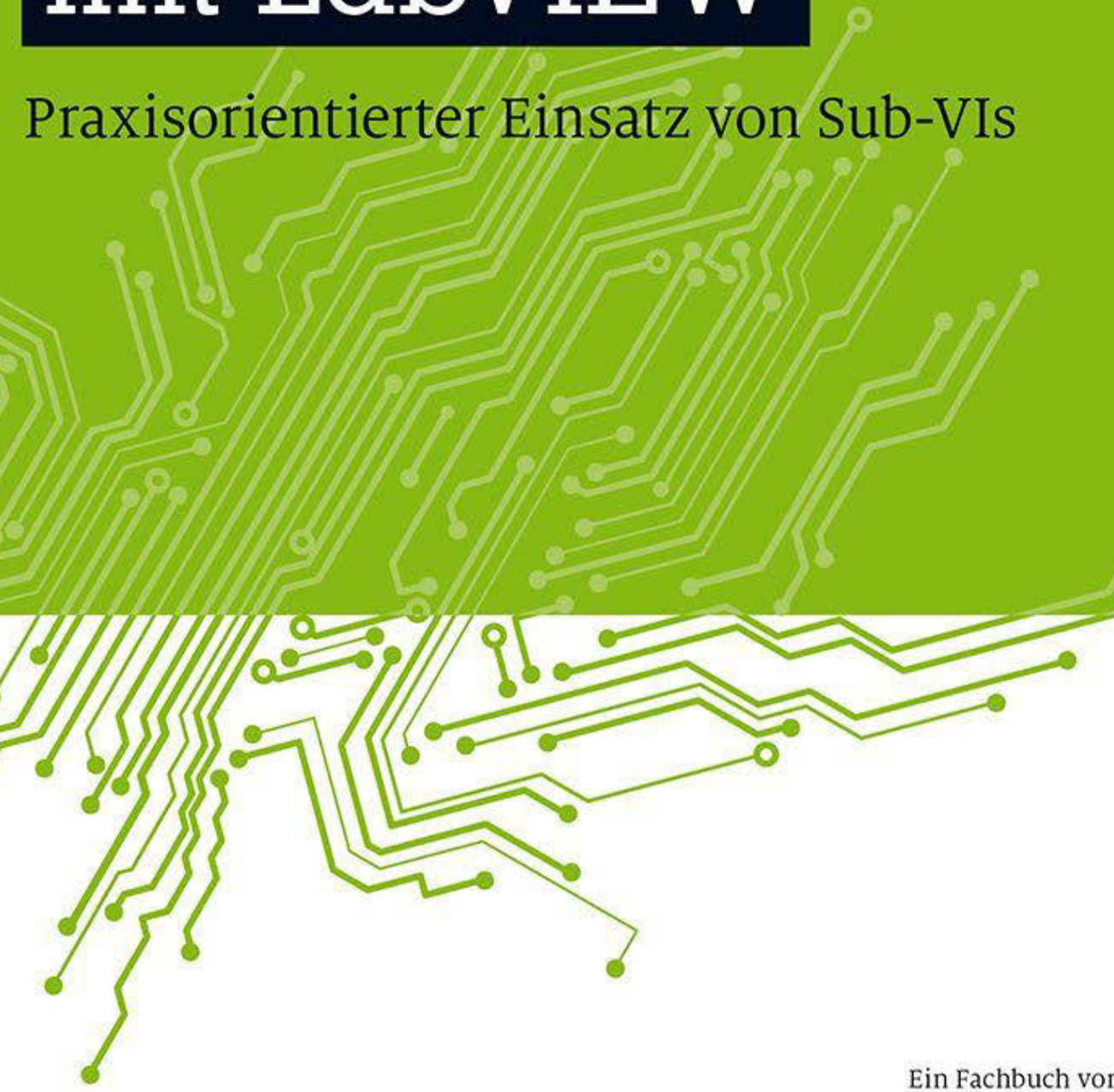
Walter Müller

Messdaten-Analyse mit LabVIEW

Praxisorientierter Einsatz von Sub-VIs



Ein Fachbuch von

elektro

Walter Müller
Messdaten-Analyse mit LabVIEW

Walter Müller

Messdaten-Analyse mit LabVIEW

Praxisorientierter Einsatz von Sub-VIs zu mathematisch–technischen Berechnungen

Vogel Business Media

Weitere Informationen:
www.vbm-fachbuch.de

ISBN 978-3-8343-3377-3
2. Auflage. 2016

Vorwort

LabVIEW ist ein Programmiersystem der Firma National Instruments, das seine Stärken besonders bei der Programmierung technischer und mathematischer Probleme zeigt. Für eine Vielzahl typischer Aufgabenstellungen sind parametrierbare Module (Sub-VIs) bereitgestellt. Durch die Programmierung mit grafischen Elementen, die durch entsprechende Signalleitungen miteinander verbunden werden, fällt der Einstieg besonders leicht. Auch der Hardware-Zugriff wird durch viele vorbereitete Elemente unterstützt. Die Vielzahl von Funktionen und Erweiterungsmöglichkeiten erschwert dem Einsteiger aber auch die Übersicht.

Es ist wohl kaum möglich, alle typischen Problemstellungen, alle standardisierten Verfahren dazu, deren Möglichkeiten und Grenzen sowie geeignete Algorithmen zu kennen. Daher wird hier der Versuch unternommen, durch eine thematische Gliederung, kurze Problembeschreibungen und möglichst einfache Programmbeispiele den Umgang mit wichtigen mathematischen Operatoren in LabVIEW zu erleichtern und damit Vertrauen und Sicherheit zu gewinnen. Angesprochen werden sollen einerseits Anwender, die bereits erste Erfahrungen mit LabVIEW besitzen oder die Lösungen eines konkreten Problems suchen, andererseits aber auch alle, die an numerischen Verfahren interessiert sind, die mit wenig Aufwand Verfahren kennen lernen, testen oder entwickeln möchten.

Zunehmend besteht im technisch-naturwissenschaftlichen Bereich ein Bedarf darin, für den Umgang mit technischen Systemen mathematische Verfahren in ihrer Wirkungsweise zu verstehen, Ergebnisse abzuschätzen und Auswirkungen von Wert- oder Parameteränderungen grob vorhersagen zu können. Dafür sind einerseits von Fachleuten abgesicherte Systeme für die korrekte Ausführung der Operationen erforderlich, andererseits muss der Anwender ein begründetes Vertrauen entwickeln, ein geeignetes Verfahren auswählen, parametrieren und interpretieren können. Auch Kritiker dieser scheinbar oberflächlichen Vorgehensweise vertrauen der Quadratwurzel-Berechnung des Taschenrechners.

Für viele Menschen stellt der mathematische Formalismus ein größeres Hindernis dar. Viele Verfahren lassen sich aber geometrisch veranschaulichen und in Schritten entwickeln, so dass die zugehörigen Begriffe, die grundlegenden Ideen und die Verhaltensweisen des mathematischen Systems Gestalt gewinnen können. Wenn die Verfahren von anderer Seite in ihrem mathematischen Kern abgesichert sind, lassen sie sich für eigene Problemlösungen einsetzen. Problematisch bleiben aber immer noch das Erkennen der Grenzen der Anwendbarkeit und die Beurteilung der Unsicherheitsgrenzen.

Ein Zugang zu Verständnis und Anwendung ergibt sich oft, wenn das Verhalten dieser mathematischen Systeme durch den Umgang mit ihnen bekannt und geläufig wird und wenn Grenzen und Möglichkeiten in der Simulation oder zusammen mit einem realen Experiment erfahren werden können. Die dargestellten Programmbeispiele unterstützen bei Weitem nicht alle Möglichkeiten. Oft ist bewusst nur ein einfacher Fall realisiert, von dem ausgehend weitere Funktionen und Eigenschaften erkundet werden können. Die Auswahl der besprochenen Verfahren orientiert sich an typischen Aufgaben bei der Auswertung bereits vorliegender Messdaten.

