Bernd vom Berg
Peter Groppe

LabVIEW-Projekte mit dem

DAQ-Modul USB-6008

~~ Ohmmeter ~~

~~ Thermometer mit KTY 81 ~~

~~ Oszilloskop ~~

~~ Treiber NI-DAQmx ~~

(Stand: 18.11.2012, V1.0)

Wichtiger Hinweis

Bei der Zusammenstellung von Texten, Abbildungen, Stückteillisten, u.s.w. wurde mit größter Sorgfalt vorgegangen. Trotzdem können Fehler nicht vollständig ausgeschlossen werden.
Die Autoren können für fehlerhafte Angaben und deren Folgen weder eine juristische Verantwortung noch irgendeine Haftung übernehmen.
Für die Mitteilung eventueller Fehler sind die Autoren dankbar.

Alle von uns im Rahmen des Projektes erstellten Programme sind zur freien, nicht kommerziellen Verwendung freigegeben, sofern die Rechte Dritter (z.B. von Seiten der Firma National Instruments (NI)) nicht betroffen sind.

I n h a l t

(Stand: 18.11.2012)

1. Einleitung

Wichtiger Hinweis

Dieses Projekt setzt einige grundlegende Kenntnisse in die Funktionsweise und in die Möglichkeiten von LabVIEW voraus.

Sollten Sie dieses Wissen noch nicht haben, so ist das aber auch kein Problem:
In unserem ersten LabVIEW-Projekt: "**LabVIEW für den Praktiker / LabVIEW meets µC**", das wir ebenfalls in diesem Forum veröffentlich haben, können Sie diese notwendigen Kenntnisse vorab einfach erwerben.
Weiterhin finden Sie im Literaturverzeichnis gute Einführungsbücher zu LabVIEW.

Arbeiten Sie also bei Bedarf zuerst die dortigen Unterlagen durch und kommen Sie dann zurück zu diesem Projekt.

Das Datenerfassungsmodul USB-6008 von National Instruments (NI) ist eine kleine kompakte Datenerfassungseinheit mit analogen und digitalen Ein-/Ausgabemöglichkeiten und einer einfachen digitalen Zählerfunktion.
Aufgrund seiner Kompaktheit und seines günstigen Preises wird es unter anderem sehr häufig als „Einsteiger-Modell" in der Lehre und in der Ausbildung eingesetzt, um dem Anfänger in LabVIEW zu demonstrieren, wie einfach, unkompliziert und dennoch leistungsfähig die fertigen passenden Treibermodule/funktionen (die **NI-DAQmx-Treiber**) für dieses Modul in der Praxis, d.h. in LabVIEW-VIs, einsetzbar sind.

In drei kleinen LabVIEW-Projekten

- einem **Ohmmeter**,
- einem **Thermometer mit KTY81-110** und
- einem **Oszilloskop**

soll der Ersteinsatz des USB-6008-Moduls unter Verwendung der NI-DAQmx-Treiber-Software praktisch vorgestellt werden.
Die dabei entwickelten LabVIEW-VIs werden genau erklärt und können daher als Basis für eigene (Weiter)Entwicklungen verwendet werden.

Diese drei Projekte sind daher der ideale Start für den ersten Einstieg in LabVIEW für Schüler/Studenten/Azubis oder andere engagierte Praktiker.

2. Die Hardware - Das DAQ-Modul USB-6008

Das USB-6008-Modul ist ein „**Busversorgtes Multifunktionales USB-Datenerfassungs-gerät (DAQ (Data Acquisition)-Modul)**“ von NI, **Abb.2.1**:

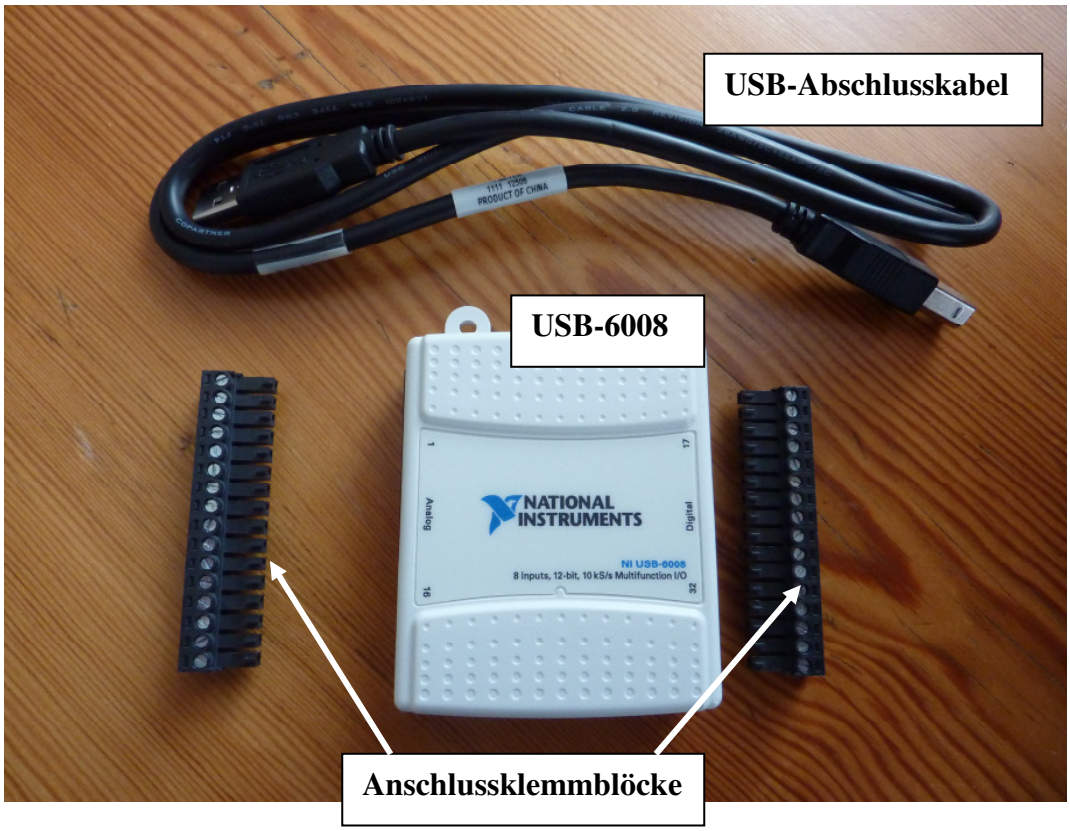


Abb.2.1: Das USB-6008-Paket von NI (incl. CD mit NI-DAQmx-Treibern)

Diese kompakte Einheit wird über eine USB-Verbindung an den PC/LapTop angeschlossen und ermöglicht so die Erfassung, Verarbeitung und Ausgabe von analogen und digitalen Signalen z.B. zum Aufbau einer einfachen Messwerterfassung und -verarbeitung bis hin zu einer kleinen Prozeßsteuerung, -überwachung oder -regelung.
Ein einfacher digitaler Ereigniszähler und zwei (Versorgungs)Spannungsausgänge runden den Funktionsumfang des USB-6008 ab.

Die **Abb.2.2** zeigt das Blockschaltbild des USB-6008:

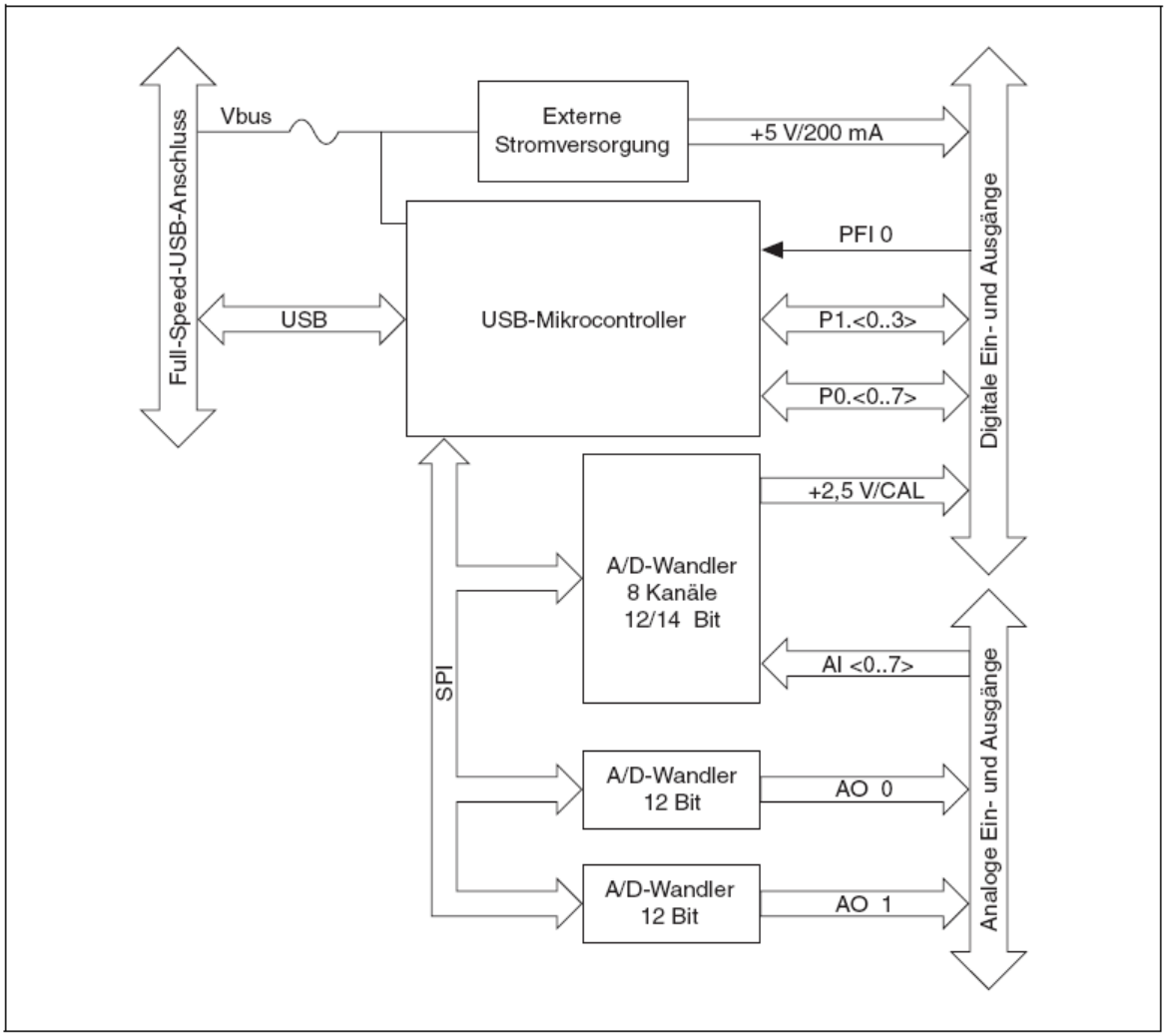


Abb.2.2: Blockschaltbild des USB-6008, [1]

Im Detail stehen dem Anwender zur Verfügung (nachfolgend einige wesentliche charakteristische Kenndaten, weitere Daten s. Datenblatt zum USB-6008, [1]):

Analoge Spannungs-Eingänge: AI0 … AI7

Kanalanzahl:

Bei Messung gegen Bezugspotential:	8 Eingänge
(RSE ≡ Referenced single-ended)	
Bei differentieller Spannungsmessung (zwischen jeweils zwei Eingängen):	4 Eingänge

Die Kanäle sind per Software anwählbar.

Auflösung:

Bei Messung gegen Bezugspotential:	11 Bit
Bei differentieller Messung:	12 Bit

Messbereiche:
Bei Messung gegen Bezugspotential: ±10 V
Bei differentieller Messung: ±20 V, ±10 V, ±5 V, ±4 V, ±2,5 V, ±2 V, ±1,25 V, ±1 V

Eingangswiderstand: 144 kΩ

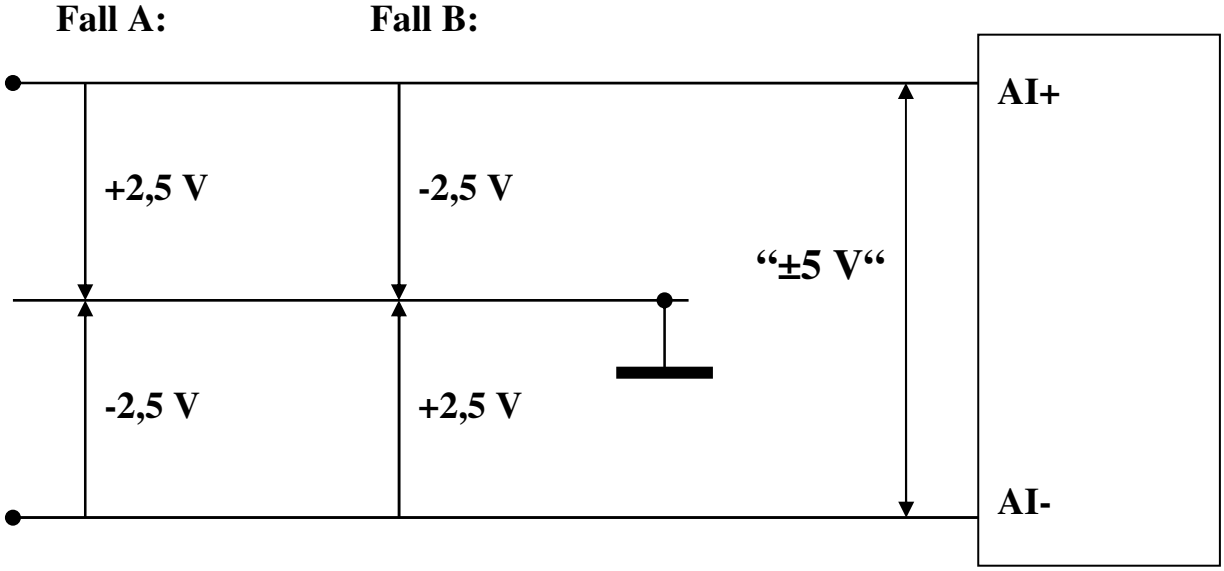
Max. Spannung gegen Bezugspotential (Arbeitsspannungsbereich): ±10 V

Überspannungsschutz bis: ±35 V

Maximale Abtastrate: 10 kS/s

Wichtig: Die differentielle Messung von Eingangsspanngen

Bei der differentiellen Messung von Eingangsspannungen muss darauf geachtet werden, dass man die Angaben zu den Messbereichen richtig interpretiert, denn fast immer ist man es in der Elektrotechnik bei Spannungsangaben gewohnt, diese gegen Masse anzugeben.
Das ist hier aber bei der **Angabe der maximalen Differenzspannungen** nicht der Fall, **Abb.2.3**:



AI+ und AI- sind die beiden Analog-Eingänge zwischen denen die Differenz-eingangsspannung gemessen werden soll.

Abb.2.3: Die differentielle Messung von Eingangsspannungen

Vom Modul gemessen wird bei der Differenzeingangsspannungsmessung immer die Spannung zwischen AI+ und AI- (AI ≡ Analog In), also:

Gemessene Spannung = (AI+) - (AI-)

Die Angabe von z.B. "**Messung der Differenzspannung im Messbereich von ±5V**" bedeutet nun, dass direkt zwischen den beiden Eingängen eine maximale Spannungsdifferenz ((AI+) – (AI-)) von +5 V bzw. -5 V anliegen darf.

Das aber ist nicht identisch mit der Spannung eines Einganges gegenüber Masse !

Die maximale Messpannung eines Eingangs **gegenüber Masse** darf bei der Differenzspannungsmessung immer nur die Hälfte der maximalen Differenzspannung betragen. Hier bei diesem gewählten Messbereich gegen Masse also nur +2,5 V bzw. -2,5 V !

Abb.2.3, Fall A:

Hier ergibt sich als Differenzeingangsspannung zwischen dem Eingang AI+ und dem Eingang AI- ein Wert von: **(+2,5 V) – (-2,5 V) = +5 V** (maximal).

Abb.2.3, Fall B:

Hier ergibt sich somit als Differenzeingangsspannung zwischen dem Eingang AI+ und dem Eingang AI- ein Wert von: **(-2,5 V) – (+2,5 V) = -5 V** (maximal).

Die maximalen Spannungssignale, die im Differenzspannungsbetrieb mit dem USB-6008 gemessen werden können, dürfen eine Grenze von ±20 V zwischen den Eingängen, also je Eingang maximal ±10 V gegen Masse, nicht überschreiten !
Darüber hinaus gehende Spannungen werden abgeschnitten, d.h. auf diese Maximalwerte begrenzt.
Die Eingänge selber vertragen Überspannung bis zur Überspannung von ±35 V.

Das alles ist zu beachten, wenn bei der späteren Konfiguration des USB-6008ers Eingangsspannungen im Differenzmodus gemessen werden sollen.

Analoge Spannungs-Ausgänge: AO0, AO1

Kanalanzahl: 2

Ausgangsspannungsbereich: 0 V … +5 V

Auflösung: 12 Bit

Ausgangsimpedanz: 50 Ω

Max. Ausgangsstrom: 5 mA

Spannung nach Einschalten: 0 V

Flankensteilheit: 1 V/µs

Max. Ausgaberate: 150 Hz

Digital I/O: P0.0 … P0.7, P1.0 … P1.3

Zwei digitale I/O-Ports vorhanden:

Port P0:	8 Leitungen (Kanäle)	(P0.0 … P0.7)
Port P1:	4 Leitungen (Kanäle)	(P1.0 … P1.3)

Jeder Kanal kann per Software als Ein- oder Ausgang programmiert werden.

Ausgangskonfiguration: Open Collector

Maximale Ausgangsströme:

Bei L-Pegel:	8,5 mA
Bei H-Pegel:	-0,6 mA

Kompatibilität: TTL, LVTTL, CMOS

Pull-Up-Widerstand (Geräte-intern): 4,7 kΩ nach +5V

Zustand nach Einschalten: Eingang

Ereigniszähler: PFI0

Zähleranzahl: 1

Auflösung (Zählerbreite): 32 Bit

Art der Zählung: zählen der fallenden Flanken

Zählrichtung: aufwärts

Notwendiger Pull-Up-Widerstand: 4,7 kΩ nach 5 V

Maximale Eingangsfrequenz: 5 MHz

Impulsbreiten:

Minimale High-Impulsbreite	100 ns
Minimale Low-Impulsbreite	100 ns

Pegel:

Minimale High-Eingangsspannung	2,0 V
Maximale Low-Eingangsspannung	0,8 V

Die Pinbelegungen des USB-6008

Sind in der **Abb.2.4** zu sehen:

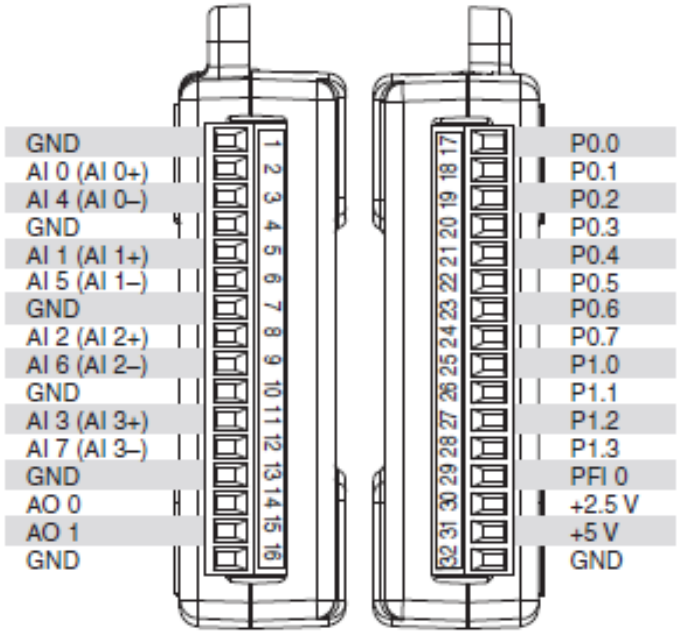


Abb.2.4: Die Pinbelegungen, [1]

Angeschlossen an einen PC/LapTop wird das USB-6008 über eine normale USB-Verbindung.

Ist auf dem Rechner bereits die NI-DAQmx-Treibersoftware installiert (s. Kap. 3), so wird dieses DAQ-Modul bereits richtig erkannt und kann unter LabVIEW verwendet werden.

2.1 Der Measurement & Automation Explorer (MAX) und erste Reaktionen

Wir gehen nun einmal davon aus, dass die benötigte Treiber-Software zum Betrieb des USB-6008er-Moduls bereits mit LabVIEW zusammen installiert wurde (s. Kap. 3.1).
Ist das noch nicht der Fall, so sollten Sie dieses jetzt nachholen (s. Kap. 3.1)

Starten Sie als erstes den NI-Measurement & Automation Explorer (MAX), **Abb.2.1.1**:

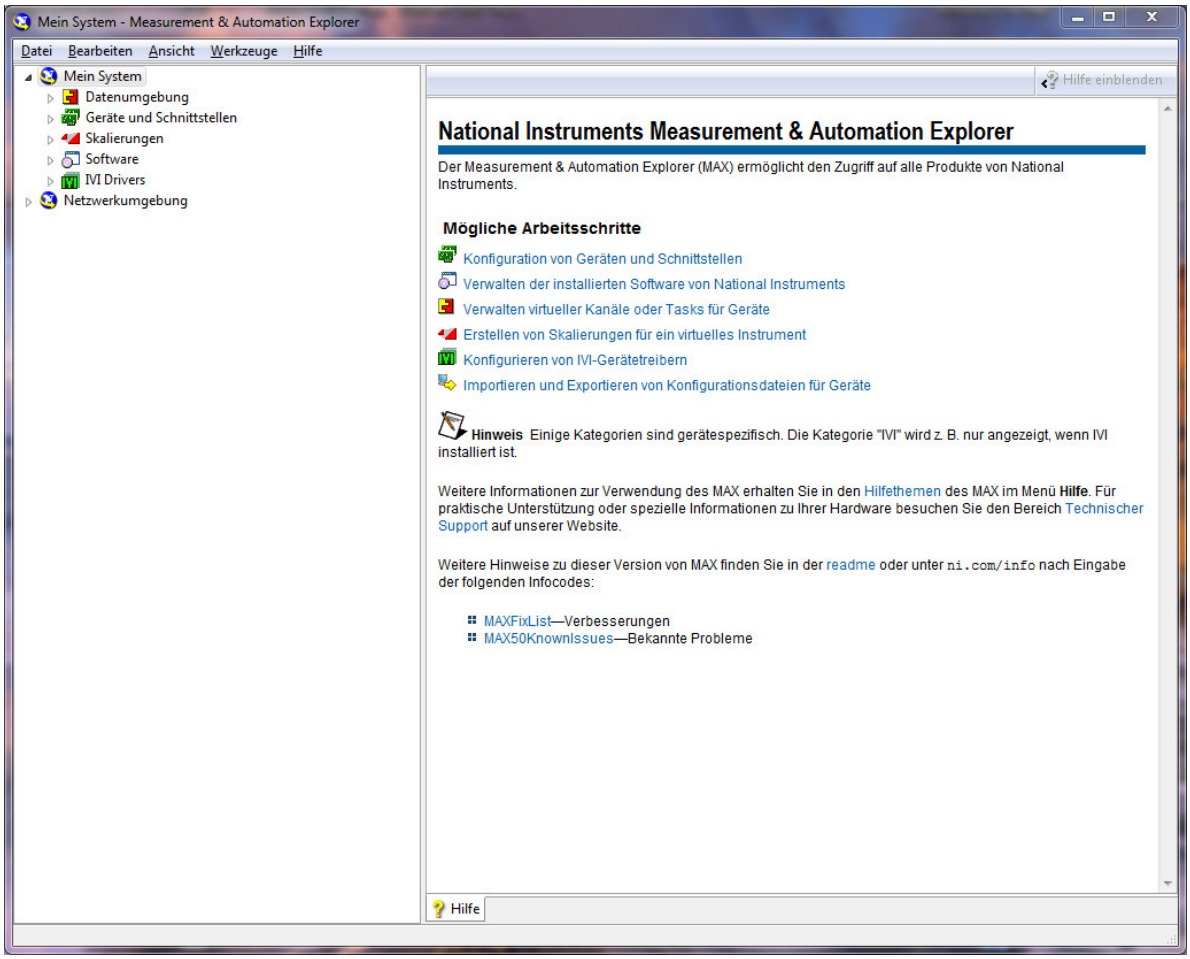


Abb.2.1.1: Der Measurement & Automation Explorer (MAX) als übergeordnete Steuerzentrale für die NI-Hard- und Software

Klicken Sie nun im linken Fenster auf den Punkt ´**Geräte und Schnittstellen**´, so erscheint eine Auflistung der an ihren Rechner angeschlossenen und vom MAX erkannten Geräte bzw. Komponenten, **Abb.2.1.2**:

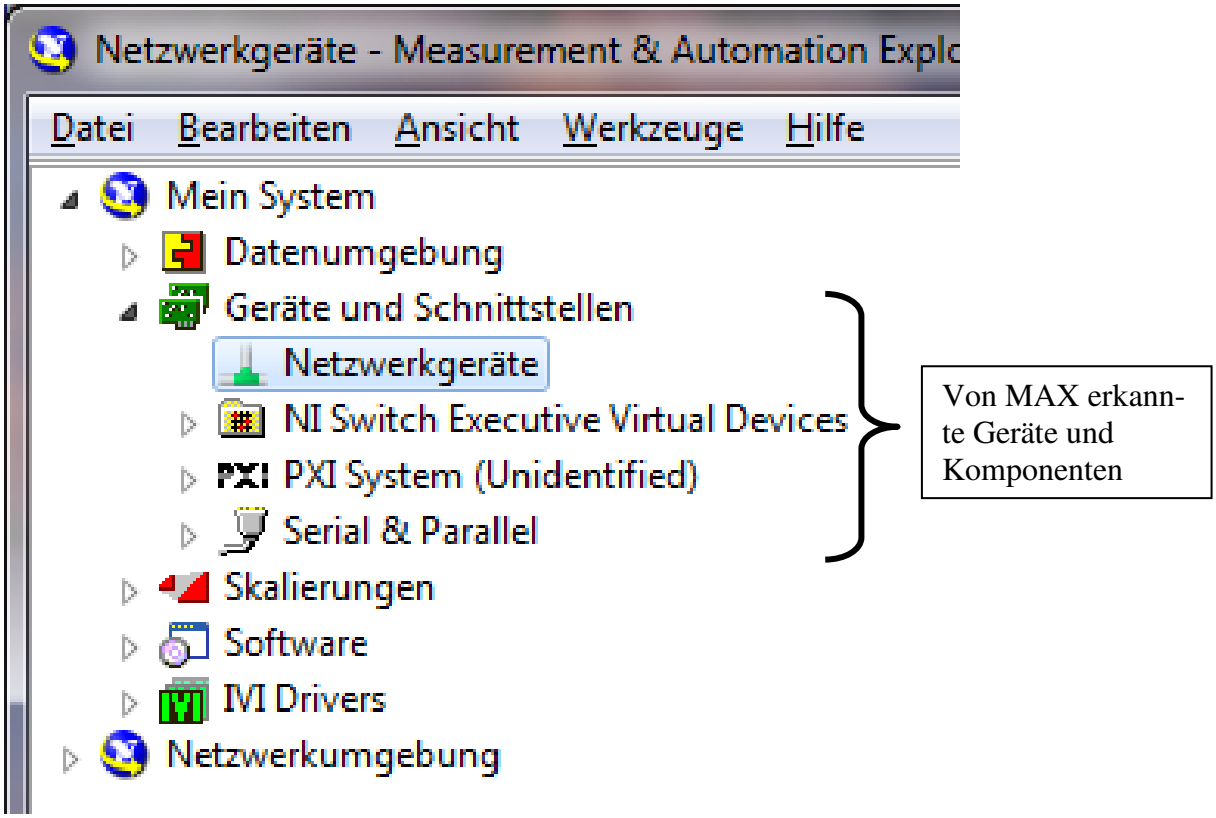


Abb.2.1.2: Die vom MAX erkannten Geräte und Komponenten

Alle diese vom MAX erkannten Einheiten können Sie dann später unter LabVIEW ansprechen und damit arbeiten.
Zur Zeit sind hier (für unseren Rechner) lediglich die erkannten seriellen und parallelen Schnittstellen aufgeführt.
Beenden Sie nun zunächst den MAX wieder.

