestehenden Beschriftungen. Diese Änderung wirkt sich nur auf Beschriftungen aus, die ab diesem Punkt erstellt werden.

Wählen Sie **Schriftsatzdialog** aus dem Pulldown-Menü **Texteinstellungen** auf dem Frontpanel, um markiertem Text bestimmte Schriftstile zuzuweisen. Wenn Sie keinen Text markiert haben, ist die Option **Standard für Panel** aktiviert. Wenn Sie **Texteinstellungen»Schriftsatzdialog** vom Blockdiagramm aus wählen, ohne dass irgendwelche Objekte ausgewählt sind, ist die Option **Standard für Diagramm** aktiviert. Sie können für das Frontpanel und das Blockdiagramm verschiedene Schriftarten festlegen. Beispielsweise können Sie im Blockdiagramm eine kleine und auf dem Frontpanel eine große Schriftart verwenden.

Das Pulldown-Menü **Texteinstellungen** enthält die folgenden integrierten Schriftarten:

- **Applikationsschriftart:** Standardschriftart, die für die Paletten **Elemente** und **Funktionen** und für Text in neuen Bedienelementen verwendet wird
- **Systemschriftart:** Wird in Menüs verwendet
- **Dialogschriftart:** Wird für den Text in Dialogfeldern verwendet

Wenn Sie ein VI, das eine dieser integrierten Schriftarten enthält, auf eine andere Plattform übertragen, werden Schriftarten verwendet, die der ursprünglichen so ähnlich wie möglich sind.

Das Pulldown-Menü **Texteinstellungen** verfügt auch über die Untermenüelemente **Anpassen**, **Stil**, **Ausrichten** und **Farbe**.

Die Auswahl, die Sie in irgendeinem dieser Untermenüs treffen, gilt für die jeweils ausgewählten Objekte. Wenn Sie beispielsweise eine neue Schriftart wählen, während ein Drehknopf oder ein Graph ausgewählt ist, werden alle Beschriftungen, Skalen und die numerischen Anzeigen mit der neuen Schriftart formatiert.

LabVIEW behält so viele Schriftattribute wie möglich bei, wenn Sie eine Änderung vornehmen. Wenn Sie beispielsweise für verschiedene Objekte die Schriftart Courier wählen, behalten die Objekte, wenn möglich, die Schriftgröße und die verwendeten Schriftstile bei. Wenn Sie das Dialogfeld **Texteinstellungen** verwenden, ändert LabVIEW für die ausgewählten Objekte die ausgewählten Texteingenschaften. Wenn Sie eine der integrierten Schriftarten oder die aktuelle Schriftart wählen, ändert LabVIEW die Schriftart der ausgewählten Objekte und verwendet die mit dieser Schriftart verbundene Größe.

Wenn Sie mit Objekten arbeiten, die über mehrere Textbereiche verfügen, wie beispielsweise Schieberegler, betreffen die Schriftänderungen die aktuell ausgewählten Objekte oder den markierten Text. Wenn Sie beispielsweise den gesamten Schieberegler markieren und **Stil>Fett** aus dem Pulldown-Menü **Texteinstellungen** wählen, wird die Schrift der Skala, der numerischen Anzeige und der Beschriftung fett gesetzt. Wenn Sie nur die Beschriftung markieren und **Fett** wählen, wird nur der Text der Beschriftung fett gesetzt. Wenn Sie den Text einer Achsenmarkierung markieren und **Fett** wählen, wird der Text aller Markierungen fett formatiert.

Gestalten von Benutzeroberflächen

Wenn ein VI als Benutzeroberfläche oder als Dialogfeld dient, sind Erscheinungsbild und Layout des Frontpanels von Bedeutung. Gestalten Sie das Frontpanel so, dass die Benutzer einfach erkennen können, welche Aktionen durchgeführt werden. Sie können Frontpanels erstellen, die ähnlich wie Instrumente oder andere Geräte aussehen.

Verwenden von Frontpanel-Bedien- und Anzeigeelementen

Bedien- und Anzeigeelemente sind die Hauptkomponenten des Frontpanels. Bei der Gestaltung des Frontpanels sollten Sie berücksichtigen, wie die Benutzer mit den VIs arbeiten, und Bedien- und Anzeigeelemente logisch gruppieren. Wenn mehrere Bedienelemente zusammengehören, versehen Sie diese mit einem dekorativen Rahmen, oder fassen Sie sie zu einem Cluster zusammen.

Sorgen Sie für ausreichend Zwischenraum zwischen den Frontpanel-Objekten, um das Frontpanel überschaulich zu gestalten. Leerräume verhindern darüber hinaus, dass der Benutzer versehentlich auf das falsche Bedienelement oder die falsche Schaltfläche klickt.

Weisen Sie Schaltflächen aussagekräftige Namen zu, und verwenden Sie die übliche Terminologie. Verwenden Sie Begriffe wie Start, Stopp und Speichern anstelle von OK. Aussagekräftige Namen erleichtern den Benutzern die Arbeit mit dem VI.

Verwenden Sie die standardmäßigen LabVIEW-Schriftarten und Farben. Beim Portieren auf andere Plattformen ersetzt LabVIEW die integrierten Schriftarten durch vergleichbare Schriftartenfamilien. Wenn Sie eine andere Schriftart verwenden, ersetzt LabVIEW die Schrift durch eine möglichst ähnliche Schrift, wenn diese Schriftart auf dem Computer nicht vorhanden ist. LabVIEW behandelt Farben ähnlich wie Schriftarten. Wenn eine Farbe auf einem Computer nicht verfügbar ist, ersetzt LabVIEW diese durch eine möglichst ähnliche Farbe.

Vermeiden Sie, Objekte auf anderen Objekten zu platzieren. Wenn Sie eine Beschriftung oder ein anderes Objekt so platzieren, dass es Bedien- oder Anzeigeelemente ganz oder teilweise überdeckt, wird die Bildschirmaktualisierung verlangsamt, was dazu führen kann, dass das Bedien- oder Anzeigeelement flimmert.

Gestalten von Dialogfeldern

Wenn ein VI aufeinanderfolgende Dialogfelder enthält, die an der gleichen Bildschirmposition angezeigt werden, ordnen Sie diese so an, dass die Schaltflächen im ersten Dialogfeld nicht direkt über den Schaltflächen im nächsten Dialogfeld liegen. Der Benutzer doppelklickt möglicherweise auf eine Schaltfläche im ersten Dialogfeld und klickt damit unwissentlich auf eine Schaltfläche im darauf folgenden Dialogfeld. Weitere Informationen zu Dialogelementen finden Sie in Abschnitt *Dialogelemente* dieses Kapitels.

Auswählen der Bildschirmgröße

Berücksichtigen Sie beim Entwurf eines VIs, dass das Frontpanel möglicherweise auch auf Computern mit unterschiedlichen Bildschirmauflösungen angezeigt wird. Wählen Sie **Datei»VI-Einstellungen**, und wählen Sie dann **Fenstergröße** aus dem Pulldown-Menü **Kategorie**. Aktivieren Sie das Kontrollkästchen **Fensterproportionen für verschiedene Bildschirmauflösungen beibehalten**, damit die Fensterproportionen des Frontpanels in Relation zur Bildschirmauflösung beibehalten werden.

Erstellen des Blockdiagramms

Nachdem Sie das Frontpanel erstellt haben, können Sie mit Hilfe der grafischen Darstellungen von Funktionen Code hinzufügen, um die Frontpanel-Objekte zu steuern. Dieser grafische Quellcode ist im Blockdiagramm enthalten.

Weitere Informationen ...

Weitere Informationen zum Gestalten und Konfigurieren des Frontpanels finden Sie in der *LabVIEW-Hilfe*.

Beziehung zwischen Frontpanel-Objekten und Blockdiagrammanschlüssen

Frontpanel-Objekte werden im Blockdiagramm als Anschlüsse dargestellt. Doppelklicken Sie auf einen Blockdiagrammanschluss, um das zugehörige Bedien- oder Anzeigeelement im Frontpanel hervorzuheben. Sie können einen Anschluss nicht aus dem Blockdiagramm löschen. Der Anschluss kann nur entfernt werden, wenn Sie das entsprechende Objekt im Frontpanel löschen.

Anschlüsse sind Eingangs- und Ausgangsports, über die Informationen zwischen dem Frontpanel und dem Blockdiagramm ausgetauscht werden. Daten, die Sie in den Frontpanel-Bedienelementen eingeben, werden über die Bedienelementanschlüsse an das Blockdiagramm übergeben. Wenn die Ausführung des VIs beendet ist, werden die Ausgabedaten an die Anzeigeelementanschlüsse übertragen, wo sie das Blockdiagramm verlassen, um erneut an das Frontpanel übergeben und dann mit Hilfe der Frontpanel-Anzeigeelemente angezeigt zu werden.

Blockdiagrammobjekte

Zu den Objekten im Blockdiagramm gehören Anschlüsse, Knoten und Funktionen. Blockdiagramme werden erstellt, indem Objekte mit Hilfe von Verbindungen verknüpft werden.

Blockdiagrammanschlüsse



Die Blockdiagrammanschlüsse repräsentieren den Datentyp des Frontpanel-Bedien- oder Anzeigeelements. Beispielsweise stellt ein DBL-Anschluss wie links dargestellt ein numerisches Bedien- oder Anzeigeelement mit doppelter Genauigkeit und Fließkommafunktion dar.

Ein Anschluss ist ein beliebiger Punkt, mit dem Sie eine Verbindung verknüpfen können, wenn diese nicht mit einer anderen Verbindung verknüpft wird. LabVIEW verfügt über Anschlüsse für Bedien- und Anzeigeelemente, Knotenanschlüsse, Konstanten und spezielle Anschlüsse an Strukturen, wie beispielsweise Eingabe- und Ausgabeanschlüsse am Formelknoten. Zum Herstellen der Verbindungen zu den Anschlüssen und zur Weitergabe von Daten an andere Anschlüsse verwenden Sie Verbindungen. Klicken Sie mit der rechten Maustaste auf ein Blockdiagrammobjekt, und wählen Sie aus dem Kontextmenü **Sichtbare Elemente»Anschlüsse**, um die Anschlüsse anzuzeigen. Klicken Sie mit der rechten Maustaste auf ein Objekt, und wählen Sie erneut **Sichtbare Elemente»Anschlüsse**, um die Anschlüsse wieder auszublenden. Dieses Kontextmenü steht für erweiterbare Funktionen nicht zur Verfügung.

Datentypen für Bedien- und Anzeigeelemente

Tabelle 5-1 enthält die Symbole für die unterschiedlichen Anschlussstypen bei Bedien- und Anzeigeelementen. Die Farbe und das Symbol des jeweiligen Anschlusses repräsentieren den Datentyp des Bedien- oder Anzeigeelements. Bedienelementanschlüsse weisen einen dickeren Rahmen als Anzeigeelementanschlüsse auf.

Tabelle 5-1. Anschlüsse von Bedien- und Anzeigeelementen

Bedienelement	Anzeigeelement	Datentyp	Farbe
		Fließkommazahl mit einfacher Genauigkeit	Orange
		Fließkommazahl mit doppelter Genauigkeit	Orange
		Fließkommazahl mit erweiterter Genauigkeit	Orange

Tabelle 5-1. Anschlüsse von Bedien- und Anzeigeelementen (Fortsetzung)













































Bedien- element	Anzeige- element	Datentyp	Farbe
		Komplexe Fließkommazahl mit einfacher Genauigkeit	Orange
		Komplexe Fließkommazahl mit doppelter Genauigkeit	Orange
		Komplexe Fließkommazahl mit erweiterter Genauigkeit	Orange
		Vorzeichenbehaftete 8-Bit-Ganzzahl	Blau
		Vorzeichenbehaftete 16-Bit-Ganzzahl	Blau
		Vorzeichenbehaftete 32-Bit-Ganzzahl	Blau
		Vorzeichenlose 8-Bit-Ganzzahl	Blau
		Vorzeichenlose 16-Bit-Ganzzahl	Blau
		Vorzeichenlose 32-Bit-Ganzzahl	Blau
		Enum-Typ	Blau
		Boolescher Wert	Grün
		String	Rosa
		Array: Der Datentyp der Elemente wird in eckige Klammern eingeschlossen, und es wird die Farbe dieses Datentyps übernommen.	Unter- schie-dlich
 	 	Cluster: Umfasst mehrere Datentypen. Cluster-Datentypen werden braun dargestellt, wenn die Elemente des Clusters vom gleichen Typ sind, oder rosa, wenn die Elemente des Clusters unterschiedliche Datentypen aufweisen.	Braun oder Rosa
		Pfad	Hellblau
		Signalverlauf	Braun
		Referenznummer (RefNum)	Hellblau
		Variant: Umfasst den Namen des Bedien- oder Anzeigeelements, die Angaben zum Datentyp und die Daten selbst. Weitere Informationen zu Daten vom Typ Variant finden Sie in Abschnitt <i>Verarbeiten von Variant-Daten</i> dieses Kapitels.	Lila

Tabelle 5-1. Anschlüsse von Bedien- und Anzeigeelementen (Fortsetzung)

Bedien- element	Anzeige- element	Datentyp	Farbe
		Polymorph: Gibt an, dass ein VI oder eine Funktion mehr als einen Datentyp akzeptiert. Weitere Informationen zu polymorphen VIs und Funktionen finden Sie in Abschnitt <i>Polymorphe VIs und Funktionen</i> dieses Kapitels.	Lila
		I/O-Name: Übergibt benutzerseitig konfigurierte DAQ-Kanalnamen, VISA-Ressourcennamen und logische IVI-Namen an I/O-VIs, damit diese mit einem Instrument oder einem DAQ-Gerät kommunizieren können. Weitere Informationen zum Datentyp I/O-Name finden Sie in Abschnitt <i>I/O-Namen-Bedien- und Anzeigeelemente</i> des Kapitels 4, <i>Erstellen des Frontpanels</i> .	Lila
		Bild: Dient zum Anzeigen von Bildern, die Linien, Kreise, Text und andere Arten von Grafikformen enthalten. Weitere Informationen zum Datentyp Bild finden Sie in Abschnitt <i>Verwenden des Bildanzeigeelements</i> des Kapitels 12, <i>VIs für Grafiken und Sound</i> .	Blau

Viele Datentypen verfügen über die entsprechenden Funktionen, mit denen die Daten bearbeitet werden können. Weitere Informationen zu den Funktionen, die mit dem jeweiligen Datentyp eingesetzt werden können, finden Sie in Abschnitt *Überblick über Funktionen* dieses Kapitels.

Konstanten

Konstanten sind Anschlüsse im Blockdiagramm, mit denen festgelegte Datenwerte an das Blockdiagramm übergeben werden, wie beispielsweise π (π) und Unendlich (∞). Benutzerdefinierte Konstanten sind Konstanten, die von Ihnen definiert und bearbeitet werden, bevor Sie das VI ausführen.

Sie beschriften eine Konstante, indem Sie mit der rechten Maustaste auf die Konstante klicken und dann aus dem Kontextmenü **Sichtbare Elemente»Beschriftung** wählen. Konstanten weisen vorher festgelegte Werte für Beschriftungen auf, die Sie mit Hilfe des Bedienwerkzeugs oder des Beschriftungswerkzeugs bearbeiten können.

Die meisten Konstanten befinden sich im oberen oder unteren Bereich der jeweiligen Palette.

Konstanten

Konstanten verwenden Sie für mathematische Berechnungen und zum Formatieren von Strings oder Pfaden. LabVIEW umfasst die folgenden Arten von Konstanten:

- **Zusätzliche numerische Konstanten:** Eine Sammlung von häufig verwendeten mathematischen und physikalischen Werten mit hoher Genauigkeit, wie beispielsweise den natürlichen Logarithmus der Basis (e) und die Lichtgeschwindigkeit. Die zusätzlichen numerischen Konstanten befinden sich auf der Palette **Funktionen»Numerisch»Zusätzliche numerische Konstanten**.
- **Stringkonstanten:** Eine Sammlung häufig verwendeter, nicht anzeigbarer Stringzeichen wie „Zeilenvorschub“ (Line Feed, LF) und „Wagenrücklauf“ (Carriage Return, CR). Die Stringkonstanten befinden sich auf der Palette **Funktionen»String**.
- **Dateikonstanten:** Eine Sammlung häufig verwendeter Pfadwerte wie beispielsweise „Kein Pfad“, „Keine RefNum“ und „Standardverzeichnis“. Die Dateikonstanten befinden sich auf der Palette **Funktionen»Datei-I/O»Dateikonstanten**.

Benutzerdefinierte Konstanten

Die Palette **Funktionen** enthält nach Typ geordnete Konstanten wie beispielsweise „Boolesch“, „Zahl“, „Ring“, „Enum-Typ“, „Farbfeld“, „Listenfeld“, „String“, „Array“, „Cluster“ und „Pfad“.

Die einfachste Methode zum Erstellen einer benutzerdefinierten Konstante besteht darin, mit der rechten Maustaste auf den Eingabe- oder Ausgabeanschluss eines VIs oder einer Funktion zu klicken und dann aus dem Kontextmenü **Erstelle Konstante** zu wählen. Wenn Sie mit Hilfe des Menüelements **Erstelle Konstante** eine Konstante erstellen, markiert LabVIEW den Wert der Konstante, so dass Sie den gewünschten Wert eingeben können. Sie können den Wert von benutzerdefinierten Konstanten nicht ändern, während das VI ausgeführt wird.

Sie können eine Konstante auch erstellen, indem Sie ein Bedienelement des Frontpanels auf das Blockdiagramm ziehen. Die hieraus resultierende Konstante enthält den Wert des Frontpanel-Bedienelements zu dem Zeitpunkt, zu dem Sie es auf das Blockdiagramm gezogen haben. Das Frontpanel-Bedienelement verbleibt auf dem Frontpanel. Wird nun der Wert des Bedienelements auf dem Frontpanel geändert, hat dies keinen Einfluss auf die Konstante und umgekehrt.

Zum Festlegen oder Ändern des Wertes einer benutzerdefinierten Konstante klicken Sie mit dem Bedienwerkzeug oder dem Beschriftungswerkzeug auf die Konstante und geben den neuen Wert ein. Um den Wert einer numerischen Konstante zu erhöhen oder zu verringern, klicken Sie mit dem Bedienwerkzeug auf die Konstante und drücken dann die Nach-oben- oder Nach-unten-Tasten der Tastatur.

Die Größe von benutzerdefinierten Konstanten wird automatisch angepasst, während Sie Daten eingeben. Nachdem Sie die Form einer Beschriftungs- oder Stringkonstante geändert oder die Größe angepasst haben, können Sie mit der rechten Maustaste darauf klicken und aus dem Kontextmenü **Größenanpassung an Text** wählen, um die Größe der Konstante oder der Beschriftung dem Inhalt entsprechend anzupassen.

Blockdiagrammknoten

Knoten sind Objekte im Blockdiagramm, die über Eingänge und/oder Ausgänge verfügen und Operationen ausführen, wenn ein VI ausgeführt wird. Knoten entsprechen Anweisungen, Operatoren, Funktionen und Subroutinen in textbasierten Programmiersprachen. LabVIEW umfasst die folgenden Arten von Knoten:

- **Funktionen:** Integrierte Ausführungselemente, die mit einem Operator, einer Funktion oder einer Anweisung vergleichbar sind. Weitere Informationen zu den in LabVIEW verfügbaren Funktionen finden Sie in Abschnitt *Überblick über Funktionen* dieses Kapitels.
- **Sub-VIs:** VIs, die in einem Blockdiagramm oder einem anderen VI verwendet werden, vergleichbar mit Subroutinen. Weitere Informationen zum Einsatz von Sub-VIs im Blockdiagramm finden Sie in Abschnitt *Sub-VIs* des Kapitels 7, *Erstellen von VIs und Sub-VIs*.
- **Strukturen:** Prozesssteuerungselemente wie beispielsweise Sequenzstrukturen, Case-Strukturen, For-Schleifen oder While-Schleifen. Weitere Informationen zum Einsatz von Strukturen finden Sie in Kapitel 8, *Schleifen und Case-Strukturen*.
- **Formelknoten:** In der Größe veränderbare Strukturen, mit denen Gleichungen direkt in ein Blockdiagramm eingegeben werden können. Weitere Informationen zum Einsatz von Formelknoten finden Sie in Abschnitt *Formelknoten* des Kapitels 20, *Formeln und Gleichungen*.
- **Eigenschaftsknoten:** Legen die Eigenschaften einer Klasse fest oder finden diese. Weitere Informationen zum Einsatz von Eigenschaftsknoten finden Sie in Abschnitt *Eigenschaftsknoten* des Kapitels 16, *Programmatische Steuerung von VIs*.

- **Methodenknoten:** Führt die Methoden einer Klasse aus. Weitere Informationen zum Einsatz von Methodenknoten finden Sie in Abschnitt *Methodenknoten* des Kapitels 16, *Programmatische Steuerung von VIs*.
- **Code-Interface-Knoten (CINs):** Ruft Code von textbasierten Programmiersprachen ab. Weitere Informationen zum Aufrufen von Code aus textbasierten Programmiersprachen finden Sie in Abschnitt *Code-Interface-Knoten* des Kapitels 19, *Aufrufen von Code aus textbasierten Programmiersprachen*.



Nachdem Sie ein VI-Frontpanel und -Blockdiagramm erstellt haben, erstellen Sie das Anschlussfeld wie links dargestellt, damit das VI als Sub-VI verwendet werden kann. Das Anschlussfeld ist eine Sammlung von Anschlüssen, die den Bedien- und Anzeigeelementen dieses VIs entsprechen, ähnlich der Parameterliste eines Funktionsaufrufs in textbasierten Programmiersprachen. Mit dem Anschlussfeld werden die Eingänge und Ausgänge definiert, die Sie mit dem VI verbinden möchten, damit Sie es als Sub-VI einsetzen können. Weitere Informationen zum Einrichten von Anschlussfeldern finden Sie in Abschnitt *Einrichten des Anschlussfeldes* des Kapitels 7, *Erstellen von VIs und Sub-VIs*.

Überblick über Funktionen

Funktionen sind die grundlegenden Betriebselemente von LabVIEW. Die Funktionssymbole auf der Palette **Funktionen** sind durch blassgelbe Hintergründe gekennzeichnet. Funktionen verfügen nicht über Frontpanels oder Blockdiagramme, weisen jedoch Anschlussfelder auf.

Die Palette **Funktionen** beinhaltet auch die VIs, die zum Lieferumfang von LabVIEW gehören. Die VI-Symbole auf der Palette **Funktionen** erkennen Sie am weißen Hintergrund. Verwenden Sie diese VIs als Sub-VIs, wenn Sie VIs für die Datensammlung, die Instrumentensteuerung, die Kommunikation oder andere VIs erstellen. Weitere Informationen zum Einsatz der integrierten VIs finden Sie in Abschnitt *Verwenden der integrierten VIs und Funktionen* des Kapitels 7, *Erstellen von VIs und Sub-VIs*.

Numerische Funktionen

Verwenden Sie die numerischen Funktionen, die sich auf der Palette **Funktionen»Numerisch** befinden, um arithmetische, trigonometrische, logarithmische und komplexe mathematische Operationen mit Zahlen zu erstellen, und um Zahlen von einem Datentyp in einen anderen umzuwandeln.

Boolesche Funktionen

Verwenden Sie die booleschen Funktionen, die sich auf der Palette **Funktionen»Boolesch** befinden, um logische Operationen mit einzelnen booleschen Werten oder mit Arrays aus booleschen Werten zu erstellen, wie beispielsweise die Folgenden:

- Ändern eines Wertes von TRUE zu FALSE und umgekehrt.
- Festlegen, welcher boolesche Wert zurückgegeben werden soll, wenn Sie zwei oder mehr boolesche Werte erhalten.
- Umwandeln eines booleschen Wertes in eine Zahl (entweder 1 oder 0).
- Ausführen einer mehrfacharithmetischen Funktion für zwei oder mehr boolesche Werte.

String-Funktionen

Verwenden Sie die String-Funktionen, die sich auf der Palette **Funktionen»String** befinden, für die folgenden Aufgaben:

- Verknüpfen von zwei oder mehr Strings.
- Extrahieren der Teilmenge eines Strings aus einem String.
- Umwandeln von Daten in Strings.
- Formatieren eines Strings zur Verwendung in einer Textverarbeitung oder Tabellenkalkulation.

Weitere Informationen zum Einsatz der String-Funktionen finden Sie in Abschnitt *Strings* des Kapitels 9, *Gruppieren von Daten mit Hilfe von Strings, Arrays und Clustern*.

Array-Funktionen

Verwenden Sie die auf der Palette **Funktionen»Array** befindlichen Array-Funktionen zum Erstellen und Bearbeiten von Arrays, wie mit den folgenden Beispielen verdeutlicht:

- Extrahieren von einzelnen Datenelementen aus einem Array.
- Hinzufügen von einzelnen Datenelementen zu einem Array.
- Teilen von Arrays.

Weitere Informationen zum Einsatz der Array-Funktionen finden Sie in Abschnitt *Arrays* des Kapitels 9, *Gruppieren von Daten mit Hilfe von Strings, Arrays und Clustern*.

Cluster-Funktionen

Verwenden Sie die auf der Palette **Funktionen»Cluster** befindlichen Cluster-Funktionen zum Erstellen und Bearbeiten von Clustern, wie mit den folgenden Beispielen verdeutlicht:

- Extrahieren von einzelnen Datenelementen aus einem Cluster.
- Hinzufügen von einzelnen Datenelementen zu einem Cluster.
- Aufteilen eines Clusters in die einzelnen Datenelemente.

Weitere Informationen zum Einsatz der Cluster-Funktionen finden Sie in Abschnitt *Cluster* des Kapitels 9, *Gruppieren von Daten mit Hilfe von Strings, Arrays und Clustern*.

Vergleichsfunktionen

Mit den Vergleichsfunktionen, die sich auf der Palette **Funktionen»Vergleich** befinden, können Sie boolesche Werte, Strings, Zahlen, Arrays und Cluster vergleichen.

Weitere Informationen zum Einsatz der Vergleichsfunktionen finden Sie in Anhang C, *Vergleichsfunktionen*.

Zeit- und Dialogfunktionen

Verwenden Sie die Zeit- und Dialogfunktionen, die sich auf der Palette **Funktionen»Zeit & Dialog** befinden, für die folgenden Aufgaben:

- Manipulieren der Geschwindigkeit, mit der eine Operation ausgeführt wird.
- Abrufen von Uhrzeit und Datum der systemeigenen Uhr.
- Erstellen von Dialogfeldern, um Benutzern Anweisungen zu erteilen.

Diese Palette enthält auch die Fehlerbehandler-VIs. Weitere Informationen zum Einsatz der Fehlerbehandler-VIs finden Sie in Abschnitt *Fehlerprüfung und Fehlerbehandlung* des Kapitels 6, *Ausführen von und Fehlersuche in VIs*.

Datei-I/O-Funktionen

Verwenden Sie die Datei-I/O-Funktionen, die sich auf der Palette **Funktionen»Datei-I/O** befinden, für die folgenden Aufgaben:

- Öffnen und Schließen von Dateien.
- Lesen aus und Schreiben an Dateien.

- Erstellen der Verzeichnisse und Dateien, die Sie im Pfad-Bedienelement angeben.
- Abrufen von Verzeichnisinformationen.
- Schreiben von Strings, Zahlen, Arrays und Clustern an Dateien.

Die Palette **Datei-I/O** enthält auch VIs zum Ausführen von häufig auftretenden Datei-I/O-Aufgaben. Weitere Informationen zum Einsatz der Datei-I/O-VIs und -Funktionen finden Sie in Kapitel 13, *Datei-I/O*.

Signalverlaufsfunktionen

Verwenden Sie die Signalverlaufsfunktionen, die sich auf der Palette **Funktionen»Signalverlauf** befinden, um Signalverläufe zu erstellen, die Signalverlaufswerte, Kanalinformationen und Informationen über den zeitlichen Verlauf enthalten. Die Palette **Signalverlauf** beinhaltet VIs, mit denen häufig auftretende Aufgaben und mit dem Signalverlauf zusammenhängende Aufgaben ausgeführt werden können. Weitere Informationen zum Erstellen und Verwenden von Signalverläufen in VIs finden Sie in Teil II, *DAQ Basics*, des *LabVIEW Measurement-Handbuchs*.

Applikationsteuerungsfunktionen

Verwenden Sie die Applikationssteuerungsfunktionen, die sich auf der Palette **Funktionen»Applikationssteuerung** befinden, um VIs und LabVIEW-Applikationen auf dem lokalen Computer oder über das Netzwerk programmatisch zu steuern. Weitere Informationen über den Einsatz der Applikationssteuerungsfunktionen finden Sie in Kapitel 16, *Programmatische Steuerung von VIs*.

Fortgeschrittene Funktionen

Verwenden Sie die fortgeschrittenen Funktionen, die sich auf der Palette **Funktionen»Fortgeschritten** befinden, um Code und Funktionen aus Bibliotheken, wie beispielsweise DLLs (Dynamic Link Libraries), abzurufen, um LabVIEW-Daten für den Einsatz in anderen Applikationen zu bearbeiten und um Codefragmente aufzurufen, die mit textbasierten Programmiersprachen erstellt wurden. Weitere Informationen zum Einsatz der fortgeschrittenen Funktionen finden Sie im Handbuch *Using External Code in LabVIEW*.

Hinzufügen von Anschlüssen zu Blockdiagrammfunktionen

Bei einigen Funktionen können Sie die Anzahl der Anschlüsse ändern. Wenn Sie beispielsweise ein Array mit zehn Elementen erstellen möchten, müssen Sie zehn Anschlüsse hinzufügen.

Sie können erweiterbaren Funktionen Anschlüsse hinzufügen, indem Sie mit Hilfe des Positionierwerkzeugs die Ecke der Funktion ziehen. Mit dem Positionierwerkzeug können Sie auch Anschlüsse von erweiterbaren Funktionen entfernen, wobei Sie allerdings keinen Anschluss entfernen können, der bereits verbunden ist. In diesem Fall müssen Sie zuerst die vorhandene Verbindung löschen, bevor Sie den Anschluss entfernen können.

Sie können Anschlüsse auch hinzufügen oder entfernen, indem Sie mit der rechten Maustaste auf das Symbol eines Anschlusses klicken und dann aus dem Kontextmenü **Eingang hinzufügen**, **Ausgang hinzufügen**, **Eingang entfernen** oder **Ausgang entfernen** wählen. Abhängig von der Funktion können Sie Anschlüsse für Eingänge, Ausgänge oder RefNum-Bedienelemente hinzufügen. Mit den Menüelementen **Eingang hinzufügen** und **Ausgang hinzufügen** aus dem Kontextmenü wird ein Anschluss unmittelbar an den Anschluss angefügt, auf den Sie mit der rechten Maustaste geklickt haben. Mit den Menüelementen **Eingang entfernen** und **Ausgang entfernen** wird der Anschluss entfernt, auf den Sie mit der rechten Maustaste geklickt haben. Wenn Sie die Elemente des Kontextmenüs zum Entfernen eines bereits verbundenen Anschlusses verwenden, entfernt LabVIEW den Anschluss und trennt die Verbindung.

Verbindung von Blockdiagrammobjekten

Sie übertragen die Daten zwischen den Blockdiagrammobjekten über Verbindungen. Jede Verbindung verfügt über eine einzige Datenquelle, jedoch können Sie diese mit vielen VIs und Funktionen verbinden, mit denen Daten gelesen werden. Verbindungen weisen unterschiedliche Farben, Stile und Stärken auf, und zwar abhängig vom jeweiligen Datentyp. Eine unterbrochene Verbindung wird als eine gestrichelte schwarze Linie dargestellt. Weitere Informationen zu den Datentypen von Verbindungen finden Sie auf der *LabVIEW-Referenzkarte*.

Verbindungsstümpfe sind die abgeschnittenen Leitungen, die um ein VI- oder Funktionssymbol angezeigt werden, wenn Sie das Verbindungswerkzeug über das Symbol bewegen. Sie zeigen den Datentyp des jeweiligen Anschlusses an. Darüber hinaus erscheint auch ein Hinweistreifen mit dem Namen des Anschlusses. Ein Punkt am Ende eines

Verbindungsstumpfes zeigt an, dass es sich um einen Eingang handelt. Verbindungsstümpfe von Ausgängen verfügen nicht über einen Punkt. Wenn Sie einen Anschluss ordnungsgemäß verbinden, verschwindet der Verbindungsstumpf für diesen Anschluss.

Ein Verbindungssegment ist ein einziges, horizontal oder vertikal verlegtes Verbindungsstück. An der Stelle, an der zwei Segmente verbunden werden, wird ein Bogen in der Verbindung angezeigt. Der Punkt, an dem drei oder vier Verbindungssegmente zusammenlaufen, wird als Knotenpunkt bezeichnet. Ein Verbindungssegment enthält alle Verbindungssegmente von Knotenpunkt zu Knotenpunkt, von Anschluss zu Knotenpunkt oder von Anschluss zu Anschluss, zwischen denen sich keine Knotenpunkte befinden. Abbildung 5-1 zeigt ein Verbindungssegment, einen Bogen und einen Knotenpunkt.

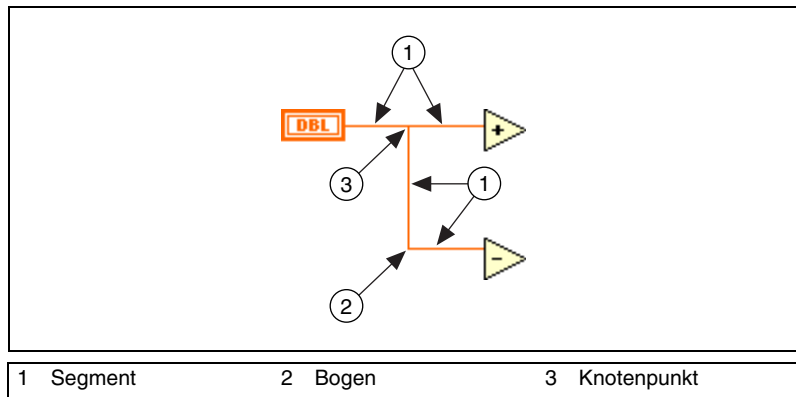


Abbildung 5-1. Verbindungssegment, Bogen und Knotenpunkt

Sie können Anschlüsse vertikal oder horizontal verbinden, und zwar abhängig von der Richtung, in der Sie das Verbindungswerkzeug zuerst bewegen. Die Verbindung wird zum Mittelpunkt des Anschlusses hergestellt, ungeachtet, an welcher Stelle des Anschlusses Sie klicken. Nachdem Sie auf den Anschluss geklickt haben, können Sie zwischen der horizontalen und vertikalen Verbindungsrichtung wechseln, indem Sie die Leertaste drücken.

Beim Verbinden eines Anschlusses fügen Sie einmal einen 90°-Bogen ein, indem Sie den Cursor entweder vertikal oder horizontal bewegen. Um die Verbindung mit Bögen in verschiedene Richtungen auszustatten, klicken Sie mit der Maustaste, um die Verbindung zu erstellen, und ziehen Sie diese dann in die neue Richtung. Sie können die Verbindung wiederholt erstellen und dann in neue Richtungen ziehen.

Um den letzten Punkt rückgängig zu machen, an dem Sie die Verbindung erstellt haben, drücken Sie die <Strg>-Taste und klicken.

(Macintosh) Drücken Sie die <Option>-Taste, und klicken.

(UNIX und Linux) Drücken Sie die mittlere Maustaste, und klicken Sie.

Wenn Sie Verbindungen kreuzweise übereinander legen, wird in der ersten von Ihnen gezogenen Verbindung ein kleiner Spalt angezeigt, um zu zeigen, dass die erste Verbindung unter der zweiten liegt.



Vorsicht! Sich kreuzende Verbindungen können ein Blockdiagramm unübersichtlich machen und die Fehlersuche erschweren.

Automatisches Verbinden von Objekten

Wenn Sie die Funktion „Automatische Verbindung“ aktiviert haben, verbindet LabVIEW Objekte automatisch, wenn Sie diese auf dem Blockdiagramm platzieren. Sie können auch Objekte automatisch verbinden, die sich bereits auf dem Blockdiagramm befinden. LabVIEW verbindet die Anschlüsse, die am besten zusammenpassen, und lässt Anschlüsse, die nicht zusammenpassen, unverbunden.

Wenn Sie ein markiertes Objekt nahe an andere Objekte im Blockdiagramm verschieben, zieht LabVIEW temporäre Verbindungen, um gültige Verbindungen anzuzeigen. Wenn Sie die Maustaste loslassen, um das Objekt im Blockdiagramm zu platzieren, stellt LabVIEW automatisch die Verbindungen her.

Sie deaktivieren die Funktion „Automatische Verbindung“, indem Sie die Leertaste drücken, während Sie ein Objekt mit Hilfe des Positionierwerkzeugs bewegen.

Die Funktion „Automatische Verbindung“ ist standardmäßig aktiviert, wenn Sie ein Objekt aus der Palette **Funktionen** wählen oder wenn Sie ein Objekt kopieren, das sich bereits im Blockdiagramm befindet, indem Sie die <Strg>-Taste drücken und das Objekt ziehen. Die Funktion „Automatische Verbindung“ ist standardmäßig deaktiviert, wenn Sie das Positionierwerkzeug verwenden, um ein bereits im Blockdiagramm befindliches Objekt zu verschieben.

(Macintosh) Drücken Sie die <Option>-Taste. **(Sun)** Drücken Sie die <Meta>-Taste. **(HP-UX und Linux)** Drücken Sie die <Alt>-Taste.

Manuelles Verbinden von Objekten

Verwenden Sie das Verbindungswerkzeug, um Anschlüsse an einem Blockdiagrammknoten mit den Anschlüssen an einem anderen Blockdiagrammknoten zu verbinden. Die Cursorspitze des Werkzeugs ist die Spitze der abgewickelten Leitungsspule. Wenn Sie das Verbindungswerkzeug über einen Knoten bewegen, blinkt der Anschluss, und auf VIs und Funktionen erscheint der Name des Anschlusses in einem Hinweisstreifen.

Wählen Sie **Hilfe»Zeige Kontext-Hilfe**, um das Fenster **Kontext-Hilfe** anzuzeigen, in dem jeder Anschluss des VIs oder der Funktion aufgeführt ist und dem Sie exakt entnehmen können, welche Verbindungen hergestellt werden müssen. Das Fenster **Kontext-Hilfe** zeigt nicht die Anschlüsse für erweiterbare Funktionen wie beispielsweise die Funktion „Array erstellen“ an.

Markieren von Verbindungen

Sie markieren Verbindungen, indem Sie mit dem Positionierwerkzeug einmal, zweimal oder dreimal darauf klicken. Mit einem einfachen Klick auf eine Verbindung wird ein Segment der Verbindung markiert. Mit dem Doppelklick wird ein Verbindungsweig markiert. Wenn Sie dreimal auf eine Verbindung klicken, wird die gesamte Verbindung markiert.

Entfernen von unterbrochenen Verbindungen

Eine unterbrochene Verbindung wird als eine gestrichelte schwarze Linie dargestellt. Für unterbrochene Verbindungen gibt es eine Vielzahl von Gründen, beispielsweise, wenn Sie versuchen, zwei Objekte mit inkompatiblen Datentypen miteinander zu verbinden. Bewegen Sie das Verbindungswerkzeug über eine unterbrochene Verbindung, um den Hinweisstreifen anzuzeigen, mit dem erläutert wird, warum die Verbindung unterbrochen ist. Klicken Sie dreimal mit dem Positionierwerkzeug auf die Verbindung, und drücken Sie die <Entf>-Taste, um eine unterbrochene Verbindung zu entfernen.



Hinweis Achten Sie darauf, dass Sie keine schwarze, gestrichelte Verbindung mit einer punktierten grünen Verbindung verwechseln. Eine punktierte grüne Verbindung stellt einen booleschen Datentyp dar.

Sie können alle unterbrochenen Verbindungen entfernen, indem Sie **Bearbeiten»Ungültige Verbindungen entfernen** wählen.



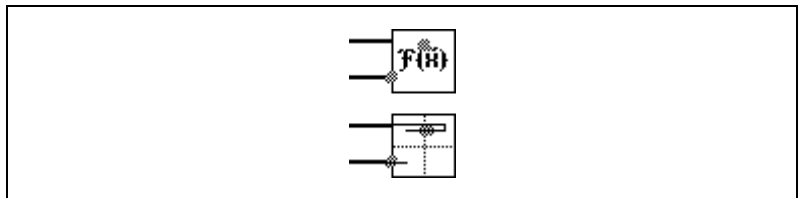
Vorsicht! Seien Sie beim Entfernen aller ungültigen Verbindungen vorsichtig. Manchmal scheint eine Verbindung ungültig zu sein, weil das Blockdiagramm noch nicht fertig ist.

Typumwandlungspunkte

Typumwandlungspunkte werden an Blockdiagrammknoten angezeigt, um Sie darauf hinzuweisen, dass Objekte mit unterschiedlichen Datentypen miteinander verbunden wurden. Der Punkt bedeutet, dass LabVIEW den Wert, der an den Knoten übergeben wurde, in eine andere Darstellungsform umgewandelt hat. Wenn Sie beispielsweise ein Bedienelement mit der Darstellungsweise Fließkommawert mit doppelter Genauigkeit und dem Wert 3,02 mit einem Anzeigeelement mit einer ganzzahligen Darstellungsweise verbinden, wird am Anzeigeelement ein Typumwandlungspunkt angezeigt, und das Anzeigeelement gibt den Wert 3 aus.

Im Blockdiagramm wird zum Zeitpunkt der Umwandlung ein Typumwandlungspunkt auf dem Rand des Anschlusses angezeigt, um anzugeben, dass eine automatische numerische Umwandlung erfolgt ist.

Da VIs und Funktionen über viele Anschlüsse verfügen können, kann ein Typumwandlungspunkt auch im Inneren eines Symbols angezeigt werden, wenn Sie einen Anschluss mit einem anderen Anschluss verbinden, wie im nachstehenden Beispiel gezeigt.



Typumwandlungspunkte können die Laufzeit verlängern und dazu führen, dass ein VI mehr Speicher benötigt. Versuchen Sie, die Datentypen Ihrer VIs konsistent zu halten.

Polymorphe VIs und Funktionen

Polymorphe VIs und Funktionen können an Eingabedaten mit unterschiedlichen Datentypen angepasst werden. Die meisten LabVIEW-Strukturen sind polymorph, so auch einige VIs und Funktionen.

Polymorphe VIs

Polymorphe VIs akzeptieren unterschiedliche Datentypen für einen einzigen Eingabe- oder Ausgabeanschluss. Ein polymorphes VI ist eine Sammlung aus Sub-VIs mit den gleichen Anschlussfeldmustern. Jedes Sub-VI ist eine Instanz des polymorphen VIs.

Beispielsweise ist das VI Schlüssel lesen [Read Key.vi] polymorph. Dessen **Standardwert**-Anschluss akzeptiert boolesche Werte, numerische Fließkommawerte mit doppelter Genauigkeit, vorzeichenbehaftete 32-Bit-Ganzzahlen, Pfad- und String-Daten oder vorzeichenlose 32-Bit-Ganzzahlen.

Erstellen von polymorphen VIs

Wenn Sie die gleiche Operation mit unterschiedlichen Datentypen durchführen, können Sie eigene polymorphe VIs erstellen.

Wenn Sie beispielsweise die gleiche mathematische Operation mit einem Fließkommawert mit einfacher Genauigkeit, mit einem aus Zahlen bestehenden Array oder mit einem Signalverlauf durchführen, könnten Sie drei unterschiedliche VIs erstellen, nämlich „Zahl berechnen“, „Array berechnen“ und „Signalverlauf berechnen“. Wenn die Operation dann durchgeführt werden soll, setzen Sie eines dieser VIs in das Blockdiagramm, und zwar abhängig vom Datentyp, der bei der Eingabe verwendet wird.

Anstatt nun jeweils manuell eine Version des VIs in das Blockdiagramm zu setzen, können Sie auch ein einziges polymorphes VI erstellen und verwenden. Das polymorphe VI Berechnen enthält drei Instanzen des VIs, wie in Abbildung 5-2 dargestellt.

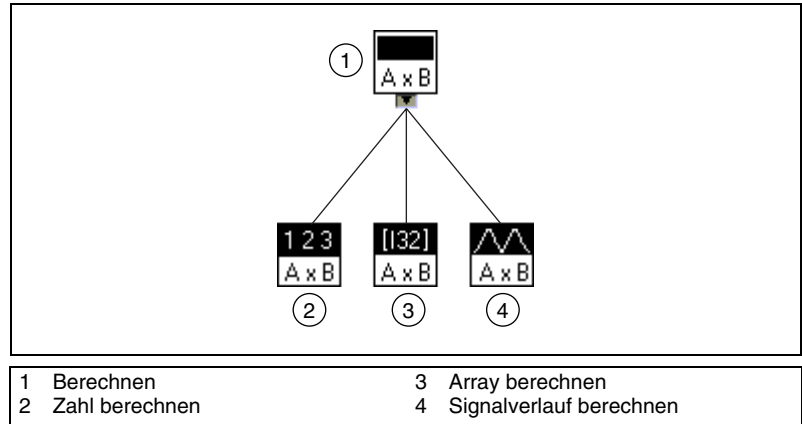


Abbildung 5-2. Beispiel für ein polymorphes VI

Das VI Berechnen verknüpft statisch die korrekte Instanz des VIs basierend auf dem Datentyp, den Sie mit dem Sub-VI Berechnen im Blockdiagramm verbinden, wie in Abbildung 5-3 dargestellt.

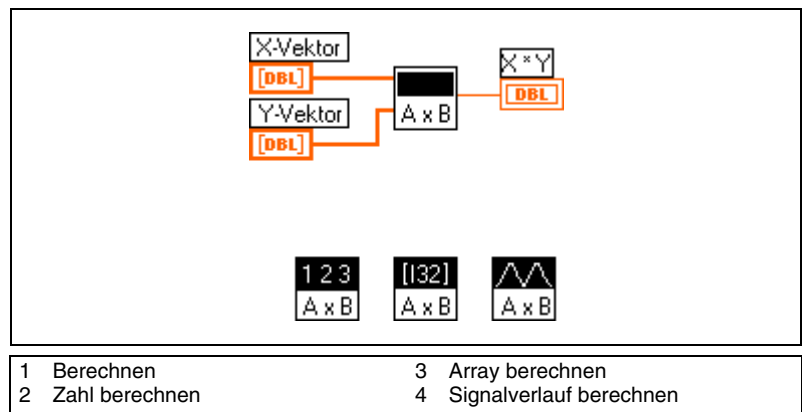


Abbildung 5-3. Polymorphes VI, das ein Sub-VI aufruft

Polymorphe VIs unterscheiden sich von den meisten VIs in der Weise, dass sie kein Blockdiagramm oder Frontpanel haben.

Berücksichtigen Sie beim Erstellen von polymorphen VIs die folgenden Aspekte:

- Alle VIs, die Sie in das polymorphe VI einschließen, müssen über das gleiche Anschlussfeldmuster verfügen, denn das Anschlussfeld des polymorphen VIs entspricht dem Anschlussfeld der VIs, die Sie zum Erstellen des polymorphen VIs verwenden.
- Die Eingänge und Ausgänge am Anschlussfeld jeder Instanz des VIs müssen den Eingängen und Ausgängen am Anschlussfeld des polymorphen VIs entsprechen.
- Die VIs, die Sie zum Erstellen von polymorphen VIs verwenden, müssen nicht aus den gleichen Sub-VIs und Funktionen bestehen.
- Die jeweiligen Frontpanels der VIs müssen nicht über die gleiche Anzahl Objekte verfügen. Allerdings muss jedes Frontpanel zumindest über die gleiche Anzahl Bedien- und Anzeigeelemente verfügen, aus denen sich das Anschlussfeld des polymorphen VIs zusammensetzt.
- Sie können ein Symbol für ein polymorphes VI erstellen.
- Sie können keine polymorphen VIs in anderen polymorphen VIs verwenden.

Wenn Sie eine vollständige Dokumentation für ein VI erstellen, das ein polymorphes Sub-VI enthält, erscheinen das polymorphe VI und die hiervon aufgerufenen VIs in der **Liste der Sub-VIs**.

Polymorphe Funktionen

Funktionen sind unterschiedlich polymorph: keine, einige oder alle Eingänge können polymorph sein. Einige Funktionseingänge akzeptieren Zahlen oder boolesche Werte. Einige akzeptieren Zahlen oder Strings. Einige lassen nicht nur skalare Zahlen, sondern auch Zahlen-Arrays, Zahlen-Cluster, Arrays aus Zahlen-Clustern und so weiter zu. Einige akzeptieren nur eindimensionale Arrays, wobei die Array-Elemente jeden beliebigen Datentyp aufweisen können. Einige Funktionen lassen alle Datentypen zu, komplexe Zahlen eingeschlossen. Weitere Informationen zu polymorphen Funktionen finden Sie in Anhang B, [Polymorphe Funktionen](#).

Verarbeiten von Variant-Daten

Variant-Daten entsprechen nicht einem bestimmten Datentyp und enthalten Attribute wie Kanalnamen und Kanaleinheiten. LabVIEW stellt Variant-Daten als Datentyp „Variant“ dar. Der Datentyp Variant unterscheidet sich von anderen Datentypen, da er den Namen des Bedien- oder Anzeigeelements, die Informationen zum Datentyp und die Daten selbst umfasst.

Verwenden Sie die Variant-Funktionen, die sich auf der Palette **Funktionen»Fortgeschritten»Datenmanipulation»Variant** befinden, zum Erstellen und Bearbeiten von Variant-Daten. Sie können jeden beliebigen LabVIEW-Datentyp in den Datentyp „Variant“ umwandeln, um Variant-Daten in anderen VIs und Funktionen zu verwenden. Wenn Sie beispielsweise einen String in einen Variant umwandeln, speichert der Variant den Text und zeigt an, dass es sich bei dem Text um einen String handelt.

Verwenden Sie den Datentyp Variant, wenn es wichtig ist, Daten unabhängig vom Datentyp zu bearbeiten. Sie können Daten auch ohne den Datentyp Variant typunabhängig darstellen, indem Sie die Daten in geglättete Strings umwandeln. Beispielsweise gibt die Methode **Alle Bedienelement-Werte lesen** Informationen über die Bedien- und Anzeigeelemente eines VIs als ein Array aus Clustern zurück. Jeder Cluster im Array enthält den Datentyp und den Wert jedes Bedien- oder Anzeigeelements. Der Cluster enthält den Namen des Bedien- oder Anzeigeelements als String, den Datentyp als ein Array aus 16-Bit-Ganzzahlen und die Daten als geglättete Strings, wie in Abbildung 5-4 dargestellt.

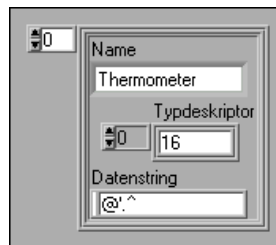


Abbildung 5-4. Cluster aus numerischen geglätteten Daten

Bei der Verwendung von geglätteten Daten gibt es jedoch auch Beschränkungen, da LabVIEW geglättete Daten nicht anpassen kann. Darüber hinaus ist es auch nicht möglich, eine in geglättete Daten umgewandelte Ganzzahl als Fließkommazahl mit erweiterter Genauigkeit darzustellen. Sie sollten daher den Datentyp „Variant“ verwenden, um mit

Daten unabhängig vom Datentyp zu arbeiten, anstatt die Daten in geglättete Daten umzuwandeln. Weitere Informationen zum Umwandeln von Daten in geglättete Daten und umgekehrt finden Sie in Abschnitt *Flattened Data* der Applikationsinformationen *LabVIEW Data Storage*.

Sie können Attribute hinzufügen, um die Variant-Daten genauer zu kennzeichnen. Beispielsweise können Sie mit einem Attribut für Variant-Daten angeben, von welchem Datenerfassungsgeräte-Kanal die Daten stammen.

Variant-Daten erweisen sich auch als nützlich, wenn Sie in LabVIEW an den Speicher schreiben und aus dem Speicher lesen und Operationen durchführen, die Stapel (Last-In-First-Out, LIFO), Queues (First-In-First-Out, FIFO), intelligente Puffer oder Bäume einbeziehen. Bei diesen Operationsarten werden Daten unabhängig vom Datentyp behandelt.

Numerische Einheiten und strikte Typenprüfung

Sie können physikalische Maßeinheiten wie beispielsweise Meter oder Sekunde mit jedem beliebigen numerischen Bedienelement verbinden, das über eine Möglichkeit zur Fließkommadarstellung verfügt.

Die Einheiten für ein Bedienelement werden in einer separaten verknüpften Beschriftung, der so genannten Einheitenbeschriftung, angezeigt. Sie zeigen die Einheitenbeschriftung an, indem Sie mit der rechten Maustaste auf das Bedienelement klicken und aus dem Kontextmenü **Sichtbare Elemente»Einheitenbeschriftung** wählen.

Wenn LabVIEW die Einheitenbeschriftung anzeigt, können Sie eine Einheit unter Verwendung der Standardabkürzungen wie beispielsweise m für Meter, ft für Fuß, s für Sekunde und so weiter eingeben.



Hinweis In Formelknoten können keine Einheiten verwendet werden.

Einheiten und strikte Typenüberprüfung

Wenn Sie einem Objekt Einheiten zuweisen, können Sie nur Objekte verbinden, die über kompatible Einheiten verfügen. LabVIEW verwendet die strikte Typenüberprüfung, um sicherzustellen, dass die Einheiten kompatibel sind. Wenn Sie zwei Objekte mit inkompatiblen Einheiten verbinden, gibt LabVIEW einen Fehler zurück. Beispielsweise gibt LabVIEW einen Fehler zurück, wenn Sie ein Objekt mit Meilen als Einheit mit einem Objekt mit Liter als Einheit verbinden, da Meilen ein Längenmaß und Liter ein Raumaß ist.

Abbildung 5-5 zeigt, wie Objekte mit kompatiblen Einheiten verbunden werden. In dieser Abbildung skaliert LabVIEW automatisch den **Entfernungs**-Anzeiger so, dass Kilometer anstelle von Metern angezeigt werden, da das Anzeigeelement auf Kilometer eingestellt ist.

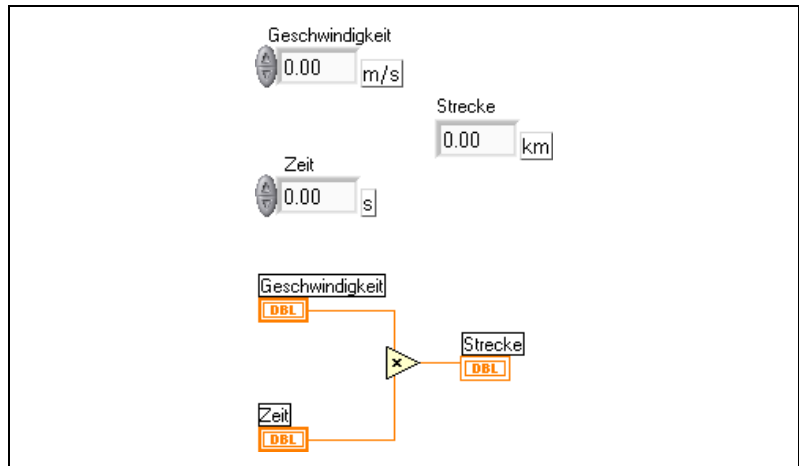


Abbildung 5-5. Verbinden von Objekten mit kompatiblen Einheiten

In Abbildung 5-6 wird ein Fehler zurückgegeben, weil für **Entfernung** die Einheit „Sekunde“ angegeben wurde. Zum Beheben dieses Fehlers ändern Sie Sekunden in ein Längenmaß wie beispielsweise Kilometer, wie in Abbildung 5-5 gezeigt.

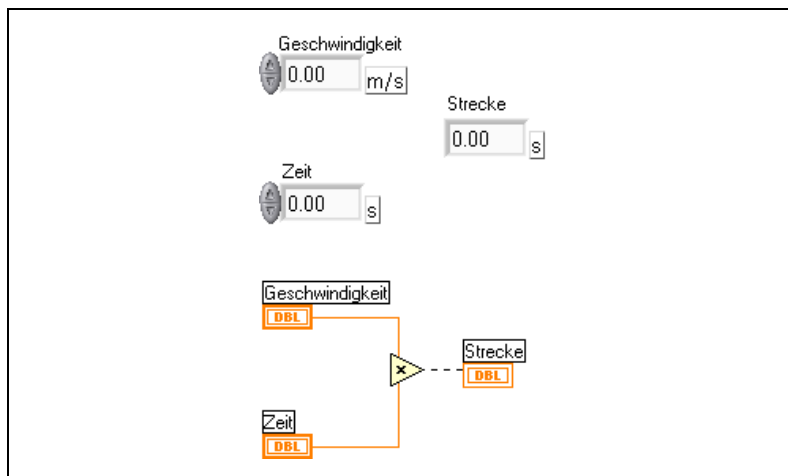


Abbildung 5-6. Das Verbinden von Objekten mit inkompatiblen Einheiten führt zu unterbrochenen Verbindungen

Zum Anzeigen des Fehlers bewegen Sie das Verbindungswerkzeug über die unterbrochene Verbindung, bis ein Hinweis angezeigt wird, oder klicken Sie mit der rechten Maustaste auf die unterbrochene Verbindung, und wählen Sie aus dem Kontextmenü **Fehler anzeigen**. Im Fenster **Fehlerliste** wird der folgende Fehler aufgeführt:

Sie haben numerische Datentypen miteinander verbunden, deren Einheiten nicht kompatibel sind.

Einige VIs und Funktionen sind im Hinblick auf Einheiten zweideutig. Diese VIs und Funktionen können Sie nicht mit anderen Anschlüssen verwenden, die Einheiten aufweisen. Beispielsweise ist die Funktion „Inkrement“ zweideutig im Hinblick auf Einheiten. Wenn Sie Längenmaße verwenden, kann die Funktion „Inkrement“ nicht feststellen, ob ein Meter, ein Kilometer oder ein Fuß hinzugefügt werden soll. Aufgrund dieser Zweideutigkeit können Sie die Funktion „Inkrement“ und andere Funktionen, mit denen Werte erhöht oder verringert werden, nicht mit Daten verwenden, denen Einheiten zugewiesen wurden.

Zur Vermeidung der Zweideutigkeit in diesem Beispiel verwenden Sie eine numerische Konstante mit der korrekten Einheit sowie die Additionsfunktion, um eine eigene Inkrement-Funktion zum Erhöhen von Werten mit Einheiten zu erstellen, wie in Abbildung 5-7 dargestellt.

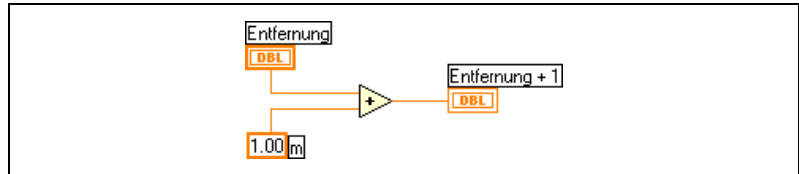


Abbildung 5-7. Erstellen einer Inkrement-Funktion mit Einheiten

Blockdiagramm-Datenfluss

Für das Ausführen von VIs folgt LabVIEW einem Datenflussmodell. Ein Blockdiagrammknoten wird ausgeführt, wenn alle Eingaben verfügbar sind. Wenn die Ausführung eines Knotens abgeschlossen ist, werden die Daten an die jeweiligen Ausgabeanschlüsse übergeben und die Ausgabedaten dann an den nächsten Knoten im Datenflusspfad weitergeleitet.

Visual Basic, C++, JAVA und die meisten anderen textbasierten Programmiersprachen folgen einem Steuerflussmodell für die Programmausführung. Im Steuerfluss bestimmt die sequenzielle Reihenfolge der Programmelemente die Ausführungsreihenfolge eines Programms.

Weil in LabVIEW der Datenfluss und nicht die sequenzielle Reihenfolge der Befehle die Ausführungsreihenfolge der Blockdiagrammelemente bestimmt, können Sie Blockdiagramme erstellen, die simultane Operationen beinhalten. LabVIEW ist ein Multitasking- und Multithread-System, bei dem mehrere Ausführungs-Threads und mehrere VIs gleichzeitig ausgeführt werden können. Weitere Informationen zum simultanen Ausführen von Aufgaben finden Sie in den Applikationsinformationen (Application Notes) *Using LabVIEW to Create Multithreaded VIs for Maximum Performance and Reliability*.

Datenabhängigkeit und künstliche Datenabhängigkeit

Das Steuerflussmodell für die Programmausführung ist anweisungs-gesteuert. Die Datenflussaufführung läuft hingegen datengesteuert oder datenabhängig ab. Ein Knoten, der Daten von einem anderen Knoten empfängt, wird immer ausgeführt, nachdem die Ausführung des anderen Knotens abgeschlossen ist.

Blockdiagrammknoten, die untereinander keine Verbindungen aufweisen, können in jeder beliebigen Reihenfolge ausgeführt werden. Obwohl in den *LabVIEW Development Guidelines* empfohlen wird, ein von links nach rechts und von oben nach unten orientiertes Layout zu verwenden, werden Knoten nicht notwendigerweise in der Reihenfolge von links nach rechts und von oben nach unten ausgeführt.

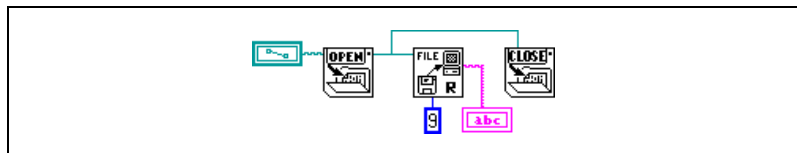
Eine Sequenzstruktur stellt eine Möglichkeit dar, um die Ausführungsreihenfolge zu steuern, wenn keine natürliche Datenabhängigkeit besteht. Weitere Informationen zu Sequenzstrukturen finden Sie in Abschnitt *Sequenzstrukturen* des Kapitels 8, *Schleifen und Case-Strukturen*. Sie können jedoch auch Durchflussparameter zum Steuern der Ausführungsreihenfolge verwenden. Weitere Informationen zu Durchflussparametern finden Sie in Abschnitt *Durchfluss-Parameter* des Kapitels 13, *Datei-I/O*.

Eine weitere Möglichkeit zum Steuern der Ausführungsreihenfolge besteht darin, eine künstliche Datenabhängigkeit zu schaffen, in welcher der empfangende Knoten die empfangenen Daten nicht wirklich nutzt. Stattdessen verwendet der empfangende Knoten den Datenempfang als Auslöser für die Programmausführung. Ein Beispiel für künstliche Datenabhängigkeit finden Sie im Timing Template (data dep) VI in `examples\general\structs.llb`.

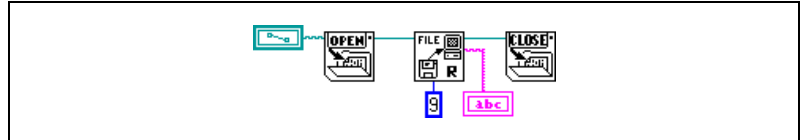
Fehlende Datenabhängigkeit

Gehen Sie nicht von einer Ausführungsreihenfolge von links nach rechts oder von oben nach unten aus, wenn keine Datenabhängigkeit existiert. Stellen Sie sicher, dass Sie die Sequenz der Ereignisse gegebenenfalls explizit definieren, indem Sie den Datenfluss durch die entsprechenden Verbindungen vorgeben.

Im folgenden Beispiel besteht keine Abhängigkeit zwischen den Funktionen „Datei lesen“ und „Datei schließen“, da die Funktion „Datei lesen“ nicht mit der Funktion „Datei schließen“ verbunden ist. Dieses Beispiel funktioniert möglicherweise nicht wie erwartet, weil es keine Möglichkeit gibt, um festzulegen, welche Funktion zuerst ausgeführt wird. Wenn die Funktion „Datei schließen“ zuerst ausgeführt wird, arbeitet die Funktion „Datei lesen“ nicht.



Mit dem folgenden Blockdiagramm wird eine Abhängigkeit eingerichtet, indem ein Ausgang der Funktion „Datei lesen“ mit der Funktion „Datei schließen“ verbunden wird. Die Funktion „Datei schließen“ wird erst ausgeführt, wenn die Ausgabe der Funktion „Datei lesen“ empfangen wird.



Datenfluss und Speicherverwaltung

Mit dem Datenfluss-Ausführungsmodell gestaltet sich die Speicherverwaltung einfacher als mit dem Steuerfluss-Ausführungsmodell. In LabVIEW werden keine Variablen zugeordnet oder diesen Werte zugewiesen. Stattdessen erstellen Sie ein Blockdiagramm mit Verbindungen, welche die Datenübergabe repräsentieren.

VI und Funktionen, die Daten generieren, weisen auch automatisch den Speicher für diese Daten zu. Wenn das VI oder die Funktion die Daten nicht mehr verarbeitet, hebt LabVIEW die Speicherzuweisung auf. Wenn Sie einem Array oder String neue Daten hinzufügen, weist LabVIEW wieder ausreichend Speicher zu, um die neuen Daten zu verwalten.

Da LabVIEW den Speicher automatisch verwaltet, haben Sie weniger Kontrolle darüber, wann Speicher zugewiesen und wann die Zuweisung aufgehoben wird. Wenn das VI mit großen Datensätzen arbeitet, müssen Sie wissen, wann die Speicherzuweisung erfolgt. Ein Verständnis der zugrunde liegenden Prinzipien kann Ihnen dabei helfen, VIs zu schreiben, die bedeutend weniger Speichermanforderungen stellen. Eine geringere Speichernutzung kann dazu beitragen, die Geschwindigkeit zu steigern, mit der VIs ausgeführt werden. Weitere Informationen zur Speicherzuweisung finden Sie in den Applikationsinformationen (Application Notes) *LabVIEW Performance and Memory Management*.

Entwerfen des Blockdiagramms

Legen Sie beim Entwurf von Blockdiagrammen die folgenden Richtlinien zugrunde:

- Verwenden Sie ein von links nach rechts und von oben nach unten orientiertes Layout. Obwohl die Position der Blockdiagrammelemente keinen Einfluss auf die Ausführungsreihenfolge hat, wirkt das Blockdiagramm geordneter und ist einfacher zu verstehen, wenn Sie eine Datenflußrichtung beibehalten. Nur die Verbindungen und die Strukturen bestimmen die Ausführungsreihenfolge.
- Vermeiden Sie, Blockdiagramme zu erstellen, die mehr als einen oder zwei Bildschirme in Anspruch nehmen. Wenn ein Blockdiagramm umfangreich und komplex wird, kann es schwer verständlich werden. Die Fehlersuche wird dadurch eventuell auch erschwert.
- Prüfen Sie, ob Sie einige Komponenten des Blockdiagramms in anderen VIs wiederverwenden können oder ob ein Abschnitt des Blockdiagramms als logische Komponente zusammengefasst werden kann. Falls ja, teilen Sie das Blockdiagramm in Sub-VIs auf, die bestimmte Aufgaben erledigen. Die Verwendung von Sub-VIs hilft bei der Änderungsverwaltung und sorgt dafür, dass Fehler in Blockdiagrammen schnell behoben werden können. Weitere Informationen über Sub-VIs finden Sie in Abschnitt *Sub-VIs* des Kapitels 7, *Erstellen von VIs und Sub-VIs*.
- Verwenden Sie die Fehlerbehandlungs-VIs, -Funktionen und -Parameter, um Fehler im Blockdiagramm zu beheben. Weitere Informationen zur Fehlerbehandlung finden Sie in Abschnitt *Fehlerprüfung und Fehlerbehandlung* des Kapitels 6, *Ausführen von und Fehlersuche in VIs*.
- Verbessern Sie das Erscheinungsbild des Blockdiagramms, indem Sie Verbindungen in effizienter Weise herstellen. Eine ungenügende Verbindungsorganisation produziert zwar möglicherweise keine Fehler, kann jedoch die Ursache dafür sein, dass das Blockdiagramm schwierig zu lesen oder die Fehlersuche erschwert ist. Oder es kann der Anschein erweckt werden, dass ein VI Funktionen ausführt, die nicht vorgesehen sind.
- Vermeiden Sie, Verbindungen unter einem Strukturrahmen oder zwischen sich überlappenden Objekten zu ziehen, da LabVIEW in diesem Fall möglicherweise einige Verbindungssegmente überdeckt.

- Vermeiden Sie, Objekte über Verbindungen zu platzieren. Mit Verbindungen werden nur die Objekte verbunden, auf die Sie klicken. Wenn Sie einen Anschluss oder ein Symbol auf eine Verbindung setzen, könnte der Eindruck entstehen, als bestünde keine Verbindung.
- Verwenden Sie das Farbwerkzeug, und klicken Sie mit der rechten Maustaste auf den Blockdiagramm-Arbeitsbereich, um die Farbe des Arbeitsbereichs zu ändern.
- Um den Leerraum zwischen gehäuften oder zu nahe gruppierten Objekten zu vergrößern, drücken Sie die <Strg>-Taste und klicken mit dem Positionierwerkzeug in den Blockdiagramm-Arbeitsbereich. Während Sie die Taste gedrückt halten, ziehen Sie einen Bereich in der Größe auf, wie er eingefügt werden soll.

(Macintosh) Drücken Sie die <Option>-Taste. **(Sun)** Drücken Sie die <Meta>-Taste. **(HP-UX und Linux)** Drücken Sie die <Alt>-Taste.

Ein von einem gestrichelten Rahmen umschlossenes Rechteck zeigt, wo der Leerraum eingefügt wird. Lassen Sie die Taste los, und fügen Sie den Leerraum ein.

Ausführen von und Fehlersuche in VIs

Zum Ausführen eines VIs müssen Sie zunächst alle Sub-VIs, Funktionen und Strukturen mit den von den Anschlüssen erwarteten Datentypen verbinden. Es kann vorkommen, dass ein VI Daten produziert oder nicht wie geplant abläuft. Mit Hilfe von LabVIEW können Sie konfigurieren, wie ein VI ausgeführt wird und Probleme bei der Blockdiagrammanordnung beziehungsweise bei den durch das Blockdiagramm laufenden Daten erkennen.

Weitere Informationen ...

Weitere Informationen über die Fehlersuche in VIs finden Sie in der *LabVIEW-Hilfe*.

Ausführen von VIs



Beim Ausführen eines VIs wird die Operation durchgeführt, für die Sie das VI erstellt haben. Sie können ein VI ausführen, wenn die Schaltfläche **Ausführen** auf der Symbolleiste als weißer Pfeil wie links dargestellt erscheint. Der weiße Pfeil zeigt darüber hinaus an, dass Sie das VI als Sub-VI verwenden können, wenn Sie ein Anschlussfeld für das VI erstellt haben.

Das VI wird ausgeführt, wenn Sie auf eine der Schaltflächen **Ausführen** oder **Wiederholt ausführen** oder auf die Einzelschrittschaltflächen auf der Blockdiagramm-Symbolleiste klicken. Mit dem Klicken auf die Schaltfläche **Ausführen** wird das VI einmal ausgeführt. Das VI stoppt, wenn dessen Datenfluss abgeschlossen ist. Wenn Sie auf die Schaltfläche **Wiederholt ausführen** klicken, wird das VI kontinuierlich ausgeführt, bis Sie am VI auf die Schaltfläche **Ausführung abbrechen** klicken. Wenn Sie auf die Einzelschrittschaltflächen klicken, wird das VI schrittweise ausgeführt. Weitere Informationen zum Einsatz der Einzelschrittschaltflächen bei der Fehlersuche in VIs finden Sie in Abschnitt [Einzelschrittausführung](#) dieses Kapitels.

Ausführungskonfiguration eines VIs

Wählen Sie **Datei»VI-Einstellungen**, und wählen Sie aus dem Pull-down-Menü **Kategorie Ausführung**, wenn Sie konfigurieren möchten, wie ein VI ausgeführt wird. Sie können ein VI beispielsweise so konfigurieren, dass es nach dem Öffnen sofort gestartet wird, oder dass es anhält, wenn es als Sub-VI aufgerufen wird. Sie können das VI auch so konfigurieren, dass es mit unterschiedlichen Prioritäten ausgeführt wird. Wenn ein VI beispielsweise unbedingt ausgeführt werden muss, ohne auf den Abschluss einer anderen Operation zu warten, konfigurieren Sie das VI zur Ausführung mit zeitkritischer (höchster) Priorität. Weitere Informationen über das Erstellen von Multithread-VIs finden Sie in den Applikationsinformationen *Using LabVIEW to Create Multithreaded VIs for Maximum Performance and Reliability*. Weitere Informationen zum Konfigurieren der Art und Weise der VI Ausführung finden Sie in Kapitel 15, *Anpassen von VIs*.

Korrigieren von nicht ausführbaren VIs



Wenn ein VI nicht läuft, handelt es sich um ein fehlerhaftes oder nicht ausführbares VI. Die Schaltfläche **Ausführen** erscheint häufig als unterbrochen (siehe links), während Sie ein VI erstellen oder bearbeiten. Wenn sie in dieser Darstellungsweise auch dann noch angezeigt wird, wenn Sie mit dem Verbinden des Blockdiagramms fertig sind, ist das VI fehlerhaft und kann nicht ausgeführt werden.

Herausfinden der Ursachen für nicht ausführbare VIs

Klicken Sie auf die gebrochene Schaltfläche **Ausführen**, oder wählen Sie **Fenster»Fehlerliste anzeigen**, um herauszufinden, warum ein VI nicht ausgeführt werden kann. Im Fenster **Fehlerliste** werden alle Fehler aufgeführt. Im Abschnitt **VI-Liste** werden die Namen aller VIs im Speicher aufgeführt, die fehlerhaft sind. Im Abschnitt **Fehler und Warnungen** werden die Fehler und Warnungen für das VI angezeigt, das Sie im Abschnitt **VI-Liste** markiert haben. Im Abschnitt **Details** werden die Fehler und in manchen Fällen auch Empfehlungen zur Fehlerbehebung und Hinweise auf weitere Informationen hierzu angezeigt. Wählen Sie **Hilfe**, um eine Online-Hilfedatei zu öffnen, in der die LabVIEW-Fehler und deren Beschreibungen aufgeführt sind.

Klicken Sie auf die Schaltfläche **Fehler anzeigen**, oder doppelklicken Sie auf die Fehlerbeschreibung, um das relevante Blockdiagramm oder Frontpanel anzuzeigen und um das Objekt hervorzuheben, das den Fehler enthält.



Wenn Sie das Kontrollkästchen **Warnungen anzeigen** im Fenster **Fehlerliste** aktiviert haben und für ein VI eine Warnung ausgegeben wird, enthält die Symbolleiste die Schaltfläche **Warnung** (siehe links).

Sie konfigurieren LabVIEW so, dass Warnungen immer im Fenster **Fehlerliste** angezeigt werden, indem Sie **Werkzeuge**»**Optionen** wählen, dann **Fehlersuche** aus dem Pulldown-Menü wählen und nun das Kontrollkästchen **Standardmäßig Warnungen im Fehlerfenster anzeigen** aktivieren. Sie können diese Änderung bei geöffnetem Fenster **Fehlerliste** vornehmen, wo die Änderung dann sofort angezeigt wird.

Warnungen verhindern nicht das Ausführen eines VIs. Sie sollen Ihnen helfen, potenzielle Probleme mit VIs zu vermeiden.

