B a c h e l o r a r b e i t

Softwareintegration eines DLIP-Bearbeitungskopfes zur automatischen Umschaltung der Strukturperiode

eingereicht an der
Fakultät Physikalische Technik/Informatik der
Westsächsischen Hochschule Zwickau
zur Erlangung des akademischen Grades eines

Bachelor of Engineering

vorgelegt von: **H e i n r i t z ,** geb. am: 11.05.2003
J a k o b

Physikalische Technik
Studienschwerpunkt Mikrotechnologie

Erstbetreuer/Zweitbetreuer:	Prof. Franke/Prof. Hartmann
Auftraggeber:	ACSYS Lasertechnik GmbH
Betreuer des Auftraggebers:	M.Sc. Tommy Knebel

Autorenreferat/Vorwort

Die vorliegende Bachelorarbeit hatte das Ziel, die bestehende Anlagensoftware der Firma ACSYS Lasertechnik GmbH so zu erweitern, dass Oberflächen-funktionalisierungen mit sehr kleinen regelmäßigen Strukturen einfach und prozesssicher hergestellt werden können.

Die dazu verwendete Hardware bestand aus einem Laser, einem DLIP-Modul der Firma Fusion Bionic und einem PC mit einer Steuerkarte der Firma Scanlab. Ein wesentlicher Bestandteil der Arbeit war die Implementierung einer Multithreading-Schnittstelle zur Kommunikation mit dem DLIP-Controller. Dazu wurde ein Zustandsautomat entwickelt, welcher die Initialisierung, das Verfahren und die Statusüberwachung des DLIP-Moduls sicherstellt.

Ein weiterer Aspekt war die Untersuchung von entstehenden Fehlern, wie Fokusverschiebung und Arbeitsfeldverzerrung. Dabei wurde der mathematische Zusammenhang zwischen Strukturperiode und Fokusverschiebung ermittelt.

Das Ergebnis dieser Arbeit ist eine Softwareerweiterung, welche die Ansteuerung des DLIP-Moduls sowie die weitestgehend automatische Korrektur der entstehenden optischen Fehler realisiert.

Die Erweiterung wurde erfolgreich auf unterschiedlichste Fehlereinflüsse (Fehleingaben, Kommunikationsabbrüche, etc.) getestet. Die Software kann so auf einer noch zu entwickelnden neuen Anlage kundentauglich eingesetzt werden.

Inhalt

I. Abbildungsverzeichnis

II. Tabellenverzeichnis

III. Abkürzungsverzeichnis

Abkürzung	Bedeutung
DLIP	Direct Laser Interference Patterning
DLL	Dynamic Link Library
DOE	Diffraktives optisches Element
USB	Universal Serial Bus

1 Einleitung/Motivation

ACSYS Lasertechnik GmbH ist eine Firma, die auf hochpräzise Maschinen im Bereich der Lasermaterialbearbeitung spezialisiert ist. Die Anwendungsmöglichkeiten sind dabei vielfältig, vom Gravieren, Markieren und Strukturieren bis zum Schneiden und Schweißen von Werkstücken [1]. Besonders interessant ist im Bereich der Strukturierung die Oberflächenfunktionalisierung. Damit können gezielt Eigenschaften der Werkstückoberfläche, wie Haftung von Beschichtungen oder Wasser, Korrosionsbeständigkeit und Oberflächenreibung manipuliert werden [2], [3].

Übliche Methoden der Oberflächenfunktionalisierung sind beispielsweise Beschichtung, Ätzen der Oberfläche oder Aktivierung mithilfe eines Plasmas. Dabei sind aber die Prozessparameter unter Umständen schwer zu steuern bzw. der Aufwand pro Fläche ist sehr hoch. Unter Umständen werden umweltschädliche Substanzen verwendet. Eine schnelle, günstige und umweltfreundliche Alternative dazu ist die Oberflächenfunktionalisierung mit Laser.

Dabei werden regelmäßige Strukturen mit Strukturperioden im µm-Bereich auf der Oberfläche des Werkstückes erzeugt. Dieses Verfahren ist mit einem üblichen Laseraufbau nur sehr umständlich zu realisieren. Kleine Strukturgrößen führen zu einer hohen Prozessdauer in der Fläche. Außerdem ist der übliche Aufbau in der Auswahl bzw. Anpassungsmöglichkeit der Strukturgröße eingeschränkt. Eine deutlich schnellere und flexiblere Alternative zum klassischen Scanner-Laser ist die direkte Laserinterferenzstrukturierung (Direct Laser Interference Patterning, DLIP) [4]. Dabei wird der Laser vor dem Eintritt in die Umlenkvorrichtung in mehrere Teilstrahlen aufgeteilt, die durch die fokussierende Optik auf dem Werkstück zur Interferenz gebracht werden. Damit können sehr regelmäßige Muster, sowohl in Strukturperiode als auch Strukturtiefe, erreicht werden. Dieses Verfahren ist also ideal für die Oberflächenfunktionalisierung geeignet. Das DLIP-Modul ist ein Interferenzmodul der CORErapid Serie der Firma Fusion Bionic und wurde im Rahmen eines öffentlich geförderten Projektes zur Verfügung gestellt.

