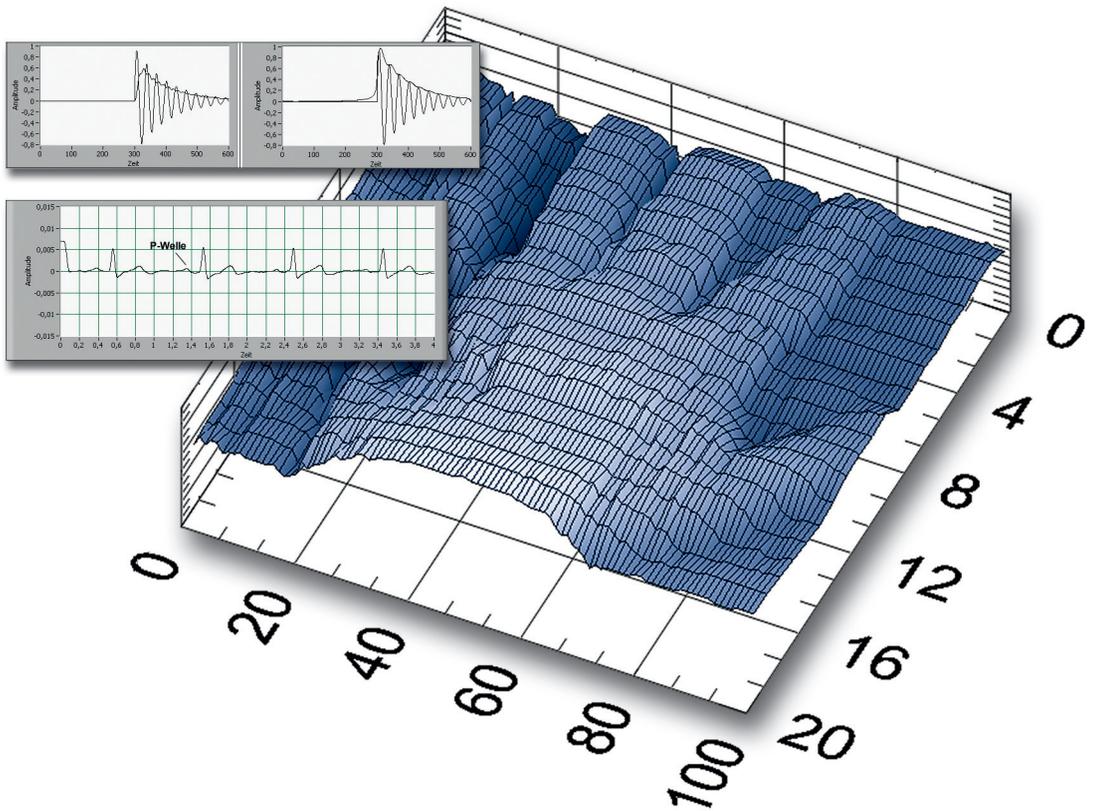


Friedrich Plötzeneder / Birgit Plötzeneder



Praxiseinstieg **LabVIEW**

Eine Einführung in die Praxis
in 12 Experimenten

Vorwort

In jeder Firma, jeder Schule oder jedem Institut gibt es jemanden, der jedem hilft und alles kann. Für alle Sonderfälle oder Probleme hat er (oder sie) den passenden Kleber, eine Bezugsquelle oder sogar schon eine Lösung in der Schublade.

LabVIEW ist dieser Praktiker in der PC-Messtechnik. Es hat viele Algorithmen, ist leicht programmierbar und kann ausgezeichnet mit externer Hardware kommunizieren.

Das vorliegende Buch ist in zwei Abschnitte gegliedert:

Der erste Teil des Buchs, Kapitel 1–19, erklärt die Programmiersprache LabVIEW Schritt für Schritt. Die Kapitel sind mit instruktiven Beispielen abgeschlossen. Aufgrund des Umfangs von LabVIEW, das mehr als tausend ausprogrammierte Algorithmen hat, sind nur die wichtigsten und Sprachelemente besprochen.

Der zweite Teil des Buchs, Kapitel 20–30, zeigt, wie leicht man mit LabVIEW technische, physikalische oder mathematische Probleme unterschiedlichster Bereiche lösen kann. Die Experimente sind mit einfacher, oft bereits vorhandener Ausrüstung möglich. Sie können zu Hause oder im Labor einer Schule oder Hochschule ohne große Investition durchgeführt werden. Sie sind bezüglich der Durchführung und Mathematik vollständig beschrieben und erprobt.

Die Experimente sind als Laborübung an Hochschulen oder HTLs und für Lehramtsstudenten, Ingenieure und Autodidakten, die mit LabVIEW arbeiten, geeignet.

Dateien zum Buch können von der Website www.ploetzeneder-labview.com heruntergeladen werden. E-Mails an f.ploetzeneder@fh-wels.at oder b.ploetzeneder@gmail.com.

Geleitwort

Am Campus Wels der Fachhochschule Oberösterreich haben wir seit nunmehr über 15 Jahren die Software-Entwicklungsumgebung *LabVIEW* im Studiengang *Automatisierungstechnik* erfolgreich im Einsatz und schon sehr viele Praktiker für unseren Wirtschaftsraum damit ausgebildet. Bis heute ist meine persönliche Begeisterung als Lehrender für die *LabVIEW* charakterisierende intuitive, grafisch-visuelle Programmierung ungebrochen. Gemeinsam mit dem Autor des vorliegenden Lehrbuchs gelingt es uns in Wels, diese Begeisterung an Studierende weiter zu vermitteln.

Die Mächtigkeit von *LabVIEW*, der Umfang an wiederverwendbaren Programmbeispielen und Dokumentationen, aber auch der jährliche „Zuwachs“ an neuer Funktionalität und Leistungsfähigkeit ist enorm. Das macht es Anfängern schwer, sich zu orientieren. Aber auch für erfahrene Praktiker, die nicht tagtäglich mit *LabVIEW* arbeiten, ist es nicht leicht, auf dem Laufenden zu bleiben. Vor diesem Hintergrund sehe ich in diesem Buch eine große Bereicherung. Es liefert einem fundierten und kompakten Einstieg und enthält eine große Anzahl konkret anwendbarer Beispiele guten Programmierstils.

In einer didaktisch ausgereiften Form wird dem Leser eine umfassende Einführung in die grafisch-visuelle Programmierung mit *LabVIEW* geboten. Ausgehend von elementaren Begriffen werden alle Kenntnisse vermittelt, die zur Lösung umfangreicher Problemstellungen notwendig sind. Das Buch ist für Anfänger ohne besondere Vorkenntnisse geeignet. Es bleibt dabei aber nicht an der Oberfläche, sondern vermittelt auch fortgeschrittene Programmiermethoden und enthält nützliche Laborexperimente als konkrete Beispiele. Selbst erfahrene *LabVIEW*-Anwender können mithilfe des Buchs Neues hinzulernen.

Positiv aufgefallen sind mir die sorgfältig erstellten realen Laborexperimente. Sie sind dank ihrer Übersichtlichkeit nicht nur leicht verständlich, sondern vermitteln auch die Funktionsvielfalt von *LabVIEW* anschaulich.

