Mobiles Labor auf Open Source Basis für die Ausbildung mechatronischer Antriebssysteme

Rolf Roskam[1]

[1]Ostfalia Hochschule für angewandte Wissenschaften, Salzdahlumer Straße 46-48, 38302 Wolfenbüttel, Deutschland; r.roskam@*ostfalia.de*

Abstract. Bedingt durch die Corona-Pandemie, in der die Präsenslabore für die Ausbildung nicht, oder nur sehr eingeschränkt nutzbar sind, wird hier eine mikrocontrollerbasierte Alternative zur häufig verwendeten Werkzeugkette Matlab/Simulink/Controldesk in Verbindung mit der Hardware DS1104 für die Entwicklung mechatronischer Antriebssysteme vorgestellt. Diese Lösung basiert auf Open Source Ansätzen und eines speziell entwickelten mobilen Labors, welches die Studierenden ausleihen und zu Hause aufbauen und durchführen können. Am Beispiel eines rotatorischen Bedienelementes, angetrieben über einen Schrittmotor zur Abbildung einer mechatronischen Raste, wird zunächst die Modellbildung und anschließend die feldorientierte Regelung dargestellt. Am Ende können die Studierenden das Ergebnis ihrer Entwicklung haptisch erleben.

Einleitung

In der ingenieurwissenschaftlichen Ausbildung von Studierenden sind Labore, in der die theoretisch erworbenen Kenntnisse praxisnah angewendet werden, elementarer Bestandteil.

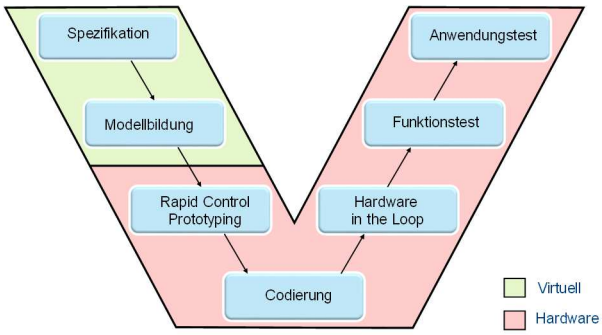

Abbildung 1: Entwicklungsprozess mechatronischer Antriebssysteme, unterteilt in virtuell und Hardware

Die Entwicklung mechatronischer Antriebssysteme folgt häufig einem V-Modell [1]. Die Umsetzung des V-Modells im Labor kann in virtuelle und hardwarenahe Elemente unterteilt werden (Abbildung 1). Die Phasen Spezifikation und Modellbildung können in der Ausbildung problemlos virtuell und damit online durchgeführt werden. Hierbei kommt entweder das kommerzielle Werkzeug Matlab/Simulink oder die Open Source Variante Scilab/XCOS zum Einsatz. Für die weiteren Phasen ist jedoch ein Hardware-Aufbau erforderlich, welcher üblicherweise über ein Labor zur Verfügung gestellt wird. An der Ostfalia steht hierzu u.a. ein Pool Raum mit 13 Arbeitsplätzen zur Verfügung. Jeder Arbeitsplatz ist ausgestattet mit einem PC, dem Werkzeug Matlab/Simulink der Firma Mathworks und einer RCP-Hardware DS1104 mit dem Werkzeug ControlDesk der Firma dSpace. Da die RCP-Hardware nur eine Signalschnittstelle zur Verfügung stellt, wurde zusätzlich eine Leistungs- und Sensorelektronik aufgebaut (Abbildung 2).

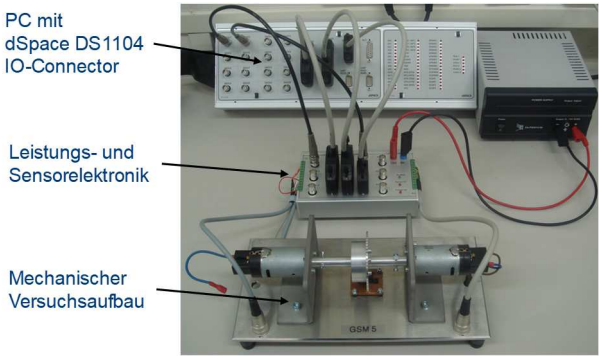

Abbildung 2: RCP-Laboraufbau DC-Motor mit Leistungs- und Sensorelektronik zum Anschluss an DS1104

Die technischen Daten des Interface sind in Tabelle 1 aufgeführt. Hierbei ist anzumerken, dass die RCP-Hardware DS1104 nicht in der Lage ist, ein winkelabhängiges Einspritzsignal in Echtzeit zu erzeugen [2].

Eingänge	Ausgänge
1x 0..5V oder Potentiometer	1x 0...5V
1x Drehzahl (Zahnzeit) (Timer-Capture)	1x Einspritzsignal (Timer-Compare)
1x Encoder mit 5V Versorgung	2x PWM-Vollbrücke mit bis zu 100KHz, 12V, 2,5A mit bipolarer Strommessung

Table 1: Kenndaten der Leistungs- und Sensorelektronik

In dem Modul „Entwicklung mechatronischer Antriebssysteme“, welches vom Autor an der Ostfalia im

Bachelor-Studiengang Maschinenbau in der Vertiefungsrichtung Mechatronik und Digitalisierung im Umfang von 8 ECTS angeboten wird, erlernen die Studierenden anhand eines Gleichstrom- und eines, als Servomotor betriebenen Schrittmotors den Entwicklungsprozess. Als Prüfungsleistung müssen sie am Ende des Semesters an einem neuen Versuchsaufbau im Rahmen einer Projektarbeit die Phasen Spezifikation, Modellbildung und RCP-Implementierung einer Mehrgrößenregelung selbstständig durchführen. Hierzu stehen weitere Versuchsaufbauten zur Verfügung.

