Jül-4384

Zentralinstitut für Engineering, Elektronik und Analytik (ZEA)
Systeme der Elektronik (ZEA-2)

Entwicklung einer Real-Time basierten Aktuator-Ansteuerung in LabVIEW

Sebastian Völkel

Mitglied der Helmholtz-Gemeinschaft

JÜLICH
FORSCHUNGSZENTRUM

Entwicklung einer Real-Time basierten Aktuator-Ansteuerung in LabVIEW

Sebastian Völkel

Berichte des Forschungszentrums Jülich; 4384
ISSN 0944-2952
Zentralinstitut für Engineering, Elektronik und Analytik (ZEA)
Systeme der Elektronik (ZEA-2)
Jül-4384

(Master, Hochschule Niederrhein, 2015)

Vollständig frei verfügbar über das Publikationsportal des Forschungszentrums Jülich (JuSER)
unter www.fz-juelich.de/zb/openaccess

Forschungszentrum Jülich GmbH
Zentralbibliothek, Verlag
52425 Jülich
Tel.: +49 2461 61-5220
Fax: +49 2461 61-6103
E-Mail: zb-publikation@fz-juelich.de
www.fz-juelich.de/zb

Abstract

Within the DFG research group “active drag reduction by transversal surface waves” the Central Institute of Engineering, Electronics and Analytics, ZEA-2 – Electronic Systems, Forschungszentrum Jülich GmbH, deals with the subproject “development of a real-time actuator and sensor network” for closed loop controlled transversal surface waves on a flat aluminum sheet in wind-tunnel experiments. The research group pursues the approach to reduce drag by influencing the turbulent boundary layer.

In the framework of the subproject mentioned above the aim of this thesis is the development of a real-time actuation control for generating transversal surface waves with high spatial resolution. The actuation control is carried out by analog signal generation of a DAQ-System of National Instruments (NI). To establish user interactions with the DAQ-System a suitable communication method has to be implemented. During the actuation wave parameter changes lead to high frequency interferences. Using LabVIEW Real-Time smooth signal transitions within a given reaction time shall be realized in order to avoid high frequency interferences.

Zusammenfassung

Das Zentralinstitut für Engineering, Elektronik und Analytik ZEA-2 – Systeme der Elektronik der Forschungszentrum Jülich GmbH bearbeitet innerhalb der DFG-Forschergruppe „Aktive Widerstandsreduktion durch wellenförmige Oberflächenoszillation“ das Teilprojekt „Entwicklung eines echtzeitfähigen Aktuator-Sensor-Netzwerkes“ für die geregelte Erzeugung von transversalen Oberflächenwellen auf einer ebenen Platte im Windkanal. Dabei wird der Ansatz verfolgt, die turbulente Grenzschicht zur Reduktion des Reibungswiderstands durch die lokale Beeinflussung der wandnächsten kohärenten Turbulenzstrukturen zu verändern.

Ziel dieser Arbeit innerhalb des o.g. Teilprojektes ist es eine echtzeitfähige Ansteuerung der Aktuatoren für die Erzeugung der räumlich hoch aufgelösten transversalen Oberflächenwellen zu entwickeln. Die Ansteuerung der Aktuatoren erfolgt über die analoge Signalausgabe eines DAQ-Systems mit der Software LabVIEW Real-Time der Firma National Instruments (NI). Um Benutzerinteraktionen mit dem DAQ-System zu ermöglichen, soll eine geeignete Kommunikationsmethode implementiert werden. Während der Aktuierung des Systems treten bei Parameterwechsel der Wellenfunktionen hochfrequente Störungen auf. Mit LabVIEW Real-Time sollen glatte Signalübergange innerhalb einer vorgegebenen Reaktionszeit des Systems entwickelt und hochfrequente Störungen vermieden werden.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1. Einleitung

Die Reduktion des Energieverbrauchs und des Schadstoffausstoßes auf dem Verkehrssektor nimmt für sämtliche Volkswirtschaften an Bedeutung zu. Ein Beitrag zur Lösung der daraus abgeleiteten Aufgabe der Verringerung der Emission ist die Reduktion des Reibungswiderstands der Verkehrssysteme Flugzeug, Schiff und Zug.

Die DFG-Forschergruppe FOR1779 [1] konzentriert sich auf das Verkehrssystem Flugzeug, da hier alle maßgeblichen Strömungsbereiche, die auch für den Schienenverkehr und die Schifffahrt relevant sind, abgedeckt werden. Durch eine möglichst energieeffiziente aktive und passive Beeinflussung der Grenzschicht wird eine Abschwächung der Turbulenzproduktion infolge der Interaktion mit dem wandnahen Strömungsfeld untersucht (Abb. 1).

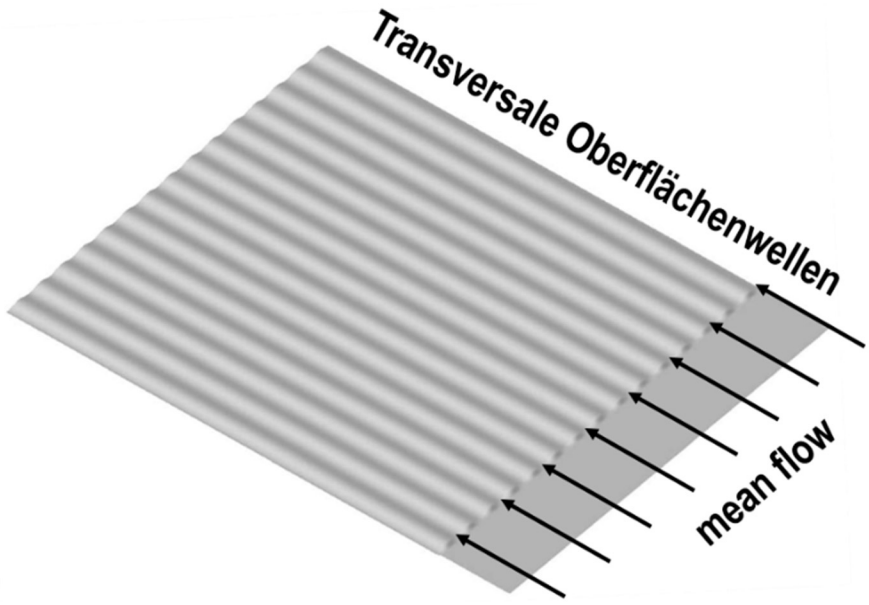

Abbildung 1: Interaktion mit dem wandnahen Strömungsfeld (mean flow) durch in Spannweitenrichtung laufende transversale Oberflächenwellen [2]

Die aktive Beeinflussung der Grenzschicht erfolgt durch in Spannweitenrichtung laufende transversale Oberflächenwellen, die passive Beeinflussung hingegen durch in der Oberfläche eingebrachte Riblet-Strukturen.

Innerhalb der DFG-Forschergruppe befasst sich das Teilprojekt „Entwicklung eines echtzeitfähigen Aktuator-Sensor-Netzwerkes“ (Abb. 2) mit

- der Entwicklung von Aktuator-Systemen zur Erzeugung von transversalen Oberflächenwellen
- der Entwicklung eines echtzeitfähigen Aktuator-Sensor-Netzwerks mit Schnittstellen zur Strömungsregelung und zu dem Aktuator-System
- der Implementierung einer zentralen Ansteuerung der Aktuatoren (Abb. 2)

Ziel dieses Teilprojektes ist die Einflussnahme auf die Grenzschicht einer umströmten Oberfläche mit flächig angeordneten Aktuatoren. Die Aktuatoren werden für die Erzeugung von Wellen quer zur Strömungsrichtung mit zeitlich sowie räumlich variabler Amplitude, Frequenz und Wellenlänge eingesetzt. Auf diese Weise können robuste Methoden zur Widerstandsreduzierung durch Strömungsregelung entwickelt werden. Die Entwicklung des Netzwerkes erfolgt modellbasiert mit Schnittstellen für die Strömungsregelung und die Aktuator-Ansteuerung sowie mit einer graphischen Benutzeroberfläche für die Durchführung von Parameterstudien im Windkanal. Die Schnittstellen erlauben schon im Rahmen der Modellentwicklung die Durchführung von „model in the loop“-Simulationen [2].

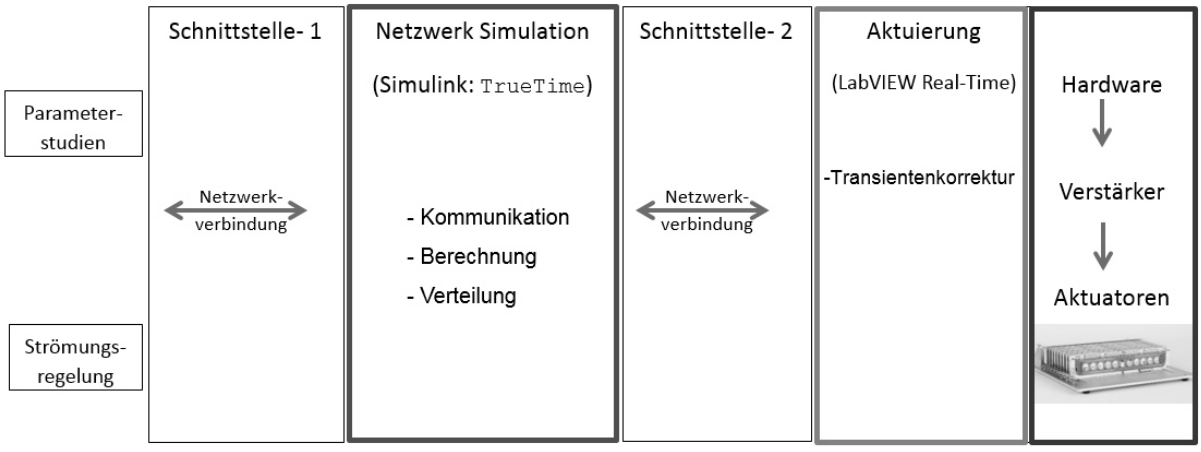

Abbildung 2: Übersicht des Teilprojektes " Entwicklung eines echtzeitfähigen Aktuator-Sensor-Netzwerkes " [1]

Diese Arbeit beschäftigt sich mit der Implementierung der zentralen Aktuator-Ansteuerung in LabVIEW Real-Time. Die bisher entwickelten Ideen und Strategien dienen als Grundlage.

2. Motivation und Aufgabenstellung

Zu Beginn der Masterarbeit „Entwicklung einer Real-Time basierten Aktuator-Ansteuerung in LabVIEW“ wurde eine Erweiterung des Anforderungsprofils der bestehenden Aktuator-Ansteuerung erarbeitet. In diesem Zusammenhang wurde festgelegt, dass über ein echtzeitfähiges DAQ-System der Firma National Instruments mindestens 20 Aktuatoren unter Verwendung der Wellenparameter *Frequenz, Amplitude, Phase, Offset, Ein-/Ausschalten* und *Wellenform* über analoge Ausgänge angesteuert werden sollen.

Zum Ansteuern aller Aktuatoren des Systems ist eine Generierung der Wellenform erforderlich, die mit einer maximalen Aktuierungsrate von 100 Hz und mindestens 200 Abtastwerten pro Periode realisiert wird. Das System soll dabei skalierbar bezüglich der Anzahl der Aktuatoren sein. Hinzu kommt, dass bei einem schnellen Wechsel der Wellenparameter hochfrequente Störungen während der Aktuierung des Systems auftreten. Um diese zu vermeiden, sollen intelligente Signalübergänge mit einer Reaktionszeit und Übergangszeit von unter einer Periode des Signals implementiert werden.

Durch das Anlegen einer Spannung an die Aktuatoren (s. Kap. 4: PCB Luftspulen-Aktuatoren) wird eine vertikal gerichtete Kraft auf die Oberfläche ausgeübt. Über Sensoren wird die Auslenkung der Aktuatoren gemessen. Damit die Ausgangswelle der gewünschten Wellenform entspricht, soll zu einem späteren Zeitpunkt des FOR1779-Projektes eine Regelung der Wellenform implementiert werden. Demnach besteht im Rahmen dieser Arbeit die zusätzliche Herausforderung, ein echtzeitiges synchrones Auslesen des Eingangs und Steuern des Ausgangs zu ermöglichen.

Die Benutzerinteraktion des DAQ-Systems erfolgt über eine Ethernet-Verbindung [3] mit einem echtzeitfähigen Netzwerkprotokoll [2]. Nach Verbindungsaufbau soll ein weiteres Kommunikationsprotokoll eingreifen das in Abhängigkeit der empfangenen Daten den Datenaustausch unter mehreren DAQ-Systemen durchführt.

3. Systemanalyse

Das Gesamtsystem besteht aus einer Benutzerschnittstelle, einen Ethernet-Netzwerk, einem DAQ-System der Firma National Instruments, einen Verstärker der Firma Thomann und einem selbst entwickelten PCB-Luftspulen-Aktuator-System des ZEA-2 (Abb. 3).

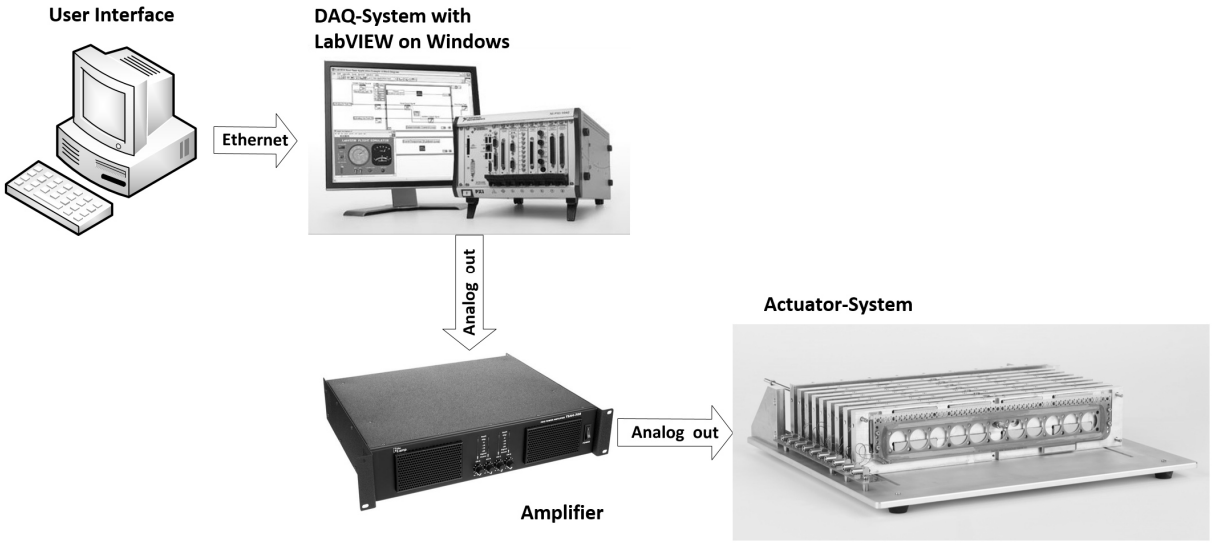

Abbildung 3: Gesamtaufbau des Systems zur Ansteuerung der Aktuatoren

Die Fernsteuerung des DAQ-Systems erfolgte über eine Ethernet-Schnittstelle und war für die Kommunikation mit einem einzelnen System konzipiert. Zur Ansteuerung der PCB-Luftspulen-Aktuatoren wurde bisher die Anwendungssoftware LabVIEW unter Windows genutzt.

Mit der entwickelten Software konnten 10 Kanäle / Aktuatoren mit jeweils einer Sinuswelle über die Wellenparameter *Frequenz, Amplitude, Phase, Offset* und *Ein-/Ausschalten* gesteuert werden. Jeder Kanal verfügte über eine Samplerate von 50.000 Hz. Zur Realisierung der Signalverläufe am Ausgang wurde eine Ringpuffermodifikation benutzt [2]. Dabei werden immer 50.000 Samples, also eine Sekunde des Sinus-Signalverlaufs vorberechnet, und die 50.000 Samples aus der letzten Berechnung gleichzeitig an den analogen Ausgang übergeben. Bei einem Parameterwechsel einer Sinuswelle mit hohen Frequenzen reagierte das System mit automatisierter Unterdrückung der hochfrequenten Transienten [2] erst nach maximal 2 Perioden des Signals.

Das war abhängig von der Pufferposition. Ereignete sich der Parameterwechsel am Anfang der Berechnung eines neuen Signalverlaufs mit 50.000 Samples, erfolgte die Reaktion auf Änderungen der Parameter erst nach Berechnung der Periode.

Die analogen Ausgangswerte des DAQ-Systems werden verstärkt und an die PCB-Luftspulen-Aktuatoren-System angelegt. Jeder analoge Ausgang steuert einen Aktuator des Systems. Das angesteuerte Aktuator-System (Abb. 4) des ZEA-2 für Windkanalversuche basiert auf dem Prinzip der Lorentzkraft [7].

Abbildung 4: Aufbau eines PCB-Luftspulen-Aktuator-Systems [6]

Die Luftspulen (Abb. 5) liegen zwischen Permanentmagneten mit unterschiedlich gerichteten Magnetfeldern.

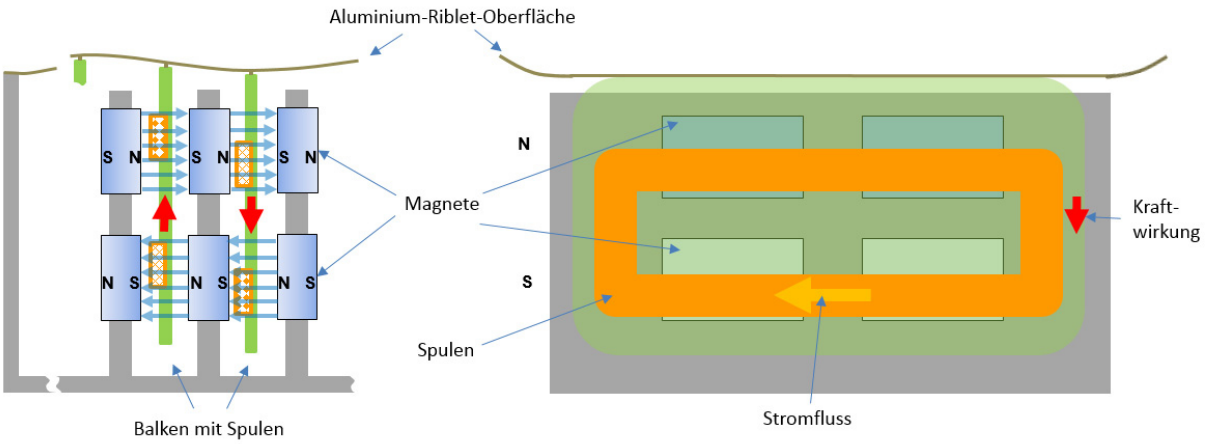

Abbildung 5: Aktuator-System mit Luftspulen-Aktuatoren [6]

Die Beeinflussung des Stromflusses durch Anlegen einer positiven oder negativen Spannung bewirkt eine vertikal gerichtete Kraft auf die Spulen. Diese wird über Balken an die Aluminium-Riblet-Oberfläche weitergegeben. Dadurch können die Aktuatoren eine transversale wandernde Oberflächenwelle erzeugen (Abb. 6).

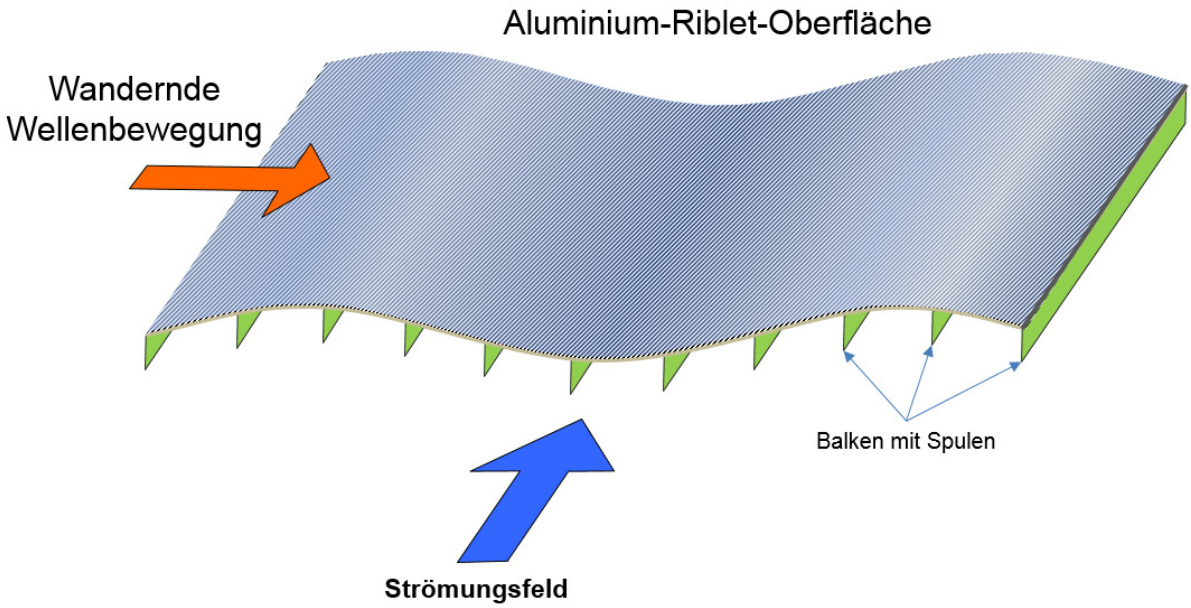

Abbildung 6: Erzeugung von wandernden Wellenbewegungen, durch Beeinflussung der Kraftwirkung auf die Balken, senkrecht zum Strömungsfeld [6]

Die Erzeugung der wandernden Wellenbewegung erfolgt durch phasenverschobene Signalverläufe der Aktuatoren (Abb. 7).

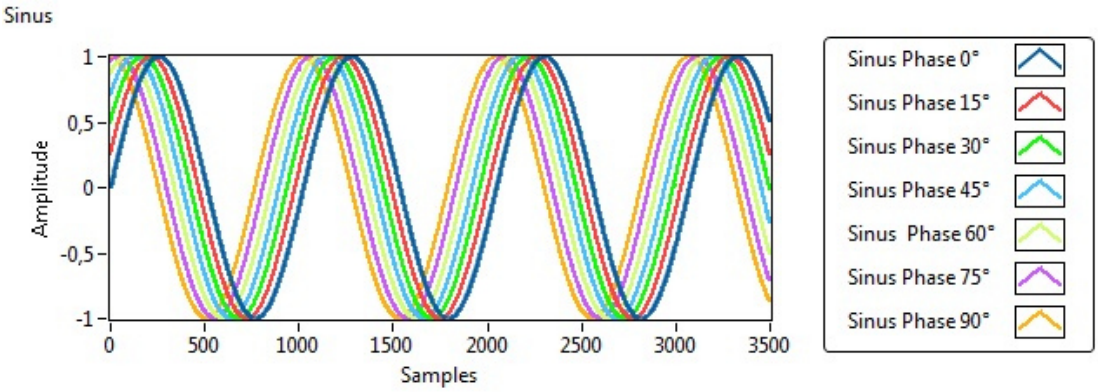

Abbildung 7: Phasenverschobene Signalverläufe der Aktuatoren

Mit Umstellung des Systems auf eine echtzeitfähige Software (LabVIEW Real-Time), können neue Anforderungen und Verbesserungen entwickelt werden.

Das Beinhaltet eine

- neue Kommunikationsmethode zum Datenaustausch unter mehreren DAQ-Systemen
- neue Ansätze einer Puffermodifikation von verschiedenen Wellenformen
- zeitnahe Reaktion des Systems auf einen Parameterwechsel unter einer Periode

4. Validierungsstrategie

Für die neuen Anforderungen der Aktuator-Ansteuerung des DAQ-Systems wurde eine Validierungsstrategie erstellt.

Kommunikationsprotokoll

Um ein Kommunikationsprotokoll für mehrere DAQ-Systeme zu entwickeln muss zunächst die Kommunikation mit einem einzelnen System entwickelt werden. Zur Analyse des entwickelten Kommunikationsprotokolls wird das freie Programm Wireshark [4] verwendet.

Echtzeitfähigkeit des Systems

Zum Testen der Echtzeitfähigkeit des Systems, kann durch zeitgesteuerte Schleifen [5] auf Ereignisse innerhalb einer fest definierten Zeit reagiert werden. Durch das Erfassen eines Ereignisses am analogen Eingang kann innerhalb dieser Zeit das analoge Signal am Ausgang ausgegeben werden. Durch ein Oszilloskop wird die zeitliche Differenz zwischen dem Anlegen einer Spannung am Eingang und dem Reagieren des Ausgangs gemessen.

Glatte Signalübergänge

Die Funktionalitäten der glatten Signalübergänge können in LabVIEW über einen Wellenform-Graphen simuliert und überprüft werden, da zur Ansteuerung der Aktuatoren die Signalverläufe an den analogen Ausgang gleichermaßen übergeben werden wie an den Graphen.

Messung der Zeitkonstanten

Zur Messung der Zeitkonstanten wird eine Sinuswelle mit einer bestimmten Frequenz am analogen Ausgang generiert. Über ein Oszilloskop wird dann die Periodendauer der erzeugten Welle gemessen.