Um einen einfachen Zugang zu ermöglichen, wird auf manche Möglichkeiten, die LabVIEW bietet, verzichtet:

- Bereitgestellte Funktionen, die hauptsächlich der Simulation dienen, rein mathematische Operationen darstellen oder zur Kommunikation mit Geräten oder anderen Programmen verwendet werden, sind weniger berücksichtigt.
- Für Standard-Anwendungen sind in LabVIEW viele Funktionen in sog. Express-VIs zusammengefasst und können leicht mit geeigneten Ein- und Ausgabeelementen versehen werden. Um

elementare Verarbeitungsverfahren und Eigenschaften herauszustellen, werden diese Bausteine hier nicht verwendet.
- Es wurde auch auf eine elegante, vielseitige Bedienoberfläche und die Fehlerverfolgung verzichtet.
- Parameteränderungen werden teilweise nur über Konstanten im Blockdiagramm vorgenommen. Alle Programme haben einen rein experimentellen Charakter, enthalten möglicherweise auch Fehler und sind in keiner Weise gegen Fehlbedienung abgesichert.

Die in den Sub-VIs des Herstellers verwendeten Algorithmen sind nur teilweise für eine weitere Untersuchung zugänglich. Daher können meist keine Aussagen über das verwendete Verfahren, die numerische Stabilität oder über eine Fehlerabschätzung gemacht werden.

Den Zugang zu Verständnis und Anwendung erleichtern die folgenden Punkte:

- Die Idee zum Verfahren, zur Wirkungsweise des Algorithmus und zum Einfluss der Parameter wird anschaulich beschrieben.
- Soweit es möglich ist, werden Ergebnisse in Diagrammen dargestellt.
- Alle Programme können über den Onlineservice des Verlags heruntergeladen werden. Alle VIs und Sub-VIs des Autors sind nicht gesperrt.
- **Beispiele** zeigen typische Anwendungen der Verfahren.
- **Infos** beschreiben Sachverhalte aus der Umgebung des Problems oder der Programmierung.
- **Anwendungen** sind umfangreichere Lösungen technisch-mathematischer Problemstellungen. Eine Beschreibung in Textform und die zugehörigen Programme können über den Onlineservice des Verlags heruntergeladen werden.
- Um die Zuordnung der Elemente in Frontpanel und Blockdiagramm zu erkennen und die Namen der Sub-VIs zu erhalten, empfiehlt es sich, mit den Programmen in der Entwicklungsumgebung zu spielen.
- Ausgehend von den in den Programmen festgelegten Daten und Parametern kann bei kleinen Änderungen die Auswirkung im Diagramm oder in übersichtlichen Tabellen betrachtet werden.

Zur grundlegenden Einführung in die Programmierung mit LabVIEW, zu den mathematisch-numerischen Grundlagen und die algorithmische Umsetzung von Verfahren wird auf bereits vorhandene Literatur und Onlinekurse verwiesen.

Das Denken in Funktionseinheiten hat einen festen Platz im Umgang mit technischen und mathematischen Systemen schon lange eingenommen. Der Mathematikunterricht in der Schule und besonders in der beruflichen Aus- und Weiterbildung sollte diese Richtung aufgreifen und eine erzählende, experimentelle und anwendungsorientierte Didaktik dafür entwickeln. Keinesfalls soll dies aber auf Kosten einer gründlichen mathematischen Bildung geschehen.

Karlsbad, August 2015 Walter Müller

Inhaltsverzeichnis

1 Grundlagen

1.1 Verfahren und Beispiele

Der Zustand technischer Systeme lässt sich durch numerische und logische Größen beschreiben. Das Verhalten dieser Systeme kann, meist nur näherungsweise, durch mathematische Beziehungen simuliert, überprüft und vorhergesagt werden. Dafür sind oft Größen notwendig, die der direkten Messung nicht zugänglich sind. Auf der anderen Seite sind die messbaren Größen in der Regel von Einflüssen überlagert, die die interessanten Informationen verbergen oder auch teilweise zerstören.

Die Verfahren der numerischen Messdatenverarbeitung dienen dazu, diese Informationen wieder zu restaurieren, hervorzuheben und für eine weitere Verarbeitung zugänglich zu machen.

Die kompletten Verfahren und Geräte, die zur Gewinnung von Messdaten geeignet sind, werden hier ausgeklammert; sie werden in den Beispielen durch Konstanten oder einfache Eingabe-Elemente zur Simulation ersetzt. Die Ergebnisse der Verarbeitung lassen sich oft in Diagrammdarstellungen am leichtesten erfassen.

Alle Programme, deren Name in den Bildunterschriften des Frontpanels (FP) und des Blockdiagramms (BD) angegeben ist, können über den Onlineservice des Verlags heruntergeladen werden (Bilder 1.1 und 1.2).

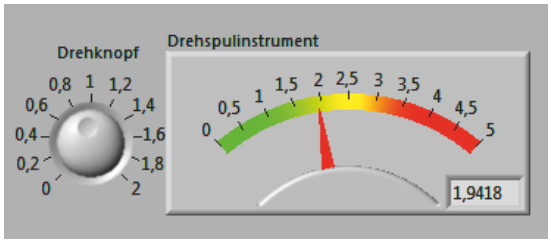


Bild 1.1 *FP „start1.vi“*

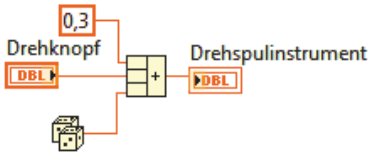


Bild 1.2 *BD „start1.vi“*

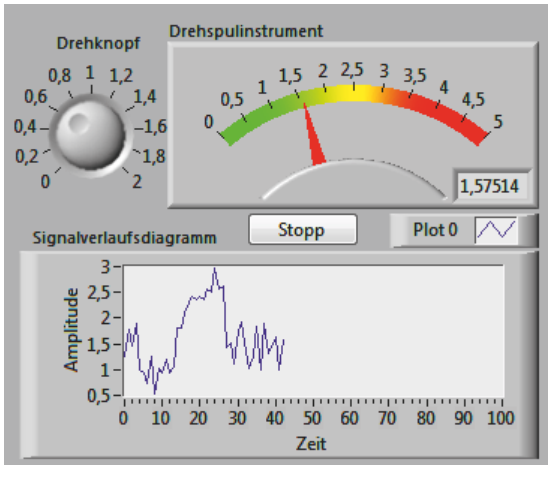


Bild 1.3 *FP „start2.vi“ mit Verlaufsdiagramm*

Bei der Programmerstellung stehen für Frontpanel (FP) und Blockdiagramm (BD) unterschiedliche Auswahlmenüs zur Verfügung. Beide Fenster zusammen bilden aber eine Einheit, so dass sich Aktionen in einem Fenster auch auf die Darstellung oder Funktion im anderen Fenster auswirken (Bilder 1.3 und 1.4).

Verschiedene Werkzeuge aus der Werkzeugpalette (Ansicht – Werkzeugpalette) werden bei eingeschalteter Automatik fast immer richtig ausgewählt, wenn der Mauszeiger entsprechende Stellen berührt.