Bedingt durch die Corona-Pandemie, in der die Präsenslabore gerade für größere Kohorten von Studierenden nur schwer genutzt werden können, fehlt ein zentrales Element in der Lehre. Daher wurde im Wintersemester 20/21 ein mobiles Labor aufgebaut, welches nachfolgend vorgestellt wird.

1 DS1104 und Alternative

Nach Vorstellung der RCP-Hardware DS1104 im Jahr 2002 [3] wurde diese in der Lehre und Forschung immer häufiger eingesetzt. So stieg die Anzahl der Veröffentlichung der IEEE-Datenbank, in der der Einsatz der DS1104 explizit erwähnt wurde, kontinuierlich an (Abbildung 3). Bereits 3 Jahre nach der Vorstellung der Hardware wurde diese auch an der Ostfalia in der Lehre und Forschung eingesetzt. Seit 2008 steht das in der Einleitung beschriebene Labor zur Verfügung.

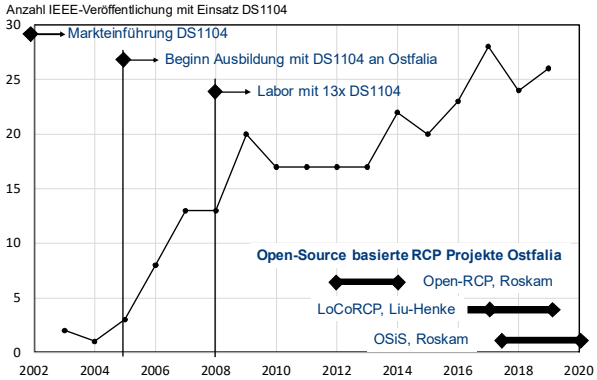

Abbildung 3: Historie Einsatz DS1104 und Projekte Ostfalia

Gerade in Zusammenarbeit mit kleinen und mittleren Industrieunternehmen musste in der Entwicklung festgestellt werden, dass die kommerzielle RCP-Hard- und Software häufig zu kostenintensiv ist. Daher begannen Entwicklungen, die auf Basis eines Open Source Ansatzes zu einer deutlich kostengünstigeren Lösung führen [4], [5]. So wurden auch am Institut für Mechatronik der Ostfalia in der Vergangenheit Forschungsprojekte für Anwendungen der Hydraulik [6], Batterietechnik [7] und Sensorcluster [8] erfolgreich bearbeitet.

Insbesondere basierend auf den Vorarbeiten in [8] konnte innerhalb von 4 Monaten eine Alternative zum bisherigen Labor entwickelt werden. Diese Entwicklung nutzt einen Dual Core Mikrocontroller STM32H745.

	DS 1104	STM32H745
Vorstellung	2002	2019
Prozessor	Zwei Prozessoren 1x PPC603 1x TMS320F240	Dual Core STM32H745 1x Cortex M7 1x Cortex M4
Technologie	500nm	40nm
Takt	250 / 20 MHz	480 / 240 MHz
RAM/ROM	32M / 8M external	1M / 2M on chip
Floating Point Unit	1x Double precision 1x Keine FPU	1x Double precision 1x Single Precision
Timer	2x Encoder 10x PWM 4x Capture	6x Encoder 12x PWM 8x Capture/Compare
A/D	4 multiplexed, 16bit 4 parallel, 12 bit	20 multiplexed, 16bit 3 parallel, 16 bit
Interface	PCI, U(S)ART	USB, U(S)ART 2x CAN, Ethernet, SPI, I2C, SDMMC
Kosten	ca. 100	1
Werkzeuge	herstellerspezifisch	Open Source

Tabelle 2: Vergleich DS1104 [9] und STM32H745 [10]

Ein Vergleich der Hardware DS1104 mit dem Mikrocontroller zeigt, dass die dynamische Entwicklung in der Mikroelektronik der letzten 20 Jahre zu Lösungen geführt hat, die bessere Leistungsdaten (doppelte Taktrate, höhere Schnittstellenanzahl) zu erheblich geringeren Kosten realisiert. Da beim Einsatz einer DS1104 eine Leistungs- und Sensorelektronik als Interface aufgebaut werden muss, kann diese auch mit geringem Aufwand um einen Mikrocontroller erweitert werden.

Das Ergebnis der Hardware-Entwicklung ist das in Abbildung 4 dargestellte RESClab (Rapid Embedded Systems Control Laboratory). Das RESClab nutzt ein kostengünstiges Discovery-Board des Mikrocontrollers, welches zusätzlich über ein Touch-Display verfügt. Somit kann in der Weiterentwicklung zukünftig auch ein modernes HMI umgesetzt werden. Die Leistungs- und Sensorelek-tronik wurde in THT (Through-hole-technology) realisiert, so dass der Aufbau von Studierenden

selbst erstellt werden kann. Im Wintersemester 20/21 wurden von den Teilnehmenden des Moduls „Mechatronische Systementwicklung“ in 3 Doppelstunden zunächst 25 RESClabs zusammengebaut und in Betrieb genommen.