FH. Prof. Univ. Doz. Dipl. Ing. Dr. Karl Kellermayr

Fachbereichsleiter Informationstechnologie

Campus Wels

Fachhochschule Oberösterreich

Inhaltsverzeichnis

1	Einführung	13
1.1	System installieren	13
2	Grundlegendes	14
2.1	Frontpanel und Blockdiagramm	14
2.2	Die fünf wichtigsten Fenster und das Fehlerfenster	15
2.3	Details der wichtigsten Fenster	17
3	Datentypen und ein erstes Programm	20
3.1	Numerische Datentypen	20
3.1.1	Erzeugen eines Eingabelements	21
3.1.2	Darstellung der Symbole der Frontpanelemente im Blockdiagramm	23
3.1.3	Einfache Berechnungen	23
3.2	Strings oder Zeichenketten	24
3.3	Boolesch	25
3.4	Weitere Datentypen	25
3.5	Umwandlung von Daten in einen anderen Datentyp	26
3.6	Konstanten	27
3.7	Hallo, Welt!	28
3.7.1	Hinweis zur Drahtspule	29
4	Einfache Frontpanelemente	30
4.1	Umwandlung von Bedien-/Anzeigeelementen und Konstanten	30
4.2	Konfiguration numerischer Frontpanelemente	31
4.3	Skalierung von Anzeigeelementen	34
4.4	Listenfelder	35
4.5	Konfigurieren eines Schalters	36
4.5.1	Schaltverhalten	36
4.5.2	Boolescher Text	37
5	Entscheidungsstrukturen	38
5.1	Funktion <i>Auswählen</i>	38
5.1.1	Auswählen in einem Programm	39
5.1.2	Beispiel zur Funktion <i>Auswählen</i>	40
5.2	Funktion einer Case-Struktur	40
5.3	Case-Struktur für boolesche Werte	42
5.4	Case-Struktur für numerische Daten	43
5.5	Case-Struktur für Strings	45
5.6	Häufige Fehler bei Case-Strukturen	46

5.6.1	Syntaxfehler.....	46
5.6.2	Fehlerhafter Case	46
5.6.3	Keine Ausgangsgröße.....	46
6	Arrays	48
6.1	Eindimensionale Arrays	48
6.1.1	Verändern der Array-Darstellung	50
6.2	Array-Funktionen (eindimensional)	51
6.2.1	Array-Länge	52
6.2.2	Array indizieren	52
6.3	Mehrdimensionale Arrays	53
6.3.1	Arrays verknüpfen	53
6.3.2	Arrays beliebigen Typs	55
6.3.3	Zweidimensionales Array im Frontpanel	56
6.4	Array-Funktionen (mehrdimensional)	57
6.4.1	Transponieren	57
6.4.2	Array indizieren	58
6.4.3	Rechnen mit einem Array	58
7	Cluster	60
7.1	Erstellen und Zerlegen eines Clusters	60
7.2	Cluster als Frontpanelement	61
7.3	Ändern einzelner Werte in einem Cluster	62
7.4	Umwandlung von Cluster in Arrays und Arrays in Cluster	63
7.5	Error-Cluster	63
8	Strings	65
8.1	Eingabe und Ausgabe	65
8.1.1	Verschiedene Darstellungsformen von Strings	66
8.2	String-Funktionen	67
8.2.1	String-Länge und String verknüpfen	67
8.2.2	Muster suchen und String suchen und ersetzen.....	68
8.2.3	Teil-String.....	70
8.3	Praxisanwendungen	70
8.3.1	Auswertung eines typischen Messgeräte-Strings	70
8.3.2	String-Format für Excel-Datei	72
9	Schleifen	74
9.1	For-Schleifen	74
9.2	While-Schleife	76
9.3	Die For-While-Schleife	77
9.4	Ein- und Ausgabe in Schleifen	78
9.5	Beispiele	80
9.5.1	Grafische Ausgabe von Arrays	80
9.5.2	Summe der arithmetischen Reihe	81

9.5.3	Erstellen einer ASCII-Tabelle	82
9.5.4	Aufstellen einer Einheitsmatrix	83
10	Schieberegister	84
10.1	Grundelemente des Schieberegisters	84
10.2	Highlight-Modus	85
10.3	Nicht initialisierte Schieberegister	86
10.4	Gestapelte Schieberegister	86
10.5	Schieberegister in Form von Rückkopplungsknoten	87
10.6	Beispiele	88
10.6.1	Flankenerkennung	88
10.6.2	Summenzeichen auswerten	89
10.6.3	Erstellen eines Arrays mit allen geraden Zahlen von 2 bis 100	89
11	Unterprogramme	91
11.1	Erstellen eines Unterprogramms	91
11.2	Automatisches Erstellen eines Unterprogramms im Hauptprogramm	95
11.3	Modi beim Aufrufen eines Unterprogramms	95
11.3.1	Frontpanel des Unterprogramms bei Aufruf öffnen	96
11.3.2	Ablaufinvariante Ausführung eines Unterprogramms	97
12	Grafische Frontpanelemente	100
12.1	Signaldiagramm und Signalgraph	100
12.1.1	Skalierung der Y-Achse	103
12.1.2	Skalierung der X-Achse	103
12.1.3	Darstellung von zwei oder mehreren Kurvenzügen	104
12.2	XY-Graph	105
12.2.1	XY-Graphik für mehrere Graphen	107
12.3	Anwendung: Signaldarstellung mit Zeitstempel	108
12.4	Eigenschaftsknoten	109
12.5	3-D-Graphen	111
13	Grafik	113
13.1	Elementare Funktionen	113
13.2	Auslesen eines JPEG-Bilds	115
13.3	Frontpanelement in eine Grafik konvertieren und abspeichern	116
14	Datenerfassung	118
14.1	Das Multifunktionsgerät USB-6008	118
14.1.1	Installation des Measurement & Automation Explorers und der Treiber	119
14.1.2	Measurement & Automation Explorer	120
14.2	Ein- und Ausgabe mit USB-6008 und LabVIEW	125
14.3	Der DAQ-Assistent in einfachen Anwendungen	129
14.3.1	Oszilloskop	129

14.3.2	Spektralanalysator	130
14.3.3	Digitalausgabe	130
14.4	Weiterführende Messungen	132
14.4.1	Messung einer Transistorkennlinie	133
14.4.2	Sprungantwort eines RC-Glieds	134
14.4.3	Datenerfassung mit Streaming	138
14.5	Das wichtigste Problem beim Umgang mit Messkarten	139
15	Dateien	141
15.1	Schreiben im Excel-Format	141
15.1.1	Dateinamen	142
15.1.2	Erstellen eines absoluten Dateinamens aus einem relativen Dateinamen	144
15.1.3	Lesen einer einfachen Excel-Datei	145
15.1.4	Erstellen einer einfachen Excel-Datei	146
15.2	Beispiel	148
15.3	Speichern von Daten mit Express Vi.	149
15.4	Binärdateien schreiben und lesen	151
15.4.1	Unterschied zwischen einer Binärdatei und einer ASCII-Datei	151
15.4.2	Schreiben in eine Binär- und einer ASCII-Datei	152
15.4.3	Lesen einer Binärdatei und eine ASCII-Datei	153
15.4.4	Weitere Dateiformate	153
15.5	Ergänzungen und weitere Funktionen	153
16	Programmablauf	155
16.1	Normale Programmausführung	155
16.1.1	Parallelbearbeitung von Schleifen	156
16.2	Ablaufsteuerung über gültige Daten – Datenflusssteuerung	157
16.3	Ablaufsteuerung mit Sequenzstruktur	158
16.3.1	Lokale Sequenz-Variablen	159
16.3.2	Lokale Variable	160
16.3.3	Beispiel: Die Eieruhr	160
17	Soundkarte	161
17.1	Test der Soundkarte	161
17.2	Ausgabe einer Sinusschwingung	162
17.2.1	Umwandlung von Datentypen, die für die Soundkarte geeignet sind	163
17.3	Beispiele	164
17.3.1	Phasenbeziehung zwischen zwei Tönen	164
17.3.2	Spektralanalysator für Töne	165
18	Serielle Schnittstelle	167
18.1	Format der seriellen Schnittstelle	167
18.2	Programmierung der Schnittstelle	168
18.3	Minimalverdrahtung von zwei Rechnern zur Datenübertragung	169