Schließen Sie danach das USB-6008er-Modul über eine (beliebige) USB-Schnittstelle an Ihren Rechner an:
In der unteren Task-Leiste erscheint darauf der Hinweis, dass ein USB-6008er-Modul erkannt worden und nun einsatzbereit ist, **Abb.2.1.3**:

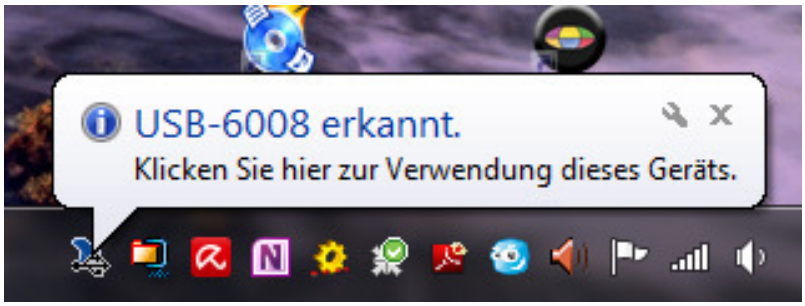


Abb.2.1.3: Das USB-6008er-Modul wurde vom Rechner erkannt

Gleichzeitig blinkt die **grüne LED am USB-6008er** regelmäßig – das ist ebenfalls ein Zeichen dafür, dass sich das USB-6008er-Modul richtig erkannt fühlt und jetzt einsatzfähig ist.

Klicken Sie NICHT auf das NI-Ikon, sondern rufen Sie jetzt wieder den MAX auf und öffnen Sie das ´Geräte und Schnittstellen´-Untermenü, **Abb.2.1.4**:

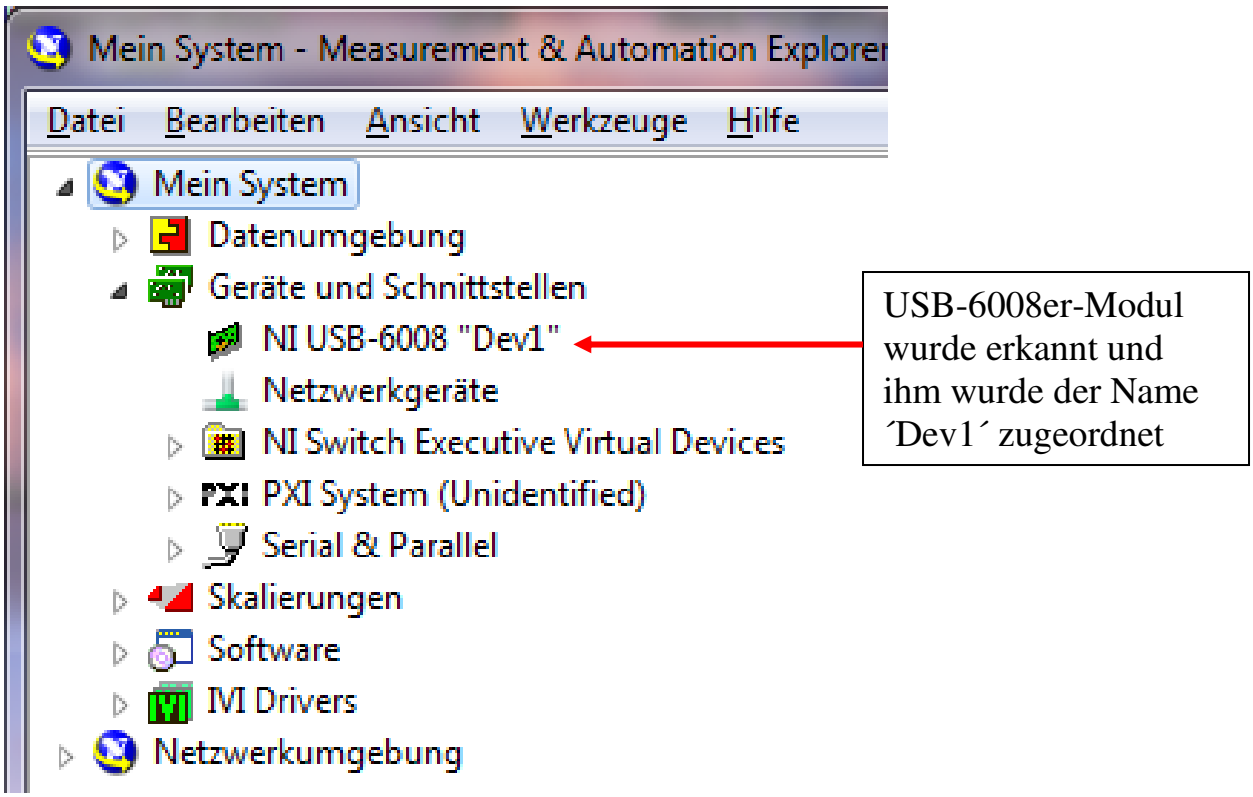


Abb.2.1.4: Der MAX hat das USB-6008er-Modul ebenfalls erkannt

Sie finden nun einen neuen Eintrag vor: das USB-6008er-Modul wurde vom MAX erkannt und es hat den Namen ´Dev1´ zugeordnet bekommen (diesen Namen können Sie natürlich beliebig ändern und so Ihrer Anwendung anpassen).
Ab jetzt ist dieses Modul unter diesem Namen in der gesamtem „NI-Welt“ auf Ihrem Rechner bekannt und insbesondere kann das Modul unter diesem Namen unter LabVIEW angesprochen werden.

Klicken Sie nun mit der Maus zweimal auf diesen Geräteeintrag und es erscheinen im mittleren und im rechten Teilfenster einige Grundoperationen, die Sie jetzt schon mit dem Modul ausführen können, **Abb.2.1.5**:

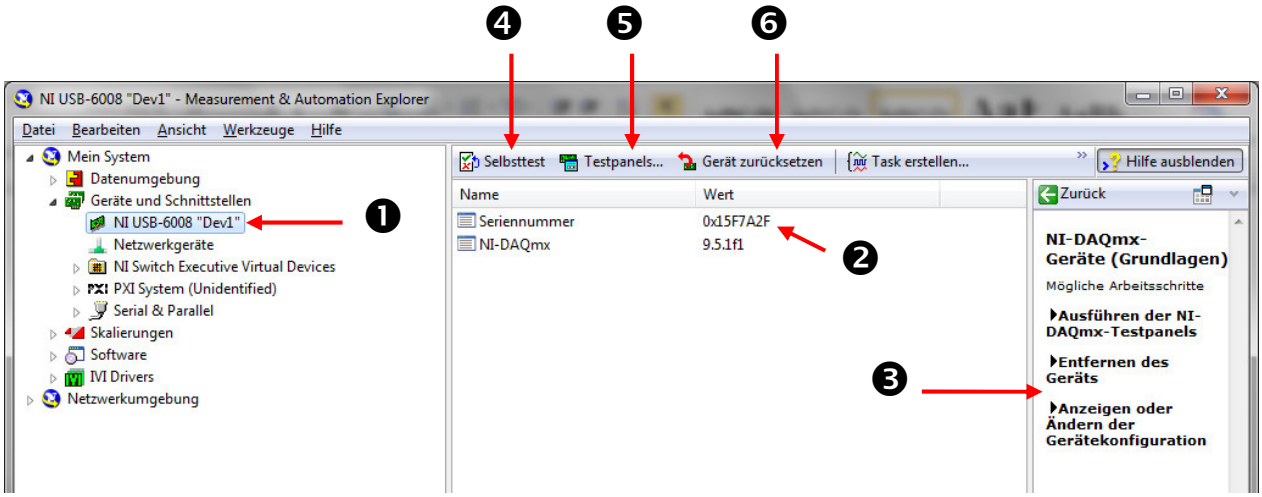


Abb.2.1.5: Das Modul kann „untersucht“ werden

Es bedeuten:

❶:
Hier zweimal mit der linken Maustaste klicken, dann erscheinen die Fenster ❷ und ❸.

❷:
In diesem Fenster finden Sie zu einen die Seriennummer des USB-6008er-Moduls und zum anderen die Versionsnummer der aktuell verwendeten NI-DAQmx-Treibersoftware (s. dazu auch Kapitel 3.1)

❸:
In diesem Fenster erhalten Sie zusätzliche Informationen darüber, was Sie jetzt noch mit dem USB-6008er alles machen können.
Darauf gehen wir hier aber nicht näher ein.

❹:

Auf diesen Knopf sollten Sie nun klicken und so den Selbsttest des Moduls auslösen, **Abb.2.1.6**:

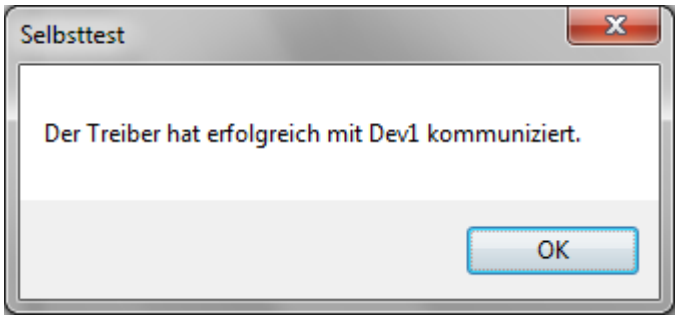


Abb.2.1.6: Alles OK mit Ihrem Modul

Klicken Sie nun auf den OK-Knopf.

❺:
Hiermit können Sie ein Testpanel aufrufen und bereits sehr einfach umfangreiche Tests mit dem USB-6008er-Modul durchführen, ohne dass Sie dazu irgendein Programm schreiben bzw. entwickeln müssten.
Hierauf gehen wir aber zur Zeit nicht näher ein.

❻:
Durch Klicken auf diesen Knopf können Sie den USB-6008er in seinen Grundzustand zurücksetzen, **Abb.2.1.7**:

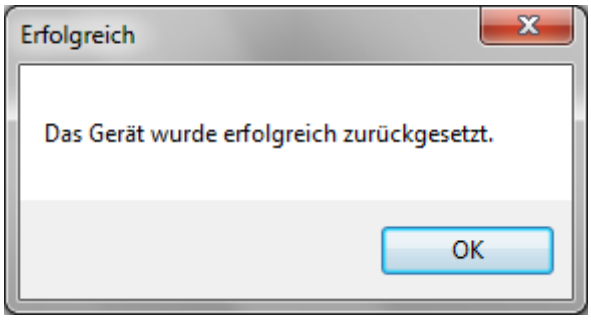


Abb.2.1.7: Der USB-6008er wurde in den Grundzustand zurückgesetzt

Hinweis für Windows XP-Benutzer:

Bei PCs mit Windows XP-Betriebssystem kann zuvor noch der Hinweis auftauchen, dass es Probleme bei der Kommunikation mit dem USB-6008-Modul geben kann und dass man zur Behebung dieses Problems einen so genannten **Hotfix** von Mikrosoft nachladen sollte, **Abb.2.1.8**:

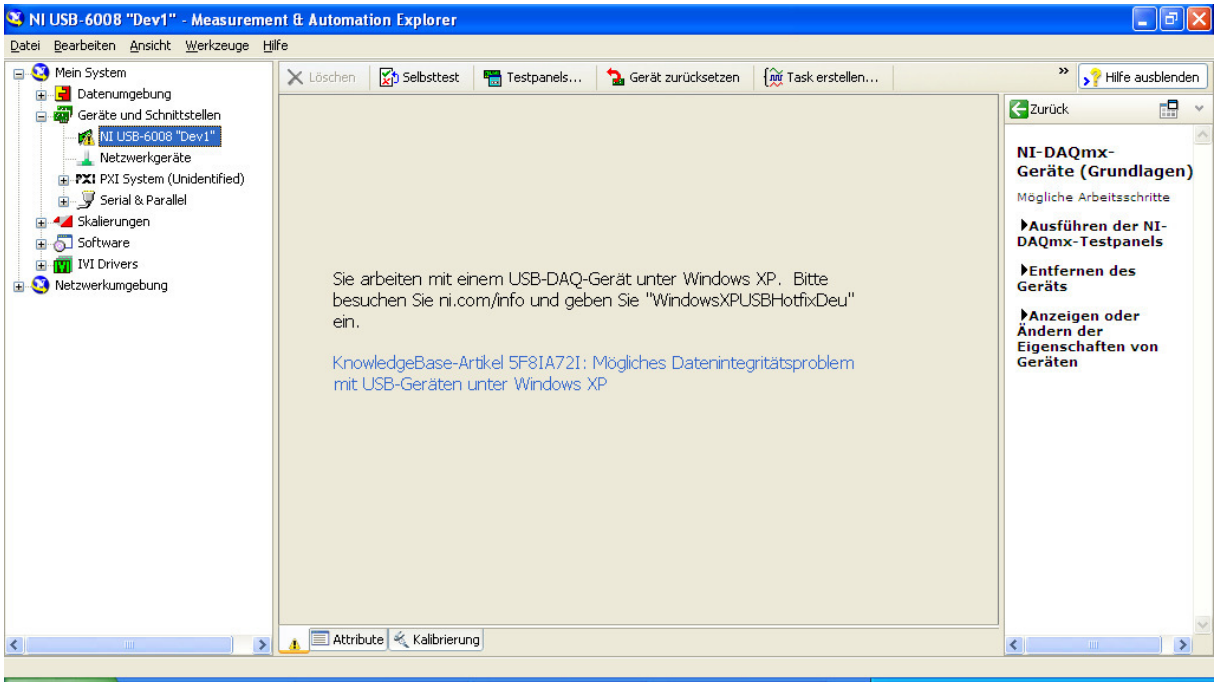


Abb.2.1.8: Mögliche Probleme unter Windows XP

Diese Fehlermeldung bzw. das Nachladen und Installieren diese Hotfixes sollte man aber nur dann ausführen, wenn wirklich Kommunikationsprobleme auftauchen, denn der Warnhinweis dazu von Microsoft selber ist recht eindeutig, **Abb.2.1.9**:

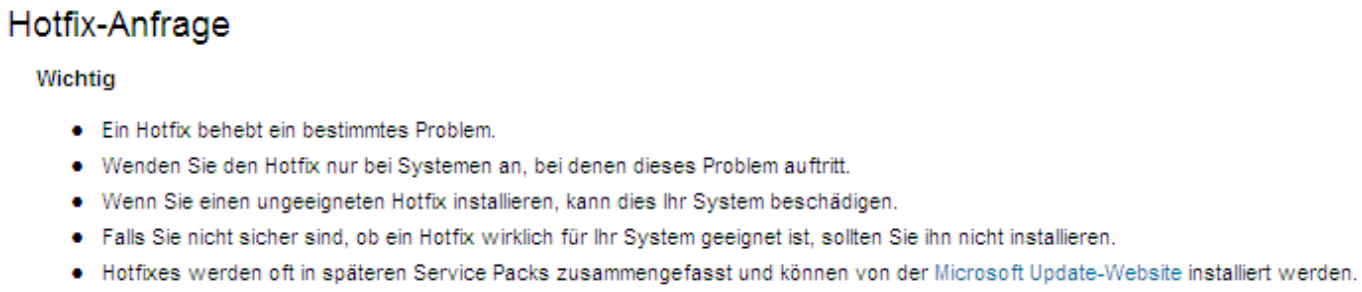


Abb.2.1.9: Die Warnung von Mircosoft

Bei all den uns zur Verfügung stehenden XP-(Test)Rechnern haben wir diesen Hotfix nicht nachinstalliert und bisher noch keine Probleme bekommen.

Fazit:

Das USB-6008er-Modul ist nun einsatzfähig und kann unter LabVIEW verwendet werden !

3. Die Treiber-Software

Bei der allgemeinen Entwicklung von PC/LapTop-gestützten Daten- bzw. genauer: Messwerterfassungssystemen (**DAQ ≡ Data Acquisition-Systeme**) geht es im Wesentlichen darum, fünf große Gruppen von Werten zu erfassen bzw. zu generieren:

- Erfassung analoger Spannungen (Analog IN)
- Erfassung binärer Zustände (Digital IN)
- Erfassung von Zählerständen (Counter/Timer)

- Ausgabe von analogen Spannungen (Analog OUT)
- Ausgabe von binären Werten (Digital OUT)

In sehr vielen Fällen werden als zentrale Komponenten eines solchen technischen Systems (Industrie)PCs oder (Industrie)LapTops verwendet, deren geforderter besonderer Funktionsumfang dann durch geeignete Einbaukarten bzw. durch geeignete externe Moduleinheiten sichergestellt wird (**DAQ-Karten bzw. DAQ-Module**).
Die Datenübertragung zwischen solchen Ein-/Ausgabekarten und dem Rechner erfolgt dann über die internen parallelen Bussysteme bzw. durch eine Ankopplung über die bekannten seriellen Schnittstellen oder Bussysteme (RS232, RS485, GPIB, CAN, WLAN, Ethernet, etc.).

Das grundlegende Problem, das sich für den Anwender/Entwickler solcher individuell zusammen gestellter Gesamtsysteme sehr schnell ergibt, ist die Tatsache, dass es auf dem internationalen Markt unzählige Anbieter solcher Zusteckkarten bzw. externer Module gibt und die Ansteuerung dieser Einheiten, also die benötigte Betriebssoftware, bei weitem nicht weltweit genormt ist.

Jede Karte von jedem Hersteller benötigt in der Regel einen eigenen kompletten Satz von Konfigurations- und Betriebsfunktionen, die so genannten **Treiber(software)pakete**, die in das jeweilige Anwendungsprogramm mit eingebunden (integriert) werden müssen, damit man die entsprechenden Karten betreiben kann, **Abb.3.1**:

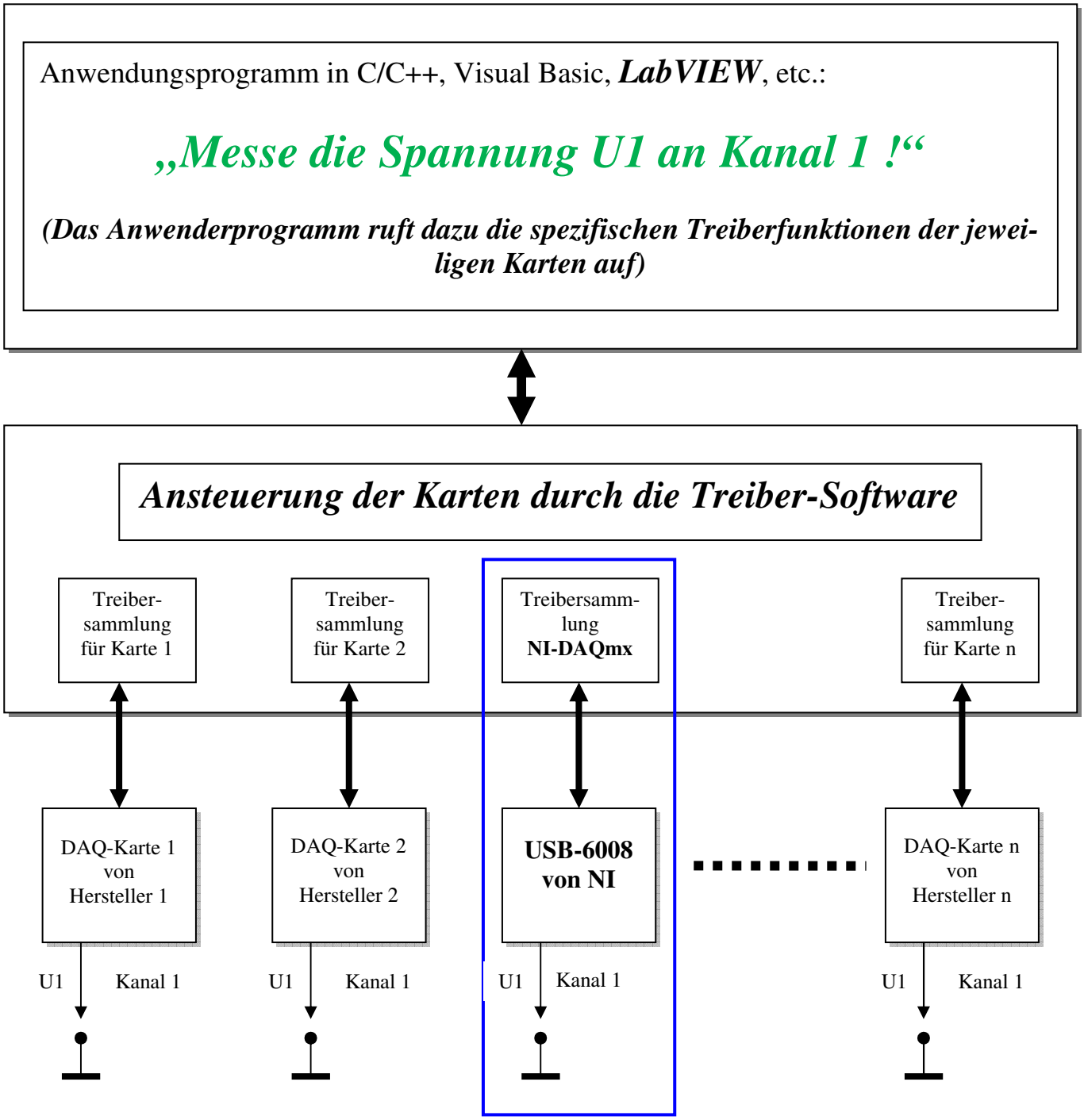


Abb.3.1: Die Ansteuerung von DAQ-Karten bzw. DAQ-Modulen

Und wie sieht hierbei nun die tägliche Praxis aus ?
Der Anwender/Entwickler programmiert in seiner Programmiersprache z.B. ganz einfach:
„Messe (mit der Karte n) die Spannung U1 am Eingangskanal 1 !“

Mehr interessiert ihn an dieser Stelle NICHT: der Anwender möchte nach diesem Mess-Aufruf einfach nur die gewünschte Spannung erhalten und diese dann weiter verarbeiten können.
Er hat kein Interesse:

- am speziellen Aufbau der Hardware der DAQ-Karte,
- am eingesetzten A/D-Wandler und dessen besonderer Ansteuerung,
- am verwendeten µC auf der Karte,
- an den besonderen Special Funktion Registern zum Betrieb des A/D-Wandlers bzw. zum Betrieb der DAQ-Karte,
- etc.

Er will einfach nur seine Spannung messen, unabhängig von der verwendeten DAQ-Karte. Und diesen Wunsch erfüllt ihm nun die **Treiber-Software**, die er in sein Anwendungsprogramm mit eingebunden hat: die hier vorhandenen Funktionen setzen diesen allgemeinen Mess-Wunsch in die jeweils speziellen, individuellen Befehle (auf der Maschinenebene) zum Betrieb dieser einen besonderen Hardware (≡ DAQ-Karte) um und liefern als Rückgabewert an das Anwenderprogramm den gewünschten Spannungsmesswert.

Aus diesem so einfachen klingenden Mess-Konzept ergeben sich jedoch einige wesentliche Konsequenzen:

Konsequenz 1:

Über die zur jeweiligen Karte gehörende Treiber-Software ist ein einfacher Betrieb der Karte mit dem eigentlichen Anwendungsprogramm (z.B. LabVIEW) möglich, ohne dass sich der Anwender um unzählige Details auf der DAQ-Karte kümmern muss.
Der Anwender „sieht“ nur eine einheitliche, für alle Karten (eines Herstellers) gleiche Anwenderschnittstelle. Was „darunter“ passiert, ist ihm völlig egal.

Konsequenz 2:

Für JEDE Karte von JEDEM Hersteller ist ein eigenes Treiber-Software-Paket notwendig, das zwar meistens kostenlos zur DAQ-Karte mitgeliefert wird, aber immer erst auf dem Zielrechner installiert werden muss.

Konsequenz 3:

Für die DAQ-Produkte von National Instruments (NI) bzw. für den Betrieb von LabVIEW in Verbindung mit DAQ-Produkten von NI gibt es im Wesentlichen drei große Gruppen von Treiber-Produkten:

- Die **VISA-Treiber** (Virtual Instruments System Architecture): zum Betrieb von externen DAQ-Einheiten unter LabVIEW, die über serielle Schnittstellen oder serielle Bussysteme an den PC/LapTop angeschlossen werden.
  Hiermit können auch DAQ-Einheiten von anderen Herstellern oder selbst entwickelte DAQ-Module unter LabVIEW betrieben werden, sofern diese über eine passende Schnittstelle verfügen.

(Diese Treiber-Funktionen haben wir in unserem Projekt "**LabVIEW für den Praktiker / LabVIEW meets µC**" kurz vorgestellt und verwendet)

- Die **NI-DAQmx-Treiber**: das ist eine Treiber-Sammlung **speziell nur** für die DAQ-Karten und –Module von NI.
  Hiermit ist ein Betrieb dieser Einheiten nicht nur unter LabVIEW sondern auch unter C/C++, Visual Basic, C#.Net, etc. möglich.
  (Die Treiber-Software für „alte" NI-DAQ-Einheiten heißt NI-DAQ)

- Die **IVI-Treiber** (Interchangeable Virtual Instrumentation): werden bevorzugt dann eingesetzt, wenn es um die Austauschbarkeit und Simulation von komplexen Messgeräten geht.

Konsequenz 4:

Zum Betrieb von **Nicht-NI**-DAQ-Produkten (unter LabVIEW):

- muss man sich die benötigten Treiber selber erstellen

  oder

- man setzt die VISA-Treiber ein, wenn die DAQ-Produkte über geeignete serielle und/oder Bus-Schnittstellen verfügen

  oder

- man setzt die (LabVIEW-)Treiber ein, die der Hersteller selbst für seine DAQ-Produkte anbietet.

Konsequenz 5:

Für den Betrieb unseres USB-6008er-Moduls unter LabVIEW verwenden wir nachfolgend die **NI-DAQmx-Treiber**.

3.1 Die NI-DAQmx-Treiber

Die NI-DAQmx-Software stellt eine Sammlung von Treibern für alle neueren DAQ-Einheiten (Karten und Module) von NI dar.

