



**MNI-Fonds für Unterrichts- und Schulentwicklung
S 6 „Anwendungsorientierung und Berufsbildung“**

LabVIEW[®] als Lern- und Experimentierwerkzeug im Physikunterricht

**Selbständige Programmierung von Simulationen zum besseren Verständnis
physikalischer Zusammenhänge**

**Eine Projektarbeit zur Weiterentwicklung des naturwissenschaftlichen Unter-
richts**

Dipl.-Ing. Wolfgang Bernhofer

**Schülerinnen und Schüler des 1. und 2. Jahrgangs der Höheren Abteilung der
HBLVA für chemische Industrie
Wien 17, Rosensteingasse 79**

Wien im Juni 2006

INHALTSVERZEICHNIS

INHALTSVERZEICHNIS	2
ABSTRACT	5
1 EINLEITUNG	6
1.1 Ich will Experimente die beeindrucken	6
1.2 Unerwartete Aufmerksamkeit.....	6
1.3 LabVIEW im Unterricht an Hochschulen.....	6
2 AUFGABENSTELLUNG	8
2.1 Neue Formen der Computersimulation	8
2.2 Neue Formen des Experiments im Unterricht	8
2.3 Das Physikpraktikum im Klassenraum.....	8
2.3.1 Demonstrationsversuche in der Klasse.....	8
2.3.2 Versuche im Praktikum	9
3 METHODEN UND DURCHFÜHRUNG	10
3.1 Voraussetzungen	10
3.2 Evaluierung	10
3.3 Erlernen der Grundfertigkeiten.....	10
3.3.1 Das erste Programm (ausführlich im Anhang)	11
3.3.2 Beispiel - Lamperl ein/aus.....	12
3.3.3 Fortführende Erklärungen der Programmiertechniken	12
3.3.4 Selbständiges Ausarbeiten von einfachen Aufgaben.....	12
3.4 Simulationen im Unterricht.....	13
3.4.1 Geradlinige gleichförmige Bewegung	13
3.4.2 Geradlinige gleichmäßig beschleunigte Bewegung	16
3.4.3 Reflexionsgesetz – Pingpong.....	18
3.4.4 Mathematisches Pendel.....	19
3.4.5 Gasgleichung einfach – manuell zu bedienen.....	20
3.4.6 Hebelgesetz	24
3.4.7 pH-Messung.....	25
3.4.8 Indikatorfarben	26
3.4.9 Interferenz.....	28

4	ERGEBNISSE	30
4.1	Unterrichtserfahrungen mit den einzelnen Übungen und Simulationen	30
	Das erste Programm	30
	Beispiel - Lamperl ein/aus	30
	Fortführende Erklärungen der Programmiertechniken.....	30
	Selbständiges Ausarbeiten von einfachen Aufgaben	30
	Geradlinige gleichförmige Bewegung	30
	Geradlinige gleichmäßig beschleunigte Bewegung.....	30
	Reflexionsgesetz – Pingpong	30
	Mathematisches Pendel	31
	Gasgleichung.....	31
	Hebelgesetz.....	31
	pH-Messung	31
	Indikatorfarben.....	31
	Interferenz	31
4.2	Verständnis, pädagogischer und didaktischer Nutzen	31
4.2.1	Verständnis	31
4.2.2	Pädagogischer Nutzen.....	32
4.2.3	Didaktischer Nutzen.....	32
4.3	Freude auf dem Weg	32
4.4	Beurteilung durch die Schülerinnen und Schüler	33
5	DISKUSSION UND AUSBLICK	34
5.1	Skepsis der Schüler	34
5.2	Die Angst des Lehrers.....	34
5.3	Diskussion.....	35
5.4	Ausblick.....	36
6	LITERATUR.....	37
6.1	Weiterführende Literatur	37
6.2	LabVIEW Internet-Adressen	38
6.2.1	National Instruments Austin Texas	38
6.2.2	LabVIEW Mailgroup.....	38
6.2.3	Brian Renken LabVIEW-pages	38
6.2.4	LabVIEW in der Ausbildung	38

7	ANHANG	39
7.1	Begegnung mit LabVIEW.....	39
7.2	Das Konzept von LabVIEW.....	41
7.3	Was ist LabVIEW?	42
7.4	Erste Übung	45
7.5	Programmierkurs - LabView - Serie 1	51
7.5.1	Aufgabe 1.....	51
7.5.2	Aufgabe 2.....	52
7.5.3	Aufgabe 3.....	52
7.5.4	Aufgabe 4.....	54
7.5.5	Aufgabe 5.....	55
7.5.6	Aufgabe 6.....	55
7.5.7	Aufgabe 7.....	56
7.5.8	Aufgabe 8.....	56
7.5.9	Aufgabe 9.....	57
7.6	Programmierkurs - LabView - Serie 2	58
7.6.1	Aufgabe 1.....	58
7.6.2	Aufgabe 2.....	59
7.6.3	Aufgabe 3.....	60
7.6.4	Aufgabe 4 – ohne Lösung	63
7.6.5	Aufgabe 5 – ohne Lösung	64
7.6.6	Aufgabe 6.....	64
7.7	Serie 3 – ohne Lösungen	64
7.7.1	Aufgabe 1.....	64
7.7.2	Aufgabe 2.....	64
7.7.3	Aufgabe 3.....	64

ABSTRACT

Diese Arbeit widmet sich der Einführung der Entwicklungsumgebung LabVIEW in den Physikunterricht der 10. und 11. Schulstufe mit dem Ziel, Schülerinnen und Schülern komplexe Zusammenhänge physikalischer Phänomene mit Hilfe des Computers näher zu bringen. Der einfache und intuitive Umgang mit der Programmiersprache G sollte sie anregen eigene Problemlösungsstrategien zu entwickeln und sich physikalische Zusammenhänge und Ergebnisse selbst programmierter Simulationen und einfachster Messapparaturen am Bildschirm darstellen lassen zu können.

Durch die spielerische Annäherung wurden gegen das Erlernen von Programmierfähigkeiten keine Barrieren aufgebaut. Anfängliche Skepsis von Schülerinnen und Schülern der graphischen Programmierung gegenüber schlug bald in Begeisterung um, eigene kreative Entwicklungen ohne lange Vorlaufzeiten stattfinden lassen zu können.

Schulstufe:	9 + 10
Fächer:	Angewandte Physik
Kontaktperson:	Dipl.-Ing. Wolfgang Bernhofer
Kontaktadresse:	HBLVA 17, 1170 Wien, Rosensteingasse 79
E-Mailadresse	wolfgang.bernhofer@tele2.at

1 EINLEITUNG

In unserer Schule spielt der Physikunterricht eine tragende Rolle. Da die Schüler eine fundierte chemische Ausbildung erhalten, muss im Physikunterricht eine Basis für das allgemeine wissenschaftliche Verständnis geschaffen werden.

1.1 Ich will Experimente die beeindrucken

Riesige Stoppuhren, veraltete Federwaagen, an ein Magnetboard geheftete Linsen die mit einer Taschenlampe als Lichtquelle zur Demonstration von Strahlengängen dienen sowie Ampere- und Voltmeter, die aussehen als wären sie zu Kaisers Zeiten schon veraltet gewesen, all das ist im Physiksaal ein sicherer Lacherfolg unter den Schülerinnen und Schülern, die im Computerzeitalter aufgewachsen sind. Aber auch die Vorführung moderner Computeranimationen, die noch vor ein paar Jahren Staunen auslösten, verlieren immer mehr an Glanz. Durch immer perfekter werdende Wissenschaftssendungen des Fernsehens, die, weil zu den besten Sendezeiten ausgestrahlt, von vielen gesehen werden, sind die Schüler aufwendigste Animationen gewohnt. Also stellte sich die Frage, wie es gelingen könnte, die Aufmerksamkeit der Schüler zurückzugewinnen.

1.2 Unerwartete Aufmerksamkeit

Als ich eines Tages im Unterricht mit dem behandelten Stoff weiter kam als ich es geplant hatte, stellte ich fest, dass ich die Simulation zur Demonstration der Überlagerung von Schwingungen nicht auf meinem Rechner vorbereitet hatte. Da es aber zum Verständnis der Addition der Amplituden essentiell ist, eine graphische, möglichst interaktive Darstellung des Sachverhaltes vorzuführen, entschloss ich mich kurzerhand mit Hilfe von LabVIEW ein solches Programm direkt während des Unterrichts zu entwickeln. Dabei hatte ich bewusst den Beamer eingeschaltet gelassen, damit die Schülerinnen und Schüler verfolgen konnten, was ich tat. Plötzlich wurde es in der Klasse ruhiger als sonst üblich, wenn ich etwas auf meinem Laptop arbeite, das nicht unmittelbar zum Unterricht gehört. Offensichtlich faszinierte die Erstellung des Programms und ich begann erklärende Worte zur Programmierung zu sprechen. Daraus entwickelte sich eine intensive Diskussion über die Entwicklung, die Simulation selbst und mögliche Variationen der physikalischen Zusammenhänge, die ich ebenfalls augenblicklich in das eben geschaffene Programm einfließen ließ. Dieses überraschende Interesse regte mich an, diese Form des Unterrichts näher zu untersuchen.

Da LabVIEW schon von vielen Unterrichtenden vor allem an Universitäten eingesetzt wurde, begann ich die Einsatzmöglichkeiten zu recherchieren.

1.3 LabVIEW im Unterricht an Hochschulen

LabVIEW wurde vor allem in Forschungslaboratorien amerikanischer Hochschulen und staatlicher Forschungsinstitutionen eingesetzt. Die in der Literatur angeführten Arbeiten kamen zu dem Schluss, dass LabVIEW ist ein ideales Werkzeug in einer stark mutierenden Laborumgebung, wo möglichst rasch komplexe Messaufgaben gelöst werden müssen, ist [1].

Mit LabVIEW kann ein Forscher selbst eine Messapplikation programmieren, ohne viel Zeit für die eigentliche Forschungsarbeit zu verlieren. Einige amerikanische Universitäten haben deshalb sehr früh LabVIEW in ihr naturwissenschaftliches Unterrichtsprogramm integriert [2].

Die Herausgabe einer LabVIEW Student Edition von National Instruments in Zusammenarbeit mit dem Verlag Prentice Hall hat dies unterstützt (www.ni.com).

Die LabVIEW Student Edition kostet rund € 10,- und enthält eine LabVIEW-Grundversion mit vereinfachten Bibliotheken und ein konsequent aufgebautes Lehrbuch.

Mit der Student Edition lassen sich bereits einfache bis mittelschwere Messapplikationen programmieren. Sie ist für den Einstieg sowohl in einer Schulklassensituation wie auch zum autodidaktischen Studium ideal. Die vernünftig dimensionierten Bibliotheken erhöhen für einen Einsteiger die Transparenz und begrenzen die Gefahr, sich in Details zu verlieren. Spezielle Hardwaretreiber für die LabVIEW Student Edition können von der National Instruments Webseite kopiert werden; sie sind auch in der Software zur LabVIEW Student Edition Lehrerversion enthalten.

Vis, so werden LabVIEW-Programme bezeichnet [siehe Anhang – Begegnung mit LabVIEW], die mit der Student Edition programmiert wurden, können ohne weiteres in die Vollversionen übernommen werden. Die gegenwärtige LabVIEW-Vollversion 8.0 kostet mit Schulvergünstigung momentan € 5850,- für die Campuslizenz bzw. €1800,- für eine 10-Platz-Lizenz, was manchen potentiellen Interessenten als übertrieben teuer erscheint.

Betrachtet man aber den enormen Umfang, die Qualität der Hardwaretreiber und die erhebliche Produktivitätssteigerung, welche die LabVIEW Entwicklungsumgebung bietet, so ist das Preis- Leistungsverhältnis sehr gut. Heute zahlt man beispielsweise für ein schnelles Oszilloskop einen höheren Preis, obschon man in diesem Fall ein Messgerät für einen relativ schmalen Anwendungsbereich besitzt [3].

2 AUFGABENSTELLUNG

Ziel der vorliegenden Arbeit war es, LabVIEW als Lern- und Experimentierwerkzeug in den Physikunterricht unserer Schule einzuführen um den Schülern zu ermöglichen eigene Simulationen zu den im Unterricht durchgenommenen physikalischen Phänomenen zu erarbeiten. Dadurch sollte die Aufmerksamkeit bei der Auseinandersetzung mit dem eben erlernten Stoff erhöht und die eigene Kreativität der Schülerinnen und Schüler gefördert werden.

2.1 Neue Formen der Computersimulation

Die Schüler sollten vom üblichen Einsatz des Computers als Demonstrationsplattform für kleine Applikationen, die meist optisch einfach gehalten und teilweise relativ veraltet sind, weggeführt werden. Das Ziel ist den Computer als ein flexibles Werkzeug zur Darstellung komplexer Sachverhalte, das rasch und individuell an die jeweilige Problemstellung angepasste Darstellungsmöglichkeiten offeriert und mit dem auf einfachste Weise Simulationen der verschiedensten Versuchsabläufe programmiert werden können, zu sehen.

2.2 Neue Formen des Experiments im Unterricht

Riesige Stoppuhren und veraltete Federwaagen haben im Computerzeitalter nichts mehr verloren. Deshalb sollte mit dieser Arbeit die Möglichkeiten zur Einführung moderner Mess- und Steuerungstechnik für die Durchführung von Experimenten im Klassenraum erhoben werden.

2.3 Das Physikpraktikum im Klassenraum

Durch die Möglichkeit einfache Versuchsabläufe am Computer zu simulieren, einfache Experimente messtechnisch zu erfassen und Daten auszuwerten, sollte es möglich sein, dass die Schülerinnen und Schüler Experimente selbst am Computer oder mit Hilfe des Computers durchführen, ohne entsprechende Laborausstattung zur Verfügung stellen zu müssen. Aufgabe sollte es sein, die Möglichkeiten für einen solchen Einsatz abzuklären.

2.3.1 Demonstrationsversuche in der Klasse

In diesem Bereich könnte sich LabVIEW als ideales, intelligentes Messsystem mit vielseitigen Analyse- und Darstellungsmöglichkeiten etablieren. Wo früher zahlreiche anwendungsspezifische Messapparate mit entsprechendem Zeit- und Kostenaufwand aufgebaut wurden, sollte es möglich werden, die anfallenden Messaufgaben in Form „virtueller Messinstrumente“ in viel kürzerer Zeit zu erfüllen. Besonders die Möglichkeit der transparenten und attraktiven Messdatendarstellung für ein großes Publikum und die vielseitige Adaptierfähigkeit von LabVIEW sollten bestehende Physikdemonstrationsversuche erheblich aufwerten und zu zahlreichen neuen Anwendungen führen.

2.3.2 Versuche im Praktikum

Versuche im Praktikum sollten mit LabVIEW-Instrumentierung durchgeführt werden, um es den Schülern zu ermöglichen, auf moderne Art und Weise mit Messungen, Computern und Messdaten umzugehen. Weil bei solchen Anwendungen die Erfassung von Messdaten weitgehend automatisch erfolgt und Daten im Rechner abgespeichert und von einem Drucker ausgedruckt werden, steht mehr Zeit für physikalisch relevante Datenanalysen zur Verfügung.

3 METHODEN UND DURCHFÜHRUNG

3.1 Voraussetzungen

Um LabVIEW so in den Unterricht einzuführen, wie es in dieser Arbeit beschrieben wird, ist es unerlässlich, dass die Schüler Zugang zu entsprechenden, mit LabVIEW ausgestatteten PCs haben. Maximal drei Schüler sollten einen Computer zur Verfügung haben, wobei aber darauf zu achten ist, dass die Rollenverteilung innerhalb dieser Gruppen regelmäßig verändert wird.

3.2 Evaluierung

Um den Lernerfolg, der sich durch den Einsatz von LabVIEW verbessern sollte zu überprüfen, wurde ein Vergleich der Prüfungsergebnisse jener Teilgebiete, die mit Hilfe von selbständig erarbeiteten Simulationen durchgenommen wurden mit den konventionell vorgetragenen Teilgebieten angedacht. Da ein solcher Vergleich praktisch aber nicht wirklich aussagekräftig wäre, wurde auf diese Form der Evaluierung verzichtet.

Zur Feststellung des Erfolgs wurden stattdessen Interviews mit den Schülerinnen und Schülern, sowie zu einem großen Teil Selbstbeobachtungen herangezogen.

Wichtige Fragestellungen für die Schülerinterviews waren:

- Der Grad der Befriedigung, der erreicht wurde, wenn die Fragestellung der Aufgabe durch eigene Leistung erarbeitet wurde.
- Die Hilfestellung, die die Schülerinnen und Schüler sich gegenseitig gaben.
- Das Verständnis für die Programmierung und der Zusammenhang zwischen der programmierten Simulation und der damit behandelten physikalischen Realität.

Als Kriterien meiner Selbstbeobachtungen zog ich folgende Punkte heran:

- Die Veränderungen der Fähigkeit der Schülerinnen und Schüler, die gestellten Aufgaben zu lösen, wurden analysiert.
- Durch Beobachtung der Arbeitsintensität der einzelnen Gruppen, versuchte ich festzustellen, ob die Schülerinnen und Schüler intensiv mit den Themen auseinander setzten.
- Diskussionen wurden beobachtet um Mängel in der Formulierung der Aufgabenstellung herauszufinden und möglichst eine Verbesserung der Verständlichkeit der Ziele zu erreichen.
- Die Kooperation der Schülerinnen und Schüler innerhalb der Gruppen stellte ebenfalls einen wichtigen Indikator dar.

3.3 Erlernen der Grundfertigkeiten

Da keiner der Schülerinnen und Schüler Erfahrung mit der Programmierung von Computern hatte, musste das Erlernen der Grundfertigkeiten voran gestellt werden. Diese Phase dauerte 8 Stunden und gliederte sich in mehrere Teile:

- Das erste Programm ($\frac{1}{2}$ -1 Stunde)
- Fortführende Erklärungen der Programmier Techniken (2 Stunden)
- Selbständiges Ausarbeiten von einfachen Aufgaben (5-6 Stunden)

3.3.1 Das erste Programm (ausführlich im Anhang)

Zur Heranführung der Schülerinnen und Schüler wurde ein Programm zur einfachen Addition zweier Zahlen sehr ausführlich besprochen und alle Details der LabVIEW-Bedienung erklärt.

In erster Linie ging es dabei um die Bedeutung der Fenster, der Icons, der Struktur und des Verbinders. Da sich der geneigte Leser aber nichts unter diesen einführenden Beschreibungen vorstellen kann, habe ich mich entschlossen ausnahmsweise genau die Vorgehensweise im Unterricht zu rekonstruieren, um einen besseren Eindruck zu vermitteln (siehe Anhang).

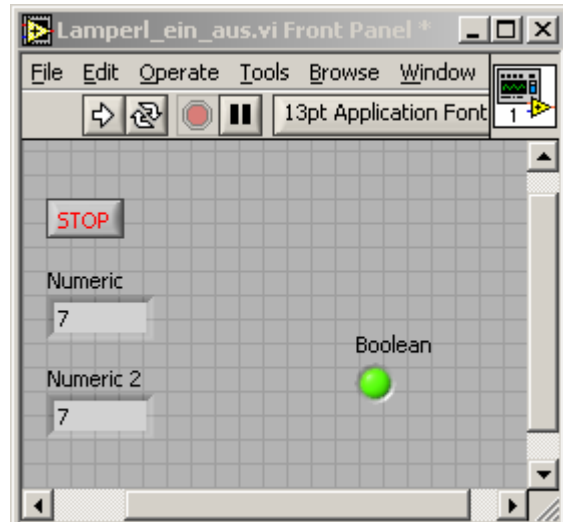
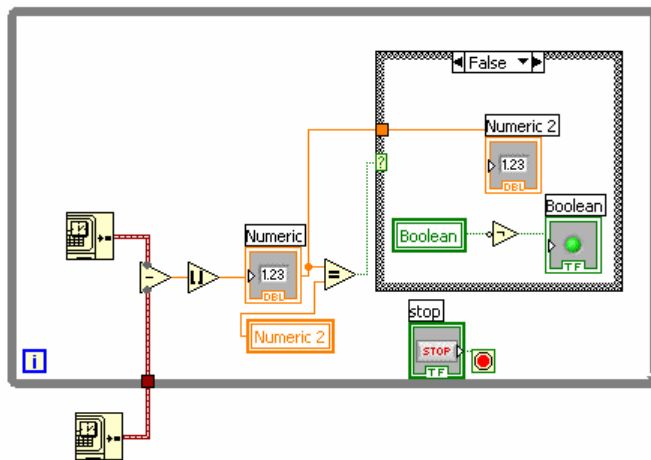
In späterer Folge werde ich nur mehr die fertigen Programme anführen und die Funktion kurz beschreiben.

Nachdem der Physikunterricht in den EDV-Saal verlegt wurde, wurde dafür gesorgt, dass die 16 zur Verfügung stehenden Arbeitsplätze mit jeweils zwei Schülern besetzt wurden.

Nach dem Bootvorgang, den ich immer dazu nützte einführende Worte zur bevorstehenden Unterrichtseinheit zu sprechen, wurde von mir über eine Beamerprojektion die im Anhang unter „einführende Übung“ beschriebene Unterrichtseinheit präsentiert.

3.3.2 Beispiel - Lamperl ein/aus

Dieses Programm wurde ebenfalls Schritt für Schritt vorgeführt und erklärt, wobei schon darauf geachtet wurde, dass alle Schülerinnen und Schüler ebenfalls jeweils eine funktionierende Version auf ihren Computern programmierten. Die Zweiergruppen tauschten anschließend die Rollen, damit auch der, der bei der Programmierung nur zugesehen hatte, selbst am Computer arbeiten konnte.



3.3.3 Fortführende Erklärungen der Programmieretechniken

Die Schülerinnen und Schüler wurden mit einem eigens entwickelten Schulungsprogramm an die Programmierung in LabVIEW herangeführt. Diese Phase, abgesehen von der einführenden Übung dauerte nur eine Doppelstunde. Dabei wurde auf den 3-Stunden Einführungskurs der Firma National Instruments zurückgegriffen, der aus urheberrechtlichen Gründen hier nicht angehängt werden kann. Dieser Kurs ist aber jederzeit nach Registrierung von der Homepage der Firma (<http://www.ni.com>) kostenlos herunterzuladen. Um keine Sprachbarriere aufzubauen, habe ich die in englischer Sprache verfassten Unterlagen übersetzt.

Das Durcharbeiten der Kursunterlagen erfolgte unter Anleitung. Dabei wurden von mir über eine Beamerprojektion die einzelnen Schritte vorgeführt und erklärt, anschließend von den Schülern nachgearbeitet. Die fertigen Übungsprogramme wurden von den Schülerinnen und Schülern anschließend noch über das Ausmaß der Kursanforderungen hinaus modifiziert, erweitert und durch optische Aufwertungen verziert.

3.3.4 Selbständiges Ausarbeiten von einfachen Aufgaben

Nach dem Durcharbeiten des Einführungskurses der Fa. National Instruments sollten weitere Programmierfertigkeiten durch Ausarbeiten einfacher Aufgaben erworben werden.

LabVIEW ist im universitären Ausbildungsbereich mittlerweile so verbreitet, dass Einführende Kurse und Unterrichtsmaterialien im Überfluss vorhanden sind. Einfache Suchabfragen im Internet ergeben hunderte von Hinweisen, verlinken auf Universitäts- oder Collageseiten, liefern Beispielprogramme und helfen mit den allseits beliebten FAQs weiter. Da ich, von dieser Angebotsflut fast erschlagen, das Rad nicht neu erfinden wollte, habe ich mich bei der Ausarbeitung der Grundübungen stark an die folgenden Vorgaben gehalten:

LabVIEW™ - eine grafische Programmiersprache geeignet für den Unterricht

Urs Lauterburg; Physikalisches Institut der Universität Bern; Sidlerstr. 5; 3012 BERN

Der von mir überarbeitete Kurs, der aus 3 Serien besteht, ist im genauen Wortlaut mit Lösungen im Anhang angeführt.

Beim Erarbeiten der Übungsbeispiele war es mir wichtig in den Lernprozess so wenig wie möglich einzugreifen, denn das Erlernen von Programmierfähigkeiten setzt einen großen Anteil an selbständigem Arbeiten voraus. Das Einüben prozeduraler Denkprozesse sowie das Umsetzen in ein Programm, sollten vom Lernenden direkt am Computer erfolgen, wobei das Abstimmen mit anderen Denkansätzen der anderen Mitschüler der Klasse von Vorteil ist.

Die Beispiele waren von unterschiedlichem Schwierigkeitsgrad und mussten in der richtigen Reihenfolge erarbeitet werden.

3.4 Simulationen im Unterricht

3.4.1 Geradlinige gleichförmige Bewegung

Nachdem die Formel für die geradlinige gleichförmige Bewegung besprochen wurde, wurde eine Simulation gemeinsam über eine Beamerprojektion so entwickelt, dass ein Freiwilliger die Umsetzung der Vorschläge der Mitschülerinnen und Mitschüler am Computer umsetzte. Dieses Grundprogramm wurde soweit entwickelt, dass eine Struktur zu erkennen war und minimale Anforderungen an die optische Aufbereitung der physikalischen Vorgänge erfüllt wurden.