Bei den hier beschriebenen, in LabVIEW vorbereiteten Funktionen, den Sub-VIs, handelt es sich um Unterprogramme, denen auf der einen Seite die Eingangsgrößen zugeführt werden (meist von links) und die bei jedem Aufruf einmal die daraus sich ergebenden Ausgangsgrößen ermitteln. Diese lassen sich in der Regel an den Anschlüssen auf der rechten Seite eines Funktionsbausteins (Blockelement; Sub-VI) abgreifen. Die einfachste Form des Programmbeispiels besteht daher aus dem Funktionsbaustein, Eingabe-Elementen als Konstanten oder Bedienelementen und zugehörigen Ausgabe-Elementen zur Ergebnisanzeige. Viele Beispiele können auf diesem Niveau getestet werden. Neben den im Programmsystem angebotenen Sub-VIs können auch eigene Sub-VIs erstellt werden.

Um sehen zu können, wie zeitliche Veränderungen oder wie innere Zustände eines Programms sich auf weitere Eingaben auswirken, bietet sich eine Erweiterung mit einer WHILE-Schleife, einer zeitlichen Taktung und der Anzeige in einem Diagramm an. Die Taktzeit wird in Millisekunden (ms) angegeben.

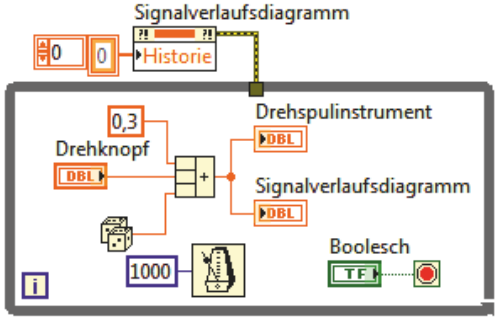


Bild 1.4
BD „start2.vi" – Diagrammanschluss

Beim Signalverlaufsdiagramm werden die anfallenden Daten intern gespeichert und synchron mit gleichem Zeitabstand auf der Abszisse angezeigt. Die Zeitachse kann über das Kontext-Menü skaliert werden. Anstelle des Drehknopfes zur Simulation kann auch (über ein Sub-VI) eine Verbindung zu einem Messgerät hergestellt werden (Bild 1.4).

Eigenschaftsknoten

Ein Diagramm kann über das Kontext-Menü manuell gelöscht werden. Alternativ lässt sich ein Eigenschaftsknoten zur Eigenschaft „Historiedaten" mit einem leeren Array als Eingang verwenden. Wird er außerhalb – oberhalb der Schleife – platziert, wird der Inhalt des Diagramms bei jedem Neustart des Programms automatisch gelöscht. Um eine feste Reihenfolge bei der Ausführung zu erzwingen, kann der Fehler-Ausgang des Eigenschaftsknotens mit dem Rand der Schleife verbunden werden.

- Erstellen – Eigenschaftsknoten – Historiedaten
- In Schreiben ändern
- Erstellen – Konstanten

Kontexthilfe
Zu jedem Programmelement kann eine kurze Hilfe angezeigt werden, die auch zu einer umfangreicheren Hilfe führt (aktivieren über „Hilfe – Kontexthilfe anzeigen").

Programmbeispiele
Eine umfangreiche Programmsammlung ist über „Hilfe – Beispiele suchen" erreichbar. Viele Beispiele sind darauf ausgerichtet, die vielfältigen Möglichkeiten bedienerfreundlich anzubieten und Fehlfunktionen abzufangen. Für den Einsteiger ist es dadurch etwas schwieriger, den eigentlichen Funktionskern zu entdecken.

Handbücher
Über www.ni.com/support findet man unterstützende Dokumente wie Handbücher (*manuals*), Grundlagenartikel (*knowledge basis*) oder Beispielcode.

1.2 Mathematik-Grundlagen

1.2.1 Konstanten

Für jede Signalart existiert ein Element, mit dem sich ein konstanter Wert festlegen lässt. Konstanten bzw. die zugehörigen Variablen unterscheiden sich darin, welche Länge und welche Bedeutung die interne binäre Darstellung des zugehörigen Speichers hat (Bild 1.5).

Bild 1.5
Numerische und logische Größen

z.B. Integer-Zahlen (blau)
Binäre Größen (grün)
Dezimalzahlen (orange)
Komplexe Zahlen (orange)

Häufig vorkommende Konstanten aus der Mathematik und der Physik findet man unter: „Funktionen – Numerisch – Konstanten".

Im Beispiel (Bild 1.6) sind numerische Konstanten vom Typ DBL (*double*: doppelte Genauigkeit) dargestellt.

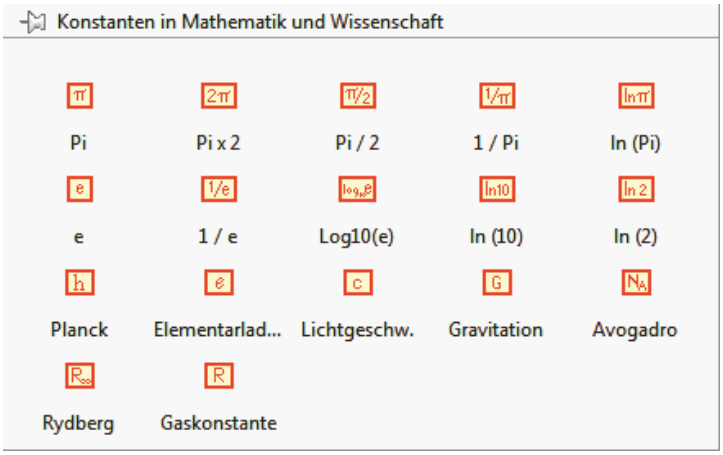


Bild 1.6 *Numerische und logische Größen*

1.2.2 Vergleichselemente

Unter „Funktionen – Vergleich" sind in den ersten beiden Zeilen leicht verstehbare Operatoren zu finden (Bild 1.7).

Bild 1.7 *Menü „Vergleichselemente"*

Die ersten drei Elemente der dritten Zeile sind bei Anwendungen der Automatisierung hilfreich:

- **Auswählen**
  schaltet wahlweise einen von zwei Werten auf den Ausgang (3. Reihe, 1. Element).
- **Max und Min**
  ordnet zwei Werte an. Liegen an den Eingängen Arrays an, werden die Werte elementweise zwischen den Arrays verglichen und in je einem Array der Minima bzw. Maxima ausgegeben (3. Reihe, 2. Element).
- **Wertebereich prüfen und erzwingen**
  begrenzt einen Eingabewert auf einen festgelegten Bereich (3. Reihe, 3. Element).

Das Begrenzungselement schneidet die Werte an der Ober- bzw. Untergrenze ab (*clipping*) und zeigt an, ob eine Begrenzung des x-Wertes stattgefunden hat (Bilder 1.8 und 1.9).

BEISPIEL: ANZEIGE WECHSELN

Die Einstellwerte für zwei Pumpen x und y sollen wahlweise auf eine Zeiger-Anzeige geschaltet werden können. Es sollen nur Werte zwischen 5 und 15 möglich sein. Einstellwerte außerhalb werden durch eine LED signalisiert. Ein Vergleich liefert Minimum und Maximum der Einstellungen.

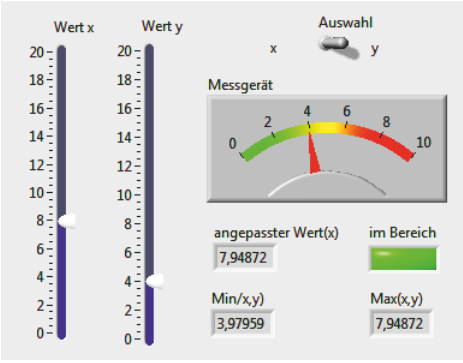


Bild 1.8 *FP „start3.vi" – Eingabe und Anzeige*

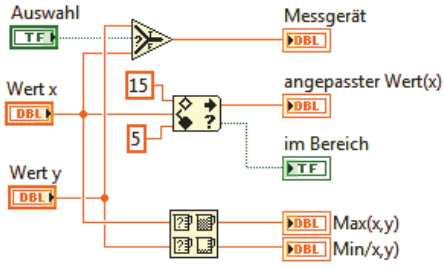


Bild 1.9 *BD „start3.vi" – Vergleichselemente*

1.2.3 Verknüpfungselemente

Wenn man aus zwei oder mehreren Werten von Variablen einen neuen Wert bildet, handelt es sich um eine Verknüpfung.