Häufige Ursachen für nicht ausführbare VIs

Die folgende Liste enthält häufige Ursachen, warum ein VI als nicht ausführbar gekennzeichnet wird, während Sie es bearbeiten:

- Das Blockdiagramm enthält eine unterbrochene Verbindung aufgrund sich nicht entsprechender Datentypen oder loser, nicht verbundener Enden. Weitere Informationen zum Verbinden von Blockdiagrammobjekten finden Sie in Abschnitt *Verbindung von Blockdiagrammobjekten* des Kapitels 5, *Erstellen des Blockdiagramms*.
- Ein obligatorischer Blockdiagrammanschluss wurde nicht verbunden. Wählen Sie **Hilfe**»**Zeige Kontext-Hilfe**, oder lesen Sie die *LabVIEW-Hilfe*, um zu erfahren, welche Parameter für einen Blockdiagrammknoten erforderlich sind.
- Ein Sub-VI ist nicht ausführbar, oder Sie haben dessen Anschlussfeld bearbeitet, seit Sie das Symbol des Sub-VIs auf das Blockdiagramm des VIs gesetzt haben. Weitere Informationen zu Sub-VIs finden Sie in Abschnitt *Sub-VIs* des Kapitels 7, *Erstellen von VIs und Sub-VIs*.

Techniken für die Fehlersuche

Wenn ein VI zwar ausführbar ist, Sie jedoch unerwartete Daten erhalten, können Sie mit Hilfe der folgenden Techniken Probleme mit dem VI oder dem Blockdiagramm-Datenfluss erkennen und beheben.

- Verbinden Sie die beiden Parameter **Fehlereingang** und **Fehlerausgang** im unteren Bereich der am meisten integrierten VIs und Funktionen. Mit diesen Parametern können Fehler erkannt werden, die in jedem Knoten im Blockdiagramm aufgetreten sind, und es wird angegeben, ob und wo ein Fehler eingetreten ist. Sie können diese Parameter auch für die von Ihnen erstellten VIs verwenden. Weitere Informationen zum Einsatz dieser Parameter finden Sie in Abschnitt [Fehlerbehandlung](#) dieses Kapitels.
- Verwenden Sie die Highlight-Funktion, wenn Sie die Daten auf ihrem Weg durch das Blockdiagramm beobachten möchten.
- Sie können sich in Einzelschritten durch das VI bewegen, um jede Aktion des VIs im Blockdiagramm anzuzeigen.
- Verwenden Sie das Probe-Daten-Werkzeug, um während des Ablaufs des VIs Zwischenwerte zu prüfen.
- Mit Hilfe von Haltepunkten können Sie die Ausführung anhalten, um in Einzelschritten fortzufahren oder um Proben einzuführen.
- Setzen Sie die Ausführung eines Sub-VIs aus, um Werte von Bedien- und Anzeigeelementen zu bearbeiten, um die Anzahl der Durchläufe zu steuern oder um zum Ausführungsbeginn des Sub-VIs zurückzukehren.
- Deaktivieren Sie einen Abschnitt des Blockdiagramms vorübergehend, um festzustellen, ob das VI ohne diesen Abschnitt bessere Leistungen zeigt.

Highlight-Funktion



Sie können eine Animation der Ausführung des Blockdiagramms anzeigen, indem Sie auf das Symbol der **Highlight-Funktion** wie links dargestellt klicken. Mit der Highlight-Funktion kann der Weg der Daten im Blockdiagramm von einem Knoten zu einem anderen mit Hilfe von Daten-Blasen angezeigt werden, die sich entlang der Verbindungen bewegen. Verwenden Sie die Highlight-Funktion in Verbindung mit der Einzelschrittausführung, um zu sehen, wie sich die Daten von Knoten zu Knoten durch ein VI bewegen.



Hinweis Mit aktivierter Highlight-Funktion wird die Ausführungsgeschwindigkeit des VIs erheblich herabgesetzt.

Einzelschrittausführung

Sie können sich in Einzelschritten durch das VI bewegen, um jede Aktion des laufenden VIs im Blockdiagramm anzuzeigen. Mit den Einzelschrittschaltflächen lässt sich die Ausführung eines VIs oder Sub-VIs nur im Einzelschrittmodus beeinflussen. Sie wechseln in den Einzelschrittmodus, indem Sie auf der Blockdiagramm-Symbolleiste auf eine der Schaltflächen **Überspringen** oder **Hineinspringen** klicken. Bewegen Sie den Cursor über die Schaltfläche **Überspringen**, **Hineinspringen** oder **Heraus-springen**, um einen Hinweisstreifen anzuzeigen, der den nächsten Schritt beschreibt, der nach dem Klicken auf diese Schaltfläche erfolgt. Sie können Sub-VIs im Einzelschrittmodus oder im Normalmodus ablaufen lassen.



Wenn Sie sich im Einzelschrittmodus mit aktivierter Highlight-Funktion durch ein VI bewegen, erscheint ein Ausführungs-Glyph wie links gezeigt auf den Symbolen der Sub-VIs, die aktuell ausgeführt werden.

Probe-Daten-Werkzeug



Verwenden Sie das Probe-Daten-Werkzeug wie links gezeigt, um Zwischenwerte auf einer Verbindung zu prüfen, während ein VI ausgeführt wird. Das Probe-Daten-Werkzeug eignet sich besonders für komplizierte Blockdiagramme mit einer Reihe von Operationen, von denen jede unrichtige Daten zurückgeben könnte. Verwenden Sie das Probe-Daten-Werkzeug zusammen mit der Highlight-Funktion, dem Einzelschrittmodus und Haltepunkten, um festzustellen, ob und wo die Daten fehlerhaft sind. Wenn Daten verfügbar sind, wird die Probe unverzüglich, während des Einzelschrittmodus oder wenn die Ausführung an einem Haltepunkt unterbrochen wurde, aktualisiert. Wenn die Ausführung aufgrund eines Einzelschrittes oder aufgrund eines Haltepunktes an einem Knoten angehalten wird, können Sie auch die Verbindung prüfen, die gerade ausgeführt wird, um den Wert zu sehen, der durch diese Verbindung fließt.

Sie können auch eine benutzerdefinierte Probe erstellen, um anzugeben, welches Anzeigeelement für die Anzeige der ermittelten Daten verwendet werden soll. Wenn Sie beispielsweise numerische Daten anzeigen, können Sie wählen, dass diese Daten in einem Diagramm in der Probe angezeigt werden.

Haltepunkte



Verwenden Sie das Haltepunkt-Werkzeug wie links dargestellt, um einen Haltepunkt in einem VI, an einem Knoten oder einer Verbindung im Blockdiagramm zu platzieren und die Ausführung an diesem Punkt anzuhalten. Wenn Sie einen Haltepunkt an einer Verbindung setzen, wird die Ausführung unterbrochen, wenn die Daten die Verbindung passiert haben. Setzen Sie einen Haltepunkt in den Arbeitsbereich des Blockdiagramms, um die Ausführung zu unterbrechen, nachdem alle Knoten im Blockdiagramm ausgeführt wurden.

Wenn Sie während der Ausführung an einen Haltepunkt gelangen, gehen Sie folgendermaßen vor:

- Bewegen Sie sich mit Hilfe der Einzelschrittschaltflächen in Einzelschritten durch die Ausführung.
- Sondieren Sie Verbindungen, um die Zwischenwerte zu prüfen.
- Setzen Sie die Ausführung bis zum nächsten Haltepunkt fort, oder lassen Sie das VI bis zum Ende durchlaufen.

LabVIEW speichert Haltepunkte mit einem VI; diese sind jedoch nur aktiv, wenn das VI ausgeführt wird.

Aussetzen der Ausführung

Setzen Sie die Ausführung eines Sub-VIs aus, um Werte von Bedien- und Anzeigeelementen zu bearbeiten, um die Anzahl der Durchläufe zu steuern, bevor der Fokus an das aufrufende VI zurückgegeben wird, oder um zum Ausführungsbeginn des Sub-VIs zurückzukehren. Sie können veranlassen, dass alle Aufrufe eines Sub-VIs bei ausgesetzter Ausführung gestartet werden, oder Sie können einen bestimmten Aufruf eines Sub-VIs aussetzen.

Um alle Aufrufe eines Sub-VIs auszusetzen, öffnen Sie das Sub-VI und wählen **Ausführen»Bei Aufruf unterbrechen**. Die Ausführung des Sub-VIs wird automatisch ausgesetzt, wenn es von einem anderen VI aufgerufen wird. Wenn Sie dieses Menüelement im Einzelschrittmodus wählen, wird die Ausführung des Sub-VIs nicht sofort ausgesetzt. Die Ausführung des Sub-VIs wird erst ausgesetzt, wenn es aufgerufen wird.

Um den Aufruf eines bestimmten Sub-VIs auszusetzen, klicken Sie im Blockdiagramm mit der rechten Maustaste auf den Sub-VI-Knoten und wählen **Einstellungen SubVI-Knoten** aus dem Kontextmenü. Aktivieren Sie das Kontrollkästchen **Bei Aufruf unterbrechen**, um die Ausführung nur bei dieser Instanz des Sub-VIs auszusetzen.

Das **Hierarchiefenster**, das Sie anzeigen, indem Sie **Durchsuchen»VI-Hierarchie anzeigen** wählen, zeigt, ob die Ausführung eines VIs angehalten oder ausgesetzt wurde. Ein Pfeil-Glyph zeigt ein VI an, das im regulären oder im Einzelschrittmodus ausgeführt wird. Eine Anhalten-Glyph steht für ein angehaltenes oder ausgesetztes VI. Ein grüner Anhalten-Glyph oder ein ungefüllter Glyph in schwarzweiß zeigt ein VI, dessen Ausführung beim Aufruf angehalten wird. Ein roter Anhalten-Glyph oder ein gefüllter Glyph in schwarzweiß zeigt ein VI, dessen Ausführung aktuell angehalten wurde. Ein Ausrufungszeichen-Glyph gibt an, dass das Sub-VI ausgesetzt wurde. Ein VI kann gleichzeitig angehalten und ausgesetzt werden.

Ermitteln der aktuellen Instanz eines Sub-VIs

Wenn Sie die Ausführung eines Sub-VIs anhalten, wird im Pull-down-Menü **Aufrufkette** die Kette der Aufrufenden vom Top-Level-VI bis zum Sub-VI aufgeführt. Dies ist nicht die gleiche Liste wie diejenige, welche angezeigt wird, wenn Sie **Durchsuchen»Aufrufende dieses VIs** wählen; in dieser Liste werden alle aufrufenden VIs angezeigt, ungeachtet, ob sie aktuell ausgeführt werden. Verwenden Sie das Menü **Aufrufkette**, um die aktuelle Instanz des Sub-VIs zu ermitteln, wenn das Blockdiagramm mehr als eine Instanz enthält. Wenn Sie ein VI aus dem Menü **Aufrufkette** wählen, wird dessen Blockdiagramm geöffnet, und LabVIEW hebt das VI hervor, welches das aktuelle Sub-VI aufgerufen hat.

Vorübergehendes Deaktivieren von Abschnitten des Blockdiagramms

Sie können ein VI ausführen, während ein Abschnitt des Blockdiagramms deaktiviert ist, ähnlich dem Deaktivieren oder Auskommentieren eines Codefragments bei textbasierten Programmiersprachen. Deaktivieren Sie einen Abschnitt des Blockdiagramms vorübergehend, um festzustellen, ob das VI ohne diesen Abschnitt bessere Leistungen zeigt.

Setzen Sie den Abschnitt, den Sie deaktivieren möchten, in eine Case-Struktur, und verwenden Sie eine boolesche Konstante, um beide Cases auszuführen. Weitere Informationen zum Einsatz von Case-Strukturen finden Sie in Abschnitt [Case-Strukturen](#) des Kapitels 8, [Schleifen und Case-Strukturen](#). Sie können auch eine Kopie des VIs erstellen und diesen Abschnitt in der Kopie aus dem Blockdiagramm löschen. Anschließend werfen Sie die Version des VIs, die Sie nicht verwenden möchten.

Deaktivieren der Fehlersuche-Werkzeuge

Sie können die Fehlersuche-Werkzeuge deaktivieren, um die Speicheranforderungen zu reduzieren und die Leistung geringfügig zu steigern. Klicken Sie mit der rechten Maustaste auf das Anschlussfeld, und wählen Sie **VI-Einstellungen**. Wählen Sie aus dem Pulldown-Menü **Kategorie Ausführung**, und deaktivieren Sie das Kontrollkästchen **Debugging aktiviert**.

Undefinierte oder unerwartete Daten

Mit undefinierten Daten, die als NaN (Not a Number, keine Zahl) oder Inf (Infinity, unendlich) ausgegeben werden, werden alle nachfolgenden Operationen ungültig. Fließkommaoperationen geben die folgenden beiden Symbolwerte zurück, um fehlerhafte Berechnungen oder sinnlose Ergebnisse anzuzeigen:

- NaN (Not a Number, keine Zahl) steht für einen Fließkommawert, der von ungültigen Operationen produziert wird, wie beispielsweise die Berechnung der Quadratwurzel aus einer negativen Zahl.
- Inf (Infinity, unendlich) steht für einen Fließkommawert, der von Operationen wie beispielsweise der Division einer Zahl durch Null produziert wird.

LabVIEW prüft ganzzahlige Werte nicht auf Bedingungen wie positiver oder negativer Überlauf. Positiver und negativer Überlauf bei Fließkommawerten entspricht IEEE 754, *Standard for Binary Floating-Point Arithmetic*.

Fließkommaoperationen geben NaN und Inf getreulich weiter. Wenn Sie NaN oder Inf in Ganzzahlen oder boolesche Werte umwandeln, verlieren die Werte ihre Bedeutung. Wenn Sie beispielsweise 1 durch Null dividieren, lautet das Ergebnis Inf. Wenn Sie Inf in eine 16-Bit-Ganzzahl umwandeln, wird der Wert 32.767 erzeugt, der wie ein normaler Wert aussieht. Weitere Informationen zum Umwandeln numerischen Werte finden Sie in Abschnitt *Numerische Konvertierung* des Anhang B, *Polymorphe Funktionen*.

Bevor Sie Daten in einen Integer-Datentyp umwandeln, verwenden Sie das Probe-Daten-Werkzeug, um die als Zwischenergebnisse produzierten Fließkommawerte auf Gültigkeit zu prüfen. Prüfen Sie auf NaN, indem Sie die Vergleichsfunktion „Keine Zahl/Kein Pfad/Keine RefNum?“ mit dem Wert verbinden, den Sie für möglicherweise ungültig halten.

Unerwartete und Standarddaten in Schleifen

For-Schleifen produzieren unerwartete Werte, wenn die Zahl der Schleifendurchläufe einer auto-indizierten For-Schleife Null ist. While-Schleifen produzieren Standarddaten, wenn das Schieberegister nicht initialisiert wurde.

For-Schleifen

For-Schleifen produzieren unerwartete Werte, wenn Sie 0 für die Anzahl der Durchläufe der For-Schleife angeben oder wenn Sie ein leeres Array an einem Eingang mit aktivierter Auto-Indizierung mit der For-Schleife verbinden. Weitere Informationen zu For-Schleifen finden Sie in Abschnitt *For-Schleifen* des Kapitels 8, *Schleifen und Case-Strukturen*. Weitere Informationen zur Auto-Indizierung finden Sie in Abschnitt *Auto-Indizierung von Schleifen* des Kapitels 8, *Schleifen und Case-Strukturen*. Die Ausgabe für jeden Tunnel, den Sie am Rand mit deaktivierter Indizierung konstruiert haben, ist ein leeres Array.

While-Schleifen

Wenn Sie das Schieberegister einer While-Schleife nicht initialisieren, ist die Ausgabe gleich dem Standardwert für den Parameter (0, FALSE, leerer String und so weiter) oder gleich dem letzten Wert, der in das Schieberegister geladen wurde, als das VI zum letzten Mal ausgeführt wurde.

Standarddaten in Arrays

Eine Indizierung über die Grenzen eines Arrays hinaus produziert den Standardwert für den Parameter des Array-Elements. Sie können die Funktion „Array-Größe“ verwenden, um die Größe des Arrays zu ermitteln. Weitere Informationen zu Arrays finden Sie in Abschnitt *Arrays* des Kapitels 9, *Gruppieren von Daten mit Hilfe von Strings, Arrays und Clustern*. Weitere Informationen zur Indizierung finden Sie in Abschnitt *Auto-Indizierung von Schleifen* des Kapitels 8, *Schleifen und Case-Strukturen*. Sie können ungewollt eine Indizierung über die Grenzen eines Arrays hinaus veranlassen, indem Sie das Array mit einer While-Schleife über das letzte Element hinaus indizieren, indem Sie einen zu großen Wert für die Eingabe **Index** der Funktion „Array indizieren“ angeben oder indem Sie ein leeres Array an die Funktion „Array indizieren“ übergeben.

Verhindern von undefinierten Daten

Verlassen Sie sich nicht auf spezielle Werte wie NaN, Inf oder leere Arrays, um zu ermitteln, ob ein VI undefinierte Daten produziert. Vergewissern Sie sich stattdessen, dass das VI definierte Daten produziert, indem Sie veranlassen, dass das VI einen Fehler zurückgibt, wenn eine Situation eintritt, in der wahrscheinlich undefinierte Daten produziert werden.

Wenn Sie beispielsweise ein VI erstellen, das ein eingehendes Array für die Auto-Indizierung einer For-Schleife verwendet, legen Sie fest, wie das VI weiter verfahren soll, wenn das Eingabe-Array leer ist. Lassen Sie entweder einen Ausgabefehlercode produzieren, oder liefern Sie ersatzweise definierte Daten für die Werte, welche die Schleife erstellt.

Fehlerprüfung und Fehlerbehandlung

Die Fehlersuche (Debugging), die in LabVIEW automatisch aktiviert ist, prüft die Blockdiagrammknoten auf Fehler. Sie können die Debugging-Einstellung prüfen, indem Sie **Datei»VI-Einstellungen** und anschließend **Ausführung** aus dem Pulldown-Menü **Kategorie** wählen. Das Kontrollkästchen **Debugging aktiviert** ist standardmäßig aktiviert. Jeder Fehler verfügt über einen numerischen Code und eine entsprechende Fehlermeldung. Verwenden Sie die LabVIEW-Fehlerbehandlungs-VIs, -Funktionen und -Parameter, um Fehler zu bearbeiten. Wenn LabVIEW beispielsweise einen Fehler feststellt, können Sie ein Dialogfeld mit einer Fehlermeldung anzeigen. Verwenden Sie die Fehlerbehandlung in Verbindung mit den Fehlersuche-Werkzeugen, um Fehler zu suchen und zu bearbeiten. National Instruments empfiehlt dringend die Verwendung der Fehlerbehandlung.

Prüfen auf Fehler

Ganz gleich, wie viel Vertrauen Sie in die von Ihnen erstellten VIs setzen, Sie können nicht jedes Problem vorhersehen, auf das ein Benutzer möglicherweise stößt. Ohne einen Mechanismus zur Fehlerüberprüfung wissen Sie nur, dass das VI nicht ordnungsgemäß funktioniert. Die Fehlerprüfung zeigt Ihnen, warum und wo Fehler auftreten.

Wenn Sie irgendeine Art der Ein- oder Ausgabe (I/O) durchführen, ziehen Sie die Möglichkeit in Betracht, dass Fehler auftreten können. Beinahe alle I/O-Funktionen geben Fehlerinformationen zurück. Versehen Sie VIs mit Fehlerprüfmechanismen, insbesondere bei I/O-Operationen (Datei, seriell,

Instrumentierung, Datenerfassung und Kommunikation), und stellen Sie ein Verfahren für eine geeignete Fehlerbehandlung bereit.

Die Fehlerprüfung in VIs kann Ihnen helfen, die folgenden Probleme zu erkennen:

- Sie haben Kommunikationskanäle falsch initialisiert oder haben fehlerhafte Daten an ein externes Gerät geschrieben.
- Ein externes Gerät ist stromlos, defekt oder arbeitet nicht korrekt.
- Sie haben die Betriebssystemsoftware aktualisiert, wodurch der Pfad zu einer Datei oder die Funktionalität eines VIs oder einer Bibliothek geändert wurde beziehungsweise beeinträchtigt wird. Möglicherweise stellen Sie ein Problem in einem VI oder einem Systemprogramm fest.

Fehlerbehandlung

LabVIEW bearbeitet Fehler nicht automatisch. Stattdessen müssen Sie die zu verwendende Fehlerbehandlungsmethode wählen. Wenn bei einem I/O-VI in einem Blockdiagramm beispielsweise eine Zeitüberschreitung auftritt, möchten Sie möglicherweise nicht, dass die gesamte Applikation gestoppt wird. Darüber hinaus möchten Sie vielleicht veranlassen, dass das VI für einen bestimmten Zeitraum erneute Versuche unternimmt. In LabVIEW können Sie diese Fehlerbehandlungsentscheidungen im Blockdiagramm des VIs treffen.

VIs und Funktionen geben Fehler in einer von zwei möglichen Weisen zurück, und zwar mit numerischen Fehlercodes oder mit einem Fehler-Cluster. Normalerweise arbeiten Funktionen mit numerischen Fehlercodes, und VIs verwenden einen Fehler-Cluster, im Normalfall mit Fehlerein- und ausgängen. Weitere Informationen zu Fehler-Clustern finden Sie in Abschnitt *Fehler-Cluster* dieses Kapitels.

Die Fehlerbehandlung in LabVIEW folgt dem Datenflussmodell. Ebenso wie die Daten durch ein VI fließen, können auch Fehlerinformationen fließen. Verbinden Sie die Fehlerinformationen vom Anfang des VIs bis zu dessen Ende. Fügen Sie am Ende des VIs ein Fehlerbehandler-VI ein, um zu ermitteln, ob das VI fehlerlos ausgeführt werden konnte. Verwenden Sie die **Fehlereingangs-** und **Fehlerausgangs-**Cluster in jedem von Ihnen verwendeten oder erstellten VI, um Fehlerinformationen durch das VI zu leiten.

Wenn das VI ausgeführt wird, prüft LabVIEW an jedem Ausführungsknoten auf Fehler. Wenn LabVIEW keine Fehler findet, wird der Knoten normal ausgeführt. Wenn LabVIEW jedoch einen Fehler findet, übergibt der Knoten den Fehler ohne Ausführung an den nächsten Knoten. Der

nächste Knoten geht ebenso vor und so weiter. Am Ende des Ausführungsflusses meldet LabVIEW den Fehler zurück.

Fehler-Cluster

Zu den Fehler-Clustern, die sich auf der Palette **Elemente»Array & Cluster** befinden, gehören die folgenden Informationskomponenten:

- **Status** ist ein boolescher Wert, der TRUE (WAHR) zurückgibt, wenn ein Fehler aufgetreten ist.
- **Code** ist eine vorzeichenbehaftete 32-Bit-Ganzzahl, die den Fehler numerisch kennzeichnet. Ein Fehlercode, der nicht Null ist, gekoppelt mit dem **Status** FALSE (FALSCH) signalisiert eine Warnung und keinen schwerwiegenden Fehler.
- **Quelle** ist ein String, der angibt, wo der Fehler aufgetreten ist.

Verwenden von While-Schleifen für die Fehlerbehandlung

Sie können einen Fehler-Cluster mit dem Bedingungsanschluss einer While-Schleife verbinden, um den Durchlauf der While-Schleife zu stoppen. Wenn Sie den Fehler-Cluster mit dem Bedingungsanschluss verbinden, wird nur der Wert TRUE oder FALSE des Parameters **Status** des Fehler-Clusters an den Anschluss übergeben. Wenn ein Fehler auftritt, wird die While-Schleife gestoppt.

Wenn ein Fehler-Cluster mit dem Bedingungsanschluss verbunden ist, ändern sich die Kontextmenüelemente **Stopp wenn TRUE** und **Weiter wenn TRUE** zu **Bei Fehler stoppen** und **Weiter solange Fehler**.

Verwenden von Case-Strukturen für die Fehlerbehandlung

Wenn Sie einen Fehler-Cluster mit dem Selektoranschluss einer Case-Struktur verbinden, zeigt die Case-Selektor-Kennung zwei Cases an, **Fehler** und **Kein Fehler**, und der Rahmen der Case-Struktur ändert die Farbe, Rot für **Fehler** und Grün für **Kein Fehler**. Wenn ein Fehler auftritt, führt die Case-Struktur das geeignete Case-Subdiagramm aus. Weitere Informationen zum Einsatz von Case-Strukturen finden Sie in Abschnitt *Case-Strukturen* des Kapitels 8, *Schleifen und Case-Strukturen*.

Erstellen von VIs und Sub-VIs

Nachdem Sie nun erfahren haben, wie ein Frontpanel und ein Blockdiagramm erstellt wird, können Sie eigene VIs und Sub-VIs erstellen und können VIs, eigenständige Applikationen und gemeinsam genutzte Bibliotheken für andere Benutzer bereitstellen.

Weitere Informationen zum Planen eines Projekts einschließlich Informationen zu häufig auftretenden Problemstellungen bei der Entwicklung sowie zu den Werkzeugen, die Sie zum Entwickeln eines Projekts verwenden können, finden Sie im Handbuch *LabVIEW Development Guidelines*.

Weitere Informationen ...

Weitere Informationen zum Erstellen und Verwenden von Sub-VIs, zum Speichern von VIs und zum Erstellen von eigenständigen Applikationen und gemeinsamen Bibliotheken finden Sie in der *LabVIEW-Hilfe*.

Planen und Entwerfen eines Projekts

Bevor Sie eigene VIs entwickeln, erstellen Sie eine Liste der Aufgaben, welche die Benutzer ausführen müssen. Legen Sie die Benutzerschnittstellenkomponenten und die Anzahl und Art der Bedien- und Anzeigeelemente fest, die Sie für die Datenanalyse, das Anzeigen der Analyseergebnisse und so weiter benötigen. Besprechen Sie mit den zukünftigen Benutzern oder anderen Mitgliedern des Projektteams, wie und wann der Benutzer auf die Funktionen und Leistungsmerkmale zugreifen können muss. Erstellen Sie beispielhafte Frontpanels, und zeigen Sie diese den zukünftigen Benutzern oder den Mitgliedern des Projektteams, um festzustellen, ob das Frontpanel den Benutzern dabei hilft, ihre Aufgaben zu erledigen. Nutzen Sie diesen interaktiven Prozess, um die Benutzeroberfläche gegebenenfalls zu verfeinern.

Teilen Sie die Applikation in verwaltbare Teile an logischen Plätzen auf. Beginnen Sie mit einem High-Level-Blockdiagramm, das die Hauptkomponenten der Applikation umfasst. Beispielsweise könnte ein Blockdiagramm einen Block für die Konfiguration, einen Block für die Datenerfassung, einen Block für die Analyse der erfassten Daten, einen

Block für die Anzeige der Analyseergebnisse, einen Block für das Speichern der Daten auf Datenträger und einen Block für die Fehlerbehandlung enthalten.

Nachdem Sie das High-Level-Blockdiagramm entworfen haben, definieren Sie die Ein- und Ausgaben. Gestalten Sie dann die Sub-VIs, aus denen die Hauptkomponenten des High-Level-Blockdiagramms bestehen. Die Verwendung von Sub-VIs vereinfacht das Lesen, Debuggen, Verstehen und Pflegen des High-Level-Blockdiagramms. Sie können auch Sub-VIs für allgemeine oder häufige Operationen erstellen, die Sie wieder verwenden können. Testen Sie Sub-VIs nach der Erstellung. Sie können zwar auch Testroutinen für die höhere Ebene erstellen, jedoch ist das Auffinden von Fehlern in kleinen Modulen einfacher als das Testen einer Hierarchie aus mehreren VIs. Möglicherweise stellen Sie auch fest, dass der anfängliche Entwurf des High-Level-Blockdiagramms unvollständig ist. Die Verwendung von Sub-VIs für das Ausführen von Einzelaufgaben vereinfacht das Ändern oder Neuordnen der Applikation. Weitere Informationen zu Sub-VIs finden Sie in Abschnitt *Sub-VIs* dieses Kapitels.

Wählen Sie **Hilfe»Beispiele**, um Beispiele für Blockdiagramme und Sub-VIs anzuzeigen.

Entwerfen von Projekten mit mehreren Entwicklern

Wenn mehrere Entwickler am gleichen Projekt arbeiten, müssen im Vorfeld die Verantwortlichkeiten für die Programmierung, die Schnittstellen sowie Codestandards definiert werden, um einen reibungslosen Ablauf des Entwicklungsprozesses und ein problemloses Funktionieren der Applikation sicherzustellen.

Speichern Sie Masterkopien der Projekt-VIs auf einem einzigen Computer, und führen Sie Richtlinien für die Quellcodekontrolle ein. Ziehen Sie die Verwendung des LabVIEW Professional Development Systems in Betracht, welches Quellcode-Kontrollwerkzeuge enthält, welche die gemeinsame Nutzung von Dateien vereinfachen. Zu diesen Werkzeugen gehört auch ein Dienstprogramm zum Vergleichen von VIs und zum Anzeigen der Änderungen, die zwischen unterschiedlichen Versionen von VIs vorgenommen wurden. Weitere Informationen zum Einsatz der Quellcodekontrolle finden Sie in Abschnitt *Source Code Control* von Kapitel 2, *Incorporating Quality into the Development Process*, in den *LabVIEW Development Guidelines*.

Verwenden der integrierten VIs und Funktionen

LabVIEW enthält VIs und Funktionen, die Sie beim Erstellen spezifischer Applikationen unterstützen, wie beispielsweise Datenerfassungs-VIs und -Funktionen, VIs, die auf andere VIs zugreifen, sowie VIs, die mit anderen Applikationen kommunizieren. Diese VIs können Sie als Sub-VIs in Ihrer Applikation verwenden, um Entwicklungszeit einzusparen. Weitere Informationen zu Sub-VIs finden Sie in Abschnitt [Sub-VIs](#) dieses Kapitels.

Erstellen von Instrumentensteuerungs- und Datenerfassungs-VIs und -Funktionen

Sie können die integrierten VIs und Funktionen verwenden, um externe Instrumente wie beispielsweise Oszilloskope zu steuern, und um Daten zu erfassen, wie beispielsweise die Ablesewerte eines Thermoelements.

Verwenden Sie die Instrumenten-I/O-VIs und -Funktionen, die sich auf der Palette **Funktionen»Instrumenten-I/O** befinden, um externe Instrumente zu steuern. Zum Steuern von Instrumenten mit LabVIEW muss die geeignete Hardware installiert und eingeschaltet sein und an Ihrem Computer betrieben werden können. Die VIs und Funktionen, die Sie zum Steuern von Instrumenten verwenden, sind abhängig von den Kommunikationsprotokollen für die Instrumentierung, die seitens Ihrer Hardware unterstützt werden. Weitere Informationen zum Erstellen von VIs zum Steuern von Instrumenten finden Sie im *LabVIEW Measurements Manual*.

Verwenden Sie die Datenerfassungs-VIs und -Funktionen, die sich auf der Palette **Funktionen»Datenerfassung** befinden, um Daten von DAQ-Geräten zu erfassen. Um diese VIs verwenden zu können, muss die NI-DAQ-Treibersoftware sowie DAQ-Hardware installiert sein. Weitere Informationen zum Installieren der NI-DAQ-Treibersoftware und von DAQ-Hardware sowie zum Erstellen von VIs für die Datenerfassung finden Sie im *LabVIEW Measurements Manual*. Nachdem Sie die gewünschten Daten erfasst haben, können Sie die integrierten Analyse-, Berichterstellungs- und Mathematik-VIs und -Funktionen verwenden, um die Daten zu analysieren, um Berichte zu erstellen und um mathematische Operationen mit diesen Daten vorzunehmen.

Erstellen von VIs, die auf andere VIs zugreifen

Verwenden Sie die Applikationssteuerungs-VIs und -Funktionen, die sich auf der Palette **Funktionen»Applikationssteuerung** befinden, um zu steuern, wie sich VIs verhalten, wenn sie als Sub-VIs aufgerufen oder vom Benutzer ausgeführt werden. Sie können diese VIs und Funktionen verwenden, um mehrere VIs gleichzeitig zu konfigurieren. Wenn Sie gemeinsam mit anderen LabVIEW-Benutzern in einem Netzwerk arbeiten, können Sie diese VIs und Funktionen auch verwenden, um von einem fernen Standort aus auf VIs zuzugreifen und diese zu steuern. Weitere Informationen zum Steuern von VIs von fernen Standorten aus finden Sie in Kapitel 16, *Programmatische Steuerung von VIs*.

Erstellen von VIs, die mit anderen Applikationen kommunizieren

Verwenden Sie die Datei-I/O-VIs und -Funktionen, die sich auf der Palette **Funktionen»Datei-I/O** befinden, um Daten an andere Applikationen wie beispielsweise Microsoft Excel zu schreiben und hieraus zu lesen. Sie können diese VIs und Funktionen verwenden, um Berichte zu erstellen oder um Daten aus einer anderen Applikation in das VI aufzunehmen. Weitere Informationen zum Übergeben von Daten an und zum Übernehmen von Daten aus anderen Applikationen finden Sie in Kapitel 13, *Datei-I/O*.

Verwenden Sie die Kommunikations-VIs und -Funktionen, die sich auf der Palette **Funktionen»Kommunikation** befinden, um LabVIEW-Daten mit Hilfe eines Kommunikationsprotokolls wie beispielsweise FTP über das Web zu übertragen und um Client-/Server-Applikationen zu erstellen, die Kommunikationsprotokolle verwenden. Weitere Informationen zur Kommunikation mit anderen Applikationen im Netzwerk oder im Web finden Sie in Kapitel 17, *Arbeiten mit LabVIEW im Netzwerk*.

(Windows) Verwenden Sie die ActiveX-Funktionen, die sich auf der Palette **Funktionen»Kommunikation»ActiveX** befinden, um VIs ActiveX-Objekte hinzuzufügen oder um ActiveX-fähige Applikationen zu steuern. Weitere Informationen zum Einsatz der ActiveX-Techniken finden Sie in Kapitel 18, *ActiveX*.

Sub-VIs

Nachdem Sie ein VI erstellt und dessen Symbol und Anschlussfeld definiert haben, können Sie es in einem anderen VI verwenden. Ein VI, das einem anderen VI untergeordnet ist, wird als Sub-VI bezeichnet. Ein Sub-VI entspricht den Subroutinen in textbasierten Programmiersprachen. Ein Sub-VI-Knoten entspricht dem Aufruf einer Subroutine in textbasierten Programmiersprachen. Der Sub-VI-Knoten ist nicht identisch mit dem Sub-VI an sich, ebenso wenig wie die Aufrufanweisung für eine Subroutine in einem Programm identisch mit der Subroutine selbst ist. Ein Blockdiagramm mit mehreren identischen Sub-VI-Knoten ruft dasselbe Sub-VI mehrere Male auf.

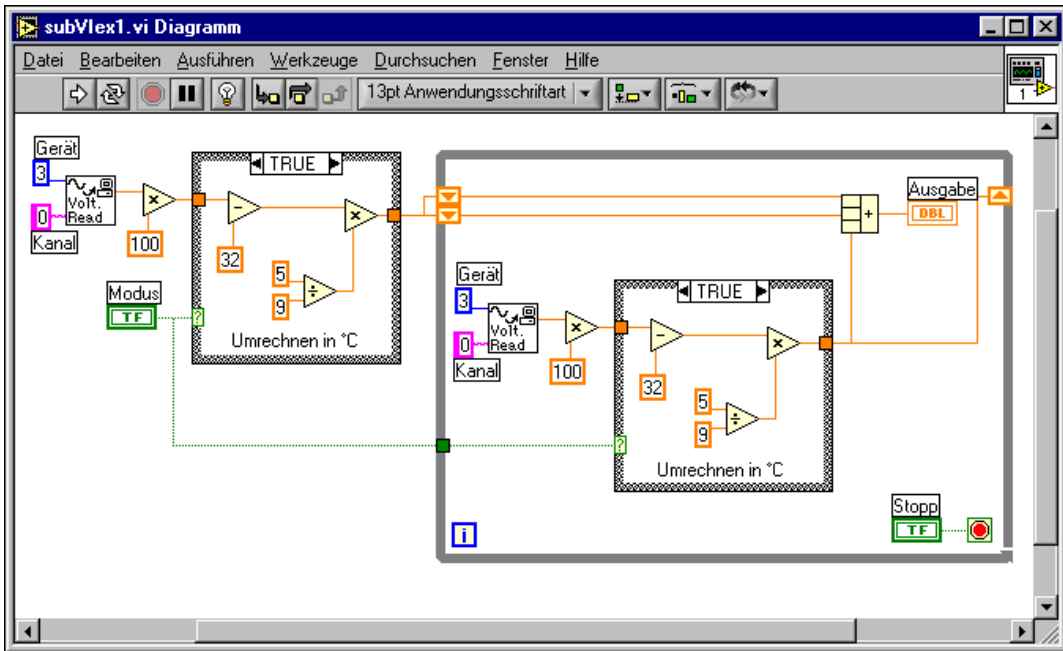
Die Bedien- und Anzeigeelemente eines Sub-VIs empfangen Daten vom und geben Daten an das Blockdiagramm des aufrufenden VIs zurück. Wählen Sie die VIs auf der Palette **Funktionen»Wählen Sie ein VI...**, und platzieren Sie diese im Blockdiagramm, um einen Sub-VI-Aufruf für dieses VI zu erstellen



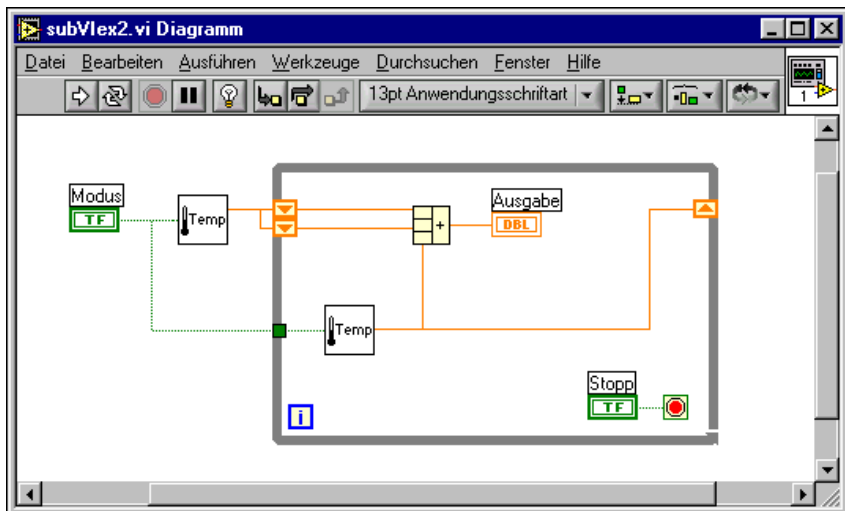
Hinweis Bevor Sie ein VI als Sub-VI verwenden können, müssen Sie ein Anschlussfeld einrichten. Weitere Informationen zum Einrichten eines Anschlussfeldes finden Sie in Abschnitt *Einrichten des Anschlussfeldes* dieses Kapitels.

Identifizieren von häufig wiederholten Operationen

Beim Erstellen von VIs stellen Sie möglicherweise fest, dass eine bestimmte Operation häufig ausgeführt wird. Ziehen Sie in diesem Fall den Einsatz von Sub-VIs oder Schleifen in Betracht, um diese Operation wiederholt auszuführen. Beispielsweise enthält das nachstehende Blockdiagramm zwei identische Operationen.



Sie können nun ein Sub-VI erstellen, das diese Operation ausführt, und dieses Sub-VI zweimal aufrufen, wie im folgenden Blockdiagramm gezeigt.



Sie können das Sub-VI auch in anderen VIs wieder verwenden. Weitere Informationen zum Einsatz von Schleifen zum Kombinieren von häufig auszuführenden Operationen finden Sie in Kapitel 8, [Schleifen und Case-Strukturen](#).

Einrichten des Anschlussfeldes



Um ein VI als Sub-VI verwenden zu können, müssen Sie ein Anschlussfeld wie links gezeigt erstellen. Das Anschlussfeld ist eine Sammlung von Anschlüssen, die den Bedien- und Anzeigeelementen dieses VIs entsprechen, ähnlich der Parameterliste eines Funktionsaufrufs in textbasierten Programmiersprachen. Mit dem Anschlussfeld werden die Eingänge und Ausgänge definiert, die Sie mit dem VI verbinden möchten, damit Sie es als Sub-VI einsetzen können. Weitere Informationen zu Anschlussfeldern finden Sie in Abschnitt [Symbol- und Anschlussfeld](#) des Kapitels 2, [Einführung in Virtuelle Instrumente](#).

Sie definieren Verbindungen, indem Sie jedem Anschluss im Anschlussfeld ein Frontpanel-Bedien- oder Anzeigeelement zuweisen. Zum Definieren eines Anschlussfeldes klicken Sie mit der rechten Maustaste auf das Symbol in der oberen rechten Ecke des Frontpanel-Fensters und wählen aus dem Kontextmenü **Anschluss anzeigen**. Das Symbol wird nun durch das Anschlussfeld ersetzt. Jedes Rechteck im Anschlussfeld entspricht einem Anschluss. Verwenden Sie die Rechtecke, um Eingänge und Ausgänge zuzuweisen. Die Anzahl der von LabVIEW im Anschlussfeld angezeigten Anschlüsse ist abhängig von der Anzahl der Bedien- und Anzeigeelemente auf dem Frontpanel.

Ein Anschlussfeld kann über maximal 28 Anschlüsse verfügen. Wenn das Frontpanel mehr als 28 Bedien- und Anzeigeelemente enthält, die programmgesteuert verwendet werden sollen, fassen Sie einige hiervon als Cluster zusammen, und weisen Sie diesem Cluster einen Anschluss des Anschlussfeldes zu. Weitere Informationen zum Gruppieren von Daten mit Hilfe von Clustern finden Sie in Abschnitt [Cluster](#) des Kapitels 9, [Gruppieren von Daten mit Hilfe von Strings, Arrays und Clustern](#).



Hinweis Versuchen Sie nicht, einem VI mehr als 16 Anschlüsse zuzuweisen. Die Übersichtlichkeit und Brauchbarkeit des VIs wird durch zu viele Anschlüsse beeinträchtigt.

Sie wählen ein anderes Anschlussmuster für ein VI, indem Sie mit der rechten Maustaste auf das Anschlussfeld klicken und aus dem Kontextmenü **Anordnung der Anschlüsse** wählen. Wählen Sie ein Anschlussfeldmuster mit zusätzlichen Anschlüssen. Sie können die

zusätzlichen Anschlüsse unverbunden lassen, bis Sie diese benötigen. Aufgrund dieser Flexibilität sind Sie in der Lage, Änderungen mit minimalen Auswirkungen auf die Hierarchie der VIs vorzunehmen.

Wenn Sie eine Gruppe aus Sub-VIs erstellen, die Sie häufig zusammen einsetzen, weisen Sie den Sub-VIs ein konsistentes Anschlussfeld mit allgemeinen Eingängen an der gleichen Position zu, damit Sie sich besser daran erinnern, wo jeder Eingang zu finden ist. Wenn Sie ein Sub-VI erstellen, das eine Ausgabe produziert, die ein anderes Sub-VI als Eingabe verwendet, richten Sie die Eingabe- und Ausgabeverbindungen aus, um die Verbindungsmuster zu vereinfachen. Setzen Sie die Cluster **Fehlereingang** in die untere linke Ecke des Frontpanels und die Cluster **Fehlerausgang** in die untere rechte Ecke.

Abbildung 7-1 enthält Beispiele für ordnungsgemäß und nicht ordnungsgemäß ausgerichtete Fehler-Cluster.

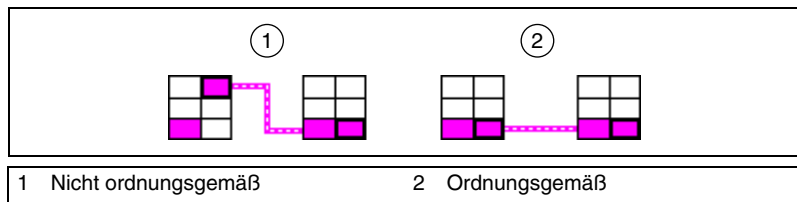


Abbildung 7-1. Ordnungsgemäß und nicht ordnungsgemäß ausgerichtete Fehler-Cluster

Festlegen von erforderlichen, empfohlenen und optionalen Ein- und Ausgängen

Sie können festlegen, welche Ein- und Ausgänge erforderlich, empfehlenswert oder optional sind, um zu verhindern, dass der Benutzer vergisst, Sub-VI-Verbindungen herzustellen.

Klicken Sie mit der rechten Maustaste auf einen Anschluss im Anschlussfeld, und wählen Sie aus dem Kontextmenü **Diese Verbindung ist**. Die aktuelle Anschlusseinstellung ist mit einem Häkchen gekennzeichnet. Wählen Sie **Erforderlich**, **Empfohlen** oder **Optional**.

Wenn ein Ein- oder Ausgang erforderlich ist, kann das VI nicht als Sub-VI ausgeführt werden, ohne dass es ordnungsgemäß verbunden ist. Wenn ein Ein- oder Ausgang empfohlen wird, können Sie das VI zwar ausführen, LabVIEW zeigt jedoch im Fenster **Fenster»Fehlerliste** eine Warnung an, wenn Sie das Kontrollkästchen **Warnungen anzeigen** im Fenster **Fehlerliste** aktiviert haben. Für unverbundene optionale Ein- und

Ausgänge verwendet LabVIEW den Standardwert und gibt keine Warnungen zurück.

Ein- und Ausgänge von VIs in `vi.lib` sind bereits als **Erforderlich**, **Empfohlen** oder **Optional** gekennzeichnet. LabVIEW setzt die Ein- und Ausgänge von VIs, die Sie erstellen, standardmäßig auf **Empfohlen**. Setzen Sie die Einstellung für einen Anschluss nur dann auf „Erforderlich“, wenn das VI über einen Ein- oder Ausgang verfügen muss, um ordnungsgemäß zu funktionieren.

Im Fenster **Kontext-Hilfe** werden erforderliche Verbindungen in Fettdruck, empfohlene Verbindungen als normaler Text und optionale Verbindungen abgeblendet dargestellt, wenn Sie die Ansicht **Ausführliche Kontext-Hilfe** gewählt haben, oder werden gar nicht angezeigt, wenn Sie die Ansicht **Einfache Kontext-Hilfe** gewählt haben.

Erstellen eines Symbols



Jedes VI verfügt über ein Symbol (siehe links), das in der oberen rechten Ecke der Frontpanel- und Blockdiagrammfenster angezeigt wird. Das Symbol ist eine grafische Darstellung eines VIs. Es kann Text, Grafiken oder eine Kombination hieraus enthalten. Wenn Sie ein VI als Sub-VI verwenden, kennzeichnet das Symbol das Sub-VI im Blockdiagramm des VIs.

Das Standardsymbol enthält eine Zahl, die angibt, wie viele neue VIs seit dem Start von LabVIEW geöffnet wurden. Sie erstellen benutzerdefinierte Symbole als Ersatz für das Standardsymbol, indem Sie mit der rechten Maustaste auf das Symbol in der oberen rechten Ecke des Frontpanels oder des Blockdiagramms klicken und aus dem Kontextmenü **Symbol bearbeiten** wählen, oder indem Sie auf das Symbol in der oberen rechten Ecke des Frontpanels doppelklicken.

Abhängig vom verwendeten Monitortyp können Sie jeweils ein separates Symbol für die Modi Schwarz/Weiß, 16 Farben oder 256 Farben erstellen. LabVIEW verwendet beim Drucken einfarbige Symbole, sofern Sie nicht über einen Farbdrucker verfügen.

Erstellen von Sub-VIs aus Teilen eines VIs

Sie wandeln einen Teil eines VIs in ein Sub-VI um, indem Sie den Abschnitt des Blockdiagramms, der wieder verwendet werden soll, mit dem Positionierwerkzeug markieren und dann **Bearbeiten»Sub-VI erstellen** wählen. Der markierte Abschnitt des Blockdiagramms wird durch ein Symbol für das neue Sub-VI ersetzt. LabVIEW erstellt Bedien- und Anzeigeelemente für das neue Sub-VI und verbindet das Sub-VI mit den vorhandenen Verbindungen.

Das Erstellen eines Sub-VIs aus einer Markierung ist zwar praktisch, erfordert jedoch auch eine durchdachte Planung, um eine logische Hierarchie der VIs zu erstellen. Überlegen Sie, welche Objekte in die Markierung aufgenommen werden sollen, und sehen Sie davon ab, die Funktionalität des hieraus resultierenden VIs zu ändern.

Entwerfen von Sub-VIs

Wenn die Benutzer das Frontpanel eines Sub-VIs nicht anzeigen müssen, können Sie bei dessen Gestaltung, beispielsweise mit Farben und Schriftarten, Zeit sparen. Dennoch ist der Aufbau des Frontpanels von Bedeutung, da Sie das Frontpanel möglicherweise bei der Fehlersuche im VI anzeigen müssen.

Platzieren Sie die Bedien- und Anzeigeelemente so, wie sie im Anschlussfeld erscheinen. Setzen Sie Bedienelemente auf die linke und Anzeigeelemente auf die rechte Seite des Frontpanels. Setzen Sie die Cluster **Fehlereingang** in die untere linke Ecke des Frontpanels und die Cluster **Fehlerausgang** in die untere rechte Ecke. Weitere Informationen zum Einrichten eines Anschlussfeldes finden Sie in Abschnitt [Einrichten des Anschlussfeldes](#) dieses Kapitels.

Wenn ein Bedienelement über einen Standardwert verfügt, setzen Sie den Standardwert als Teil des Namens des Bedienelements in Klammern. Fügen Sie dem Bedienelementnamen gegebenenfalls auch die Maßeinheiten hinzu. Wenn ein Bedienelement also beispielsweise die obere Grenztemperatur festlegt und über den Standardwert 20 °C verfügt, nennen Sie das Bedienelement **Obere Grenztemperatur (20 degC)**. Wenn Sie das Sub-VI auf mehreren Plattformen einsetzen möchten, vermeiden Sie Sonderzeichen in Bedienelementnamen. Verwenden Sie also beispielsweise **degC** anstelle von °C.

Benennen Sie boolesche Bedienelemente so, dass die Benutzer erkennen können, welche Funktion das Bedienelement im Status TRUE (WAHR) ausführt. Verwenden Sie Namen wie **Abbrechen**, **Zurücksetzen** und **Initialisieren**, mit denen die Aktion aussagekräftig beschrieben wird.

Anzeigen der VI-Hierarchie

Im **Hierarchiefenster** wird eine grafische Darstellung der Aufrufhierarchie aller VIs im Speicher angezeigt, einschließlich der Typdefinitionen und globalen Variablen. Wählen Sie **Durchsuchen» VI-Hierarchie anzeigen**, um das **Hierarchiefenster** anzuzeigen. Verwenden Sie dieses Fenster, um Sub-VIs und andere Knoten anzuzeigen, aus denen sich das aktuelle VI zusammensetzt.

Wenn Sie das Bedienwerkzeug über Objekte im **Hierarchiefenster** bewegen, zeigt LabVIEW den Namen des jeweiligen VIs an. Sie können ein VI oder ein Sub-VI mit Hilfe des Positionierwerkzeugs aus dem **Hierarchiefenster** in das Blockdiagramm ziehen, um das VI oder Sub-VI als Sub-VI in einem anderen VI zu verwenden. Sie können auch einen oder mehrere Knoten auswählen und in die Zwischenablage kopieren und anschließend in andere Blockdiagramme einfügen. Doppelklicken Sie im **Hierarchiefenster** auf einen VI- oder Sub-VI-Knoten, um das Frontpanel dieses VIs anzuzeigen.

Ein VI, das Sub-VIs enthält, verfügt über eine Pfeilschaltfläche im Rahmen. Klicken Sie auf diese Pfeilschaltfläche, um Sub-VIs ein- oder auszublenden. Eine rote Pfeilschaltfläche wird angezeigt, wenn alle Sub-VIs ausgeblendet sind. Eine schwarze Pfeilschaltfläche wird angezeigt, wenn alle Sub-VIs eingeblendet sind.

Speichern von VIs

Sie können VIs als einzelne Dateien speichern, oder Sie können mehrere VIs gruppieren und diese in einer VI-Bibliothek speichern. VI-Bibliotheksd Dateien verfügen über die Dateinamenerweiterung `.lib`. National Instruments empfiehlt, VIs als verzeichnisweise organisierte Einzeldateien zu speichern, insbesondere, wenn mehrere Entwickler am gleichen Projekt arbeiten.

Vorteile des Speicherns von VIs als Einzeldateien

In der folgenden Liste werden die Gründe für das Speichern von VIs als Einzeldateien aufgeführt:

- Sie können das Dateisystem zum Verwalten der Einzeldateien verwenden.
- Sie können mit Unterverzeichnissen arbeiten.
- VIs und Bedienelemente lassen sich in einzelnen Dateien stabiler speichern als ein gesamtes Projekt in einer Einzeldatei.
- Sie können die integrierten Quellcode-Kontrollwerkzeuge des Professional Development Systems oder Quellcode-Kontrollwerkzeuge von Fremdanbietern verwenden.

Vorteile des Speicherns von VIs als Bibliotheken

In der folgenden Liste werden die Gründe für das Speichern von VIs als Bibliotheken aufgeführt:

- Sie können bis zu 255 Zeichen zum Benennen von Dateien verwenden. **(Mactintosh)** Unter MacOS 9.x oder niedriger sind Dateinamen auf 31 Zeichen begrenzt.
- Sie können eine VI-Bibliothek einfacher auf andere Plattformen portieren als mehrere Einzel-VIs.
- Sie können die Dateigröße Ihres Projekts geringfügig verringern, da VI-Bibliotheken komprimiert werden, um die Anforderungen an den Plattenspeicher zu reduzieren.



Hinweis LabVIEW speichert viele der integrierten VIs und Beispiele in VI-Bibliotheken, um konsistente Speicherpositionen auf allen Plattformen zu gewährleisten.

Wenn Sie mit VI-Bibliotheken arbeiten, ziehen Sie in Betracht, Ihre Applikation in mehrere VI-Bibliotheken aufzuteilen. Setzen Sie die High-Level-VIs in eine VI-Bibliothek, und richten Sie weitere Bibliotheken ein, die VIs nach Funktionen getrennt enthalten.

Verwalten von VIs in Bibliotheken

Verwenden Sie den VI Dateimanager, auf den Sie durch Auswahl von **Werkzeuge»Verwaltung von VI-Bibliotheken** zugreifen, um das Kopieren, Umbenennen und Löschen von Dateien in VI-Bibliotheken zu vereinfachen. Mit diesem Werkzeug können Sie auch neue VI-Bibliotheken und -Verzeichnisse erstellen und VI-Bibliotheken in Verzeichnisse umwandeln und umgekehrt. Das Erstellen von neuen VI-Bibliotheken und -Verzeichnissen und das Umwandeln von VI-Bibliotheken in Verzeichnisse und umgekehrt ist wichtig, wenn Sie VIs mit Quellcode-Kontrollwerkzeugen verwalten müssen.

Bevor Sie den VI Dateimanager aufrufen, schließen Sie alle VIs, die möglicherweise betroffen sind, um zu vermeiden, dass eine Dateioperation mit einem VI vorgenommen wird, das sich bereits im Speicher befindet.

Benennen von VIs

Verwenden Sie beim Speichern von VIs aussagekräftige Namen. Beschreibende Namen wie `Temperaturüberwachung.vi` und `Serielles Lesen & Schreiben.vi` vereinfachen die Identifikation eines VIs und geben Auskunft, wie es zu verwenden ist. Wenn Sie zweideutige Namen wie beispielsweise `VI#1.vi` verwenden, werden Sie möglicherweise Schwierigkeiten beim Identifizieren von VIs bekommen, insbesondere dann, wenn Sie mehrere VIs gespeichert haben.

Überlegen Sie, ob die Benutzer die VIs auf einer anderen Plattform ausführen werden. Vermeiden Sie Zeichen, die bei einigen Betriebssystemen für spezielle Zwecke vorbehalten sind, wie beispielsweise `\`, `:`, `/`, `?`, `*`, `<`, `>` und `#`.

(Macintosh) Sorgen Sie dafür, dass VI-Namen weniger als 31 Zeichen aufweisen, wenn die Benutzer möglicherweise unter MacOS 9.x oder niedriger arbeiten.

Speichern für eine Vorläuferversion

Sie können VIs für Vorläuferversionen von LabVIEW speichern, um das Aktualisieren von LabVIEW komfortabel zu gestalten und um es zu ermöglichen, die VIs gegebenenfalls in zwei Versionen von LabVIEW zu pflegen. Wenn Sie auf eine neue Version von LabVIEW aktualisieren, können Sie auf die letzte Version der VIs zurückgreifen.

Wenn Sie ein VI für eine Vorläuferversion speichern, wandelt LabVIEW nicht nur dieses VI um, sondern auch alle VIs in dessen Hierarchie mit Ausnahme der `vi.lib`-Dateien.

Oftmals nutzt ein VI Funktionen, die in der Vorläuferversion von LabVIEW nicht verfügbar waren. In solchen Fällen speichert LabVIEW so viel von dem VI wie möglich und erstellt einen Bericht der Komponenten, die nicht umgewandelt werden konnten. Dieser Bericht wird unverzüglich im Dialogfeld **Warnungen** angezeigt. Klicken Sie auf **OK**, um diese Warnungen zu bestätigen und das Dialogfeld zu schließen. Klicken Sie auf **Speichern**, um die Warnungen für eine spätere Prüfung in einer Textdatei zu speichern.

Bereitstellen von VIs

Wenn Sie VIs auf anderen Computern oder für andere Benutzer bereitstellen möchten, überlegen Sie, ob Sie auch benutzerseitig bearbeitbaren Blockdiagramm-Quellcode beifügen möchten oder ob Sie das Blockdiagramm ausblenden oder entfernen möchten. Wählen Sie **Datei»Mit Optionen speichern**, um VIs ohne die Blockdiagramme zu speichern und so die Dateigröße zu reduzieren, und um zu verhindern, dass Benutzer den Quellcode ändern. Wenn Sie ein VI ohne Blockdiagramm speichern, verhindern Sie darüber hinaus, dass Benutzer das VI auf eine andere Plattform übertragen oder dass sie das VI auf eine zukünftige Version von LabVIEW aktualisieren.



Vorsicht! Wenn Sie VIs ohne Blockdiagramme speichern, stellen Sie sicher, dass Sie die Originalversion der VIs nicht überschreiben. Speichern Sie die VIs in anderen Verzeichnissen, oder verwenden Sie unterschiedliche Namen.

Anstatt das Blockdiagramm zu entfernen, können Sie Blockdiagramme auch mit einem Passwort schützen. In diesem Fall ist das Blockdiagramm weiterhin verfügbar, der Benutzer muss jedoch ein Passwort eingeben, damit er das Blockdiagramm anzeigen oder bearbeiten kann.

Eine weitere Option für das Bereitstellen von VIs besteht darin, eine eigenständige Applikation oder eine gemeinsam genutzte Bibliothek zu erstellen. Eine eigenständige Applikation oder eine gemeinsam genutzte Bibliothek ist die richtige Wahl, wenn Sie nicht davon ausgehen, dass die Benutzer die VIs bearbeiten. Die Benutzer können zwar die Applikation oder die gemeinsame Bibliothek nutzen, sie können jedoch die Blockdiagramme nicht bearbeiten oder anzeigen. Eigenständige Applikationen enthalten vereinfachte Menüs.

Erstellen von eigenständigen Applikationen und gemeinsamen Bibliotheken

Wählen Sie **Werkzeuge»Applikation oder „Shared Library“ (DLL) erzeugen**, um mit Hilfe des Application Builders eigenständige Applikationen und Installationsprogramme oder gemeinsame Bibliotheken (Shared Libraries, DLLs) für VIs zu erstellen. Arbeiten Sie mit gemeinsamen Bibliotheken, wenn Sie die VIs in der gemeinsamen Bibliothek unter Verwendung von textbasierten Programmiersprachen aufrufen möchten.



Hinweis Mit dem LabVIEW Professional Development System wird der Application Builder bereitgestellt. Wenn Sie das LabVIEW Base Package oder Full Development System verwenden, können Sie den Application Builder separat erwerben.

Verwenden Sie die Registerkarten des Dialogfeldes **Applikation oder „Shared Library“ (DLL) erzeugen**, um die verschiedenen Einstellungen für die Applikation oder die gemeinsam genutzte Bibliothek zu konfigurieren, die Sie erstellen möchten. Nachdem Sie diese Einstellungen definiert haben, speichern Sie diese in einem Skript, damit Sie die Applikation gegebenenfalls auf einfache Weise erneut erstellen können.

Die Benutzer können nun Ihre Applikation ausführen oder die gemeinsame Bibliothek nutzen, sie können die Blockdiagramme jedoch weder bearbeiten noch anzeigen.

Weitere Informationen zum Installieren des Application Builders finden Sie in den *LabVIEW Application Builder Versionshinweise*.

Erstellen und Bearbeiten von VIs

Dieser Teil beschreibt LabVIEW-Merkmale, -VIs und -Funktionen, mit denen Sie die jeweilige Funktionsweise Ihrer Applikationen festlegen können. Die Kapitel in diesem Abschnitt beschreiben die Einsatzmöglichkeit jedes einzelnen LabVIEW-Merkmals und die einzelnen Klassen von VIs und Funktionen.

Teil II, *Erstellen und Bearbeiten von VIs*, enthält die folgenden Kapitel:

- Kapitel 8, *Schleifen und Case-Strukturen*, beschreibt die Verwendung von Strukturen im Blockdiagramm, um Code-Blöcke zu wiederholen und Code bedingt oder in einer bestimmten Reihenfolge auszuführen.
- Kapitel 9, *Gruppieren von Daten mit Hilfe von Strings, Arrays und Clustern*, beschreibt die Verwendung von Strings, Arrays und Clustern zur Gruppierung von Daten.
- Kapitel 10, *Lokale und globale Variablen*, beschreibt die Verwendung von lokalen und globalen Variablen, um Daten, die nicht durch die übliche Verbindungsmethode transportiert werden können, innerhalb einer Applikation zu übergeben.
- Kapitel 11, *Graphen und Diagramme*, beschreibt die Verwendung von Graphen und Diagrammen zur grafischen Darstellung von Daten.
- Kapitel 12, *VIs für Grafiken und Sound*, beschreibt die Verwendung von Grafiken und Sound in VIs.
- Kapitel 13, *Datei-I/O*, beschreibt das Durchführen von Datei-I/O-Operationen.
- Kapitel 14, *Dokumentieren und Drucken von VIs*, beschreibt das Dokumentieren und Drucken von VIs.
- Kapitel 15, *Anpassen von VIs*, beschreibt das Konfigurieren der Funktionsweise von VIs und Sub-VIs entsprechend den Anforderungen der Applikation.

- Kapitel 16, *Programmatische Steuerung von VIs*, beschreibt die Kommunikation mit VIs und anderen Instanzen von LabVIEW, um VIs und LabVIEW in Programmen zu steuern.
- Kapitel 17, *Arbeiten mit LabVIEW im Netzwerk*, beschreibt die Verwendung von VIs für die Kommunikation oder Vernetzung mit anderen Prozessen, die beispielsweise in anderen Applikationen oder auf Netzwerkrechnern ausgeführt werden.
- Kapitel 18, *ActiveX*, beschreibt das Bereitstellen von öffentlichen Objekten, Befehlen und Funktionen, auf die andere Windows-Applikationen zugreifen können.
- Kapitel 19, *Aufrufen von Code aus textbasierten Programmiersprachen*, beschreibt Code-Aufrufe aus textbasierten Programmiersprachen und die Verwendung von DLLs.
- Kapitel 20, *Formeln und Gleichungen*, beschreibt die Verwendung von Gleichungen in VIs.

Schleifen und Case-Strukturen

Strukturen sind grafische Darstellungen der Schleifen und Case-Anweisungen in textbasierten Programmiersprachen. Verwenden Sie Strukturen im Blockdiagramm, um Codeblöcke zu wiederholen und Code bedingungsabhängig oder in einer bestimmten Reihenfolge auszuführen.

Wie andere Knoten verfügen auch Strukturen über Anschlüsse, über die sie mit anderen Blockdiagrammknoten verbunden werden können, werden automatisch ausgeführt, wenn Eingabedaten verfügbar sind, und liefern Daten an Ausgabeverbindungen, wenn die Ausführung abgeschlossen ist.

Jede Struktur verfügt über einen auffälligen, in der Größe veränderbaren Rahmen, mit dem der Abschnitt des Blockdiagramms umschlossen wird, der entsprechend den Regeln der Struktur ausgeführt wird. Der Abschnitt des Blockdiagramms im Inneren des Strukturrahmens wird als Subdiagramm bezeichnet. Die Anschlüsse, die Daten an Strukturen übergeben beziehungsweise Daten aus Strukturen übernehmen, werden Tunnel genannt. Ein Tunnel ist ein Verbindungspunkt an einem Strukturrahmen.

Weitere Informationen ...

Weitere Informationen zum Verwenden von Strukturen finden Sie in der *LabVIEW-Hilfe*.

Verwenden Sie die folgenden Strukturen, die sich auf der Palette **Funktionen»Strukturen** befinden, um zu steuern, wie ein Blockdiagramm Prozesse ausführt:

- **For-Schleife:** Wiederholt die Ausführung eines Subdiagramms so oft wie vorgegeben.
- **While-Schleife:** Führt ein Subdiagramm aus, bis eine Bedingung zutrifft.
- **Case-Struktur:** Enthält mehrere Subdiagramme, von denen abhängig von dem an die Struktur übergebenen Eingabewert nur eines ausgeführt wird.

- **Sequenzstruktur:** Enthält ein oder mehrere Subdiagramme, die in sequenzieller Reihenfolge ausgeführt werden.
- **Formelknoten:** Führt mathematische Operationen basierend auf der numerischen Eingabe durch. Weitere Informationen zum Einsatz von Formelknoten finden Sie in Abschnitt [Formelknoten](#) des Kapitels 20, [Formeln und Gleichungen](#).

Klicken Sie mit der rechten Maustaste auf den Rahmen der Struktur, um das Kontextmenü aufzurufen.

For-Schleifen- und While-Schleifenstrukturen

Verwenden Sie die For-Schleife und die While-Schleife, um sich wiederholende Operationen zu steuern.

For-Schleifen



Eine For-Schleife wie links gezeigt wiederholt die Ausführung eines Subdiagramms so oft wie vorgegeben.



Der Wert im Zähl-Anschluss (ein Eingabeanschluss, siehe links) gibt an, wie viele Male das Subdiagramm wiederholt werden soll. Sie können die Anzahl explizit einstellen, indem Sie einen Wert von außerhalb der Schleife mit der linken oder der oberen Seite des Zähl-Anschlusses verbinden, oder Sie können die Anzahl implizit mit der Auto-Indizierung festlegen. Weitere Informationen zum impliziten Festlegen der Anzahl finden Sie in Abschnitt [Verwenden der Auto-Indizierung zum Einstellen der For-Schleifenanzahl](#) dieses Kapitels.



Der Iterationsanschluss (ein Ausgabeanschluss, siehe links) enthält die Anzahl der abgeschlossenen Wiederholungen. Die Wiederholungszählung beginnt immer bei Null. Während der ersten Wiederholung gibt der Iterationsanschluss 0 zurück.

Sowohl der Anzahl- als auch der Iterationsanschluss entsprechen dem Datentyp „Vorzeichenbehaftete Ganzzahl vom Typ Long“ mit einem Bereich von 0 bis $2^{31}-1$. Wenn Sie eine Fließkommazahl mit dem Zähl-Anschluss verbinden, rundet LabVIEW diese Zahl und zwingt sie innerhalb des Bereichs. Wenn Sie 0 mit dem Zähl-Anschluss verbinden, wird die Schleife nicht ausgeführt.

Fügen Sie der For-Schleife Schieberegister hinzu, um Daten aus der aktuellen Wiederholung an die nächste Wiederholung zu übergeben. Weitere Informationen zum Hinzufügen von Schieberegistern zu einer Schleife finden Sie in Abschnitt *Schieberegister in Schleifen* dieses Kapitels.

While-Schleifen



Ähnlich einer Do-Schleife oder einer Repeat-Until-Schleife in textbasierten Programmiersprachen führt eine While-Schleife wie links gezeigt ein Subdiagramm aus, bis eine Bedingung zutrifft.



Die While-Schleife führt das Subdiagramm aus, bis der Bedingungsanschluss, ein Eingabeanschluss, einen bestimmten booleschen Wert empfängt. Verhalten und Erscheinungsbild des Bedingungsanschlusses ist standardmäßig auf **Weiter wenn TRUE** (siehe links) eingestellt. Wenn ein Bedingungsanschluss auf **Weiter wenn TRUE** eingestellt ist, führt die While-Schleife das Subdiagramm aus, bis der Bedingungsanschluss den Wert FALSE (FALSCH) erhält. Sie können das Verhalten und das Erscheinungsbild des Bedingungsanschlusses ändern, indem Sie mit der rechten Maustaste auf den Anschluss oder auf den Rahmen der While-Schleife klicken und **Stopp wenn TRUE** (siehe links) wählen. Wenn ein Bedingungsanschluss auf **Stopp wenn TRUE** eingestellt ist, führt die While-Schleife das Subdiagramm aus, bis der Bedingungsanschluss den Wert TRUE (WAHR) erhält. Da das VI den Bedingungsanschluss am Ende jeder Wiederholung prüft, wird die While-Schleife immer mindestens einmal ausgeführt. Das VI wird nicht ausgeführt, wenn Sie den Bedingungsanschluss nicht verbinden.



Über den Bedingungsanschluss einer While-Schleife können Sie auch eine grundlegende Fehlerbehandlung durchführen. Wenn Sie einen Fehler-Cluster mit dem Bedingungsanschluss verbinden, wird nur der Wert TRUE (WAHR) oder FALSE (FALSCH) des Parameters **Status** des Fehler-Clusters an den Anschluss übergeben. Darüber hinaus ändern sich auch die Kontextmenüelemente **Stopp wenn TRUE** und **Weiter wenn TRUE** zu **Bei Fehler stoppen** und **Weiter solange Fehler**. Weitere Informationen zu Fehler-Clustern und die Fehlerbehandlung finden Sie in Abschnitt *Fehlerprüfung und Fehlerbehandlung* des Kapitels 6, *Ausführen von und Fehlersuche in VIs*.



Der Iterationsanschluss (ein Ausgabeanschluss, siehe links) enthält die Anzahl der abgeschlossenen Wiederholungen. Die Wiederholungszählung beginnt immer bei Null. Während der ersten Wiederholung gibt der Iterationsanschluss 0 zurück.

Fügen Sie der While-Schleife Schieberegister hinzu, um Daten aus der aktuellen Wiederholung an die nächste Wiederholung zu übergeben. Weitere Informationen zum Hinzufügen von Schieberegistern zu einer Schleife finden Sie in Abschnitt *Schieberegister in Schleifen* dieses Kapitels.

Vermeiden von endlosen While-Schleifen

Wenn Sie den Anschluss des booleschen Bedienelements außerhalb der While-Schleife platzieren, wie in Abbildung 8-1 gezeigt, und das Bedienelement auf FALSE (FALSCH) und der Bedingungsanschluss beim Start der Schleife auf **Stopp wenn TRUE** eingestellt ist, ergibt sich eine Endlosschleife. Eine Endlosschleife kann auch entstehen, wenn das Bedienelement außerhalb der Schleife auf TRUE und der Bedingungsanschluss auf **Weiter wenn TRUE** gesetzt wurde.

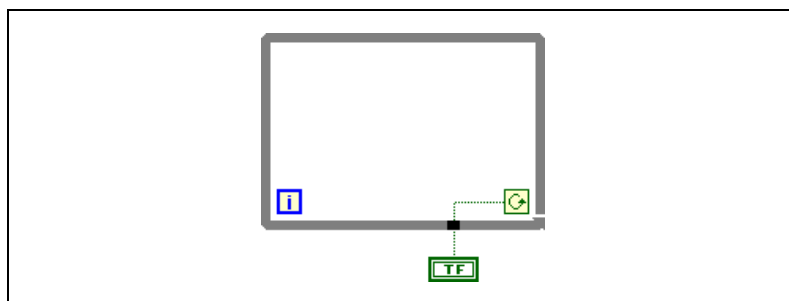


Abbildung 8-1. Endlose While-Schleife

Mit dem Ändern des Wertes des Bedienelements lässt sich die Endlosschleife nicht stoppen, da der Wert nur einmal gelesen wird, bevor die Schleife beginnt. Zum Stoppen einer Endlosschleife müssen Sie die Ausführung des VIs abbrechen, indem Sie auf der Symbolleiste auf die Schaltfläche **Abbrechen** klicken.

Auto-Indizierung von Schleifen

Wenn Sie ein Array mit einer For- oder While-Schleife verbinden, können Sie jedes Element in diesem Array lesen und verarbeiten, indem Sie die Auto-Indizierung aktivieren. Weitere Informationen zu Arrays finden Sie im Kapitel 9, *Gruppieren von Daten mit Hilfe von Strings, Arrays und Clustern*.

Wenn Sie ein Array mit einem Eingabetunnel auf dem Rand der Schleife verbinden und die Auto-Indizierung am Eingabetunnel aktivieren, werden die Elemente des Arrays beginnend mit dem ersten Element nacheinander

an die Schleife übergeben. Klicken Sie mit der rechten Maustaste auf den Rahmen der Schleife, und wählen Sie aus dem Kontextmenü **Indizierung aktivieren** oder **Indizierung deaktivieren**, um die Auto-Indizierung zu aktivieren oder zu deaktivieren. Ist die Auto-Indizierung deaktiviert, wird das gesamte Array an die Schleife übergeben. Die Auto-Indizierung für While-Schleifen ist standardmäßig deaktiviert.

Die Schleife indiziert Skalarelemente aus 1D-Arrays, 1D-Arrays aus 2D-Arrays und so weiter. Das Gegenteil geschieht an Ausgabetunneln. Skalare Elemente werden sequenziell zu 1D-Arrays zusammengefasst, 1D-Arrays zu 2D-Arrays und so weiter.

Verwenden der Auto-Indizierung zum Einstellen der For-Schleifenanzahl

Wenn Sie die Auto-Indizierung für ein Array aktivieren, das mit einer For-Schleife verbunden ist, müssen Sie den Zähl-Anschluss nicht explizit verbinden. Da For-Schleifen verwendet werden können, um Arrays elementweise zu verarbeiten, aktiviert LabVIEW die Auto-Indizierung standardmäßig für jedes Array, das Sie mit einer For-Schleife verbinden. Deaktivieren Sie die Auto-Indizierung, wenn Arrays nicht elementweise verarbeitet werden müssen.

Wenn Sie die Auto-Indizierung für mehr als einen Tunnel aktivieren oder wenn Sie die Anzahl explizit festlegen, erhält die Anzahl den niedrigsten Wert. Wenn beispielsweise zwei auto-indizierte Arrays mit 10 beziehungsweise 20 Elementen an die Schleife übergeben werden und Sie den Zähl-Anschluss mit dem Wert 15 verbinden, wird die Schleife zehnmal ausgeführt, und somit indiziert die Schleife nur die ersten 10 Elemente des zweiten Arrays. Wenn Sie Daten aus zwei Quellen in einem Graphen zeichnen und die ersten 100 Elemente gezeichnet werden sollen, verbinden Sie den Zähl-Anschluss mit dem Wert 100. Wenn eine der Datenquellen nur 50 Elemente umfasst, wird die Schleife 50-mal ausgeführt, und es werden nur die ersten 50 Elemente indiziert. Verwenden Sie die Funktion „Array-Größe“, um die Größe des Arrays festzulegen.

Wenn Sie einen Array-Ausgabetunnel indizieren, empfängt das Ausgabe-Array ein neues Element aus jeder Wiederholung der Schleife. Daher entsprechen auto-indizierte Ausgabe-Arrays in der Größe immer der Anzahl der Wiederholungen. Wenn die Schleife beispielsweise zehnmal ausgeführt wird, weist das Ausgabe-Array 10 Elemente auf. Wenn Sie die Auto-Indizierung für einen Ausgabetunnel deaktivieren, wird nur das Element der letzten Wiederholung der Schleife an den nächsten Knoten im Blockdiagramm übergeben. Die Dicke der Verbindung zwischen dem

Ausgabetunnel und dem nächsten Knoten zeigt an, ob die Schleife mit Auto-Indizierung arbeitet. Mit Auto-Indizierung ist die Verbindung dicker, denn die Verbindung enthält ein Array und keinen Skalar.