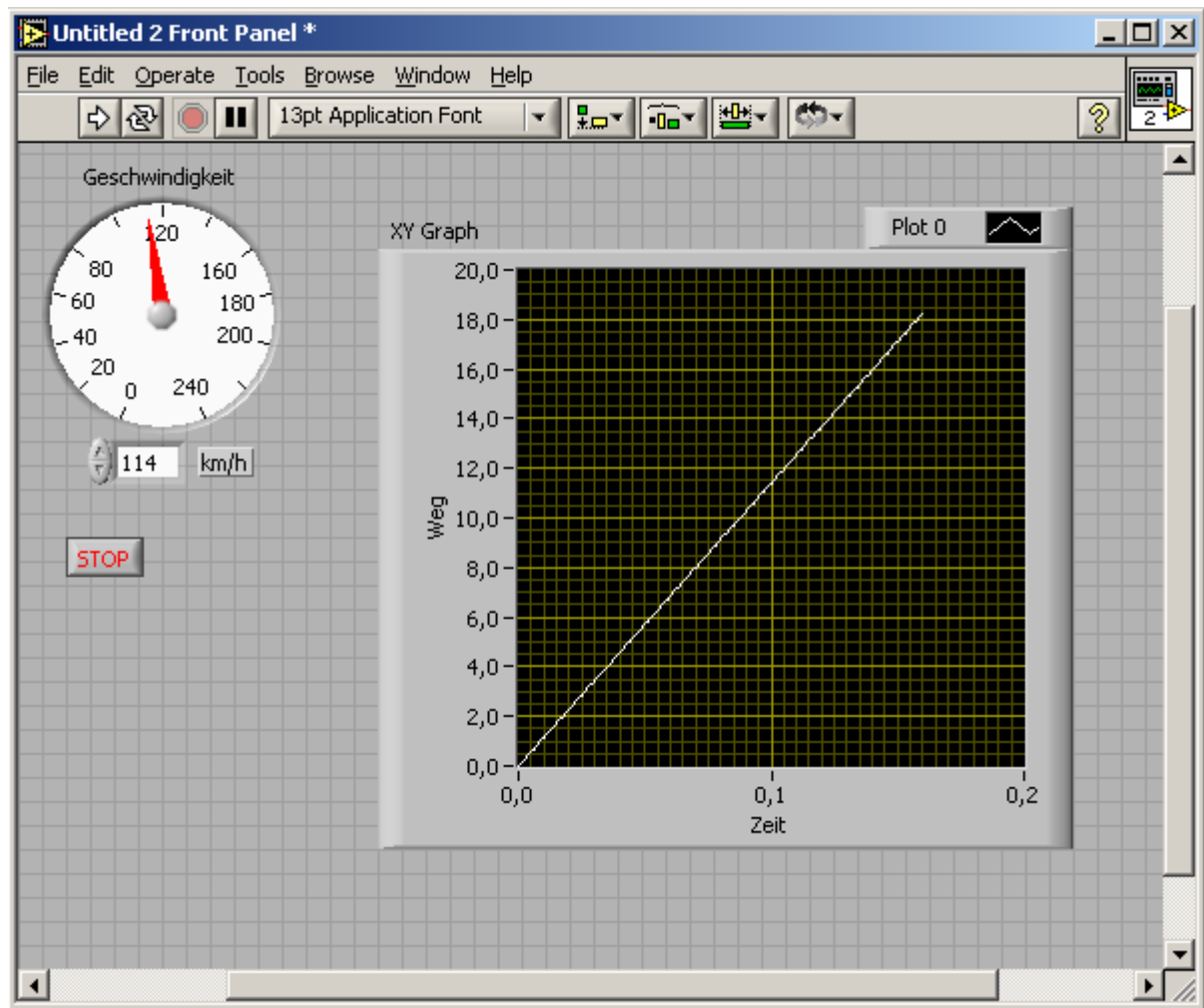
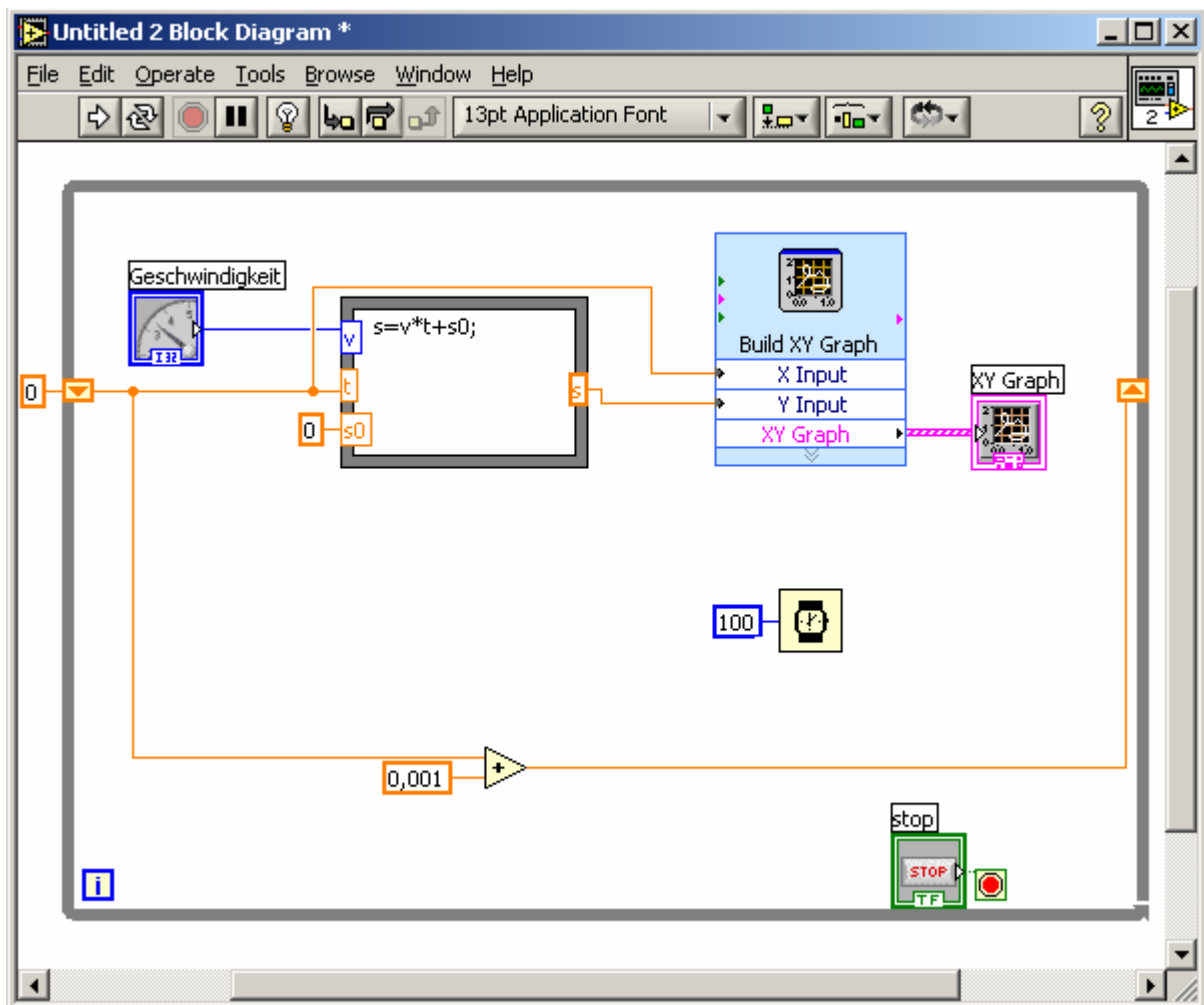


Abbildung 1

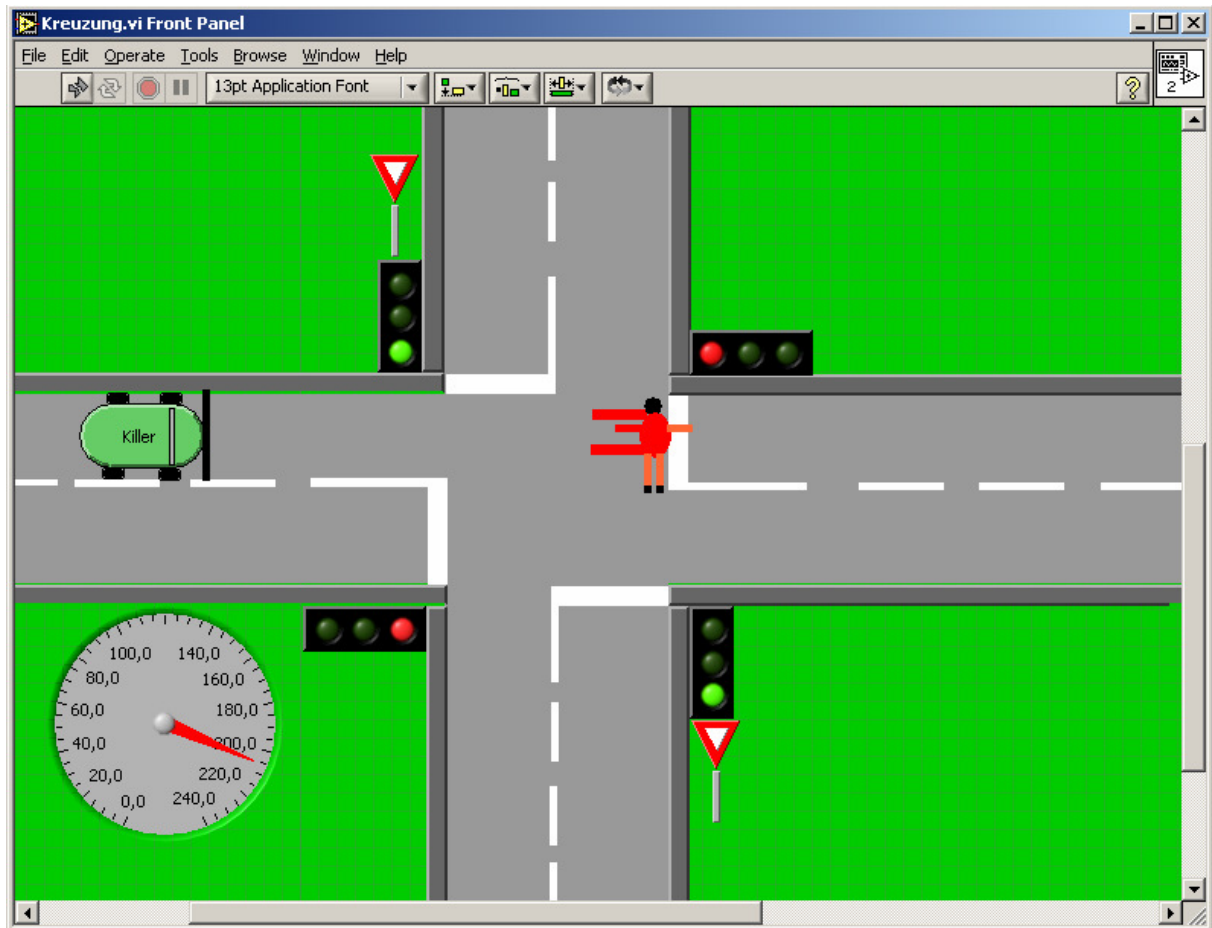
Front Panel der Simulation „geradelinige gleichförmige Bewegung“



Auf dieses Programm aufbauend entwickelten die Schülerinnen und Schüler, ihre erworbenen Programmierkenntnisse anwendend, aufwendigere Applikationen.

Nach Internetrecherchen wurden Vorlagen von Schülern der HTL Braunau so erweitert, dass sich Autos über Kreuzungen bewegten, die durch Ampeln geregelt waren. Die Programmvorlagen dazu sind im Internet erhältlich: www.htl-braunau.at

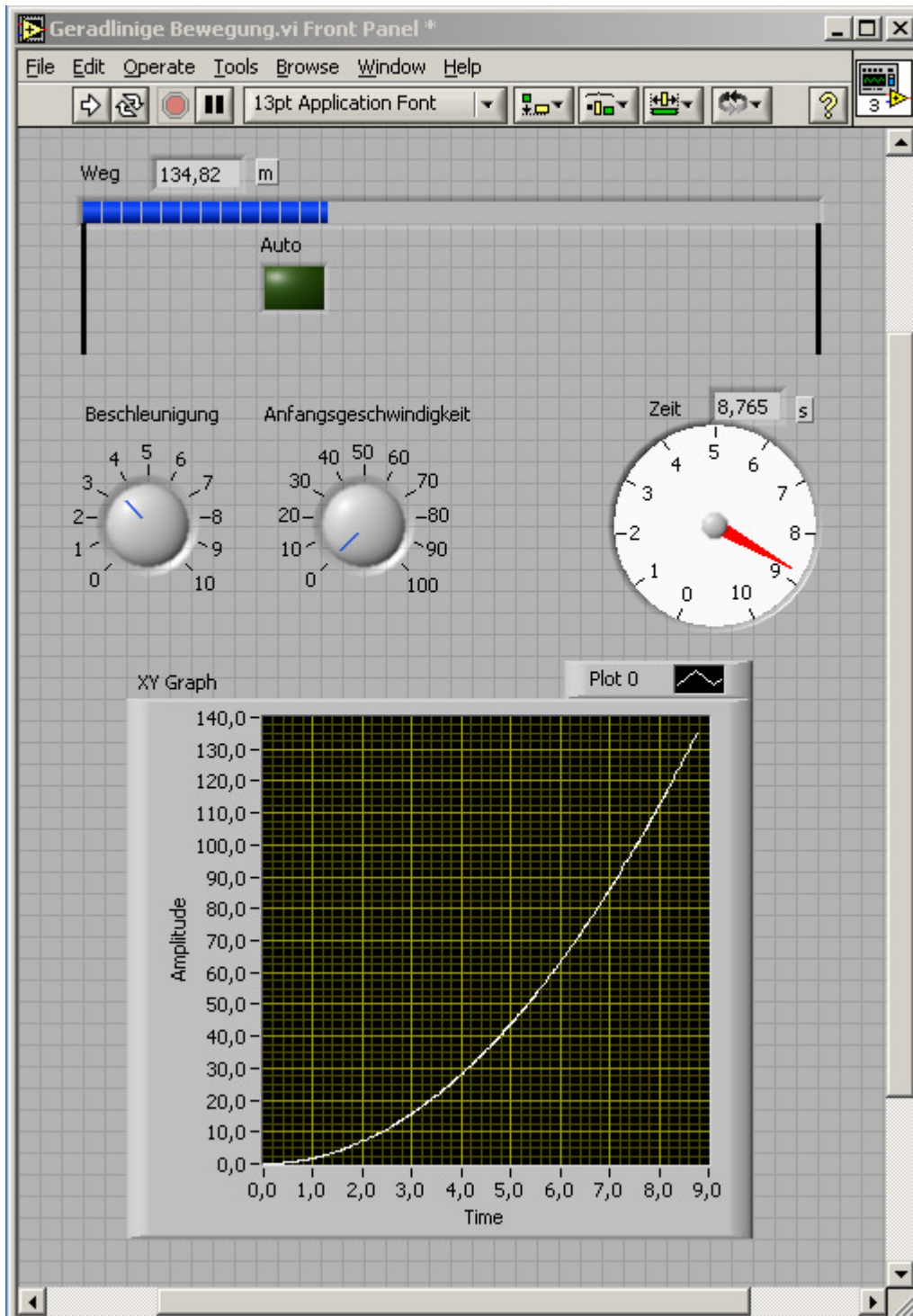
Als Beispiel sei hier ein Programm angeführt, bei dem ein unvorsichtiger Fußgänger überfahren wird oder sicher die Straße überquert, je nachdem wie schnell das Auto fährt. Dabei wurde von den Schülern zwar großes Augenmerk auf die simulierte Blutspur gelegt, der Spaß kam dabei aber sicher nicht zu kurz.

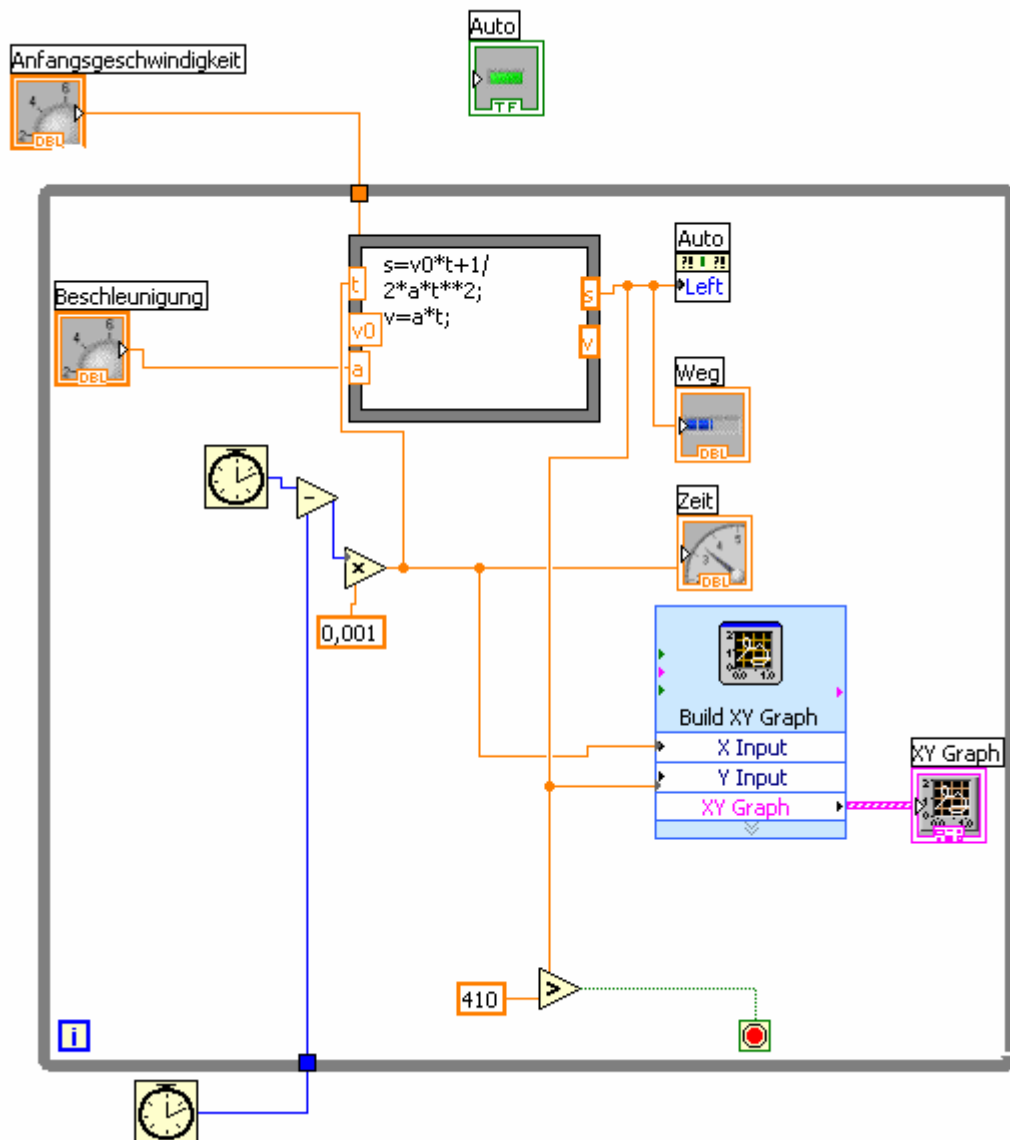


3.4.2 Geradlinige gleichmäßig beschleunigte Bewegung

Die Vorgehensweise war wieder völlig analog, wobei die graphische Aufbereitung schon in der Grundversion viel anspruchsvoller ausfiel als im vorherigen Beispiel, weil die Schülerinnen und Schüler durch intensive Mitarbeit die Komplexität immer weiter in die Höhe trieben. Um nicht schon vor der Phase, die dafür vorgesehen ist, die Schüler selbständig die Aufgabe ausarbeiten zu lassen, wurde der Lehrerrechner in einem Rotationsprinzip von den Schülern selbst bedient, sodass schon in der ersten Fassung des Programms die Hauptentwicklungsarbeit von den Schülerinnen und Schülern selbst geleistet wurde.

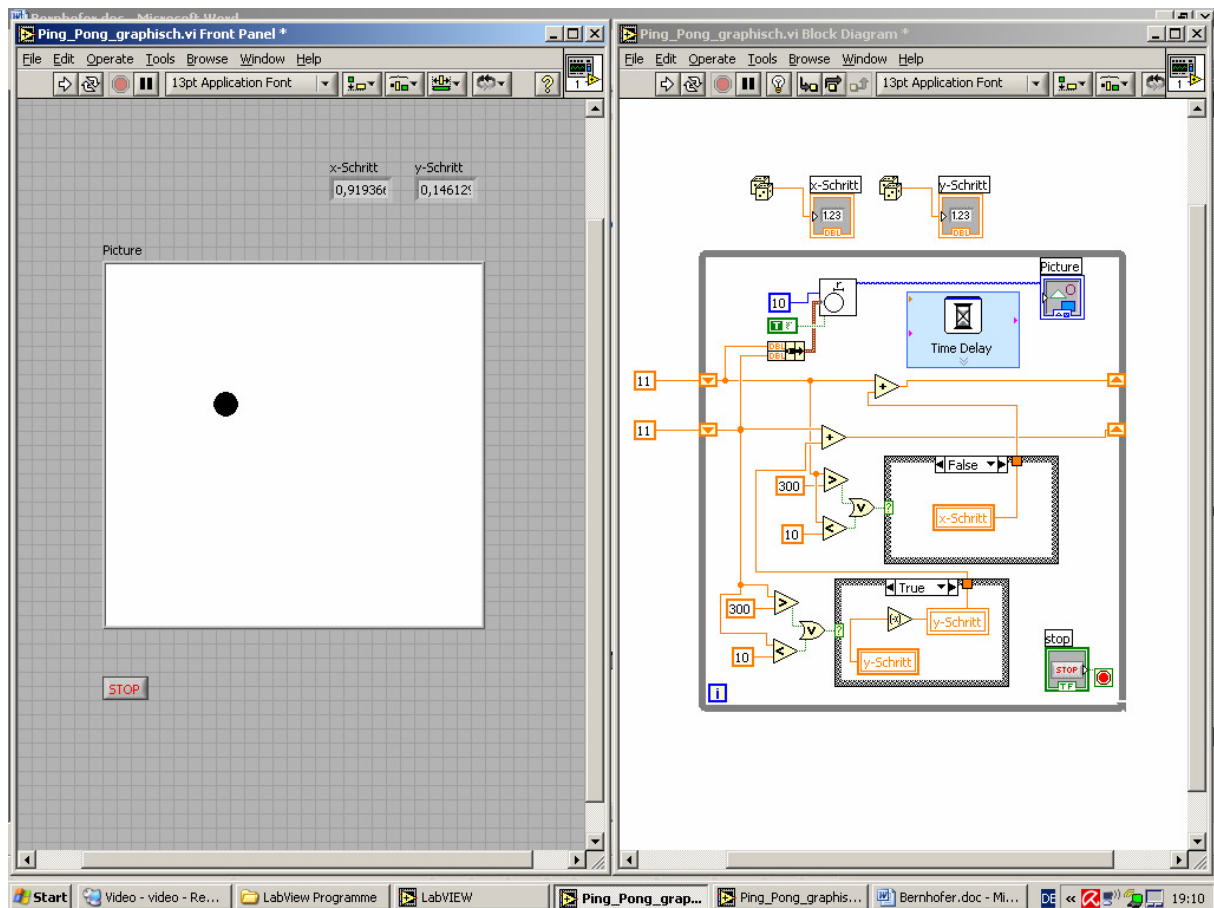
Anschließend wurden wieder Verfeinerungen an dem Grundprogramm durch die Schülerinnen und Schüler vorgenommen.





3.4.3 Reflexionsgesetz – Pingpong

Der Einfallswinkel ist gleich dem Ausfallswinkel jedes Lichtstrahles, der regulärer Reflexion unterliegt. Nachdem Einigkeit darüber erzielt wurde, dass auch Billardkugeln nach diesem Gesetz von der Bande abprallen, wurde eine Billardkugel-von-der-Bande-Abprallsimulation gemeinsam erarbeitet, wobei ich den Schülern bei der Umlegung ihrer Ideen in die Berechnung der Bewegung in einem kartesischem Koordinatensystem behilflich sein musste.

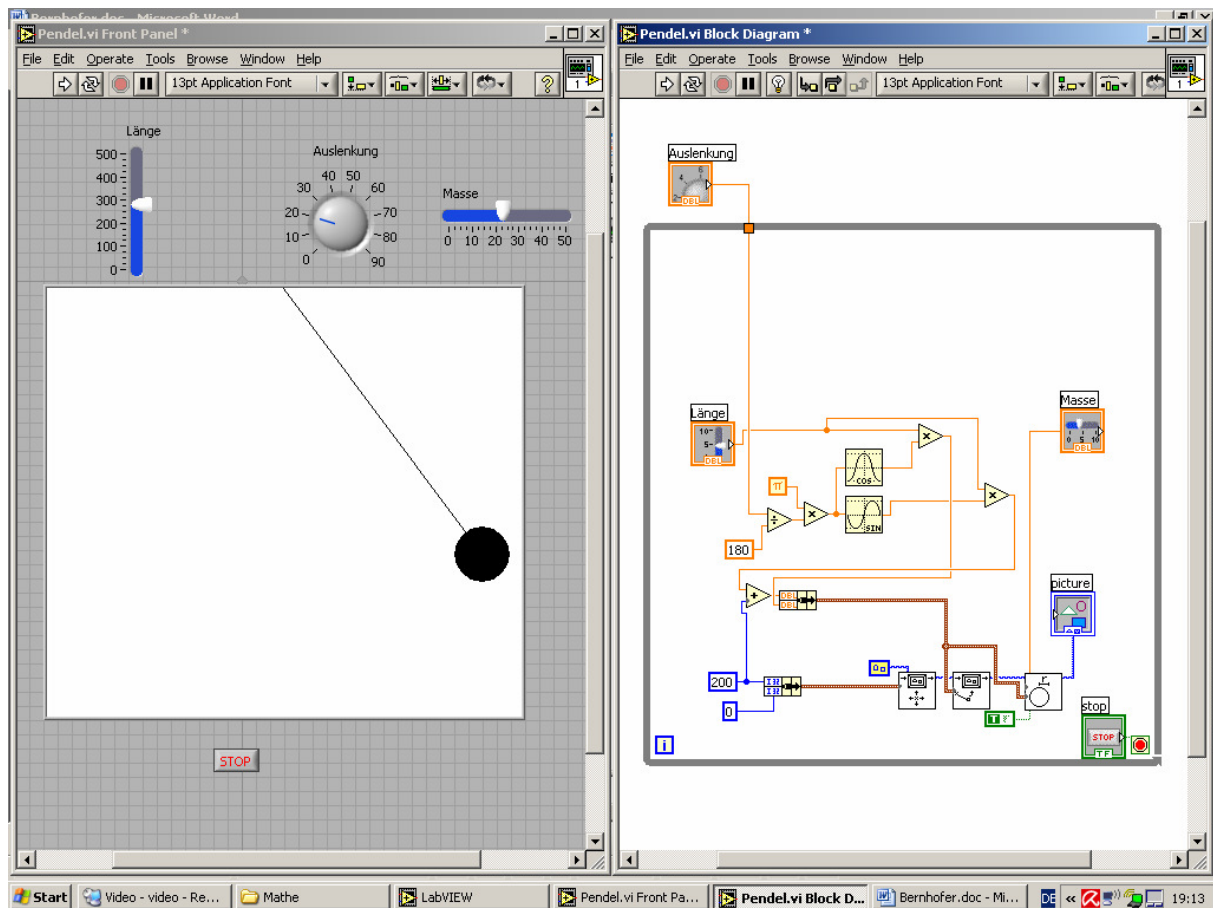


In diesem Beispiel wurde auch erstmalig von der Vollgraphikfähigkeit von LabVIEW Gebrauch gemacht, weil Kugel und Banden nicht wie bei den vorangegangenen Beispielen Graphikobjekte waren, sondern frei aufgebaut wurden.