5. Entwicklungsumgebung

In diesem Kapitel werden die verwendeten Methoden und Werkzeuge zum Ansteuern der PCB-Luftspulen-Aktuatoren vorgestellt.

Entwicklungssystem

Als Entwicklungssystem wird ein DAQ-System der Firma National Instruments (NI) mit verschiedenen Einschubkarten verwendet.

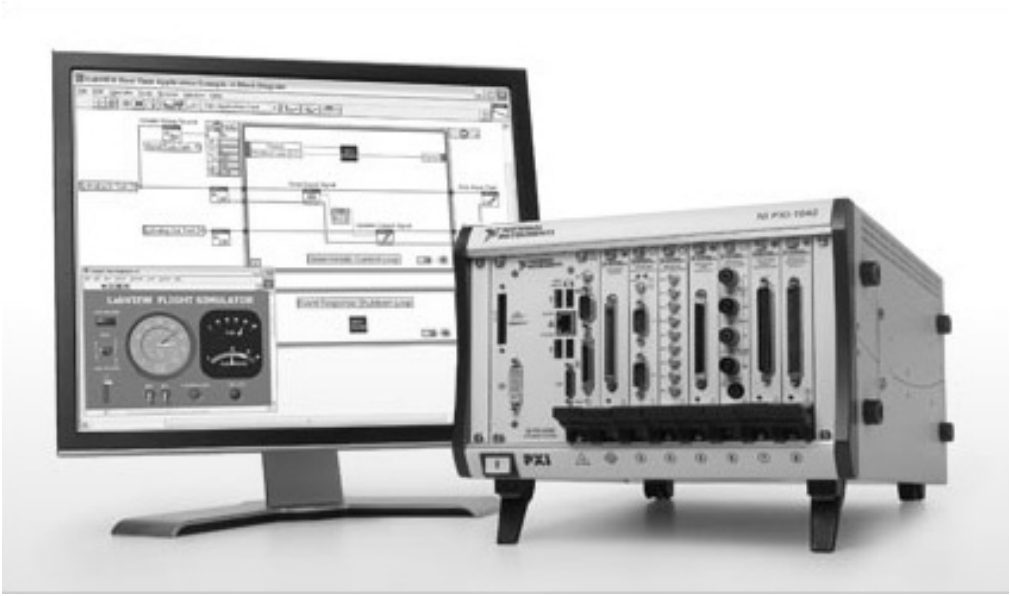

Abbildung 8: Beispiel eines DAQ-Systems mit verschiedenen Einschubkarten der Firma NI [8]

Das DAQ-System besteht aus einem PXIe-1065 Chassis mit folgenden Einschubkarten:

- PXIe-8133 Core i7-820QM 1.73 GHz Controller mit 4 Kernen [9]
- 4x PXI-6723 32 Kanal analoge Ausgänge DAQ [10]
- PXI-6254 32 Analoge & 48 Digitale Eingänge DAQ [11]

Der PXIe-8133 Controller wird als Echtzeitsystem mit LabVIEW Real-Time genutzt und verfügt unter anderem über eine Ethernet-Schnittstelle zur Kommunikation mit dem Netzwerk. Die analogen Ausgangskarten PXI-6723 dienen zur Ansteuerung der PCB-Luftspulen-Aktuatoren und die analoge/digitale Eingangskarte PXI-6254 zum Einlesen der Sensordaten.

Zum Datenaustausch untereinander wird eine Weiterentwicklung der PCI-Express-Technologie verwendet [12]. Dadurch ist eine Gesamtsystembandbreite von bis zu 8 GB/s pro Karte möglich. Wenn alle 32 Kanäle angesteuert werden, schränkt das die Samplerate laut Spezifikation der PXI-Karten [13] auf 50.000 Hz ein.

Entwicklungssoftware

Auf dem PXIe-8133 der Firma NI wurde das Echtzeitbetriebssystem [14] LabVIEW Realtime 2013 installiert. Beim Vergleich der beiden Betriebssysteme LabVIEW unter Windows und LabVIEW unter NI Real-Time OS wurden folgende Unterschiede festgestellt:

Tabelle 1: Vergleich: LabVIEW Windows/LabVIEW Real-Time

	LABVIEW (WINDOWS)	LABVIEW (NI Real-Time OS)
Ressourcen	Windows teilt sich Ressourcen mit LabVIEW (z.B. grafische Oberfläche)	LabVIEW verfügt über alle Ressourcen des Systems (keine grafische Oberfläche)
Schleifen	Schleifen können nur über einen Zeitstempel gesteuert werden (Max. 1 KHz Takt, min. 1 ms)	Zeitgesteuerte Schleifen (bis > 1Mhz Takt, min. 1ns) zur zeitkritischen Abarbeitung (z.B. innerhalb 20 µs) von Programmsequenzen.
Prozess	Aufgrund von Windows ist keine Prozesssteuerung möglich	Variable Prozesssteuerung von Schleifen
Multithreading	Passiv: LabVIEW teilt die Tasks auf, beinflussbar durch Programmierarchitektur	Aktiv: Tasks können selbst aufgeteilt und mit unterschiedlichen Prioritäten versehen werden

Anhand des Vergleichs ist abzusehen, dass eine signifikante Steigerung der Performanz durch LabVIEW Real-Time OS gegenüber einer Standard LabVIEW-Applikation erzielt werden kann. Dies ist insbesondere wichtig für die zeitkritischen Anforderungen an die Ansteuerung der Aktuatoren.

6. Systemkonzept

In diesem Kapitel werden die entwickelten Ansätze zur Lösung der Anforderungen zur Ansteuerung der Luftspulen-Aktuatoren vorgestellt.

6.1. Generierung der Wellenform

Zur Erzeugung der Wellenform wurde zuerst der Ansatz verfolgt, eine direkte Berechnung der Werte für alle Kanäle vorzunehmen. Da aber für jeden Kanal eine eigene Welle für jede Frequenz berechnet werden musste, bedeutete das einen großen Rechenaufwand für das DAQ-System.

Um diese prozessorbelastenden Berechnungen der Signalverläufe zu umgehen, wurde ein neuer Ansatz basierend auf der Abtastung einer einzigen abgespeicherten Wellenform von einer Periode untersucht. Aufgrund der gezeigten Einschränkungen der PXI-6723 Einschubkarten (Kap. 5: Entwicklungssystem) wurde dabei eine Abtastrate von 20 µs gewählt. Anschließend konnte eine Umsetzungstabelle (Tab. 2) für alle ganzzahligen Frequenzen bis 100 Hz erstellt werden.

Tabelle 2: Umsetzungstabelle (Lookup-Table)

Abtastrate:	20 µs	Speicher:	1.000.000 * 4 Bytes
Frequenz / Hz	Periode / s	Signal /Abtastrate	Datei/ Abtastung
1	1	50000	20
2	0,5	25000	40
3	0,33	16666,66667	60
100	0,01	500	2000
250	0,004	200	5000

Um ein fehlerfreies zeitdiskretes Signal durch die Abtastung des Speichers zu gewährleisten, muss der Speicher in Abhängigkeit der Abtastrate eine Größe von mindestens 100.000 Werten haben. Wird der Speicher größer oder die Abtastrate höher, können mehr Frequenzen im Nachkommabereich als zeitdiskretes Signal dargestellt werden.

Durch eine Vergrößerung des Speichers auf 1.000.000 Werten konnten Frequenzen bis zu einer Nachkommastelle erzeugt werden. Die Abtastung des Speichers ist abhängig von der benötigten Frequenz der Abtastrate und der Anzahl der Speicherwerte.

$$Abtastwert_{Speicher} = Frequenz * Abtastrate * Speicher_{Werte}$$

Unter der hier entwickelten Wellenform (Abb. 9) ist sogar eine maximale Aktuierungsrate von 250 Hz mit mindestens 200 Abtastwerten pro Periode möglich. Damit konnte die erste Anforderung an das System erfüllt werden.

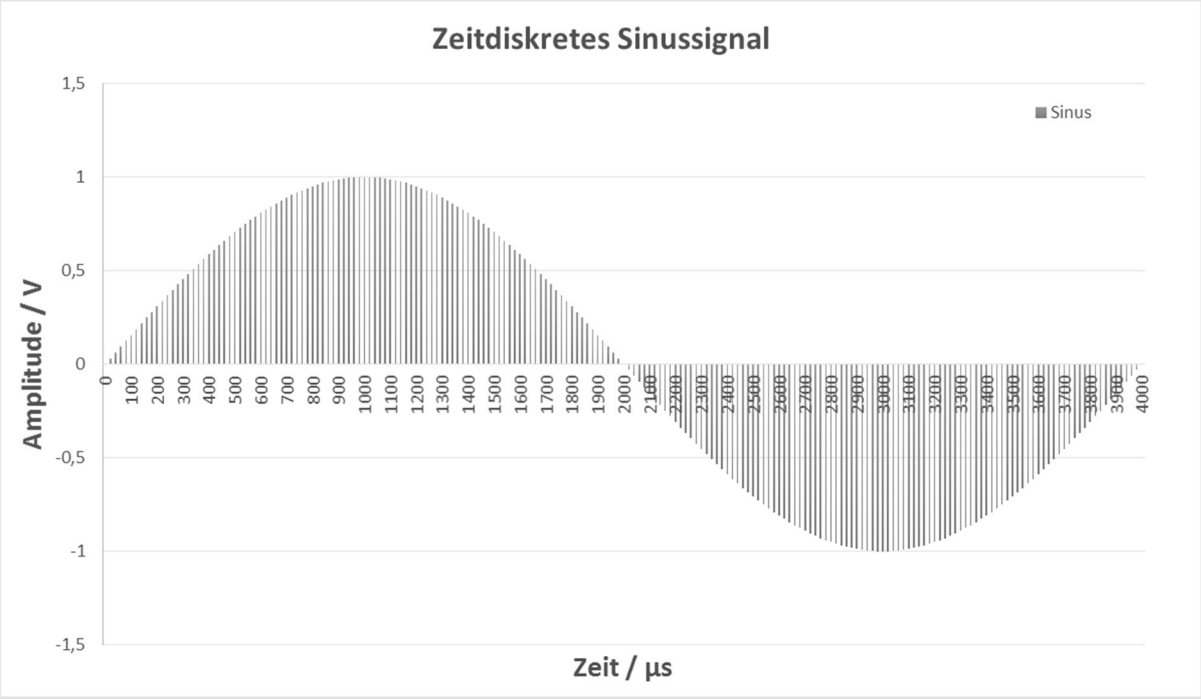

Abbildung 9: Zeitdiskretes Sinussignal mit 200 Abtastwerten je 20 µs und einer Periode von 4000 µs = 250 Hz

6.2. Glatte Signalübergänge

Die Änderungen der verschiedenen Wellenparameter führen zu unterschiedlichen Signalübergängen. Daher wird eine Änderung der Amplitude, Wellenform, und Frequenz anders behandelt als die der Phase oder des Offsets.

Hierfür wurden drei unterschiedliche Ansätze der Signalanpassung entwickelt. Zwei dieser Ansätze befassen sich mit der Änderung der Wellenparameter am Nullpunkt der aktuierten Wellenfunktion. Zum Nulldurchgang baut sich die Kraftwirkung des Magnetfelds der PCB-Luftspulen (s. Kap. 3) ab. Dadurch ist es energetisch am sinnvollsten die Wellenänderung zu diesem Zeitpunkt vorzunehmen.

Änderung von Amplitude, Wellenform und Frequenz

Bei den Wellenparametern Amplitude, Wellenform, und Frequenz, wird die Wellenänderung des Signals zum nächsten Nulldurchgang ausgeführt (Abb. 10).

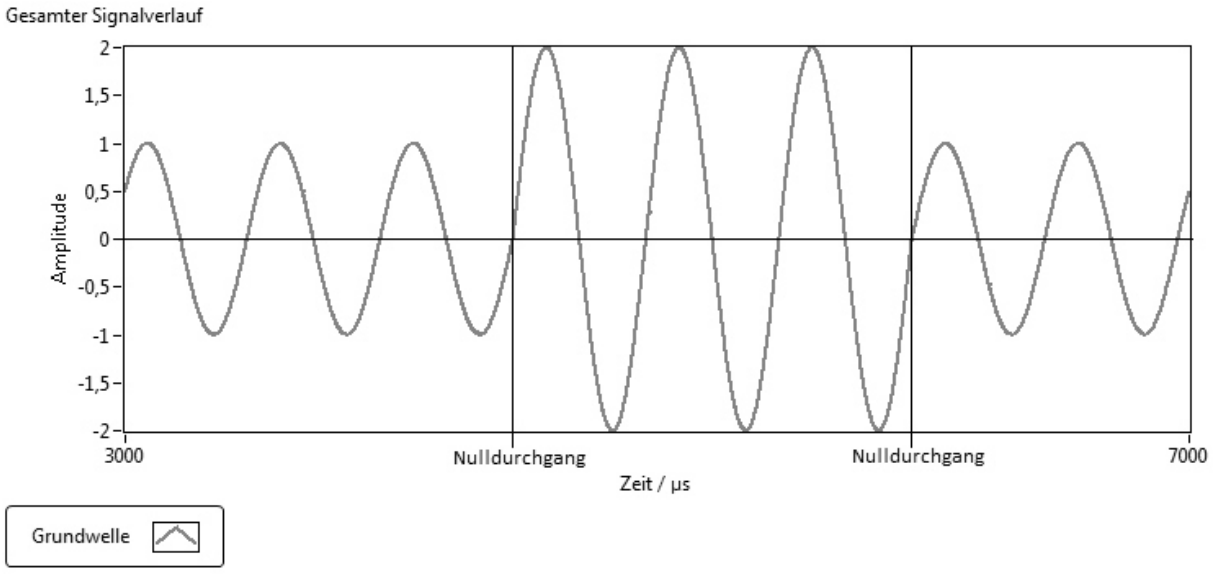

Abbildung 10: Amplitudenwechsel beim Nulldurchgang

Unter der Annahme, dass es sich bei dem Signalverlauf im Speicher um ein periodisches Signal handelt, kann der Nulldurchgang anhand des aktuellen abgetasteten Samples zur Laufzeit des Signals bestimmt werden (Abb. 11).

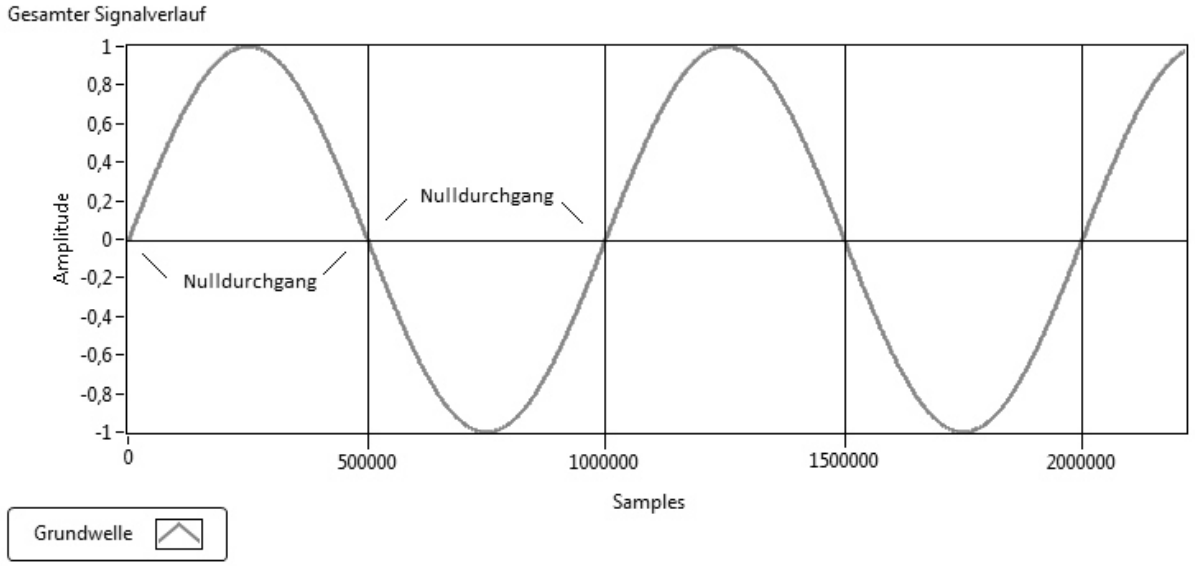

Abbildung 11: Nulldurchgänge einer abgespeicherten Sinuswelle von 1.000.000 Werten

Die Samples, bei denen sich die Nulldurchgänge befinden, lassen sich wie folgt definieren:

$$Sample = 0$$

$$Sample = \left(\frac{Speicher_{Werte}}{2}\right) - 1 = 49.999$$

$$Sample = Speicher_{Werte} - 1 = 999.999$$

Zu jeder halben Periode des Signalverlaufs kann eine Signalanpassung dieser Wellenparameter stattfinden.

Änderung der Phase

Wie im vorangehenden Kapitel erwähnt, besteht eine Signalperiode aus 1.000.000 Werten. In Abhängigkeit der Wellenfrequenz wird der Speicher mit den Signalwerten unterschiedlich abgetastet. Jeder Amplitudenwert des Signals hat ein eigenes Sample im Speicher der als Speicherpunkt P0 (Sample, Amplitudenwert) bekannt ist. Für weitere Untersuchungen der Phasenverschiebung gehen wir von einem abgespeicherten Sinussignals aus (Abb. 12).

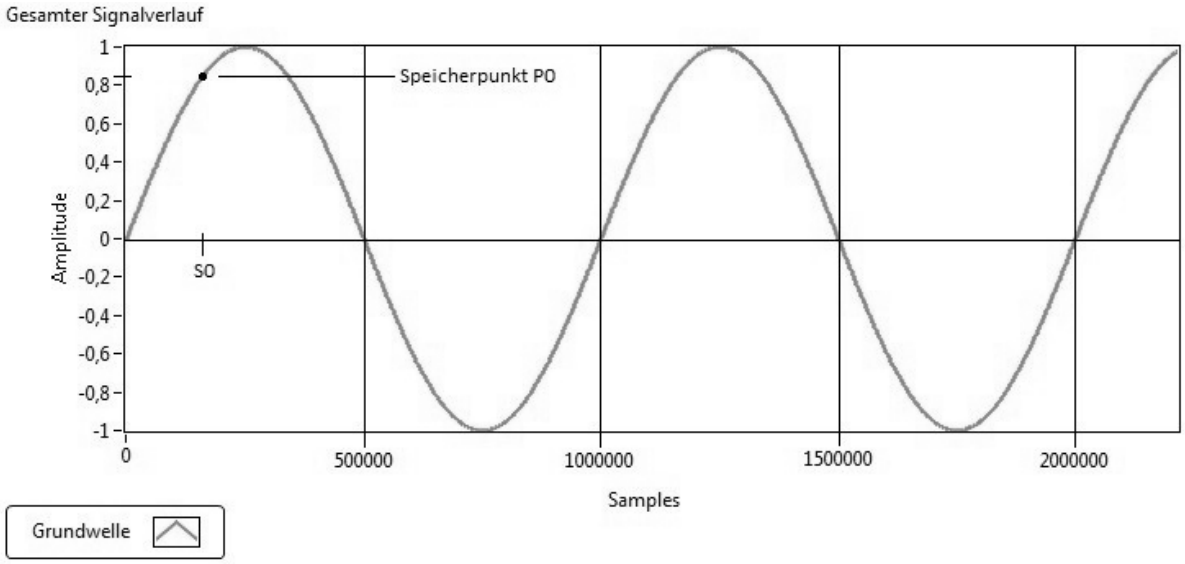

Abbildung 12: Position des Speicherpunktes des Sinussignals

Der Ansatz zur Änderung der Phase ohne hochfrequente Störungen durch schnelle Änderungen der Spannung und dessen Auswirkungen auf das Magnetfeld der PCB-Luftspulen zu verursachen, besteht in der Verschiebung des Nullpunktes zur nächsten halben Periode. Hierfür wird bei einer Änderung der Phase, der Sinuswert bei $\varphi = 60°$ der geänderten Phase berechnet.

$$x(t) = \sin\left(2 * \pi * f * t + \frac{\pi * \varphi}{180°}\right)$$

Beispiel:

$Zeit: t = 0$ $Phase: \varphi = 60°$

$$x(0) = \sin\left(2 * \pi * f * 0 + \frac{\pi * 60°}{180°}\right)$$

$$x(0) \cong 0{,}87$$

Die Abtastung des Speichers erfolgt so lange bis der absolute Wert der errechneten Phase erreicht ist. Durch die Eigenschaft eines periodischen und nullzentrierten Signals, dass der Phasenwert immer wieder positiv oder negativ auftritt, konnte folgende Annahme getroffen werden (Abb. 13):

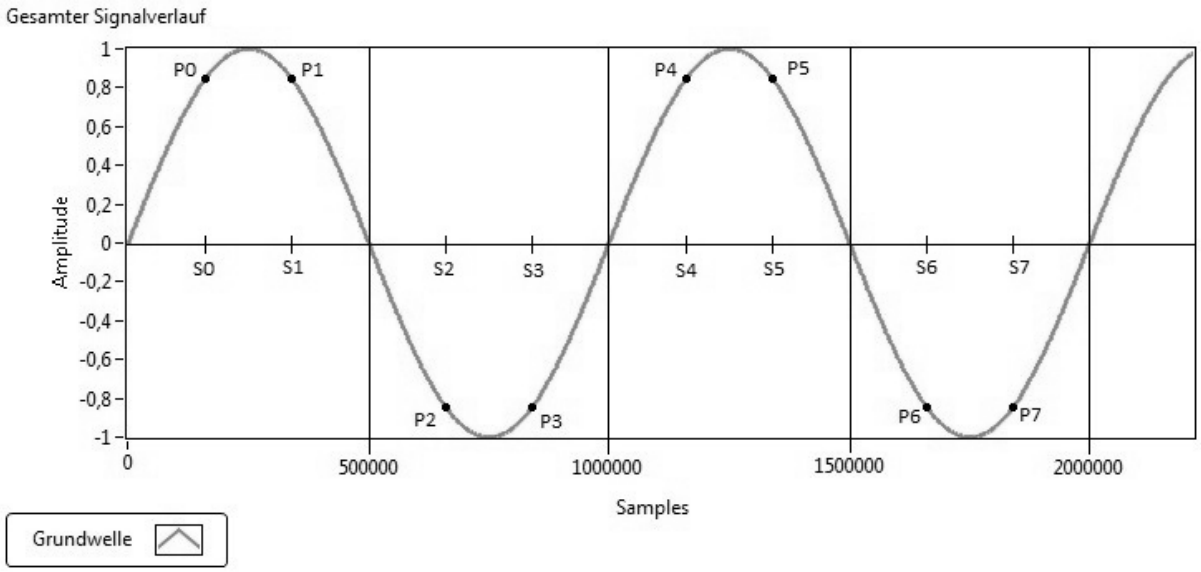

Abbildung 13: Immer wieder auftretender positiver oder negativer Phasenwert (P0,1…7)

Wenn der Phasenwert am Speicherpunkt P0 (S0, 0,87) auftritt, und die Phasenänderung somit φ <= 60° ist, dann wird der Nulldurchgang vom Speicherpunkt der nächsten halben Periode an den Speicherpunkt P3 (S3, -0,87) verschoben. Daraus folgt eine schnellere Abtastung des Speichers und der Nulldurchgang wird früher erreicht (Abb. 14).

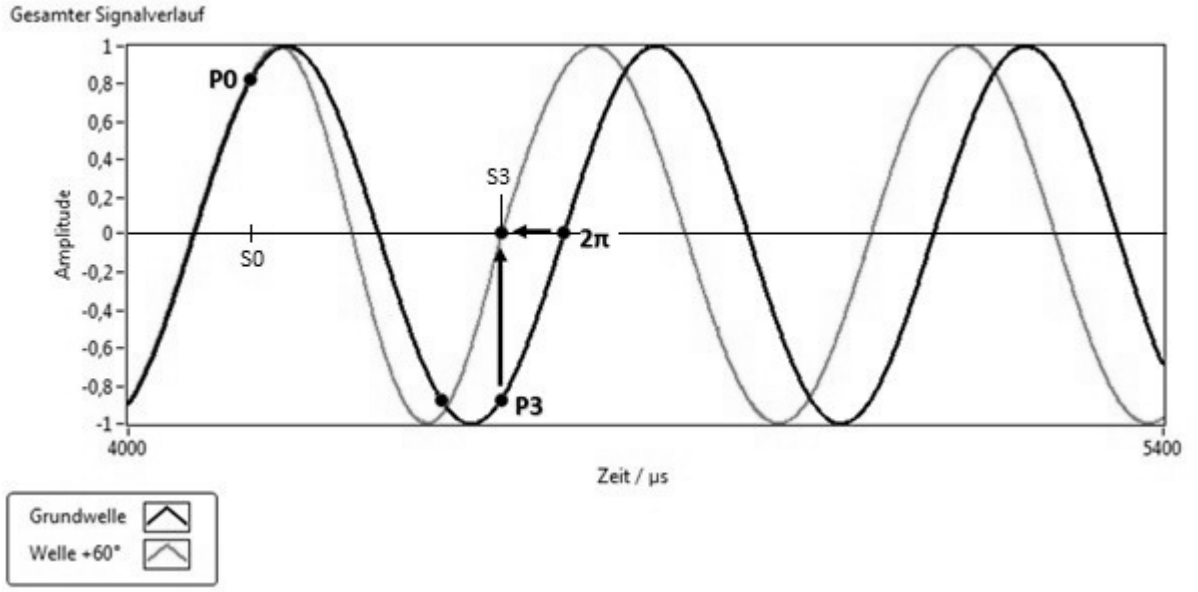

Abbildung 14: Verschiebung des Nulldurchgangs an den Speicherpunkt P3.