Grundverknüpfungen und einige gebräuchliche Funktionen stehen als Funktionsblock (Sub-VI) unter „Funktionen – Numerisch" bereit (Bild 1.10).

Die Untermenüs (rechte Spalte) enthalten:

- Funktionen zur Umwandlung von Datentypen,
- Operationen zur Art der digitalen Speicherung eines Wertes,
- Operationen mit komplexen Zahlen,
- Spezialfunktionen zur Sensorik,
- Formen zur Dezimal-Darstellung,
- weitere Konstanten.

Über „Hilfe – Kontexthilfe anzeigen" erhält man weitere Informationen.

Die Verknüpfungen orientieren sich an den Erfordernissen bei der Programmerstellung. Funktionen zum Rest bei der Division (1. Zeile, 5. Spalte), „inkrementieren" und „dekrementieren" und diverse Rundungsoperatoren sind typisch. Ein Element zum Potenzieren findet man unter

„Mathematik – Grund- und Spezialfunktionen“, es lässt sich jedoch auch mit dem Ausdrucksknoten realisieren.

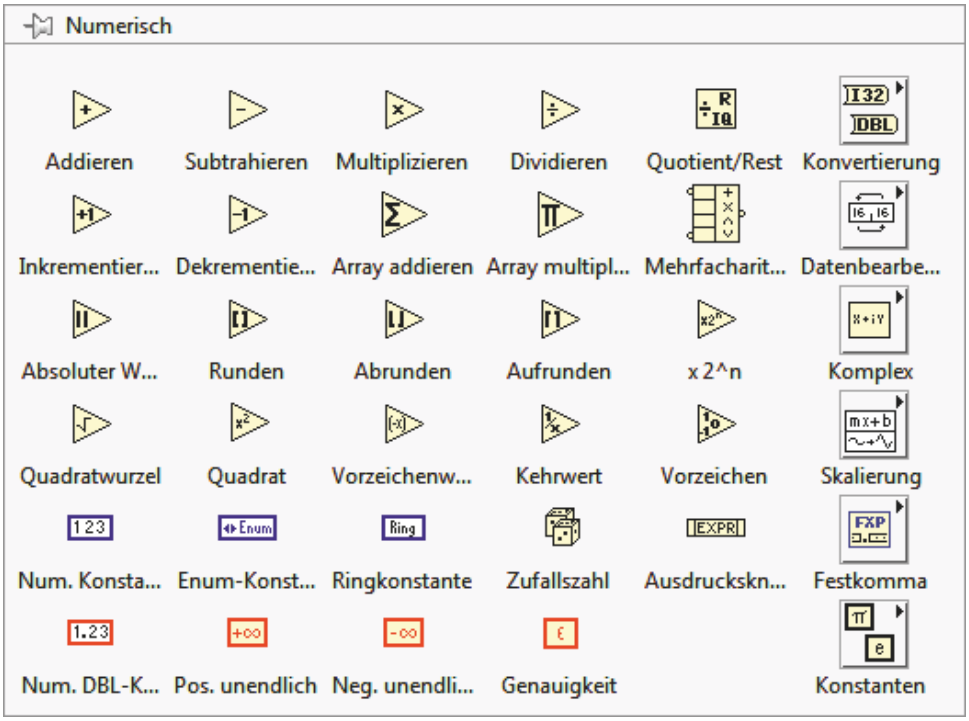


Bild 1.10 *Menü zu den numerischen Verknüpfungen*

Das Würfelsymbol erzeugt eine Zufallszahl zwischen 0 und 1.

Beim Ausdrucksknoten (Bild 1.11) wird der links angeschlossene Eingangswert mit x bezeichnet und kann in einer Formel zum Ausgangswert verknüpft werden, dem keine feste Bezeichnung zugeordnet ist.

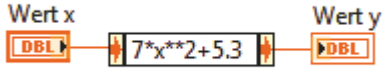


Bild 1.11
Ausdrucksknoten

ACHTUNG
Im Ausdrucksknoten muss x als Variable verwendet werden, Potenzieren wird mit ** ausgeführt und das Dezimaltrennzeichen ist der Punkt.

1.2.4 Funktionen einer Variablen

Bei einer Funktion wird durch eine Vorschrift den Elementen x einer Definitionsmenge je ein Element y einer Bildmenge zugeordnet. Bei den Grund- und Spezialfunktionen in LabVIEW stammen die Elemente der Definitionsmenge aus den möglichen Werten einer (hier eindimensionalen) Variablen. Besitzt ein Sub-VI mehrere Eingänge, sind die Größen neben dem Eingang der Variablen x als Konstanten (Parameter) zu betrachten.

Der Ausdrucksknoten (aus „Funktionen – Numerisch") erlaubt die Eingabe einer Formel mit allen gängigen Funktionen mit Parametern, die als numerische Werte in die Formel eingetragen werden müssen.

Im Menü „Mathematik" sind übersichtlich gegliedert viele Funktionen aus unterschiedlichen Anwendungsbereichen bereitgestellt (Bild 1.12). Weitere Funktionen (auch mehrdimensional) können mit der Struktur „Formelknoten" in einer textbasierten Programmierung definiert werden.

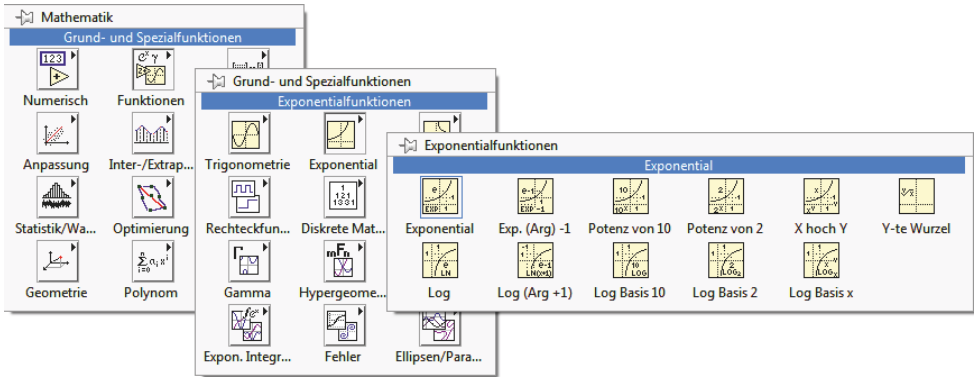
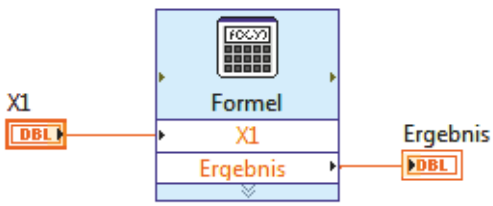


Bild 1.12 *Gegliedertes Menü mathematischer Funktionen*

1.2.5 Express-VI: Formel

Das Express-VI „Formel" (Bild 1.13) ermöglicht die Eingabe oder Editierung einer Formel über eine taschenrechner-ähnliche Oberfläche. Mehrere Eingabegrößen sind (bei einer Ergebnisgröße) möglich.

Bild 1.13
Programmiersymbol Express-VI: Formel



1.2.6 Formelknoten

Der Formelknoten (Bild 1.14) ermöglicht umfangreiche Berechnungen und komplizierte Funktionsbeschreibungen in Formelschreibweise. Dabei können auch Programmstrukturen wie Fallunterscheidungen (Verzweigung, Alternation) oder Wiederholungen (Schleife, Iteration) verwendet werden.