3.4.4 Mathematisches Pendel

Nachdem die Vollgraphikfähigkeiten von LabVIEW von den Schülerinnen und Schülern sehr gut aufgenommen worden waren, wurde auf ihren Wunsch hin auch diese Simulation nicht mit Graphikobjekten aufgebaut, sondern frei gezeichnet.

Grundlage für die Berechnungen der Pendelbewegung war nicht die bekannte Formel des mathematischen Pendels, sondern die Betrachtungen über den Zusammenhang der rücktreibenden Kraft. Diese rücktreibende Kraft, verursacht durch die Erdanziehung, wird in zwei Komponenten zerlegt, wobei der radiale bzw. tangentielle Anteil durch sin- und cos Terme errechnet wurde.



Die graphische Darstellung wurde sehr einfach gehalten. Von den Schülerinnen und Schülern wurde trotzdem Wert darauf gelegt, dass das Kugelr, welches die Pendelmass symbolisiert, mit verändernder Masse seinen Durchmesser ändert.

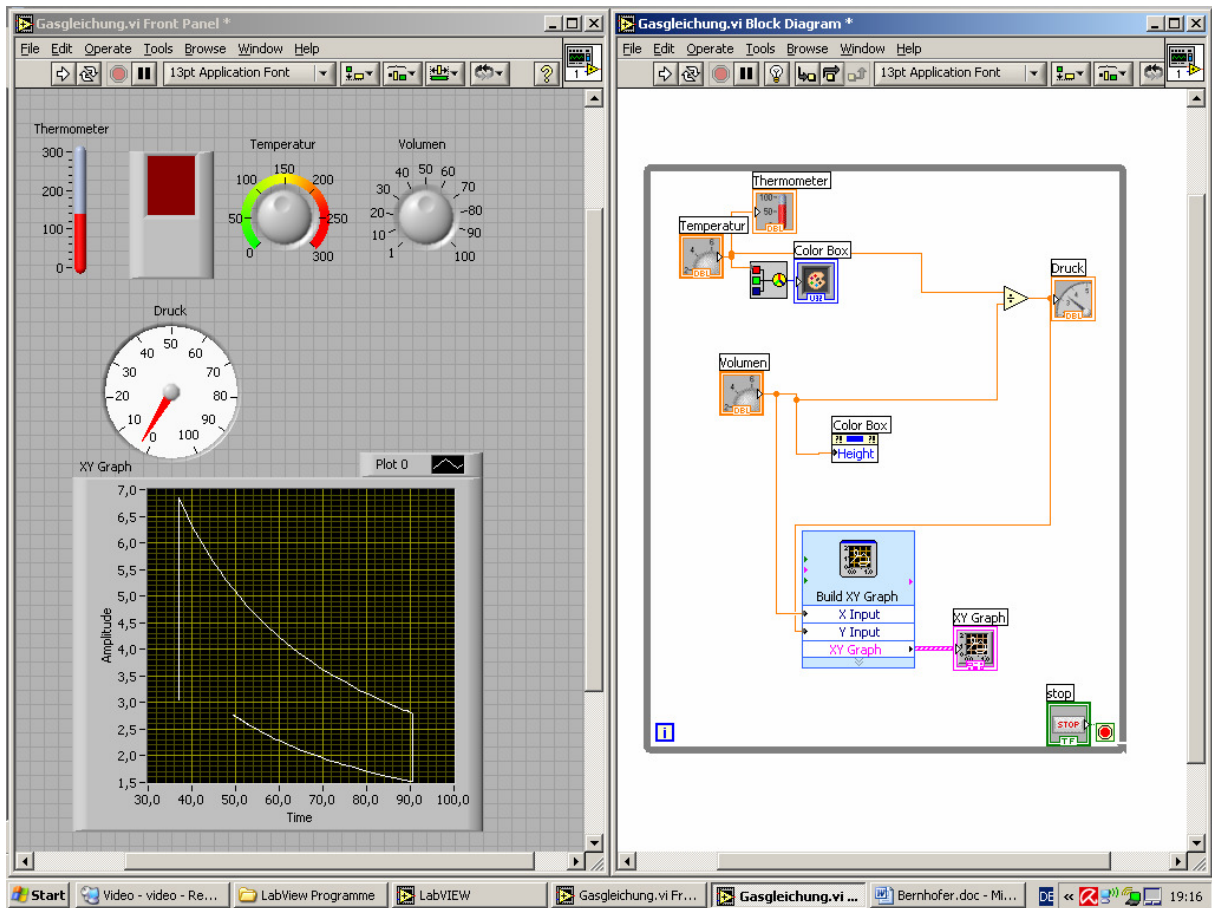
Diese zusätzliche Anforderung konnte von ihnen selbständig ausgeführt werden.

3.4.5 Gasgleichung einfach – manuell zu bedienen

Da unsere Schule eine HTL für Chemie ist, stellt die Gasgleichung und die zu ihr führenden Gesetze eine wichtige Grundlage für die Schülerinnen und Schüler dar, die sie in den Folgejahren ihrer weiteren Ausbildung immer wieder benötigen.

Aus diesem Grund wurde auf diese Problematik besonderes Augenmerk gerichtet.

In mehreren Schritten wurden die Schülerinnen und Schüler an die Programmierung der Simulation einer Wärmekraftmaschine herangeführt, wobei besonderer Wert darauf gelegt wurde, dass das Erarbeiten der Lösungen zur Gasgleichung möglichst selbständig erfolgte.

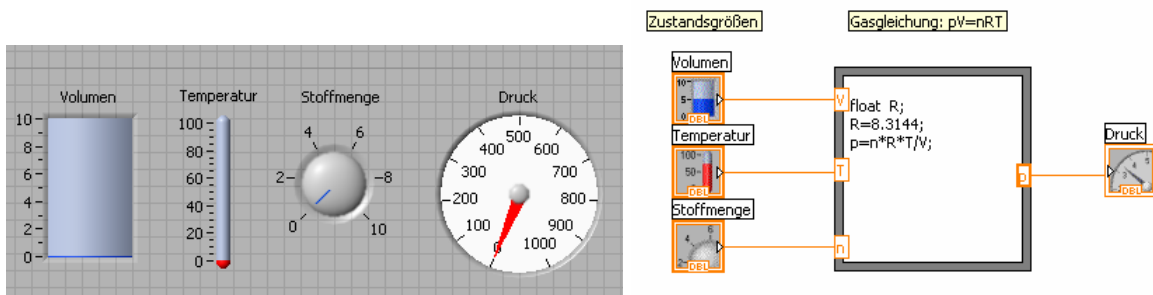


Variation mit Entstehungsgeschichte:

Gasgleichung 1:

Das erste Programm lief noch nicht alleine, der Benutzer musste das Programm zu jeder Berechnung erneut starten. Eine simulierte Lauffunktion konnte aber durch den „Dauerlauf-Button“ von LabVIEW erreicht werden, mit dessen Hilfe ein Programm in eine fortlaufende Exekution gezwungen werden kann.

Ausgaben von Volumen, Druck und Temperatur wurden dabei dem Selbstverständnis des Chemikers in den verwendeten Symbolen angepasst.

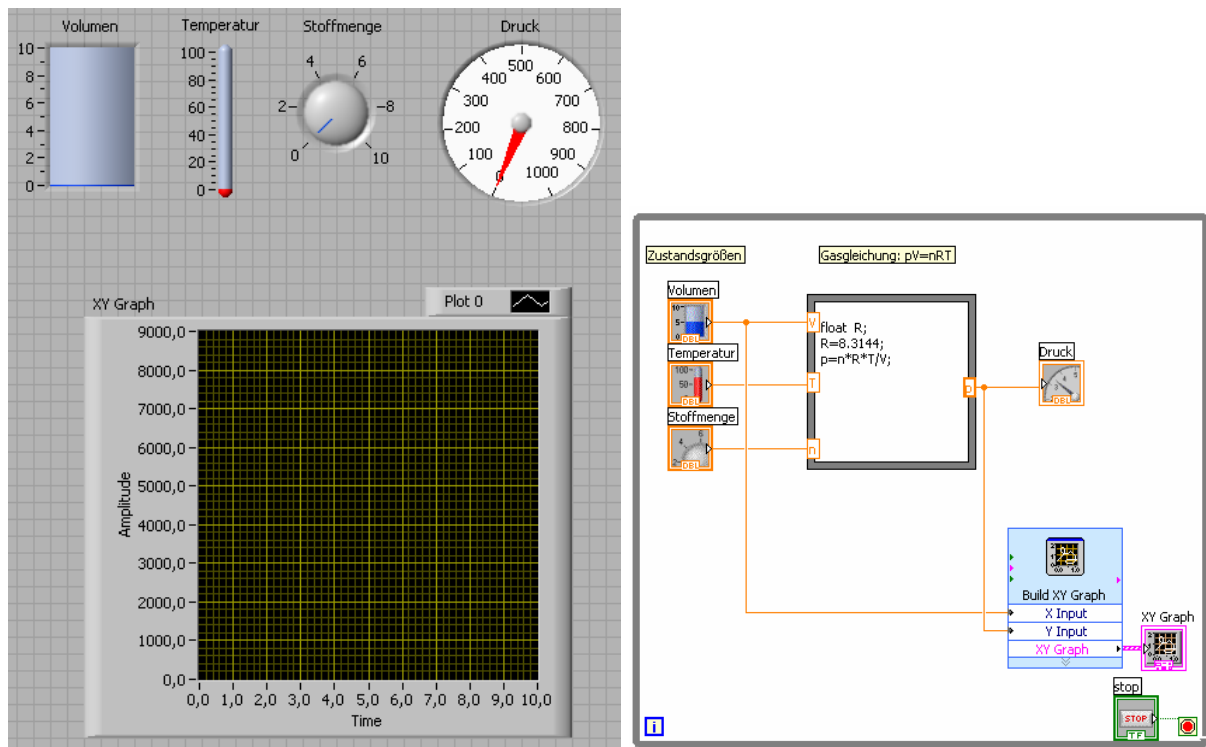


Gasgleichung 2:

Dieses Programm war schon selbständig lauffähig und bot die Möglichkeit ein p,V-Diagramm des jeweiligen Prozesses zu liefern. Änderungen der Zustandsgrößen mussten aber noch händisch vorgenommen werden.

Durch einfachste Änderungen des Programms konnten auch V,T, bzw. p,T-Diagramme erzeugt werden. Die Schülerinnen und Schüler waren angehalten diese Modifikationen selbständig durchzuführen.

Auch wurden Achsbeschriftung und Verwendung von Einheiten in ihren Verantwortungsbereich übergeben.



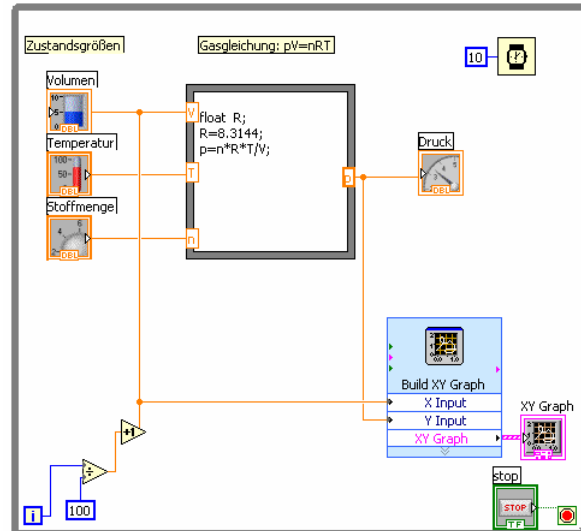
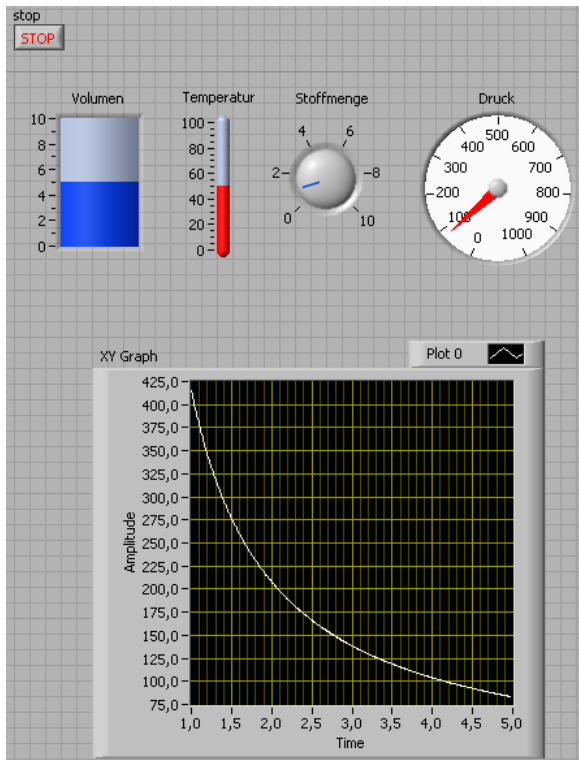
Gasgleichung 3:

In diesem Schritt der Entwicklung erfolgte ein Umdenkprozess, der zu einer vollständigen Automatisierung der Simulation eines Versuchsablaufs führte.

Der Schleifenzähler der „while-Schleife“ wurde verwendet um den kontinuierlichen Veränderungsprozess einer der Zustandsgrößen vorzugeben.

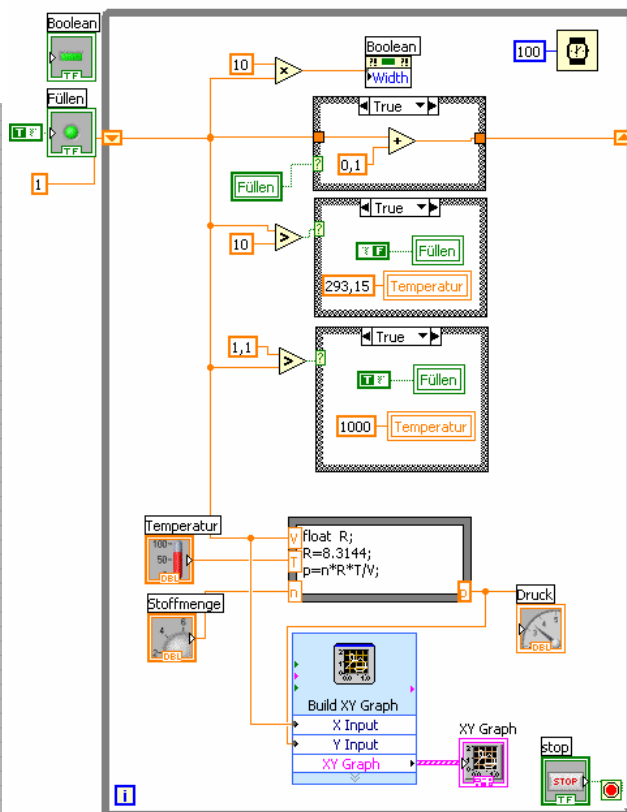
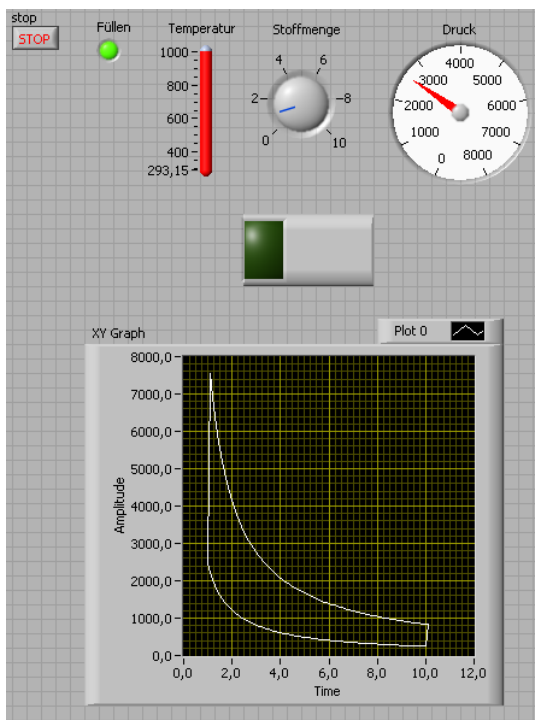
Da der Schleifenzähler nur ganzzahlig inkrementiert, was zu große Sprünge im Diagramm zur Folge hätte, musste die Erhöhung des Zählers durch Division verringert werden.

Schließlich konnte der hyperbolische Verlauf des p,V-Diagramms wiedergegeben werden.



Motor:

Obschon der Sprung zur nächsten Evolutionsstufe riesig erscheint, ist dem bisherigen Programm nicht viel hinzuzufügen um die Simulation einer Wärmekraftmaschine zu erhalten.



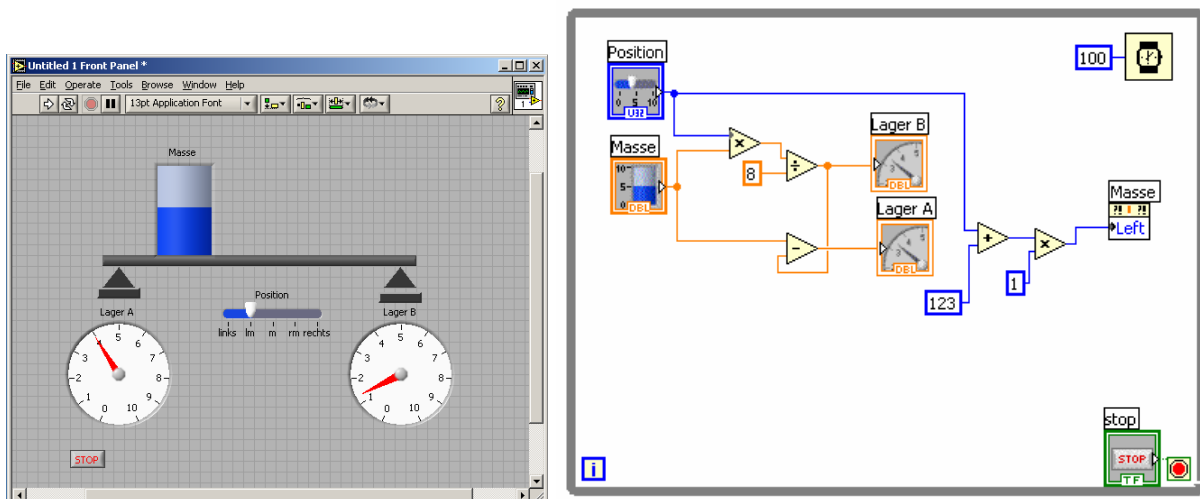
Prinzipiell musste nur ein Weg gefunden werden, dass der immer wiederkehrende Prozess des Füllens, Komprimierens, Zündens, Expandierens und Auspuffens programmtechnisch umsetzbar wurde.

Der Weg dazu wurde von mir vorgezeichnet indem ich die programmtechnische Umsetzung dieses Schleifenprozesses vorzeigte, von den Schülerinnen und Schülern wurde anschließend die Detailausarbeitung umgesetzt.

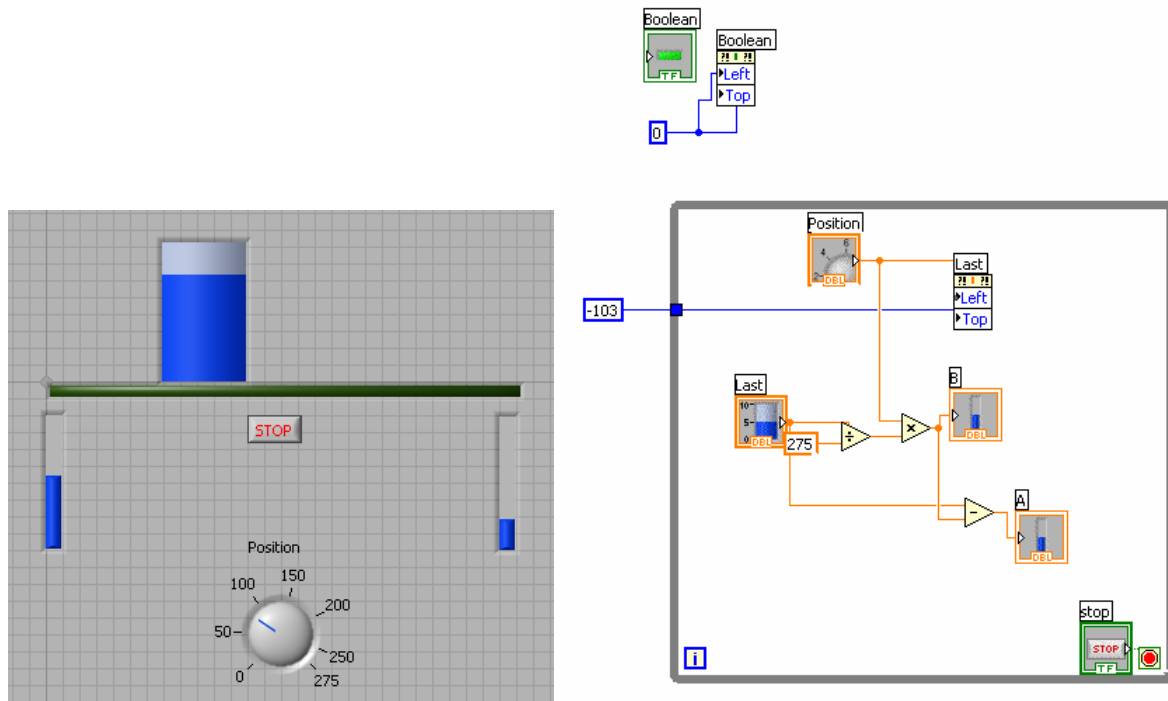
3.4.6 Hebelgesetz

Der Ansatzpunkt für diese Simulation war, dass sich das Gewicht eines auf zwei Lagern ruhenden Balkens indirekt proportional zum Abstand des Gewichtes vom Lager aufteilt.

Diese Aufteilung erfolgt damit völlig analog zum Hebelgesetz und wurde programmtechnisch so umgesetzt, dass ein Tank-Symbol, das die Masse durch seinen Füllgrad darstellt, auf einem Balken während des Programmlaufs verschiebbar ist und die Lastverteilung auf die Lager während der Verschiebung angezeigt wird.



Variation:

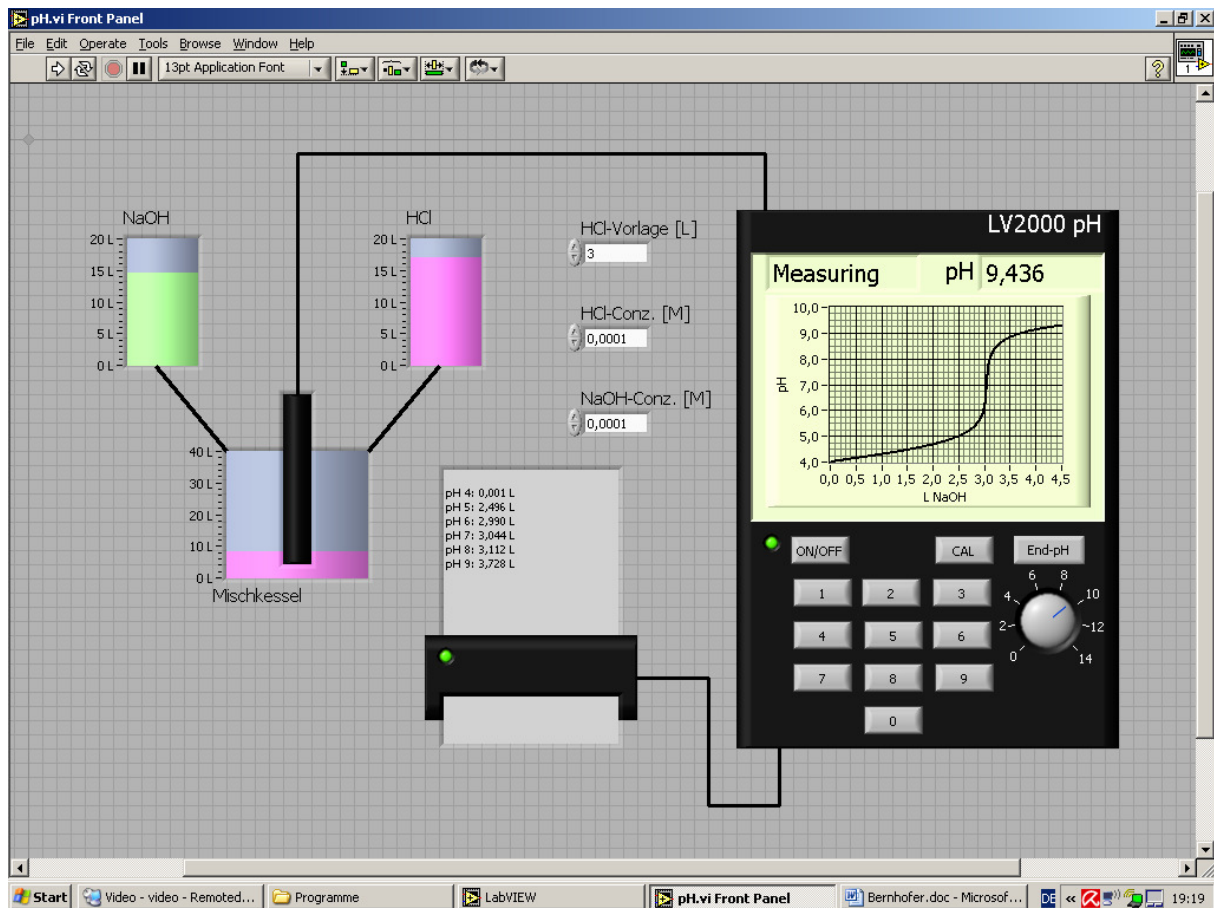


3.4.7 pH-Messung

Eine der eindrucksvollsten und programmtechnisch schwierigsten Simulationen wurde von einem Schüler in seiner Freizeit programmiert und der Klasse zur Verfügung gestellt. Dabei handelt es sich um eine liebevoll gestaltete Darstellung einer Säure-Basen-Titration, die sich durch eine spezielle Lösung des Problems auszeichnet, dass eine Berechnung von $\log(0)$ am Umschlagspunkt geschickt umgangen werden konnte.

Nachdem einfachere Beispiele im Unterricht erarbeitet wurden und die Berechnung des pH-Werts erklärt war, setzten sich die Schüler mit einfachen Titrationsprogrammen auseinander. Fast alle Ansätze führten zu funktionierenden Simulationen, die bis zum Umschlagspunkt exakte Ergebnisse lieferten.