Verwendung Discovery-Board STM32H745
mit Ethernet, USB, 2x CAN-Bus und zusätzlich
2x64MByte SPI Flash, 16MByte SDRAM, 4GByte eMMC
IO-Platinen (2 Etagen) im TH-Aufbau
- 2x Vollbrücke 12V/2,5A
  mit integrierter biploarer Strommessung
- 1x Drehzahlmessung (Capture-Timer)
- 1x externer Encoder
- 1x analoger Eingang 0…5V
- 1x DA-Ausgang 0…5V
- 1x Injektor-Ausgang (Compare-Timer)
HMI:
- 3+1 Taster
- 4+3 LEDs
- 4,3" Touch-Display 480x272
- Internes Poti
- Interner Encoder

Abbildung 4: RESClab mit technischen Daten

2 RCP-Entwicklungswerkzeuge

Neben der Hardware sind durchgängige Software-Werkzeuge elementarer Bestandteil des mechatronischen Entwicklungsprozesses. Als Quasi-Standard hat sich in den letzten Jahren die Software Matlab/Simulink in der modellbasierten, mechatronischen Systementwicklung etabliert. Hiermit können Modelle und Regelalgorithmen blockorientiert entwickelt und automatisch in Quellcode und damit ausführbaren Programmen umgesetzt werden, die in Echtzeit auf einer speziellen Hardware ablaufen können. Um darüber hinaus während der Programmausführung Parameter ändern und Messgrößen aufzeichnen und analysieren zu können, wird für die DS1104 die Software ControlDesk der Firma dSpace eingesetzt.

Der Einsatz der durchgängigen, modellbasierten Werkzeugkette Matlab/Simulink/ControlDesk ist aus Kostengründen für den Einsatz in mobilen Laboren nicht geeignet. Daher wurde auf Basis des Forschungsprojektes OSiS [8] eine separate Werkzeugkette Scicos/XCOS/OSiS-IDE entwickelt, die ausschließlich mit Open Source Komponenten auskommt. Nach dem Start des Programms Scilab ist eine Auswahl zur gewünschten Hardware zu treffen, vergleichbar zur Hardware-Auswahl beim Start von Matlab in Verbindung mit RCP-Hardware der Firma dSpace. Anschließend stehen in Scilab/XCOS spezielle Hardware-Blöcke für das RESClab zur Verfügung. Im Hauptmenü kann nach Fertigstellung des Algorithmus in XCOS die automatische Code-Generierung über den OSiS-Codegenerator gestartet werden. Das Compilieren, Linken sowie der USB-Download des erstellten Programms erfolgt über eine selbstentwickelte, integrierte Entwicklungsumgebung (OSiS-IDE). Hiermit können auch Parameter online geändert sowie Messgrößen erfasst und analysiert werden.

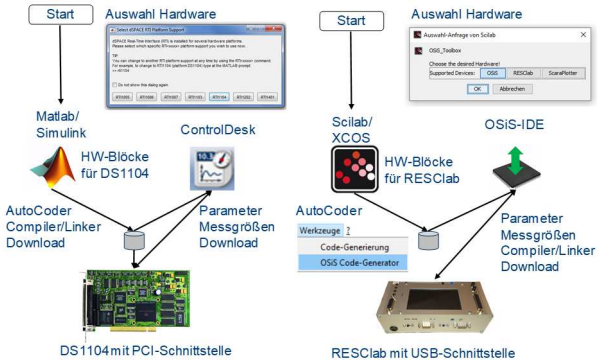

Abbildung 5: RCP-Entwicklungswerkzeuge im Vergleich

Der Funktionsumfang dieser Werkzeugkette ist deutlich geringer als bei der kommerziellen Variante. Allerdings ist die Funktionalität in den meisten Fällen und insbesondere in der Ausbildung des mechatronischen Entwicklungsprozesses vollkommen ausreichend, wie im nachfolgenden Kapitel gezeigt wird. Vorteilhaft ist dabei der, im Vergleich zur kommerziellen Werkzeugkette Matlab/Simulink/dSpace, geringere Einarbeitungsaufwand.

3 Einsatz RESClab

Nach Fertigstellung der Hard- und Software wurden in der zweiten Hälfte des Wintersemesters 20/21 die mobilen Labore verteilt. Das mobile Labor wird in einer Stapelbox (Größe 400x300x186mm) transportiert und umfasst das RESClab, unterschiedliche Versuchsaufbauten, Verbindungskabel, CAN-Bus mit Steckernetzteil 12V/5A sowie CAN-USB-Interface (Abbildung 6).

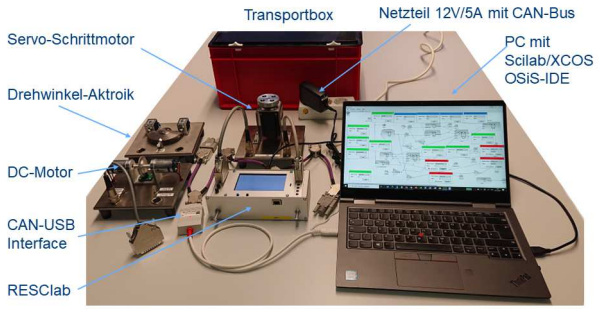

Abbildung 6: Mobiles Labor

Um das mobile Labor nutzen zu können, müssen die Studierenden einen PC oder Laptop mit Windows 7 oder 10 Betriebssystem und USB-Schnittstelle zur Verfügung stellen. Über einen Ausleihvertrag verpflichten sich die Studierenden zur Rückgabe der Geräte bis zum Ende des

Semesters.

Am Beispiel des Versuchsaufbaus Servo-Schrittmotor soll der Einsatz exemplarisch dargestellt werden. In der Spezifikation der Projektaufgabe wird eine Umsetzung eines rotatorischen Bedienelementes definiert, welches an einer einstellbaren Position über eine Raste verfügen soll (Abbildung 7).