Zur Berechnung des neuen Abtastwertes müssen die Position des Speicherpunktes P0, die aktuelle Abtastrate und die Position des Speicherpunktes P3 bekannt sein.

Speicherpunkt P0:

Der Speicherpunkt P0 ist die Position des errechneten Phasenwertes und somit bekannt.

Speicherpunkt P3:

Die Position des Speicherpunktes P3 kann über den Speicherpunkt P0 bestimmt werden.

$$2\,\pi \rightarrow 360° \rightarrow Speicher_{Werte} \rightarrow 1000000\,Werte$$

$$P3 = Speicher_{Werte} - P0$$

Aktuelle Abtastrate:

Die aktuelle Abtastrate ist der errechnete Abtastwert der erzeugten Wellenform (Kap. 6.1).

$$Abtastwert_{Speicher} = Frequenz * Abtastrate * Speicher_{Werte}$$

Neue Abtastrate:

Über den Speicherpunkt P0 und der aktuellen Abtastrate können die noch verbleibenden Abtastungen vom Speicherpunkt P0 bis zum Ende des Speichers berechnet werden.

$$Anzahl\ der\ Abtastungen_{P0 \rightarrow Ende\ des\ Speichers} = \frac{Speicher_{Werte} - P0}{Abtastwert_{Speicher}}$$

Um die Differenz der Abtastungen des Speicherpunktes P0 und die des Speicherpunktes P3 zu ermitteln, werden die Abtastungen vom Speicherpunkt P3 bis zum Ende des Speichers berechnet.

$$Anzahl\ der\ Abtastungen_{P3 \rightarrow Ende\ des\ Speichers} = \frac{Speicher_{Werte} - P3}{Abtastwert_{Speicher}}$$

Für alle Phasenverschiebungen von φ <= 180° erfolgt die Nulldurchgangsverschiebung in negativer x-Richtung.

$$Anzahl\ der\ Abtastungen_{\varphi <= 180°}$$

$$= Anzahl\ der\ Abtastwerte_{P0 \rightarrow Nulldurchgang} - Anzahl\ der\ Abtastwerte_{P3 \rightarrow Nulldurchgang}$$

Für alle Phasenverschiebungen von φ >= 180° erfolgt die Nulldurchgangsverschiebung in positiver x-Richtung.

$$Anzahl\ der\ Abtastungen_{\varphi >= 180°}$$

$$= Anzahl\ der\ Abtastwerte_{P0 \rightarrow Nulldurchgang} + Anzahl\ der\ Abtastwerte_{P3 \rightarrow Nulldurchgang}$$

Durch die errechnete Anzahl der Abtastungen kann nun ein neuer Abtastwert in Abhängigkeit der aktuellen Position vom Speicherpunkt P0 ermittelt werden.

$$Neuer\ Abtastwert_{Speicher} = \frac{Speicher_{Werte} - P0}{Anzahl\ der\ Abtastungen_{\varphi<=180°\ or\ \varphi>=180°}}$$

Beispiel:

$f = 100\ Hz = 100\frac{1}{s}, \qquad Samplerate: 20\ \mu s, \qquad \varphi = +60°, \qquad P0 = 164500$

$Abtastwert_{Speicher} = Frequenz * Abtastrate * Speicher_{Werte}$

$Abtastwert_{Speicher} = 100\ \frac{1}{s} * 0{,}00002\ s * 1000000 = 2000$

$P3 = Speicher_{Werte} - P0 = 1000000 - 164500 = 835500$

$$Anz.der\ Abtast._{P0\rightarrow Nulldurchgang} = \frac{Speicher_{Werte} - P0}{Abtastwert_{Speicher}} = \frac{1000000 - 164500}{2000} = 417{,}75$$

$$Anz.der\ Abtast._{P3\rightarrow Nulldurchgang} = \frac{Speicher_{Werte} - P3}{Abtastwert_{Speicher}} = \frac{1000000 - 835500}{2000} = 82{,}25$$

$\varphi = +60° \leq 180°$

$Anzahl\ der\ Abtastungen_{\varphi <= 180°}$

$= Anzahl\ der\ Abtastwerte_{P0\rightarrow Nulldurchgang} - Anzahl\ der\ Abtastwerte_{P3\rightarrow Nulldurchgang}$

$= 417{,}75 - 82{,}25 = 335{,}50$

$$Neuer\ Abtastwert_{Speicher} = \frac{Speicher_{Werte} - P0}{Anzahl\ der\ Abtastungen_{\varphi\leq 180°\ or\ \varphi\geq 180°}}$$

$$= \frac{1000000 - 164500}{335{,}50} = 2490{,}31 \ldots \approx 2490$$

Unter der Voraussetzung, dass nach dem Erreichen des Nulldurchgangs die Differenz zum alten Abtastwert berücksichtig wird, können die Laufzeitfehler aufgrund der Nachkommastellen vernachlässigt werden.

$$\begin{aligned} Differenz_{Neu \rightarrow Alt} &= \left(Neuer\ Abtastwert_{Speicher} * Anzahl\ der\ Abtastungen\right) \\ &\quad -(Speicher_{Werte} - P0) + \ Alter\ Abtastwert_{Speicher} \\ &= (2490 * 335{,}5) - (1000000 - 164500) + 2000 \\ &= \ 835395 - 835500 + 2000 = 1895 \end{aligned}$$

Änderung des Offsets

Der Ansatz zum Signalübergang des Offsets beruht auf einer Rampenfunktion. Hierbei werden die Änderungen des Offsets in negativer und positiver y-Richtung aufgeteilt.

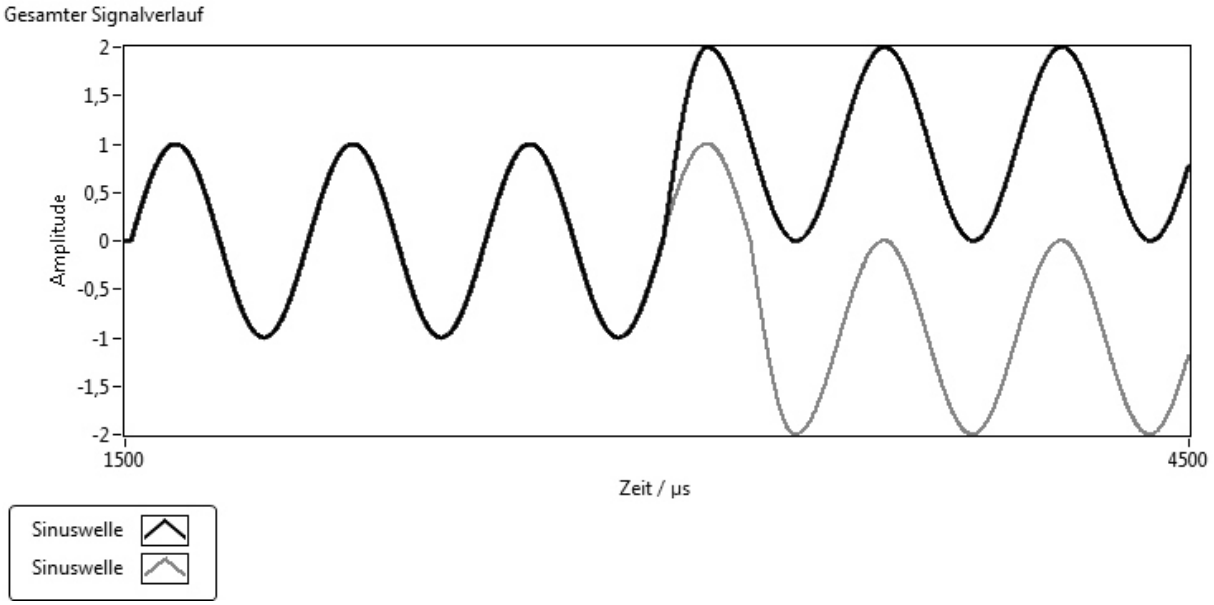

Abbildung 15: Negative (Grau) und positive (Schwarz) Offsetänderung zum selben Zeitpunkt

Positive Offsetänderung:

Bei der positiven Offsetänderung des Signals wird bis zum Nulldurchgang in positiver Richtung gewartet.

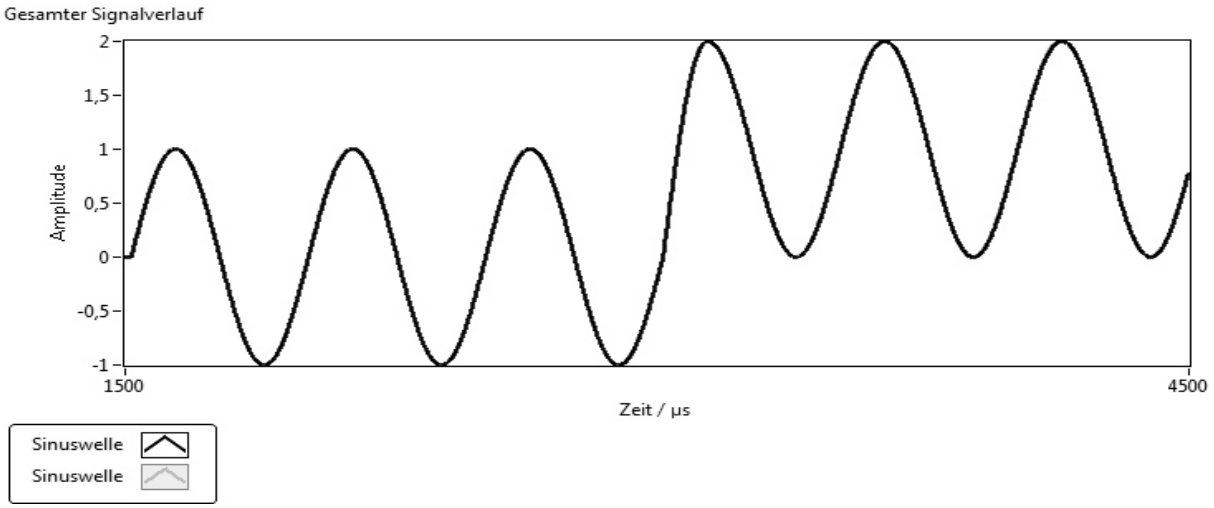

Abbildung 16: Offsetänderung nach Nulldurchgang in positiver Richtung

Der Signalwert der Sinuswelle wird mit der Addition des Offsetwertes und der Amplitude multipliziert.

$$Sinuswert = Speicherwert * (Amplitude + Offset)$$

Die Steigung der Welle nimmt solange zu, bis der höchste Sinuswert des Signals zum Zeitpunkt von $T = \frac{\pi}{2}$ auftritt. Danach wird der Sinuswert über die ursprüngliche Signalfunktion berechnet.

$$Sinuswert = (Speicherwert * Amplitude) + Offset$$

Negative Offsetänderung:

Um die negative Offsetänderung des Signals durchzuführen, wird bis zum Nulldurchgang in negativer Richtung gewartet.

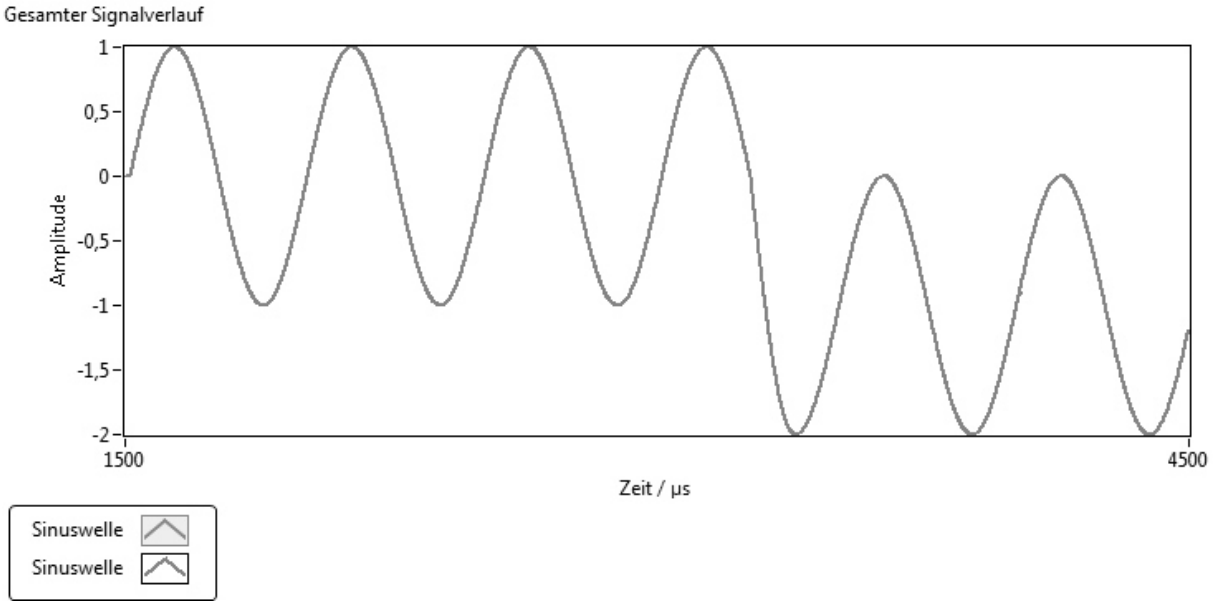

Abbildung 17: Offsetänderung nach Nulldurchgang in negativer Richtung

Die Steigung der Welle erfolgt über dieselbe Signalfunktion der positiven Offsetänderung. Nur der Übergang zur ursprünglichen Signalfunktion verändert sich zum Zeitpunkt des niedrigsten Sinuswertes bei $T = \frac{2\pi}{2}$.

6.3. Kommunikationsprotokoll

Um mehrere DAQ-Systeme zu verbinden, wurde ein Kommunikationsprotokoll auf Basis eines Master/Slave-Prinzips [15] über die Kombination eines UDP- [16] und TCP/IP-Protokolls [17] gewählt. Dabei hat nur ein DAQ-System (Master) das Recht, die übertragenen Daten der Benutzerschnittstelle entgegenzunehmen und an andere DAQ-Systeme weiterzureichen.

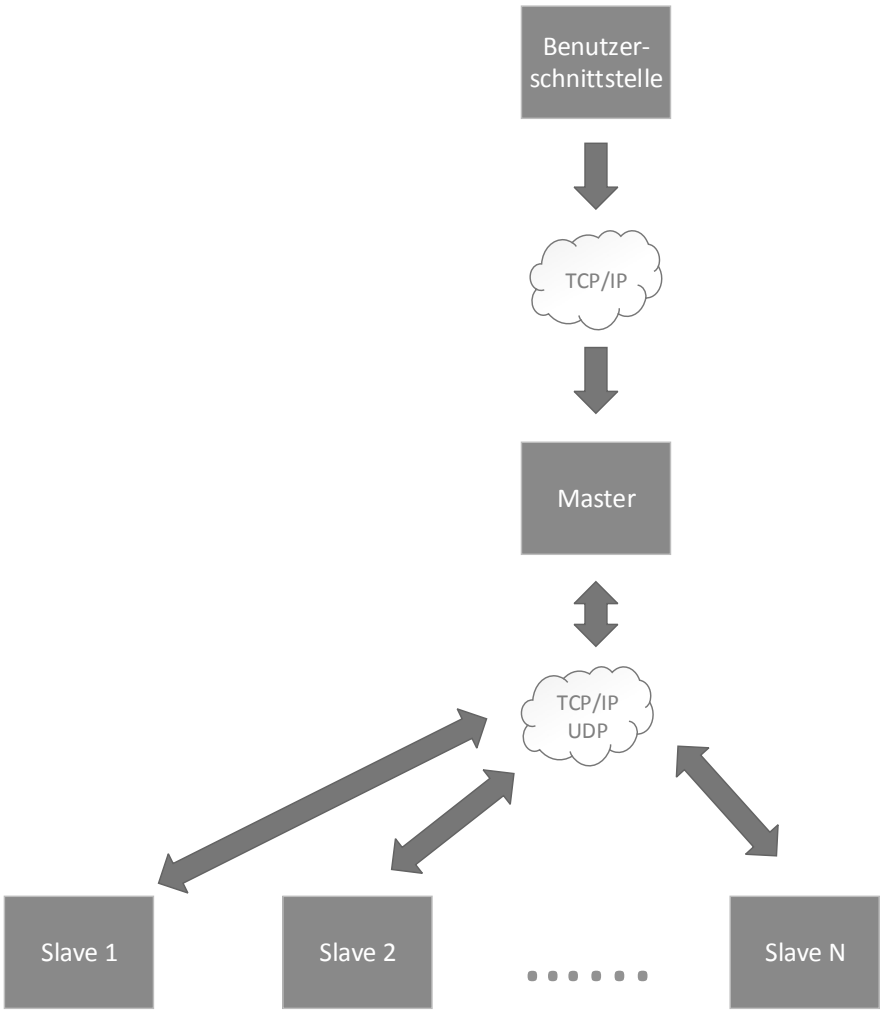

Abbildung 18: Kommunikations-Topologie

Die Benutzerschnittstelle stellt eine TCP/IP-Verbindung zu einem DAQ-System her. In diesem Moment bestätigt das DAQ-System die Kommunikation und wird zum Master-System. Das Master-System sendet ein UDP-Broadcast übers Netzwerk und wartet auf Antwort der anderen DAQ-Systeme. Nach der Identifizierung der anderen DAQ-Systeme, verbindet sich das Master-System mit den anderen DAQ-Systemen über ein TCP/IP-Protokoll, um den Datenaustausch der Benutzerschnittstelle zu übernehmen.

7. Systemimplementierung

Nach der Erstellung des Systemkonzeptes wurde das Hauptprogramm in unabhängig voneinander programmierbare Teilprogramme aufgeteilt (Abb. 19).

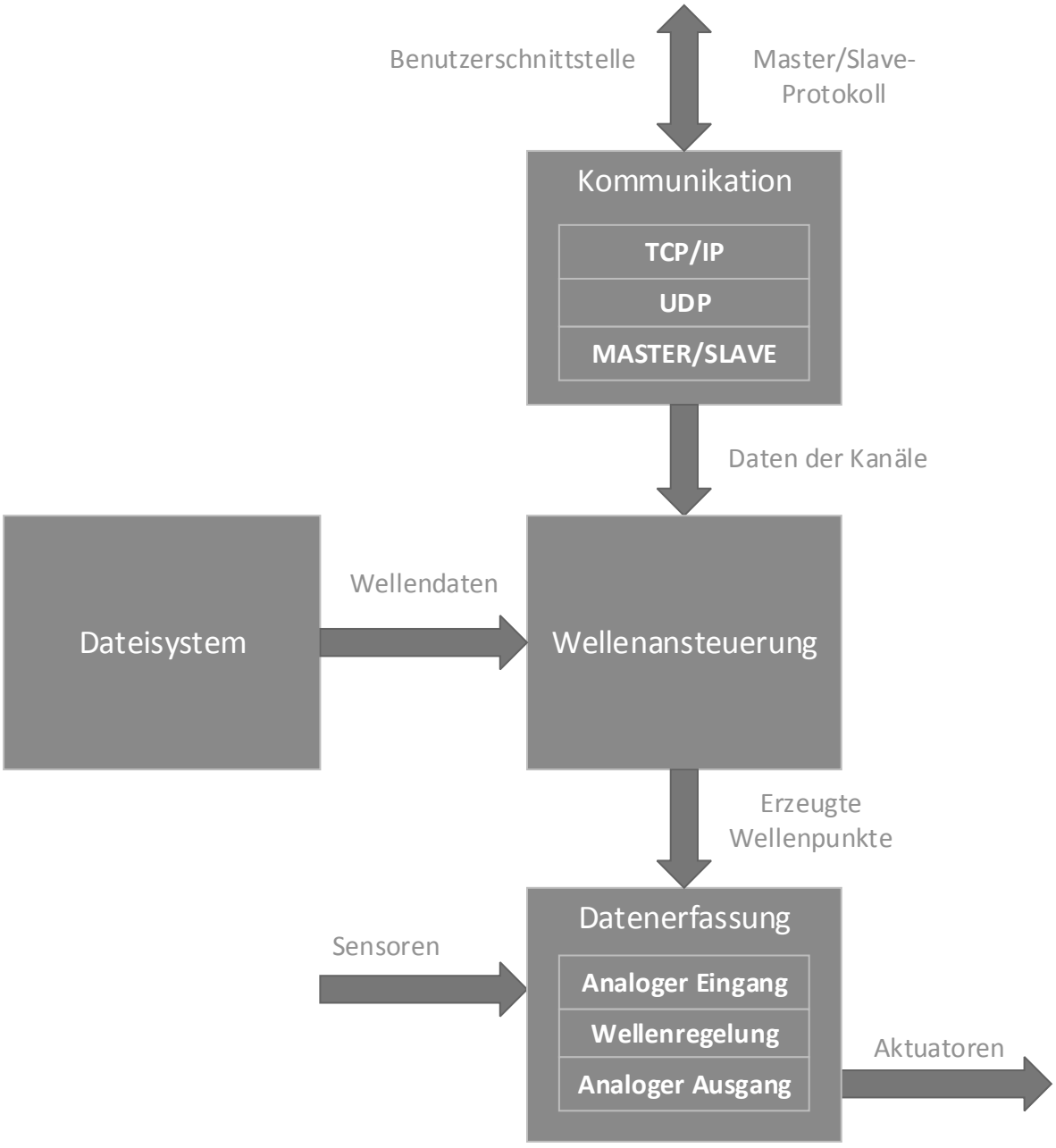

Abbildung 19: Aufbau des Programms zur Aktuator-Ansteuerung

Die beiden Teilprogramme *Kommunikation*, und *Wellenansteuerung* basieren auf der Architektur von Zustandsautomaten. Diese Architektur wurde deshalb ausgewählt, da sie sehr flexibel, leicht lesbar und erweiterbar ist.

Der Programmteil *Kommunikation* dient ausschließlich zum Datenaustausch mit der Benutzerschnittstelle oder anderen DAQ-Systemen im Netzwerk (Kapitel 6.3). Mit den beiden Verbindungen werden die Daten zur Ansteuerung der Aktuatoren übertragen oder empfangen und an das Teilprogramm *Wellenansteuerung* übermittelt.

Das Teilprogramm *Dateisystem* liest einmalig bei der Initialisierung des DAQ-Systems die erzeugten Wellenformen (Kapitel 6.1) für eine Sinus- und Dreieckwelle aus. Die ausgelesenen Wellendaten aus diesen Dateien werden dem Teilprogramm *Wellenansteuerung* zur Verfügung gestellt.

Der Programmteil der *Wellenansteuerung* realisiert die glatten Signalübergänge (Kapitel 6.2) bei Parameterwechsel der Kanäle. Zusätzlich werden spezielle Ein-/ und Ausschaltverhalten der Aktuatoren berücksichtigt. Danach wird für jeden Kanal ein Wellenpunkt erzeugt und an das Programm *Datenerfassung* geschickt.

Das Teilprogramm *Datenerfassung* basiert auf einer zeitgesteuerten Schleife. Dadurch ist eine genaue Einhaltung der Abtastrate (Kapitel 6.1) des analogen Ein- und Ausgangs möglich. Zusätzlich ermöglicht die zeitgesteuerte Schleife eine echtzeitfähige Vorbereitung der Wellenregelung.

Die Teilprogramme *Kommunikation*, *Wellenansteuerung* und *Datenerfassung* laufen in parallel programmierten Schleifen ab. Das Programm *Dateisystem* wird einmal durchgeführt. Als Daten-Schnittstellen zwischen den Teilprogrammen dienen gemeinsame Speicher.

7.1. Speicherverwaltung

Zum Datenaustausch zwischen den Teilprogrammen *Kommunikation*, *Wellenansteuerung* und *Datenerfassung* wird als Datenstruktur eine Warteschlange [18] (Queue) verwendet. Die Daten aus dem Teilprogramm *Dateisystem* werden einmalig in den Arbeitsspeicher des DAQ-Systems geladen und stehen dem Teilprogramm *Wellenansteuerung* dauerhaft zur Verfügung. Zusätzlich benötigt das Programm *Wellenansteuerung* einen Zwischenspeicher für Kanalinformationen vor Änderung der Wellenparameter. Jede Queue erhält bei der Initialisierung des DAQ-Systems einen eigenen Namen und die Programme greifen lesend und schreibend auf diese Datenstruktur zu (Abb. 20).

