Technische Universität Ilmenau
Fakultät für Informatik und Automatisierung
Institut für Theoretische und Technische Informatik
Fachgebiet Rechnerarchitekturen

Verantwortlicher Hochschullehrer: Prof.Dr.-Ing.habil. W. Fengler

Betreuer: Dr.-Ing. Bernd Däne

Modellierung eines Messsystems mit dem Modellierungswerkzeug ML-Designer

Studienarbeit 2002

Vorgelegt von: Kirke Rimbach
Geboren am: 3. September 1977
Studiengang: Informatik
Beginn der Arbeit: 1. November 2001
Abgabe der Arbeit: 11. April 2002

Ilmenau, den 11. April 2002

TABLE OF CONTENTS

1 Zusammenfassung 3
2 Überblick über das Gesamtsystem 4
3 Das Modellierungswerkzeug ML-Designer 6
4 Die Regelstrecke, das Atomkraftmikroskop (AFM) 9
4.1 Das Masse-Feder-Dämpfer Modell eines ATM 9
4.2 Erste mathematische Vorbetrachtungen 10
4.3 Das Modell der Abtastspitze 11
4.4 Die Interaktionskräfte zwischen Spitze und Probe 12
4.5 Erweiterungsmöglichkeiten 13
5 Die interferometrische Positionsmessung 14
5.1 Das Funktionsprinzip 14
5.2 Das Störungsmodell 15
5.3 Die Implementierung 16
6 Der Rechnerteil 19
6.1 Die Offset- und Verstärkungsfehlerkorrektur 19
6.2 Der adaptive Filter 21
6.3 Die digitale Vierfachinterpolation 22
6.4 Das Zählen der Quadranten 23
6.5 Die resultierende Positionsbeschreibung 24
7 Der Regler 25
7.1 Die X,Y Regler 25
7.2 Das Problem 26
7.3 Der Z Regler 26
8 Weitere Module des Simulationsmodells 27
9 Integration und Tests 28
9.1 System ohne Filter und geringer Simulationszeit 29
9.1.1 Beschreibung des Aufbaus 29
9.1.2 Eingestellte Parameter 30
9.1.3 Veränderbare und getestete Parameteränderungen bzw. Fehler 31
9.2 System mit Filter und sehr hoher Simulationszeit 32
9.2.1 Anfangsstabilitätsanpassung 32
9.2.2 Anpassung der Reglerparameter 32
9.2.3 Abhängigkeit der Anzahl Filterkoeffizienten von der Frequenz 33
10 Bewertung und Abschluss 34
10.1 Konventioneller Entwurfsvorgang 34
10.2 Geänderte Entwurfsmethodik 35
10.3 Mögliche Verbesserungen 36
10.4 Abschluss 37
11 Erste Schritte mit dem Modell 38
11.1 Die Quellen 38
11.2 Extrahieren der Quelle 38
11.3 Starten von mld 38
11.4 Designansicht des Systems „modell“ 38
11.5 Starten der Simulation 39
11.6 Ende der Simulation und Ergebnisauswertung 40
12 Quellen 41
13 Abbildungsverzeichnis und Anhang 42

1 Zusammenfassung

Für den Aufbau eines realen Meßsystems soll mit dieser Arbeit untersucht werden, in wie weit es möglich ist, dieses vor seiner eigentlichen Realisierung zu simulieren. Es ist mit Hilfe des Modellierungswerkzeuges „ML-Designer“ ein simulierbares Systemmodell aufgebaut worden. In ihm sollen physikalische, technische und algorithmisch softwaretechnische Verbindungen und Abläufe sichtbar gemacht werden.
Das betrachtete reale Objekt ist ein Atom Kraft Mikroskop (Atomic Force Microscope AFM). Mit dem Design von Differentialgleichungen wurde das dynamische Verhalten des AFM nachgebildet. Es liefert relativ realistische Messwerte, die für die weitere Verarbeitung benötigt werden, um das AFM in gewünschter Art und Weise benutzen zu können. Das Bewegungsverhalten wird mit Laserinterferometern gemessen. Die Störungsbehafteten Signale sind der Ausgangspunkt zur weitern Verarbeitung in Digital Signal Prozessoren (DSP). Durch digitale Filter, Regler und Ablaufsequenzen werden System und Benutzerinformationen verarbeitet und in eine Kenngröße für das AFM umgewandelt. Durch eine Vielzahl von Modellparametern kann eine Anpassung an die Wirklichkeit gemacht werden. Auf die Implementierungen im Modellierungswerkzeug wird tief greifend eingegangen, weil sie Ausgangspunkt für die DSP Softwareentwicklung (Codegenerierung) sein kann. Über die im System beobachtbaren Regelungs- und Messwertverarbeitungs-vorgänge sollen Aussagen über Qualität und Anforderungen an das Messwertverarbeitungssystem gemacht werden können. In den abschließenden Betrachtungen werden momentane Schwächen des Entwicklungswerkzeuges ML-Designer gezeigt und wünschenswerte Erweiterungen, die speziell im Bereich des „rapid protyping“ von Vorteil sind, genannt. Für eine Art „starting guid“ wurde das Schlusskapitel benutzt, um einen schnellen Start beim Umgang und Testen des entworfenen Systemmodells zu gewährleisten.

2 Überblick über das Gesamtsystem

Das Raster-Tunnel-Mikroskop wurde 1982 in einem Schweizer IBM Labor entwickelt. Damit war es erstmals möglich atomare Strukturen und Ausmaße aufzulösen. Die so genannten Elektronenmikroskope können aber nur unter bestimmten Umständen die atomare Oberflächenbeschaffenheit von Materialien wiedergeben. Da man auch die Eigenschaften nicht elektrisch leitfähiger Stoffe untersuchen wollte, wurde 4 Jahre später das Atom-Kraft-Mikroskop vorgestellt. Durch immer komplexer werdende Technologien stieg auch die Anforderung an die Genauigkeit dieser Mikroskope. Um feine Auflösungen im Nanometerbereich zu erzeugen mussten hochgenaue Positionier und Längenmeßsysteme entwickelt werden. [1][2]

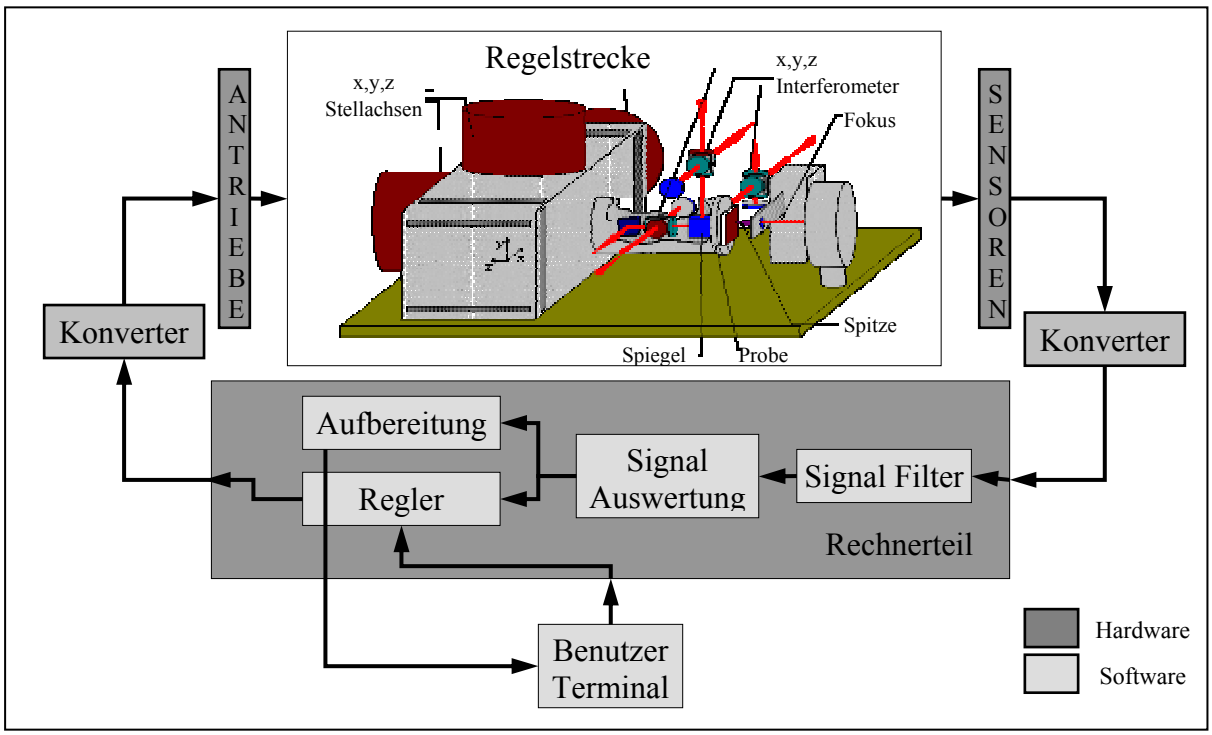


Abbildung 2-1: Schematische Darstellung des Gesamtsystems

Das Atom-Kraft-Mikroskop

In der Abbildung 2-1 ist der prinzipielle Aufbau eines Kraft Mikroskops als Regelstrecke gekennzeichnet. Das Funktionsprinzip ist am leichtesten mit dem eines Plattenspielers zu vergleichen. Eine Nadel, dessen Spitze annähernd den Radius eines Atoms hat, wird gleichmäßig über eine Probenoberfläche geführt. Dabei wird Zick Zackförmig eine Fläche in x und y Richtung abgetastet. Wenn man von einer konstanten Kraft ausgeht, die zwischen Spitze und Materialoberfläche wirkt, muss in z Richtung eine Wegänderung messbar sein, die direkt proportional zur Oberflächenbeschaffenheit der Probe ist. Die Auslenkung in x, y und z Richtung wird durch Stellachsen realisiert, die nach piezoelektrischen, dynamischen oder magnetischen Wandlerprinzip arbeiten. Die Wandler weisen bei Einprägung von Strom oder Spannung eine relative Längenänderung auf. Der Zusammenhang zwischen Strom/Spannung und Länge ist hochgradig nicht linear. Um trotzdem die geforderten konstanten Wegausbreitung zu erzeugen müssen Regelungstechnische Prinzipien benutzt werden. Das hat zur Folge, das die Längenausbreitung in einer geeigneten Art und Weise gemessen werden muss, um für die Regelung benötigte Soll <-> Istwert Vergleiche durchzuführen. Die

Messung der Bewegung in x, y und z Richtung erfolgt nach interferenzoptischem Prinzip mit Hilfe von drei Interferometern. Gleichzeitig kann in einem kleinen Bereich die Bewegung der Spitze über Fokussierung (ähnlich der Focussteuerung von CD Spielern) gemessen werden.

Die Schnittstellen

Die Eingangsgröße für die Regelstrecke ist konstant Strom/Spannung für die Stellachsen. Diese werden durch Treiber realisiert, die je nach Anforderung an Leistung, Fehlertoleranz und Leistungsverlust mehr oder weniger komplex sind. Diese Treiber sind meist überwiegend in analoger Hardware realisiert und aufgebaut. D.h. der Rechner macht eine analoge oder digitale Sollwertvorgabe und die Konverter/Leistungsendstufen erzeugen die geforderte Amplitude. Dabei ist die Ausgabe mit Bauform bedingten Fehlern behaftet.
Die Ausgangsgröße werden durch die Interferometer und den Fokus erzeugt. Die interferenzoptische Längenmessung hat den Nachteil, dass sie die absoluten Abmessungen der Messanordnung in relative Maße umwandelt. Für jede Achse werden zwei analoge periodische Signale erzeugt, die mit Hilfe von Operationsverstärkern aufbereitet werden müssen, bevor sie über AD Wandler digitalisiert werden können. Der Fokus liefert ein Signal, das direkt proportional zur Auslenkung der Spitze ist.
Somit gehen 3 Signalwege in die Regelstrecke, werden dort transformiert, und 7 wieder heraus.

Der Rechnerteil

Die Verarbeitungs- und Steuereinheit hat die Aufgabe die Bedienung des Kraft Mikroskops durch ein externes Benutzer Terminal zu gewährleisten. Es muss die Steuerung des Mess- und Positioniersystems erfüllen. Sowie Messwerte erfassen und Algorithmen auf diese Messwerte anwenden. Dabei werden große Datenmengen verarbeitet. Zur Fehlerkorrektur und Rauschbeseitigung soll unter anderem digitale Filter mit etwa 100 Filterkoeffizienten benutzt werden. Bei der Verarbeitung der Messwerte wird mit einer hohen Auflösung (32-bit-integer-Meßwerte) und einer hohen Abtastrate gearbeitet (120 kHz). Als Prozessor soll ein DSP TMS320C6701 von Texas Instruments eingesetzt werden. Dabei handelt es sich um einen mit maximal 200 MHz getakteten Signalprozessor mit VelociTI-Architektur. Die Architektur implementiert das VLIW-Prinzip (Very Long Instruction Word). Der Prozessor verfügt über 8 funktionelle Einheiten und kann somit bis zu 8 Befehle parallel ausführen. Die neue Architektur wird besonders im Zusammenspiel mit den von Texas Instruments angebotenen Hochoptimierenden Code Generation Tools (C-Compiler, Assembler, Assembly Optimizer) ausgenutzt. Bei der Wahl des Prozessortyps wurde zugunsten dieses Prozessors entschieden, um dem hohen Anspruch an die Rechenleistung zu genügen. Die parallele Abarbeitung von Befehlen ermöglicht es, den Echtzeitanforderungen gerecht zu werden. Der interne Speicher ist in Datenspeicher und Programmspeicher (jeweils 64 kBytes) aufgeteilt. Der Prozessor verfügt weiterhin über 2 Multichannel Buffered Serial Ports (MCBSP), 2 Timer, 4 DMA-Controller und ein 16-bit Host-Interface. Die DMA-Controller spielen eine wichtige Rolle für den Datenzugriff. Interne DMA Controller dienen dazu, die Daten einzuholen, ohne gleichzeitig Prozessorzeit zu verbrauchen. Das macht sich speziell beim Lesen der vielen Messwerte deutlich, da der Zugriff auf den externen Bus wesentlich langsamer ist, als die Verarbeitung.

3 Das Modellierungswerkzeug ML-Designer

ML-Designer ist ein Software System mit dessen Hilfe man besser Systeme, Produkte und sogar Schaltkreise entwickeln kann. [3] Es ist ein Simulationswerkzeug mit dem man physikalische, mathematische, theoretische und technische Modelle aufbauen und virtuell testen kann. ML-Designer soll helfen ein schnelles Design zu erzeugen. Darauf aufbauend kann dann eine Validierung von Funktionalität und Architektur (Systemarchitektur) erfolgen. Dabei geht ML-Designer weit über das eigentliche System-Level Design hinaus und verbindet es sehr stark mit der Entwicklung (auf Hardware und Softwareebene). Durch die als Primitive zur Verfügung gestellte Schnittstelle ist es für Programmierer sehr leicht eigene C++ Programme in die Umgebung von ML-Designer einzubinden und zu benutzen. Es vereinheitlicht den Weg vom Konzept bis zur Implementierung und verringert somit später Kosten bei Designänderungen. ML-Designer baut auf das Konzept von PTOLEMY auf und erweitert es um eine Vielzahl von Features. Es kann in vielen Anwendungsbereichen genutzt werden:

- Netzwerk Architekturen (Warteschlangennetze, Routing, Protokolle)
- System Architekturen (GPS, Telemetrie, Sende und Empfangsstationen)
- Micro und Komponentenarchitektur (Decoder/Encoder, Digitale Bildverarbeitung)
- Funktionale Spezifikationen (Signal Verarbeitungsalgorithmen, Controling)

Durch die Verbindung mit Programmen wie Matlab, Satlab und Tc/tkl können auch deren Eigenschaften genutzt werden. Auch den Anforderungen von Entwicklerteams kann es gerecht werden. Es bietet

- Einfaches Modellieren durch graphische Editor und Einsatz allgemeiner Programmiersprachen für die Beschreibung von Funktionen, Schnittstellen und Architekturen.
- Unterstützt die Validierung vom abstrakten Modell, bis hin zur Implementierung.
- Koordiniert verteiltes Arbeiten von Gruppen über Netzwerk.
- Gute Überschaubarkeit von Komplexen Strukturen durch Hierarchischen Aufbau.

Für die Modellierung des in Kapitel 2 dargestellten Systems wird nur ein Bruchteil der zur Verfügung stehenden Möglichkeiten benötigt.

Simulationsdomänen

Eine der am meisten genutzten Domänen ist die des synchronen Datenflusses (SDF). Da ML-Designer in der Version 2.p04 noch keine kontinuierliche Zeitdomäne besitzt, in der bspw. kontinuierliche Differentialgleichungen simuliert werden können, wird versucht das SDF Konzept dafür zu benutzen. Der SDF ist gekennzeichnet durch einen Datenflussgraph, in dem der Kontrollfluss zur Zeit des Kompilierens vorbestimmt ist. D.h. die Berechnung ist Zeitunabhängig und somit erreicht man ein Verhalten das mit f(x)=y beschrieben werden kann.
Die Domäne der diskreten Ereignisse (Discrete Event DE) funktioniert auf einer anderen Basis. Jedes Ereignis bekommt einen Zeitstempel. Somit werden alle auftretenden Ereignisse in der Simulationszeit aufgereiht und nacheinander abgearbeitet. Es gibt keine Gleichzeitigkeit von Ereignissen. Deswegen ist diese Domäne speziell für Netzwerkanwendungen und Hardwaresimulation geeignet.
Der letzte Bereich der betrachtet werden soll ist die Finite State Machine (FSM). Diese noch experimentelle Domäne ist dafür konzipiert endliche Automaten zu modellieren. Die

Interaktion mit der Umgebung erfolgt durch logische Werte (Boolsche Werte = TRUE/FALSE = 1/0).
Prinzipiell ist es möglich die drei unterschiedlichen Domänen untereinander, durch ein geeignetes Schnittstellenkonzept, zu verbinden. Der unterschied zwischen den Domänen ist, wie der Simulationskernel die einzelnen Teile bearbeitet. Eine detaillierte Zusammenfassung über alle Domänen ist in „MLDesigner Documentation preview Version 2.0“ auf Seite 140 zu finden. [3]

Hierarchische Struktur

Die hierarchische Strukturierung der Modelle ist ein wesentlicher Vorteil dieses Modellierungswerkzeuges. In Abbildung 3-1 ist dieser kurz vorgestellt.

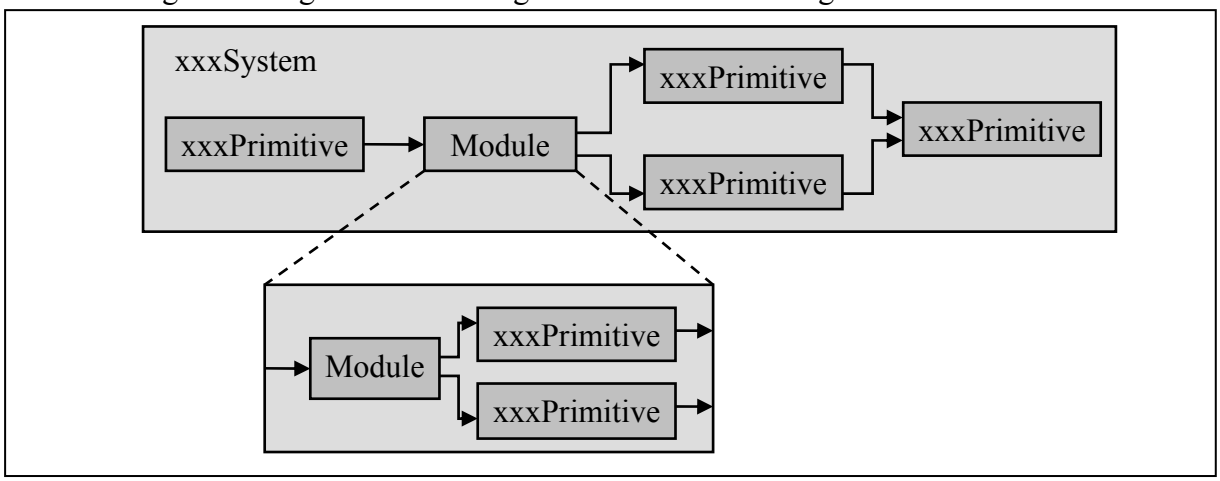


Abbildung 3-1: Hirarchie in ML-Designer

Es gibt folgende Modellkomponenten:

Primitve — Primitive sind die kleinste Modellkomponente in ML-Designer, dessen Funktionalitäten mit C++ Code Fragmenten realisiert sind. Primitive können wohl definierte Ein/Ausgangs Ports und Parameter als Schnittstelle haben. Die Funktionalität von FSM Primitiven wird durch ein FSM Modell (Graph) beschrieben. Sie werden in Ptolemy als Stars bezeichnet.