Die Problematik des $\log(0)$ konnte aber von der Klasse nicht umgangen werden.



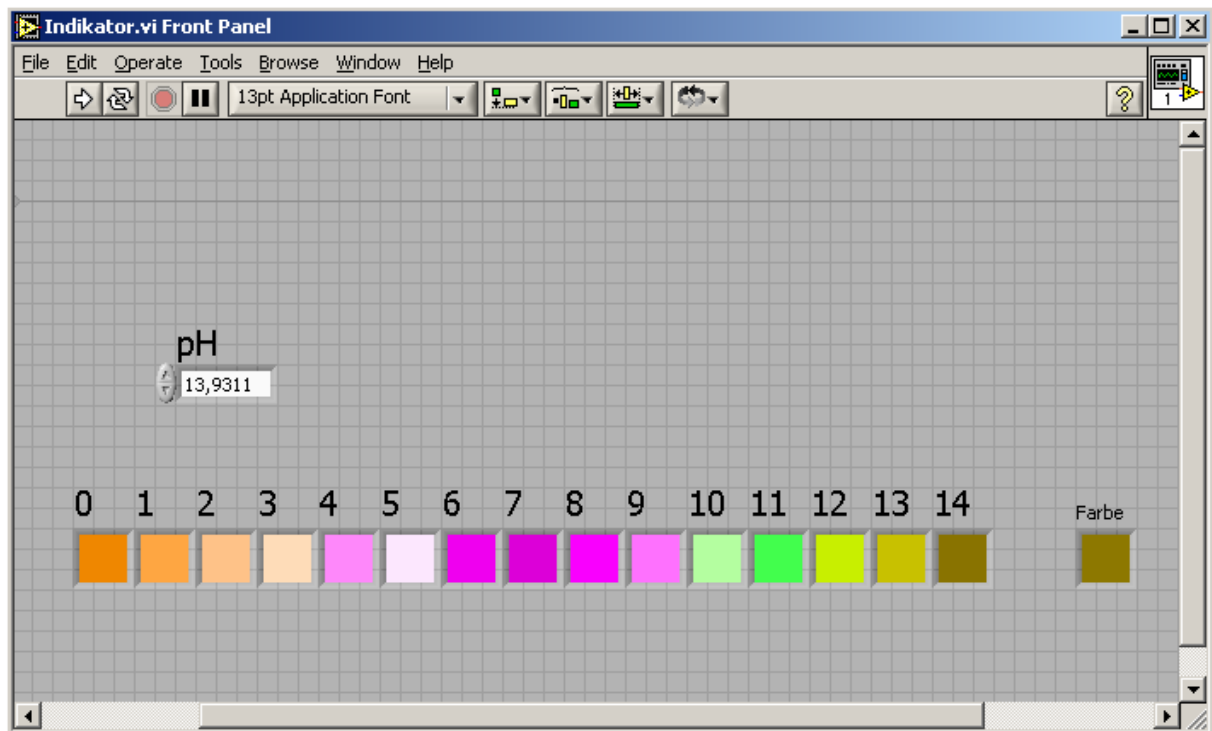
Der Lösungsweg dieses Beispiels besteht darin, nach Annäherung an eine Gewisse Umgebung von Null die Berechnung auf den pOH-Wert umzustellen und den Rest der Kurve knapp oberhalb von Null fortzusetzen.

3.4.8 Indikatorfarben

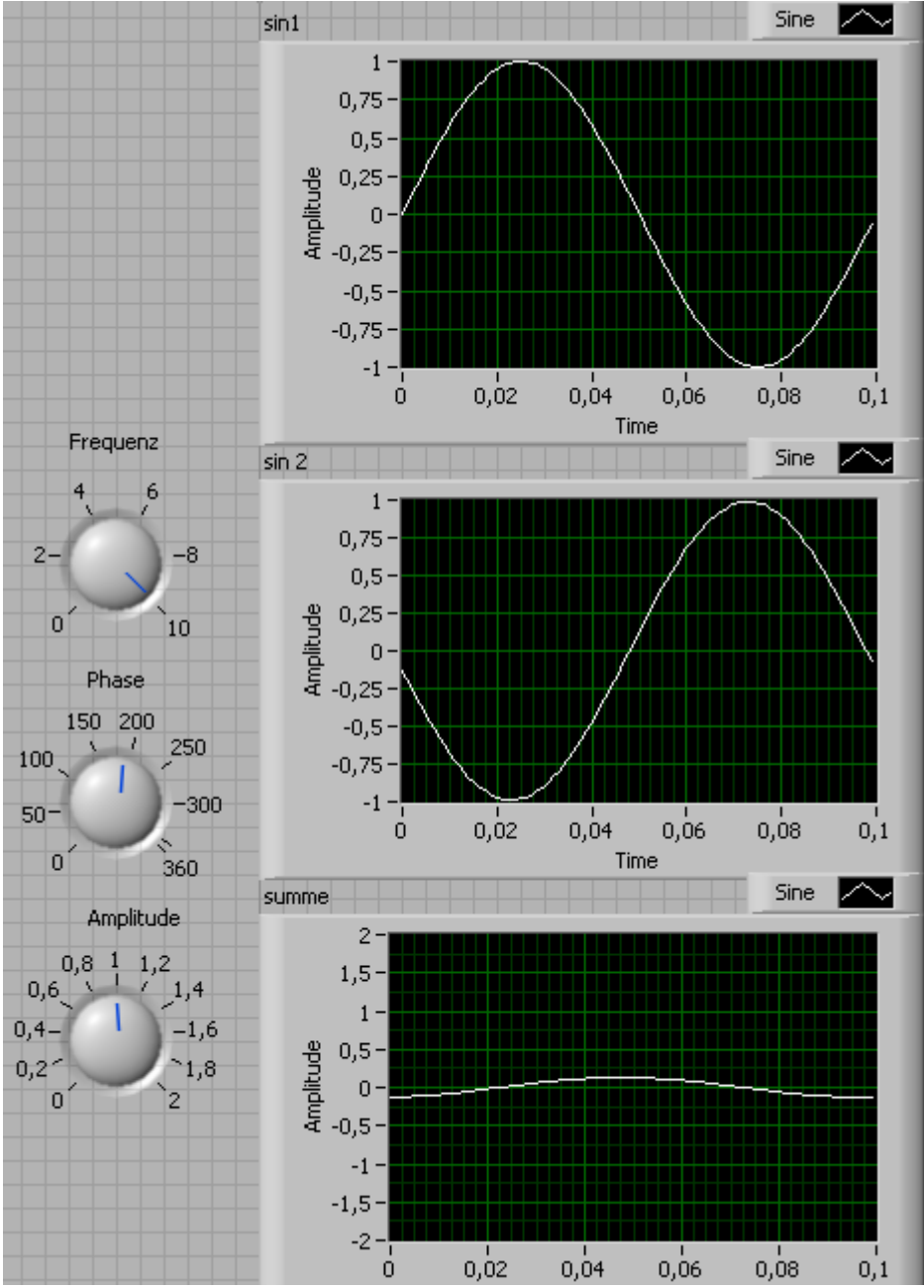
Da ein möglicherweise zugesetzter Indikator während der Titration seine Farbe in Abhängigkeit vom pH-Wert der Lösung ändert, wurde von den Schülern sehr bald der Vorschlag unterbreitet, diese Farbänderung ebenfalls in die Simulation zu übernehmen. Da prinzipiell alle Graphikobjekte von LabVIEW in allen Eigenschaften skalierbar sind, war die Änderung der Farbe des Tanksymbols, das das Titrationsgefäß darstellte, kein Problem.

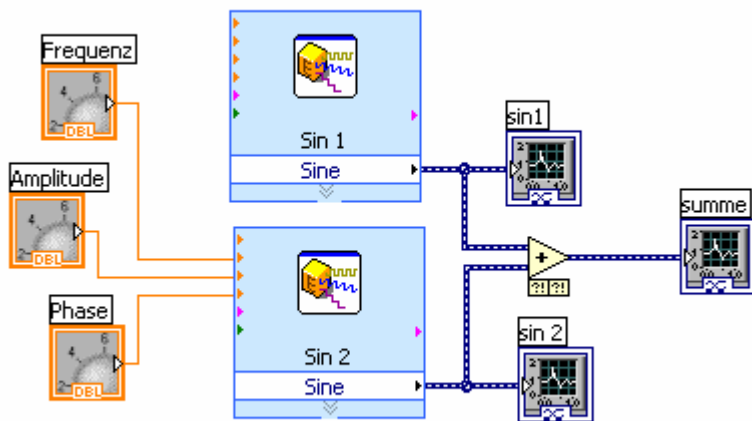
Schwieriger war nur, die Farbänderung an den realen Farbumschlag eines existierenden Indikators anzupassen.

Diese Aufgabe wurde von dem vorliegenden Programm gelöst, wobei der Geduld der Schüler zu verdanken ist, dass die Farbskala, die durch das Programm errechnet wurde fast exakt dem Farbverlauf eines Universalindikators entspricht.



3.4.9 Interferenz





Dieses letzte Beispiel stellt ein sehr anspruchsvolles Teilgebiet der Physik sehr einfach aber eindrucksvoll dar. Durch die besondere Datenstrukturverwaltung von LabVIEW ist es möglich, Daten aller Art einfach mathematisch zu verknüpfen.

Dieses Prinzip wird bei diesem Beispiel genutzt und führt dazu, dass das Programm, das zwei sin-Wellen interferieren lässt einfachste Struktur hat, obwohl man damit alle möglichen Überlagerungsszenarien durchspielen kann.

Vollständige Auslöschung sowie Maximalverstärkung sind nur zwei Spezialfälle der möglichen Simulationen.

Gerade bei diesem Beispiel zeigt sich die Mächtigkeit von LabVIEW.

4 ERGEBNISSE

4.1 Unterrichtserfahrungen mit den einzelnen Übungen und Simulationen

Die in den Grundkursen angegebenen Zeiten, die zum Erarbeiten der Inhalte nötig sind, sind leicht einzuhalten.

Das erste Programm

Das erste Programm in seiner gesamten Ausführlichkeit durchzunehmen, wäre nicht wirklich nötig gewesen, da die Schülerinnen und Schüler durch selbständiges Probieren sehr schnell zu viel besseren Ergebnissen kamen, als durch diese Übung gefordert.

Beispiel - Lamperl ein/aus

Die Zeitsteuerung dieses Programms stellte sich als komplex dar und wurde von den Schülerinnen und Schülern nicht unmittelbar verstanden. Nach intensiver Auseinandersetzung mit der Systemuhr, die jeder Rechner hardwaremäßig hat, konnte der Algorithmus nachvollzogen werden.

Fortführende Erklärungen der Programmier Techniken

Dieser Teil der Einführung stellte sich als wichtig heraus. Die Schülerinnen und Schüler machten während dieses Abschnitts die größten Fortschritte, was ihre Programmierfähigkeiten betrifft.

Selbständiges Ausarbeiten von einfachen Aufgaben

Das selbständige Ausarbeiten von einfachen Aufgaben stellte für die Schülerinnen und Schüler kein Problem dar.

Geradlinige gleichförmige Bewegung

Insgesamt stellte diese Aufgabe keine besondere Herausforderung dar und war schnell durchgearbeitet.

Geradlinige gleichmäßig beschleunigte Bewegung

Wie an der unten angeführten Lösung ersichtlich ist, sind die Schülerinnen und Schüler an die Lösung dieser Problemstellung schon viel ernsthafter herangegangen. Das fertige Programm wurde anschließend genutzt um Experimente zu simulieren, wobei die verschiedenen Parameter von den Schülerinnen und Schülern spielerisch verändert und alle möglichen Szenarien durchprobiert wurden.

Reflexionsgesetz – Pingpong

Die in diesem Beispiel eingeführten Graphikmöglichkeiten in der LabVIEW-Programmierung faszinierten.

Mathematisches Pendel

Die Graphikmöglichkeiten wurden von den Schülerinnen und Schülern in diesem Beispiel selbständig erarbeitet.

Gasgleichung

Diese Beispielgruppe beschäftigte die Klasse am längsten. Wichtig wurde dabei, den physikalischen Hintergrund mit der Simulation von Schritt zu Schritt aufwendiger zu verbinden. Das bereitete vielen Schülerinnen und Schülern Schwierigkeiten, die aber durch gegenseitiges Erklären bei den Gruppenarbeiten überwunden werden konnten.

Hebelgesetz

Dieses Beispiel war von der Programmierung sehr einfach und wurde von allen Gruppen in Rekordzeit von nicht einmal einer Stunde gelöst.

pH-Messung

Die Königin unter den Anwendungen konnte von den Schülerinnen und Schülern nicht selbständig entwickelt werden, weil sich bei der pH-Berechnung immer wieder eine Division durch Null ergab, welche von den Ausführenden nicht umgangen werden konnte. Nach gemeinsamer Entwicklung der Lösung wurde von den Gruppen noch die graphische Aufbereitung übernommen.

Indikatorfarben

Dieser Programmteil wurde einfach durch Ausprobieren und Vergleichen mit der Farbtafel des Universalindikatorpapiers gelöst.

Interferenz

Dieses Beispiel, das gleichzeitig mein Lieblingsbeispiel ist, weil ich durch dieses Programm auf die Idee der Einführung von LabVIEW gekommen bin, wurde von mir als Vorführprogramm entwickelt.

4.2 Verständnis, pädagogischer und didaktischer Nutzen

4.2.1 Verständnis

Es hat sich gezeigt, dass nach sehr kurzer Einschulungsphase die graphische Entwicklungsumgebung von LabVIEW verstanden wurde.

Schülerinnen und Schüler hatten dabei nicht nur von der Möglichkeit profitiert, programmieren zu lernen, sondern auch Spaß beim Austesten der Programme und dem individuellen Gestalten der Bedienoberflächen.

Der Wissenszuwachs der Schülerinnen und Schüler wurde durch den Einsatz des Computers im Unterricht zwar vielleicht nicht vergrößert, das Verständnis für Zusammenhänge wurde aber sicher gefördert.

4.2.2 Pädagogischer Nutzen

Den pädagogischen Nutzen dieser Arbeit sehe ich darin, dass die Beschäftigung mit den physikalischen Hintergründen eines Phänomens oder eines Gesetzes viel intensiver stattfand, sobald die Möglichkeit gegeben war, sich mit der Ausarbeitung einer Simulation zu beschäftigen.

4.2.3 Didaktischer Nutzen

Allgemein ist anzumerken, dass der strategisch durchdachte Einsatz von LabVIEW im Physikunterricht von den Schülerinnen und Schülern sehr positiv und motivierend aufgenommen wurde.

Die Erfahrung hat gezeigt, dass auch Schülerinnen und Schüler mit wenig oder ohne Programmiererfahrung den Einstieg in die Programmentwicklung rasch schafften, da LabVIEW mit der grafischen Syntax eine gestalterisch intuitive Annäherung zu Problemlösungen begünstigte und dadurch motivierend wirkte.

Dieser Tatsache entsprechend könnte der Einsatz von LabVIEW auch als Einführungssprache im Informatikunterricht durchaus positive Wirkung zeigen.

Das selbständige Erarbeiten von Simulationen zu physikalischen Problemstellungen war sehr rasch möglich und konnte nach kurzer Zeit fast ohne Mithilfe des Lehrers durchgeführt werden.

Die nach der Entwicklung der eigenen Programme notwendige Debugging-Phase stellte den Einstieg in eine vertiefte Auseinandersetzung mit der Materie dar, denn wenn Programme falsche oder keine Ergebnisse lieferten, war man gezwungen auf der Suche nach den Fehlern die Hintergründe zu erfragen.

Auch setzten sich die Schülerinnen und Schüler erstmalig damit auseinander, welcher Wertebereich für die Ergebnisse ihrer Berechnungen in Frage kam, welche Berechnungen zu unrealistischen Aussagen führten, warum Simulationen Unendlich oder Null als Resultat lieferten und, als vielleicht wichtigster Punkt, wie groß die Relevanz ihrer Programme im Vergleich zum realen Verhalten der Dinge waren.

Diese Frage nach dem Zusammenhang zwischen der theoretischen Berechnung und der Realität, wurde von den Schülern ebenfalls selbständig erarbeitet und Antworten in einfachen Experimenten gesucht.

4.3 Freude auf dem Weg

Nach der relativ kurzen Einarbeitungsphase in die Entwicklungsumgebung von LabVIEW wurde jede weitere Unterrichtseinheit, die im EDV Saal stattfand von den Schülerinnen und Schülern durchwegs positiv aufgenommen. Die Möglichkeit erlerntes Wissen am Computer praktisch umzusetzen wurde von den Schülerinnen und Schülern geschätzt und die Zusammenarbeit in Gruppen sorgt dabei für Abwechslung. Gut gefiel allen die Möglichkeit gestalterisch tätig zu werden, da ohne große Programmierfertigkeiten Benutzeroberflächen und Programmabläufe gestaltet werden konnten. Dadurch konnten sich alle einbringen und wurden nicht in passive Rollen gedrängt, wenn sie beim Umgang mit dem Computer nicht so geschickt waren.

4.4 Beurteilung durch die Schülerinnen und Schüler

Die meisten der Schülerinnen und Schüler, die an diesem Projekt teilgenommen haben, haben mittlerweile begonnen LabVIEW auch zur Lösung von Aufgabestellungen in anderen Gegenständen zu nutzen, weil sie den Vorteil für sich erkannt haben.

Speziell sei dabei auf Mathematik hingewiesen, wo sich die Probleme der Optimierung, der Kurvendiskussion, der Integral- und Differentialrechnung und der Wahrscheinlichkeitsrechnung einfach analysieren lassen.

Das zeigt deutlich auf, dass die Verwendung von LabVIEW zur Problemlösung von den Schülerinnen und Schülern positiv beurteilt wurde.

Ich glaube, durch dieses Projekt gezeigt zu haben, dass es sich im Physikunterricht lohnt, neue Wege zu gehen und die Schüler an moderne Technologien heranzuführen, auch wenn der Weg noch lang sein mag.

Wird LabVIEW den Schülern zur eigenen Entwicklung von Anwendungen zur Verfügung gestellt, erhalten sie nicht nur einen Einblick in die Konzepte und Entwicklungsstrategien der LabVIEW-Programmierung sondern entwickeln auch ein tieferes Verständnis für die naturwissenschaftlichen Hintergründe.

5 DISKUSSION UND AUSBLICK

5.1 Skepsis der Schüler

Ganz zu Beginn dieser Arbeit stand die erste Physikstunde, in der ich mit der Absicht die Klasse betrat, eine kleine physikalische Problemstellung direkt am Laptop in der Klasse zu programmieren. Ein Beamer bot den Schülern Einblick in die Entstehung des Programms.

Die Reaktion der Schüler war interessant, denn sofort nachdem ich mit Laptop und Beamer erschienen war, schallte es mir aus vielen Mündern entgegen: „Juhu – Wir schauen einen Film an!“.

Damit hatte ich nicht gerechnet. Offensichtlich wurde der Laptop von den Schülerinnen und Schülern nicht als Hilfsmittel für die Lösung wissenschaftlicher Problemstellungen angesehen.

Nachdem ich erklärt hatte, dass ich ein Programm schreiben wollte, mit dem man die geradlinige Bewegung berechnen konnte, machten mich die Schülerinnen und Schüler drauf aufmerksam, dass sie ihre Taschenrechner nicht mitgebracht hätten.

Ich erklärte, dass ich das Programm auf meinem Laptop entwickeln würde und startete nach einer kurzen Aufbauphase.

Nach dem Start von LabVIEW und dem Setzen weniger Icons und Verbinder wurde ich mit dem Hinweis unterbrochen, dass das was ich da mache nicht programmieren sei. Schließlich hätte der eine oder andere einen Bruder oder eine Schwester, die programmieren gelernt hätte, und das, was ich da auf meinen Bildschirm zauberte, schaute so ganz anders aus als die endlosen Textzeilen, die sie gewohnt waren.

Dieses falsche Bild, das bei der Einführung von LabVIEW schon in der ersten Stunde entstand, zog sich durch das gesamte Projekt und es gelang mir nicht, gegen die vorgefassten Meinungen wie eine Programmiersprache zu sein und wie ein Programm auszusehen hat anzukommen.

Wie sich später bei den Befragungen zeigte, wurde und wird LabVIEW nicht als vollwertige Programmiersprache anerkannt, obschon sich die Schülerinnen und Schüler ihrer mittlerweile auch in anderen Bereichen mit großer Freude bedienen.

5.2 Die Angst des Lehrers

Die von mir verwendete Technologie ist weder neu noch kompliziert. Seit über 25 Jahren wird LabVIEW eingesetzt und weiterentwickelt. Niemand sollte davor zurückschrecken LabVIEW in seinem Unterricht zu verwenden, es sollte allerdings auch nicht der Fehler begangen werden, den Lernaufwand, den der Lehrer erbringen muss, zu unterschätzen.

Ich unterrichte LabVIEW als Programmierumgebung seit 6 Jahren und bin nach wie vor weit davon entfernt, mich als Pro bezeichnen zu können. Das stellt aber kein Problem dar, da die meisten Programmpakete, die LabVIEW zur Verfügung stellt, beim Einsatz im Unterricht gar nicht benötigt werden. Die Grundfunktionen dieser Entwicklungsumgebung sind von jemandem, der zumindest Grundkenntnisse vom Programmieren hat, in kürzester Zeit erlernbar, aber auch mit Kenntnissen der Soft-

warentwicklung völlig unbelastete sollten mit absehbarem Einsatz verwertbare Ergebnisse erzielen können.

Auch kann man sich bei der Einführung dieser Entwicklungsumgebung ein interessantes Phänomen zu nutze machen: Wenn es um LabVIEW geht, laufen einzelne Schülerinnen und Schüler zur Höchstform auf und der Lehrer kann darauf vertrauen, wenn er etwas nicht weiß, so braucht er in den meisten Fällen nur seine Schülerinnen und Schüler fragen, die finden sicher eine Lösung, sei es durch Ausprobieren oder surfen im Internet.

5.3 Diskussion

- LabVIEW ist grundsätzlich ohne Programmierkenntnisse erlernbar.
- Es hat sich gezeigt, dass besonders junge Menschen die grafische Syntax rasch lernen.
- Diese Syntax und die interaktive Ein- und Ausgabe von Variablen fördert den intuitiv spielerischen Umgang mit der Programmierkunst.
- Auf Grund der grafischen Struktur ist LabVIEW auch ein vielseitiges Präsentationspaket.
- Die Datenausgabe kann in der gewünschten Form gestaltet werden.
- Datenschreiber, Grafiken und benutzerdefinierte Darstellungen sind nur ein kleiner Teil der Optionen.
- Die Messdatenerfassung, die Analyse- und Darstellungswerkzeuge machen LabVIEW zu einem mächtigen Entwicklungssystem.
- Alle Problemlösungsmöglichkeiten einer konventionellen Programmiersprache sind als virtuelle Instrumente möglich.

Trotz der Eigenart von LabVIEW, dass, im Unterschied zu konventionellen Programmiersprachen wie „C“ oder „Pascal“, durch grafisches Zeichnen von Blockdiagrammen programmiert wird, ist LabVIEW eine vollumfängliche Programmiersprache und somit ein Teilbereich der Informatik.

Dem Argument, LabVIEW sei durch die grafische Syntax eine zu spezielle Entwicklungsumgebung und daher ungeeignet, können folgende Aspekte entgegengesetzt werden:

1. LabVIEW ist eine vollumfängliche Programmiersprache, alle syntaktischen Strukturen konventioneller Programmiersprachen sind enthalten und können dementsprechend schrittweise behandelt werden.
2. Die in LabVIEW voll integrierte, rasch zu realisierende Gestaltung einer Benutzeroberfläche mit entsprechenden grafischen Ein- und Ausgabemöglichkeiten erlaubt eine sofortige Einsicht von Ein- und Ausgabeparametern, was das Experimentieren mit Programmen oder Programmmodulen zulässt, eine kreative Programmierung fördert und die rasche Überprüfung des Funktionierens eines Programms zulässt.
3. Ein Programmierer wird von syntaktischen Kleinarbeiten und Details entlastet und kann sich auf die Gesamtstruktur und somit auf den wesentlichen Aspekt seines Programms konzentrieren. Die grafische Syntax und die Möglichkeit zur einfachen Ges-

taltung attraktiver Benutzeroberflächen erleichtern auch künstlerisch begabten Leuten den Zugang zur Softwareentwicklung.

4. Da LabVIEW als Programmiersprache für Messdatenerfassung den unkomplizierten Zugriff auf eine breites Spektrum von Messdaten ermöglicht, sind interessante Applikationen in diesem Bereich auch für Anfänger realisierbar, was wiederum den „lustvoll spielerischen“ Umgang beim Erlernen der Programmierkunst fördert.

5. Die zunehmende Popularität von LabVIEW unter professionellen Programmierern in den wichtigen Bereichen der universitären und industriellen Forschung und Entwicklung, der Qualitätskontrolle von Industrieprodukten und der Industrieautomation hat LabVIEW in den letzten Jahren zu einem Industriestandard gemacht. Diese Entwicklung wird sich in den kommenden Jahren fortsetzen. Die große Nachfrage nach qualifizierten LabVIEW-Programmierern in der Industrie ist ein guter Beweis für diese Dynamik. Zunehmend integrieren auch konventionelle Programmientwicklungsumgebungen grafische Programmierhilfen und verlangen dementsprechende Fähigkeiten in diesem Bereich. Die Tatsache, dass mit LabVIEW im Vergleich zu konventionellen Sprachen qualitativ hoch stehende und sehr stabile Industrieapplikationen in viel kürzerer Zeit entwickelt werden können, hat LabVIEW in eine starke Position gebracht.

5.4 Ausblick

Aufgrund der positiven Erfahrungen die ich im Austausch mit Kollegen gemacht habe, wäre eine Weiterentwicklung des Einsatzes von LabVIEW in andere Unterrichtsbereiche wünschenswert. Ich könnte mir z.B. eine Verwendung von LabVIEW im Informatikunterricht vorstellen.

Programmieren wird aufgrund der Komplexität moderner Entwicklungsumgebungen wie Visual Studio oder Java basierender IDEs in den meisten Fällen nicht mehr unterrichtet. Kollegen, die aber eine Einführung in die Grundzüge des Programmierens nach wie vor für sinnvoll halten, könnten mit LabVIEW ein Werkzeug an die Hand bekommen, das den Schülern viel von ihrer Hemmschwelle nimmt.