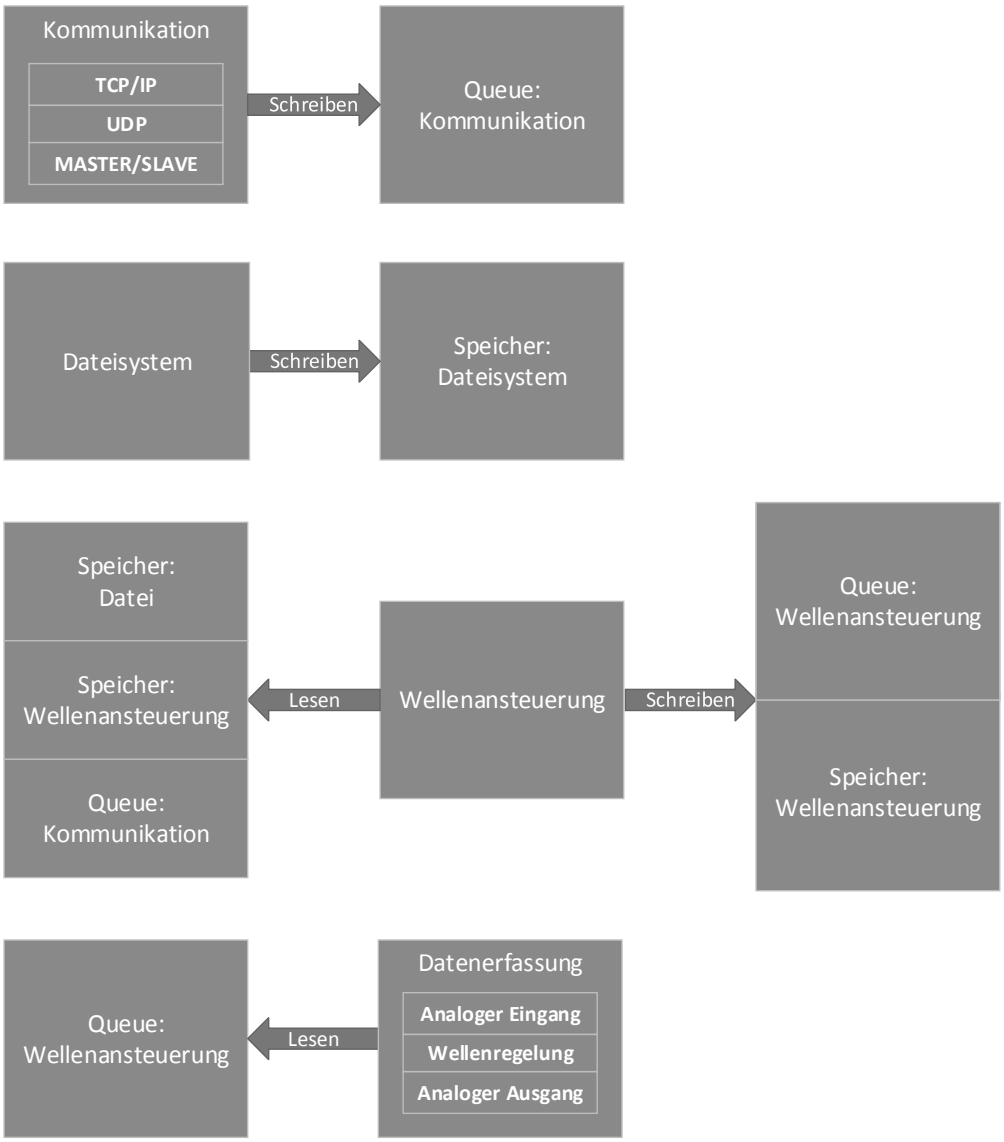

Abbildung 20: Speicherzuordnungen der Programme

7.2. TCP/IP-Zustandsautomat

Entsprechen der Aufgabenstellung (s. Kap. 2) sollte eine Strategie zur Verbindung mehrere Systeme erarbeitet werden. Hierfür wurde zuerst die Kommunikation zwischen Benutzerschnittstelle und einem einzelnen Real-Time System über eine TCP/IP-Verbindung (Abb. 21) auf der Basis eines Zustandsautomaten [19] mit insgesamt drei Zuständen entwickelt.

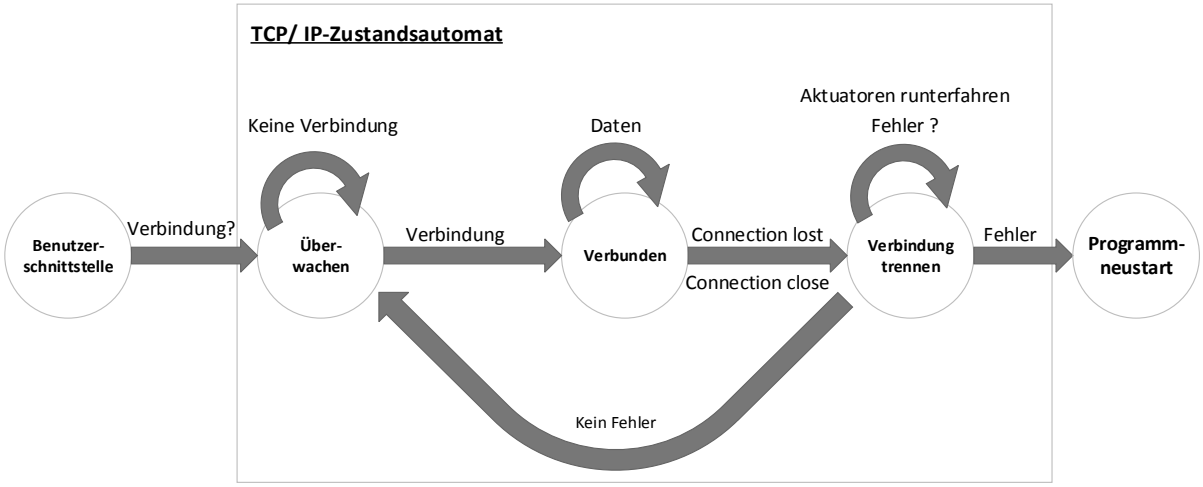

Abbildung 21: TCP/IP einfacher Zustandsautomat für ein System

Zustand: Überwachen

In diesem Zustand wird auf eine TCP/IP-Verbindung mit der Benutzerschnittstelle an dem Port 2055 gewartet. Solange keine Verbindung besteht, werden keine Aktuatoren angesteuert. Wenn eine Verbindung besteht, wechselt der Zustandsautomat in den Zustand *Verbunden*.

Zustand: Verbunden

In diesem Zustand werden die gesendeten Daten der Benutzerschnittstelle kontinuierlich abgefragt und in eine Queue geschrieben.

Die gesendeten Daten enthalten Kanalinformationen zu jedem Aktuator-System im Netzwerk mit den Wellenparametern *Frequenz, Amplitude, Phase, Offset, Ein-/Ausschalten* und *Wellenform* (Abb. 22).

Abbildung 22: Gesendete Daten der Benutzerschnittstelle

Zur Fehlerverarbeitung wurde eine TCP/IP Fehlerbehandlung entwickelt, die nur auf zwei Fehler der TCP/IP-Verbindung reagiert:

- Die Verbindung wird von der Benutzerschnittstelle geschlossen
- Die Verbindung wurde im Netzwerk unterbrochen

Zustand: Verbindung trennen

In diesem Zustand werden die Daten in der Queue zurückgesetzt und die Aktuatoren herunterfahren. Anschließend wird in Abhängigkeit des anliegenden Fehlers ein *Programmneustart* durchgeführt oder in den Zustand *Überwachen* gewechselt.

Liegt ein Systemfehler vor, wird das Programm neu gestartet. Handelt es sich um einen TC/IP-Fehler, wechselt der Zustand wieder zu *Überwachen*.

7.3. Kommunikation

Auf den Grundlagen des TCP/IP-Zustandsautomaten konnte der Ansatz einer Master/Slave Verbindung zwischen mehreren Systemen entwickelt werden. In Abhängigkeit des Verbindungstypen konnte ein Kommunikationsschema aufgebaut werden (Abb. 23).

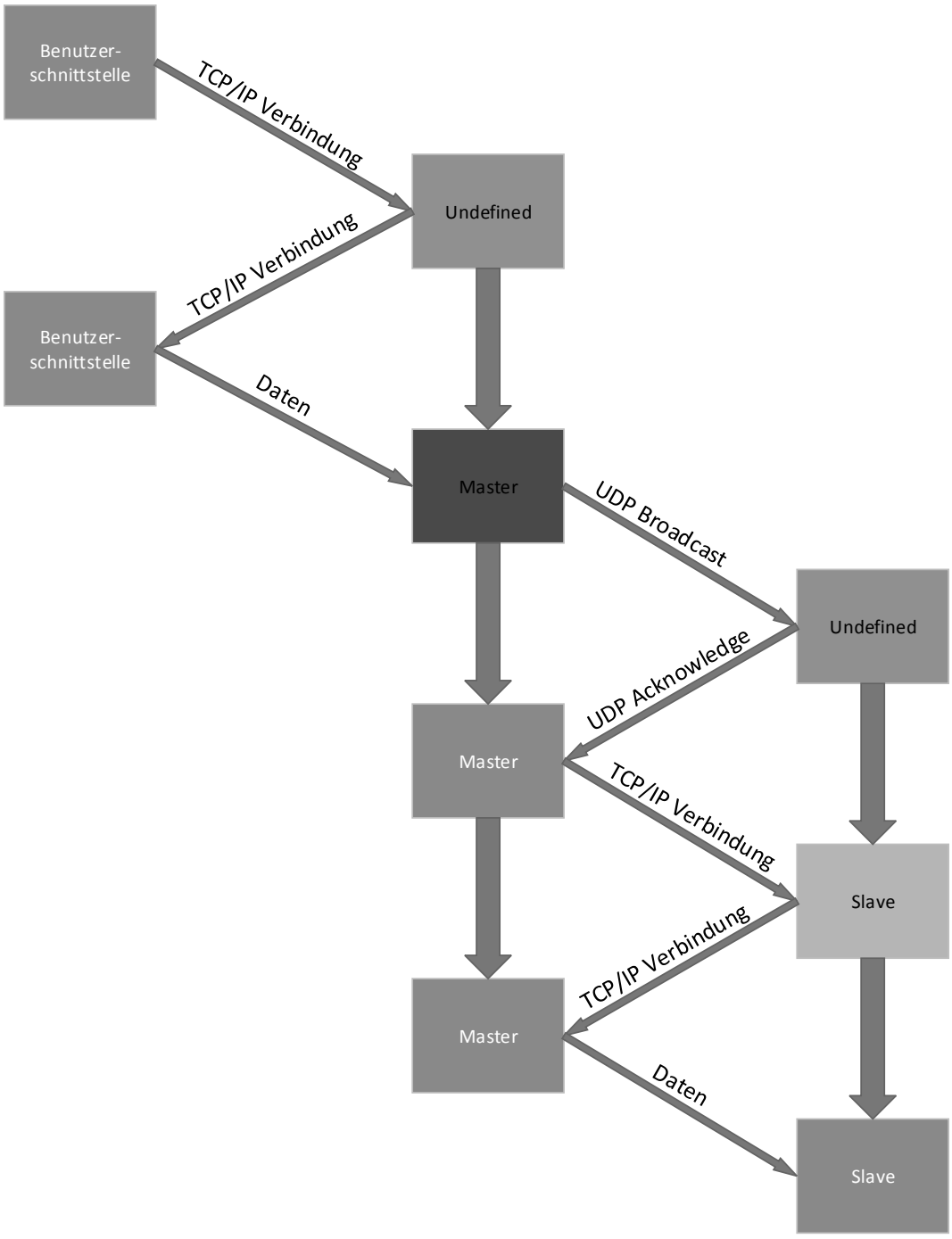

Abbildung 23: Master/Slave-Kommunikationsschema

Dafür wurden drei Zustände definiert, die ein DAQ-System während der Kommunikation annehmen kann:

Undefined:

Der Ausgangszustand jedes Systems ist *Undefined* (Abb. 23). Das System wartet in diesem Zustand auf eine Verbindung mit der Benutzerschnittstelle oder auf eine Verbindung mit einem Master-System. Wird zuerst eine TCP/IP-Verbindung über die Benutzerschnittstelle hergestellt, wechselt das System in den Zustand *Master* (Abb. 23). (vgl. Kap. 7.4 Zustand *Undefined*)

Master:

In dem Zustand *Master* nimmt das System Daten der Benutzerschnittstelle entgegen und „sucht“ über ein UDP-Broadcast nach anderen Systemen im Netzwerk. Über die eingehenden UDP-Acknowledgements erstellt das Master-System eine IP-Adressliste. Mit dieser Liste wird eine TCP/IP-Verbindung zu den anderen DAQ-Systemen hergestellt. Erhält ein System ein UDP-Broadcast und befindet sich im Zustand *Undefined* (Abb. 23), wechselt das System nach Absetzen eines UDP-Acknowledgements in den Zustand *Slave* (Abb. 23). (vgl. Kap. 7.4 Zustand *Master*)

Slave:

In dem Zustand *Slave* wartet das System auf eine TCP/IP-Anfrage des Master-Systems, um nach Verbindung beider Systeme Daten zu erhalten. (vgl. Kap. 7.4 Zustand *Slave*)

7.4. Zustandsautomat: Kommunikation

Um das gezeigte Kommunikationsschema zu realisieren, wurde ein Zustandsautomat mit drei Zuständen entwickelt (Abb. 24). In diesem Abschnitt werden der Ablauf und die Funktionsweise der im letzten Kapitel dargestellten Zustände erläutert.

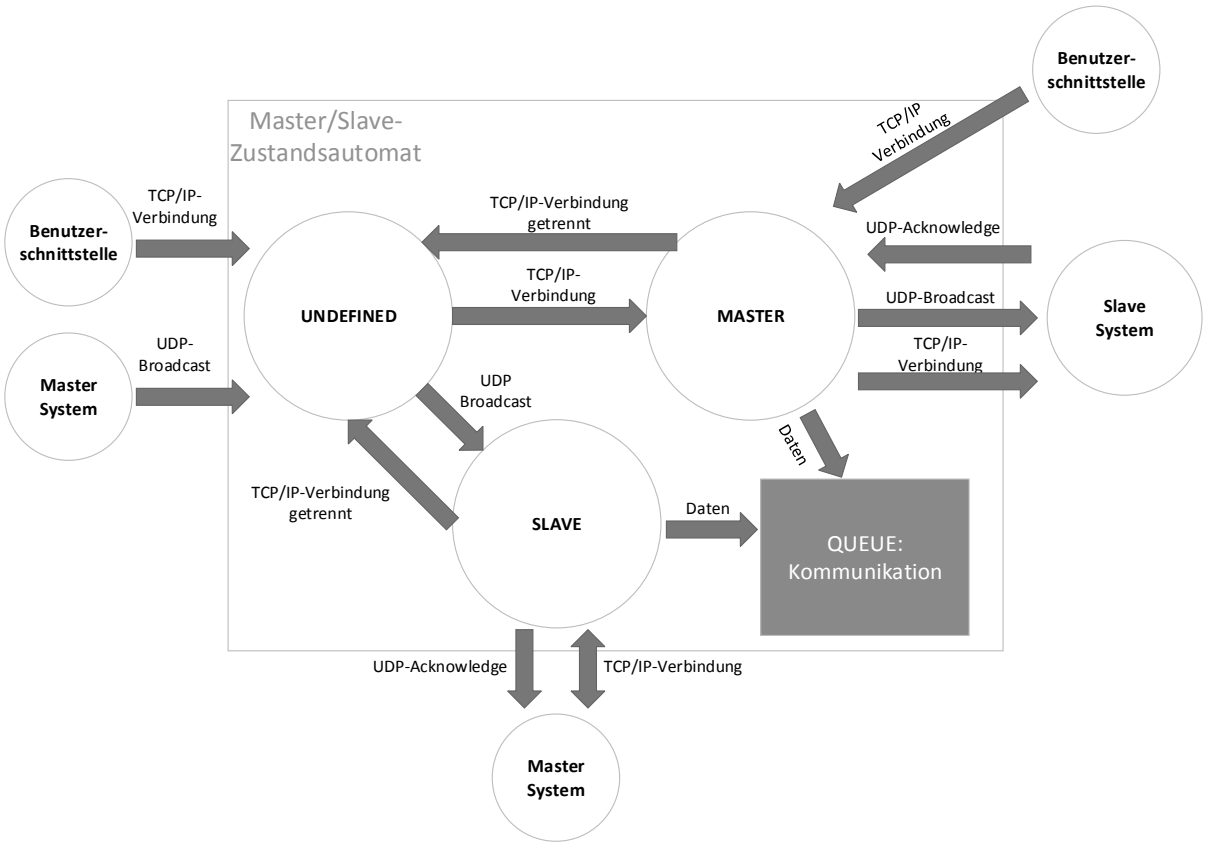

Abbildung 24: Zustandsautomat auf Basis des entwickelten Kommunikationsschemas zur Bestimmung der Master oder Slave Zugehörigkeit

Diese Darstellung dient als Übersicht der benötigten Ein- und Ausgangsparameter für die Kommunikation. Zusätzlich wurde ein Datenspeicher für einen programmübergreifenden Datenaustausch integriert.

Zustand: Undefined

Beim Programmstart befindet sich das DAQ-System im Zustand *Undefined*. In diesem Zustand erfolgt keine Ansteuerung der Aktuatoren und das System wartet auf eine Verbindung von der Benutzerschnittstelle über TCP/IP oder auf ein UDP-Broadcast des Master-Systems (Abb.25).

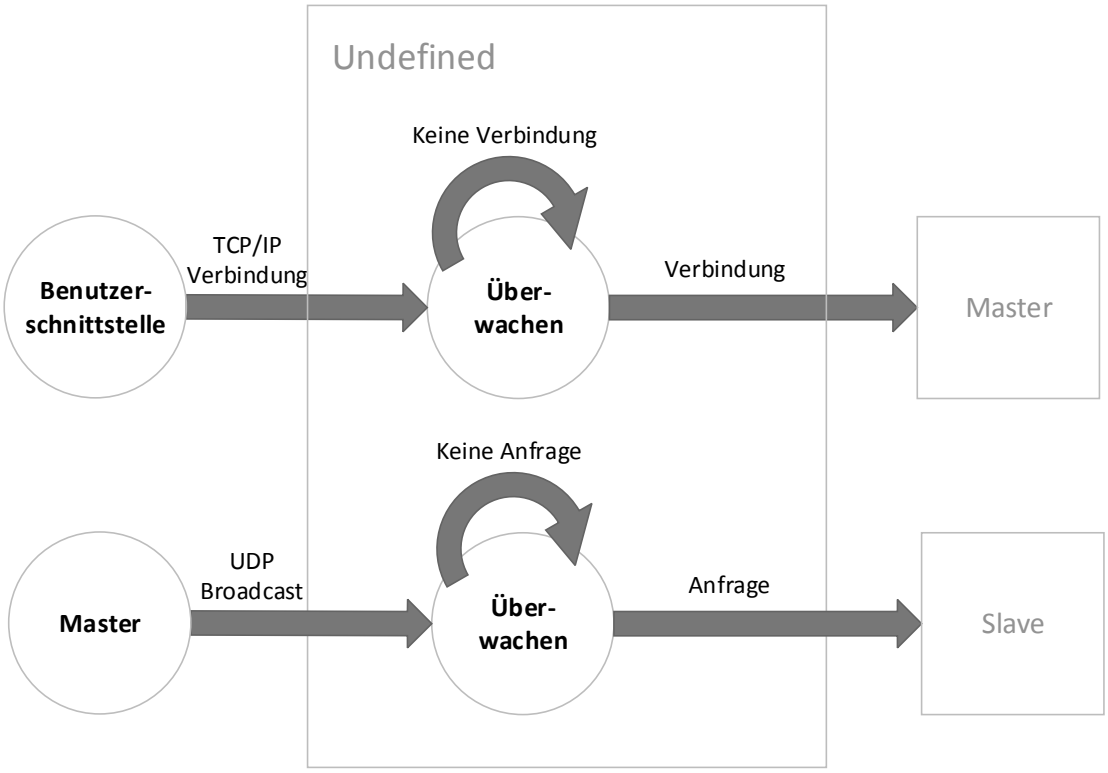

Abbildung 25: Funktionsdiagramm des Zustandes: Undefined

Wenn die Benutzerschnittstelle eine Verbindung herstellt, wechselt das System in den Zustand *Master*. Wenn ein UDP-Broadcast eines Master-Systems eintrifft, wechselt das System in den Zustand *Slave*.

Zustand: Master

Wenn sich das System im Zustand *Master* befindet (Abb. 26), werden die übertragenen Daten der Benutzerschnittstelle auf die Anzahl der Kanäle des DAQ-Systems überprüft.

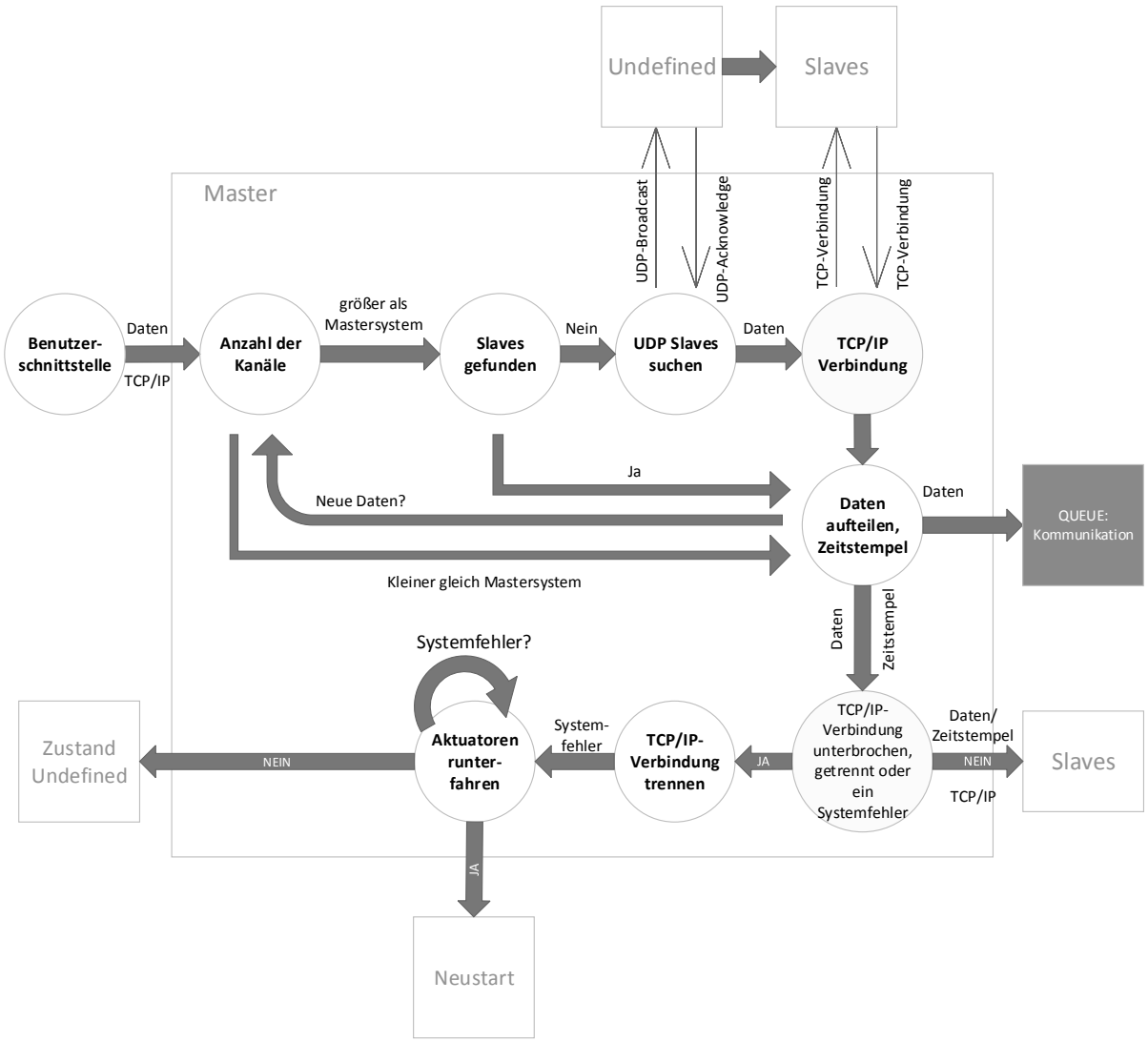

Abbildung 26: Funktionsdiagramm des Zustandes: Master

Ist die Anzahl der übertragenen Kanäle größer als die verfügbaren Ausgänge am Master-System, wird über ein UDP-Broadcast nach anderen echtzeitfähigen Systemen im Netzwerk gesucht. Diese Systeme wechseln nach Beantwortung des UDP Broadcasts über die Rücksendung eines UDP Acknowledgements mit den Informationen ihrer IP-Adresse in den Zustand *Slave*.

Das Master-System erstellt anhand der beantworteten UDP Telegramme eine IP-Adressliste. Die Liste wird aufsteigend von der niedrigsten zur höchsten IP-Adresse sortiert. Danach wird eine Verbindung über das TCP/IP-Protokoll hergestellt. Wenn die Benutzerschnittstelle Daten für eine Anzahl von Ausgängen schickt, welche kleiner als die gemeinsamen Ausgänge aller Master/Slave-Systeme sind werden die Ausgänge nicht gleichmäßig aufgeteilt sondern anhand der Adressliste (Tab. 3).

Tabelle 3: IP-Adressliste des Master-Systems

System	IP-Adresse	Ausgänge (insgesamt 60)
Master	192.168.0.2	20
Slave 1	192.168.0.3	20
Slave 2	192.168.0.4	18 (20)

In diesem Fall erhält das Slave-System mit der höchsten IP-Adresse (Slave 2) nur Daten für 18 Ausgänge obwohl das System 20 Ausgänge zur Verfügung hat.