Module — Module sind Modellkomponenten, die weitere Blöcke (Module, Primitive) einbetten. Sie können wie Primitive Ein/Ausgangs Ports und Parameter besitzen. Module werden als Galaxie in Ptolemy gekennzeichnet.

System — Das System ist die höchste Schicht und enthält eine Menge von untereinander verbundenen Modulen und Primitiven. Es hat keine Ein/Ausgangs Ports. Parameter können dazu benutzt werden, das System zu parametrisieren. Nur Systeme können in Abhängigkeit ihrer festgelegten Domäne simuliert werden. Sie stellen die eigentliche Umgebung (reale Welt) des abstrakten Modells dar und werden in Ptolemy als Universum bezeichnet.

Library — Bibliotheken sind keine Modellkomponenten. Sie dienen der Gruppierung von Primitiven und Modulen nach Verwendung und Aufgabenbereich. ML-Designer stellt eine Reihe vorgefertigter zur Verfügung und erleichtern damit einen ersten Einstieg und dessen Benutzung.

Die Daten, die zwischen Ports, ausgetauscht werden heißen Partikel oder Tokens. Auf die nähere Beschreibung des Port- und Verbindungskonzeptes soll an dieser Stelle nicht näher eingegangen werde.

Elementare Vorgehensweise beim Modellieren

Um komplex dynamische Systeme modellieren zu können, muss als erstes ein mathematisches Modell der benötigten Komponenten entstehen. Erst wenn man die Abläufe versteht, kann man sie richtig interpretieren und modellieren. Ein mathematisches Modell ist ein Algorithmus oder Menge von Gleichungen, die mit einer Menge von Werten ein signifikantes Verhalten des Systems oder Modells hervorrufen. Um Modelle entwickeln zu können, muss man Techniken nutzen die relevant im Vergleich zum betrachteten Modell sind. Der Prozess der Modellentwicklung beginnt mit der Spezifikation der Forderung, die das Modell haben muss. Dabei sind folgende Fragen zu klären.

- Welche Wirkungen sollten in dem Modell integriert sein? Man sollte nur die Effekte benutzen, die unbedingt nötig sind. Dadurch ist es wesentlich besser aufzubauen, weniger komplex und leichter zu testen. Weiterhin benötigt es weniger Rechenleistung.
- Wie detailliert muss das Modell sein? In den meisten Fällen ist ein einfaches Teilmodell alles was benötigt wird.
- Welche Schnittstellen zwischen dem System und der Umgebung gibt es?
- Welche Techniken sollen bei der Systementwicklung benutzt werden?
- Welche Daten müssen gesammelt werden um das Modell anzupassen?
- Welche Rechenleistung steht für die Simulation des Modells zur Verfügung?
- Wie soll das Modell (Teilmodell) implementiert werden für die Verifikation und Validierung?

Diese Fragen sollten zu Beginn an der höchsten Schicht geklärt werden. Das bedeutet im ML-Designer „System“ oder bei Ptolemy dem Universum. Das ist somit das Anfangssystem, welches simuliert werden soll. Diese Fragen sollten dann immer und immer wieder gestellt werden, solange das System in Subsysteme (Module) untergliedert wird. Daraus wird ersichtlich, dass in den höheren Schichten eine wesentlich höhere Abstraktion erfolgt. Die niedrigen Ebenen (Levels) stellen die Implementierung dar.
Der Modellentwickler muss abwägen, welche Details wichtig sind und wie hoch die Abstraktion (Schichten) gewählt werden muss. Denn nicht zu letzt sind das Kriterien, die Kosten und Entwicklungszeit maßgeblich beeinflussen.
Ein guter Ansatzpunkt ist deswegen:

- Starte mit eine relativ simplen Modell.
- Wähle die Anzahl der Details und Schichten nach dem Modell Details.
- Dieser erste Ansatz sollte lauffähig sein und simulierbar sein.
- Nach und nach können weiter Erweiterung vorgenommen werden.
- Da zu beginn meist nicht klar ist, welche Feinheiten das System benötigt, entstehen so sehr große und komplex Simulationen.

Sind die Modul Schnittstellen (Ports) gut definiert, ist festzustellen das selbst tief greifende Erweiterungen keinen Einfluss auf den Rest des Modells haben. Diese Möglichkeiten müssen von einem guten Entwicklungswerkzeug unterstützt werden.

4 Die Regelstrecke, das Atomkraftmikroskop (AFM)

Es gibt unterschiedliche Möglichkeiten eine Oberfläche mit Hilfe einer Spitze im Nanometerbereich abzutasten. Hier sind als erstes voll Kontakt Mode Mechanismen, wie die Reibungskraft Mikroskopie, zu nennen. Hierbei ist die Spitze kontinuierlich in Kontakt mit dem Muster (Probe). Das hat den Nachteil, das die Adhäsionskraft der Spitze auf die Oberfläche der Probe zurückwirkt und diese somit teilweise zerstört. Bei der „intermittent contact AFM“ schwingt die Spitze mit einer sehr hohen Eigenfrequenz. Beim Auftreten von Interaktionen zwischen Spitze und Probe ändert sich diese in Amplitude, Phase und Frequenz. Da bei diesem Verfahren die Spitze überwiegend frei schwingt, können auch organische Materialien untersucht werden.
Doch eins haben alle Verfahren gemein, nämlich die hochgenaue 3-d Positionierung der zu untersuchenden Probe. Von ihr soll ein im Nanometerbereich genaues Abbild der Oberflächenstruktur erzeugt werden. Somit ist die eigentliche Güte, die man auch als Auflösungsgenauigkeit bezeichnen kann, sehr stark von den eingesetzten Technologien abhängig.

4.1 Das Masse-Feder-Dämpfer Modell eines ATM

Aus den Forderungen, dass die Halterung der Probe in 3 Achsen beweglich gelagert sein muss, entsteht eine mechanische Struktur. [4]

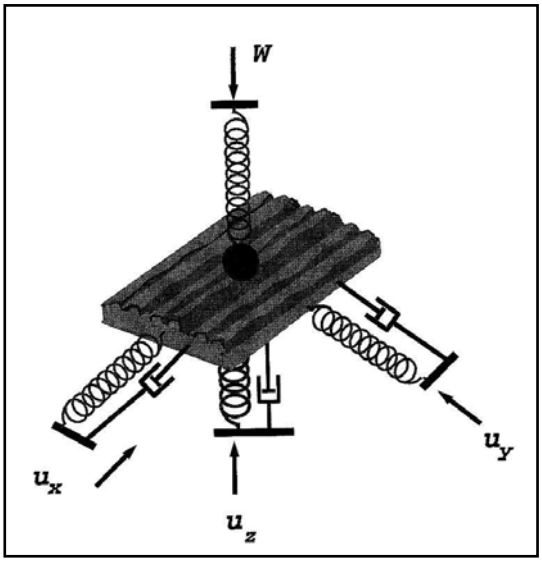


Abbildung 4-1: mechanische Struktur eines AFM

Der eigentliche Mittelpunkt ist die Probe, die durch ihre Masse repräsentiert wird. An ihr greifen in x, y und z Richtung die drei Stellachsen an. Die Eingangsgrößen, welche durch die Stellachsen erzeugt werden, sind mit Ux, Uy und Uz gekennzeichnet. Diese können Kraft oder Weg sein. Die Übertragung auf die Probenmasse erfolgt über das Feder-Dämpfer System. Dabei wird davon ausgegangen, dass die Feder die Kraft nur in die dafür vorgesehen Richtung überträgt. Mit drei Dämpfungs- und Federkonstanten sind die Parameter für die Umwandlung eindeutig bestimmt. Es wird weiterhin davon ausgegangen, dass die Spitze in konstanten Kontakt mit der Probe ist. Da diese mit einer konstanten Last W auf die Oberfläche einwirkt, erzeugt sie eine Gegenkraft in z Richtung. Die Spitze ist als Punktmasse dargestellt, welche mit relativ kleiner Fläche auf die Probenoberfläche einwirkt.

Folgende Formeln repräsentieren die Dynamik der entsprechenden Koordinatenachsen des Modells.

$$\ddot{x} + 2\xi_x \omega_x \dot{x} + \omega_x^2 x = u_x - \frac{1}{m_p} F_{rx}$$

$$\ddot{y} + 2\xi_y \omega_y \dot{y} + \omega_y^2 y = u_y - \frac{1}{m_p} F_{ry}$$

$$\ddot{z} + 2\xi_z \omega_z \dot{z} + \omega_z^2 z = u_z - \frac{1}{m_p} N$$

Mit F_{rx}, F_{ry} sind Reibungskräfte der Spitze auf der Oberfläche in die jeweilige Richtung. Diese Größen werden auf Null gesetzt, da sie nur in einem Reibungskraft Mikroskop von großer Bedeutung ist. N ist die Kraft, die der z Achse entgegenwirkt und m_p die Masse der Probe. $[\xi_x, \xi_y, \xi_z]$ sind die Dämpfungskonstanten und $[\omega_x, \omega_y, \omega_z]$ die Federkonstanten. Die erste Ableitung ist die Geschwindigkeit und die zweite Ableitung die Beschleunigung, jeweils in die betrachtete Richtung. Es sollte ersichtlich sein, dass die Werte für die Koordinatenachsen die Einheit Meter [m] haben. Das bedeutet bei Auflösungsgenauigkeiten von einigen Nanometern [nm] muss mit Potenzen von 10^-9 gearbeitet werden. Da weiterhin in dem Modellierungswerkzeug ML-Designer noch nicht die Möglichkeit besteht kontinuierliche Differentialgleichungen zu simulieren müssen diese diskretisiert werden. D.h. es wird eine Umwandlung aus der CT Domäne (continuous time Domain) in die SDF Domäne durchgeführt. Dafür werden zwei Skalierungsfaktoren eingeführt. ω_0 ist die Skalierung für die Frequenz und $[x_0, y_0, z_0]$ sind die Skalierungsfaktoren für die Achsen (im Modell sind die drei Parameter gleich groß und als achs_norm gekennzeichnet). Folgende Transformationen sind erforderlich:

Alte Variablen	Neue Variablen
x, y, z	$\frac{x}{x_0}, \frac{y}{y_0}, \frac{z}{y_0}$
F_{rx}, F_{ry}, N	$\frac{F_{rx}}{m_p \omega_0^2 x_0}, \frac{F_{ry}}{m_p \omega_0^2 x_0}$
u_x, u_y, u_z	$\frac{u_x}{\omega_0^2 x_0}, \frac{u_y}{\omega_0^2 y_0}, \frac{u_z}{\omega_0^2 z_0}$

Die Realisierung solcher Gleichungen erfolgt über Integratoren. Es müssen somit alle Eingänge und Ausgänge aus dem Modul entsprechend transformiert werden.

4.2 Erste mathematische Vorbetrachtungen

Um einen ersten Eindruck zu gewinnen, wie sich die Dynamik einer Achse verhält wurde ein kleines Modell erzeugt. Dieses wurde in Matlab/Simulink aufgebaut und zeigt auf einfache Weise recht gut die Eigenschaften. Hierzu wurden folgende Werte für Dämpfungs- und Federkonstanten aus den Quellen zu Grunde gelegt.

$[\omega_x, \omega_y, \omega_z]$	[669.07, 407.07, 686.13]
$[\xi_x, \xi_y, \xi_z]$	[0.0366, 0.0293, 0.041]
m_p [kg]	6e-7

Die folgenden Bilder zeigen den Aufbau und das erzeugte Ergebnis. Es ist die Antwort des Systems auf einen Sprung von 40nm zu sehen.

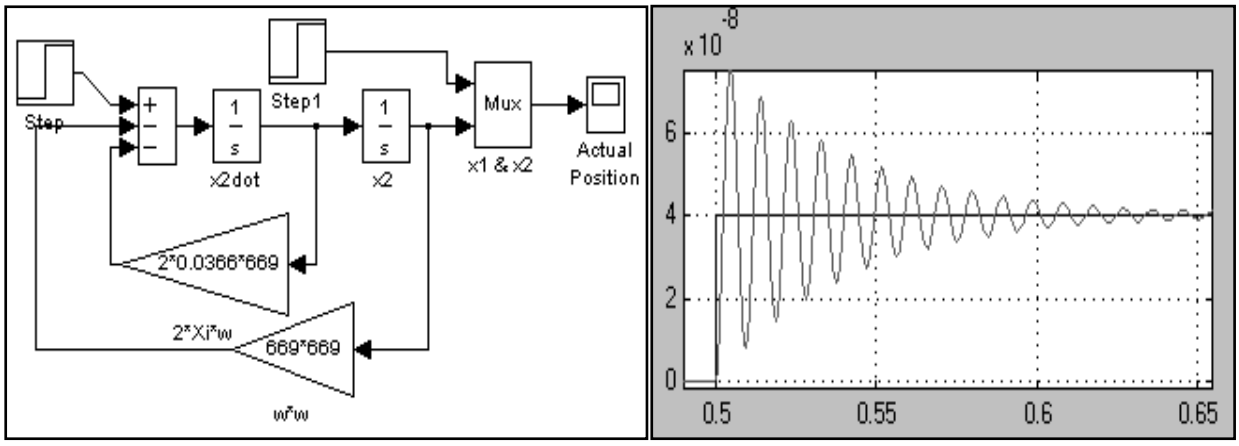


Abbildung 4-2: Aufbau u. Ergebnis der Vorbetrachtung

Diese Ergebnisse sind auch rechnerisch nachvollziehbar. Man hat es mit einem System zu tun, welches mit einem fast 90%igem Überschwingen auf einen Schritt antwortet. Die erste Amplitude ist schon nach 1.49 ms zu finden und erst nach ca. 128 ms ist die Schwingung auf etwa 5% abgeklungen.
Das bedeutet, um exakte schnelle Schritte machen zu können müssen die Kenngrößen der Stellachsen geregelt werden.

4.3 Das Modell der Abtastspitze

Obwohl die Nadel mit der Spitze in Abbildung 4-1 schon kurz dargestellt wurde ist es nötig näher darauf einzugehen. Am einfachsten ist sie wie dargestellt als Masse-Feder System zu interpretieren. Trotzdem wurde im Modell eine Dämpfungskonstante eingeführt, die eine bessere Anschauung und Verständlichkeit realisiert. Die folgende Skizze soll wieder kurz die physikalischen Zusammenhänge darstellen.

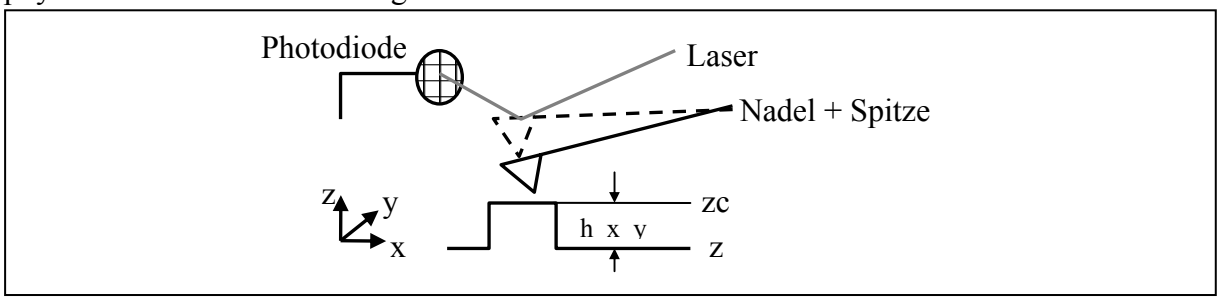


Abbildung 4-3: Koordinatensystem der Spitze

An diesem Punkt sollen nun absolute Positionsbezüge eingeführt werden. Da die Nadel fest montiert ist soll die Position der Spitze zc ohne Vorhandensein einer Probe (h_x_y=0) gleich Null sein. Befindet sich die Spitze in Kontakt so ist auch die Position der z Achse gleich Null. Die Oberfläche der Probe wird durch eine Funktion, welche von x und y (Scannposition) abhängig ist, dargestellt. Somit ist die Position der Spitze durch $z_c = z + h(x, y)$ beschrieben. Hierbei wird die Eindringtiefe (delta(x,y)) in das Material und der Radius der Spitze vernachlässigt. Die eigentliche Bewegung wird über einen Laser und eine Photodiode ermittelt. Es ist zu beachten, dass die Intensität den die Photodiode empfängt auch bei zc=0 einen bestimmten Wert hat. Somit können auch negative zc Werte detektiert werden.

Die komplette Beschreibung der Dynamik der Spitze sieht folgendermaßen aus und beruht auf Werten des Kapitels 4.1.:

$$\ddot{z_c} = -2\xi_c\omega_c(\dot{z_c} - \dot{z}) - \omega_c^2(z_c - h_{x,y} - z) + \frac{1}{m_c} f_z$$

m_c ist die Masse und ω_c, ξ_c sind Feder- Dämpfungskonstanten der Spitze. Für dieses Modul gibt es drei Eingangsgrößen. Das sind die Geschwindigkeit und Position der z Achse und eine nichtlineare Kraft f_z deren Beschreibung im folgenden Kapitel folgt. Für die Federkonstante wurde ein Wert von 2400 Hz festgelegt. Bei (z+h_x_y) Werte die kleiner als Null sind erfolgt das Lösen der Spitze von der Probe. Damit für die Kraft f_z nicht mit sehr hohen Werten gerechnet werden muss, wird zu diesem Zeitpunkt Geschwindigkeit und Position der z Achse, sowie Höhe der Probe für die Eingänge des Moduls auf Null gesetzt.

4.4 Die Interaktionskräfte zwischen Spitze und Probe

Beim abtasten von harten Oberflächen kann das JKR (Johnson-Kendall-Roberts) Kraftmodell genutzt werden. Diese Modell Theorie beschreibt das mechanische Modell des kontinuierlichen Kontaktes. Es enthält sowohl energetische als auch Elastizitätsbetrachtungen. Die Funktion $f_z(z_c)$ berechnet die Kraft aus einer Reihe gegebener Parameter und der Position der Spitze. Aufgrund umfangreicher Berechnungen wurde diese Funktion komplett als Primitive umgesetzt. [5][6]

$$f(z) = \begin{cases} f_0 R\left[-\left(\frac{\sigma}{z}\right)^2 + \frac{1}{30}\left(\frac{\sigma}{z}\right)^8\right] & \text{if } z > z_0 \quad * \\ g_0(z_0 - z)^{3/2} & \text{if } z \le z_0 \quad ** \end{cases}$$

$$\text{mit} \quad g_0 = \frac{8\sqrt{2}}{3\pi\left(\frac{1-\nu_1^2}{\pi E_1} + \frac{1-\nu_2^2}{\pi E_2}\right)}\sqrt{R} \quad \text{und} \quad f_0 = \frac{2}{3}\pi^2\varepsilon\rho_1\rho_2\sigma^4$$

Das Antasten der Probe an die Spitze erzeugt den unten dargestellten Signalverlauf. Auf der y Achse ist die Kraft $f_z(z_c)$ in Newton [N] abgebildet und die x Achse beschreibt die Position der z Achse in Nanometer [nm]. Diese Funktion liefert nur Beobachtbare Werte in einem Interaktionsbereich der z Achse zwischen 0 und 1 Nanometer. Das ist der Bereich, in dem solche kleinen Kräfte messbar und spürbar sind. Das bedeutet, benutzt man eine Antastung in

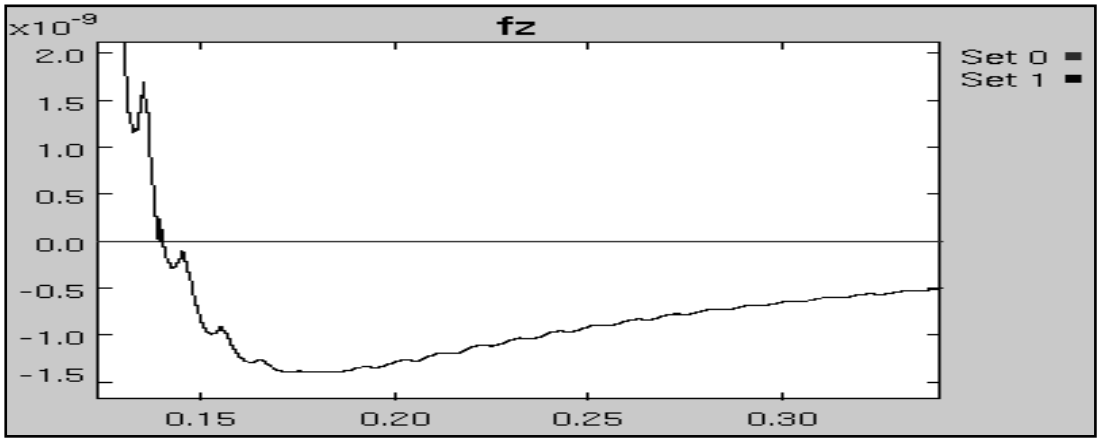


Abbildung 4-4: JKR Kraft

bspw. 5 nm Schritten für die z Achse so ist die Kraft $f_z(z_c)$ hinreichend klein. Deutlich sichtbar ist der Nullpunkt des Leonard –Jones Potential z_0, der durch ** beschrieben ist und Kräfte größer Null liefert. Im zweiten Teil des Signals dominieren „Van der Waals Kräfte“ und „Pauli Abstoßungskräfte“. Sie werden durch * berechnet. Für die Simulation stehen aus den Quellen [4][5] folgende Parameterwerte zur Verfügung:

achs_norm σ [m]	2.5e-10
Spitzenradius R [m]	40e-9
Nullpunkt Leonard Jones Potential z0 [m]	0.15e-9
Epsilon [J]	3.7e-22
p1	5e28
p2	5e28
Querkontraktionszahl d. Spitze v1	0.5
Querkontraktionszahl d. Probe v2	0.5
Elastizitätsmodul d. Spitze E1 [Nm^-2]	179e9
Elastizitätsmodul d. Probe E2 [Nm^-2]	179e9
Masse der Spitze m_c [kg]	6e-9

Aus den Einheiten der Werte wird ersichtlich, das die Berechnung mit dem Datentype double zwingend erforderlich ist.