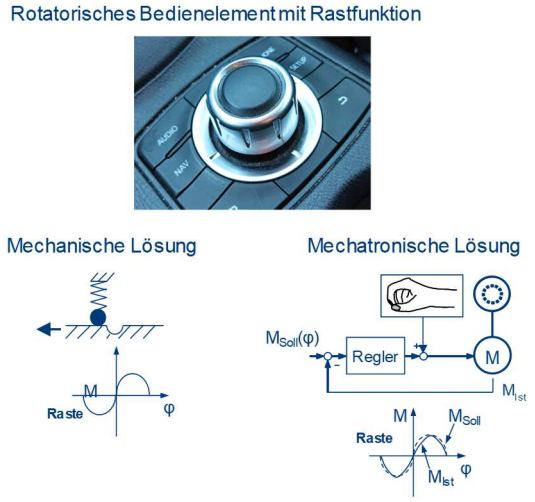

Abbildung 7: Rotatorisches Bedienelement mit Raste

Rastfunktionen werden mechanisch über eine federvorgespannte Kugel realisiert, die an der gewünschten Position in einer Mulde einrastet. Um das Bedienelement aus der Raste zu bewegen, muss ein definiertes Moment überwunden werden. Diese Rastfunktion kann über einen Schrittmotor als mechatronische Raste umgesetzt werden [11].

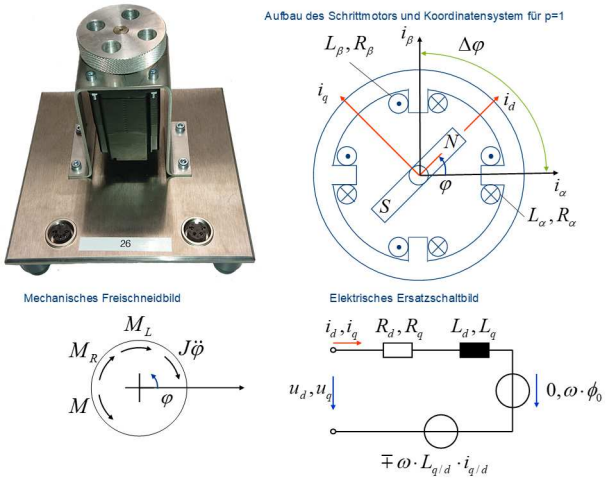

Abbildung 8: 2-Phasen Schrittmotor, Koordinatensystem und elektromechanische Ersatzschaltbilder

Der Schrittmotor ist dabei als Servoantrieb anzusteuern, der winkelabhängig ein definiertes Sollmoment ausregelt, welches der mechanischen Raste entspricht. Der Vorteil einer mechatronischen Realisierung ist die freie Konfiguration der Rastposition und Haptik sowie auch die Umsetzung weiterer haptischer Rückmeldungen (Anschlag, Vibration, Reibung, usw.).

Nach der Spezifikation erfolgt die Modellbildung des Schrittmotors (Abbildung 8). Hierbei werden Reibmomente und Temperatureinflüsse vernachlässigt. Die mechanische Bewegungsgleichung des Produktes der Winkelbeschleunigung $\ddot{\varphi}$ und des Gesamtmassenträgheitsmomentes J des Rotors und der Drehscheibe ergibt sich aus der Summe der Momente:

$$J \cdot \ddot{\varphi} = M - M_L - M_R \,. \tag{1}$$

Hierbei ist M das vom Schrittmotor erzeugte Moment, M_L das durch den Bediener wirkende Lastmoment und M_R das interne Rastmoment des Schrittmotors, welches über eine Sinusfunktion mit einer Amplitude M_{R0} winkelabhängig vereinfacht nachgebildet wird.

$$M_R = M_{R0} \cdot \sin(4 \cdot \vartheta) \tag{2}$$

Dabei ist zu berücksichtigen, dass der elektrische Winkel ϑ abhängig ist von der Polpaarzahl p des Motors und der aktuellen Winkelposition des Rotors φ.

$$\vartheta = p \cdot \varphi \tag{3}$$

Schrittmotoren besitzen üblicherweise eine Polpaarzahl von $p=50$, so dass sich im Schrittbetrieb als kleinster Winkelschritt ergibt:

$$\Delta\varphi = 360° / (4 \cdot p) = 1{,}8° \tag{4}$$

Das Motormoment M wird maßgeblich bestimmt durch den Strom $\boldsymbol{i_q}$ und der Motorkonstante ϕ_0:

$$M = p \cdot \left(\phi_0 \cdot i_q + \left(L_d - L_q\right) \cdot i_d \cdot i_q\right) \tag{5}$$

Der Strom $\boldsymbol{i_q}$ ist der, in dem rotierenden d-q-Koordinatensystems quer zum Rotormagnetfeld wirkenden Strom, der mit Hilfe der Park-Transformation (6) aus den, in den beiden Spulen des Schrittmotors fließende Strömen i_α und i_β berechnet werden kann. Als Störgröße in der Momentengleichung wirkt die Differenz der, in das d-q-Koordinatensystem transformierten Induktivitäten L_d und L_q der beiden Spulen des Schrittmotors multipliziert mit den beiden Strömen i_d und i_q.

Die Modellbildung der elektrischen Kreise erfolgt im rotierenden d-q-Koordinatensystem des Rotors unter Verwendung der Park- und inversen Park-Transformation (7) in Bezug zum ortsfesten α-β-Koordinatensystem.

$$\begin{pmatrix} i_d \\ i_q \end{pmatrix} = \begin{pmatrix} \cos\varphi & \sin\varphi \\ -\sin\varphi & \cos\varphi \end{pmatrix} \cdot \begin{pmatrix} i_\alpha \\ i_\beta \end{pmatrix} \quad (6)$$

$$\begin{pmatrix} i_\alpha \\ i_\beta \end{pmatrix} = \begin{pmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{pmatrix} \cdot \begin{pmatrix} i_d \\ i_q \end{pmatrix} \quad (7)$$