2 Oberflächenbearbeitung mit DLIP

Direct Laser Interference Patterning (kurz DLIP) bietet eine schnelle und genaue Möglichkeit, Oberflächen mit sehr kleinen regelmäßigen Strukturen zu bearbeiten. Dafür wird sich das Prinzip der Interferenz zunutze gemacht.

2.1 Üblicher Laseraufbau

Ein Laseraufbau besteht grundlegend aus drei Bestandteilen: einer Laserquelle, die den Laserstrahl in die Maschine einbringt, eine Umlenkvorrichtung, die den Laserstrahl auf dem zu bearbeitenden Werkstück positioniert und eine fokussierende Optik, die den Laserstrahl in der Arbeitsebene fokussiert. Im verwendeten Aufbau übernimmt ein Galvanometerscanner die Umlenkung und Fokussierung des Laserstrahls. Dieser enthält zwei Galvanometerspiegel als Umlenkung und ein F-Theta-Objektiv als fokussierende Optik.

Abbildung 1: Aufbau Galvanometerscanner [5]

Die Spiegel lenken, gesteuert von Galvanometerantrieben, den Laser in x- bzw. y-Richtung vom Mittelpunkt des Objektivs ab. Das F-Theta-Objektiv ist eine spezielle Planfeldoptik und fokussiert den eingehenden Laserstrahl in die Arbeitsebene. Die x- und y-Position des fokussierten Lasers in der Arbeitsebene hängt dabei nicht von der Eintrittsposition, sondern nur vom Winkel des Lasers in x- bzw. y-Richtung beim Eintreffen in das F-Theta-Objektiv ab [6]. Aufgrund der Spiegelanordnung im Scanner ist der Winkel des Lasers nicht linear vom Spiegelwinkel abhängig, es entsteht eine kissenförmige Bildfeld-Verzeichnung.

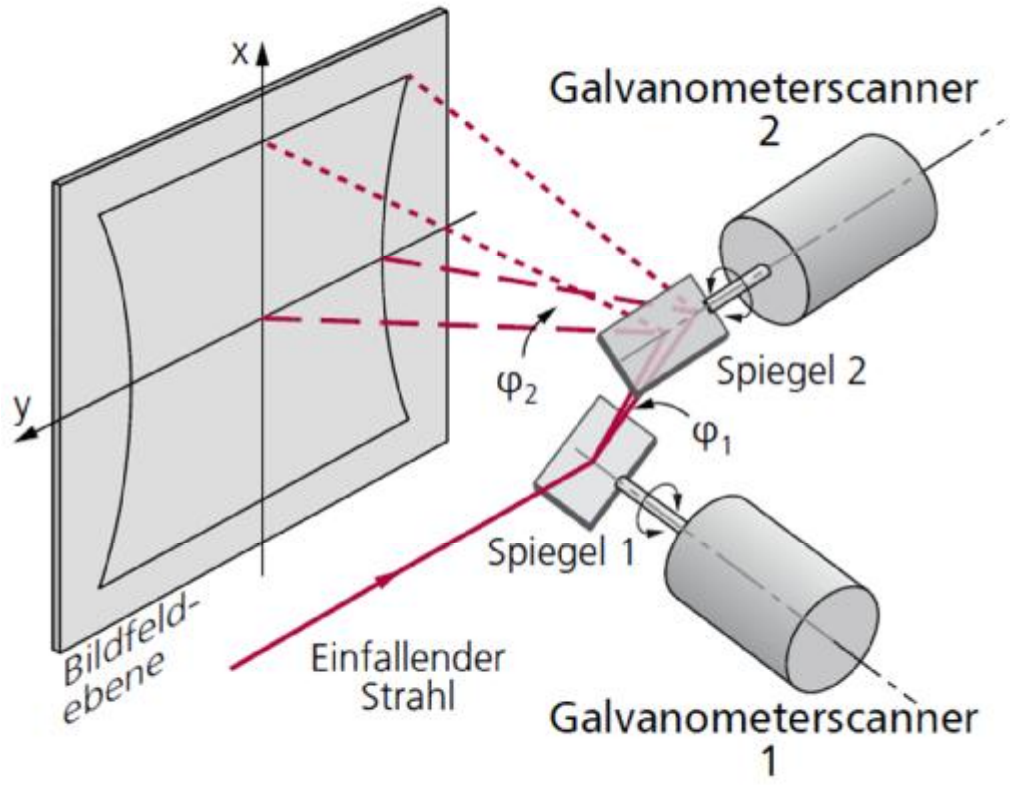

Abbildung 2: Verzeichnung durch Zwei-Spiegel-System [7, S. 186]

Eine weitere Art der Bildfeld-Verzeichnung tritt bedingt durch das F-Theta-Objektiv auf. [7]. Die Formen der Verzeichnungen durch Spiegelanordnung und F-Theta-Objektiv sowie die resultierende Verzeichnung sind in Abbildung 3 dargestellt.