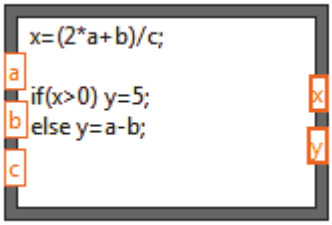


Bild 1.14
Formelknoten

Die Anweisungen werden von oben nach unten abgearbeitet, ein Bezug auf bereits oben berechnete Werte ist möglich. Für alle Ausgabegrößen muss auch eine Wertzuweisung vorgenommen werden. Die Ausgabe erfolgt erst, wenn alle Berechnungen abgeschlossen sind.

ACHTUNG
Ausdrücke müssen mit Semikolon abgeschlossen werden. Zwischenergebnisse können zur weiteren Berechnung verwendet werden. Zu beachten ist die Schreibweise für Potenz (**). Dezimalzahlen müssen mit Dezimalpunkt geschrieben werden.

1.2.7 Komplexe Zahlen

Eine komplexe Zahl z lässt sich durch ein geordnetes Paar reeller Zahlen darstellen. Entweder werden Realteil „re“ und Imaginärteil „im“ angegeben oder es werden Betrag „r“ und Winkel „θ“ verwendet. Das erste Element im Menü „Komplex“ erzeugt die konjugiert-komplexe Zahl (mit negiertem Imaginärteil). Die weiteren dargestellten Blockelemente dienen der Umwandlung der Formate (Bild 1.15).

Im Datentyp „Complex“ wird die Form Realteil und Imaginärteil (auch bei den Bedien- und Anzeigeelementen) als Voreinstellung verwendet.

Bild 1.15 *Menü „Komplex“*

Die Blockelemente für Verknüpfungen und Funktionen erkennen den angelegten Datentyp und führen die entsprechenden Operationen aus (polymorphe Funktionen; Bild 1.16).

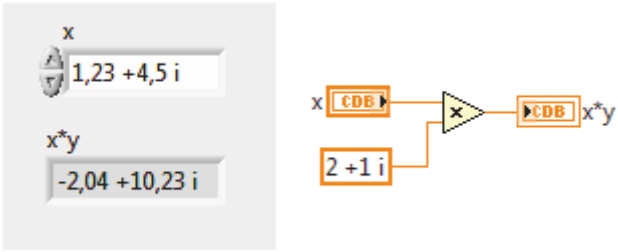


Bild 1.16 *Verknüpfung von Zahlen des Datentyps Complex CDB*

INFO: POLYMORPHE FUNKTIONEN

Polymorphe Funktionen sind als Sub-VIs realisiert, die auch selbst erstellt werden können. Bei polymorphen Funktionen ist für jeden gewünschten Datentyp eine eigene Programmversion erstellt. Diese lassen sich zu einer polymorphen Version zusammenbinden. Beim Aufruf des Sub-VIs wird der am Eingang angeschlossene Datentyp erkannt. Das Sub-VI wählt dann die zugehörige Programmversion aus und führt die zugehörige Berechnung durch.

1.2.8 Zeitfunktionen

Die Zeit wird als eigenständige Signalart (Zeitstempel) geführt. Dabei werden die Sekunden seit Freitag, 01. Januar 1904 12.00 Uhr Weltzeit gezählt. Mit dem Blockelement „Sekunden nach Datum/Zeit" wird ein Cluster aus numerischen Werten erzeugt. Auf diese Elemente kann z.B. mit dem Cluster-Element „Nach Namen aufschlüsseln" zugegriffen werden.

Zur Steuerung zeitlicher Abläufe in Schleifen und Sequenzen können die ersten drei Elemente aus dem Menü „Timing" verwendet werden:

- Timer-Wert (ms)
  gibt die aktuelle Zeit in s aus;
- Warten (ms)
  startet die Wartezeit nach der Ausführung aller Elemente;
- Bis zum nächsten Vielfachen von ms warten
  startet die Wartezeit vor der Ausführung aller Elemente.

Das Sub-VI „Datum/Zeit in Sekunden ermitteln" liefert die Informationen, die mit dem Sub-VI „Sekunden nach Datum/Zeit" in die angezeigten Einzelwerte aufgeschlüsselt werden können (Bilder 1.17 bis 1.19).

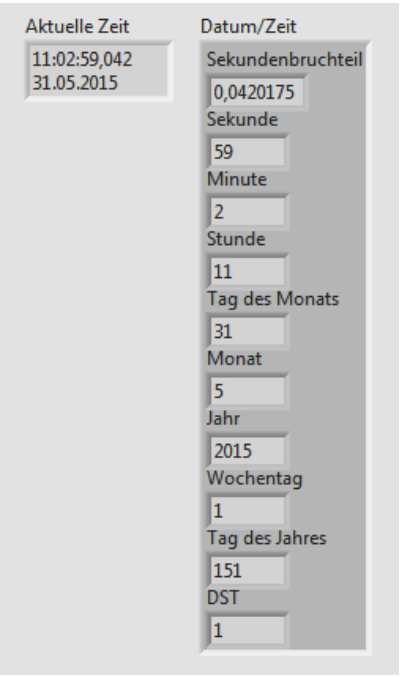


Bild 1.17 *Frontpanel zur „Zeit"*

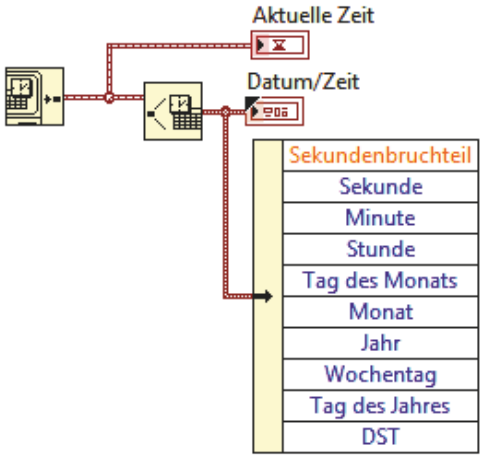


Bild 1.18 *Elemente des Zeitstempels*

Bild 1.19 *Elemente des Menüs „Timing"*

1.2.9 Array (numerisch)

Ein eindimensionales numerisches Array (Bild 1.20) ist eine Anordnung von Speicherstellen in einer Zeile oder in einer Spalte (Darstellung im Frontpanel beliebig). Bei zwei Dimensionen wird jede Zelle durch die Angabe von Zeilen- und Spaltennummer adressiert. Die Nummerierung beginnt immer bei null.

Für die Verarbeitung von Messdaten sind die Sub-VIs „Max.&Min.vi", „1D-Array interpolieren.vi" und „Schwellwert.vi" (in der vierten Zeile) hervorzuheben.

Mit dem Sub-VI „1D-Array interpolieren.vi“ wird zum eingegebenen Wert eine lineare Interpolation im zugehörigen Intervall ausgeführt. Im einfachsten Fall steht eine Folge von y-Werten im Eingangs-Array. Der Index bildet den jeweils zugehörigen x-Wert. Bei zweidimensionalen Daten muss jeder Datenpunkt (nach den x-Werten sortiert) als Cluster im Array stehen (siehe Beispiel unter Abschnitt 3.4.1).