Auto-Indizierung mit While-Schleife

Wenn Sie die Auto-Indizierung für ein Array aktivieren, das an eine While-Schleife übergeben wird, indiziert die While-Schleife das Array auf die gleiche Weise wie eine For-Schleife. Allerdings ist die Anzahl der von einer While-Schleife ausgeführten Wiederholungen nicht durch die Größe des Arrays beschränkt, da die While-Schleife wiederholt wird, bis eine bestimmte Bedingung zutrifft. Wenn eine While-Schleife über das Ende des Eingabe-Arrays hinaus indiziert, wird der Standardwert des Elementtyps des Arrays an die Schleife übergeben. Sie können verhindern, dass der Standardwert an die While-Schleife übergeben wird, indem Sie die Funktion „Array-Größe“ verwenden. Die Funktion „Array-Größe“ gibt an, wie viele Elemente das Array beinhaltet. Richten Sie die While-Schleife so ein, dass die Ausführung beendet wird, wenn die Anzahl der Wiederholungen der Größe des Arrays entspricht. Wenn Sie einen Array-Ausgabetunnel auto-indizieren, empfängt das Ausgabe-Array ein neues Element aus jeder Wiederholung der Schleife. Daher entsprechen auto-indizierte Ausgabe-Arrays in der Größe immer der Anzahl der Wiederholungen.



Vorsicht! Da Sie die Größe des Ausgabe-Arrays nicht im Vorfeld bestimmen können, ist das Aktivieren der Auto-Indizierung für die Ausgabe einer For-Schleife effizienter als für die einer While-Schleife. Zu viele Wiederholungen können dazu führen, dass das System Probleme aufgrund von ungenügendem Speicher zurückmeldet. Wenn Sie also nur über begrenzten Speicher verfügen, ziehen Sie die Verwendung einer For-Schleife in Betracht.

Schieberegister in Schleifen

Sie verwenden Schieberegister in Verbindung mit For- und While-Schleifen, um Werte von einer Wiederholung in die nächste zu übernehmen.



Ein Schieberegister erscheint als ein Paar Anschlüsse (siehe links), die einander direkt gegenüberliegend auf den vertikalen Seiten des Schleifenrahmens angeordnet sind. Der rechte Anschluss enthält einen nach oben weisenden Pfeil und speichert die Daten beim Abschluss der Wiederholung. LabVIEW übergibt die mit der rechten Seite des Registers verbundenen Daten in die nächste Wiederholung. Sie erstellen ein Schieberegister, indem Sie mit der rechten Maustaste auf den linken oder rechten Rand einer Schleife klicken und dann aus dem Kontextmenü **Schieberegister hinzufügen** wählen.

Ein Schieberegister überträgt jeden beliebigen Datentyp und stellt sich automatisch auf den Datentyp des ersten Objekts ein, das mit dem Schieberegister verbunden ist. Die Daten, die Sie mit den Anschlüssen des jeweiligen Schieberegisters verbinden, müssen vom gleichen Typ sein. Sie können an einer Struktur mehrere Schieberegister erstellen, und Sie können mehr als einen linken Anschluss verbinden, um auf diese Weise mehr als einen Vorläuferwert zu registrieren.

Nachdem die Schleife ausgeführt wurde, verbleibt der letzte im Schieberegister gespeicherte Wert im rechten Anschluss. Wenn Sie den rechten Anschluss außerhalb der Schleife verbinden, überträgt die Verbindung den letzten im Schieberegister gespeicherten Wert.

Wenn Sie das Register nicht initialisieren, verwendet die Schleife den Wert, der an das Register geschrieben wurde, als die Schleife zuletzt ausgeführt wurde, oder den Standardwert für den Datentyp, wenn die Schleife noch nie ausgeführt wurde.

Verwenden Sie eine Schleife mit einem nicht initialisierten Schieberegister, um ein VI wiederholt so auszuführen, dass bei jeder Ausführung des VIs die Anfangsausgabe des Schieberegisters gleich dem letzten Wert der vorherigen Ausführung ist. Lassen Sie den Eingang des linken Schieberegisteranschlusses bei einem nicht initialisierten Schieberegister unverbunden, um die Statusinformationen zwischen aufeinander folgenden Ausführungen eines VIs beizubehalten.

Steuern des Timings

Möglicherweise möchten Sie die Geschwindigkeit steuern, mit der ein Prozess ausgeführt wird, wie beispielsweise die Geschwindigkeit, mit der Daten in einem Diagramm gezeichnet werden. Sie können die Warten-Funktion in der Schleife verwenden, um die Zeitspanne in Millisekunden anzugeben, für die gewartet werden soll, bevor die Schleife erneut ausgeführt wird.

Case- und Sequenzstrukturen

Case- und Sequenzstrukturen enthalten mehrere Subdiagramme, von denen immer nur jeweils eines sichtbar ist. Eine Case-Struktur führt ein Subdiagramm abhängig von dem an die Struktur übergebenen Eingabewert aus. Eine Sequenzstruktur führt alle Subdiagramme in sequenzieller Reihenfolge aus.

Case-Strukturen



Eine Case-Struktur (siehe links) verfügt über zwei oder mehr Subdiagramme oder Cases. Hiervon ist immer nur jeweils ein Subdiagramm sichtbar, und die Struktur führt immer nur jeweils einen Case aus. Welches Subdiagramm ausgeführt wird, bestimmt der Eingabewert.



Die Case-Selektorkennung am Anfang der Case-Struktur (siehe links) umfasst in der Mitte die Case-Selektorkennung sowie Dekrement- und Inkrement-Schaltflächen auf jeder Seite. Mit den Dekrement- und Inkrement-Schaltflächen können Sie in den verfügbaren Cases blättern.



Um festzulegen, welcher Case ausgeführt wird, verbinden Sie einen Eingabewert mit dem Selektoranschluss (siehe links). Sie müssen eine Ganzzahl, einen booleschen Wert, einen String oder einen Wert vom Enum-Typ mit dem Selektoranschluss verbinden. Sie können den Selektoranschluss an einer beliebigen Stelle am linken Rand der Case-Struktur platzieren.

Sie können einen Standard-Case für die Case-Struktur angeben. Sie müssen einen Standard-Case angeben, um Werte zu verarbeiten, die außerhalb des Bereichs liegen, oder Sie müssen jeden möglichen Eingabewert explizit aufführen. Wenn Sie beispielsweise Cases für 1, 2 und 3 angegeben haben, jedoch den Eingabewert 4 erhalten, führt die Case-Struktur den Standard-Case aus.

Case-Selektorwerte und Datentypen

Geben Sie einen einzelnen Wert oder Listen und Bereiche mit Werten an, um den Case auszuwählen. Im Falle von Listen verwenden Sie Kommata als Trennzeichen zwischen den Werten. Geben Sie Bereiche als `10 . . 20` an, was bedeutet, dass alle Zahlen von 10 bis einschließlich 20 Teil des Bereichs sind. Sie können auch nach oben oder unten offene Bereiche verwenden. Beispielsweise steht `. . 100` für alle Zahlen die kleiner oder gleich 100 sind. Sie können Listen und Bereiche auch kombinieren, also beispielsweise `. . 5, 6, 7 . . 10, 12, 13, 14`. Wenn Sie einen Selektor angeben, der überlappende Bereiche enthält, zeigt die Case-Struktur den Selektor in einer kompakteren Form erneut an. Das vorstehende Beispiel wird als `. . 10, 12 . . 14` erneut angezeigt.

Wenn Sie String- und Enum-Werte in einem Case-Selektor verwenden, werden die Werte in Anführungszeichen angezeigt, zum Beispiel `"rot"`, `"grün"` und `"blau"`. Diese Anführungszeichen müssen Sie jedoch nicht zusammen mit den Werten eingeben, sofern der String oder der Enum-Wert

kein Komma oder Bereichssymbol enthält („ , “ oder „ . . “). Bei einem String-Case-Selektor verwenden Sie spezielle Backslash-Codes für nicht alphanumerische Zeichen wie beispielsweise `\r` für Wagenrücklauf (Carriage Return, CR), `\n` für einen Zeilenvorschub (Linefeed, LF) und `\t` für einen Tabulator. Eine Liste dieser Backslash-Codes finden Sie in der *LabVIEW-Hilfe*.

Wenn Sie den Datentyp der Verbindung ändern, die mit dem Selektoranschluss einer Case-Struktur verbunden ist, wandelt die Case-Struktur automatisch die Case-Selektorwerte, sofern möglich, in den neuen Datentyp um. Wenn Sie einen numerischen Wert, also beispielsweise 19, in einen String umwandeln, lautet der String-Wert „19“. Wenn Sie einen String in einen numerischen Wert umwandeln, konvertiert LabVIEW nur die Selektor-Strings, die eine Zahl darstellen. Die anderen Werte bleiben Strings. Wenn Sie eine Zahl in einen booleschen Wert umwandeln, konvertiert LabVIEW 0 als FALSE (FALSCH) und 1 als TRUE (WAHR), und alle anderen numerischen Werte werden zu Strings.

Wenn Sie einen Selektorwert eingeben, der nicht den gleichen Datentyp wie das Objekt aufweist, das mit dem Selektoranschluss verbunden ist, erscheint der Wert in Rot, um anzuzeigen, dass Sie den Wert löschen oder bearbeiten müssen, bevor die Struktur ausgeführt werden kann, und das VI funktioniert nicht. Darüber hinaus können Sie auch aufgrund der möglichen Rundungsfehler in der Fließkomma-Arithmetik keine Fließkommazahlen als Case-Selektorwerte verwenden. Wenn Sie einen Fließkommawert mit dem Case verbinden, rundet LabVIEW den Wert auf die nächste gerade Ganzzahl. Wenn Sie einen Fließkommawert in den Case-Selektor eingeben, erscheint der Wert in Rot, um anzuzeigen, dass Sie den Wert löschen oder bearbeiten müssen, bevor die Struktur ausgeführt werden kann.

Eingabe- und Ausgabetunnel

Sie können mehrere Eingabe- und Ausgabetunnel für eine Case-Struktur erstellen. Eingaben stehen für alle Subdiagramme zur Verfügung, jedoch müssen Subdiagramme nicht jede Eingabe verwenden. Allerdings müssen Sie jeden Ausgabetunnel für jeden Case definieren. Wenn Sie für einen Case einen Ausgabetunnel erstellen, erscheinen bei allen anderen Cases an derselben Position am Rahmen ebenfalls Tunnel. Nicht verbundene Ausgabetunnel erscheinen als weiße Quadrate. Sie können bei jedem Case eine andere Datenquelle für den gleichen Ausgabetunnel definieren, wobei jedoch die Datentypen kompatibel sein müssen.

Verwenden von Case-Strukturen für die Fehlerbehandlung

Wenn Sie einen Fehler-Cluster mit dem Selektoranschluss einer Case-Struktur verbinden, zeigt die Case-Selektor-Kennung zwei Cases an, Fehler und Kein Fehler, und der Rahmen der Case-Struktur ändert die Farbe, Rot für Fehler und Grün für Kein Fehler. Wenn ein Fehler auftritt, führt die Case-Struktur das geeignete Case-Subdiagramm aus. Weitere Informationen zur Fehlerbehandlung finden Sie in Abschnitt [Fehlerbehandlung](#) des Kapitels 6, [Ausführen von und Fehlersuche in VIs](#).

Sequenzstrukturen



Eine Sequenzstruktur (siehe links) enthält ein oder mehrere Subdiagramme oder Rahmen, die in sequenzieller Reihenfolge ausgeführt werden. Die Sequenzstruktur führt zunächst Rahmen 0, dann Rahmen 1, Rahmen 2 und so weiter aus, bis der letzte Rahmen ausgeführt wurde. Die Sequenzstruktur schließt die Ausführung nicht ab beziehungsweise gibt keine Daten zurück, bis der letzte Rahmen ausgeführt wurde.

Verwenden Sie die Sequenzstruktur, um die Ausführungsreihenfolge zu steuern, wenn keine natürliche Datenabhängigkeit vorliegt. Ein Knoten, der Daten von einem anderen Knoten erhält, ist im Hinblick auf die Daten von diesem Knoten abhängig und wird immer erst ausgeführt, wenn die Ausführung des anderen Knotens abgeschlossen ist.

Innerhalb der einzelnen Rahmen einer Sequenzstruktur bestimmt, wie im Rest des Blockdiagramms, die Datenabhängigkeit die Ausführungsreihenfolge der Knoten. Weitere Informationen zur Datenabhängigkeit finden Sie in Abschnitt [Datenabhängigkeit und künstliche Datenabhängigkeit](#) des Kapitels 5, [Erstellen des Blockdiagramms](#).

Die Tunnel von Sequenzstrukturen können, im Gegensatz zu Case-Strukturen, immer nur über eine Datenquelle verfügen. Die Ausgabe kann von jedem Rahmen ausgehen, jedoch verlassen die Daten die Sequenzstruktur erst, wenn die Ausführung aller Rahmen abgeschlossen ist und nicht, wenn die Ausführung einzelner Rahmen beendet ist. Wie bei Case-Strukturen stehen die Daten an den Eingabetunneln für alle Rahmen zur Verfügung.



Um Daten aus einem Rahmen an einen beliebigen nachfolgenden Rahmen zu übergeben, verwenden Sie einen lokalen Sequenzanschluss wie links dargestellt. Im lokalen Sequenzanschluss des Rahmens, der die Datenquelle enthält, wird ein nach außen weisender Pfeil angezeigt. Der Anschluss im nachfolgenden Rahmen enthält einen nach innen weisenden Pfeil, womit angezeigt wird, dass der Anschluss als Datenquelle für den

Rahmen fungiert. Sie können den lokalen Sequenzanschluss nicht in Rahmen verwenden, die vor dem ersten Rahmen liegen, mit dem Sie den lokalen Sequenzanschluss verbunden haben.

Vermeiden, dass Sequenzstrukturen zu häufig verwendet werden

Um die Vorteile der von LabVIEW gebotenen Parallelverarbeitung zu nutzen, sollten Sie Sequenzstrukturen nicht zu häufig verwenden. Sequenzstrukturen stellen zwar die Ausführungsreihenfolge sicher, machen jedoch parallele Operationen unmöglich. Beispielsweise können asynchrone Aufgaben, bei denen I/O-Geräte wie PXI, GPIB, serielle Ports und DAQ-Geräte verwendet werden, gleichlaufend mit anderen Operationen ausgeführt werden, wenn dies nicht von Sequenzstrukturen verhindert wird. Sequenzstrukturen blenden auch Abschnitte des Blockdiagramms aus und unterbrechen den natürlichen Datenfluss von links nach rechts.

Wenn Sie die Ausführungsreihenfolge steuern müssen, ziehen Sie die Herstellung einer Datenabhängigkeit zwischen Knoten in Betracht. Beispielsweise können Sie die Fehler-I/O verwenden, um die Ausführungsreihenfolge von I/O zu steuern. Weitere Informationen zur Fehler-I/O finden Sie in Abschnitt [Fehlerbehandlung](#) des Kapitels 6, [Ausführen von und Fehlersuche in VIs](#).

Beispiele zur Verwendung von Sequenzstrukturen finden Sie in `examples\general\structs.llb`.

Gruppieren von Daten mit Hilfe von Strings, Arrays und Clustern

Verwenden Sie zum Gruppieren von Daten Strings, Arrays und Cluster. Mit Strings werden Abfolgen aus ASCII-Zeichen gruppiert. Mit Arrays werden Datenelemente des gleichen Typs gruppiert. Mit Clustern lassen sich Datenelemente unterschiedlichen Typs gruppieren.

Weitere Informationen ...

Weitere Informationen zum Gruppieren von Daten mit Strings, Arrays und Clustern finden Sie in der *LabVIEW-Hilfe*.

Strings

Ein String ist eine Abfolge von anzeigbaren oder nicht anzeigbaren ASCII-Zeichen. Mit Strings steht ein plattformunabhängiges Format für Informationen und Daten zur Verfügung. Zu den häufigeren Einsatzbereichen von Strings gehören die Folgenden:

- Erstellen von einfachen Textmeldungen.
- Übergeben von numerischen Daten als Zeichenstrings an Instrumente und anschließendes Konvertieren der Strings in Zahlen.
- Speichern von numerischen Daten auf Datenträger. Zum Speichern von Zahlen in einer ASCII-Datei müssen Sie die Zahlen zunächst in Strings umwandeln, bevor Sie diese in eine Festplattendatei schreiben.
- Erstellen von Anweisungen und Aufforderungen für den Benutzer mit Hilfe von Dialogfeldern.

Auf dem Frontpanel erscheinen Strings als Tabellen, Texteingabefelder und Beschriftungen. Sie bearbeiten und manipulieren Strings mit den String-Funktionen im Blockdiagramm. Sie formatieren Strings für die Verwendung in anderen Applikationen wie beispielsweise Textverarbeitungs- und Tabellenkalkulationsprogramme oder für den Einsatz in anderen VIs oder Funktionen.

Strings auf dem Frontpanel

Verwenden Sie das String-Bedienelement, das sich auf der Palette **Elemente»String & Pfad** befindet, um Texteingabefelder und Beschriftungen zu simulieren. Weitere Informationen zu den String-Bedien- und Anzeigeelementen finden Sie in Abschnitt *String-Bedien- und Anzeigeelemente* des Kapitels 4, *Erstellen des Frontpanels*.

String-Anzeigearten

Klicken Sie mit der rechten Maustaste auf ein String-Bedien- oder Anzeigeelement auf dem Frontpanel, um aus den in Tabelle 9-1 dargestellten Anzeigearten zu wählen. Die Tabelle enthält auch eine Beispielmeldung für jede Anzeigeart.

Tabelle 9-1. String-Anzeigearten

Anzeigeart	Beschreibung	Meldung
Normale Anzeige	Zeigt die druckbaren Zeichen unter Verwendung der Schriftart des Bedienelements an. Nicht druckbare Zeichen werden als Kästchen angezeigt.	Es gibt vier Anzeigearten. \ ist ein Backslash.
'\'-Code-Anzeige	Zeigt Backslash-Codes für alle nicht druckbaren Zeichen an.	Es\sgibt\svier\sAnzeige-\sarten.\n\\\sist\sein\sBackslash.
Passwort-Anzeige	Zeigt ein Sternchen (*) für jedes Zeichen einschließlich Leerzeichen an.	***** *****
Hex-Anzeige	Zeigt den ASCII-Wert jedes Zeichens in Hexadezimalschreibweise anstelle des Zeichens selbst an.	5468 6572 6520 6172 6520 666F 7572 2064 6973 706C 6179 2074 7970 6573 2E0A 5C20 6973 2061 2062 6163 6B73 6C61 7368 2E

Tabellen

Verwenden Sie das Tabellen-Bedienelement, das sich auf der Palette **Elemente**»**Liste & Tabelle** befindet, um auf dem Frontpanel eine Tabelle zu erstellen. Jede Zelle in einer Tabelle ist ein String, und jede Zelle befindet sich in einer Spalte und einer Zeile. Daher ist die Tabelle die Anzeigeform für ein 2D-Array aus Strings. Abbildung 9-1 zeigt eine Tabelle mit allen zugehörigen Teilen. Weitere Informationen zu Arrays finden Sie in Abschnitt *Arrays* dieses Kapitels.

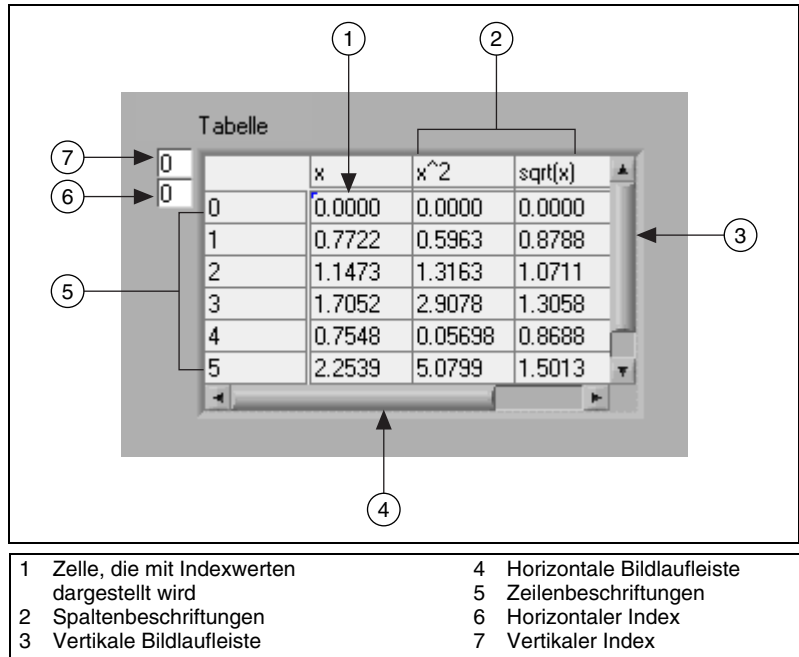


Abbildung 9-1. Bestandteile einer Tabelle

Programmgesteuertes Bearbeiten von Strings

Verwenden Sie die String-Funktionen, die sich auf der Palette **Funktionen**»**String** befinden, um Strings auf folgende Weisen zu bearbeiten:

- Suchen, Abrufen und Ersetzen von Zeichen oder Substrings in einem String.
- Ändern des gesamten Textes in einem String in Groß- oder Kleinbuchstaben.

- Suchen und Abrufen von gleichlautenden Mustern innerhalb eines Strings.
- Abrufen einer Zeile aus einem String.
- Drehen und Umkehren von Text innerhalb eines Strings.
- Verknüpfen von zwei oder mehr Strings.
- Löschen von Zeichen aus einem String.

Beispiele für die Verwendung der String-Funktionen zum Bearbeiten von Strings finden Sie in `examples\general\strings.llb`.

Formatieren von Strings

Um Daten in einem anderen VI, einer anderen Funktion oder Applikation verwenden zu können, müssen Sie die Daten oftmals in einen String konvertieren und diesen String dann in einer Weise formatieren, die das VI, die Funktion oder Applikation lesen kann. Beispielweise erwartet Microsoft Excel Strings, die Trennzeichen enthalten, die Excel dann verwendet, um die Zahlen oder Wörter in Zellen aufzuteilen.

Wenn Sie beispielsweise ein aus Zahlen bestehendes 1D-Array mit der Funktion „Datei schreiben“ in eine Tabelle schreiben möchten, müssen Sie das Array als String formatieren und jede Zahl mit einem Trennzeichen wie beispielsweise einem Tabulator trennen. Um ein aus Zahlen bestehendes Array mit dem VI „In Spreadsheet-Datei schreiben“ in eine Tabelle zu schreiben, müssen Sie das Array mit der Funktion „Array in Tabellen-String“ formatieren und ein Format und ein Trennzeichen angeben.

Verwenden Sie die String-Funktionen, die sich auf der Palette **Funktionen»String** befinden, für die folgenden Aufgaben:

- Verknüpfen von zwei oder mehr Strings.
- Extrahieren der Teilmenge eines Strings aus einem String.
- Umwandeln von Daten in Strings.
- Formatieren eines Strings zur Verwendung in einer Textverarbeitung oder Tabellenkalkulation.

Verwenden Sie die Datei-I/O-VIs und -Funktionen, die sich auf der Palette **Funktionen»Datei-I/O** befinden, um Strings in Text- und Tabellenkalkulationsdateien zu speichern.

Format-Bezeichner

In vielen Fällen müssen Sie einen oder mehrere Format-Bezeichner im Parameter **Format String** einer String-Funktion angeben, um einen String zu formatieren. Ein Format-Bezeichner ist Code, der angibt, wie Daten in einen oder aus einem String umgewandelt werden sollen. LabVIEW verwendet Konvertierungs-codes, um das Textformat des Parameters festzulegen. Beispielsweise konvertiert der Format-Bezeichner %x eine hexadezimale Ganzzahl in einen oder aus einem String.

Bei den Funktionen „In String formatieren“ und „Aus String suchen“ können mehrere Format-Bezeichner im **Format String**-Parameter verwendet werden, und zwar einer für jede Eingabe an die oder Ausgabe aus der erweiterbare(n) Funktion.

Bei den Funktionen „Array in Tabellenstring“ und „Tabellenstring in Array“ wird im **Format String**-Parameter nur ein Format-Bezeichner verwendet, weil diese Funktionen nur jeweils eine Eingabe zu konvertieren haben. LabVIEW behandelt alle zusätzlichen Bezeichner, die Sie in diese Funktionen einfügen, als Buchstabenketten ohne besondere Bedeutung.

Zahlen und Strings

Numerische Daten und String-Daten unterscheiden sich voneinander, da es sich bei String-Daten um ASCII-Zeichen handelt, bei numerischen Daten jedoch nicht. Text- und Tabellenkalkulationsdateien akzeptieren nur Strings. Wenn Sie numerische Daten an eine Text- oder Tabellenkalkulationsdatei schreiben möchten, müssen Sie die numerischen Daten zunächst in einen String umwandeln.

Wenn Sie einem vorhandenen String einen Satz Zahlen hinzufügen möchten, konvertieren Sie die numerischen Daten zunächst in einen String und verwenden dann die Funktion „Strings verknüpfen“ oder eine andere String-Funktion, um den neuen String dem vorhandenen String hinzuzufügen. Verwenden Sie die String-/Zahlkonvertierungsfunktionen, die sich auf der Palette **Funktionen»String»String-/Zahlkonvertierung** befinden, um Zahlen in Strings umzuwandeln.

Ein String kann einen Satz Zahlen enthalten, die Sie in einem Graphen oder einem Diagramm anzeigen. Beispielsweise können Sie eine Textdatei lesen, die einen Satz Zahlen enthält, die Sie in einem Diagramm zeichnen möchten. Allerdings weisen diese Zahlen das Format ASCII-Text auf, daher müssen Sie die Zahlen als String lesen und diesen String dann in einen Satz Zahlen konvertieren, bevor Sie die Zahlen in einem Diagramm zeichnen können.

Abbildung 9-2 zeigt einen String, der einen Satz Zahlen enthält, wie der String in Zahlen umgewandelt wird, ein Zahlen-Array erstellt wird und wie die Zahlen in einem Diagramm gezeichnet werden.

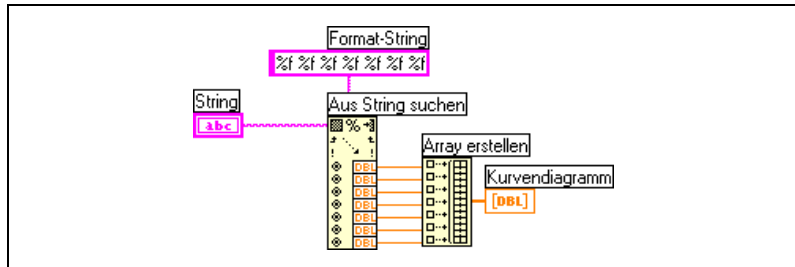


Abbildung 9-2. Konvertieren eines Strings in Zahlen

Gruppieren von Daten mit Arrays und Clustern

Verwenden Sie die Array- und Cluster-Bedienelemente und -Funktionen, die sich auf den Paletten **Elemente»Array & Cluster**, **Funktionen»Array** und **Funktionen»Cluster** befinden, um Daten zu gruppieren. Mit Arrays werden Datenelemente des gleichen Typs gruppiert. Mit Clustern lassen sich Datenelemente unterschiedlichen Typs gruppieren.

Arrays

Ein Array besteht aus Elementen und Dimensionen. Elemente sind die Daten, aus denen sich das Array zusammensetzt. Eine Dimension ist die Länge, die Höhe oder die Tiefe eines Arrays. Ein Array kann eine oder mehrere Dimensionen und bis zu $2^{31} - 1$ Elemente pro Dimension aufweisen, abhängig vom verfügbaren Speicher.

Sie können Arrays aus numerischen, booleschen, Pfad-, String-, Signalverlauf- und Cluster-Datentypen erstellen. Ziehen Sie die Verwendung von Arrays in Betracht, wenn Sie mit einer Sammlung aus ähnlichen Daten arbeiten und wenn Sie sich wiederholende Berechnungen ausführen.

Arrays eignen sich ideal für das Speichern der Daten, die aus Signalverläufen gesammelt werden, oder für das Speichern von Daten, die in Schleifen erzeugt werden, bei denen jede Wiederholung der Schleife ein Element des Arrays erzeugt.

Sie können keine aus Arrays bestehenden Arrays erzeugen. Sie können jedoch ein Array aus Clustern erzeugen, von denen jeder ein oder mehrere Arrays enthält. Weitere Informationen zu den Elementtypen, die ein Array enthalten kann, finden Sie in Abschnitt *Beschränkungen für Arrays* dieses

Kapitels. Weitere Informationen zu Clustern finden Sie in Abschnitt *Cluster* dieses Kapitels.

Indizes

Um ein bestimmtes Element in einem Array zu finden, muss ein Index pro Dimension vorhanden sein. In LabVIEW können Sie mit Hilfe von Indizes in einem Array navigieren und Elemente, Zeilen, Spalten und Seiten aus einem Array im Blockdiagramm abrufen.

Beispiele für Arrays

Ein Beispiel für ein einfaches Array ist ein Text-Array, in dem die neun Planeten unseres Sonnensystems aufgeführt sind. LabVIEW stellt dies als ein aus Strings bestehendes 1D-Array mit neun Elementen dar.

Array-Elemente sind geordnet, ebenso wie die neun Planeten eine festgefügte Reihenfolge im Hinblick auf ihre Entfernung zur Sonne aufweisen. Bei einem Array wird ein Index verwendet, so dass Sie auf jedes spezielle Element unverzüglich zugreifen können. Der Index ist nullbasiert, was bedeutet, dass er im Bereich 0 bis $n - 1$ liegt, wobei n für die Anzahl der Elemente im Array steht. In diesem Beispiel steht $n = 9$ für die neun Planeten, also rangiert der Index von 0 bis 8. Die Erde ist der dritte Planet, also gilt hierfür der Index 2.

Ein weiteres Beispiel für ein Array ist ein Signalverlauf, der als ein numerisches Array dargestellt wird, in dem jedes nachfolgende Element den Spannungswert in den nachfolgenden Zeitintervallen wiedergibt, wie in Abbildung 9-3 gezeigt.

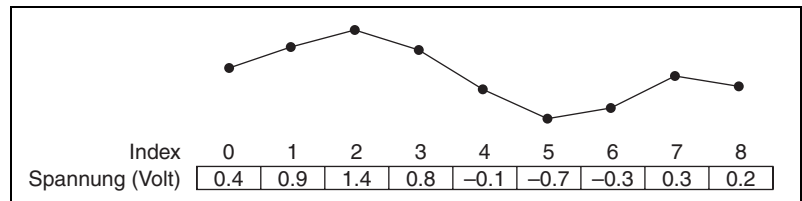


Abbildung 9-3. Signalverlauf als ein aus Zahlen bestehendes Array

Ein etwas komplexeres Beispiel für ein Array ist ein Graph, der als ein Array aus Punkten dargestellt wird und bei dem jeder Punkt ein Cluster ist, der wiederum ein Zahlenpaar enthält, das die X-/Y-Koordinaten darstellt, wie in Abbildung 9-4 gezeigt.

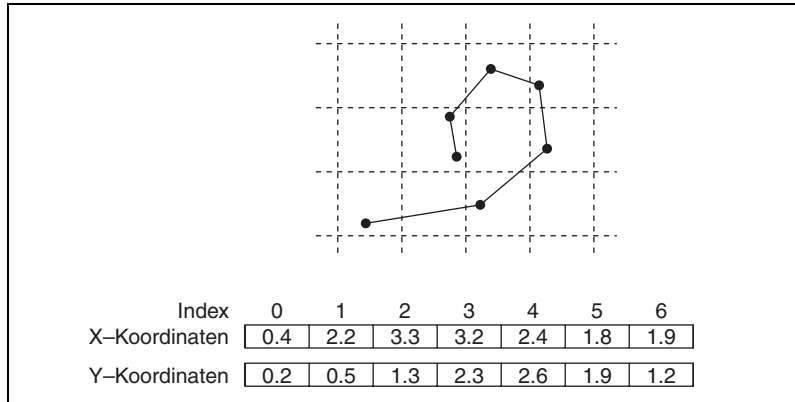


Abbildung 9-4. Graph als ein aus Punkten bestehendes Array

In den vorstehenden Beispielen wird ein 1D-Array verwendet. Bei einem 2D-Array werden die Elemente in einem Raster gespeichert. Dies setzt einen Spalten- und einen Zeilenindex zum Suchen eines Elements voraus, die beide jeweils nullbasiert sind. Abbildung 9-5 zeigt ein aus 6 Spalten und 4 Zeilen bestehendes 2D-Array, das $6 \times 4 = 24$ Elemente enthält.

		Spalten index					
		0	1	2	3	4	5
Reihenindex	0						
	1						
	2						
	3						

Abbildung 9-5. 2D-Array mit 6 Spalten und 4 Zeilen

Beispielsweise enthält ein Schachbrett acht Spalten und acht Zeilen für insgesamt 64 Positionen. Jede Position kann leer sein oder eine Schachfigur enthalten. Ein Schachbrett können Sie also als ein aus Strings bestehendes 2D-Array darstellen. Hierbei ist jeder String der Name der Figur, welche die entsprechende Position auf dem Schachbrett einnimmt, oder ein leerer String, wenn die Position leer ist.

Sie können die Beispiele für 1D-Arrays in den Abbildungen 9-3 und 9-4 zweidimensional machen, indem Sie dem Array eine Zeile hinzufügen. Abbildung 9-6 zeigt eine Sammlung von Signalverläufen, die als ein 2D-Array aus Zahlen dargestellt wird. Mit dem Zeilenindex wird der Signalverlauf gewählt, mit dem Spaltenindex ein Punkt im Signalverlauf.

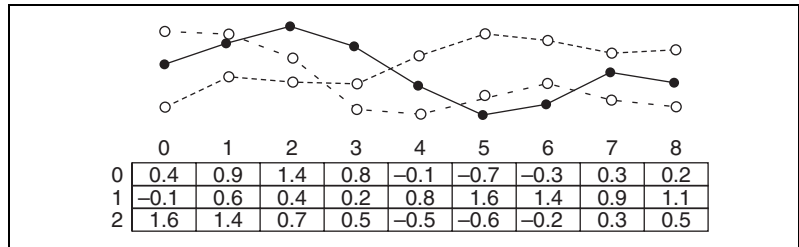


Abbildung 9-6. Mehrere Signalverläufe in einem aus Zahlen bestehenden 2D-Array

Weitere Beispiele für Arrays finden Sie in `examples\general\arrays.llb`.

Beschränkungen für Arrays

Sie können aus beinahe jedem Datentyp ein Array erstellen, wobei die folgenden Ausnahmen gelten:

- Sie können keine aus Arrays bestehenden Arrays erzeugen. Sie können jedoch ein mehrdimensionales Array verwenden oder ein Array aus Clustern erzeugen, von denen jeder ein oder mehrere Arrays enthält.
- Sie können kein Array aus Nicht-XY-Graphen erzeugen, da ein Graph ein Array-Datentyp ist und ein Array kein anderes Array enthalten kann. Sie können jedoch ein Array aus Nicht-XY-Graphen erstellen, wenn sich der Graph in einem Cluster befindet.
- Sie können kein aus Diagrammen bestehendes Array erzeugen.

Erstellen von Array-Bedien- und Anzeigeelementen und -Konstanten

Sie erstellen ein Array-Bedien- oder Anzeigeelement auf dem Frontpanel, indem Sie einen Array-Container, wie in Abbildung 9-7 gezeigt, platzieren und ein Datenobjekt oder Element, bei dem es sich um ein Zahl-, boolesches, String-, Pfad-, Refnum- oder Cluster-Bedien- oder Anzeigeelement handeln kann, in den Array-Container ziehen.

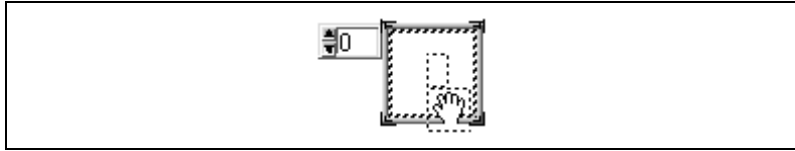


Abbildung 9-7. Array-Container

Der Array-Container passt ihre Größe automatisch an das neue Objekt an, ungeachtet, ob es sich um ein kleines boolesches Bedienelement oder um einen großen 3D-Graphen handelt.

Um ein bestimmtes Element auf dem Frontpanel anzuzeigen, geben Sie entweder die Indexnummer in die Indexanzeige ein, oder verwenden Sie die Pfeile der Indexanzeige, um zu dieser Nummer zu navigieren.

Zum Erstellen einer Array-Konstanten im Blockdiagramm wählen Sie **Funktionen»Array»Array-Konstante** zum Platzieren des Array-Containers und setzen dann eine String-Konstante, eine numerische Konstante oder eine Cluster-Konstante in diesen Array-Container. Sie können eine Array-Konstante als Basis für den Vergleich mit einem anderen Array verwenden.

Array-Indexanzeige

Ein 2D-Array enthält Zeilen und Spalten. Wie in Abbildung 9-8 gezeigt ist die obere Anzeige mit den zwei Feldern links der Zeilenindex, und die untere Anzeige ist der Spaltenindex. Die kombinierte Anzeige rechts der Zeilen- und Spaltenanzeigen zeigt den Wert an der angegebenen Position an. Abbildung 9-8 zeigt, dass der Wert an Zeile 6, Spalte 13 66 lautet.

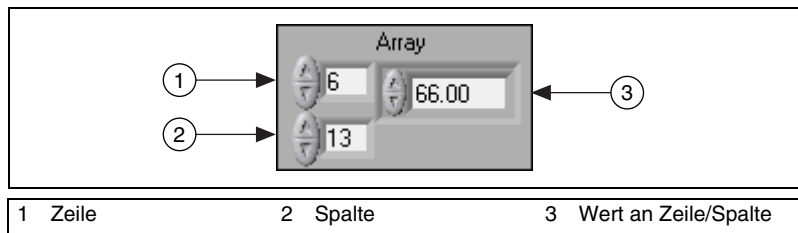


Abbildung 9-8. Array-Bedienelement

Zeilen und Spalten sind nullbasiert, was bedeutet, dass die erste Spalte 0, die zweite Spalte 1 und so weiter ist. Wird die Indexanzeige des folgenden Arrays auf Zeile 1, Spalte 2 geändert, wird der Wert 6 angezeigt.

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

Wenn Sie versuchen, eine Spalte oder eine Zeile anzuzeigen, die außerhalb des Bereichs der Array-Dimensionen liegt, wird das Array-Bedienelement abgeblendet, um anzuzeigen, dass kein Wert definiert ist, und LabVIEW zeigt den Standardwert des Datentyps an. Der Standardwert des Datentyps ist abhängig vom Datentyp des Arrays.

Verwenden Sie das Positionierwerkzeug, um mehr als eine Zeile oder Spalte gleichzeitig anzuzeigen.

Array-Funktionen

Verwenden Sie die auf der Palette **Funktionen»Arrays** befindlichen Array-Funktionen zum Erstellen und Bearbeiten von Arrays, wie mit den folgenden Beispielen verdeutlicht:

- Extrahieren von einzelnen Datenelementen aus einem Array.
- Einfügen, Löschen oder Ersetzen von Datenelementen in einem Array.
- Teilen von Arrays.

Automatisches Ändern der Größe von Array-Funktionen

Die Funktionen „Array indizieren“, „Array-Teilmenge ersetzen“, „In Array einfügen“, „Aus Array entfernen“ und „Array-Teilmenge“ passen ihre Größe automatisch an, um den Dimensionen des Eingabe-Arrays, das Sie verbinden, zu entsprechen. Wenn Sie beispielsweise ein 1D-Array mit einer dieser Funktionen verbinden, weist die Funktion einen einzelnen Indexeingang auf. Wenn Sie ein 2D-Array mit der gleichen Funktion verbinden, werden zwei Indexeingänge angezeigt, einer für die Zeile und einer für die Spalte.

Sie können mit diesen Funktionen auf mehr als ein Element oder Teil-Array (Zeile, Spalte oder Seite) zugreifen, indem Sie die Größe der Funktion mit Hilfe des Positionierwerkzeugs manuell ändern. Wenn Sie eine dieser Funktionen erweitern, werden die Funktionen in Inkrementen erweitert, die von den Dimensionen des Arrays bestimmt werden, das mit der Funktion

verbunden wird. Wenn Sie ein 1D-Array mit einer dieser Funktionen verbinden, wird die Funktion um einen einzelnen Indexeingang erweitert. Wenn Sie ein 2D-Array mit der gleichen Funktion verbinden, wird die Funktion um zwei Indexeingänge erweitert, einer für die Zeile und einer für die Spalte.

Die Indexeingänge, die Sie verbinden, bestimmen die Form des Teil-Arrays, auf das Sie zugreifen oder das Sie ändern möchten. Wenn der Eingang einer Array indizieren-Funktion ein 2D-Array ist und Sie nur den Eingang **Index (Zeile)** verbinden, extrahieren Sie eine vollständige 1D-Zeile des Arrays. Wenn Sie nur den Eingang **Index (Spalte)** verbinden, extrahieren Sie eine vollständige 1D-Spalte des Arrays. Wenn Sie den Eingang **Index (Zeile)** und den Eingang **Index (Spalte)** verbinden, extrahieren Sie ein einzelnes Element des Arrays. Jede Eingangsgruppe ist unabhängig und kann auf einen beliebigen Teil jeder Dimension des Arrays zugreifen.

Bei dem in Abbildung 9-9 gezeigten Blockdiagramm wird die Array indizieren-Funktion verwendet, um eine Zeile und ein Element aus einem 2D-Array abzurufen.

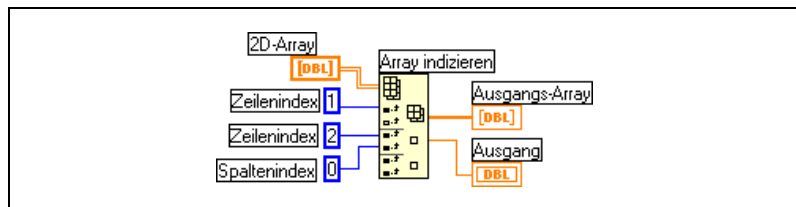


Abbildung 9-9. Indizieren eines 2D-Arrays

Um auf mehrere aufeinander folgende Werte in einem Array zuzugreifen, erweitern Sie die Array indizieren-Funktion, verbinden jedoch keine Werte mit den Index-Eingängen in jedem Inkrement. Um beispielsweise die erste, die zweite und die dritte Zeile eines 2D-Arrays abzurufen, erweitern Sie die Array indizieren-Funktion um drei Inkremente und verbinden 1D-Array-Anzeigeelemente mit jedem **Teil-Array**-Ausgang.

Cluster

Mit Clustern werden Datenelemente mit unterschiedlichen Datentypen gruppiert, wie beispielsweise ein Kabelbündel in einem Telefonkabel, bei dem jedes Kabel ein anderes Element des Clusters darstellt.

Das Bündeln von mehreren Datenelementen zu Clustern vermeidet unübersichtliche Verbindungen im Blockdiagramm und verringert die Anzahl der Anschlussfeldanschlüsse, die für Sub-VIs benötigt werden.

Ein Anschlussfeld kann über maximal 28 Anschlüsse verfügen. Wenn das Frontpanel mehr als 28 Bedien- und Anzeigeelemente enthält, die programmgesteuert verwendet werden sollen, fassen Sie einige hiervon als Cluster zusammen und weisen diesem Cluster einen Anschluss des Anschlussfeldes zu.

Obwohl Cluster- und Array-Elemente jeweils geordnet sind, müssen Sie alle Cluster-Elemente auf einmal aufschlüsseln anstatt jeweils ein Element gleichzeitig zu indizieren. Sie können auch die Funktion „Nach Namen aufschlüsseln“ verwenden, um auf bestimmte Cluster-Elemente zuzugreifen. Cluster unterscheiden sich auch insofern von Arrays, als dass sie eine feste Größe aufweisen. Wie ein Array kann auch ein Cluster entweder ein Bedien- oder ein Anzeigeelement sein. Ein Cluster kann keine Mischung aus Bedien- und Anzeigeelementen enthalten.

Die meisten Cluster im Blockdiagramm verfügen über ein rosafarbenes Verbindungsmuster und ein Datentypsymbol. Cluster aus Zahlen, manchmal auch als Punkte bezeichnet, verfügen über ein braunes Verbindungsmuster und ein Datentypsymbol. Sie können braune numerische Cluster mit numerischen Funktionen wie beispielsweise „Addieren“ oder „Quadratwurzel“ verbinden, um die gleiche Operation gleichzeitig für alle Elemente des Clusters durchzuführen.

Zum Verbinden von Clustern müssen beide Cluster über die gleiche Anzahl Elemente verfügen. Die entsprechenden Elemente, bestimmt durch die Clusterreihenfolge, müssen kompatible Datentypen aufweisen. Wenn Sie beispielsweise eine Fließkommazahl doppelter Genauigkeit und einen String im gleichen Cluster verwenden, erscheint die Verbindung im Blockdiagramm als unterbrochen, und das VI kann nicht ausgeführt werden. Wenn Sie Zahlen mit unterschiedlicher Darstellungsform im gleichen Cluster verwenden, erzwingt LabVIEW hierfür die gleiche Darstellungsform. Weitere Informationen zur Umwandlung von Zahlen finden Sie in Abschnitt *Numerische Konvertierung* des Anhangs B, *Polymorphe Funktionen*.

Verwenden Sie die auf der Palette **Funktionen»Cluster** befindlichen Cluster-Funktionen zum Erstellen und Bearbeiten von Clustern, wie mit den folgenden Beispielen verdeutlicht:

- Extrahieren von einzelnen Datenelementen aus einem Cluster.
- Hinzufügen von einzelnen Datenelementen zu einem Cluster.
- Aufteilen eines Clusters in die einzelnen Datenelemente.

Lokale und globale Variablen

In LabVIEW lesen Sie Daten aus einem oder schreiben Daten an ein Frontpanel-Objekt, indem Sie dessen Blockdiagrammanschluss verwenden. Ein Frontpanel-Objekt verfügt jedoch nur über einen Blockdiagrammanschluss, und in Ihrer Applikation müssen Sie möglicherweise von mehr als einer Position aus auf die Daten dieses Anschlusses zugreifen können.

Lokale und globale Variablen übergeben Informationen zwischen Positionen in der Applikation, die nicht über eine Verbindung miteinander verbunden werden können. Sie verwenden lokale Variablen, um von mehr als einer Position in einem einzelnen VI auf Frontpanel-Objekte zugreifen zu können. Sie verwenden globale Variablen, um auf Daten in mehreren VIs zuzugreifen und um Daten zwischen mehreren VIs zu übergeben.

Weitere Informationen ...

Weitere Informationen zum Einsatz von lokalen und globalen Variablen finden Sie in der *LabVIEW-Hilfe*.

Lokale Variablen

Sie verwenden lokale Variablen, um von mehr als einer Position in einem einzelnen VI auf Frontpanel-Objekte zuzugreifen und um Daten zwischen Blockdiagrammstrukturen zu übergeben, die nicht über eine Verbindung miteinander verbunden werden können.

Mit einer lokalen Variablen können Sie an ein Bedien- oder Anzeigeelement auf dem Frontpanel schreiben oder dieses auslesen. Das Schreiben an eine lokale Variable ist mit der Übergabe von Daten an einen beliebigen anderen Anschluss vergleichbar. Mit einer lokalen Variablen können Sie jedoch auch dann an den Anschluss schreiben, wenn es sich um ein Bedienelement handelt, und Sie können ihn auch dann auslesen, wenn es sich um ein Anzeigeelement handelt. Tatsächlich können Sie mit einer lokalen Variablen auf ein Frontpanel-Objekt sowohl als Eingang als auch als Ausgang zugreifen.

Wenn die Benutzeroberfläche beispielsweise voraussetzt, dass die Benutzer sich anmelden, können Sie die Eingabeaufforderungen **Anmeldename** und **Password** jedes Mal löschen, wenn sich ein neuer Benutzer anmeldet. Verwenden Sie eine lokale Variable, um die String-Bedienelemente **Anmeldename** und **Password** auszulesen, wenn sich der Benutzer anmeldet, und um leere Strings an diese Bedienelemente zu schreiben, wenn sich der Benutzer abmeldet.

Erstellen von lokalen Variablen

Klicken Sie mit der rechten Maustaste auf ein Frontpanel-Objekt oder einen Blockdiagrammanschluss, und wählen Sie aus dem Kontextmenü **Erstelle»Lokale Variable**, um eine lokale Variable zu erstellen. Sie können auch **Funktionen»Strukturen»Lokale Variable** wählen, um eine lokale Variable zu erstellen.



Ein lokaler Variablenknoten (siehe links), der noch nicht mit einem Bedien- oder Anzeigeelement verbunden ist, wird nun im Blockdiagramm angezeigt. Klicken Sie mit der rechten Maustaste auf den lokalen Variablenknoten, und wählen Sie **Objekt wählen** aus dem Kontextmenü, um anzugeben, welches Frontpanel-Objekt Sie mit der lokalen Variablen verbinden möchten. Das Untermenü **Objekt wählen** des Kontextmenüs enthält eine Liste aller Frontpanel-Bedienelemente, die über verknüpfte Beschriftungen verfügen.

LabVIEW verwendet verknüpfte Beschriftungen, um lokale Variablen mit Frontpanel-Objekten zu verbinden, also sollten Sie Frontpanel-Bedien- und Anzeigeelemente mit aussagekräftigen verknüpften Beschriftungen versehen. Weitere Informationen zu verknüpften und freien Beschriftungen finden Sie in Abschnitt [Beschriftungen](#) des Kapitels 4, [Erstellen des Frontpanels](#).

Globale Variablen

Sie verwenden globale Variablen, um auf Daten in mehreren VIs zuzugreifen und um Daten zwischen mehreren VIs zu übergeben, die gleichzeitig ausgeführt werden. Globale Variablen sind integrierte LabVIEW-Objekte. Wenn Sie eine globale Variable erstellen, erstellt LabVIEW automatisch ein spezielles globales VI, das über ein Frontpanel, jedoch nicht über ein Blockdiagramm verfügt. Fügen Sie dem Frontpanel des globalen VIs Bedien- und Anzeigeelemente hinzu, um die Datentypen der globalen Variablen zu definieren, die es enthält. Tatsächlich ist dieses Frontpanel ein Container, von dem aus mehrere VIs auf Daten zugreifen können.

Nehmen wir beispielsweise einmal an, Sie verfügen über zwei VIs, die gleichzeitig ausgeführt werden. Jedes VI enthält eine While-Schleife und schreibt Daten an ein Signalverlaufsdiagramm. Das erste VI enthält ein boolesches Bedienelement, mit dem beide VIs beendet werden. Sie müssen nun eine globale Variable verwenden, um beide Schleifen mit einem einzigen booleschen Bedienelement beenden zu können. Wenn sich beide Schleifen in einem einzigen Blockdiagramm im gleichen VI befänden, könnten Sie eine lokale Variable verwenden, um die Schleifen zu beenden.

Erstellen von globalen Variablen



Wählen Sie **Funktionen»Strukturen»Globale Variable**, um eine globale Variable zu erstellen. Nun wird ein globaler Variablenknoten im Blockdiagramm angezeigt. Doppelklicken Sie auf den globalen Variablenknoten, um das Frontpanel des globalen VIs anzuzeigen. Setzen Sie Bedien- und Anzeigeelemente in der gleichen Weise auf dieses Frontpanel, wie Sie auch bei einem standardmäßigen Frontpanel vorgehen.

Sie können mehrere einzelne globale VIs erstellen, von denen jedes über ein Frontpanel-Objekt verfügt, oder Sie können ein globales VI mit mehreren Frontpanel-Objekten erstellen. Ein globales VI mit mehreren Objekten ist effizienter, da Sie zusammenhängende Variablen gruppieren können. Das Blockdiagramm eines VIs kann mehrere globale Variablenknoten enthalten, die mit Bedien- und Anzeigeelementen auf dem Frontpanel eines globalen VIs verbunden sind. Diese globalen Variablenknoten sind entweder Kopien des ersten globalen Variablenknotens, den Sie im Blockdiagramm des globalen VIs platziert haben, oder es sind globale Variablenknoten von globalen VIs, die Sie im aktuellen VI platziert haben. Sie platzieren globale VIs in der gleichen Weise in anderen VIs, wie Sie Sub-VIs in anderen VIs platzieren. Jedes Mal, wenn Sie einen neuen globalen Variablenknoten in einem Blockdiagramm platzieren, erstellt LabVIEW ein neues VI, das nur mit diesem globalen Variablenknoten verbunden ist, sowie Kopien hiervon. Weitere Informationen zu Sub-VIs finden Sie in Abschnitt *Sub-VIs* des Kapitels 7, *Erstellen von VIs und Sub-VIs*.

Nachdem Sie mit dem Platzieren von Objekten auf dem Frontpanel des globalen VIs fertig sind, speichern Sie dieses und kehren zum Blockdiagramm des ursprünglichen VIs zurück. Anschließend müssen Sie das Objekt im globalen VI auswählen, auf das Sie zugreifen möchten. Klicken Sie mit der rechten Maustaste auf den globalen Variablenknoten, und wählen Sie aus dem Kontextmenü **Objekt wählen**, um das Objekt anzugeben, auf das Sie zugreifen möchten. Das Untermenü **Objekt wählen** des Kontextmenüs

enthält eine Liste aller Frontpanel-Objekte des globalen VIs, die über verknüpfte Beschriftungen verfügen.

LabVIEW verwendet verknüpfte Beschriftungen, um globale Variablen zu kennzeichnen, also sollten Sie Frontpanel-Bedien- und Anzeigeelemente mit aussagekräftigen verknüpften Beschriftungen versehen. Weitere Informationen zu verknüpften und freien Beschriftungen finden Sie in Abschnitt *Beschriftungen* des Kapitels 4, *Erstellen des Frontpanels*.

Lesen und Schreiben von Variablen

Nachdem Sie eine lokale oder globale Variable erstellt haben, können Sie Daten aus der Variablen lesen oder Daten an die Variable schreiben. Standardmäßig empfängt eine neue Variable Daten. Diese Art Variable fungiert als Anzeigeelement und ist eine lokale oder globale Variable mit Schreibzugriff. Wenn Sie neue Daten an die lokale oder globale Variable schreiben, wird das verbundene Frontpanel-Bedien- oder Anzeigeelement mit den neuen Daten aktualisiert.

Sie können eine Variable auch so konfigurieren, dass sie als Datenquelle fungiert oder als lokale oder globale Variable mit Lesezugriff. Klicken Sie mit der rechten Maustaste auf die Variable, und wählen Sie aus dem Kontextmenü **In Lesen ändern**, um die Variable als Bedienelement zu konfigurieren. Wenn der Knoten ausgeführt wird, liest das VI die Daten im zugehörigen Frontpanel-Bedien- oder Anzeigeelement.

Um die Variable so zu ändern, dass sie Daten aus dem Blockdiagramm empfängt anstatt Daten bereitzustellen, klicken Sie mit der rechten Maustaste auf die Variable und wählen aus dem Kontextmenü **In Schreiben ändern**.

Im Blockdiagramm können Sie lokale und globale Variablen mit Lesezugriff in der gleichen Weise von lokalen oder globalen Variablen mit Schreibzugriff unterscheiden, wie Sie Bedienelemente von Anzeigeelementen unterscheiden. Eine lokale oder globale Variable mit Lesezugriff verfügt ähnlich wie ein Bedienelement über einen dickeren Rahmen. Eine lokale oder globale Variable mit Schreibzugriff verfügt ähnlich wie ein Anzeigeelement über einen dünnen Rahmen.

Beispiele für den Einsatz von lokalen und globalen Variablen finden Sie in `examples\general\locals.llb` und `example\general\globals.llb`.

Umsichtiges Verwenden von lokalen und globalen Variablen

Lokale und globale Variablen gehören zu den fortgeschrittenen LabVIEW-Konzepten. Sie sind naturgemäß kein Bestandteil des LabVIEW-Datenfluss-Ausführungsmodells. Das Lesen von Blockdiagrammen wird möglicherweise erschwert, wenn Sie lokale und globale Variablen verwenden, daher sollten sie umsichtig eingesetzt werden. Der Missbrauch von lokalen und globalen Variablen, wie beispielsweise deren Einsatz anstelle eines Anschlussfeldes, kann zu unerwartetem Verhalten von VIs führen. Der übermäßige Einsatz von lokalen und globalen Variablen, wie beispielsweise deren Einsatz anstelle von Verbindungen oder anstelle des Datenflusses, verringert die Leistungsfähigkeit. Weitere Informationen zum LabVIEW-Datenfluss-Ausführungsmodell finden Sie in Abschnitt *Blockdiagramm-Datenfluss* des Kapitels 5, *Erstellen des Blockdiagramms*.

Initialisieren von lokalen und globalen Variablen

Vergewissern Sie sich, dass die lokalen und globalen Variablen bekannte Daten enthalten, bevor das VI ausgeführt wird. Andernfalls könnten die Variablen Daten enthalten, die dazu führen, dass das VI nicht korrekt funktioniert.

Wenn Sie die Variable nicht initialisieren, bevor das VI die Variable zum ersten Mal liest, enthält die Variable den Standardwert des verbundenen Frontpanel-Objekts.

Ablaufunterschiede

Da VIs einem Datenfluss-Ausführungsmodell folgen, verhalten sich die lokalen und globalen LabVIEW-Variablen nicht wie die lokalen und globalen Variablen in textbasierten Programmiersprachen. Ein Ablaufunterschied tritt auf, wenn zwei oder mehr Codefragmente, die parallel ausgeführt werden, den Wert derselben gemeinsam genutzten Ressource (normalerweise eine globale oder lokale Variable) ändern. In Abbildung 10-1 finden Sie ein Beispiel für einen Ablaufunterschied.

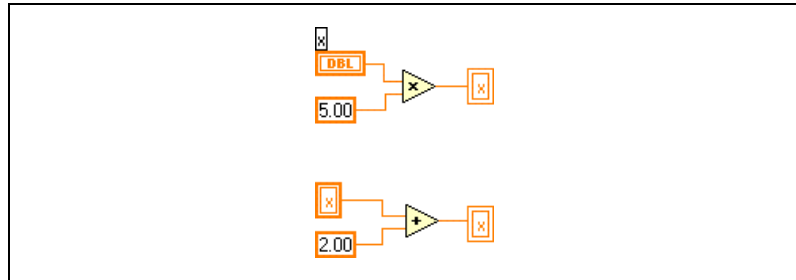


Abbildung 10-1. Ablaufunterschied

Die Ausgabe dieses VIs ist abhängig von der Reihenfolge, in der die Operationen ausgeführt werden. Da zwischen den beiden Operationen keine Datenabhängigkeit besteht, gibt es keine Möglichkeit, um festzustellen, welche zuerst ausgeführt wird. Zur Vermeidung von Ablaufunterschieden sollten Sie nicht an die gleiche Variable schreiben, aus der Sie lesen. Weitere Informationen zur Datenabhängigkeit finden Sie in Abschnitt *Datenabhängigkeit und künstliche Datenabhängigkeit* des Kapitels 5, *Erstellen des Blockdiagramms*.

Den Speicher betreffende Überlegungen beim Einsatz von lokalen Variablen

Beim Erstellen von Sub-VIs erstellen Sie ein Anschlussfeld, welches definiert, wie Daten an das Sub-VI übergeben und aus dem Sub-VI übernommen werden. Das Anschlussfeld erstellt keine Kopien der Datenpuffer der aufrufenden VIs.

Lokale Variablen erstellen Kopien der Datenpuffer. Wenn Sie eine lokale Variable auslesen, erstellen Sie einen neuen Puffer für die Daten des hiermit verbundenen Bedienelements.

Wenn Sie lokale Variablen verwenden, um große Datenmengen von einer Stelle im Blockdiagramm an eine andere zu übergeben, benötigen Sie im Allgemeinen mehr Speicher, und dementsprechend verlangsamt sich die Ausführungsgeschwindigkeit gegenüber der Datenübergabe über eine Verbindung.

Den Speicher betreffende Überlegungen beim Einsatz von globalen Variablen

Wenn Sie eine globale Variable auslesen, erstellt LabVIEW eine Kopie der in dieser globalen Variablen gespeicherten Daten.

Wenn Sie große Arrays und Strings bearbeiten, kann die Zeit und die Speicherkapazität, die für das Bearbeiten von globalen Variablen benötigt wird, beträchtlich sein. Das Bearbeiten globaler Variablen ist besonders ineffizient, wenn es sich um Arrays handelt, denn auch wenn Sie nur ein einziges Array-Element ändern, speichert LabVIEW das gesamte Array. Wenn Sie die globale Variable an mehreren Stellen einer Applikation auslesen, erstellen Sie mehrere Speicherpuffer, was ineffizient ist und die Leistungsfähigkeit verringert.

Graphen und Diagramme

Mit Graphen und Diagrammen können Sie Daten in grafischer Form darstellen.

Graphen und Diagramme unterscheiden sich hinsichtlich Anzeige und Aktualisierung von Daten. VIs mit Graphen erfassen die Daten normalerweise in einem Array und stellen die Daten anschließend in einem Graphen dar, ähnlich wie in einer Tabellenkalkulation, bei der die Daten zuerst gespeichert und dann grafisch dargestellt werden. Im Gegensatz dazu erweitert ein Diagramm die bereits angezeigten Datenwerte. In einem Diagramm können Sie den aktuellen Messwert beziehungsweise die aktuelle Messung in Zusammenhang mit den zuvor erfassten Daten anzeigen.

Weitere Informationen ...

Weitere Informationen über die Verwendung von Graphen und Diagrammen finden Sie in der *LabVIEW-Hilfe*.

Grafik- und Diagrammtypen

In der Palette **Elemente»Graph** sind die folgenden Typen von Graphen und Diagrammen enthalten:

- **Kurvendiagramm und Kurvengraph:** Zeigt die erfassten Daten mit konstanter Rate an.
- **XY-Graph:** Zeigt die erfassten Daten mit nicht konstanter Rate an, wie zum Beispiel Daten, die getriggert erfasst werden.
- **Intensitätsdiagramm und Intensitätsgraph:** Zeigt dreidimensionale Daten in einer zweidimensionalen Darstellung an, wobei die Werte der dritten Dimension anhand von Farben angezeigt werden.
- **Digitaler Kurvengraph:** Zeigt Daten als Pulse oder als Gruppen digitaler Linien an. Computer übertragen digitale Daten als Pulse zu anderen Computern.
- **(Windows) 3D-Graphen:** Zeigen die 3D-Daten einer dreidimensionalen Darstellung auf dem Frontpanel in einem ActiveX-Objekt an.

Beispiele für Graphen und Diagramme finden Sie in `examples\general\graphs`.

Optionen für Graphen und Diagramme

Obwohl Graphen und Diagramme Daten unterschiedlich darstellen, verfügen sie über mehrere gemeinsame Optionen, auf die Sie über das Kontextmenü zugreifen können. Weitere Informationen zu den nur für Graphen beziehungsweise Diagrammen verfügbaren Optionen finden Sie in den Abschnitten *Anpassen von Graphen* und *Anpassen von Diagrammen* in diesem Kapitel.

Kurven- und XY-Graphen und -Diagramme verfügen über andere Optionen als Intensitäts-, digitale und 3D-Graphen und -Diagramme. Weitere Informationen zu den Optionen für Intensitäts-, digitale und 3D-Graphen sowie -Diagramme finden Sie in den Abschnitten *Intensitätsgraphen und -diagramme*, *3D-Graphen* und *Digitale Graphen* in diesem Kapitel.

Mehrere X- und Y-Achsen in Graphen und Diagrammen

Alle Graphen und Diagramme unterstützen mehrere X- und Y-Achsen. Mit Hilfe von mehreren Achsen in einem Graphen oder Diagramm können Sie mehrere Kurven darstellen, die verschiedene X- beziehungsweise Y-Achsen verwenden. Klicken Sie mit der rechten Maustaste auf die X- oder Y-Achse des Kurvengraphen oder -diagramms, und wählen Sie **Maßstab duplizieren** aus dem Kontextmenü, um mehrere X- oder Y-Achsen zum Graphen oder Diagramm hinzuzufügen.

Kantengeglättete Liniendarstellungen für Graphen und Diagramme

Sie können die Darstellungen von Linien in Diagrammen und Graphen mit Hilfe von kantengeglätteten Linien verbessern. Wenn Sie die Darstellung von kantengeglätteten Linien aktivieren, werden Linien geglättet dargestellt. Bei der Darstellung von kantengeglätteten Linien werden weder Linienbreite, Liniensstil noch Punktstil und so weiter geändert.

Zum Aktivieren von kantengeglätteten Liniendarstellungen klicken Sie mit der rechten Maustaste auf die Diagrammlegende und wählen aus dem Kontextmenü **Kantengeglättet**. Wird die Diagrammlegende nicht angezeigt, klicken Sie mit der rechten Maustaste auf das Diagramm oder den Graphen und wählen aus dem Kontextmenü **Sichtbare Objekte» Legende der Kurve**.



Hinweis Da die Darstellung von kantengeglätteten Linien sehr rechenintensiv sein kann, kann die Verwendung der kantengeglätteten Liniendarstellung die Systemleistung verringern.

Anpassen der Darstellung von Graphen und Diagrammen

Sie können die Darstellung von Graphen und Diagrammen durch Ein- oder Ausblenden von Optionen anpassen. Klicken Sie mit der rechten Maustaste auf den Graphen oder das Diagramm und wählen aus dem Kontextmenü **Sichtbare Objekte**, um die folgenden Optionen ein- oder auszublenden:

- **Legende der Kurve:** Definiert Farbe und Stil der Kurven. Verändern Sie die Größe der Legende, um mehrere Kurven anzuzeigen.
- **Achsenlegende:** Definiert Achsenbeschriftungen und konfiguriert Achseneigenschaften.
- **Graphen-Palette:** Ändert Skalierung und Formatierung während der Ausführung eines VIs.
- **X-Achse** und **Y-Achse:** Formatiert die X- und Y-Achsen. Weitere Informationen zu Achsen finden Sie in dem Abschnitt [Achsoptionen](#) dieses Kapitels.
- **Cursor-Legende** (nur bei Graphen): Zeigt an der definierten Punktkoordinate eine Markierung an. Sie können in einem Graphen mehrere Cursor anzeigen.
- **Bildlaufleiste** (nur bei Diagrammen): Blättert durch die Daten im Diagramm. Mit Hilfe der Bildlaufleiste können Sie Daten anzeigen, die zum aktuellen Zeitpunkt nicht vom Diagramm angezeigt werden.

Anpassen von Graphen

Sie können das Verhalten von Graphen-Cursor, Achsoptionen und Graphen-Achsen ändern. Abbildung 11-1 veranschaulicht die Elemente eines Graphen.

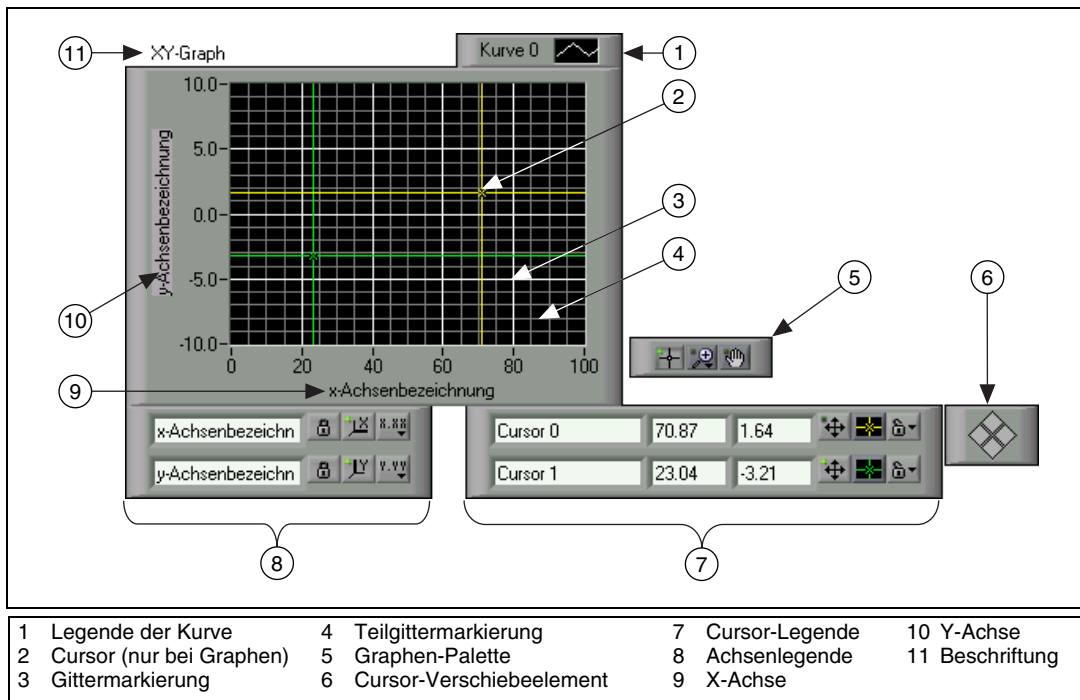


Abbildung 11-1. Graphen-Elemente

Sie können die meisten in der obigen Legende dargestellten Objekte hinzufügen, indem Sie auf den Graphen klicken, aus dem Kontextmenü **Sichtbare Objekte** und dann das entsprechende Element auswählen.

Graphen-Cursor

Mit Hilfe von Cursor können Sie in Graphen den genauen Wert eines Punktes einer Kurve messen. Der Cursor-Wert wird in der Cursor-Legende angezeigt. Sie fügen einen Cursor zu einem Graphen hinzu, indem Sie mit der rechten Maustaste auf den Graphen klicken, aus dem Kontextmenü **Sichtbare Objekte»Cursor-Legende** auswählen und auf eine beliebige Stelle in einer Zeile der Cursor-Legende klicken, um einen Cursor zu aktivieren. Mit dem Positionierwerkzeug können Sie die Cursor-Legende erweitern und mehrere Cursor hinzufügen.

Sie können Cursor und eine Cursor-Anzeige auf allen Graphen einfügen und den Cursor in der Darstellung beschriften. Sie können einen Cursor in einer Darstellung verankern und gleichzeitig mehrere Cursor verschieben. Ein Graph kann eine beliebige Anzahl von Cursor besitzen.

Ein Beispiel zum Lesen von Cursor-Werten und zum programmatischen Vergrößern oder Verkleinern eines Graphen mit Hilfe von Cursor finden Sie in „Zoom Graph.vi“ in der Datei `examples\general\graphs\zoom.llb`.

Achsoptionen

In Graphen können die horizontalen und vertikalen Achsen automatisch angepasst werden, um die darzustellenden Daten anzuzeigen. Dieses Verhalten wird automatische Skalierung genannt. Sie schalten die automatische Skalierung ein oder aus, indem Sie mit der rechten Maustaste auf den Graphen klicken und aus dem Kontextmenü **X-Achse»Autom. Skalierung X** oder **Y-Achse»Autom. Skalierung Y** auswählen. Standardmäßig ist die automatische Skalierung für Graphen aktiviert. Die automatische Skalierung kann jedoch die Leistung beeinträchtigen.

Mit dem Bedienwerkzeug oder dem Beschriftungswerkzeug können Sie die horizontale oder vertikale Achse direkt ändern.

Achsenlegende für Kurvengraphen

Mit der Achsenlegende können Sie Achsenbeschriftungen definieren und Achseneigenschaften konfigurieren.

Klicken Sie unter Verwendung des Bedienwerkzeugs auf die links angezeigte Schaltfläche **Achsenstil**, um Format, Genauigkeit und Achsen zu konfigurieren.



Mit der auf der linken Seite angezeigten Schaltfläche **Scale Lock** können Sie die automatische Skalierung für jede Achse sowie die Sichtbarkeit von Achsen, Achsenbeschriftungen und Kurven umschalten, und Achsenbeschriftungen, Gitter, Gitterlinien und Gitterfarben formatieren.



Formatieren der Graphen-Achsen

Sie können die Achsen eines Graphen so formatieren, dass Zeit beziehungsweise Amplitude absolut oder relativ dargestellt werden. Verwenden Sie das absolute Zeitformat, um für eine Achse Zeit und/oder Datum anzuzeigen. Wenn LabVIEW kein Datum anzeigen soll, verwenden Sie das relative Zeitformat. Zum Auswählen des absoluten oder relativen Zeitformats klicken Sie mit der rechten Maustaste auf den Graphen, wählen die Achse aus, die geändert werden soll, und wählen aus dem Kontextmenü **Formatieren**. In dem angezeigten Dialogfeld **Formatieren** können Sie verschiedene Eigenschaften der Graphen-Achse festlegen. Standardmäßig

werden auf der X-Achse die Zeit und auf der Y-Achse die Amplitude dargestellt.

Glätten von Grafiken

Klicken Sie mit der rechten Maustaste auf den Graphen und wählen aus dem Kontextmenü **Fortgeschritten»Grafik glätten**, um einen Puffer außerhalb des Bildschirms zu verwenden und das Blinken der Darstellung zu minimieren. Die Verwendung von **Grafik glätten** kann abhängig von dem verwendeten Computer und Grafiksystem die Leistung beeinträchtigen.

Anpassen von Diagrammen

Im Gegensatz zu Graphen, bei denen ein ganzer Signalverlauf angezeigt wird, der die bereits gespeicherten Daten überschreibt, werden Diagramme periodisch aktualisiert und eine Historie der zuvor gespeicherten Daten verwaltet.

Sie können die Kurven- und Intensitätsdiagramme in der Palette **Elemente»Graph** so anpassen, dass sie mit Ihren Datenanzeigeanforderungen übereinstimmen oder mehr Informationen anzeigen. Für Diagramme stehen die folgenden Optionen zur Verfügung: Bildlaufleiste, Legende, Palette, digitale Anzeige und Darstellung der Achsen in Bezug auf die Zeit. Sie können das Verhalten von Historienlänge, Aktualisierungsmodus und Kurvanzeige ändern.

Historienlänge

LabVIEW speichert Datenwerte, die bereits zu dem Diagramm hinzugefügt wurden, in einem Puffer – der sogenannten Diagrammhistorie. Die Standardgröße für den Puffer der Diagrammhistorie beträgt 1.024 Datenwerte. Sie können den Historienpuffer konfigurieren, indem Sie mit der rechten Maustaste auf das Diagramm klicken und aus dem Kontextmenü **Historienlänge** auswählen. Früher erfasste Daten können mit der Diagrammbildlaufleiste angezeigt werden. Klicken Sie mit der rechten Maustaste auf das Diagramm und wählen aus dem Kontextmenü **Sichtbare Objekte»Bildlaufleiste**, um eine Bildlaufleiste anzuzeigen.

Aktualisierungsmodi von Diagrammen

Diagramme verwenden drei unterschiedliche Modi zur fortlaufenden Anzeigen von Daten. Klicken Sie mit der rechten Maustaste auf das Diagramm und wählen aus dem Kontextmenü **Fortgeschritten»Aktualisierungsmodus**. Wählen Sie **Streifendiagramm**,

Oszilloskopdiagramm oder **Laufdiagramm**. Der Standardmodus ist **Streifendiagramm**. In einem Streifendiagramm werden die Daten im Diagramm kontinuierlich fortlaufend von links nach rechts angezeigt. In einem Oszilloskopdiagramm wird ein Datenobjekt, wie zum Beispiel ein Puls oder Signal, über das Diagramm von links nach rechts periodisch dargestellt. Ein Laufdiagramm ähnelt einem EKG-Diagramm. Es funktioniert ähnlich wie ein Oszilloskopdiagramm, mit der Ausnahme, dass die alten Daten auf der rechten Seite und die neuen Daten getrennt durch eine vertikale Linie auf der linken Seite angezeigt werden.