Auch einer Verwendung im Mathematikunterricht stehe ich positiv gegenüber. Ohne lange an Ausgabemöglichkeiten feilen zu müssen, können Funktionen dargestellt und analysiert werden, iterative Algorithmen bearbeitet und numerische Verfahren verstanden werden.

Der Computer wird verstärkt Einzug in das Klassenzimmer halten – und die Lehrer müssen diese Entwicklung nutzen und die zusätzlichen Möglichkeiten richtig einsetzen.

Durch die Verwendung der DAQ (Data Acquisition) Kästchen, die einen einfachen Anschluss von Sensoren ermöglichen, waren Kraftmessungen, Zeitmessungen und Temperaturmessungen direkt am Computer und damit von jedem Schüler selbst durchführbar. Dies führte zu einem spielerischen Umgang mit wissenschaftlichen Fragestellungen und damit zu einer Intensität der Auseinandersetzung, die sonst nicht zu erreichen gewesen wäre.

6 LITERATUR

[1]

Teaching Modern Data Acquisition Systems with a Departmental Requirement for Student Laptop Ownership

Stephen T. McClain; The University of Alabama at Birmingham; Department of Mechanical Engineering; BEC 358B, 1530 3rd Ave S; Birmingham, AL 35294-4461; smcclain@uab.edu

Bruce Cain; Mississippi State University; Department of Mechanical Engineering; P.O. Box ME; Mississippi State, MS 39762; cain@me.msstate.edu

[2]

VIRTUAL MEASUREMENT TECHNOLOGY IN THE EDUCATION OF PHYSICISTS AND COMMUNICATION ENGINEERS

ZOLTÁN GINGL; Department of Experimental Physics, University of Szeged; Dóm tér 9., H-6720 Szeged, Hungary; gingl@physx.u-szeged.hu

ZOLTÁN KÁNTOR; Department of Optics and Quantum Electronics, University of Szeged; Dóm tér 9., H-6720 Szeged, Hungary

[3]

LabVIEW - Eine industrielle Standardsoftware für die Schule genutzt

W. Bube, N. Fischer; Gymnasium Puchheim, Bgm.-Ertl-Str.11, 82178 Puchheim/München

[4]

LabVIEW™ - eine grafische Programmiersprache geeignet für den Unterricht

Urs Lauterburg; Physikalisches Institut der Universität Bern; Sidlerstr. 5; 3012 BERN

6.1 Weiterführende Literatur

Eine aktualisierte Bücherliste findet man unter:

<http://www.ni.com/devzone/reference/books/>

LabVIEW Student Edition; Autorin: Lisa K. Wells; Verlag: Prentice Hall; Mac und PC Version: LVSE6i ISBN: 0-13-032550-3

LabVIEW Graphical Programming, Practical Applications in Instrumentation and Control; Autor: Gary W. Johnson (LabVIEW Programmierer seit V 1.2); Verlag: McGraw-Hill, Inc.; ISBN 0-07-032915-X

LabVIEW Lernhandbuch; Autoren: Lisa K. Wells & Jeffrey Travis; Verlag: Prentice Hall PTR; ISBN 0-13-268194-3

Das LabVIEW-Buch; Autoren: Lisa K. Wells & Jeffrey Travis; Verlag: Prentice Hall PTR; ISBN 3-8272-9540-8

Joint Time Frequency Analysis, Methods and Distribution, Cohen's Class usw.; Autoren: Shie Qian & Dapang Chen; Verlag: Prentice Hall PTR; ISBN 0-13-254384-2

LabVIEW Beispiele, Die Kunst der grafischen Programmierung; Autoren: Ralph Griemert & Wolfgang Erhart; Verlag: Eigenverlag

Praxisbuch LabVIEW 3; Autor: Hans-Günter Dahn; Verlag: IWT Verlag GmbH; ISBN 3-88322-445-6 (ca. Fr. 95.-)

LabVIEW Power Programming; Editor: Gary.W. Johnson; Verlag: McGraw-Hill; ISBN 0-07-913666-4

6.2 LabVIEW Internet-Adressen

6.2.1 National Instruments Austin Texas

- Web-site: <http://www.ni.com> (National Instruments home page)
- Ftp-site: <ftp.ni.com> (for drivers, Info usw.)

6.2.2 LabVIEW Mailgroup

- Website: <http://www.info-labview.org/> Eine Internetseite der LabVIEW mailgroup, wo zurzeit ca. 20000 LabView-Programmierer registriert sind und international über E-Mail kommunizieren.

6.2.3 Brian Renken LabVIEW-pages

- Website: <http://LabVIEW.BrianRenken.com> Brian Renken ist ein engagierter LabVIEW-Programmierer; er unterhält eine Internetseite mit vielen links zu anderen LabVIEW Sites in Form eines Web Ring. Er offeriert dort einen eine gute Suchmaschine für die info-labview Archive.

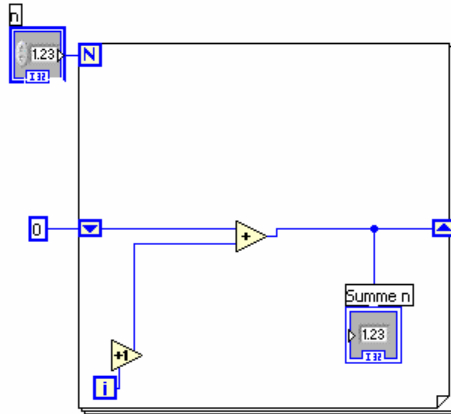
6.2.4 LabVIEW in der Ausbildung

- Website: <http://www.ni.com/academic> für eine Liste von Institutionen, die LabVIEW im Bildungsbereich einsetzen.
- Website: <http://www.ni.com/company/robofab.htm> Stellt ein Lego-Bausteinsystem vor, welches von Lego in Zusammenarbeit mit National Instruments für den Unterricht in Schulen konzipiert wurde. Das System erlaubt es Schülern Maschinen, Fahrzeuge, Automaten oder Roboter aus Lego-Bausteinen zu konstruieren und sie in einer auf LabVIEW basierenden Umgebung zu programmieren. Die erstellten Programme können über eine drahtlose Infrarotverbindung in einen kleinen "Controller" der Konstruktion kopiert werden um sie autonom intelligent zu steuern.
- Website: <http://ldaps.arc.nasa.gov/LEGOEngineer/> Beispiele von Lego-Robolab-Applikationen.

7 ANHANG

7.1 Begegnung mit LabVIEW

Schon an einfachen Beispielen wird die Faszination, die von LabVIEW ausgeht, klar:



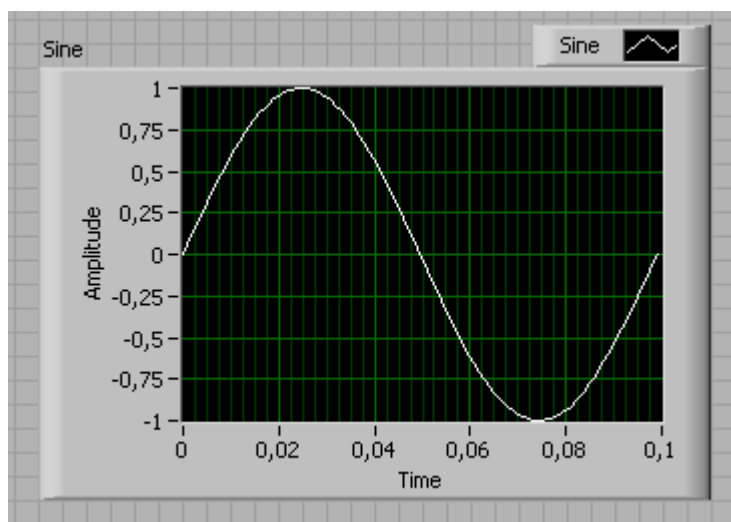
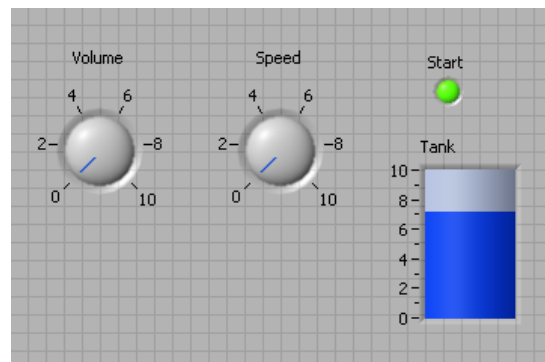
Grafisches Programmieren durch das Verbinden von Icons

Dieses kleine Programm rechnet die Summe aller Zahlen bis zu dem in der Variable n vorgegebenen Wert.

Dabei handelt es sich aber nicht um das Flussdiagramm des Programms sondern um den tatsächlichen ausführbaren Quellcode.

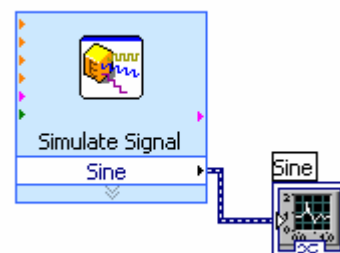
Interaktive Bedienelemente für Benutzer-eingaben

Diese Bedienelemente werden einfach zur Verfügung gestellt. Der Programmierer braucht sie nur aus einem Menü per Mausklick auswählen und auf die Fensteroberfläche des „Front Panel“ zu legen.



Ausgabemöglichkeiten als skalierbare Grafen

Einfach auswählen, auf die Fensteroberfläche legen und die Datenquelle mit einem Verbinder graphisch anhängen, schon wird die Kurve dargestellt.

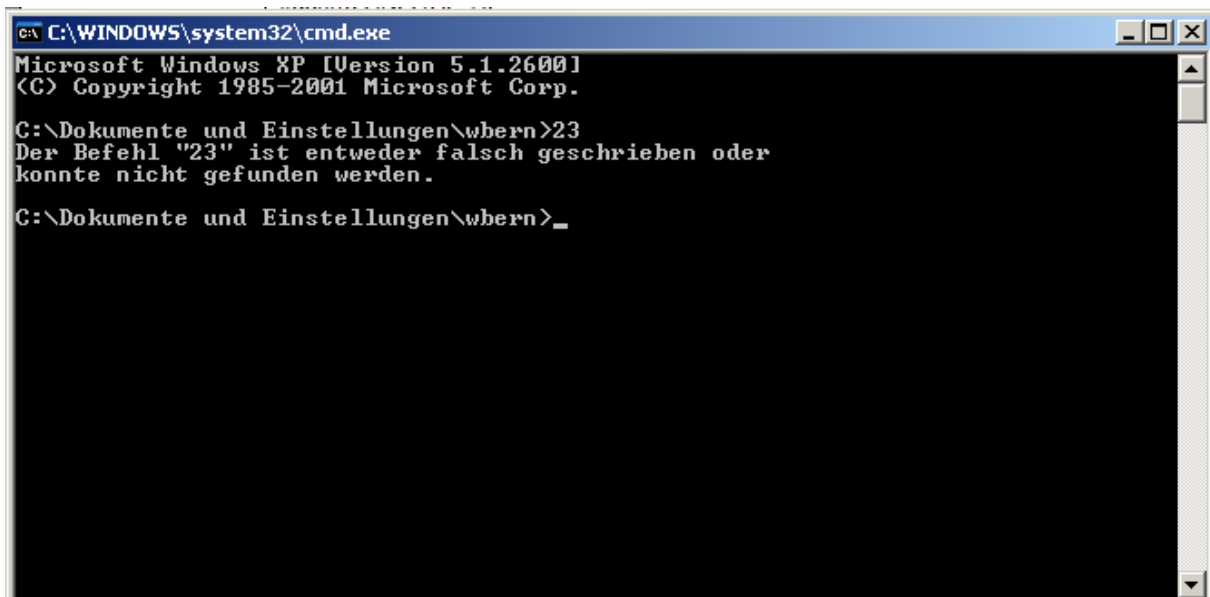


Dagegen ist selbst ein einfaches Programm in einer üblichen Hochsprache wie C oder Pascal zu verfassen eine umständliche Arbeit mit vielen Programmzeilen, die gute Kenntnisse der benutzten Programmierumgebung verlangt.

```
#include <stdio.h>
#define ANZAHL 10
/*
Dieses Programm gibt die Liste der ersten ANZAHL Quadratzahlen aus.
*/
int main(void) {
    int zahl = 1; /* nimmt die aktuelle Zahl auf */

    printf("Liste der ersten %i Quadratzahlen:\n", ANZAHL);
    while (zahl <= ANZAHL)
    {
        printf("%i\n", zahl*zahl);
        zahl = zahl + 1;
    }
    return 0;
}
```

Dabei ist aber keine graphische Ausgabe vorgesehen, sondern nur weiße Zahlen auf schwarzem Hintergrund eines Fensters, das so aussieht, als wäre es aus der DOS-Ära übrig geblieben.



The image shows a screenshot of a Windows command prompt window. The title bar reads "C:\WINDOWS\system32\cmd.exe". The window content displays the following text: "Microsoft Windows XP [Version 5.1.2600] (C) Copyright 1985-2001 Microsoft Corp." followed by the prompt "C:\Dokumente und Einstellungen\wbern>23". Below this, an error message is shown: "Der Befehl '23' ist entweder falsch geschrieben oder konnte nicht gefunden werden." The prompt continues with "C:\Dokumente und Einstellungen\wbern>_" and a cursor.

Wegen des großen Aufwands Programme herzustellen, die nicht nur den Anforderungen des verwendeten Algorithmus sondern auch optischen Mindestansprüchen

genügen, werden Versuchsabläufe im Unterricht praktisch nie über Computer gesteuert oder dabei notwendige Messungen über Schnittstellen erfasst.

7.2 Das Konzept von LabVIEW

Das grundlegende LabVIEW-Konzept beruht auf der Hierarchie von „virtuellen Instrumenten“, VIs. Diese haben als intuitive Benutzerschnittstelle die Nachahmung eines wirklichen Instrumentes mit Knöpfen, Schaltern und grafischen Anzeigen.

Ein virtuelles Instrument ist aus darunter liegenden virtuellen Instrumenten, den so genannten „subVIs“ zusammengesetzt, ähnlich einer elektronischen Schaltung, die ebenfalls aus untergeordneten Bausteinen besteht.

Die übergeordneten Funktionen werden in der untersten Hierarchiestufe von den fundamentalen Funktionsblöcken, die von LabVIEW zur Verfügung gestellt werden, gebildet. Diese werden aus den Bibliotheken der Entwicklungsumgebung geladen und kompiliert. Diese Vielschichtarchitektur mit universal austauschbaren Modulen erlaubt die Konstruktion komplexer Programme, die über eine einheitliche Schnittstellenstruktur verknüpft werden.

Dabei besitzt LabVIEW alle Eigenschaften einer „normalen“ Programmiersprache.

Der Ein- und Ausgabemechanismus ist wichtiger Bestandteil von virtuellen Instrumenten. Jedes VI hat seine eigene, integrierte Benutzerschnittstelle, die eine einfache Bedienung auf jeder Hierarchiestufe erlaubt und das Austesten einzelner VIs als Bausteine erleichtert.

LabVIEW benutzt das Konzept des Datenfluss-Modells in Anlehnung an Blockdiagramme, wie sie in den Ingenieurwissenschaften üblich sind. Daten und Variablen eines Programms werden in Analogie zu Signalen in der Elektrotechnik durch Verbindungen übermittelt.

Ein grafischer Compiler ist in der Lage, das gezeichnete Programm in lauffähige Maschinensprache umzusetzen.

Er ist direkt ins Entwicklungssystem eingebunden und arbeitet so schnell, dass ein Programmierer den Prozess des Kompilierens nicht wahrnimmt.

Durch die konsequente Weiterentwicklung zu einem mächtigen Programmiersystem wurde LabVIEW im Bereich der modernen Mess- und Testtechnik sowie der Prozesssteuerung zu einem Industriestandard.

Die LabVIEW-Vollversion enthält Werkzeuge zur Entwicklung komplexer Applikationen:

- Multithreading: unterschiedliche Programmteile können unabhängig voneinander vom darunter liegenden Betriebssystem auf parallel laufende Prozesse oder Prozessoren verteilt werden

- Unterstützung für Applikationen, die auf Netzwerken verteilt Messdaten sammeln beziehungsweise Daten austauschen

- Programmverbindungen zu Fremdapplikationen; Stichwörter sind hier OLE-Automation, Active X, WWW usw.

7.3 Was ist LabVIEW?

LabVIEW (Abkürzung für Laboratory Virtual Instrument Engineering Workbench) ist ein mächtiges, vielseitiges Software-Messwerkzeug und Analysepaket für PCs und Workstations (Apple Macintosh, Microsoft Windows, Sun SPARC und HP 9000/700 unter HP-UX).

LabVIEW ist in erster Linie eine Entwicklungsumgebung wie professionelle C- oder BASIC-Systeme.

Mit einem großen Unterschied allerdings: Herkömmliche Programmiersysteme verlangen die Anweisungen in Textform, LabVIEW aber besitzt seine eigene grafische Programmiersprache „G“.

Programme werden grafisch in Form von Flussdiagrammen mit der Computermaus gezeichnet und konstruiert, man spricht dabei von „Blockdiagrammen“. Solche Diagramme eliminieren viele syntaktische Details konventioneller Hochsprachen.

LabVIEW benutzt Terme, Module und Konzepte, die dem Ingenieur und Wissenschaftler vertraut sind, und definiert im Gegensatz zu Textsprachen ein Programm mit grafischen Symbolen.

LabVIEW ist grundsätzlich ohne Programmierkenntnisse erlernbar.

Es hat sich gezeigt, dass besonders junge Menschen die grafische Syntax rasch lernen.

Diese Syntax und die interaktive Ein- und Ausgabe von Variablen fördert den intuitiv spielerischen Umgang mit der Programmierkunst.

Da sich die elementare Programmierstruktur nicht von denjenigen herkömmlicher Programmiersprachen unterscheidet, kann Programmiererfahrung für das Studium von LabVIEW immerhin nützlich sein.

LabVIEW kümmert sich für uns um viele kleine Details. Es stellt umfangreiche Bibliotheken mit Funktionen und Subroutinen für großzügiges Programmieren zur Verfügung.

LabVIEW besitzt zahlreiche applikationsspezifische Funktionen zur Messdatenerfassung, zur Verwendung mit dem GPIB (General-Purpose Interface Bus), dem Standardbus zur Steuerung externer Messinstrumente, für Geräte, welche über die serielle Schnittstelle kommunizieren, sowie zur Datenanalyse, Datenpräsentation und Datenspeicherung.

Die Analysebibliotheken umfassen zahlreiche Module zur Messdatenverarbeitung wie Filter, Windows, Statistiken, Regressionen, Lineare Algebra, Matrizen-Arithmetik usw.

LabVIEW bietet Programmentwicklungswerkzeuge zur Fehlerbeseitigung wie das Setzen von breakpoints, das Schritt-für-Schritt-Abarbeiten eines Programms sowie die Möglichkeit der Datenflussanimation im Blockdiagramm.

Auf Grund der grafischen Struktur ist LabVIEW auch ein vielseitiges Präsentationspaket.

Die Datenausgabe kann in der gewünschten Form gestaltet werden.

Datenschreiber, Grafiken und benutzerdefinierte Darstellungen sind nur ein kleiner Teil der Optionen.

Die Messdatenerfassung, die Analyse- und Darstellungswerkzeuge machen LabVIEW zu einem mächtigen Entwicklungssystem.

Alle Problemlösungsmöglichkeiten einer konventionellen Programmiersprache sind als virtuelle Instrumente möglich.

LabVIEW-Programme werden virtuelle Instrumente („Virtual Instruments“), kurz VIs genannt, da ihre Erscheinung und Funktion Hardwaregeräten ähnlich sind. Diese VIs funktionieren je nachdem als Hauptprogramme, Funktionen oder Subroutinen wie in den konventionellen Programmiersprachen C, Pascal, BASIC usw.

Jedes VI besitzt eine interaktive Benutzeroberfläche, die gleichzeitig die Schnittstelle definiert, über welche die VIs untereinander Daten austauschen.

LabVIEW-VIs bestehen aus drei Komponenten:

Dem „Front Panel“:

Dies ist die interaktive Benutzeroberfläche oder Variablenschnittstelle eines VIs. Es simuliert die Frontplatte eines Gerätes mit Knöpfen, Schaltern, Grafiken und vielen andern controls (Benutzereingaben) und indicators (Programmausgaben). Der Benutzer gibt die Daten über die Tastatur oder mit der Maus ein und sieht das Resultat nach der Verarbeitung durch das Programm.

Dem „Block Diagram“:

Es handelt sich dabei um den Quellcode des VIs, konstruiert mit LabVIEWs grafischer Programmiersprache „G“. Dieses „Block Diagram“, obschon es sehr bildhaft aussieht, ist das lauffähige Programm. Die Teile des Blockdiagramms, die Symbole (icons) stellen untergeordnete VIs dar: Entweder sind dies von LabVIEW zur Verfügung gestellte Funktionsmodule und Programmstrukturen oder vom Benutzer selbst programmierte VIs. Die Blöcke werden mit Drähten zweckmäßig verbunden. Die Verbindungen symbolisieren die Kanäle, durch welche die Daten zur Programmlaufzeit fließen. Der Datenfluss im Blockdiagramm bestimmt somit den Ablauf eines Programms. Jede LabVIEW-Funktion wird erst dann ausgeführt, wenn an all ihren „verdrahteten“ Eingängen Daten anliegen; sie gibt danach die verarbeiteten Werte an das aufrufende Diagramm zurück.

Dem Symbol (icon) mit dem Verbinder (connector) eines VIs:

Dies ist die Schnittstelle, die den Datentransfer zu anderen VIs regelt. Das Symbol repräsentiert ein VI im Blockdiagramm eines übergeordneten VI. Der Verbinder definiert die Ein- und Ausgangsparameter des VI. VIs können hierarchisch und modular in Baumstrukturen aufgebaut werden; sie können entweder ein top-level-Programm oder ein beliebiges Unterprogramm sein. Ein VI in einem übergeordneten VI wird in Analogie zur Subroutine „subVI“ genannt. Das Symbol (icon) repräsentiert die Funktion im Diagramm, die Parameter werden über die Schnittstelle (connector) definiert.

Mit seiner Struktur unterstützt LabVIEW das modulare Programmieren. Zuerst wird eine Applikation in eine Reihe einfacher Prozesse aufgeteilt, danach baut man die funktionsfähigen Teilbausteine zusammen und kombiniert sie in einem top-level-Diagramm. Jedes subVI ist autonom lauffähig und kann somit leicht individuell getestet werden. Die subVIs können von Programmen beliebig an unterschiedlichen Stellen aufgerufen werden.

Neben den bereits erwähnten Eigenschaften wie grafisches Programmieren, Datenfluss- gesteuerter Programmablauf und interaktives Benutzerinterface in Anlehnung an reale Geräte, besitzt LabVIEW noch weitere charakteristische Konzepte, die sich von denjenigen konventioneller Programmiersprachen unterscheiden.

Einige davon sind im Folgenden aufgelistet und kurz erläutert:

Die Möglichkeit des automatischen Indizierens und Aufbauens (auto indexing) von n-dimensionalen Matrizen durch die Schleifenstrukturen „For“ und „While“.

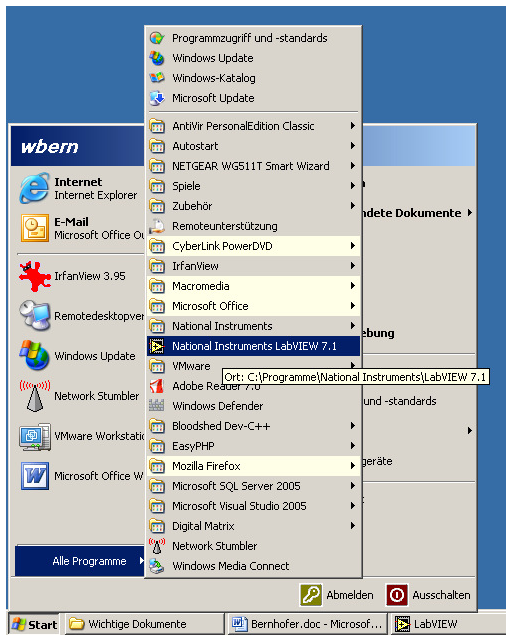
Das transparente Anbringen eines Schieberegisters (shift register) an eine Schleifenstruktur, was den Zugang zu den Werten in vorausgegangenen Iterationen erleichtert.

Die Anpassung unterschiedlicher Datentypen durch automatische Adaption (polymorphism). Der Polymorphismus erlaubt zum Beispiel die skalare Addition zweier Matrizen, wobei jedes Element der ersten Matrix zu jedem entsprechend indizierten Wert der zweiten Matrix addiert wird, ebenso wie die Addition einer Matrix mit einem einfachen Skalar, wobei der Wert des Skalars zu jedem Element der Matrix addiert wird, dies wohlverstanden mit der selben Additionsfunktion. Dabei spielt es keine Rolle, ob die Eingänge unterschiedliche numerische Datentypen aufweisen; der Datentyp des Ausgangsparameters wird automatisch entsprechend konvertiert.

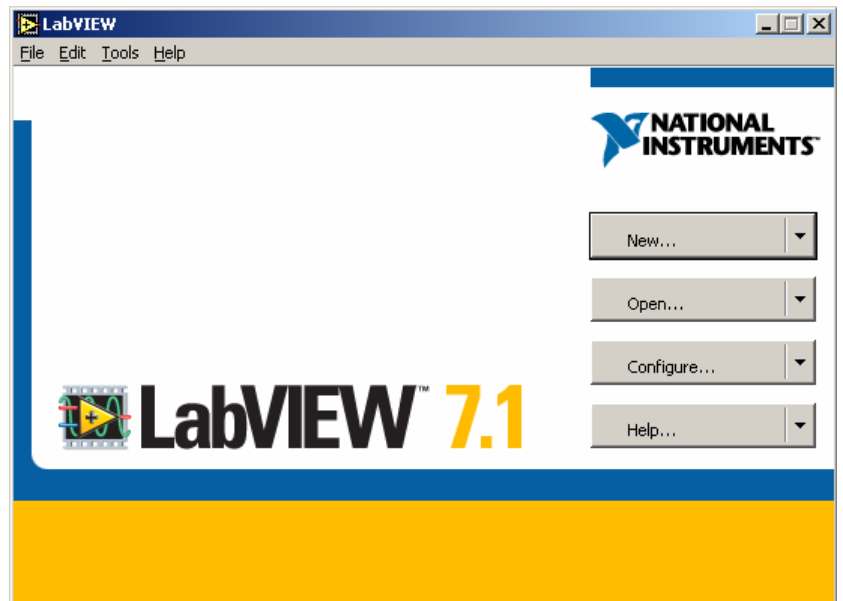
Die grafische Syntax macht es sehr einfach, Schleifen in einem LabVIEW-Programm mit unterschiedlichen Geschwindigkeiten parallel laufen zu lassen. So ist es bei-

spielsweise möglich, eine schnelle Schleife zur Messdatenerfassung parallel zu einer langsameren zu programmieren, welche gelegentlich einen Wert in einem Grafen darstellt. Der Datenaustausch zwischen parallel laufenden Schleifen geschieht durch entsprechend definierte globale oder lokale Variablen (server-client model). Solche Vorgänge sind in einer konventionellen Programmiersprache viel schwieriger zu realisieren.