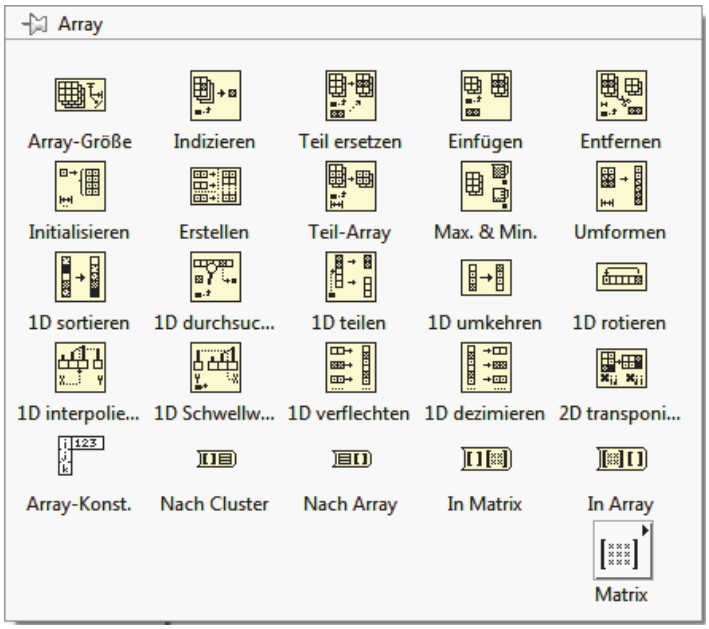


Bild 1.20 *Funktionsmenü „Array“*

Mit dem Sub-VI „Schwellwert.vi“ wird zum eingegebenen Wert eine Zahl ausgegeben, deren ganzzahliger Anteil die Nummer des Intervalls angibt und der gebrochene Anteil die Lage innerhalb des Intervalls. Im zweidimensionalen Array mit Daten muss jeder Datenpunkt (nach den x-Werten sortiert) als Cluster festgelegt sein.

Im Untermenü „Matrix“ findet man die Funktionen, bei denen einzelne Zellen einer Matrix ausgewählt, umgeordnet oder gezählt werden (siehe Abschnitt 1.2.11). Rechenoperationen finden sich unter „Mathematik – Lineare Algebra“.

1.2.10 Array (boolesch, binär)

In den ersten beiden Zeilen aus „Mathematik – Boolesch“ befinden sich die Elemente für paarweise Verknüpfungen, Negation und Implikation. Die Mehrfacharithmetik kann jeweils eine der angebotenen Funktionen für mehrere Eingänge ausführen. Besonders hilfreich ist die Möglichkeit, einzelne Eingänge oder den Ausgang direkt zu invertieren. Für Boolesche Arrays sind die bitweisen Verknüpfungen mit UND bzw. ODER vorbereitet (Bild 1.21).

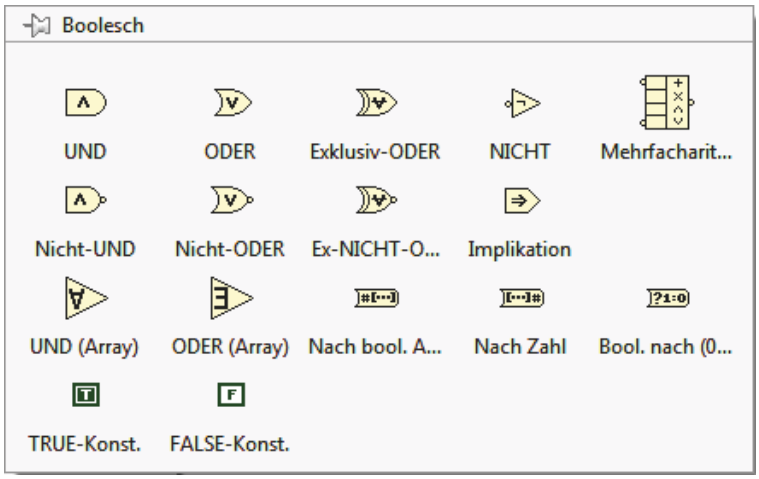


Bild 1.21 *Menü Funktionen mit Booleschen Arrays (dritte Zeile)*

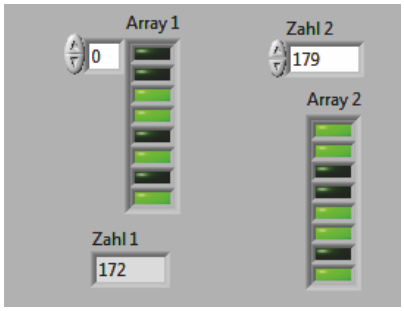


Bild 1.22 *FP „binaer-zahl.vi“*

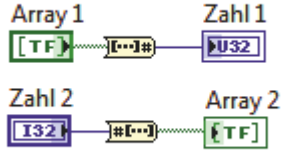


Bild 1.23 *BD „binaer-zahl.vi“*

Die Elemente im Booleschen Array sind logische Werte (Bild 1.22). Ein Boolesches Array kann als binäre Codierung einer Zahl verstanden werden. Eine Konvertierung ist in beide Richtungen möglich. Die numerische Seite kann über „Eigenschaften“ in verschiedenen Zahlenbasen dargestellt werden.

Im Beispiel „binaer-zahl.vi“(Bild 1.23) befindet sich das höchstwertige Bit unten. Die bei den numerischen Arrays dargestellten Funktionen lassen sich auch bei den Booleschen Arrays anwenden (Bild 1.24). Es sind aber nicht alle sinnvoll.

1.2.11 Matrix

Im Unterschied zu den Arrays werden die Zellinhalte bei den Matrizen im Sinne der Linearen Algebra interpretiert (Bild 1.26). Für die Berechnung mit Matrizen stehen alle gängigen Rechenoperationen und Verfahren zum Umgang mit speziellen numerischen Problemstellungen zur Verfügung („Mathematik – Lineare Algebra“).

Mit dem dargestellten Beispiel „lgs.vi“ (Bild 1.25) lässt sich ein lineares Gleichungssystem lösen.

Bild 1.24 *FP „lgs.vi“ – Lösung eines linearen Gleichungssystems*

Bild 1.25
BD „lgs.vi“

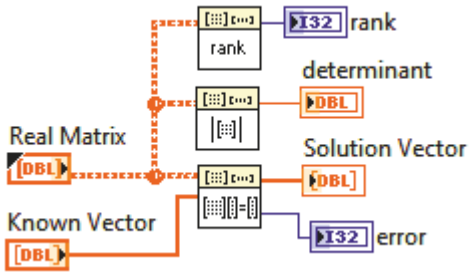
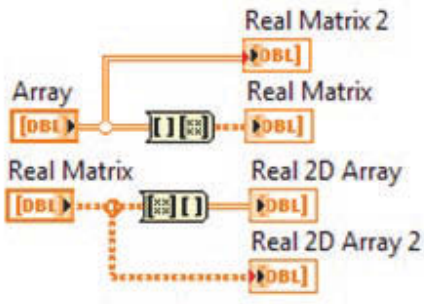


Bild 1.26
Array und Matrix



Eine Matrix bzw. ein Vektor stellt je einen eigenen Datentyp dar, der auf einem Array aufgebaut ist. Matrix-Operationen akzeptieren eindimensionale numerische Arrays als Vektoren.

Matrizen und numerische Arrays lassen sich ineinander umwandeln (Bild 1.27). Sie werden auch bei der Anzeige automatisch in den entsprechenden Typ konvertiert. Arrays und Matrizen unterscheiden sich jedoch in der Art der mathematischen Operationen.

Ein weiteres umfangreiches VI aus den LabVIEW-Beispielen innerhalb des Programmpaketes ist „Linear Algebra Calculator.vi“.