4.5 Erweiterungsmöglichkeiten

Als Erweiterungen sind alle möglichen Arten von Verfeinerungen denkbar, die das Gesamtmodell das Atomkraftmikroskop noch genauer in seiner Funktionsweise beschreiben. Hierbei sollte jedoch eine Anpassung an die zur Verfügung stehende Rechenleistung gemacht werden.

Berechnung von N

N ist die Kraft, die durch eine nichtdimensionale Last W der Nadel erzeugt wird. D.h. die Bewegung der z Achse im Kontaktmodus wird in einem kleinen Bereich gehemmt. Diese Erweiterung wurde noch vorgenommen und wird durch folgende Formel beschrieben.

$N = m_c \ddot{z_c} + m_c \omega_c^2 z_c + W$; mit einem Parameterwert W=1.5e-12 , der sich aus Normierungsbedingungen ergeben hat.

Reibungskraft in x und y Richtung

Die Berechnung dieser Werte erfolgt in ähnlicher Weise wie die von fz. Sie ist jedoch wesentlich aufwendiger und trägt nur noch geringe Veränderungen des eigentlichen Verhaltens bei. In den Quellen zu diesem Kapitel wird tief greifender auf diesen Sachverhalt eingegangen.

Nichtlineares Verhalten der Stellachsen

Momentan ist das System so aufgebaut, das ein Input an Ux, Uy und Uz linear verarbeitet wird. In realen Systemen ist das nicht der Fall. D.h. bspw. eine Verdopplung der Eingangsspannung führt nicht zu einer Verdopplung der Stellachsenlänge. Auch das zeitliche Verhalten muss berücksichtigt werden. Man kann aber davon ausgehen das die Stellachsen wesentlich schneller sind als das zeitliche Verhaltend des Masse-Feder-Dämpfer Systems. Deswegen wurde die Anstiegszeit der diskreten Schritte (bspw. Spannungsschritte durch DA Wandler Auflösung festgelegt) auf Null gesetzt. Somit ist mit jedem Simulationsschritt eine Änderung der Eingänge möglich.

5 Die interferometrische Positionsmessung

Dieses Kapitel beschäftigt sich mit dem Messprinzip, welches die beschriebene Dynamik des Kapitel 4 detektieren soll. Um Belastungen der Positioniereinheit durch das Längenmesssystem zu minimieren, können keine Kontaktbasierenden Prinzipien eingesetzt werden. Auch die im Nanometerbereich liegende Auflösungsgenauigkeit stellt sehr hohe Anforderungen. Die Laserinterferometrie ist ein kontaktloses Verfahren zum Messen von Positionen und ihren relativen Bewegungsänderungen. Seit den Anfängen dieser Technologischen wurde dieses Prinzip sehr stark weiterentwickelt. So konnten Anfangsweise nur maximal halbe oder viertel Wellenlänge genaue Auflösungen erreicht werden. Bei einer Wellenlänge von 400 Nanometer sind das nur ganze 100 Nanometer genau. Wie das mit heutigen Mitteln noch erhöht werden kann folgt in einem späteren Kapitel.

5.1 Das Funktionsprinzip

Bei einem Laserinterferometer existiert kein fester Maßstab, wie man es von herkömmlichen Verfahren kennt. Vielmehr wird die Wellenlänge des Lasers als Skalierung genutzt. Im allgemeinem werden zwei Lichtwellen überlagert, die unterschiedliche Wegstrecken zurückgelegt haben. Mit Hilfe der Intensität der resultierenden Welle ist es möglich die Wegdifferenz zu bestimmen. In Abbildung 5-1 ist der schematische Aufbau eines

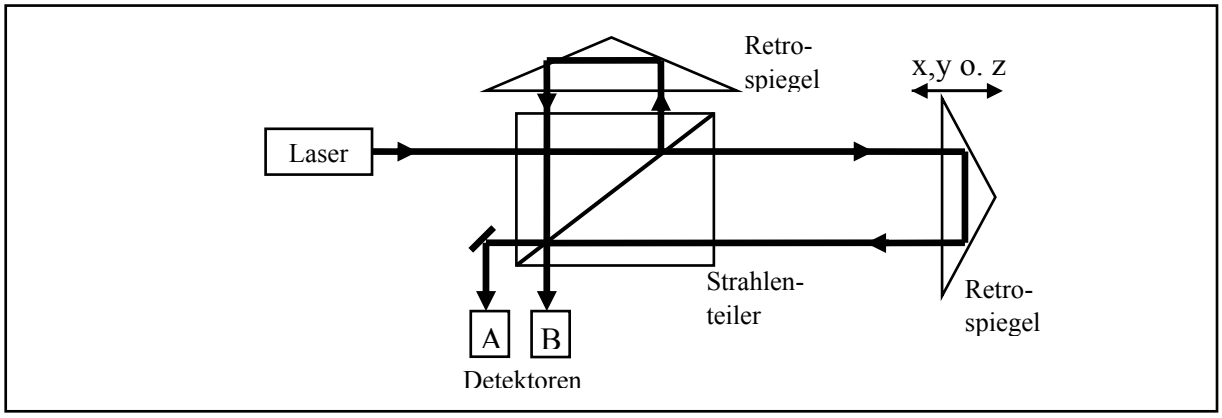


Abbildung 5-1: Schematischer Aufbau eines Polarisationsinterferometer

Polarisationsinterferometer dargestellt, mit welchem wir uns näher beschäftigen wollen. Ein Laser, meist He-Ne, erzeugt einen frequenzstabilen Lichtstrahl. Dabei wir ein kleiner ausgekoppelter Anteil des Lichtstrahles der Lasersteuerungselektronik rückwirkend zugeführt. Dadurch ist es möglich die Frequenz und somit auch die Wellenlänge sehr stabil zu halten. Der Strahlenteiler koppelt einen Mess- und Referenzstrahl gleicher Intensität aus. Der Referenzstrahl wird an einem Spiegel mit festem Abstand zum Strahlenteiler umgelenkt. Der Messstrahl wird durch einen am Messobjekt befindlichen Spiegel zurück auf den Strahlenteiler geführt. Auf den beiden Detektoren A und B werden die nochmals geteilten Lichtstrahlen zur Interferometrie gebracht. Auf die ausführliche Beschreibung der Phasenverschiebung und der Polarisation soll an dieser Stelle verzichtet werden.[7] Durch eine geeignete Elektronik an den Detektoren könne die Ströme in Spannungen umgewandelt werden. Diese kann dann für eine weitere Verarbeitung genutzt werden. Der Verlauf in Abhängigkeit der Verschiebung ist in Abbildung 5-2 im Bereich zwischen 0.0 und 0.2 der X-Koordinatenachse dargestellt. Die beiden periodischen Signale sind um 90°

phasenverschoben. Der Betrag der Verschiebung lässt sich durch einfaches abzählen der Minima und Maxima bestimmen. Bei einer Laserwellenlänge von 630 Nanometern ist durch die 3 Perioden eine Verschiebung von etwa 1900 Nanometern in eine Richtung bestimmbar. Die X-Koordinatenachse ist in Meter eingeteilt. Die Amplitude der beiden Signale ist maßgeblich durch die Detektorenelektronik vorgeschrieben und über Verstärkerschaltungen in bestimmten Bereichen frei einstellbar. Für die exakte Bestimmung sind zwei Signale zwingend nötig. Befindet sich bspw. die Achse an einer Position, an der der Wert eines Signals das Maximum (oder Minimum) ist. Ändert sich nun die Position in positive oder negative Richtung, so ist der Signalverlauf in beiden Fällen gleich. Nur durch zu Hilfenahme des zweiten Signalverlaufes ist die Richtung eindeutig bestimmbar. Das bedeutet, dass sich mit jedem Vielfachen der Wellenlänge der Werteverlauf der Signale A und B wiederholt. Um prinzipiell eine absolute Position zu bestimmen, muss in einer Initialiesierungsphase ein Reset der Zähleinrichtung durchgeführt. Diese Position wird somit zum Nullpunkt und von hieraus können die Anzahl der Perioden gezählt werden.

5.2 Das Störungsmodell

Bevor man diese Signale durch analog digital Umsetzer zur Weiterverarbeitung vorbereiten kann, muss man die möglichen Fehlerausprägungen untersuchen. Eine erste sehr wichtige Fehlerquelle ist die Änderung der Wellenlänge. Diese wurde im eigentlichen Modell nicht berücksichtigt, da sie von einer Reihe schwer modellierbaren Umweltfaktoren abhängig ist. Sie soll aber auf die elektronische Komplexität der Stabilisierung der Laserwellenlänge hinweisen.

$$\Delta\lambda/\lambda = 10^{-6} * (0.9 * \Delta t - 0.3 * \Delta p + 0.01 * \Delta H)$$

mit λ Wellenlänge, p Druck in hPa, t Temperatur 20° , H Luftfeuchtigkeit 50%

Wie zu sehen ist werden ein paar wichtige zu messender Umweltparameter benötigt. Viel wichtiger für die eigentliche Positionsbestimmung sind die folgenden auftretenden Fehler:

- Offsetfehler
- Gainfehler
- Phasenfehler
- Signalrauschen

Der Offsetfehler gibt an, wie stark die eigentlichen normierten Sinus / Cosinus Signale um den Nullpunkt verschoben sind. Der Gain- bzw. Verstärkungsfehler ist ein Maß für die Abweichung der maximalen Signalamplitude von der Norm. Die beiden Signale sind im idealen Fall 90° phasenverschoben. Auch hier können Fehler auftreten, die im Bogenmaß angegeben werden können. Im weiteren werden alle Angaben in Bezug auf die eigentlich Amplitude (einstellbarer Parameter) gemacht. Bspw. hat ein Gainfehler von 0,3 bei einer Amplitude von 1,0 die Auswirkung, dass das Maximum des periodischen Signals zwischen 0,7 und 1,3 liegen kann. Derselbe Gainfehler führt bei einer Amplitude von 2,0 zu einem Ausdehnungsbereich von 1,4 bis 2,6.

Speziell bei den ersten beiden Fehlerquellen gibt es jeweils eine dynamische und statische Ausprägung. Der dynamische Fall tritt dann ein, wenn sich die Achse, von der die Position bestimmt werden soll, kontinuierlich bewegt. Der Fehler soll sich in diesem Fall, nach einer bestimmten Anzahl von Periodendauern, einmal kontinuierlich in seiner positiven und negativen Signalbeeinflussung ausgeprägt haben. Die Anzahl der benötigten Perioden wird durch die Modulationslänge beschrieben. Diese Betrachtung ist somit von der Position abhängig, aber keinesfalls Zeitabhängig. Für die zeitabhängige Beeinflussung gibt es noch den statischen Fall. Die Achse steht ein einer bestimmten Position und erzeugt somit zwei

Werte. In diesem Fall sollen sich die beiden Werte nicht über den gesamten Fehlerbereich ändern, sondern nur nahe den eigentlich erzeugten Messwerten. Für den dynamischen Fall soll die Abbildung 5-2 einen visuellen Eindruck bieten.

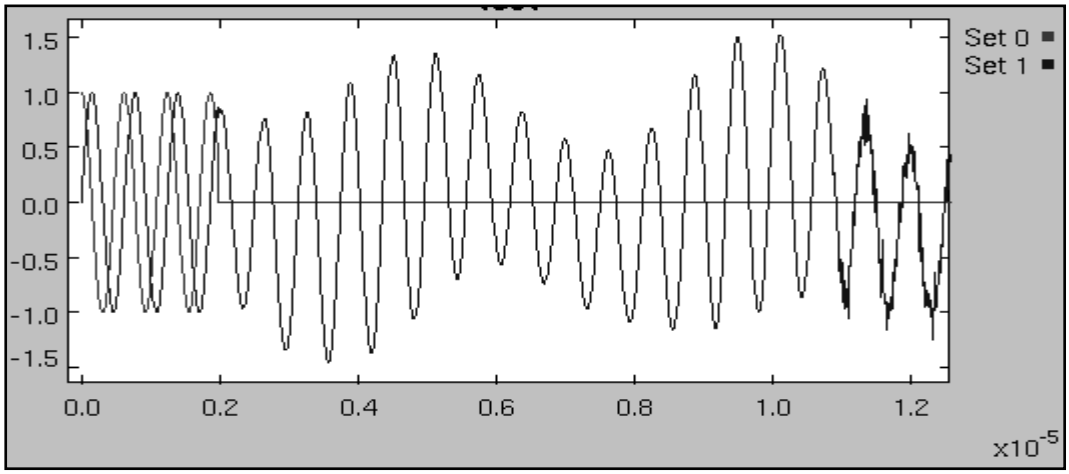


Abbildung 5-2: Detektorsignale und Signalfehler

Ab etwa 0,2 wurde ein Offset- und Gainfehler von 0,3 mit einer Modellierungslänge von etwa 3,7 erzeugt. Dabei wurden für beide Fehler unterschiedliche Werte verwendet. Es ist sehr deutlich zu sehen, dass der Verstärkungsfehler nach etwa 4 Wellenlängen einen Bereich von 0,7 bis 1,3 abgedeckt hat. Der Offsetfehler verhält sich in ähnlicher Art und Weise. Ab etwa 1,1 wurde noch ein Signalrauschen erzeugt. Der Übersichtlichkeit halber wurde auf die Darstellung des zweiten Signals und des Phasenfehlers verzichtet. Deutlich ist auch eine sehr schlechte Konstellation zu sehen. Bei etwa 0,7 wirken Offset und Gain in dieselbe Richtung und bewirken somit eine extreme Verzerrung.

5.3 Die Implementierung

Die beschriebene Funktion und Wirkungsweise wird durch das Modul laser_noise realisiert. Es enthält 4 relativ gleiche parallel arbeitende Verarbeitungsströme.

Parameter

Folgende Parameter werden für die richtige Funktionsweise und Darstellung benötigt:

Wellenlänge [nm]	Wellenlänge des Lasers
Amplitude	Spezifiziert die Amplitude der periodischen Signale. Kann eine durch Hardware vorgegeben Größe sein. (bspw. als Spannung in [V] angegeben)
Norm_update	Dient der zeitlichen Steuerung eines kleinen Fensters, in dem das Signal-Rausch Verhältnis angezeigt wird. Nur zur Visualisierung.
Zc_faktor	Faktor zur speziellen Behandlung des Signal Rauschens auf dem Intensitätswert des Spitzensignals.
Zc_nullpegel	Repräsentiert den Intensitätswert der Photodiode der Nadel im Nichtkontakt und Stillstand. Kann eine Spannungswertvorgabe sein.
Schrittweite	Wird benötigt um maximales Signal Rauschen anzupassen und für das Anzeigefenster.
Modulations_länge	Siehe. 5.2
Noise_power	Dient für Steuerung des statischen Falles, siehe. 5.2.

Alle Fehler können mit Hilfe von Tck Schieberegler zur eigentlichen Simulationslaufzeit dynamisch eingestellt und verändert werden.

Input / Output

Dieses Modul hat als Eingang die 4 absoluten Positionsausgänge des Atomkraftmikroskop Modells. Es wandelt diese absoluten Positionsangaben in relative um und erzeugt pro Achse ein Paar Sinus / Cosinus Signale. Somit erhält man für die drei Achsen 6 Ausgangssignale, wobei diese immer Paarweise korrespondieren. Für die Nadelposition wird nur ein Ausgang erzeugt, da hier nur eine einfache Intensitätsumsetzung erfolgt.

Internes Verhalten

Es folgt eine kurze Skizze der Vorgehensweise.

1. Erzeugung des Sinus und Cosinus Signals. Dazu wird der Eingang mit Hilfe der Wellenlänge auf einen Bereich 0..2PI skaliert. Durch Anwendung der zugehörigen trigonometrischen Funktion werden zwei Signale erzeugt, die noch um den Wertebereich der Amplitude erweitert werden. Da sich gezeigt hat, dass eine spätere Phasenverschiebung schlecht modelliert werden kann wird der Phasenfehler schon an dieser Stelle erzeugt.
2. Hinzufügen des Gainfehlers, der durch Multiplikative Verknüpfung gekennzeichnet ist.
3. Der Offsetfehler wird durch Addition zum eigentlichen Signal hinzugeführt.
4. Erzeugen des Signalrauschens. Diese wird durch Weißes Rauschen dargestellt. Das Rauschen muss jeder Zeit kleiner, als die minimale Information, sein. Deswegen wurde die Intensität der Rauschquelle auf $\frac{\text{Amplitude} * \text{Schrittweite} * 4}{\text{Wellenlänge}} * 0.1$ begrenzt.

   Dieser Sachverhalt beinhaltet die Anzahl der erkennbaren diskreten Schritte einer bestimmten Größe pro Wellenlänge.

Die Abbildung 5-3 soll den Unterschied zwischen dem statischem und dynamischen Verhalten noch einmal näher erläutern. Als Eingang ist der Port weg, welcher die Achsposition zur Verfügung stellt, und der durch den Schieberegler einstellbare Gainfehler. Ohne Betrachtung des Integrators wird ein periodisches Signal erzeugt, welches eine

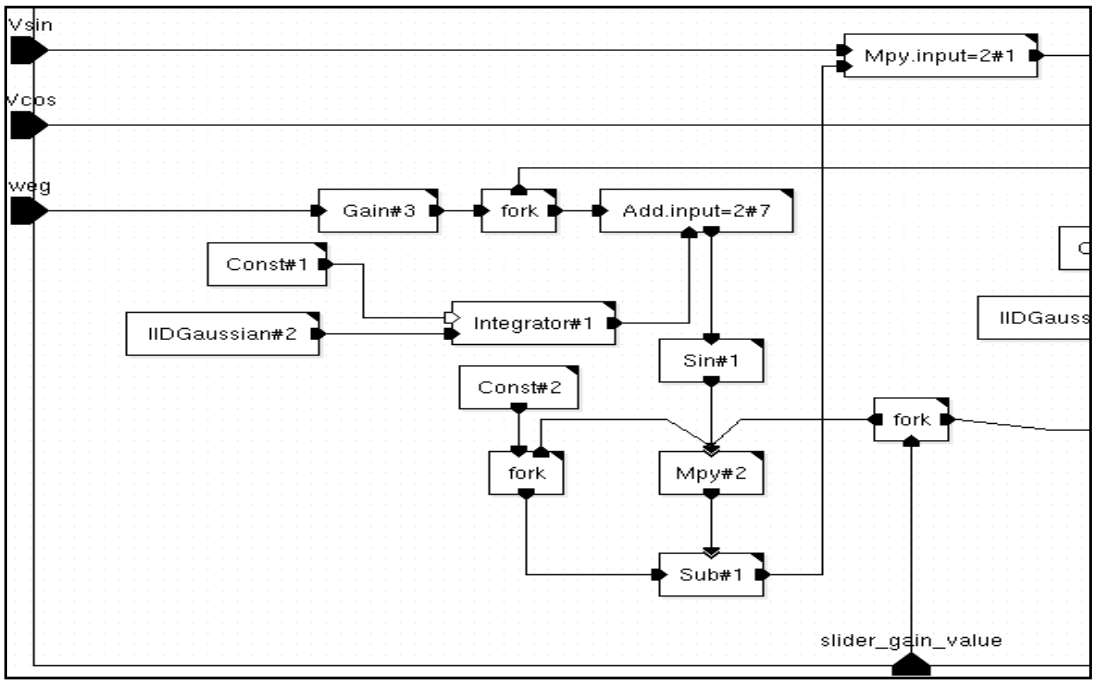


Abbildung 5-3: dyn. und stat. Gainfehler Prinzip

Amplitude von slider_gain_value und eine Periodendauer von etwa Modulationslänge (Faktor Gain#3) hat. Soll ein Gainfehler von 0,3 erzeugt werden, so entsteht nach der Multiplikation des Sinus mit 0,3 das entsprechend periodische Signal. Damit man die Multiplikative Verknüpfung mit dem eigentlichem Signal (V_sin) durchführen kann muss der Nullpegel künstlich (Const#2) auf 1.0 verschoben werden. Nun wird fortlaufend, solang sich der weg kontinuierlich ändert, V_sin mit Werten (der Modellierungswelle) aus dem Bereich (1–Gainfehler) bis (1+Gainfehler) multipliziert.
Da sich mit diesem Sachverhalt alle 6 Signale gleichermaßen ändern würden, wurde ein Zufallsprozess integriert. Ein Weißes Rauschsignal (Intensität über Parameter noise_power einstellbar) wird kontinuierlich aufsummiert. Dieser Anteil wird vor der Sinusbildung zu dem eigentlichem Weg addiert. Im mittel ist die Summe des Rauschens Null, aber trotzdem liefert der Integratorausgang kleine positive und negative Werte. Damit wird gewährleistet, dass die Anstiege aller 6 Fehlermodellierungsignale unterschiedlich sind. Je größer die Intensität des Rauschens gewählt wird, desto ungleichmäßiger ist die Modellierungswelle.
Ist der weg statisch, so ändert sich die Modellierungswelle nur noch in dem durch den Integrator vorgegebenem Bereich.