Bevor die Slave-Systeme die Aktuierung starten, schickt das Master-System einen gemeinsamen Zeitstempel. Ist der Zeitwert dieses Zeitstempels erreicht, starten alle Systeme gleichzeitig mit der Aktuierung.

Wird eine der beiden TCP-Verbindungen mit der Benutzerschnittstelle oder dem Slave-System unterbrochen/getrennt werden die Aktuatoren des Master-Systems heruntergefahren. Anschließend wechselt der Zustandsautomat in den Zustand *Undefined*. Liegt ein Systemfehler vor, dann wird das Programm nach dem Herunterfahren der Aktuatoren neu gestartet.

Zustand: Slave

Befindet sich das System im Zustand *Slave* (Abb.27), kommt der ursprünglich entwickelte TCP/IP-Zustandsautomat (Kapitel 7.2) zum Einsatz.

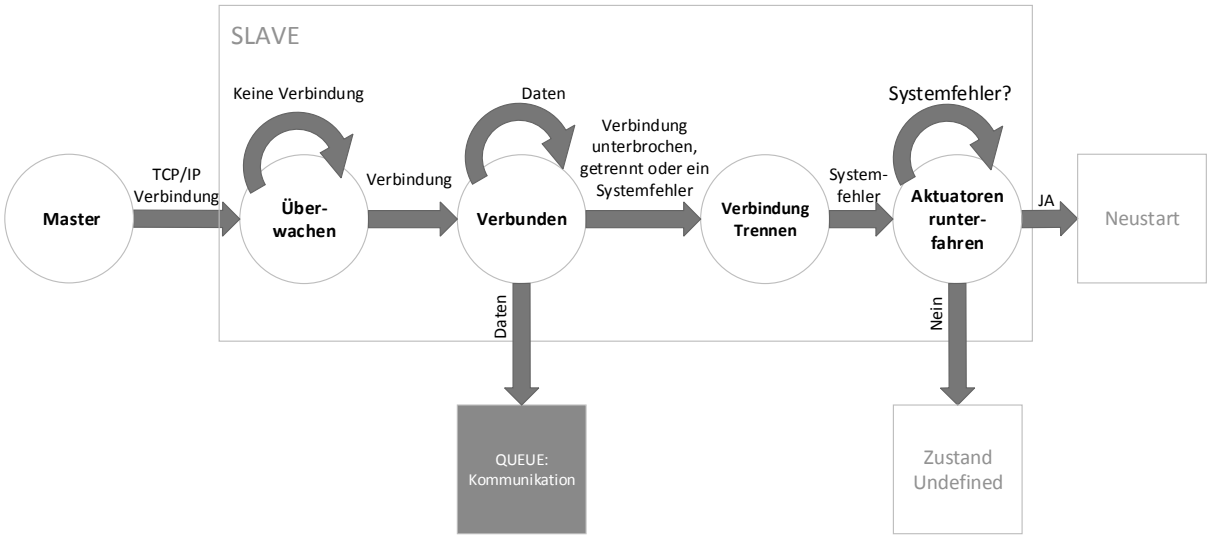

Abbildung 27: Funktionsdiagramm des Zustandes: Slave

Das Slave-System wartet auf eine TCP/IP-Verbindung mit dem Master-System. Wird eine Verbindung hergestellt, nimmt das Slave-System die Daten der Kanäle mit dem Zeitstempel auf und beginnt mit der Aktuierung beim Erreichen des Zeitwerts des Zeitstempels. Bei jeder Änderung der Kanal-Daten wird ein neuer Zeitwert des Zeitstempels gesetzt. Dadurch erfolgt die Transientenkorrektur aller Systeme zum gleichen Zeitwert. Tritt während der Aktuierung ein Verbindungs- oder Systemfehler auf, werden die Aktuatoren bis zum nächsten Nullpunkt der Wellenfunktion heruntergefahren. Das Slave-System startet neu oder wechselt, wenn die Verbindung getrennt/unterbrochen wurde, in den Zustand *Undefined*.

7.5. Echtzeitige Datenerfassung

Als Vorbereitung für die Wellenregelung der Aktuatoren ist es erforderlich, die Datenerfassung zu synchronisieren (Abb. 28).

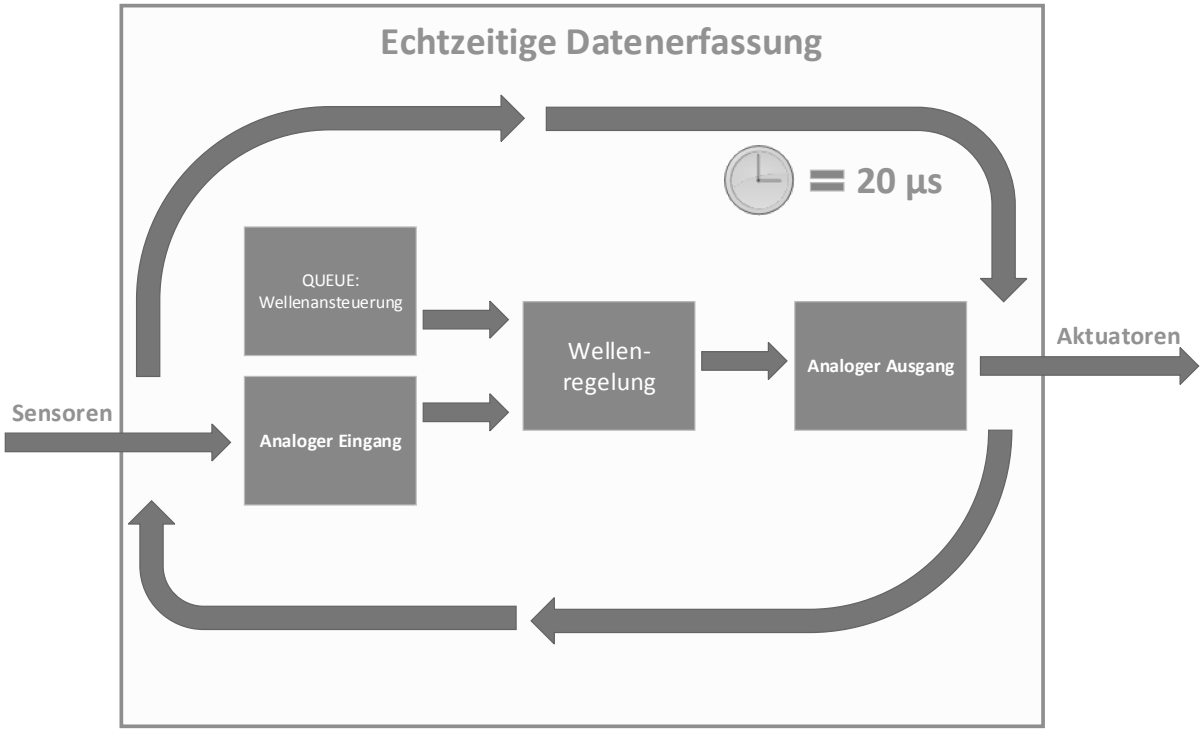

Abbildung 28: Synchrone Messung der Sensoren und Stellung des geregelten Wertes

Zur Synchronisation der analogen Ein- und Ausgänge wird ein gemeinsamer Taktgeber (angelehnt an die Generierung der Wellenform Kapitel 6.1) über das DAQ-System beim Starten des Programms erstellt. Über den Taktgeber wird eine echtzeitige zeitgesteuerte Schleife ausgeführt, welche innerhalb von 20 µs die Bearbeitung abschließt.

Dazu gehört:

- Das Einlesen des generierten Signalwertes aus der Queue des Teilprogramms *Wellenansteuerung*
- Das Einlesen der Eingangswerte der Sensoren
- Die Neuberechnung des Signalwertes durch die zu entwickelnde *Wellenregelung*
- Die Ausgabe der geregelten analogen Ausgangswerte an die Aktuatoren

Dieser Vorgang wiederholt sich kontinuierlich alle 20 µs. Zur Verdeutlichung des Programmablaufs wurde ein Laufzeitdiagramm der Schleife erstellt (Abb. 29).

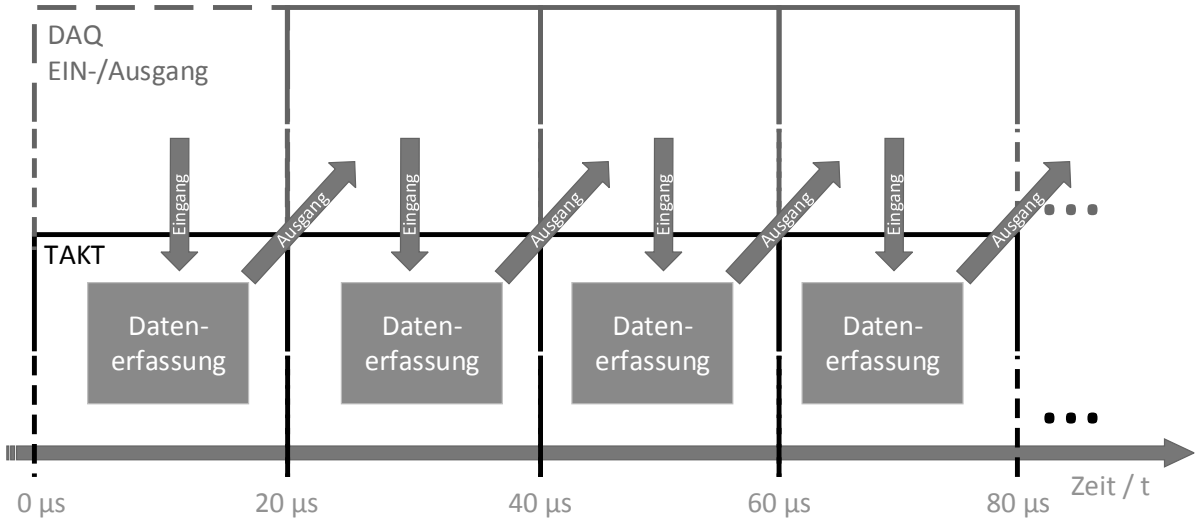

Abbildung 29: Laufzeitdiagramm der zeitgesteuerten Schleife

Nachdem der erste Takt der DAQ-Karte die zeitgesteuerte Schleife zum Zeitpunkt $t = 0\ \mu s$ gestartet hat, werden die Sensordaten am Eingang eingelesen. In Abhängigkeit der Sensordaten, und des Signalwertes aus der Queue des Teilprogramms *Wellenansteuerung*, kann eine *Wellenregelung* eingreifen. Nach der Regelung des Signalwertes wird innerhalb von $t = 20\ \mu s$ der Wert dem analogen Ausgang übergeben. Danach wird die zeitgesteuerte Schleife zum nächsten Takt wiederholt.

7.6. Dateisystem

Unter Verwendung des entwickelten Systemkonzeptes zur Generierung von Wellenformen wurden Dateien mit jeweils 1.000.000 Sinus- und Dreieck-Wellenpunkten erstellt und im Aktuator-System abgespeichert (Abb. 30).

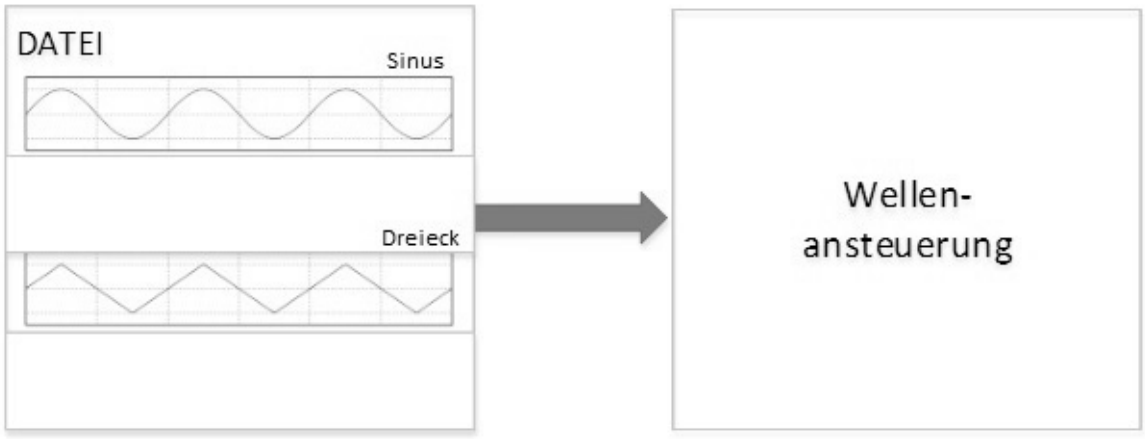

Abbildung 30: Die Daten werden aus dem Dateisystem geladen und an das Programm Wellenansteuerung übergeben

Die Wellendaten werden beim Start des Systems in den Arbeitsspeicher geladen, sodass keine prozessorbelasteten Speicherfunktionen aufgerufen werden müssen. Danach stehen die Daten dem Teilprogramm *Wellenansteuerung* für weitere Bearbeitungen zur Verfügung.

7.7. Zustandsautomat: Wellenansteuerung

In diesem Programmabschnitt werden alle eingegangenen Welleninformationen aus dem *Dateisystem* und dem Teilprogramm *Kommunikation* (Kapitel 7.4) zusammengeführt und die Ausgangswelle für alle Kanäle erzeugt. Der Zustandsautomat verfügt über Vier Zustände in dem er sich befinden kann (Abb. 31).

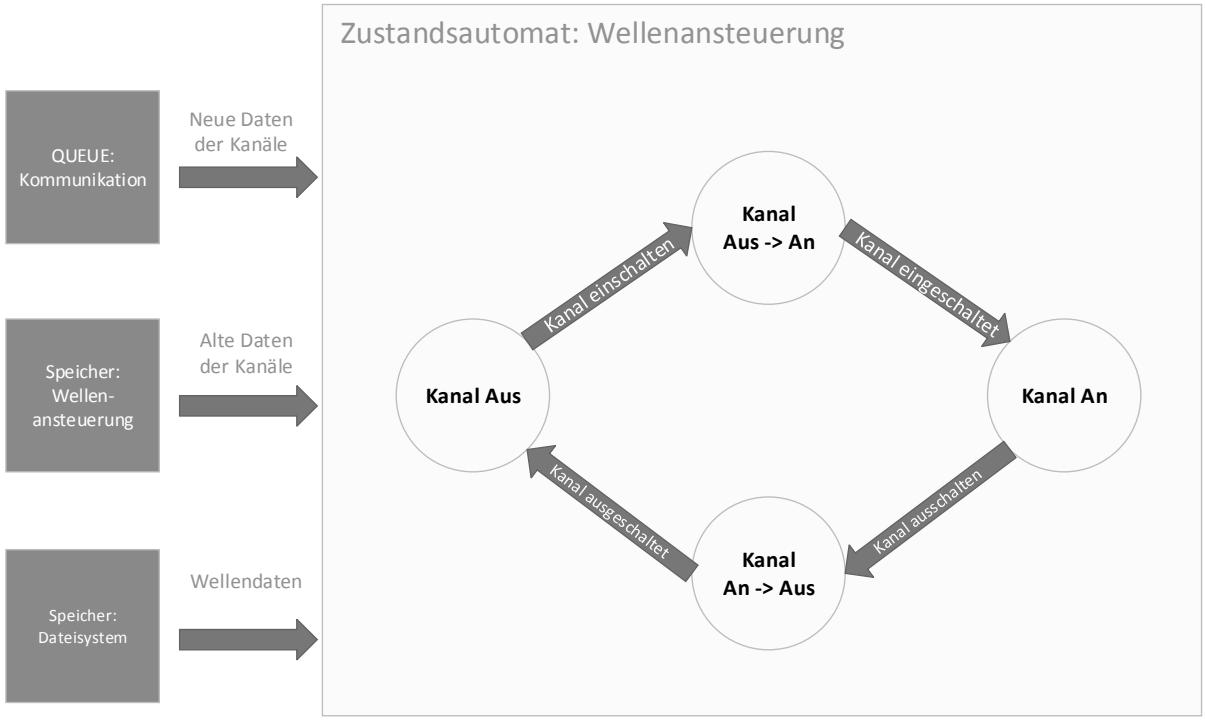

Abbildung 31: Kanalzustände des Zustandsautomat: Wellenansteuerung

Diese Zustände sind von dem Wellenparameter *Ein-/Ausschalten* abhängig. Wird ein Kanal eingeschaltet und der Zustandsautomat befindet sich im Zustand *Kanal Aus*, wechselt der Zustand in *Kanal Aus->An*. In diesem Zustand wird der Kanal unter Berücksichtigung der Kanaldaten mit einem bestimmten Einschaltverhalten eingeschaltet. Erst nach einer Periode des generierten Signals wechselt der Zustand dann in *Kanal An*. Das ist der wichtigste Zustand des Zustandsautomaten. Hier erfolgt die Generierung von glatten Signalübergängen. Bei Ausschalten des Kanals wechselt der Zustand in *Kanal An->Aus*. In diesem Zustand wird der Kanal unter Berücksichtigung der Kanaldaten mit einem bestimmten Ausschaltverhalten ausgeschaltet. Danach nimmt der Zustandsautomat seinen Ausgangszustand *Kanal Aus* an.

Zustand: Kanal Aus

Bei einem Start des DAQ-Systems wird mit diesem Zustand begonnen und der Speicher der *Wellenansteuerung* hat für den aktuellen Abtastwert der Wellendaten den Anfangswert gleich Null (Abb. 32).

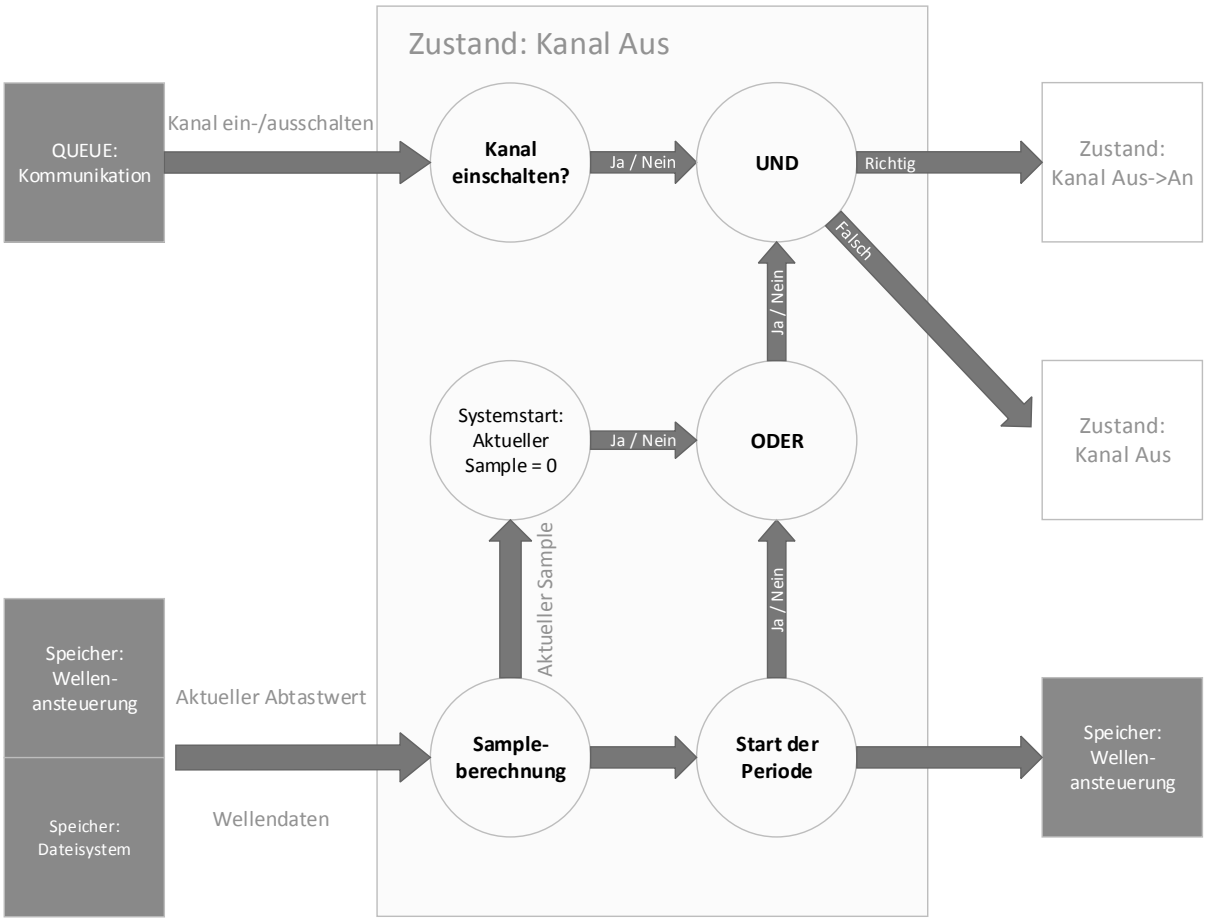

Abbildung 32: Funktionsdiagramm des Ausgangszustand Kanal Aus im Zustandsautomat Wellenansteuerung

Nachdem die ersten Welleninformationen des Teilprogramms *Kommunikation* übertragen wurden, wird anhand des Wellenparameters *Kanal ein-/ausschalten überprüft*, ob der Kanal eingeschaltet werden soll. Ist das nicht der Fall, bleibt der Zustandsautomat in seinem alten Zustand. Ansonsten durchläuft das Programm mehrere Zwischenschritte, um seinen Zustand in *Kanal Aus->An* zu wechseln.

Sampleberechnung

In dieser Funktion werden die aktuellen Wellendaten für eine Sinus- oder Dreieckwelle in Abhängigkeit des Wellenparameters *Wellenform* ausgelesen. Anschließend wird mit dem aktuellen Abtastwert der letzten Sampleberechnung ein neuer Abtastwert berechnet.

$$Neuer\ Abtastwert = Aktueller\ Abtastwert + Abtastwert_{Datei}$$

Ist der neue Abtastwert größer als der Speicher der Wellendaten, wird die Differenz als neuer Abtastwert übernommen und abgespeichert.

$$Neuer\ Abtastwert = Neuer\ Abtastwert - Speicher_{Werte}$$

Gleichzeitig wird der Start einer neuen Periode des erzeugten Signals bestätigt.

Diese Funktion ist beim ersten Start des DAQ-Systems unerheblich, da im Anfangszustand die *Frequenz* des Kanals gleich Null ist, und somit der neue sowie der aktuelle Abtastwert gleich Null sind. Und der Zustand *Kanal Aus* wechselt nach der Abfrage ob der aktueller Abtastwert gleich Null ist in den Zustand *Kanal Aus->An*.

Wurde während der Laufzeit des DAQ-Systems der Kanal ein- und wieder ausgeschaltet, läuft die Berechnung des Abtastwertes im Zustand *Kanal Aus* solange weiter, bis der Kanal eingeschaltet wird und ein Start der neuen Periode bestätigt ist. Das ist notwendig, um mit den anderen Kanälen zeitgleich zur nächsten Periode starten zu können (Abb. 33).

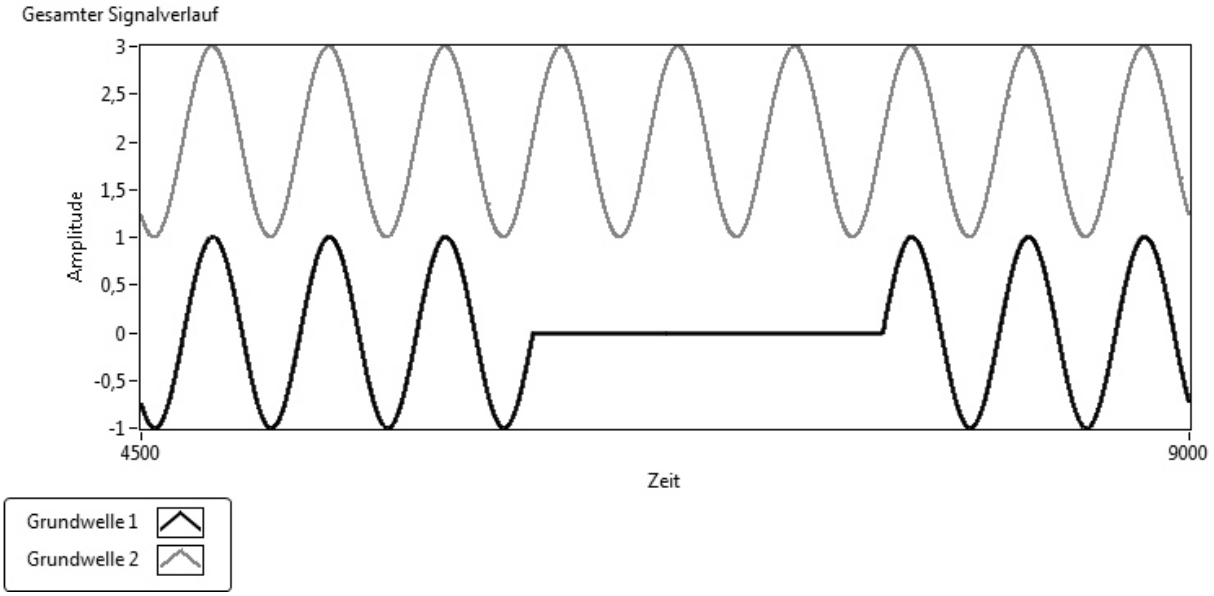

Abbildung 33: Nach Ausschalten des Kanals beginnt die neue Welle des Kanals zum gleichen Zeitpunkt t=0

Das neue Sample und der Start einer neuen Periode werden in den Zwischenspeicher des Teilprogramms *Wellenansteuerung* geschrieben.