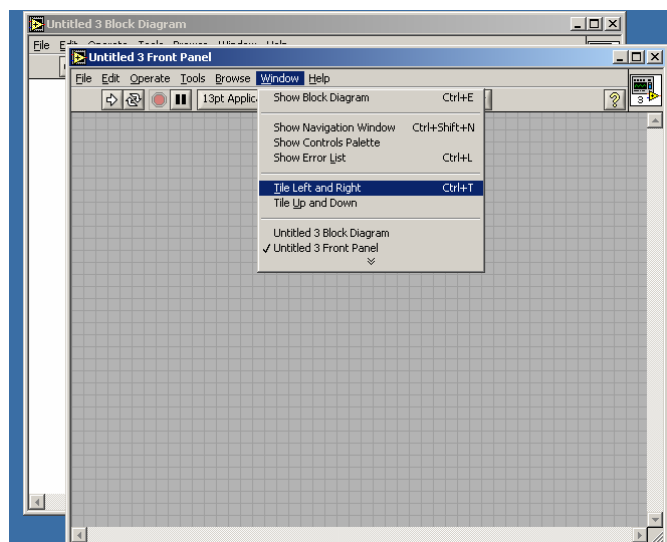
7.4 Erste Übung



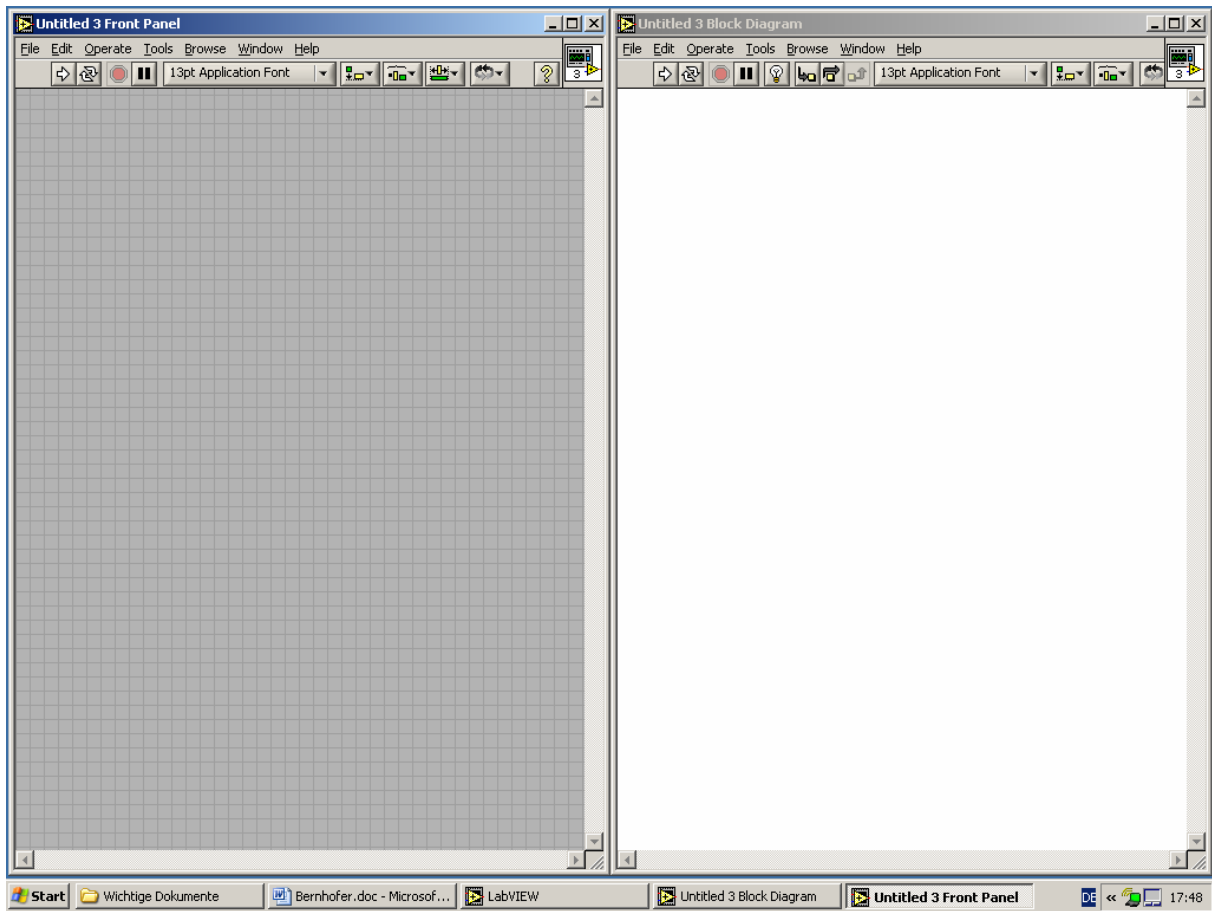
Nach Öffnen von LabVIEW erscheint am Bildschirm:



Nachdem auf den kleinen Pfeil neben „New“ geklickt wurde, wird aus dem erscheinenden Pop-upmenü der Unterpunkt „Blank VI“ gewählt. In der Menüzelle eines der beiden erscheinenden Fenster wählt man unter „Window“ den Eintrag „Tile Left and Right“.

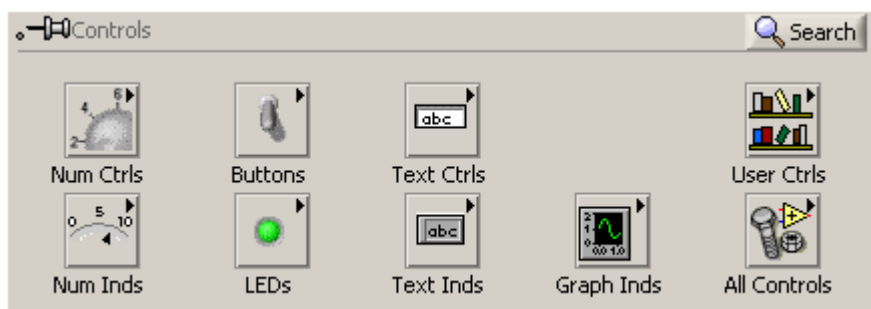


Danach sieht der Bildschirm wie folgt aus:

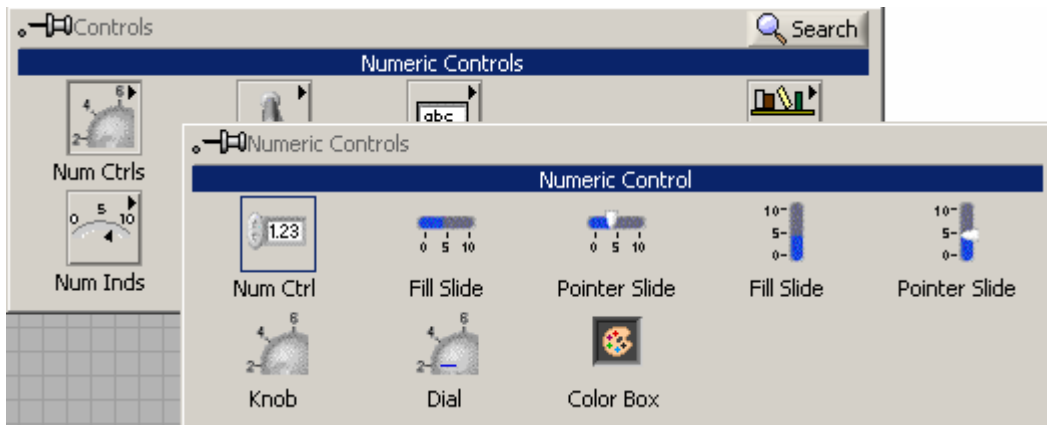


Dabei handelt es sich beim linken Fenster um das „Front Panel“, das in späterer Folge die Bedienelemente und Ausgabeelemente aufnehmen wird, das rechte Fenster ist das „Block Diagram“, in das der Programmcode mit der Maus gezeichnet wird.

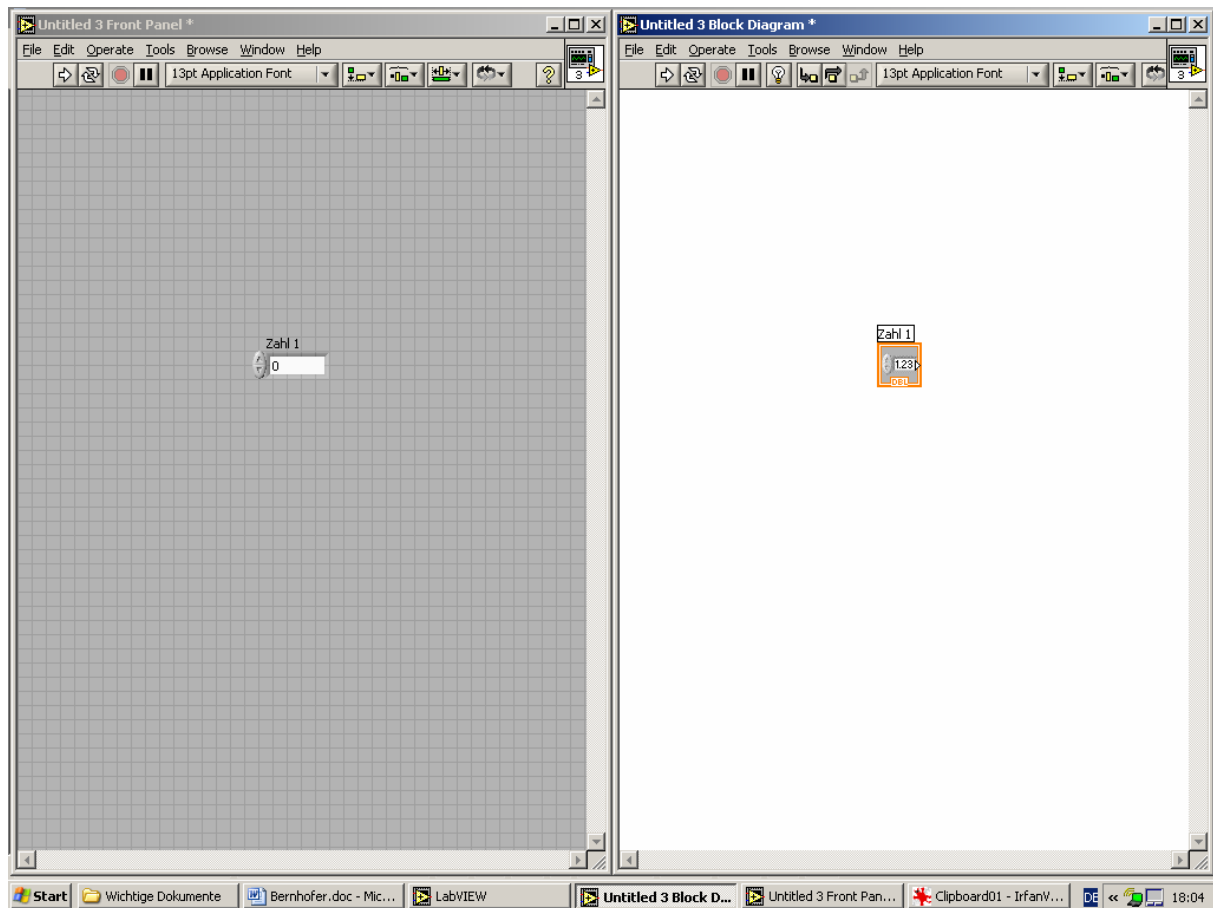
Klickt man mit der rechten Maustaste irgendwo ins „Front Panel“ so erhält man das „Controls“ Menü.



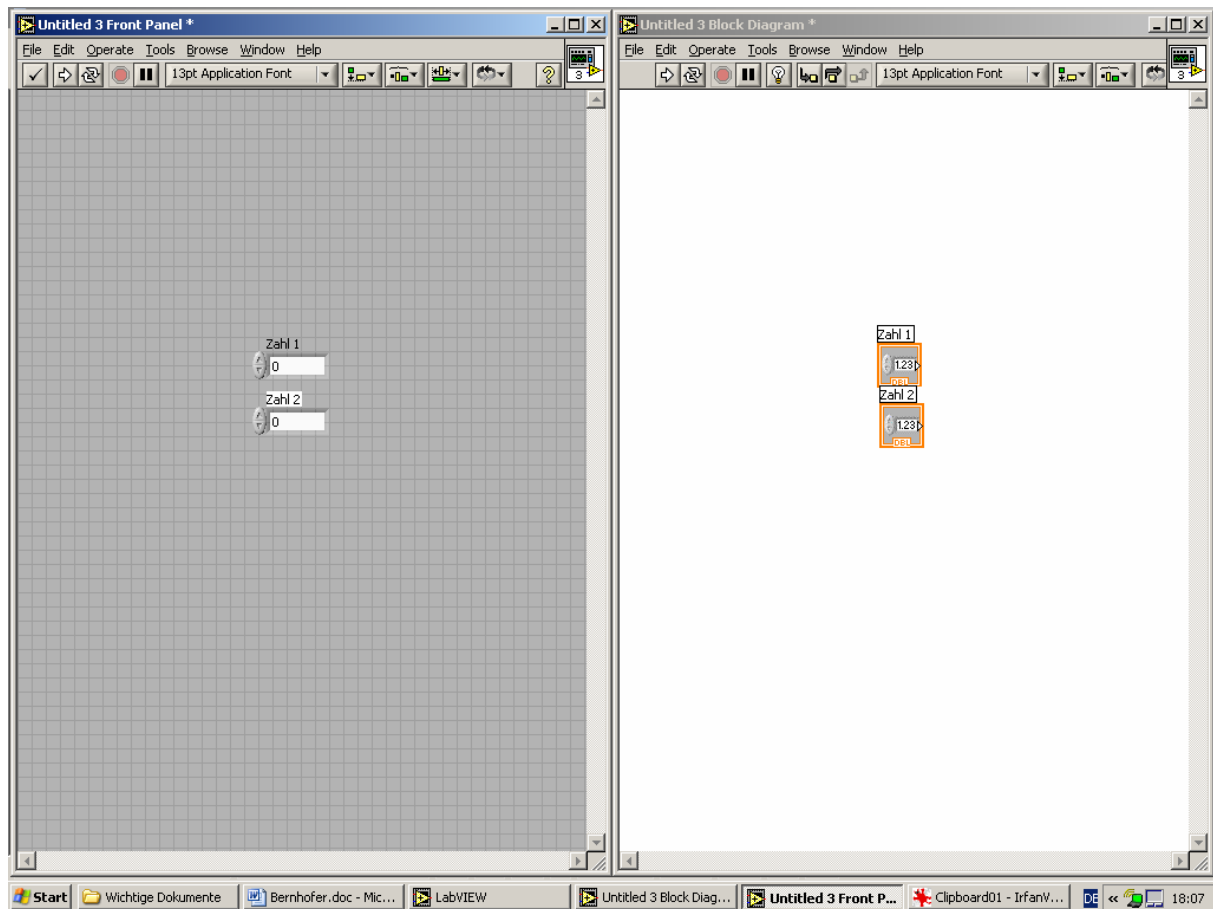
Dort wählt man durch darüber führen der Maus das Untermenü „Num Ctrls“ und in diesem das Icon „Numeric Control“ aus.



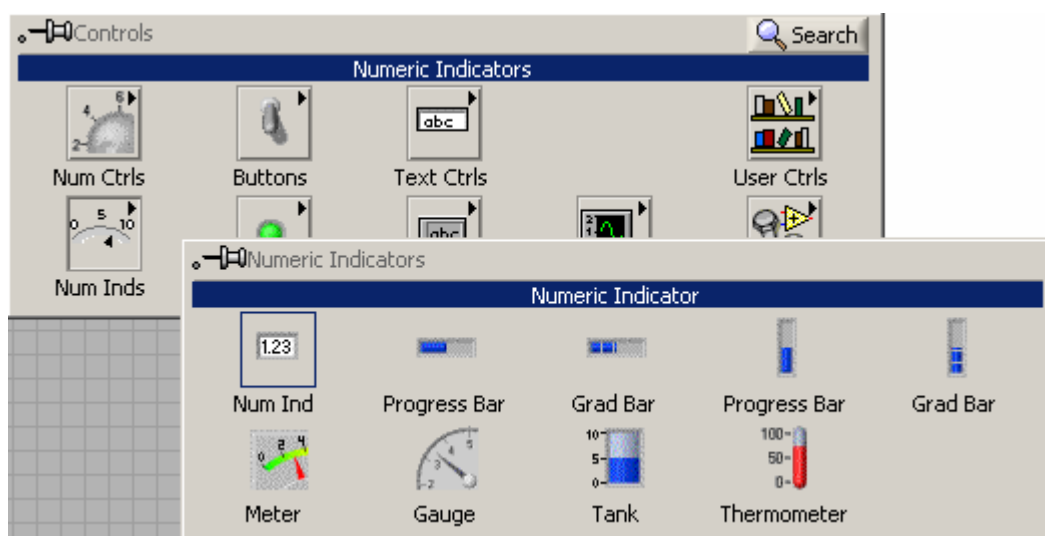
Dieses wird nun irgendwo in das Front Panel durch klicken abgelegt. Unmittelbar nach dem Ablegen, ist die Beschriftung oberhalb des Icons schwarz unterlegt und kann durch unmittelbares Tippen verändert werden.



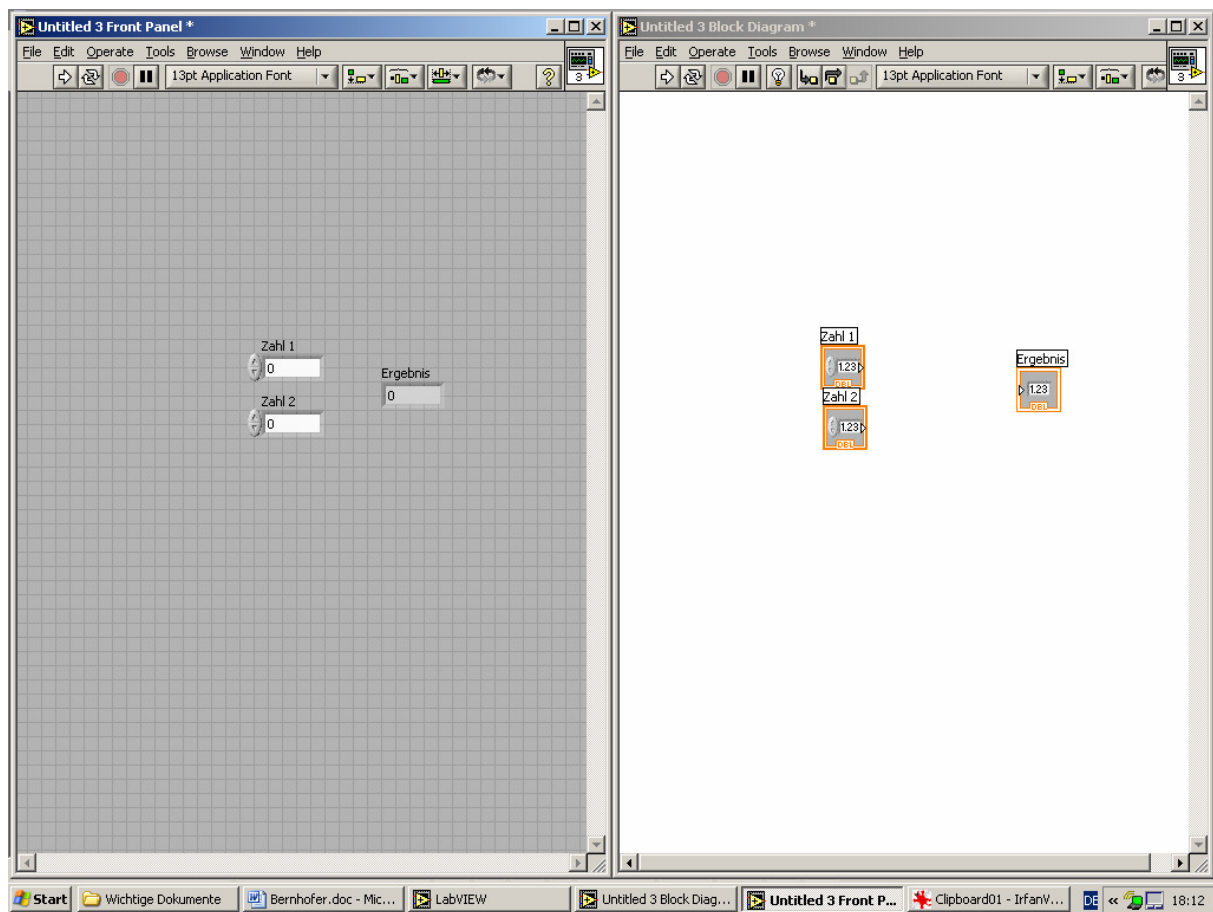
Analog wird nun noch einmal vorgegangen um ein zweites Icon, welches Zahl 2 heißen soll zu erzeugen.



Eine ähnliche Vorgehensweise wird ein drittes Mal angewandt, mit dem Unterschied, dass im „Controls“-Menü die Maus über den Punkt „Num Inds“ geführt wird und aus dem sich öffnenden Menü „Numeric Indicator“ gewählt wird.

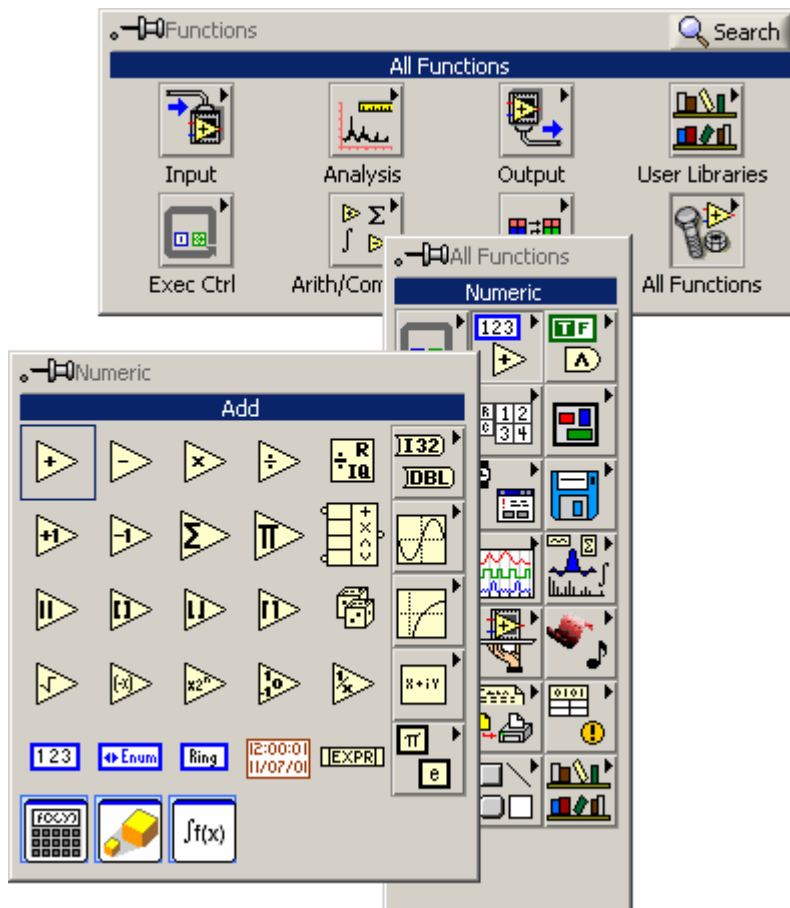


Die beiden Fenster sollten nun wie folgt aussehen:

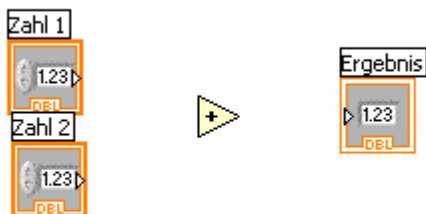


In der weiteren Folge wenden wir uns dem andern Fenster zu. In diesem Fenster, dem „Block Diagram“, sind während der Erzeugung der Ein- und Ausgabefelder analog der obigen Anleitung, zusätzliche Icons entstanden. Diese stellen die Variablen dar, die mit den eben erzeugten Feldern korrespondieren. Um ein Programm zu zeichnen, müssen diese Variablenicons intelligent verbunden werden.

Zuerst wird ganz analog zu der vorangegangenen Vorgehensweise in das „Block Diagram“ mit der rechten Maustaste geklickt, dann die Maus über den Punkt „All Funktions“ gebracht, anschließend im aufgeklappten Untermenü über den Punkt „Numeric“ und weiter über das Icon „Add“ gebracht, welches durch klicken ausgewählt und auf das „Block Diagram“ gelegt wird.

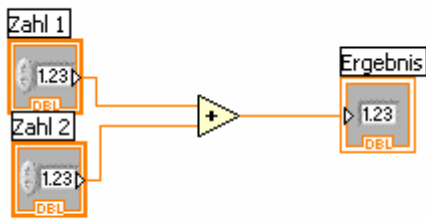


Danach sollten die Icons so gruppiert sein:

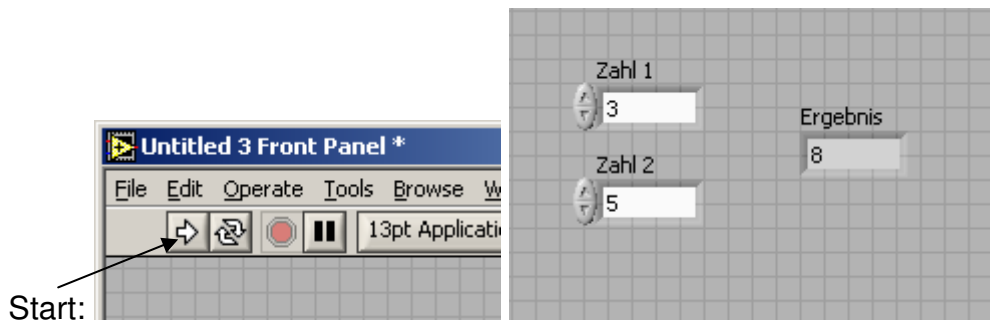


Nun muss noch der Datenfluss hergestellt werden, indem mit dem „Verbinder“ die entsprechenden Ablaufstrukturen geschaffen werden.