Die Installation

Die Installation von NI-DAQmx wurde entweder bereits mit der Installation von LabVIEW durchgeführt oder muss jetzt noch abgewickelt werden.
Wenn Sie den ´**Measurement & Automation Explorer (MAX)**´ aufrufen und dort den Punkt ´**Software**´ anwählen, so können Sie erkennen, welche Software-Komponenten von NI auf Ihrem Rechner installiert sind, **Abb.3.1.1**:

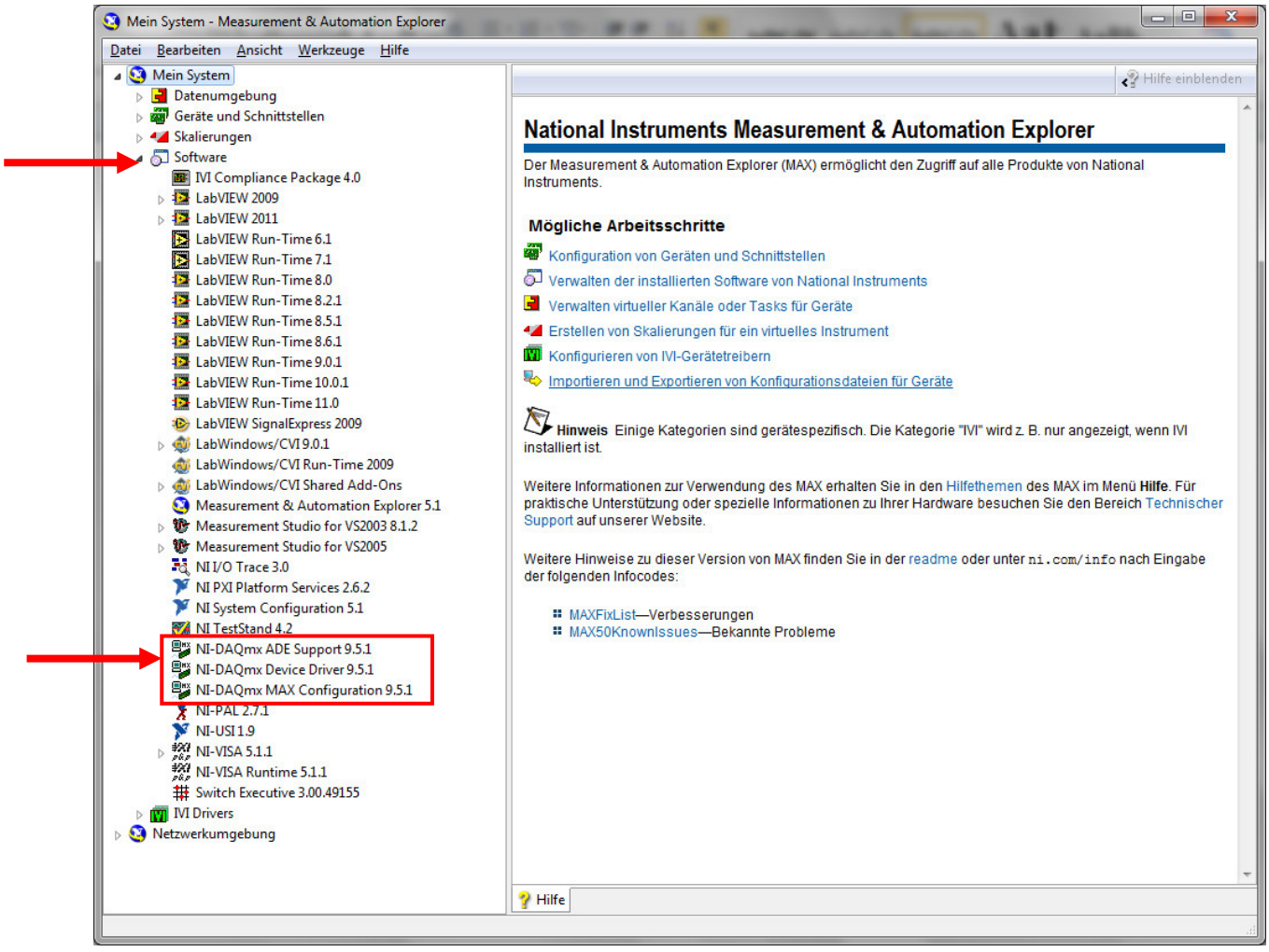


Abb.3.1.1: Die NI-DAQmx-Software ist bereits auf dem Rechner installiert

Sollten Sie die entsprechenden NI-DAQmx-Einträge nicht vorfinden, so müssen Sie diese Software zuerst noch nachinstallieren.
Dazu können Sie zum einen die mitgelieferte CD verwenden oder Sie laden sich die aktuellste Version der Treibersoftware aus dem Internet herunter.

Sie finden dieses Paket am einfachsten, indem Sie z.B. bei Google den Suchbegriff ´NI-DAQmx´ eingeben.
Bereits eines der ersten Suchergebnisse führt Sie zur entsprechenden Seite von NI, von der Sie die Treibersoftware NI-DAQmx herunterladen können.
Die aktuelle Version (Stand: Juli 2012) hat die Nummer 9.5.1 und einen Umfang von 1,31 GB.
Nach dem Download können Sie dieses Treiber-Paket wie gewohnt installieren und darauf hin sollten im MAX die entsprechenden Einträge vorhanden sein.

Die NI-DAQmx-Treiber in LabVIEW2011

Wenn Sie nun LabVIEW aufrufen, so finden Sie auf dem Blockdiagramm unter

BD\Mess-I/O\DAQmax - Datenerfassung

alle vorhandenen NI-DAQmx-Funktionen, **Abb.3.1.2**:

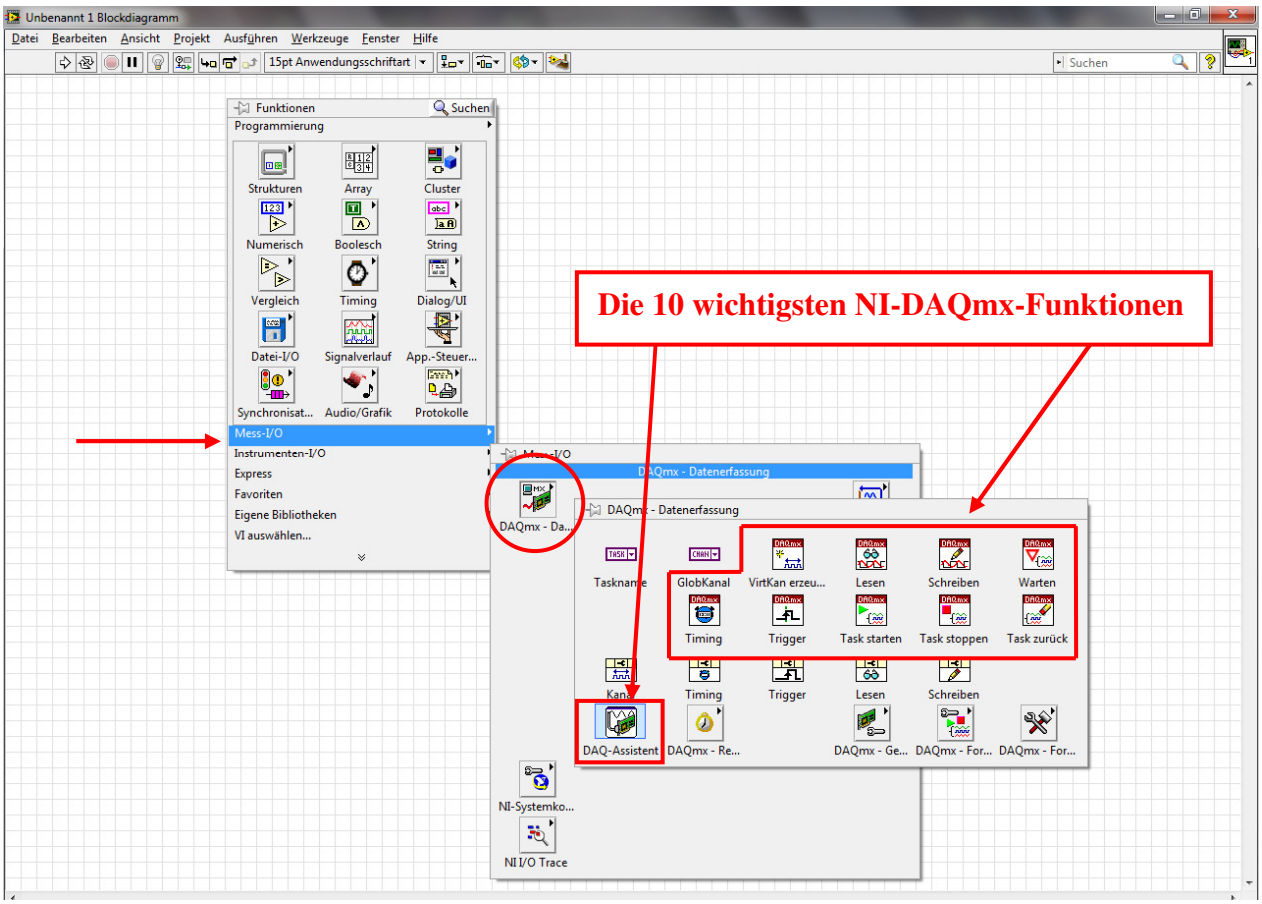


Abb.3.1.2: Der Weg zu den NI-DAQmx-Funktionen

Hier sind zunächst **die 10 wichtigsten NI-DAQmx-Funktionen** aufgeführt, mit denen man bereits eine Vielzahl von DAQ-Aufgaben **mit allen neueren NI-DAQ-Modulen** erledigen kann (in einigen Untermenüs (unten rechts in der letzten Zeile des ´DAQmx-Datenerfassungs´-Fensters) sind noch weitergehende NI-DAQmx-Funktionen für die „aller letzten Feinheiten“ zu finden).
Die direkte Verwendung dieser zehn Grundfunktionen

- spart Entwicklungszeit,
- ergibt eine verbesserte Leistungsfähigkeit des gesamten DAQ-Systems und
- reduziert somit die Entwicklungskosten für das System

da der interne Aufbau dieser Funktionen unmittelbar und optimal auf die jeweilige NI-DAQ-Hardware zugeschnitten ist.

Mit anderen Worten:
Diese zehn Grundfunktionen reichen aus, um mit allen neueren NI-DAQ-Modulen die fünf am Anfang von Kapitel 3 erwähnten DAQ-Aufgaben zu erledigen.
Der Anwender hat es also immer nur mit diesem gleichen Satz dieser zehn Funktionen zu tun, egal welche NI-DAQ-Hardware er verwendet !

Ein kleiner Nachteil ergibt sich jedoch für den LabVIEW-Erstanwender/Einsteiger an dieser Stelle:
Wie bei LabVIEW üblich, sind auch diese NI-DAQmx-Funktionen äußerst leistungsfähig und damit sehr komplex aufgebaut, oder anders formuliert: der Anwender kann bei jeder Funktion sehr viel selber einstellen und konfigurieren, **Abb.3.1.3**:

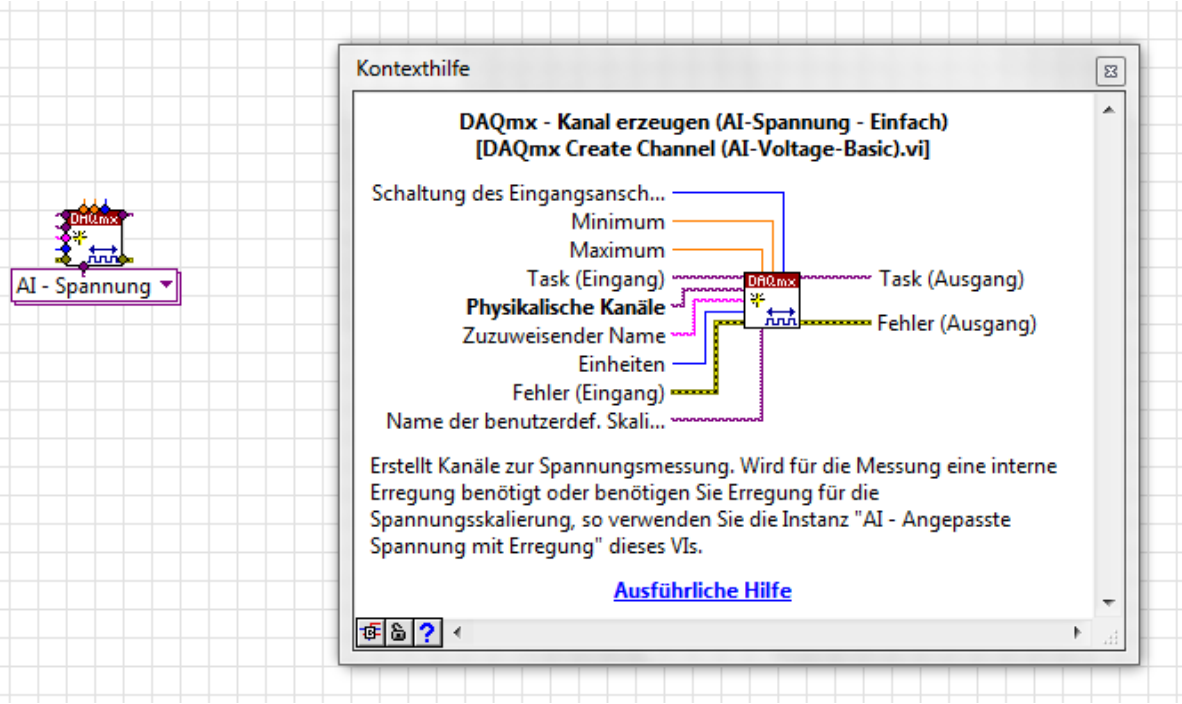


Abb.3.1.3: Der konfigurierbare Umfang der einzelnen NI-DAQmx-Funktionen ist teilweise beachtlich

Hier ist einmal die Funktion ´**DAQmx-Kanal erzeugen**´ mit der zugehörigen Kontexthilfe dargestellt.
Man erkennt, dass man sehr viele Eigenschaften auswählen und festlegen kann.
Nach einer gewissen Einarbeitungszeit (und nach viel Lesen in der Kontexthilfe) ist diese direkte und unmittelbare Verwendung der NI-DAQmx-Funktionen sicherlich kein Problem mehr.

Für den Anfänger und Ersteinsteiger in LabVIEW bzw. für kleine bis mittelgroße DAQ-Projekte bietet LabVIEW aber noch **einen einfacheren Weg** zum Einsatz der NI-DAQmx-Funktionen an: in der Abb.3.1.2 ist in der unteren linken Ecke der so genannte **DAQ-Assistent** (auch **DAQ-Wizard** genannt) aufrufbar.
Dieses Tool dient zur vereinfachten Erstellung von DQA-Anwendungen (unter LabVIEW auf der Grundlage von NI-DAQ-Modulen), **OHNE** dass die einzelnen benötigten NI-DAQmx-Funktionen vom Anwender selber explizit ausgewählt, zusammengestellt, verbunden und konfiguriert werden müssen.
In einem einfachen, **anwenderfreundlichen interaktiven Dialog** erfasst der DAQ-Assistent die „Wünsche“ der Anwenders, stellt darauf hin (intern) die benötigten NI-DAQmx-Funktionen zusammen, verbindet und konfiguriert diese und übergibt dem Anwender auf dem LabVIEW-Blockdiagramm ein fertiges Express-VI, das die gewünschten Aufgaben erfüllt und vom Anwender sofort genutzt werden kann.
Will man z.B. eine bestimmte Spannung an einen Eingangskanal des DAQ-Moduls messen, so gibt das vom DAQ-Assistent erstellte Express-VI diese Spannung an seinem Ausgang aus und um mehr muss sich der Anwender gar nicht kümmern. Er kann diesen Messwert dann sofort in seinem VI weiter verarbeiten.

Fazit:

Zur Erstellung eines DAQ-VIs unter LabVIEW gibt es daher **zwei verschiedene Realisierungsansätze**:

1) **Man verwendet direkt die einzelnen NI-DAQmx-Funktionen**: stellt sie selber zusammen und konfiguriert diese entsprechend seine Bedürfnissen.
   So lässt sich „das Allerletzte“ aus der das DAQ-Hardware „herausholen“.
   Dieses Verfahren ist nach einer gewissen Einarbeitungszeit sehr gut anwendbar und wird für große oder ganz spezielle DAQ-Aufgaben empfohlen.

2) Man beginnt als Anfänger/Ersteinsteiger mit dem **DAQ-Assistenten** und erhält damit innerhalb kürzester Zeit auf einfachem Wege optimal zusammengestellte LabVIEW-Express-VIs, die die DAQ-Aufgabe genauso gut erfüllen.
   Somit lernt man zunächst den Umgang mit der jeweiligen DAQ-Hardware und die Möglichkeiten der DAQmx-Funktionen kennen (die ja intern im Express-VI vom DAQ-Assistenten eingesetzt werden).

Möchten man dann später einmal, das gesamte VI bis an die Grenze optimieren, so kann man sich dann selber die NI-DAQmx-Funktionen geeignet zusammen stellen. Das ist aber sehr oft gar nicht notwendig, **da der DAQ-Assistent schon hervorragende Ergebnisse liefert**.

Wir werden Ihnen nachfolgend den Einsatz des DAQ-Assistenten in Verbindung mit dem USB-6008 näher vorstellen.

Und zum Abschluss dieses Kapitels müssen Sie noch einige Begriffe kennen lernen, die im LabVIEW-Sprachgebrauch sehr häufig bei DAQ-Anwendungen benutzt werden:

Physikalischer Kanal (Physical channel):
Ein direkter Anschluss-Pin an der DAQ-Hardware, über den Signale ausgegeben oder eingelesen werden können.
Also z.B. der direkte Eingangs-Pin eines A/D-Wandler-Bausteins im DAQ-Modul oder der Ausgabe-Pin bei einem D/A-Wandler.

Virtueller Kanal (Virtual channel)
Das ist dann ein eigentlicher Mess-/Ausgabe-Kanal unter LabVIEW. Dem rein physikalischen Kanal werden noch weitere Informationen bzw. Eigenschaften zugeordnet, die für seine endgültige Verwendung im DAQ-System wesentliche sind, z.B.:

- ein entsprechender Name für den Kanal,
- Angabe der Abtastrate für diesen Kanal, also Anzahl der Abtastungen (Messungen) je Sekunde,
- Einstellung von Triggerbedingungen zur Auslösung der Messungen / der Ausgabe,
- Festlegung des Messbereiches und der Skalierung,
- etc.

All diese Zusatzinformationen, zusammen mit dem zu Grunde liegenden physikalischen Kanal, bilden dann den virtuellen Kanal, der in LabVIEW verwendet wird (≡ Gesamtheit aller Einstellungen zu einem physikalischen Kanal).
Die virtuellen Kanäle bilden somit die Grundlage für LabVIEW-DAQ-VIs, wobei es solche Kanäle für die Erfassung von Messwerten und auch für die Ausgabe von Signalen gibt.

Messaufgabe (Messprojekt oder Task)
Eine komplette (in sich abgeschlossene) Mess- und Steuerungsaufgabe wird als Task bezeichnet und besteht i.a. aus mehreren virtuellen Kanälen und den entsprechenden Verarbeitungsroutinen für die Messwerte bzw. aus den Ausgaberoutinen für die Steuersignale.

3.2 Der DAQ-Assistent von NI

Zum Einsatz des DAQ-Assistenten

- schließen Sie das USB-6008 an Ihren Rechner an und überprüfen Sie, ob dieses Modul ordnungsgemäß funktioniert, d.h. führen Sie den Selbsttest im MAX aus (s. Kap. 2.1)

- öffnen Sie unter LabVIEW ein neues VI und schalten Sie um auf das Blockdiagramm.

Fügen Sie nun den DAQ-Assistenten auf dem Blockdiagramm ein.
Sie finden ihn unter:

BD\Mess-I/O\DAQmax – Datenerfassung\DAQ-Assistent

Nach der Platzierung des Funktionssymbols auf dem Blockdiagramm beginnt sofort die erste Grundinitialisierung des DAQ-Assistenten, **Abb.3.2.1**:

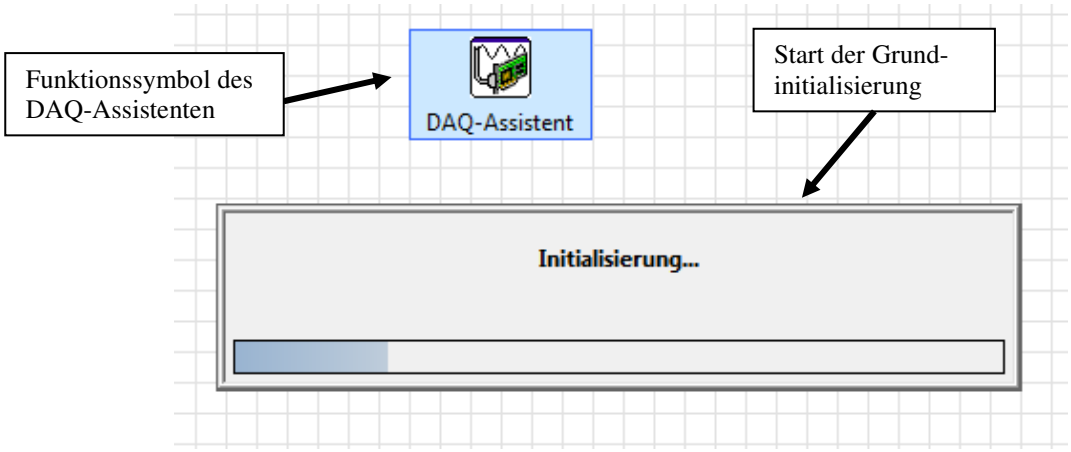
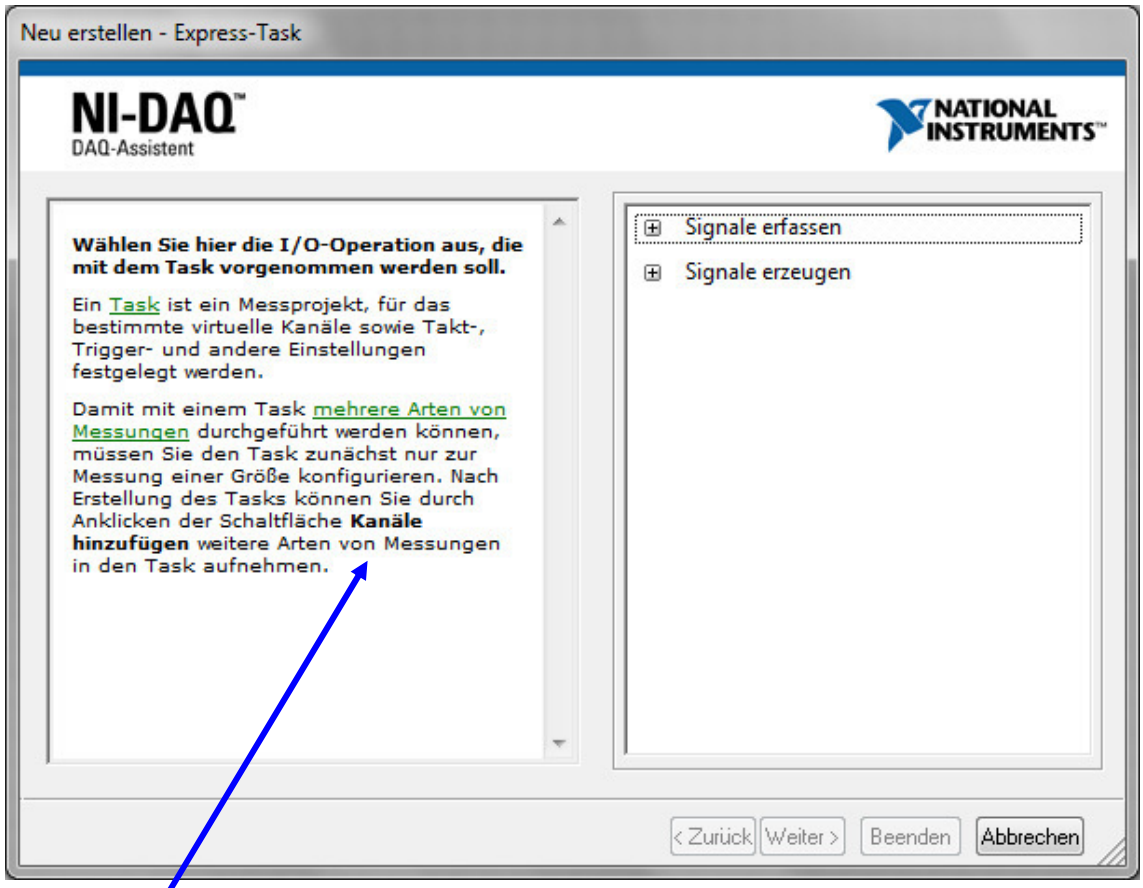


Abb.3.2.1: Der erste Aufruf des DAQ-Assistenten

Und danach startet der **interaktive Einstellungsprozess** zur Generierung des Express-VIs, **Abb.3.2.2**:



Erklärungen zu den jeweiligen Inhalten bzw. zu den Auswahl-/Einstellmöglichkeiten im rechten Fenster. Durch Klicken auf die **grün hinterlegten Stichworte** erhält man weitergehenden Informationen dazu.

Abb.3.2.2: Der Start des interaktiven Einstellungsprozesses

Wir machen den weiteren Ablauf am **Beispiel einer Spannungsmessung** klar:
Erweitern Sie daher den Punkt ´**Signale erfassen**´ im rechten Fenster durch Klicken auf das ´+´-Kästchen**, Abb.3.2.3**:

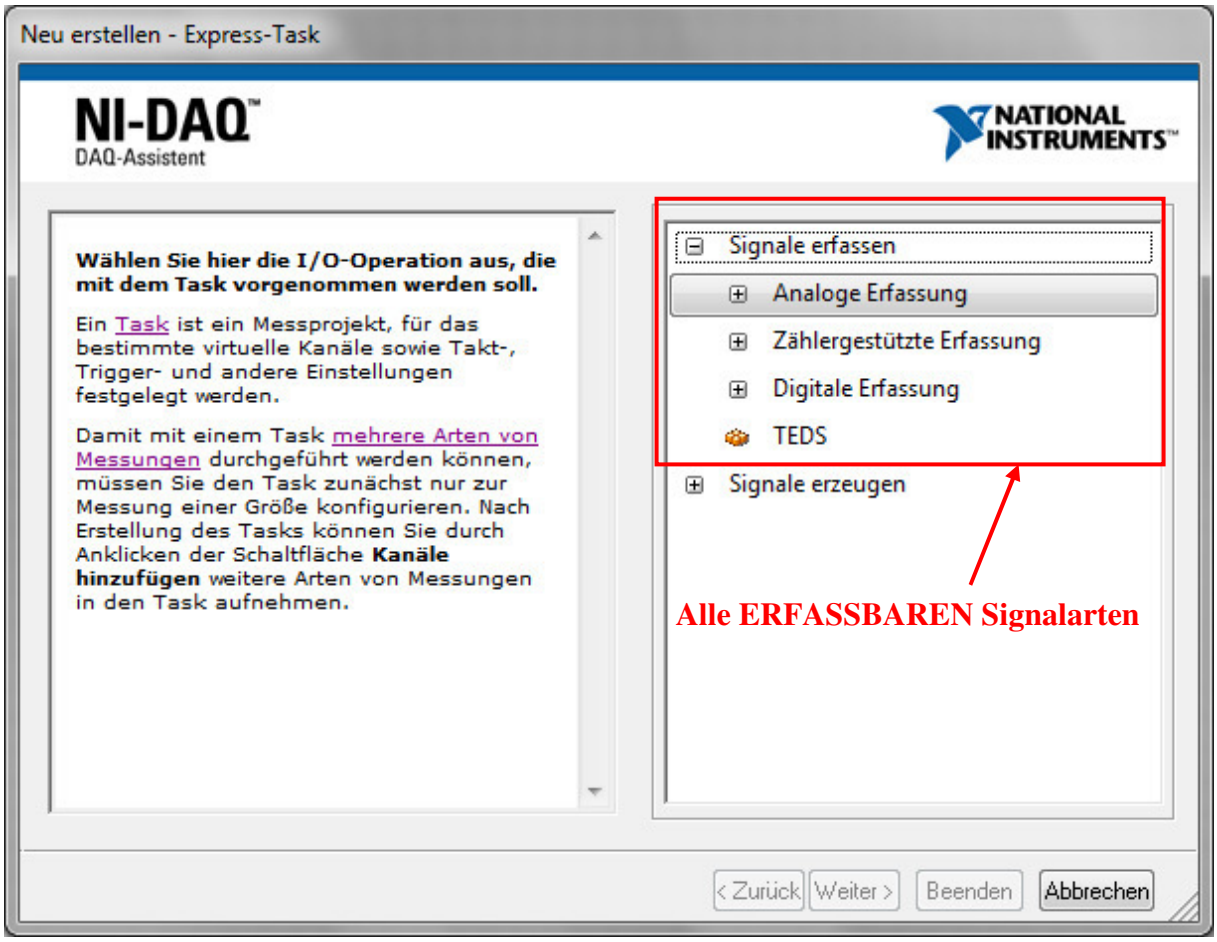


Abb.3.2.3: Die Signalerfassung – ganz allgemein

Es werden nun grundsätzlich erst einmal ganz allgemein alle in einem DAQ-System erfassbaren Signalarten (Signalgruppen) angezeigt (s. Kap.3).
Für die Spannungsmessung wählen wir die ´**Analoge Erfassung**´ aus, **Abb.3.2.4**:

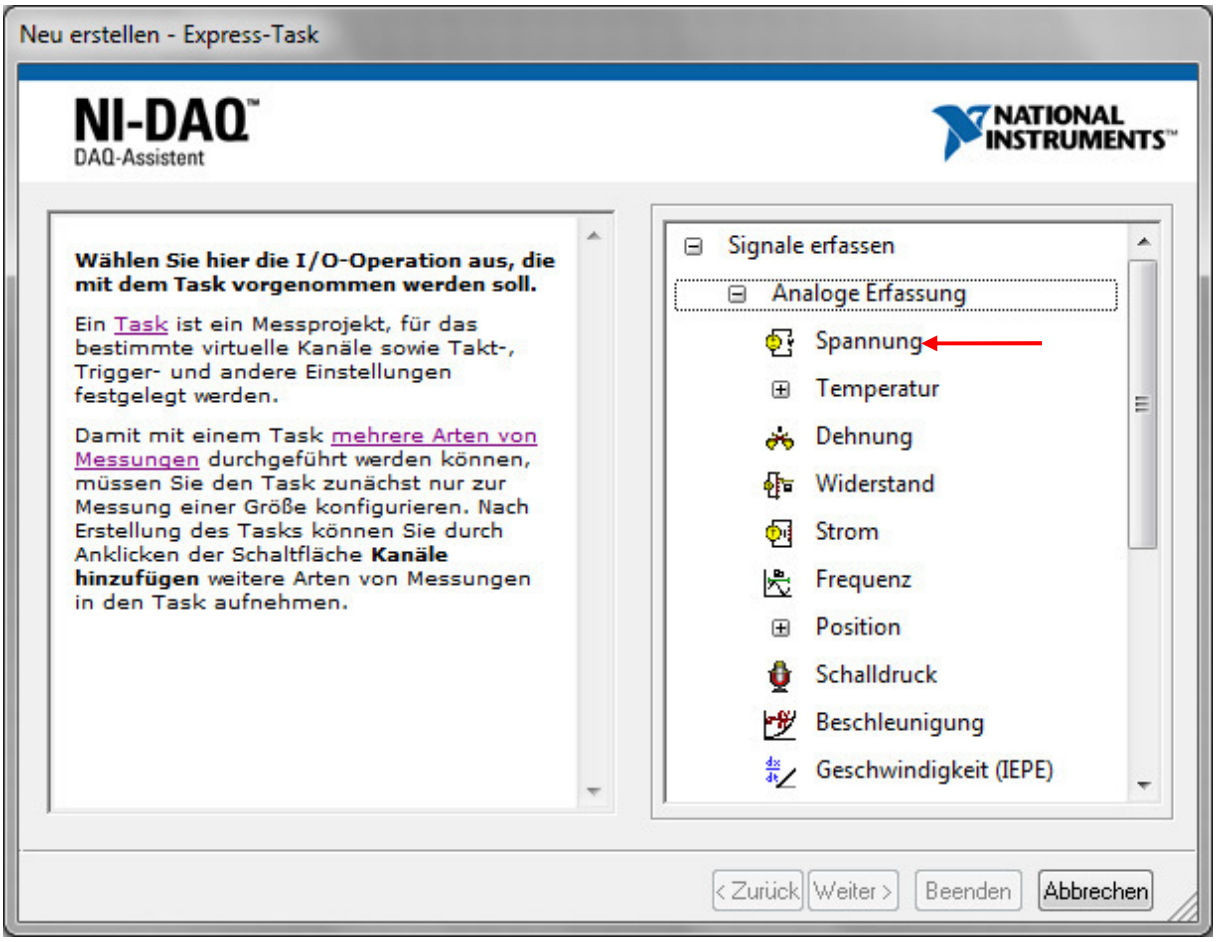


Abb.3.2.4: Die „weite Welt“ der erfassbaren analogen Eingangssignale

Im nun erscheinenden Menü wird eine große Vielfalt von erfassbaren analogen Größen aufgelistet, wobei es zu einigen Punkten sogar noch Untermenüs mit weiteren Eintragungen gibt (Temperatur, Position, Kraft, …).
Wir wählen gemäß unserem Mess-Wunsch die analoge Größe ´Spannung´ aus und nun zeigt sich „schlagartig“ **die große Leistungsfähigkeit des DAQ-Assistenten, Abb.3.2.5**:

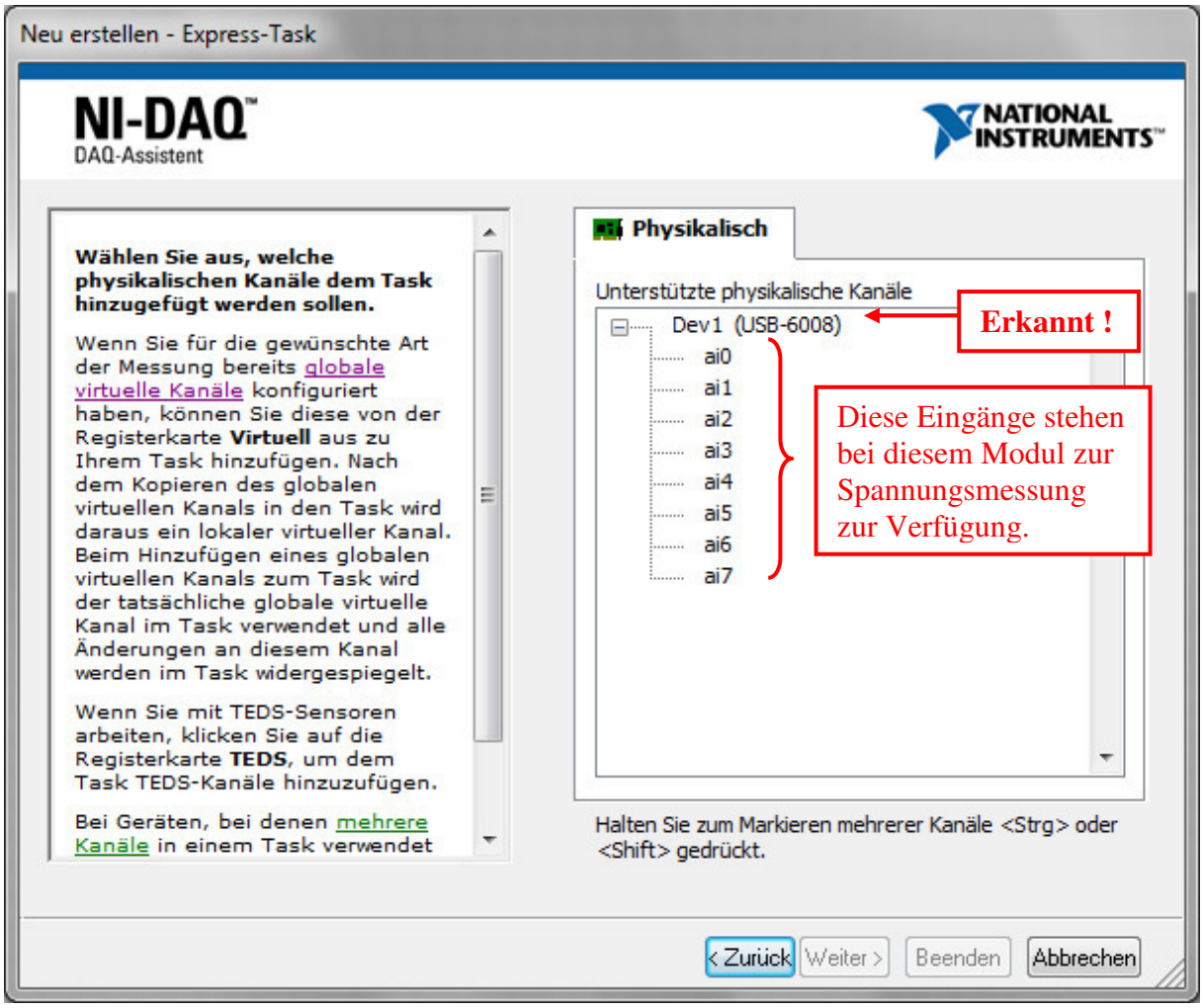


Abb.3.2.5: Der DAQ-Assistent erkennt sie alle !

Mit Hilfe von MAX erkennt der DAQ-Assistent sofort alle angeschlossenen DAQ-Module von NI mit allen Ihren Eingängen, die zur Spanungsmessung geeignet sind und bietet diese dem Anwender nun zur individuellen Auswahl an !

In unserem Beispiel bedeutet das:
Der DAQ-Assistent hat ein USB-6008-Modul erkannt und diesem zunächst dem Namen ´Dev1´ gegeben bzw. er hat diesen Namen vom MAX übernommen, s. Abb.2.1.4.
Diese Bezeichnung kann man natürlich später individuell den Bedürfnissen der Anwendung anpassen (Kontextmenü zu diesem Gerät im MAX aufrufen und den Punkt ´Umbenennen´ auswählen).
Weiterhin „weiß“ der DAQ-Assistent nun, dass eine USB-6008er-Einheit insgesamt acht Eingänge (physikalische Kanäle) zur analogen Spannungsmessung besitzt und bietet diese dem Anwender daher zur Auswahl an.

Für einen kleinen (Zwischen)Test haben wir einmal ein zweites USB-6008er-Modul über einen weiteren USB-Anschluss an unseren Rechner angeschlossen: nach einige automatisch ablaufenden Installationsvorgängen (wie man diese von USB-Geräten her kennt), hat der MAX diese neue Einheit erkannt und stellt sie dem DAQ-Assistenten zur Verfügung, d.h. wenn man jetzt den DAQ-Assistenten (neu) aufruft, erscheinen die beiden Module zur Auswahl, **Abb.3.2.6**:

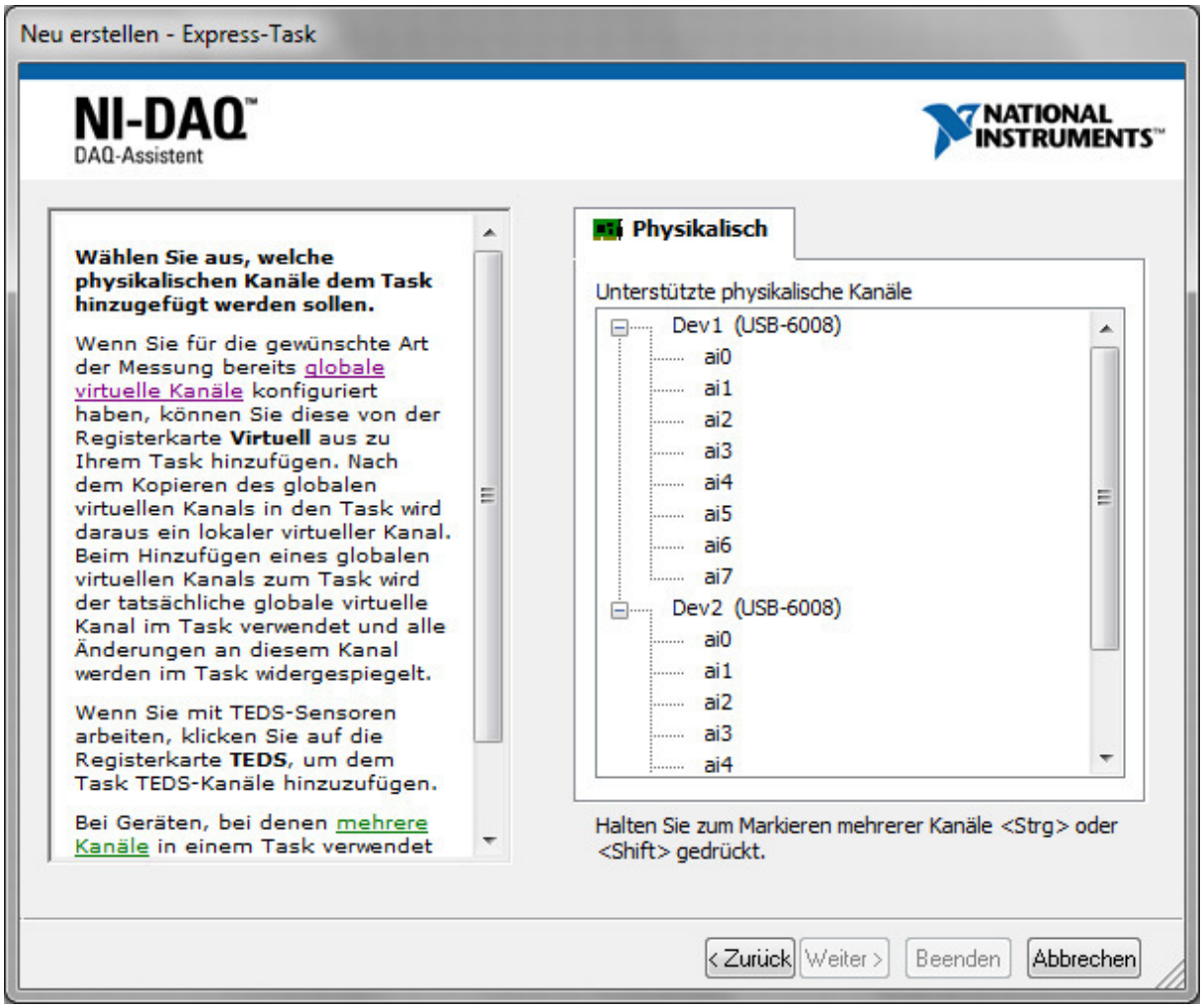


Abb.3.2.6: Dem DAQ-Assistenten entgeht nichts !

Kommen wir nun zurück zu unserer Aufgabe der Spannungsmessung und der Abb.3.2.5.
Wir müssen nun einen Anschluss (Kanal) zur Spannungsmessung auswählen: wir nehmen den Kanal ´**ai0**´ (≡ Analog-IN-Kanal 0) und klicken auf ´Beenden´ (das ist zwar ein komischer Name zur Bestätigung der Auswahl, aber so ist nun einmal an einigen Stellen die deutsche Übersetzung der Knopf-Beschriftungen von NI gewählt worden).
Und nun erscheint **das Hauptfenster des DAQ-Assistenten**, in dem alle wesentlichen Eingaben und Festlegungen zur Durchführung der Spannungsmessung gemachten werden (also zur Erzeugung des virtuellen Kanals mit dem Eingang ´ai0´, der hinterher in LabVIEW benutzt werden kann).
Zunächst ist hier die Registerkarte ´**Konfiguration**´ geöffnet, **Abb.3.2.7**:

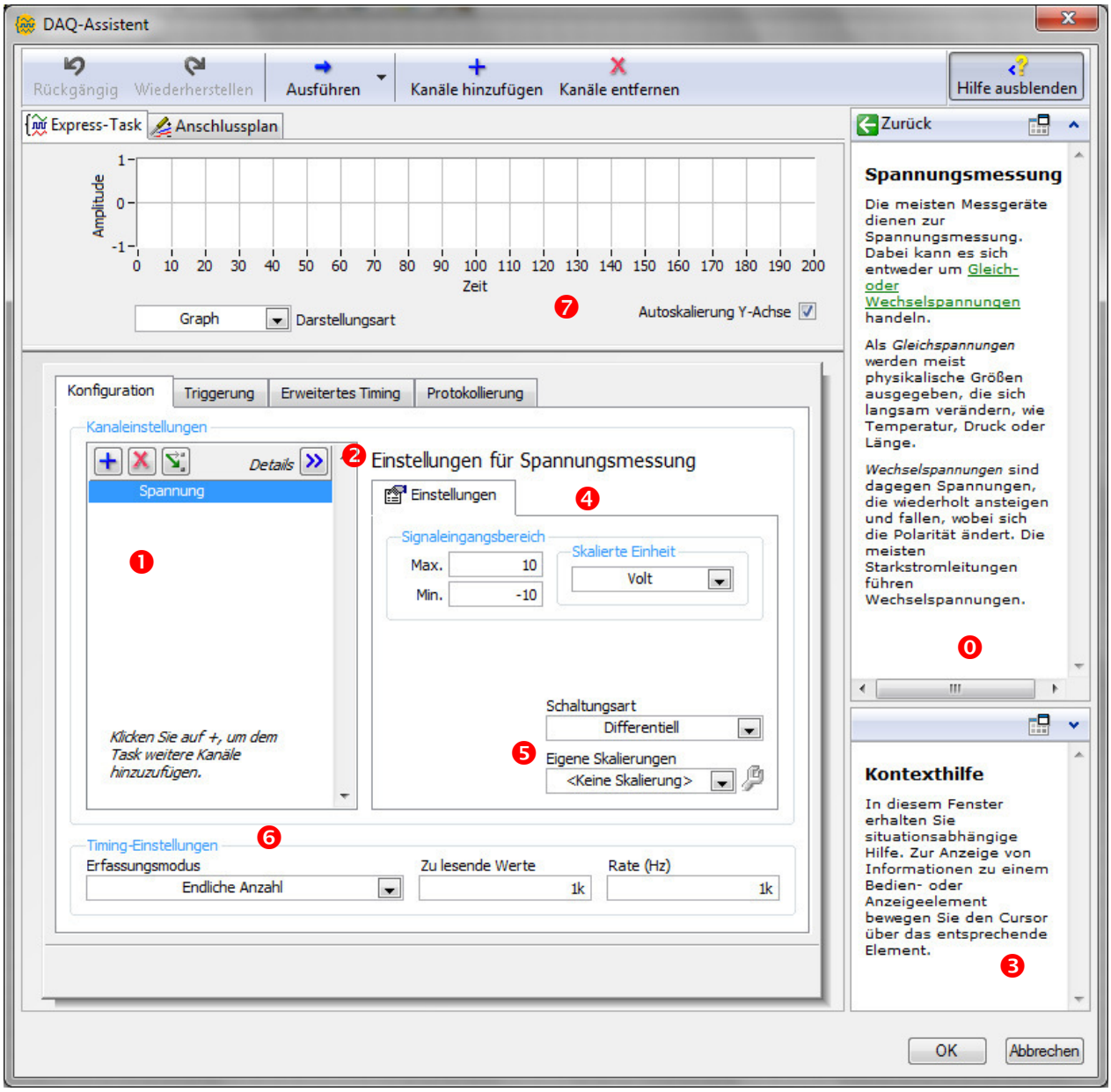


Abb.3.2.7: Das Hauptfenster des DAQ-Assistenten

⓪

In diesem Fenster finden Sie ganz allgemeine Zusatzinformationen zu der von Ihnen ausgewählten bzw. festgelegten Messung.
Hier also: zur analogen Spannungsmessung.

❶

In diesem Fenster sind die (physikalischen) Kanäle aufgeführt, die zur Messung eingerichtet wurden.
In unserem Fall: ein Kanal zur Spannungsmessung.

Fahren Sie mit dem Mauszeiger in dieses Feld, so erhalten Sie im Fenster ❸ zusätzliche Informationen zu den dortigen Eintragungen.

❷

Wenn Sie jetzt auf den Doppelpfeil rechts neben ´**Details**´ klicken, erhalten Sie weitere Informationen zu diesem Kanal, **Abb.3.2.8**:

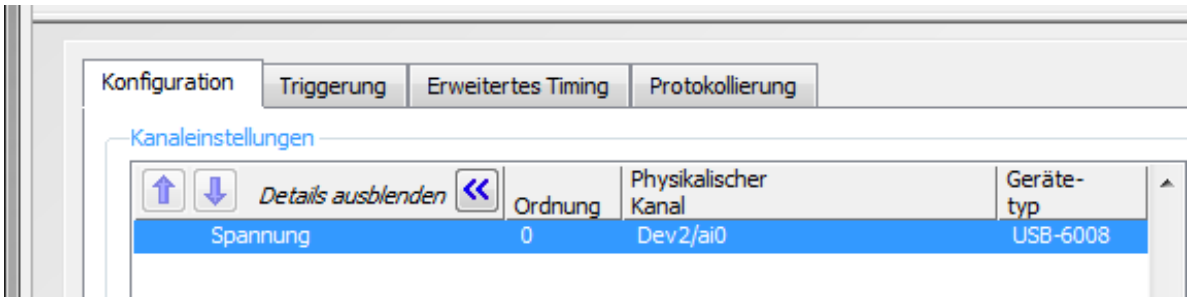


Abb.3.2.8: Weitere Informationen zum eingerichteten Spannungsmess-Kanal

Sie erkennen:
Es wird hier die Spannung am Kanal ´**ai0**´ am einem **USB-6008**-Gerät mit der Bezeichnung ´**Dev2**´ gemessen.
(Wir benutzen ab jetzt das USB-6008er-Modul mit dem Namen ´Dev2´)

❹

In diesem Feld können Sie nun die Einstellungen für die Spannungsmessung durchführen: minimale und maximale Meßspannung und die Einheit dazu.
Tragen Sie hier einmal ein: Messbereich von 0 … 10 V und Einheit ´V´ (eine andere Einheit lässt sich hier auch nicht auswählen).
Und beachten Sie: immer wenn Sie mit der Maus über ein Eingabe- bzw. Festlegungsfeld fahren, erscheinen im Fenster ❸ zusätzliche Erläuterungen dazu.

❺

Hier legen Sie die Art der Spannungsmessung fest:

- **RSE** (≡ Referenced single-ended): Spannungsmessung gegen Masse.
- **Differentiell**: Spannungsmessung zwischen jeweils zwei Eingängen (s. dazu auch Kap. 2).

Wählen Sie hier ´**RSE**´ aus.
Im Fenster ´**Eigene Skalierungen**´ machen wir keine weiteren Eintragungen.

❻

In den drei Feldern zu den ´**Timing-Einstellungen**´ wird festgelegt, wie viele Messwerte und wie schnell diese erfasst werden:

- ´**Rate (Hz)**´: hier gibt man an, wie schnell das Meßsignal abgetastet werden soll, also die Abtastrate in Abtastungen pro Sekunde.
  Das Maximum beim USB-6008er beträgt 10 kHz, also 10.000 Abtastungen pro Sekunde.
  Stellen Sie hier einen Wert von 4 ein.

- ´**Zu lesende Werte**´: hier wird angegeben, wie viele Messwerte insgesamt bei jedem Start dieser Messaufgabe (Task) erfasst werden sollen.
  Tragen Sie hier einen Wert von 20 ein.

  Diese beiden zuletzt getätigten Einstellungen bedeuten nun: wenn die Task (das Express-VI) gestartet wird, werden insgesamt 20 Messwert mit einer Abtastfrequenz von 4 Messwerten pro Sekunde erfasst. Die gesamte Messung dauert insgesamt also 5 Sekunden.

- ´**Erfassungsmodus**´: Mit dieser Einstellung wird festgelegt, ob die Messaufgabe nach einer endlichen Anzahl von Messungen erledigt ist, ob kontinuierlich (durch) gemessen werden soll, oder ob bei jedem Aufruf der Task (bei jedem Aufruf des erzeugten Express-VIs) immer nur ein Wert gemessen werden soll.
  Im letzten Fall haben die Festlegungen bei ´Zu lesende Werte´ und bei ´Rate(Hz)´ natürlich keine Wirkung und diese Felder sind daher ausgegraut.
  Lassen Sie hier die Einstellung auf ´Endliche Anzahl´ stehen.

Damit sollten Ihre Einstellungen so aussehen, wie in **Abb.3.2.9** dargestellt:

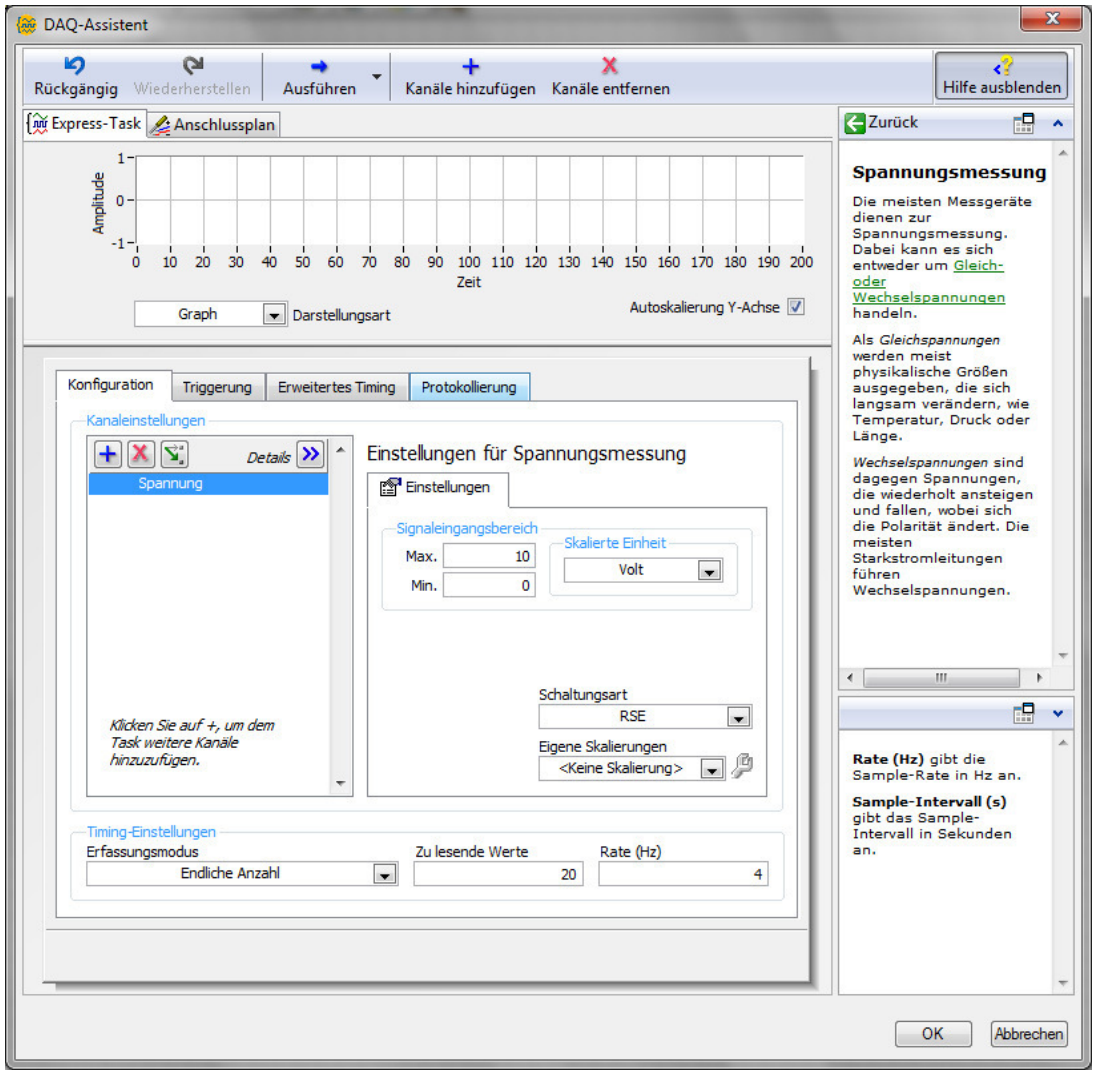


Abb.3.2.9: Die ersten Grundeinstellungen im DAQ-Assistenten

❼

In diesem Fenster findet man eine weitere, sehr interessante Eigenschaft des DAQ-Assistenten: hier können Sie ihre Einstellungen testen, schon reale Messwerte mit dem USB-6008er erfassen und auf einfache Art und Weise darstellen.
Klicken Sie dazu auf den Button ´**Ausführen**´ und das Messprojekt (Task) wird einmal ausgeführt, d.h. in unserem Fall: 20 Messwerte mit einer Abtastrate von 4 Hz werden erfasst (es wird also 5 Sekunden lang gemessen) und die Ergebnisse anschließend graphisch dargestellt, **Abb.3.2.10**:

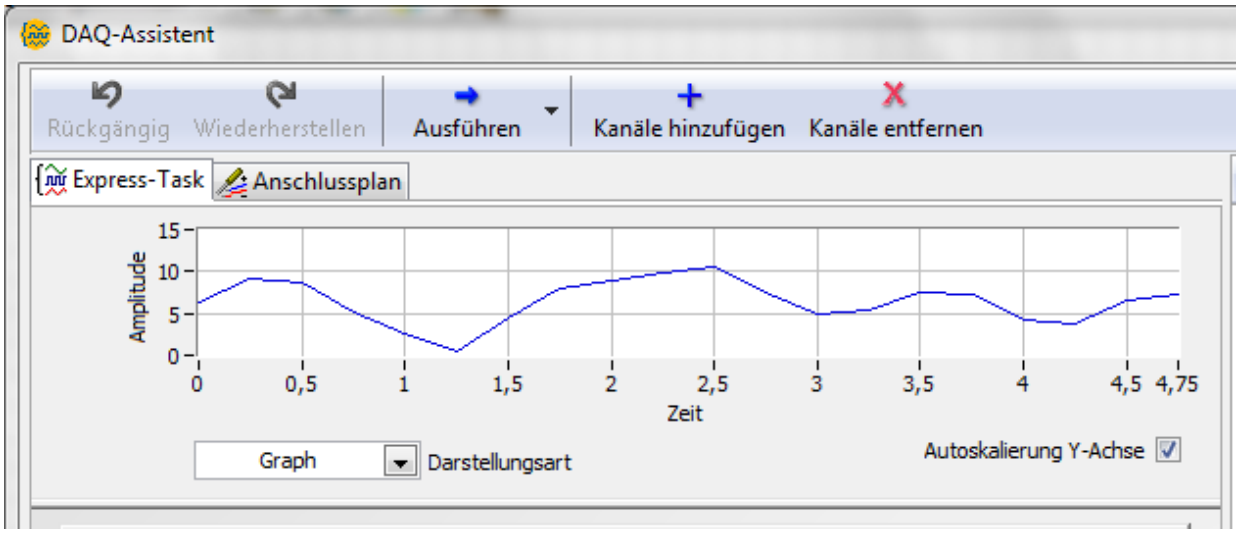


Abb.3.2.10: Eine erste Testmessung wird durchgeführt

In diesem Beispiel haben wir einfach ein regelbares Netzteil am Anschluss ´ai0´ angeschlossen und die Spannung fünf Sekunden lang rauf und runter geregelt.
So kann man sehr schnell und einfach überprüfen, ob und wie die Spannungsmessung an diesem Kanal funktioniert.