Zustand: Kanal Aus->An

In diesem Zustand werden in Abhängigkeit der Wellenparameter *Phase* und *Offset* verschiedene Einschaltvorgänge durchgeführt (Abb. 34). Bei Start des DAQ-Systems hat der Speicher der *Wellenansteuerung* für die *Phase* und den *Offset* den Anfangswert Null. Sofern im Speicher die Phaseninformation eines zuvor aktuierten Kanals nicht mehr vorliegen (vgl. Kap. 7.7 Zustand *Kanal Aus*).

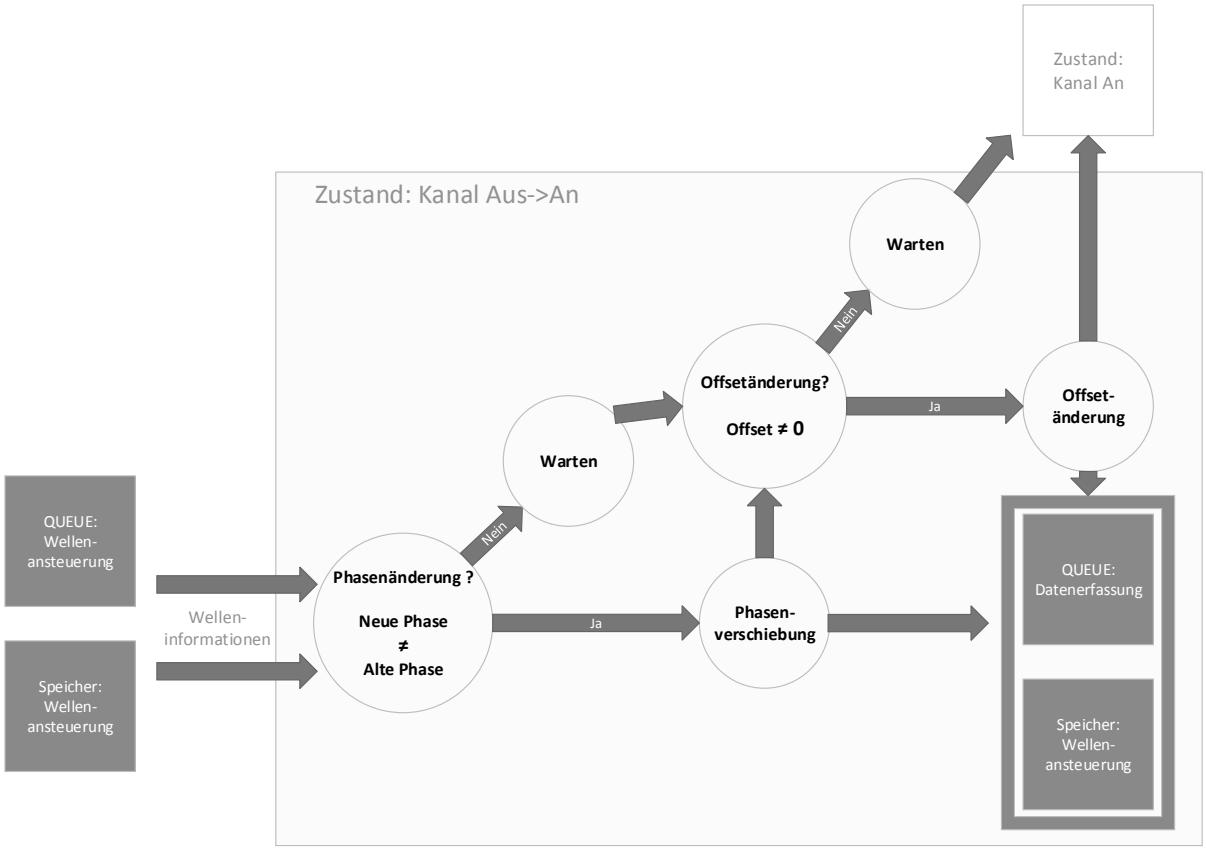

Abbildung 34: Funktionsdiagramm des Zustands Kanal Aus->An

Unterscheidet sich die Phase der neuen Welleninformationen von der Phase des Speichers, findet eine Phasenverschiebung der Welle statt.

Phasenverschiebung beim Einschaltvorgang

Für die Phasenverschiebung beim Einschaltvorgang werden zwei Perioden des Signals als Vorlaufzeit verwendet. Dies ist notwendig, um hochfrequente Störungen durch Sprünge des Signals vom Nullwert bis zum Phasenwert (Bsp. φ = 90°) zu vermeiden (Abb. 35).

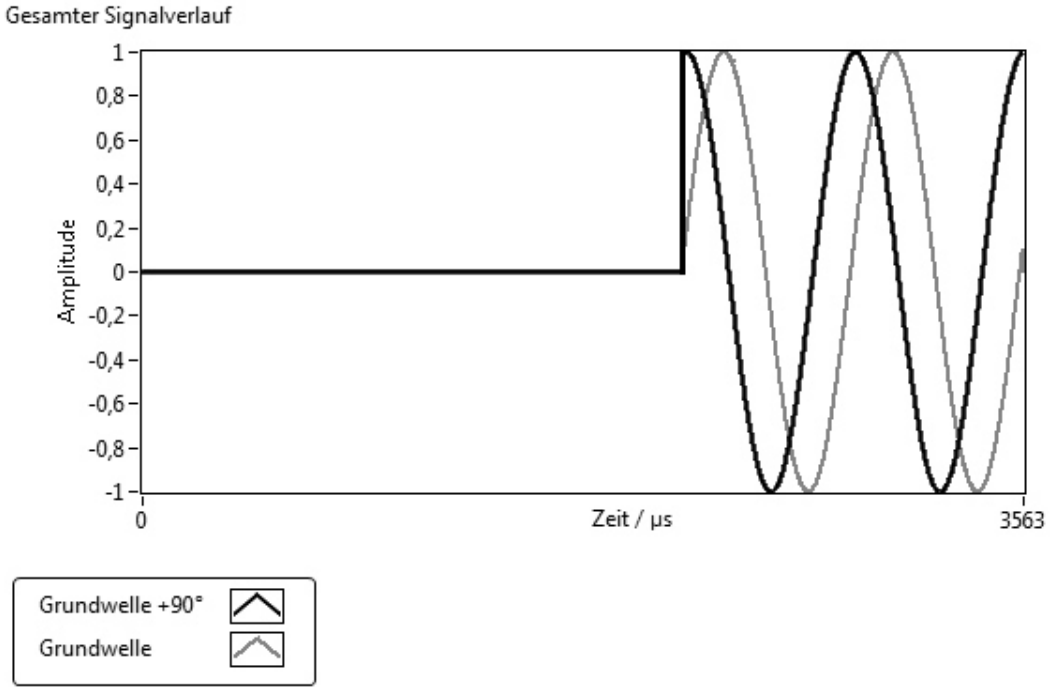

Abbildung 35: Sprung der Welle vom Wert 0 zum Phasenwert 1 ohne eine Signalanpassung

Die Phasenverschiebung der Welle erfolgt über das entwickelte Konzept der glatten Signalübergänge (s. Kap. 6.2). Dabei muss die Differenz zwischen den beiden Phasen gebildet werden.

Dafür gilt

$$Phase_{neu} > Phase_{alt}$$
$$Phase = |\, Phase_{alt} - Phase_{neu}|$$

und

$$Phase_{neu} < Phase_{alt}$$
$$Phase = 360° - |\, Phase_{alt} - Phase_{neu}|$$

Dann wird der Speicher für einen Speicherdurchlauf bzw. eine Periode mit dem neu errechneten Abtastwert abgetastet (Abb. 36: „gestrichelte Welle“). Während der Abtastung des Speichers findet keine Aktuierung des Systems statt. Der Wellenpunkt erhält den Ausgangswert 0 (Abb. 36).

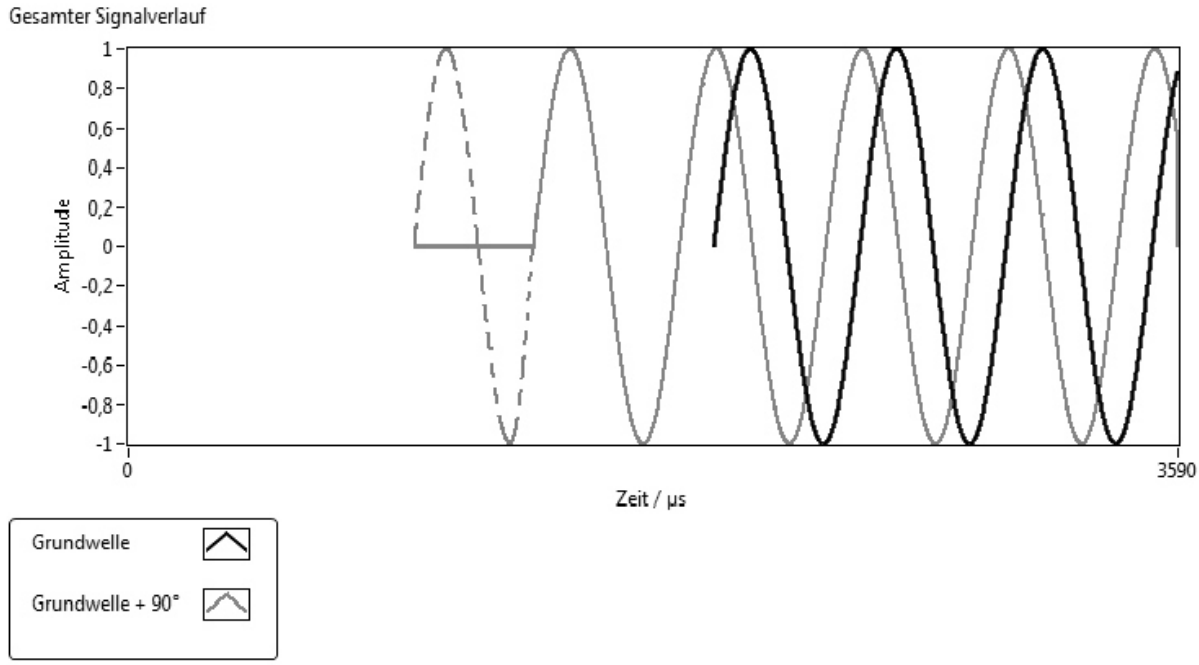

Abbildung 36: Abtastung des Speichers vor Beginn der Aktuierung

Dieses Einschaltverhalten entsteht, wenn eine Phasenänderung am Kanal vorliegt. Ist das nicht der Fall, wird der Speicher anhand des errechneten Abtastwertes (s. Kap. 6.1) für einen Speicherdurchlauf (einer Periode des Signals) abgetastet. Dann wird zum nächsten Speicherdurchlauf (zur nächsten Periode) überprüft, ob der Offsetwert der neuen Welleninformationen ungleich Null ist.

<u>Offsetänderung beim Einschaltvorgang</u>

Die Offsetänderung beim Einschaltvorgang wird entweder nach der Phasenverschiebung oder am Ende der Laufzeit von zwei Speicherdurchläufen ausgeführt.

Dadurch werden hochfrequente Störungen durch Sprünge des Signals vom Nullwert zum Offsetwert vermieden (Abb. 37).

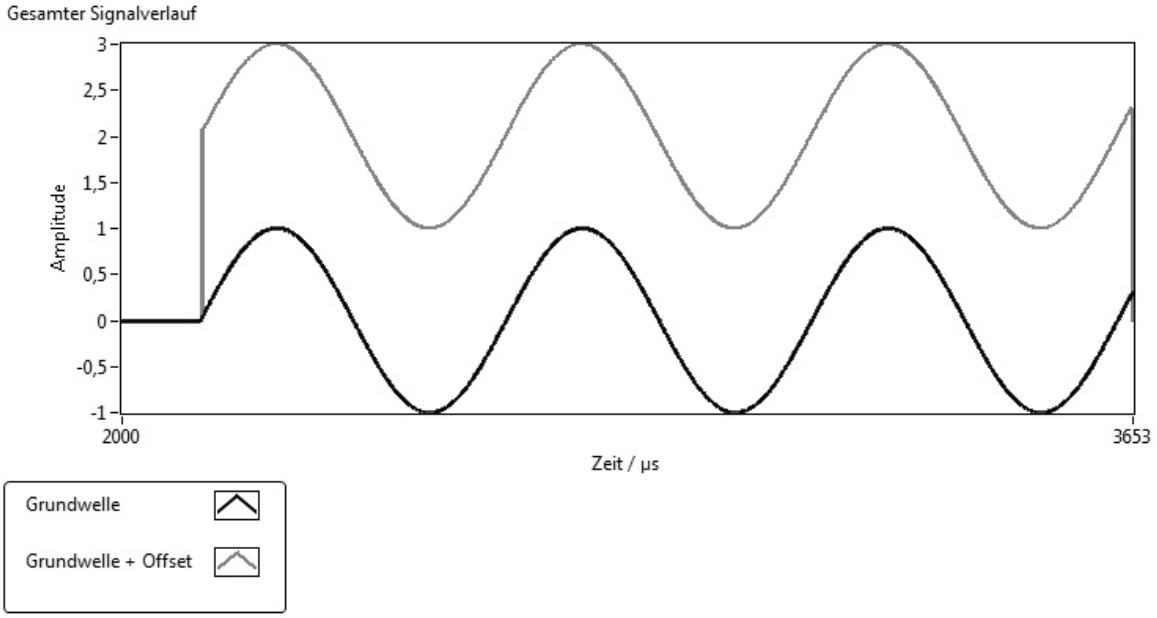

Abbildung 37: Offsetänderung ohne Signalanpassung

Die Offsetänderung der Welle erfolgt über die Rampenfunktion (Abb. 38) aus dem Konzept der glatten Signalübergängen (s. Kap. 6.2).

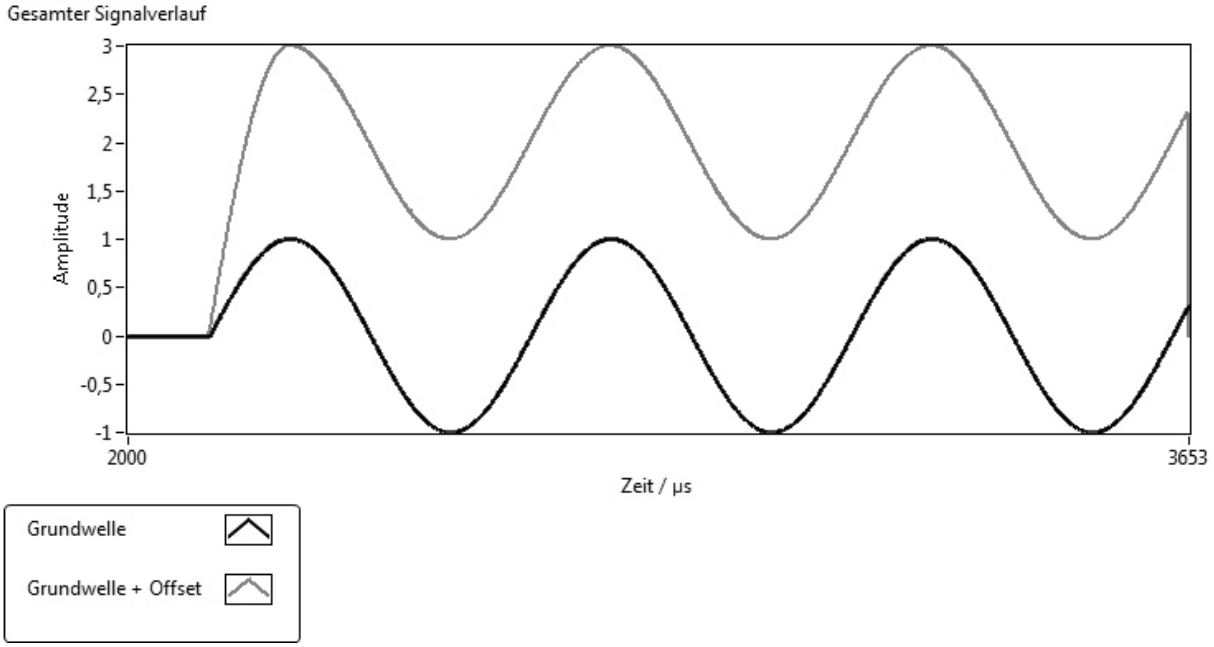

Abbildung 38: Offsetänderung mit Signalanpassung

Phasenverschiebung und Offsetänderung beim Einschaltvorgang

Bei Änderung der Wellenparameter werden beide vorgestellten Konzepte vereint. Zuerst findet die Phasenänderung statt, und danach die Offsetänderung.

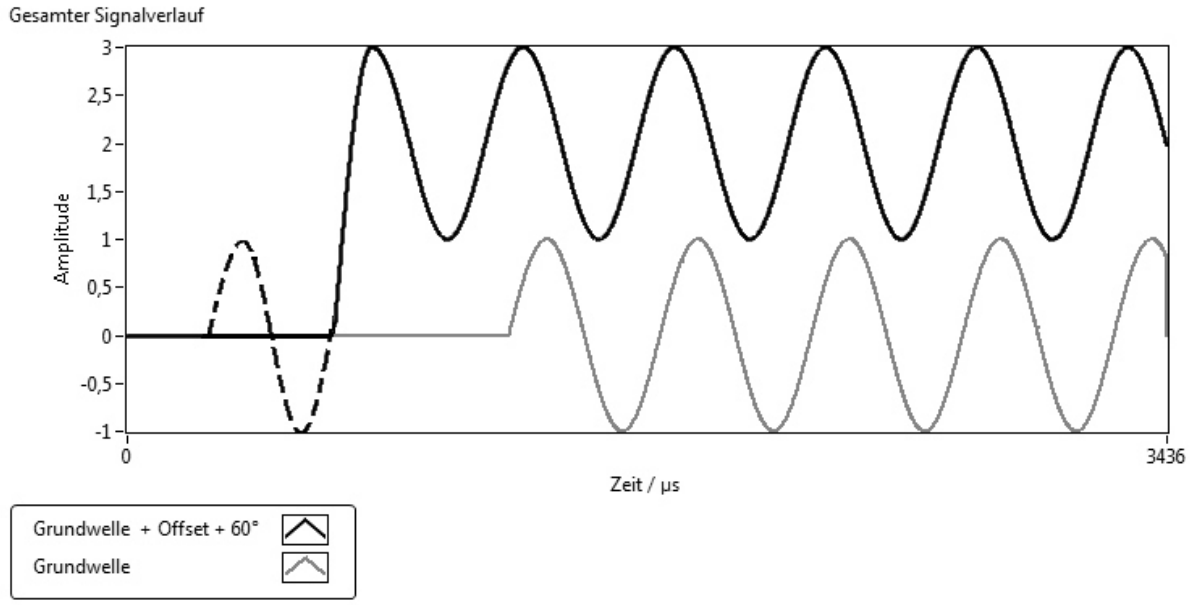

Abbildung 39: Phasen- und Offsetänderung des Signals.

Nachdem der Einschaltvorgang abgeschlossen ist, wechselt der Zustand nach Ablauf von zwei Speicherdurchläufen in den Zustand *Kanal An*. Die Daten der Kanäle werden in den Speicher der *Wellenansteuerung* geschrieben. Die erzeugten Wellenpunkte werden in die Queue des Teilprogramms *Datenerfassung* geschrieben.

Zustand: Kanal An

Der Zustand *Kanal An* dient zur Generierung glatter Signalübergänge der aktuierten Welle bei Änderung der Wellenparameter *Frequenz*, *Amplitude*, *Phase*, *Offset*, *Ein-/Ausschalten* und *Wellenform* (Abb. 40).

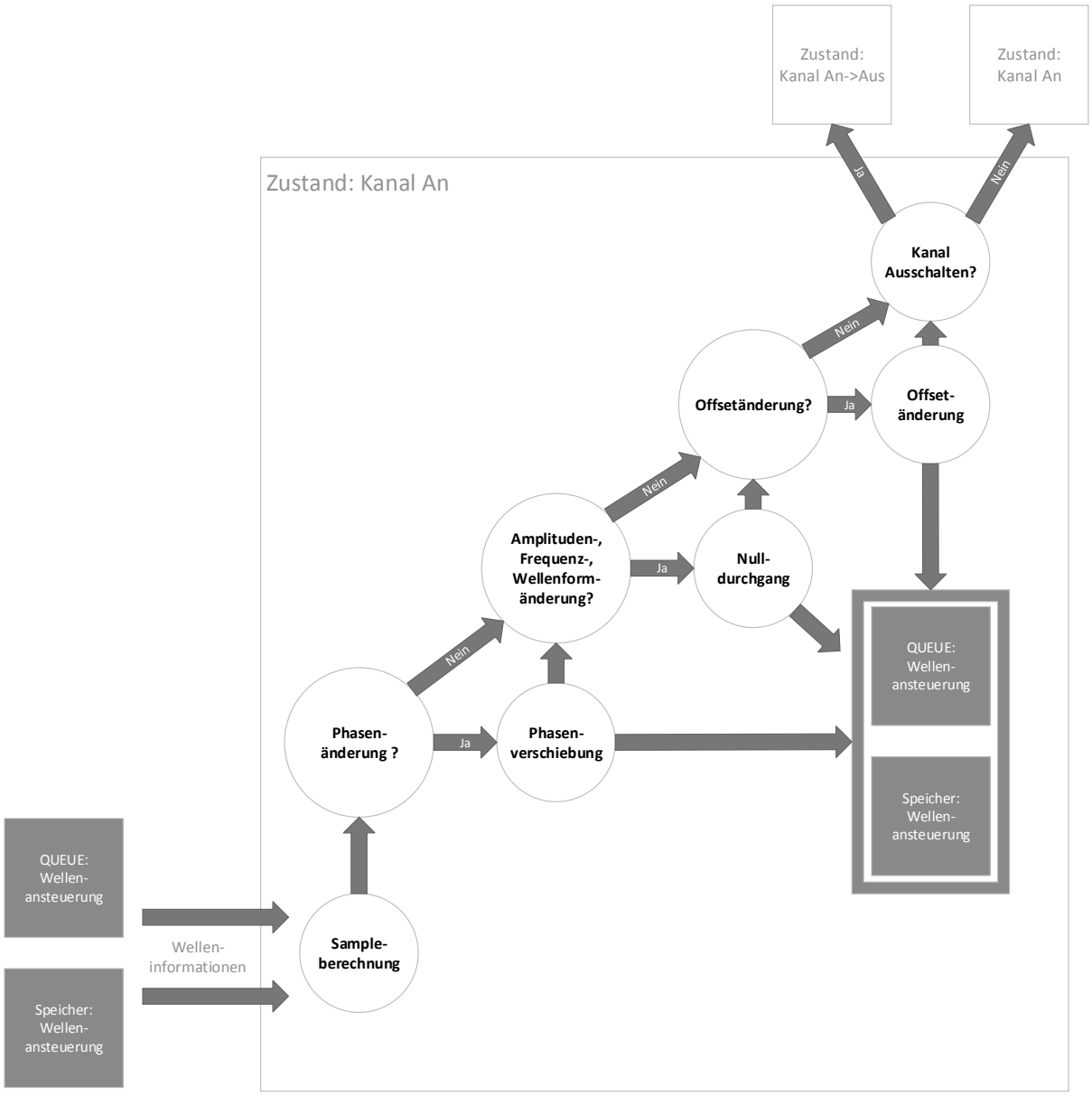

Abbildung 40: Funktionsdiagramm des Zustands: Kanal An

Das Programm startet mit der Sampleberechnung des Abtastwertes. Wenn keine Änderung der Wellenparameter vorliegt wird der Speicher mit diesem Wert abgetastet.

Änderung der *Phase*:

Es wird dieselbe Funktion wie im Zustand *Kanal Aus->An* benutzt, mit dem Unterschied, dass die berechneten Wellenpunkte in den Speicher der Datenerfassung geschrieben und an den analogen Ausgang übergeben werden (Abb. 41).