Überlagerte Kurven und Stapelplots

Sie können in einem Diagramm mehrere Kurven unter Verwendung einer einzigen vertikalen Achse darstellen, den sogenannten überlagerten Kurven, oder unter Verwendung mehrerer vertikaler Achsen, den sogenannten Stapelplots. Abbildung 11-2 zeigt Beispiele für überlagerte Kurven und Stapelplots.

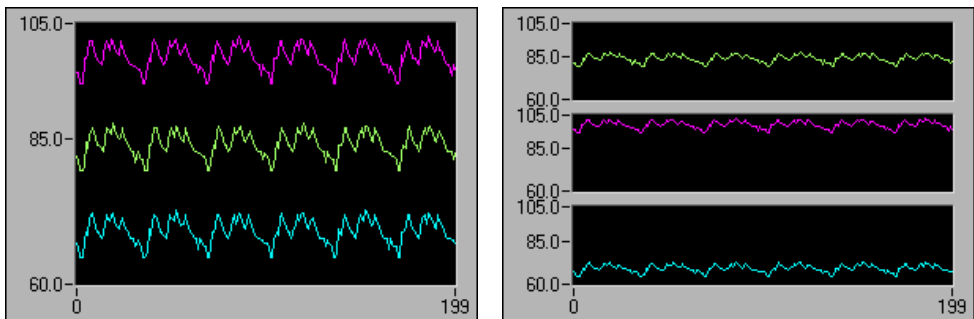


Abbildung 11-2. Diagramme mit überlagerten Kurven und Stapelplots

Beispiele für die verschiedenen Diagramme und die jeweils akzeptierten Datentypen finden Sie in „Charts.vi“ in `examples\general\graphs\charts.llb`.

Kurven- und XY-Graphen

In Kurvengraphen werden in gleichen Abständen abgetastete Messungen angezeigt. XY-Graphen zeigen eine beliebige Wertemenge an, die gleichmäßig abgetastet sein kann oder nicht. Abbildung 11-3 zeigt Beispiele eines Kurvengraphs und eines XY-Graphs.

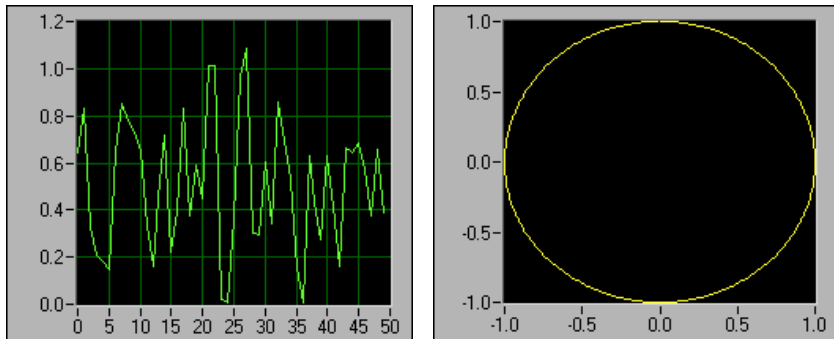


Abbildung 11-3. Kurven- und XY-Graphen

Der Kurvengraph stellt nur einwertige Funktionen dar, wie in $y = f(x)$, wobei die Werte gleichmäßig über die x -Achse verteilt sind, wie zum Beispiel bei erfassten zeitabhängigen Signalen. Beim XY-Graph handelt es sich um ein kartesisches Objekt zur grafischen Darstellung für allgemeine Zwecke, das mehrwertige Funktionen wie zum Beispiel Kreisformen oder Signalverläufe mit einer variablen Zeitbasis darstellt. Beide Graphen können Kurven mit einer beliebigen Anzahl von Werten anzeigen.

Beide Graphentypen akzeptieren mehrere Datentypen, sodass der Aufwand für die Bearbeitung der Daten vor der Anzeige minimiert wird.

Datentypen für Kurvengraphen mit einer Kurve

Der Kurvengraph akzeptiert bei Kurvengraphen mit einer Kurve zwei Datentypen.

Der Graph akzeptiert ein einzelnes Array mit Werten, interpretiert die Daten als Werte im Graph und erhöht den x -Index beginnend bei $x = 0$ um eins. Der Graph akzeptiert außerdem einen Cluster mit einem x -Anfangswert, einem Δx -Wert und einem Array von y -Daten.

Beispiele für die Datentypen, die Kurvengraphen für eine Kurve akzeptieren, finden Sie im VI Waveform Graph in `examples\general\graphs\gengraph.llb`.

Kurvengraph mit mehreren Kurven

Ein Kurvengraph mit mehreren Kurven akzeptiert ein 2D-Array mit Werten, wobei jede Zeile des Arrays aus einer einzelnen Kurve besteht. Der Graph interpretiert die Daten als Werte im Graph und erhöht den x -Index beginnend bei $x = 0$ um eins. Verbinden Sie ein 2D-Array mit dem Graph, klicken mit der rechten Maustaste auf den Graph und deaktivieren Sie **Array transponieren** aus dem Kontextmenü, um jede Spalte in dem Array als Kurve zu behandeln. Dies ist insbesondere dann nützlich, wenn Sie mehrere Kanäle eines DAQ-Gerätes abtasten, da das Gerät die Daten als 2D-Arrays zurückgibt, wobei jeder Kanal als eigenständige Spalte gespeichert wird. Ein Beispiel für einen Graph, der diesen Datentyp akzeptiert, finden Sie im Graph (Y) Multi Plot 1 im VI Waveform Graph in der Datei `examples\general\graphs\gengraph.11b`.

Ein Kurvengraph mit mehreren Kurven akzeptiert außerdem einen Cluster mit einem x -Wert, einem Δx -Wert und einem 2D-Array von y -Daten. Der Graph interpretiert die y -Daten als Werte im Graph und erhöht den x -Index beginnend bei $x = 0$ um Δx . Dieser Datentyp eignet sich zum Anzeigen mehrerer Signale, die alle mit derselben regelmäßigen Rate abgetastet werden. Ein Beispiel für einen Graph, der diesen Datentyp akzeptiert, finden Sie im Graph „(Xo, dX, Y) Multi Plot 3“ im VI Waveform Graph in der Datei `examples\general\graphs\gengraph.11b`.

Ein Kurvengraph mit mehreren Kurven akzeptiert ein Kurvenarray, wobei das Array Cluster enthält. Jeder Cluster enthält ein Wertearray, das die y -Daten enthält. Das innere Array beschreibt die Werte einer Kurve, das äußere Array enthält einen Cluster für jede Kurve. Abbildung 11-4 zeigt dieses Array mit y -Clustern.



Abbildung 11-4. Array mit Clustern von y -Werten

Verwenden Sie einen Kurvengraphen mit mehreren Kurven anstelle eines 2D-Arrays, wenn die Anzahl der Elemente in den einzelnen Kurven nicht übereinstimmt. Verwenden Sie diese Datenstruktur anstelle eines 2D-Arrays, wenn Sie beispielsweise Daten mehrerer Kanäle abtasten und für jeden Kanal unterschiedliche Zeitintervalle verwenden, da jede Zeile eines 2D-Arrays die gleiche Anzahl von Elementen enthalten muss. Die Anzahl der Elemente in den inneren Arrays eines Arrays von Clustern kann

variieren. Ein Beispiel für einen Graph, der diesen Datentyp akzeptiert, finden Sie im Graph „(Y) Multi Plot 2“ im VI Waveform Graph in der Datei `examples\general\graphs\gengraph.llb`.

Ein Kurvengraph für mehrere Kurven akzeptiert einen Cluster mit einem x -Anfangswert, einem Δx -Wert und einem Array, das Cluster enthält. Jeder Cluster enthält ein Wertearray, das die y -Daten enthält. Mit der Funktion „Elemente bündeln“ können Sie die Arrays in Cluster bündeln, mit der Funktion „Array erstellen“ können Sie aus den resultierenden Clustern ein Array erstellen. Sie können auch die Funktion „Cluster-Array erstellen“ verwenden, die Arrays von Clustern erstellt, welche die angegebenen Eingänge enthalten. Ein Beispiel für einen Graph, der diesen Datentyp akzeptiert, finden Sie im Graph „(Xo, dX, Y) Multi Plot 2“ im VI Waveform Graph in der Datei `examples\general\graphs\gengraph.llb`.

Ein Kurvengraph für mehrere Kurven akzeptiert ein Array von Clustern mit einem x -Wert, einem Δx -Wert und einem Array von y -Daten. Dieser Datentyp bildet die universellste Form eines Kurvengraphen für mehrere Kurven, da Sie Anfangswert und Schrittweite für die x -Achse jeder Kurve eindeutig festlegen können. Ein Beispiel für einen Graph, der diesen Datentyp akzeptiert, finden Sie im Graph „(Xo, dX, Y) Multi Plot 1“ im VI Waveform Graph in der Datei `examples\general\graphs\gengraph.llb`.

Datentypen für XY-Graphen mit einer Kurve

Der XY-Graph für eine Kurve akzeptiert einen Cluster, der ein x -Array und ein y -Array enthält. Der XY-Graph akzeptiert außerdem ein Wertearray, in dem ein Wert aus einem Cluster besteht, der einen x -Wert und einen y -Wert enthält.

Beispiele für die Datentypen, die XY-Graphen für eine Kurve akzeptieren, finden Sie im XY-Graph-VI in `examples\general\graphs\gengraph.llb`.

Datentypen für XY-Graphen mit mehreren Kurven

Der XY-Graph mit mehreren Kurven akzeptiert ein Kurvenarray, in dem eine Kurve aus einem Cluster besteht, der ein x -Array und ein y -Array enthält. Der XY-Graph für mehrere Kurven akzeptiert außerdem ein Array mit Clustern von Kurven, wobei eine Kurve aus einem Wertearray besteht. Ein Wert besteht aus einem Cluster, der einen x -Wert und einen y -Wert enthält.

Beispiele für die Datentypen, die XY-Graphen für mehrere Kurven akzeptieren, finden Sie im VI XYGraph in `examples\general\graphs\gengraph.llb`.

Kurvendiagramme

Bei einem Kurvendiagramm handelt es sich um einen speziellen Typ eines numerischen Anzeigeelements, das eine oder mehrere Kurven anzeigt. Beispiele für Kurvendiagramme finden Sie in `examples\general\graphs\charts.llb`.

Sie können an Diagramme einen Wert oder gleichzeitig mehrere Werte übergeben. Wie beim Kurvengraph interpretiert LabVIEW die Daten als Werte im Graph und erhöht den x -Index beginnend bei $x = 0$ um eins. Das Diagramm betrachtet diese Eingangswerte als neue Daten für eine einzelne Kurve.

Das Diagramm wird weniger häufig aktualisiert, wenn gleichzeitig mehrere Werte übergeben werden.

Um Daten für mehrere Kurven in einem Kurvendiagramm zu übergeben, können Sie die Daten in einem Cluster skalarer Zahlen bündeln, wobei jede Zahl einen einzelnen Wert für die jeweilige Kurve darstellt.

Wenn Sie mehrere Werte für Kurven in einem einzigen Aktualisierungsvorgang übergeben möchten, verbinden Sie ein Array mit Clustern von Zahlen mit dem Diagramm. Jede Zahl stellt einen einzelnen y -Wert für die einzelnen Kurven dar.

Wenn die Anzahl der anzuzeigenden Kurven erst zur Laufzeit ermittelt werden kann oder wenn Sie in einem einzigen Aktualisierungsvorgang mehrere Werte für mehrere Kurven übergeben möchten, verbinden Sie ein 2D-Array von Daten mit dem Diagramm. Wie beim Kurvengraph interpretieren Kurvendiagramme Zeilen standardmäßig als neue Daten für die einzelnen Kurven. Verbinden Sie einen 2D-Array-Datentyp mit dem Diagramm, klicken mit der rechten Maustaste auf das Diagramm und wählen aus dem Kontextmenü **Array transponieren**, um die Spalten in dem Array für die einzelnen Kurven als neue Daten zu verwenden.

Intensitätsgraphen und -diagramme

Mit Intensitätsgraphen und -diagrammen können Sie 3D-Daten zweidimensional darstellen, indem Sie auf einer kartesischen Ebene Farbböcke anzeigen. Intensitätsgraph und -diagramme akzeptieren ein 2D-Array von Zahlen. Jede Zahl im Array stellt eine bestimmte Farbe dar. Die Indizes der Elemente im 2D-Array legen die Position der Farben in der Darstellung fest. Abbildung 11-5 zeigt die Funktionsweise des Intensitätsgraphen.

Die Datenzeilen werden im Graph oder Diagramm zur Anzeige als neue Spalten übergeben. Wenn die Zeilen in der Anzeige als Zeilen angezeigt werden sollen, verbinden Sie ein 2D-Array mit dem Graph oder Diagramm, klicken mit der rechten Maustaste auf den Graph oder das Diagramm und wählen aus dem Kontextmenü **Array transponieren**.

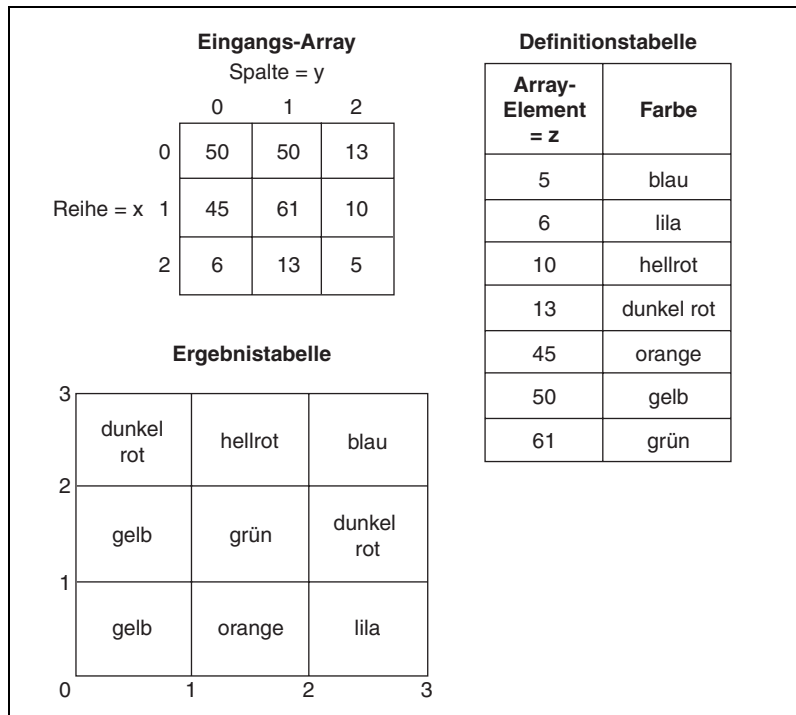


Abbildung 11-5. Farbzunordnung bei Intensitätsgraphen

Die Array-Indizes entsprechen dem unteren linken Eckpunkt des Farbblocks. Der Farbblock verfügt über einen Einheitsbereich. Dabei handelt es sich um den Bereich zwischen den beiden Punkten, wie durch

die Array-Indizes festgelegt. Ein Intensitätsdiagramm oder -graph kann bis zu 256 diskrete Farben anzeigen. Um in einem Intensitätsdiagramm umfangreiche Daten anzuzeigen, muss ausreichend Speicher zur Verfügung stehen.

Wenn Sie einen Datenblock auf einem Intensitätsdiagramm dargestellt haben, verschiebt sich der Ursprung der kartesischen Ebene auf die rechte Seite des letzten Datenblocks. Werden in dem Diagramm neue Daten verarbeitet, werden die neuen Daten rechts von den alten Daten angezeigt. Ist eine Diagrammanzeige voll, laufen die ältesten Daten auf der linken Seite aus dem Diagramm. Dieses Verhalten ähnelt dem Verhalten von Streifendiagrammen. Weitere Informationen zu diesen Diagrammen finden Sie im Abschnitt [Diagrammaktualisierungsmodi](#) dieses Kapitels.

Beispiele für Intensitätsgraphen und -diagramme finden Sie in `examples\general\graphs\intgraph.llb`.

Farbzuordnung

Sie können die Farbzuordnung für Intensitätsgraphen und -diagramme genauso interaktiv festlegen, wie Sie die Farben für einen numerischen Farbverlaufsbalken definieren. Weitere Informationen zu Farbrampen finden Sie in Abschnitt [Farbrampen](#) des Kapitels 4, [Erstellen des Frontpanels](#).

Sie können die Farbzuordnung für Intensitätsgraph und -diagramm programmatisch festlegen, indem Sie den Eigenschaftsknoten auf zweierlei Weise verwenden. Normalerweise legen Sie die Zuordnung von Wert und Farbe im Eigenschaftsknoten fest. Bei dieser Methode legen Sie die Eigenschaft **Z-Achse: Marker-Werte** fest. Diese Eigenschaft besteht aus einem Array mit Clustern, in dem jeder Cluster einen numerischen Grenzwert und die für den betreffenden Wert anzuzeigende Farbe enthält. Wenn Sie die Farbzuordnung auf diese Weise festlegen, können Sie mit der Eigenschaft **Z-Achse: High Colour (16-Bit)** eine Farbe für eine Bereichsüberschreitung und mit der Eigenschaft **Z-Achse: Untere Farbe** eine Farbe für eine Bereichsunterschreitung festlegen. Intensitätsgraph und -diagramm sind auf insgesamt 254 Farben beschränkt. Zusammen mit den Farben für eine Bereichsüberschreitung oder -unterschreitung ergeben sich damit insgesamt 256 Farben. Wenn Sie mehr als 254 Farben angeben, erstellt der Intensitätsgraph beziehungsweise das Intensitätsdiagramm die Tabelle aus 254 Farben durch Interpolation der festgelegten Farben.

Wenn Sie im Intensitätsgraph eine Bitmap anzeigen, legen Sie die Farbtabelle mit Hilfe der Eigenschaft **Farbtabelle** fest. Bei dieser Methode können Sie ein Array mit bis zu 256 Farben festlegen. Die an das

Diagramm übergebenen Daten werden anhand der Farbskala des Intensitätsdiagramms den Indizes in dieser Farbtabelle zugeordnet. Reicht die Farbskala von 0 bis 100, wird der Datenwert 0 dem Index 1 zugeordnet und der Wert 100 dem Index 254, wobei die dazwischenliegenden Werte zwischen 1 und 254 interpoliert werden. Alle Werte unter 0 werden der Farbe für eine Bereichsunterschreitung (Index 0) zugeordnet und alle Werte über 100 der Farbe für eine Bereichsüberschreitung (Index 255).



Hinweis Die Farben, die im Intensitätsgraph oder -diagramm angezeigt werden sollen, sind auf diejenigen Farben und die Anzahl der Farben begrenzt, die von der Grafikkarte angezeigt werden können. Darüber hinaus sind Sie auf die Anzahl der für die Anzeige zugeordneten Farben beschränkt.

Optionen für Intensitätsdiagramme

Das Intensitätsdiagramm verwendet viele der optionalen Bestandteile der Kurvendiagramme, die Sie ein- oder ausblenden können, indem Sie mit der rechten Maustaste auf das Diagramm klicken und **Sichtbare Objekte** auswählen. Da das Intensitätsdiagramm als dritte Dimension eine Farbe enthält, definiert darüber hinaus eine Skala (ähnlich wie beim Farbverlaufsbalken) den Bereich und die Zuordnungen von Werten und Farben.

Wie bei Kurvendiagrammen speichert das Intensitätsdiagramm den Datenverlauf, oder Puffer, der vorherigen Aktualisierungen. Sie können diesen Puffer konfigurieren, indem Sie mit der rechten Maustaste auf das Diagramm klicken und aus dem Kontextmenü **Historienlänge** auswählen. Die Standardgröße für ein Intensitätsdiagramm beträgt 128 Datenwerte. Die Anzeige eines Intensitätsdiagramms kann sehr speicherintensiv sein. Die Anzeige eines Diagramms mit einfacher Genauigkeit, einer Historie von 512 Werten und 128 y-Werten erfordert beispielsweise $512 * 128 * 4$ Bytes (Größe einer Zahl mit einfacher Genauigkeit) oder 256 KB.

Optionen für Intensitätsgraphen

Der Intensitätsgraph funktioniert genauso wie das Intensitätsdiagramm, mit der Ausnahme, dass vorhergehende Daten nicht gespeichert werden und keine Aktualisierungsmodi enthalten sind. Immer, wenn neue Daten an einen Intensitätsgraph übergeben werden, ersetzen diese neuen Daten die alten.

Der Intensitätsgraph kann wie andere Graphen über Cursor verfügen. Jeder Cursor zeigt die x -, y - und z -Werte für einen bestimmten Punkt im Graph an.

Digitale Graphen

Mit dem Kurvengraph können Sie digitale Daten anzeigen, insbesondere dann, wenn Sie mit Zeitdiagrammen oder Logikanalysatoren arbeiten. Informationen zum Erfassen digitaler Daten finden Sie in Kapitel 8, *Digital-I/O*, des *LabVIEW Measurement Manual*.

Der Kurvengraph in Abbildung 11-6 zeigt ein Array von sechs vorzeichenlosen 8-Bit-Integern. Der Graph stellt diese ganzen Zahlen in einem Binärformat dar, wobei jedes Bit im Graph eine Kurve darstellt.

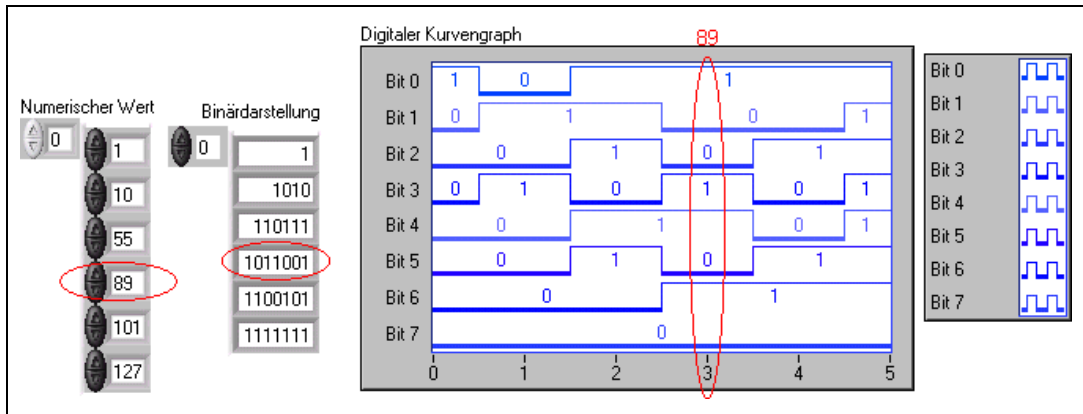


Abbildung 11-6. Digitale Grafikdarstellung von ganzen Zahlen

Die Spalte **Binärdarstellung** in Abbildung 11-6 entspricht der ersten Kurve, Bit 0, im digitalen Graph. Die Spalte **Numerischer Wert** entspricht der zweiten Kurve, Bit 1, und so weiter. Die Zahl 89 benötigt beispielsweise sieben Bit Speicher. Die binäre Grafikdarstellung der Zahl 89 wird in Punkt 3 des Graphen angezeigt.

Zum Erstellen eines VIs, das digitale Daten anzeigt, verwenden Sie die Funktion „Elemente bündeln“, um Triggerzeit (X0), Δx (dX), das Zahlenarray und die Anzahl der Anschlüsse zusammenzufassen. Siehe dazu Abbildung 11-7.

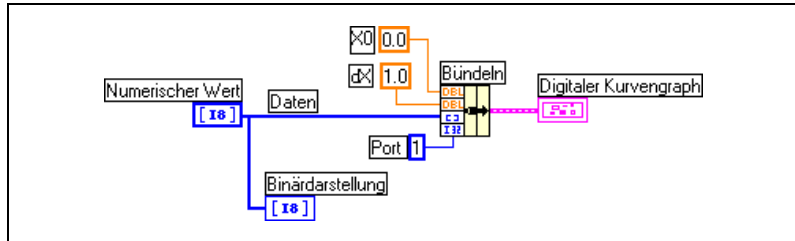


Abbildung 11-7. Verwenden der Funktion „Elemente bündeln“ bei digitalen Graphen

Port legt die Anzahl der Datenelemente fest, die als eine einzelne ganze Zahl behandelt werden sollen. Bei 8-Bit-Daten, die als 8-Bit-Integer dargestellt werden sollen, beträgt die Anzahl der Ports 1. Bei 32-Bit-Daten, die als 32-Bit-Integer dargestellt werden sollen, beträgt die Anzahl der Ports ebenfalls 1. Bei drei Quellen mit 8-Bit-Daten müssen 24 Bits als eine ganze Zahl dargestellt werden. Verwenden Sie in diesem Fall die Funktion „1D-Arrays überführen“, um die 8-Bit-Daten zu überführen, und legen Sie die Anzahl der Ports, wie in Abbildung 11-8 gezeigt, als 3 fest.

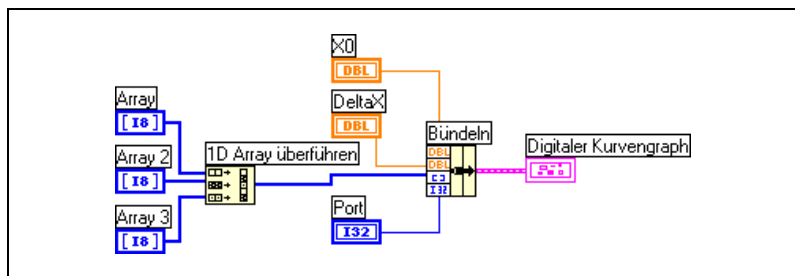


Abbildung 11-8. Verwenden der Funktion „1D-Arrays überführen“ bei digitalen Graphen

Geben Sie in Abbildung 11-8 die Zahl 3 in **Port** ein, um alle drei Arrays der 8-Bit-Daten grafisch darzustellen. Geben Sie 2 in **Port** ein, um zwei Arrays mit 8-Bit-Daten grafisch darzustellen. Geben Sie 5 in **Port** ein, um fünf Arrays mit 8-Bit-Daten grafisch darzustellen. Dies führt dazu, dass 16 der 40 Bits keine Werte enthalten.

Maskieren von Daten

Das VI in Abbildung 11-6 erzeugt einen Graph, in dem jede Kurve ein Bit der Daten darstellt. Sie können vor der Anzeige im Graph auch Daten-Bits auswählen, neu anordnen und kombinieren. Auswählen, Anordnen und Kombinieren von Bits wird Maskieren der Daten genannt.

Mit Hilfe einer Maske können Sie die Kurven von zwei oder mehr verschiedenen Bits kombinieren und auf einer einzelnen Kurve darstellen. Bei einem Array von 8-Bit-Integern können bis zu 8 Bits in einer einzigen Kurve dargestellt werden. Bei einem Array von 16-Bit-Integern können bis zu 16 Bits in einer einzigen Kurve angezeigt werden und so weiter. Sie können dasselbe Bit auch zweimal oder mehrmals in einer einzelnen Kurve darstellen. Weitere Informationen zum Maskieren von digitalen Daten finden Sie in Anhang D, *Maskieren von digitalen Daten*.

3D-Graphen



Hinweis Die Bedienelemente für 3D-Graphen stehen unter Windows nur im Full Development System und Professional Development System von LabVIEW zur Verfügung.

Bei vielen in der Realität vorkommenden Datenmengen, wie zum Beispiel einer Oberflächentemperaturverteilung, verknüpften Zeit-/Frequenzanalysen und der Bewegung eines Flugzeugs, müssen Daten in drei Dimensionen visualisiert werden. Mit den 3D-Graphen können Sie dreidimensionale Daten visualisieren und die Anzeige der Daten ändern, indem Sie die Eigenschaften des 3D-Graphs ändern.

Es stehen die folgenden 3D-Graphen zur Verfügung:

- **3D-Oberflächengraph:** Zeichnet eine Oberfläche im 3D-Raum. Wenn Sie dieses Bedienelement auf dem Frontpanel ablegen, wird es von LabVIEW mit einem Sub-VI verbunden, das die Daten erhält, welche die Oberfläche darstellen.
- **Parametrischer 3D-Graph:** Zeichnet eine komplexe Oberfläche im 3D-Raum. Wenn Sie dieses Bedienelement auf dem Frontpanel ablegen, wird es von LabVIEW mit einem Sub-VI verbunden, das die Daten erhält, welche die Oberfläche darstellen.
- **3D-Kurvengraph:** Zeichnet eine Linie im 3D-Raum. Wenn Sie dieses Bedienelement auf dem Frontpanel ablegen, wird es von LabVIEW mit einem Sub-VI verbunden, das die Daten erhält, welche die Linie darstellen.

Sie können die 3D-Graphen zusammen mit den 3D-Graph-VIs verwenden, die sich in der Funktionenpalette unter **Grafiken & Sound» 3D-Graph-Eigenschaften** befinden, um Kurven und Oberflächen darzustellen. Eine Kurve enthält einzelne Punkte im Graph, wobei jeder Punkt eine x -, y - und z -Koordinate besitzt. Das VI verbindet diese Punkte dann zu einer Linie. Eine Kurve eignet sich ideal für die Visualisierung des Weges eines sich bewegenden Objekts, wie zum Beispiel des Flugwegs eines Flugzeugs.

Die 3D-Graphen verwenden ActiveX-Technologie sowie VIs, die der 3D-Darstellung dienen. Sie können die Eigenschaften der VIs, die sich in der Funktionenpalette unter **Grafiken & Sound» 3D-Graph-Eigenschaften** befinden, einstellen, um das Laufzeitverhalten zu ändern, wie zum Beispiel Einstellen der Basis-, Achsen-, Raster- und Projektionseigenschaften.

Ein Oberflächengraph verwendet x -, y - und z -Daten, um die Punkte im Graph darzustellen. Der Oberflächengraph verbindet anschließend diese Punkte und bildet eine dreidimensionale Oberflächenansicht der Daten. Sie können einen Oberflächengraph beispielsweise zur topologischen Darstellung verwenden.

Bei Auswahl eines 3D-Graphen fügt LabVIEW auf dem Frontpanel einen ActiveX-Container ein, der ein Bedienelement für einen 3D-Graphen enthält. LabVIEW erstellt im Blockdiagramm außerdem eine Referenz auf diesen 3D-Graphen. LabVIEW verbindet diese Referenz mit einem der drei 3D-Graph-VIs.

Der Datentyp „Signalverlauf“

Der Datentyp „Signalverlauf“ enthält Daten, Anfangszeit und Δx eines Signalverlaufs. Sie können Signalverläufe mit der Funktion „Signalverlauf erstellen“ erstellen, die sich in der Funktionenpalette unter **Signalverlauf** befindet. Viele VIs und Funktionen, mit denen Sie Signalverläufe erfassen oder analysieren, akzeptieren standardmäßig den Datentyp „Signalverlauf“ oder geben ihn zurück. Wenn Sie den Datentyp „Signalverlauf“ mit einem Kurvengraph oder -diagramm verbinden, stellt der Graph beziehungsweise das Diagramm automatisch eine Kurve dar, die auf Daten, Anfangszeit und Δx dieses übergebenen Signalverlaufs basiert. Wenn Sie ein Array des Datentyps „Signalverlauf“ mit einem Kurvengraph oder -diagramm verbinden, stellt der Graph beziehungsweise das Diagramm automatisch alle Signalverläufe dar. Informationen zur Verwendung des Datentyps „Signalverlauf“ finden Sie in Kapitel 6, *Analog Input*, des *LabVIEW Measurements Manual*.

VIs für Grafiken und Sound

Mit den VIs und Funktionen für Grafiken und Sound in der Palette **Funktionen»Grafiken & Sound** können Sie Grafiken und Sound in VIs anzeigen oder ändern.

Weitere Informationen ...

Weitere Informationen zum Verwenden von Grafiken und Sound finden Sie in der *LabVIEW-Hilfe*.

Verwenden des Bildanzeigeelements

Bei dem Bildanzeigeelement in der Palette **Elemente»Graph»Cntls** handelt es sich um ein Allzweckanzeigeelement zum Anzeigen von Bildern, die Linien, Kreise, Text und andere Arten von Grafikformen enthalten. Da das Bildanzeigeelement auf Pixel-Ebene gesteuert werden kann, können Sie nahezu jedes beliebige Grafikobjekt erstellen.

Das Bildanzeigeelement besitzt ein pixel-basiertes Koordinatensystem, in dem der Ursprung (0,0) das Pixel in der oberen linken Ecke des Bedienelements bildet. Die horizontale Komponente (x) einer Koordinate nimmt nach rechts hin zu, die vertikale Koordinate (y) nach unten hin.

Ist ein Bild zu groß, sodass es von dem Bildanzeigeelement nicht angezeigt werden kann, schneidet LabVIEW das Bild so zu, dass nur die Pixel angezeigt werden, die in den Anzeigebereich des Anzeigeelements passen. Mit dem Positionierwerkzeug können Sie die Größe des Anzeigeelements ändern und das VI erneut ausführen, damit das gesamte Bild angezeigt wird.

Mit der Eigenschaft **Bereichsgröße darstellen** können Sie die Größe des Bildbereichs des Bildanzeigeelements programmatisch feststellen und ändern. Mit der Eigenschaft **Abmessungen** können Sie die Größe des gesamten Bildanzeigeelements feststellen. Sie können das Anzeigeelement auch mit dem Positionierwerkzeug auswählen, indem Sie **Bearbeiten»Bedienelement anpassen** wählen und im Fenster **Bedienelemente** des

Bedienelement-Editors die Größe des gesamten Bildanzeigeelements und die Größe des Anzeigebereichs des Bildanzeigeelements bestimmen.



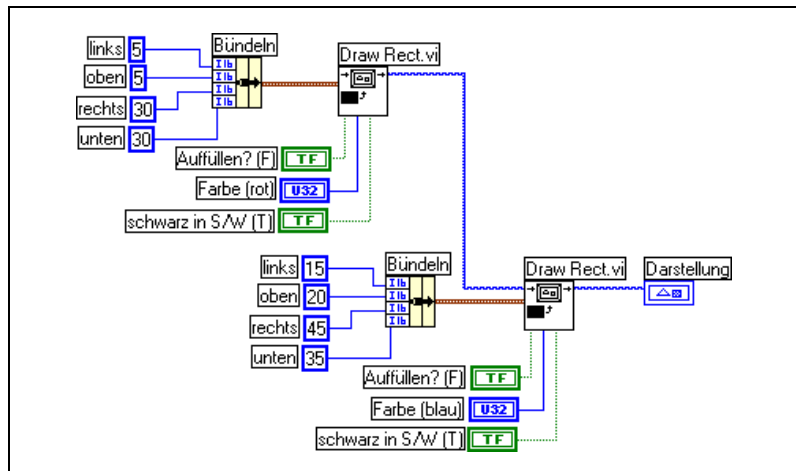
Wenn Sie auf dem Frontpanel ein Bildanzeigeelement einfügen, wird es als leerer rechteckiger Bereich angezeigt. Im Blockdiagramm wird ein dazugehöriger Anschluss auf der linken Seite angezeigt. Verwenden Sie die Bildfunktionen-VIs in der Palette **Funktionen»Grafiken & Sound»Bildfunktionen** anstelle einer Grafikanwendung, um die Funktionen von Grafiken in LabVIEW zu ändern oder zu erweitern.

Um ein Bild in einem Bildanzeigeelement anzuzeigen, müssen Sie mit den Bildfunktionen-VIs Zeichenanweisungen festlegen. Jedes VI übernimmt mehrere Eingaben, die eine Zeichenanweisung beschreiben. Basierend auf diesen Eingaben erstellt das VI eine kompakte Beschreibung dieser Festlegungen, die zur Anzeige an das Bildanzeigeelement übergeben werden.

Sie können die mit den Bildfunktionen-VIs erstellten Bilder nur an ein Bildanzeigeelement oder den Eingang **Bild** eines Bildfunktionen-VIs übergeben. LabVIEW zeichnet das Bild beim Aktualisieren des Bildanzeigeelements in einem geöffneten Frontpanel.

Jedes Bildfunktionen-VI verkettet seine Zeichenanweisung mit den vorherigen Zeichenanweisungen der Ein- und Ausgänge **Bild**. Ist der Eingang **Bild** nicht verbunden, gibt der Ausgang **Bild** einen leeren rechteckigen Darstellungsbereich zurück.

Das folgende Blockdiagramm verwendet „Draw Rect.vi“, um zwei überlappende Rechtecke zu zeichnen.



Das boolesche Bedienelement **schwarz in S/W (T)?** zeigt an, ob das Rechteck bei der Anzeige auf einem Monochrommonitor schwarz oder weiß angezeigt wird. Wenn Sie mehrere VIs erstellen, die das Bildanzeigeelement verwenden, und die VIs mit allen Monitoren ausgeführt werden sollen, müssen Sie das Bedienelement **schwarz in S/W (T)?** verwenden.

VIs für Bildplots

Verwenden Sie die Bildplot-VIs in der Palette **Funktionen»Grafiken & Sound»Bildplots**, um mit dem Bildanzeigeelement gebräuchliche Typen von Graphen zu erstellen. Diese Graphen umfassen ein Polardiagramm, eine Kurvengraphanzeige, ein Smith-Diagramm und einen Graphenmaßstab.

Die Bildplot-VIs verwenden systemnahe Zeichenfunktionen, um eine grafische Anzeige der Daten zu erstellen, und passen den Zeichencode an, um die Funktionalität zu erweitern. Diese grafischen Anzeigen verfügen nicht über eine so hohe Interaktivität wie die integrierten LabVIEW-Bedienelemente. Sie können mit ihnen jedoch Informationen auf eine solche Weise visualisieren, wie dies momentan in den integrierten Bedienelementen nicht möglich ist. Mit „Plot Waveform.vi“ können Sie einen Plot mit einer geringfügig anderen Funktion erstellen als der von LabVIEW-Graphen.

Verwenden von „Polar Plot.vi“ als Sub-VI

Mit „Polar Plot.vi“ können Sie bestimmte, zusammenhängende Quadranten eines polaren Graphen oder den gesamten Graphen zeichnen. Wie bei den integrierten LabVIEW-Graphen können Sie die Farbe der Komponenten angeben, Gitter einfügen, und Bereich und Format der Achsen festlegen.

„Polar Plot.vi“ bietet in einem einzelnen VI eine Vielzahl von Funktionen. Das VI enthält daher komplizierte Cluster als Eingabe. Sie können Standardwerte und benutzerspezifische Bedienelemente verwenden, um die Komplexität des VIs zu verringern. Kopieren Sie ein benutzerspezifisches Bedienelement aus „Polar Plot Demo.vi“ in der Datei `examples\picture\demos.llb` und fügen es in das Frontpanel ein, anstatt im Blockdiagramm eine eigene Cluster-Eingabe zu erstellen.

Verwenden von „Plot Waveform.vi“ als Sub-VI

Verwenden Sie „Plot Waveform.vi“, welches das Verhalten des integrierten Kurvengraphen emuliert, um Kurven mit verschiedenen Stilen, zum Beispiel Punkten, Verbindungslinien und Balken, zu zeichnen. Wie bei den integrierten LabVIEW-Kurvengraphen können Sie die Farbe der Komponenten angeben, Gitter einfügen, und Bereich und Format der Achsen festlegen.

„Plot Waveform.vi“ bietet in einem einzelnen VI eine Vielzahl von Funktionen. Das VI enthält daher komplizierte Cluster als Eingabe. Sie können Standardwerte und benutzerspezifische Bedienelemente verwenden, um die Komplexität der VIs zu verringern. Kopieren Sie ein allgemeines Bedienelement aus „Waveform and XY Plots.vi“ in der Datei `examples\picture\demos.llb` und fügen es in das Frontpanel ein, anstatt eine eigene Cluster-Eingabe zu erstellen.

„Plot XY.vi“ und „Plot Multi-XY.vi“ ähneln dem „Plot Waveform.vi“. Sie können mit verschiedenen Bedienelementen die Darstellung ändern, da die VIs für die XY-Darstellung drei Darstellungsarten bieten: zwei Streudiagrammartentypen und eine Darstellungsart, bei der bei jeder eindeutigen x-Position eine Linie gezeichnet wird, um die minimalen und maximalen y-Werte für den betreffenden x-Wert zu markieren.

Verwenden von „Smith Plot.vi“ als Sub-VI

Mit Hilfe von „Smith Plot.vi“ können Sie das Verhalten von Übertragungsleitungen untersuchen, wie zum Beispiel in der Telekommunikationsbranche. Bei einer Übertragungsleitung handelt es sich um ein Medium, über das Energie und Signale übertragen werden. Bei einer Übertragungsleitung kann es sich um ein Kabel handeln oder die Atmosphäre, über die das Signal übertragen wird. Übertragungsleitungen beeinflussen das übertragene Signal. Dieser Effekt, die sogenannte Impedanz der Übertragungsleitung, kann ein Wechselstromsignal dämpfen oder eine Phasenverschiebung bewirken.

Die Impedanz der Übertragungsleitung bildet ein Maß für Widerstand und Reaktanz der Leitung. Die Impedanz, z , wird in der Regel als komplexe Zahl in der Form $z = r + jx$ aufgeführt, wobei sowohl Widerstand (r) als auch Reaktanz (x) Komponenten darstellen.

Mit „Smith Plot.vi“ können Sie die Impedanzen von Übertragungsleitungen anzeigen. Das Diagramm besteht aus Kreisen mit konstantem Widerstand und konstanter Reaktanz.

Sie können eine gegebene Impedanz, $r + jx$ darstellen, indem Sie den Schnittpunkt des entsprechenden r -Kreises und x -Kreises suchen. Nach dem Darstellen der Impedanz können Sie „Smith Plot.vi“ als visuelle Hilfe verwenden, um die Impedanz anzupassen und den Reflektionskoeffizienten einer Übertragungsleitung zu berechnen.

„Smith Plot.vi“ bietet in einem einzelnen VI eine Vielzahl von Funktionen. Viele dieser VIs enthalten daher komplizierte Cluster als Eingabe. Sie können Standardwerte und benutzerspezifische Bedienelemente verwenden, um die Komplexität der VIs zu verringern. Kopieren Sie ein benutzerspezifisches Bedienelement aus den Beispielen für „Smith Plot.vi“ in der Datei `examples\picture\demos.llb` und fügen es in das Frontpanel ein, anstatt eine eigene Cluster-Eingabe zu erstellen.

Wenn Sie Lastimpedanzen grafisch darstellen, können Sie eine Impedanz als komplexe Zahl der Form $r + jx$ darstellen.

Um den Verlust von Einzelheiten im Smith-Diagramm zu vermeiden, können Sie die Daten mit „Normalize Smith Plot.vi“ normalisieren. Sie können die mit „Normalize Smith Plot.vi“ normalisierten Daten direkt an „Smith Plot.vi“ übergeben. Die Daten des Smith-Diagramms werden normalerweise in Bezug auf die charakteristische Impedanz (Z_0) des Systems normalisiert.

Bildfunktionen-VIs

Verwenden Sie die Bildfunktionen-VIs in der Palette **Funktionen» Grafiken & Sound» Bildfunktionen**, um Formen zu zeichnen und in ein Bildanzeigeelement Text einzugeben. Sie können Punkte, Linien, Formen und Pixmaps zeichnen. Bei Pixmaps umgewandelter Daten handelt es sich um 2D-Arrays mit Farben, wobei jeder Wert einer Farbe entspricht.

Die erste Zeile der Palette **Bildfunktionen** enthält VIs, mit denen Sie Punkte und Linien zeichnen können. Ein Punkt besteht aus einem Cluster aus 16-Bit-Ganzzahlen mit Vorzeichen, welche die x - und y -Koordinaten eines Pixels darstellen. Beim Zeichnen wird die Position des Grafikstifts für das Bild gespeichert.

Bei den meisten Bildfunktionen-VIs müssen Sie absolute Koordinaten angeben, das heißt, relativ zum Ursprung (0,0). Bei „Draw Line.vi“ und „Move Pen.vi“ können Sie absolute oder relative Koordinaten angeben. Relative Koordinaten sind relativ zur aktuellen Position des Stifts. Mit „Move Pen.vi“ können Sie die Stiftposition, ohne zu zeichnen, ändern.

Lediglich bei „Draw Point.vi“, „Move Pen.vi“, „Draw Line.vi“ und „Draw Multiple Lines.vi“ wird die Stiftposition geändert.

Die zweite Zeile der Palette **Bildfunktionen** enthält VIs, mit denen Sie gefüllte Formen zeichnen können. Die einzelnen VIs zeichnen in einem rechteckigen Bereich eines Bildes eine Form. Sie legen ein Rechteck als Cluster mit vier Werten fest, welche die linken, oberen, rechten und unteren Pixel darstellen. Bei „Draw Arc.vi“ beschreibt das Rechteck die Größe eines Ovals. Zusätzliche Parameter legen den Teil des Ovals fest, der gezeichnet werden soll (den Bogen).

Die dritte Zeile der Palette **Bildfunktionen** enthält VIs, mit denen Sie in einem Bild Text zeichnen können. „Get Text Rect.vi“ zeichnet keinen Text, sondern berechnet die Größe des umgebenden Rechtecks eines Strings.

Die vierte Zeile der Palette **Bildfunktionen** enthält VIs, mit denen Sie in einem Bild Pixmaps zeichnen können. Zusätzlich zu dem 2D-Array mit Daten müssen Sie auch die Farbtabelle für die Zuordnung von Array-Werten und Farben festlegen. Das 2D-Array entspricht einem Gitter aus Pixeln. Diese VIs konvertieren jeden Wert im Daten-Array anhand der Werte als Index in das Farbtabelle-Array in eine Farbe.

Die letzte Zeile der Palette **Bildfunktionen** enthält die Konstante „Empty Picture“, die Sie immer dann verwenden, wenn Sie mit einem leeren Bild beginnen oder ein leeres Bild ändern müssen.

Erstellen und Ändern von Farben mit den Bildfunktionen-VIs

Viele Bildfunktionen-VIs besitzen den Eingang **color**, um die Farbe von Formen und Text zu ändern. Die einfachste Möglichkeit, eine Farbe zu erstellen, besteht darin, eine Farbfeldkonstante zu verwenden, die sich in der Palette **Funktionen»Numerisch»Zusätzliche numerische Konstanten** befindet, und zur Auswahl der Farbe auf die Konstante zu klicken.

Um Farben als Ergebnis von Berechnungen und nicht mit den Farbfeldkonstanten zu erstellen, müssen Sie wissen, wie ein Farbfeld eine Farbe mit einem numerischen Wert festlegt.

Eine 32-Bit-Ganzzahl mit Vorzeichen stellt eine Farbe dar. Die unteren drei Bytes stellen die roten, grünen und blauen Bestandteile der Farbe dar. Erstellen Sie für einen blauen Farbbereich ein Array mit 32-Bit-Ganzzahlen, in dem nur die Werte des untersten Bytes geändert werden, da das unterste Byte die blaue Komponente enthält. Erstellen Sie

für einen grauen Farbbereich ein Array mit 32-Bit-Ganzzahlen, in dem die roten, grünen und blauen Werte jedes Elements gleich sind.

Grafikformate-VIs

Mit den Grafikformate-VIs in der Palette **Funktionen»Grafiken & Sound»Grafikformate** können Sie Daten aus mehreren Standardgrafikdateiformaten lesen und schreiben, wie zum Beispiel Bitmap- (.bmp), Portable Network Graphics (.png)- und Joint Photographic Experts Group (.jpg)-Dateien. Mit Hilfe dieser VIs können Sie die folgenden Aufgaben durchführen:

- Lesen von Daten aus einer Bitmap-Datei zur Anzeige in einem Bildanzeigeelement.
- Lesen von Daten für die Bildbearbeitung.
- Schreiben eines Bildes für die Anzeige in anderen Applikationen.

Bei Bitmap-Daten handelt es sich um 2D-Arrays, in denen der Datentyp jedes Punktes abhängig von der Farbtiefe variiert. Bei einem Schwarz-/Weißbild (1-Bit-Bild) ist jeder Punkt vom Typ Boolesch. Bei 4-Bit- und 8-Bit-Bildern stellt jeder Punkt einen Index in einer Farbtabelle dar. Bei 24-Bit-Bildern mit Echtfarben bildet jeder Punkt eine Mischung aus roten, grünen und blauen Werten (RGB).

Die VIs, die Grafikdateien lesen und schreiben, arbeiten mit Daten in einem einfachen, unstrukturierten Format, das eher der Art entspricht, wie Grafikdateien auf einem Datenträger geschrieben werden, als den in einem eindimensionalen Array gespeicherten Daten. Bei diesen Grafikdateien handelt es sich um Pixmaps, deren Konzept dem von Bitmaps ähnelt. Sie können diese umgewandelten Daten mit „Draw Flattened Pixmap.vi“, das sich in der Palette **Funktionen»Grafiken & Sound»Bildfunktionen** befindet, direkt schreiben. Diese Palette enthält VIs zum Zeichnen von Pixmaps, bei denen es sich um 2D-Arrays aus 1-Bit- (schwarz und weiß), 4-Bit-, 8-Bit- und 24-Bit-Daten handelt.

Um die Daten als 2D-Array zu bearbeiten, können Sie diese mit „Unflatten Pixmap.vi“ und „Flatten Pixmap.vi“, die sich in der Palette **Funktionen»Grafiken & Sound»Grafikformate** befinden, in das entsprechende Format konvertieren.

Sound-VIs

Mit den Sound-VIs, die sich in der Palette **Funktionen»Grafiken & Sound»Sound** befinden, können Sie Klangdateien und Sound-Funktionen in VIs integrieren. Mit Hilfe dieser VIs können Sie die folgenden Aufgaben durchführen:

- Erstellen von VIs, die Klangdateien wiedergeben, wie beispielsweise eine aufgezeichnete Warnung, wenn Anwender bestimmte Aktionen ausführen.
- Erstellen eines VIs, das eine Klangdatei wiedergibt, wenn das VI die Ausführung beginnt oder beendet, beziehungsweise wenn im VI eine bestimmte Stelle erreicht wird.
- Konfigurieren eines Sound-Eingabegerätes, um Klangdaten zu erfassen. Mit den Sound-Eingang-VIs können Sie Klangdaten erfassen. Sie können weiterhin beliebige Klangdaten lesen, die durch das Gerät laufen.
- Konfigurieren eines Sound-Ausgabegerätes, das Klangdaten von anderen Sound-VIs akzeptiert. Sie können die Lautstärke des Sounds durch das Gerät steuern, den Sound wiedergeben oder anhalten und den Sound vom System löschen.

Datei-I/O

Datei-I/O-Operationen übergeben Daten an und von Dateien. Mit den Datei-I/O-VIs und -Funktionen, die sich in der Palette **Funktionen» Datei-I/O** befinden, können Sie alle Aspekte der Datei-I/O durchführen, wie beispielsweise:

- Öffnen und Schließen von Datendateien.
- Lesen von Daten aus und Schreiben von Daten in Dateien.
- Lesen aus und Schreiben in eine Datei im Tabellenkalkulationsformat.
- Verschieben und Umbenennen von Dateien und Verzeichnissen.
- Ändern von Dateieigenschaften.
- Erstellen, Ändern und Lesen einer Konfigurationsdatei.

Verwenden der High-Level-VIs, um häufige I/O-Operationen durchzuführen. Verwenden der Low-Level-VIs und -Funktionen, um jede Datei-I/O-Operation einzeln zu steuern.

Weitere Informationen ...

Weitere Informationen zum Durchführen von Datei-I/O-Operationen finden Sie in der *LabVIEW-Hilfe*.

Grundlagen der Datei-I/O

Eine normale Datei-I/O-Operation umfasst den folgenden Prozess.

1. Erstellen oder Öffnen einer Datei. Angeben, wo sich eine vorhandene Datei befindet, oder wo eine neue Datei erstellt werden soll, indem Sie einen Pfad angeben oder in einem Dialogfeld von LabVIEW den Speicherort der Datei auswählen. Nach dem Öffnen der Datei stellt eine Refnum die Datei dar. Weitere Informationen zu Refnums finden Sie in dem Abschnitt *Referenzen auf Objekte oder Applikationen* des Kapitels 4, *Erstellen des Frontpanels*.
2. Lesen aus oder Schreiben in die Datei.
3. Schließen der Datei.

Die meisten Datei-I/O-VIs und -Funktionen führen in einer Datei-I/O-Operation nur einen Schritt aus. Einige High-Level-Datei-I/O-VIs für häufige Datei-I/O-Operationen führen jedoch alle drei Schritte durch. Obwohl diese VIs nicht so effizient sind wie die Low-Level-Funktionen, sind sie eventuell einfacher zu verwenden.

Auswählen eines Datei-I/O-Formats

Die verwendeten Datei-I/O-VIs hängen von dem Format der Dateien ab. Sie können drei Formate verwenden, um Daten aus Dateien zu lesen beziehungsweise Daten in Dateien zu schreiben: Text, Binär und Datenprotokoll. Das verwendete Format hängt sowohl von den erfassten oder erstellten Daten ab als auch von den Applikationen, die auf die Daten zugreifen.

Anhand der folgenden grundlegenden Richtlinien können Sie das zu verwendende Format ermitteln:

- Wenn Sie die Daten anderen Applikationen zur Verfügung stellen möchten, wie zum Beispiel Microsoft Excel, verwenden Sie Textdateien, da diese am häufigsten verwendet werden und am einfachsten portiert werden können.
- Wenn Sie Schreib- oder Lesezugriffe durchführen müssen oder Geschwindigkeit und Festplattenspeicherplatz von Bedeutung sind, verwenden Sie Binärdateien, da sie hinsichtlich Festplattenspeicherplatz und Geschwindigkeit effizienter sind als Textdateien.
- Wenn Sie in LabVIEW komplexe Datensätze oder unterschiedliche Datentypen bearbeiten müssen, verwenden Sie Datenprotokolldateien, da sie sich optimal zum Speichern von Daten eignen, sofern der Zugriff auf die Daten lediglich von LabVIEW aus erfolgen soll und komplexe Datenstrukturen gespeichert werden müssen.

Einsatz von Textdateien

Verwenden Sie Dateien im Textformat, wenn die Daten anderen Anwendern oder Applikationen zur Verfügung gestellt werden sollen, wenn Festplattenspeicherplatz und Datei-I/O-Geschwindigkeit ohne Bedeutung sind, wenn kein wahlfreier Schreib- oder Lesezugriff erfolgen muss, und wenn die numerische Genauigkeit unwichtig ist.

Textdateien bilden das einfachste einsetzbare Format für die gemeinsame Nutzung. Nahezu jeder Computer kann Textdateien lesen und schreiben. Eine Vielzahl von textbasierten Programmen kann textbasierte Dateien

lesen. Die meisten Applikationen zur Gerätesteuerung verwenden Text-Strings.

Sie sollten Daten, auf die Sie von einer anderen Applikation, beispielsweise von einer Textverarbeitungs- oder Tabellenkalkulationsapplikation, aus zugreifen möchten, in Textdateien speichern. Sie können Daten mit den String-Funktionen, die sich in der Palette **Funktionen»String** befinden, im Textformat speichern und alle Daten in Text-Strings konvertieren. Textdateien können Informationen unterschiedlicher Datentypen enthalten.

Textdateien belegen normalerweise mehr Speicherplatz als Binär- oder Datenprotokolldateien wenn die Daten ursprünglich nicht in Textform vorliegen, wie zum Beispiel Graphen- oder Diagrammdaten, da die ASCII-Darstellung von Daten normalerweise umfangreicher ist als die eigentlichen Daten. Sie können beispielsweise die Zahl `-123,4567` als Fließkommazahl einfacher Genauigkeit in 4 Bytes speichern. Jedoch belegt die entsprechende ASCII-Darstellung 9 Bytes (ein Byte für jedes Zeichen).

Darüber hinaus ist der wahlfreie Zugriff auf numerische Daten in Textdateien schwierig. Obwohl jedes Zeichen in einer Zeichenfolge genau 1 Byte belegt, ist der Speicherplatz zum Ausdruck einer Zahl als Text normalerweise variabel. Um die neunte Zahl in einer Textdatei zu finden, muss LabVIEW zuerst die vorhergehenden acht Zahlen lesen und konvertieren.

Beim Speichern von numerischen Daten in Textdateien kann sich die Genauigkeit verringern. Computer speichern numerische Daten als Binärdaten, und numerische Daten werden in einer Textdatei normalerweise in dezimaler Schreibweise geschrieben. Beim Lesen der Daten aus der Textdatei kann die Genauigkeit verloren gehen. Ein Verlust der Genauigkeit tritt bei Binärdateien nicht auf.

Beispiele für die Verwendung von Datei-I/O bei Textdateien finden Sie in `examples\file\smpfile.llb` und `examples\file\sprdsht.llb`.

Einsatz von Binärdateien

Eine Binärdatei verwendet eine feste Anzahl Speicherbytes auf dem Datenträger. Wenn beispielsweise eine Zahl von 0 bis 4 Milliarden im Binärformat gespeichert wird (zum Beispiel `1, 1000` oder `1000000`), belegt jede Zahl 4 Bytes.

Verwenden Sie Binärdateien, um numerische Daten zu speichern und um entweder auf bestimmte Zahlen in einer Datei oder wahlfrei auf die Zahlen in einer Datei zuzugreifen. Binärdateien sind im Gegensatz zu Textdateien,

die von Menschen gelesen werden können, nur von Rechnern lesbar. Binärdateien bieten das kompakteste und schnellste Format zum Speichern von Daten. Sie können in Binärdateien mehrere Datentypen verwenden. Dies ist jedoch unüblich.

Binärdateien sind effizienter, da sie weniger Festplattenspeicherplatz belegen und Daten beim Speichern und Abrufen nicht in eine oder aus einer Textdarstellung konvertiert werden müssen. Eine Binärdatei kann in 1 Byte Festplattenspeicherplatz 256 Werte darstellen. Binärdateien enthalten oftmals ein Byte-für-Byte Abbild der Daten im Speicher, außer bei erweiterten und komplexen Zahlen. Enthält die Datei ein Byte-für-Byte Abbild der Daten im Speicher, erfolgt das Lesen der Daten schneller, da keine Konvertierung notwendig ist.



Hinweis Text und Binärdateien werden auch Byte-Stream-Dateien bezeichnet, das heißt, Daten werden als Sequenz von Zeichen oder Bytes gespeichert.

Beispiele zum Lesen und Schreiben eines Arrays mit Fließkommawerten doppelter Genauigkeit aus einer beziehungsweise in eine Binärdatei finden Sie in den VIs „Read Binary File.vi“ und „Write Binary File.vi“ in `examples\file\smp1file.llb`.

Einsatz von Datenprotokolldateien

Verwenden Sie Datenprotokolldateien, um nur in LabVIEW auf Daten zuzugreifen und zu bearbeiten sowie komplexe Datenstrukturen schnell und einfach zu speichern.

Datenprotokolldateien können nur von LabVIEW erstellt und gelesen werden. Im Gegensatz zu Binärdateien, die beim Speichern ähnliche Datentypen erfordern, können Sie mit Datenprotokolldateien in einem Datensatz einer Datei mehrere verschiedene Datentypen speichern. In einer Datenprotokolldatei können Sie beispielsweise Strings, Zahlen und Cluster speichern.

Eine Datenprotokolldatei speichert Daten als Folge von identisch aufgebauten Datensätzen, ähnlich wie bei einer Tabellenkalkulation, bei der jede Zeile einen Datensatz darstellt. Jedem Datensatz in einer Datenprotokolldatei müssen dieselben Datentypen zugeordnet sein. Die Komponenten eines Datenprotokolldatensatzes können jedoch beliebige Datentypen aufweisen, die beim Erstellen der Datei festgelegt werden.

Sie können beispielsweise ein Datenprotokoll erstellen, dessen Datentyp ein Cluster aus einem String und einer Zahl ist. Jeder Datensatz des Datenprotokolls bildet dann einen Cluster aus einem String und einer Zahl. Der erste Datensatz könnte zum Beispiel („abc“, 1) lauten und der zweite Datensatz („xyz“, 7).

Die Verwendung von Datenprotokolldateien erfordert wenig Bearbeitung, sodass Schreib- und Lesevorgänge wesentlich schneller erfolgen. Außerdem wird der Abruf von Daten vereinfacht, da die ursprünglichen Datenblöcke als Datensatz zurückgelesen werden können, ohne alle Datensätze vorher in der Datei lesen zu müssen. Der wahlfreie Zugriff erfolgt bei Datenprotokolldateien schnell und einfach, da für den Zugriff lediglich die Datensatznummer benötigt wird. LabVIEW weist die Datensatznummer beim Erstellen der Datenprotokolldatei fortlaufend den einzelnen Datensätzen zu.

Sie können vom Frontpanel und vom Blockdiagramm aus auf Datenprotokolldateien zugreifen. Weitere Informationen über den Zugriff auf Datenprotokolldateien vom Frontpanel aus finden Sie in dem Abschnitt [Protokollieren von Frontpanel-Daten](#) dieses Kapitels.

LabVIEW schreibt immer dann einen Datensatz in die Datenprotokolldatei, wenn das entsprechende VI ausgeführt wird. Sie können einen Datensatz nicht überschreiben, nachdem er von LabVIEW in die Datenprotokolldatei geschrieben wurde. Beim Lesen einer Datenprotokolldatei können Sie einen Datensatz oder mehrere Datensätze gleichzeitig lesen.

Beispiele zum Lesen und Schreiben von Datenprotokolldateien finden Sie in `examples\file\data\log.llb`.

Verwenden von High-Level-Datei-I/O-VIs

Mit Hilfe der High-Level-Datei-I/O-VIs, die sich in der obersten Reihe der Palette **Funktionen»Datei-I/O** befinden, können Sie häufige I/O-Operationen durchführen, wie zum Beispiel Lesen und Schreiben der folgenden Datentypen:

- Zeichnen in oder aus Textdateien
- Zeilen aus Textdateien
- 1D- oder 2D-Arrays aus Zahlen einfacher Genauigkeit in oder aus Textdateien einer Tabellenkalkulation
- 1D- oder 2D-Arrays aus Zahlen einfacher Genauigkeit oder 16-Bit-Ganzzahlen mit Vorzeichen in oder aus Binärdateien

Sie können beim Schreiben und Lesen von Dateien Zeit und Programmieraufwand sparen, wenn Sie die High-Level-VIs verwenden. Die High-Level-VIs führen zusätzlich zum Öffnen und Schließen der Datei Lese- und Schreiboperationen durch. Die High-Level-VIs erfordern die Eingabe des Dateipfads. Wenn Sie keinen Pfadnamen angeben, wird ein Dialogfenster angezeigt, in dem Sie eine Datei für den Schreib- oder Lesevorgang angeben können. Bei einem Fehler zeigen die High-Level-VIs ein Dialogfeld an, das den Fehler beschreibt. Sie können die Ausführung beenden oder fortsetzen.

Abbildung 13-1 zeigt die Verwendung des High-Level-VIs In Spreadsheet-Datei schreiben (Tabelle) [Write To Spreadsheet File.vi], um Zahlen an eine Microsoft Excel-Tabellenkalkulationsdatei zu senden. Bei Ausführung dieses VIs werden Sie von LabVIEW aufgefordert, die Daten in eine vorhandene Datei zu schreiben oder eine neue Datei zu erstellen.

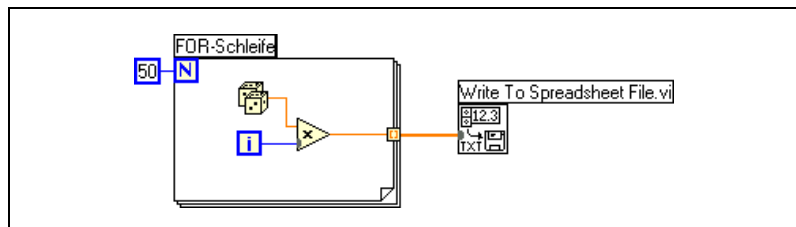


Abbildung 13-1. Verwenden eines High-Level-VIs zum Schreiben in eine Tabelle

Mit den Binärdatei-VIs, die sich in der Palette **Funktionen»Datei-I/O»Binärdatei-VIs** befinden, können Sie Dateien im Binärformat schreiben und lesen. Bei den Daten kann es sich um Ganzzahlen oder Fließkommazahlen einfacher Genauigkeit handeln.

Verwenden von Low-Level- und fortgeschrittenen Datei-I/O-VIs und -Funktionen

Mit den Low-Level-Datei-I/O-VIs und -Funktionen, die sich in der mittleren Zeile der Palette **Funktionen»Datei-I/O** befinden, sowie den fortgeschrittenen Datei-I/O-Funktionen, die sich in der Palette **Funktionen»Datei-I/O»Fortgeschrittene Dateifunktionen** befinden, können Sie jede Datei-I/O-Operation einzeln steuern.

Mit den Low-Level-Hauptfunktionen können Sie eine Datei erstellen oder öffnen, Daten in die Datei schreiben beziehungsweise Daten aus der Datei

lesen sowie die Datei schließen. Mit den anderen Low-Level-Funktionen können Sie die folgenden Aufgaben durchführen:

- Erstellen von Dateien.
- Verschieben, Kopieren oder Löschen von Dateien.
- Auflisten von Verzeichnisinhalten.
- Ändern von Dateieigenschaften.
- Bearbeiten von Pfaden.



Bei einem Pfad, links gezeigt, handelt es sich um einen LabVIEW-Datentyp, der den Speicherort einer Datei auf dem Datenträger bezeichnet. Der Pfad beschreibt den Datenträger, der die Datei enthält, die Verzeichnisse zwischen der obersten Ebene des Dateisystems und der Datei sowie den Namen der Datei. Sie geben einen Pfad mit dem Dateipfad-Bedienelement ein oder zeigen einen Pfad mit dem Dateipfad-Anzeigeelement an, wobei die Syntax der jeweiligen Plattform verwendet werden muss. Weitere Informationen zu Pfadbedien- und -anzeigeelementen finden Sie in dem Abschnitt *Dateipfad-Bedien- und Anzeigeelemente* des Kapitels 4, *Erstellen des Frontpanels*.

Abbildung 13-2 zeigt die Verwendung der Low-Level-VIs und -Funktionen, um Zahlen an eine Datei im Microsoft Excel-Tabellenkalkulationsformat zu senden. Wenn Sie dieses VI ausführen, öffnet das VI Öffnen/Erstellen/Ersetzen einer Datei [Open/Create/Replace.vi] die Datei `numbers.xls`. Die Funktion „Datei schreiben“ schreibt den Zahlen-String in die Datei. Die Funktion „Datei schließen“ schließt die Datei. Wird die Datei nicht geschlossen, bleibt die Datei im Speicher und ist für andere Applikationen oder Benutzer nicht zugänglich.

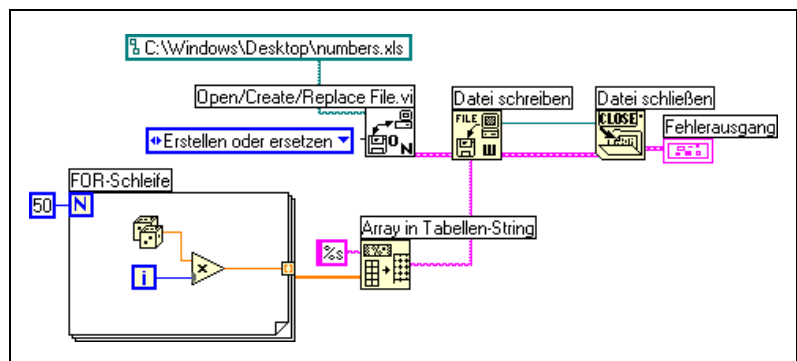


Abbildung 13-2. Verwenden eines Low-Level-VIs zum Schreiben in eine Tabelle

Vergleichen Sie das VI in Abbildung 13-2 mit dem VI in Abbildung 13-1, das die gleiche Aufgabe durchführt. In Abbildung 13-2 müssen Sie mit der Funktion „Array in Tabellen-String“ das Array aus Zahlen als String formatieren. Das VI In Spreadsheet-Datei schreiben (Tabelle) [Write To Spreadsheet File.vi] in Abbildung 13-1 öffnet die Datei, konvertiert das Array aus Zahlen in einen String und schließt die Datei.

Ein Beispiel zum Verwenden von Low-Level-Datei-I/O-VIs und -Funktionen finden Sie in „Write Datalog File Example.vi“ in der Datei `examples\file\datalog.llb`.

Datenträger-Streaming

Sie können die Low-Level-Datei-I/O-VIs und -Funktionen auch für Datenträger-Streaming verwenden, das Speicherressourcen spart. Bei Datenträger-Streaming handelt es sich um ein Verfahren, bei dem Dateien geöffnet bleiben, wenn mehrere Schreiboperationen (beispielsweise in einer Schleife) durchgeführt werden. Obwohl die High-Level-Schreiboperationen einfach eingesetzt werden können, wird die Datei bei jeder Ausführung geöffnet und geschlossen. Sie können VIs effizienter gestalten, wenn Sie ein häufiges Öffnen und Schließen der Datei vermeiden.

Datenträger-Streaming verringert die Häufigkeit, mit der die Funktion mit dem Betriebssystem zusammenarbeitet, um die Datei zu öffnen und zu schließen. Fügen Sie für eine typische Datenträger-Streaming-Operation das VI „Open/Create/Replace File.vi“ vor der Schleife und die Funktion „Datei schließen“ hinter der Schleife ein. In der Schleife kann die Datei ohne den Overhead für das Öffnen und Schließen der Datei kontinuierlich beschrieben werden.

Datenträger-Streaming eignet sich ideal bei langwierigen Datenerfassungsoperationen, bei denen die Geschwindigkeit von Bedeutung ist. Die Daten können bei laufender Datenerfassung kontinuierlich in eine Datei geschrieben werden. Vermeiden Sie die Ausführung anderer VIs und Funktionen, wie zum Beispiel Analyse-VIs und -Funktionen, bis die Erfassung abgeschlossen ist, um optimale Ergebnisse zu erhalten.

Erstellen von Text- und Spreadsheet-Dateien

Um Daten in eine Text- oder Spreadsheet-Datei zu schreiben, müssen die Daten in Strings konvertiert werden. Um die Daten in eine Spreadsheet-Datei zu schreiben, müssen Sie den String als Tabellen-String formatieren. Dabei handelt es sich um einen String, der Trennzeichen enthält (beispielsweise Tabulatorzeichen). Weitere Informationen zum Formatieren von Strings finden Sie in dem Abschnitt *Formatieren von Strings* des Kapitels 9, *Gruppieren von Daten mit Hilfe von Strings, Arrays und Clustern*.

Beim Schreiben von Text in Textdateien ist keine Formatierung erforderlich, da die meisten Textverarbeitungsapplikationen, die Text lesen, keinen formatierten Text benötigen. Mit dem VI Zeichen in Datei schreiben [Write Characters To File.vi], das die Datei automatisch öffnet und schließt, können Sie einen Text-String in eine Textdatei schreiben.

Mit dem VI In Spreadsheet-Datei schreiben (Tabelle) [Write To Spreadsheet File.vi] oder der Funktion „Array in Tabellen-String“ können Sie eine Zahlenreihe aus einem Graphen, einem Diagramm oder einer Erfassung in einen Tabellen-String konvertieren. Weitere Informationen über die Verwendung dieser VIs und Funktionen finden Sie in den Abschnitten *Verwenden von High-Level-Datei-I/O-VIs* und *Verwenden von Low-Level- und fortgeschrittenen Datei-I/O-VIs und -Funktionen* dieses Kapitels.

Beim Lesen von Text einer Textverarbeitungsapplikation kann es zu Fehlern kommen, da Textverarbeitungsapplikationen Text mit unterschiedlichen Schriftarten, Farben, Formatierungen und Größen formatieren, die von den Spreadsheet-Datei VIs nicht verarbeitet werden können.

Wenn Sie Zahlen und Text in eine Tabelle oder Textverarbeitungsapplikation schreiben möchten, verwenden Sie die String-Funktionen in der Palette **Funktionen»String** und die Array-Funktionen in der Palette **Funktionen»Array**, um die Daten zu formatieren und die Strings zu kombinieren. Schreiben Sie die Daten anschließend in eine Datei. Weitere Informationen zum Verwenden dieser Funktionen, um Daten zu formatieren und zu kombinieren, finden Sie in Kapitel 9, *Gruppieren von Daten mit Hilfe von Strings, Arrays und Clustern*.

Formatieren und Schreiben von Daten in Dateien

Mit der Funktion „In Datei formatieren“ können Sie String-, Zahlen-, Pfad- und boolesche Daten als Text formatieren und den Text in eine Datei schreiben. In vielen Fällen können Sie diese Funktion verwenden, anstatt den String mit der Funktion „In String formatieren“ zu formatieren und den resultierenden String mit dem VI Zeichen in Datei schreiben [Write Characters To File.vi] oder der Funktion „Datei schreiben“ zu schreiben.

Mit der Funktion „In Datei formatieren“ können Sie die Reihenfolge bestimmen, in der die Daten in der Textdatei erscheinen. Mit dieser Funktion können Sie jedoch keine Daten an eine Datei anhängen oder in einer Datei vorhandene Daten überschreiben. Verwenden Sie für diese Operationen die Funktion „In String formatieren“ zusammen mit der Funktion „Datei schreiben“.

Weitere Informationen zum Formatieren von Strings finden Sie in dem Abschnitt *Formatieren von Strings* des Kapitels 9, *Gruppieren von Daten mit Hilfe von Strings, Arrays und Clustern*.

Einlesen von Daten aus Dateien

Mit der Funktion „Aus Datei einlesen“ können Sie Text in einer Datei nach String-, Zahlen-, Pfad- und booleschen Werten absuchen und den Text anschließend in einen Datentyp konvertieren. In vielen Fällen können Sie diese Funktion verwenden, anstatt Daten mit der Funktion „Datei lesen“ oder mit dem VI Zeichen aus Datei lesen [Read Characters From File.vi] zu lesen und den resultierenden String mit der Funktion „Aus String suchen“ zu durchsuchen.

Mit der Datei „Aus Datei einlesen“ können Sie den gesamten Text in der Datei lesen. Mit dieser Funktion können Sie jedoch nicht die Stelle in der Datei bestimmen, an der das Einlesen beginnt. Verwenden Sie für diese Operation das VI Zeichen aus Datei lesen [Read Characters From File.vi] zusammen mit der Funktion „Aus String suchen“.

Erstellen von Binärdateien

Zum Erstellen einer Binärdatei erfassen Sie eine Zahlenreihe und schreiben sie in eine Datei. Sie brauchen die Daten nur dann zu formatieren, wenn Sie die High-Level-Binärdatei-I/O-VIs in der Palette **Funktionen»Datei-I/O»Binärdatei-VIs** verwenden möchten.

Verwenden Sie die VIs In I16-Datei schreiben [Write To I16.vi] und In SGL-Datei schreiben [Write To SGL.vi], um ein- und zweidimensionale Arrays aus Zahlen in einer Datei zu speichern. Sie müssen die Zahlen zuerst als 16-Bit-Ganzzahlen oder Fließkommazahlen einfacher Genauigkeit formatieren. Mit den VIs Aus I16-Datei lesen [Read I16.vi] und Aus SGL-Datei lesen [Read SGL.vi] können Sie die erstellten Dateien lesen.

Um Zahlen eines anderen Datentyps zu schreiben, wie zum Beispiel Fließkommazahlen doppelter Genauigkeit oder 32-Bit-Ganzzahlen ohne Vorzeichen, verwenden Sie die Low-Level- oder die fortgeschrittenen Dateifunktionen, die sich in der Palette **Funktionen»Datei-I/O»Fortgeschrittene Dateifunktionen** befinden. Verwenden Sie beim Lesen der Datei die Funktion „Datei lesen“ und geben den Datentyp der Zahl an.