6 Der Rechnerteil

Durch die Kombination der Informationen aus dem Kapitel 4 und 5 sollte eigentlich sehr deutlich hervorgehen, welche Signalverläufe der eigentliche Rechnerteil verarbeiten muss. Trotzdem soll die Abbildung 6-1 den Sachverhalt noch einmal näher beschreiben.

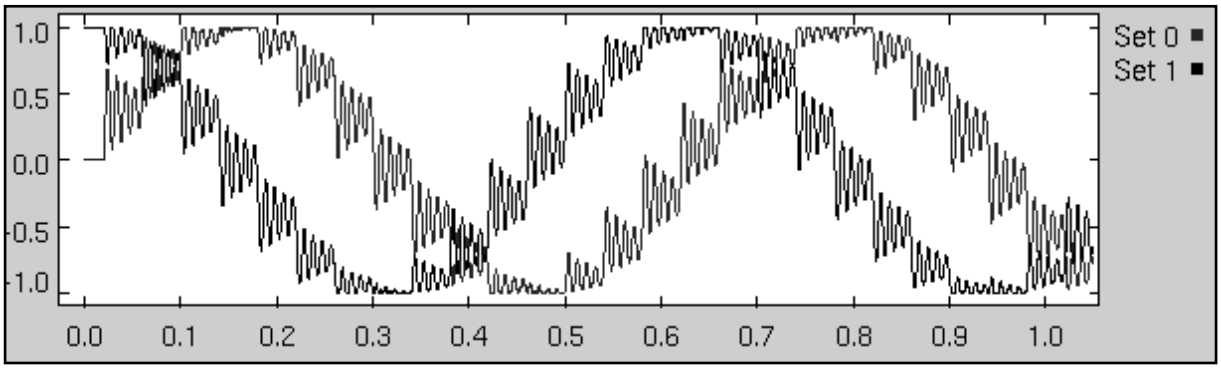


Abbildung 6-1: Signalinput des Rechnerteils für eine Achse

Es ist der störfreie Ausgang des Laserinterferometers zu sehen, der durch eine Treppenfunktion mit der Schritthöhe von 40 Nanometer am Eingang des AFM erzeugt wird. Die x Achse ist in Sekunden angegeben. Auf jeder Stufe der Treppe wird 200/5000 Sekunden lang gewartet. Im Amplitudensignal ist eindeutig die Dynamische Antwort der Stellachse zu sehen und die Höhe der Treppe mit vorschreitender Zeit. Nach etwa 16 Schritten in dieselbe Richtung erfolgt die Wiederholung des Signalverlaufes. In der Charakteristik der Periodischen Signale ist begründet, dass ein Schritt bei Extrema der Funktion wesentlich weniger Signalinformationsänderung liefert als andere. Der zugehörige Positionsverlauf der Achse ist im Anhang in Abbildung 13-1 dargestellt.
Im folgenden Kapitel werden die Mechanismen beschrieben und erläutert, wie man aus den zur Verfügung stehenden relativen Signalen mit Hilfe von DSP Verfahren absolute Positionswerte erzeugen kann. Doch zuvor müssen alle auftretenden Signalfehler, deren Ausprägung für den Rechnerteil unbekannt ist, beseitigt werden. Das bedeutet bis auf die Wellenlänge des Lasers ist in diesem Modul kein weiterer Parameter (speziell Fehlerwerte) bekannt. Es kann davon ausgegangen werden, dass die ADU's des Rechners den kompletten Signalbereich abtasten können. D.h. selbst maximale Konstellationen auftretender Fehler werden exakt abgetastet und somit erkannt.

6.1 Die Offset- und Verstärkungsfehlerkorrektur

Das hier vorgestellte Verfahren findet seine eigentliche Anwendung in digitalen Hochinterpolatoren von Linearantrieben.[10] Es berechnet den Offset- und Verstärkungsfehler und korrigiert daraufhin die Eingangssignale. Dabei soll es überwiegend unabhängig arbeiten und somit keine eventuell Fehlerbehafteten Informationen aus späteren Berechnungen (digitale Perioden counts) benutzen. Die Korrektur beruht auf dem erkennen der Extrema und somit kann sie effektiv nur nach jeweils einer Periode gemacht werden. Es wird immer eine Reihe der letzten Minima und Maxima gespeichert um eine Mittelwertbildung über diese durchzuführen. Somit kompensiert das Verfahren in seiner eigentlichen Form speziell langfristige Offset- und Temperaturdrifts von Bauteilen.

Parameter

Der wichtigste Parameter ist die Anzahl der gespeicherten letzten Maxima und Minima Werte. Für Motoren ist ein typischer Wert 8. Hierdurch werden langfristig kleine Fehler und kurzzeitig starke Fehlerschwankungen ausgeglichen. Das würde mit einem Modulationsindex ab 40 des Laserinterferometers korrespondieren. Auf schnelle Fehlerschwankungen kann nur mit Pufferlänge (Anzahl der Minima und Maxima) von 2 oder 3 reagiert werden. Extreme Ausreißer können dadurch schlecht gemittelt werden und sind somit nicht so stark geglättet. Mit einem Modulationsindex von etwa 8 kann das nachgebildet werden.
Der zweite Parameter wird für die Erkennung der Extrema benötigt. Mit ihm wird die minimale Signalamplitude der Eingänge festgelegt. Es darf zu keiner Zeit ein Extrema unter diesem Level liegen.

Implementierung

Die Vorgehensweise für die Korrektur wird jeweils für beide Signale (Sinus und Cosinus) je Stellachse getrennt auf die gleiche Art und Weise vorgenommen. Die Abbildung 6-2 stellt die Vorgehensweise graphisch dar.

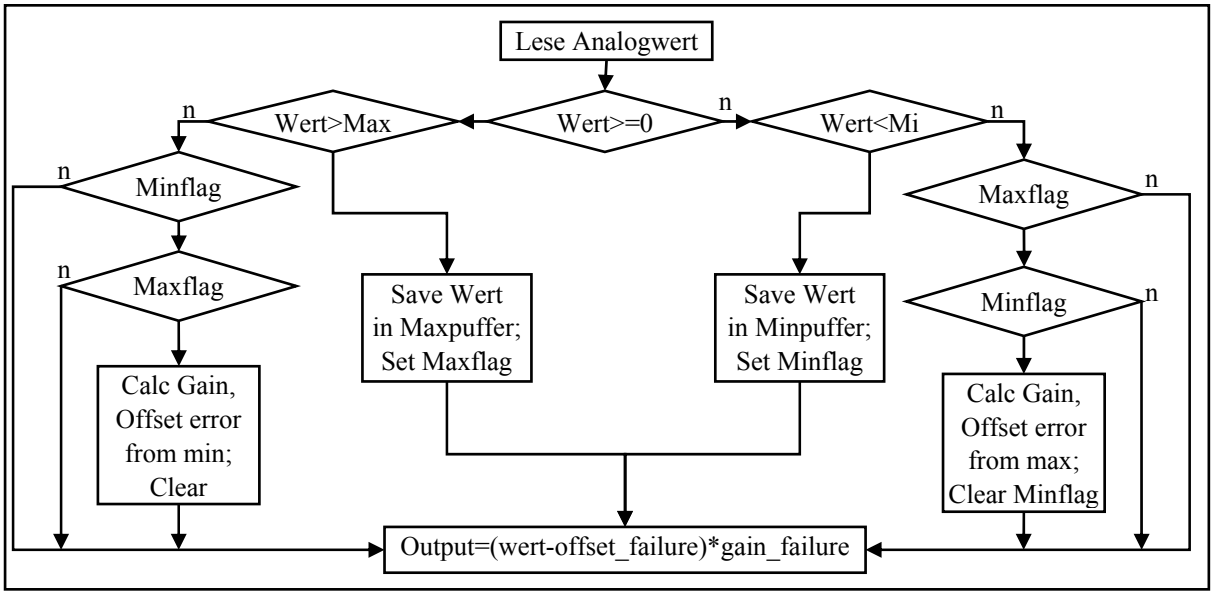


Abbildung 6-2: Flussdiagramm des Algorithmus

Zur Initilaisierung wird der ringförmige Maximumpuffer und Minimumpuffer mit den zugehörigen maximalem und minimalem Signallevel initialisiert. Die Pufferlänge kann durch einen Parameter angepasst werden. Es wird jeweils ein Pointer auf den Beginn des Puffers gesetzt. Dadurch entsteht ein Hysteresebereich, der es erlaubt eine falsche Korrektur bei Stillstandsschwankungen in dem Bereich zu unterdrücken. Alle Flags werden auf FALSE gesetzt. Es ist ganz wichtig, das eine Update der Korrekturwerte (offste_failure und gain_failure) mit Hilfe der Maximumwerte erst nach der Erkennung eines Minimums erfolgt. Gleiches gilt auch in die andere Richtung. Würde eine Neuberechnung gleich nach dem Erkennen eines Extrema erfolgen, so ist nicht sichergestellt ob es wirklich eines ist. D.h. erst nach dem detektieren eines Minimas wird das letzte gespeichert Maximum benutzt. Addiere als erstes alle Werte des Puffers zusammen und teile sie durch die Länge des Puffers. Dieser Wert wird in der Liste gespeichert. Addiere die beiden so gemittelten Werte der beiden Listen zusammen und teile sie durch zwei (!!! In der anderen Liste steht der letzte gemittelte Wert an Stelle Pointer-1 und beachte die Vorzeichen!!!). Dieses Ergebnis ist der Offsetfehler. Den

Verstärkungsfehler erhält man, wenn man das doppelte Produkt des Reziproken aus der Subtraktion der beiden Werte errechnet. Erhöhe den zugehörige Pointer um eins und rage den Initialwert ein. Das Verhalten ist in einem kleinen System Namens gain_offset_error_test zu sehen. Es existiert noch ein weiteres Modul namens gain_offset_correct_mini, das für Experimente mit schnellen Korrekturen genutzt wurde. Leider liefert keines der beiden Module wirklich gute und überzeugende Ergebnisse.

6.2 Der adaptive Filter

Für die Beseitigung des dritten Fehlers, dem Signalrauschen, werden nun DSP Filter benutzt. Ein erster Ansatz auf dem Gebiet ist immer ein so genannter FIR Filter. Es wird davon ausgegangen, dass die Funktionsweise eines solchen Signal Filters bekannt ist. Mit ihnen ist es möglich stochastische und deterministisch periodische Verfälschungen von Signalen auf digitalem Weg zu beheben. Für die Benutzung von FIR Filtern müssen Filterkoeffizienten berechnet werden. In die Berechnung fließen meistens sehr viele Informationen aus den statischen Parametern des Systems mit ein. Damit man bei der Benutzung und Anpassung des Modellsystems relativ viele Freiheiten hat, ohne jedes mal die Koeffizienten neu berechnen zu müssen, wurde die Nutzung adaptiver Filter untersucht. [11] Für den Aufbau und die Funktionsweise stehen sehr viele Demos in Mathlab und ML-Desiner zur Verfügung. Diese sind aber mit größter Vorsicht zu betrachten. Das verwendete Prinzip ist ein LMS Filter und der Aufbau ist in Abbildung 6-3 schematisch dargestellt.

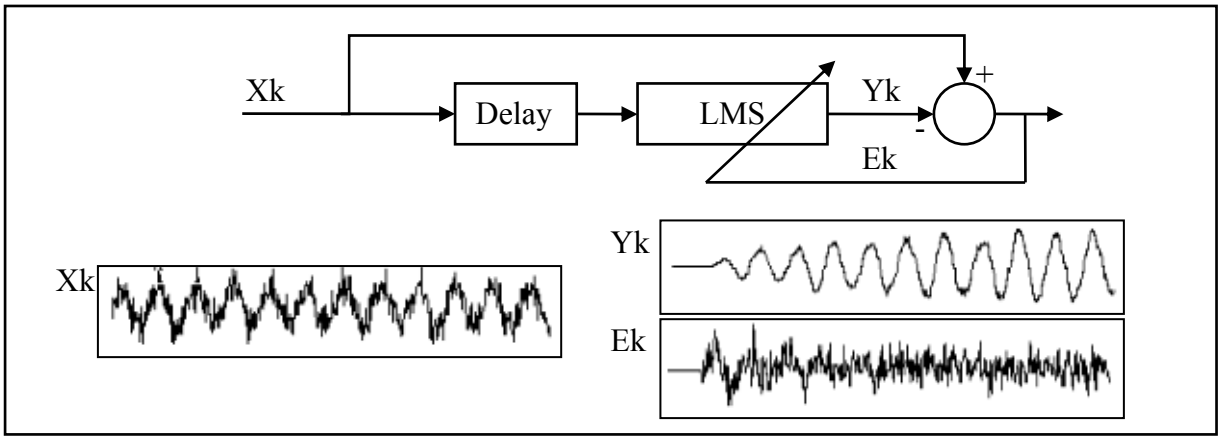


Abbildung 6-3: adaptiver Filter

Die Anpassung der Filterkoeffizienten (taps) kann nur mit Hilfe eines Referenzsignals durchgeführt werden. Dazu muss kontinuierlich der Fehler Ek zwischen Referenzsignal Xk und Filterausgang Yk berechnet werden. Es gibt drei gebräuchliche Möglichkeiten eine Referenz zu dem eigentlichen Signal Xk zu erhalten. Das wäre das Signal selber, der Rauschanteil oder ein verzögertes Xk. Die ersten beiden Referenzsignale fallen weg, da sie unbekannte Größen sind. In der digitalen Signalverarbeitung ist es sehr leicht durch Zwischenspeicherung Signale kurzzeitig zu verzögern. Dabei ist die Anzahl der Verzögerung genau so groß, wie die Anzahl der Koeffizienten des Filters. Die Adaption der Taps (bi) erfolgt nach folgender Berechnung:

bi_new = bi_old + cstep*Ek*X(k-d)-i (bi = 0 Anfangsbedingung; i = Anzahl der Taps = Delay)

Sehr gute Ergebnisse wurden mit den Parametern i=32 und cstep=1/(10*i*signal_amplitude) erzielt. Dabei ist cstep die Schrittgröße mit der die Anpassung der Koeffizienten erfolgt. Sollte diese Zahl zu groß gewählt werden wird der Filter instabil und der Fehler wird wider

größer. Ist die Zahl zu klein kann es passieren, das wichtige Signalinformation weggefiltert werden. In unserem Fall würde es ein abflachen des dynamischen Anstiegsverhalten bedeuten und das fast 90%ige Überschwingen geht teilweise verloren. Der LMS (least mean square) Algorithmus minimiert den Quadratischen Fehler Ek.

6.3 Die digitale Vierfachinterpolation

Die einfachste Art und Weise aus dem analogen Signalverlauf eine Positionsaussage zu treffen besteht in der digitalen Interpolation.[8] Hierbei werden über einfachste Komperatorschaltung jeweils Sinus und Cosinus mit einem Referenzpegel verglichen und es entstehen somit zwei 1 Bit Aussagen. Diese beiden digitalen Signale können mit Hilfe von Zählstufen weiter zu Positionsangaben Aufsummiert werden. Diese Aufgaben werden überwiegend durch Hochintegrierte externe Hardware übernommen. Die eigentliche Benutzung erfolgt dann meist durch das Lesen der zugehörigen Speicheradressen dieser. Da dieser Informationsbestandteil eine wichtige Größe des Modells ist, musste das Interpolationsverfahren in einfachster Funktionsweise nachgebildet werden. Um die Arbeitsweise deutlich zu machen ist die einfachste Interpolation in Abbildung 6-4 kurz verdeutlicht.

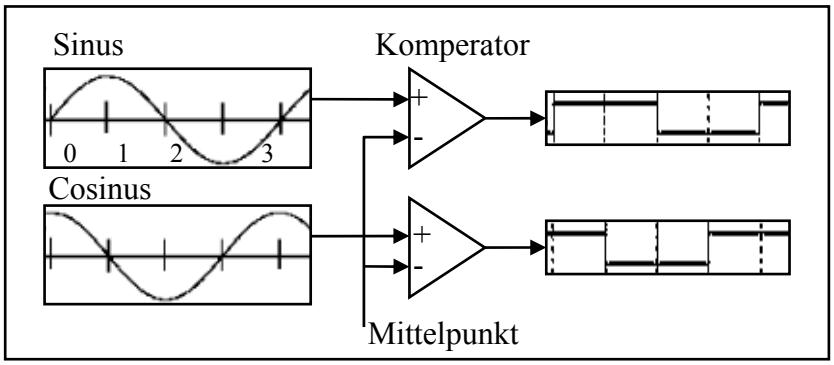


Abbildung 6-4: zweifache digitale Interpolation

Als Ausgang entstehen zwei um 90° verschobene digitale Signale. Diese werden in der Fachliteratur auch als Incremente (bspw. bei Incremetal Drehgeber) bezeichnet. Was passiert bei diesem Verfahren, wenn ein Eingangssignal um den Nullpunkt oszilliert? Es würde zu einem ständigen Bitwechsel eines der beiden Binärsignale führen. Dadurch entsteht ein sehr großer Fehler in der weiteren Auswertung. Um das in dem Modell zu vermeiden werden beide Signale durch FIR Filter bearbeitet. Dieser Filter entfernt alle Frequenzen die im Bereich der Dynamik der Stellachsen liegen. Somit entsteht aus dem Signal (siehe Abbildung 6-1) eine geglättet monotone Funktion. Dabei ist zu beachten, dass bei höheren Schrittweiten die Funktion entsprechend eckiger wird. Es wird ein FIR Filter mit 64 Koeffizienten verwendet. Diese Filtertaps wurden mit Hilfe von Matlab Funktionen berechnet. Der Aufruf in Matlab war "%fir1(64,0.001)", es gibt aber noch eine Reihe weiterer Funktionen die benutzt werden können.

Bei der Vierfachinterpolation werden, wie der Name es sagt, vier Komparatoren benötigt. Mit Hilfe der beiden Eingangssignale werden 4 Hilfssignale erzeugt, die immer 45° phasenverschoben zueinander sind. Diese Hilfssignale werden den vier Komparatoren zugeführt. Dabei benutzen die Komparatoren eine 1-dimensionale einstellbare Hysterese, um Restschwingungen (die der FIR nicht beseitigen konnte) zu kompensieren. Das führt zu einer kleinen zeitliche Verzögerung, die später behoben werden muss. Die Ausgangssignale haben somit die doppelte Frequenz der Eingänge (ist dadurch eigentlich nur eine zweifach

Interpolation). Damit können genau vier Quadranten einer Wellenlänge exakt gemessen werden. Die maximal zu ereichende Positionsgenauigkeit ist Wellenlänge/n, wobei n=Interpolationsfaktor ist. Die vier Hilfssignale haben folgende Form:

$$S0 = a * \sin(0°) + b * \cos(0°) = b$$

$$S1 = a * \sin(45°) + b * \cos(45°)$$

$$S2 = a * \sin(90°) + b * \cos(90°) = a$$

$$S3 = a * -\sin(135°) + b * -\cos(135°)$$

mit a und b die beiden Eingangssignale

S0 bis S3 werden den Komparatoren, die auf Null vergleichen, zugeführt. Die jeweils 90° phasenverschobenen Komparatorenausgänge werden an ein XOR angeschlossen. Es bilden sich die Paare (S0,S2) und (S1,S3). Diese doch recht simple Vorgehensweise kann sehr leicht an höhere Genauigkeiten angepasst werden. Dabei ist durch die Frequenz der digitalen Signale eine Obergrenze gesetzt. Zur weitern Verarbeitung müssen die Anzahl der Signalwechsel gezählt werden und das kann nur mit einer bestimmten maximalen Frequenz durchgeführt werden. Somit ist die maximale Genauigkeit dieses Verfahrens abhängig von der maximalen Geschwindigkeit der Stellachsenbewegung und der zur Verfügung stehenden Rechenleistung des Rechners.