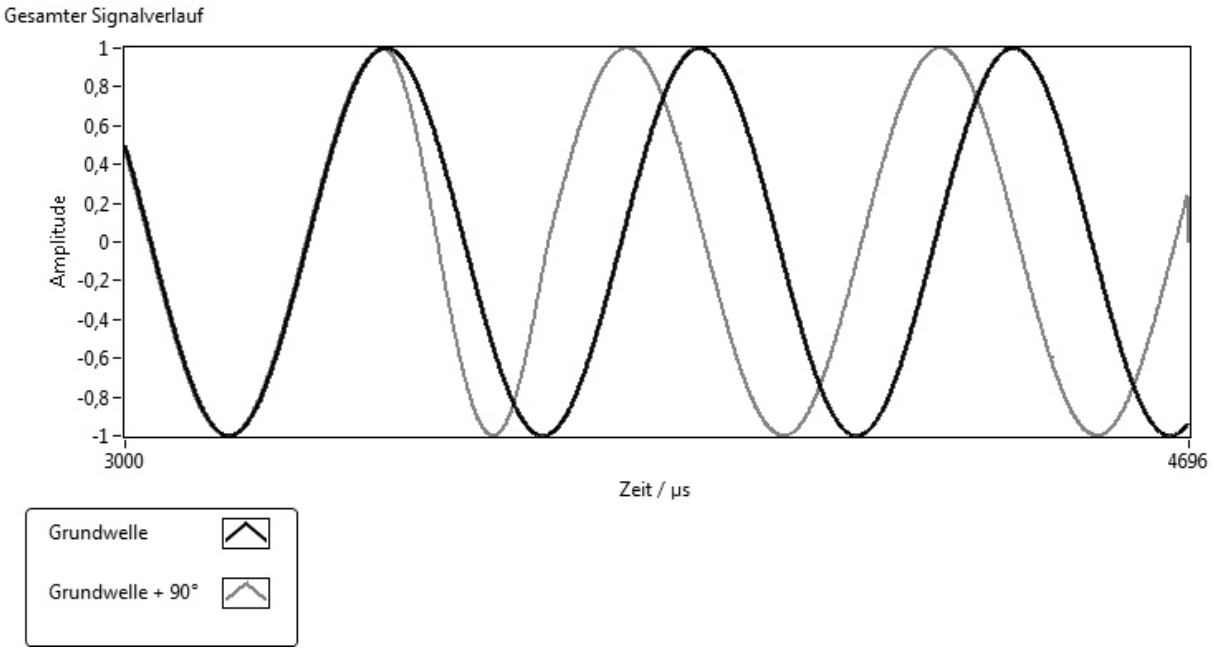

Abbildung 41: Phasenverschiebung mit Übergabe der errechneten Wellenpunkt

Änderung der *Frequenz*, *Amplitude* oder *Wellenform*:

In diesem Fall wird auf die Bestätigung des Starts der neuen Periode von der Funktion Sampleberechnung gewartet und beim Nulldurchgang der Parameterwechsel vollzogen (Abb. 42).

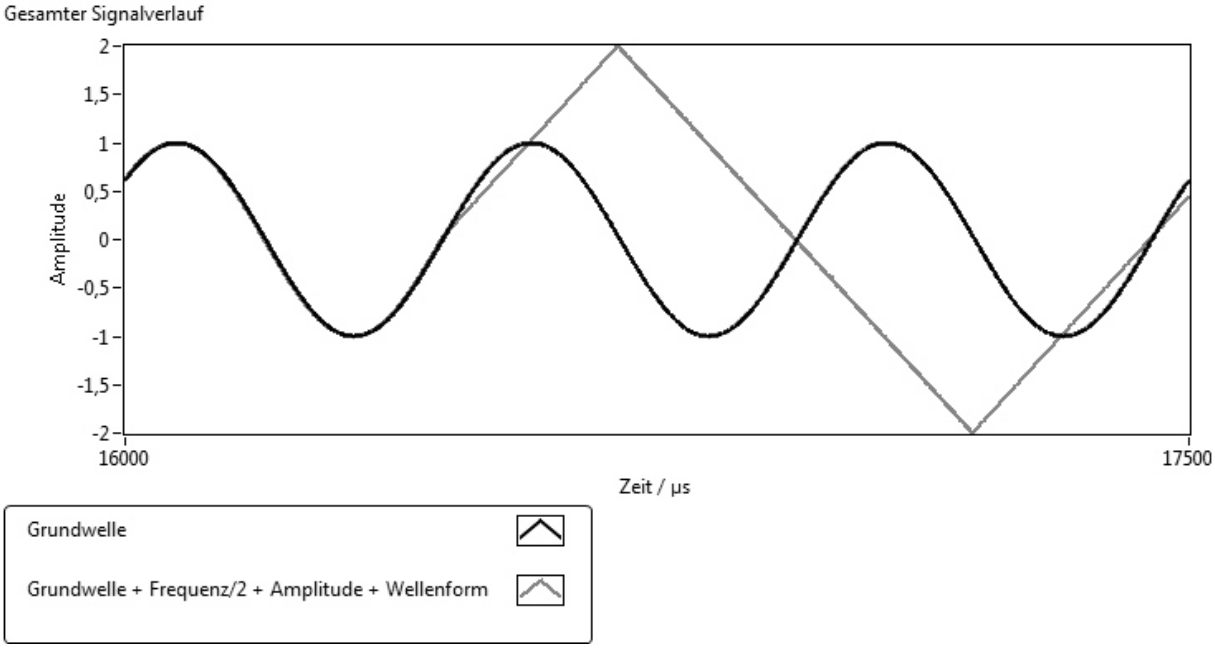

Abbildung 42: Parameterwechsel beim Nulldurchgang

Änderung des *Offsets*:

Bei einer Änderung des Offsets wird bei einer positiven Offsetänderung auf die Bestätigung des Starts der neuen Periode von der Funktion Sampleberechnung gewartet und beim Nulldurchgang der Parameterwechsel vollzogen (Abb. 43: „schwarze Sinuswelle“). Und bei einer negativen Offsetänderung, wird bis zum nächsten Nulldurchgang der Welle gewartet (Abb. 43: „graue Sinuswelle“). (vgl. Kap. 8 Glatte Signalübergänge.

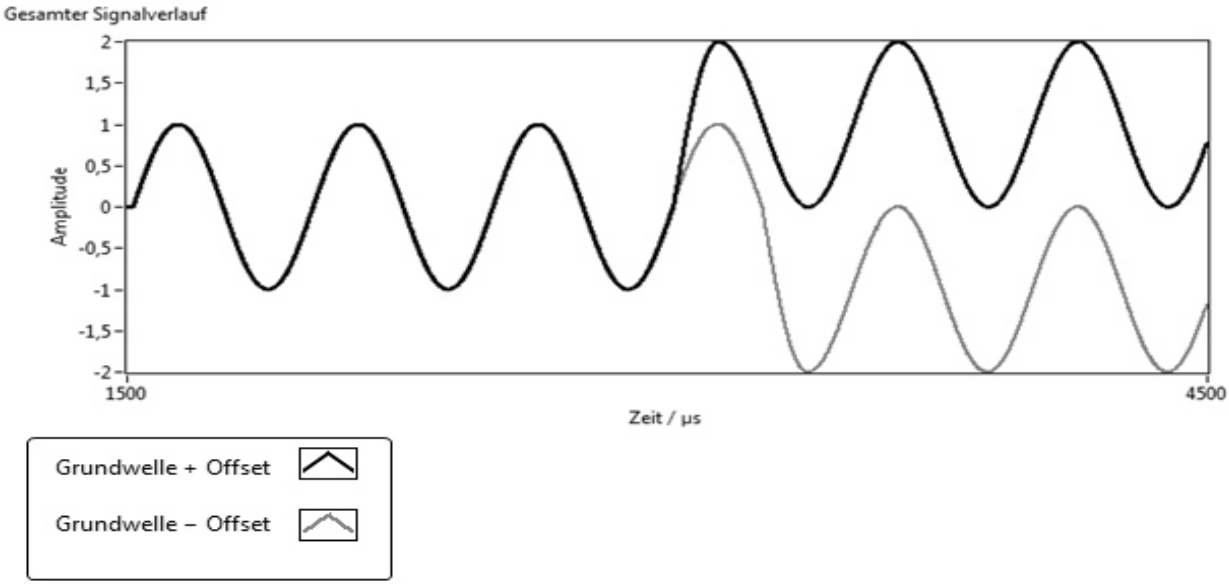

Abbildung 43: Negative und positive Offsetänderung

Änderung aller Wellenparameter:

Erfolgt eine gleichzeitige Änderung aller Wellenparameter eines Signals werden alle Funktionen gemäß des gezeigten Funktionsdiagramms (s. Abb. 40) nacheinander vom Programm abgearbeitet. Zuerst wird die Phasenänderung durchgeführt, anschließend werden bei Erreichen des Nulldurchgangs die anderen Wellenänderungen der *Frequenz*, *Amplitude* und des *Offsets* übernommen (Abb. 44).

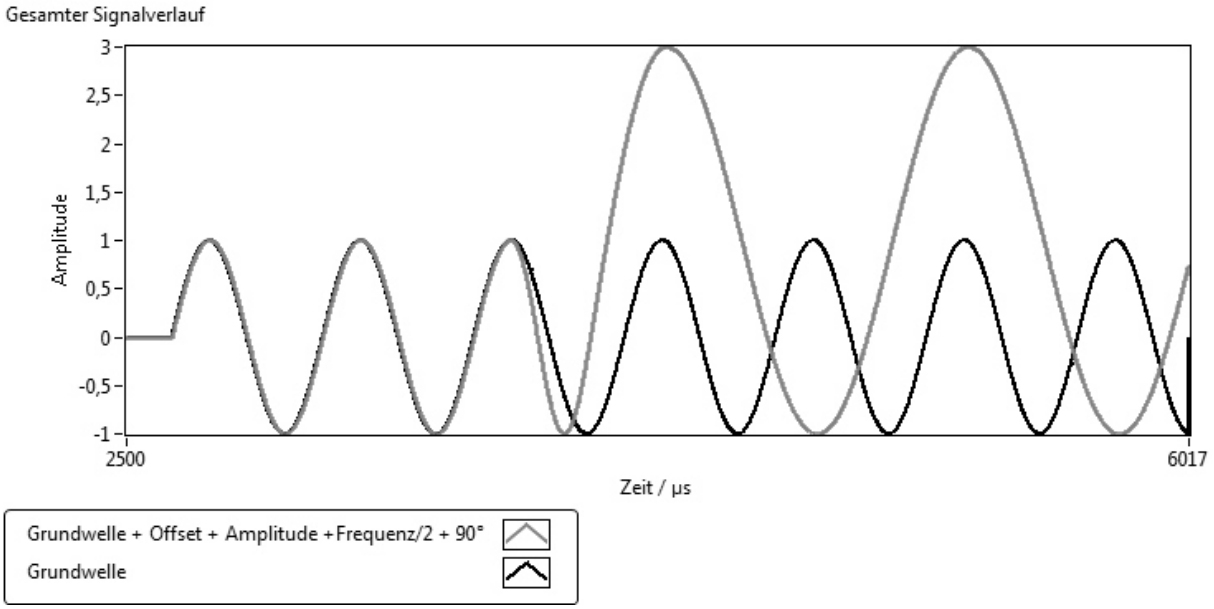

Abbildung 44: Signalanpassung aller geänderten Wellenparameter

Nachdem die Änderungen der Wellenparameter abgeschlossen sind, wird zuletzt überprüft, ob der Kanal ausgeschaltet werden soll. Ist das der Fall, wechselt der Zustandsautomat in den Zustand *Kanal An->Aus*. Alle Welleninformationen der einzelnen Kanäle werden beim Wechsel des Zustandes in den Speicher des Teilprogramms *Wellenansteuerung* geschrieben.

Zustand: Kanal An->Aus

In diesem Zustand werden die Aktuatoren bis zum Nullpunkt heruntergefahren. Dies ist notwendig um hochfrequente Störungen durch Sprünge des Signals zum Nullpunkt zu vermeiden (Abb. 45).

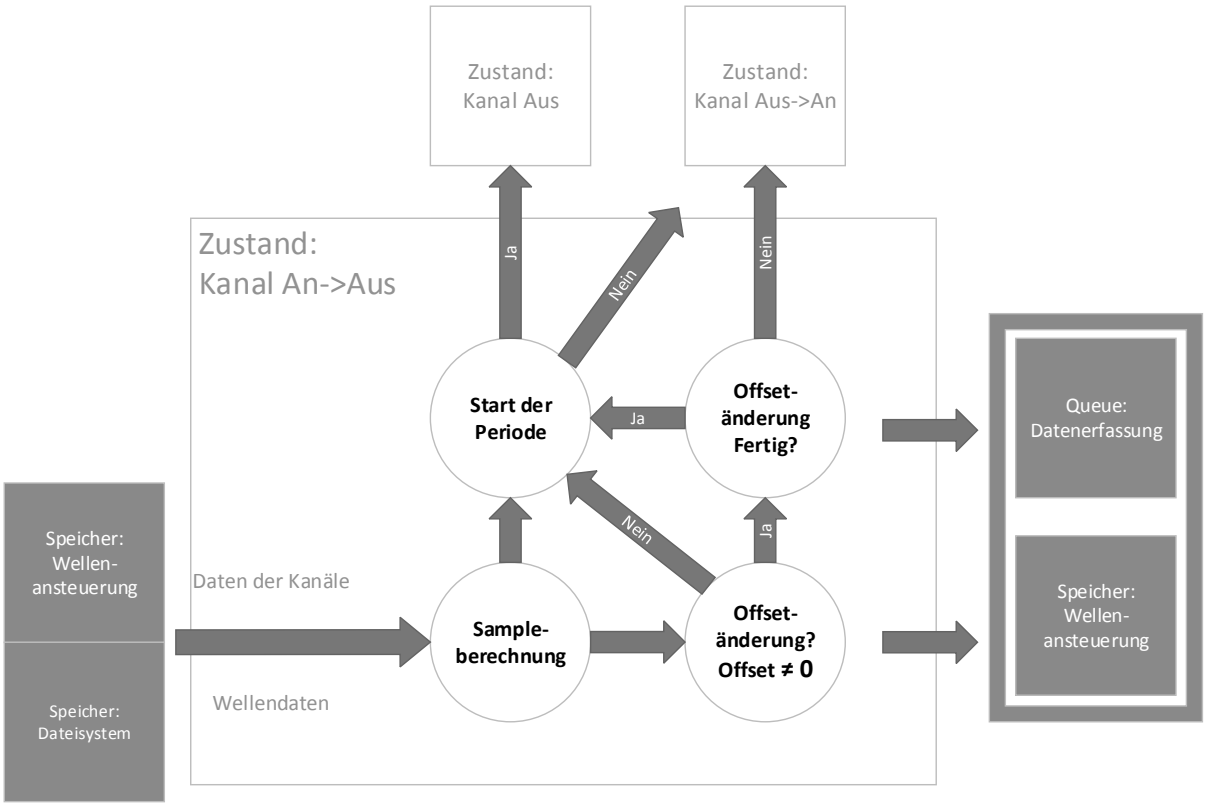

Abbildung 45: Funktionsdiagramm Zustand Kanal An->Aus

Die Funktion Sampleberechnung wird aufgerufen, um den Start einer neuen Periode des Signals abzufragen. Liegt eine Offsetänderung am Kanal vor, fährt der Aktuator über eine Rampenfunktion an den Nullpunkt. Während der Offsetänderung werden die Wellenpunkte an die Queue des Teilprogramms *Datenerfassung* übergeben. Ist der Nullpunkt bei der Offsetänderung erreicht, wechselt der Zustandsautomat in den *Zustand Kanal->Aus* und übergibt die Welleninformationen an den Speicher des Teilprogramms *Wellenansteuerung*. Der Zustandsautomat bleibt so lange in diesem Zustand, bis der Kanal wieder eingeschaltet wird und die Durchführung der Zustände von vorne beginnt.

8. Systemvalidierung

Entsprechend der vorab entworfenen Validierungsstrategie wurden das Kommunikationsprotokoll, die Echtzeitfähigkeit und die Reaktionszeit des Systems sowie die Implementierung zur automatischen Erzeugung der glatten Signalübergänge bei Parameterwechseln getestet.

Kommunikation

Zur Überprüfung des entwickelten Kommunikationsprotokolls, wurde die Software Wireshark benutzt. Bei dieser Software handelt es sich um ein Paket-Sniffer, der den Datenverkehr im Netzwerk aufzeichnet. Die Inhalte der Datenpakete werden dekodiert und lesbar dargestellt. Über diese Software wurde der Datenverkehr zwischen der Benutzerschnittstelle und zwei Master-Slave-Systemen aufgezeichnet (Abb. 46).

626 17.787142000	134.94.232.251	134.94.232.46	TCP	386
627 17.787521000	134.94.232.46	255.255.255.255	UDP	48
628 17.788615000	134.94.232.141	255.255.255.255	UDP	60
683 18.789264000	134.94.232.46	134.94.232.141	TCP	66
686 18.790045000	134.94.232.141	134.94.232.46	TCP	66
727 19.389703000	134.94.232.46	134.94.232.141	TCP	242
735 19.587678000	134.94.232.251	134.94.232.46	TCP	390
736 19.589684000	134.94.232.141	134.94.232.46	TCP	60
737 19.589734000	134.94.232.46	134.94.232.141	TCP	242
741 19.787262000	134.94.232.46	134.94.232.251	TCP	54
742 19.787681000	134.94.232.251	134.94.232.46	TCP	390
743 19.789678000	134.94.232.141	134.94.232.46	TCP	60
744 19.789719000	134.94.232.46	134.94.232.141	TCP	242
752 19.987285000	134.94.232.46	134.94.232.251	TCP	54
753 19.987688000	134.94.232.251	134.94.232.46	TCP	390
754 19.989706000	134.94.232.141	134.94.232.46	TCP	60
755 19.989760000	134.94.232.46	134.94.232.141	TCP	242
769 20.187275000	134.94.232.46	134.94.232.251	TCP	54
770 20.187898000	134.94.232.251	134.94.232.46	TCP	390
771 20.189693000	134.94.232.141	134.94.232.46	TCP	60
772 20.189733000	134.94.232.46	134.94.232.141	TCP	242

Abbildung 46: Datenverkehr zwischen der Benutzerschnittstelle und zwei Master-Slave-Systemen in der Software Wireshark

Die IP-Adressen sind wie folgt aufgeteilt (Tab. 4).

Tabelle 4: IP-Adressen der Benutzerschnittstelle und zwei Master-Slave-Systemen

IP-Adresse	System
134.94.232.251	Benutzerschnittstelle
134.94.232.46	Master-System
134.94.232.141	Slave-System

Nach dem Kommunikationsschema (s. Kap. 7.3) stellt die Benutzerschnittstelle eine TCP/IP-Verbindung mit dem Master-System her (Abb. 46: Zeile 1). Danach sendet das Master-System ein UDP Broadcast ins Netzwerk (Abb. 46: Zeile 2). Das UDP-Broadcast wird vom Slave-System mit einem UDP-Acknowledge beantwortet (Abb. 46: Zeile 3). Das Master-System stellt dann eine TCP/IP-Verbindung mit dem Slave-System her (Abb. 46: Zeile 4,5). Anschließend werden die erhaltenen Daten der Benutzerschnittstelle kontinuierlich vom Master-System ans Slave-System gesendet (Abb. 46: ab Zeile 6).

Durch Überprüfung des Datenverkehrs der verschiedenen Pakete konnte das entwickelte Kommunikationsprotokoll nachvollzogen werden.

Echtzeitige Datenerfassung

Zur Verifizierung des Teilprogramms *Datenerfassung*, wurde die Messung der Zeitkonstante und der Echtzeitfähigkeit des Systems durchgeführt.

Messung der Zeitkonstante:

Zur Messung der Zeitkonstante wurde eine Sinuswelle mit einer Amplitude von 1 Volt und einer Frequenz von 100 Hz erzeugt. Die errechneten Wellenpunkte wurden dem Teilprogramm *Datenerfassung* übergeben und mit einer Samplerate von 20 µs am analogen Ausgang übergeben (Abb. 47).

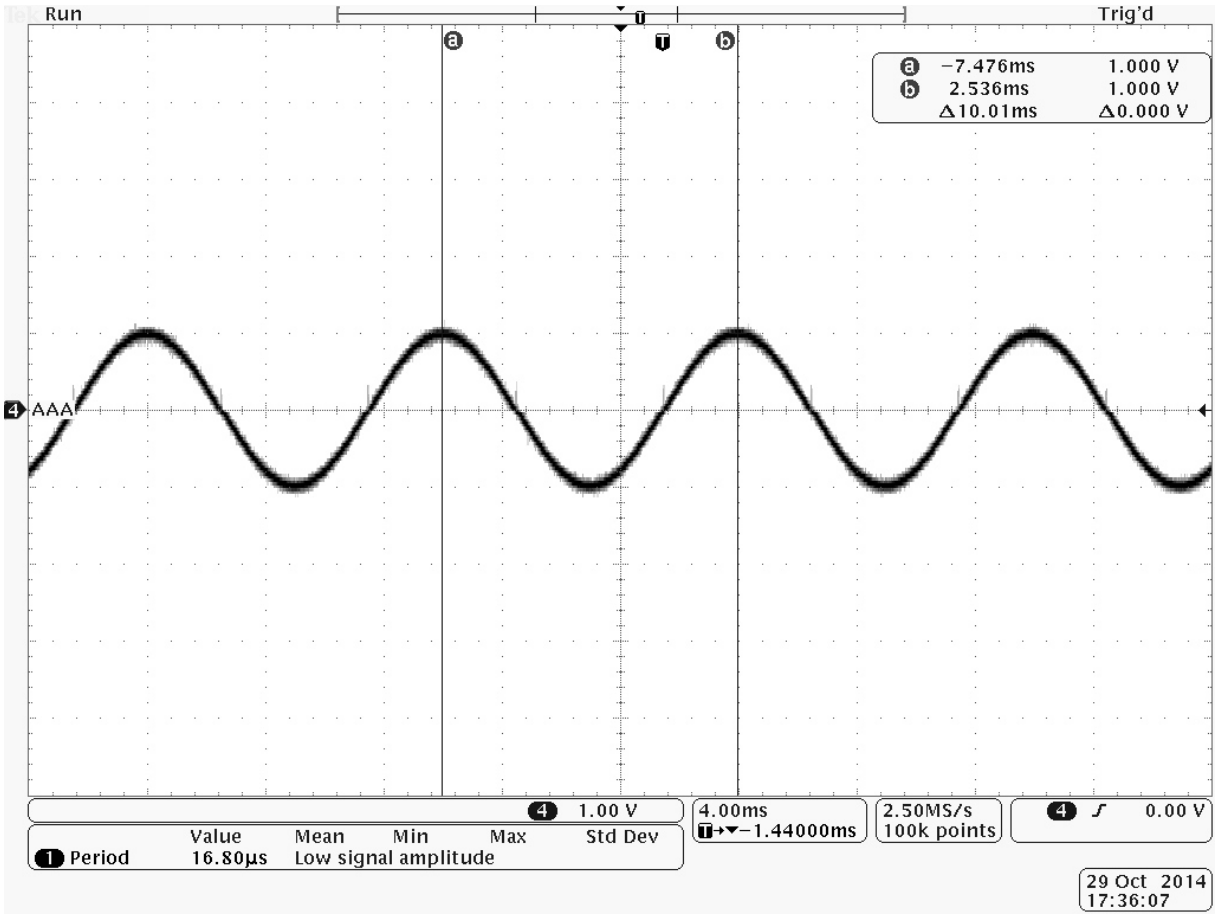

Abbildung 47: Messung der Zeitkonstante eines Sinussignals von 100 Hz

Die Messung mit dem Oszilloskop [20] ergab eine Periodendauer von 10 ms. Der gemessene Signalverlauf entsprach exakt der vorgegebenen Sinuskurve. Damit wurde die Funktion der zeitgesteuerten Schleife erfolgreich validiert.

Echtzeitfähigkeit des Systems:

Mit einem dafür entwickelten Testprogrammkann die Echtzeitfähigkeit des Systems überprüft werden. Der analoge Ausgang 1 wird mit den analogen Eingang 0 verbunden. Wenn der Schalter im Programm betätigt wird, schaltet der analoge Ausgang 1 auf ein Volt hoch. Liegt eine Spannung am analogen Eingang 1 an, soll der analoge Ausgang 0 innerhalb von 20 µs folgen (Abb. 48).

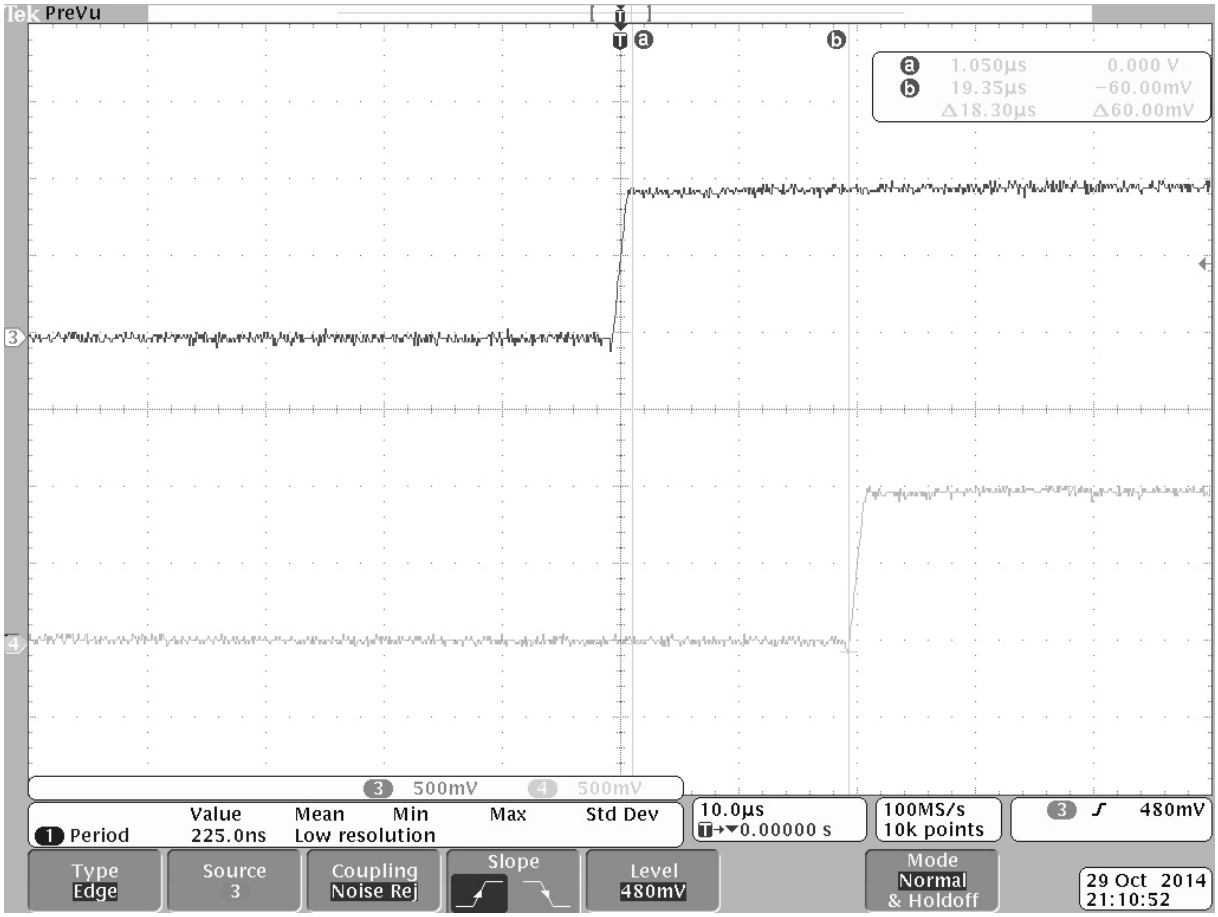

Abbildung 48: Ausgang („grau“) folgt im Idealfall dem Eingang („schwarz“) innerhalb von 20 µs

Die Messung mit dem Oszilloskop ergab eine Reaktionszeit des Systems von unter 20 µs. Mit dem Nachweis der Einhaltung einer fest definierten Reaktionszeit konnte die Echtzeitfähigkeit des Systems erfolgreich validiert werden.