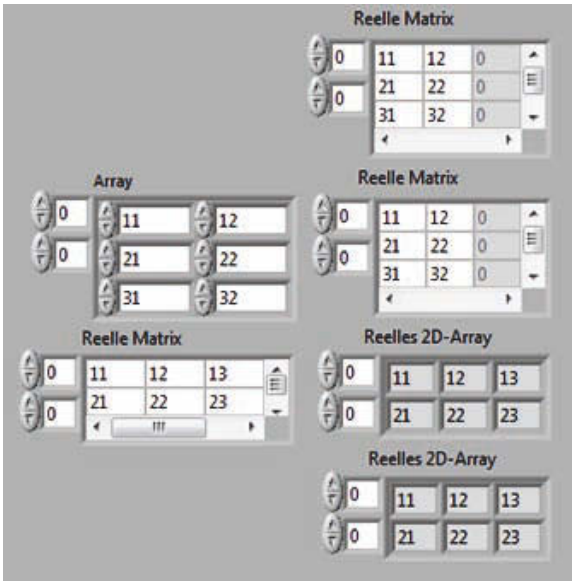


Bild 1.27
Umwandlung Array–Matrix

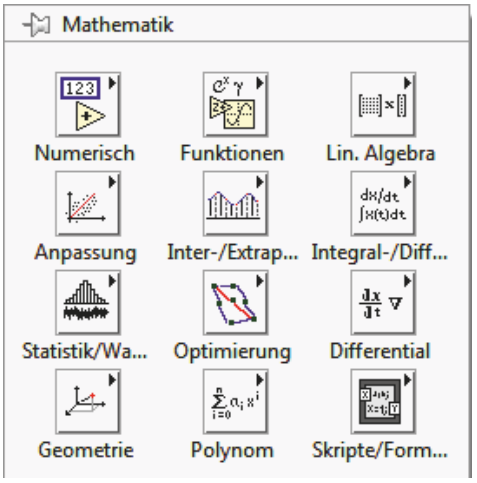


Bild 1.28
Menü „Mathematik“

INFO: WEITERE MATHEMATIK-SUB-VIS

Die umfangreiche Sammlung an Sub-VIs zu mathematischen Funktionen und Operationen sind im Kontext-Menü unter „Mathematik“ zu finden.
Ein großer Teil davon findet direkt Anwendung bei der Analyse von Messdaten. Dieser Teil ist der zentrale Inhalt dieses Buches.
Der andere Teil widmet sich allgemeineren Verfahren aus der numerischen und auch der symbolischen Mathematik und kann nur auszugsweise dargestellt werden.

1.2.12 Scriptsprachen

Beim Skriptknoten (Bild 1.29) wird die MATLAB®-Software aufgerufen und ein MATLAB-Skript ausgeführt (nur unter WINDOWS® mit ActiveX; das Programm MATLAB muss installiert sein). Über den Rand des Knotens werden Daten an den Script-Server übergeben. Dieser führt das Script aus. Der Knoten enthält das Skript (Bild 1.30).

INFO

Ähnliche Funktionalität besitzt das freie Programm Scilab. Auf der Homepage ni.com werden unter „Community" Beispiele zum Download angeboten. Mit der Suche „Scilab" kann von der Seite
http://search.ni.com/nisearch/app/main/p/bot/no/ap/tech/lang/de/pg/1/sn/catnav:ex,n6:SignalProcessingAnalysis/q/scilab/
der Scriptknoten für die Scriptsprache „Scilab" heruntergeladen werden. Nach der Installation erfolgt der Zugriff über das dargestellte Menü. Die Software Scilab muss auf dem Rechner installiert sein. Der Knoten ruft den Script-Server auf und führt das Skript aus.

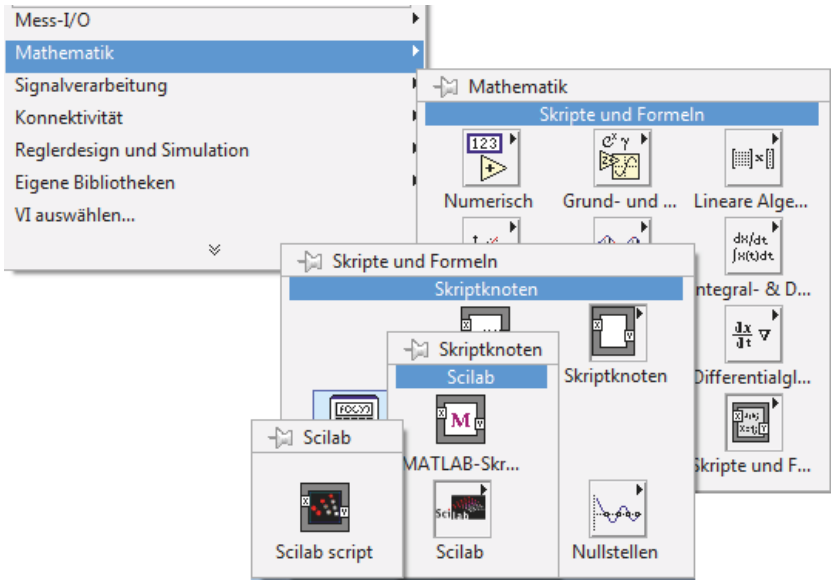


Bild 1.29 *Menü zum Scriptknoten für Scilab*

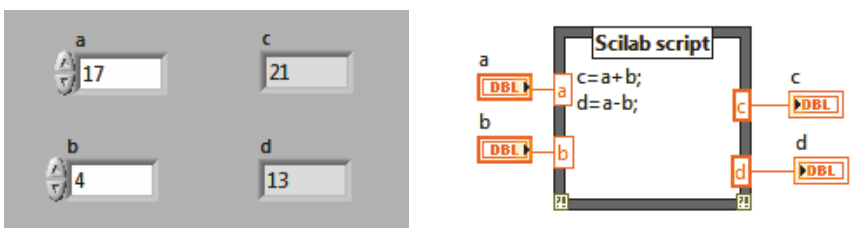


Bild 1.30 *FP und BD „skriptknoten.vi" – Einfache Verknüpfung mit Scilab*

1.3 Signalarten

1.3.1 Werte und Signale

Werte (oder Inhalte) von Programmelementen in LabVIEW sind Zahlen, logische Werte, Zeichenketten, Zeitstempel oder Pfade im Dateisystem. Im Diagrammfenster sind die zugehörigen Ein- und Ausgabe-Elemente oder die entsprechenden Elemente für Konstanten und Variablen durch Farben unterschieden. Ihrem Wert entspricht jeweils ein Bereich im Arbeitsspeicher. Die zugehörigen Signalleitungen sind ebenfalls farbcodiert. Transportierte Werte oder gespeicherte Wertefolgen sind Signale.

1.3.2 Grundtypen

Bild 1.31 *Zeichenkette (string)*

Zeichenketten (rot; Bild 1.31)
Zeichenketten (*strings*) enthalten beliebige Zeichen der in LabVIEW zur Verfügung stehenden Zeichensätze (einschließlich der möglichen Steuerzeichen).

Einzelwerte numerisch (orange, blau, Bild 1.32)