Beispiele zum Lesen und Schreiben von Fließkommazahlen aus einer beziehungsweise in eine Binärdatei finden Sie in den VIs „Read Binary File.vi“ und „Write Binary File.vi“ in `examples\file\smplfile.llb`.

Erstellen von Datenprotokolldateien

Sie können Datenprotokolldateien erstellen und lesen, indem Sie die Frontpanel-Datenprotokollierung aktivieren beziehungsweise die Low-Level- oder die fortgeschrittenen Dateifunktionen in der Palette **Funktionen»Datei-I/O»Fortgeschrittene Dateifunktionen** verwenden, um Daten zu erfassen und die Daten in eine Datei zu schreiben. Weitere Informationen zum Erstellen und Zugreifen auf Datenprotokolldateien vom Frontpanel aus finden Sie in dem Abschnitt [Protokollieren von Frontpanel-Daten](#) dieses Kapitels.

Die Daten in einer Datenprotokolldatei müssen nicht formatiert werden. Beim Schreiben oder Lesen von Datenprotokolldateien müssen Sie jedoch den Datentyp angeben. Wenn Sie beispielsweise einen Temperaturmesswert mit Datum und Uhrzeit erfassen, zu der die Temperatur aufgezeichnet wurde, schreiben Sie die Daten in eine Datenprotokolldatei und geben die Daten als Cluster einer Zahl und zwei Strings an. Ein Beispiel für das

Schreiben von Daten in eine Datenprotokolldatei finden Sie in „Simple Temp Datalogger.vi“ in der Datei `examples\file\datalog.11b`.

Wenn Sie eine Datei lesen, die einen Temperaturmesswert mit Datum und Uhrzeit enthält, zu der die Temperatur aufgezeichnet wurde, geben Sie an, dass ein Cluster aus einer Zahl und zwei Strings gelesen werden soll. Ein Beispiel für das Lesen einer Datenprotokolldatei finden Sie in „Simple Temp Datalog Reader.vi“ in der Datei `examples\file\datalog.11b`.

Schreiben von Signalverläufen in Dateien

Mit den VIs Signalverläufe in Datei schreiben [Write Waveforms to File.vi] und Signalverläufe in Spreadsheet-Datei exportieren [Export Waveforms to Spreadsheet File.vi] in der Palette **Funktionen»Signalverlauf»Datei-I/O für Signalverlauf** können Sie Signalverläufe an Dateien senden. Sie können Signalverläufe in Tabellen-, Text- oder Datenprotokolldateien schreiben.

Wenn Sie beabsichtigen, den erstellten Signalverlauf nur in einem VI zu verwenden, speichern Sie den Signalverlauf als Datenprotokolldatei (.log). Weitere Informationen zur Datenprotokollierung finden Sie in dem Abschnitt *Einsatz von Datenprotokolldateien* dieses Kapitels.

Das VI in Abbildung 13-3 erfasst mehrere Signalverläufe, zeigt sie in einem Graphen an und schreibt sie in eine Datei im Microsoft Excel-Tabellenformat.

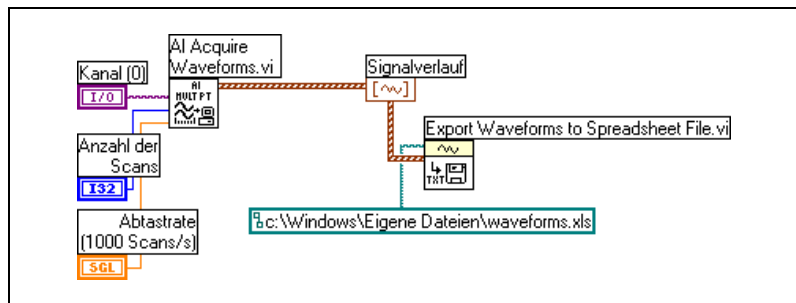


Abbildung 13-3. Schreiben mehrerer Signalverläufe in eine Spreadsheet-Datei

Lesen von Signalverläufen aus Dateien

Mit dem VI Signalverlauf aus Datei lesen [Read Waveform from File.vi] in der Palette **Funktionen»Signalverlauf»Datei-I/O für Signalverlauf** können Sie einen Signalverlauf aus einer Datei lesen. Nach dem Lesen eines einzelnen Signalverlaufs können Sie Komponenten des Signalverlaufdatentyps mit der Funktion „Signalverlauf erstellen“, die sich in der Palette **Funktionen»Signalverlauf** befindet, hinzufügen oder bearbeiten, oder Signalverlaufkomponenten mit der Funktion „Signalverlaufsattribut lesen“ in der Palette **Funktionen»Signalverlauf** extrahieren.

Das VI in Abbildung 13-4 liest einen Signalverlauf aus einer Datei, bearbeitet die **dt**-Komponente des Signalverlaufs und stellt den bearbeiteten Signalverlauf in einem Graphen dar.

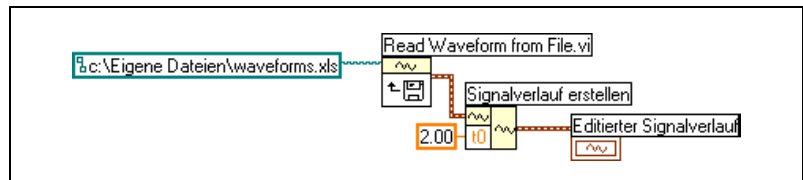


Abbildung 13-4. Lesen eines Signalverlaufs aus einer Datei

Das VI „Signalverlauf aus Datei lesen“ [Read Waveform from File.vi] liest ebenfalls mehrere Signalverläufe aus einer Datei. Das VI gibt ein Array aus Signalverlaufdatentypen zurück, die in einem Graphen mit mehreren Kurven angezeigt werden können. Wenn Sie auf einen einzelnen Signalverlauf einer Datei zugreifen möchten, müssen Sie das Array aus Signalverlaufdatentypen indizieren (siehe Abbildung 13-5). Weitere Informationen zum Indizieren von Arrays finden Sie in dem Abschnitt *Arrays* des Kapitels 9, *Gruppieren von Daten mit Hilfe von Strings, Arrays und Clustern*. Das VI greift auf eine Datei zu, die mehrere Signalverläufe umfasst. Die Funktion „Index-Array“ listet den ersten und den dritten Signalverlauf in der Datei und stellt sie in zwei verschiedenen Kurvengraphen dar. Weitere Informationen zu Graphen finden Sie in dem Kapitel 8, *Schleifen und Case-Strukturen*.

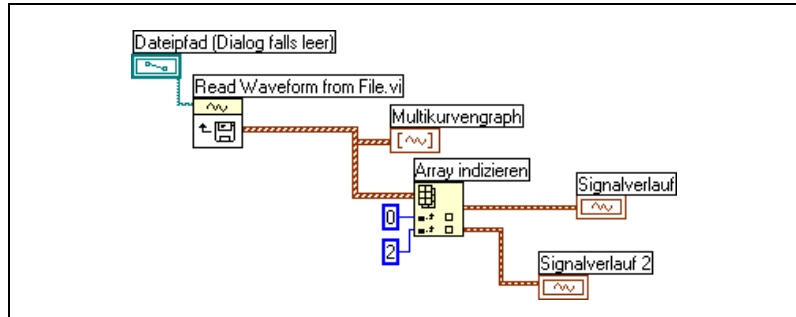


Abbildung 13-5. Lesen mehrerer Signalverläufe aus einer Datei

Durchfluss-Parameter

Viele Datei-I/O-VIs und -Funktionen enthalten Durchfluss-Parameter, normalerweise eine Refnum oder einen Pfad, die denselben Wert zurückgeben wie der entsprechende Eingangsparameter. Sie können diese Parameter einsetzen, um die Ausführungsreihenfolge der Funktionen zu steuern. Durch Verbinden des Durchfluss-Ausgangs des ersten Knotens, der ausgeführt werden soll, mit dem entsprechenden Eingang des nächsten Knotens, der ausgeführt werden soll, schaffen Sie eine künstliche Datenabhängigkeit. Ohne diese Durchfluss-Parameter müssten Sie Sequenzstrukturen verwenden, um sicherzustellen, dass Datei-I/O-Operationen in der gewünschten Reihenfolge stattfinden. Weitere Informationen zur künstlichen Datenabhängigkeit finden Sie in dem Abschnitt *Datenabhängigkeit und künstliche Datenabhängigkeit* des Kapitels 5, *Erstellen des Blockdiagramms*.

Erstellen von Konfigurationsdateien

Mit den Konfigurationsdatei-VIs in der Palette **Funktionen»Datei-I/O» Konfigurationsdatei-VIs** können Sie Standard-Windows-Dateien für Konfigurationseinstellungen lesen und erstellen (.ini-Dateien) und plattformspezifische Daten wie Pfade in einem plattformunabhängigen Format schreiben, damit Sie die von den jeweiligen VIs erstellten Dateien auf mehreren Plattformen verwenden können.

Beispiele zum Verwenden der Konfigurationsdatei-VIs finden Sie in `examples\file\config.llb`.



Hinweis Die Standarderweiterung für Windows-Konfigurationseinstellungsdateien lautet `.ini`, die Konfigurationsdatei-VIs arbeiten jedoch mit Dateien beliebiger Erweiterung zusammen, sofern der Inhalt im richtigen Format vorliegt. Weitere Informationen zum Konfigurieren des Inhalts finden Sie in dem Abschnitt *Windows-Dateiformat für Konfigurationseinstellungen* dieses Kapitels.

Verwenden von Konfigurationseinstellungsdateien

Bei einer Standard-Windows-Datei mit Konfigurationseinstellungen handelt es sich um ein spezielles Format zum Speichern von Daten in einer Textdatei. Sie können auf einfache Weise programmatisch auf die Daten in der `.ini`-Datei zugreifen, da sie einem speziellen Format folgt.

Betrachten Sie beispielsweise eine Datei mit Konfigurationseinstellungen mit dem folgenden Inhalt:

```
[Data]
Wert=7.2
```

Mit den Konfigurationsdatei-VIs können Sie diese Daten, wie in Abbildung 13-6 gezeigt, lesen. Dieses VI verwendet das VI Schlüssel lesen [Read Key.vi], um den **Schlüssel** mit dem Namen `Wert` aus dem **Abschnitt** mit dem Namen `Data` zu lesen. Dieses VI funktioniert unabhängig von Änderungen der Datei, sofern die Datei das Windows-Dateiformat für Konfigurationseinstellungen verwendet.

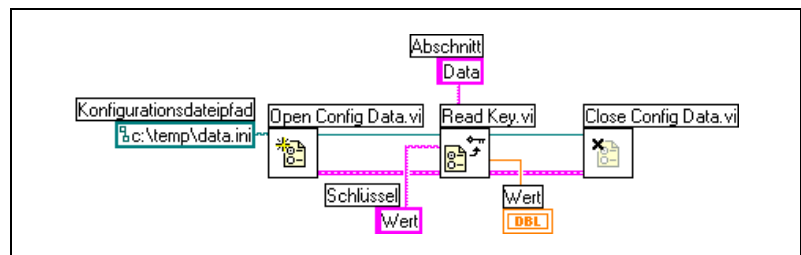


Abbildung 13-6. Lesen von Daten aus einer `.ini`-Datei

Windows-Dateiformat für Konfigurationseinstellungen

Bei Windows-Konfigurationseinstellungsdateien handelt es sich um Textdateien, die in benannte Abschnitte unterteilt sind. Abschnittsnamen stehen in eckigen Klammern. Alle Abschnittsnamen in einer Datei müssen eindeutig sein. Die Abschnitte enthalten Schlüssel/Wert-Paare, die durch ein Gleichheitszeichen (=) getrennt sind. Innerhalb eines Abschnitts müssen alle Schlüsselnamen eindeutig sein. Der Schlüsselname stellt einen

Konfigurationsparameter dar und der Wertname die Einstellung des jeweiligen Parameters. Das folgende Beispiel zeigt den Aufbau der Datei:

```
[Abschnitt 1]
Schlüssel1=Wert
Schlüssel2=Wert

[Abschnitt 2]
Schlüssel1=Wert
Schlüssel2=Wert
```

Verwenden Sie bei Konfigurationsdatei-VIs für den Wertbestandteil des Parameters **Schlüssel** die folgenden Datentypen:

- String
- Pfad
- Boolesch
- 64-Bit-Fließkommazahl mit doppelter Genauigkeit
- 32-Bit-Ganzzahl mit Vorzeichen
- 32-Bit-Ganzzahl ohne Vorzeichen

Die Konfigurationsdatei-VIs können unformatierte oder konvertierte String-Daten lesen und schreiben. Die VIs lesen unformatierte Daten Byte-für-Byte, ohne die Daten nach ASCII zu konvertieren. Bei konvertierten Strings speichert LabVIEW alle Textzeichen, die nicht angezeigt werden können, in der Konfigurationseinstellungsdatei als entsprechende Hexadezimalcodes, wie zum Beispiel `\0D` für einen Wagenrücklauf. Zusätzlich speichert LabVIEW einen umgekehrten Schrägstrich in der Konfigurationseinstellungsdatei doppelt, wie zum Beispiel `\\` für `\`. Setzen Sie den Eingangsparameter **Standardwert** der Konfigurationsdatei-VIs für unformatierte Daten auf TRUE und für konvertierte Daten auf FALSE.

Wenn VIs in eine Konfigurationsdatei schreiben, werden Strings oder Pfaddaten, die ein Leerzeichen enthalten, in Anführungszeichen eingeschlossen. Enthält ein String Anführungszeichen, speichert LabVIEW sie als `\`. Wenn Sie Konfigurationsdateien mit einem Texteditor lesen oder schreiben, stellen Sie eventuell fest, dass LabVIEW Anführungszeichen durch `\` ersetzt.

LabVIEW speichert Pfaddaten in `.ini`-Dateien in einem plattformunabhängigen Format, dem UNIX-Standardformat für Pfade. Die VIs interpretieren den in einer Konfigurationseinstellungsdatei gespeicherten, absoluten Pfad `/c/temp/data.dat` wie folgt:

- **(Windows)** `c:\temp\data.dat`
- **(Macintosh)** `c:temp:data.dat`
- **(UNIX)** `/c/temp/data.dat`

Die VIs interpretieren den relativen Pfad `temp/data.dat` wie folgt:

- **(Windows)** `temp\data.dat`
- **(Macintosh)** `:temp:data.dat`
- **(UNIX)** `temp/data.dat`

Protokollieren von Frontpanel-Daten

Mit der Frontpanel-Datenprotokollierung können Sie Daten für die Verwendung in anderen VIs und in Berichten aufzeichnen. Sie können beispielsweise Daten aus einem Graphen protokollieren und die betreffenden Daten in einem anderen Graphen eines anderen VIs verwenden.

Bei jeder Ausführung eines VIs speichert die Frontpanel-Datenprotokollierung die Frontpanel-Daten in einer getrennten Datenprotokolldatei, die ein Textformat mit Trennzeichen verwendet. Sie können die Daten folgenderweise abrufen:

- Verwenden desselben VIs, aus dem die Daten aufgezeichnet wurden, um die Daten interaktiv abzurufen.
- Verwenden des VIs als Sub-VI, um die Daten programmatisch abzurufen.
- Verwenden der Datei-I/O-VIs und -Funktionen, um die Daten abzurufen.

Jedes VI unterhält eine Protokolldateibindung, welche den Speicherort der Datenprotokolldatei aufzeichnet, in der LabVIEW die protokollierten Frontpanel-Daten verwaltet. Die Protokolldateibindung ist die Zuordnung zwischen einem VI und der Datenprotokolldatei, in der die VI-Daten protokolliert werden.

Eine Datenprotokolldatei enthält Datensätze, die eine Zeitmarkierung und alle Daten enthält, wenn ein VI ausgeführt wurde. Beim Zugriff auf eine Datenprotokolldatei wählen Sie den gewünschten Datensatz aus, indem das

VI im Abrufmodus ausgeführt wird und die Frontpanel-Bedienelemente zum Anzeigen der Daten verwendet werden. Wenn Sie das VI im Abrufmodus ausführen, wird oben im Frontpanel ein numerisches Bedienelement angezeigt, damit Sie durch die Datensätze navigieren können. In Abbildung 13-7 finden Sie ein Beispiel für dieses numerische Bedienelement.

Automatische und interaktive Frontpanel-Datenprotokollierung

Wählen Sie **Ausführen»Protokoll nach Beendigung**, um die automatische Protokollierung zu aktivieren. Bei der ersten Protokollierung von Frontpanel-Daten für ein VI fordert LabVIEW Sie auf, einen Namen für die Datenprotokolldatei anzugeben. LabVIEW protokolliert die Daten bei jeder Ausführung des VIs und hängt bei jeder weiteren Ausführung des VIs einen Datensatz an die Datenprotokolldatei an. Sie können einen Datensatz nicht überschreiben, nachdem er von LabVIEW in die Datenprotokolldatei geschrieben wurde.

Um die Daten interaktiv zu protokollieren, wählen Sie **Ausführen»Datenprotokollierung»Protokoll**. LabVIEW hängt die Daten unmittelbar an die Datenprotokolldatei an. Protokollieren Sie die Daten interaktiv, damit Sie auswählen können, wann die Daten protokolliert werden. Bei automatischer Protokollierung der Daten werden die Daten bei jeder Ausführung des VIs protokolliert.



Hinweis Ein Kurvendiagramm protokolliert bei der Frontpanel-Datenprotokollierung zu einem Zeitpunkt nur einen Datenpunkt. Wenn Sie ein Array mit dem Diagrammanzeigeelement verbinden, enthält das Datenprotokoll eine Untermenge des vom Diagramm angezeigten Arrays.

Interaktive Anzeige der protokollierten Frontpanel-Daten

Nach dem Protokollieren von Daten können Sie diese interaktiv anzeigen, indem Sie **Ausführen»Datenprotokollierung»Protokolldaten lesen** wählen. Die Datenabrufsymbolleiste wird angezeigt (siehe Abbildung 13-7).

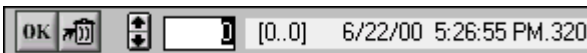


Abbildung 13-7. Datenabrufsymbolleiste

Die hervorgehobene Zahl kennzeichnet den angezeigten Datensatz. Die Zahlen in eckigen Klammern zeigen den Datensatzbereich an, der für das aktuelle VI protokolliert wurde. Bei jeder Ausführung des VIs

wird ein Datensatz protokolliert. Datum und Uhrzeit zeigen an, wann der ausgewählte Datensatz protokolliert wurde. Sie können den nächsten oder vorherigen Datensatz anzeigen, indem Sie auf die entsprechenden Pfeile klicken. Sie können auch die Nach-Unten- und Nach-Oben-Pfeiltasten auf der Tastatur verwenden.

Zusätzlich zu der Datenabrufsymbolleiste ändert sich das Erscheinungsbild des Frontpanels abhängig von dem in der Symbolleiste ausgewählten Datensatz. Wenn Sie beispielsweise auf den Pfeil für den nächsten Datensatz klicken und zu einem anderen Datensatz wechseln, zeigen die Bedien- und Anzeigeelemente die Daten für den betreffenden Datensatz zum Zeitpunkt der Protokollierung der Daten an. Klicken Sie auf die Schaltfläche **OK**, um den Abrufmodus zu verlassen und zu dem VI zurückzukehren, dessen Datenprotokolldatei angezeigt wurde.

Löschen eines Datensatzes

Im Abrufmodus können Sie bestimmte Datensätze löschen. Sie markieren einen einzelnen Datensatz im Abrufmodus zum Löschen, indem Sie den betreffenden Datensatz anzeigen und auf die Schaltfläche **Datensatz löschen** klicken. Wenn Sie erneut auf die Schaltfläche **Datensatz löschen** klicken, wird die Löschkennzeichnung des Datensatzes wieder entfernt.

Wählen Sie im Abrufmodus **Ausführen»Datenprotokollierung»Daten löschen**, um alle zum Löschen markierten Daten zu löschen.

Wenn Sie nicht **Ausführen»Datenprotokollierung»Daten löschen** wählen, bevor Sie auf die Schaltfläche **OK** klicken, werden Sie von LabVIEW gefragt, ob die markierten Datensätze gelöscht werden sollen.

Löschen der Protokolldateibindung

Mit der Protokolldateibindung ordnen Sie einem VI die Datenprotokolldatei zu, die beim Protokollieren oder Abrufen von Frontpanel-Daten verwendet werden soll. Sie können einem VI zwei oder mehr Datenprotokolldateien zuordnen. Dies ist eventuell beim Testen und Vergleichen der VI-Daten hilfreich. Sie können beispielsweise die Daten, die bei der ersten Ausführung des VIs protokolliert wurden, mit den Daten vergleichen, die bei der zweiten Ausführung des VIs protokolliert wurden. Um einem VI mehr als eine Datenprotokolldatei zuzuordnen, müssen Sie die

Protokolldateibindung löschen, indem Sie **Ausführen»Datenprotokollierung»Lösch Protokolldateibindung** wählen. LabVIEW fordert Sie auf, eine Datenprotokolldatei auszuwählen, wenn das VI das nächste Mal ausgeführt wird, egal ob die automatische Protokollierung aktiviert ist oder die Daten interaktiv protokolliert werden sollen.

Ändern der Protokolldateibindung

Sie ändern die Protokolldateibindung, um Frontpanel-Daten zu protokollieren oder Frontpanel-Daten aus einer anderen Protokolldatei abzurufen, indem Sie **Ausführen»Datenprotokollierung»Protokolldateibindung ändern** auswählen. LabVIEW fordert Sie auf, eine andere Protokolldatei auszuwählen oder eine neue zu erstellen. Sie können die Protokolldateibindung auch ändern, wenn Sie andere Daten in ein VI einlesen oder Daten aus dem VI an ein anderes Datenprotokoll anhängen möchten.

Programmatischer Abruf von Frontpanel-Daten

Sie können protokollierte Daten auch mit Hilfe eines Sub-VIs beziehungsweise den Datei-I/O-VIs und -Funktionen abrufen.

Abrufen von Frontpanel-Daten mit Hilfe eines Sub-VIs

Wenn Sie mit der rechten Maustaste auf ein Sub-VI klicken und aus dem Kontextmenü **Datenbankzugriff aktivieren** wählen, wird um das Sub-VI ein gelbes Kästchen angezeigt (siehe Abbildung 13-8).

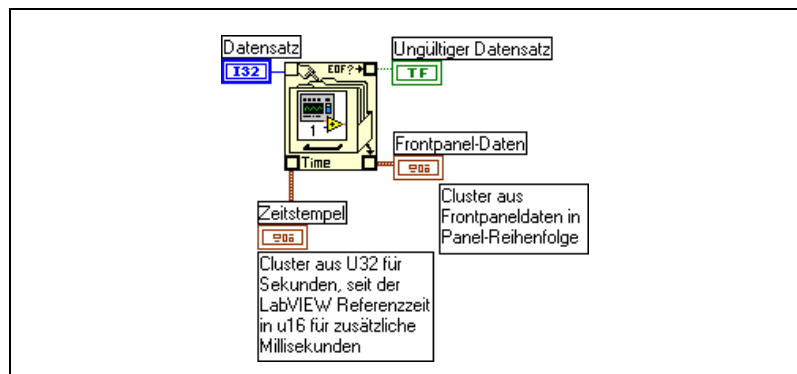


Abbildung 13-8. Abrufen protokollierter Daten

Das gelbe Kästchen, das wie ein Aktenschrank aussieht, enthält Anschlüsse für den Zugriff auf die Daten in der Datenprotokolldatei. Wenn Sie den

Datenbankzugriff für das Sub-VI aktivieren, fungieren die Ein- und Ausgänge des Sub-VIs als Ausgänge, welche die protokollierten Daten zurückgeben. **Datensatz #** zeigt den abzurufenden Datensatz an, **ungültiger Datensatz #** zeigt das Vorhandensein der Datensatznummer an, **Zeitstempel** ist die Uhrzeit, zu welcher der Datensatz erstellt wurde, und **Frontpaneldaten** ist ein Cluster der Frontpanel-Objekte. Sie können auf die Daten eines Frontpanel-Objekts zugreifen, indem Sie den Cluster **Frontpaneldaten** mit der Funktion „Aufschlüsseln“ verbinden.

Sie können außerdem Werte für bestimmte Ein- und Ausgänge abrufen, indem Sie den entsprechenden Anschluss des Sub-VIs anschließen (siehe Abbildung 13-9).

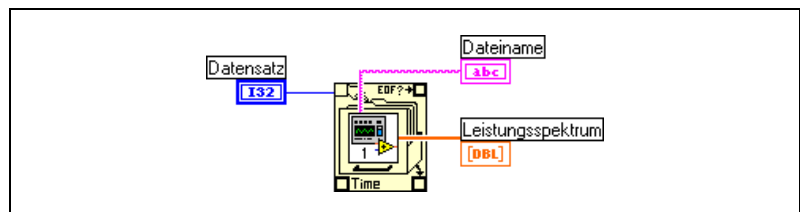


Abbildung 13-9. Abrufen protokollierter Daten über Sub-VI-Anschlüsse

Wenn Sie das VI ausführen, wird das Sub-VI nicht ausgeführt. Statt dessen werden die protokollierten Daten von dem entsprechenden Frontpanel als Cluster an das VI-Frontpanel zurückgegeben.

Festlegen von Datensätzen

Das Sub-VI besitzt n protokollierte Datensätze, und Sie können eine beliebige Zahl von $-n$ bis $n - 1$ mit dem Anschluss **Datensatz #** des Sub-VIs verbinden. Sie können auf Datensätze unter Verwendung von nichtnegativen Datensatzzahlen relativ zu dem ersten protokollierten Datensatz zugreifen. 0 stellt den ersten Datensatz dar, 1 den zweiten Datensatz und so weiter bis $n - 1$, wodurch der letzte Datensatz dargestellt wird.

Sie können auf Datensätze unter Verwendung von negativen Datensatzzahlen relativ zu dem letzten protokollierten Datensatz zugreifen. -1 stellt den letzten Datensatz dar, -2 den vorletzten Datensatz und so weiter bis $-n$, wodurch der erste Datensatz dargestellt wird. Wenn Sie eine Zahl außerhalb des Bereichs $-n$ bis $n - 1$ mit dem Anschluss **Datensatz #** verbinden, ist der Ausgang **ungültiger Datensatz #** TRUE, und das Sub-VI ruft keine Daten ab.

Abrufen von Frontpanel-Daten mit Datei-I/O-Funktionen

Sie können im Frontpanel protokollierte Daten auch mit den Datei-I/O-VIs und -Funktionen, wie zum Beispiel der Funktion „Datei lesen“, abrufen. Der Datentyp jedes Datensatzes in der Frontpanel-Datenprotokolldatei erstellt zwei Cluster. Ein Cluster enthält eine Zeitmarkierung, und der andere Cluster enthält die Frontpanel-Daten. Der Cluster mit der Zeitmarkierung besteht aus einer 32-Bit-Ganzzahl ohne Vorzeichen, welche die Sekunden darstellt, und einer 16-Bit-Ganzzahl ohne Vorzeichen, welche die Millisekunden darstellt, die seit der LabVIEW-Referenzzeit (1. Januar 1904, 00:00 Uhr) verstrichen sind.

Sie können mit denselben Datei-I/O-Funktionen auf Datensätze der Frontpanel-Datenprotokolldateien zugreifen, die für den Zugriff auf programmatisch erstellte Datenprotokolldateien verwendet werden. Geben Sie den **Typ des Datenprotokolldatensatzes** am Typeingang der Funktion „Datei öffnen“ ein (siehe Abbildung 13-10).

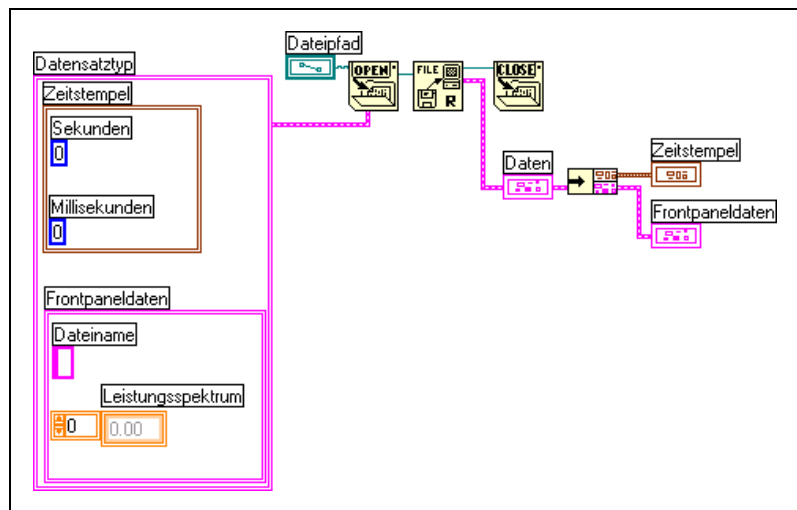


Abbildung 13-10. Abrufen protokollierter Daten mit der Funktion „Datei öffnen“

Dokumentieren und Drucken von VIs

Mit Hilfe von LabVIEW können Sie VIs dokumentieren und drucken.

Der Abschnitt *Dokumentieren von VIs* dieses Kapitels beschreibt das Aufzeichnen von Informationen über das Blockdiagramm und/oder das Frontpanel in einem beliebigen Entwicklungsstadium in einer gedruckten Dokumentation über VIs.

Der Abschnitt *Drucken von VIs* dieses Kapitels beschreibt die Optionen zum Drucken von VIs. Einige Optionen eignen sich besser für das Drucken von Informationen über VIs, und andere für die Berichterstellung der von den VIs erzeugten Daten und Ergebnisse. Mehrere Faktoren beeinflussen die verwendete Methode, wie beispielsweise, ob der Druck manuell oder programmatisch erfolgen soll, wie viele Optionen für das Berichtsformat benötigt werden, ob die Funktionalität in den erstellten unabhängigen Applikationen benötigt wird, und auf welchen Plattformen die VIs ausgeführt werden.

Der Abschnitt *Drucken von Berichten* dieses Kapitels beschreibt, wie in LabVIEW Berichte gedruckt werden.

Weitere Informationen ...

Weitere Informationen zum Dokumentieren und Drucken von VIs finden Sie in der *LabVIEW-Hilfe*.

Dokumentieren von VIs

Mit LabVIEW können Sie die Entwicklung verfolgen, ein fertiggestelltes VI dokumentieren und Anweisungen für Anwender von VIs erstellen. Sie können die Dokumentation in LabVIEW erstellen, drucken und als HTML- oder RTF-Dateien speichern.



Hinweis In erstellten Applikationen können Sie die VI-Dokumentation nicht in jedem Format drucken.

Um die gedruckte Dokumentation besonders effektiv zu verwenden, erstellen Sie VI- und Objektbeschreibungen und richten die VI-Revisions-Historie ein.

Erstellen von VI- und Objektbeschreibungen

Erstellen Sie Beschreibungen für VIs und den dazugehörigen Objekten, wie zum Beispiel Bedienelemente und Anzeigeelemente, um den Zweck des VIs oder Objekts zu beschreiben, und Benutzern Anweisungen zur Verwendung des VIs oder Objekts zur Verfügung zu stellen. Sie können die Beschreibungen in LabVIEW anzeigen oder drucken.

Sie erstellen, bearbeiten und zeigen VI-Beschreibungen an, indem Sie **Datei»VI-Einstellungen** und anschließend aus dem Pulldown-Menü **Kategorie** die Option **Dokumentation** auswählen. Sie erstellen, bearbeiten und zeigen Objekt- und Sub-VI-Beschreibungen an, indem Sie mit der rechten Maustaste auf das Objekt klicken und aus dem Kontextmenü **Beschreibung und Tipp** auswählen. Bei Hinweisstreifen handelt es sich um kurze Beschreibungen, die angezeigt werden, wenn der Cursor über ein Objekt bewegt wird. Wenn Sie im Dialogfeld **Beschreibung und Tipp** keinen Tipp eingeben, wird kein Hinweisstreifen angezeigt. Die VI- oder Objektbeschreibung wird ebenfalls im Fenster **Kontext-Hilfe** angezeigt, wenn Sie den Cursor über das VI-Symbol beziehungsweise das Objekt bewegen. Wählen Sie **Hilfe»Zeige Kontext-Hilfe**, um das Fenster **Kontext-Hilfe** anzuzeigen.

Einrichten der VI-Revisions-Historie

In jedem VI können Sie mit dem Fenster **Historie** die Entwicklungsgeschichte samt Versionsnummer des VIs anzeigen. Zeichnen Sie die an den VIs durchgeführten Änderungen im Fenster **Historie** auf, wenn Sie Änderungen vornehmen, und verfolgen Sie sie. Wählen Sie **Werkzeuge»VI-Revisions-Historie**, um das Fenster **Historie** anzuzeigen. Sie können die Revisions-Historie auch ausdrucken. Weitere Informationen

zum Drucken der Revisions-Historie finden Sie in Abschnitt [Drucken der Dokumentation](#) dieses Kapitels.

Versionsnummern

Die Versionsnummer bildet eine einfache Möglichkeit, Änderungen eines VIs zu verfolgen. Die Versionsnummer beginnt bei Null und wird bei jedem Speichern des VIs erhöht. Die aktuelle auf dem Datenträger gespeicherte Versionsnummer wird im Fenster **Historie** angezeigt. Um die aktuelle Versionsnummer in der Titelleiste des VIs anzuzeigen, wählen Sie **Werkzeuge»Optionen**, aus dem Pulldown-Menü **Revisions-Historie**, und aktivieren das Kontrollkästchen **Versionsnummer in der Titelleiste anzeigen**.

Die in LabVIEW in der Titelleiste des Fensters **Historie** und der Titelleiste des VIs angezeigte Nummer ist die nächste Versionsnummer, das heißt die aktuelle Versionsnummer plus eins. Wenn Sie einen Kommentar zu der Historie hinzufügen, enthält die Kopfzeile des Kommentars die nächste Versionsnummer. Die Versionsnummer wird nicht erhöht, wenn Sie ein VI speichern und lediglich das Fenster **Historie** geändert wurde.

Die Versionsnummern sind von den Kommentaren im Fenster **Historie** unabhängig. Lücken in den Versionsnummern zwischen Kommentaren zeigen an, dass das VI ohne Kommentar gespeichert wurde.

Da es sich bei der Historie genau genommen um ein Entwicklungswerkzeug handelt, entfernt LabVIEW die Historie automatisch, wenn das Blockdiagramm eines VIs gelöscht wird. Weitere Informationen zum Löschen des Blockdiagramms finden Sie in Abschnitt [Bereitstellen von VIs](#) des Kapitels 7, [Erstellen von VIs und Sub-VIs](#). Das Fenster **Historie** steht in der Laufzeitversion eines VIs nicht zur Verfügung. Auf der Seite **Allgemein** des Dialogfelds **VI-Einstellungen** wird die Versionsnummer auch für VIs ohne Blockdiagramm angezeigt. Klicken Sie im Fenster **Historie** auf die Schaltfläche **Zurücksetzen**, um die Revisions-Historie zu löschen und die Versionsnummer auf Null zurückzusetzen.

Drucken der Dokumentation

Wählen Sie **Datei»Drucken**, um die VI-Dokumentation zu drucken oder in einer HTML- oder RTF-Datei zu speichern. Sie können ein integriertes Format verwenden oder für die Dokumentation ein benutzerspezifisches Format erstellen. Die erstellte Dokumentation kann die folgenden Objekte umfassen:

- Anschlussfeld und Symbol
- Frontpanel und Blockdiagramm
- Bedienelemente, Anzeigeelemente und Datentypen
- VI-Hierarchie
- Liste der Sub-VIs
- Revisions-Historie

Speichern in HTML- oder RTF-Dateien

Sie können die VI-Dokumentation in HTML- oder RTF-Dateien speichern. Sie können HTML- und RTF-Dateien in die meisten Textverarbeitungsanwendungen importieren und HTML- und RTF-Dateien verwenden, um Hilfedateien zu erstellen. Verwenden Sie HTML für Dokumentation die im Web veröffentlicht werden soll. Weitere Informationen zum Verwenden von HTML- und RTF-Dateien zum Erstellen von Hilfedateien finden Sie in Abschnitt *Erstellen eigener Hilfedateien* dieses Kapitels. Weitere Informationen zum programmatischen Erstellen von HTML- und RTF-Dateien finden Sie in Kapitel 16, *Programmatische Steuerung von VIs*.

Wenn eine Dokumentation in einer RTF-Datei gespeichert wird, legen Sie fest, ob eine Datei als Online-Hilfedatei oder zur Textverarbeitung erstellt werden soll. Im Hilfedateiformat speichert LabVIEW Grafiken in externen Bitmap-Dateien. Im Textverarbeitungsdateiformat bettet LabVIEW Grafiken in das Dokument ein. Bei HTML-Dateien speichert LabVIEW alle Grafiken extern im JPEG- oder PNG-Format.

Auswählen des Grafikformats für HTML-Dateien

Wenn Sie die Dokumentation in einer HTML-Datei speichern, können Sie das Format der Grafikdateien und die Farbtiefe auswählen.

Das JPEG-Format komprimiert Grafiken. Dabei können jedoch einige Grafikdetails verloren gehen. Dieses Format eignet sich optimal für Fotos. Bei Strichzeichnungen, Frontpanels und Blockdiagrammen kann die JPEG-Komprimierung zu verschwommenen Bildern und ungleichmäßigen Farben führen. JPEG-Grafiken sind immer 24-Bit-Grafiken. Wenn Sie eine

geringere Farbtiefe auswählen, zum Beispiel schwarzweiß, werden Grafiken mit der erforderlichen Tiefe gespeichert, das Ergebnis ist aber trotzdem eine 24-Bit-Grafik.

Das PNG-Format komprimiert Grafiken ebenfalls, wenn auch nicht immer so gut wie beim JPEG-Format. Die PNG-Komprimierung führt jedoch nicht zu einem Detailverlust. Außerdem werden 1-Bit-, 4-Bit-, 8-Bit- und 24-Bit-Grafiken unterstützt. Bei einer geringeren Bit-Tiefe lässt sich die resultierende Grafik sehr viel besser komprimieren als das JPEG-Format. Das PNG-Format ersetzt das GIF-Format (Graphics Interchange Format). Obwohl das PNG-Format eine Reihe von Vorteilen im Vergleich zu JPEG und GIF bietet, wird es jedoch von den meisten Webbrowsern nicht optimal unterstützt.

Um GIF-Grafiken zu verwenden, konvertieren Sie die JPEG- oder PNG-Grafiken mit einem Konvertierprogramm für Grafikformate. Wenn Sie GIF-Grafiken erstellen, sollten Sie PNG-Grafiken nutzen, da sie verlustlose Reproduktionen der Originalgrafiken sind. Konvertieren Sie anschließend die PNG-Grafiken in GIF-Grafiken. Ändern Sie die HTML-Datei, in der die VI-Dokumentation gespeichert wurde, so, dass sie auf die GIF-Grafiken mit der Erweiterung `.gif` verweisen. Aus lizenzrechtlichen Gründen speichert LabVIEW Grafiken nicht als GIF-Dateien; dies kann jedoch in Zukunft der Fall sein.

Namenskonventionen für Grafikdateien

Wenn Sie HTML-Dateien oder RTF-Dateien mit externen Grafiken speichern, speichert LabVIEW die Anschlüsse der Bedien- und Anzeigeelemente mit konsistenten Namen in den Grafikdateien. Verfügt ein VI über mehrere Anschlüsse desselben Typs, erstellt LabVIEW eine Grafikdatei, die jeden Anschluss enthält. Wenn ein VI beispielsweise über drei vorzeichenbehaftete 32-Bit-Ganzzahleneingänge verfügt, erstellt LabVIEW eine einzelne Datei mit dem Namen `ci32.jpg`.

Erstellen eigener Hilfedateien

Sie können mit den von LabVIEW erstellten HTML- oder RTF-Dateien Hilfedateien erstellen. Sie können HTML-Dateien für die Verwendung im Web oder zum Erstellen von HTML-basierten Hilfedateien erzeugen. Sie können Verknüpfungen von LabVIEW zu HTML-basierten Hilfedateien erstellen.

Windows-Hilfdateien

Windows-Hilfdateien basieren auf RTF-Dateien. Sie können in diesen Dateien Themen erstellen, um Verbindungen zu VIs zu erstellen, und Sie können Verknüpfungen von LabVIEW zu einer Windows-Hilfdatei erstellen.

Mit einem Hilfecompiler können Sie aus den RTF-Dateien Hilfdateien erstellen. Wenn Sie Hilfdateien für mehrere Plattformen erstellen möchten, müssen Sie den Hilfecompiler für die Plattform verwenden, auf der die Datei verwendet werden soll.

Drucken von VIs

Sie können in LabVIEW die folgenden Hauptmethoden verwenden, um VIs zu drucken:

- Wählen Sie **Datei»Fenster drucken**, um den Inhalt des aktiven Fensters zu drucken.
- Wählen Sie **Datei»Drucken**, um umfassendere Informationen über ein VI zu drucken, wie beispielsweise Informationen über das Frontpanel, das Blockdiagramm, Sub-VIs, Bedienelemente, VI-Historie und so weiter. Weitere Informationen zum Verwenden dieser Methoden zum Drucken von VIs finden Sie in Abschnitt *Drucken der Dokumentation* dieses Kapitels.
- Mit den Berichterzeugungs-VIs können Sie Papier- oder HTML-Berichte der vom VI erzeugten Informationen drucken.
- Mit dem VI-Server kann jederzeit jedes VI-Fenster oder jede VI-Dokumentation programmatisch gedruckt werden. Weitere Informationen zum Verwenden dieser Methode zum Drucken von VIs finden Sie in Kapitel 16, *Programmatische Steuerung von VIs*.

Drucken des aktiven Fensters

Wählen Sie **Datei»Fenster drucken**, um den Inhalt des aktiven Frontpanel- oder Blockdiagrammfensters mit den wenigsten Eingabeaufforderungen zu drucken. LabVIEW druckt den Arbeitsbereich des aktiven Fensters, der jedoch nicht durch die Größe des aktiven Fensters beschränkt wird. LabVIEW druckt weder Titelleiste, Menüleiste, Symbolleiste noch Bildlaufleisten.

Wählen Sie **Datei»VI-Einstellungen** und dann aus dem Pulldown-Menü **Kategorie** den Eintrag **Druck-Optionen**, um genauer steuern zu können, wie LabVIEW ein VI druckt, wenn Sie **Datei»Fenster drucken** auswählen

oder programmatisch drucken. Weitere Informationen zum programmatischen Drucken finden Sie in Abschnitt *Programmatisches Drucken* dieses Kapitels.

Drucken von Berichten

Mit den Berichterzeugungs-VIs, die sich in der Palette **Funktionen»Berichterzeugung** befinden, können Sie Papier- oder HTML-Berichte der vom VI erzeugten Informationen drucken. Sie erstellen mit „Einfacher Text-Report.vi“ einen einfachen Bericht. Mit den anderen Berichterzeugungs-VIs können Sie komplexere Berichte erstellen.

Mit Hilfe der Berichterzeugungs-VIs können Sie die folgenden Aufgaben durchführen:

- Festlegen von Kopf- und Fußzeilen des Berichts.
- Festlegen von Textschriftart, -größe, -stil und -farbe.
- Festlegen von Rändern und Tabulatoren.
- Bestimmen, welche Informationen in welcher Zeile oder auf welcher Seite eines Berichts angezeigt werden.
- Festlegen der Ausrichtung des Berichts: Hochformat oder Querformat.
- Einschließen von Text aus anderen Dateien in einem Bericht.
- Löschen von Informationen aus einem vorhandenen Bericht.
- Automatisches Drucken eines Papierberichts oder Speichern eines HTML-Berichts.
- Anhängen von Text, Grafiken oder Tabellen an einen Bericht.
- Löschen eines Berichts nach dem Druck.

Programmatisches Drucken

Mit den folgenden Methoden können Sie VIs programmatisch drucken, anstatt interaktiv die Dialogfelder **Drucken** zu verwenden, die bei Auswahl von **Datei»Fenster drucken** und **Datei»Drucken** angezeigt werden:

- Festlegen, dass das Frontpanel eines VIs nach jeder Ausführung automatisch gedruckt wird.
- Erstellen eines Sub-VIs, um das VI zu drucken.
- Mit dem VI-Server kann jederzeit jedes VI-Fenster oder jede VI-Dokumentation programmatisch gedruckt werden. Weitere Informationen zum Verwenden dieser Methode zum Drucken von VIs finden Sie in Kapitel 16, *Programmatische Steuerung von VIs*.

Drucken nach Ausführung

Wählen Sie **Ausführen»Nach Ausführung drucken**, um ein VI nach seiner Ausführung zu drucken. Sie können auch **Datei»VI-Einstellungen** wählen, aus dem Pulldown-Menü **Kategorie** den Eintrag **Druck-Optionen** auswählen und dann das Kontrollkästchen **Panel automatisch drucken, wenn die Ausführung des VIs beendet ist** aktivieren.

Wenn Sie diese Option auswählen, druckt LabVIEW den Inhalt des Frontpanels immer dann, wenn die Ausführung eines VIs beendet wird. Wenn das VI als Sub-VI verwendet wird, druckt LabVIEW, wenn die Ausführung des Sub-VIs beendet wird, bevor es zum aufrufenden VI zurückkehrt.

Verwenden eines Sub-VIs für einen selektiven Ausdruck bei Beendigung

In einigen Fällen soll ein VI eventuell nicht nach jeder Ausführung ausgedruckt werden. Sie möchten den Druck vielleicht so einstellen, dass er nur dann erfolgt, wenn der Benutzer auf eine Schaltfläche klickt oder wenn irgendein Zustand auftritt, wie zum Beispiel ein Testfehler. Eventuell möchten Sie auch mehr Kontrolle über das Format des Ausdrucks oder nur eine Teilmenge der Bedienelemente drucken. In diesen Fällen können Sie ein Sub-VI verwenden, das nach der Ausführung gedruckt wird.

Erstellen Sie ein Sub-VI und formatieren Sie das Frontpanel so, wie es von LabVIEW gedruckt werden soll. Anstatt **Ausführen»Nach Ausführung drucken** auszuwählen, während Sie sich im VI befinden, können Sie den Menüpunkt vom Sub-VI aus auswählen. Wenn Sie drucken möchten, rufen Sie das Sub-VI auf und übergeben die zu druckenden Daten.

Weitere Druckverfahren

Erfüllen die Standarddruckmethoden von LabVIEW nicht Ihre Anforderungen, können Sie die folgenden zusätzlichen Verfahren verwenden:

- Zeilenweises Drucken von Daten. Wenn Sie einen Zeilendrucker besitzen, der an die serielle Schnittstelle angeschlossen ist, können Sie mit den Seriellen Kompatibilitäts-VIs Text an den Drucker senden. Dies erfordert in der Regel einige Kenntnisse über die Befehlssprache des Druckers.
- Exportieren von Daten in andere Applikationen, wie zum Beispiel Microsoft Excel, Speichern der Daten in einer Datei und Drucken aus der anderen Applikation.

- **(Windows und UNIX)** Verwenden Sie „System Exec.vi“.
- **(Macintosh)** Verwenden Sie „AESend Print Document.vi“.
- **(Windows)** Verwenden Sie die ActiveX-Automation, um Daten von anderen Applikationen drucken zu lassen. Weitere Informationen zu ActiveX finden Sie in Kapitel 18, [ActiveX](#).

Anpassen von VIs

Sie können die Funktionsweise von VIs und Sub-VIs entsprechend den Anforderungen der Applikation konfigurieren. Wenn Sie beispielsweise beabsichtigen, ein VI als Sub-VI zu verwenden, das eine Benutzereingabe erfordert, konfigurieren Sie das VI so, dass das dazugehörige Frontpanel bei jedem Aufruf angezeigt wird.

Sie können ein VI auf viele Arten konfigurieren, entweder im VI selbst oder programmatisch unter Verwendung des VI-Servers. Weitere Informationen, wie mit dem VI-Server ein identisches Verhalten von VIs konfiguriert werden kann, finden Sie in Kapitel 16, [Programmatische Steuerung von VIs](#).

Weitere Informationen ...

Weitere Informationen über das Anpassen von VIs finden Sie in der *LabVIEW-Hilfe*.

Konfigurieren von Erscheinungsbild und Verhalten von VIs

Wählen Sie **Datei»VI-Einstellungen**, um Erscheinungsbild und Verhalten eines VIs zu konfigurieren. Das Dialogfeld **VI-Einstellungen** steht nicht zur Verfügung, wenn ein VI ausgeführt wird.

Klicken Sie oben im Dialogfeld auf **Kategorie**, um aus verschiedenen Optionskategorien auszuwählen, zum Beispiel:

- **Allgemein:** Zeigt den aktuellen Pfad an, in dem ein VI gespeichert ist, die Versionsnummer, die Versionshistorie und alle seit dem letzten Speichern des VIs durchgeführten Änderungen. Sie können mit Hilfe dieser Seite auch ein Symbol bearbeiten.
- **Dokumentation:** Auf dieser Seite können Sie eine Beschreibung des VIs und eine Verknüpfung zu einem Thema in einer Hilfedatei erstellen. Weitere Informationen zu den Dokumentationsmöglichkeiten

finden Sie in dem Abschnitt *Dokumentieren von VIs* des Kapitels 14, *Dokumentieren und Drucken von VIs*.

- **Sicherheit:** Auf dieser Seite können Sie ein VI sperren oder mit einem Kennwort schützen.
- **Fenstererscheinungsbild:** Auf dieser Seite können Sie verschiedene Fenstereinstellungen konfigurieren.
- **Fenstergröße:** Auf dieser Seite können Sie die Fenstergröße festlegen.
- **Ausführung:** Auf dieser Seite können Sie die Ausführung eines VIs konfigurieren. Sie können ein VI beispielsweise so konfigurieren, dass es beim Öffnen sofort gestartet wird, oder anhält, wenn es als Sub-VI aufgerufen wird. Sie können das VI auch so konfigurieren, dass es mit unterschiedlichen Prioritäten ausgeführt wird. Wenn ein VI beispielsweise unbedingt ausgeführt werden muss, ohne auf den Abschluss einer anderen Operation zu warten, konfigurieren Sie das VI zur Ausführung mit zeitkritischer (höchster) Priorität. Weitere Informationen zum Erstellen von Multithread-VIs finden Sie in den Applikationsinformationen *Using LabVIEW to Create Multithreaded VIs for Maximum Performance and Reliability*.

Anpassen von Menüs

Sie können für jedes erstellte VI angepasste Menüs erstellen und die VIs so konfigurieren, dass Menüleisten ein- oder ausgeblendet werden. Sie können Menüleisten ein- oder ausblenden, indem Sie **Datei»VI-Einstellungen** wählen, auf **Kategorie** klicken, die Option **Fenstererscheinungsbild** auswählen, auf die Schaltfläche **Anpassen** klicken und das Kontrollkästchen **Menüleiste anzeigen** aktivieren beziehungsweise deaktivieren.

Das Konfigurieren von Menüs umfasst das Erstellen des Menüs und das Bereitstellen des Blockdiagrammcodes, der ausgeführt wird, wenn der Anwender die verschiedenen Menüobjekte auswählt.



Hinweis Benutzerspezifische Menüs werden nur angezeigt, während das VI ausgeführt wird.

Erstellen von Menüs

Sie können benutzerspezifische Menüs erstellen oder die Standardmenüs von LabVIEW statisch ändern, wenn Sie das VI bearbeiten, beziehungsweise programmatisch, wenn das VI ausgeführt wird. Wenn Sie **Bearbeiten»Laufzeitmenü** auswählen und im Dialogfeld **Menü-Editor** ein Menü erstellen, erstellt LabVIEW eine Laufzeit-Menü-Datei (.rtm), damit in einem VI anstelle der Standardmenüleiste eine benutzerspezifische Menüleiste angezeigt wird. Nach dem Erstellen und Speichern der .rtm-Datei müssen Sie denselben relativen Pfad zwischen dem VI und der .rtm-Datei beibehalten.

Im Dialogfeld **Menü-Editor** können Sie eine benutzerspezifische .rtm-Datei einem VI zuordnen. Bei Ausführung des VIs wird das Menü aus der .rtm-Datei geladen. Mit Hilfe der Menüfunktionen, die sich in der Palette **Funktionen»Applikationssteuerung»Menü** befinden, können Sie Menüobjekte programmatisch zur Laufzeit einfügen, löschen und ändern. Sie können alle Benutzerobjekte bearbeiten, jedoch nur Anwendungsobjekte hinzufügen oder entfernen. Weitere Informationen zur programmatischen Bearbeitung von Menüs finden Sie in dem Abschnitt *Verarbeitung der Menüauswahl* dieses Kapitels.

Sie können zu einem benutzerspezifischen Menü Anwendungsobjekte, Benutzerobjekte und Zeilenumbrüche hinzufügen. Bei Anwendungsobjekten handelt es sich um Menüobjekte, die von LabVIEW bereitgestellt werden. Wenn Sie ein Anwendungsobjekt auswählen, definiert LabVIEW das Verhalten des betreffenden Objekts. Benutzerobjekte sind Objekte, die von Ihnen hinzugefügt werden. Das Verhalten von Benutzerobjekten wird durch das Blockdiagramm gesteuert.

Verarbeitung der Menüauswahl

Wenn Sie ein benutzerspezifisches Menü erstellen, weisen Sie jedem Menüobjekt einen eindeutigen Stringbezeichner, einen sogenannten Hinweisstreifen, zu, der Groß-/Kleinschreibung unterscheidet. Wählt der Anwender ein Menüobjekt aus, wird der dazugehörige Hinweisstreifen programmatisch mit der Funktion „Menüauswahl holen“ abgerufen. LabVIEW stellt basierend auf dem Hinweisstreifen-Wert des einzelnen Menüobjekts im Blockdiagramm für jedes Menüobjekt einen Handler zur Verfügung. Der Handler ist eine Kombination aus While-Schleife und Case-Struktur, mit der Sie bestimmen können, welches Menü gegebenenfalls ausgewählt wurde, und den entsprechenden Code ausführen können.

Erstellen Sie nach dem Erstellen eines benutzerspezifischen Menüs im Blockdiagramm eine Case-Struktur, welche die einzelnen Objekte im benutzerspezifischen Menü ausführt oder bearbeitet. Dieser Prozess wird Verarbeitung der Menüauswahl genannt. Mit den Funktionen „Menüauswahl holen“ und „Aktivieren der Menüüberwachung“ können Sie definieren, welche Aktionen durchgeführt werden sollen, wenn ein Anwender ein Menüobjekt auswählt. LabVIEW verarbeitet alle Anwendungsobjekte implizit.

In Abbildung 15-1 liest die Funktion „Menüauswahl holen“ das vom Anwender ausgewählte Menüobjekt und übergibt das Menüobjekt an die Case-Struktur, in der das Menüobjekt ausgeführt wird.

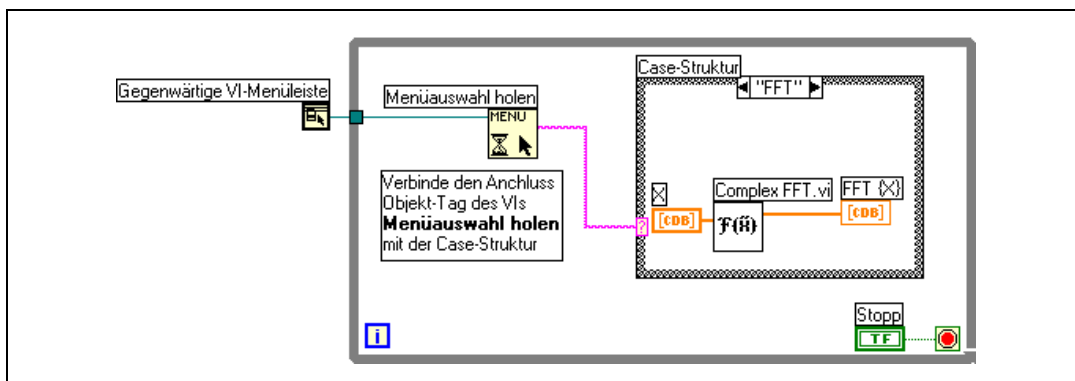


Abbildung 15-1. Blockdiagramm mit Menüverarbeitung

Wenn Sie wissen, dass die Verarbeitung eines bestimmten Menüobjekts lange dauert, verbinden Sie ein boolesches Bedienelement mit dem Eingang **Menü fixieren** der Funktion „Menüauswahl holen“ und setzen das boolesche Bedienelement auf TRUE, um die Menüleiste zu deaktivieren, damit der Benutzer im Menü keine andere Auswahl vornehmen kann, während LabVIEW das Menüobjekt verarbeitet. Verbinden Sie den Wert TRUE mit der Funktion „Aktivieren der Menüüberwachung“, um die Menüleiste zu aktivieren, nachdem das Menüobjekt von LabVIEW verarbeitet wurde.

Programmatische Steuerung von VIs

Sie können über Blockdiagramme, ActiveX-Technologie und das TCP-Protokoll auf den VI-Server zugreifen, um mit VIs und anderen Instanzen von LabVIEW zu kommunizieren, damit Sie VIs und LabVIEW programmatisch steuern können. Sie können VI-Server-Operationen auf einem lokalen Computer oder über ein Netzwerk ausführen.

Weitere Informationen ...

Weitere Informationen zur programmatischen Steuerung von VIs finden Sie in der *LabVIEW-Hilfe*.

Fähigkeiten des VI-Servers

Mit dem VI-Server können Sie die folgenden programmatischen Operationen durchführen:

- Aufrufen eines VIs über das Netzwerk.
- Konfigurieren Sie eine Instanz von LabVIEW als Server, der VIs exportiert, die Sie von anderen Instanzen von LabVIEW im Web aufrufen können. Wenn Sie über eine Applikation zur Datenerfassung verfügen, die Daten von Standorten im Netzwerk erfasst und protokolliert, können Sie diese Daten gelegentlich von Ihrem lokalen Computer aus abrufen. Durch Ändern der LabVIEW-Voreinstellungen können Sie manche VIs im Web zugänglich machen, sodass die Übertragung der neuesten Daten so einfach ist wie ein Sub-VI-Aufruf. Der VI-Server erledigt die Einzelheiten der Vernetzung. Der VI-Server arbeitet außerdem plattformübergreifend, sodass Client und Server auf verschiedenen Plattformen ausgeführt werden können.
- Bearbeiten der Eigenschaften eines VIs und von LabVIEW. Sie können beispielsweise die Position eines VI-Fensters dynamisch ermitteln und einen Bildlauf im Frontpanel durchführen, damit ein Teil davon sichtbar ist. Sie können weiterhin alle Änderungen programmatisch auf der Festplatte speichern.

- Aktualisieren der Eigenschaften mehrerer VIs, anstatt für jedes VI manuell das Dialogfenster **Datei»VI-Einstellungen** zu verwenden.
- Abrufen von Informationen über eine Instanz von LabVIEW, wie zum Beispiel Versionsnummer und Ausgabe. Sie können auch Umgebungsinformationen abrufen, wie zum Beispiel die Plattform, auf der LabVIEW ausgeführt wird.
- Dynamisches Laden von VIs in den Speicher, wenn es von einem anderen VI aufgerufen werden muss, anstatt alle Sub-VIs beim Öffnen eines VIs zu laden.
- Erstellen einer Modularchitektur für die Applikation, um die Funktion der Applikation nach der Auslieferung an Kunden zu erweitern. Angenommen, Sie besitzen mehrere VIs, die Daten filtern und alle dieselben Parameter verwenden. Wenn Sie die Applikation so gestalten, dass diese VIs aus einem Modulverzeichnis dynamisch geladen werden, können Sie die Applikation mit einem Teil dieser VIs ausliefern und Benutzern zusätzliche Filterungsoptionen dadurch zur Verfügung stellen, indem Sie die neuen Filter-VIs in diesem Modulverzeichnis speichern.

Erstellen von VI-Server-Applikationen

Das Programmiermodell für VI-Server-Applikationen basiert auf RefNums. RefNums werden auch bei Datei-I/O, Netzwerkverbindungen und anderen Objekten in LabVIEW verwendet. Weitere Informationen zu RefNums finden Sie in dem Abschnitt [Referenzen auf Objekte oder Applikationen](#) des Kapitels 4, [Erstellen des Frontpanels](#).

Normalerweise öffnen Sie eine RefNum auf eine Instanz von LabVIEW oder einem VI. Die RefNum wird dann als Parameter für andere VIs verwendet. Die VIs lesen oder schreiben Eigenschaften, führen Methoden aus oder laden dynamisch ein referenziertes VI. Schließlich wird die RefNum geschlossen, wodurch die Instanz von LabVIEW oder das VI aus dem Speicher entfernt wird.

Verwenden Sie die folgenden Funktionen und Knoten, die sich in der Funktionspalette unter **Funktionen»Applikationssteuerung** befinden, um VI-Server-Applikationen zu erstellen:

- **Applikationsreferenz öffnen:** Öffnet eine RefNum auf eine lokale oder Netzwerkapplikation, auf die Sie über den Server zugreifen, oder um auf eine Netzwerkinstanz von LabVIEW zuzugreifen. Mit der Funktion „VI-Referenz öffnen“, können Sie auf ein VI auf dem lokalen Computer oder dem Netzwerkrechner zugreifen.

- **Eigenschaftsknoten:** Ruft VI- oder Applikationseigenschaften ab beziehungsweise legt sie fest. Weitere Informationen zu Eigenschaften finden Sie in dem Abschnitt *Eigenschaftsknoten* dieses Kapitels.
- **Methodenknoten:** Ruft Methoden für ein VI oder eine Applikation auf. Weitere Informationen zu Methoden finden Sie in dem Abschnitt *Methodenknoten* dieses Kapitels.
- **Aufruf über Referenz:** Lädt ein VI dynamisch in den Speicher.
- **LV-Objektreferenz schließen:** Schließt die RefNum zum VI oder zur Applikation, auf die über den VI-Server zugegriffen wurde.

Applikations- und VI-Referenzen

Der Zugriff auf die Funktionalität des VI-Servers erfolgt über Referenzen auf zwei Hauptklassen von Objekten: das Applikationsobjekt und das VI-Objekt. Nachdem eine Referenz auf eines dieser Objekte erstellt wurde, können Sie die Referenz an ein VI oder eine Funktion übergeben, das/die eine Operation für das Objekt durchführt.

Eine Applikations-RefNum verweist auf eine lokale oder Netzwerkinstanz von LabVIEW. Mit Hilfe von Applikationseigenschaften und -methoden können Sie die LabVIEW-Voreinstellungen ändern und Systeminformationen zurückgeben. Eine VI-RefNum verweist auf ein VI in der Instanz von LabVIEW.

Mit einer RefNum auf eine Instanz von LabVIEW können Sie Informationen zur LabVIEW-Umgebung abrufen, wie zum Beispiel die Plattform, auf der LabVIEW ausgeführt wird, die Versionsnummer oder eine Liste aller VIs, die sich momentan im Speicher befinden. Sie können auch Informationen, wie zum Beispiel den aktuellen Benutzernamen oder die Liste der in andere Instanzen von LabVIEW exportierten VIs festlegen.

Wenn Sie eine RefNum zu einem VI erhalten, wird das VI in den Speicher geladen. Nach Erhalt der RefNum bleibt das VI solange im Arbeitsspeicher, bis diese RefNum geschlossen wird. Sind gleichzeitig mehrere RefNums zu einem geöffneten VI vorhanden, bleibt das VI im Speicher, bis alle RefNums zu diesem VI geschlossen wurden. Mit einer RefNum zu einem VI können Sie alle Eigenschaften eines VIs, die im Dialogfeld **Datei» VI-Einstellungen** verfügbar sind, sowie dynamische Eigenschaften, wie zum Beispiel die Fensterposition des Frontpanels, aktualisieren. Sie können das VI auch programmatisch drucken, an einem anderen Ort speichern und seine Strings für Übersetzungszwecke in eine andere Sprache exportieren und importieren.

Bearbeiten von Applikations- und VI-Einstellungen

Mit dem VI-Server können Sie Applikations- und VI-Einstellungen mit Hilfe der Eigenschafts- und Methodenknoten abrufen und festlegen. Viele Applikations- und VI-Einstellungen können nur mit Hilfe der Eigenschafts- und Methodenknoten abgerufen und festgelegt werden.

Beispiele zur Verwendung der Applikations- und VI-Klasseneigenschaften und -methoden finden Sie in `examples\viserver`.

Eigenschaftsknoten

Mit dem Eigenschaftsknoten können Sie verschiedene Eigenschaften einer Applikation oder eines VIs abrufen oder festlegen. Sie wählen die Eigenschaften des Knotens mit Hilfe des Bedienwerkzeugs, indem Sie auf den Eigenschaftsanschluss klicken oder mit der rechten Maustaste auf den weißen Bereich des Knotens klicken und aus dem Kontextmenü **Eigenschaften** auswählen.

Sie können mit einem einzelnen Knoten mehrere Eigenschaften lesen oder schreiben. Mit dem Positionierwerkzeug können Sie die Größe des Eigenschaftsknotens ändern, um neue Anschlüsse hinzuzufügen. Ein kleiner Richtungspfeil rechts neben der Eigenschaft zeigt eine gelesene Eigenschaft an. Ein kleiner Richtungspfeil links neben der Eigenschaft zeigt eine geschriebene Eigenschaft an. Klicken Sie mit der rechten Maustaste auf die Eigenschaft und wählen aus dem Kontextmenü **In Lesen ändern** oder **In Schreiben ändern**, um die Funktionsweise der Eigenschaft zu ändern.

Der Knoten wird von oben nach unten ausgeführt. Der Eigenschaftsknoten wird nicht ausgeführt, wenn vor der Ausführung ein Fehler auftritt. Überprüfen Sie daher immer die Möglichkeit von Fehlern. Tritt in einer Eigenschaft ein Fehler auf, ignoriert LabVIEW die restlichen Eigenschaften und gibt einen Fehler zurück. Der Cluster **Fehlerausgang** enthält Informationen darüber, welche Eigenschaft zu dem Fehler geführt hat.

Implizit verknüpfte Eigenschaftsknoten

Wenn Sie einen Eigenschaftsknoten aus einem Frontpanel-Objekt erstellen, indem Sie mit der rechten Maustaste auf das Objekt klicken und aus dem Kontextmenü **Erstelle»Eigenschaftsknoten** auswählen, wird der neue Eigenschaftsknoten implizit mit dem Objekt verknüpft. Da die betreffenden Eigenschaftsknoten implizit mit dem Objekt, aus dem sie

erstellt wurden, verknüpft sind, besitzen Sie keinen Eingang **RefNum** und Sie brauchen den Eigenschaftsknoten nicht mit dem Bedienelement oder der RefNum für dieses Objekt zu verbinden. Weitere Informationen zu Eigenschaftsknoten für Bedienelemente finden Sie in dem Abschnitt *Objekt-RefNums* dieses Kapitels.

Methodenknoten

Mit dem Methodenknoten können Sie Aktionen oder Methoden einer Applikation oder eines VIs ausführen. Im Gegensatz zum Eigenschaftsknoten führt ein einzelner Methodenknoten nur eine einzige Methode einer Applikation oder eines VIs aus. Sie wählen eine Methode mit Hilfe des Bedienwerkzeugs, indem Sie auf den Methodenanschluss klicken oder mit der rechten Maustaste auf den weißen Bereich des Knotens klicken und aus dem Kontextmenü **Methoden** auswählen.

Der Name der Methode ist im Methodenknoten stets der erste Anschluss in der Parameterliste. Wenn die Methode einen Wert zurückgibt, zeigt der Methodenanschluss den Rückgabewert an. Sonst besitzt der Methodenanschluss keinen Wert.

Der Methodenknoten listet die Parameter von oben nach unten auf, wobei der Name der Methode zuerst und die optionalen Parameter am Ende grau angezeigt werden.

Manipulieren von Eigenschaften und Methoden der Applikationsklasse

Sie können Eigenschaften in einer lokalen oder Netzwerkinstanz von LabVIEW abrufen oder festlegen und/oder Methoden in LabVIEW ausführen. Abbildung 16-1 zeigt, wie alle VIs im Speicher eines lokalen Computers in einem String-Array auf dem Frontpanel angezeigt werden können.

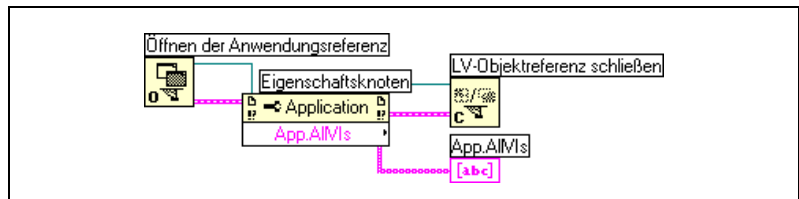


Abbildung 16-1. Anzeigen aller VIs im Speicher eines lokalen Computers

Um die VIs im Speicher eines Netzwerkrechners zu finden, verbinden Sie ein String-Bedienelement mit dem Eingang **Rechnername** der Funktion „Applikationsreferenz öffnen“ (siehe Abbildung 16-2) und geben die IP-Adresse oder den Domännennamen ein. Sie müssen auch die Eigenschaft **Exportierte VIs im Speicher** auswählen, da die Eigenschaft **Alle VIs im Speicher** der Abbildung 16-1 nur für lokale Instanzen von LabVIEW gilt.

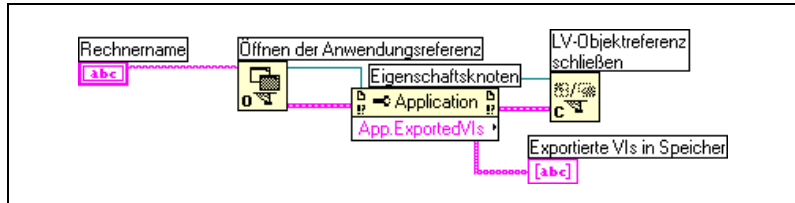


Abbildung 16-2. Anzeigen aller VIs im Speicher eines Netzwerkrechners

Manipulieren von Eigenschaften und Methoden der VI-Klasse

Sie können Eigenschaften eines VIs abrufen oder festlegen und/oder Methoden eines VIs durchführen. In Abbildung 16-3 initialisiert LabVIEW die Frontpanel-Objekte eines VIs mit Hilfe des Methodenknotens erneut mit den Standardwerten. Das Frontpanel wird geöffnet und zeigt die Standardwerte mit dem Eigenschaftsknoten an.

Bevor Sie auf Eigenschaften und Methoden eines VIs zugreifen können, müssen Sie mit der Funktion „VI-Referenz öffnen“ eine RefNum zu dem betreffenden VI erstellen. Um eine Methode eines VIs aufzurufen, verwenden Sie den Methodenknoten.

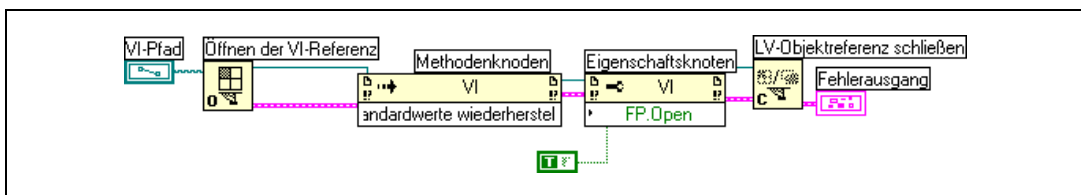


Abbildung 16-3. Verwenden von Eigenschafts- und Methodenknoten für VI-Klassen

Sobald Sie die Funktion „VI-Referenz öffnen“ mit dem Methodenknoten verbunden haben, können Sie auf alle VI-Klassenmethoden zugreifen.

Der Eigenschaftsknoten funktioniert ähnlich wie der Methodenknoten. Nachdem Sie eine VI-RefNum mit dem Eigenschaftsknoten verbunden haben, können Sie auf alle VI-Klasseneigenschaften zugreifen.

Manipulieren von Eigenschaften und Methoden der Applikationsklasse und VI-Klasse

In manchen VIs müssen Sie auf Eigenschaften und Methoden der Applikationsklasse und der VI-Klasse zugreifen. Sie müssen die RefNums zur Applikations- und VI-Klasse getrennt öffnen (siehe Abbildung 16-4).

Abbildung 16-4 zeigt, wie die VIs im Speicher des lokalen Computers ermittelt werden und der Pfad jedes dieser VIs auf dem Frontpanel angezeigt werden kann. Um alle im Speicher befindlichen VIs zu finden, müssen Sie auf eine Applikationsklasseneigenschaft zugreifen. Um die Pfade zu allen betreffenden VIs zu ermitteln, müssen Sie auf eine VI-Klasseneigenschaft zugreifen. Die Anzahl der im Speicher befindlichen VIs bestimmt die Anzahl der Durchläufe der For-Schleife. Fügen Sie die Funktion „VI-Referenz öffnen“ und „LV-Objektreferenz schließen“ in der For-Schleife ein, da für jedes im Arbeitsspeicher befindliche VI eine VI-RefNum benötigt wird. Schließen Sie die Applikations-RefNum erst, nachdem die For-Schleife alle VI-Pfade abgerufen hat.

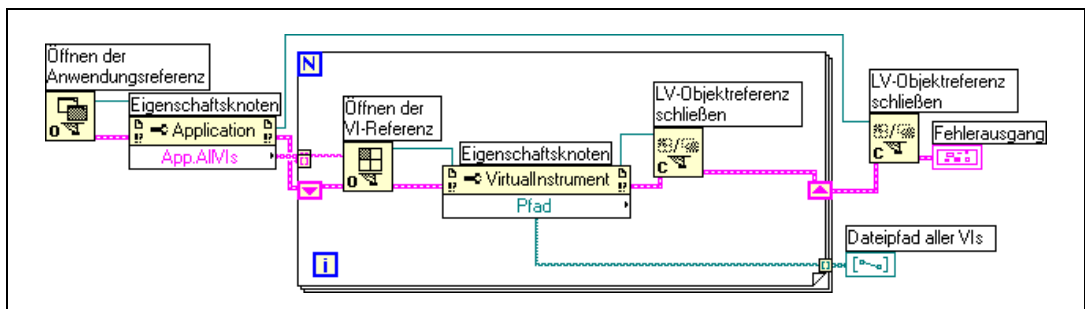


Abbildung 16-4. Verwenden von Eigenschaften und Methoden der Applikationsklasse und VI-Klasse

Dynamisches Laden und Aufrufen von VIs

Sie können VIs dynamisch laden, anstatt statisch verknüpfte Sub-VI-Aufrufe zu verwenden. Bei einem statisch verknüpften Sub-VI handelt es sich um ein VI, das direkt in das Blockdiagramm eines aufrufenden VIs eingefügt wird. Es wird gleichzeitig mit dem aufrufenden VI geladen.

Im Gegensatz zu statisch verknüpften Sub-VIs werden dynamisch geladene Sub-VIs erst dann geladen, wenn das aufrufende VI das Sub-VI aufruft. Bei einem größeren aufrufenden VI können Sie Ladezeit und Speicher sparen,

wenn das Sub-VI dynamisch geladen wird, da das Sub-VI erst dann geladen wird, wenn es vom aufrufenden VI benötigt wird. Nach Abschluss der Operation kann es aus dem Speicher entfernt werden.

Der Knoten „Aufruf über Referenz“ und strikt typisierte VI-RefNums

Mit dem Knoten „Aufruf über Referenz“ können VIs dynamisch aufgerufen werden.

Der Knoten „Aufruf über Referenz“ erfordert eine strikt typisierte VI-RefNum. Die strikt typisierte VI-RefNum bezeichnet das Anschlussfeld des aufgerufenen VIs. Es erstellt keine permanente Zuordnung zum VI und enthält keine anderen VI-Informationen, wie zum Beispiel Name und Speicherort. Sie können die Ein- und Ausgänge des Knotens „Aufruf über Referenz“ wie jedes andere VI verbinden.