Aber der Komfort des DAQ-Assistenten geht noch weiter: durch Klicken auf den Button ´**Anschlussplan**´ (in Abb.3.2.10) erhält man einen detaillierten Übersichtsplan, wie der entsprechende Anschluss am USB-6008er Hardware-mäßig zu verdrahten ist, damit die Spannungsmessung wie gewünscht, d.h. wie eingestellt, funktioniert, **Abb.3.2.11**:

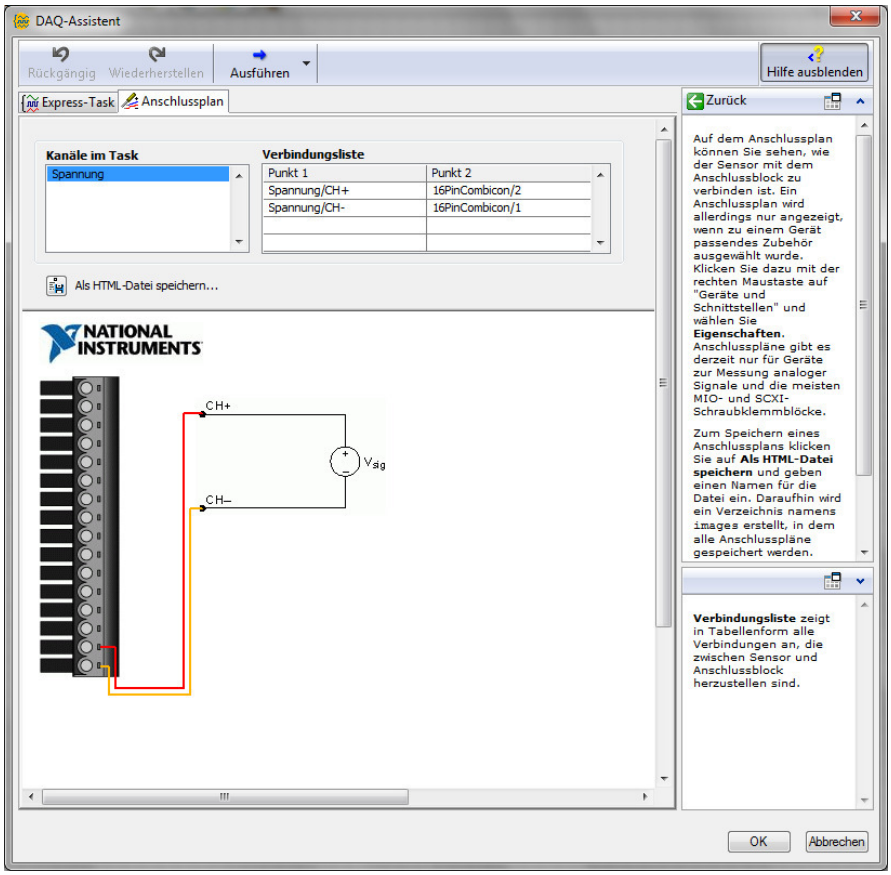


Abb.3.2.11: Ein detaillierter Anschlussplan mit Verbindungsliste hilft weiter

Die anderen Buttons und Registerkarten in diesem DAQ-Fenster sind für unser Projekt zunächst nicht weiter von Interesse und daher können Sie jetzt auf ´OK´ klicken.
Aus den von Ihnen gemachten Eingaben bzw. Einstellungen erzeugt der DAQ-Assistent nun „blitzschnell“ im Hintergrund das benötigte **Express-VI**, das auf den NI-DAQmx-Funktionen beruht, **Abb.3.2.12**:

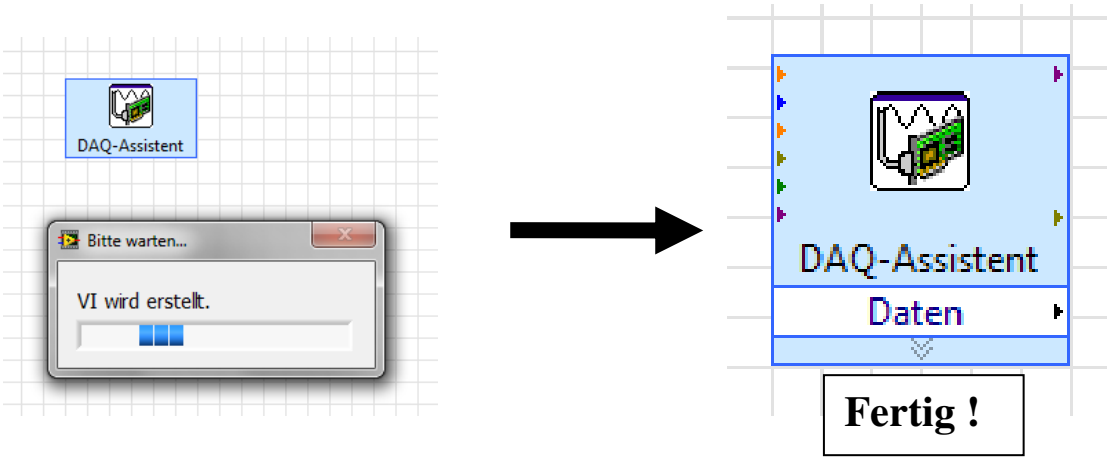


Abb.3.2.12: Das Express-VI entsteht

Am Ausgang ´**Daten**´ des Express-VIs kann nun die gemessene Spannung abgegriffen und weiter verarbeitet werden (dazu kommen wir gleich).

Möchte man zuvor noch wissen, wie der DAQ-Assistent das Express-VI eigentlich aus den einzelnen NI-DAQmx-Funktionen zusammengebaut hat, d.h. möchte man einen Blick in den internen Aufbau des Express-VIs werfen, so ist das über das **Kontextmenü** zum Express-VI recht einfach möglich, **Abb.3.2.13**:

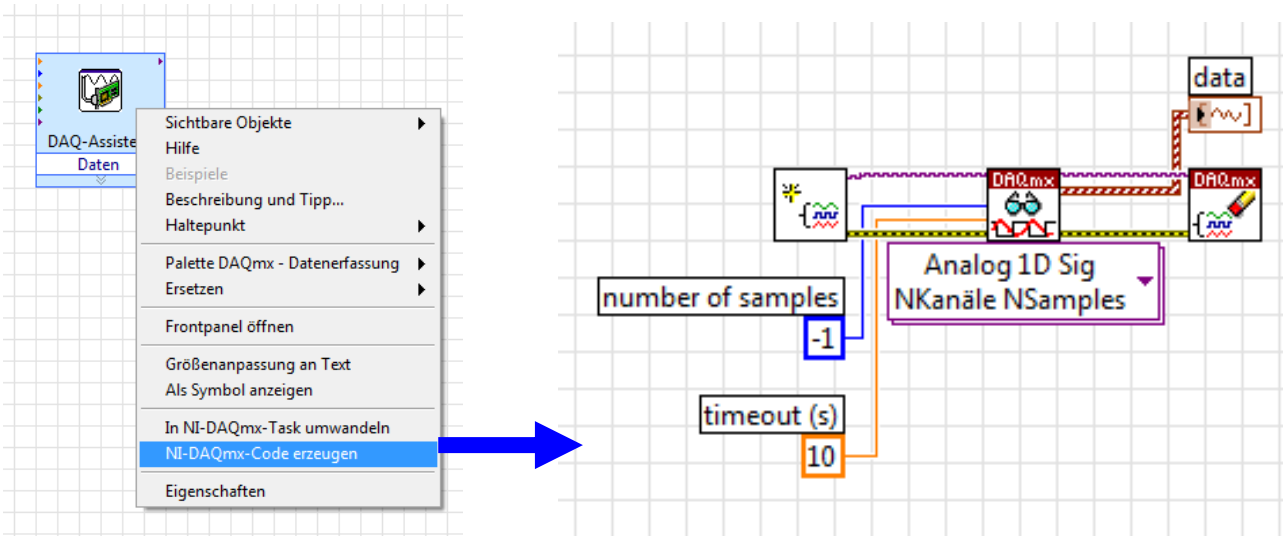


Abb.3.2.13: Die Interna des erzeugten Express-VIs

Wählt man hier den Menüpunkt ´**NI-DAQmx-Code erzeugen**´ an, so stellt LabVIEW nach einigen Sekunden die interne Struktur des Express-VIs dar.
Man erkennt somit, welche NI-DAQmx-Funktionen wie verwendet worden sind und wie die Parametrierung dieser Funktionen aussieht.
Jetzt könnte man noch eigene Änderungen/Verbesserungen durchführen, was wir hier allerdings nicht machen, denn das Express-VI ist vom DAQ-Assistenten schon optimal aufgebaut worden.
Machen Sie daher diese Ansicht mit dem Short-Cut ´Strg+Z´ wieder rückgängig (die Frage nach der Abspeicherung der Änderungen verneinen Sie).

Sie können dem Express-VI nun noch einen passenden Namen geben, der die Aufgabe dieses VIs besser beschreibt:
Klicken Sie dazu mit der Maus zweimal auf das Textfeld ´**DAQ-Assistent**´: der Text wird schwarz hinterlegt und Sie können ihn ändern, z.B. in ´**Spannungsmessung an ai0**´, **Abb.3.2.14**:

Abb.3.2.14: Passende Namen verbessern das Verständnis im Blockdiagramm

Zum Abschluss schließen wir jetzt noch eine ´**Graphanzeige**´ und ein ´**Numerisches Anzeigeelement**´ an den Datenausgang des Express-VIs an, **Abb.3.2.15**:

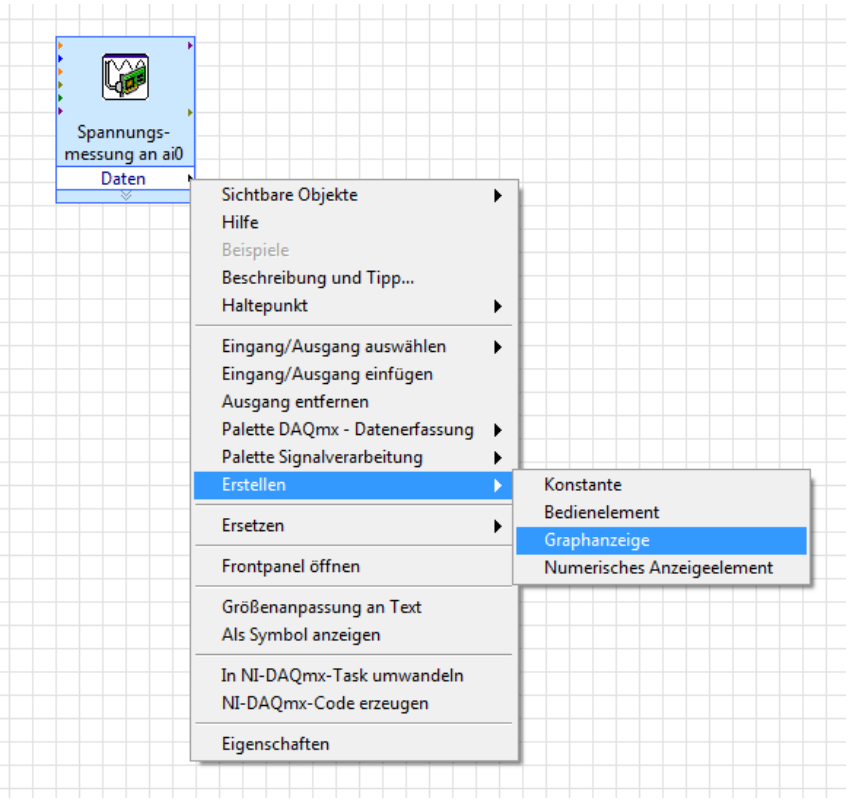


Abb.3.2.15: Die gemessenen Spannungswerte werden sichtbar gemacht

Wählen Sie dazu im Kontextmenü des Datenausgangsanschlusses den Menüpunkt ´Erstellen´ und wählen Sie dann die Punkte ´**Graphanzeige**´ und danach ´**Numerisches Anzeigeelement**´ aus.
Auf dem Blockdiagramm werden diese beiden Visualisierungselemente eingefügt. Nennen Sie die graphische Anzeige ´**Spannungsverlauf**´ und die Zahlenanzeige ´**Spannung / V**´ und verbinden Sie diese Anzeigen mit dem Datenausgang des Express-VIs, **Abb.3.2.16**:

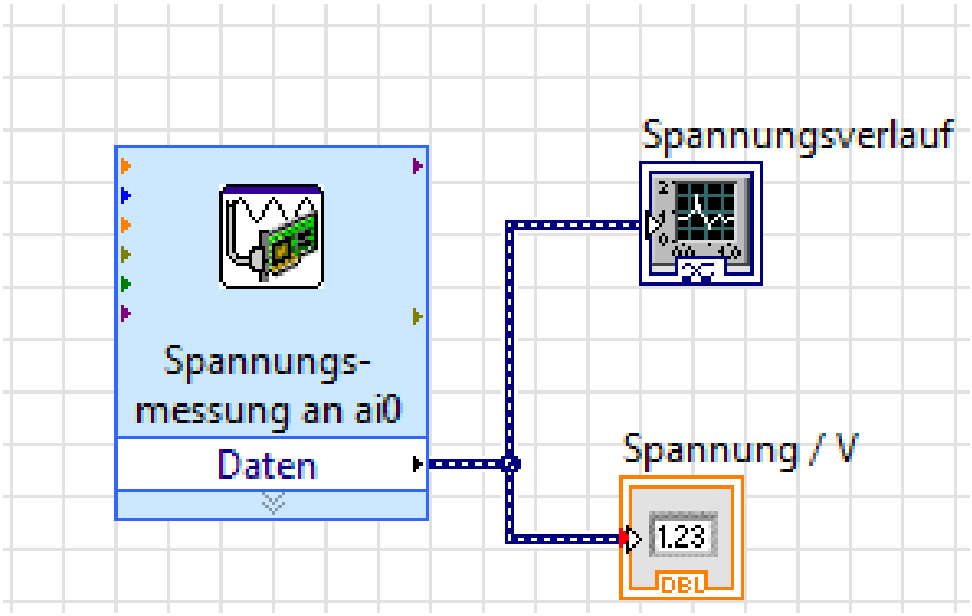


Abb.3.2.16: Die Visualisierung des gemessenen Spannungswerts

Somit können wir nun die erfassten Spannungswerte zahlenmäßig und in Form eines Kurvenverlaufs sichtbar machen.
Schalten Sie daher um auf das Frontpanel und gestalten Sie dieses nach Ihren Vorstellungen, z.B. wie in **Abb.3.2.17**:

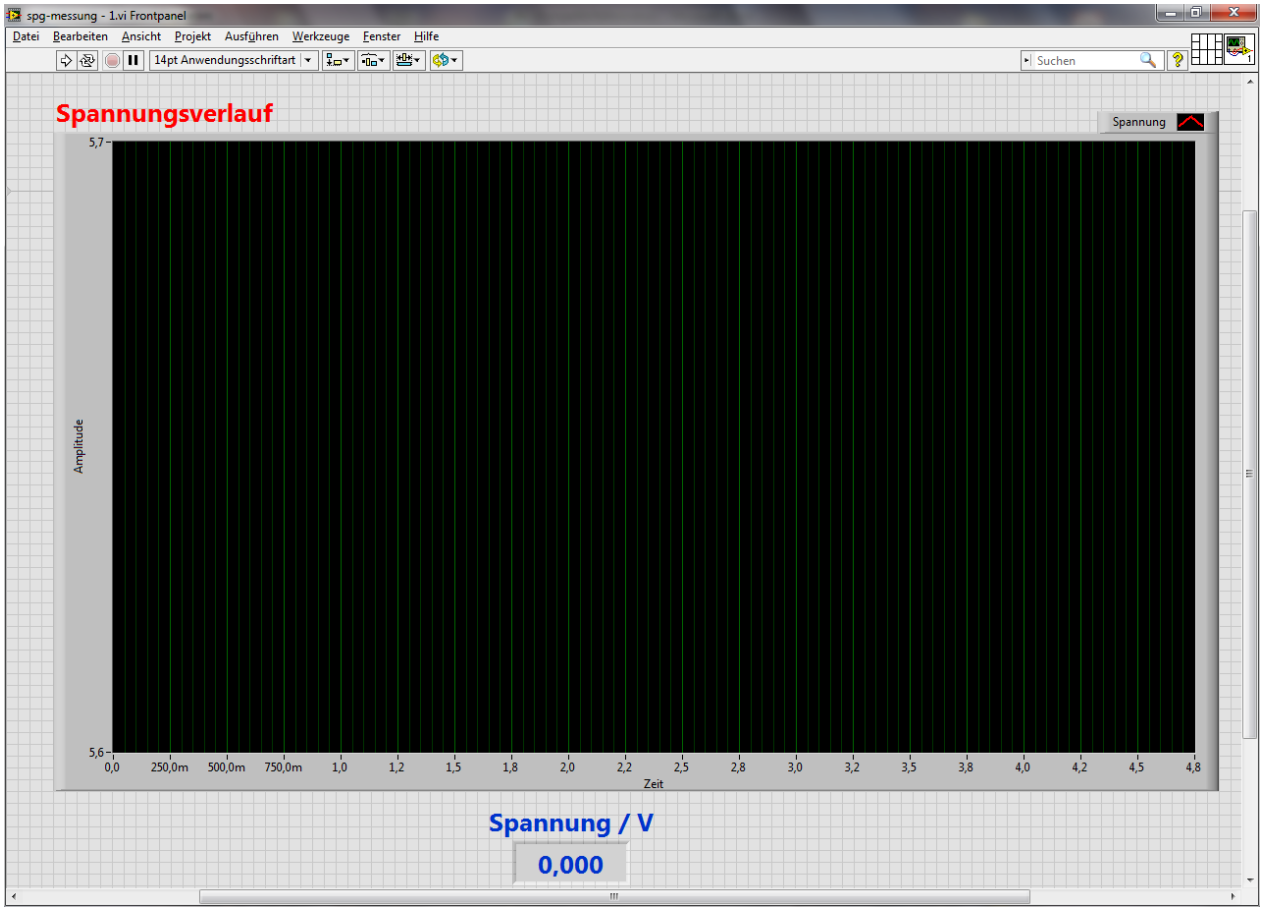


Abb.3.2.17: Das passende Frontpanel

Speichern Sie jetzt das VI ab, z.B. unter den Namen ´**spg-messung.vi - 1**´.

Nun endlich können Sie das VI auf dem Frontpanel starten, in dem Sie auf den Button ´**Wiederholt ausführen**´ klicken, **Abb.3.2.18**:

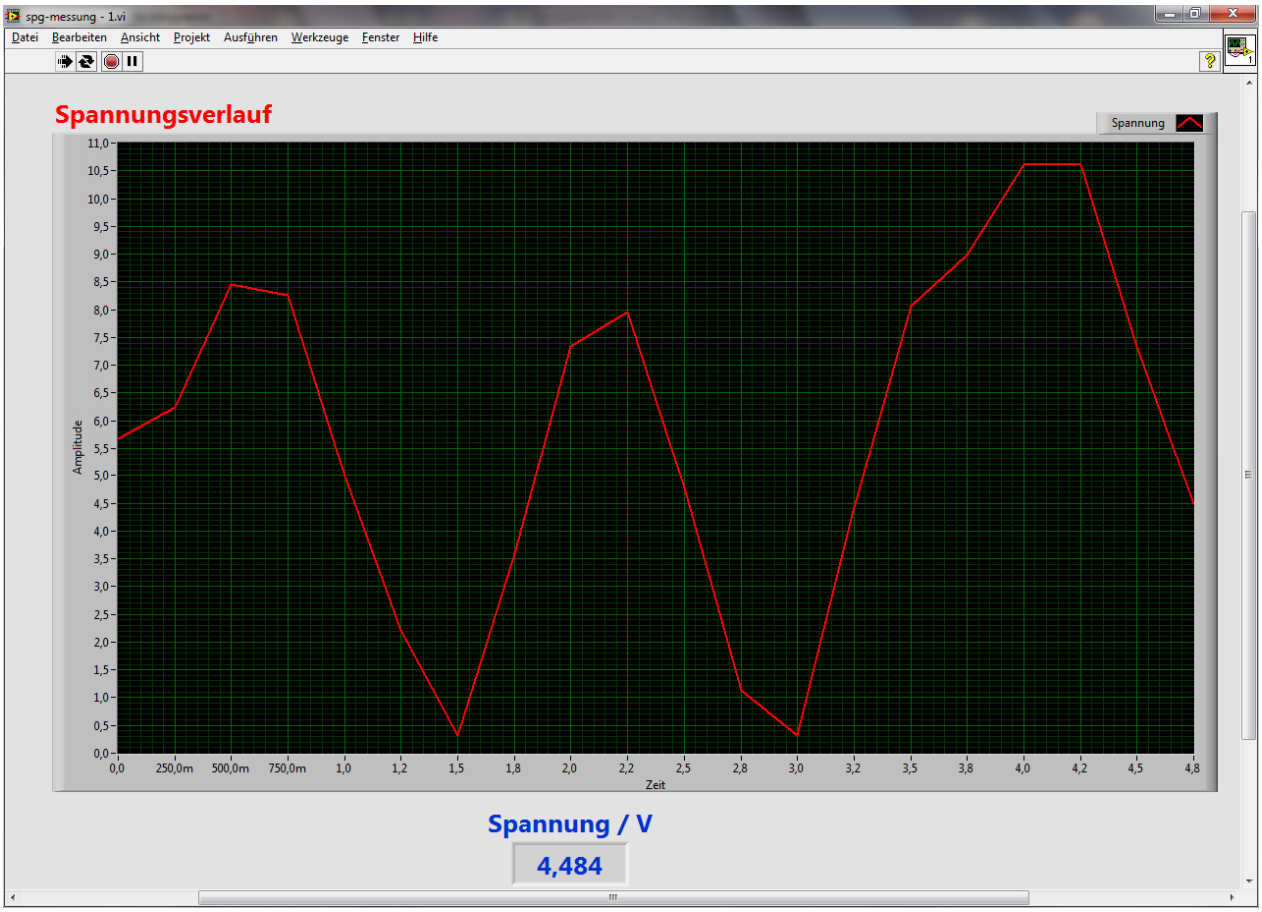


Abb.3.2.18: Die Spannungserfassung läuft …

Wir haben hier wieder das regelbare Netzteil an den Eingang ´ai0´ angeschlossen und die Spannung einfach rauf und runter geregelt: es werden nun bei jedem Aufruf des Express-VIs 5 Sekunden lang 20 Spannungsmesswerte erfasst und dies dann dargestellt, wobei die Zahlenanzeige immer den letzten Messwert anzeigt.

Dieses VI wird nun in unserem nachfolgend beschriebenen dritten Projekt (s. Kap. 6) zu einem vollwertigen, kleinen Oszilloskop-VI ausgebaut.

Nach diesen ausführlichen Beschreibungen zum DAQ-Assistenten und der Durchführung einiger einfacher Messungen kommen wir nun zu den eigentlichen Projekt-Applikationen …

4. Projekt „Ohmmeter“

In unserem ersten kleinen Projekt soll ein einfaches Ohmmeter zur Ausmessung von unbekannten Widerständen Rx mit dem USB-6008 realisiert werden.
Dazu zunächst ein kleiner Blick auf die elektrotechnischen Grundlagen, **Abb.4.1**:

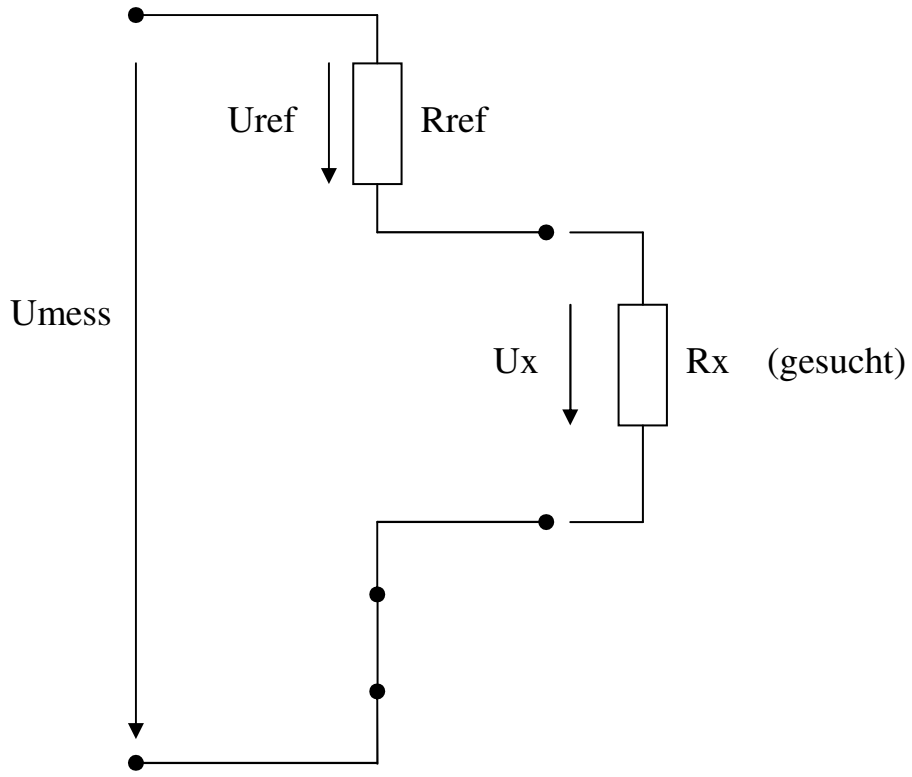


Abb.4.1: Die Grundlagen - Der Spannungsteiler

Eine Möglichkeit, unbekannte Widerstände zu bestimmten liegt in der Anwendung der bekannten „**Spannungsteiler-Regel**“:

Bei einem Spannungsteiler (Reihenschaltung von Widerständen) verhalten sich die (Teil)Spannungen wie die (Teil)Widerstände.