Ein Maschenumlauf ergibt die nachfolgenden Differenzialgleichungen:

$$u_d = R_d \cdot i_d + L_d \cdot \frac{di}{dt} - \omega \cdot L_q \cdot i_q \quad (8)$$

$$u_q = R_q \cdot i_q + L_q \cdot \frac{di_q}{dt} + \omega \cdot L_d \cdot i_d + \omega \cdot \phi_0 \quad (9)$$

Die Spannungen u_d und u_q sind die, in das d-q-Koordinatensystem transformierten Spannungen, die an den beiden Spulen des Schrittmotors angelegt werden. Die Berechnung erfolgt ebenfalls über die Park-Transformation. Neben dem PT1-Verhalten der beiden Stromkreise, welches über den jeweiligen Widerstand und Induktivität definiert ist, gibt es einen zusätzlichen Einfluss durch die jeweils andere Induktivität in Abhängigkeit der elektrischen Drehzahl ω und des, in dieser Induktivität fließenden Stroms. Die elektrische Drehzahl ω wird aus der Polpaarzahl und der mechanischen Drehzahl $\dot{\varphi}$ berechnet zu:

$$\omega = p \cdot \dot{\varphi} \quad (10)$$

In der q-Achse der Gleichung des Maschenumlaufs (9) muss zusätzlich die induzierte Spannung, die sich aus dem Produkt der elektrischen Drehzahl ω und der Motorkonstante ϕ_0 ergibt, berücksichtigt werden.

Typ: RTELLIGENT 42A03EC	Wert	Einheit
Haltemoment M_H	0,3	Nm
Schrittwinkel / Polpaarzahl p	1,8 / 50	° / -
Nennstrom I_N	2	A
Massenträgheit Rotor J_M	77	gcm²
Schrittanzahl Encoder k	1000	-

Tabelle 3: Technische Daten des Schrittmotors [12]

Einige Parameter des Simulationsmodells können dem Datenblatt des Herstellers entnommen werden (Tabelle 3). Im Normbetrieb ist der Strom i_d Null, so dass sich die Motorkonstante ϕ_0 anhand der technischen Daten des Motors aus der Momentengleichung (5) berechnen lässt zu:

$$\phi_0 = M_H / (p \cdot I_N) = 3mNm / A \quad (11)$$

Zusätzlich erfolgen Messungen über das mobile Labor. Der Strom wird dabei über ein Anti-Aliasing-Filter mit einer Zeitkonstante T_{AA} von 0,12ms mit einer Abtastfrequenz von 5kHz gemessen. Der Winkel wird über einen Encoder mit einer 4-fach Auswertung und somit einer Auflösung von 4000 Inkrementen erfasst

Nach einmaliger Ausrichtung des Rotors durch Aktivierung einer Spule in der α-Achse, bei der gleichzeitig der gemessene Winkel des Encoders zu Null gesetzt wird, kann bei anschließend festgestelltem Rotor ($\omega=0$) unmittelbar die Sprungantwort des Stroms in der d-Achse (erneute Aktivierung der Spule in der α-Achse) und q-Achse (Aktivierung der Spule in der β-Achse) ermittelt werden (Abbildung 9).

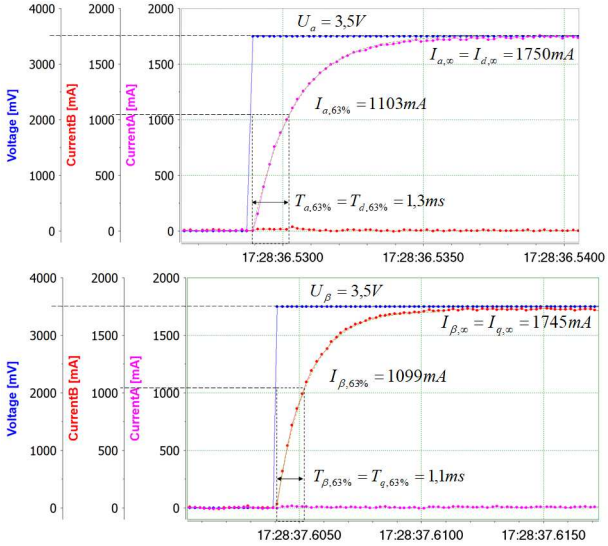

Abbildung 9: Sprungantwort Strom bei festgestelltem Rotor

Anschließend werden die Widerstände und Induktivitäten ermittelt. Die Sprungantwort des Stroms entspricht einem PT1-Verhalten, so dass die charakteristische Zeitkonstante bei Erreichen von 63% des Endwertes $T_{d/q,63\%}$ abgelesen werden kann. Um den Verzögerungseinfluss durch die Messung bei der Ermittlung der Induktivität zu berücksichtigen, wird die Zeitkonstante des Anti-Aliasing Filters T_{AA} vereinfacht subtrahiert:

$$R_{d/q} = U_{d/q} / I_{d/q,\infty} \quad (12)$$

$$L_{d/q} = \left(T_{d/q,63\%} - T_{AA}\right) \cdot R_{d/q} \quad (13)$$

Beim Massenträgheitsmoment muss neben dem Rotor die montierte Scheibe berücksichtigt werden. Dieses

wurde mit Hilfe der CAD-Daten zu $J_S=120gcm^2$ bestimmt. Ein Vergleich der Dynamik im Schrittbetrieb in Simulation und Messung zeigt eine gute Übereinstimmung, wenn das Massenträgheitsmoment 10% größer angesetzt wird (Abbildung 10),