18.4	Flusssteuerung mit Handshake	170
18.5	Ausgabe eines Strings an die RS-232-Schnittstelle	171
18.6	Einlesen eines Strings von der RS-232-Schnittstelle	171
18.7	Setzen der Handshake-Leitungen der seriellen Schnittstelle ...	172
19	Erstellung einer EXE-Datei und eines Installationsprogramms ..	174
19.1	Erstellung einer EXE-Datei	174
19.2	Erstellung eines Installationsprogramms	177
20	EKG	182
20.1	Hintergrund	182
20.2	Schaltung	182
20.3	Programm	184
20.4	Gemessenes EKG	184
21	Schrittmotoransteuerung	185
21.1	Hintergrund	185
21.1.1	Wave-Mode	185
21.1.2	Two-Phase-on-Mode	186
21.1.3	Half-Step-Mode	186
21.2	Versuchsaufbau	186
21.3	Programme	189
21.3.1	Wave-Mode	189
21.3.2	Two-Phase-on- und Half-Step-Mode	190
22	Drehstrom aus dem Laptop	192
22.1	Hintergrund	192
22.2	Versuchsaufbau	193
22.3	Programme	195
22.3.1	Ausgabe von zwei phasenversetzten Sinusspannungen	195
22.3.2	Drehstrom mit in feinen Stufen verstellbarer Frequenz	196
23	Dehnungsmessstreifen	198
23.1	Hintergrund	198
23.1.1	Umgang mit Störungen	199
23.2	Versuchsaufbau	199
23.3	Programm	200
24	Terminalprogramm	202
24.1	Version 1: Terminalprogramm mit zwei Threads	202
24.2	Version 2: Ereignis-/Eventgesteuertes Terminalprogramm	205
24.2.1	Aufbau: Ereignisstruktur in While-Schleife	205
24.2.2	Empfangen (Ereignis: <i>Zeichen lesen</i>)	206
24.2.3	Beenden (Ereignis: <i>Stopp</i>)	207
24.2.4	Senden (Ereignis: <i>String zu Senden</i>)	207

25	Signalverarbeitung in der Praxis	208
25.1	Interpolation von Daten	208
25.2	Störungen herausfiltern: Spikes und Rauschen	210
25.3	Maximumsuche	212
25.4	Bestimmung einer Einhüllenden	213
26	Akustisches GPS	215
26.1	Hintergrund	215
26.2	Versuchsaufbau	216
26.3	Versuchsanordnung	217
26.4	Programm	218
26.4.1	Ausgabe/Korrelation	218
26.4.2	Zeichnen der Hyperbel	219
27	Bildverarbeitung mit zweidimensionaler Fourier-Transformation	221
27.1	Hintergrund	221
27.2	Programm	223
27.2.1	Wirkung des Tiefpassfilters auf das Bild	223
27.2.2	Anmerkungen zur FFT	224
28	Temperaturverteilung in einem Ring	225
28.1	Hintergrund	225
28.2	Lösung mit FFT	225
28.3	Versuchsaufbau	229
28.4	Programm	230
29	3-D-Scanner mit Laptop und Beamer	232
29.1	Hintergrund	232
29.2	Versuchsaufbau	233
29.3	Programme	234
29.3.1	Bilder aufnehmen	234
29.3.2	Grafik erzeugen	235
30	Praktische Bildverarbeitung	237
30.1	Elementare Bildberechnungen: Bildrätsel überlisten	237
30.2	Anwenden von IMAQ: Münzen zählen	238
30.3	Kantenschärfung in einem Bild (Kernel-Operationen)	241
31	Literatur	243
32	Bauteile	244
	Die Autoren	246
	Stichwortverzeichnis	247

11 Unterprogramme

Unterprogramme sind ein wichtiges Konzept, um Komplexität in den Griff zu bekommen. Dabei trennt man Teile des Programms vom Gesamtprogramm und verwendet sie nur noch symbolisch. Sie kennen bereits viele Unterprogramme: Die meisten LABVIEW-Funktionen sind genau das! Die Nützlichkeit liegt nicht nur in der Übersichtlichkeit, sondern auch darin, dass man Programmteile sehr gut wiederverwerten kann.

11.1 Erstellen eines Unterprogramms

Eine wichtige Entwicklungsmethode für Software heißt *Bottom-up*. Bei dieser Methode wird das unterste Unterprogramm zuerst entwickelt und gründlich getestet. Danach werden die nächsten Ebenen der Unterprogramme programmiert, die auf die ersten zugreifen. Stellen Sie sich vor, sie müssten eine beliebig komplexe Funktion, die zwei Eingangsvariablen benötigt und zwei Werte ausgibt, in Ihr Programm einbauen. Eine sinnvolle Möglichkeit ist es, diese Funktion erst einmal fehlerfrei umzusetzen, und sie dann einfach einzufügen. In unserem Beispiel ist es nur eine einfache Addition/Subtraktion.

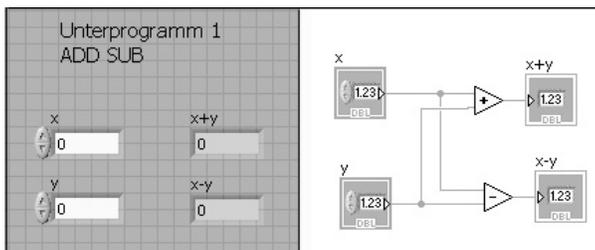


Abb. 11.1: Dieses Programm soll als Unterprogramm eingesetzt werden.

Wenn Sie dieses Programm jetzt als Unterprogramm wünschen, müssen Sie bestimmen, wie es im eigentlichen Programm (Hauptprogramm) aussieht. Dazu wählen Sie oben rechts im Frontpanel mit Rechtsklick *Symbol bearbeiten*.

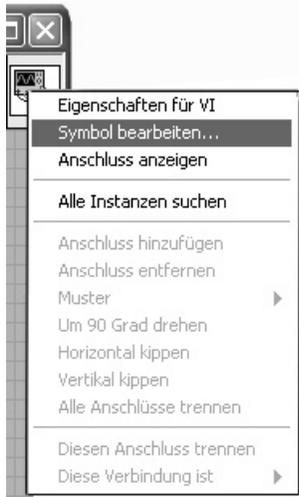


Abb. 11.2: Öffnen des Kontextmenüs am Symbol des Programms

In einer grafischen Programmiersprache hat ein Unterprogramm keinen Namen, sondern wird als Symbol eingesetzt. Gestalten Sie also ein neues Symbol. Wenn Sie Ihr Symbol erstellt haben, müssen Sie noch die Anschlüsse erstellen.

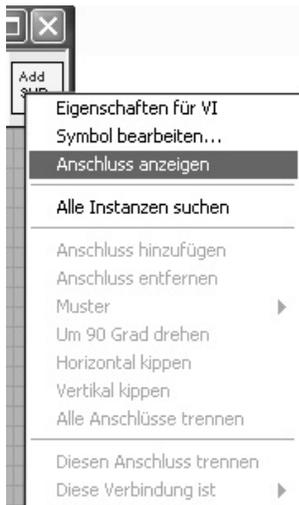


Abb. 11.3: Öffnen des Anschlussfensters

Sie sehen jetzt sehr viele Anschlussmöglichkeiten und benötigen genau vier Anschlüsse. Öffnen Sie das Kontextmenü und wählen Sie das Muster mit 4 Anschlüssen aus.