Und damit ergibt sich bei der hier vorliegenden Schaltung:

$$\frac{Rx}{Ux} = \frac{Rref}{Uref}$$

Aufgelöst nach Rx erhält man:

$$Rx = Ux * \frac{Rref}{Uref}$$

Und da weiterhin gilt: Uref = Umess - Ux ergibt sich als endgültige **Bestimmungsgleichung für Rx**:

$$Rx = Ux * \frac{Rref}{Umess - Ux}$$

Wenn also die Meßspannung Umess gegeben und bekannt ist, ebenso wie der Widerstand Rref ((hoch)genauer Messwiderstand) und die Spannung Ux am unbekannten Widerstand gemessen wird, so lässt sich Rx einfach bestimmen.
Und genau dieses Szenario wird hier mit dem USB-6008er-Modul nachgebildet, **Abb.4.2**:

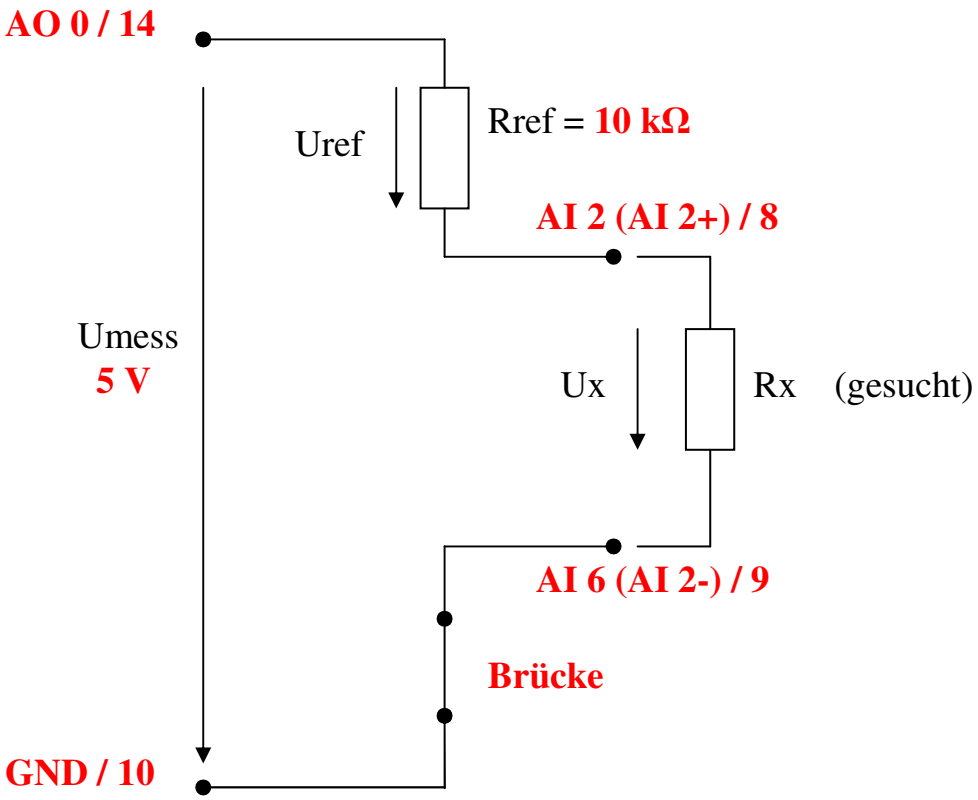


Abb.4.2: Die Verdrahtung des USB-6008ers

Die benötigte Meßspannung von +5V wird mit Hilfe des **Analog-Ausgangs AO 0** des USB-6008er erzeugt.
Als Referenzwiderstand Rref wird ein genauer Messwiderstand von zunächst 10 kΩ eingesetzt.

Hinweis:
Dieser Widerstand Rref erfüllt hier zwei wichtige Aufgaben:

1) Mit ihm wird der **Messbereich des Ohmmeters** festgelegt.
   Bei einem Wert von 10 kΩ ergibt sich ein sinnvoller Messbereich von ca. 100 Ω bis ca. 10 kΩ (s. auch nachfolgende Ausführungen).

2) Rref sorgt dafür, dass die **maximale Ausgangsbelastung** des Analogausgangs AO 0 von 5 mA nicht überschritten wird.
   Im Worst-Case-Fall (Rx = 0 Ω ≡ Kurzschluss an den Klemmen von Rx) ergibt sich somit ein maximaler Ausgangsstrom von 5 V / 10 kΩ = 500 µA.

Die Spannung Ux am unbekannten Widerstand Rx wird nun als Differenzspannung zwischen den beiden Eingangskanälen AI 2 und AI 6 gemessen (s. auch Kap. 2).
Eine Brücke zwischen AI 6 und GND vervollständigt die Hardwarebeschaltung des USB-6008.
Die **Abb.4.3** zeigt die verdrahtete Klemmleiste am Modul:

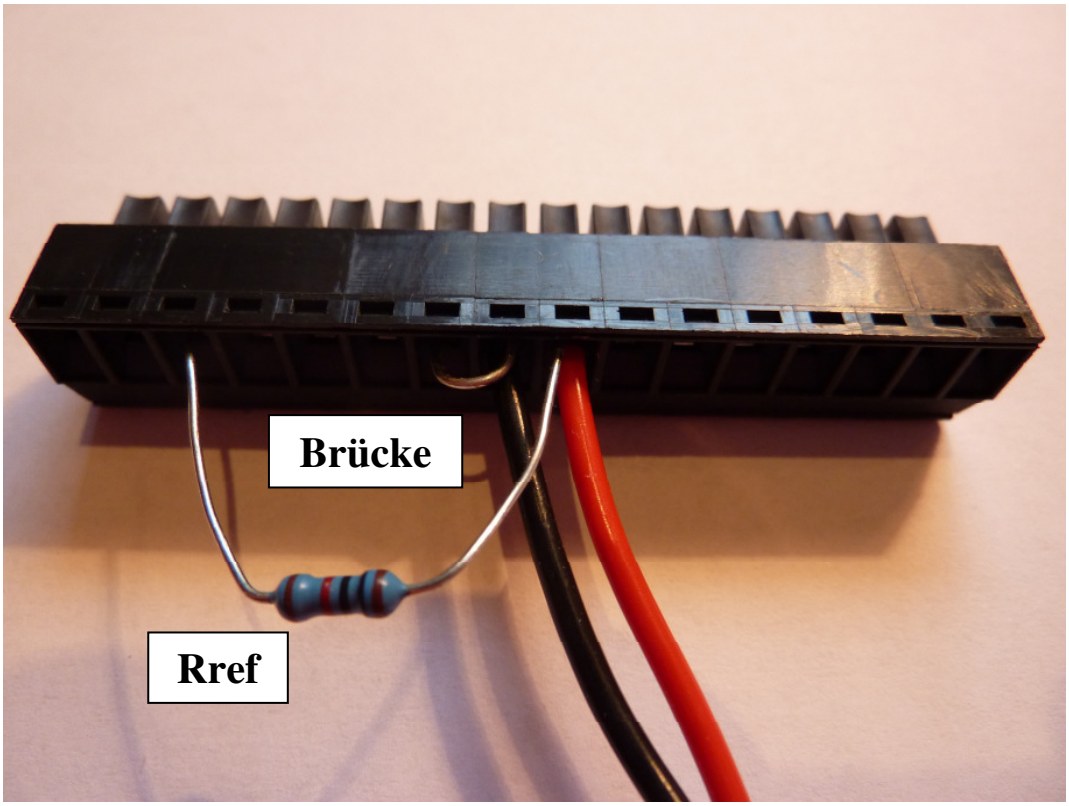


Abb.4.3: Die Verdrahtung der Klemmleiste
(die rote und die schwarze Leitung führen zu Rx)

Kommen wir nun zum Aufbau des LabVIEW-VIs.

Abb.4.4 zeigt das zu Grunde liegende Blockdiagramm des Ohmmeters:

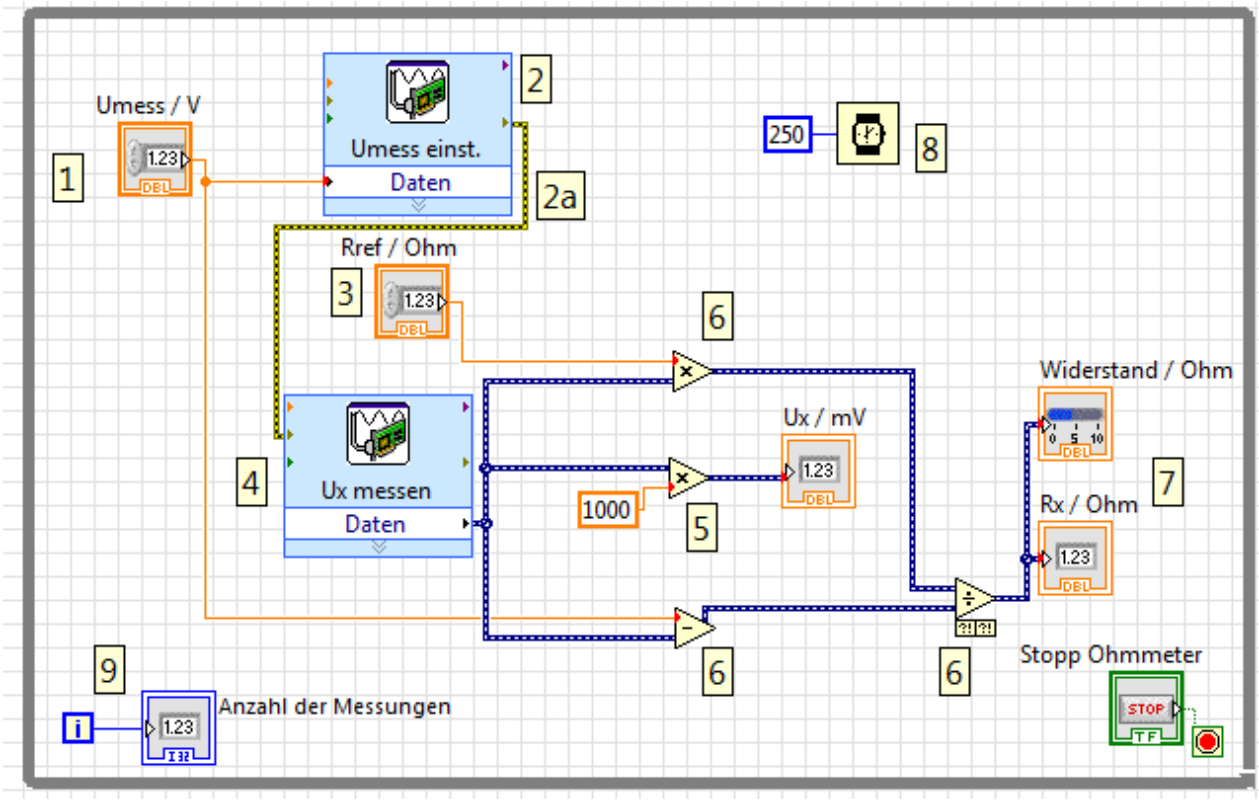


Abb.4.4: Das Blockdiagramm des Ohmmeters

Bei ❶ wird über das Frontpanel die jeweils einzustellende Meßspannung Umess vorgegeben und an den DAQ-Assistenten bei ❷ weitergeleitet.
Dieses Express-VI wird zur Ausgabe einer Spannung wie folgt konfiguriert:

- Auswahl: „**Signale erzeugen**“
- Dann „**Analoge Ausgabe**“ wählen
- Hier dann „**Spannung**“ wählen
- Als Ausgabekanal legen wir „**ao0**“ fest (≡ Analog-Out, Kanal 0)
- Nach dem Klick auf „**Beenden**“ erscheint das eigentliche **Konfigurationsfenster** für diesen analogen Spannungsausgang, **Abb.4.5**:

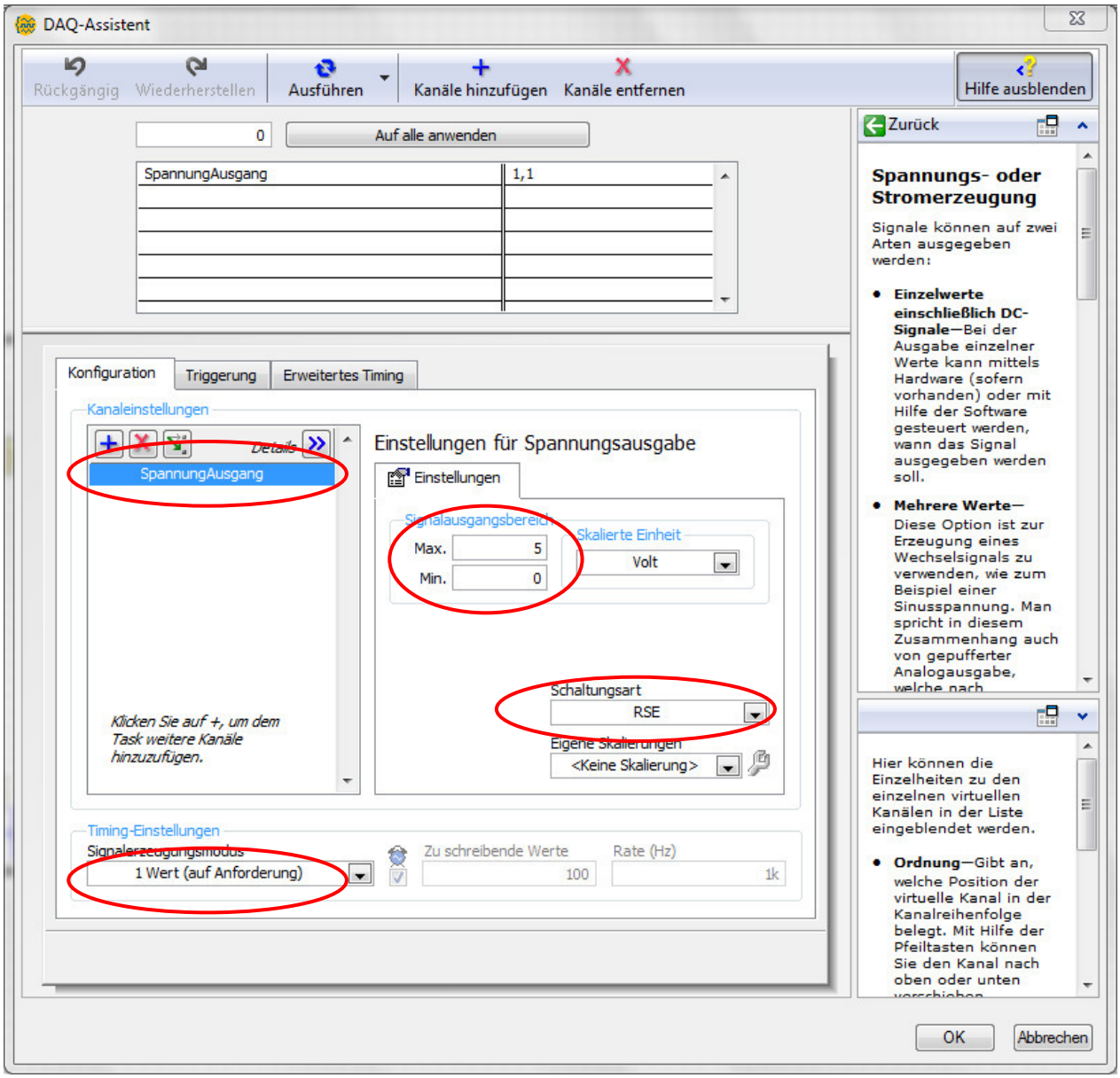


Abb.4.5: Die Konfiguration des analogen Spannungsausgangs ao0

Notwendige Einstellungen:

- **SpannungAusgang**: wurde schon eingestellt.

- **Signalausgangsbereich**: hier wählen wir den maximalen Bereich von 0 … +5 V. Der reale Ausgangsspannungswert kann später, zur Laufzeit des VIs, jederzeit über das Frontpanel-Bedienelement ❶ wahlfrei eingestellt bzw. festgelegt werden. Wir stellen in den meisten Fällen über das Frontpanel sinnvoller Weise einen Wert von Umess = +5 V ein.

Sehr wichtig:

Hier wird also nur der möglich Ausgangsspannungs**bereich** eingestellt, der vom Express-VI auch überwacht wird.

Die eigentlich gewünschte Ausgangsspannung MUSS IMMER von außen an das Express-VI angelegt werden ! (z.B. über eine Konstante, über ein Bedienelement oder durch eine Berechnung innerhalb des VIs selber)

Allerdings wird die so eingestellte Ausgangsspannung überwacht, z.B.:
Stellt man einen Ausgangsspannungsbereich von 0 .. 3 V ein und schließt von außen (z.B. via Bedienelement ❶) eine Spannung von 2,5 V an, so ist alles ok.
Schließt man dagegen eine Spannung von 3,5 V an, so erkennt das Express-VI diese Verletzung der oberen Grenze, gibt eine Fehlermeldung aus und hält das VI an.

- **Schaltungsart**: RSE (ein andere lässt sich auch gar nicht einstellen).

- **Signalerzeugungsmodus**: 1 Wert (auf Anforderung), d.h. bei jedem Aufruf des VIs, also bei bzw. vor jeder Messung wird die Ausgangsspannung Umess neu eingestellt.

Zur Beendigung der Einstellungen klicken Sie auf ´OK´.
Damit ist die Konfiguration des Analogausgangs zur Erzeugung von Umess abgeschlossen.

Das Express-VI ❹ wird nun so eingestellt, dass es die unbekannte Differenzspannung Ux zwischen den analogen Eingangskanälen AI 2 und AI 6 misst (s. Abb.4.2), **Abb.4.6**:

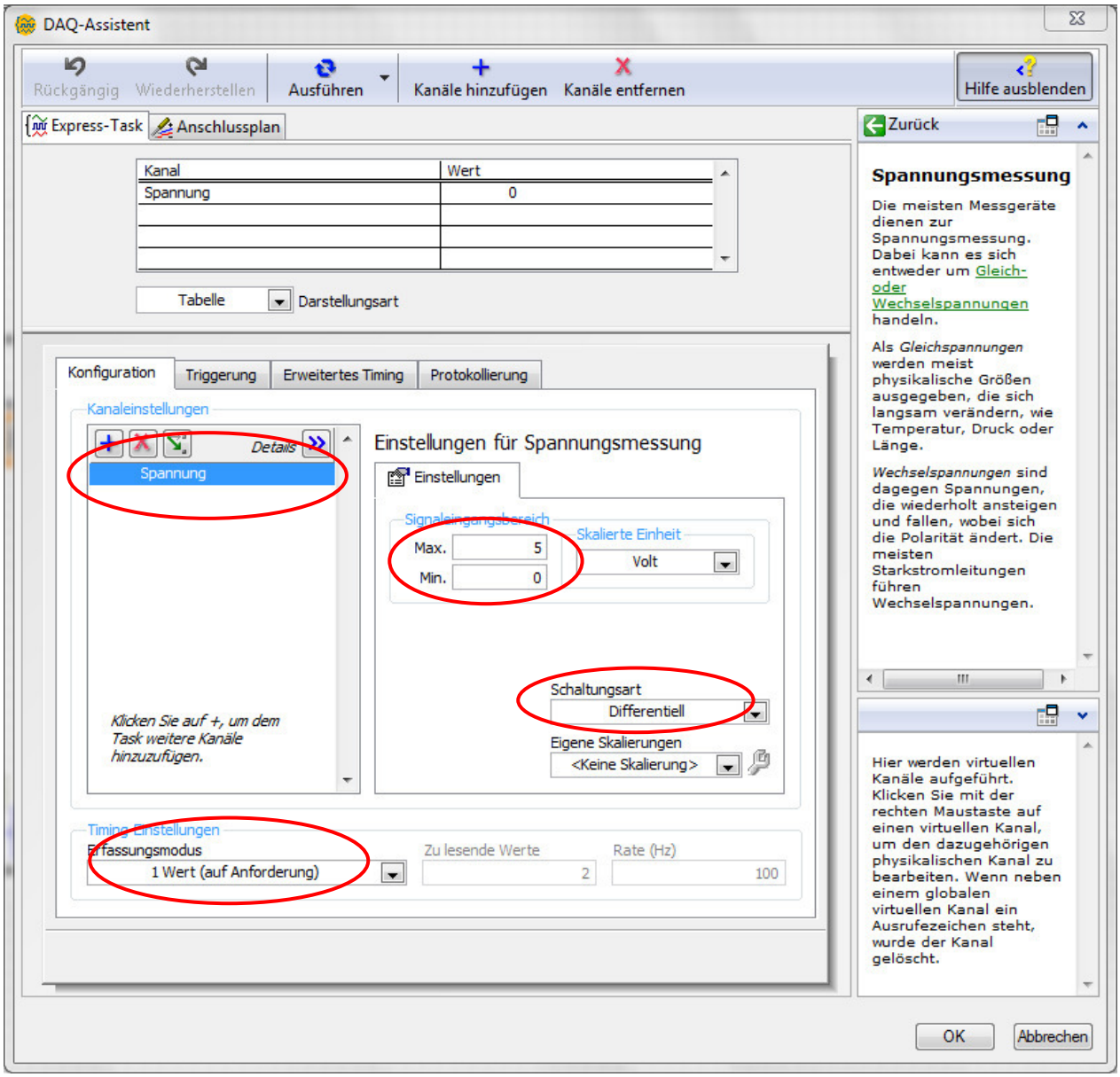


Abb.4.6: Die Messung der unbekannten Eingangsspannung Ux

Einstellungen:

- **Spannung(smessung)**: wurde bereits festgelegt.
- **Signaleingangsbereich**: 0 … 5 V, kleinere/größere Spannungen können ja gar nicht auftreten.
- **Schaltungsart**: Differentiell
- **Erfassungsmodus**: 1 Wert (auf Anforderung)

Zur Beendigung der Einstellungen klicken Sie auf ´OK´.
Damit ist die Konfiguration des Analogeingangs zur Messung von Ux abgeschlossen.
Am Datenausgang dieses Express-VIs kann daher Ux abgegriffen und weiter verarbeitet werden.

Sehr wichtig ist die Verbindung ❷a zwischen dem Fehlerausgang des VIs ❷ und dem Fehlereingang des VIs ❹ (obwohl diese Verbindung hier scheinbar überflüssig ist, da auftretende Fehler gar nicht angezeigt bzw. gar nicht weiter ausgewertet werden).
Fehlt diese Verbindung, dann passiert beim Start des Gesamt-VIs folgendes: die Express-VIs ❷ und ❹ sind Datenfluss-mäßig voneinander unabhängig und das bedeutet, dass LabVIEW beide Express-VIs **GLEICHZEITIG** startet.
Mit anderen Worten: die Meßspannung Umess wird eingestellt und gleichzeitig wird schon Ux gemessen.
Und das kann zu Problemen führen, wenn Umess noch nicht (ganz) stabil am Spannungsteiler anliegt und Ux somit falsch (ebenfalls noch nicht stabil) gemessen wird.
(Man kann hier leider nicht eindeutig sagen, dass das Express-VI ❷ vor Start des Express-VIs ❹ fertig abgearbeitet ist, da im Hintergrund noch das Betriebssystem Windows agiert. Und dessen Verhalten, in Verbindung mit der Kommunikation über die USB-Schnittstelle zum USB-6008-Modul hin, ist eben nicht deterministisch).

Dieses Verhalten muss also auf jeden Fall verhindert werden.
Es muss vielmehr gelten: zuerst muss Umess stabil anliegen und dann erst darf Ux gemessen werden.
Man muss daher an dieser Stelle von der parallelen Verarbeitung unabhängiger Threads unter LabVIEW abweichen (Datenfluss-Prinzip) und LabVIEW zu einer sequentiellen Abarbeitung der beiden Threads (der beiden Express-VIs) „zwingen."
Und genau das macht die Verbindung ❷a: Das Express-VI ❹ wartet jetzt nämlich mit seiner Ausführung solange, bis an seinem Fehlereingang gültige Daten vom Express-VI ❷ anliegen.
Das bedeutet: es wird zunächst das Express-VI ❷ gestartet und abgearbeitet. Wenn das fehlerfrei erfolgt ist (und davon gehen wir hier einmal aus), wird die Fehlerausgangsmeldung von ❷ an den Fehlereingang des Express-VIs ❹ weitergeleitet und erst jetzt wird dieses Express-VI gestartet und somit Ux gültig und stabil gemessen.
Obwohl also die Fehlermeldungen der beiden Express-VIs gar nicht ausgewertet werden, sorgt die Verbindung ❷a für einen ordnungsgemäßen Ablauf des Gesamt-VIs.

Der Rest unseres Ohmmeter-Blockdiagramms ist nun schnell erklärt:
Bei ❺ wird die gemessene Spannung Ux so aufbereitet, dass sie in mV angezeigt werden kann.

Die ❻er-Blöcke sorgen für die Realisierung der Bestimmungsgleichung für Rx, wobei über das Frontpanel-Bedienelement ❸ der jeweils verwendetet Referenzwiderstand Rref eingegeben werden kann (in unserem Beispiel: Rref = 10 kΩ).
Bei ❼ erfolgt die Anzeige von Rx auf zwei Arten, ❽ erzeugt eine Wartezeit von 250 ms zwischen den einzelnen Messungen und bei ❾ wird einfach die Anzahl der bisher durchgeführten Messungen gezählt.

Als letzter Punkt bleibt nur noch die „schöne“ Gestaltung des Frontpanels übrig, das z.B. wie folgt aussehen könnte, **Abb.4.7**:

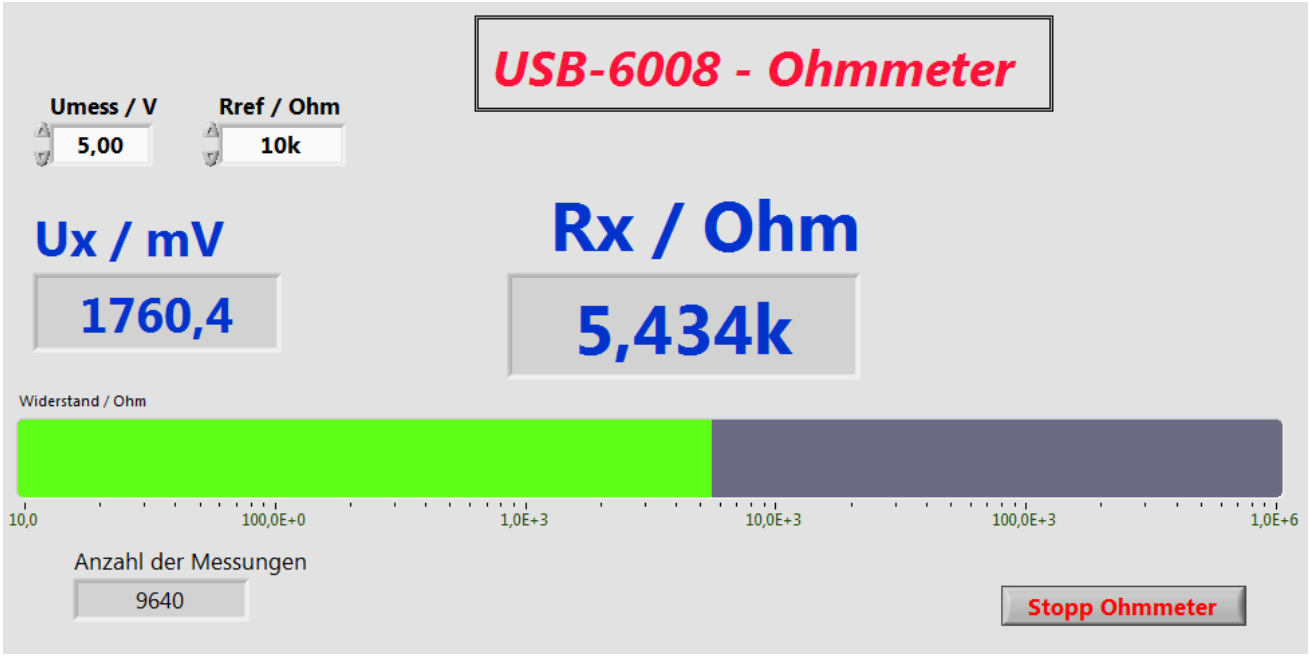


Abb.4.7: Das Frontpanel des Ohmmeters

Praxiseinsatz:

Ab jetzt können Sie das USB-6008er-Modul bequem als Ohmmeter einsetzen.

5. Projekt „Thermometer“

Das vorhergehende Projekt lässt sich nun sehr einfach zu einem elektronischen **Thermometer** mit dem USB-6008er-Modul erweitern.