6.4 Das Zählen der Quadranten

Mit der Problematik, wie man diese Impulse zählen kann wird jeder Informatiker während seines Studiums konfrontiert. Die Lösung lautet in dem Fall Zustandsautomaten. Die Stellachse kann sich in zwei Richtungen bewegen und liefert dadurch zwei unterschiedliche Binärsignalfolgen. Es existiert eine Vorwärts- und Rückwärtsfolge. Der Zustandsautomat hat die Aufgabe das zu erkennen und bei jedem Quadrantenwechsel (es können 4 Quadranten pro Wellenlänge unterschieden werden) einen entsprechenden Impulse (vor oder zurück) zu erzeugen. Diese Impulse können durch einen Binärzähler verarbeitet werden und dieser erzeugt eine Zahl, die der Anzahl der zurückgelegten Quadranten (1/4 Wellenlängen) entspricht. Auf die graphische Darstellung des Zustandsautomattes sei an dieser Stelle verzichtet.[12]

Implementation

Zur Zeit der Entstehung des Gesamtsystems funktioniert die FSM Domain (finit state machine) in dem Entwicklungswerkzeug ML-Designer gar nicht. Um trotzdem einmal die Kombination unterschiedlicher Domains zu benutzen wurde dieses Modul in der DE Domaine realisiert. Die Aussage der MLD Dokumentation, das bei Interdomainkommunikation Anpassungen automatisch vorgenommen werden kann nicht bestätigt werden. Beim Einbinden von DE Modulen in SDF Systeme wird bei jedem Simulationsschritt der SDF Domain genau ein Event an den Eingangsports des DE Moduls erzeugt und genau ein Event am Ausgangsport gefordert. Wird am Ausgang kein Event erzeugt, dann schlägt die Simulation mit der Fehlermeldung „inconsistent sampling rate“ fehl. Der DE UpDownCounter muss somit immer ein Event (Zählerstand) in jedem Simulationsschritt erzeugen. Auch wenn sich der Zählerstand nicht geändert hat. Deshalb wurde der „demand port“ des erweiterten UpDownCounters (der jetzt nur Eins Events zählt) direkt an einen Eingangsport angeschlossen. Dasselbe Verhalten kann man auch mit dem DE Primitiv „sampler“ herstellen. Dieses wiederholt zur Sampleratenanpassung das letzte Event. Um den

eigentlichen Vorteil der DE Domain zu nutzen werden über Schwellwerttest nur dann Events erzeugt, wenn sich die Binärsignale ändern.

Einschränkungen

Das Gesamtverhalten des AFM und der Interferomter ist so aufgebaut, das zum Simulationsbeginn sich alle Stellachsen in einer Art Endlage (bspw. mechanisch) befinden. Weiterhin liefern die Interferometer Wertepaare, die im Bereich 0..PI/4 oder 7PI/4..2PI liegen. Damit ist gewährleistet, das die beiden digitalen Signale 0 sind und der erste mögliche Signalwechsel Reihenfolgemäßig ein incB=1 und danach incA=1 ist. Dadurch sind alle Zählerstände größer gleich Null. Durch ein Reset port besteht die Möglichkeit den Zähler zurückzusetzen, um diesen Initialzustand herzustellen.

6.5 Die resultierende Positionsbeschreibung

Da sich diese Arbeit mit hochgenauer Positionierung von Stellachsen eines AFM beschäftigt muss die vorliegende Position noch genauer bestimmt werden. Die durch das Verfahren in 6.4 beschrieben Position wird im Modell mit dig_position bezeichnet. Sie enthält eine grobe Abschätzung der bisher zurückgelegten Quadranten. Um pro Wellenlänge eine höhere Auflösung zu erzeugen wird eine trigonometrische Berechnung durchgeführt. Man bildet den inversen Tangens aus dem Quotient des Sinus und Cosinus Abtastwertes.[10]

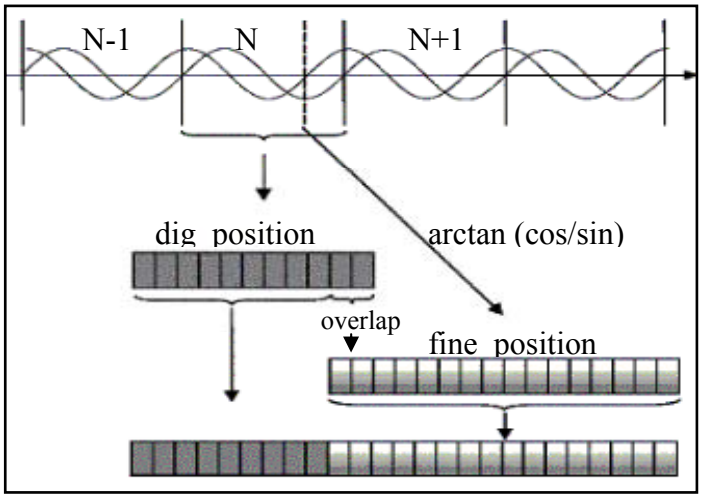


Abbildung 6-5: Darstellung der Position

Durch eine geeignete Transformation (mathematisch, logisch) kann man den so erhaltenen Wertebereich kontinuierlich, bspw. zwischen 0 und 1, skalieren. In Abbildung 6-5 ist der Zusammenhang zwischen den beiden Werten graphisch dargestellt. Die fine_position kann man prinzipiell als Nachkommaanteil der dig_position ansehen. Durch geeignete Transformation kann man eine Festkommazahl erzeugen, die die Position mit bspw. 16 Bit genauen Nachkommaanteil beschreiben kann. Durch die Hystereseanteile und dem FIR Filter kann es bei der dig_position zu verspäteter Quadrantenerkennung kommen. Dieses kann sehr einfach durch die Überlappung der unteren 2 Bit der dig_position mit den oberen 2 Bit der fine_position korrigiert werden. Ist der Nachkommaanteil schon im Quadrant 0 und der Ganzzahlenanteil noch in Quadrant 3, so muss eine Anpassung auf volle Wellenlänge gemacht werden. Gleiches gilt für eine Korrektur eines Q3 zu Q0 Überganges. Dazu muss man nur +4 oder -4 auf dig_position addieren. Multipliziert man die so erhaltenen Werte mit der Wellenlänge erhält man die Position der jeweiligen Achse.

7 Der Regler

Prinzipiell ist der Regler ein Teil des Rechners. Er schließt den eigentlichen Kreis des Signalflusses und soll aufgrund seiner doch recht hohen Bedeutung ein eigenes Kapitel bekommen. Im System befinden sich 3 Reglermodule. Zwei gleiche, die x und y Achse regeln. Der andere arbeitet auf einem etwas anderen Prinzip und ist für die Regelung der Position der z Achse zuständig. Die Funktion eines PID Reglers sollte bekannt sein. Er hat zwei Eingänge, jeweils ein Soll- und Istwert. Über die Fehlerabweichung versucht er auf gewünschte Art und Weise einen Output zu erzeugen, mit dem Ziel nach endlicher Zeit Gleichheit der beiden Eingänge zu erreichen. Dabei wird der Sollwert von einem Nutzerterminal vorgegeben. Das kann beispielsweise ein Bahnsteueralgorithmus sein, der für die x und y Achse ein 2-dimensionales Raster erzeugt. Als Quelle für den Aufbau der Regler diente der Quellcode [13] der „Nanomeßmaschine“ der Fakultät für Maschinenbau.

7.1 Die X,Y Regler

Die Istwertvorgabe wird durch den Ausgang des Rechnerteils erzeugt. Für jede Achse stehen zwei Werte im Festkommazahlenformat zur Verfügung. Die Anzahl der Quadranten in den 16.0 Formaten und die Verfeinerte Position je Wellenlänge in den 3.16 Formaten. Dabei sind die 2 LSB des fine_position Vorkomaanteils gleich der Quadrantennummer. Das MSB sollte immer Null sein, weil keine negativen Zahlen zugelassen sind. Im Regler werden die beiden Festkommazahl zur Überlagerung gebracht. Eine wiederholte Überprüfung der Korrektheit wird nicht vorgenommen (das sollte durch die Quadrantenkorrektur des Rechnerteils sichergestellt werden). Nach dem Umwandeln in eine Gleitkommazahl und der Multiplikation mit der Wellenlänge des Lasers kann ein direkter Soll- Istwertvergleich vorgenommen werden. Der Sollwert wird ebenfalls als Gleitkommazahl in der Einheit Meter [m] durch das Module Ablaufsteuerung vorgegeben. Die Parametervorgaben aus den Quellen konnten nicht direkt übernommen werden. Das Simulationsmodell unterscheidet sich relativ stark von dem wirklichen Aufbau. Da die Berechnungsgrundlage für die Bestimmung der Parameter fehlte wurden diese experimentell ermittelt. Hierzu wurden die adaptiven Filter der digitalen Signalverarbeitung überbrückt. Der Grund wird in dem Unterkapitel 7.2 näher geschildert. Folgende Reglerparameter haben sich als brauchbar erwiesen.

AdaptionGain	1e9
AdaptionLowerLimit	0.01
AdaptionUpperLimit	100
IntegratorLowerLimit	-1
IntegratorUpperLimit	1
GainKn	3000
GainKv	5
GainKp	47e4
Wellenlaenge	632e-9
Cycle_time [s]	0.0002
Steuer_const	450000

Es wurden die Schrittweiten von 10, 50 und 120 Nanometer getestet. Dabei wurden wider 200 Simulationsschritte je Stufe zugrunde gelegt. Wie in den frühen Kapiteln zu sehen ist erreicht das System nach dieser Zeit keinen akzeptablen Endwert. Mit dem Regler ist es jedoch möglich nach einer Zeit von 0.04 Sekunden (200/5000) den gewünschten Istwert zu erreichen. Dabei werden natürlich die kleinen Schritthöhen wesentlich schneller erzeugt als

bei den hohen Größen. Die cycle_time ergibt sich aus dem Frequenzskallierungsfaktor (ohmega0) des AFM Modelle. Der Regler besitzt einen Resetport. Solang dieser den Wert 1 trägt erzeugt die Ablaufsteuerung eine Initialisierungssequenz. Dabei wird der Sollwert mit der Steuer_const multipliziert und am Ausgang angelegt. Der Parameter steuer_const ist eine Linearitätskonstante, mit der es möglich ist auch ohne Regler ein paar Schritte durchzuführen. Während dieser Zeit initialisieren sich die Integratoren. Das kann als eine Vorsteuerung gesehen werden, da es unter Umständen zu einem falschen Schrittbeginn kommen kann. Dieser wird durch die Quadrantenkorrektur verschleiert und führt erst beim nächsten Quadrantenübergang zu einem Aufschwingen des Reglers.

7.2 Das Problem

Die Anpassung der Reglerparameter mit den adaptiven Filtern hat sich als äußerst schwierig erwiesen. Um den vorliegenden Sachverhalt sichtbar zu machen wurde ein extra Simulationssystem angelegt. Der Filter kann die erste Amplitudenschwingung der Sprungantwort des Systems nicht genau interpretieren. Für ihn ist der schnelle Anstieg ein Hochfrequenzanteil. Die eigentliche Aufgabe des Filters ist es hochfrequentes Signalrauschen zu filtern. Demzufolge macht er es an dieser unerwünschten Stelle auch. Er Verzerrt die Antwortschwingung in Amplitude und Zeit. Die Amplitudenverfälschung ist nicht so kritisch, wie die der Zeit. Der Regler misst beispielsweise einen negativen Fehler (Sollwert-Istwert) obwohl im realen System dieser eigentlich positiv ist. Das ist in dem Fall recht schlecht, da die erste Antwortamplitude den Regler prägt.
Ein Ausweg ist möglich indem der Regler dementsprechend langsamer gemacht wird. Dadurch müssen kleinere Reglerkonstanten und längere Laufzeit (gleich Simulationsschritte) genutzt werden, um ähnliche Ergebnisse wie unter 7.1 zu erzielen. An dieser Stelle wäre es sehr wichtig mathematisch nachweisbare Zusammenhänge zwischen Filter und Regler zu finden und diese beim eigentlichen Reglerdesign einfließen zu lassen. Prinzipiell ist festgestellt worden, dass ein Integrationsbetonter Regler an dieser Stelle besser, aber auch langsamer wäre. Trotzdem folgt eine kleine Tabelle mit angepassten Parametern für unterschiedliche Schritthöhen (jeweils bei 200 Simulationsschritten je Schritt).

Schritthöhe	1nm	10nm	20nm	40nm	80nm	100nm
GainKn	50	700	1400	2450	2850	4000
GainKv	5	7	7	7	3	3
GainKp	25e4	16e4	16e4	19.35e4	19.35e4	21e4

Alle Werte sind nur experimentelle zu betrachten. Für das Abstimmen des Gesamtmodelles wird trotzdem der gegebene Regler verwendet mit experimentell angepassten Parametern.

7.3 Der Z Regler

Für die Positionsregelung der z Achse wird das Prinzip ein wenig abgewandelt. Für denn Ist- und Sollwertvergleich wird an dieser Stelle die wesentlich genauere Intensitätsmessung der Abtastnadel genutzt. Die Photodiode erzeugt im Freilauf der Nadel (zc=0) einen entsprechenden messbaren Nullpegel. In der Simulation soll ein kontinuierlicher Kontakt Mode (Kontaktkraft zwischen Spitze und Probe konstant) genutzt werden. Spitze und Probe befinden sich somit im kontinuierlichen Kontakt, wenn der Intensitätswert zu jeder Zeit ein wenig größer ist als der des Nullpegels. Dadurch ist der Regleristwert während der ganzen Simulation konstant. Er hat den Wert zc_Nullpegel+Delta. Wird die Nadel, durch Bewegung

der x und y Achse, über die Probe geführt ändert sich die Intensität in Abhängigkeit der Probenoberfläche (h_x_y). Durch Verstellung der z Achse wird das ausgeglichen. Damit wird der Ausgang des Reglers für die Positionsstellung der z Achse genutzt. O.B.d.A ist klar, das die geregelte z Position direkt proportional zur Oberflächenbeschaffenheit der Probe ist. Durch eine geeignete Aufbereitung und Speicherung der Koordinaten von x, y und z Achse kann später eine Detailgetreue graphische und messtechnische Verarbeitung erfolge

8 Weitere Module des Simulationsmodells

Der Großteil aller für das Gesamtsystem benötigten Module wurden in den vier vorhergehenden Kapiteln beschreiben. Damit eine komplette Simulation durchgeführt werden kann müssen weitere Primitive und Module genutzt werden, welche den Ablauf steuern und am Ende der Simulation eine geeignete graphische Aufbereitung durchführen.

Sollwertgenerator

Dieser generiert für den x und y Regler die eigentlichen diskreten Schrittsollwerte. Dabei wird in x und y Richtung Treppenstufen erzeugt. In x Richtung werden Stufen erzeugt, die eine Höhe der Rasterauflösung haben. Die Rasterauflösung (width_per_step) ist die Auflösung mit der die Probenoberfläche abtastet werden soll. Typisch sind hier Werte 10e-9, 20e-9 oder 40e-9 Meter. Nachdem eine bestimmte Anzahl (anz_steps) von Stufen erreicht wurde, werden diese in umgekehrter Richtung zurückgegangen. Bei Null angekommen widerholt sich das gleiche Spiel. Es entsteht eine diskrete Dreiecksfunktion. Bei den Extrema dieser Funktion erfolgt für die y Achse eine Stufenbildung. Die Breite der x Stufen wird als Parameter (shedulsteps_per_step) im Maßstab der Systemsimulationsschritte angegeben. Systemtypischer Wert ist hier 200. Die Wechselfrequenz der x Werte ist somit anz_steps höher als die der y Werte. Ein weiterer Parameter ist der Startwert der Treppe. Mit ihm ist es möglich eine Nullpunktverschiebung der Sollwertvorgaben zu machen.
Das Modul hat zwei binäre Eingänge. Der erste ist für den Startpunkt der Generierung. Dieser ist eins aktiv. Mit dem anderen Eingang kann die Wertgenerierung anhalten und fortsetzen werden.

Oberflächendarstellung

Damit man die Oberfläche, die eigentliche gemessene z Position, graphisch darstellen kann wurde die Senke „Waterfall“ angepasst. Hierdurch wird die Oberfläche auf einfache nicht maßstabsgerechte Art und Weise 3-dimensional dargestellt.

Start und Initialisierungssignal

Der Resetport des z Moduls für Filter und Positionsberechnung wird gleich zu beginn der Simulation deaktiviert (0 Pegel). Die beiden anderen Positionssysteme erhalten ihre Resetaufhebung nachdem ein erster Kontakt von Probe und Spitze festgestellt wurde. Das erfolgt über einen Testprimitiv, welches Spitzen Sollwert und Istwert vergleicht. Über ein Triggerprimitiv erfolgt eine genau einmalige Auswertung, so dass sich das Signal für den Rest der Simulation nicht mehr ändert.

Stop der Sollwerterzeugung

Über einen Tcl/Tk Button ist es möglich die Sollwertvorgabe zu stoppen. Diese Primitiv erzeugt aber nur Einsimpulse. Das alternate_trigger Primitiv erzeugt daraus eine alternierende Folge von Werten. Jedes wiederholte drücken des Buttons erzeugt ein Wechsel des Signals.

9 Integration und Tests

Nachdem alle für den Aufbau des Gesamtsimulationssystems benötigten Einzelmodule mehr oder weniger ausführlich in den letzten acht Kapiteln beschrieben wurden kann man sich der Integration aller Teile widmen. Eigentlich entstand das System schrittweise. Immer wenn Teilmodule in ihrer Beschreibung und Implementation fertig gestellt waren wurden sie Integriert. Sofern eine Validierungsmöglichkeit bestand wurde diese in den frühen Stadien schrittweise durchgeführt. Dadurch konnte man schon zu Beginn grobe Designfehler erkennen und teilweise beheben. Das so entstandne Gesamtsystem ist im Anhang als Abbildung graphisch dargestellt. Es gibt prinzipiell zwei Typen von Systemmodellierungen die für diesen Zweck in Betracht kamen.

Veranschaulichung von Sachverhalten und Funktionsabläufen

Es ist sehr gut möglich Objekte der realen Welt in ihrem meist sichtbaren Verhalten zu beschreiben. Komplexe und große Systeme werden hierdurch beschrieben und möglich gemacht. Trotzdem bestechen sie meist durch die Einfachheit und Verständlichkeit. In solchen System können Parameter dadurch grob abgeschätzt und in weiten Grenzen genutzt werden. Die Ergebnisse, welches eine solche Gestaltungsweise liefert, sind nur eingeschränkt nutzbar. Damit können die Resultate maximal so genau und aussagekräftig sein wie der Aufbau des Systems selber. Die Simulationszeit (speziell in der SDF Domain) ist weitaus geringer als bei Modellen des folgenden Typs.

Exakte mathematische Simulation

Eine Voraussetzung für diese ist, die detaillierte mathematische Beschreibung aller Abläufe. Sie kann nur dann eingesetzt werden, wenn das reale Objekt mathematisch in irgendeiner Weise beschreibbar ist und dessen Funktionsparameter bestimmbar sind. Hohe Komplexität bei großen Systemen und lange Simulationszeiten folgen daraus. Von ganzen Funktionsabläufen können deshalb nur Teilsimulationen gemacht werden. Solche Entwürfe lassen aber weitaus höhere Interpretation der Ergebnisse zu. Falsche oder ungenaue Beschreibungen führen zu Fehlern, die nicht unbedingt sofort sichtbar sind. Hierdurch entstehen so genannte verschliffene Fehler die das Ergebnis Fehlerbehaften. Es muss eine Abschätzung gemacht werden, in wie weit diese Fehler akzeptabel sind.
Folgende Teile sollten in dem hier vorgestellten System sichtbar sein:

- Steuerungsablauf für die Benutzung eines Atomkraftmikroskop
- Ablauf der Steuerung eines 3-d Achsensystems
- Darstellung eines Ergebnisses, in dem Fall eine simulierte Probenoberfläche
- Darstellung des Messprinzips (Interferometer) und dessen Fehlermodell
- Darstellung der Dynamik und Kraftwirkungen im Mikroskop
- Darstellung der Signalfilterung und Verarbeitung
- Benutzung eines vorgegebenen Regelungskonzeptes für die Stellachsen

Es ist deutlich sichtbar, dass die Teilaufgaben eine Verknüpfung beider zuvor beschriebener Typen darstellt. Dadurch werden exakt mathematisch beschriebene Teile mit oberflächlich arbeitenden Modulen verbunden. Diese sollen in gewünschter Weise zusammenarbeiten. Das führte zu sehr starken Problemen im Gesamttest. Hier speziell in den Bereichen der Modulübergreifenden Initialisierung und Parametrisierung. Hierdurch wurden Schwachstellen sichtbar und in einer sehr späten Phase des Entwurfs beschlossen zwei getrennte Systeme aufzubauen.