Bewegt man den Mauszeiger über den Bereich der Icons, der durch das kleine Dreieck gekennzeichnet ist, so beginnt dieser Teil des Icons zu blinken und der Cursor verwandelt sich in eine kleine Drahtspule, mit der nun die Icons jeweils mit den Ecken des „+“ Symbols verbunden werden können, indem nach dem Klicken ein Draht gezogen wird.



Wählt man nun im „Front Panel“ bestimmte Werte für die Zahlen 1 und 2, so kann nach Start des Programms durch Klicken auf den Startbutton in Ausgabefeld Ergebnis das Ergebnis der Addition der beiden Zahlen abgelesen werden.

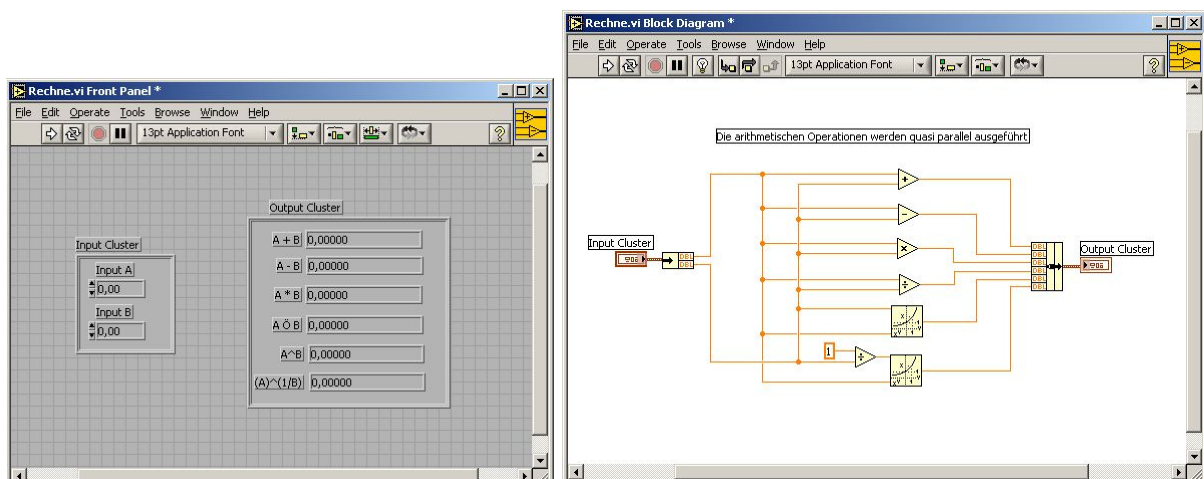


7.5 Programmierkurs - LabView - Serie 1

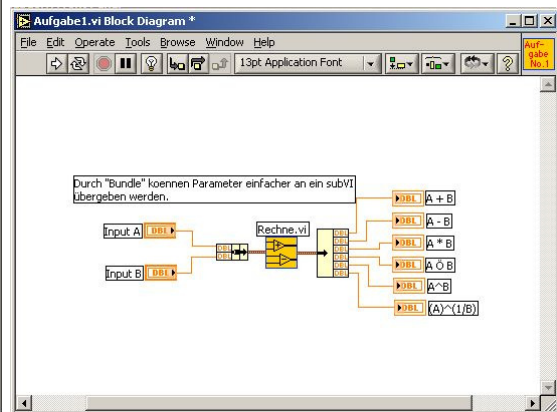
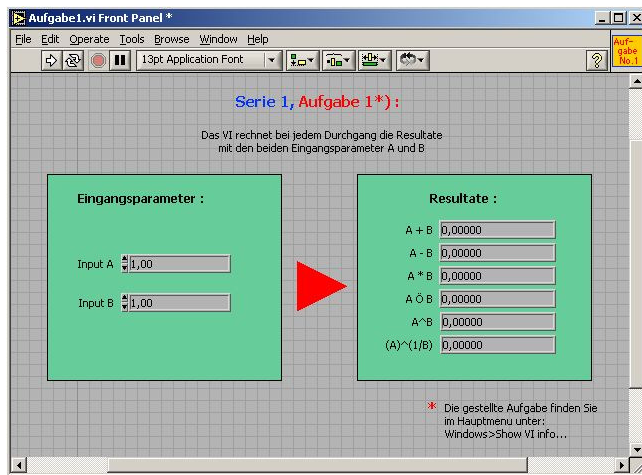
7.5.1 Aufgabe 1

Programmieren Sie ohne Verwendung von Schleifen ein VI das zwei Benutzereingaben mit den einfachen arithmetischen Operationen +, -, * und / verknüpft und bei jedem Programmdurchlauf die Resultate anzeigt.

Die Operationen sollen dabei parallel ausgeführt werden und gleichzeitig richtig angeschrieben als Ausgaben auf dem „Front Panel“ erscheinen.



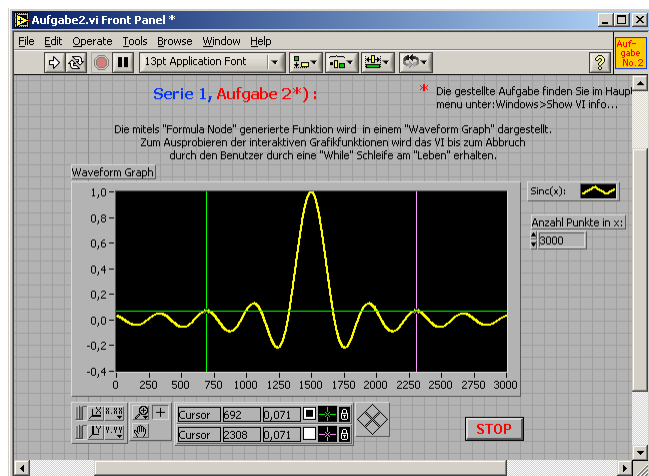
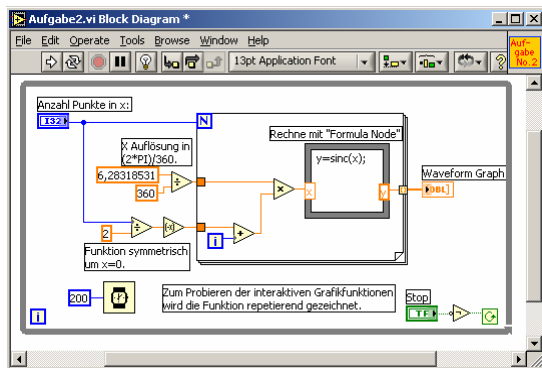
Machen Sie aus diesem VI ein subVI, das Sie aufrufen können. (Tipp: zur Parameterübergabe sind clusters vorteilhaft)



7.5.2 Aufgabe 2

Generieren Sie mittels „formula node“ eine Funktion $y=f(x)$ Ihrer Wahl. Stellen Sie die Funktion nach Berechnen in einem „waveform graph“ zweckmäßig dar.

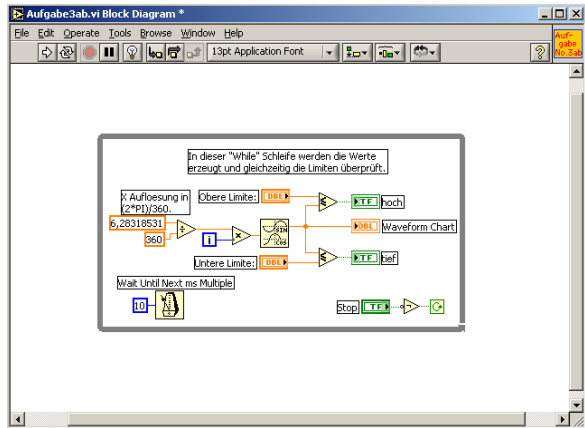
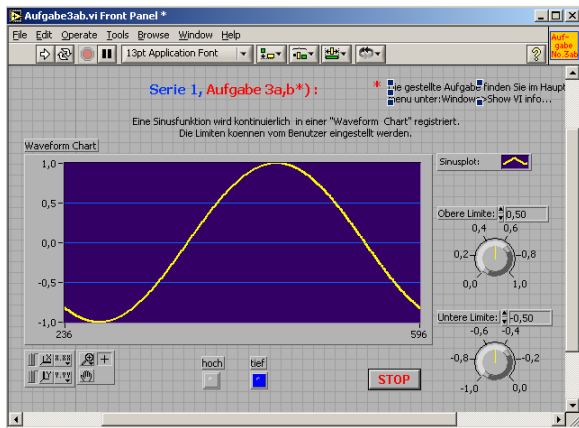
Versuchen Sie die Darstellungsoptionen des Graphen optimal anzupassen. Probieren Sie die eingebauten Cursorfunktionen aus.



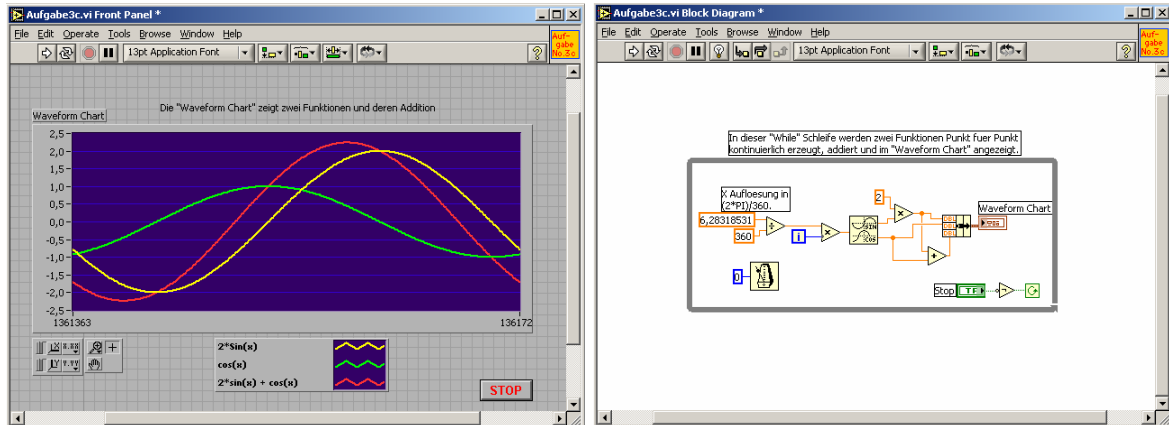
7.5.3 Aufgabe 3

a) Generieren Sie in einer Schleife numerisch eine periodische Funktion $y=f(x)$ Ihrer Wahl die Sie in einem „waveform chart“ kontinuierlich darstellen können. Implementieren Sie einen „sauberen“ Stop des VIs.

b) Definieren Sie für Aufgabe 3a) je eine minimale und maximale Limite die beim Unter- bzw. Überschreiten einen Alarm ihrer Wahl auslöst.

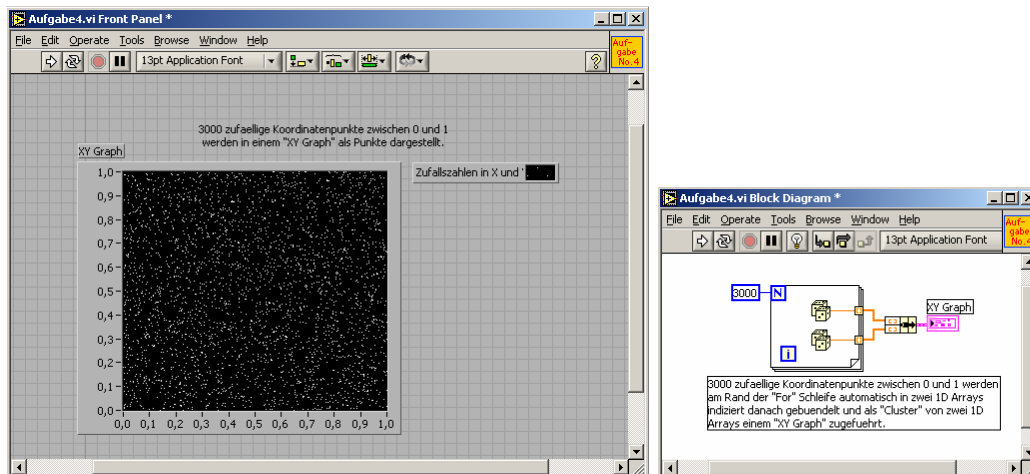


c) Bauen Sie das VI von Aufgabe 3b) so aus, dass Sie zwei unterschiedliche Funktionen und deren Addition gleichzeitig auf dem „waveform chart“ darstellen können.



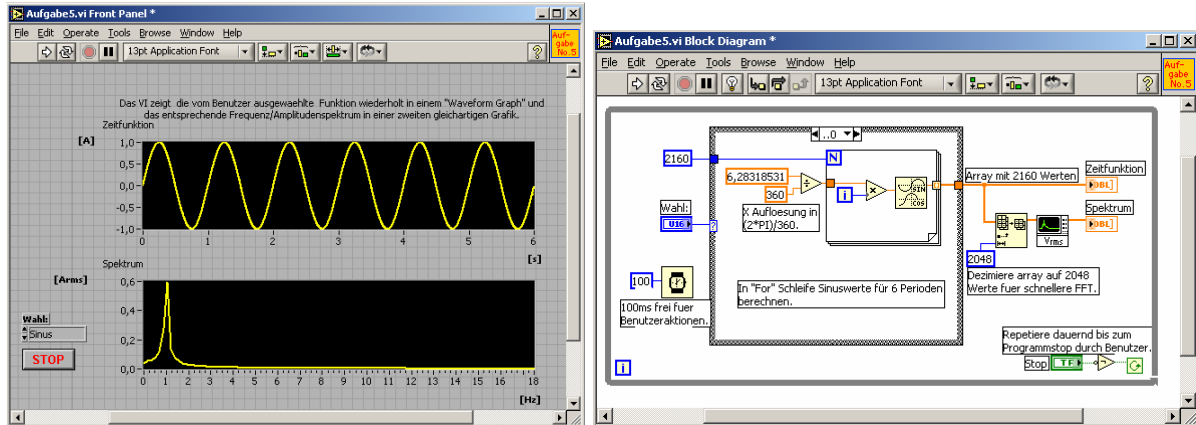
7.5.4 Aufgabe 4

Testen Sie LabVIEWs „random generator“, indem Punkte mit zufälligen Koordinaten in einem „XY chart“ dargestellt werden.



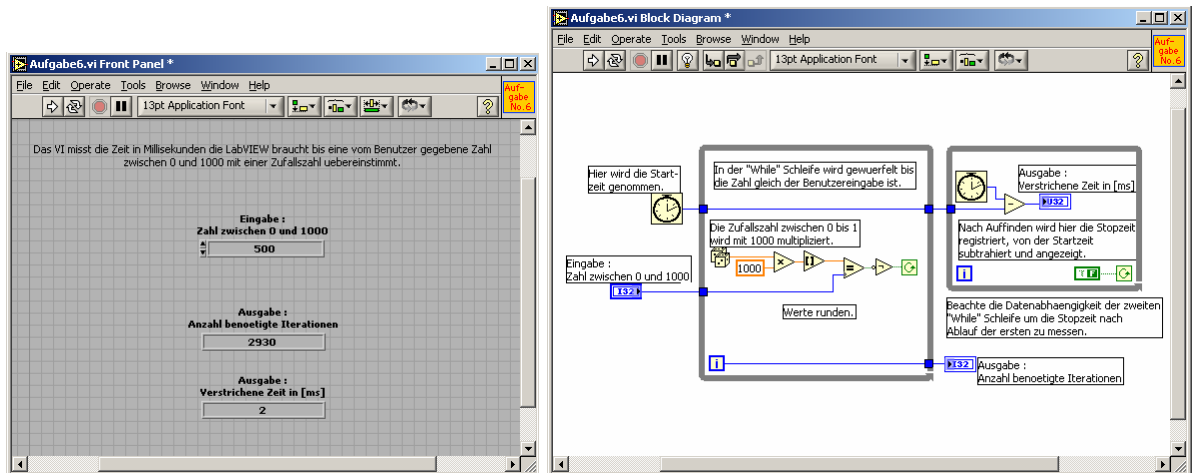
7.5.5 Aufgabe 5

Stellen Sie eine vom Benutzer ausgewählte periodische Funktion repetierend ähnlich einem Oszilloskop in einem „waveform graph“ dar. Realisieren Sie in einem zweiten Graphen die Darstellung der Fourier-Transformierten der aktiven Funktion.



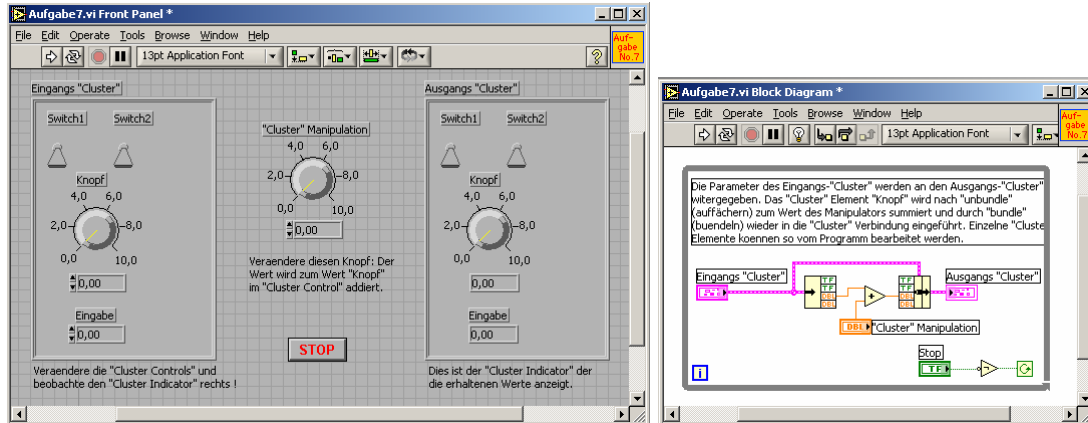
7.5.6 Aufgabe 6

Erstellen Sie ein VI welches die Zeit misst die LabVIEW braucht bis eine vom Benutzer eingegebene Zahl in einem vordefinierten Bereich mit derjenigen einer passend formatierten Zufallszahl übereinstimmt.



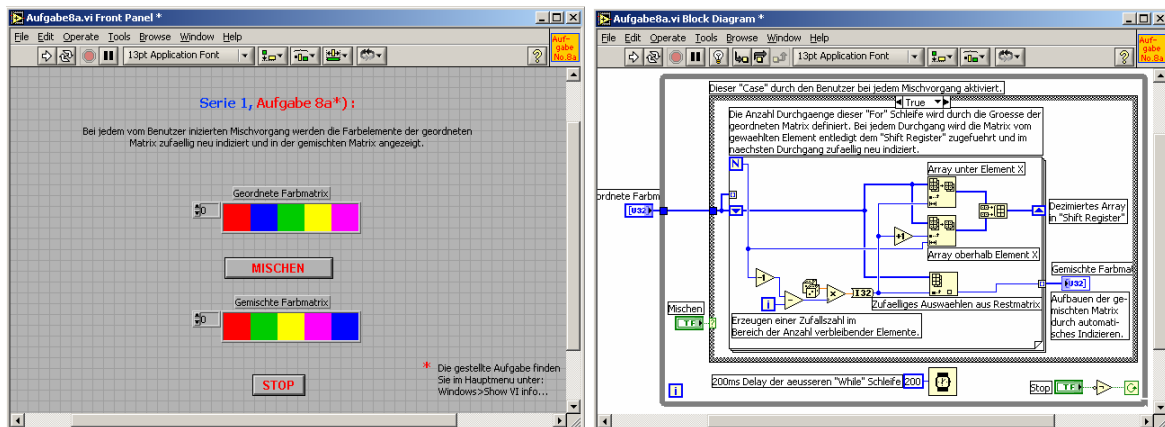
7.5.7 Aufgabe 7

Plazieren Sie je einen „cluster control“ und „cluster indicator“ in einer „While“-Schleife. Verbinden Sie beide Objekte und manipulieren Sie eines der cluster-Elemente durch bundle / unbundle interaktiv.

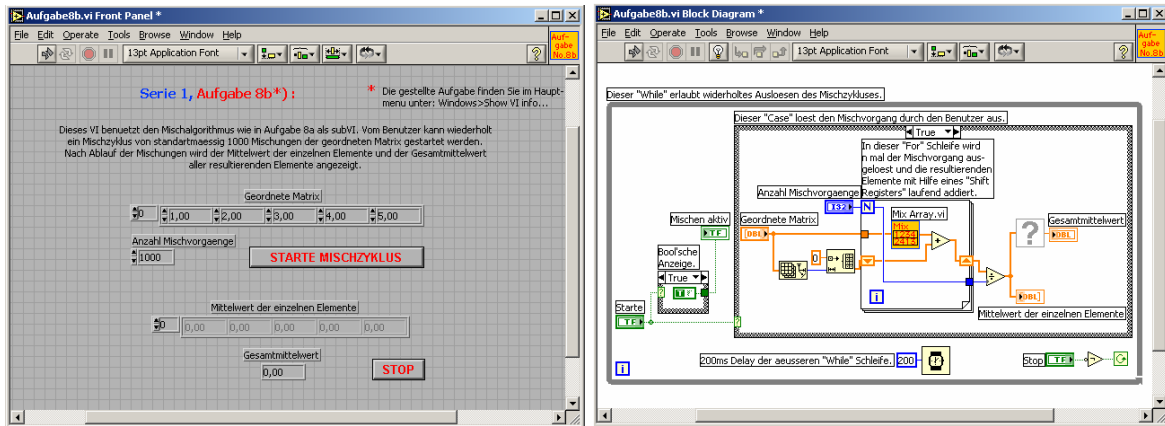


7.5.8 Aufgabe 8

a) Mischen Sie in einer Schleife die Reihenfolge eines Farbarrays mit fünf unterschiedlichen Farbelementen bei jedem Durchgang zufällig neu. Jedes Mischen sollte vom Benutzer auf Knopfdruck gestartet werden können.



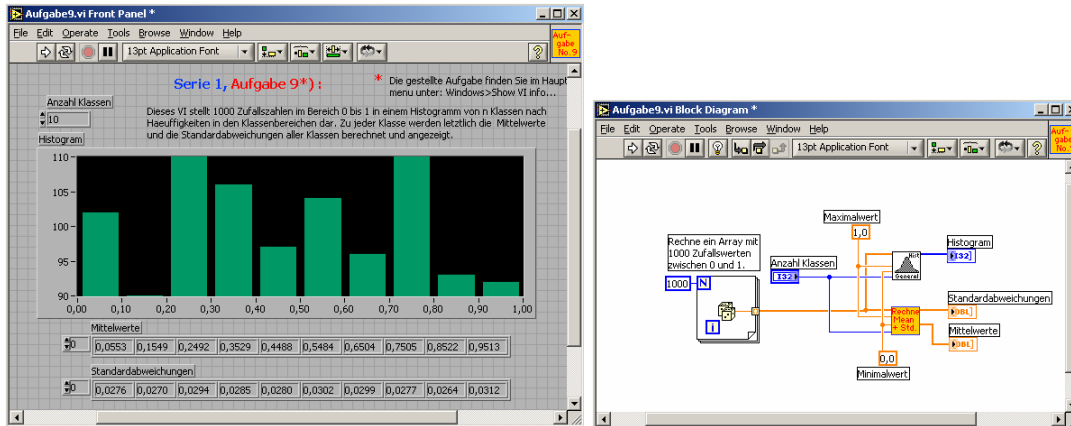
b) Ersetzen Sie die Farbelemente durch Integerzahlen von 1 bis 5, mischen Sie ähnlich Aufgabe 8a) 10000mal. Stellen Sie die fünf arithmetischen Mittelwerte der einzelnen Elemente und den Gesamtmittelwert der fünf Einzelmittelwerte nach Ablauf des VIs dar.

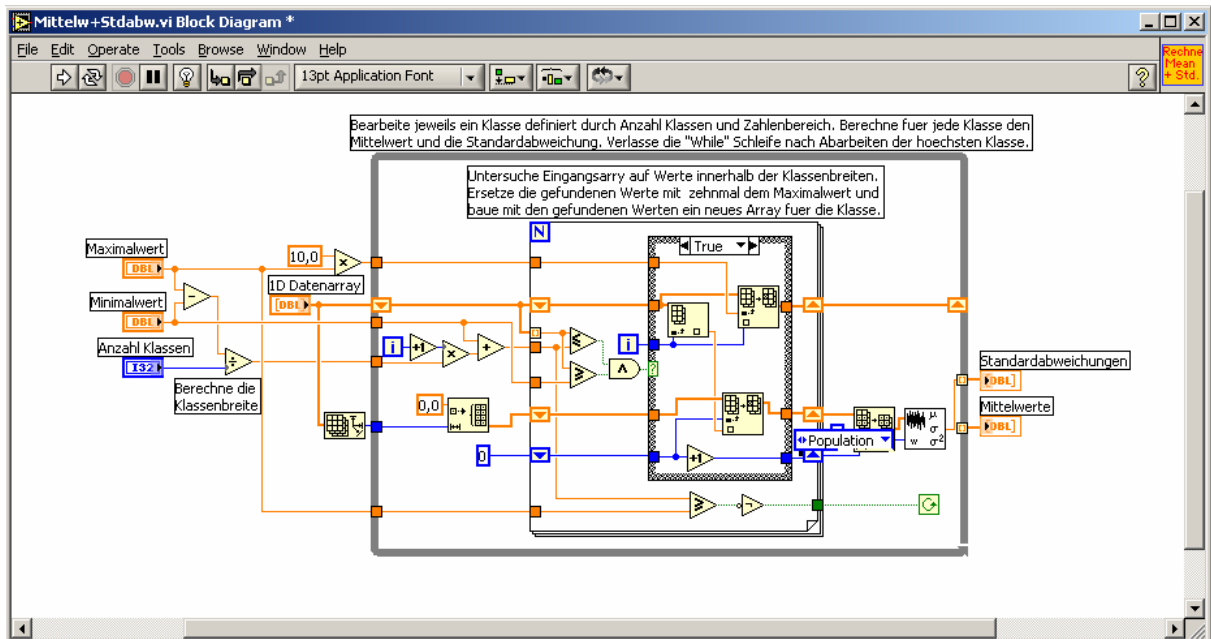


7.5.9 Aufgabe 9

Für fortgeschrittene Programmierer:

Stellen Sie in einem Histogramm 1000 Zufallswerte in 10 Klassen (bins) gleicher Intervalle dar. Berechnen Sie die arithmetischen Mittelwerte und die Standardabweichungen für jede Klasse. (Tipp: Brauchen Sie dazu die autoindexing Fähigkeiten von LabVIEW und shift registers.)