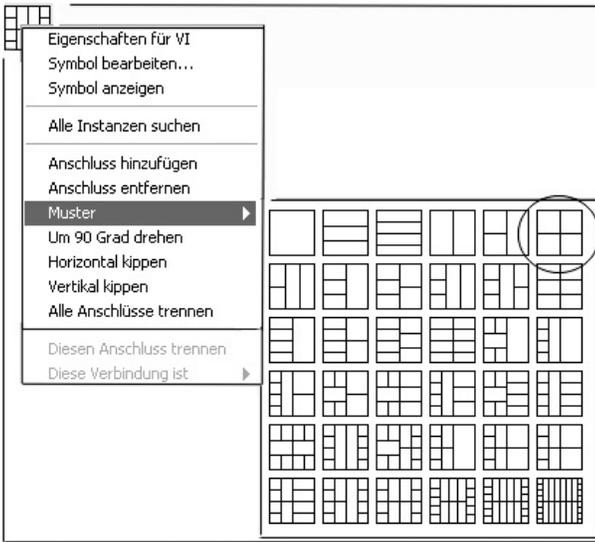


Abb. 11.4: Auswahl des Anschlussmusters

Schließen Sie die Frontpanelelemente am Anschlussmuster an. Klicken Sie das Frontpanel und das Anschlussmuster in der angegebenen Reihenfolge 1, 2, 3, 4, 5, 6, 7, 8 an. Sie wählen also jeweils die Variable und danach den entsprechenden Anschluss aus.

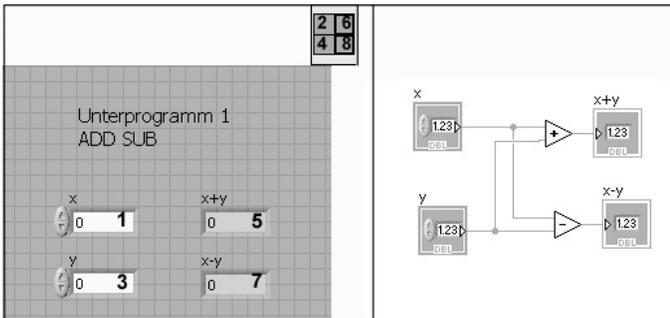


Abb. 11.5: Zuordnung von Frontpanelelementen und die Anschlüsse im Unterprogramm

Speichern Sie über *Datei >> Speichern unter* das Programm unter dem Namen *UP1.vi*. Erstellen Sie ein neues Programm mit zwei numerischen Eingaben und einem Zeigerinstrument.

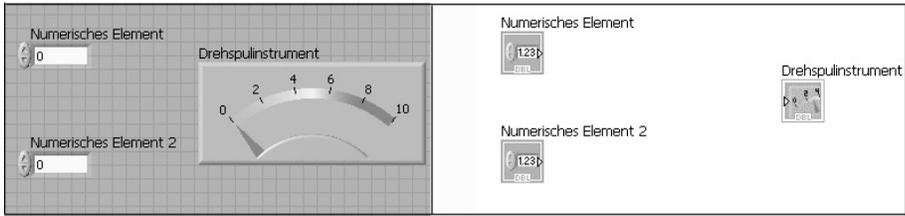


Abb. 11.6: Ein neues Programm wird das Hautprogramm.

Gehen Sie über das Blockdiagramm und öffnen Sie die Funktionspalette (rechte Maustaste):

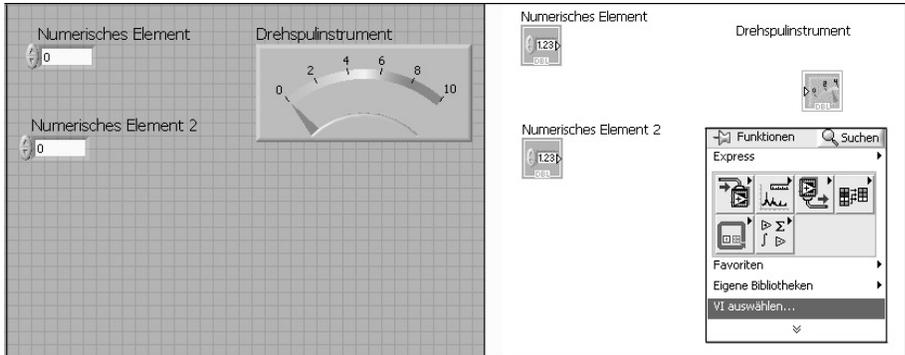


Abb. 11.7: Wählen des eigenen Unterprogramms in der Funktionspalette

Wählen Sie das Unterprogramm *UPI* aus und setzen Sie es in das Hauptprogramm ein.

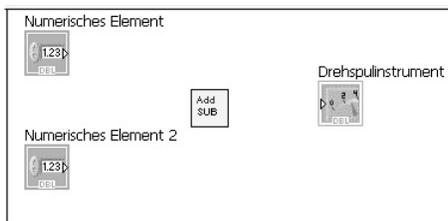


Abb. 11.8: Eingesetztes Unterprogramm

Verbinden Sie das Unterprogramm mit den Frontpanelementen und testen Sie das Programm.

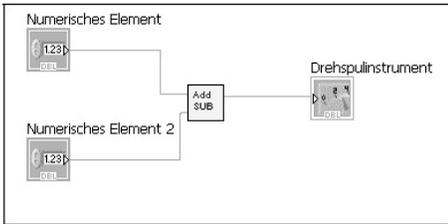


Abb. 11.9: Hauptprogramm mit Unterprogramm nach Verdrahtung

11.2 Automatisches Erstellen eines Unterprogramms im Hauptprogramm

Bei der Entwicklung eines LabVIEW-Programms kommt es vor, dass das Programm größer wird, als ursprünglich geplant. Schnell kommt der Wunsch auf, einen Teil des Hauptprogramms in ein Unterprogramm zu verwandeln. Dies wird in der Entwicklungsumgebung von LabVIEW auf folgende Weise unterstützt:

- Den Teil markieren, der in ein Unterprogramm verwandelt werden soll
- In der Symbolleiste *Bearbeiten* >> *SubVI erstellen* aufrufen

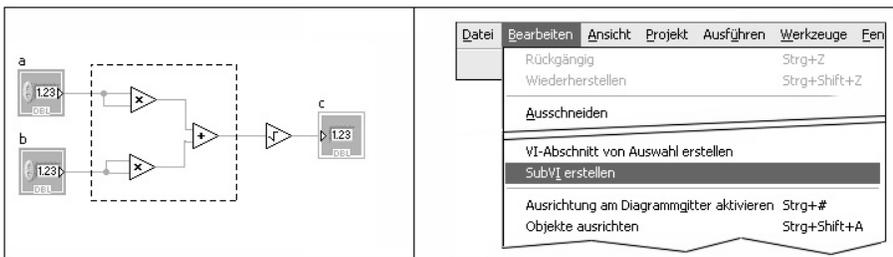


Abb. 11.10: Automatisches Erstellen eines Unterprogramms

Danach ist das Unterprogramm aus dem Hauptprogramm erstellt und schon in dieses eingesetzt.

11.3 Modi beim Aufrufen eines Unterprogramms

Das Aufrufen eines Unterprogramms kann auf unterschiedliche Arten durchgeführt werden. Beispielsweise können Frontpanels neu geöffnet werden (etwa für einen Anmeldedialog).

11.3.1 Frontpanel des Unterprogramms bei Aufruf öffnen

Beispiel:

Es soll ein Programm erstellt werden, das einen Anmeldedialog hat. Beim Start des Hauptprogramms soll das Frontpanel des Unterprogramms (der Dialog) geöffnet werden. Im Fenster des Unterprogramms kann der Name des Anwenders eingegeben werden. Nach einer Bestätigung soll das Fenster automatisch geschlossen werden und der String mit dem Namen ist vom Unterprogramm an das Hauptprogramm zu übergeben.