9.1 System ohne Filter und geringer Simulationszeit

Eine graphische Darstellung des gesamten Systems ist aus Platzgründen im Anhang zu finden. Alle Beschreibungen in diesem und nachfolgendem Kapitel sind an der Darstellung nachvollziehbar. Die Parameter der einzelnen Module stimmen mit den Vorgaben aus den vorigen Kapiteln teilweise überein. Anpassungen, Hinweise und bekannte Probleme bei Parameteränderungen werden in diesem Kapitel gemacht. Prinzipiell soll dieses System die Korrektheit der Ablaufsteuerung zeigen. Das bedeutet, dass Grenzwerte (Achsensteuerung) erreichbar sind und Maßstabsgerechte (Achsenposition und zeitliches Verhalten) Simulation erfolgt. Das dynamische Verhalten der Regler und der Achsen ist sichtbar. Die Interaktion zwischen Probe (repräsentiert durch z Achse) und der eigentlichen Abtastnadel ist deutlich erkennbar.

9.1.1 Beschreibung des Aufbaus

Der Aufbau der beiden unterschiedlichen Simulationsmodelle ist gleich, deswegen sollte die Überschrift der Abbildung nicht irreleiten. Es folgt noch mal eine kurze Beschreibung der einzelnen Module und aus der kronologischen Reihenfolge sollte der eigentliche Signalweg ersichtlich werden.

nano_scanner

Er enthält alle dynamischen Berechnungen des 3-d Achsensystems (3d_system) und der Nadelspitze (cantilever). Im Zusammenhang mit der Nadelspitze erfolgt die Berechnung der Kraft fz im Kontaktmodus und wird den Dynamikberechnungen der Spitze rückgeführt. Es wird die Kraft (N) berechnet, die von der Spitze auf die z Achse einwirkt. Das Modul hat 4 relevante Ausgänge für absolute Position x, y, z und zc (Spitzenposition). Es gibt einen Ausgang für die experimentelle Oberfläche (h_x_y) der Probe, welche abhängig von der Position der x und y Achse ist. Der andere terminierte Ausgang zeigt fz. Die Anzeige der beiden Ports ist für den Ablaufvorgang nicht relevant. Die vier Positionen werden einem Graphik Primitive zugeführt, das nach der Simulation die Werte an den Ports graphisch darstellt. Die Signale werden dem nächsten Modul als Eingang zugeführt.

laser_noise

Dieses bearbeitet das interferenzoptische Messprinzip und dessen mögliche Fehlermodell. Für die Spitzenposition wird eine Intensitätsdarstellung durchgeführt. In dem Modul existieren deshalb 4 parallele unabhängige Verarbeitungswege. Es enthält die vier Tcl/Tk Slider für die Einstellung der Fehlertolleranzen und die Refernzanzeige. Die Ausgänge nach rechts sind das Paar periodische Signal für die y Achse und der Intensitätswert der Spitze. Nach links für die z Achse und nach unten die der x Achse.

filter_dig_pos

Für jede Achse getrennt erfolgen hier die messtechnische Verarbeitung und die Berechnung der Position. Aus den eigentlichen relativen Messsignalbezügen werden wieder absolute hergestellt. Diese liegt dann im Maßstab der Wellenlänge als geteilte Festkommazahl oder Gleitkomma am Referenzausgang (ref) an. Um einen Maßstab Meter zu erhalten muss man die Werte mit der Laserwellenlänge multiplizieren.

**_regler*

Die Eingänge der Regler sind der gemessene/berechnete Istwert und der vorgegebene Sollwert (r_val). Für x und y wird der Sollwert durch die Bahnsteuerung (triangle_gen) und

für z ein konstanter Wert vorgegeben. Die Reglerausgänge werden über eine „fork“ Primitive und einem nötigen Delay den Stellachseneingängen des AFM rückgeführt. Jede Rückführung in ML-Designer muss mit einem Delay versehen werden, da sonst der Sheduler in einen Deadlock endet und die Simulation nicht durchgeführt werden kann.
Alle anderen Teile sind in der Beschreibung schon aufgetaucht und müssen an dieser Stelle nicht noch einmal näher erläutert werden.

9.1.2 Eingestellte Parameter

Der Großteil aller Parameter wurde in den entsprechenden Kapiteln bereits vorgenommen. Die Simulation des Systems erfolgt mit einem Frequenzskalierungsfaktor von 5000. Das entspricht in der SDF Domain 5000 Simulationsschritte (Systemparameter ohmega0) pro simulierte Systemsekunde. So kann in akzeptabler endlicher Zeit das Verhalten dargestellt werden. Dieser Parameter sollte nicht geändert werden.

Warum ohne Filter?

Wie im Kapitel 7.2 schon erwähnt war es nicht möglich die Reglerparameter experimentell auf den gegebenen Sachverhalt abzustimmen. Das schien an dieser Stelle an den Filtern zu liegen, wurde mit späterer Erkenntnis aber auf die zu niedrige Abtastung zurückgeführt. Es ist einfach nicht möglich bei einer 5 KHz Abtastung Digitalfilter einzusetzen, ohne das Signal stark zu verfälschen. Mit diesem Signal kam der Regler nicht zurecht. Deswegen wurde in diesem Modell auf die Filter verzichtet.

Warum einen geänderter z Regler?

Die Eigenfrequenz der Abtastnadel ist sehr hoch. Des Weiteren fehlte jegliche Information, wie man von einem solchen Verhalten der Spitze ausgehend geeignete Reglerparameter entwerfen kann. Deswegen war es nicht möglich die gegebene Reglerstruktur zu benutzen. Das neue Reglerkonzept ist von einem Schema diskret arbeitender PID Regler abgeschaut. Die Berechnung des neuen Reglerausgangs beruht auf dem vorhergehenden Wert. Dieser wird durch den neuen gemessenen, integrierten und differenzierten Fehler angepasst und ausgegeben. Es ist ein Hochintegrierender Regler entstanden, der eine gegebene Problematik beseitigt hat. Doch zuvor noch ein paar Hinweise zum Entwurf. Die Intensitätsberechnung ist so aufgebaut, das pro 1 nm Positionsänderung ein Intensitätswert von etwa 0.005 gemessen werden kann (1e-9*PI/Wellenlänge). Um diese Änderung kompensieren zu können muss der letzte Stellwert um etwa 0.09 angepasst werden (1e-9*450000/0.005). Bei der Berechnung floss die Linearitätskonstante, deren Bedeutung schon erklärt worden ist, mit ein. Folgende Werte sind für den z Regler benutzt worden:

AdaptionGain	0.143
GainKn	0.0009
GainKv	0.0009
KainKp	0.012

Die eigentliche Problematik besteht noch in der Initialisierungsphase eine Antastung der Probe (z Achse) mit der Spitze durchzuführen. Dazu wurde eine Erweiterung für die Initialstellung der z Achse im nano_scanner Modul vorgenommen.

z_nullpos	-632e-9

Damit ist klar, dass der z Regler eine Antaststrecke von einer ganzen Wellenlänge durchführt. Die Darstellung der Probenoberfläche am Ausgang des filter_dig_pos_z Moduls sind logischerweise mit einem Offset von 632 Nanometer behaftet. Das spielt in diesem Zusammenhang keine große Bedeutung, weil die Oberflächenabtastung nur absolute Bezüge

zur Oberfläche schafft, jedoch es nur schwer möglich ist absolute Bezüge zur Probe selber zu erzeugen.

9.1.3 Veränderbare und getestete Parameteränderungen bzw. Fehler

Es gibt zwei Arten von Parametern im System. Zum einen Parameter die den Ablauf steuern und zum anderen Systemparameter der einzelnen Module.

Modulsystemparameter

Sie entsprechen den Wertvorgaben aus unterschiedlichen Quellen und sind in ihrem jeweiligen Gültigkeitsbereich anzupassen. Sollten einmal Probleme nach dem Anpassen der Masse von der Spitze auftreten (im Modul nano_scanner) liegt es mit hoher Wahrscheinlichkeit an dem Skalierungsfaktor der Interaktionskraft fz. Hier ist nicht eindeutig klar, auf welchen Bereich diese Kraft normiert werden muss. Auf alle Fälle müssen Spitzenmasse (mc) und Skalierungsfrequenz (ohmega0) enthalten sein. Weiterhin wirkt sich eine Änderung der Probenmasse nicht aus. Die Masse (mp) ist für die Eingänge der Achsen ein Faktor. Das ist prinzipiell nur ein Konvertierungsproblem. Darunter ist der Sachverhalt zu sehen, wie der Ausgang der Regler (real sind das DAU's) durch die Treiberelektronik (Konverter) verstärkt oder angepasst wird und auf die Eingänge der Stellachsen einwirkt.
Für die Parameteranpassung des Lasermoduls existiert ein kleines System namens „gain_offset_error_test" welches die Änderungen gut sichtbar macht. Wenn die Wellenlänge des Lasers geändert wird, muss diese an unterschiedlichen Stellen angepasst werden (im Regler, nano_scanner und in show_pos_error). Es ist kein Systemparameter, wird aber von einer Großzahl andere Module als Initialmarkierung und Maßeinheit genutzt.
Bei der Berechnung der eigentlichen Position gibt es keinen Parameter, dessen Änderung sinnvoll ist. Der Interpolationsfaktor für die digital Interpolation darf nicht geändert werde, da hierfür eine technische Anpassung (siehe Kapitel 6.3) nötig ist. Die Parameter für die Offset und Verstärkungsfehlerkorrektur sind in Kapitel 6.1 ausreichend beschrieben. Wie sich der Parameter für die Hysteresekomparatoren bei der Interpolation auswirkt kann in dem System „up_down_test" nachvollzogen werden. Desto höher dieser Wert, desto später wird der neue Quadrantenübergang von Q3 auf Q0 erkannt.
Die Fehler bei falschen Wertanpassungen an den Reglern sieht man recht deutlich. Entweder arbeitet der Regler falsch oder es kommt zu einem aufschwingen der Achsenpositionen. Die Qualität der gemachten Anpassung kann sehr deutlich an der sichtbaren Antwort der x Achse auf die Stufenvorgabe gesehen werden. Man sollte gut kontrollieren, ob nach einer bestimmten Anzahl von Schritten auch die dazugehörige reale Position erreicht ist. Dasselbe muss nach dem Richtungswechsel gelten. Somit müssen die Extrema der Dreiecksfunktion eindeutig ausregelbar sein.

Ablaufparameter

Die erzeugte Oberflächenstruktur ist ein Parameter, der hervorragend die Funktionsweise des z Reglers beeinflussen kann. Er ist im AFM Module zu ändern. Die experimentelle Oberfläche wird durch eine Funktion sin(x)*sin(y) erzeugt. Der Wert der eingetragen wird multipliziert mit 10 gibt an, das nach so einer Länge genau ein Extrema der 2-d Funktion überquert wurde. Das bedeutet, wenn der Wert gleich der Schritthöhe ist und die Anzahl der Schritte 20 ist wird etwa ein Minima und Maxima der Funktion überquert. Macht man den Wert entsprechend kleiner, so wird bei gleichem zurückgelegtem Weg mehr erzeugt. Man wird erkennen, dass der z Regler wesentlich mehr Probleme damit hat. Ganz deutlich ist es an dem sehr langen Kontaktverlust der Nadelspitze zu erkennen.

Die Schritthöhe wurde in einem Bereich von 10, 20, 40 und 80 Nanometer getestet. Die erkennbaren Unterschiede liegen nur in dem Verhalten der Regler. Bei größeren Schritten ist die Zeit zum erreichen des Sollwertes dementsprechend länger. Auch die Oberschwingung nach dem erreichen und dem halten des Wertes sehen anders aus.
Eine Verlängerung der Stufenweite gibt dem Regler bei noch höheren Schrittweiten mehr Zeit seine Vorgaben auszuführen. Die Zeitliche Länge der Stufe ist simulationszeit_je_schritt/ohmega0 in Sekunden. Die Anzahl der Schritte in Verbindung mit der Schritthöhe ist ein Maß für die Größe der quadratischen Fläche, von der ein Abbild erzeugt werden soll. Natürlich ist diese nur quadratisch, wenn man auch die Gesamtzahl der Simulationsschritte richtig setzt.

9.2 System mit Filter und sehr hoher Simulationszeit

Das System welches im vorigen Teil aufgebaut wurde hat einen sehr stabilen Zustand über eine Vielzahl von Änderungsmöglichkeiten gezeigt. Um das neue System an die geforderten hohen Abtastraten anzupassen lag es deswegen nahe dieses als Basis zu nutzen. Zum Einstellen der hohen Frequenzen dient ein Parameter frequenz_faktor. Mit dessen Hilfe ist es nun möglich die Simulationsfrequenz als ganzzahliges Vielfaches von ohmega0 einzustellen. Getestet wurde hier bis zu einem Faktor von 20 der somit einer Simulationsfrequenz von 100 KHz entspricht.

9.2.1 Anfangsstabilitätsanpassung

Prinzipiell fangen die x und y Achse Initial bei einer absoluten Position von 0 an (das entspricht dem Quadrant 0). Da das digitale Zählverfahren nur positive counts zulässt ist von dieser Position ein Start sehr schlecht. Erfolgt der erste Quadrantenübergang nicht von 0 zu 1, sondern von 0 auf 3 (negativ counts) stimmt die Initialisierung nicht mehr. Das führt zu einer falschen Positionsangabe von dem Modul filter_dig_pos. In dem Ablaufverfahren wurde kein Referenzfahrtverhalten vorgesehen. An der z Achse ist zu sehen, das so etwas von Vorteil ist. Deswegen wurde für x und y eine künstliche Nullpunktverschiebung durchgeführt. Dies wird aber nicht im nano_scanner Modul vorgenommen. Der Grund dafür folgt gleich noch. Es wurde der digital count Anfangswert nicht 0 sondern 4 gewählt. Das führt zu einem berechneten Initialwert der Festkommazahl auf 1 (das entspricht einer Position von 1*wellenlänge). Somit muss auch die Sollwertvorgabe für die Regler bei einer Wellenlänge beginnen. Diese Anpassung ist kein Parameter des Moduls, deswegen ist natürlich bei sonst gleichem Aufbau die z Achse zur Oberfläche um zwei Wellenlängen offsetbehaftet. Der Regler hat deswegen eine stabile Vorsteuerung. Er erzeugt bei der ersten künstlichen Nullpunktregelung einen stabilen positiven Reglerausgang. Die Filter zeigen ein Verhalten, das die erste Sprungamplitude nicht unbedingt positiv ist. Das führt nun zu einer richtigen Erkennung und kann durch die Regler beseitigt werden.

9.2.2 Anpassung der Reglerparameter

Eine sehr einfache Lösung der Anpassung von den Reglern wäre die Nutzung von den Primitiven „UpSample“ und „DownSample“ gewesen. Dadurch hätte man sich die Anpassung der Reglerparameter an die geänderte Frequenz ersparen können. Wieder einmal hat ML-Designer gezeigt, dass es damit nicht umgehen kann. Fehlermeldungen der Verklemmung des Shedulers konnten nicht beseitigt werden. Wie schon erwähnt fehlte jegliche Dokumentation zu den Reglern und die Parameter wurden experimentell ermittelt. Es war überraschend, dass durch eine kleine Änderung die Regler über alle Frequenzbereiche stabil arbeiten. Das sind für x und y Regler folgende:

GainKn	4000.0/$frequenz_faktor
GainKv	5.0/$frequenz_faktor

Und für den z Regler wie folgt:

AdaptionGain	0.143/$frequenz_faktor
GainKn	0.0008/$ frequenz_faktor
GainKv	0.0009/$ frequenz_faktor

Die neuen Reglerparameter sind somit nur von dem Parameter ferquenz_faktor abhängig. Somit kann eine Abstimmung des Reglertyps aus dem einfachen Simulationsmodell sehr gut in dieses integriert werden. Das ist eine wünschenswerte Forderung beim Reglerentwurf.

9.2.3 Abhängigkeit der Anzahl Filterkoeffizienten von der Frequenz

Nachdem sich die Verwendung von adaptiven Filtern als nicht akzeptabel erwies wurden zu Letzt doch reine FIR Filter eingesetzt. Im Modul filter_dig_pos gibt es einen weiteren Parameter, mit dem die Koeffizienten in einem Floatarray angegeben werden können. Bei Tests hat sich ein Zusammenhang der Anzahl der Taps und der Simulationsfrequenz herausgestellt. In einem Testszenario mit folgenden Einstellungen kann das nachgewiesen werden:

frequenz_faktor	6	6	12	12
Anzahl Filter Koeff.	32	64	32	64
Regler Verhalten	stabil	instabil	stabil	stabil

Bei sonst gleich bleibender Grenzfrequenz des Filters. Bei zu hoher Anzahl der Filterkoeffizienten kommt es zu einem aufschwingen des Reglers. Die Ursache hierfür ist in dem Zusammenspiel von Filterausgang und Regler zu suchen. Eine erste Deutung weist auf die Geschwindigkeit des Reglers mit der er versucht den Fehler auszugleichen und der Signalverschiebung, die unweigerlich durch den Filter erzeugt wird. Es wird sich an dieser Stelle vorbehalten einen mathematischen Nachweis oder Beweis zu führen, da die eigentlichen Grundlagen hierfür fehlen.
Die exakte Berechnung der Koeffizienten kann erst dann vorgenommen werden, wenn Eigenfrequenzen der realen Mikroskopachsen gemessen werden können und Fehlertoleranz der Interferometer bzw. des Photodetektor vorliegen.

10 Bewertung und Abschluss

Der Umgang und die Einarbeitung in das Entwurfswerkzeug ML-Designer am Beispiel der gestellten Aufgabe resultierte in einigen Erkenntnissen. Das eigentliche Potential, welches dieses Entwicklungswerkzeug für den Systementwurf und Analyse zur Verfügung stellt, wurde nur teilweise genutzt. Um gute und effektive Entwürfe zu erzeugen bedarf es einem geschulten Umgang mit den Mitteln von ML-Designer. Das Sprichwort „Übung macht den Meister“ ist speziell im Bereich des schöpferischen Entwurfs sehr zutreffend. Im Folgenden sollen gemachte Erfahrungen genannt und erläutert werden. Ein Einsatzgebiet für ML-Designer im Fachbereich „Rechnerarchitekturen“ liegt im Rechnerentwurf eingebetteter Systeme.

10.1 Konventioneller Entwurfsvorgang

Speziell das „Rapid Prototyping“ verlangt einen durchgängig durch Entwicklungswerkzeuge geführten Entwurfsvorgang. Somit ist es möglich in kürzester Zeit lauffähige Prototypen zu entwickeln und die gemachten Entwürfe für ein Serienprodukt zu nutzen. Die Zeit von der Idee bis zum Betrieb des Systems könnte in folgende Phasen zerteilt werden.

a) Analyse
b) Entwurf
c) Validierung und Verifikation
d) Implementierung und Realisierung
e) Validierung und Verifikation auf der Zielhardware
f) Betrieb

In der Phase der Analyse werden Informationen gesammelt die Aussagen über Funktionen und Anforderungen machen. Diese Phase ist geprägt durch sammeln und auswerten von Fachliteratur, Veröffentlichungen und Spezifikationen. Für den eigentlichen Entwurf kann dann die Designsoftware genutzt werden. Wie der Entwurf hierarchisch gegliedert werden kann, wurde in einem frühen Kapitel schon erläutert. Da zu Beginn keinerlei Realisierungsinformationen vorliegen ist der „Top-Down-Entwurf“ eine der häufigsten genutzten Vorgehensweisen. Der Entwurf erfolgt in Abstraktionsgraden (Levels). Das eigentliche Simulationssystem (Level 0, das höchste) wird durch das Umgebungs- und Systemmodell gebildet. Bei komplexen Systemen enthalten die Ebenen 1, 2 oder auch 3 mit hoher Wahrscheinlichkeit keinerlei Realisierungs- und Implementierungsdetails. Diese Funktionalität wird in ML-Designer erst auf der Ebene von Primitiven bereitgestellt. Es erfolgt eine Unterteilung in Funktionsblöcke auf den jeweiligen Abstraktionsniveaus und den Zuordnungen zu den entsprechenden Domainen. Trotzdem weisen diese ersten Schichten einen hohen Informationsfluss in den Bereichen Datenaustausch (Schnittstellen zwischen Blöcken und Modulen), Parameterabhängigkeiten, Synchronisation und Initialisierung auf. Erst wenn alle Ebenen soweit Verfeinert wurden bis sie nur noch aus Primitiven bestehen kann mit der Validierung und Verifikation Fortgefahren werden. Dies ist die zweite Stärke von ML-Designer. Durch die unterschiedlichen Simulations- und Testverfahren erfolgt die Validierung mit den Anforderungen aus der Analyse. Dazu ist es nötig Signale/Daten (Testfälle) zu erzeugen (Quellen) und die entstandenen Resultate durch Senken Primitive darzustellen. Für die Realisierung und Anpassung auf die Zielhardware (sofern sie sich von der Entwicklungshardware unterscheidet) gibt es einen synthetischen oder einen heuristischen Weg. Partitionierung und Spezifikation aus der Entwurfsphase bilden für die heuristische Implementierung das Grundgerüst. Die gute Interpretierbarkeit und die Implementierungs-

nähe der Primitive ermöglichen eine schnelle Umsetzung im Vergleich zu herkömmlichen Entwurfsvorgängen. Die synthetische und automatische Realisierung erfolgt mit Hilfe der Code Generierungsdomains. Diese werden momentan experimentell für VHDL und C Code für x86, Motorola 56000 und Texas Instrument TMS320C5x angeboten. Dabei sei darauf hingewiesen, dass es keine automatische Konvertierung gibt, die das Entwurfsmodell umsetzt. In den Code Generierungsdomains ist keine Simulation möglich. Nach erfolgreichem Test mit Hilfe von „Remote Debuggern“ oder Emulatoren kann von einem funktionstüchtigen System ausgegangen werden und einem Betrieb steht nichts mehr im Wege. Natürlich unterscheiden sich Hardware und Software Prozesse in Details, aber das Vorgehensmodell ist ähnlich.