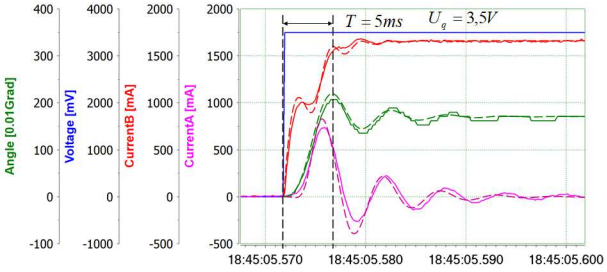

Abbildung 10: Validierung Schrittbetrieb (Simulation gestrichelt) mit angepasstem Massenträgheitsmoment

Zur Bestimmung des motorinternen Rastmomentes wird der Schrittmotor mit cosinus- und sinusförmigen Spannungen für U_α und U_β mit einer Amplitude von 3,5V und einer Frequenz ω von 50Hz angesteuert. Die Vorgabe einer Amplitude M_{R0} des Rastmomentes von 2,4% des Haltemoments M_H führt dabei zu einer guten Übereinstimmung des Drehzahlverlaufs in Simulation und Praxis (Abbildung 11). Die Phasenverschiebung im Drehzahlverlauf ist bedingt durch die Zahnzeitmessung für nur einen Kanal des Encodersignals sowie eine Plausibilitätsüberprüfung (Erkennung einer Referenzmarke).

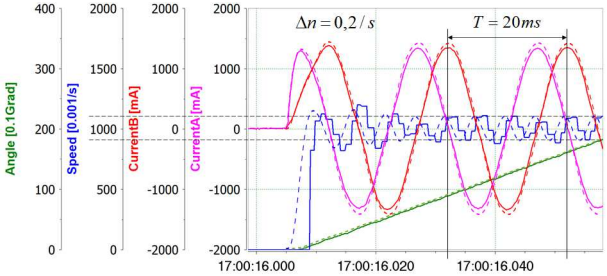

Abbildung 11: Validierung Sinusbetrieb (Simulation gestrichelt) mit angepasstem Rastmoment

Eine Übersicht der durch Messungen und mit Hilfe der Validierung bestimmten Parameter zeigt Tabelle 4.

Simulationsparameter	Wert	Einheit
Widerstand R_d	2,0	Ohm
Widerstand R_q	2,0	Ohm
Induktivität L_d	2,4	mH
Induktivität L_q	2,0	mH
Rastmoment M_{R0}	7,2	mNm
Massenträgheitsmoment J	217	gcm²

Tabelle 4: Durch Messung und Validierung ermittelte Parameter des Schrittmotors

Nach Erstellung und Validierung der Simulation erfolgt die Auslegung des PI-Stromregelkreise sowohl für die d- als auch für die q-Achse (Abbildung 12) nach der Methode des Betragsoptimums einer PT2-Regelstrecke [13].

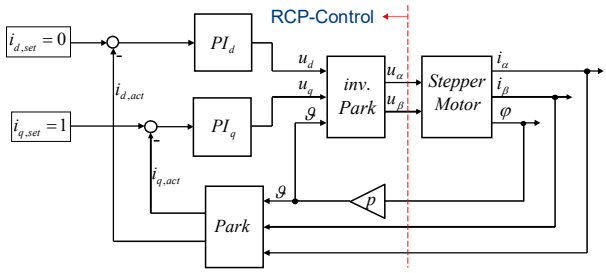

Abbildung 12: Stromregelkreise für d- und q-Achse

Die PT2-Glieder ergeben sich aus den beiden Zeitkonstanten der LR-Netzwerke sowie des Anti-Aliasing-Filters in der d- und q-Achse. Die Gesamtverstärkung des PT2-Gliedes wird über dem Widerstand des Netzwerkes definiert. Bei der Reglerauslegung wird die größte Zeitkonstante der Strecke (hier das LR-Netzwerk) kompensiert und für das PT2-Ersatzsystem des geschlossenen Regelkreises eine Dämpfung von 1 vorgegeben. Somit ergeben sich die Regelparameter in nichtnormierter Form zu:

Regelparameter	Wert	Einheit
d-Kreis Proportional $K_{p,d}$	4,9167	Ohm
d-Kreis Integral $K_{i,d}$	4167	Ohm/s
q-Kreis Proportional $K_{p,q}$	4,0950	Ohm
q-Kreis Integral $K_{i,q}$	4179	Ohm/s

Tabelle 5: PI-Regelfaktoren gemäß Betragsoptimum

In der Lehre wird im Modul „Mechatronische Systementwicklung“ bei der Auslegung der Regelung zusätzlich auf eine drehzahlabhängige Vorsteuerung zur Kompensation der Induktionsstöreinflüsse des PT1-Gliedes im Stromkreis, auf die Limitierung der Regelung durch Abtastung und Diskretisierung der Aktor- und Sensorgrößen, sowie auf die Begrenzung der Stellgröße und den dadurch erforderlichen Anti-Wind-Up Regler eingegangen. Die Analyse erfolgt dabei zunächst in der Simulation und anschließend mit Hilfe des mobilen Labors am realen Versuchsaufbau.

Hier wird im Folgenden lediglich die Stromregelung ohne Vorsteuerung dargestellt. Mit den analytisch berechneten Regelparametern wird ein Sollwertsprung auf 1A in der d-Achse sowohl in der Simulation als auch in der Praxis nach ca. 1ms ohne Überschwinger ausgeregelt (Abbildung 13). Bei Vorgabe des gleichen Sollwertsprungs in der q-Achse verbleibt eine Regelabweichung

(Abbildung 14). Diese kann mit einer Vorsteuerung der drehzahlabhängigen Induktionsstöreinflüsse des PT1-Gliedes im Stromkreis vollständig ausgeglichen werden, setzt aber eine zusätzliche Messtechnik oder einen Beobachtungsalgorithmus für die Drehzahl voraus. Da die Drehzahl in der Anwendung als mechatronische Raste bei manueller Bedienung üblicherweise klein ist, wird die verbleibende Regelabweichung deutlich kleiner ausfallen, so dass eine Vorsteuerung entfallen kann.