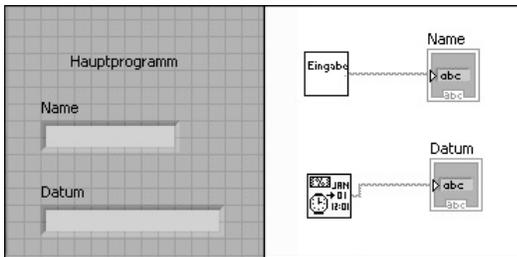


Abb. 11.11: Hauptprogramm; nach dem Start wird automatisch das Frontpanel des Unterprogramms geöffnet.

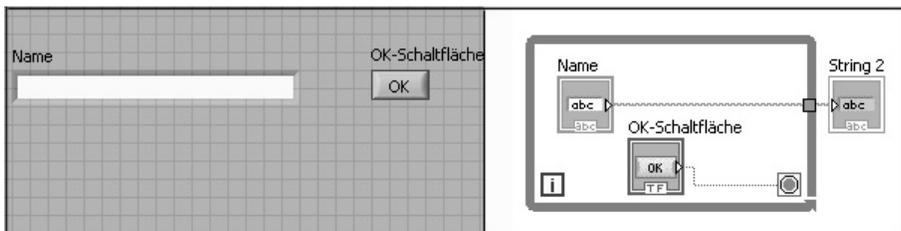


Abb. 11.12: Unterprogramm-Eingabe, das vom Hauptprogramm geöffnet wird.

In die String-Eingabe des Unterprogramms kann der Name eingegeben werden. Die *While*-Schleife wird so lange ausgeführt, bis auf den Button *OK* gedrückt wird. Danach wird das Unterprogrammfenster geschlossen und der String an das Hauptprogramm übergeben.

Wie wird erreicht, dass bei Aufruf des Unterprogramms das Frontpanel geöffnet und nach Ausführung wieder geschlossen wird? Dazu ist im Hauptprogramm mit der rechten Maustaste das Unterprogramm anzuklicken. Im entstehenden Kontextmenü kann man die *SubVI*-Einstellungen konfigurieren. Wie ist ein Unterprogramm zu konfigurieren, damit bei seinem Aufruf das Frontpanel geöffnet wird?

Tabelle 11.1: Konfiguration des Unterprogramms; Bedingung für das Öffnen des Frontpanels

Kontextmenü des Unterprogramms	Einstellung des Unterprogramms
<p>Konfiguration</p>	<p>a) Öffnen bei Aufruf b) Schließen bei Beendigung</p> <p>Öffnen</p>

11.3.2 Ablauffinvariante Ausführung eines Unterprogramms

Wird ein Unterprogramm mehrmals in ein Hauptprogramm eingesetzt, kann es zu einem besonderen Problem kommen. Soll beim Verlassen des Unterprogramms ein Wert einer Variablen erhalten bleiben, muss auch für jedes eingesetzte Unterprogramm ein Speicher bereitgestellt werden.

Spezialbeispiel aus der Elektronik:

Ein Binärzähler besteht aus zwei T-Flipflops. Jedes Flipflop soll in Form eines Unterprogramms realisiert werden.

Hinweis zum T-Flipflop: Der Ausgang Q ändert sich bei einer positiven Flanke am T-Eingang. *T-Eingang* steht dabei für „Trigger-Eingang“.

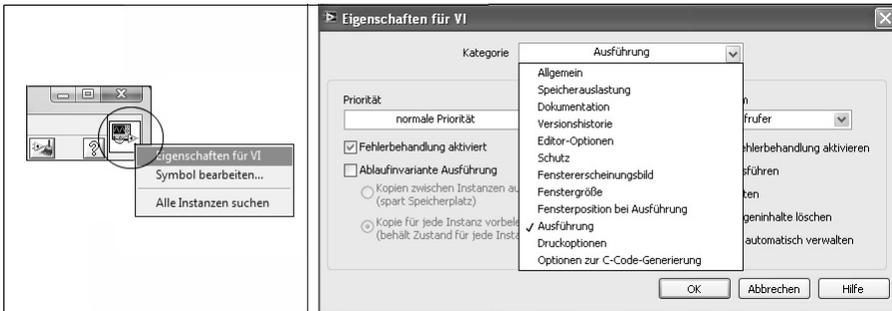


Abb. 11.15: Eigenschaften für VI und danach Ausführungsmodus wählen

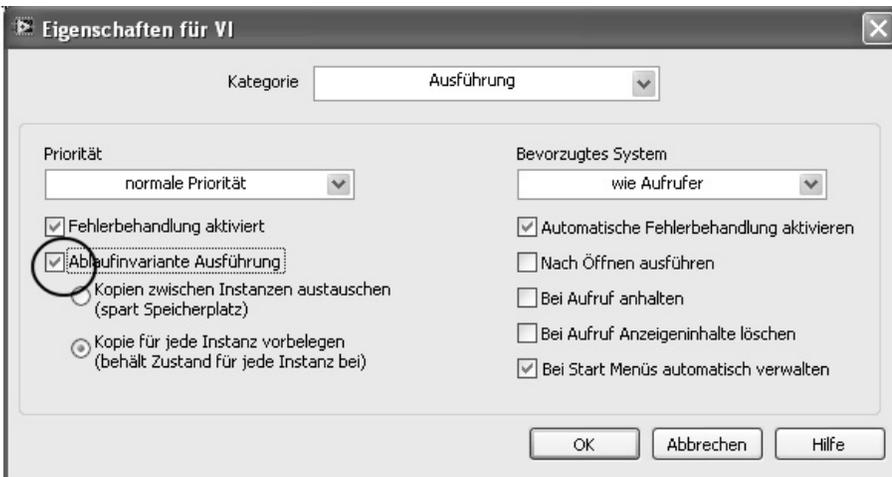


Abb. 11.16: Ablaufinvariante Ausführung wählen.

Ablaufinvariant (oder wiedereintrittsfähig) wird in der englischen Version als reentrant bezeichnet.

21 Schrittmotoransteuerung

Zur Ansteuerung eines Schrittmotors ist nur ein einziger Treiber-IC nötig, der direkt an die USB-6008 angeschlossen wird. Damit können seine Eigenschaften im Labor sehr gut untersucht werden.

21.1 Hintergrund

Der Vorteil eines Schrittmotors ist, dass er ohne Sensor eine genaue Position anfahren kann. Daher wird er vor allem für Präzisionsgeräte häufig eingesetzt. Im Gegensatz zum Gleichstrommotor hat er ein Haltemoment (wirkt also im Stillstand Bewegung von außen entgegen). Von der Wirkungsweise ist der Schrittmotor (englisch: *Stepper*) ein Synchronmotor, d. h., seine Bewegung ist zur Wechselspannung synchron. Meistens ist der Rotor (der sich bewegende Läufer) ein Permanentmagnet, aber es sind auch weichmagnetische Ausführungen möglich. Der Stator (umgebender, nicht drehbarer Teil) besteht im Wesentlichen aus zwei Spulen, die als Elektromagnete wirken. Diese Spulen werden in Schritten angesteuert und ziehen den Rotor im Drehsinn um einen Schritt weiter. Es gibt mehrere Arten, den Motor zu betreiben. Die wichtigsten sind der Wave-Mode, der Two-Phase-on-Mode und der Half-Step.

21.1.1 Wave-Mode

Bei dieser Ansteuerungsart ist immer eine Spule stromlos.

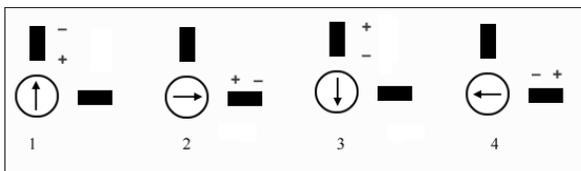


Abb. 21.1: Wave-Mode; die Magnetisierung wird durch den Pfeil im Rotor dargestellt.