Glatte Signalübergänge

Zur Überprüfung der Funktionalität glatter Signalübergänge wurden die Wellenänderungen während der Entwicklungsphase über einen Wellenform-Graphen in LabVIEW simuliert. Die Erzeugung einer veränderten Welle erfolgt innerhalb einer halben Periode.

Phasenverschiebung:

Bei einer Phasenverschiebung der Sinuswelle um φ = 90° ersetzt eine „Übergangswelle“ eine halbe Periode des Signals. Die Wellenlänge wird verkürzt (Abb. 49).

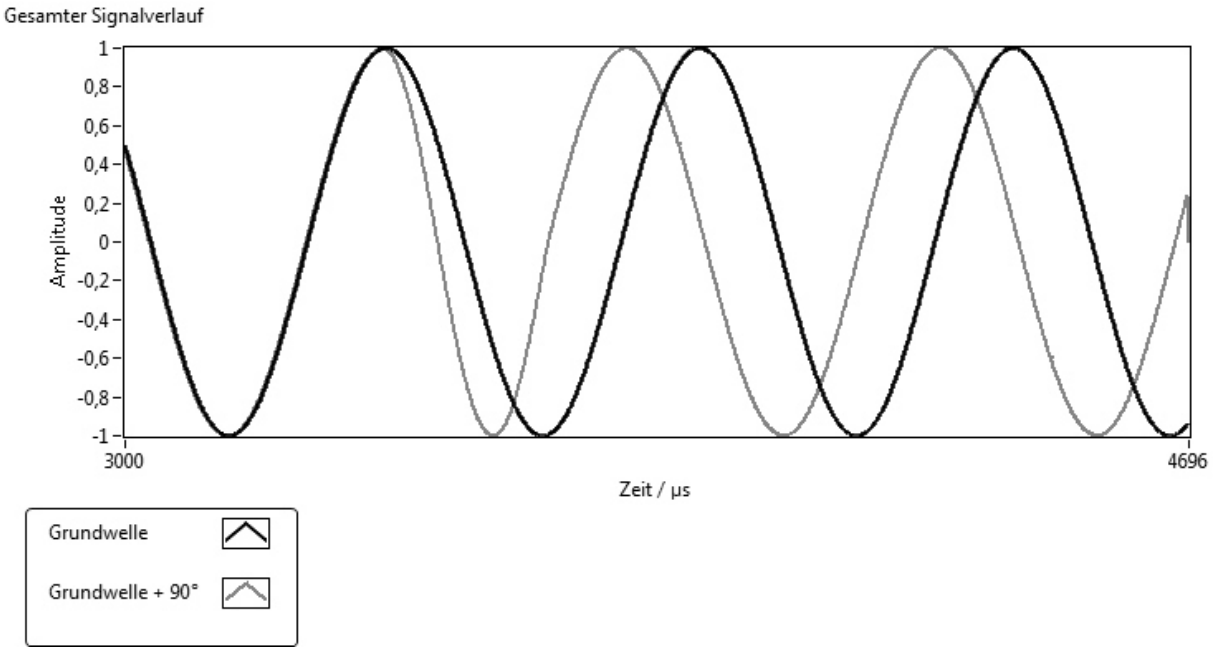

Abbildung 49: Simulation einer Phasenverschiebung um φ = 90° der laufenden Sinuswelle

Offsetänderung:

Die Änderung des Signaloffsets kann in positiver und negativer Richtung erfolgen. Der Zeitpunkt der eingehenden Parameteränderung beeinflusst die Zeit bis zur Umsetzung der Offsetänderung. Tritt eine positive Änderung auf, wird bis zur nächsten Flanke mit positiver Steigung gewartet. Negative Änderungen werden innerhalb der nächsten Flanke mit negativer Steigung durchgeführt.

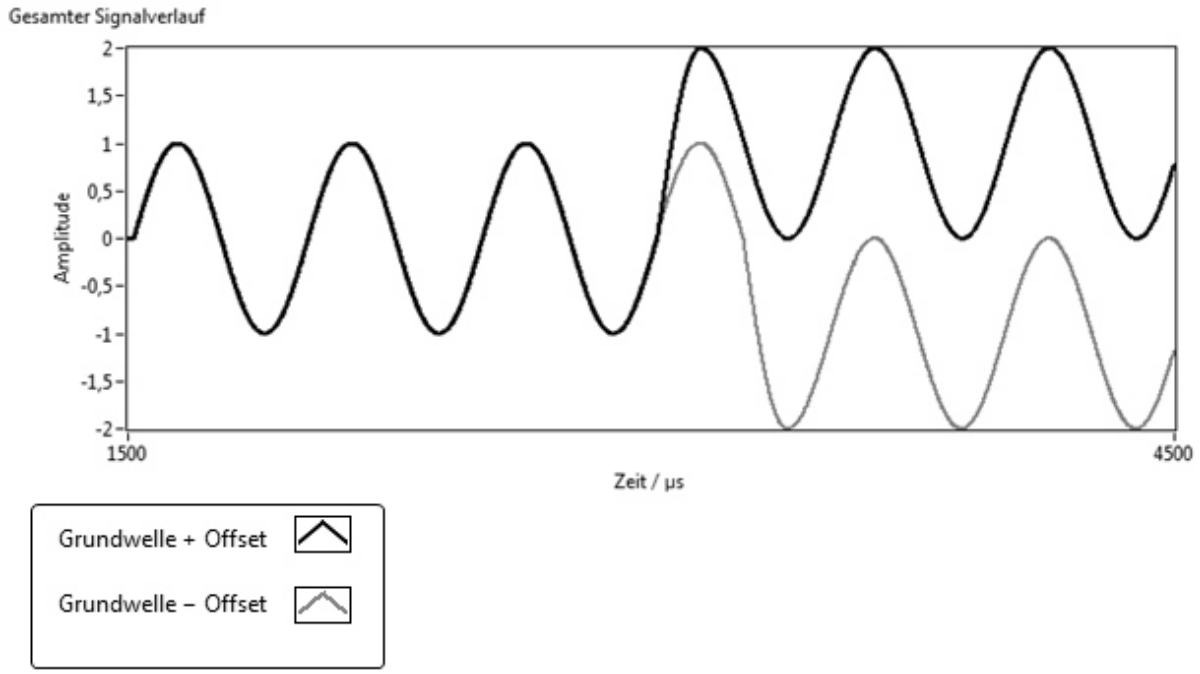

Abbildung 50: Simulation einer positiven und negativen Änderung des Signaloffsets einer laufenden Sinuswelle

Amplitudenänderung:

Die Amplitudenänderungen werden wie die Offsetänderungen umgesetzt. Der Wert der Amplitude kann beliebige positive Werte annehmen. Durch die analoge Ausgabe ist dieser Wert auf 10 Volt beschränkt.

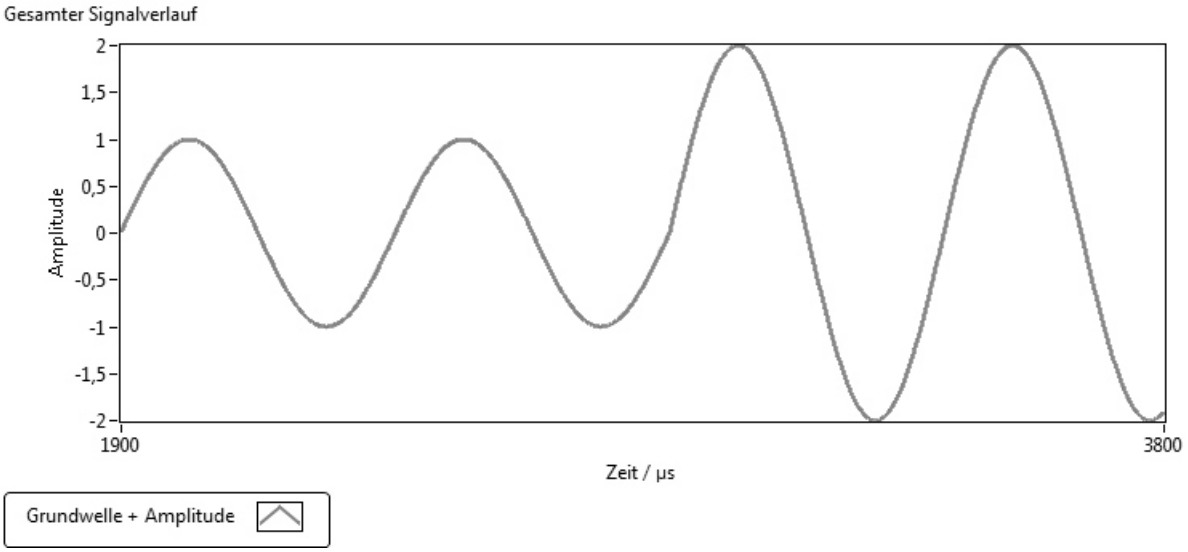

Abbildung 51: Simulation einer Amplitudenänderung der laufenden Sinuswelle

Frequenzänderung:

Mit einer Frequenzänderung von 100 Hz nach 200 Hz ändert sich der Signalverlauf entsprechend der Abbildung 49. Der Übergang ist glatt und erfolgt innerhalb einer halben Periode.

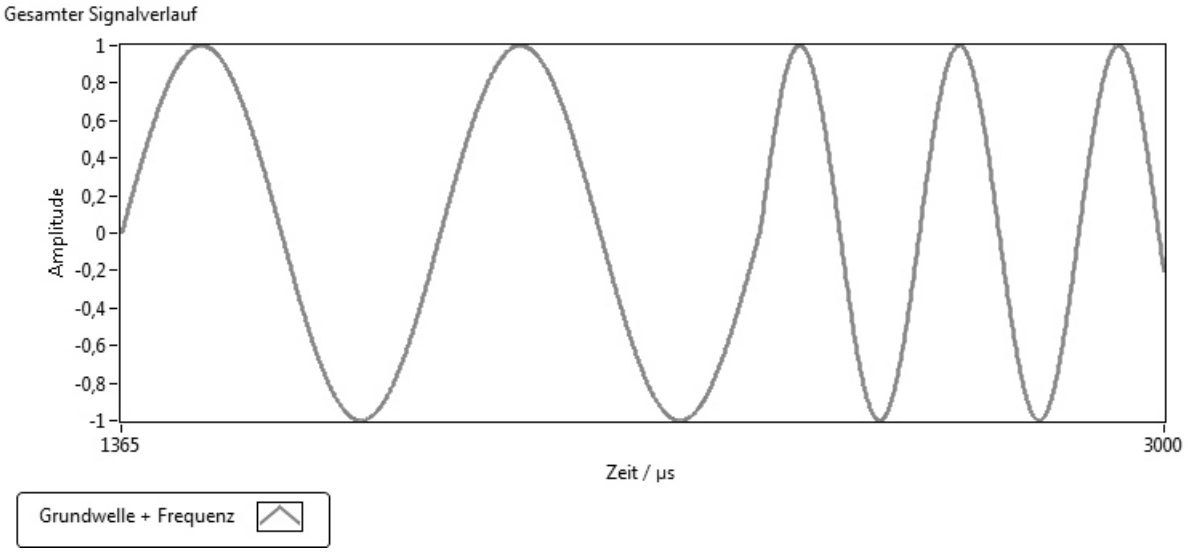

Abbildung 52: Simulation einer Frequenzänderung der laufenden Sinuswelle

Wellenformänderung:

Für die Wellenformänderung ist ein zusätzlicher Datensatz im DAQ-System notwendig. Der Übergang erfolgt zu Beginn einer neuen Periode. Die vorher beschriebenen Änderungen der Wellenparameter funktionieren, sofern der abgespeicherte Signalverlauf periodisch ist.

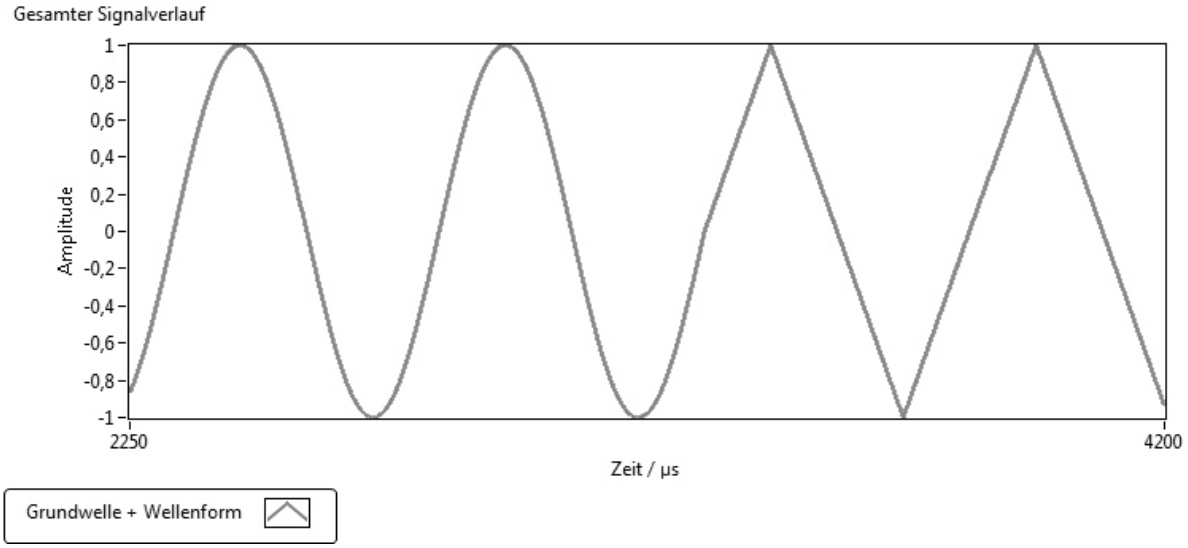

Abbildung 53: Simulation einer Wellenformänderung der laufenden Welle

Wellenänderung:

Ändert der Benutzer alle Wellenparameter gleichzeitig mit einer Nachricht, werden die beschriebenen Wellenänderungen in folgender Reihenfolge ausgeführt:

1. Phasenverschiebung
2. Amplituden- und Frequenzänderung
3. Offsetänderung

Innerhalb einer Periode ist dieser Vorgang abgeschlossen. Die Änderung der Wellenform wird gesondert behandelt. Es wurde gezeigt, dass die entwickelten glatten Signalübergänge korrekt umgesetzt werden. Bei gleichzeitiger Änderung von Phase, Amplitude und Frequenz ist eine Umsetzungszeit unterhalb von einer halben Periode sichergestellt. Wenn dazu noch das Offset innerhalb der Wellenform verändert wird, erhöht sich diese Zeit noch auf maximal eine Periode. Da bei dem verwendeten Aktuator-System allerdings das Offset über eine Gleichstromansteuerung realisiert wird, bleibt die Reaktionszeit unterhalb einer halben Periode.

9. Zusammenfassung und Ausblick

Das Ziel dieser Arbeit bestand darin, eine echtzeitfähige Aktuator-Ansteuerung für die geregelte Erzeugung von transversalen Oberflächenwellen auf einer ebenen Platte im Windkanal zu implementieren. Hintergrund dieser Aufgabenstellung ist die DFG-Forschergruppe „Aktive Widerstandsreduktion durch wellenförmige Oberflächenoszillation“ innerhalb derer das Zentralinstitut für Engineering, Elektronik und Analytik ZEA-2 (Systeme der Elektronik) der Forschungszentrum Jülich GmbH das Teilprojekt „Entwicklung eines echtzeitfähigen Aktuator-Sensor-Netzwerks“ bearbeitet.

Die Ansteuerung der Aktuatoren erfolgt über die analogen Ausgänge des in Kapitel 4 beschriebenen DAQ-Systems mit dem Betriebssystem LabVIEW Real-Time OS der Firma National Instruments. Mit der Anwendungssoftware LabVIEW Real-time 2013, sollten Programme entwickelt werden, um:

- mind. 20 Aktuatoren anzusteuern
- eine Wellenform mit einer Aktuierungsrate von 100 Hz mit mind. 200 Abtastwerten pro Periode zu generieren
- die Skalierbarkeit des Systems bezüglich der Anzahl der Aktuatoren zu gewährleisten
- ein echtzeitiges synchrones Auslesen des Eingangs und Steuern des Ausgangs als Vorbereitung einer Wellenregelung zu implementieren
- ein Kommunikationsprotokoll zum Verbinden mehrerer DAQ-Systeme zu realisieren
- eine Reaktionszeit des Systems auf Wellenänderungen von unter einer Periode zu verwirklichen

Für die Umsetzung der Aufgaben wurde zuerst das bestehende nicht echtzeitfähige System zur Ansteuerung der Aktuatoren analysiert. Das bestehende System beinhaltete

- eine Ansteuerung für 10 Aktuatoren
- eine Kommunikation zu einem DAQ-System
- eine Reaktionszeit des Systems auf Wellenänderungen von max. 2 Perioden
- kein synchrones Auslesen der Eingänge und Steuern der Ausgänge

Durch die Umstellung auf ein Echtzeitbetriebssystem konnten die Funktionen des bestehenden Systems erweitert und verbessert werden.

Dafür wurde ein neues Systemkonzept für die Kommunikation, die echtzeitfähige Wellenänderung und Wellenregelung erstellt. Für die Überprüfung der Entwicklungen wurde zu Beginn der Arbeit eine Validierungsstrategie festgelegt.

Die Kommunikation mit mehreren DAQ-Systemen erfolgt über ein selbst entwickeltes Master/Slave-Kommunikationsprinzip. Die erhaltenen Daten der „model in the loop“ Netzwerk-Simulation werden mit gemeinsamen Zeitstempeln versehen, um eine gleichzeitige Aktuierung der Systeme zu ermöglichen.

Auf Basis des Echtzeitbetriebssystems konnte unter Verwendung einer fest definierten Samplerate ein Algorithmus zur Umsetzung einer Wellenänderung schneller als eine halbe Wellenperiode verwirklicht werden. Zudem konnte durch eine zeitgesteuerte Schleife die Ein- und Ausgänge synchronisiert werden. Zu einem Späteren Zeitpunkt kann die Wellenregelung in diese zeitgesteuerte Schleife integriert werden. Für die in der zweiten Förderphase der DFG-Forschergruppe zu entwickelnde Wellenregelung konnte somit die technische Grundlage geschaffen werden.

Aufbauend auf den Ergebnissen dieser Arbeit können nun Funktionstests des Gesamtsystems mit Aktuatoren, Sensoren und Ankopplung an die „model in the loop“ Netzwerk-Simulation des Teilprojektes für Versuche im Windkanal zur Überprüfung der Robustheit der Aktuator-Ansteuerung durchgeführt werden.

Bei einem Ausbau des Systems für den Betrieb von deutlich mehr Aktuatoren muss eine Synchronisation der Zeitgeber im Netzwerk sichergestellt werden. Dies kann mit einem entsprechenden Zeitsynchronisationsprotokoll realisiert werden.

Mit Erhöhung der Samplerate kann das DAQ-System an seine Grenzen stoßen. Die Umsetzung des hier entwickelten Systems auf FPGA-Module würde deutlich höhere Sampleraten ermöglichen. Zusätzlich wäre damit eine Erweiterung des Algorithmus für nicht periodische Signale realisierbar. Entsprechend können die Ergebnisse dieser Arbeit mit Hilfe eines zukünftigen Einsatzes von FPGA-Karten die Reaktionszeit des Systems in Bezug auf die Wellenänderung/-regelung noch deutlich verbessern.

Literaturverzeichnis

[1] DFG (Deutsche Forschungsgemeinschaft), GEPRIS (Geförderte Projekte) FOR1779 [22.10.2014], http://gepris.dfg.de/gepris/projekt/202175528

[2] Dück, M.; Kaparaki, M.; Srivastava, S.; van Waasen, S.; Schiek, M., "Development of a real time actuation control in a network-simulation framework for active drag reduction in turbulent flow," Automatic Control Conference (CACS), 2013 CACS International , vol., no., pp.256,261, 2-4 Dec. 2013 doi: 10.1109/CACS.2013.6734142

[3] IT-Lexikon, Fachwissen für IT-Professionals, [22.10.2014], http://www.itwissen.info/definition/lexikon/Ethernet-Ethernet.html

[4] Wireshark, [22.10.2014], https://www.wireshark.org/

[5] NI (National Instruments), Zeitgesteuerte Schleifen [22.10.2014], http://zone.ni.com/reference/de-XX/help/371361H-0113/glang/timed_loop/

[6] Dück, M; S. van Waasen, S.; Schiek, M.; „Entwicklung eines echtzeitfähigen Aktuator-Sensor-Netzwerk“, interner Vortrag, [24.10.2014]

[7] Lorrain, P.; Corson, D.; Lorrain, F.; "Elektromagnetische Felder und Wellen : unter Berücksichtigung elektrischer Stromkreise", S.293, "Lorentzkraft", [22.10.2014], ISBN: 3110122324

[8] NI (National Instruments), PXI-System [22.10.2014], http://www.ni.com/images/features/us/101020_fg_pxi.jpg

[9] NI (National Instruments), 1.73 GHz Quad-Core Embedded Controller for PXI Express NI PXIe-8133 [22.10.2014], http://sine.ni.com/ds/app/doc/p/id/ds-267/lang/de

[10] NI (National Instruments), High-Speed Voltage Output – Up to 1 MS/s/Channel, up to 16 Bits, up to 32 Channels [22.10.2014], http://sine.ni.com/ds/app/doc/p/id/ds-157/lang/de

[11] NI (National Instruments), High-Speed M Series Multifunction Data Acquisition - 16-Bit, up to 1.25 MS/s, up to 80 Analog Inputs [22.10.2014], http://sine.ni.com/ds/app/doc/p/id/ds-22/lang/de

[12] IT-Lexikon, Fachwissen für IT-Professionals, [22.10.2014], http://www.itwissen.info/definition/lexikon/PCI-Express-PCIe-PCI-express.html

[13] NI (National Instruments), Manual NI-6722/6723, [29.10.2014], http://www.ni.com/pdf/manuals/370822c.pdf

[14] NI (National Instruments), Was versteht man unter einem Echtzeitbetriebssystem? [22.10.2014], http://www.ni.com/white-paper/3938/de/

[15] IT-Lexikon, Fachwissen für IT-Professionals, [22.10.2014], http://www.itwissen.info/definition/lexikon/Master-Slave-Betrieb-master-slave-operations.html

[16] IT-Lexikon, Fachwissen für IT-Professionals, [22.10.2014], http://www.itwissen.info/definition/lexikon/user-datagram-protocol-UDP-UDP-Protokoll.html

[17] IT-Lexikon, Fachwissen für IT-Professionals, [22.10.2014], http://www.itwissen.info/definition/lexikon/transmission-control-protocol-internet-protocol-TCP-IP-TCP-IP-Protokolle.html

[18] IT-Lexikon, Fachwissen für IT-Professionals. [22.10.2014], http://www.itwissen.info/definition/lexikon/Warteschlange-Q-queue.html

[19] John E. Hopcroft, Jeffrey Ullman: Einführung in die Automatentheorie, formale Sprachen und Komplexitätstheorie. 2. Auflage. Addison-Wesley, Bonn, München 1990 (Originaltitel: Introduction to automata theory, languages and computation), ISBN: 3-89319-181-X.

[20] Tektronix, MDO4000 Mixed Domain Oscilloscope [22.10.2014], http://de.tek.com/datasheet/mdo4000/mdo4000-series-datasheet

Jül-4384
März 2015
ISSN 0944-2952