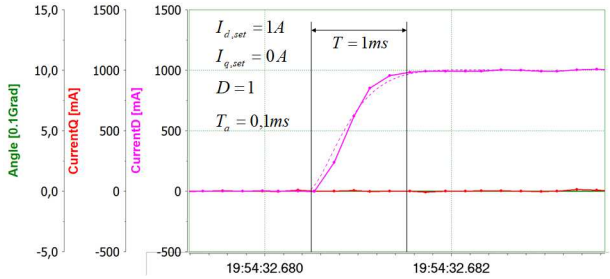

Abbildung 13: Ergebnisse der Stromregelung ind der d-Achse ohne Bewegung (Simulation gestrichelt)

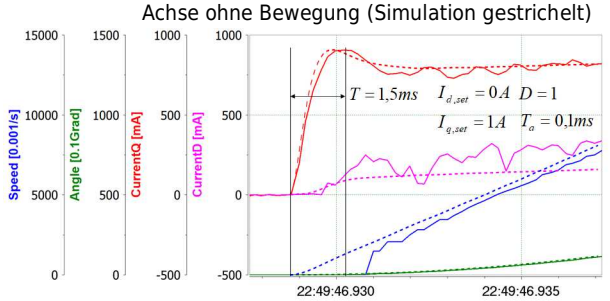

Abbildung 14: Ergebnisse der Stromregelung in der q-Achse mit Bewegung (Simulation gestrichelt)

Um eine Rastfunktion abzubilden, muss eine winkelabhängige Momentvorgabe realisiert werden. Hierzu können unterschiedliche Verläufe definiert werden, die die Qualität der haptische Rückmeldung bestimmen. Eine gute Haptik wird gemäß [11] durch eine Sinusfunktion des Sollmomentes erreicht (Abbildung 15).

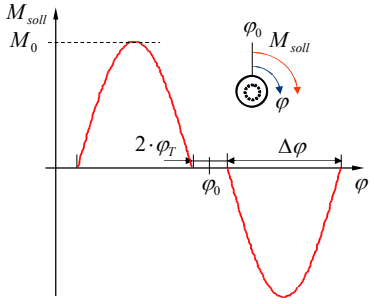

Abbildung 15: Stromsollwert einer Raste mit Nullpunkt

Da das Moment näherungsweise proportional zum Strom i_q ist, ist für die haptische Rückmeldung eine winkelabhängige Sollwertvorgabe des Stromes ausreichend. Hierzu wird eine sinusförmige Stromsollwertvorgabe mit der Amplitude I_0 definiert. Der Stromsollwert ergibt sich zu:

$$
\begin{aligned}
&\textit{für} \quad \varphi_0 + \varphi_T < \varphi < \varphi_0 + \varphi_T + \Delta\varphi: \\
&\qquad i_{q,soll} = I_0 \cdot \sin\left(\frac{\pi}{\Delta\varphi} \cdot \left(\varphi - \left(\varphi_0 + \varphi_T\right)\right)\right) \\
&\textit{für} \quad \varphi_0 - \varphi_T > \varphi > \varphi_0 - \varphi_T - \Delta\varphi: \\
&\qquad {}_{q,soll} = I_0 \cdot \sin\left(\frac{\pi}{\Delta\varphi} \cdot \left(\varphi - \left(\varphi_0 - \varphi_T\right)\right)\right) \\
&\textit{sonst}: \quad i_{q,soll} = 0
\end{aligned}
\tag{14}
$$

Hierbei ist φ_0 die gewünschte Rastposition, φ_T eine symmetrische Breite um die Rastposition, in der kein Moment wirkt und $\Delta\varphi$ die Breite, auf der eine positive Halbwelle der Sinusfunktion wirken soll. Der Strom i_d wird zu Null vorgegeben.

Die Umsetzung des Stromsollwertgenerators erfolgt in einem Superblock in XCOS mit Parametern, die über die OSiS-IDE online einstellbar sind (Abbildung 16).

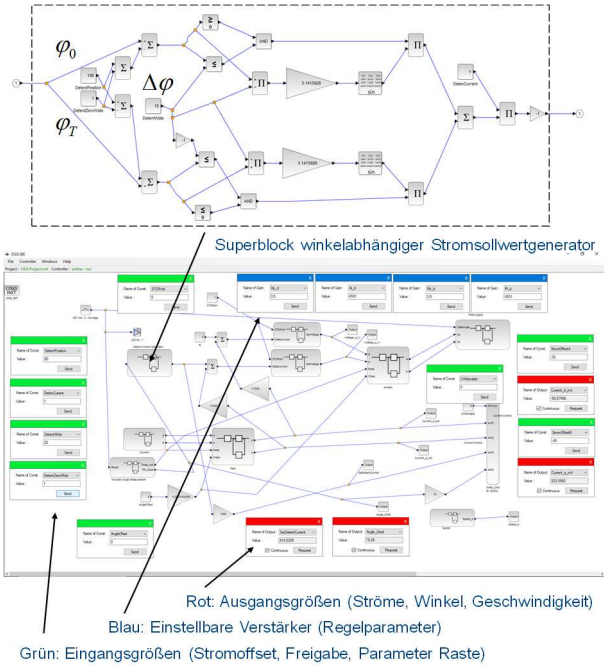

Abbildung 16: Stromsollwertgenerator und OSiS-IDE

Abschließend kann die haptische Rückmeldung der Raste, die durch die Regelung erzeugt wird, vom Bediener mit Hilfe des mobilen Labors „gefühlt“ werden. Mit Hilfe der einstellbaren Parameter ist eine Optimierung der Haptik möglich. Exemplarisch wurden die in der Tabelle 6 angegebenen Parameter gewählt. Gerade durch das reale Erleben der Regelung einer mechatronischen Raste in der Praxis zeigen sich sehr deutlich die Grenzen der virtuellen Ausbildung und die Vorteile eines mobilen Labors. Auch in dieser Veröffentlichung kann nur das Messergebnis der Prozessgrößen beim Durchfahren der

mechatronischen Raste dargestellt (Abbildung 17), nicht jedoch die Haptik erlebt werden.