10.2 Geänderte Entwurfsmethodik

ML-Designer gewährleistet momentan noch keine durchgängige Entwurfsmethodik. Das grundlegende Konzept von ML-Designer liegt in der Nutzung des Simulators. Wie schon erwähnt lässt sich dieser jedoch in den frühen Stadien beim entwerfen der höheren Abstraktionsschichten nicht einsetzen. Hier fehlen Beschreibungsmittel, die auch in irgendeiner Art und Weise ein Testen zulassen. In dieser Zeit gemachte Entwurfsfehler werden erst beim erreichen der niedrigsten Ebene sichtbar. Denn erst jetzt sind alle Teilmodelle simulationsfähig, nur lassen sich dann höher Fehler nur schwer lokalisieren. Das Verfeinern jeder Abstraktionsebene nach erfolgreichem Testen ist nicht möglich. Deswegen wurde eine etwas andere Vorgehensweise gewählt.

a) frühes teilen in Funktionsblöcke (Analyse Phase)
b) Spezifikation des Ein- und Ausgangsverhalten je Block
c) Entwurf bis in die niedrigste Ebene anhand des Funktionsumfangs
d) Entwurf von Testumgebungen für jeden Block extra
e) Testen aller Blöcke in getrennten Umgebungen und Überprüfung des korrekten Verhaltens
f) falls Fehler auftreten müssen diese Blockweise behoben werden
g) Verknüpfung der Teilblöcke zu einem Gesamtmodell

In dem hier vollzogenen Entwurf sind 4 Teilblöcke entstanden. Das sind die Regelstrecke, das Laserinterferometer, der Rechnerteil und die digitalen Regler. Es ist relativ wichtig, dass diese Teilsysteme erhalten bleiben. Sie dienen später zur Konsistenzprüfung. D.h. beim Aufbau des Gesamtmodells aus den einzelnen Teilen können Probleme auftreten.

a) Im Verhalten der Initialisierung.
b) In der Synchronisation der Teilblöcke.
c) In der unterschiedlichen Hierarchieauflösung in den einzelnen Blöcken.

Speziell Systeminitialisierung die bei der Umgebungsmodellierung auftreten sind recht schwer beherrschbar und lassen sich nur durch geeignete Festlegungen beheben. Bei Änderungen in den Modulen die zur Beseitigung gemacht werden, sollte unbedingt nachträglich überprüft werden, ob das geänderte Modul noch 100%ig in seiner Testumgebung funktioniert. Die Gefahr verschleifender Fehler ist hier besonders hoch. Die Auswirkungen von Entwurfsfehlern sind nicht am Entstehungsort erkennbar. Auch das nachträgliche überprüfen in den untersten Ebenen durch Verwendung von Anzeigeelementen hat sich als schwierig herausgestellt. In dem Beispiel wurden bspw. drei Instanzen des „filter_dig_pos“ Moduls eingesetzt. Ein einfügen einer „XGraph“ Primitive (das entspricht einer Moduländerung) zur Visualisierung in eines der Module bewirkt aufgrund der Konsistenzerhaltung ein dreimaliges auftreten. Die Auswertung erweist sich in diesem Fall als schwierig, da auch die automatische Beschriftung ein wenig kryptisch und verwirrend erfolgt. Auch das Durchschleifen von Beobachtungssignalen bis in die Ebene 0 ist nicht

empfehlenswert. Dadurch wird die graphische Struktur zerstört. Man merkt sehr schnell, das eine Struktur die auch beim Schaltplanentwurf genutzt wird sehr vorteilhaft ist. Der Übersichtlichkeit halber sollte immer versucht werden von Links nach Rechts zu entwerfen und Signalwege wenn möglich nicht zu überlagern.

10.3 Mögliche Verbesserungen

Es gibt eine Reihe von Erweiterungen, die ein Entwurfswerkzeug für einen durchgängigen Systementwurf unterstützen sollte. Welche das sein könnten und für was man sie verwendet wird nachfolgend kurz erläutert.

Beschreibungselemente aus der Objektorientierten Softwareentwicklung

Sie sind zwingend nötig, um ein Beschreibungsmittel in den höheren Abstraktionsschichten bereit zu stellen. Da in diesen Ebenen keinerlei Implementierung oder Realisierung vorhanden ist könnten trotzdem durch diese Elemente Tests durchgeführt werden. Dadurch ist eine frühe Validierung des Entwurfs möglich und die Fehleranfälligkeit sinkt. „Use-Cases“ beschreiben Aktionen zwischen Umgebungs- und Systemmodell auf der Abstraktionsebene 0. Mit Hilfe von Sequenzdiagrammen kann man das zeitliche Verhalten und Interaktionen von Teilblöcken auf einer Abstraktionsebene darstellen. Das Zusammenwirken der Einzelteile in einem Modul bildet die Verfeinerung eines logisch getrennten Aufgabenfeldes einer höheren Ebene. Bei Modellierungen in denen es mehr um die Abstraktion des Datenaustausches geht sind „Message-Sequnce-Charts“ als Erweiterung günstig. Der Einsatz von Zustands-, Kollaborations- und Klassendiagrammen rundet das ganze Konzept ab. Natürlich muss Aufwand und Nutzen in einem guten Verhältnis stehen, weil diese Beschreibungsmittel im Vergleich recht viel Zeit in Anspruch nehmen.

Debugger

Der Funktionsumfang, wie er in Programmiersprachen benutzt wird ist nicht nötig. Wichtigstes Kriterium wäre ein „watch Modus“. Es müsste möglich sein, ohne graphische Modellierungseingriffe, jeden beliebigen Zustand oder Signalfluss dynamisch zur Simulationszeit zu beobachten. Das ist mit einer hierarchischen Auflistung aller Parameter und den dazugehörigen Werten aus dem Simulationssystem vergleichbar. Eine erleichterte Fehlersuche und ein größere Beobachtungsbereich wären möglich.

Code Generation

Diese Domain fordert einen recht praxisnahen Bezug. Besonders in Systementwürfen die immer wieder auf gleiche unterliegende Zielhardware aufbauen entstehen speziell angepasste Funktionsbibliotheken. Diese wurden im Laufe der Zeit durch die Entwickler aufgebaut und schrittweise optimiert. Man kann hier von Firmenspezifischen (persönlichen) Bibliotheken reden. Es ist ein Nachteil, das Code Generierungs- und Simulationsdomaine getrennt sind. Dieser muss irgendwie kompensiert werden. Das bedeutet, zu der speziell angelegten Zielhardwareabhängigen Code Generierungsdomain (unterschiedlichste Primitiven) muss es funktionsäquivalente Primitiven in der Simulationsdomaine geben. So das nach einem Aufbau, Test und Validierung des Modells in der Simulationsdomaine ein automatisches direktes mapping in ein Modell der Code Domaine erfolgen kann. Das setzt natürlich eine Kennzeichnung der relevanten Module voraus. Es ist bspw. für die Software Generierung in eingebetteten Rechnern nicht sinnvoll ausführbaren Code für Module, die Umgebungsmodelle darstellen, zu generieren. Weiterhin muss es Mechanismen geben, die es ermöglichen Funktionalitäten aus dem Bereich der Betriebssysteme in die Generierung von Code

einfließen zu lassen. Bspw. benötigt die Software Realisierung einer FSM kaum Betriebssystemfunktionalität. Stattdessen sind bei DSP Anwendungen wie Timer Interrupte für digitale Regler und Filter, sowie Sheduler für die Verwaltung mehrerer gleichzeitiger Aufgaben zwingend nötig. Der geschilderte Mechanismus ist für „Rapid Prototyping“ mit schneller Anpassung an unterschiedliche Zielhardware von großem Nutzen. Zumal die Unterstützungsinitiative vom eigentlichen Hardware Hersteller (bspw. CPU Hersteller) kommen kann.

Schnittstelle für Emulatoren und Remote Debugger

Um das Gesamtkonzept des schnellen Entwurfs abzurunden muss noch eine Validierung von Simulationsergebnis und ausgeführtem synthetisch erzeugten Programm der Zielhardware erfolgen. Natürlich sind mit solch einfachen Mitteln keine harten Echtzeitanwendungen prüfbar, aber Fehler im Logischen oder in Abläufen könnten erkannt werden. Eine sehr hilfreiche Erweiterung ist auch das aufnehmen von realen Messwerten über das Zielsystem, die anschließende Einspeisung in ML-Designer und die Simulation über diese Werte. Die große Anzahl unterschiedlicher Schnittstellen von Emulatoren und Debugger wird solche Features nur für eine kleine Menge ausgewählter Produkte (CPU’s) zulassen und effizient erscheinen lassen.

10.4 Abschluss

Aus rein programmiertechnischer Sicht bleibt zu sagen, dass eine rein graphische Modellierung immer mehr Ressourcen (Zeit, Material und Kosten) benötigt. Aus rein Komplexitätsbewältigenden Sicht bietet die Nutzung von graphischen Mitteln enorme Vorteile in Hinblick auf Dokumentiertheit, Verständlichkeit und Überschaubarkeit. Im Nachhinein betrachtet würde der Entwurf eines ähnlichen Beispieles, aus persönlichen Erfahrungen heraus, anders aufgebaut werden. So würden speziell das Modul „nano_scanner“ und der digitale Interpolator Zweig viel stärker programmiertechnischer gelöst. Die Vereinfachungen aufgrund schlechter graphischer Modellierbarkeit an diesen Stellen haben zu den im Kapitel 9 dargestellt Problemen geführt. Zu viele Abstraktionsebenen und Benutzung von Bibliotheksprimitiven sind nicht förderlich für die Vorteile, die ein solches Entwurfswerkzeug bieten soll.

11 Erste Schritte mit dem Modell

Um den Einstieg, in die Benutzung des entstandenen Modells, zu erleichtern sollen die folgenden sechs Schritte ein kleines Nutzungsschema darstellen.

11.1 Die Quellen

Es gibt zwei gepackte Archive Namens *„afm.zip“* [1]und *„afm_zu.zip“*. Das erste dieser beiden enthält das Modell mit geringer Simulationszeit und ohne Filter. Das Andere ist das Größere und wurde deshalb afm_zusatz benannt. Diese Archive sollten nicht unter Windows verändert werden. Die Dateinamen der Daten im Archiv dürfen nicht geändert werden (in Groß- und Kleinschreibung), was unter Windows automatisch passieren kann.

11.2 Extrahieren der Quelle

Bei den Erläuterungen wird sich ab hier auf das erweiterte Modell bezogen. Aufgrund der Namensgleichheit in beiden Archivinhalten kann immer nur genau eines von beiden genutzt werde. Dieses Archiv enthält ein Verzeichnis *„afm_steuerung.lib“* und eins Namens *„filter“*. Bevor die Verzeichnisse entpackt werden, sollte ML-Designer mindestens einmal gestartet werden. ([3] quickguide.pdf bietet eine gute Grundlage für dessen Benutzung) Ein Verzeichnis im Home Bereich des Nutzers wurde automatisch von ML-Designer angelegt (*$home/MLD*). Das Verzeichnis *„afm_steuerung.lib“* kann nun nach *$home/MLD* extrahiert werden. Das Verzeichnis *„filter“* kann an eine geeignete Stelle im Home Bereich entpackt werden. Es enthält eine Reihe von Dateien, die Filterkoeffizienten enthalten. Die Kennzeichnung erfolgte nach dem Schema *„FilterTyp_AnzahlTaps_Grenzfrequens“* des Filters. Die Anzahl der Filterkoeefizienten wird durch die Anzahl der Werte in der Datei bestimmt.

11.3 Starten von mld

Nach dem starten von ML-Designer ist eine neu Bibliothek *„afm_steuerung“* im *„Library View“* Fenster erkennbar. Sie enthält alle im modellierten System neu geschaffenen Module und Primitive. In der ersten Verzeichnishierarchie sind die Simulationssysteme *„ablaufsteuerung“*, *„gain_offset_error_test“*, *„show_error_timing“*, *„up_down_test“* und *„modell“* zu finden. Die Unterbibliotheken *„de_module“*, *„logic“*, *„math“* und *„regler“* enthalten entsprechende Module und Primitive die zum Aufbau des Simulationsmodells *„modell“* nötig sind. Dieses kann durch einen Doppelclick geöffnet werden. Im Anhang befindet sich eine komplette Dokumentation des gesamten Verzeichnisbaums.

11.4 Designansicht des Systems „modell“

Im Design Fenster von ML-Designer wird nun das Gesamtsystem dargestellt. Die Abbildung ist äquivalent zu der Struktur, die im Anhang dieser Ausarbeitung zu finden ist. Eine kurze Beschreibung, wie diese Struktur zu deuten ist, findet man im Kapitel 9.1.1. Bevor zur Simulation übergegangen werden kann, müssen zwei Aspekte erläutert werden.

1. **Zeitverhalten:** Um einen Zeitlichen Sachverhalt in einer diskreten Zeitsimulation herzustellen ist es nötig zu wissen, wie viele Simulationsschritte pro Systemsekunde ML-Designer durchführt. Der Systemparameter *„ohmega0“* ist standardmäßig mit

[1] Alle spezifischen Namen und Bezeichner werden der besseren Übersicht in Anführungszeichen und Kursiv dargestellt.

5000 vorbelegt (sollte nicht geändert werden), was einer Frequenz von 5 KHz entspricht. Anpassungen an die geforderten 100 KHz Filterabtastfrequenz kann unter zu Hilfenahme des Systemparameters *„frequenz_faktor“* durchgeführt werden. Die Anzahl der Simulationsschritte (Anzahl der Berechnungen) je Sekunde sind somit Vielfache von 5000.

2. **Ablaufverhalten:** Damit das Verhalten des AFM (Atom Kraft Mikroskop) sichtbar wird müssen Positionssollwertvorgaben für die x und y Achse gemacht werden. Für die x Achse wird eine diskrete Dreiecksfunktion erzeugt. Es gibt *„anzahl_steps“* viele Stufen (da es eine diskrete Funktion ist), die jeweils eine Höhe von *„schritthöhe“* haben. Ist ein Weg von „*anzahl_steps*schritthöhe“* zurückgelegt, wird derselbe Weg umgekehrt durchgeführt. Zuvor jedoch der y Sollwert um *„schritthöhe“* erhöht. Dem Modell wird eine Zeit von *„simulationszeit_je_schritt/ohmega0“* (das sind 40 ms) zugesichert, um den vorgegebenen Sollwert auszuregeln. Das entspricht einer Anzahl von *„simulationszeit_je_schritt*frequenz_faktor“* vielen Berechnungen die ML-Designer für jede Sollwertvorgabe berechnen muss. Die Voreinstellungen für die „schritthöhe“ ist 40e-9 Meter (40 nm) und mit 20 Stufen wird somit ein Weg von 800 nm beschrieben.

11.5 Starten der Simulation

Aus den zuvor gemachten Betrachtungen kann der Simulationsparameter *„Run length“* bestimmt werden. Bspw. müssen für die Abtastung eines quadratischen Feldes in x und y Richtung *„simulationszeit_je_schritt*frequenzfaktor*anzahl_steps²“* viele Berechnungen durchgeführt werden. Nach dem starten der Simulation werden zwei Fenster aktiviert.

1. ***„tclRunControl“*:** Hier werden alle graphischen Nutzerelemente angezeigt.
   a) Der Wert für die z Achsen Positionsangabe (dritte Wertanzeigeelement von oben) wird kontinuierlich zu beginn steigen.
   b) Hat dieser einen Wert von etwa zwei Wellenlängen (2*632e-9) erreicht, ist die Antastung der Probe an die Nadelspitze abgeschlossen.
   c) Die Sollwertvorgabe für die x und y Achse wird aktiviert.
   d) In den ersten beiden Anzeigeelementen werden die Positionssollwertvorgaben der x und y Achse angezeigt. Das der Beginn bei einer Wellenlänge erfolgt ist reine Festlegungssache (siehe 9.2.1). Die Frequenz, mit der sich die x Werte ändern, ist *„anzahl_steps“* größer als die Frequenz der y Werte. Die z Werte ändern sich ständig, weil sie die Probenoberfläche repräsentieren.
   e) Das unter 11.4 beschrieben Ablaufverhalten sollte anhand des Wertepaars für x und y nachvollzogen werden können.
   f) Die beiden *„BarGraph“* Anzeigeelemente zeigen den Fehler zwischen dem durch die modellierte Rechnerfunktion berechneten (mit Signalfiltern, digitaler Interpolation und trigonometrischer Funktion) und der am Modell des AFM messbaren Position. Der obere ist für den y Positionsfehler und der untere für x.
   g) Wenn die Filterkoeffizienten (fir_32_0.10) nicht geändert werden, sind die Fehlerschwankungen bei kleinen „frequnez_faktor“ (bspw. 5) Werten höher als bei großen Werte (bis 20).
   h) Mit dem Stopp Taster kann man die Sollwertvorgabe anhalten. Durch wiederholtes betätigen wird sie fortgesetzt. Dies ist später in den Simulationsergebnissen sichtbar. An den Stellen ist das zeitliche halten auf einer Stufe erkennbar, da diese viel länger als alle anderen ist.

2. ***„tkPlotnoiseReference“*:** Hier wird der Ausschnitt eines Signals angezeigt, dem der Rechnerteil als Eingang zur Verfügung steht. Das ist nur eine Repräsentative Anschauung eines Stufendynamischen Bereiches für einen ungeregelten Achsenschritt. In Abbildung 6-1 ist eine wirkliche Schrittfolge sichtbar. An diesem Teilausschnitt werden die Auswirkungen von Signalfehlern der Laserinterferometer sichtbar gemacht.
   a) Die Benutzung der Schieberegler aus dem *„tclRunControl“* Fenster ermöglicht die Wertmäßige Vorgabe der Fehlergrößen.
   b) Die Fehler werden dynamisch modelliert und ihre Auswirkung in dem Fenster angezeigt.
   c) Phasenfehler werden nicht angezeigt, da hierfür zwei Signale nötig sind.
   d) Möchte man den Restlasersignalfehler im erzeugten Positionssignal der *„filter_dig_pos“* Module sehen, so kann am *„ref“* Port eine *„XGraph“* Primitive angebracht werden.

11.6 Ende der Simulation und Ergebnisauswertung

Die Hauptauswertung erfolgt über das Fenster *„zeigt absolute Position“*. Ein vorzeitiger Abbruch der Simulation und Anzeigen der Ergebnisse ist durch Betätigung des *„early end“* Tasters vom *„Run modell“* Fensters möglich.