1. An der senkrechten Spule wird eine Spannung mit der eingezeichneten Polarität angelegt und der Pfeil nach oben gezogen.
2. Die horizontale Spule zieht die Magnetisierung, die durch den Pfeil dargestellt wird, nach rechts. Der Motor hat sich um 90° im Uhrzeigersinn gedreht.
3. Die senkrechte Spule wird mit umgepolter Spannung betrieben und der Pfeil nach unten gezogen. Eine weitere Drehung im Uhrzeigersinn entsteht.

- Durch Umpolen der Spannung in horizontaler Wicklung wird die nächste viertel Umdrehung bewirkt.

21.1.2 Two-Phase-on-Mode

Hier fließt immer durch zwei Wicklungen gleichzeitig Strom. Das bewirkt, dass der Rotor im Fall A nach rechts oben zeigt. Der Motor hat bei dieser Ansteuerung ein größeres Drehmoment, aber auch einen größeren Stromverbrauch, da immer zwei Wicklungen angeschaltet sind.

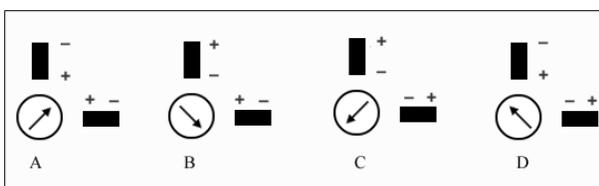


Abb. 21.2: Two-Phase-on Mode

21.1.3 Half-Step-Mode

Dieser Modus ist eine Kombination von Wave- und Two-Phase-on-Mode. Die Drehung des Rotors ist bei einem Schritt nur halb so groß wie bei den anderen Ansteuermethoden.

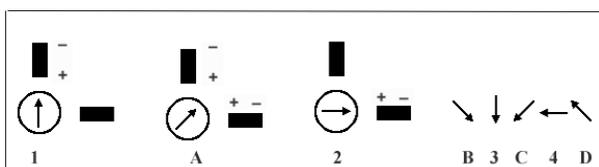


Abb. 21.3: Half-Step-Mode

21.2 Versuchsaufbau

Bei einem realen Schrittmotor ist die erläuterte Drehung von 90° oder 45° pro Schritt schon zu groß. Kleinere Schritte erhält man, wenn man mit den Spulen jeweils mehrere Nord-Süd-Pol-Paare erzeugt. Der im Versuch verwendete Schrittmotor hat eine Auflösung der Schritte von 7,5°. Dies kann man schon durch Drehen mit der Hand nachprüfen (drehen: man spürt ein Ruckeln).

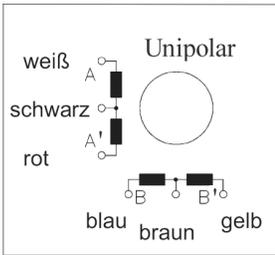


Abb. 21.4: Anschlussbild des verwendeten Schrittmotors nach Datenblatt von Nanotec

Bei diesem Schrittmotor hat jede Wicklung eine Mittelanzapfung (schwarz und braun). Legt man an diese +12 V an, kann man mit einem einfachen Transistor den weißen oder roten Abschluss auf Masse ziehen und damit das Magnetfeld umpolen. Mit der zweiten Spule ist die gleiche Methode an den Anschlüssen braun und blau/gelb möglich. Diese Typen von Schrittmotoren werden als *Unipolar* bezeichnet, da keine Umpolung der Spannung notwendig ist. Im angegebenen Versuch werden nur die Wicklungen schwarz/rot und braun/gelb verwendet, dafür wird aber mit einer Spannung verschiedener Polarität gearbeitet. Der Unipolarmotor wird also im Bipolarbetrieb eingesetzt. Der Strom, den die USB-6008 direkt am Port (Digitalausgang) ausgibt, ist allerdings viel zu klein, um den Schrittmotor anzutreiben. Deswegen ist ein Treiber-IC notwendig, um den nötigen Strom durch den Motor zu schicken. Man verwendet den Baustein L293D, der vom 5-V-Spannungsausgang der Messkarte versorgt wird.

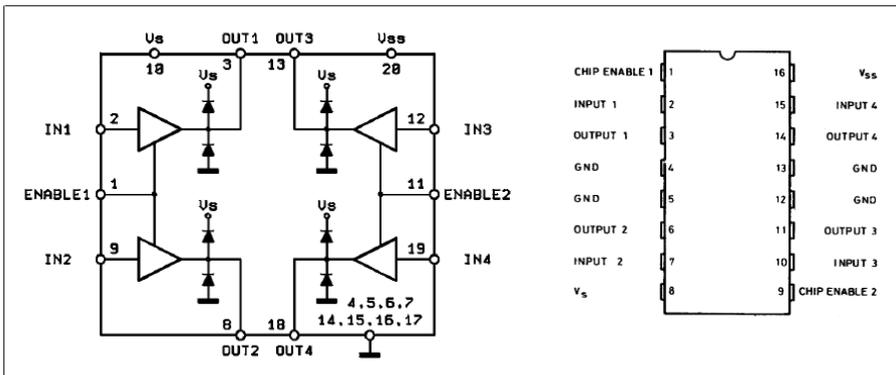


Abb. 21.5: L293-IC zur Ansteuerung des Schrittmotors [6]

Obwohl der Motor für 12 V pro Wicklung ausgelegt ist, funktioniert er auch noch mit 5 V. Wie auch aus dem Bild ersichtlich ist, ist kein zusätzliches Netzgerät erforderlich. Der Wicklungswiderstand des Schrittmotors von 50Ω garantiert, dass die Strombelastung für die USB-6008 nicht zu groß wird.