Parameter der Raste	Wert	Einheit
Amplitude Raststrom I_0	1	A
Rastposition φ_0	90	Grad
Rastbreite $\Delta\varphi$	15	Grad
Breite Nullpunkt φ_T	1	Grad

Tabelle 6: Parameter der Raste

Durch die manuelle Bedienung des Stellelementes ist der Winkelverlauf nichtlinear ansteigend. Deutlich ist zu erkennen, dass die Regelung des Stromes jederzeit in der Lage ist, den Istwert ohne Abweichungen dem Sollwert folgen zu lassen.

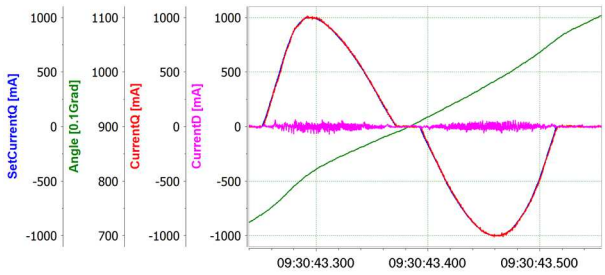

Abbildung 17: Prozessgrößen beim manuellen Durchfahren einer mechatronischen Raste

Die für die Anwendung mechatronische Raste von der OSiS-IDE berechnete Auslastung des Prozessorkerns M7 beträgt, bei einer Abtastzeit von 0,2ms ohne den Einsatz von Optimierungen, maximal 47%. Aufgrund der begrenzten Entwicklungszeit wurde der Prozessorkern M4 noch nicht genutzt. In der Weiterentwicklung des RESClabs ist somit noch eine deutliche Leistungssteigerung möglich. Der Speicherplatzbedarf des automatisch erzeugten Programms umfasst ca. 92KByte inkl. IO-Handling sowie CAN-Bus und USB-Kommunikation, so dass der interne Speicher des Mikrocontrollers von 2MByte vollkommen ausreichend ist.

4 Zusammenfassung

In dieser Veröffentlichung wurde ein mobiles Labor auf Open Source Basis vorgestellt, welches auf einem neuen Mikrocontroller basiert. Aufgrund der dynamischen Entwicklung in der Mikroelektronik ist dieses Labor von den Leistungsdaten vergleichbar zu der, in der Entwicklung elektrischer Antriebssysteme häufig eingesetzten Hardware DS1104 der Firma dSpace mit den Software-Werkzeugen Matlab/Simulink/ControlDesk.

Anhand der Projektaufgabe „Mechatronische Raste“ konnten sowohl Messungen zur Parameterermittlung wie auch die Umsetzung einer feldorientierten Stromregelung dargestellt werden. Das Ergebnis der mechatronischen Raste kann durch eine Messung validiert werden, jedoch nur durch das reale Fühlen der Haptik mit Hilfe des mobilen Labors von den Studierenden erlebt werden.

References

[1] Verein Deutscher Ingenieure (Hrsg.). VDI 2206 - Entwicklungsmethodik für mechatronische Systeme. Berlin, Beuth Verlag GmbH 2004

[2] Roskam, R. Mechatronic Evaluation Board for Use in Education of the Mechatronic Engineering Process. 11th International Workshop on Research and Education in Mechatronics, Ostrava, 2010

[3] Pöhlmann, T. Antriebsregler leicht gemacht. Embedded engineering, München, Hanser 2002

[4] Bucher, R.; Balemi, S. Scilab/Scicos and Linux RTAI - A unified approach. IEEE Conference on Control Applications, Toronto, 2006

[5] Grzegorz et al. Rapid Control Prototyping with Scilab/Scicos/RTAI for PC-based and ARM-based Platforms. Proceedings of the IMCSIT Volume 3, Wisla, Poland, 2008

[6] Dobkowitz, D. et al. Serientauglicher Hydraulikcontroller auf Open Source Basis für die modellbasierte Entwicklung hydraulischer Systeme. 8. Kolloquium Mobilhydraulik, Braunschweig 2014

[7] Jacobitz, S; Liu-Henke, X. The Seamless Low-cost Development Platform LoRra for Model based Systems Engineering. Proceedings of the 8th International Conference on Model-Driven Engineering and Software Development, Valetta 2020

[8] Roskam, R. et al. Abschlussbericht zum Forschungsvorhaben .ZW 6- 85007817 OSiS - Offene Steuerung mit integriertem Sensorcluster. Ostfalia 2020

[9] NN.: Product Information DS1104. Paderborn, dSpace 2020

[10] N.N.: Reference Manual RM0399 STM32H745/755 and STM32H747/757 advanced Arm®-based 32-bit MCUs V3.0, STMicroelectronics, 2020

[11] Roskam, R. Abschlussbericht zum BMBF Forschungsvorhaben 03FH049PX2 "Integratives Werkzeug zur Entwicklung haptischer Bedienelemente", Ostfalia 2016

[12] N.N. Datenblatt Servo Stepping Motor 42A03EC. Rtelligent, 2020

[13] Schröder, D. Elektrische Antriebe - Regelung von Antriebssystemen. Berlin, Springer-Verlag 2015