Die Grundlage dazu bildet ein temperaturabhängiger Widerstand, ein **KTY81-110** der Firma NXP.
Dieses Bauteil ist ein **Silizium-Halbleiter-Temperatursensor (PTC-Verhalten)**, der seinen Gesamtwiderstandswert in Abhängigkeit von der Umgebungstemperatur ändert.
Mit anderen Worten: die Temperaturmessung wird hier zunächst auf die Messung eines ohmschen Widerstandes zurückgeführt, und genau das haben wir gerade ja gemacht.
Bevor wir starten, zunächst aber die unerlässlichen …

Grundlagen

Aus dem Datenblatt des KTY81-110er lässt sich der Widerstandswert (tabellarisch) in Abhängigkeit von der jeweiligen äußeren Umgebungstemperatur ermitteln.
Trägt man diese Tempertaturwerte in Abhängigkeit vom gemessenen Widerstand R_x graphisch auf, also den Funktionsgraphen $\mathbf{T = f(R_x)}$, so erkennet man, dass dieser Verlauf (leider) mehr oder weniger stark nichtlinear ist, **Abb.5.1**:

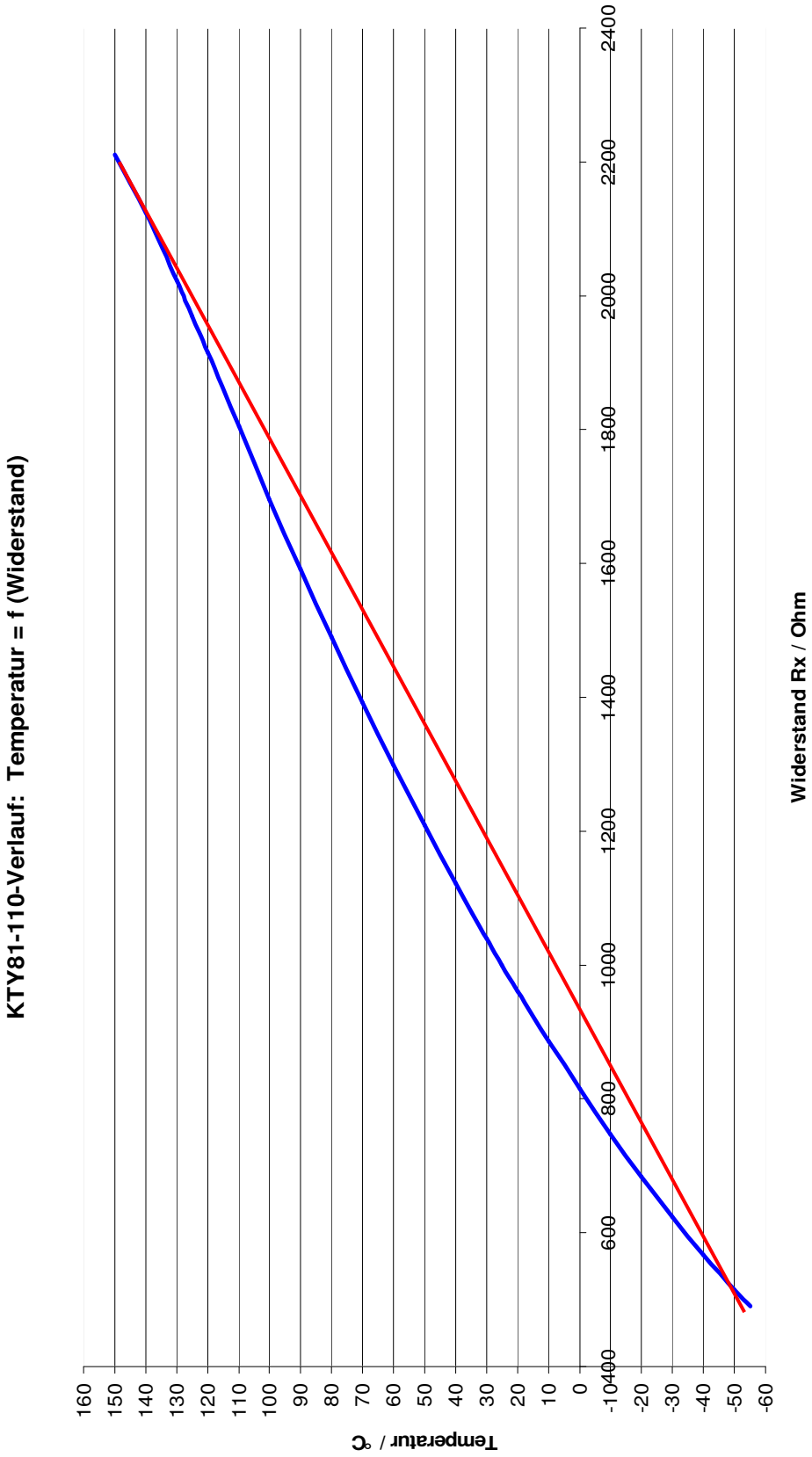


Abb.5.1: Der Widerstands-/Temperaturverlauf des KTY81-110 - $T = f(R_x)$

Die blaue Kurve entspricht dem Widerstandsverlauf des KTY81-110ers und die rote Kurve zeigt zum Vergleich einen angenommenen linearen Temperatur-/Widerstands-Verlauf.

Diese „Krümmung“ im KTY81-110er-Verlauf hat nun zur Folge, dass man aus dem gemessenen Widerstand nicht so einfach, z.B. über eine lineare Gleichung, die zu Grunde liegende Temperatur ermitteln kann.
Will man genaue Messungen/Berechnungen durchführen, muss man mit linearen Interpolationen arbeiten.
Das hört sich zunächst sehr stark mathematisch an, aber auch hier bietet LabVIEW natürlich bereits fertige Funktionen, die dieses einfach für den Anwender erledigen.
Somit muss unser zu entwickelndes Thermometer-VI **grundsätzlich zwei Aufgaben erfüllen:**

- Bestimmung des Widerstandes des KTY81-110ers (so etwas wurde bereits im vorherigen Projekt durchgeführt)

  und

- Berechnung der Temperatur durch Interpolation des gemessenen Widerstandswertes.

Die Beschaltung des USB-6008er-Moduls

zeigt die **Abb.5.2**:

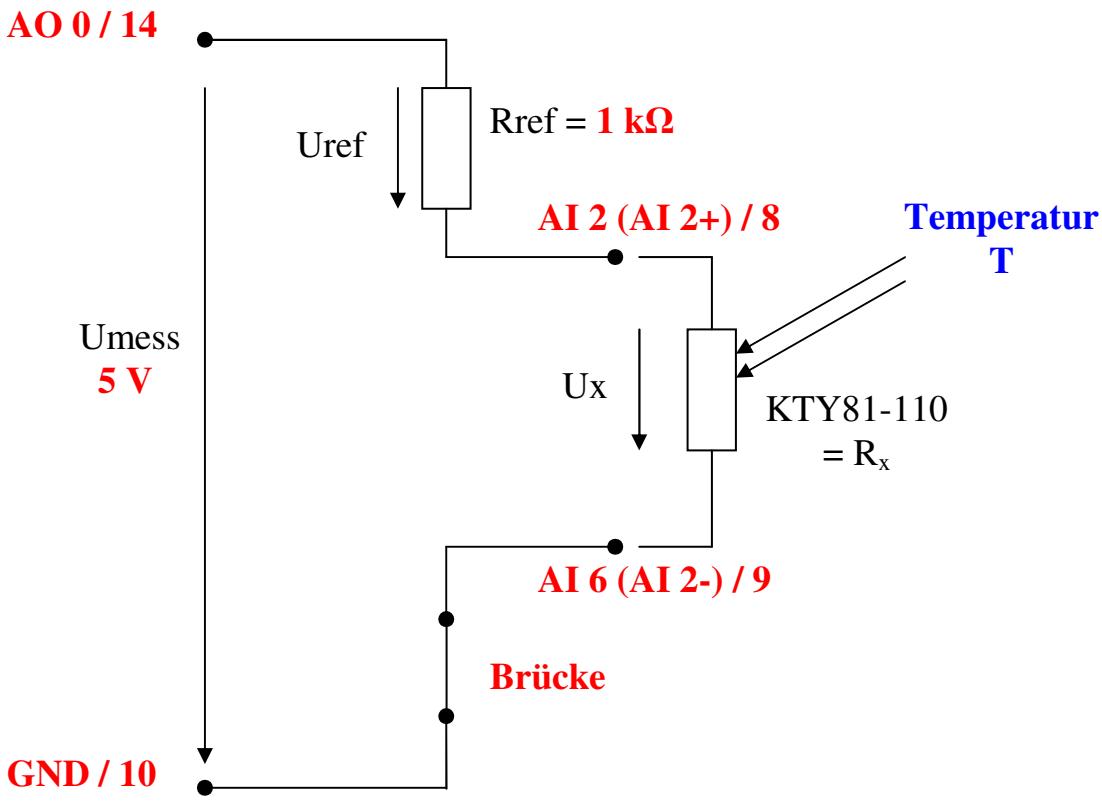


Abb.5.2: Die Beschaltung des USBer-6008-Moduls

Die äußere Beschaltung ist identisch mit der Schaltung zur Widerstandsmessung aus der Abb.4.2 mit dem einzigen Unterschied, dass R_{ref} hier einen Wert von 1 kΩ hat und wir mit einer festen (unverstellbaren) Meßspannung U_{mess} von +5 V arbeiten.
Abb.5.3 zeigt die Verdrahtung der Steckleiste am USB-6008er:

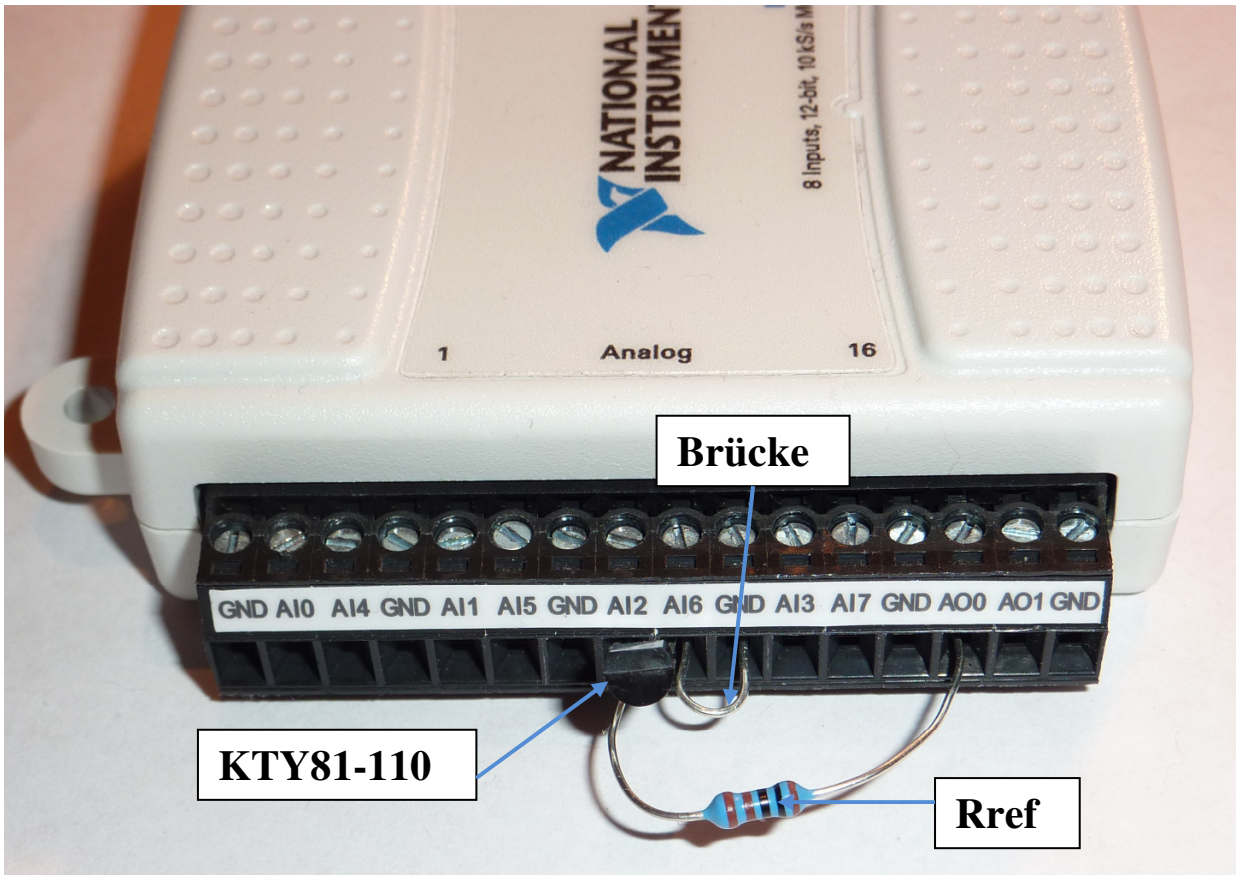


Abb.5.3: Der Verdrahtung am USB-6008er

Damit erhält man als Bestimmungsgleichung für den aktuellen KTY81-110er-Widerstandswert (≡ R_x):

$$Rx = Ux * \frac{1\,k\Omega}{5\,V - Ux}$$

Den Messbereich für Ux kann man hier nun etwas genauer einstellen, denn es gilt ja zunächst gem. der Spannungsteiler-Regel:

$$Ux = Umess * \frac{Rx}{Rx + Rref}$$

Hier also:

500

$$Ux = 5\,V * \frac{Rx}{Rx + 1\,k\Omega}$$

Der minimal mögliche Wert $R_{x,min}$ des KTY81-110ers bei einer Temperatur von -55°C ist nun 475 Ω.
Damit ergibt sich für

Ux,min = 1,61 V

Der maximal mögliche Wert $R_{x,max}$ des KTY81-110ers bei einer Temperatur von +150°C ist nun 2277 Ω.
Damit ergibt sich für

Ux,max = 3,47 V

Im Express-VI zur Messung von Ux kann daher ein Messbereich von 1,5 V … 3,5 V eingestellt werden, um die Spannung Ux mit größt möglicher Auflösung zu messen.

Das Blockdiagramm des VIs

Nach diesem kleinen theoretischen Ausflug kann das Blockdiagramm des Thermometers entwickelt werden, **Abb.5.4**:

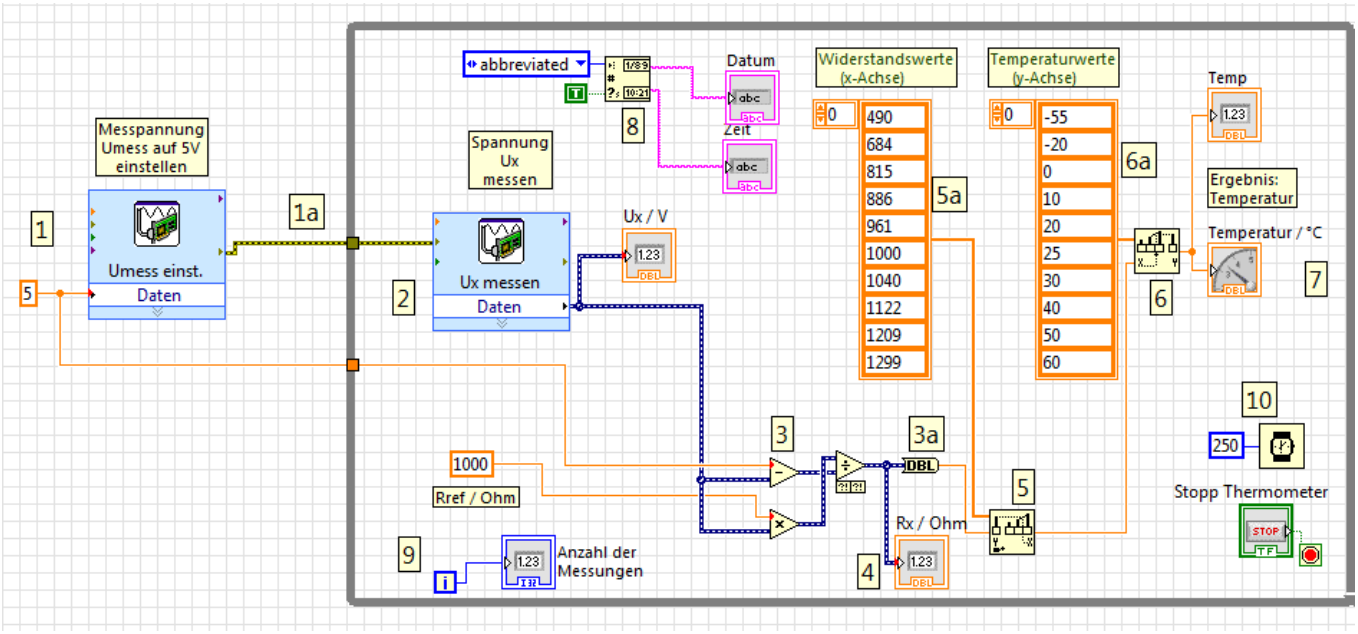


Abb.5.4: Das Blockdiagramm des Thermometers

Bei ❶ wird die benötigte Meßspannung von +5 V eingestellt.
Dazu ist dieses Express-VI wie folgt zu konfigurieren:

- Spannungsausgang: ao0
- Signalausgangsbereich: 0 V … 5 V
- Schaltungsart: RSE
- Signalerzeugungsmodus: 1 Wert (auf Anfrage), d.h. Umess wird einmal zu Beginn der Messungen eingestellt und das ist ja ausreichend.

Die eigentlich benötigte Meßspannung wird danach außerhalb des Express-VIs über eine Konstante mit dem Wert ´5´ eingestellt und zusätzlich in die While-Schleife zu den Berechnungsstufen reingeführt.

Über die Verbindung ❶a wird der korrekte Datenfluss sichergestellt, d.h. zuerst wird die Meßspannung eingestellt, dann können die Messungen beginnen.

Innerhalb der While-Schleife wird bei ❷ die Spannung Ux am KTY81-110er gemessen.
Dieses Express-VI ist somit wie folgt zu parametrieren:

- Spannungsmessung über: ai2
- Signaleingangsbereich: 1,5 V … 3,5 V
- Schaltungsart: Differentiell
- Erfassungsmodus: 1 Wert (auf Anforderung)

Die so erfasste Spannung Ux wird bei ❸ in den Widerstandswert des KTY81-110ers umgerechnet und bei ❹ als Zwischenwert angezeigt.
Bis hier hin ist der Ablauf noch identisch wie beim Ohmmeter.

Bei ❸a erfolgt nun die Separation des eigentlichen Widerstandswerts aus dem dynamischen Datensatz:
Das Express-VI ❷ gibt als Ausgangsdaten nicht einfach nur die reine Meßspannung Ux sondern einen kompletten Datensatz aus, in dem noch weitere Werte, wie z.B. Zeit- und Datumsinformationen über den Zeitpunkt der Messung enthalten sind.
(Dynamische Daten werden im Blockdiagramm durch eine blau/weiß-geringelte Leitung gekennzeichnet)
Dementsprechend werden bei allen folgenden Berechnungen auch dynamische Daten weiter verarbeitet.
Insbesondere liegt am Ausgang der Divisionsfunktion solch ein zusammengesetzter Datensatz vor, in dem der gewünschte Wert von Rx enthalten ist, aber auch noch die anderen Daten.

Diese Zusatzdaten sind für uns hier nicht relevant, wir benötigen für die weiteren Berechnungen lediglich den reinen Widerstandswert Rx.
Diesen erhält man, indem man einfach eine entsprechende Konvertierungsfunktion verwendet, die an ihrem Ausgang den Wert Rx separat zur Verfügung stellt.
Man findet diese Funktion unter:

BD\Programmierung\Numerisch\Konvertierung\Nach Fließkommawert

In den ❺er- und ❻er-Funktionsblöcken erfolgt nun die lineare Interpolation:

Die Funktion ❺ findet man unter:

BD\Programmierung\Array\Schwellwert (1D-Array)

Mit dieser Funktion wird zunächst festgestellt, in welchen Intervall der Widerstandswert Rx liegt.

Dazu muss man bei ❺a (und später bei ❻a) entsprechende Stützwerte aus dem Datenblatt des KTY81-110er angeben (Anschluss der Arrays über: Kontextmenü zum jeweiligen Anschluss-Pin und dann ´Erstellen\Konstante´).
Mit anderen Worten:

In den beiden 1-dimensionalen Arrays ❺a und ❻a werden Widerstands-/Temperatur-Wertepaare (die (typischen) TYP-Werte) aus dem Datenblatt eingetragen, z.B.:

490 Ω bei -55°C
684 Ω bei -20 °C
815 Ω bei 0°C
usw.

Je mehr Wertepaare man hier einträgt, desto genauer wird die Interpolation.
Wir haben uns hier erst einmal auf 10 Wertepaare beschränkt, Erweiterungen können Sie selber aber sehr einfach durchführen.

Nachdem durch ❺ nun der interpolierte x-Wert bestimmt worden ist, wird dieser bei ❻ eingespeist.
Man findet diese Funktion unter:

BD\Programmierung\Array\1D-Array interpolieren

Hiermit wird nun die Interpolation des gesuchten y-Wertes, also der **gesuchten Temperatur**, durchgeführt.

Der so gefundene Temperaturwert steht am Ausgang von ❻ zur Verfügung und wird bei ❼ auf zwei verschiedene Arten angezeigt.

Durch den Block bei ❽ erhält man noch zusätzlich eine Uhrzeit und Datumsanzeige. Man findet diese Funktion unter:

BD\Programmierung\Timing\Datum-/Zeit-String lesen

Bei ❾ wird ganz einfach die Anzahl der bisher durchgeführten Messungen gezählt und ❿ sorgt für eine Zeitverzögerung von 250 ms zwischen den einzelnen Messungen.

Das Frontpanel des VIs

Nachdem man nun so das funktionsfähige Blockdiagramm erstellt hat, schaltet man um auf das Frontpanel und gestaltet dieses nach seinen Wünschen und Vorstellungen.
Eine Möglichkeit der Realisierung könnte so aussehen, **Abb.5.5**:

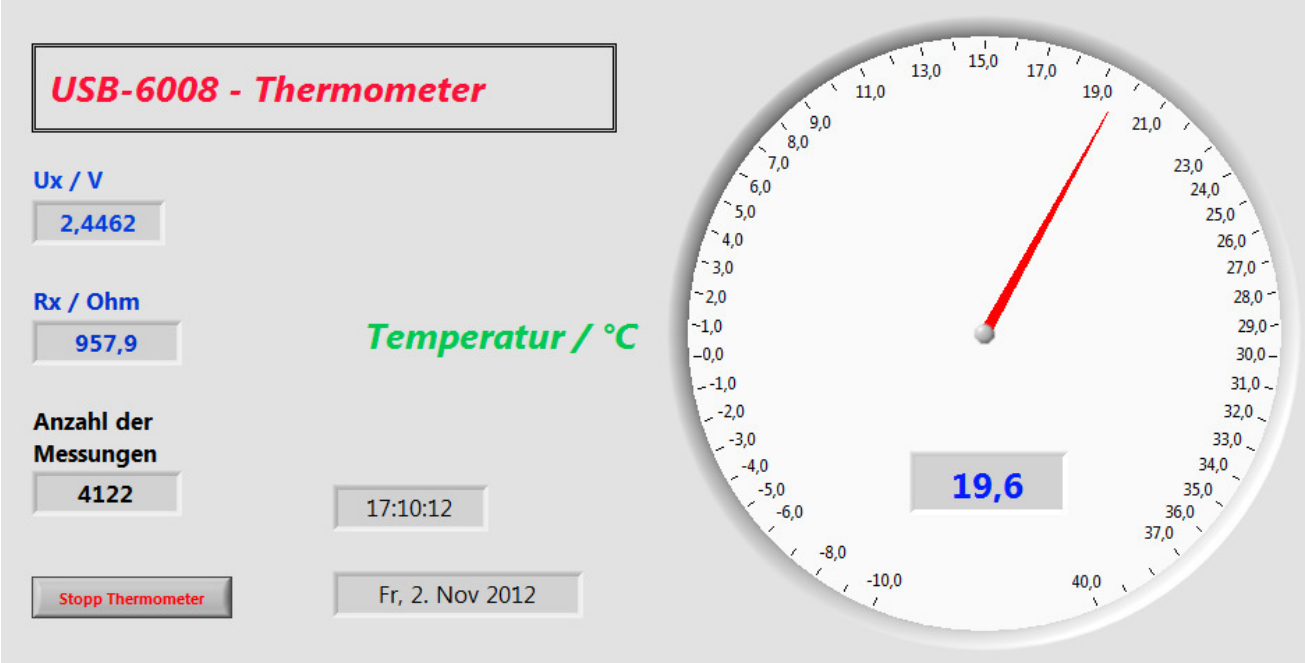


Abb.5.5: Der Frontpanel des Thermometers

Jetzt können Sie das VI starten und Ihre Messungen durchführen.

Natürlich sollten Sie die Ergebnisse mit einem Referenzmessgerät vergleichen.

6. Projekt „Oszilloskop“

In unserem vorerst letzten kleinen Praxisprojekt mit dem USB-6008er wollen wir ein einfaches Oszilloskop entwickeln.
Schaut man sich das erstellte Express-VI zur Spannungsmessung mit Hilfe der Kontexthilfe etwas näher an, so erkennt man, dass einige der Konfigurationsdaten, die wir bisher „von Hand“ intern eingestellt und festgelegt haben (s. z.B. Abb.3.2.7) auch von extern, d.h. über das LabVIEW-Programm selber, während der Programmablaufzeit („programmatisch“) eingestellt werden können, **Abb.6.1**:

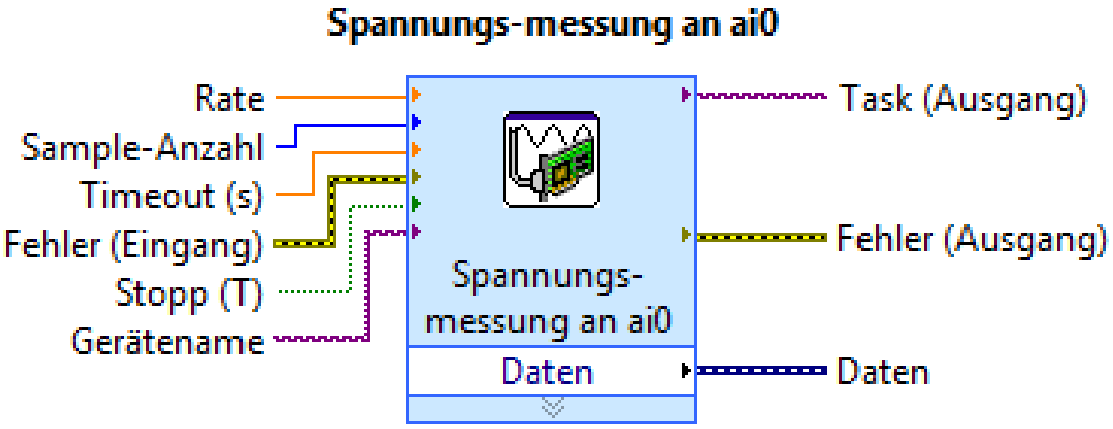


Abb.6.1: Aus der Kontexthilfe zum DAQ-Assistenten-Express-VI: die Ein- und Ausgänge bei der Spannungsmessung im Detail

Unter anderem können die Parameter ´**(Sample)Rate**´, ´**Sample-Anzahl**´ und ´**Timeout(s)**´ so ganz einfach von außen festgelegt werden.
Das machen wir uns nun zunutze, um die Betriebswerte des Oszilloskops dynamisch, während des Betriebs, einzustellen.