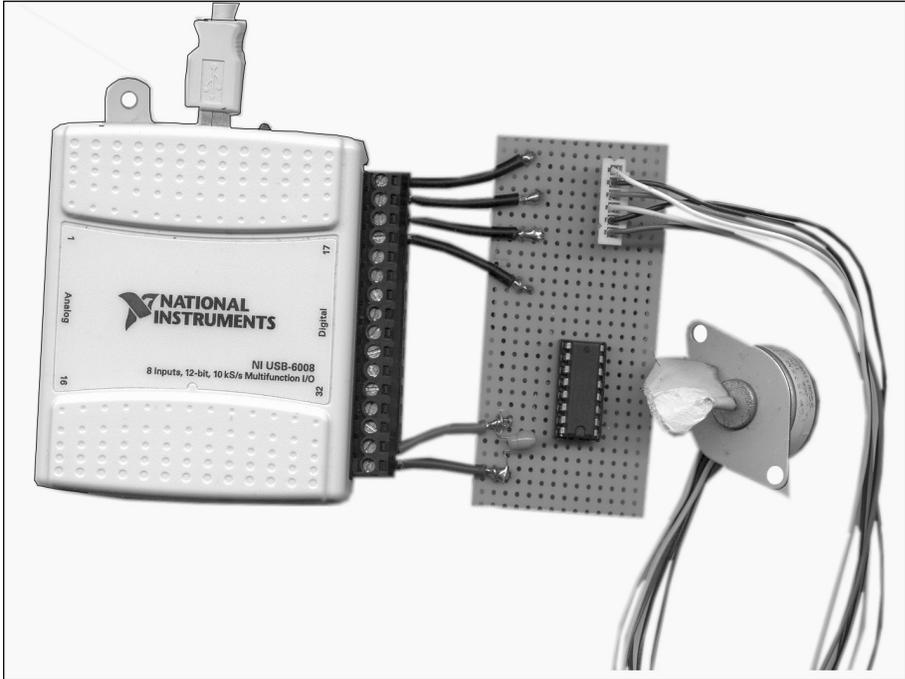


Abb. 21.6: Aufbau der Schaltung

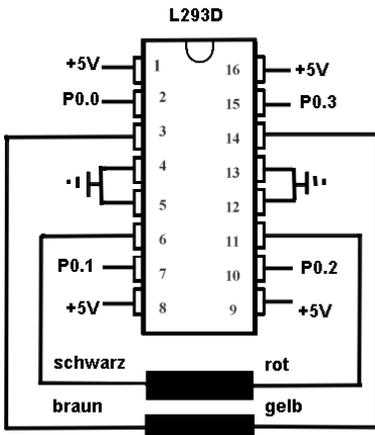


Abb. 21.7: Anschluss des Schrittmotors am Treiber-IC

21.3 Programme

21.3.1 Wave-Mode

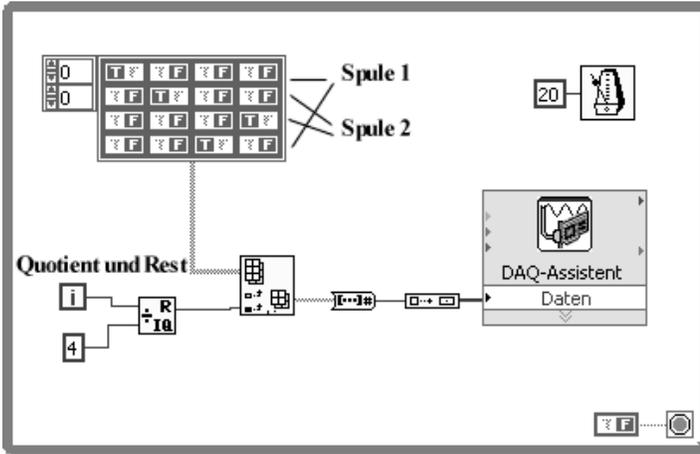


Abb. 21.8: Programm zum Ansteuern des Schrittmotors im Wave-Mode

Aus dem zweidimensionalen Array wird mit *Array indizieren* eine Spalte herausgeschnitten. Mit den Werten aus dieser Spalte werden die Spulen des Schrittmotors angesteuert. Zuerst erfolgt dazu mit *Boolesches Array nach Zahl* die Umwandlung in einen Integer. Danach wird mit *Array erstellen* ein Array gebildet, das die Integerwerte enthält (also Einsen und Nullen, statt T und F). Diese spezielle Art der Formatierung ist für die Digitalausgabe an einen Port notwendig, wenn man mit dem DAQ-Assistenten arbeiten möchte.

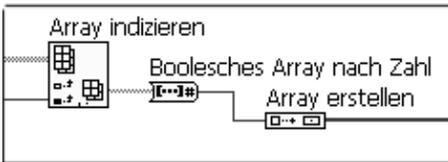


Abb. 21.9: Spaltenweise Ausgabe des 2-D-Arrays in den DAQ-Assistenten

Welche Spalte genau zur Steuerung verwendet wird, soll sich zyklisch ändern (0, 1, 2, 3, 0, 1, 2, 3, 0 usw.). Das erreicht man mit der *Modulo*-Funktion (Bildung des Rests bei Division). Bei der verwendeten Funktion mit Modulo 4 wird aus der Folge $i = 0, 1, 2, 3, 4, 5, \dots$ (von der Schleife stammend) am Iterationsterminal die Folge 0, 1, 2, 3, 0, 1, ...

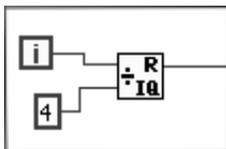


Abb. 21.10: Funktion Quotient und Rest; aus 0, 1, 2, 3, 4, 5, ... wird 0, 1, 2, 3, 0, 1, ...

Man betrachtet nun eine einzelne Spalte, also eine Steuerungsanweisung. Das oberste und das unterste Element (Zeilen 0 und 3 im zweidimensionalen Array) sind für die senkrechte Spule verantwortlich. Die mittleren Elemente (Zeilen 1 und 2) beeinflussen die horizontale Spule. Die Spalte 0 mit den Werten T, F, F bewirkt, dass der magnetisierte Rotor nach oben gezogen wird. Die horizontale Spule ist auf F, F gesetzt und somit stromlos. Das gleiche Ergebnis erhält man mit T, T, T, F , da auch hier die horizontale Spule keinen Spannungsunterschied hat und somit stromlos wird. Damit ist das erste Bild im Diagramm für den Wave-Mode realisiert. Spalte 1 hat die Werte F, T, F, F . Damit wird die senkrechte Spule stromlos und die horizontale Spule zieht den Rotor nach rechts. Dies entspricht dem zweiten Bild im Diagramm des Wave-Mode. Die beiden weiteren Fälle werden durch Umkehr der Stromrichtung aus den Fällen 1 und 2 gebildet.

Rücklauf

Eine Umkehr der Drehrichtung des Motors kann durch eine veränderte Tabelle oder durch einen Tabellenzugriff in umgekehrter Reihenfolge erreicht werden. Es werden dabei die Spalten 3, 2, 1, 0, 3, 2, 1, 0 ... aus dem zweidimensionalen Array herausgeschnitten.

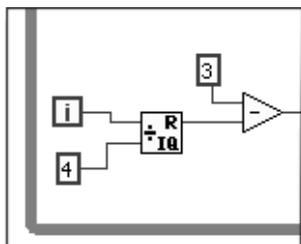


Abb. 21.11: Rücklauf des Motors durch veränderten Array-Zugriff

21.3.2 Two-Phase-on- und Half-Step-Mode

Der Two-Phase-on- und der Half-Step-Mode können einfach durch Austausch der Tabellen im Programm realisiert werden.

Stichwortverzeichnis

A

- Ablaufinvariante Ausführung 99
 - mit Sequenzstruktur 158
 - über gültige Daten 157
- Addieren 24
- Aktualisierungsmodi 102
- Akustisches GPS 215
- 2D-Array transponieren 57
- Array 49
 - als numerisches Eingabeelement 49
 - Array erstellen 53
 - Arrayfunktionen 53
 - eindimensional 51
 - mehrdimensional 57
 - Array, gerade Zahlen 90
 - Array indizieren 52, 58
 - Array, indizieren 79
 - Array initialisieren 55
 - Array-Länge 52
 - Array nach Tabellen-String 72
 - Arrays 48
 - Arrays beliebigen Typs 55
 - Arrays in Cluster 63
 - Arrays, mehrdimensional 53
 - Arrays, neu erstellte 49
 - Array, zweidimensional 56
- ASCII-Datei 153
- Ausführungsmode 98
- Ausgabe einer Sinuskurve 81
- Ausgabe einer Sinusschwingung 162
- Auswählen 38
 - Beispiel 40

B

- Bedingungsanschluss 76
- Beschriftung 33
 - verschieben 34
- Bessel-Filter 184

- Bildrätsel 237
- Bildverarbeitung 221, 237
- Binärdateien 151
- Blockdiagramm 15, 16
- Boolesch 25
- Boolescher Text 37

C

- Case-Struktur 40
 - Bereichsauswahl 44
 - Fall einfügen 45
 - für boolesche Werte 42
 - für numerische Daten 43
 - für Strings 45
 - häufige Fehler 46
 - Voreinstellung 43
- Chirp 215
- Cluster 60
 - als Frontpanelement 61
 - aufschlüsseln 60
 - erstellen 60
 - in Arrays 63
 - Wertänderung 62
 - zerlegen 60
- CSV-Format 147