Abbildung 16-5 zeigt die Verwendung des Knotens „Aufruf über Referenz“, um „Frequency Response.vi“ dynamisch aufzurufen. Der Knoten „Aufruf über Referenz“ erfordert wie die für Eigenschaftsknoten und Methodenknotten verwendeten Funktionen die Verwendung der Funktionen „Öffnen der VI-Referenz“ und „LV-Objektreferenz schließen“. Der Knoten „Aufruf über Referenz“ erfordert jedoch die Verbindung einer strikt typisierten VI-RefNum mit dem Eingang **Typ-Bezeichner VI-RefNum** der Funktion „Öffnen der VI-Referenz“. Sie schließen eine strikt typisierte VI-RefNum an, indem Sie **Elemente»RefNum»VI-RefNum** auswählen und die RefNum im Frontpanel einfügen. Klicken Sie mit der rechten Maustaste auf die RefNum und wählen aus dem Kontextmenü **VI-Serverklasse auswählen»Durchsuchen**. Das Dialogfeld **Wählen Sie ein VI** fordert Sie zur Auswahl eines VIs auf.

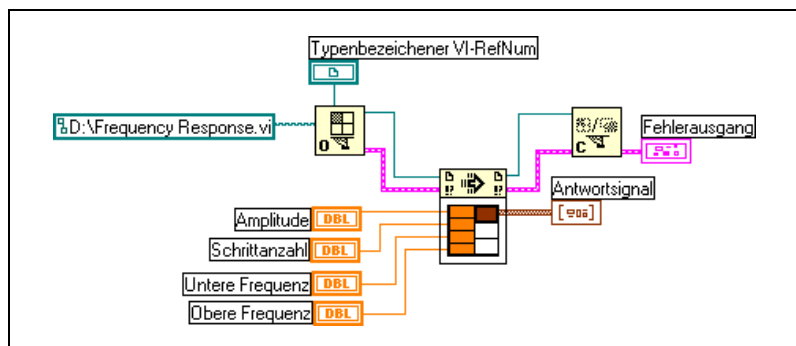


Abbildung 16-5. Verwenden des Knotens „Aufruf über Referenz“

Das für strikt typisierte RefNums angegebene VI stellt lediglich die Anschlussfeldinformationen bereit. Das bedeutet, dass keine permanente Verknüpfung zwischen der RefNum und dem VI hergestellt wird. Vermeiden Sie insbesondere die Verwechslung der Auswahl eines VI-Anschlussfeldes mit dem Erhalt einer RefNum zum ausgewählten VI. Sie geben mit dem Eingang **VI-Pfad** der Funktion „VI-Referenz öffnen“ ein bestimmtes VI an.

Bearbeiten und Ausführen von VIs auf Netzwerkrechnern

Ein wichtiger Aspekt sowohl der Applikations- als auch der VI-RefNums ist ihre Netzwerktransparenz. Dies bedeutet, dass Sie RefNums zu Objekten auf Netzwerkrechnern auf dieselbe Art und Weise öffnen können wie RefNums zu diesen Objekten auf Ihrem Computer.

Nachdem Sie eine RefNum zu einem Objekt im Netzwerk geöffnet haben, können Sie es mit wenigen Einschränkungen in genau derselben Weise behandeln wie ein lokales Objekt. Bei Operationen an einem Netzwerkobjekt sendet der VI-Server die Informationen zur Operation über das Netzwerk und anschließend die Ergebnisse wieder zurück. Die Applikation sieht nahezu identisch aus, ganz gleich ob die Operation über das Netzwerk oder lokal abläuft.

Objekt-RefNums

Mit Objekt-RefNums können Sie Informationen von Frontpanel-Objekten an Sub-VIs übergeben. Nachdem eine Objekt-RefNum an ein Sub-VI übergeben wurde, können Sie mit den Eigenschaftsknoten und Methodenknotten Eigenschaften lesen und konfigurieren und Methoden des referenzierten Frontpanel-Objekts aufrufen.

Strikt typisierte und schwach typisierte Objekt-RefNums

Strikt typisierte Objekt-RefNums akzeptieren nur Objekt-RefNums desselben Datentyps. Wenn der Typ einer strikt typisierten Objekt-RefNum ein Schieberegler mit 32-Bit-Integern ist, können Sie einen Schieberegler mit 32-Bit-Integern, einen Schieberegler mit 8-Bit-Integern oder einen Schieberegler mit Double-Fließkommazahlen an den Anschluss der Objekt-RefNum anschließen, aber keinen Schieberegler mit einem Cluster aus 32-Bit-Integern oder einen Schieberegler mit einem Array aus 32-Bit-Integern.

Objekt-RefNums, die aus einem Bedienelement erstellt werden, sind standardmäßig strikt typisiert. Ein roter Stern in der linken unteren Ecke der Objekt-RefNum auf dem Frontpanel zeigt an, dass die Objekt-RefNum strikt typisiert ist. Im Blockdiagramm wird im Eigenschafts- oder Methodenknoten, der an den Anschluss der Objekt-RefNum angeschlossen ist (`strict`) angezeigt, um darauf hinzuweisen, dass die Objekt-RefNum strikt typisiert ist.



Hinweis Da die mechanischen Aktionen mit Latch mit strikt typisierten Objekt-RefNums nicht kompatibel sind, erzeugen boolesche Bedienelemente mit mechanischen Schaltaktionen schwach typisierte Objekt-RefNums.

Schwach typisierte Objekt-RefNums sind hinsichtlich der akzeptierten Datentypen flexibler. Wenn der Typ einer schwach typisierten Objekt-RefNum ein Schieberegler ist, können Sie einen Schieberegler mit 32-Bit-Ganzzahlen, einen Schieberegler mit einfacher Genauigkeit oder einen Cluster aus Schiebereglern mit 32-Bit-Ganzzahlen mit dem Anschluss der Objekt-RefNum verbinden. Wenn der Typ einer schwach typisierten Objekt-RefNum ein Bedienelement ist, können Sie eine Objekt-RefNum eines beliebigen Bedienelementtyps mit dem Anschluss der Objekt-RefNum verbinden.



Hinweis Wenn Sie einen Eigenschaftsknoten mit dem Anschluss einer schwach typisierten Objekt-RefNum verbinden, erzeugen die Eigenschaften von Werten und Diagrammhistorien Variant-Daten, die eventuell konvertiert werden müssen, bevor sie verwendet werden können. Weitere Informationen zu Variant-Daten finden Sie in dem Abschnitt [Verarbeiten von Variant-Daten](#) des Kapitels 5, [Erstellen des Blockdiagramms](#).

Arbeiten mit LabVIEW im Netzwerk

VIs können mit anderen Prozessen kommunizieren oder vernetzt werden, die beispielsweise in anderen Applikationen oder auf Netzwerkrechnern ausgeführt werden. Mit den Netzwerkfunktionen in LabVIEW können Sie die folgenden Aufgaben durchführen:

- Sie können Direktübertragungsdaten mit Hilfe der National Instruments DataSocket-Technologie gemeinsam mit anderen VIs nutzen, die in einem Netzwerk ausgeführt werden.
- Veröffentlichen von Frontpanel-Bildern und VI-Dokumentation im Web.
- Erstellen von VIs, die über Low-Level-Protokolle, wie zum Beispiel TCP, UDP, DDE, Apple-Ereignisse und PPC-Toolbox, mit anderen Applikationen und VIs kommunizieren.

Weitere Informationen ...

Weitere Informationen zum Arbeiten mit LabVIEW im Netzwerk finden Sie in der *LabVIEW-Hilfe*.

Auswahl zwischen Datei-I/O, VI-Server, ActiveX und Arbeiten im Netzwerk

Ein Arbeiten im Netzwerk bildet eventuell nicht die optimale Lösung für eine Applikation. Wenn Sie eine Datei erstellen möchten, die Daten enthält, die andere VIs und Applikationen lesen können, verwenden Sie Datei-I/O-VIs und -Funktionen. Weitere Informationen zum Verwenden der Datei-I/O-VIs und -Funktionen finden Sie in Kapitel 13, *Datei-I/O*.

Verwenden Sie den VI-Server, wenn Sie andere VIs steuern möchten. Weitere Informationen zum Steuern von VIs und anderen LabVIEW-Applikationen auf lokalen Computern oder Netzwerkrechnern finden Sie in Kapitel 16, *Programmatische Steuerung von VIs*.

(Windows) Wenn Sie auf die Merkmale von Microsoft-Anwendungen zugreifen möchten, wie zum Beispiel Einbetten eines Kurvengraphen in eine Excel-Tabelle, verwenden Sie ActiveX-VIs und -Funktionen. Weitere Informationen zum Zugriff auf ActiveX-fähige Applikationen und Zulassen des Zugriffs von anderen ActiveX-Applikationen auf LabVIEW finden Sie in Kapitel 18, [ActiveX](#).

LabVIEW als Netzwerk-Client und -Server

Sie können LabVIEW als Client verwenden, um Daten zu abonnieren und Merkmale in anderen Applikationen zu verwenden, beziehungsweise als Server, um LabVIEW-Eigenschaften anderen Applikationen zur Verfügung zu stellen. Weitere Informationen zum Verwenden des VI-Servers, um VIs auf lokalen Computern oder Netzwerkrechnern zu steuern, finden Sie in Kapitel 16, [Programmatische Steuerung von VIs](#). Sie steuern VIs, indem Sie mit dem Eigenschaftsknoten auf Eigenschaften zugreifen beziehungsweise mit dem Methodenknoten Methoden aufrufen.

Bevor Sie auf die Eigenschaften und Methoden einer anderen Applikation zugreifen können, müssen Sie das verwendete Netzwerkprotokoll einrichten, mit dem der Zugriff auf die Eigenschaften und Methoden erfolgt. Sie können beispielsweise HTTP und TCP/IP als Protokolle verwenden. Das ausgewählte Protokoll hängt von der Applikation ab. Das HTTP-Protokoll eignet sich beispielsweise ideal für die Veröffentlichung im Web. Das HTTP-Protokoll kann jedoch nicht verwendet werden, um ein VI zu erstellen, das von einem anderen VI erzeugte Daten empfängt. Verwenden Sie dazu das TCP-Protokoll.

Weitere Informationen zu den von LabVIEW unterstützten Kommunikationsprotokollen finden Sie in Abschnitt [Low-Level-Kommunikationsanwendungen](#) dieses Kapitels.

(Windows) Weitere Information zum Verwenden der ActiveX-Technologie mit LabVIEW als ActiveX-Server oder -Client finden Sie in Kapitel 18, [ActiveX](#).

Verwenden der DataSocket-Technologie

Mit der National Instruments DataSocket-Technologie können Sie Direktübertragungsdaten gemeinsam mit anderen VIs und anderen Applikationen, wie zum Beispiel National Instruments ComponentWorks, im Web oder auf dem lokalen Computer nutzen. DataSocket fasst anerkannte Kommunikationsprotokolle für Messtechnik und Automatisierung zusammen, ähnlich wie ein Webbrowser unterschiedliche Internet-Technologien zusammenfasst.

Die DataSocket-Technologie ermöglicht im Frontpanel über das Dialogfeld **DataSocket Verbindung** Zugriff auf mehrere Ein- und Ausgabe-mechanismen. Klicken Sie mit der rechten Maustaste auf ein Frontpanel-Objekt und wählen aus dem Kontextmenü **Datenoperationen»DataSocket-Verbindung**, um das Dialogfeld **DataSocket-Verbindung** anzuzeigen. Sie veröffentlichen (schreiben) oder abonnieren (lesen) Daten, indem Sie eine URL angeben, ungefähr so wie die URLs in einem Webbrowser.

Wenn Sie beispielsweise die Daten in einem Thermometeranzeigeelement auf dem Frontpanel mit anderen Computern im Web gemeinsam nutzen möchten, veröffentlichen Sie die Thermometerdaten, indem Sie im Dialogfeld **DataSocket-Verbindung** eine URL angeben. Die Benutzer der anderen Computer abonnieren die Daten, indem sie ein Thermometer in ihr Frontpanel einfügen und die URL im Dialogfeld **DataSocket-Verbindung** auswählen. Weitere Informationen zum Verwenden der DataSocket-Technologie im Frontpanel finden Sie in Abschnitt *Frontpanel-DataSocket* dieses Kapitels.

Weitere Informationen zur DataSocket-Technologie finden Sie in der wissenschaftlichen Ausarbeitung *Integrating the Internet into Your Measurement System*. Diese Ausarbeitung steht als PDF-Datei auf der Installations-CD im Verzeichnis `manuals` oder auf der National Instruments-Website unter `ni.com` zur Verfügung.

Angeben einer URL

URLs verwenden Kommunikationsprotokolle, wie zum Beispiel `dstp`, `ftp` und `file`, um Daten zu übertragen. Das in einer URL verwendete Protokoll hängt von der Art der Daten ab, die veröffentlicht werden sollen, und von der Konfiguration des Netzwerks.

Beim Veröffentlichen oder Abonnieren von Daten mit DataSocket können Sie die folgenden Protokolle verwenden:

- **DataSocket Transport Protocol (`dstp`)**: Das systemspezifische Protokoll für DataSocket-Verbindungen. Wenn Sie dieses Protokoll verwenden, kommuniziert das VI mit dem DataSocket-Server. Sie müssen die Daten mit einem benannten Tag versehen, das an die URL angehängt wird. Die DataSocket-Verbindung verwendet das benannte Tag, um ein bestimmtes Datenelement auf einem DataSocket-Server zu adressieren. Um dieses Protokoll zu verwenden, muss ein DataSocket-Server ausgeführt werden.
- **(Windows) OLE for Process Control (`opc`)**: Dient speziell für die gemeinsame Nutzung von Echtzeitproduktionsdaten, wie zum Beispiel Daten, die von industriellen Automatisierungsvorgängen erzeugt werden. Um dieses Protokoll zu verwenden, muss ein OPC-Server ausgeführt werden.
- **(Windows) logos**: Eine interne National Instruments-Technologie für die Übertragung von Daten zwischen Netzwerk und dem lokalen Computer.
- **File Transfer Protocol (`ftp`)**: Bei diesem Protokoll können Sie eine Datei angeben, aus der Daten gelesen werden sollen.
- **file**: Mit diesem Protokoll können Sie eine Verbindung zu einer lokalen oder Netzwerkdatei bereitstellen, die Daten enthält.

Die Tabelle 17-1 zeigt Beispiele für die einzelnen Protokoll-URLs.

Tabelle 17-1. Beispiele für DataSocket-URLs

URL	Beispiel
dstp	dstp://servername.com/numerische_Daten, wobei <i>numerische_Daten</i> den benannten Tag darstellt
opc	opc:\National Instruments.OPCTest\objekt1 opc:\\machine\National Instruments.OPCModbus\Modbus Demo Box.4:0 opc:\\machine\National Instruments.OPCModbus\Modbus Demo Box.4:0?updaterate=100&deadband=0.7
logos	logos://computer_name/process/datenobjektname
ftp	ftp://ftp.natinst.com/datasocket/ping.wav
file	file:ping.wav file:c:\meinedaten\ping.wav file:\\rechner\meinedaten\ping.wav

Mit den `dstp`-, `opc`- und `logos`-URLs können Sie Direktübertragungsdaten gemeinsam nutzen, da diese Protokolle Bedien- und Anzeigeelemente lokal und im Netzwerk aktualisieren können. Mit den `ftp`- und `file`-URLs können Sie Daten aus Dateien lesen, da diese Protokolle keine Bedien- und Anzeigeelemente lokal und im Netzwerk aktualisieren können.

Beispiele für die Verwendung von DataSocket-Verbindungen finden Sie in der Datei `examples/comm/datasktx.llb`.

Unterstützte DataSocket-Datenformate

Mit DataSocket können Sie die folgenden Daten veröffentlichen und abonnieren:

- **Unformatierter Text:** Verwenden Sie unformatierten Text, um einen String an ein String-Anzeigeelement zu übermitteln.
- **Text mit Tabulatortrennzeichen:** Verwenden Sie Text mit Tabulatortrennzeichen, wie in einer Tabelle, um Daten in Arrays zu veröffentlichen. LabVIEW interpretiert Text mit Tabulatortrennzeichen als Daten-Array.

- **wav-Daten:** Verwenden Sie .wav-Daten, um Sound für ein VI oder eine Funktion zu veröffentlichen.
- **Variant-Daten:** Verwenden Sie Variant-Daten, um Daten aus einer anderen Applikation, wie zum Beispiel einem Component-Works-Bedienelement, zu abonnieren.

Verwenden von DataSocket im Frontpanel

Mit Frontpanel-DataSocket-Verbindungen können Sie Direktübertragungsdaten in einem Frontpanel-Objekt veröffentlichen oder abonnieren. Wenn Sie die Daten eines Frontpanel-Objekts mit anderen Benutzern gemeinsam nutzen, veröffentlichen Sie die Daten. Wenn Benutzer die veröffentlichten Daten abrufen und in ihrem Frontpanel anzeigen, abonnieren die Benutzer die Daten.

DataSocket-Verbindungen unterscheiden sich von Webserver- und ActiveX-Verbindungen, da Sie DataSocket-Verbindungen direkt im Frontpanel ohne Programmierung eines Blockdiagramms verwenden können. Jedes Frontpanel-Bedien- oder -Anzeigeelement kann Daten über eine eigene DataSocket-Verbindung veröffentlichen oder abonnieren. Frontpanel-DataSocket-Verbindungen veröffentlichen nur die Daten, aber keine Grafik des Frontpanel-Bedienelements, sodass VIs, welche die Daten über eine DataSocket-Verbindung abonnieren, für die Daten eigene Operationen durchführen können.

Sie können den Wert eines Frontpanel-Bedienelements direkt im Frontpanel einstellen und anschließend die Daten veröffentlichen. Sie können jedoch auch ein Blockdiagramm erstellen, den Ausgang eines VIs oder einer Funktion mit einem Anzeigeelement verbinden und die Daten vom Anzeigeelement aus veröffentlichen. DataSocket-Verbindungen werden normalerweise in Szenarios verwendet, in denen Bedien- und Anzeigeelemente die folgenden Aufgaben erfüllen:

- Veröffentlichen eines Werts von einem Frontpanel-Bedienelement, um ein Bedienelement zu bearbeiten und die Daten für andere Benutzer zu veröffentlichen, die über ein Bedien- oder Anzeigeelement abonniert werden können. Sie können beispielsweise auf dem Frontpanel einen Drehknopf einfügen, der die Temperatur erhöht oder verringert, und ein Benutzer an einem anderen Computer kann die Daten abonnieren und in einem Bedienelement verwenden, das mit einem Sub-VI oder einer Funktion verbunden ist, oder die Daten in einem Anzeigeelement anzeigen. Ein Benutzer kann ein Frontpanel-Bedienelement nicht mit dem eigenen VI verändern, wenn das Bedienelement die Daten über eine DataSocket-Verbindung abonniert hat.

- Veröffentlichen eines Werts, der in einem Frontpanel-Anzeigeelement angezeigt wird, sodass ein anderer Benutzer die Daten abonnieren und in einem Bedien- oder Anzeigeelement in dem eigenen Frontpanel anzeigen oder die Ergebnisse als Daten in einem Bedienelement verwenden kann, das in einem Sub-VI oder einer Funktion mit einem Eingang verbunden ist. Ein VI, das die mittlere Temperatur berechnet und die Temperatur in einem Thermometer auf dem Frontpanel anzeigt, kann beispielsweise die Temperaturdaten veröffentlichen.
- Abonnieren eines Werts aus einem Frontpanel-Anzeigeelement, um die Daten in einem Frontpanel-Anzeigeelement in einem VI anzuzeigen, das in einem Bedien- oder Anzeigeelement auf dem Frontpanel eines anderen VIs angezeigt wird.
- Abonnieren eines Werts aus einem Frontpanel-Bedienelement, um die Daten in einem Frontpanel-Bedienelement in einem VI anzuzeigen, das in einem Bedien- oder Anzeigeelement auf dem Frontpanel eines anderen VIs angezeigt wird. Wenn Sie die Daten über ein Bedienelement abonnieren, können Sie die Daten in einem VI verwenden, wenn das Bedienelement mit dem Eingang eines Sub-VIs oder einer Funktion verbunden wird.
- Veröffentlichen aus und Abonnieren von einem Frontpanel-Bedienelement, sodass Benutzer ein Bedienelement im Frontpanel Ihres VIs von den Frontpanels ihrer VIs aus verändern können. Wenn Sie das VI ausführen, ruft das Frontpanel-Bedienelement in Ihrem VI den aktuellen Wert ab, der von einem anderen VI oder einer anderen Applikation über die DataSocket-Verbindung veröffentlicht wurde. Wenn ein Benutzer den Wert des Bedienelements im eigenen Frontpanel ändert, veröffentlicht die DataSocket-Verbindung den neuen Wert im Frontpanel-Bedienelement Ihres VIs. Wenn Sie anschließend den Wert Ihres Frontpanel-Bedienelements ändern, wird der Wert von Ihrem VI an die Frontpanel der anderen Benutzer veröffentlicht.

Die Frontpanel-Objekte, die Daten abonnieren, brauchen nicht mit der Art des Objekts überein zu stimmen, das die Daten veröffentlicht. Die Frontpanel-Objekte müssen jedoch denselben Datentyp besitzen, beziehungsweise bei numerischen Daten den denselben Datentyp erzwingen. Sie können beispielsweise in Ihrem VI ein digitales Anzeigeelement verwenden, um die Daten eines Thermometers anzuzeigen, die in einem anderen VI erzeugt werden. Bei dem Thermometer kann es sich um eine Fließkommazahl handeln, und bei dem digitalen Anzeigeelement um ein Ganzzahl.

Frontpanel-DataSocket-Verbindungen dienen hauptsächlich dazu, Direktübertragungsdaten gemeinsam zu nutzen. Um Daten in lokalen Dateien, auf FTP-Servern oder Webservern zu lesen, verwenden Sie die Funktion

„DataSocket-Lesen“ in der Palette **Funktionen»Kommunikation»DataSocket**, die Datei-I/O-VIs und -Funktionen in der Palette **Funktionen»Datei-I/O** oder die Applikationssteuerungs-VIs und -Funktionen in der Palette **Funktionen»Applikationssteuerung**. National Instruments empfiehlt, für Frontpanel-Direktaktualisierungen nur unformatierten Text und Variant-Daten zu verwenden.

Lesen und Schreiben von Direktübertragungsdaten über das Blockdiagramm

Im Blockdiagramm können Sie Daten mit Hilfe der DataSocket-Funktionen programmatisch lesen oder schreiben, die sich in der Palette **Funktionen»Kommunikation»DataSocket** befinden.

Mit der Funktion „DataSocket-Schreiben“ können Sie Direktübertragungsdaten programmatisch über eine DataSocket-Verbindung schreiben. Abbildung 17-1 zeigt, wie ein numerischer Wert geschrieben wird.

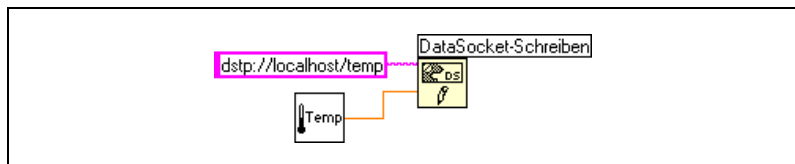


Abbildung 17-1. Veröffentlichen von Daten mit „DataSocket-Schreiben“

Die Funktion „DataSocket-Schreiben“ ist polymorph, sodass Sie die meisten Datentypen mit dem Eingang **Daten** verbunden werden können. Weitere Informationen zu polymorphen VIs und Funktionen finden Sie in Abschnitt *Polymorphe VIs und Funktionen* des Kapitels 5, *Erstellen des Blockdiagramms*.

Mit der Funktion „DataSocket-Lesen“ können Sie Direktübertragungsdaten programmatisch über eine DataSocket-Verbindung lesen. Abbildung 17-2 zeigt, wie Daten gelesen und in eine Fließkommazahl doppelter Genauigkeit konvertiert werden.

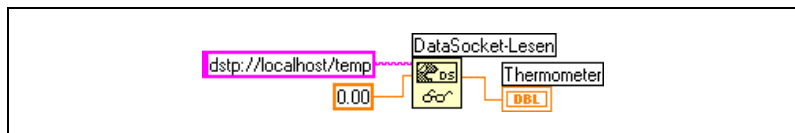


Abbildung 17-2. Lesen eines einzelnen Werts mit „DataSocket-Lesen“

Sie konvertieren die Direktübertragungsdaten in einen bestimmten Datentyp, indem Sie ein Bedienelement oder eine Konstante mit dem Eingang **Typ** von „DataSocket-Lesen“ verbinden. Ohne Angabe eines Typs gibt der Ausgang **Daten** von „DataSocket-Lesen“ Variant-Daten zurück, die mit der Funktion „Variant in Daten“ konvertiert werden müssen, die sich in der Palette **Funktionen»Kommunikation»DataSocket»Variant** befindet. In einigen Fällen müssen Sie Variant-Daten in LabVIEW-Daten konvertieren.

DataSocket und Variant-Daten

In einigen Fällen kann das VI oder die Applikation, das/die die Daten programmatisch liest, die Daten nicht in den ursprünglichen Datentyp zurückwandeln, beispielsweise wenn Daten aus einer anderen Applikation abonniert werden. Darüber hinaus müssen Sie eventuell ein Attribut zu den Daten hinzufügen, wie zum Beispiel eine Zeitmarkierung oder eine Warnung, welche die Datentypen nicht zulassen.

In diesen Fällen können Sie mit der Funktion „In Variant“, die sich in der Palette **Funktionen»Kommunikation»DataSocket»Variant** befindet, die an eine DataSocket-Verbindung geschriebenen Daten programmatisch in Variant-Daten konvertieren. Abbildung 17-3 zeigt ein Blockdiagramm, das kontinuierlich einen Temperaturmesswert erfasst, die Daten in Variant-Daten konvertiert und eine Zeitmarkierung als Attribut zu den Daten hinzufügt.

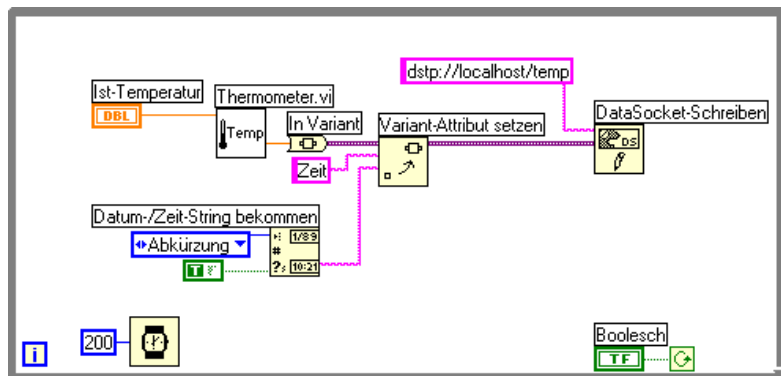


Abbildung 17-3. Konvertieren von Direktübertragungstemperaturdaten in Variant-Daten

Wenn ein VI die Direktübertragungsdaten liest, muss das VI die Variant-Daten in einen Datentyp konvertieren, der verarbeitet werden kann. Abbildung 17-4 zeigt ein Blockdiagramm, das kontinuierlich Temperaturdaten von einer DataSocket-Verbindung liest, die Variant-Daten in Temperaturmesswerte konvertiert, das zu den einzelnen Messwerten gehörende Zeitmarkierungsattribut abrufen und Temperatur und Zeitmarkierung auf dem Frontpanel anzeigt.

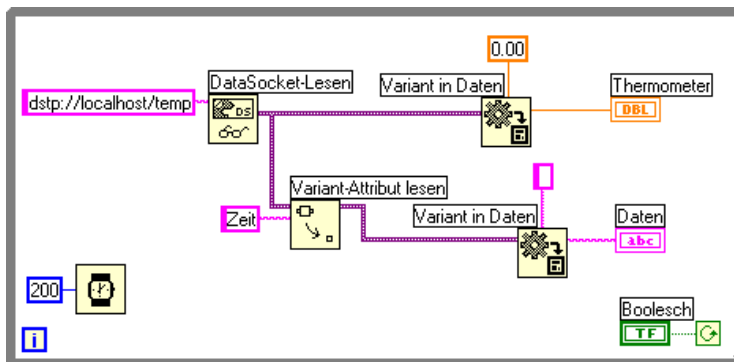


Abbildung 17-4. Konvertieren von Direktübertragungs-Variant-Daten in Temperaturdaten

Veröffentlichen von VIs im Web

Mit Hilfe des LabVIEW-Webservers können Sie HTML-Dokumente erstellen und Frontpanel-Bilder im Web veröffentlichen. Sie können den Browser-Zugriff auf die veröffentlichten Frontpanels steuern und konfigurieren, welche VIs im Web sichtbar sind.



Hinweis Mit dem LabVIEW Enterprise Connectivity Toolset können Sie VIs im Web steuern und zusätzliche Sicherheitsmerkmale zu VIs hinzufügen, die im Web veröffentlicht werden. Weitere Informationen zu diesem Toolset finden Sie auf der National Instruments-Website unter ni.com.

Webserver-Optionen

Wählen Sie **Werkzeuge»Optionen** und aus dem oberen Pulldown-Menü eines der **Webserver**-Elemente, um die folgenden Optionen einzustellen:

- Einrichten von Stammverzeichnis und Protokolldatei.
- Aktivieren des Webservers.

- Steuern des Browser-Zugriffs auf VI-Frontpanel.
- Konfigurieren, welche VI-Frontpanel im Web sichtbar sind.

Sie müssen den Webserver auf der Seite **Webserver: Konfiguration** des Dialogfelds **Werkzeuge»Optionen** aktivieren, bevor Sie VIs im Web veröffentlichen können. Sie können den Webserver auch mit dem Web Publishing Tool aktivieren, das im folgenden Abschnitt beschrieben wird. Die VIs müssen sich vor dem Veröffentlichen im Speicher befinden.

Erstellen von HTML-Dokumenten

Wählen Sie **Werkzeuge»Web-Dokumentationswerkzeug**, um mit dem Web-Dokumentationswerkzeug die folgenden Aufgaben durchzuführen:

- Erstellen eines HTML-Dokuments
- Einbetten statischer oder animierter Bilder des Frontpanels in ein HTML-Dokument.
- Hinzufügen von Text über und unter dem eingebetteten VI-Frontpanel-Bild.
- Einfügen eines Rahmen um das VI-Frontpanel-Bild.
- Animieren des Bilds. Zurzeit unterstützen nur Netscape-Browser animierte Bilder.
- Voransicht des Dokuments.
- Speichern des Dokuments auf der Festplatte.
- Aktivieren des Webservers zum Veröffentlichen von HTML-Dokumenten und Frontpanel-Bildern im Web.

Veröffentlichen von Frontpanel-Bildern

Mit einer `.snap`-URL können Sie in einem Webbrowser oder einem HTML-Dokument ein statisches Abbild des Frontpanels eines VIs zurückgeben, das sich momentan im Speicher befindet. Die Abfrageparameter in der URL geben VI-Name und die Attribute des Bilds an. Verwenden Sie zum Beispiel `http://web.server.adresse/.snap?VI-Name.vi`, wobei `VI-Name.vi` für den Namen des VIs steht, das angezeigt werden soll.

Mit einer `.monitor`-URL können Sie ein animiertes Abbild des Frontpanels eines VIs zurückgeben, das sich momentan im Speicher befindet. Die Abfrageparameter in der URL geben VI-Name, Attribute der Animation und die Attribute des Bilds an. Verwenden Sie zum Beispiel `http://web.server.adresse/.monitor?VI-Name.vi`, wobei `VI-Name.vi` für den Namen des VIs steht, das angezeigt werden soll.

Frontpanel-Bildformate

Der Webserver kann Frontpanel-Bilder im JPEG- und PNG-Bildformat erzeugen.

Das JPEG-Format komprimiert Grafiken. Dabei können jedoch einige Grafikdetails verloren gehen. Dieses Format eignet sich optimal für Fotos. Bei Strichzeichnungen, Frontpanels und Blockdiagrammen kann die JPEG-Komprimierung zu verschwommenen Bildern und ungleichmäßigen Farben führen. Das PNG-Format komprimiert Grafiken ebenfalls, wenn auch nicht immer so gut wie beim JPEG-Format. Die PNG-Komprimierung führt jedoch nicht zu einem Detailverlust.

Low-Level-Kommunikationsanwendungen

LabVIEW unterstützt mehrere Low-Level-Protokolle, die für die Kommunikation zwischen Computern verwendet werden können.

Jedes Protokoll ist verschieden, insbesondere in der Hinsicht, wie der Verweis auf die Netzwerkadresse einer Netzwerkanwendung erfolgt. Jedes Protokoll ist normalerweise inkompatibel zu anderen Protokollen. Wenn Sie beispielsweise zwischen Macintosh und Windows kommunizieren möchten, müssen Sie ein Protokoll verwenden, das auf beiden Plattformen funktioniert, wie zum Beispiel TCP.

TCP und UDP

TCP (Transmission Control Protocol) und UDP (User Datagram Protocol) stehen auf allen von LabVIEW unterstützten Plattformen zur Verfügung. Bei TCP handelt es sich um ein zuverlässiges, verbindungsorientiertes Protokoll. Es bietet Fehlererkennung und gewährleistet, dass die Daten in der richtigen Reihenfolge ohne Duplizierung ankommen. Aus diesen Gründen bildet TCP normalerweise die optimale Auswahl für Netzwerkanwendungen.

Obwohl UDP eine höhere Leistung als TCP bieten kann, ist keine Verbindung erforderlich und keine Übermittlung garantiert. Normalerweise wird UDP in Anwendungen verwendet, bei denen eine garantierte Übermittlung ohne Bedeutung ist. Eine Anwendung kann beispielsweise Daten häufig genug an ein Ziel senden, sodass einige wenige verlorene Datensegmente unproblematisch sind. Weitere Informationen zum Verwenden von TCP und UDP in LabVIEW-Anwendungen finden Sie in der Applikationsinformation *Using LabVIEW with TCP/IP and UDP*.

DDE (Windows)

Der dynamische Datenaustausch (DDE) arbeitet auf einer höheren Ebene als TCP, um Befehle und Daten zwischen Clients und Servern auszutauschen. DDE eignet sich insbesondere bei der Kommunikation mit Standardanwendungen wie Microsoft Excel. Weitere Informationen zum Verwenden von DDE in LabVIEW-Applikationen finden Sie in der Applikationsinformation *Using DDE in LabVIEW*.

Apple-Ereignisse und PPC-Toolbox (Macintosh)

Eine häufige Form der Kommunikation, die nur bei Macintosh möglich ist, sind Apple-Ereignisse. Mit Apple-Ereignissen können Sie wie bei DDE Nachrichten senden, die Aktionen anfordern oder Informationen von anderen Macintosh-Anwendungen zurückgeben.

Bei der PPC-Toolbox (Programm-zu-Programm-Kommunikation) handelt es sich um eine untergeordnete Form, Daten zwischen Macintosh-Anwendungen zu senden und zu empfangen. Die PPC-Toolbox bietet eine höhere Leistung als Apple-Ereignisse, da der zum Übertragen von Informationen erforderliche Overhead geringer ist. Da die PPC-Toolbox jedoch nicht die Art der übertragbaren Informationen festlegt, wird sie von vielen Applikationen nicht unterstützt. Die PPC-Toolbox eignet sich optimal zum Senden umfangreicher Informationen zwischen Applikationen, welche die PPC-Toolbox unterstützen. Weitere Informationen zum Verwenden von Apple-Ereignissen und der PPC-Toolbox in LabVIEW-Applikationen finden Sie in der Applikationsinformation *Using Apple Events and the PPC Toolbox to Communicate with LabVIEW Applications on the Macintosh*.

Pipe-VIs (UNIX)

Mit den Pipe-VIs können Sie UNIX-Named Pipes öffnen, schließen, lesen und schreiben. Mit Named Pipes können Sie mit LabVIEW und nicht verwandten Prozessen kommunizieren.

Ausführen von Befehlen auf Systemebene (Windows und UNIX)

Mit „System Exec.vi“ können Sie in VIs andere Windows-Anwendungen oder UNIX-Befehlszeilenanwendungen ausführen oder starten. Mit „System Exec.vi“ führen Sie eine Befehlszeile auf Systemebene aus, die beliebige Parameter enthält, die von der zu startenden Anwendung unterstützt werden.

ActiveX

Mit Hilfe der ActiveX-Automation stellt eine Windows-Applikation, wie zum Beispiel LabVIEW, Objekte, Befehle und Funktionen öffentlich zur Verfügung, auf die andere Windows-Applikationen zugreifen können. Sie können LabVIEW als ActiveX-Client einsetzen, um auf die Objekte, Eigenschaften, Methoden und Ereignisse zuzugreifen, die von anderen ActiveX-fähigen Applikationen stammen. LabVIEW kann auch als ActiveX-Server fungieren, sodass andere ActiveX-fähige Applikationen auf LabVIEW-Objekte, -Eigenschaften und -Methoden zugreifen können.

In diesem Handbuch bezieht sich *ActiveX* auf die ActiveX- und OLE-Technologie von Microsoft. Diese Technologie steht nur unter Windows zur Verfügung. Weitere Informationen zu ActiveX finden Sie in der Microsoft Developer's Network-Dokumentation *Inside OLE*, von Craig Brockschmidt, zweite Ausgabe, und *Essential COM*, von Don Box.

Weitere Informationen ...

Weitere Informationen zur Verwendung der ActiveX-Technologie finden Sie in der *LabVIEW-Hilfe*.

ActiveX-Objekte, -Eigenschaften, -Methoden und -Ereignisse

ActiveX-fähige Applikationen umfassen Objekte, die Eigenschaften und Methoden bereitstellen, auf die andere Applikationen zugreifen können. Objekte können für den Benutzer sichtbar sein, wie zum Beispiel Schaltflächen, Fenster, Bilder, Dokumente und Dialogfelder, oder unsichtbar, wie zum Beispiel Registrierungsobjekte von Applikationen. Sie greifen auf eine Applikation zu, indem Sie auf ein Objekt zugreifen, das der betreffenden Applikation zugeordnet ist, und eine Eigenschaft einstellen oder eine Methode des jeweiligen Objekts aufrufen.

Weitere Informationen zu Objekten, Eigenschaften und Methoden finden Sie in dem Abschnitt *Bearbeiten von Applikations- und VI-Einstellungen* des Kapitels 16, *Programmatische Steuerung von VIs*.

Ereignisse sind die Aktionen, die für ein Objekt durchgeführt werden, wie zum Beispiel das Klicken mit der Maus, das Drücken einer Taste oder das Überlaufen des Speichers. Immer dann, wenn eine dieser Aktionen für das Objekt auftritt, sendet das Objekt ein Ereignis, um den ActiveX-Container zusammen mit ereignisspezifischen Daten darauf hinzuweisen.

Beispiele für die Verwendung von ActiveX-Ereignissen in LabVIEW und Beispiele für die Verwendung von ActiveX-Ereignis-VIs finden Sie in „ActiveX Event Template.vi“ und „List ActiveX Events.vi“ in der Datei `examples\comm\axevent.llb`, sowie in „FamilyTree.vi“ und „MultiEvents.vi“ im Verzeichnis `examples\comm`.

ActiveX-VIs, -Funktionen, -Bedienelemente und -Anzeigeelemente

Verwenden Sie die folgenden VIs, Funktionen, Bedien- und Steuerelemente, um auf die Objekte, Eigenschaften, Methoden und Ereignisse zuzugreifen, die von anderen ActiveX-fähigen Applikationen stammen:

- Verwenden Sie die ActiveX-Objekt-Refnum, zu finden in der Elementpalette unter **ActiveX**, um eine Referenz auf ein ActiveX-Objekt zu erstellen. Klicken Sie im Frontpanel mit der rechten Maustaste auf dieses Bedienelement, um ein Objekt aus der ActiveX-Klasse auszuwählen, auf die Sie zugreifen möchten.
- Verwenden Sie die Funktion „ActiveX-Objekt öffnen“ in der Funktionspalette unter **Kommunikation»ActiveX**, um ein ActiveX-Objekt zu öffnen.
- Verwenden Sie den ActiveX-Container in der Elementpalette unter **ActiveX**, um auf ein ActiveX-Objekt zuzugreifen und es auf dem Frontpanel anzuzeigen. Klicken Sie mit der rechten Maustaste auf dieses Bedienelement, um das Objekt auszuwählen, auf das Sie zugreifen möchten.
- Verwenden Sie den Eigenschaftsknoten in der Funktionspalette unter **Kommunikation»ActiveX**, um die zu einem ActiveX-Objekt gehörenden Eigenschaften abzurufen (zu lesen) und einzustellen (zu schreiben).
- Verwenden Sie den Methodenknoten in der Funktionspalette unter **Kommunikation»ActiveX**, um die zu einem ActiveX-Objekt gehörenden Methoden aufzurufen.

- Verwenden Sie die ActiveX-Ereignis-VIs in der Funktionspalette unter **Kommunikation»ActiveX»ActiveX-Ereignisse**, um die Ereignisse zu verwalten, die für ein ActiveX-Objekt auftreten, das in den ActiveX-Container auf dem Frontpanel eingefügt wurde.
- Verwenden Sie den ActiveX-Variant in der Elementpalette unter **ActiveX** zur Übergabe von Daten an das und von dem Sub-VI. Sie können damit auch Daten anzeigen, aber nicht ändern. Weitere Informationen zu Variant-Daten finden Sie in dem Abschnitt *Verarbeiten von Variant-Daten* des Kapitels 5, *Erstellen des Blockdiagramms*.

LabVIEW als ActiveX-Client

Wenn LabVIEW auf die Objekte einer anderen ActiveX-fähigen Applikation zugreift, fungiert LabVIEW als ActiveX-Client. Sie können LabVIEW folgendermaßen als ActiveX-Client verwenden:

- Öffnen Sie eine Applikation, wie zum Beispiel Microsoft Excel, erstellen Sie ein Dokument und fügen Sie Daten in das Dokument ein.
- Betten Sie im Frontpanel ein Dokument ein, beispielsweise ein Microsoft Word-Dokument oder eine Excel-Tabelle.
- Fügen Sie im Frontpanel eine Schaltfläche oder ein anderes Objekt aus einer anderen Applikation ein, wie zum Beispiel eine Schaltfläche **Hilfe**, welche die Hilfedatei der anderen Applikation aufruft.
- Erstellen Sie eine Verknüpfung zu einem ActiveX-Objekt, das mit einer anderen Applikation erstellt wurde.

LabVIEW greift über die ActiveX-Objekt-Refnum und den ActiveX-Container, bei denen es sich jeweils um Frontpanel-Objekte handelt, auf ein ActiveX-Objekt zu. Wählen Sie mit der ActiveX-Objekt-Refnum ein ActiveX-Objekt aus. Verwenden Sie den ActiveX-Container zur Auswahl eines ActiveX-Objekts, das angezeigt werden kann, wie zum Beispiel eine Schaltfläche oder ein Dokument, und fügen Sie es im Frontpanel ein. Beide Objekte werden im Blockdiagramm als ActiveX-Objekt-Refnums angezeigt.

Zugriff auf eine ActiveX-fähige Applikation

Verwenden Sie die ActiveX-Objekt-Refnum im Blockdiagramm, um eine Referenz auf eine ActiveX-fähige Applikation zu erstellen und auf die Applikation zuzugreifen. Verbinden Sie das Objekt mit der Funktion „ActiveX-Objekt öffnen“, welche die aufrufende Applikation öffnet. Verwenden Sie den Eigenschaftsknoten, um die zu dem Objekt gehörenden Eigenschaften auszuwählen und darauf zuzugreifen. Verwenden Sie den Methodenknoten, um die zu dem Objekt gehörenden Methoden auszuwählen und darauf zuzugreifen. Schließen Sie die Referenz auf das Objekt mit Hilfe der Funktion „ActiveX-Objekt schließen“. Durch Schließen des Verweises wird das Objekt aus dem Speicher entfernt.

Sie können beispielsweise ein VI erstellen, das Microsoft Excel öffnet, sodass Excel auf dem Bildschirm des Benutzers angezeigt wird, ein Arbeitsblatt erstellt, ein Spreadsheet erstellt, eine Tabelle in LabVIEW erstellt und diese Tabelle in das Excel-Spreadsheet geschrieben wird.

Ein Beispiel zur Verwendung von LabVIEW als Excel-Client finden Sie in „Write Table To XL.vi“ in der Datei `examples\comm\ExcelExamples.llb`.

Einfügen eines ActiveX-Objekts oder -Dokuments im Frontpanel

Sie fügen ein ActiveX-Objekt oder -Dokument im Frontpanel mit dem ActiveX-Container ein, um das ActiveX-Objekt oder -Dokument auszuwählen und auf die zu dem Objekt gehörenden Eigenschaften und Methoden zuzugreifen. Verwenden Sie den Eigenschaftsknoten, um auf die zu dem Objekt gehörenden Eigenschaften zuzugreifen. Verwenden Sie den Methodenknoten, um die zu dem Objekt gehörenden Methoden aufzurufen.

Sie können auf dem Frontpanel mit Hilfe eines ActiveX-Containers beispielsweise eine Internetseite anzeigen, indem Sie auf das Objekt „Microsoft Web Browser“ zugreifen, die Methodenklasse „Navigate“ und die Methode „URL“ auswählen und den URL angeben.

Wenn Sie den ActiveX-Container verwenden, brauchen Sie die ActiveX-Objekt-Refnum im Blockdiagramm nicht mit der Funktion „ActiveX-Objekt öffnen“ zu verbinden oder die Referenz auf das Objekt mit der Funktion „ActiveX-Objekt schließen“ zu schließen. Sie können eine direkte Verbindung zum Methoden- oder Eigenschaftsknoten erstellen, da der ActiveX-Container die aufrufende Applikation in LabVIEW einbettet. Enthält der ActiveX-Container jedoch Eigenschaften oder Methoden, die andere ActiveX-Objekt-Refnums zurückgeben, müssen Sie die betreffenden anderen Refnum-Objekte schließen.

LabVIEW als ActiveX-Server

Die LabVIEW-Applikation, VIs sowie Eigenschaften und Methoden von Objekten stehen über ActiveX-Aufrufe anderen Applikationen zur Verfügung. Andere ActiveX-fähige Applikationen, wie zum Beispiel Microsoft Excel, können Eigenschaften, Methoden und einzelne VIs von LabVIEW anfordern, wobei LabVIEW als ActiveX-Server fungiert.

Sie können beispielsweise einen VI-Graphen in eine Excel-Tabelle einfügen, die Daten für die VI-Eingänge in der Tabelle eingeben und das VI ausführen. Beim Ausführen des VIs werden die Daten im Graphen dargestellt.

Ein Beispiel für die Verwendung der LabVIEW-Eigenschaften und -Methoden in einer Excel-Tabelle finden Sie in `examples\comm\freqresp.xls`.

Verwenden von Konstanten zum Festlegen von Parametern in ActiveX-VIs

Einige Parameter in ActiveX-Knoten übernehmen eine diskrete Liste zulässiger Werte. Wählen Sie den beschreibenden Namen in der Ring-Konstante, um die jeweiligen Parameterwerte festzulegen. Um beim Erstellen eines ActiveX-VIs auf die Ring-Konstante zuzugreifen, klicken Sie mit der rechten Maustaste auf den Parameter des Knotens, der die Datenwerte entgegennimmt, und wählen aus dem Kontextmenü **Konstante erzeugen**. Die in der Ring-Konstante verfügbaren Auswahlmöglichkeiten hängen von der an den Knoten übergebenen Refnum ab. Die Abbildungen 18-1 und 18-2 zeigen Beispiele für die Verwendung der Ring- und numerischen Konstanten zur Einstellung von Parameterwerten.

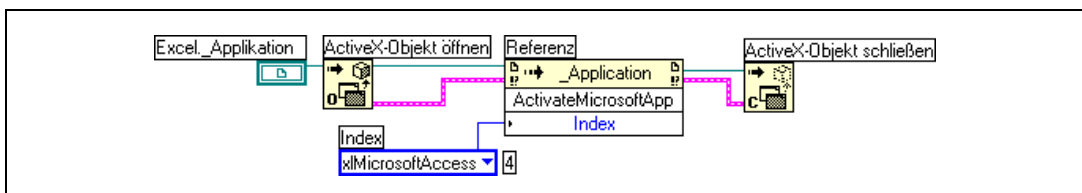


Abbildung 18-1. Festlegen eines Datenwerts mit einer Ring-Konstante

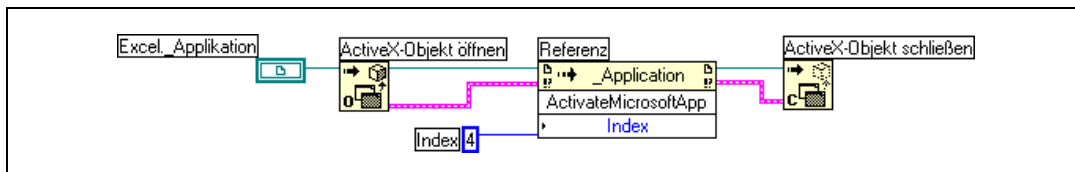


Abbildung 18-2. Festlegen eines Datenwerts mit einer numerischen Konstante

Parameter, die Datenwerte übernehmen, verfügen über einen kleinen Pfeil links neben dem Parameternamen. Klicken Sie zur Anzeige des entsprechenden numerischen Datenwerts mit der rechten Maustaste auf die Ring-Konstante, und wählen Sie aus dem Kontextmenü **Sichtbare Objekte»Zahlenwertanzeige**.

Die VIs in den Abbildungen 18-1 und 18-2 greifen jeweils auf die Applikation Microsoft Excel zu und führen eine Methode aus. Der Parameter **Index** der Methode **ActivateMicrosoftApp** verfügt über mehrere Optionen: **MicrosoftWord**, **MicrosoftPowerPoint**, **MicrosoftMail**, **MicrosoftAccess**, **MicrosoftFoxPro**, **MicrosoftProject** und **MicrosoftSchedulePlus**.

Um den numerischen Wert des Parameters **Index** zu ermitteln, welcher der Option **MicrosoftAccess** in Abbildung 18-1 entspricht, wählen Sie aus der Liste in der Ring-Konstante die Option **MicrosoftAccess** aus. Der numerische Wert der aktuell ausgewählten Option wird in einem Feld neben der Ring-Konstante angezeigt. Anstatt eine Ring-Konstante zu verwenden, können Sie den numerischen Wert einer Option in eine numerische Konstante eingeben (siehe Abbildung 18-2).

Aufrufen von Code aus textbasierten Programmiersprachen

Sie können die meisten gemeinsam genutzten Standardbibliotheken in LabVIEW mit der Funktion „Aufruf externer Bibliotheken“ aufrufen, die sich in der Funktionenpalette unter Fortgeschritten befindet. Sie können in LabVIEW mit Hilfe von Code-Interface-Knoten (CIN) auch C-Code aufrufen.

Weitere Informationen zum Aufrufen von Code textbasierter Programmiersprachen finden Sie im Handbuch *Using External Code in LabVIEW*.

Weitere Informationen ...

Weitere Informationen zum Aufrufen von Code aus textbasierten Programmiersprachen finden Sie in der *LabVIEW-Hilfe*.

Funktion „Aufruf externer Bibliotheken“

Mit der Funktion „Aufruf externer Bibliotheken“ können Sie die meisten gemeinsam genutzten Standardbibliotheken aufrufen. Mit dieser Funktion können Sie in LabVIEW eine Schnittstelle erstellen, um vorhandene Bibliotheken oder neue Bibliotheken, die speziell für die Verwendung mit LabVIEW geschrieben wurden, aufzurufen. National Instruments empfiehlt die Verwendung der Funktion „Aufruf externer Bibliotheken“, um eine Schnittstelle zu externem Code zu erstellen.

Code-Interface-Knoten

Verwenden Sie CIN als alternative Methode zum Aufrufen von Quellcode, der in C geschrieben ist. Die Funktion „Aufruf externer Bibliotheken“ ist in der Regel einfacher zu verwenden als CIN.

Formeln und Gleichungen

Wenn Sie in LabVIEW eine komplexe Formel verwenden möchten, brauchen Sie verschiedene arithmetische Funktionen im Blockdiagramm nicht zu verbinden. Sie können Gleichungen in einer vertrauten, mathematischen Umgebung entwickeln und diese Gleichungen anschließend in eine Applikation integrieren.

Mit dem Formelknoten können Sie in der LabVIEW-Umgebung mathematische Operationen ausführen. Um die Funktionalität zu erweitern, können Sie zu den mathematischen Applikationen HiQ und MATLAB Verknüpfungen herstellen, um Gleichungen zu entwickeln. HiQ und MATLAB sind Softwareprogramme, die Ihnen beim Verwalten und Visualisieren von realen mathematischen, wissenschaftlichen und technischen Problemen helfen.

Weitere Informationen ...

Weitere Informationen zu Gleichungen und der zu verwendenden Syntax, den verfügbaren Funktionen und Operatoren sowie Beschreibungen möglicher Fehler finden Sie in der *LabVIEW-Hilfe*.

Methoden zur Nutzung von Gleichungen in LabVIEW

Sie können den Formelknoten, den HiQ-Skript-Knoten und den MATLAB-Skript-Knoten verwenden, um im Blockdiagramm mathematische Operationen durchzuführen.



Hinweis Auf dem Computer muss HiQ oder MATLAB installiert sein, um Skript-Knoten zu verwenden, da LabVIEW die ActiveX-Technologie verwendet, um das Skript zur Ausführung an HiQ beziehungsweise MATLAB zu übergeben. LabVIEW verwendet für die Implementierung der Skript-Knoten die ActiveX-Technologie, die nur unter Windows zur Verfügung steht. Skript-Knoten stehen daher nur unter Windows zur Verfügung.

Da National Instruments mit LabVIEW auch HiQ ausliefert, können Sie die Software ohne Zusatzkosten installieren, um bei der Verarbeitung von Gleichungen eine höhere Funktionalität zu erreichen.

Die Skript-Knoten lassen sich mit dem Formelknoten vergleichen. Sie ermöglichen allerdings den Import eines vorhandenen HiQ- oder MATLAB-Skripts in ASCII-Form und die Ausführung des Skripts in LabVIEW. Wie beim Formelknoten können Daten zum und vom Knoten übergeben werden.

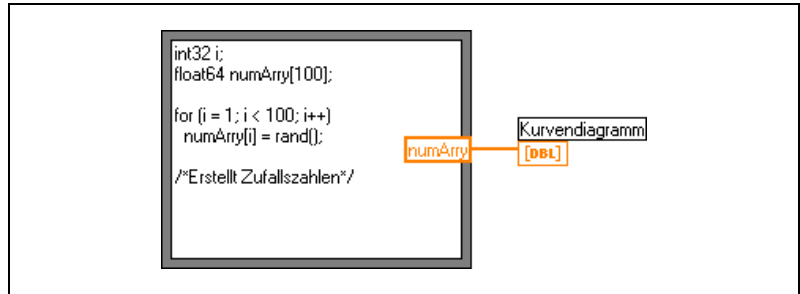
Formelknoten

Beim Formelknoten handelt es sich um einen praktischen textbasierten Knoten, mit dem Sie im Blockdiagramm mathematische Operationen durchführen können. Sie benötigen keinen Zugriff auf externen Code oder Applikationen, und Sie brauchen keine rudimentären arithmetischen Funktionen zu verbinden, um Gleichungen zu erstellen. Zusätzlich zu textbasierten Gleichungsausdrücken kann der Formelknoten textbasierte Versionen von If-Anweisungen, While-Schleifen, For-Schleifen und Do-Schleifen akzeptieren, die C-Programmierern vertraut sind. Diese Programmiererelemente sind denen der C-Programmierung ähnlich, jedoch nicht identisch.

Formelknoten sind nützlich für Gleichungen, die viele Variablen besitzen oder sonst kompliziert sind, sowie für die Nutzung vorhandenen textbasierten Codes. Sie können den vorhandenen textbasierten Code kopieren und in einem Formelknoten einfügen, anstatt den Code erneut grafisch zu erstellen.

Verwenden des Formelknotens

Der Formelknoten, der sich in der Funktionspalette unter **Strukturen** und **Mathematisch»Formel** befindet, ist ein Feld, dessen Größe wie bei der For-Schleife, While-Schleife, Case-Struktur und Sequenzstruktur verändert werden kann. Der Formelknoten enthält jedoch anstelle eines Unterdiagramms eine oder mehrere C-ähnliche Anweisungen, die wie in dem folgenden Beispiel gezeigt, durch Semikola getrennt sind. Wie bei C können Sie Kommentare zwischen einem Schrägstrich/Sternchen-Paar (`/*Kommentar*/`) hinzufügen.



Ein Beispiel für die Verwendung eines Formelknotens finden Sie in „Equations.vi“ in `examples\general\structs.llb`.

Variablen im Formelknoten

Bei der Arbeit mit Variablen müssen Sie die folgenden Punkte beachten:

- Die Anzahl der Variablen oder Gleichungen in einem Formelknoten ist unbegrenzt.
- Die Namen von Eingängen beziehungsweise Ausgängen müssen eindeutig sein, ein Ausgang kann jedoch denselben Namen haben wie ein Eingang.
- Sie deklarieren eine Eingangsvariable, indem Sie auf den Rand des Formelknotens klicken und aus dem Kontextmenü **Eingang hinzufügen** auswählen. Innerhalb des Formelknotens können keine Eingangsvariablen deklariert werden.
- Sie deklarieren eine Ausgangsvariable, indem Sie auf den Rand des Formelknotens klicken und aus dem Kontextmenü **Ausgang hinzufügen** auswählen. Sie müssen einen Namen für die Eingangsvariable beziehungsweise den Namen einer Variablen, die im Formelknoten deklariert wird, angeben.
- Sie können ändern, ob es sich bei einer Variablen um einen Eingang oder einen Ausgang handelt, indem Sie mit der rechten Maustaste auf die Variable klicken und aus dem Kontextmenü **In Eingang umwandeln** beziehungsweise **In Ausgang umwandeln** auswählen.
- Sie können im Formelknoten eine Variable deklarieren und verwenden, ohne sie einem Eingang oder Ausgang zuzuordnen.
- Sie müssen alle Eingangsanschlüsse verbinden.
- Variablen können skalare Fließkommazahlen sein, deren Genauigkeit von der Konfiguration des jeweiligen Computers abhängt. Sie können auch ganze Zahlen und Arrays aus Zahlen als Variablen verwenden.
- Variablen dürfen keine Einheiten besitzen.

Ausdrucksnoten

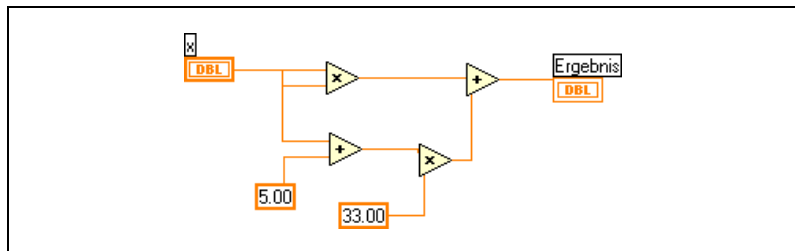
Mit Hilfe von Ausdrucksnoten können Sie Ausdrücke oder Gleichungen berechnen, die eine einzelne Variable enthalten. Ausdrucksnoten sollten verwendet werden, wenn eine Gleichung nur eine Variable hat, jedoch anderweitig kompliziert ist.

Ausdrucksnoten verwenden den Wert, den Sie an den Eingangsanschluss als Wert der Variablen übergeben. Der Ausgangsanschluss gibt den Wert der Berechnung zurück.

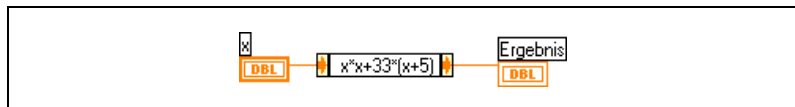
Betrachten Sie zum Beispiel die folgende einfache Gleichung:

$$x \times x + 33 \times (x + 5)$$

Das Blockdiagramm in der folgenden Abbildung verwendet zur Darstellung dieser Gleichung numerische Funktionen.



Mit Hilfe eines Ausdrucksnotens können Sie, wie in der folgenden Abbildung gezeigt, ein wesentlich einfacheres Blockdiagramm erstellen.



Polymorphismus in Ausdrucksnoten

Der Eingangsanschluss eines Ausdrucksnotens besitzt denselben Datentyp wie das angeschlossene Bedienelement oder die angeschlossene Konstante. Der Ausgangsanschluss besitzt denselben Datentyp wie der Eingangsanschluss. Der Datentyp des Eingangs darf eine beliebige nichtkomplexe skalare Zahl, ein Array nichtkomplexer skalarer Zahlen oder ein Cluster nichtkomplexer skalarer Zahlen sein. Bei Arrays und Clustern wendet der Ausdrucksnoten die Gleichung für jedes Element eines Eingangs-Arrays oder -Clusters an.

Verwenden von HiQ mit LabVIEW

HiQ ist eine hochleistungsfähige, interaktive Umgebung zur Problemlösung, in der Sie reale wissenschaftliche und technische Probleme analysieren, visualisieren und dokumentieren können. HiQ verwendet eine virtuelle Notebook-Schnittstelle und Programmiersprache mit dem Namen HiQ-Script. Das Notebook verfügt über Seiten, Abschnitte und Registerkarten, auf denen Sie Objekte wie Text, Zahlen und Graphen einfügen und anordnen können.

Mit den VIs in der Funktionspalette unter **Kommunikation»HiQ** können Sie HiQ von LabVIEW aus steuern und Daten in verschiedenen Formaten zwischen den beiden Applikationen übertragen. Um diese VIs zu verwenden, muss HiQ installiert sein. Mit diesen HiQ-VIs können Sie HiQ starten, auf HiQ-Notebooks zugreifen und manipulieren, um Daten zu analysieren und zu visualisieren, und ein HiQ-Script in einem Notebook ausführen.

Weitere Informationen zur HiQ-Funktionalität, einschließlich der mathematischen Merkmale, Funktionsreferenz und Syntaxinformationen, finden Sie in der *HiQ-Hilfe*.

Beispiele zum Einrichten und Verwenden der HiQ-VIs finden Sie in `examples\comm\hiq`.

HiQ- und MATLAB-Skript-Knoten

Mit den HiQ- und MATLAB-Skript-Knoten in der Funktionspalette unter **Mathematisch»Formel** können Sie HiQ- und MATLAB-Skripte im Blockdiagramm bearbeiten, damit LabVIEW mit den erweiterten mathematischen Funktionen arbeiten kann.

HiQ oder MATLAB muss installiert sein, um Skript-Knoten zu verwenden. Verfügen Sie bereits über ein in HiQ oder MATLAB geschriebenes Skript, können Sie es in den Skript-Knoten importieren.











Sie weisen im Skript als Eingänge oder Ausgänge für die Variablen Skript-Knotenanschlüsse zu, um Werte zwischen HiQ beziehungsweise MATLAB und LabVIEW zu übergeben. Sie können die Funktion eines Anschlusses anhand der Schreibweise der Gleichung festlegen. Wenn das Skript beispielsweise die Zuordnungsanweisung $X = i + 3$ enthält, können Sie i einem Eingangsanschluss zuweisen, der steuert, wie der

Skript-Knoten X berechnet, und Sie können X einem Ausgangsanschluss zuweisen, um das Endergebnis der Skript-Berechnung abzurufen.

Haben Sie noch kein Skript geschrieben, können Sie im Blockdiagramm einen Skript-Knoten einfügen und unter Verwendung der HiQ-beziehungswise MATLAB-Syntax ein Skript erstellen. LabVIEW kommuniziert mit dem Skript-Server-Modul. Dabei handelt es sich um das Programm, welches das Skript ausführt. LabVIEW kommuniziert und steuert das Skript-Server-Modul über ein industrierkanntes Protokoll. Die Skript-Server-Module werden mit HiQ oder MATLAB installiert.

Aufgrund der Eigenheiten der Skript-Sprachen von HiQ und MATLAB kann der Skript-Knoten nicht den Datentyp der erstellten Anschlüsse ermitteln. Sie müssen jedem Skript-Knotenanschluss einen LabVIEW-Datentyp zuordnen. Der Skript-Knoten in LabVIEW erkennt die von HiQ oder MATLAB unterstützten Datentypen. Tabelle 20-1 zeigt die LabVIEW-Datentypen sowie die entsprechenden Datentypen in HiQ und MATLAB.

Tabelle 20-1. LabVIEW-, HiQ- und MATLAB-Datentypen

LabVIEW Datentyp	HiQ Datentyp	MATLAB Datentyp
	Ganzzahl	—
	Reell	Reell
	Text	—
	Ganzzahliger Vektor	—
	Reeller Vektor	Reeller Vektor
	Ganzzahlige Matrix	—
	Reelle Matrix	Reelle Matrix
	Komplex	Komplex
	Komplexer Vektor	Komplexer Vektor
	Komplexe Matrix	Komplexe Matrix

Programmervorschläge für HiQ- und MATLAB-Skripts

Die folgenden Programmier Techniken erleichtern die Fehlersuche in einem Skript:

- Schreiben Sie das Skript und führen es in HiQ oder MATLAB für Test- und Fehlersuchzwecke aus, bevor Sie es in LabVIEW importieren.
- Klicken Sie mit der rechten Maustaste auf den HiQ-Skript-Knoten und wählen Sie aus dem Kontextmenü **Im Server bearbeiten**, um das Skript zu bearbeiten, die Fehlersuche auszuführen, zu kompilieren und in einem eigenständigen HiQ-Skript-Fenster auszuführen.
- Überprüfen Sie die Datentypen. Wenn Sie einen neuen Eingang oder Ausgang erstellen, müssen Sie sicherstellen, dass der Datentyp des Anschlusses richtig ist. Sowohl in HiQ als auch in MATLAB kann während der Berechnung der Typ einer Variablen geändert werden. Mit Hilfe der Funktionen „Fehlereingang“ und „Fehlerausgang“ können Sie eine Typenänderung feststellen.
- Erstellen Sie Bedien- und Anzeigeelemente für die Ein- und Ausgänge, damit Sie die Werte überwachen können, die zwischen LabVIEW und HiQ oder MATLAB übergeben werden. Auf diese Weise können Sie gegebenenfalls feststellen, wo ein Skript-Knoten einen Wert falsch berechnet.
- Nutzen Sie die Fehlerprüfparameter für Fehlersuchinformationen. Erstellen Sie ein Anzeigeelement für den Anschluss **Fehlerausgang** eines Skript-Knotens, damit die Fehlerinformationen zur Laufzeit angezeigt werden. Bei Formelknoten werden Fehler beim Kompilieren angezeigt.

Erforderliche HiQ-Supportdateien für eine LabVIEW-Applikation

Nachdem Sie eine LabVIEW-Applikation erstellt haben, die Aufrufe von HiQ-VIs enthält, müssen Sie den Zielcomputer berücksichtigen, auf dem die Applikation ausgeführt wird.

Wenn der HiQ-Skript-Knoten im LabVIEW-Code aufgerufen wird, wird HiQ gestartet. Wenn Sie eine LabVIEW-Applikation vertreiben, die ein HiQ-Skript enthält, muss auf dem Zielrechner HiQ installiert sein.

In der folgenden Tabelle ist aufgeführt, welche Dateien abhängig von Einsatz und Verteilung von LabVIEW und HiQ benötigt werden.

Zu verteilende Dateien	Aktuelle Software auf dem Zielsystem	Erforderliche Installation
LabVIEW-Applikations-VIs mit HiQ-Skript-Knoten	LabVIEW (beliebige Version) installiert; HiQ nicht installiert	Lizenziertes Exemplar von HiQ, das im Lieferumfang von LabVIEW enthalten ist.
Programmdatei	Weder LabVIEW noch HiQ installiert	HiQ Reader ermöglicht es anderen Benutzern, HiQ Professional Notebooks anzuzeigen und Skripte auszuführen. Sie können den HiQ Reader von der LabVIEW Full Development System- oder Professional Development System-CD kostenlos verteilen. Sie können den HiQ Reader von der National Instruments-Internetseite unter ni.com herunterladen.

Die Organisation von LabVIEW

Dieser Anhang beschreibt die Struktur des LabVIEW-Dateisystems und die vorgeschlagenen Verzeichnisse zum Speichern von Dateien.

Organisation der LabVIEW-Verzeichnisstruktur

Dieser Anhang beschreibt die Struktur des LabVIEW-Dateisystems für Windows, Macintosh und UNIX. LabVIEW installiert Treibersoftware für GPIB-, DAQ- und VXI-Hardware. Informationen zum Konfigurieren der Hardware finden Sie in Kapitel 3, *Installing and Configuring Your Measurement Hardware*, des *LabVIEW Measurement Manual*.

Wie in den mit der Software gelieferten *LabVIEW-Versionshinweisen* beschrieben, enthält das LabVIEW-Verzeichnis nach der Installation die folgenden Unterverzeichnisse.

Bibliotheken

- `user.lib`: Enthält die von Ihnen erstellten Bedienelemente und VIs. LabVIEW zeigt die Bedienelemente in der Elementpalette unter **Benutzerdef. Elemente** und VIs in der Funktionpalette unter **Eigene Bibliotheken** an.
- `vi.lib`: Enthält Bibliotheken integrierter VIs, wie zum Beispiel GPIB-, Analyse- und DAQ-VIs. LabVIEW zeigt diese VIs in der **Funktionpalette** in den jeweiligen Unterpalletten an. Speichern Sie keine Dateien im Verzeichnis `vi.lib`, da LabVIEW diese Dateien beim Installieren neuer Versionen überschreibt.
- `instr.lib`: Enthält Instrumententreiber, mit denen PXI-, VXI-, GPIB-, serielle und computerbasierte Instrumente gesteuert werden. Speichern Sie Instrumententreiber von National Instruments in diesem Verzeichnis, wenn Sie diese unabhängig von LabVIEW installieren. Sie werden von LabVIEW zu der Unterpalette **Instrumenten-I/O» Instrumententreiber** hinzugefügt.

Struktur und Support

- `menus`: Enthält Dateien, mit denen LabVIEW die Struktur der Paletten **Elemente** und **Funktionen** konfiguriert.
- `resource`: Enthält zusätzliche Support-Dateien für die LabVIEW-Applikation. Speichern Sie keine Dateien in diesem Verzeichnis, da LabVIEW diese Dateien beim Installieren neuer Versionen überschreibt.
- `project`: Enthält Dateien, die Objekte im LabVIEW-Menü **Werkzeuge** darstellen.
- `templates`: Enthält Vorlagen für allgemeingebrauchliche VIs.
- `www`: Verzeichnis für HTML-Dateien, auf die Sie über den Web-Server zugreifen können.

Lernen und Anleitung

- `activity`: Enthält VIs, mit denen Sie die Aktivitäten im *LabVIEW-Tutorium* durchführen. Das Verzeichnis `activity\solution` enthält die durchgearbeiteten VIs der einzelnen Aktivitäten.
- `examples`: Enthält Beispiel-VIs. Wählen Sie **Hilfe»Beispiele...**, um die Beispiele zu durchsuchen.

Dokumentation

- `manuals`: Enthält die Dokumentation im PDF-Format. Dieser Ordner enthält keine Hilfedateien. Sie können die PDF-Dateien anzeigen, indem Sie **Hilfe»Zeige gedruckte Handbücher** auswählen.
- `help`: Enthält die Hilfedateien. Sie können die *LabVIEW-Hilfe* anzeigen, indem Sie **Hilfe»Inhalt und Index** auswählen.

Datei für Sonstiges

- `serpdrv`: Support-Datei für den Zugriff auf den seriellen Anschluss unter Windows und UNIX. Verteilen Sie diese Datei mit allen Applikationen, die auf diesen Plattformen den seriellen Anschluss nutzen.

Macintosh

Zusätzlich zu den obigen Dateien verfügen Macintosh-Anwender über einen Ordner für gemeinsam genutzte Bibliotheken, der Support-Dateien für die LabVIEW-Applikation enthält.

Vorgeschlagenes Verzeichnis zum Speichern von Dateien

LabVIEW installiert die Verzeichnisse `vi.lib` und `resource` ausschließlich für Zwecke des LabVIEW-Systems. Speichern Sie in diesen Verzeichnissen keine Dateien.

Sie können Dateien in den folgenden Verzeichnissen speichern:

- `user.lib`: Alle häufig verwendeten VIs, die in der **Funktionenpalette** unter **Eigene Bibliotheken** angezeigt werden sollen. Verwenden Sie dieses Verzeichnis zur Speicherung von `vi.lib`-Erweiterungen.



Hinweis Speichern Sie im Verzeichnis `user.lib` nur dann Sub-VIs, wenn sie ohne Änderungen projektübergreifend portiert werden können. Die Pfade zu VIs in `user.lib` sind absolut. Pfade zu Sub-VIs, die in anderen Verzeichnissen gespeichert wurden, werden relativ zum übergeordneten VI gespeichert. Wenn Sie ein VI aus `user.lib` kopieren, um es für einen Spezialfall zu modifizieren, wird daher der Pfad zu den dazugehörigen Sub-VIs in `user.lib` nicht geändert.

- `instr.lib`: Alle Instrumententreiber-VIs, die in der Funktionenpalette unter **Instrumenten-I/O»Instrumententreiber** angezeigt werden sollen.
- `project`: VIs, mit denen die LabVIEW-Eigenschaften erweitert werden. Die in diesem Verzeichnis gespeicherten VIs werden im Menü **Werkzeuge** angezeigt.
- `www`: Verzeichnis für HTML-Dateien, auf die Sie über den Web-Server zugreifen können.
- `help`: Alle VIs, PDF-Dateien und `.hlp`-Dateien, die im Menü **Hilfe** zur Verfügung gestellt werden sollen.

Sie können auf der Festplatte an beliebiger Stelle Verzeichnisse erstellen, um von Ihnen erstellte LabVIEW-Dateien zu speichern.

Polymorphe Funktionen

Funktionen sind unterschiedlich polymorph: keine, einige oder alle Eingänge können polymorph sein. Einige Funktionseingänge akzeptieren Zahlen oder boolesche Werte. Einige akzeptieren Zahlen oder Strings. Einige lassen nicht nur skalare Zahlen, sondern auch Zahlen-Arrays, Zahlen-Cluster, Arrays von Zahlen-Clustern und so weiter zu. Einige akzeptieren nur eindimensionale Arrays, wobei die Array-Elemente von jedem Datentyp sein können. Einige Funktionen lassen alle Datentypen zu, komplexe Zahlen eingeschlossen.

Weitere Informationen ...

Weitere Informationen zu polymorphen Funktionen finden Sie in der *LabVIEW-Hilfe*.

Numerische Konvertierung

Jede numerische Darstellung kann in jede andere numerische Darstellung konvertiert werden. Wenn zwei oder mehrere numerische Eingänge verschiedener Darstellungen mit einer Funktion verbunden werden, gibt die Funktion normalerweise die Ausgabe in dem größeren oder breiteren Format zurück. Die Funktionen wandeln die kleineren Darstellungen vor der Ausführung in die größte Darstellung um.

Einige Funktionen, wie zum Beispiel Dividieren, Sinus und Kosinus, erzeugen immer Ausgaben mit Fließkommazahlen. Wenn Sie Ganzzahlen an den Eingängen eingeben, wandeln diese Funktionen die Ganzzahlen vor der Ausführung der Berechnung in Double-Fließkommazahlen um.

Für skalare Mengen mit Fließkommazahlen werden normalerweise am besten Double-Fließkommazahlen verwendet. Single-Fließkommazahlen sparen wenig oder keine Ausführungszeit und laufen sehr viel schneller über. Die Analysebibliotheken verwenden beispielsweise Double-Fließkommazahlen. Extended-Fließkommazahlen sollten nur wenn notwendig verwendet werden. Die Leistung und Genauigkeit von arithmetischen Operationen mit Extended-Fließkommazahlen hängt von der jeweiligen Plattform ab. Weitere Informationen zum Überlauf von Fließkommazahlen

finden Sie in dem Abschnitt *Undefinierte oder unerwartete Daten* des Kapitels 6, *Ausführen von und Fehlersuche in VIs*.