1. ***„zeigt absolute Position“*:** Hier wurden die vier Positionssignale des AFM über den Simulationszeitraum aufgezeichnet.
   a) X und y Achse weisen ein geregeltes stufenförmiges Verhalten auf. Wie die Stufen ungeregelt aussehen ist in Abbildung 13-1 deutlich zu erkennen. Die Restschwingung (Oberwellenschwingung), um den eigentlichen Sollwert, wird durch schlecht abgestimmte Reglerparameter hervorgerufen.
   b) Für die Position der z Achse ist im ersten Teil der Antastvorgang zu erkennen. Für den weiteren Verlauf folgt die Kurve einer Ausgeregelten *„sin(x)*sin(y)“* Form (simulierte Probenoberfläche). Aufgrund der 2-dimensinalen Darstellung ist die Oberflächenstruktur in diesem Signal nicht gut sichtbar, aber trotzdem nachvollziehbar.
   c) Die Position der Nadel ist relativ konstant im Bereich der Null. Ihre Schwingung ist mal stärker oder schwächer, in Abhängigkeit des Kontaktes mit der Probenoberfläche. Deutlich sichtbar ist das Verhalten bei Vollkontakt während der Antastung.
2. **Anzeige zu Testzwecken:** Es wird das gemessene Intensitätssignal der Nadelspitze angezeigt, welches durch die Focussteuerung erzeugt wird. Das Signal ist durch freies Schwingen oder Kontakt der Nadel geprägt. Bei Kontaktverlust schwingt die Nadel gleichmäßig um den *„zc_nullpegel“* Wert. Beim Kontakt folgt sie der Dynamik der z Achse, sofern ihr das dynamisch gesehen möglich ist.
3. **„waterfall_invert“:** Zeigt die 3-dimensionale Struktur der Oberfläche aus den gefilterten und berechneten Signalen des z Interferomters. Eine wirklich anschauliche Repräsentation wird nur deutlich sichtbar, wenn die abgetastete Fläche groß genug ist. Diese Anzeigeprimitiv war in erster Linie für die einfache *„afm_steuerung“* in der Datei *„afm.zip“* gedacht. Bei hinreichend hoher Rechenleistung oder Zeit sollte sie auch hier funktionieren.

12 Quellen

[1] http://www.theoinf.tu-ilmenau.de/nanomess/

[2] Neue DSP-Hardware- und Softwarelösungen für den Einsatz in Mehrkoordinaten-Nanomeß- und Positioniersystemen
Iwk2000_dsp_pap.pdf
Iwk2000_dsp_fol.pdf

[3] http://www.mldesigner.com
mld2.pdf
quickguide.pdf

[4] S.Salapaka, M.Dahleh: "Friction in Atomic Force Microskopy : Modeling & Control" Department of Mechaincal and Environmental Engineering, University of Califarnia, Santa Barbara

[5] G. Schitter, P.Menold, H.F.Knapp, F.Allgöwer, A.Stemmer: "High performance feedback for fats scanning atomic force microscopes" Review of Scientec Instruments Volume 72, Number 8 August 2001

[6] N A Burnham, O P Behrend, F Oulevey, G Gremaud,P-J Gallo, D Gourdon, E Dupas, A J Kulik, H M Pollock and G A D Briggs : "How does a tip tap?" Nanotechnology (1997) 67–75.

[7] Axel Donges; Reinhard Noll: Lasermesstechnik: Grundlagen und Anwendungen; Heidelberg : Huethig, 1993

[8] Ernst, Alfons: „Digitale Längen- und Winkelmesstechnik" ; Positionsmeßsysteme für den Maschinen- und Gerätebau sowie die Elektroindustrie ; Verlag „modern industrie"; Heidenhain

[9] Universität Paderborn; Physikalisches Fortgeschrittenenpraktikum PD; Anleitung zum Versuch: Längenmessung; Ausgabe: 16. Oktober 2000

[10] Analog Device tech. Paper: Closed Loop Position Estimation for Sinusoidal Encoders with the ADMC401 AN401-23
www.analog.com/marketSolutions/motorControl/applicationCode/admc401/pwm401_1.html

[11] Adaptive Filters - Matlab Tools; FH-Mannheim - Institut für Digitale Signalverarbeitung; Prof. Dr. B. Wirnitzer ver 1.0 Oct. 99

[12] M. Krapp; „Digitale Automaten" 1988 S.89 ff.

[13] Technische Universität Ilmenau Fakultät für Maschinenbau Institut für Prozeßmeß- und Sensortechnik; Tino Hausotte Project: Nanomeßmaschine

[14] M. Wolf ; Script „Objektorientiert Prozessmodellierung"

13 Abbildungsverzeichnis und Anhang

ABBILDUNG 2-1: SCHEMATISCHE DARSTELLUNG DES GESAMTSYSTEMS 4
ABBILDUNG 3-1: HIRARCHIE IN ML-DESIGNER 7
ABBILDUNG 4-1: MECHANISCHE STRUKTUR EINES AFM 9
ABBILDUNG 4-2: AUFBAU U. ERGEBNIS DER VORBETRACHTUNG 11
ABBILDUNG 4-3: KOORDINATENSYSTEM DER SPITZE 11
ABBILDUNG 4-4: JKR KRAFT 12
ABBILDUNG 5-2: DETEKTORSIGNALE UND SIGNALFEHLER 16
ABBILDUNG 5-3: DYN. UND STAT. GAINFEHLER PRINZIP 17
ABBILDUNG 6-1: SIGNALINPUT DES RECHNERTEILS FÜR EINE ACHSE 19
ABBILDUNG 6-2: FLUSSDIAGRAMM DES ALGORITHMUS 20
ABBILDUNG 6-3: ADAPTIVER FILTER 21
ABBILDUNG 6-4: ZWEIFACHE DIGITALE INTERPOLATION 22
ABBILDUNG 6-5: DARSTELLUNG DER POSITION 24
ABBILDUNG 13-1: ACHSVERHALTEN BEI STEUERSCHRITTEN 42

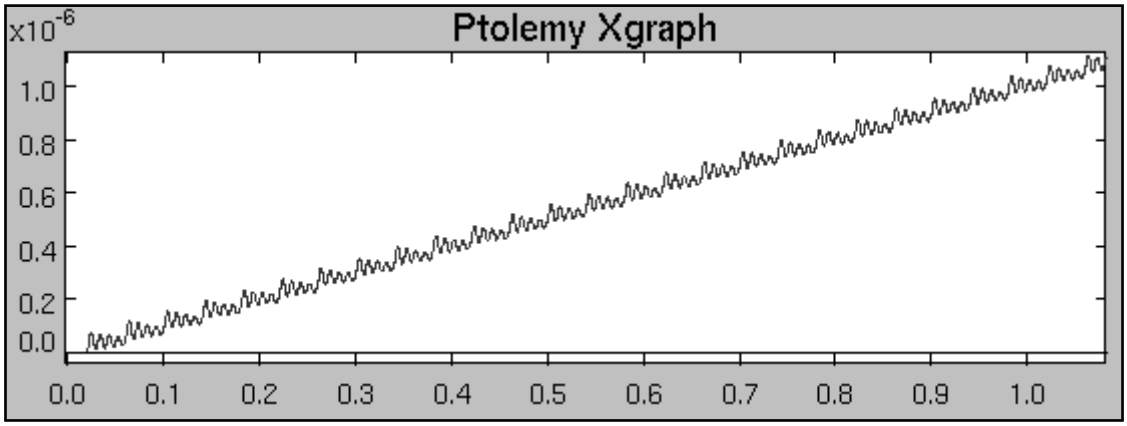


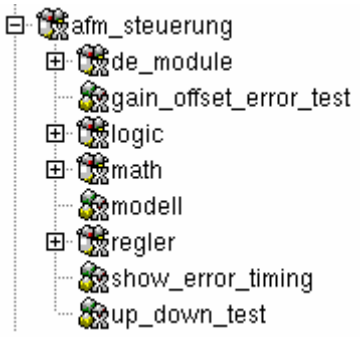
Abbildung 13-1: Achsverhalten bei Steuerschritten

Die Verwaltung von Projekten unter Ml-Designer wird mit Hilfe von Verzeichnishierarchien durchgeführt. Dabei obliegt die Anordnung von eigenen entworfenen Primitiven und Modulen zu Bibliotheken dem Entwickler. Für einen ungehinderten Entwurf ist es deshalb wichtig diese intuitiv schnell wieder zu finden und für den eigentlichen Verwendungszweck einzusetzen. Jedes Elementarverzeichnis kann folgende Arten von Dateien enthalten:

*.htm	enthält die Dokumentation eines Elementes, speziell auch die Ports
*.mml	enthält die Modullbeschreibung in Form von XML Text
*.std	Repräsentation einer FSM
*.pl	ist die Quelldatei einer Primitve, nimmt Quellcode des Nutzers geordnet auf
*.h	aus *.pl generierte Headerdatei
*.cc	generierte und kompilierbare Datei
*.o	Objektdatei für die Ausführung der Simulation
makefile	für eine geordnete Benutzung des Compilers

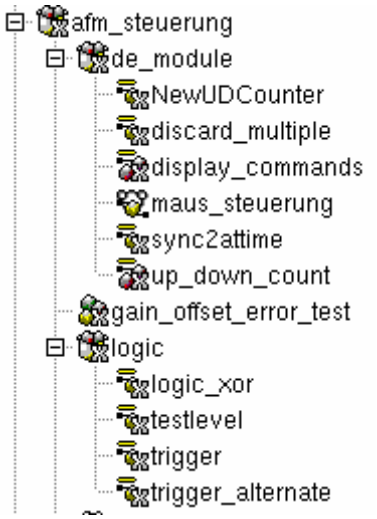
Es existiert immer genau ein Elemententyp für jedes Elementarverzeichnis (dieses enthält keine Unterverzeichnisse mehr). Der Typ des Elements kann ein System, Modul, Primitve oder eine FSM sein. Wie die Verzeichnisstrukturen und somit auch die Projektstrukturen am Beispiel aussehen, sollen die folgenden Abbildungen verdeutlichen. In den wichtigsten Modulen und Primitiven ist, über die Option „online Dokumentation„ direkt in ML-Designer, meist die bei der Entwicklung entstandene Beschreibung zu finden. Einige der Elemente haben für das derzeitige Modell keine Bedeutung mehr. Das lag an der doch recht hohen Anzahl von Versionsneuerungen, die während der Zeit der Studienarbeit vollzogen worden sind.

1. Hierarchieebene



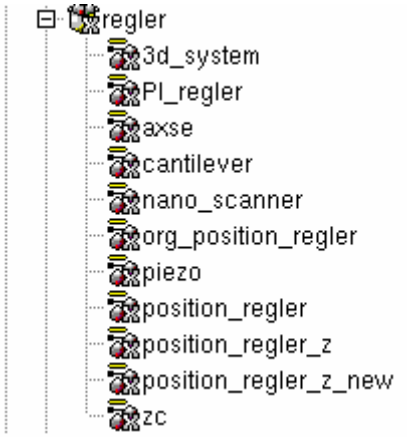
Die Elementarsysteme auf dieser Ebene sind Simulationssysteme der SDF Domain. Die durch ein „+“ gekennzeichneten Verzeichnisse sind Bibliotheken, welche Primitive und Module funktional zusammenfassen. Sie werden unter 2. näher erläutert.
Das System *„gain_offset_error_test“* zeigt, wie Offset- und Verstärkungsfehler eines Sinus Signals mit Hilfe der Primitive „math/offset_gain_correct_mini“ ausgeglichen wird. Zur Anzeige kommt der erkannte Offset- und Gainfehler, sowie korrigierte und unkorrigierte Sinussignal.
Das System *„modell“* enthält das in dieser Arbeit entstandene Gesamtsimulationsmodell.
In „show_error_timing“ wurde zu letzt das zeitliche Verhalten zwischen realer Achsenposition und errechneter untersucht. Das ist noch ein Überbleibsel aus der Zeit, in der das Gesamtmodell noch nicht funktionstüchtig war. (nur im kleinen Modell)
Der *„up_down_test“* wurde zur Validierung des Berechnungsverhaltens von der Achsenposition verwendet. Aus einem Sin/Cos Signal werden incremente erzeugt, diese gezählt und um den Kommaanteil aus der trigonometrischen Berechnung erweitert.

2. Hierarchieebene



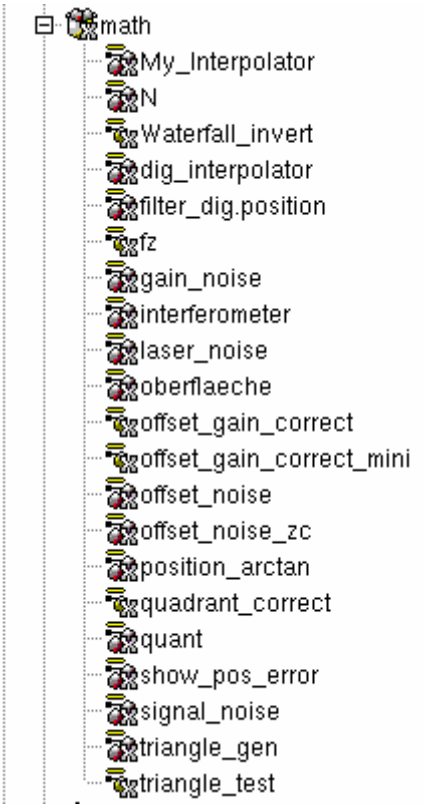
Bibliothek, für Elemente der Domain DE o. FSM
->Erweiterter UpDownCounter, der nur eins Events zählt
->Aufeinanderfolgende gleiche Events werden beseitigt.
Modul für Ausgabe von Steuerkommandos einer *.txt an den Nutzer
->FSM nach [12] (funktioniert nicht)
->Softwaremäßige Realisierung der FSM
->Modul enthält die FSM und den Zähler

->in einer frühen Version gab es das Primitiv nicht in Standard lib
->wie ein Komparator nur mit konstanten Schwellwert
->schaltet genau einmal nach auftreten eines triggerwertes am Input
->toggelt Ausgangsbit in Abhängigkeit der Eingangspulse



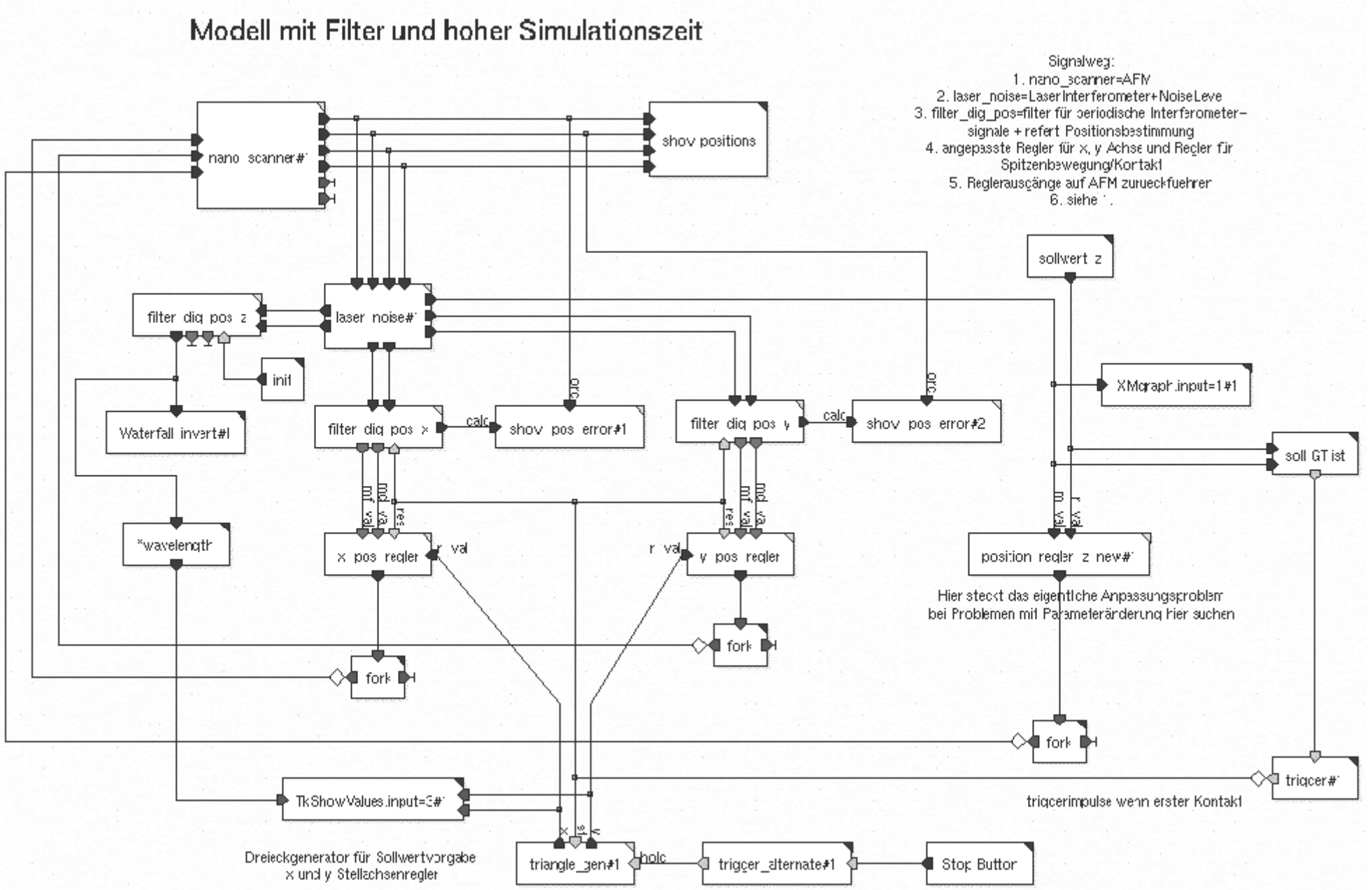
Bibliothek enthält alles, was mit Integratoren zu tun hat (AFM).
->Normierung aller Ein-/Ausgangs Ports für die 3 Achsen
->kleiner einfacher PID Regler
->Berechnung der Differentialgleichung 2. Ordnung.
->Normierung für die Spitze mit Modul zc und fz.
->bildet AFM aus Parametern und Modul cantilever, 3d_system
->ein anderer Weg als arctan für fineposition zu nutzen
->nicht funktionstüchtig, Ansatz Modell eines Piezo Aktors
->aus der Quelle [13] entworfener Regler
->z Regler erhält keine Festkommazahl, deswegen geändert
->geändertes Reglerprinzip + Filter für Intensitätswerte
->Berechnet Dynamik der Spitze über Diff.gleichung 2. Ordnung

Differentialgleichungen 2. Ordnung werden durch zwei Integratoren dargestellt. Dadurch ist es möglich Weg, Geschwindigkeit und Beschleunigung zu berechnen.



->1. Ansatz Sin/Cos Signal in digitale incremente zu formen
->errechnet Kraft die von der Spitze auf Probe rückwirkt
->3-d Anzeigeelement für Probenoberfläche
->eingesetztes und erläutertes Interpolationsprinzip
->Filter + Quad. Zählverfahren + Pos. Berechnung + Korrektur
->Primitive berechnet Interaktionskraft Probe <-> Spitze
->erzeugt Verstärkungsfehler des Laserinterferometersignals
->generiert aus abs. Pos. ein Sin/Cos Signal + Phasenfehler
->Gesamt Lasermodell mit 4 Fehlerquellen+Nutzungselemente
->simulierte Probenoberfläche des Moduls nano_scanner
->korrigiert den Offset-/Verstärkungsfehler in filter_dig.pos
->abgewandelte Mechanismus, der aber nicht besser ist (TEST)
->erzeugt die Offsetfehler für die Paare Sin/Cos Signal
->musste für Intensitätssteuerung der Spitze geändert werden
->berechnet arctan(sin/cos) und skaliert Ergebnis (0..2PI)
->korrigiert Quadrantenunterschied zwischen dig und fine Pos.
->Element zum quantisieren von Signalen
->zeigt in Bargraph Tck/Tk den gemachten Positionsfehler
->sollte das hochfrequente Signalrauschen erzeugen
->Sollwerte für Regler in Form diskreter Stufen-/Dreiecksfkt.
->prüft erreichen der Extrema der Fkt. für Richtungswechsel

Abkürzungen:
Pos = Position
Quad = Quadrant
dig = digital
fine = genau
abs = absolut
Diff = Differential


Modell mit Filter und hoher Simulationszeit
Signalweg:
1. nano_scanner=AFM
2. laser_noise=LaserInterferometer+NoiseLeve
3. filter_dig_pos=filter für periodische Interferometer-signale + refert Positionsbestimmung
4. angepasste Regler für x, y Achse und Regler für Spitzenbewegung/Kontakt
5. Reglerausgänge auf AFM zurueckfuehrer
6. siehe '.
nano_scanner#'
show positions
sollwert z
filter_dig_pos_z
laser_noise#'
init
XMgraph.input=1#1
Waterfall invert#1
filter_dig_pos_x
calc
show_pos_error#1
filter_dig_pos_y
calc
show_pos_error#2
soll GT ist
*wavelength
x_pos_regler
r_val
y_pos_regler
r_val
position_regler_z_new#'
Hier steckt das eigentliche Anpassungsproblem
bei Problemen mit Parameteränderung hier suchen
fork
fork
fork
trigger#'
triggerimpulse wenn erster Kontakt
TkShowValues.input=3#'
Dreieckgenerator für Sollwertvorgabe
x und y Stellachsenregler
triangle_gen#1
hold
trigger_alternate#1
Stop Buttor