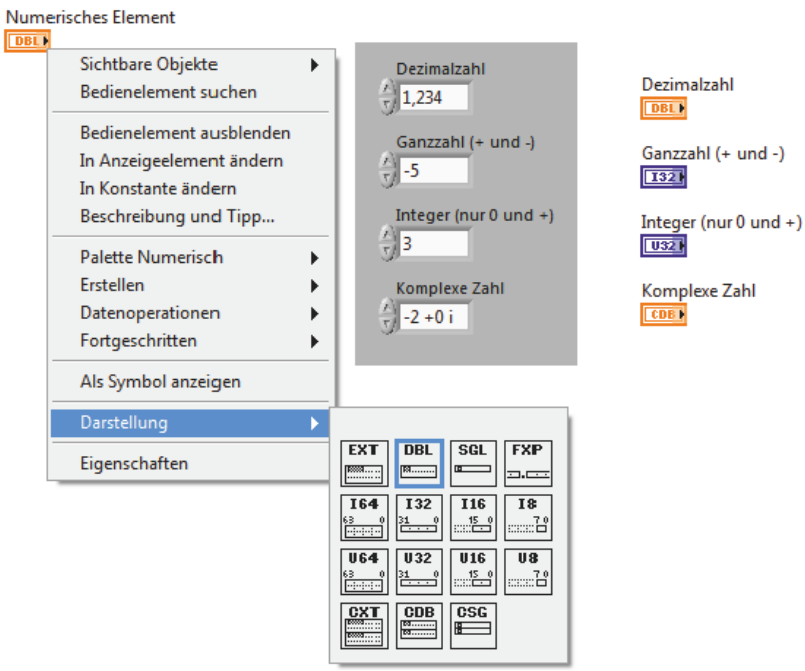


Bild 1.32 *Menü zu den Zahlenformaten bei numerischen Daten*

Dezimalzahlen mit unterschiedlicher Stellenzahl benötigen im Vergleich zu Ganzzahlen und Integer-Zahlen am meisten Speicherplatz. Bei gleichem Datentyp (z.B. Darstellung SGL, DBL, I32, ...) können in der Regel unterschiedliche Anzeigeformate gewählt werden (z.B. Fließkomma, Zehnerpotenz, Zeitformat bzw. dezimal, oktal, hexadezimal, binär).

Komplexe Zahlen werden mit Real- und Imaginärteil dargestellt.

Einzelwerte logisch (grün; Bild 1.33)

Bild 1.33 *Logisches Eingabe-Element*

Logik-Signale können nur die Werte „True" (wahr, 1) oder „False" (falsch, 0) annehmen. Logische Signale lassen sich mit Logik-Funktionen verknüpfen.

Array

In einem Array (Feld) werden gleichartige Werte (Inhalte) zusammengefasst. Zur Zusammenführung bzw. zur Auftrennung gibt es jeweils ein Blockelement, das auf mehrere Kanäle erweiterbar ist (Bilder 1.34 und 1.35).

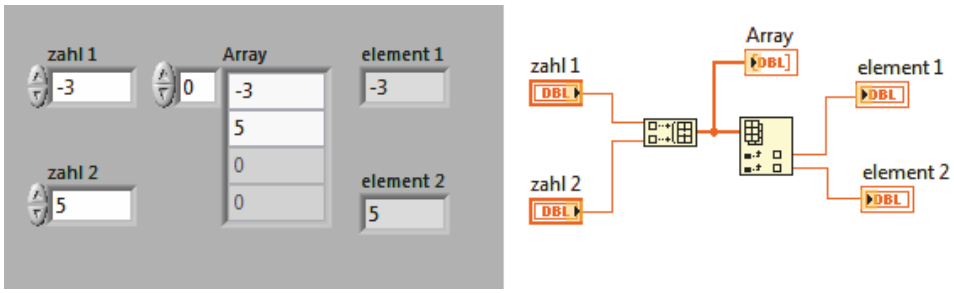


Bild 1.34 *Array mit zwei numerischen Elementen*

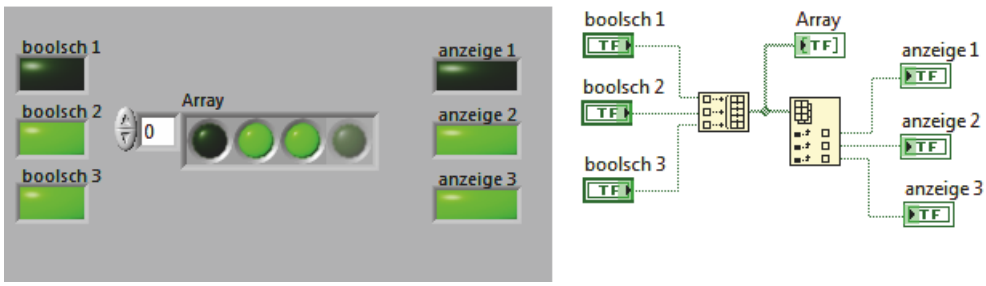


Bild 1.35 *Array mit drei logischen Elementen*

Signalverlauf (Waveform) analog (braun)

Ein Signalverlauf stellt ein Zeitsignal dar, das in gleichen Zeitabständen (äquidistant) abgetastet wurde. Für die vollständige Information müssen daher neben den Signalwerten der Startzeitpunkt

und das Taktintervall angegeben werden. Die Signalwerte sind numerische Größen in einem beliebigen numerischen Format (Bild 1.36).

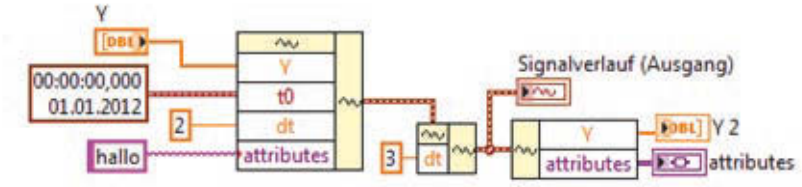


Bild 1.36 *Signalverlauf (Waveform) mit Zeitstempel und Attribut*

Der Signalverlauf kann eine zusätzliche Information in einer beliebigen Datenform als Attribut mit sich führen (Beispiele: Auftragsnummer, Mitarbeiter, Maschinenkennung). Es sind mehrere Attribute möglich (z.B. Einzelwert, Array, Cluster).

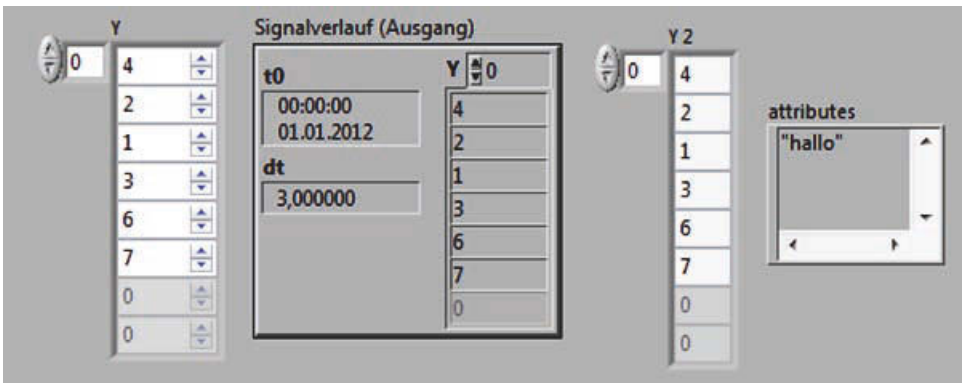


Bild 1.37 *Analoger Signalverlauf Ein-und Ausgabe*

Das Beispiel in Bild 1.37 zeigt die Zusammenführung der einzelnen Elemente zum analogen Signalverlauf (Waveform). Mit dem gleichen Element lassen sich Teile eines vorhandenen Signalverlaufs ändern. Die einzelnen Komponenten können auch wieder in einzelne Variablen getrennt werden.

Mit dem Signalverlauf sind spezielle Verknüpfungen möglich (z.B. aneinander hängen, werteweise multiplizieren). Der Signalverlauf ist eine spezielle Form eines Clusters.

Signalverlauf (Waveform) digital (grün; Bild 1.38)

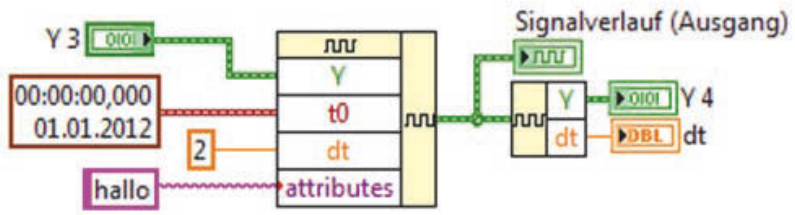


Bild 1.38 *Aufbau eines digitalen Signalverlaufs*