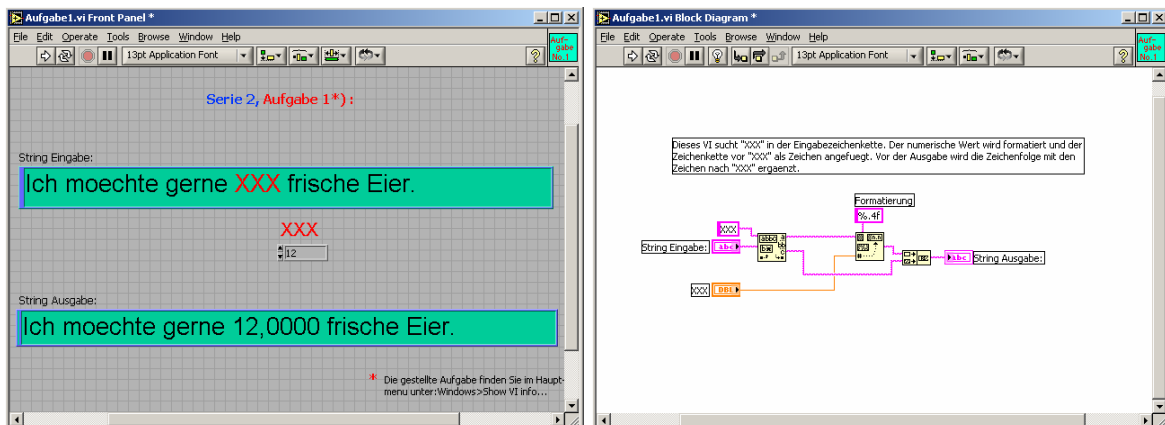




7.6 Programmierkurs - LabView - Serie 2

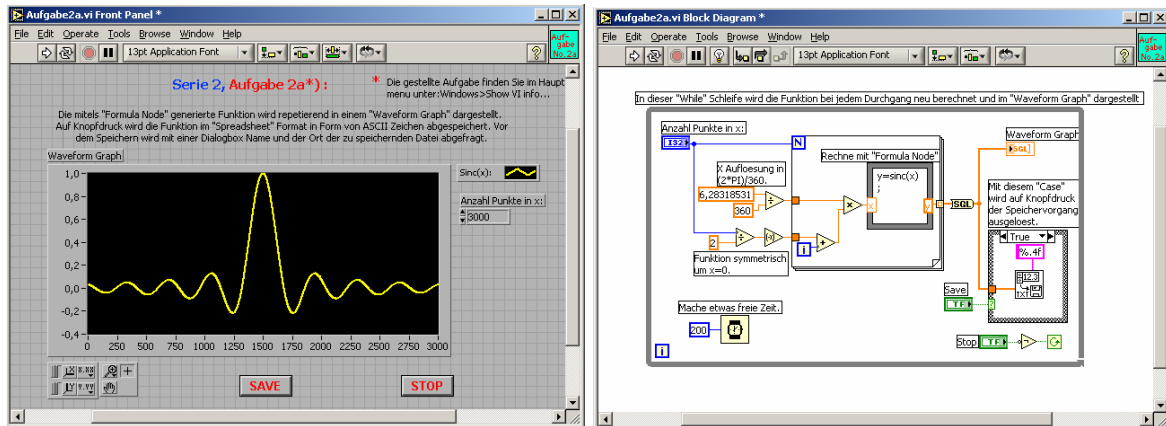
7.6.1 Aufgabe 1

Setzen Sie in einen beliebigen Text einen numerischen Wert des Typs DBL mit 4 Nachkommastellen ein.

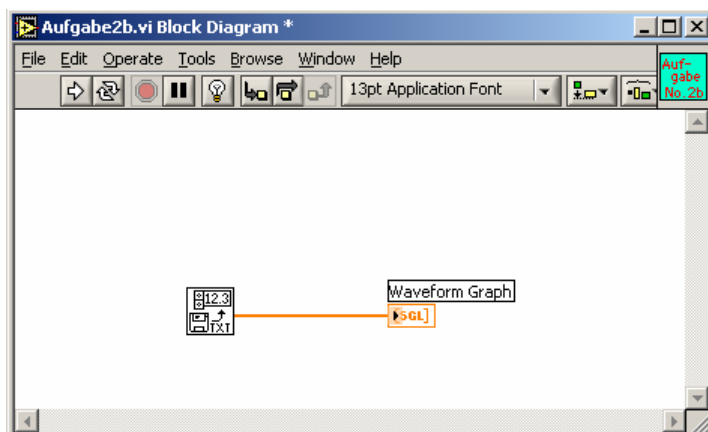


7.6.2 Aufgabe 2

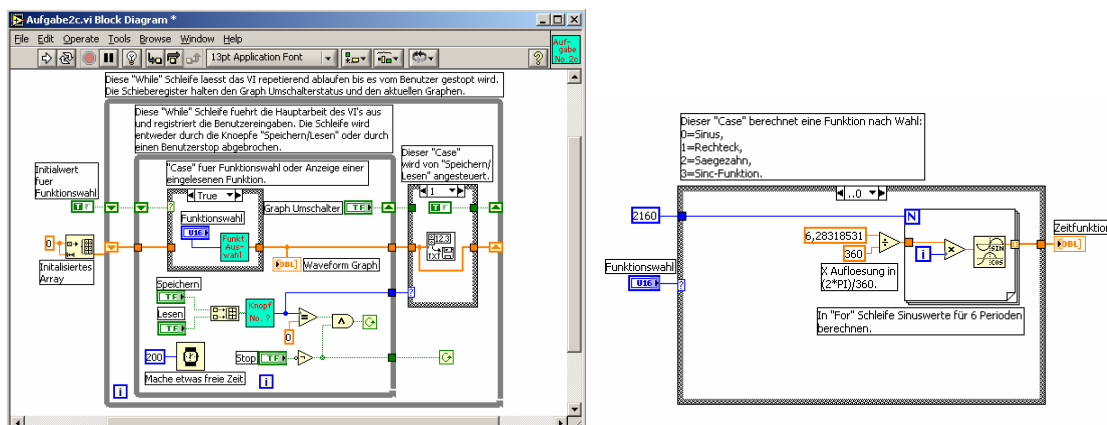
a) Nehmen Sie das VI von Aufgabe 2 der Serie 1 und speichern Sie die Funktion im „Spreadsheet“ Format (ASCII Zeichen) auf der Harddisk in einer Datei ab.

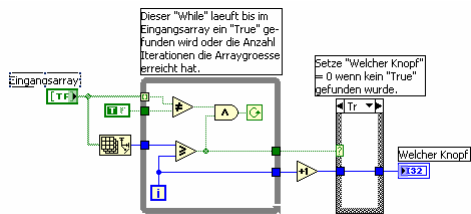


b) Lesen Sie die Funktion von der Datei wieder ein und stellen Sie sie in einem Graphen dar.



c) Automatisieren Sie den Prozess des Abspeicherns und Einlesens so dass unterschiedliche, vom Benutzer ausgewählte Funktionen, im ASCII Format wahlweise abgespeichert und eingelesen werden können.

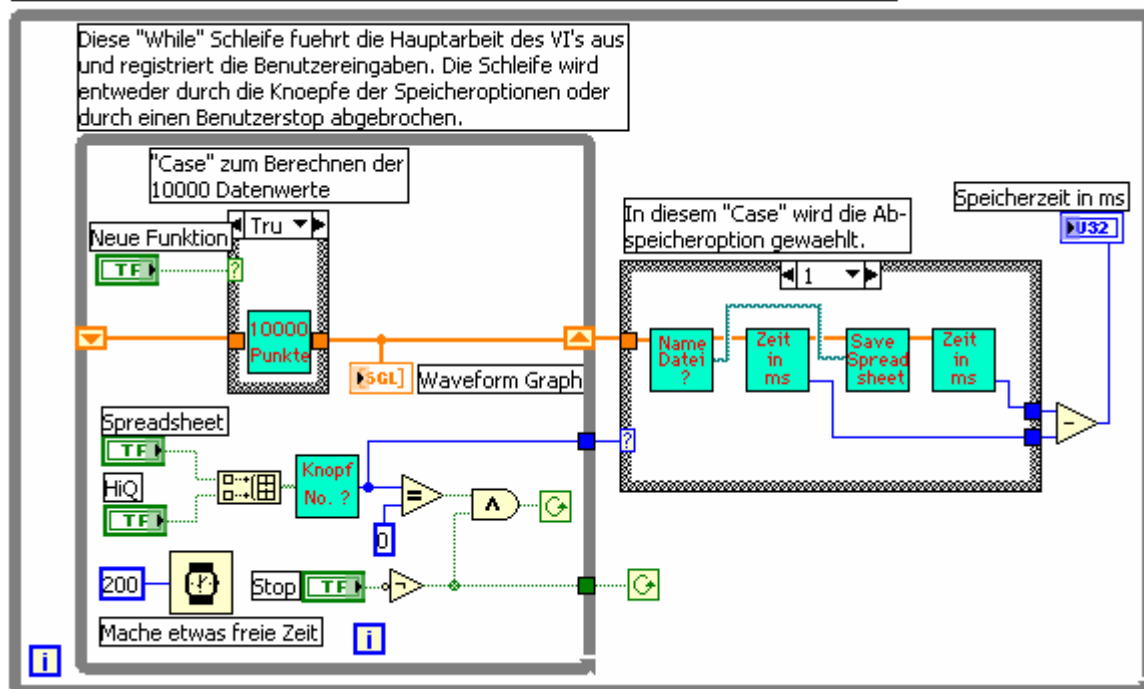


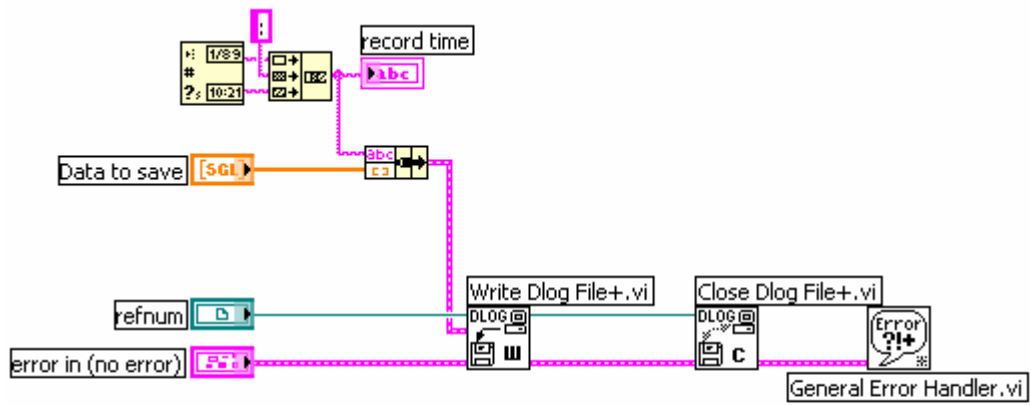


7.6.3 Aufgabe 3

Speichern Sie 10000 Datenpunkte unter den w ahlbaren Optionen von Textdatei oder LabVIEW eigener Datalog-Datei (Bin rformat) auf Harddisk ab. Messen sie die Ab-speicherzeiten und vergleichen Sie diese f ur die beiden Formate.

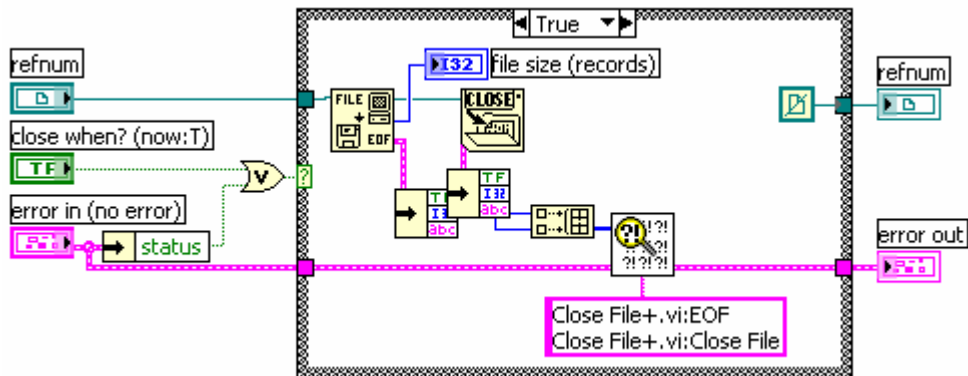
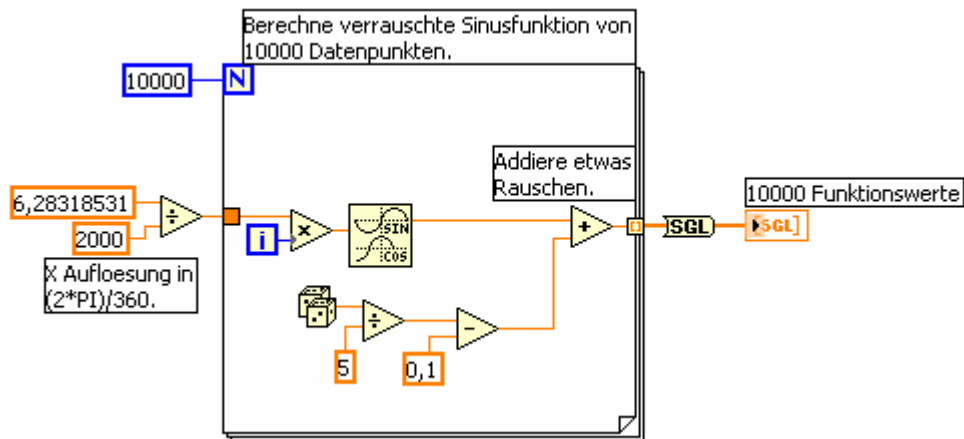
Diese "While" Schleife l asst das VI repetierend ablaufen bis es vom Benutzer gestopt wird.



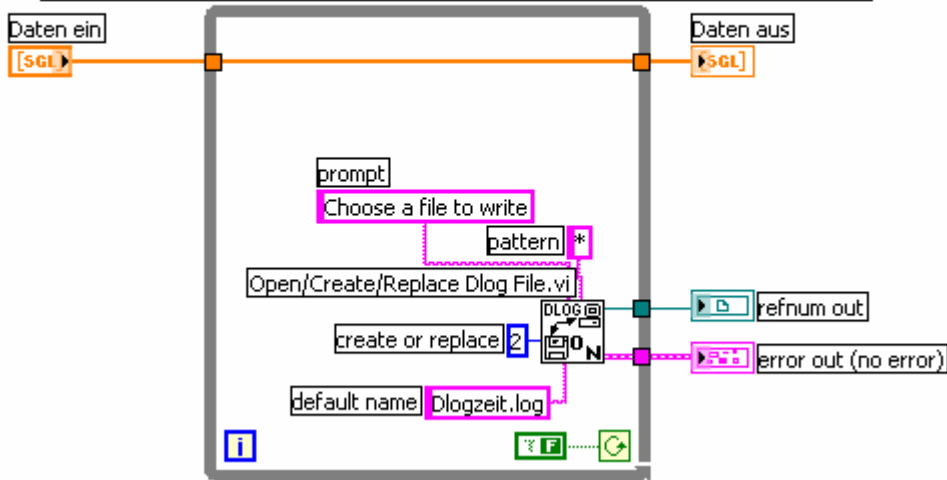


note

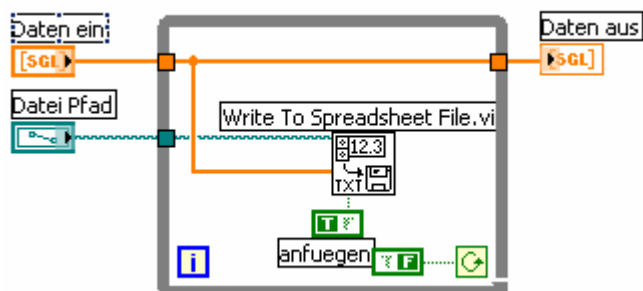
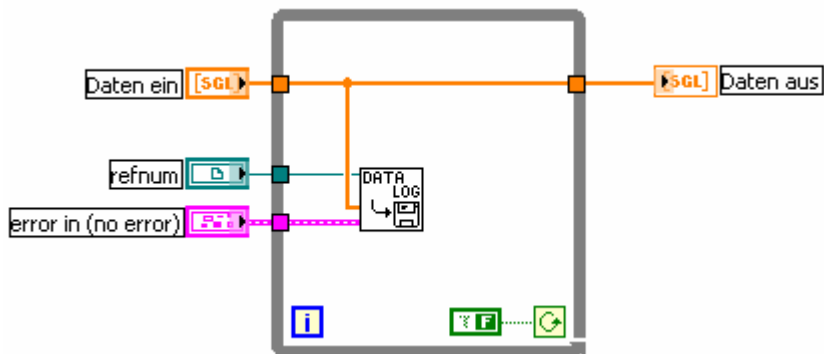
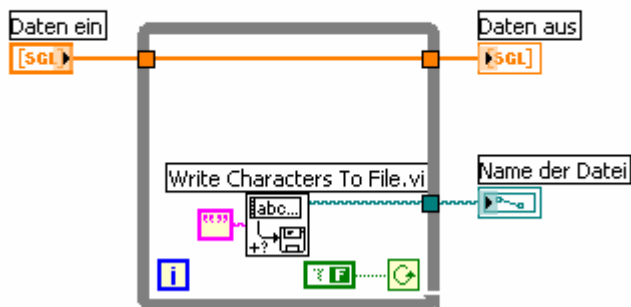
The three datalog (DLOG) file subVIs are not polymorphic and can be used only with the datalog record type shown here (that is, a cluster of a string and a 1D SGL array). If you want to use a different type, modify a copy of these VIs appropriately. These VIs are themselves modifications of VIs in the Functions:Utilities:File palette.

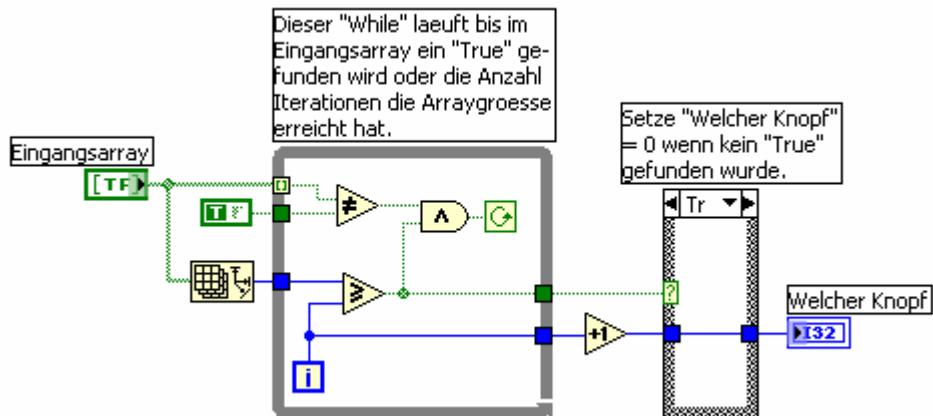


Dieses subVI braucht "Open/Create/Replace Dlog File.vi" um vom Benutzer durch eine Dialogbox ein Dateiname zu erhalten. Die Referenznummer dieser Datei kann zu einem spaetern Zeitpunkt zum Abspeichern der Datalogdatei benutzt werden.



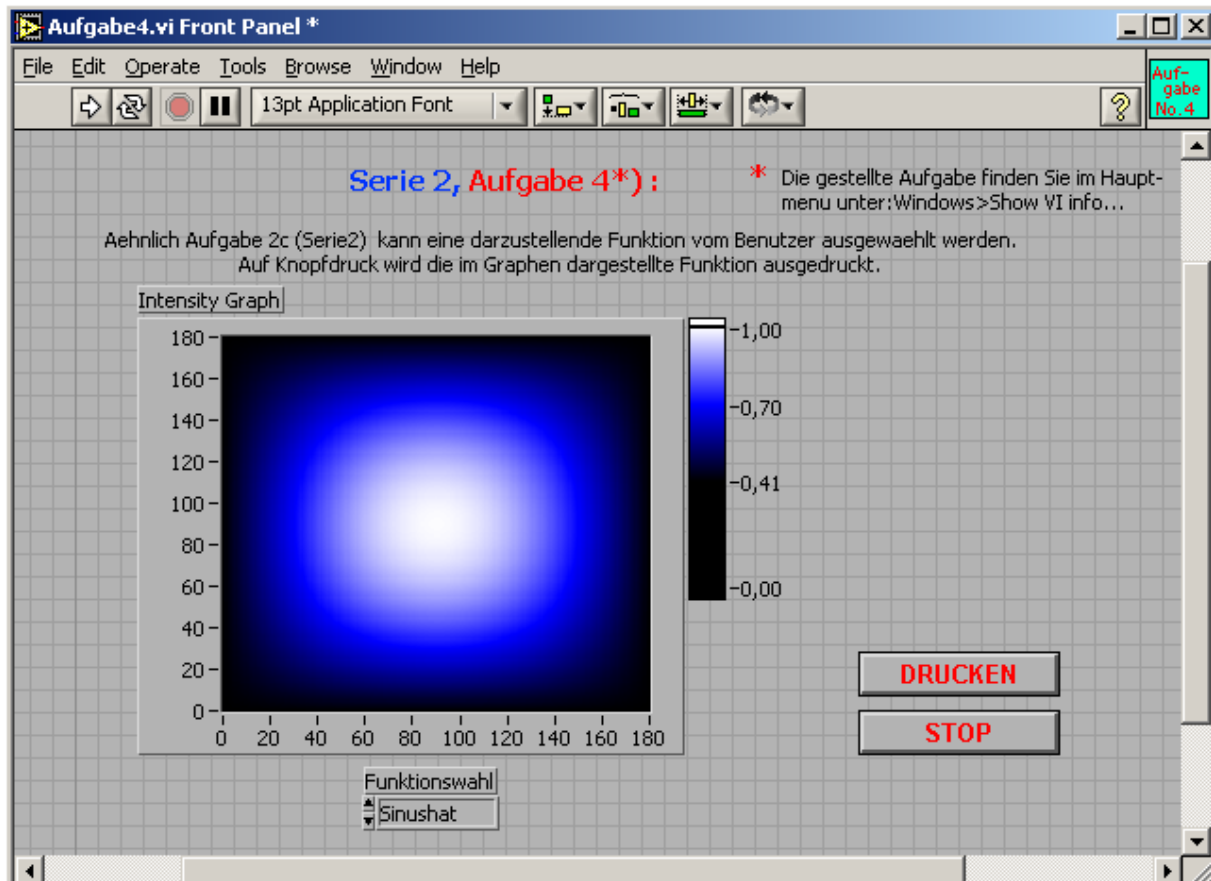
Dieses subVI braucht "Write Characters To File.vi" um vom Benutzer durch eine Dialogbox ein Dateiname zu erhalten. Der Pfad dieser Datei kann zu einem spaetern Zeitpunkt zum Abspeichern benutzt werden.





7.6.4 Aufgabe 4 – ohne L sung

Programmieren Sie ein Druck VI das einen von Ihnen generierten intensity graphen mit 3D Daten auf Knopfdruck ausdrucken l sst.



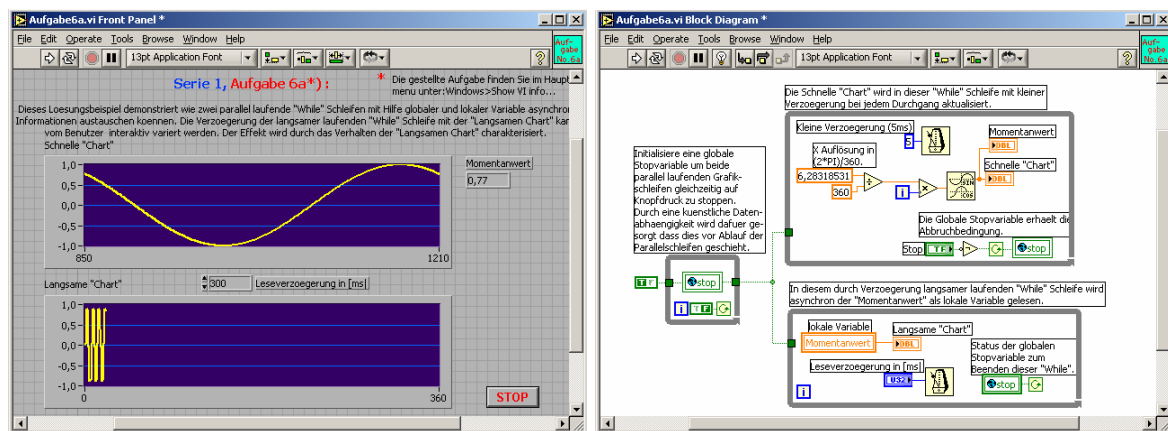
7.6.5 Aufgabe 5 – ohne Lösung

Bauen Sie ein Wahl VI welches drei gelungene VIs von Ihnen wahlweise auf Knopfdruck aufrufen und zeigen kann.

7.6.6 Aufgabe 6

Für fortgeschrittene Programmierer:

Generieren Sie eine Periodische Funktion mit großer Auflösung in der x-Achse in einer schnellen „While“-Schleife. Übermitteln Sie mittels einer lokalen oder globalen Variable jeden n-ten Datenpunkt an eine parallel laufende langsamere „While“-Schleife die eine waveform chart mit den übermittelten Datenpunkten laufend aktualisiert. Sorgen Sie dafür dass das VI von einem Benutzer „sauber“ beendet werden kann.



7.7 Serie 3 – ohne Lösungen

7.7.1 Aufgabe 1

Programmieren Sie eine zeitlich limitierte Datenerfassung und speichern Sie die Messdaten als ASCII-Zeichen im spreadsheet Format nach Abschluss der Messung ab.

7.7.2 Aufgabe 2

Programmieren Sie eine kontinuierliche Messdatenerfassung mit der Möglichkeit die Daten laufend im LabVIEW datalog Format abzuspeichern. Konsultieren Sie dazu die LabVIEW Beispielprogramme im LabVIEW Ordner unter examples: daq oder examples: file.

7.7.3 Aufgabe 3

Lesen Sie die Datei aus Aufgabe 2 ein und berechnen Sie aus den Daten ein Frequenzspektrum mit Hilfe einer LabVIEW FFT Funktion. Stellen Sie das Spektrum in einem Graphen dar.