D

- Daisy Chain 64
- DAQ-Assistant 126
- Darstellung, Datentyp 22
- Darstellung von drei Kurven 105
- Darstellung von zwei Kurven 105
- Dateidialog 143
- Dateien 141
- Dateinamen 142
 - absolut 145
 - relativ 145

Datenerfassung 118
Datenspeicherung, Express 149
DBL 21
Dehnungsmessstreifen 198
Digitalausgabe 130
Dithering 199
Drahtspule 29
Drehstrom 192
DTR 172
Durchführungen 78

E

Eieruhr 160
Eigenschaftsknoten 109
Eingabeelement 21
Einhüllende 213
EKG 182
Elementfenster 16
Entscheidungsstrukturen 38
Entscheidungsterminal 41
Error-Cluster 63
Excel-Datei
 erstellen 146
 lesen 145
Excel-Format 141
EXE-Datei 174
Expressfunktionen 18

F

Fehlerfenster 17
Flankenerkennung 88
Flusssteuerung 170
Format-String 72
For-Schleifen 74
For-While-Schleife 77
Fourier-Transformation, 2D 221
Frequenzgang 138
Frontpanel 15, 16
Frontpanelement 15
 in eine Grafik konvertieren 116
 Wert 33
Funktionspalette 17

G

Grafik 113
 elementare Funktionen 113
Grafische Frontpanelemente 100

H

Half-Step-Mode 186
Hallo, Welt! 28
Handshake 170
Highlight-Modus 85
Hilbert-Transformation 214
Historienlänge 101

I

I8 21
I16 21
IMAQ 234, 238
Indexanzeige 48
Installationsprogramm 177
In String suchen 71
Interpolation 208
Iterationsanschluss 74

J

JPEG-Bild 115

K

Kantenschärfung 241
Kernel-Operationen 241
Klassische Funktionen 18
Konfiguration numerischer
 Frontpanelemente 31
Konstanten 27
Konversion 26
Koordinaten 114

L

Lichtschnittverfahren 232
Listenfelder 35
Lokale Sequenz-Variable 159
Lokale Variable 160

M

Masseschleife 140
Maximumsuche 212
Measurement & Automation Explorer 119
Median 211
Messgeräte-Strings 70
Messkarten, Probleme 139
Modulo-Funktion 289
Münzen zählen 238
Muster suchen 68

N

Null-Modemkabel 170
Numerische Datentypen 20

O

Oberflächengraph 111
Oszilloskop 129

P

Parallelarbeit 156
Pfad 25
Pin-Belegung anzeigen 121
Programmablauf 155
Pulsweitenmodulation 192

R

Rauschen 210
RC-Glied 134

Rechnen mit einem Array 58
reentrant 99
RS-232 167
RS-232, Einlesen 172
RS-232, String-Ausgabe 171
RSE 140
RTS 172
RxD 169

S

3-D-Scanner 232
Schalter 25
 konfigurieren 36
 Schaltverhalten 36
Schieberegister 84
 gestapelt 86
 in Form von Rückkopplungsknoten 87
 nicht initialisiert 86
Schleifen 74
 Ein- und Ausgabe 78
Schrittmotoransteuerung 185
Selektorfeld 41
Serielle Schnittstelle 167
Signaldiagramm 100
Signalgraph 80, 100
Signalverarbeitung 208
Signalverlaufsgraph 80
Skalierung 103
 der X-Achse 103
 der Y-Achse 102
 logarithmisch 35
 von Anzeigeelementen 34
Soundkarte 161, 215
 Test 161
Spektralanalysator 130, 165
Spikes 210
Spitzenwerterkennung 212
Störungen 199, 210
Streaming 138
String-Funktionen 66
String-Konstanten 28
String-Länge 67
Strings 24, 65
 Darstellungsformen 65

suchen und ersetzen 68
verknüpfen 67
SubVi erstellen 95
Summe 82
 aller Elemente in einem Array 82
 der arithmetischen Reihe 81

T

Tabellen-String 73
 in Array 73
Teil-String 70
Temperatursensor 229
Temperaturverteilung 225
Terminalprogramm 202
Testpanel 121
T-Flipflop 98
Threads 202
Tiefpass 211
Timer 74
Transistorkennlinie 133
Transponieren 57
Trigger 135
Two-Phase-on-Mode 186
TxD 169

U

U8 21
U16 21
Umwandlung 30
 Anzeigeelement 30
 Bedienelement 30

Datentyp 26
 eines Arrays 163
 eines Signalverlaufs 164
 Konstante 31
 von Datentypen, Soundkarte 163
 von dynamischen Daten 164
Unterprogramm 91
 automatisch erstellen 95
 erstellen 91
 Konfiguration 97
USB 6008 118

W

Wave-Mode 185
Webcam 234
Wechselrichter 192
Werkzeugpalette 17, 19
While-Schleife 76

X

XON-XOFF 170
XY-Graph 105

Z

Zählerterminal 74
Zeichenketten 24
Zeitstempel 108, 149

Friedrich Plötzeneder / Birgit Plötzeneder

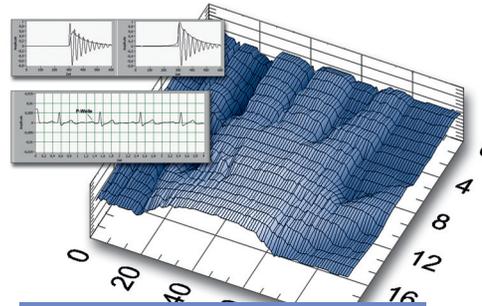
Praxiseinstieg **LabVIEW**

Dieses Buch führt Anfänger und Fortgeschrittene Schritt für Schritt mit vielen Praxisbeispielen in die führende Programmiersprache für Messtechnik, „LabVIEW“ ein.

Der erste Teil des Buchs erklärt ausführlich und umfassend die Programmiersprache LabVIEW. Die Kapitel schließen jeweils mit instruktiven Beispielen. Aufgrund des Umfangs von LabVIEW, das mehr als tausend ausprogrammierte Algorithmen hat, beschränkt sich das Buch auf die wichtigsten Algorithmen und Sprachelemente, sodass Sie sich nicht mit unnötigem Ballast befassen müssen.

Der zweite Teil des Buchs zeigt, wie leicht man mit LabVIEW technische, physikalische und mathematische Probleme unterschiedlichster Bereiche lösen kann. Die Experimente sind mit einfacher, oft bereits vorhandener Ausrüstung möglich. Praxisrelevante Themen wie Messdatenerfassung, serielle Schnittstelle und Soundkartenprogrammierung werden ausführlich besprochen. Anhand von einfach durchzuführenden Experimenten wie dem Messen eines EKGs, der Ansteuerung eines Schrittmotors oder dem Bau eines 3-D-Scanners werden wichtige Kenntnisse für die Realisierung eigener Projekte vermittelt. Sie können ohne große Investition durchgeführt werden und sind hinsichtlich der Durchführung und Mathematik vollständig beschrieben und erprobt.

Die Experimente sind auch als Laborübung an Hochschulen sowie für Auszubildende, Ingenieure und Autodidakten, die mit LabVIEW arbeiten, geeignet.



Praktische Experimente :

- EKG
- Schrittmotoransteuerung
- Wechselrichter – Drehstrom aus dem Laptop
- Dehnungsmessstreifen
- Terminalprogramm
- Signalverarbeitung in der Praxis
- Akustisches GPS
- Bildverarbeitung mit zweidimensionaler Fourier-Transformation
- Temperaturverteilung in einem Ring
- 3-D-Scanner mit Laptop und Beamer
- Praktische Bildverarbeitung
- und viele mehr

ISBN 978-3-7723-4039-0



Euro **29,95** [D]