Für ganze Zahlen wird normalerweise am besten ein vorzeichenbehafteter 32-Bit-Integer verwendet.

Wenn ein Ausgang mit einem Ziel verbunden wird, das mit einer anderen numerischen Darstellung arbeitet, konvertiert LabVIEW die Daten entsprechend den folgenden Regeln:

- **Ganzzahl mit oder ohne Vorzeichen in Fließkommazahl:** Die Umwandlung ist genau, außer bei der Umwandlung von Long-Integern in Single-Fließkommazahlen. In diesem Fall reduziert LabVIEW die Genauigkeit von 32 Bit auf 24 Bit.
- **Fließkommazahl in Ganzzahl mit oder ohne Vorzeichen:** LabVIEW wandelt Werte außerhalb des Bereichs in den Maximal- oder Minimalwert der Ganzzahl. Die meisten der ganzzahligen Objekte, wie zum Beispiel der Iterationsanschluss einer For-Schleife, runden Fließkommazahlen auf beziehungsweise ab. LabVIEW rundet einen Bruchteil von 0,5 in die nächste gerade Ganzzahl. LabVIEW rundet beispielsweise 6,5 auf 6 und nicht auf 7.
- **Ganzzahl in Ganzzahl:** LabVIEW wandelt Werte außerhalb des Bereichs nicht in den Maximal- oder Minimalwert der Ganzzahl. Wenn die Quelle kleiner als das Ziel ist, erweitert LabVIEW das Vorzeichen einer vorzeichenbehafteten Quelle und fügt in die überzähligen Bits einer vorzeichenlosen Quelle Nullen ein. Wenn die Quelle größer als das Ziel ist, kopiert LabVIEW von dem Wert nur die niederwertigen Bits.

Polymorphismus von numerischen Funktionen

Die arithmetischen Funktionen akzeptieren numerische Eingangsdaten. Mit einigen Ausnahmen, auf die bei den Funktionsbeschreibungen hingewiesen wird, besitzt der Ausgang dieselbe numerische Darstellung wie der Eingang, beziehungsweise bei verschiedenen Darstellungen der Eingänge ist die Ausgabe breiter als die Eingänge.

Die arithmetischen Funktionen arbeiten mit Zahlen, Arrays mit Zahlen, Clustern mit Zahlen, Arrays mit Clustern von Zahlen, komplexen Zahlen, und so weiter. Die formale und rekursive Definition des zulässigen Eingabetyps lautet:

Numerischer Typ = numerischer Skalar ODER Array [*numerischer Typ*]
ODER Cluster [*numerische Typen*]

Die numerischen Skalare können Fließkommazahlen, Ganzzahlen oder komplexe Fließkommazahlen sein. LabVIEW lässt keine Verwendung von Arrays aus anderen Arrays zu.

Arrays können eine beliebige Anzahl Dimensionen jeder Größe besitzen. Cluster können eine beliebige Anzahl von Elementen besitzen. Der Ausgangstyp von Funktionen besitzt dieselbe numerische Darstellung wie der Eingangstyp. Bei Funktionen mit nur einem Eingang verarbeiten die Funktionen jedes Element des Arrays oder Clusters.

Bei Funktionen mit zwei Eingängen können Sie die folgenden Eingangskombinationen verwenden:

- **Ähnlich:** Beide Eingänge besitzen dieselbe Struktur, und der Ausgang hat dieselbe Struktur wie die Eingänge.
- **Ein Skalar:** Ein Eingang ist ein numerischer Skalar, der andere ein Array oder Cluster, und der Ausgang ist ein Array oder Cluster.
- **Array:** Ein Eingang ist ein numerisches Array, der andere der numerische Typ selbst, und der Ausgang ist ein Array.

Bei ähnlichen Eingängen führt LabVIEW die Funktion für die jeweiligen Elemente der Strukturen durch. LabVIEW kann beispielsweise zwei Arrays Element für Element addieren. Beide Arrays müssen dieselbe Dimension besitzen. Sie können Arrays mit einer unterschiedlichen Anzahl von Elementen addieren. Die Ausgabe einer solchen Addition besitzt die gleiche Anzahl von Elementen wie die kleinste Eingabe. Cluster müssen dieselbe Anzahl von Elementen besitzen, und die jeweiligen Elemente müssen vom selben Typ sein.

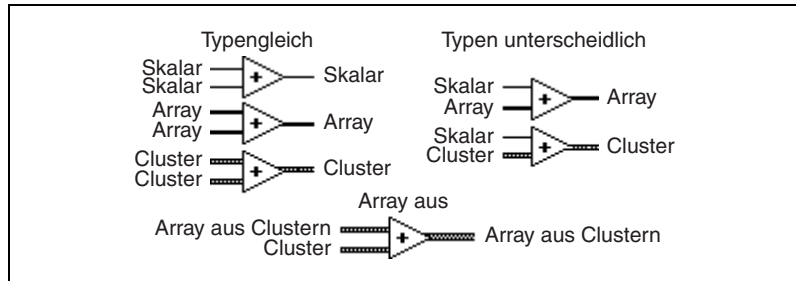
Mit der Multiplikationsfunktion kann keine Matrixmultiplikation durchgeführt werden. Wenn Sie die Multiplikationsfunktion für zwei Matrizen verwenden, multipliziert LabVIEW die erste Zahl in der ersten Zeile der ersten Matrix mit der ersten Zahl in der ersten Zeile der zweiten Matrix, und so weiter.

Bei Operationen mit einem Skalar und einem Array oder Cluster führt LabVIEW die Funktion mit dem Skalar und den entsprechenden Elementen der Struktur durch. LabVIEW kann beispielsweise eine Zahl von allen Elementen eines Arrays subtrahieren, unabhängig von der Dimension des Arrays.

Bei Operationen mit einem numerischen Typ und einem Array des betreffenden Typs führt LabVIEW die Funktion für jedes Array-Element durch. Bei einem Graphen handelt es sich beispielsweise um ein Array von Punkten, wobei ein Punkt ein Cluster zweier numerischer Typen, x und y ,

bildet. Um einen Graphen 5 Einheiten in x -Richtung und 8 Einheiten in y -Richtung zu verschieben, können Sie den Punkt (5, 8) zum Graphen addieren.

Das folgende Beispiel zeigt einige mögliche polymorphe Kombinationen der Additionsfunktion.



Polymorphismus von booleschen Funktionen

Die logischen Funktionen akzeptieren boolesche oder numerische Eingangsdaten. Bei einem numerischen Eingang führt LabVIEW die Operation bitweise durch. Wenn der Eingang eine Ganzzahl ist, besitzt der Ausgang dieselbe Darstellung. Ist der Eingang eine Fließkommazahl, rundet LabVIEW die Zahl in eine Long-Integer-Zahl, und der Ausgang ist vom Typ Long-Integer.

Die logischen Funktionen arbeiten mit Zahlen oder booleschen Werten, Clustern mit Zahlen oder booleschen Werten, Arrays mit Clustern von Zahlen oder booleschen Werten, und so weiter.

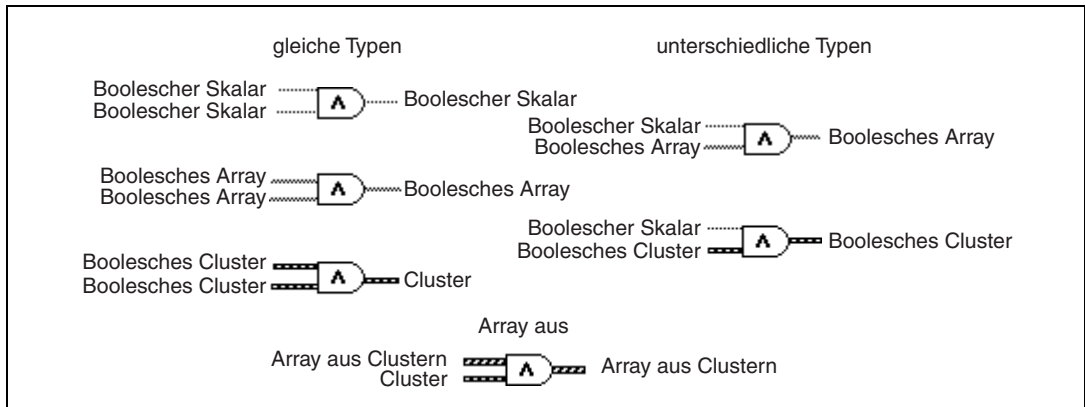
Die formale und rekursive Definition des zulässigen Eingabetyps lautet:

Logischer Typ = boolescher Skalar ODER numerischer Skalar ODER Array [*logischer Typ*] ODER Cluster [*logische Typen*]

mit der Ausnahme, dass komplexe Zahlen und Arrays von Arrays nicht zugelassen sind.

Logische Funktionen mit zwei Eingängen können dieselben Eingangskombinationen wie die arithmetischen Funktionen haben. Die logischen Funktionen unterliegen jedoch der weiteren Beschränkung, dass die Grundoperationen nur zwischen zwei booleschen Werten oder zwei Zahlen erfolgen können. Eine UND-Operation zwischen einem booleschen

Wert und einer Zahl ist beispielsweise nicht möglich. Das folgende Beispiel zeigt einige Kombinationen boolescher Werte für die UND-Funktion.



Polymorphismus von Array-Funktionen

Die meisten Array-Funktionen akzeptieren n -dimensionale Arrays beliebigen Typs. Die Verbindungsdiagramme in den Funktionsbeschreibungen zeigen als Standarddatentyp numerische Arrays.

Polymorphismus von String-Funktionen

Die Funktionen „String-Länge“, „In Großbuchstaben“, „In Kleinbuchstaben“, „String umkehren“ und „String rotieren“ akzeptieren Strings, Cluster und Arrays mit Strings sowie Arrays mit Clustern. „In Großbuchstaben“ und „In Kleinbuchstaben“ akzeptieren auch Zahlen, Cluster mit Zahlen sowie Arrays mit Zahlen, wobei die Zahlen als ASCII-Zeichen interpretiert werden. Die Eingänge für Breite und Genauigkeit müssen skalar sein.

Polymorphismus von String-Konvertierungsfunktionen

Die Funktionen „Pfad in String“ und „String in Pfad“ sind polymorph. Das heißt, sie arbeiten mit skalaren Werten, Arrays von Skalaren, Clustern von Skalaren, Arrays von Clustern mit Skalaren, und so weiter. Der Ausgang besitzt dieselbe Struktur wie der Eingang, jedoch mit dem neuen Typ.

Polymorphismus von zusätzlichen String_in_Zahl-Funktionen

Die Funktionen „Zahl in Dezimal-String“, „Zahl in Hexadezimal-String“, „Zahl in Oktal-String“, „Zahl in String in techn. Notation“, „Zahl in String in gebrochener Notation“ und „Zahl in String in Exponentialnotation“ akzeptieren Cluster und Arrays mit Zahlen und erzeugen Cluster und Arrays mit Strings. Die Funktionen „Dezimal-String in Zahl“, „Hexadezimal-String in Zahl“, „Oktal-String in Zahl“, „Bruch-/Exponential-String in Zahl“ akzeptieren Cluster und Arrays mit Strings und erzeugen Cluster und Arrays mit Zahlen. Die Eingänge für Breite und Genauigkeit müssen skalar sein.

Polymorphismus von Cluster-Funktionen

Die Funktionen „Elemente bündeln“ und „Aufschlüsseln“ zeigen erst dann die Datentypen für die einzelnen Eingangs- oder Ausgangsanschlüsse, wenn Objekte mit den betreffenden Anschlüssen verbunden werden. Wenn Sie diese Anschlüsse verbinden, sehen sie ähnlich wie die Datentypen der entsprechenden Frontpanel-Anschlüsse der Bedien- oder Anzeigeelemente aus.

Polymorphismus von Vergleichsfunktionen

Die Funktionen „Gleich?“, „Ungleich?“ und „Wählen“ arbeiten mit Eingaben jeden Typs, so lange die Eingaben denselben Typ besitzen.

Die Funktionen „Größer oder gleich?“, „Kleiner oder gleich?“, „Kleiner?“, „Größer?“, „Max & Min“ und „Wertebereich prüfen und erzwingen“ arbeiten mit Eingaben jeglichen Typs, außer komplexen, Pfad- oder Refnum-Eingaben, solange die Eingaben denselben Typ besitzen. Sie können Zahlen, Strings, boolesche Werte, Arrays aus Strings, Cluster aus Zahlen, Cluster aus Strings und so weiter vergleichen. Sie können jedoch keine Zahl mit einem String oder einen String mit einem booleschen Wert und so weiter vergleichen.

Die Funktionen, die Werte mit Null vergleichen, akzeptieren numerische Skalare, Cluster und Arrays aus Zahlen. Diese Funktionen geben boolesche Werte in derselben Datenstruktur wie der Eingang aus.

Die Funktion „Keine Zahl/Pfad/RefNum?“ akzeptiert dieselben Eingabetypen wie die Funktionen, die Werte mit Null vergleichen. Diese Funktion akzeptiert außerdem Pfade und Refnums. Die Funktion

„Keine Zahl/Pfad/RefNum?“ gibt boolesche Werte in derselben Datenstruktur wie der Eingang aus.

Die Funktionen „Dezimalziffer?“, „Hexadezimalziffer?“, „Oktalziffer?“, „Druckbar?“ und „Nicht darstellbares Zeichen?“ akzeptieren die Eingabe eines skalaren Strings oder einer Zahl, von Clustern aus Strings oder nichtkomplexen Zahlen, von Arrays aus Strings oder nichtkomplexen Zahlen und so weiter. Die Ausgabe besteht aus booleschen Werten in derselben Datenstruktur wie die Eingabe.

Die Funktion „Leerer String/Pfad?“ akzeptiert einen Pfad, einen skalaren String, Cluster aus Strings, Arrays aus Strings und so weiter. Die Ausgabe besteht aus booleschen Werten in derselben Datenstruktur wie die Eingabe.

Die Funktionen „Gleich?“, „Ungleich?“, „Keine Zahl/Pfad/RefNum?“, „Leerer String/Pfad?“ und „Wählen“ können mit Pfaden und Refnums eingesetzt werden; andere Vergleichsfunktionen jedoch akzeptieren keine Pfade oder Refnums als Eingabe.

Vergleichsfunktionen, die Arrays und Cluster verwenden, erzeugen normalerweise boolesche Arrays und Cluster mit derselben Struktur. Sie können mit der rechten Maustaste auf die Funktion klicken und **Vergleichen mehrerer Elemente** auswählen, wodurch die Funktion einen einzelnen booleschen Wert ausgibt. Die Funktion vergleicht mehrere Elemente, indem der erste Satz von Elementen zur Erzeugung der Ausgabe verglichen wird, außer wenn die ersten Elemente gleich sind; in diesem Fall vergleicht die Funktion den zweiten Satz von Elementen und so weiter.

Polymorphismus von logarithmischen Funktionen

Die logarithmischen Funktionen akzeptieren numerische Eingangsdaten. Wenn die Eingabe eine Ganzzahl ist, ist die Ausgabe eine Fließkommazahl mit doppelter Genauigkeit. Andernfalls besitzt der Ausgang dieselbe numerische Darstellung wie der Eingang.

Diese Funktionen arbeiten mit Zahlen, Arrays mit Zahlen, Clustern mit Zahlen, Arrays mit Clustern von Zahlen, komplexen Zahlen, und so weiter. Die formale und rekursive Definition des zulässigen Eingabetyps lautet:

Numerischer Typ = numerischer Skalar ODER Array [numerischer Typ]
ODER Cluster [numerische Typen]

mit der Ausnahme, dass Arrays aus Arrays nicht zugelassen sind.

Arrays können eine beliebige Größe und eine beliebige Anzahl Dimensionen besitzen. Cluster können eine beliebige Anzahl von Elementen besitzen. Der Ausgangstyp besitzt dieselbe numerische Darstellung wie der Eingangstyp, und die Funktionen verarbeiten jedes Element des Clusters oder Arrays. Weitere Informationen zu polymorphen Funktionen mit zwei Eingängen finden Sie in dem Abschnitt [Polymorphismus von numerischen Funktionen](#) dieses Kapitels. Für logarithmische Funktionen mit zwei Eingängen sind die folgenden Kombinationen von Eingangstypen zulässig:

- **Ähnlich:** Beide Eingänge besitzen dieselbe Struktur und der Ausgang hat dieselbe Struktur wie die Eingänge.
- **Ein Skalar:** Der eine Eingang ist ein numerischer Skalar, der andere ein numerisches Array oder ein numerischer Cluster, und der Ausgang ist ein Array oder Cluster.



Vergleichsfunktionen

Mit den Vergleichsfunktionen, die sich in der Palette **Funktionen» Vergleich** befinden, können Sie boolesche Werte, Strings, Zahlen, Arrays und Cluster vergleichen. Die Mehrzahl der Vergleichsfunktionen prüft einen Eingang oder vergleicht zwei Eingänge und gibt einen booleschen Wert aus.

Weitere Informationen ...

Weitere Informationen zu Vergleichsfunktionen finden Sie in der *LabVIEW-Hilfe*.

Vergleichen von booleschen Werten

Für die Vergleichsfunktionen ist der Boolesche Wert TRUE größer als der boolesche Wert FALSE.

Vergleichen von Strings

LabVIEW vergleicht Strings basierend auf dem numerischen Äquivalent der ASCII-Zeichen. Demnach ist a (mit einem Dezimalwert von 97) größer als A (65), was wiederum größer als 0 (48) ist, was wiederum größer als das Leerzeichen (32) ist. LabVIEW vergleicht die einzelnen Zeichen am Anfang des String beginnend Zeichen für Zeichen, bis eine Ungleichheit auftritt. An diesem Punkt endet dann der Vergleich. LabVIEW vergleicht beispielsweise die Strings abcd und abef bis c gefunden wird, dessen Wert kleiner ist als e. Das Vorhandensein eines Zeichens wird als größer als das Nichtvorhandensein eines Zeichens interpretiert. Der String abcd ist also größer als abc, weil der erste String länger ist.

Die Funktionen, welche die Kategorie eines Stringzeichens prüfen (beispielsweise die Funktionen „Dezimalstellen?“ und „Druckbar?“), werten nur das erste Zeichen eines Strings aus.

Vergleichen von Zahlen

Die Vergleichsfunktionen konvertieren numerische Werte vor dem Vergleichen in dieselbe Darstellung. Vergleiche eines Eingangs oder von zwei Eingängen mit dem Wert `Keine Zahl` geben einen Wert zurück, der Ungleichheit anzeigt. Weitere Informationen zu dem Wert `Keine Zahl` finden Sie in dem Abschnitt *Undefinierte oder unerwartete Daten* des Kapitels 6, *Ausführen von und Fehlersuche in VIs*.

Vergleichen von Arrays und Clustern

Einige Vergleichsfunktionen verfügen über zwei Modi, um Arrays oder Cluster mit Daten zu vergleichen. Wenn im Modus „Vergleichen mehrerer Elemente“ zwei Arrays oder Cluster verglichen werden, gibt die Funktion einen einzelnen skalaren Wert zurück. Im Modus „Vergleichen der Elemente“ vergleicht die Funktion die Elemente einzeln und gibt dann ein Array oder einen Cluster aus booleschen Werten zurück.

Im Modus „Vergleichen mehrerer Elemente“ folgen die String- und Array-Vergleichsoperationen demselben Verfahren – der String wird als Array von ASCII-Zeichen betrachtet.

Einige Vergleichsfunktionen arbeiten nur im Modus „Vergleichen mehrerer Elemente“, sodass keine entsprechenden Optionen im Kontextmenü angezeigt werden.

Arrays

Beim Vergleichen von mehrdimensionalen Arrays müssen alle in die Funktion eingegebenen Arrays dieselbe Dimension besitzen. Die Vergleichsfunktionen, die nicht über die Modi „Vergleichen mehrerer Elemente“ und „Vergleichen der Elemente“ verfügen, vergleichen Arrays in derselben Weise wie Strings – mit dem ersten Element beginnend ein Element nach dem anderen, bis eine Ungleichheit auftritt.

Modus „Vergleichen der Elemente“

Im Modus „Vergleichen der Elemente“ geben die Vergleichsfunktionen ein Array mit booleschen Werten derselben Dimension wie die Eingabe-Arrays aus. Jede Dimension des Ausgabe-Arrays stellt die Größe des kleineren der beiden Eingabe-Arrays in der jeweiligen Dimension dar. Bei jeder Dimension (Zeile, Spalte oder Seite) vergleichen die Funktionen die

entsprechenden Elementwerte in allen Eingabe-Arrays, um den entsprechenden booleschen Wert im Ausgabe-Array zu erzeugen.

Modus „Vergleichen mehrerer Elemente“

Im Modus „Vergleichen mehrerer Elemente“ geben die Vergleichsfunktionen ein einziges boolesches Ergebnis zurück. LabVIEW betrachtet die jeweiligen Werte, die in den Eingabe-Arrays später auftreten, als untergeordnete Werte der Werte, die in den Eingabe-Arrays früher vorkommen. LabVIEW führt die folgenden Schritte durch, um das Ergebnis des Vergleichs zu ermitteln:

- LabVIEW vergleicht zugehörige Elemente in jedem Eingabe-Array, beginnend am Anfang des Arrays.
- Stimmen die entsprechenden Elemente *nicht* überein, beendet LabVIEW den Vergleich – das Ergebnis dieses Vergleichs wird als Ausgabe der Vergleichsfunktion verwendet.
- Stimmen die entsprechenden Elemente überein, verarbeitet LabVIEW das nächste Wertepaar, bis eine Ungleichheit festgestellt oder das Ende eines Eingabe-Arrays erreicht wird.
- Stimmen alle Werte in den Eingabe-Arrays überein, besitzt ein Array am Ende jedoch weitere Elemente, wird das längere Array als größer angesehen als das kürzere Array. LabVIEW betrachtet das Array [1, 2, 3, 2] beispielsweise als größer als das Array [1, 2, 3].

Cluster

Cluster, die verglichen werden sollen, müssen dieselbe Anzahl Elemente enthalten, die Typen der einzelnen Elemente in den Clustern müssen kompatibel sein und die Elemente müssen in derselben Cluster-Reihenfolge vorkommen. Ein Cluster mit Daten vom Typ DBL und einem String kann beispielsweise mit einem Cluster mit Daten vom Typ I32 und einem String verglichen werden.

Modus „Vergleichen der Elemente“

Im Modus „Vergleichen der Elemente“ geben die Vergleichsfunktionen einen Cluster mit booleschen Werten zurück, wobei jedes Element den dazugehörigen Elementen im Eingabe-Cluster entspricht.

Modus „Vergleichen mehrerer Elemente“

Im Modus „Vergleichen mehrerer Elemente“ geben die Vergleichsfunktionen ein einziges boolesches Ergebnis zurück. LabVIEW vergleicht die entsprechenden Elemente, bis eine Ungleichheit festgestellt wird, die das Ergebnis bestimmt. Die Funktion betrachtet die beiden Cluster nur dann als gleich, wenn alle Elemente übereinstimmen.

Verwenden Sie den Modus „Vergleichen mehrerer Elemente“, wenn Sie Datensätze mit sortierten Daten vergleichen, wobei später in dem Cluster auftretende Elemente als untergeordnete Elemente der früher im Cluster vorkommenden Elemente betrachtet werden. In einem Cluster mit zwei Strings, Nachname gefolgt von Vorname, werden die Felder für die Vornamen beispielsweise nur dann verglichen, wenn die Felder für die Nachnamen übereinstimmen.

Maskieren von digitalen Daten

Erstellen Sie im Blockdiagramm eine Maske, indem Sie den digitalen Signalverlauf mit einem 2D-Array aus ganzen Zahlen bündeln (siehe Abbildung D-1).

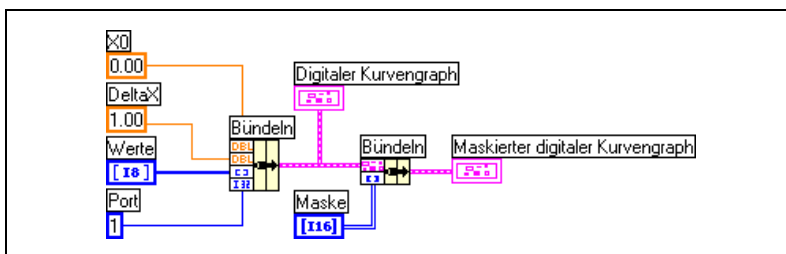


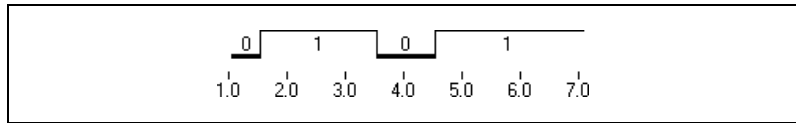
Abbildung D-1. Maskieren von digitalen Daten

Kombinieren Sie die Bits, indem Sie in jeder Zeile des 2D-Arrays **Maske** Werte festlegen.

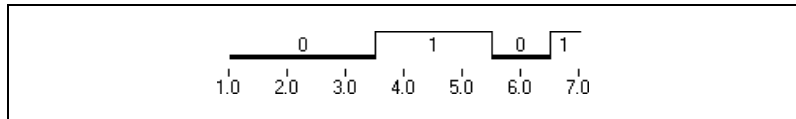
Betrachten Sie beispielsweise ein eindimensionales Array, das aus sieben 8-Bit-Zahlen besteht: 1, 2, 7, 32, 55, 82 und 127. Die folgende Tabelle zeigt die binäre Darstellung dieser Zahlen.

	1	2	7	32	55	82	127
Bit 0	1	0	1	0	1	0	1
Bit 1	0	1	1	0	1	1	1
Bit 2	0	0	1	0	1	0	1
Bit 3	0	0	0	0	0	0	1
Bit 4	0	0	0	0	1	1	1
Bit 5	0	0	0	1	1	0	1
Bit 6	0	0	0	0	0	1	1
Bit 7	0	0	0	0	0	0	0

Mit Hilfe einer Maske können Sie ein Bit in einem Graph darstellen und ein anderes Bit derselben Kurve überlagern. Bit 1 enthält beispielsweise, wie in der folgenden Darstellung gezeigt, die Werte 0, 1, 1, 0, 1, 1 und 1.



Bit 5 enthält, wie in der folgenden Darstellung gezeigt, die Werte 0, 0, 0, 1, 1, 0 und 1.



Bei einer Maske handelt es sich um ein 2D-Array. Jede Zeile in dem Array stellt eine Kurve in dem digitalen Graph dar. Um die Bits 1 und 5 in der Kurve darzustellen, geben Sie 1 in einem Element des Arrays ein und 5 in einem anderen Element (siehe Abbildung D-2). Geben Sie in den anderen Elementen des Arrays, die kein Bit darstellen sollen, -1 ein.

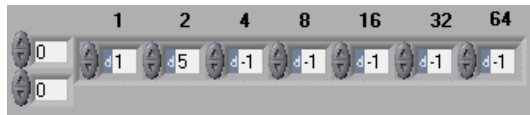
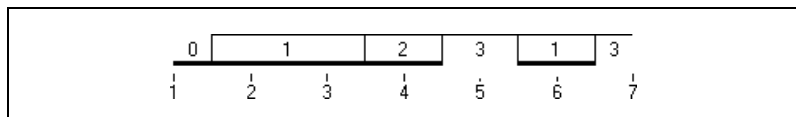


Abbildung D-2. Beispiel eines Array-Bedienelements für eine Maske

In diesem Beispiel wird jedes Element im Array durch eine Zweier-Potenz dargestellt (2^0 bis 2^6). Anhand dieser Zahlen können Sie festlegen, welches Element im Array ein Bit im Graph darstellt. In diesem Beispiel stellt Element 1 im Graph Bit 1 dar. Der Wert 1 im Bit wird daher in der Kurve als 1 dargestellt und der Wert 0 in der Kurve als 0. Element 2 stellt im Graph Bit 5 dar. Der Wert 1 im Bit wird daher in der Kurve als 2 dargestellt und der Wert 0 in der Kurve als 0.

Das Array in Abbildung D-2 führt zu der folgenden Darstellung im digitalen Kurvengraph.



Diese Darstellung zeigt:

- Bit 1 und Bit 5 im ersten Element beinhalten den Wert 0.
- Bit 1 beinhaltet im zweiten und im dritten Element den Wert 1. Bit 5 beinhaltet im zweiten und im dritten Element den Wert 0.
- Bit 1 im vierten Element beinhaltet den Wert 0. Bit 5 im vierten Element beinhaltet den Wert 1.
- Bit 1 und Bit 5 im fünften Element beinhalten den Wert 1.
- Bit 1 im sechsten Element beinhaltet den Wert 1. Bit 5 im sechsten Element beinhaltet den Wert 0.
- Bit 1 und Bit 5 im siebten Element beinhalten den Wert 1.

Die in der Darstellung angezeigten Zahlen werden aus der folgenden Gleichung abgeleitet:

$$\begin{aligned}
 & [\text{Bit-Wert (1 or 0)}] \times 2^{[\text{Elementennummer}]} \\
 & + [\text{Weiterer Bit-Wert (1 or 0)}] \times 2^{[\text{weitere Elementennummer}]} \\
 & \quad \vdots \\
 & + [n \text{ Bit-Wert (1 or 0)}] \times 2^{[n \text{ Elementennummer}]} \\
 \hline
 & \text{Wert der Kurve}
 \end{aligned}$$

In der Abbildung D-3 stellen das vierte (8) und sechste (32) Element Bit 5 beziehungsweise Bit 1 dar.

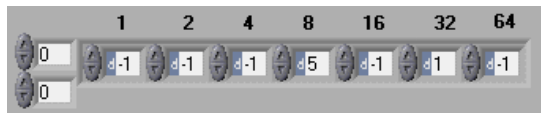
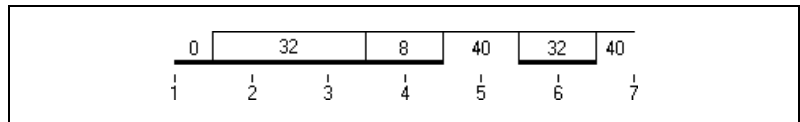


Abbildung D-3. Angeben derselben Bits in verschiedenen Elementen

Das Array in Abbildung D-3 führt zu der folgenden Darstellung. Beachten Sie, dass die Kurven ähnlich aussehen, sich die Werte aber unterscheiden.



Jede Zeile in der Maske entspricht einer Kurve im Graph. Fügen Sie eine neue Zeile zur Maske hinzu, um den digitalen Graph, wie in Abbildung D-4 gezeigt, um eine neue Kurve zu erweitern.

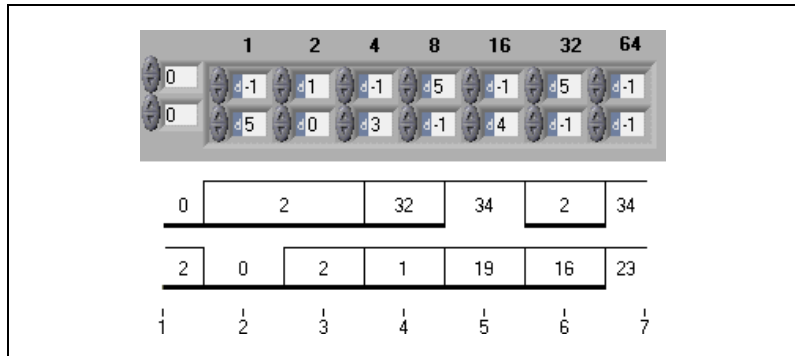


Abbildung D-4. Erstellen von zwei Kurven maskierter Daten in einem digitalen Kurvengraph

Technischer Support

Support im Internet

National Instruments Support im Internet gibt Antworten auf Ihre Fragen und bietet Lösungen zu Problemen mit Installation, Konfiguration und Applikation. Die Online-Problemlösung und Diagnostic umfasst eine Auflistung häufig gestellter Fragen (FAQs), Datenbanken (Knowledge Database), produktspezifische Assistenten zur Fehlersuche und -behebung, Handbücher, Treiber, Softwareaktualisierungen und vieles mehr. Support im Internet auf den entsprechenden Webseiten unter ni.com

NI Developer Zone

Die NI Developer Zone finden Sie im Internet unter ni.com/zone und ist die wichtigste Quelle zum Einrichten von Mess- und Automationssystemen. In der Developer Zone finden Sie ganz einfach die neusten Beispielprogramme, Konfigurationsdaten, Lernhilfen, Aktuelles und den Kontakt zu anderen Entwicklern, mit denen Sie Erfahrungen austauschen können.

Veranstaltungen (Customer Education)

National Instruments bietet viele Möglichkeiten, Sie Ihren Wünschen entsprechend zu unterrichten oder weiterzubilden. Sie können zwischen Handbüchern für ein Selbststudium, Videos, interaktiven CDs oder weltweit von Ausbildern geleiteten Hands-On-Kursen wählen. Mehr Informationen zu Kursterminen, Kursbeschreibungen, Ausbildungszentren und Anmeldung finden Sie auf unserer Webpage ni.com/germany unter Veranstaltungen.

Systemintegration

Sollten Sie nicht genügend Zeit, eingeschränkte technische Mittel oder andere Probleme haben, können Sie sich auch jeder Zeit an unsere Berater und Dienstleister wenden. Sie können sich auf ein hervorragendes Sachverständnis unseres weltweiten Netzes von Mitgliedern des Alliance

Programmes verlassen. Mehr Informationen zu Alliance-Lösungen bezüglich Systemintegration finden Sie auf unserer Webpage ni.com unter Alliance Programm.

Weltweiter Support

Die Niederlassungen von National Instruments garantieren weltweit schnelle Hilfe durch Ansprechpartner in Ihrer näheren Umgebung. Sie können auf unserer Webpage ni.com unter Worldwide Offices auf die Seiten der verschiedenen Niederlassungen zugreifen. Diese Seiten bieten Ihnen neuste Kontaktinformationen, Support-Telefonnummern, E-Mail-Adressen und Informationen zu aktuellen Veranstaltungen.

Falls Sie nach dem Durchsuchen unserer Webpage zum technischen Support keine Antworten zu Ihren Fragen gefunden haben, wenden Sie sich an Ihre lokale Niederlassung oder an die Hauptniederlassung von National Instruments. Die Telefonnummern der weltweiten Niederlassungen finden Sie am Anfang dieses Benutzerhandbuchs.

Glossar

Präfix	Bedeutung	Wert
m-	milli-	10^{-3}
k-	kilo-	10^3
M-	Mega-	10^6

Zahlen/Symbole

Δ	Delta; Differenz. Δx bezeichnet den Wert, um den x von einem Index zum nächsten geändert wird.
π	Pi.
∞	Unendlich.
1D	Eindimensional.
2D	Zweidimensional.
3D-Kurve	Spezielle parametrische Kurve $(x(t), y(t), z(t))$, wobei der Parameter t ein gegebenes Intervall durchläuft.

A

A	Ampere.
Ablaufunterschied	Tritt auf, wenn zwei oder mehrere Teile eines Codes, die parallel ausgeführt werden, den Wert der gleichen gemeinsam genutzten Ressource ändern, in der Regel eine globale oder lokale Variable.
Absolute Koordinaten	Bildkoordinaten relativ zum Ursprung (0,0) des Monitors.
Absoluter Pfad	Datei- oder Verzeichnispfad, der den Speicherort relativ zur höchsten Ebene des Dateisystems beschreibt.
Abtastwert	Einzelner analoger oder digitaler Ein- oder Ausgabedatenwert.

AC	Wechselstrom.
Aktives Fenster	Das Fenster, in dem Anwendereingaben vorgenommen werden können; in der Regel das Fenster, das den Fokus besitzt. Die Titelleiste des aktiven Fensters ist hervorgehoben. Sie können ein Fenster aktivieren, indem Sie darauf klicken oder indem Sie es im Menü Fenster auswählen.
Aktuelles VI	VI, dessen Frontpanel, Blockdiagramm oder Symbol-Editor das aktive Fenster darstellt.
Anschluss	Ein Teil eines VIs oder eines Funktionsknotens mit Eingangs- und Ausgangsanschlüssen. Die Daten werden durch einen Anschluss zum Knoten und wieder zurück geleitet.
Anschluss	Objekt oder Bereich an einem Knoten, durch den Daten fließen.
Anschlussfeld	Bereich oben rechts auf dem Frontpanel oder Blockdiagramm, in dem das Anschlussmuster des VIs angezeigt wird. Hier werden die Ein- und Ausgänge definiert, die Sie mit einem VI verbinden können.
Anzeigeelement	Ein Frontpanel-Objekt, das Ausgabewerte anzeigt, wie zum Beispiel ein Graph oder eine LED.
Applikations-Software	Applikation, die mit Hilfe des LabVIEW-Entwicklungssystems erstellt und im LabVIEW-Laufzeitsystem ausgeführt wird.
Array	Geordnete, indizierte Liste aus Datenelementen des gleichen Typs.
ASCII	American Standard Code for Information Interchange.
Auto-Indizierung	Fähigkeit von Schleifenstrukturen, an ihren Begrenzungen Arrays zusammenzuführen und zu trennen. Wenn ein Array mit aktivierter Auto-Indizierung in eine Schleife eintritt, wird es automatisch von der Schleife zerlegt, indem Skalare aus 1D-Arrays, 1D-Arrays aus 2D-Arrays und so weiter extrahiert werden. Schleifen fassen Daten in Arrays zusammen, wenn die Daten eine Schleife wieder verlassen.
automatische Skalierung	Die Fähigkeit von Skalen, sich an den Bereich der dargestellten Werte anzupassen. In Diagrammskalen bestimmt die automatische Skalierung die maximalen und minimalen Skalierungswerte.

B

Bedienelement	Frontpanel-Objekt zur interaktiven Eingabe von Daten in ein VI oder zur programmatischen Eingabe von Daten in ein Sub-VI, wie zum Beispiel ein Drehknopf, eine Taste oder ein Drehregler.
Bedienelemente und Anzeigen für Strings	Frontpanel-Objekte zum Ändern und Anzeigen von Text.
Bedienwerkzeug	Werkzeug zur Eingabe von Daten in Bedienelemente.
Bedingungsanschluss	Anschluss einer While-Schleife, der einen booleschen Wert enthält, mit dem bestimmt wird, ob ein neuer Schleifendurchlauf ausgeführt wird.
Benutzerdefinierte Konstante	Blockdiagrammobjekt, das einen von Ihnen festgelegten Wert enthält.
Bereich	Bereich zwischen den Grenzwerten, innerhalb dessen eine Größe gemessen, empfangen oder gesendet wird. Wird anhand der unteren und oberen Bereichsgrenzwerte angegeben.
Beschriftung	Textobjekt, das zur Benennung oder Beschreibung von Objekten oder Bereichen auf dem Frontpanel oder im Blockdiagramm verwendet wird.
Beschriftungswerkzeug	Werkzeug zum Erstellen von Beschriftungen und zur Eingabe von Text in Textfenster.
Bibliothek	<i>Siehe VI-Bibliothek.</i>
Bild	Serie grafischer Anweisungen, die von einem Bildanzeigeelement verwendet werden, um ein Bild zu erzeugen.
Bildanzeigeelement	Allgemeines Anzeigeelement zur Anzeige von Bildern, die Linien, Kreise, Text und andere Arten grafischer Formen enthalten.
Bildlaufwerkzeug	Werkzeug, mit dem man einen Bildlauf durch Fenster durchführen kann.
Blockdiagramm	Grafische Beschreibung oder Darstellung eines Programms oder Algorithmus. Das Blockdiagramm besteht aus ausführbaren Symbolen, sogenannten "Knoten" und Verbindungen, die Daten zwischen den Knoten transportieren. Das Blockdiagramm ist der Quell-Code für das VI. Das Blockdiagramm befindet sich im Blockdiagramm-Fenster des VIs.

Boolesche Bedien- und Anzeigeelemente	Frontpanel-Objekte zum Verändern und Anzeigen boolescher Daten (TRUE oder FALSE).
Byte-Stream-Datei	Eine Datei, in der Daten als eine Sequenz von ASCII-Zeichen oder Bytes gespeichert werden.

C

Case-Struktur	Konditionale Abschnittssteuerungsstruktur, mit der eines der Unterdiagramme basierend auf der Eingabe in der Case-Struktur ausgeführt wird. Es handelt sich hier um eine Kombination aus den Angaben IF, THEN, ELSE und CASE in Steuerflusssprachen.
Case-Zweig	Ein Unterdiagramm einer Case-Struktur.
CIN	<i>Siehe</i> Code-Interface-Knoten (CIN).
Cluster	Ein Satz unindizierter, geordneter Datenelemente beliebiger verschiedener Datentypen wie numerisch, boolesch, String, Array oder Cluster. Die Elemente müssen entweder alle Bedien- oder Anzeigeelemente sein.
Code-Interface-Knoten (CIN)	CIN. Ein spezieller Blockdiagrammknoten, über den Sie textbasierten Code mit einem VI verknüpfen können.

D

D	Delta; Differenz. Δx bezeichnet den Wert, um den x von einem Index zum nächsten geändert wird.
DAQ	<i>Siehe</i> Datenerfassung .
DAQ-Kanalassistent	Utility, das Sie durch das Benennen und Konfigurieren von analogen und digitalen DAQ-Kanälen führt. Steht in der Datenumgebung von Measurement & Automation Explorer (Windows) oder im DAQ Channel Wizard (Macintosh) zur Verfügung.
Darstellung	Untertyp des numerischen Datentyps, in dem es Byte mit und ohne Vorzeichen, Word- und Long-Integer sowie Single-, Double- und Extended-Fließkommazahlen gibt.
Datei-Refnum	<i>Siehe</i> Refnum .

Datenabhängigkeit	Bedingung in einer Datenflussprogrammiersprache, bei der ein Knoten solange nicht ausgeführt werden kann, bis Daten von einem anderen Knoten empfangen wurden. <i>Siehe auch</i> Künstliche Datenabhängigkeit .
Datenerfassung	DAQ. Das Verfahren zur Erfassung von Daten, normalerweise von A/D- oder digitalen Eingangs-Plug-In-Geräten.
Datenfluss	Programmiersystem, das aus ausführbaren Knoten besteht, die nur ausgeführt werden, wenn alle erforderlichen Eingangsdaten empfangen wurden. Die Ausführung erzeugt automatisch die Ausgangsdaten. LabVIEW ist ein Datenflusssystem.
Datenprotokolldatei	Datei, in der Daten als Sequenz von Datensätzen eines einzigen, beliebigen Datentyps gespeichert werden, den Sie beim Erstellen der Datei angeben müssen. Obwohl alle Datensätze in einer Datenprotokolldatei vom gleichen Typ sein müssen, kann dieser Typ komplex sein. Sie können beispielsweise festlegen, dass es sich bei jedem Datensatz um einen Cluster handelt, der einen String, eine Zahl und ein Array enthält.
Datenprotokollierung	Erfassen von Daten und deren gleichzeitige Speicherung in einer Datei. LabVIEW-Datei-I/O-VIs und -Funktionen können Daten protokollieren.
Datentyp	Format für Informationen. In LabVIEW sind die für die meisten VIs und Funktionen akzeptablen Datentypen numerisch, Array, String, boolesch, Pfad, Refnum, Enum, Signalform und Cluster.
DDE	<i>Siehe</i> Dynamischer Datenaustausch.
Diagramm	Zweidimensionale Anzeige einer oder mehrerer Kurven, in der vorherige Daten bis zu einem definierten Maximum erhalten bleiben. Die Daten werden in das Diagramm übertragen und die Anzeige wird Wert für Wert oder Array für Array aktualisiert. Eine bestimmte Anzahl vorhergehender Werte wird für Anzeigezwecke in einem Puffer gespeichert. <i>Siehe auch</i> Oszilloskop-Diagramm , Streifenschreiberdiagramm und Laufdiagramm .
diskret	Nicht kontinuierliche Werte einer unabhängigen Variable, in der Regel die Zeit.
DLL	Dynamic Link Library.
Dynamischer Datenaustausch	DDE. Methode zur Weiterleitung von Daten zwischen Applikationen ohne Eingreifen oder Überwachung durch den Anwender.

E

Eigenschaftsknoten	Setzt oder liest die Eigenschaften eines VIs oder einer Applikation.
Eindimensional	Das Vorhandensein einer Dimension, wie bei einem Array mit nur einer Zeile von Elementen.
Ereignis	Bedingung oder Status eines analogen oder digitalen Signals.
Externer Trigger	Spannungsimpuls von einer externen Quelle, der ein Ereignis auslöst, wie zum Beispiel eine A/D-Wandlung.

F

Farbkopierwerkzeug	Kopiert Farben, um diese mit dem Farbwerkzeug einzufügen.
Farbwerkzeug	Werkzeug, mit dem Vorder- und Hintergrundfarben eingestellt werden können.
Fehlerausgang	Fehlerstruktur, die ein VI verlässt.
Fehlereingang	Fehlerstruktur, die an ein VI übergeben wird.
Fehlermeldung	Angabe eines Software- oder Hardware-Fehlers oder des Versuchs einer inakzeptablen Dateneingabe.
Fehlerstruktur	Besteht aus einem booleschen Status-Anzeigeelement, einem numerischen Code-Anzeigeelement und einem String-Quellen-Anzeigeelement.
FIFO	Durchlaufspeicherpuffer (First-In-First-Out). Die ersten gespeicherten Daten werden als erste Daten an den Empfänger gesendet.
Filterung	Typ der Signalkonditionierung, der das Filtern unerwünschter Signale aus dem zu messenden Signal ermöglicht.
Formatumwandlungspunkt	Punkt auf einem Anschluss, der angibt, dass einer von zwei miteinander verbundenen Anschlüssen von LabVIEW so konvertiert wurde, dass er mit dem Datentyp des anderen Anschlusses übereinstimmt.
Formelknoten	Knoten, der Gleichungen ausführt, die als Text eingegeben wurden. Besonders hilfreich für lange Gleichungen, die in Form von Blockdiagrammen zu komplex werden.

For-Schleife	Iterative Schleifenstruktur, die ihre Unterdiagramme so oft wie vorher festgelegt ausführt. Entspricht dem textbasierten Code: <code>For i = 0 to n - 1, do....</code>
Freie Beschriftung	Beschriftung auf dem Frontpanel oder Blockdiagramm, das zu keinem anderen Objekt gehört.
Frontpanel	Interaktive Anwenderschnittstelle eines VIs. Die Frontpanel-Darstellung bildet physische Instrumente nach, wie zum Beispiel Oszilloskope und Multimeter.
Funktion	Integriertes Ausführungselement; vergleichbar mit einem Operator, einer Funktion oder einer Anweisung in einer textbasierten Programmiersprache.

G

Gebrochene Schaltfläche Ausführung	Die Schaltfläche, die anstelle der Schaltfläche Ausführung angezeigt wird, wenn ein VI aufgrund von Fehlern nicht ausgeführt werden kann.
Geglättete Daten	Daten eines beliebigen Typs, die in einen String aus Binärwerten formatiert wurden, um sie in der Regel in eine Datei zu schreiben.
General Purpose Interface Bus (Schnittstellenbus für allgemeine Zwecke)	GPIB ist synonym mit HP-IB. Der Standardbus zum Steuern von elektronischen Instrumenten mit einem Computer. Wird auch IEEE 488-Bus genannt, da er in den ANSI/IEEE-Standards 488-1978, 488.1-1987 und 488.2-1992 definiert ist.
Gerät	Instrument oder Controller, der als einzelne Einheit adressiert werden kann und reale I/O-Werte steuert oder überwacht. Ein Gerät wird oftmals über irgendein Datenübertragungsnetzwerk an einen Host-Computer angeschlossen. Es kann sich auch um ein Plug-In-Gerät handeln. Bei Datenerfassungsapplikationen befindet sich ein DAQ-Gerät in einem Computer beziehungsweise ist ein DAQ-Gerät direkt an den parallelen Anschluss des Computers angeschlossen. Einsteckkarten, PCMCIA-Karten und Geräte wie DAQPad-1200, die an den parallelen Anschluss eines Computers angeschlossen werden, sind Beispiele für DAQ-Geräte. SCXI-Module unterscheiden sich mit Ausnahme von SCXI-1200, bei dem es sich um ein hybrides Gerät handelt, von Geräten.
Globale Variable	Greift auf Daten verschiedener VIs in einem Blockdiagramm zu und leitet sie weiter.

Glyph	Kleines Bild oder Symbol.
GPIB	<i>Siehe</i> General Purpose Interface Bus (Schnittstellenbus für allgemeine Zwecke)
Graph	Zweidimensionale Anzeige einer oder mehrerer Kurven. Ein Graph empfängt und stellt die Daten als Block dar.
Graph-Bedienelement	Frontpanel-Objekt, das Daten in einer kartesischen Ebene anzeigt.
Gruppe	<p>Sammlung von Eingangs- und Ausgangskanälen oder -Anschlüssen, die Sie definieren. Gruppen können Analogeingangskanäle, Analogausgangskanäle, Digitaleingangskanäle, Digitalausgangskanäle oder Zähler/Timer-Kanäle enthalten. Eine Gruppe kann nur einen Kanaltyp enthalten. Verwenden Sie eine Task-ID, um nach dem Erstellen auf eine Gruppe Bezug nehmen zu können. Sie können gleichzeitig bis zu 16 Gruppen definieren.</p> <p>Um eine Gruppe zu löschen, leiten Sie ein leeres Kanal-Array und die Gruppennummer zum VI Gruppenkonfiguration weiter. Sie müssen eine Gruppe nicht löschen, um deren Mitgliedschaft zu ändern. Wenn Sie eine Gruppe mit aktiver Task neu konfigurieren, wird von LabVIEW die Task gelöscht und es wird eine Warnung ausgegeben. Die Task wird von LabVIEW auch nach der Neukonfiguration der Gruppe nicht erneut gestartet.</p>
H	
Haltepunkt	Hält die Ausführung für die Fehlersuche an.
Haltepunkt-Werkzeug	Ein Werkzeug zum Festlegen eines Haltepunkt in einem VI, Knoten oder an einem Verbindungsstück.
Handle	Zeiger auf einen Speicherblock-Zeiger, der Referenz-Arrays und -Strings verwaltet. Ein String-Array ist ein Handle für einen Speicherblock, der Handles für Strings beinhaltet.
Handles zur Größenänderung	Gewinkelte Handles in der Ecke von Objekten, welche die Punkte zum Ändern der Größe angeben.
Hex	Hexadezimal. Zahlensystem mit der Basis 16.
Hierarchiefenster	Fenster, das die Hierarchie von VIs und Sub-VIs grafisch darstellt.

Highlight-Funktion Technik zur Fehlersuche, bei der die VI-Ausführung animiert wird, um den Datenfluss im VI darzustellen.

Hinweisstreifen Kleines gelbes Textbanner, das einen Anschlussnamen anzeigt und damit die Identifizierung des Anschlusses für Verbindungen erleichtert.

I

I/O Input/Output (Eingabe/Ausgabe). Die Übertragung von Daten zu oder von einem Computer-System einschließlich Kommunikationskanälen, Eingangsgeräten und/oder Datenerfassungs- und Steuerschnittstellen.

IEEE Institute for Electrical and Electronic Engineers.

Inf Numerischer Anzeigewert für die Fließkomma-Darstellung von unendlich.

Instrumententreiber Ein VI, das programmierbare Instrumente steuert.

Integer (Ganze Zahl) Eine beliebige ganze Zahl, deren Wert negativ, null oder positiv sein kann.

Intensitätsgraph Methode, drei Dimensionen von Daten in einer zweidimensionalen Darstellung mit Hilfe von Farben anzuzeigen.

IP Internet-Protokoll.

Iterationsanschluss Anschluss für eine For- oder While-Schleife, der die aktuelle Anzahl der ausgeführten Schleifendurchläufe enthält.

K

Kanal Pin oder verdrahteter Anschluss, von/zu dem ein analoges oder digitales Signal gelesen oder geschrieben wird. Analogsignale können einseitig oder differentiell sein. Bei digitalen Signalen werden Kanäle zur Bildung von Ports gruppiert. Ports bestehen in der Regel aus vier oder acht digitalen Kanälen.

Kanalname Eindeutiger Name, mit dem eine Kanalkonfiguration im Measurement & Automation Explorer bezeichnet wurde.

Klonen	<p>Das Kopieren eines Bedienelements oder eines anderen Objekts durch Klicken auf das Objekt bei gedrückter <Strg>-Taste und Ablegen der Kopie an einer neuen Position.</p> <p>(Macintosh) Drücken Sie die Wahltaste. (Sun) Drücken Sie die <Meta>-Taste. (HP-UX und Linux) Drücken Sie die <Alt>-Taste.</p> <p>(UNIX) Sie können ein Objekt auch duplizieren, indem Sie mit der mittleren Maustaste auf das Objekt klicken und die Kopie dann an die neue Position ziehen.</p>
Knoten	<p>Programmausführungselement. Knoten sind analog zu Anweisungen, Operatoren, Funktionen und Subroutinen in textbasierten Programmiersprachen. In einem Blockdiagramm umfassen Knoten Funktionen, Strukturen und Sub-VIs.</p>
Kompilieren	<p>Ein Verfahren, durch das Code einer höheren Programmiersprache in maschinenlesbaren Code umgewandelt wird. In LabVIEW werden VIs automatisch kompiliert, bevor diese nach dem Erstellen oder Ändern zum erstenmal ausgeführt werden.</p>
Konfigurationsprogramm	<p>Bezeichnet unter Windows den Measurement & Automation Explorer und das NI-DAQ-Konfigurationsprogramm auf einem Macintosh.</p>
Konstante	<p><i>Siehe</i> Universelle Konstante und Benutzerdefinierte Konstante.</p>
Kontexthilfe -Fenster	<p>Spezielles Fenster in LabVIEW, in dem die Namen und Positionen der Anschlüsse für ein VI oder eine Funktion, die Beschreibung von Bedien- und Anzeigeelementen, die Werte von Universalkonstanten und Beschreibungen der Datentypen von Eigenschaften der Bedienelemente angezeigt werden.</p>
Kontextmenü	<p>Menü, auf das durch Klicken mit der rechten Maustaste auf ein Objekt zugegriffen wird. Menüoptionen gehören speziell zu diesem Objekt.</p>
Kontrollkästchen	<p>Kleines rechteckiges Kästchen in einem Dialogfeld, das Sie aktivieren oder deaktivieren können. Kontrollkästchen sind normalerweise mehreren Optionen zugeordnet, die festgelegt werden können. Sie können mehr als ein Kontrollkästchen aktivieren.</p>
Konvertierung	<p>Ändern des Typs eines Datenelements.</p>
Künstliche Datenabhängigkeit	<p>Bedingung in einer Datenflussprogrammiersprache, in der die Datenankunft, nicht deren Werte, die Ausführung des Knotens auslösen.</p>

Kurve	Grafische Darstellung eines Daten-Arrays, das entweder in Form eines Graphen oder Diagramms angezeigt wird.
Kurvendiagramm	Anzeigeelement, das Daten mit einer bestimmten Rate grafisch darstellt.
L	
LabVIEW	Laboratory Virtual Instrument Engineering Workbench. LabVIEW ist eine grafische Programmiersprache, in der zum Erstellen von Programmen Symbole an Stelle von Textzeilen verwendet werden.
Laufwerk	Ein Buchstabe im Bereich a-z, gefolgt von einem Doppelpunkt (:), um logische Laufwerke zu bezeichnen.
LED	Leuchtdiode (Light Emitting Diode).
Leeres Array	Array, das null Elemente beinhaltet, aber einen definierten Datentyp aufweist. Beispielsweise ist ein Array, das in seinem Datenfenster ein numerisches Bedienelement anzeigt, aber über keine definierten Werte für alle Elemente verfügt, ein leeres numerisches Array.
Legende	Objekt aus einem Diagramm oder einem Graphen, mit dem die Namen und Darstellungsstile von Kurven im Diagramm oder Graphen angezeigt werden.
Listenfeld	Ein Feld in einem Dialogfeld, in dem alle verfügbaren Möglichkeiten für eine Aktion aufgelistet werden, Beispielsweise eine Liste der Dateinamen auf einer Festplatte.
LLB	VI-Bibliothek.
Lokale Sequenz-Variable	Anschluss zur Weiterleitung von Daten zwischen Rahmen einer Sequenzstruktur.
Lokale Variable	Eine Variable, die es ermöglicht, von/in ein Bedien- oder Anzeigeelement auf dem Frontpanel eines VIs zu lesen oder zu schreiben.
M	
Matrix	zweidimensionales Array.
Measurement & Automation Explorer	Die National Instruments-Standardumgebung unter Windows für die Konfiguration und Diagnose von Hardware.

Menüleiste	Horizontale Leiste, in der die Namen des Hauptmenüs einer Applikation aufgelistet werden. Die Menüleiste wird unter der Titelleiste eines Fensters angezeigt. Jede Applikation verfügt über ihre eigene charakteristische Menüleiste. Einige Menüs und Befehle sind jedoch vielen Applikationen gemeinsam.
Methode	Eine Aktion, die ausgeführt wird, wenn ein Objekt eine Meldung empfängt. Eine Methode ist immer einer Klasse zugeordnet.
Multithread-Applikation	Eine Applikation, die mehrere verschiedene Threads unabhängig voneinander ausführt. Auf einem Computer mit mehreren Prozessoren können diese verschiedenen Threads simultan auf verschiedenen Prozessoren ausgeführt werden.

N

NaN	Digitaler Anzeigewert für die Fließkommazahldarstellung <i>Keine Zahl</i> . Normalerweise das Ergebnis einer nicht definierten Operation, wie zum Beispiel $\log(-1)$.
Nicht ausführbares VI	Ein VI, das aufgrund von Fehlern nicht ausgeführt werden kann; wird durch einen unterbrochenen Pfeil in der gebrochenen Schaltfläche Ausführung angezeigt.
NI-DAQ	Umfassende Treibersoftware, die zum Lieferumfang aller National Instruments-Datenerfassungshardware gehört.
Numerische Bedien- und Anzeigeelemente	Frontpanel-Objekte zum Ändern und Anzeigen von numerischen Daten.

O

Objekt	Generischer Begriff für ein beliebiges Element auf dem Frontpanel oder im Blockdiagramm, einschließlich Bedienelementen, Anzeigeelementen, Knoten, Verbindungsstücken und importierten Bildern.
Objektkontextmenü-Werkzeug	Werkzeug für den Zugriff auf das Kontextmenü eines Objekt.

OLE	Object Linking and Embedding.
Oszilloskop-Diagramm	Numerisches Anzeigeelement, modelliert nach dem Betrieb eines Oszilloskops.

P

Palette	Anzeige von Symbolen, die mögliche Optionen darstellen.
Palette Elemente	Eine Palette, die Bedien-, Anzeige- und Gestaltungselemente für das Frontpanel enthält.
Palette Funktionen	Eine Palette, die VIs, Funktionen, Blockdiagrammstrukturen und Konstanten enthält.
Palette Werkzeuge	Palette, die Werkzeuge enthält, die Sie zum Bearbeiten und zur Fehlersuche von/in Objekten im Frontpanel und Blockdiagramm verwenden können.
Panel-Fenster	VI-Fenster, in dem das Frontpanel, die Symbolleiste sowie das Symbol- und Anschlussfeld enthalten sind.
Pixel	Kleinste Einheit eines digitalisierten Bildes.
Pixmap	Standardformat zur Bildspeicherung, bei der ein Farbwert jedes Pixel darstellt. Eine Bitmap ist die Schwarz/Weiß-Version einer Pixmap.
Polymorphismus	Die Fähigkeit eines Knotens, sich automatisch auf Daten unterschiedlicher Darstellungsformen, Typen oder Strukturen einzustellen.
Positionierwerkzeug	Werkzeug zum Verschieben von Objekten und zum Ändern derer Größe.
PPC	Program-to-program communication (Kommunikation zwischen Programmen).
Probe	Fehlersuchfunktion zum Überprüfen von Zwischenwerten in einem VI.
Probedaten-Werkzeug	Werkzeug zum Erstellen von Probedaten an Verbindungen.
programmgesteuertes Drucken	Automatisches Drucken von VI-Frontpanels nach der Ausführung.
Puffer	Temporärspeicher für erfasste oder erzeugte Daten.

Pulldown-Menüs	Menüs, auf die in der Menüleiste zugegriffen werden kann. Pulldown-Menüs enthalten in der Regel allgemeine Menüoptionen.
Punkt	Ein Cluster, das zwei 16-Bit-Integer enthält, die horizontale und vertikale Koordinaten darstellen.
PXI	PCI eXtensions for Instrumentation. Eine modulare, computerbasierte Instrumentierungsplattform.

R

Rahmen	Unterdiagramm einer Sequenzstruktur.
Rechteck	Ein Cluster, der vier 16-Bit-Integer enthält. Die ersten beiden Werte beschreiben die vertikalen und horizontalen Koordinaten der oberen linken Ecke. Die letzten beiden Werte beschreiben die vertikalen und horizontalen Koordinaten der unteren rechten Ecke.
Refnum	Referenz. Ein Bezeichner, den LabVIEW einer geöffneten Datei zuordnet. Verwenden Sie Refnums, um anzugeben, dass eine Funktion oder ein VI eine Operation in der offenen Datei ausführen soll.
relative Koordinaten	Bildkoordinaten, die relativ zur aktuellen Cursor-Position sind.
Ring-Bedienelement	Spezielles numerisches Bedienelement, das 32-Bit-Ganzzahlen, beginnend bei 0 sequentiell ansteigend, einer Reihe von Textbeschriftungen oder Grafiken zuordnet.

S

Schieber	Der bewegliche Teil von Schieberegler-Bedien- und Anzeigeelementen.
Schieberegister	Optionalen Mechanismus in Schleifenstrukturen, um Variablenwerte von einem Schleifendurchlauf an den nächsten zu übergeben.
SCXI	Signal Conditioning eXtensions for Instrumentation. Die National Instruments-Produktlinie für aufbereitete Signale mit geringem Pegel in einem externen Gehäuse in der Nähe der Sensoren, sodass in einer gestörten Umgebung nur Signale mit hohen Pegeln an DAQ-Geräte gesendet werden.

Sequenz-Struktur	Programmsteuerstruktur, bei der die Unterdiagramme in numerischer Reihenfolge ausgeführt werden. Verwenden Sie diese Struktur, um datenunabhängige Knoten zu zwingen, Vorgänge in der von Ihnen gewünschten Reihenfolge auszuführen.
Signalverlauf	Mehrere Spannungswerte, die mit einer spezifischen Abtastrate erfasst wurden.
Skalar	Zahl, die durch einen Punkt auf einer Skala dargestellt werden kann. Im Gegensatz zu einem Array ein einzelner Wert. Boolesche Skalarwerte und Cluster sind explizit singuläre Instanzen ihrer jeweiligen Datentypen.
Skalierung	Teil eines Diagramms, Graphen und einiger numerischer Bedien- und Anzeigeelemente, die eine Reihe von Markierungen oder Punkten in bekannten Intervallen enthalten, um Messeinheiten zu bezeichnen.
Speicherpuffer	<i>Siehe Puffer.</i>
Standard	Voreingestellter Wert. Für viele VI-Eingänge werden Standardwerte verwendet, wenn Sie keinen Wert angeben.
Steuerfluss	Programmierungssystem, in dem die sequentielle Reihenfolge von Anweisungen die Ausführungsreihenfolge bestimmt. Die meisten textbasierten Programmiersprachen sind Steuerflusssprachen.
Streifenschreiberdiagramm	Numerisches Anzeigeelement grafischer Darstellungen modelliert nach einem Papierstreifendiagrammrekorder, der sich verschiebt, wenn Daten grafisch dargestellt werden.
String	Darstellung eines Wertes als Text.
Struktur	Ein Programmsteuerelement, zum Beispiel eine Sequenzstruktur, eine Case-Struktur, eine For-Schleife oder eine While-Schleife.
Sub-VI	Ein VI, das im Blockdiagramm eines anderen VIs verwendet wird. Vergleichbar mit einer Subroutine.
Laufdiagramm	Numerisches Anzeigeelement, modelliert nach dem Betrieb eines Oszilloskops. Es ist ähnlich dem Oszilloskopdiagramm, mit der Ausnahme, dass eine Linie über die Anzeige gleitet, um alte von neuen Daten zu trennen.
Symbol	Grafische Darstellung eines Knotens in einem Blockdiagramm.

Symbolleiste	Leiste, die Befehlsschaltflächen zum Ausführen von VIs und zur Fehlersuche enthält.
Syntax	Ein Regelsatz, dem Anweisungen einer bestimmten Programmiersprache entsprechen müssen.

T

TCP	Transmission Control Protocol.
Treiber	Software, die ein bestimmtes Hardware-Gerät, wie zum Beispiel ein DAQ-Gerät, steuert.
Tunnel	Dateneingangs- oder -ausgangsanschluss in einer Struktur.

U

Überlagerung	Die Addition von Gauß'schem Rauschen zu einem analogen Eingangssignal. Durch Überlagerung eines Hilfssignals und anschließende Mittelung der Eingangsdaten können Sie die Auflösung um ein weiteres halbes Bit erhöhen.
UDP	User Datagram Protocol.
Universelle Konstante	Nicht bearbeitbares Blockdiagramm-Objekt, das ein bestimmtes ASCII-Zeichen oder eine numerische Standardkonstante ausgibt, zum Beispiel π .
Unterdiagramm	Blockdiagramm innerhalb der Grenzen einer Struktur.

V

Vektor	1D-Array.
Verbindungsbogen	Ein Punkt, an dem zwei Verbindungssegmente aufeinandertreffen.
Verbindungsknotenpunkt	Ein Punkt, an dem drei oder mehr Verbindungssegmente aufeinandertreffen.
Verbindungssegment	Einzelnes horizontales oder vertikales Verbindungsstück.
Verbindungsstichleitung	Abgeschnittene Leitung, die um ein VI- oder Funktionssymbol angezeigt wird, wenn Sie das Verbindungswerkzeug über das Symbol bewegen.

Verbindungsstück	Datenpfad zwischen Knoten.
Verbindungswerkzeug	Werkzeug zum Definieren von Datenpfaden zwischen Anschlüssen.
Verbindungsweig	Abschnitt einer Verbindung, der alle Verbindungssegmente von Kreuzung zu Kreuzung, Anschluss zu Kreuzung oder Anschluss zu Anschluss enthält, wenn es dazwischen keine Kreuzungen gibt.
Verzeichnis	Struktur zum Organisieren von Dateien in zweckmäßige Gruppen. Ein Verzeichnis ist mit einer Adresse vergleichbar, die den Speicherort der Dateien anzeigt. Ein Verzeichnis kann Dateien oder Unterverzeichnisse mit Dateien enthalten.
VI	<i>Siehe</i> Virtuelles Instrument (VI).
VI der höchsten Programmebene	Das oberste VI einer VI-Hierarchie. Dieser Begriff dient zur Unterscheidung des VIs von seinen Sub-VIs.
VI-Bibliothek	Spezielle Datei, die eine Sammlung in Beziehung stehender VIs für einen bestimmten Zweck enthält.
VI-Klasse	Eine Referenz auf ein virtuelles Instrument, mit dem auf die Eigenschaften und Methoden eines VIs zugegriffen werden kann.
Virtual Instrument Software Architecture	VISA. (VISA) Schnittstellenbibliothek zur Steuerung von GPIB, VXI, RS-232 und anderen Instrumententypen.
Virtuelles Instrument (VI)	Programm in LabVIEW, mit dem das Erscheinungsbild und die Funktion physikalischer Instrumente modelliert werden kann.
VISA	<i>Siehe</i> Virtual Instrument Software Architecture.
VI-Server	Mechanismus zur programmatischen Steuerung von VIs und LabVIEW-Applikationen (lokal und über Netzwerk).
VXI	VME eXtensions for Instrumentation (Bus).

W

Werkzeug	Spezieller Cursor zur Ausführung bestimmter Operationen.
Wertebereichs- erzwingung	Automatische Umwandlung, die in LabVIEW ausgeführt wird, um die numerische Darstellung von Datenelementen zu ändern.
While-Schleife	Schleifenstruktur, die einen Code-Abschnitt solange wiederholt, bis eine bestimmte Abbruchbedingung erfüllt ist.

Z

Zählanschluss	Anschluss für eine For-Schleife, dessen Wert bestimmt, wie oft die For-Schleife ihr Unterdiagramm ausführt.
Ziehen	Verwenden des Cursors auf dem Bildschirm, um Objekte auszuwählen, zu verschieben, zu kopieren oder zu löschen.
Zweidimensional	Zwei Dimensionen, wie zum Beispiel bei einem Array, das mehrere Zeilen und Spalten enthält.