Das Frontpanel des fertigen Oszilloskops ist in **Abb.6.2** dargestellt und gibt schon einen sehr guten Überblick über die (Mess)Möglichkeiten dieses VIs:

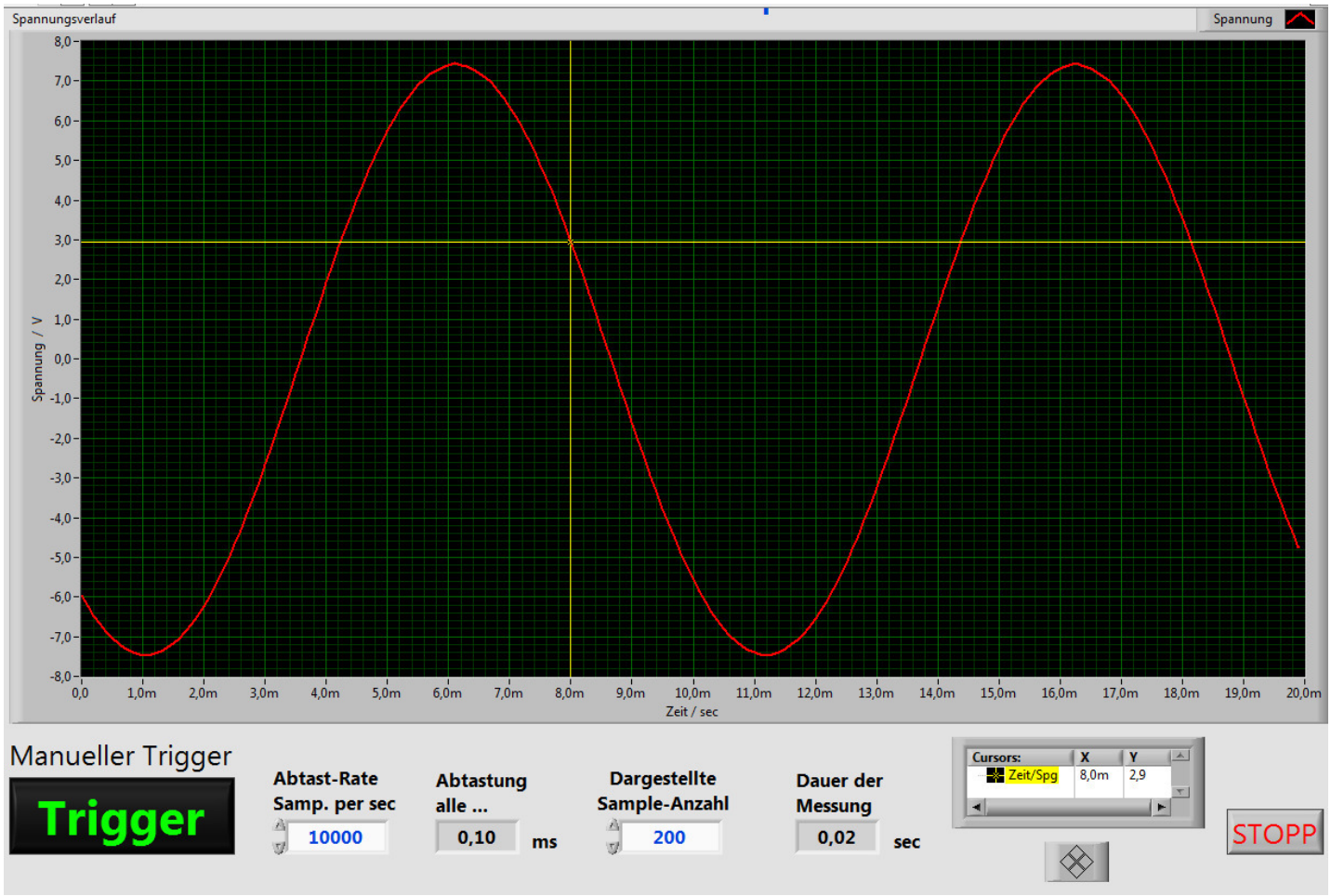


Abb.6.2: Das Frontpanel des Oszilloskops

Das komplette Blockdiagramm zum Oszilloskop zeigt **Abb.6.3**. und ist schnell erklärt:

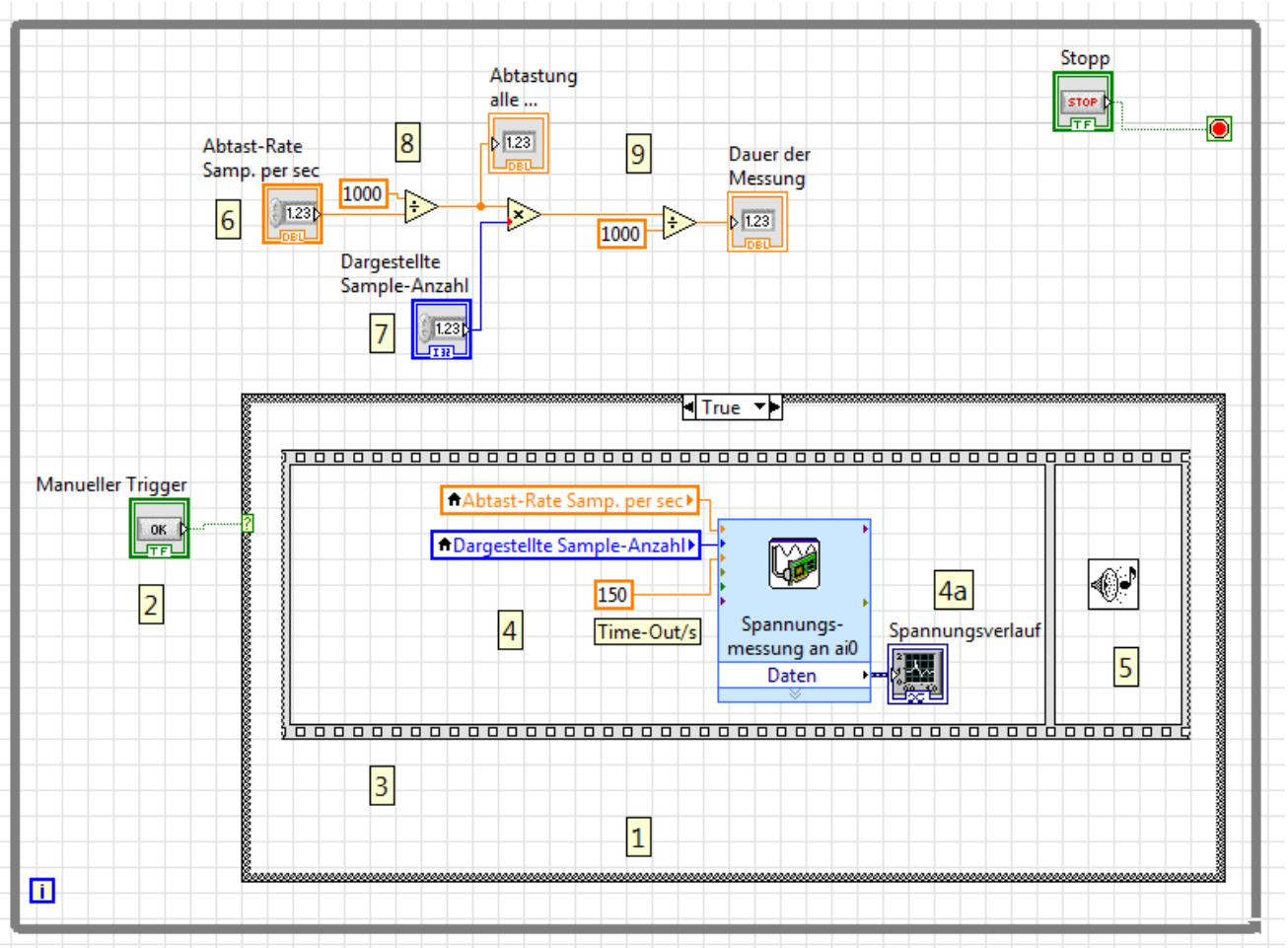


Abb.6.3: Das Blockdiagramm des Oszilloskops

Kernpunkt des VIs ist zunächst eine Case-Struktur, die als einfache if-Struktur betrieben wird, ❶.

Das ´**True-Unterdiagramm**´ realisiert die Oszilloskop-Funktion (≡ Messung und Darstellung des Eingangssignals) und das ´**False-Unterdiagramm**´ bleibt ganz einfach leer.

Der Start des Oszilloskops, genauer: der Start einer Messung mit anschließender Kurvendarstellung der Messwerte, wird mit Hilfe des Tasters ´**Manueller Trigger**´, ❷, ausgelöst: beim Betätigen und wieder Loslassen des Tasters wird das True-Unterdiagramm einmalig ausgeführt --> Messung und Darstellung.

Wird der Taster dagegen nicht betätigt, wird permanent das False-Unterdiagramm, also ´Nichts´, abgearbeitet.

Hinweis:

Aufgrund der „Einfachheit“ des USB-6008er-Moduls und des hier entwickelten VIs ist eine permanente Darstellung des Eingangssignals, wie man sie von einem „normalen Oszilloskop“ her kennt, nicht realisierbar.

Es fehlt einfach die Möglichkeit, eine geeignete Triggerung zu implementieren, um ein dauerhaft stehendes Bild zu erhalten. Daher wird hier, jeweils einfach auf „Knopfdruck“, eine Messung und eine Darstellung der Messergebnisse durchgeführt.
Der Leistungsfähigkeit dieses Oszilloskops tut das aber keinen Abbruch, denn die Ergebnisse sind sehr gut darstellbar, insbesondere wenn es sich um periodisch verlaufende Meßsignale handelt (was ja sehr oft der Fall ist).

Innerhalb des True-Unterdiagramms befindet sich nun eine Flache Sequenzstruktur, ❸, mit zwei Rahmen.
Im Rahmen ❹ ist der eigentliche Mess- und Darstellungsteil enthalten. Der Rahmen ❺ beinhaltet eine kleine akustische Spielerei: wenn der gestartete Messdurchgang incl. der Darstellung der Messwerte beendet ist (wenn also ein komplettes Oszillogramm vorliegt), ertönt ein kleiner ´Piep-Ton´, der diesen Fertig-Zustand signalisiert.
Das ist besonders bei der Durchführung längerer Messungen/Aufzeichnungen (> 5 Sekunden Dauer) sehr sinnvoll.

Das Unterdiagramm ❹ besteht nun im Wesentlichen aus dem bereits hinlänglich bekannten DAQ-Assistenten-Express-VI zur Spannungsmessung, allerdings hier mit dem kleinen Unterschied, dass einige Betriebsparameter jetzt von außen her eingestellt werden.
Zuerst wird aber, wie gewohnt, der Assistent aufgerufen und intern die Grundwerte dieses VIs eingestellt, **Abb.6.4**:

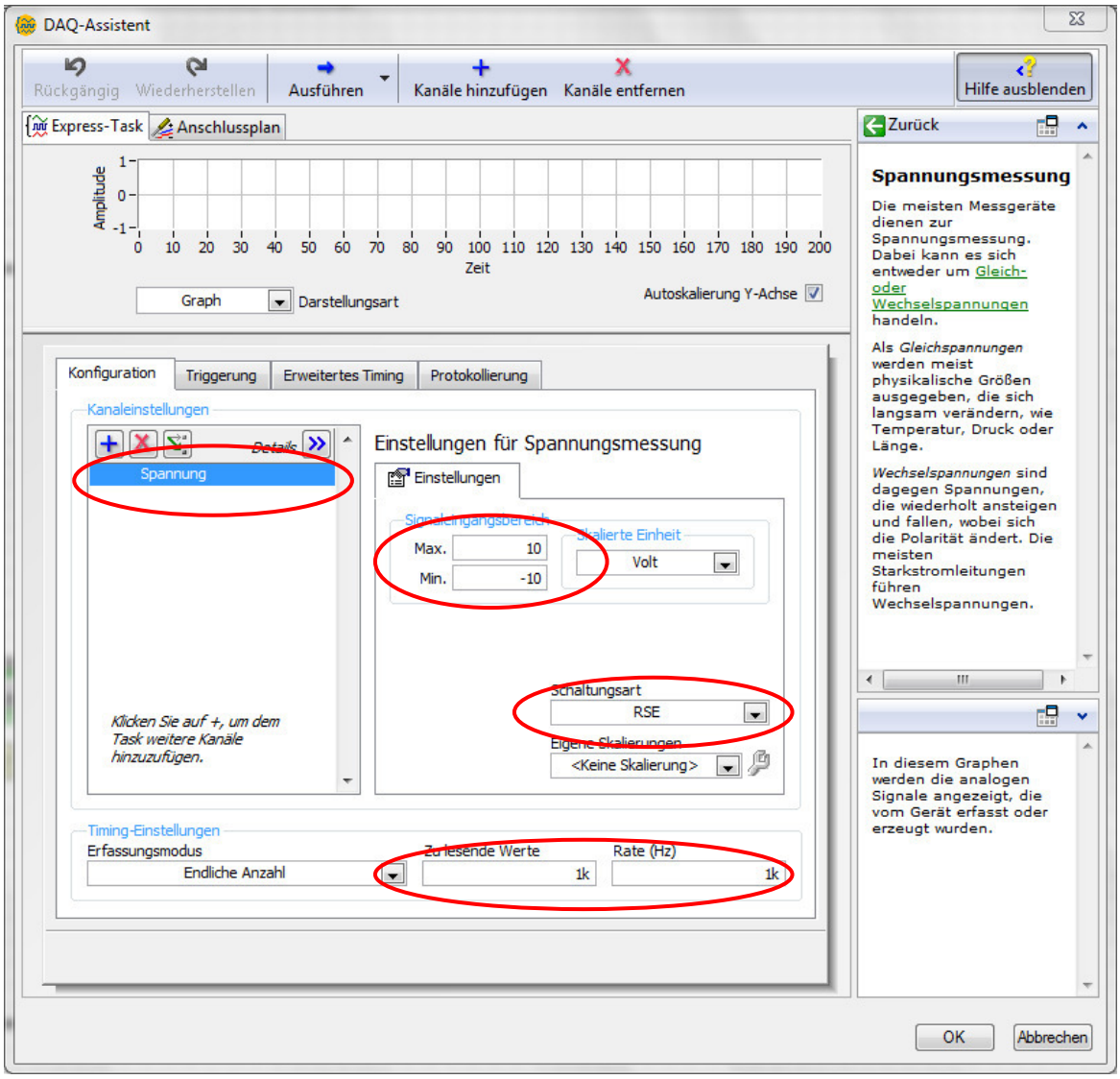


Abb.6.4: Die Grundeinstellung des DAQ-Assistenten

- Spannungsmessung über Kanal ai0 beim USB-6008er-Device 2.

  Hinweis:
  Das ist die einzige Stelle im gesamten VI, wo Sie eventuell etwas ändern müssen: hier tragen Sie nämlich den Kanal des von Ihnen benutzten NI-DAQ-Moduls ein.

- Ausnutzung des vollen Messbereiches von -10 V … 10 V.

- Messung der Spannung gegen Masse (Schaltungsart RSE).

- Die Werte für die zu lesende Anzahl der Messwerte und für die Abtastrate bleiben zunächst unverändert, diese Werte werden nachher von außen geändert bzw. festgelegt.

Danach wird der DAQ-Assistent geschlossen.

Die äußeren Anschlüsse dieses VIs werden nun wie folgt beschaltet:

- **Rate**: Bei ❻ wird, via Frontpanel, ein Bedienelement namens ´**Abtast-Rate Samp. per sec**´ erzeugt, über das die Abtastrate eingegeben werden kann.
  Über eine lokale Variable dazu, wird diese Eingabe an den ´**Rate**´-Anschluss des VIs gelegt.
  Die Eingabe über das Frontpanel überschreibt somit die im Inneren des VIs getroffene Festlegung zu diesem Parameter.

- **Sample-Anzahl**: Bei ❼ wird, via Frontpanel, ein Bedienelement namens ´**Dargestellte Sample-Anzahl**´ erzeugt, über das dieser Wert eingegeben werden kann.
  Über eine lokale Variable dazu, wird diese Eingabe an den ´**Sample-Anzahl**´-Anschluss des VIs gelegt.
  Die Eingabe über das Frontpanel überschreibt somit die im Inneren des VIs getroffene Festlegung zu diesem Parameter.

- **Timeout(s)**: Hier wird festlegt, wie viele Sekunden das VI auf die Messwerte (vom USB-6008er) warten soll, bevor eine Fehlermeldung ausgegeben wird, wenn sich das Modul bis dahin nicht mit **allen** angeforderten Messwerten zurück gemeldet hat.
  Dieser Wert legt letztendlich die maximale Zeitdauer für eine Messung fest.
  Ein Wert von 150 Sekunden ist hier sicherlich ausreichend.

- Am **Daten-Ausgang** des VIs, bei ❹a, wird zur graphischen Visualisierung ein ´**Signalverlaufsgraph**´ angeschlossen, den Sie zuerst über das Frontpanel einfügen müssen und der in ´Spannungsverlauf´ umbenannt wird.

Weitere Festlegungen an diesem Express-VI sind nicht notwendig.

Die Berechnungen bei ❽ und ❾ dienen zur Bestimmung der Zeiten für eine Abtastung und zur Ermittlung, wie lange die gesamte, hier eingestellte Messwerterfassung an sich dauert. Diese Werte werden ergänzend auf dem Frontpanel dargestellt.
Damit ist das Blockdiagramm fertig erstellt und Sie können jetzt das Frontpanel nach Ihren Vorstellungen gestalten, z.B. so wie in Abb.6.2.

Zum Schluss kann man noch ein kleines, aber sehr feines Extra dem Frontpanel hinzufügen: die (gelben) **Mess-Cursor-Linien mit Auswertebox** in x- und y-Richtung.
Rufen Sie dazu im Kontextmenü zum Signalverlaufsgraph den Punkt ´**Eigenschaften**´ auf und wählen Sie dort die Registerkarte ´**Cursor**´, **Abb.6.5**:

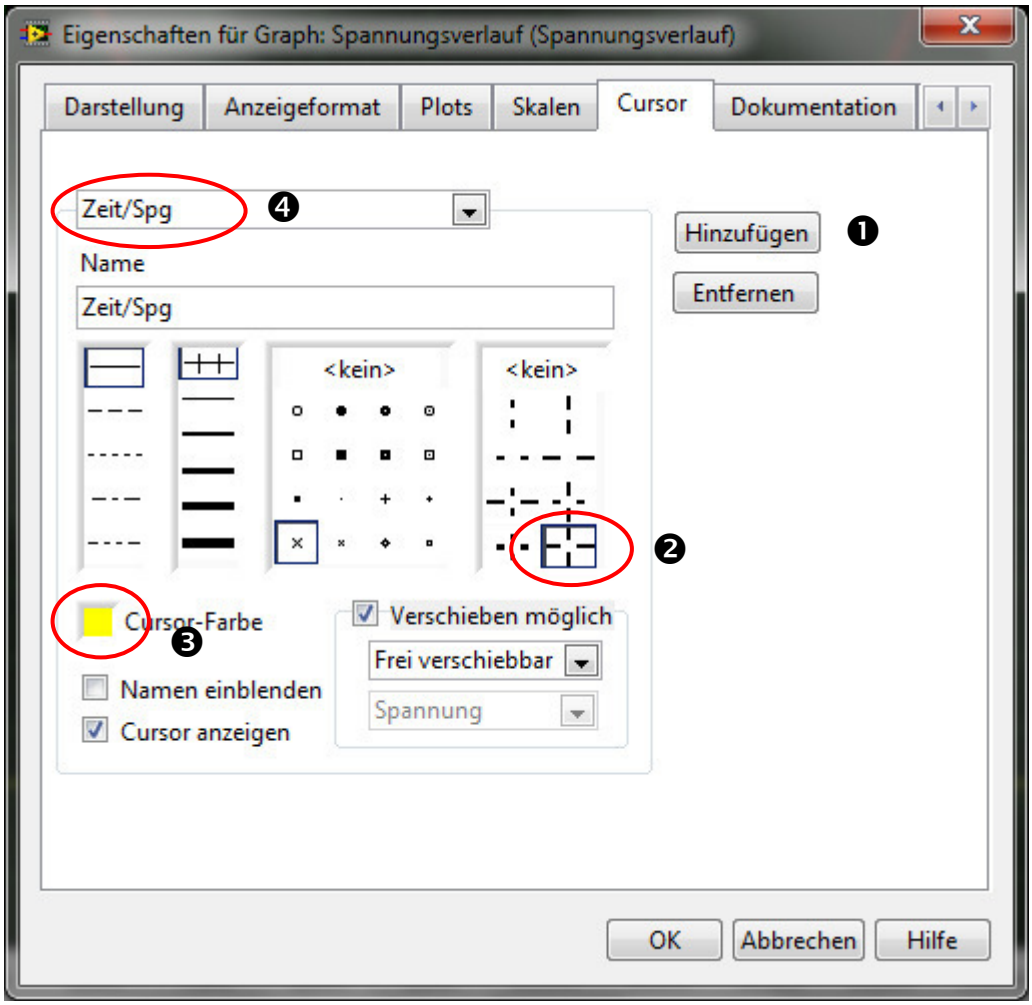


Abb.6.5: Das Einfügen von x- und y-Mess-Cursor-Linien

Klicken Sie auf ´Hinzufügen´, ❶, wählen Sie dann die Cursor-Form (zwei sich kreuzende Cursor-Linien), ❷, dann suchen Sie sich eine Farbe für die Cursor-Linien aus, ❸, und zum Schluss geben Sie eine passende Bezeichnung für das Cursor-Paar an, ❹.
Wenn Sie nun auf ´OK´ klicken, erscheint das Cursor-Linienpaar im Signalverlaufsgraphen.

Aber nicht nur das: es erscheint auch ein kleines **Cursor-Kontroll-Feld** auf dem Frontpanel, **Abb.6.6**:
(Sollte das Cursor-Feld nicht angezeigt werden, so können Sie dieses erreichen, indem Sie das Kontextmenü des Signalverlaufgraphens aufrufen und dort wählen: ´Sichtbare Objekte\Cursor-Legende´)

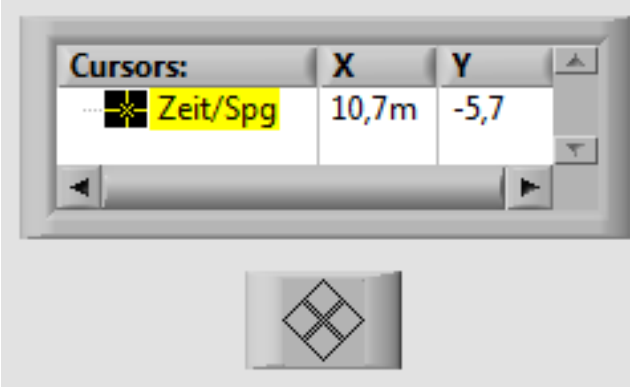


Abb.6.6: Das Cursor-Kontroll-Feld

Im Anzeige-Bereich dieses Feldes werden nun sofort die aktuellen Messwerte angezeigt, die der Cursor-Position entsprechen.
Umgekehrt können Sie hier auch direkt Werte eintragen, und die Cursor-Linien werden dann sofort an die entsprechenden Positionen gesetzt (vergessen Sie jedoch bei den Zeitangaben nicht den Buchstaben ´m´ für Milli (Sekunden) mit anzugeben, sonst verschwindet die senkrechte Cursor-Linie im Nirwana).
Zur genauen Positionierung der Cursor-Linien gibt es jetzt noch zwei weitere Möglichkeiten:

- Sie können im Signalverlaufsgraphen die Cursor-Linien direkt mit dem Maus-Bildschirmcursor „anfassen“ und verschieben.

- Zur Feinpositionierung können Sie auch das **rautenförmige Cursor-Steuerfeld** mit den vier Pfeilschaltflächen benutzen und so die Cursor-Linien ganz genau auf ihre Zielpunkte setzen.

Nun steht der praktischen Anwendung dieses USB-6008-Oszilloskops nichts mehr im Wege:

Gewünschte Abtastrate und Anzahl der dargestellten Samples eintragen und auf Trigger drücken --> jeweils eine Messung und Darstellung wird durchgeführt !

Praxis:

Dieses VI kann jetzt u.a. sehr gut dafür eingesetzt werden, um in der Lehre und in der Ausbildung die große Bedeutung des **Abtasttheorems** bei der Erfassung von analogen Meßsignalen sehr klar herauszustellen.
Schließen Sie dazu am Analogeingang einfach einen Frequenzgenerator an, mit dem Sie ein Sinus-Signal mit der Frequenz von ca. 100 Hz (Periodendauer: 10 ms) und einer Amplitude von ca. 7,5 V erzeugen.

Stellen Sie dann die Abtastraten und Anzahl der darzustellenden Samples gemäß der nachfolgenden Tabelle ein und beurteilen Sie das angezeigte Ergebnis !
(Führen Sie zur jeder Einstellung auch durchaus mehrere Messungen durch)

Beachten:
Die **maximale Abtastrate** beim USB-6008er-Modul beträgt **10 kHz** und bei darüber eingestellten Werten stürzt das VI ab, da dieser Wert bei der Eingabe zunächst nicht überprüft wird. Die NI-DAQmx-Funktionen (in Verbindung mit dem USB-6008er) erkennen diese Fehleingabe aber und stoppen das VI mit einer Fehlermeldung.

Abtastrate (Samples per sec)	**Dargestellte Sample-Anzahl**	**Abtastung alle …. ms**	**Samples pro Periode**	**Beurteilung des erfassten Signals**
Messdauer: 20 ms (≡ 2 Signalperioden)				
10.000	200			
5.000	100			
1.000	20			
500	10			
Messdauer: 200 ms (≡ 20 Signalperioden)				
5.000	1.000			
500	100			
400	80			
300	60			
200	40			
100	20			
444	89			
333	67			
222	44			

Und damit sind wir am Ende unserer kleinen Praxisprojekte mit dem USB-6008er-Modul angekommen.

7. Und wie geht´s weiter ?

Diese kleine Projekt-Sammlung solle Ihnen zeigen, wie einfach sich der erste Einsatz des USB-6008er-Moduls gestaltet, wenn man die fertig vorliegenden NI-DAQmx-Funktionen in Verbindung mit dem DAQ-Assistenten verwendet.

Wenn Sie noch viel mehr über LabVIEW erfahren wollen und vor allen Dingen LabVIEW in der täglichen Praxis einsetzen möchten, so gibt es jetzt verschiedene Möglichkeiten, Ihr Wissen zu erweitern:

Im September 2012 erschien im Elektor-Verlag der erste Band einer von uns herausgegebenen **Lehrbuchreihe** zum Thema "**LabVIEW für den Praktiker**", [2].
Hiermit wird ein fundierter Einstieg in LabVIEW mit vielen praktischen Beispielen, Projekten und Übungen (mit Musterlösungen) möglich.
Angefangen von den Grundlagen über den Betrieb von seriellen Schnittstellen bis hin zur graphischen Darstellung und Auswertung von Daten reicht zunächst das Spektrum in den ersten beiden Bände.

Weiterhin veranstalten wir in verschiedenen Orten Deutschlands mehrtägige **Seminare und Workshops** zum Themenbereich „**LabVIEW meets µC**".

Nähere Informationen dazu finden Sie im Kapitel ´Literatur, Seminare und Bezugsquellen´.

8. Literatur, Seminare und Bezugsquellen

[1] National Instruments
BEDIENUNGSANLEITUNG UND SPEZIFIKATIONEN
NI USB-6008/6009
Busversorgtes multifunktionales USB-Datenerfassungsgerät
www.ni.com

[2] Bernd vom Berg, Peter Groppe
„LabVIEW – Einstieg in die Praxis (Band 1)“
Elektor-Verlag Aachen, September 2012

Weiterführende Literatur:

Wolfgang Georgi, Ergun Metin
„Einführung in LabVIEW“
5. überarbeitete und erweiterte Auflage, Januar 2012
Carl Hanser Verlag GmbH & CO. KG
ISBN-13: 978-3446423862

Das große deutschsprachige Hilfeforum zu LabVIEW:

http://www.labviewforum.de/index.php

Mehrtägige Seminare und Workshops für LabVIEW-Anfänger, Neueinsteiger, Schüler, Auszubildende und Lehrer/Ausbilder und LabVIEW-Projekte, Mikrocontroller-Systeme für Lehre und Ausbildung , u.v.a.m. von der Firma PalmTec:

www.palmtec.de

Weiterführende Informationen zu LabVIEW und Seminare für fortgeschrittene (Pro-fi)Anwender bei National Instruments:

www.ni.com

9. Versions History

19. November 2012: Version 1.0 des Projektes veröffentlicht.