Stichwortverzeichnis

Zahlen

2D-Bedien- und Anzeigeelemente, 4-9
3D-Bedien- und Anzeigeelemente, 4-9
3D-Graphen, 11-17

A

Abbildungen. *Siehe* Grafiken.
Abbrechen bei vorhandenen Fehlern. *Siehe* *LabVIEW-Hilfe*.
Ablaufunterschiede, 10-5
Abrufen von Daten
 mit Datei I/O-Funktionen, 13-22
 mit Sub-VIs, 13-20
 programmatisch, 13-20
ActiveX, 18-1
 Anzeigeelemente, 18-2
 Arbeiten im Netzwerk und, 17-1
 Bedienelemente, 18-2
 Clients, 18-3
 Container, 18-2
 Eigenschaften, 18-1
 Einfügen von Objekten im Frontpanel, 18-4
 Ereignisse, 18-2
 Erstellen von Unterpaletten, 3-6
 Festlegen von Parametern mit Hilfe von
 Konstanten, 18-5
 Funktionen, 18-2
 Konstanten zum Festlegen von
 Parametern, 18-5
 Methoden, 18-1
 Objekte, 18-1
 Server, 18-5
 Überblick, 18-1
 VIs, 18-2
 VI-Server, 16-1

 Zugriff auf ActiveX-fähige
 Applikationen, 18-4
 zum Ausführen von Skript-Knoten, 20-2
Aktualisieren von Paletten. *Siehe* *LabVIEW-Hilfe*.
Aktualisieren von VIs, 7-14
Ändern von Palettenansichten. *Siehe* *LabVIEW-Hilfe*.
Animierte Frontpanel-Bilder, 17-11
Anmeldefenster beim Start anzeigen. *Siehe* *LabVIEW-Hilfe*.
Anmerkungen, 4-20
 bearbeiten. *Siehe* *LabVIEW-Hilfe*.
Anpassen
 Arbeitsumgebung, 3-4
 Erscheinungsbild und Verhalten
 von VIs, 15-1
 Frontpanel-Objekte, 4-1
 Menüs, 15-2
 Paletten, 3-4
Anschlüsse, 2-3
 Anzahl, Auto-Indizierung zum
 Einstellen, 8-5
 anzeigen, 5-2
 Anzeigen von Hinweisstreifen. *Siehe* *LabVIEW-Hilfe*.
 Bedingung, 8-3
 Blockdiagramm, 5-2
 Entfernen von Funktionen, 5-11
 Frontpanel-Objekte und, 5-1
 hinzufügen zu Funktionen, 5-11
 Iteration
 For-Schleifen, 8-2
 While-Schleifen, 8-3
 Konstanten, 5-4
 lokale Sequenzanschlüsse, 8-11
 Muster, 7-8
 Selektor-, 8-8

- suchen. *Siehe LabVIEW-Hilfe.*
- Typumwandlungspunkte, 5-15
- Überblick, 2-3
- Verbindung, 5-11
- von Bedien- und Anzeigeelem.
(Tabelle), 5-2
- Zähl, 8-2
- Anschlussfelder, 2-5
 - Drucken, 14-4
 - einrichten, 7-7
 - erforderliche und optionale Ein- und
Ausgänge, 7-8
 - Überblick, 2-5
- Ansichten, 3-5
 - ändern. *Siehe LabVIEW-Hilfe.*
 - bearbeiten, 3-5
 - erstellen, 3-5
 - gemeinsam nutzen. *Siehe
LabVIEW-Hilfe.*
 - löschen. *Siehe LabVIEW-Hilfe.*
- Anwendungsobjekt
 - Bearbeiten von Einstellungen, 16-4
 - VI-Server, 16-3
- Anzahl-Anschlüsse, Auto-Indizierung zum
Einstellen, 8-5
- Anzeigeelemente, 4-1
 - 2D, 4-9
 - 3D, 4-9
 - ActiveX, 18-2
 - Anschlüsse (Tabelle), 5-2
 - Anzeigen von optionalen Elementen, 4-2
 - Array, 4-14
 - ausblenden
 - optionale Elemente, 4-2
 - Siehe auch LabVIEW-Hilfe.*
 - ausgeblendet. *Siehe LabVIEW-Hilfe.*
 - boolesch, 4-12
 - verwenden. *Siehe LabVIEW-Hilfe.*
 - Cluster, 4-14
 - Datentypen (Tabelle), 5-2
 - Drucken, 14-4
 - Enum-Typ
 - erweitert, 4-17
 - erforderliche, 7-8
 - ersetzen, 4-2
 - erstellen im Blockdiagramm. *Siehe
LabVIEW-Hilfe.*
 - Farbbox, 4-11
 - Farbrampe, 4-11
 - Gestaltung der Benutzeroberfläche, 4-22
 - Größe ändern, 4-6
 - in Relation zur Fenstergröße, 4-7
 - gruppieren und sperren, 4-6
 - I/O-Name, 4-17
 - klassische, 4-9
 - konfigurieren, 4-1
 - mit geringer Farbanzahl, 4-9
 - mit hoher Farbanzahl, 4-9
 - numerisch, 4-10
 - verwenden. *Siehe LabVIEW-Hilfe.*
 - numerische, 4-11
 - optionale, 7-8
 - Pfad verwenden. *Siehe LabVIEW-Hilfe.*
 - Pfad-Anzeigeelemente, 4-13
 - RefNum, 4-18
 - verwenden. *Siehe LabVIEW-Hilfe.*
 - Registerkarte, 4-14
 - Richtlinien für den Einsatz auf dem
Frontpanel, 4-22
 - rotatorische, 4-10
 - Schieberegler, 4-10
 - String
 - Anzeigearten, 9-2
 - String-Anzeigeelemente, 4-12
 - Überblick, 4-1
 - umwandeln in Bedienelemente, 4-2
 - Zuweisen von Farbe, 4-5
- Anzeigen
 - Anschlüsse, 5-2
 - ausgeblendete Frontpanel-Objekte. *Siehe
LabVIEW-Hilfe.*

- Beschriftung von automatisch erstellten Konstanten. *Siehe LabVIEW-Hilfe.*
- Fehler, 6-2
- Hinweisstreifen. *Siehe LabVIEW-Hilfe.*
- Kette der Aufrufenden, 6-7
- optionale Elemente in Frontpanel-Objekten, 4-2
- Warnungen, 6-3
- Apple-Ereignisse, 17-13
- Application Builder. *Siehe eigenständige Applikationen.*
- Applikationen
 - Erstellen von eigenständigen Applikationen, 7-15
 - Bereitstellen von VIs, 7-15
 - Erstellen von VI-Server-Applikationen, 16-2
- Applikationsinformationen, 1-3
- Applikationsschriftart, 4-21
- Applikationssteuerungsfunktionen, 5-10
- Arbeiten im Netzwerk. *Siehe Kommunikation.*
- Arbeitsbereich, Hinzufügen zum Frontpanel oder zum Blockdiagramm, 4-9
- Arrays, 9-6
 - Auto-Indizierung von Schleifen, 8-4
 - Bedien- und Anzeigeelemente, 4-14
 - Datentyp (Tabelle), 5-3
- Beispiele, 9-7
 - 1D-Arrays, 9-7
 - 2D-Arrays, 9-8
- Beschränkungen, 9-9
- Dimensionen, 9-6
- Einfügen von Elementen. *Siehe LabVIEW-Hilfe.*
- Ersetzen von Elementen. *Siehe LabVIEW-Hilfe.*
- erstellen, 9-9
- Funktionen, 5-8
- globale Variablen, 10-7
- Größe ändern. *Siehe LabVIEW-Hilfe.*
- Größe von, 6-9
- Indizes, 9-7
 - Anzeige, 9-10
 - Konstanten, 9-9
- Löschen von Elementen. *Siehe LabVIEW-Hilfe.*
- Polymorphismus, B-5
- Standarddaten, 6-9
- Überblick, 9-6
- Umwandeln von Clustern in Arrays und umgekehrt. *Siehe LabVIEW-Hilfe.*
- vergleichen, C-2
- verschieben. *Siehe LabVIEW-Hilfe.*
- Aufheben der Gruppierung von Frontpanel-Obj., 4-6
- Aufheben der Sperrung Frontpanel-Obj., 4-6 VIs. *Siehe LabVIEW-Hilfe.*
- Aufrufen von Code textbasierter Programmiersprachen, 19-1
- Aufrufen von VIs über das Netzwerk, 16-1
- Aufrufenden
 - anzeigen, 6-7
 - Kette der, 6-7
- Ausblenden
 - Bildlaufleisten, 4-19
 - Menüleiste, 4-19
 - Objekte auf dem Frontpanel. *Siehe LabVIEW-Hilfe.*
 - optionale Elemente in Frontpanel-Objekten, 4-2
- Ausdrucks-knoten, 20-4
- Ausführbare VIs
 - Fehlersuche. *Siehe LabVIEW-Hilfe.*
- Ausführen von Befehlen auf Systemebene, 17-13
- Ausführen von VIs, 6-1
- Ausführung
 - Fluss, 5-23
 - steuern mit Sequenzstrukturen, 8-10
- Highlight-Funktion, Fehlersuche in VIs, 6-4

Ausführungsfluss, 5-23
Ausführungsgeschwindigkeit
 steuern, 8-7
Ausführungsmodus
 Öffnen von VIs im Ausführungsmodus.
 Siehe LabVIEW-Hilfe.
Ausführungsreihenfolge, 5-23
 steuern mit Sequenzstrukturen, 8-10
Ausgeblendete Frontpanel-Objekte. *Siehe
 LabVIEW-Hilfe.*
Auskommentieren
 Abschnitte eines Blockdiagramms
 Fehlersuche in VIs, 6-7
Ausnahmesteuerung. *Siehe LabVIEW-Hilfe.*
Ausrichten von Objekten. *Siehe
 LabVIEW-Hilfe.*
Außerkräftsetzen der
 Funktionstasteneinstellungen des Systems.
 Siehe LabVIEW-Hilfe.
Aussetzen der Ausführung
 Fehlersuche in VIs, 6-6
Auswählen
 Objekte. *Siehe LabVIEW-Hilfe.*
 Standardinstanz eines polymorphen VIs.
 Siehe LabVIEW-Hilfe.
 Werkzeuge. *Siehe LabVIEW-Hilfe.*
Auto-Indizierung, 8-4
 For-Schleifen, 8-5
 unerwartete Daten, 6-9
 While-Schleifen, 8-6
Automations-Refnum-Objekt, 18-2
Automatische Verbindung, 5-13
Automatische Vervollständigung, 4-15
 Listenfelder, 4-15
 Ring-Bedienelemente, 4-16
Automatisches Anmelden. *Siehe
 LabVIEW-Hilfe.*

B

Bäume
 Variant-Daten, 5-20
Bearbeiten
 Frontpanel-Objekte, 4-1
 Kontextmenüs von polymorphen VIs.
 Siehe LabVIEW-Hilfe.
 Menüs, 15-2
 Palettenansichten, 3-5
Bedienelemente, 4-1
 2D, 4-9
 3D, 4-9
 ActiveX, 18-2
 Anschlüsse (Tabelle), 5-2
 Anzeigen von optionalen Elementen, 4-2
 Array, 4-14
 ausblenden
 Optionale Elemente, 4-2
 Siehe auch LabVIEW-Hilfe.
 ausgeblendet. *Siehe LabVIEW-Hilfe.*
 Automations-Refnum, 18-2
 benennen, 7-10
 boolesch, 4-12
 verwenden. *Siehe LabVIEW-Hilfe.*
 Cluster, 4-14
 Datentypen (Tabelle), 5-2
 Dialogelement, 4-19
 verwenden. *Siehe LabVIEW-Hilfe.*
 Drucken, 14-4
 Enum-Typ, 4-16
 erweitert, 4-17
 verwenden. *Siehe LabVIEW-Hilfe.*
 erforderliche, 7-8
 ersetzen, 4-2
 erstellen im Blockdiagramm. *Siehe
 LabVIEW-Hilfe.*
 Farbbox, 4-11
 Farbrampe, 4-11

- Gestaltung der Benutzeroberfläche, 4-22
- Größe ändern, 4-6
 - in Relation zur Fenstergröße, 4-7
- gruppieren und sperren, 4-6
- hinzufügen von Bibliotheken, 3-4
- I/O-Name, 4-17
- klassische, 4-9
- konfigurieren, 4-1
- Listenfeld, 4-15
 - verwenden. *Siehe LabVIEW-Hilfe.*
- mit geringer Farbanzahl, 4-9
- mit hoher Farbanzahl, 4-9
- numerisch, 4-10
 - verwenden. *Siehe LabVIEW-Hilfe.*
- numerische, 4-11
- optionale, 7-8
- Palette, 3-1
 - anpassen, 3-4
 - navigieren und suchen, 3-2
- Pfad verwenden. *Siehe LabVIEW-Hilfe.*
- Pfad-Bedienelemente, 4-13
- RefNum, 4-18
 - verwenden. *Siehe LabVIEW-Hilfe.*
- Registerkarte, 4-14
- Richtlinien für den Einsatz auf dem Frontpanel, 4-22
- Ring-Bedienelement, 4-16
 - verwenden. *Siehe LabVIEW-Hilfe.*
- rotatorische, 4-10
- Schieberegler, 4-10
- String
 - Anzeigearten, 9-2
 - Tabellen, 9-3
- String-Bedienelemente, 4-12
- Tastenkombination, 4-3
- Überblick, 4-1
- umwandeln in Anzeigeelemente, 4-2
- Untertitel für Sub-VI-Hinweisstreifen. *Siehe LabVIEW-Hilfe.*
- Zuweisen von Farbe, 4-5
- Bedienelemente vom Enum-Typ, 4-16
 - Datentyp (Tabelle), 5-3
 - verwenden. *Siehe LabVIEW-Hilfe.*
- Bedienelemente vom Typ Enum, 4-16
 - erweitert, 4-17
- Bedingungsanschluss, 8-3
- Befehle auf Systemebene, 17-13
- Befehlszeile
 - Starten von VIs. *Siehe LabVIEW-Hilfe.*
- Beispiele, 1-4
 - Arrays, 9-7
 - 1D-Arrays, 9-7
 - 2D-Arrays, 9-8
 - Überblick, 1-4
- Benachrichtigung bei Fehlern. *Siehe LabVIEW-Hilfe.*
- Benennen
 - Bedienelemente, 7-10
 - VIs, 7-13
- Benutzerbibliothek
 - Hinzufügen von VIs und Bedienelementen, 3-4
- Benutzerdefinierte Fehler. *Siehe LabVIEW-Hilfe.*
- Benutzerdefinierte Konstanten, 5-5
- Benutzeroberfläche. *Siehe Frontpanel.*
- Bereitstellen von VIs, 7-14
- Berichte
 - Berichterzeugungs-VIs, 14-7
 - Drucken, 14-7
 - erzeugen
 - Fehler-Cluster. *Siehe LabVIEW-Hilfe.*
- Beschriftung von automatisch erstellten Konstanten anzeigen. *Siehe LabVIEW-Hilfe.*
- Beschriftungen, 4-19
 - anzeigen für automatisch erstellte Konstanten. *Siehe LabVIEW-Hilfe.*
 - globale Variablen, 10-4

- Größe ändern. *Siehe LabVIEW-Hilfe.*
- Konstanten, 5-4
- lokale Variablen, 10-2
- Maßeinheiten, 5-20
- Schriftarten, 4-20
- transparent. *Siehe LabVIEW-Hilfe.*
- Untertitel, 4-20
- Bibliotheken
 - Anwender, A-1
 - Entfernen von VIs aus. *Siehe LabVIEW-Hilfe.*
 - gemeinsame, 7-15
 - Bereitstellen von VIs, 7-15
 - Hinzufügen von VIs und Bedienelementen, 3-4
 - Instrument, A-1
 - konvertieren in Verzeichnisse. *Siehe LabVIEW-Hilfe.*
 - Konvertieren von Verzeichnissen in. *Siehe LabVIEW-Hilfe.*
 - Markieren von VIs als Top-Level. *Siehe LabVIEW-Hilfe.*
 - Organisation der, A-1
 - Speichern von VIs als, 7-12
 - vorgeschlagenes Verzeichnis zum, A-3
 - verwalten, 7-13
 - Verzeichnisstruktur, A-1
 - VI, A-1
- Bildanzeigeelemente und Anzeigeelemente
 - Verwenden, 12-1
- Bild-Bedien- und Anzeigeelemente
 - Datentyp (Tabelle), 5-4
- Bilder. *Siehe Grafiken.*
- Bildlauf durch ein Diagramm, 11-3
- Bildlaufleisten
 - ausblenden, 4-19
 - Listenfelder, 4-15
 - Ring-Bedienelemente, 4-16
- Bildring-Bedienelemente, 4-16
- Bildschirmauflösung, 4-23
- Binär
 - Datei-I/O, 13-3
 - Erstellen von Dateien, 13-11
 - Fließkomma-Arithmetik, 6-8
- Bitmap-Dateien, 12-7
- Blinkgeschwindigkeit. *Siehe LabVIEW-Hilfe.*
- Blockdiagramm, 2-3
 - Anschlüsse, 5-2
 - anzeigen, 5-2
 - Entfernen von Funktionen, 5-11
 - Frontpanel-Objekte und, 5-1
 - hinzufügen zu Funktionen, 5-11
 - von Bedien- und Anzeigeelem. (Tabelle), 5-2
 - Ausrichten von Objekten. *Siehe LabVIEW-Hilfe.*
 - Beschriftungen, 4-19
 - erstellen. *Siehe LabVIEW-Hilfe.*
 - Größe ändern. *Siehe LabVIEW-Hilfe.*
 - DataSocket, 17-8
 - Datenfluss, 5-23
 - Datentypen (Tabelle), 5-2
 - Drucken, 14-6
 - Einfügen von Objekten. *Siehe LabVIEW-Hilfe.*
 - Entfernen von Objekten. *Siehe LabVIEW-Hilfe.*
 - entwerfen, 5-26
 - Ersetzen von Objekten. *Siehe LabVIEW-Hilfe.*
 - Erstellen von Bedien- und Anzeigeelementen. *Siehe LabVIEW-Hilfe.*
 - Funktionen, 5-7
 - Gleichmäßiges Verteilen von Objekten. *Siehe LabVIEW-Hilfe.*
 - Hinzufügen von Leerraum ohne Größenänderung, 5-27
 - Knoten, 5-6
 - Konstanten, 5-4
 - Kontrollieren des Quellcodes, 7-2

Kopieren von Objekten. *Siehe LabVIEW-Hilfe.*
 Löschen von Objekten. *Siehe LabVIEW-Hilfe.*
 Neuordnen von Objekten. *Siehe LabVIEW-Hilfe.*
 Objekte, 5-2
 Optionen, 3-6
 Passwortschutz, 7-14
 planen, 7-1
 Schriftarten, 4-20
 Strukturen, 8-1
 verwenden. *Siehe LabVIEW-Hilfe.*
 Suchen von Anschlüssen. *Siehe LabVIEW-Hilfe.*
 Typumwandlungspunkte, 5-15
 Überblick, 2-3
 Variant-Daten, 5-19
 Verbindung, 5-11
 VI-Server, 16-1
 vorübergehendes Deaktivieren von
 Abschnitten, 6-7
 BMP-Dateien, 12-7
 Boolesche Bedien- und
 Anzeigeelemente, 4-12
 Datentyp (Tabelle), 5-3
 verwenden. *Siehe LabVIEW-Hilfe.*
 boolesche Bedien- und Anzeigeelemente
 Vergleichen von Werten, C-1
 Boolesche Funktionen, 5-8
 Polymorphismus, B-4
 Byte-Stream-Dateien, 13-4

C

CASE-Strukturen, 8-8
 Datentypen, 8-8
 Fehlerbehandlung, 6-12
 Selektoranschlüsse, 8-8
 Werte, 8-8
 verwenden. *Siehe LabVIEW-Hilfe.*

C-Code aufrufen aus LabVIEW, 19-1
 CIN, 19-1
 Clients
 ActiveX, 18-3
 Cloning von Objekten auf dem Frontpanel
 oder im Blockdiagramm. *Siehe
 LabVIEW-Hilfe.*
 Cluster, 9-12
 Bedien- und Anzeigeelemente, 4-14
 Datentyp (Tabelle), 5-3
 Fehler, 6-11
 Berichte. *Siehe LabVIEW-Hilfe.*
 Komponenten, 6-12
 Funktionen, 5-9
 Größe ändern. *Siehe LabVIEW-Hilfe.*
 Polymorphismus, B-6
 Überblick, 9-12
 Umwandeln von Arrays in Cluster und
 umgekehrt. *Siehe LabVIEW-Hilfe.*
 Verbindungsmuster, 9-13
 vergleichen, C-2
 verschieben. *Siehe LabVIEW-Hilfe.*
 Code-Interface-Knoten, 19-1
 computerbasierte Instrumente
 konfigurieren, 1-4
 Container
 ActiveX, 18-2
 Cursor
 Graph, 11-4
 hinzufügen zu Graphen. *Siehe
 LabVIEW-Hilfe.*

D

DAQ
 Kanal-Assistent, 1-4
 Kanal-Monitor, 1-4
 Konfigurationsprogramm, 1-4
 Lösungsassistent, 1-4
 Übergeben von Kanalnamen, 4-17
 VIs und Funktionen, 7-3

- DataSocket, 17-3
 - Blockdiagramm, 17-8
 - Formate für Daten, 17-5
 - Frontpanel, 17-6
 - Protokolle, 17-4
 - Überblick, 17-3
 - URLs, 17-4
 - Variant-Daten, 17-9
- DataSocket-Protokoll dstp, 17-4
- DataSocket-Protokoll file, 17-4
- DataSocket-Protokoll ftp, 17-4
- DataSocket-Protokoll logos, 17-4
- DataSocket-Protokoll OLE for Process Control, 17-4
- DataSocket-Protokoll opc, 17-4
- Datei "serpdrv", A-2
- Datei-I/O, 13-1
 - Arbeiten im Netzwerk und, 17-1
 - Binärdateien, 13-3
 - Erstellen, 13-11
 - Byte-Stream-Dateien, 13-4
 - Datenprotokolldateien, 13-4
 - Erstellen, 13-11
 - Datenträger-Streaming, 13-8
 - Durchfluss-Parameter, 13-14
 - Formate, 13-2
 - Fortgeschrittene Dateifunktionen, 13-6
 - Funktionen, 5-9
 - Grundlegende Funktionen, 13-1
 - High-Level-VIs, 13-5
 - Konfigurationsdatei-VIs
 - Format, 13-15
 - Lesen und Schreiben von .ini-Dateien, 13-14
 - Zweck, 13-15
 - Lesen von Signalverläufen, 13-13
 - Low-Level-VIs und -Funktionen, 13-6
 - Pfade, 13-7
 - Protokollieren von Frontpanel-Daten, 13-17
 - Refnums, 13-1
 - Schreiben von Signalverläufen, 13-12
 - Spreadsheet-Dateien-erstellen, 13-9
 - Textdateien, 13-2
 - Erstellen, 13-9
 - Überblick, 13-1
- Datenabhängigkeit, 5-23
 - Ablaufunterschiede, 10-5
 - Durchfluss-Parameter, 13-14
 - fehlende, 5-24
 - künstlich, 5-24
 - steuern mit Sequenzstrukturen, 8-10
- Daten-Bubbles
 - anzeigen während Highlight-Funktion. *Siehe LabVIEW-Hilfe.*
- Datenerfassung. *Siehe DAQ.*
- Datenfluss beobachten, 6-4
- Datenfluss-Programmierungsmodell, 5-23
 - Verwalten von Speicher, 5-25
- Datenprotokolldatei-I/O, 13-4
 - Erstellen von Dateien, 13-11
- Datenprotokollierung, 13-17
 - ändern der Protokolldateibindung, 13-20
 - Automatisch, 13-18
 - Interaktiv, 13-18
 - löschen der Protokolldateibindung, 13-19
 - Löschen von Datensätzen, 13-19
 - Programmatischer Abruf von Daten, 13-20
- Datensätze, 13-18
 - Festlegen während des Abrufens von Frontpanel-Daten mit Sub-VIs, 13-21
 - Löschen, 13-19
- Datenträger-Streaming, 13-8
- Datentypen
 - CASE-Strukturen, 8-8
 - Drucken, 14-4
 - HiQ (Tabelle), 20-6
 - MATLAB (Tabelle), 20-6
 - Signalverlauf, 11-18
 - von Bedien- und Anzeigeelem. (Tabelle), 5-2

- Datumsformat
 - Optionen, 3-6
- DDE, 17-13
- Deaktivieren
 - Fehlersuche-Werkzeuge, 6-8
- Debuggen
 - ausgeblendete Verbindungen, 5-26
- Definieren von Fehlern. *Siehe LabVIEW-Hilfe.*
- Dezimalpunkt, lokaler. *Siehe LabVIEW-Hilfe.*
- Diagramme, 11-1
 - Anpassen der Darstellung, 11-3
 - Anpassen des Verhaltens, 11-6
 - Bildlauf, 11-3
 - erstellen. *Siehe LabVIEW-Hilfe.*
 - Hinzufügen von Kurven. *Siehe LabVIEW-Hilfe.*
 - Historienlänge, 11-6
 - Intensität, 11-12
 - Optionen, 11-14
 - Kantengeglättete
 - Liniendarstellungen, 11-2
 - Mehrere Achsen, 11-2
 - Optionen, 11-2
 - Signalverlauf, 11-11
 - Stapelplots, 11-7
 - Typen, 11-1
 - Überblick, 11-1
 - Überlagerte Kurven, 11-7
 - zoomen. *Siehe LabVIEW-Hilfe.*
- Dialogfelder
 - Bedienelemente, 4-19
 - verwenden. *Siehe LabVIEW-Hilfe.*
 - für systemeigene Dateien. *Siehe LabVIEW-Hilfe.*
 - gestalten, 4-23
 - Ring-Bedienelemente, 4-16
 - Schriftart, 4-21
- Dialogfelder für systemeigene Dateien. *Siehe LabVIEW-Hilfe.*
- Dialogfunktionen, 5-9
- Digitale Graphen, 11-15
 - Maskieren von Daten, 11-17, D-1
- Dimensionen
 - Arrays, 9-6
- Direktklick. *Siehe LabVIEW-Hilfe.*
- DLLs
 - aufrufen aus LabVIEW, 19-1
 - erstellen, 7-15
 - Bereitstellen von VIs, 7-15
- Dokumentation
 - Einführung zu diesem Handbuch, *xvii*
 - Gliederung dieses Handbuchs, *xviii*
 - Handbuch, 1-1
 - PDF-Bibliothek, 1-3
 - Schreibkonventionen in diesem Handbuch, *xviii*
 - verwenden mit anderen Ressourcen, 1-1
 - Verwendung dieses Handbuchs, *xvii*
 - Verzeichnisstruktur, A-2
- Dokumentieren von VIs, 14-2
 - Drucken, 14-4
 - Erstellen von Hinweisstreifen, 14-2
 - Erstellen von Objekt- und VI-Beschreibungen, 14-2
 - Hilfdateien, 14-5
 - programmatisch, 14-4
 - Revisions-Historie, 14-2
 - Verknüpfen mit erstellten Hilfdateien. *Siehe LabVIEW-Hilfe.*
- Drag and Drop. *Siehe LabVIEW-Hilfe.*
- Drehknöpfe
 - Frontpanel, 4-10
 - Hinzufügen von Farbrampen, 4-12
- Drehregler
 - Frontpanel, 4-10
 - Hinzufügen von Farbrampen, 4-12
- Drehspulinstrumente
 - Frontpanel, 4-10
 - Hinzufügen von Farbrampen, 4-12

Drucken, 14-6

- Aktives Fenster, 14-6
- Berichte, 14-7
- Dokumentation von VIs, 14-4
 - mit Sub-VIs, 14-8
 - nach Ausführung, 14-8
- Optionen, 3-6
- programmatisch, 14-7
- Speichern
 - als HTML, 14-4
 - als RTF, 14-4
- Verfahren, 14-8

Drucken von PostScript. *Siehe LabVIEW-Hilfe.*

Durchfluss-Parameter, 13-14

Dynamischer Datenaustausch, 17-13

E

Eigenschaften

- ActiveX, 18-1

Eigenschaftsknoten, 16-4

- ActiveX, 18-2
- Ändern von Listenfeldeinträgen, 4-15
- Suchen von Objekten oder Anschlüssen.
Siehe LabVIEW-Hilfe.

Eigenständige Applikationen

- erstellen, 7-15
 - Bereitstellen von VIs, 7-15

Einbetten von Objekten mit Hilfe von

- ActiveX, 18-4

Einfügen

- Array-Elemente. *Siehe LabVIEW-Hilfe.*
- Objekte im Blockdiagramm. *Siehe LabVIEW-Hilfe.*
- Objekte in Paletten. *Siehe LabVIEW-Hilfe.*

Einheitenbeschriftungen, 5-20

Einzelstapelausführung

- Fehlersuche in VIs, 6-5

Endlose While-Schleifen, 8-4

Entfernen

- Anschlüsse von Funktionen, 5-11
- Objekte auf dem Frontpanel oder im Blockdiagramm. *Siehe LabVIEW-Hilfe.*
- Strukturen. *Siehe LabVIEW-Hilfe.*
- Sub-VIs aus polymorphen VIs. *Siehe LabVIEW-Hilfe.*
- unterbrochene Verbindungen, 5-14
- VIs aus Bibliotheken. *Siehe LabVIEW-Hilfe.*

Entwerfen

- Blockdiagramm, 5-26
- Projekte, 7-1
- Sub-VIs, 7-10

Entwickeln

- Instrumententreiberapplikationen. *Siehe LabVIEW-Hilfe.*

Entwickeln von VIs, 7-1

- Richtlinien, 1-3
- Verfolgen der Entwicklung. *Siehe Dokumentieren von VIs.*

Ereignisse

- ActiveX, 18-2

Ersetzen

- Array-Elemente. *Siehe LabVIEW-Hilfe.*
- Objekte auf dem Frontpanel, 4-2
- Objekte im Blockdiagramm. *Siehe LabVIEW-Hilfe.*
- Text in Strings. *Siehe LabVIEW-Hilfe.*

Erste Schritte, 1-1

Erstellen

- Arrays, 9-9
- benutzerdefinierte Konstanten, 5-5
- Binärdateien, 13-11
- Blockdiagramm, 5-1
- Datenprotokolldateien, 13-11
- Diagramme. *Siehe LabVIEW-Hilfe.*
- eigenständige Applikationen, 7-15
 - Bereitstellen von VIs, 7-15
- Frontpanel, 4-1

gemeinsam genutzte Bibliotheken, 7-15
 Bereitstellen von VIs, 7-15
 Graphen. *Siehe LabVIEW-Hilfe.*
 Hinweisstreifen, 14-2
 Menüs, 15-3
 Objektbeschreibungen, 14-2
 Objekt-Refnums. *Siehe LabVIEW-Hilfe.*
 Palettenansichten, 3-5
 polymorphe VIs, 5-16
 Revisions-Historie, 14-2
 Spreadsheet-Dateien, 13-9
 Sub-VIs, 7-5, 7-10
 Situationen, die vermieden werden
 sollten. *Siehe LabVIEW-Hilfe.*
 Symbole, 7-9
 Textdateien, 13-9
 Unterpalletten. *Siehe LabVIEW-Hilfe.*
 VI-Beschreibungen, 14-2
 VIs, 7-1
 VI-Server-Applikationen, 16-2
 Erstellen von Berichten, 14-7
 Erweiterungswerkzeugsätze, 1-1
 in Paletten, 3-6
 Erzeugen von Berichten
 Fehler-Cluster. *Siehe LabVIEW-Hilfe.*

F

Farbe

Ändern in Grafiken, 12-6
 Bedien- und Anzeigeelemente mit
 geringer Farbanzahl, 4-9
 Bedien- und Anzeigeelemente mit hoher
 Farbanzahl, 4-9
 Erstellen in Grafiken, 12-6
 Farbboxen, 4-11
 Farbrampen, 4-11
 Rotatorische Bedienelemente und
 Anzeigen, 4-12
 Farbwahltabelle, 4-11

Optionen, 3-6
 Zuordnen, 11-13
 Farbzuzuordnung, 11-13
 Fehler
 abbrechen bei vorhandenen. *Siehe
 LabVIEW-Hilfe.*
 anzeigen, 6-2
 Ausnahmesteuerung. *Siehe
 LabVIEW-Hilfe.*
 Behandlung, 6-10
 Methoden, 6-11
 Verwenden von
 CASE-Strukturen, 6-12
 Verwenden von
 While-Schleifen, 6-12
 Benachrichtigung. *Siehe LabVIEW-Hilfe.*
 Cluster, 6-11
 Anschlussfeld, 7-8
 Berichte. *Siehe LabVIEW-Hilfe.*
 Komponenten, 6-12
 Codes, 6-11
 definieren. *Siehe LabVIEW-Hilfe.*
 Fenster, 6-2
 I/O, 6-11
 inkompatible Einheiten, 5-21
 Instrumentensteuerung. *Siehe
 LabVIEW-Hilfe.*
 Liste, 6-2
 nicht ausführbare VIs, 6-2
 Normalbedingungen als. *Siehe
 LabVIEW-Hilfe.*
 prüfen auf, 6-10
 suchen, 6-2
 Techniken für die Fehlersuche, 6-4
 Fehlersuche
 ausführbare VIs. *Siehe LabVIEW-Hilfe.*
 Deaktivieren der
 Fehlersuche-Werkzeuge, 6-8
 HiQ- und MATLAB-Skripts, 20-7
 nicht ausführbare VIs, 6-2
 Optionen, 3-6

- Schleifen, 6-9
- Standarddaten, 6-9
- Strukturen. *Siehe LabVIEW-Hilfe.*
- Techniken, 6-4
 - Aussetzen der Ausführung, 6-6
 - Einzelschrittausführung, 6-5
 - Fehlerbehandlung, 6-10
 - Haltepunkt-Werkzeug, 6-6
 - Highlight-Funktion, 6-4
 - Probe-Daten-Werkzeug, 6-5
 - vorübergehendes Deaktivieren von Abschnitten, 6-7
- undefinierte Daten, 6-8
- Werkzeuge
 - deaktivieren, 6-8
- Fenstergröße definieren. *Siehe LabVIEW-Hilfe.*
- Fenstertitel in Funktionspalette. *Siehe LabVIEW-Hilfe.*
- Festlegen
 - Kooperationsebene. *Siehe LabVIEW-Hilfe.*
 - Optionen für die Arbeitsumgebung, 3-6
 - Siehe auch LabVIEW-Hilfe.*
- Festplattenspeicher
 - Optionen, 3-6
 - prüfen. *Siehe LabVIEW-Hilfe.*
- FIFO. *Siehe First-In-First-Out.*
- First-In-First-Out
 - Variant-Daten, 5-20
- Fließkommazahlen
 - Konvertieren, B-1
 - positiver und negativer Überlauf, 6-8
- Format String-Parameter, 9-5
- Formate für Datei-I/O, 13-2
 - Binärdateien, 13-3
 - Datenprotokolldateien, 13-4
 - Textdateien, 13-2
- Formatieren
 - Graphen-Achsen, 11-5
- Strings, 9-4
 - Bezeichner, 9-5
 - Text auf dem Frontpanel, 4-20
- Formelknoten, 20-2
 - Abbildung, 20-3
 - Eingeben von C-ähnlichen Anweisungen, 20-2
 - Eingeben von Gleichungen, 20-2
 - Überblick, 20-2
 - Variablen, 20-3
- Formeln. *Siehe Gleichungen.*
- For-Schleifen, 8-2
 - Auto-Indizierung zum Einstellen der Anzahl, 8-5
 - Iterationsanschlüsse, 8-2
 - Schieberegister, 8-6
 - Steuern des Timings, 8-7
 - unerwartete Daten, 6-9
 - verwenden. *Siehe LabVIEW-Hilfe.*
 - Zähl-Anschlüsse, 8-2
- Fortgeschrittene Funktionen, 5-10
- Freie Beschriftungen, 4-20
 - erstellen. *Siehe LabVIEW-Hilfe.*
- Freigeben von Speicher. *Siehe LabVIEW-Hilfe.*
- Frontpanel, 2-2
 - Abrufen von Daten
 - mit Datei I/O-Funktionen, 13-22
 - mit Sub-VIs, 13-20
 - Ändern der Größe von Objekten, 4-6
 - in Relation zur Fenstergröße, 4-7
 - Anzeigeelemente, 4-9
 - anzeigen mit unterschiedlichen Bildschirmauflösungen, 4-23
 - Anzeigen von optionalen Objektelementen, 4-2
 - ausblenden
 - Objekte. *Siehe LabVIEW-Hilfe.*
 - optionale Objektelemente, 4-2
 - ausgeblendete Objekte. *Siehe LabVIEW-Hilfe.*

- Ausrichten von Objekten. *Siehe LabVIEW-Hilfe.*
- Bedienelemente, 4-9
- Beschriftungen, 4-19
 - erstellen. *Siehe LabVIEW-Hilfe.*
 - Größe ändern. *Siehe LabVIEW-Hilfe.*
- DataSocket, 17-6
- Datenprotokollierung, 13-17
- Definieren der Fenstergröße. *Siehe LabVIEW-Hilfe.*
- Drucken, 14-6
- Einfügen von Objekten mit Hilfe von ActiveX, 18-4
- Entfernen von Objekten. *Siehe LabVIEW-Hilfe.*
- Ersetzen von Objekten, 4-2
- Festlegen der Navigationsreihenf., 4-4 gestalten, 4-22
- Gleichmäßiges Verteilen von Objekten. *Siehe LabVIEW-Hilfe.*
- Gruppieren und Sperren von Obj., 4-6
- Hinzufügen von Leerraum ohne Größenänderung, 4-9
- Importieren von Grafiken, 4-5
- Konfigurieren von Objekten, 4-1
- Kopieren von Objekten. *Siehe LabVIEW-Hilfe.*
- Löschen von Objekten. *Siehe LabVIEW-Hilfe.*
- Neuanordnen von Objekten. *Siehe LabVIEW-Hilfe.*
- Objekte, Blockdiagrammanschlüsse und, 5-1
- Objekten Farbe zuweisen, 4-5
 - Kopieren von Farben. *Siehe LabVIEW-Hilfe.*
 - Vorder- und Hintergrund. *Siehe LabVIEW-Hilfe.*
- Optionen, 3-6
- planen, 7-1
- Programmatische Steuerung von Objekten, 16-9
- Protokollieren von Daten, 13-17
- Reihenfolge der Objekte, 4-4
- Schriftarten, 4-20
- Skalieren von Objekten, 4-7
- Sub-VIs, 7-10
- Suchen von Objekten. *Siehe LabVIEW-Hilfe.*
- Tastenkombination, 4-3
- Texteigenschaften, 4-20
- transparente Objekte. *Siehe LabVIEW-Hilfe.*
- Überblick, 2-2
- überlappende Objekte, 4-14
- Umwandeln von Bedienelementen in Anzeigeelemente und umgekehrt, 4-2
- Untertitel, 4-20
 - erstellen. *Siehe LabVIEW-Hilfe.*
- Veröffentlichen von Bildern im Web, 17-11
- Funktion
 - Highlight
 - Anzeigen von Daten-Blasen. *Siehe LabVIEW-Hilfe.*
 - Probe-Daten. *Siehe LabVIEW-Hilfe.*
- Funktion "Aufruf externer Bibliotheken", 19-1
- Funktionen, 5-7
 - Applikationssteuerung, 5-10
 - Array, 5-8
 - Blockdiagramm, 5-7
 - boolesch, 5-8
 - Cluster, 5-9
 - Datei-I/O, 5-9
 - Dialog, 5-9
 - Entfernen von Anschlüssen, 5-11
 - fortgeschritten, 5-10
 - Hinzufügen von Anschlüssen, 5-11
 - numerisch, 5-7
 - Palette, 3-1

- anpassen, 3-4
- Fenstertitel. *Siehe LabVIEW-Hilfe.*
- navigieren und suchen, 3-2
- Polymorph, B-1
- Referenz. *Siehe LabVIEW-Hilfe.*
- Signalverlauf, 5-10
- String, 5-8
- suchen. *Siehe LabVIEW-Hilfe.*
- Zeit, 5-9
- Funktionstasteneinstellungen
 - Systemeinstellungen außer Kraft setzen.
Siehe LabVIEW-Hilfe.

G

- Ganzzahlen
 - Konvertieren, B-1
 - positiver und negativer Überlauf, 6-8
- Gemeinsam genutzte Bibliotheken
 - aufrufen aus LabVIEW, 19-1
 - erstellen, 7-15
 - Bereitstellen von VIs, 7-15
- Gemeinsam nutzen
 - Dateien, 7-2
 - Direktübertragungsdaten mit anderen VIs und Applikationen, 17-3
 - Palettenansichten. *Siehe LabVIEW-Hilfe.*
 - VIs, 7-14
- Gemeinsame Nutzung von Dateien, 7-2
- Gepufferte Daten
 - lokale Variablen, 10-6
- Gestalten
 - Dialogfelder, 4-23
 - Frontpanel, 4-22
- GIF-Dateien, 14-5
- Gleichmäßiges Verteilen von Objekten. *Siehe LabVIEW-Hilfe.*
- Gleichungen
 - Ausdrucks-knoten, 20-4
 - Formelknoten, 20-2

- HiQ
 - ActiveX, 20-2
 - Fehlersuche in Skripten, 20-7
 - Skript-Knoten, 20-5
 - VIs, 20-5
- Integration in LabVIEW, 20-1
- MATLAB
 - ActiveX, 20-2
 - Fehlersuche in Skripten, 20-7
 - Skript-Knoten, 20-5
 - Methoden zur Nutzung, 20-1
- Globale Variablen, 10-2
 - Ablaufunterschiede, 10-5
 - erstellen, 10-3
 - initialisieren, 10-5
 - lesen und schreiben, 10-4
 - Speicher, 10-7
 - Überblick, 10-2
 - umsichtig verwenden, 10-5
- Globale Variablen mit Lesezugriff, 10-4
- Globale Variablen mit Schreibzugriff, 10-4
- GPIB konfigurieren, 1-4
- Grafik glätten
 - bei Graphen, 11-6
 - beim Zeichnen. *Siehe LabVIEW-Hilfe.*
- Grafiken
 - Ändern von Farben, 12-6
 - Bildanzeigeelemente und Anzeigeelemente
 - Verwenden, 12-1
 - Bild-Bedien- und Anzeigeelemente
 - Datentyp (Tabelle), 5-4
 - drag and drop. *Siehe LabVIEW-Hilfe.*
 - Eingabe von Text, 12-5
 - Erstellen von Farben, 12-6
 - Formate, 12-7
 - für HTML-Dateien, 14-4
 - Graphen, 12-3
 - Hinzufügen zu VI-Symbol. *Siehe LabVIEW-Hilfe.*
 - importieren, 4-5

Pixmaps, 12-5
 Veröffentlichen von Frontpanels
 im Web, 17-11
 Zeichnen von Formen, 12-5
 Graphen, 11-1
 3D, 11-17
 Achsen-Formatieren, 11-5
 Anpassen der Darstellung, 11-3
 Anpassen des Verhaltens, 11-3
 Cursor, 11-4
 hinzufügen. *Siehe LabVIEW-Hilfe.*
 digital, 11-15
 Maskieren von Daten, 11-17
 erstellen. *Siehe LabVIEW-Hilfe.*
 Grafik glätten, 11-6
 Grafiken, 12-3
 Hinzufügen von Kurven. *Siehe
 LabVIEW-Hilfe.*
 Intensität, 11-12
 Optionen, 11-14
 Kantengeglättete
 Liniendarstellungen, 11-2
 Mehrere Achsen, 11-2
 Optionen, 11-2
 Polar, 12-3
 Signalverlauf, 11-8
 Datentypen, 11-8, 11-9
 Skalieren, 11-5
 Smith-Diagramme, 12-4
 Typen, 11-1
 Überblick, 11-1
 Übertragungsleitungen, 12-4
 XY, 11-8
 Datentypen, 11-10
 zoomen. *Siehe LabVIEW-Hilfe.*
 Größe ändern
 Arrays. *Siehe LabVIEW-Hilfe.*
 benutzerdefinierte Konstanten, 5-6
 Beschriftungen. *Siehe LabVIEW-Hilfe.*
 Cluster. *Siehe LabVIEW-Hilfe.*

Frontpanel-Objekte, 4-6
 in Relation zur Fenstergröße, 4-7
 Tabellen. *Siehe LabVIEW-Hilfe.*
 Größe anpassen. *Siehe* Größe ändern.
 Gruppieren
 Daten
 Arrays, 9-6
 Cluster, 9-12
 Strings, 9-1
 Frontpanel-Objekte, 4-6
 VIs in Bibliotheken, 7-12

H

Haltepunkt-Werkzeug
 Fehlersuche in VIs, 6-6
 Hervorheben von Haltepunkten. *Siehe
 LabVIEW-Hilfe.*
 Handbuch. *Siehe* Dokumentation.
 Hierarchiefenster, 7-11
 Drucken, 14-4
 suchen. *Siehe LabVIEW-Hilfe.*
 Highlight-Funktion
 Fehlersuche in VIs, 6-4
 Hilfedateien, 1-2
 Erstellen eigener, 14-5
 HTML, 14-5
 RTF, 14-5
 Überblick, 1-2
 Verknüpfen mit VIs. *Siehe
 LabVIEW-Hilfe.*
 Hintergrundfarbe von Frontpanel-Objekten.
Siehe LabVIEW-Hilfe.
 Hinweisstreifen
 anzeigen an Anschlüssen. *Siehe
 LabVIEW-Hilfe.*
 anzeigen. *Siehe LabVIEW-Hilfe.*
 Bedienelement-Hinweisstreifen. *Siehe
 LabVIEW-Hilfe.*
 Erstellen, 14-2

Hinzufügen

- Anschlüsse zu Funktionen, 5-11
- Bedienelemente zu Bibliotheken, 3-4
- Grafik zu VI-Symbol. *Siehe LabVIEW-Hilfe.*
- Verzeichnisse zum VI-Suchpfad. *Siehe LabVIEW-Hilfe.*
- VI's zu Bibliotheken, 3-4

HiQ

- ActiveX, 20-2
- Datentypen, 20-6
- Fehlersuche in Skripts, 20-7
- Skript-Knoten, 20-5
- Starten aus LabVIEW, 20-5
- Supportdateien für LabVIEW-Applikationen, 20-7
- VI's, 20-5

Historie

- Diagramme, 11-6
- Optionen, 3-6
- Siehe auch* Revisions-Historie.

Hot-Menüs. *Siehe LabVIEW-Hilfe.*

HTML

- Erstellen von Dokumenten, 17-11
- Grafikformate, 14-4
- Hilfdateien, 14-5
- Speichern als, 14-4
- Veröffentlichen von VI's im Web, 17-10

I

I/O

- Bedien- und Anzeigeelemente
Datentyp (Tabelle), 5-4
- Datei. *Siehe* Datei-I/O.
- Fehler, 6-11
- I/O-Namen-Bedien- und Anzeigeelemente, 4-17

Impedanz von Übertragungsleitungen, 12-4

Importieren von Grafiken, 4-5

In geglättete Daten umgewandelte Daten Variant-Daten und, 5-20

Indizes von Arrays, 9-7

Anzeige, 9-10

Indizieren von Schleifen, 8-4

For-Schleifen, 8-5

While-Schleifen, 8-6

Inf (Infinity, unendlich), Fließkommawert undefinierte Daten, 6-8

ini-Dateien, Lesen und Schreiben, 13-14

Inkrementelles Ausführen von VI's, 6-5

Installationsprogramme erstellen, 7-15

Instanzen von Sub-VI's

- Aussetzen der Ausführung, 6-6
- ermitteln, 6-7

Instrumente

konfigurieren, 1-4

steuern, 7-3

Instrumentenbibliothek

- Hinzufügen von VI's und Bedienelementen, 3-4

Instrumententreiber. *Siehe LabVIEW-Hilfe.*

Intelligente Puffer

Variant-Daten, 5-20

Intensitätsdiagramme, 11-12

Farbzuordnung, 11-13

Optionen, 11-14

Intensitätsgraphen, 11-12

Farbzuordnung, 11-13

Optionen, 11-14

Iterationsanschlüsse

For-Schleifen, 8-2

While-Schleifen, 8-3

IVI

- Instrumententreiber. *Siehe LabVIEW-Hilfe.*

Übergeben von logischen Namen, 4-17

J

Joint Photographic Experts
Group-Dateien, 12-7
JPEG-Dateien, 12-7, 14-5
Websserver, 17-12

K

Kantengeglättete Liniendarstellungen, 11-2
Kein Pfad, 4-13
Kette aus Aufrufenden
 anzeigen, 6-7
Klassische Bedien- und Anzeigeelemente, 4-9
Knoten, 2-4
 Aufruf über Referenz, 16-8
 Ausführungsfluss, 5-23
 Blockdiagramm, 5-6
 Eigenschaft, 16-4
 HiQ-Skript-Knoten, 20-5
 MATLAB-Skript-Knoten, 20-5
 Methode, 16-5
 Überblick, 2-4
Kommunikation, 17-1
 ActiveX, 18-1
 Apple-Ereignisse, 17-13
 Ausführen von Befehlen auf
 Systemebene, 17-13
 DataSocket, 17-3
 Datei-I/O, 13-1
 DDE, 17-13
 Funktionen, 7-4
 Low-Level, 17-12
 Macintosh, 17-13
 Pipes, 17-13
 PPC, 17-13
 Protokolle, 17-12
 System Exec.vi, 17-13
 TCP, 17-12
 Überblick, 17-1
 UDP, 17-12

UNIX, 17-13

VIs, 7-4

VI-Server, 16-1

Komprimieren des Speichers. *Siehe
LabVIEW-Hilfe.*

Konfigurationsdatei-VIs

Format, 13-15

Lesen und Schreiben von
.ini-Dateien, 13-14

Zweck, 13-15

Konfigurieren

Erscheinungsbild und Verhalten
von VIs, 15-1

Frontpanel-Objekte, 4-1

Menüs, 15-2

Konstanten, 5-4, 5-5

Arrays, 9-9

benutzerdefiniert, 5-5

erstellen. *Siehe LabVIEW-Hilfe.*

Festlegen von Parametern mit
ActiveX, 18-5

Überblick, 5-4

Kontextmenüs, 3-3

im Ausführungsmodus, 3-3

Kontinuierliches Ausführen von VIs, 6-1

Kontrollieren

Quellcode, 7-2

Konvertieren

Bibliotheken in Verzeichnisse. *Siehe
LabVIEW-Hilfe.*

Konvertieren von Arrays in Cluster und
umgekehrt. *Siehe LabVIEW-Hilfe.*

Verzeichnisse in Bibliotheken. *Siehe
LabVIEW-Hilfe.*

Zahlen in Strings, 9-5

Kooperationsebene festlegen. *Siehe
LabVIEW-Hilfe.*

Kopieren

Objekte auf dem Frontpanel oder im
Blockdiagramm. *Siehe
LabVIEW-Hilfe.*

- kopieren
 - VIs, A-3
- Korrigieren
 - VIs, 6-2
 - Techniken für die Fehlersuche, 6-4
- Künstliche Datenabhängigkeit, 5-24
- Kurven
 - gestapelte, 11-7
 - hinzufügen zu Graphen und Diagrammen.
Siehe LabVIEW-Hilfe.
 - kantengeglättet, 11-2
 - überlagerte, 11-7
- L**
- LabVIEW, 1-1
 - anpassen, 3-6
 - Optionen, 3-6
 - Überblick, 1-1
- labview.ini, 3-7
- Last-In-First-Out
 - Variant-Daten, 5-20
- Laufdiagramm, 11-7
- Leere Pfade, 4-13
- Leistung
 - Deaktivieren der Fehlersuche-Werkzeuge, 6-8
 - lokale und globale Variablen, 10-5
 - Optionen, 3-6
- Lesen aus Dateien, 13-1
- Leuchten auf dem Frontpanel, 4-12
- LIFO
 - Variant-Daten, 5-20
- Liniendarstellungen
 - kantengeglättet, 11-2
- Liste. *Siehe Anzeigen.*
- Listenfeld-Bedienelemente, 4-15
 - verwenden. *Siehe LabVIEW-Hilfe.*
- Logarithmische Funktionen
 - Polymorphismus, B-7
- Lokale Sequenzanschlüsse, 8-11
- Lokale Variablen, 10-1
 - Ablaufunterschiede, 10-5
 - erstellen, 10-2
 - initialisieren, 10-5
 - lesen und schreiben, 10-4
 - Speicher, 10-6
 - Suchen von Objekten oder Anschlüssen.
Siehe LabVIEW-Hilfe.
 - Überblick, 10-1
 - umsichtig verwenden, 10-5
- Lokale Variablen mit Lesezugriff, 10-4
- Lokale Variablen mit Schreibzugriff, 10-4
- Lokalen Dezimalpunkt verwenden. *Siehe LabVIEW-Hilfe.*
- Löschen
 - Array-Elemente. *Siehe LabVIEW-Hilfe.*
 - Datenprotokolldatensätze, 13-19
 - Objekte auf dem Frontpanel oder im Blockdiagramm. *Siehe LabVIEW-Hilfe.*
 - Palettenansichten. *Siehe LabVIEW-Hilfe.*
 - Strukturen. *Siehe LabVIEW-Hilfe.*
 - unterbrochene Verbindungen, 5-14
- Löschen von
 - Datenprotokolldatensätzen, 13-19
- Low-Level-Kommunikation, 17-12
- M**
- Markieren
 - Verbindungen, 5-14
- Markieren von VIs als Top-Level in Bibliotheken. *Siehe LabVIEW-Hilfe.*
- Maskieren digitaler Daten, 11-17, D-1
- Maßeinheiten, 5-20
- Mathematik. *Siehe Gleichungen.*
- MATLAB
 - ActiveX, 20-2
 - Datentypen, 20-6
 - Fehlersuche in Skripts, 20-7
 - Skript-Knoten, 20-5

Measurement & Automation Explorer, 1-4
 Mehrere Threads ausführen. *Siehe LabVIEW-Hilfe.*
 Menü-Editor, 15-3
 Menüleiste ausblenden, 4-19
 Menüs, 3-3
 Bearbeiten, 15-2
 Hot-Menüs. *Siehe LabVIEW-Hilfe.*
 Kontextmenüs, 3-3
 bearbeiten für polymorphe VIs. *Siehe LabVIEW-Hilfe.*
 Referenz. *Siehe LabVIEW-Hilfe.*
 Ring-Bedienelemente, 4-16
 Verarbeiten der Auswahl, 15-3
 Methoden
 ActiveX, 18-1
 Methodenknoten, 16-5
 ActiveX, 18-2
 Muster
 Anschluss, 7-8

N

NaN (Not a Number, keine Zahl),
 Fließkommawert
 undefinierte Daten, 6-8
 National Instruments Support im Internet, E-1
 Navigationsreihenf. festlegen, 4-4
 negativer Zahlenüberlauf, 6-8
 Neuordnen von Objekten. *Siehe LabVIEW-Hilfe.*
 NI Developer Zone, E-1
 Nicht ausführbare VIs
 Anzeigen von Fehlern, 6-2
 häufige Ursachen, 6-3
 korrigieren, 6-2
 NI-DAQ-Konfigurationsprogramm, 1-4
 Normalbedingungen als Fehler. *Siehe LabVIEW-Hilfe.*

Not a Number (NaN, keine Zahl),
 Fließkommawert
 undefinierte Daten, 6-8
 Numerische Bedienelemente und
 Anzeigen, 4-11
 Numerische Werte
 Bedien- und Anzeigeelemente, 4-10
 verwenden. *Siehe LabVIEW-Hilfe.*

O

Objekte
 ActiveX, 18-1
 Anzeigen von optionalen Elementen, 4-2
 ausblenden auf dem Frontpanel
 Siehe auch LabVIEW-Hilfe.
 optionale Elemente, 4-2
 ausgeblendet auf dem Frontpanel. *Siehe LabVIEW-Hilfe.*
 ausrichten. *Siehe LabVIEW-Hilfe.*
 auswählen. *Siehe LabVIEW-Hilfe.*
 beschriften, 4-19
 Beschriftungen
 erstellen. *Siehe LabVIEW-Hilfe.*
 Größe ändern. *Siehe LabVIEW-Hilfe.*
 Blockdiagramm, 5-2
 einfügen im Blockdiagramm. *Siehe LabVIEW-Hilfe.*
 Einfügen im Frontpanel mit Hilfe von
 ActiveX, 18-4
 ersetzen im Blockdiagramm. *Siehe LabVIEW-Hilfe.*
 ersetzen, auf dem Frontpanel, 4-2
 Erstellen von Beschreibungen, 14-2
 Erstellen von Hinweistreifen, 14-2
 Farbe zuweisen auf dem Frontpanel, 4-5
 Kopieren von Farben. *Siehe LabVIEW-Hilfe.*
 Festlegen der Navigationsreihenf.
 Frontpanel, 4-4

Frontpanel und
 Blockdiagrammanschlüsse, 5-1
 gleichmäßig verteilen. *Siehe LabVIEW-Hilfe.*
Größe ändern auf dem Frontpanel, 4-6
 in Relation zur Fenstergröße, 4-7
Gruppieren und Sperren auf dem Frontpanel, 4-6
in Paletten einfügen. *Siehe LabVIEW-Hilfe.*
konfigurieren auf dem Frontpanel, 4-1
neu anordnen. *Siehe LabVIEW-Hilfe.*
Programmatische Steuerung, 16-9
skalieren auf dem Frontpanel, 4-7
suchen. *Siehe LabVIEW-Hilfe.*
transparent. *Siehe LabVIEW-Hilfe.*
überlappend auf dem Frontpanel, 4-14
Umwandeln von Bedienelementen in Anzeigeelemente und umgekehrt, 4-2
Untertitel auf dem Frontpanel, 4-20
 erstellen. *Siehe LabVIEW-Hilfe.*
verbinden im Blockdiagramm, 5-11
verschieben. *Siehe LabVIEW-Hilfe.*
Objekt-Refnums, 16-9
 erstellen. *Siehe LabVIEW-Hilfe.*
 schwach typisierte, 16-10
 strikt typisierte, 16-9
Öffnen von VIs im Ausführungsmodus. *Siehe LabVIEW-Hilfe.*
Optionen
 festlegen, 3-6
 Siehe auch LabVIEW-Hilfe.
 speichern, 3-7
Optionen für die Arbeitsumgebung
 festlegen, 3-6
 Siehe auch LabVIEW-Hilfe.
 speichern, 3-7
Optionen für Rückgängigmachen, 3-6
Oszilloskopdiagramm, 11-7

P

Paletten, 3-1, 3-4
 aktualisieren. *Siehe LabVIEW-Hilfe.*
 ändern. *Siehe LabVIEW-Hilfe.*
 Ansichten, 3-5
 Einfügen von Objekten. *Siehe LabVIEW-Hilfe.*
 Elementepalette, 3-1
 Farbwahltabelle, 4-11
 Funktionspalette, 3-1
 gemeinsam nutzen. *Siehe LabVIEW-Hilfe.*
 navigieren und suchen, 3-2
 organisieren, 3-5
 Referenz. *Siehe LabVIEW-Hilfe.*
 Werkzeugpalette, 3-2
Panel-Anordnung festlegen, 4-4
Parameter
 Datentypen (Tabelle), 5-2
Passwortschutz, 7-14
PDF-Bibliothek, 1-3
Pfade
 Bedien- und Anzeigeelemente, 4-13
 Datentyp (Tabelle), 5-3
 verwenden. *Siehe LabVIEW-Hilfe.*
Datei-I/O, 13-7
Hinzufügen von Verzeichnissen zum VI-Suchpfad. *Siehe LabVIEW-Hilfe.*
Konstanten, 5-5
 leer, 4-13
 Optionen, 3-6
 ungültig, 4-13
Pipes-Kommunikation, 17-13
Pixmaps, 12-5
Planen von Projekten, 7-1
PNG-Dateien, 12-7, 14-5
 Webserver, 17-12
Polare Graphen, 12-3

Polymorph
 Ausdrucks-knoten, 20-4
 Bedien- und Anzeigeelemente
 Datentyp (Tabelle), 5-4
 Funktionen, B-1
 VIs
 Auswählen einer Standardinstanz.
Siehe LabVIEW-Hilfe.
 Bearbeiten von Kontextmenüs. *Siehe
 LabVIEW-Hilfe.*
 Entfernen von Sub-VIs aus. *Siehe
 LabVIEW-Hilfe.*
 erstellen, 5-16

Polymorphe
 VIs, 5-16

Popup-Menüs. *Siehe* Kontextmenüs.

Portable Network Graphics-Dateien, 12-7

positiver Zahlenüberlauf, 6-8

PPC-Toolbox, 17-13

Probe-Daten-Werkzeug, Fehlersuche
 in VIs, 6-5

Problembhebung. *Siehe* Fehlersuche.

Programm-zu-Programm-
 Kommunikation, 17-13

Projektentwurf, 7-1

Projektplanung, 7-1

Protokolldateibindung, 13-17
 ändern, 13-20
 löschen, 13-19

Protokolle
 DataSocket, 17-4
 Low-Level-Kommunikation, 17-12

Protokollieren von Daten. *Siehe*
 Datenprotokollierung.

Prüfen des verfügbaren Festplattenspeichers.
Siehe LabVIEW-Hilfe.

Puffer
 Variant-Daten, 5-20

Pull-down-Menüs auf dem Frontpanel, 4-16

Punkte
 Typumwandlung, 5-15

Q

Quellanschlüsse. *Siehe* Bedienelemente.
 Quellcode. *Siehe* Blockdiagramm.
 Quellcodekontrolle, 7-2
 Queues
 Variant-Daten, 5-20

R

RefNums
 Bedien- und Anzeigeelemente, 4-18
 Datentyp (Tabelle), 5-3
 verwenden. *Siehe LabVIEW-Hilfe.*

Refnums
 Automatisierung, 18-2
 Bedienelement, 16-9
 Datei-I/O, 13-1
 Knoten „Aufruf über Referenz“, 16-8
 strikt typisierte, 16-8

Register-Bedien- und Anzeigeelemente, 4-14

Reparieren
 VIs, 6-2
 Techniken für die Fehlersuche, 6-4

Revisions-Historie
 Drucken, 14-4
 Erstellen, 14-2
 Nummern, 14-3

Richtlinien für die Entwicklung, 1-3

Ring-Bedienelemente, 4-16
 verwenden. *Siehe LabVIEW-Hilfe.*

Rotatorische Bedienelemente und
 Anzeigen, 4-10

RTF
 Hilfedateien, 14-6
 Speichern als, 14-4

rtn-Datei, 15-3

Runduminstrumente
 Frontpanel, 4-10
 Hinzufügen von Farbrampen, 4-12

Runtime-Menü-Datei, 15-3

S

- Schalter auf dem Frontpanel, 4-12
- Schaltflächen
 - steuern mit Tastenkombinationen, 4-3
- Schieber
 - hinzufügen, 4-10
- Schieberegister, 8-6
 - Standarddaten, 6-9
- Schieberegler und Anzeigen, 4-10
- Schleifen
 - Auto-Indizierung, 8-4
 - endlos, 8-4
 - For, 8-2
 - Schieberegister, 8-6
 - Steuern des Timings, 8-7
 - unerwartete Daten, 6-9
 - verwenden. *Siehe LabVIEW-Hilfe.*
 - While, 8-3
- Schreiben in Dateien, 13-1
- Schreibkonventionen in diesem Handbuch, *xviii*
- Schriftarten
 - Applikation, 4-21
 - Dialog, 4-21
 - Einstellungen, 4-20
 - Optionen, 3-6
 - System, 4-21
- Schrittweises Bewegen durch VIs
 - Fehlersuche in VIs, 6-5
- Schwach typisierte Objekt-Refnums, 16-10
- Selektoranschlüsse, 8-8
 - Werte, 8-8
- Senken-Anschlüsse. *Siehe Anzeigeelemente.*
- Sequenzstrukturen, 8-10
 - Steuern der
 - Ausführungsreihenfolge, 5-24
 - verwenden. *Siehe LabVIEW-Hilfe.*
 - zu häufige Verwendung, 8-11
- Server
 - ActiveX, 18-5
- Signalverlauf
 - Bedien- und Anzeigeelemente
 - Datentyp (Tabelle), 5-3
 - Datentyp, 11-18
 - Diagramme, 11-11
 - Funktionen, 5-10
 - Graphen, 11-8
 - Datentypen, 11-8, 11-9
 - Grafiken, 12-4
- Signalverläufe
 - Lesen aus Dateien, 13-13
 - Schreiben in Dateien, 13-12
- Skalieren
 - Frontpanel-Objekte, 4-7
 - Graphen, 11-5
- Skript-Knoten
 - HiQ, 20-5
 - MATLAB, 20-5
- Smith-Diagramme, 12-4
- Sound, 12-8
- Speicher
 - Deaktivieren der
 - Fehlersuche-Werkzeuge, 6-8
 - freigeben. *Siehe LabVIEW-Hilfe.*
 - globale Variablen, 10-7
 - komprimieren. *Siehe LabVIEW-Hilfe.*
 - lesen aus und schreiben an mit
 - Variant-Daten, 5-20
 - lokale Variablen, 10-6
 - Typumwandlungspunkte, 5-15
 - verwalten mit dem Datenfluss-
 - Programmierungsmodell, 5-25
- Speichern
 - Optionen für die Arbeitsumgebung, 3-7
- Speichern von Dateien
 - vorgeschlagenes Verzeichnis zum, A-3
- Speichern von VIs, 7-11
 - Bibliotheken, 7-12
 - Einzeldateien, 7-12
 - für Vorläuferversionen, 7-14

- zurückkehren zur zuletzt gespeicherten Version. *Siehe LabVIEW-Hilfe.*
- Sperren
 - Frontpanel-Obj., 4-6
 - VIs. *Siehe LabVIEW-Hilfe.*
- Spreadsheet-Dateien
 - Erstellen, 13-9
- Standarddaten
 - Arrays, 6-9
 - While-Schleifen, 6-9
- Stapel
 - Variant-Daten, 5-20
- Stapelplots, 11-7
- Starten von VIs von der Befehlszeile aus. *Siehe LabVIEW-Hilfe.*
- Statische Frontpanel-Bilder, 17-11
- Steuerfluss-Programmierungsmodell, 5-23
- Steuern
 - Instrumente, 7-3
 - VIs programmatisch, 16-1
 - VIs, die als Sub-VIs aufgerufen werden, 7-4
- Steuerung
 - Programmatische Steuerung von Frontpanel-Objekten, 16-9
 - VIs im Web, 17-10
- Streifendiagramm, 11-7
- Strikt typisierte Refnums
 - Bedienelement, 16-9
 - VI, 16-8
- Strikte Typenüberprüfung, 5-21
- Strings, 9-1
 - aufteilen. *Siehe LabVIEW-Hilfe.*
 - Bedien- und Anzeigeelemente, 4-12
 - Anzeigearten, 9-2
 - Bedienelemente
 - Datentyp (Tabelle), 5-3
 - Ersetzen von Text. *Siehe LabVIEW-Hilfe.*
 - formatieren, 9-4
 - Bezeichner, 9-5
 - Funktionen, 5-8
 - globale Variablen, 10-7
 - Konstanten, 5-5
 - Polymorphismus, B-5
 - programmgesteuert bearbeiten, 9-3
 - Tabellen, 9-3
 - Vergleichen, C-1
 - Zahlen zu, 9-5
- Struktur der LabVIEW-Verzeichnisse, A-1
- Struktur- und Support-Verzeichnis
 - menus, A-2
 - project, A-2
 - resource, A-2
 - templates, A-2
 - WWW, A-2
- Strukturen, 8-1
 - Case-, 8-8
 - entfernen. *Siehe LabVIEW-Hilfe.*
 - Fehlersuche. *Siehe LabVIEW-Hilfe.*
 - For-Schleifen, 8-2
 - Globale Variablen, 10-2
 - im Blockdiagramm, 2-4
 - lokale Variablen, 10-1
 - löschen. *Siehe LabVIEW-Hilfe.*
 - Sequenz, 8-10
 - Überblick, 8-1
 - Verbindungen. *Siehe LabVIEW-Hilfe.*
 - verwenden. *Siehe LabVIEW-Hilfe.*
 - While-Schleifen, 8-3
- Subroutinen. *Siehe Sub-VIs.*
- Sub-VIs, 7-5
 - Abrufen von Frontpanel-Daten, 13-20
 - Anzeigen der Kette der Aufrufenden, 6-7
 - Anzeigen von Namen nach der Positionierung. *Siehe LabVIEW-Hilfe.*
 - Aussetzen der Ausführung, 6-6
 - Bedienelement-Untertitel für Hinweisstreifen. *Siehe LabVIEW-Hilfe.*
 - Drucken von VIs, 14-8
 - entfernen aus polymorphen VIs. *Siehe LabVIEW-Hilfe.*

- entwerfen, 7-10
 - Ermitteln der aktuellen Instanz, 6-7
 - erstellen, 7-5, 7-10
 - Situationen, die vermieden werden sollten. *Siehe LabVIEW-Hilfe.*
 - Frontpanel, 7-10
 - Hierarchie, 7-11
 - kopieren, A-3
 - polymorphe VIs, 5-16
 - steuern des Verhaltens, 7-4
 - Überblick, 7-5
 - Suchen
 - Bedienelemente, VIs und Funktionen in den Paletten, 3-2
 - Fehler, 6-2
 - Objekte, Text und VIs. *Siehe LabVIEW-Hilfe.*
 - Suchen nach
 - Bedienelementen, VIs und Funktionen in den Paletten, 3-2
 - VI-Hierarchie. *Siehe LabVIEW-Hilfe.*
 - Support im Internet, E-1
 - Symbole, 2-5
 - bearbeiten, 7-9
 - Drucken, 14-4
 - erstellen, 7-9
 - Überblick, 2-5
 - Symbolleiste, 3-4
 - System Exec.vi, 17-13
 - Systemintegration, E-1
 - Systemschriftart, 4-21
- ## T
- Tabellen, 9-3
 - Größe ändern. *Siehe LabVIEW-Hilfe.*
 - Tabellenkalkulationsdateien
 - Schreiben von numerischen Daten an, 9-5
 - Tank-schieberegler und Anzeigen, 4-10
 - Tasten
 - Frontpanel, 4-12
 - Tastenkombination, 4-3
 - Festlegen der Navigationsreihenf., 4-4
 - Steuern von Schaltflächen, 4-3
 - TCP, 17-12
 - VI-Server, 16-1
 - Technischer Support, E-1
 - Text
 - drag and drop. *Siehe LabVIEW-Hilfe.*
 - Eingabefelder, 4-12
 - formatieren, 4-20
 - Ring-Bedienelemente, 4-16
 - suchen. *Siehe LabVIEW-Hilfe.*
 - Textdateien
 - Datei-I/O, 13-2
 - Erstellen, 13-9
 - Schreiben von numerischen Daten an, 9-5
 - Thermometer
 - Schieberegler und Anzeigen, 4-10
 - Threads
 - Ausführen mehrerer Threads. *Siehe LabVIEW-Hilfe.*
 - Timing
 - steuern, 8-7
 - Transmission Control Protocol, 17-12
 - Transparent
 - Beschriftungen. *Siehe LabVIEW-Hilfe.*
 - Transparente
 - Objekte. *Siehe LabVIEW-Hilfe.*
 - Treiber
 - für Instrumente. *Siehe LabVIEW-Hilfe.*
 - Tunnel, 8-1
 - Eingabe und Ausgabe, 8-9
 - Tutorium, 1-2
 - Tutorium- und Anleitungsverzeichnis
 - activity, A-2
 - examples, A-2
 - tutorial, A-2
 - Typumwandlungspunkte, 5-15

U

Überlagerte Kurven, 11-7
 Überlappende Frontpanel-Objekte, 4-14
 Übertragungsleitungen, 12-4
 UDP, 17-12
 Uhrzeitformat
 Optionen, 3-6
 undefinierte Daten, 6-8
 Arrays, 6-9
 For-Schleifen, 6-9
 Inf (Infinity, unendlich), 6-8
 NaN (Not a Number, keine Zahl), 6-8
 prüfen auf, 6-8
 verhindern, 6-10
 While-Schleifen, 6-9
 Ungültige Pfade, 4-13
 Unterbrochene Verbindungen, 5-14
 Unterpalletten
 Erstellen von ActiveX, 3-6
 erstellen. *Siehe LabVIEW-Hilfe.*
 organisieren, 3-5
 verschieben. *Siehe LabVIEW-Hilfe.*
 Unterstützung
 Dateien für den Aufruf von HiQ, 20-7
 Untertitel, 4-20
 erstellen. *Siehe LabVIEW-Hilfe.*
 Sub-VI-Hinweisstreifen. *Siehe LabVIEW-Hilfe.*
 URLs für DataSocket, 17-4
 User Datagram Protocol, 17-12

V

Variablen
 global, 10-2
 Ablaufunterschiede, 10-5
 erstellen, 10-3
 initialisieren, 10-5
 lesen und schreiben, 10-4
 Speicher, 10-7

 Überblick, 10-2
 umsichtig verwenden, 10-5
 lokal, 10-1
 Ablaufunterschiede, 10-5
 erstellen, 10-2
 initialisieren, 10-5
 lesen und schreiben, 10-4
 Speicher, 10-6
 Überblick, 10-1
 umsichtig verwenden, 10-5
 Variant-Daten, 5-19
 ActiveX, 18-3
 Bearbeiten von Attributen. *Siehe LabVIEW-Hilfe.*
 Bedien- und Anzeigeelemente
 Datentyp (Tabelle), 5-3
 DataSocket, 17-9
 in geglättete Daten umgewandelte Daten
 und, 5-20
 umwandeln in, 5-19
 verarbeiten, 5-19
 Veranstaltungen (Customer Education), E-1
 Verbinden von Anschlüssen, 5-11
 Verbindung
 automatisch, 5-13
 Blockdiagramm, 5-11
 Einheiten, 5-21
 manuell, 5-14
 Strukturen. *Siehe LabVIEW-Hilfe.*
 Techniken, 5-26
 Werkzeug, 5-14
 Verbindungen, 2-4
 markieren, 5-14
 Überblick, 2-4
 unterbrochen, 5-14
 Verbindungshilfen. *Siehe LabVIEW-Hilfe.*
 verschieben. *Siehe LabVIEW-Hilfe.*
 Verfolgen der Entwicklung. *Siehe Dokumentieren von VIs.*

- Vergleichen
 - boolesche Werte, C-1
 - Strings, C-1
 - Versionen von VIs, 7-2
 - Zahlen, C-2
- vergleichen
 - Arrays, C-2
 - Cluster, C-2
- Vergleichsfunktionen, C-1
 - Polymorphismus, B-6
- Verknüpfen von VIs mit erstellten
Hilfdateien. *Siehe LabVIEW-Hilfe.*
- Verknüpfte Beschriftungen, 4-20
 - bearbeiten. *Siehe LabVIEW-Hilfe.*
- Veröffentlichen von VIs im Web, 17-10
- Verschieben
 - Arrays. *Siehe LabVIEW-Hilfe.*
 - Cluster. *Siehe LabVIEW-Hilfe.*
 - Objekte. *Siehe LabVIEW-Hilfe.*
 - Unterpaletten. *Siehe LabVIEW-Hilfe.*
 - Verbindungen. *Siehe LabVIEW-Hilfe.*
- Versionen
 - Speichern von VIs für
 - Vorläuferversionen, 7-14
 - vergleichen, 7-2
 - zurückkehren zur zuletzt gespeicherten
Version. *Siehe LabVIEW-Hilfe.*
- Versionsnummer
 - anzeigen in Titelleiste. *Siehe
LabVIEW-Hilfe.*
- Verzeichnis zum Speichern von Dateien, A-3
- Verzeichnispfade. *Siehe* Pfade.
- Verzeichnisse
 - konvertieren in Bibliotheken. *Siehe
LabVIEW-Hilfe.*
 - Konvertieren von Bibliotheken in. *Siehe
LabVIEW-Hilfe.*
- Verzeichnisstruktur von LabVIEW
 - Macintosh, A-2
 - Organisation, A-1
- VI-Objekt
 - Bearbeiten von Einstellungen, 16-4
 - VI-Server, 16-3
- Virtuelle Instrumente. *Siehe* VIs.
- VIs, 2-1
 - aktualisieren, 7-14
 - Aufheben der Sperrung. *Siehe
LabVIEW-Hilfe.*
 - Aufrufen über das Netzwerk, 16-1
 - ausführbare Fehlersuche. *Siehe
LabVIEW-Hilfe.*
 - ausführen, 6-1
 - Beispiele, 1-4
 - benennen, 7-13
 - bereitstellen, 7-14
 - Bibliotheken, 7-12
 - Dokumentieren, 14-2
 - drag and drop. *Siehe LabVIEW-Hilfe.*
 - Drucken, 14-6
 - Dynamisches Aufrufen, 16-7
 - Dynamisches Laden, 16-7
 - entfernen aus Bibliotheken. *Siehe
LabVIEW-Hilfe.*
 - entwickeln, 7-1
 - erstellen, 7-1
 - Erstellen von Beschreibungen, 14-2
 - Erstellen von Hinweisstreifen, 14-2
 - Fehlerbehandlung, 6-10
 - gemeinsam nutzen, 7-14
 - Hierarchie, 7-11
 - hinzufügen zu Bibliotheken, 3-4
 - Konfigurieren von Erscheinungsbild und
Verhalten, 15-1
 - kopieren, A-3
 - korrigieren, 6-2
 - markieren als Top-Level in Bibliotheken.
Siehe LabVIEW-Hilfe.
 - nicht ausführbar, 6-2
 - öffnen im Ausführungsmodus. *Siehe
LabVIEW-Hilfe.*
 - polymorphe, 5-16

- Programmatische Steuerung, 16-1
- Referenz. *Siehe LabVIEW-Hilfe.*
- speichern, 7-11
- sperren. *Siehe LabVIEW-Hilfe.*
- starten von der Befehlszeile aus. *Siehe LabVIEW-Hilfe.*
- Steuern im Web, 17-10
- steuern, wenn sie als Sub-VIs aufgerufen werden, 7-4
- Strikt typisierte Refnums, 16-8
- suchen. *Siehe LabVIEW-Hilfe.*
- Techniken für die Fehlersuche, 6-4
- Überblick, 2-1
- Vergleichen von Versionen, 7-2
- Veröffentlichen im Web, 17-10
- zurückkehren zur zuletzt gespeicherten Version. *Siehe LabVIEW-Hilfe.*
- VISA
 - Übergeben von Ressourcennamen, 4-17
- VI-Server, 16-1
 - Anwendungsobjekt, 16-3
 - Arbeiten im Netzwerk und, 17-1
 - Aufruf über Referenz, 16-8
 - Aufrufen von anderen Instanzen von LabVIEW im Web, 16-1
 - Aufrufen von VIs über das Netzwerk, 16-1
 - Bearbeiten von Einstellungen, 16-4
 - Eigenschaftsknoten, 16-4
 - Erstellen von Applikationen, 16-2
 - Fähigkeiten, 16-1
 - Methodenknoten, 16-5
 - Netzwerkapplikationen, 16-9
 - Steuern von Frontpanel-Objekten, 16-9
 - Strikt typisierte VI-Refnums, 16-8
 - Überblick, 16-1
 - VI-Objekt, 16-3
- VI-Suchpfad bearbeiten. *Siehe LabVIEW-Hilfe.*
- Vordergrundfarbe von Frontpanel-Objekten. *Siehe LabVIEW-Hilfe.*
- Voreinstellungen. *Siehe* Optionen.
- Vorübergehendes Deaktivieren von Abschnitten des Blockdiagramms
 - Fehlersuche in VIs, 6-7
- VXI
 - VIs, 1-3
 - VXI konfigurieren, 1-4
- W**
- Warnungen
 - anzeigen, 6-3
 - Schaltfläche, 6-3
 - standardmäßig anzeigen. *Siehe LabVIEW-Hilfe.*
- Web
 - Aufrufen von anderen Instanzen von LabVIEW, 16-1
 - Erstellen von HTML-Dokumenten, 17-11
 - Steuern von VIs, 17-10
 - Veröffentlichen von VIs, 17-10
- Web-Dokumentationswerkzeug, 17-11
- Webserver, 17-10
 - Aktivieren, 17-11
 - Optionen, 17-10
- Wechseln zwischen Frontpanel-Objekte mit der Tab-Taste, 4-4
- Weltweiter technischer Support, E-2
- Werkzeuge
 - auswählen. *Siehe LabVIEW-Hilfe.*
 - Palette, 3-2
- Werkzeugsätze, 1-1
 - in Paletten, 3-6
- While-Schleifen, 8-3
 - Auto-Indizierung, 8-6
 - Bedingungsanschluss, 8-3
 - endlos, 8-4
 - Fehlerbehandlung, 6-12
 - Iterationsanschlüsse, 8-3
 - Schieberegister, 8-6

Standarddaten, 6-9
Steuern des Timings, 8-7
verwenden. *Siehe LabVIEW-Hilfe.*
wiederholen
Codeblöcke
 For-Schleifen, 8-2
 While-Schleifen, 8-3
Windows-Hilfedateien, 14-6

X

X-Achsen
 Formatieren, 11-5
 mehrere, 11-2
XY-Graphen, 11-8
 Datentypen, 11-10

Y

Y-Achsen
 Formatieren, 11-5
 mehrere, 11-2

Z

Zähl-Anschlüsse, 8-2
Zahlen
 ändern der Darstellung. *Siehe LabVIEW-Hilfe.*
 außerhalb des Bereichs, 4-17
 Datentypen (Tabelle), 5-2
 Funktionen, 5-7

Konstanten, 5-5
Konvertieren, B-1
Maßeinheiten, 5-20
Polymorphismus, B-2
positiver und negativer Überlauf, 6-8
Schreiben von Daten an
 Tabellenkalkulations- oder
 Textdateien, 9-5
Strings und, 9-5
Vergleichen, C-2
Zahlen außerhalb des Bereichs, 4-17
Zeichenformatierung, 4-20
Zeichnen
 Grafik glätten. *Siehe LabVIEW-Hilfe.*
Zeichnen. *Siehe auch Grafiken.*
Zeiger
 hinzufügen, 4-11
Zeitfunktionen, 5-9
Zoomen in Grafiken und Diagrammen. *Siehe LabVIEW-Hilfe.*
Zurückkehren zur zuletzt gespeicherten
 Version. *Siehe LabVIEW-Hilfe.*
Zuweisen
 Passwörter zu Blockdiagrammen, 7-14
Zuweisen von Farbe
 Frontpanel-Objekte, 4-5
 Hintergrund. *Siehe LabVIEW-Hilfe.*
 Kopieren von Farben. *Siehe LabVIEW-Hilfe.*
 Vordergrund. *Siehe LabVIEW-Hilfe.*
transparente Objekte. *Siehe LabVIEW-Hilfe.*