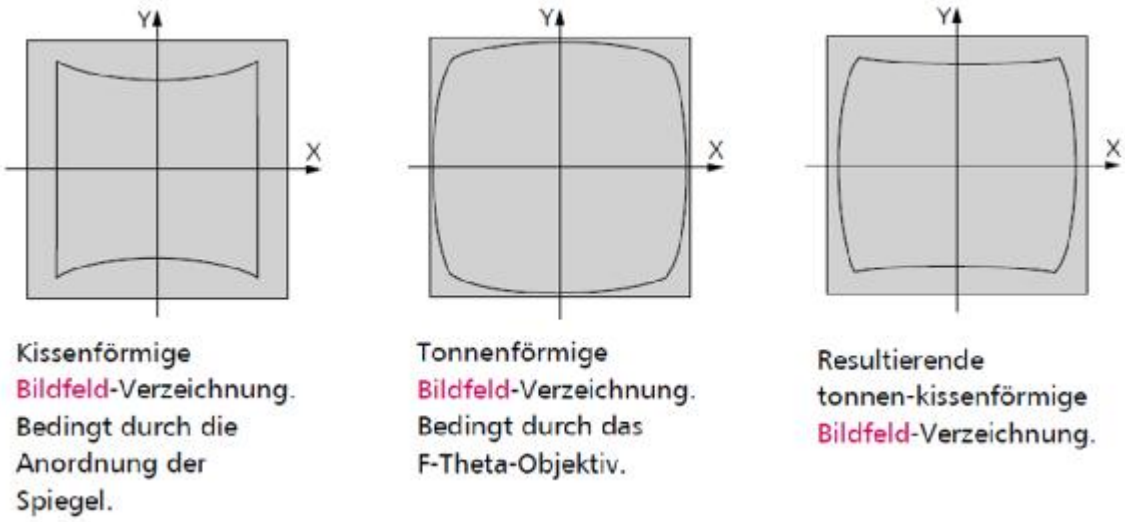

Abbildung 3: Resultierende Bildfeldverzeichnung [7, S. 186]

Die Korrektur der Bildfeldverzerrung kann von der Steuereinheit des Galvanometer-scanners mithilfe von Korrekturtabellen vorgenommen werden. Damit stimmt die Soll-Position des Lasers mit der tatsächlichen Position auf dem Werkstück überein.

Zur exakten Ausrichtung des Arbeitsbereiches ist weiterhin die Einstellung von Skalierung, Rotation sowie Verschiebung des Arbeitsbereiches durch entsprechende Matrizen möglich.

Dabei wird das Produkt aus Rotations- und Skalierungsmatrix berechnet, anschließend mit den ursprünglichen Befehlskoordinaten multipliziert und die resultierenden Koordinaten zur Offset-Matrix addiert.

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = M \times \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} \quad (1)$$

$$M = M_T \times M_R \times M_S \quad (2)$$

Die erhaltenen Koordinaten werden nun mit der Korrekturtabelle korrigiert und die korrespondierenden Positionsbefehle an die beiden Spiegel geschickt [7, S. 244ff].

2.2 Oberflächenfunktionalisierung mit üblichem Aufbau

Bei der Funktionalisierung von Oberflächen müssen regelmäßige Muster mit Strukturperioden im Mikro- bis Nanometerbereich in der Oberfläche erzeugt werden. Mit dem üblichen Aufbau einer Lasermaschine müssten also präzise Strukturen im Abstand von wenigen Mikro- bis Nanometern zueinander von der Oberfläche einzeln abgetragen werden. Bei Änderung der Größe einer Struktur müsste die Breite des Lasers in der Arbeitsebene geändert werden, was unweigerlich mit einer Veränderung der optischen Konfiguration verbunden wäre. Zudem kann ein Werkstück mit dem üblichen Aufbau nur linienweise bearbeitet werden. Das führt zu großen Bearbeitungszeiten für die Funktionalisierung von größeren Oberflächen.

2.3 Prinzip Interferenz

Interferenz beschreibt den Prozess der Überlappung zweier oder mehrerer kohärenter Wellen. Je nach Phasenverschiebung der Wellen am Treffpunkt tritt konstruktive oder destruktive Interferenz auf. Beträgt die Phasenverschiebung der Lichtwellen ein ganzzahliges Vielfaches der Wellenlänge, so tritt konstruktive Interferenz auf, die Amplituden beider Wellen addieren sich. Beträgt die Phasenverschiebung eine halbe Wellenlänge bzw. geradzahlige Vielfache der halben Wellenlänge, so tritt destruktive Interferenz auf, die Einzelamplituden subtrahieren sich voneinander. Die Intensität der Interferenzmaxima berechnet sich nach der Gleichung

$$I = I_1 + I_2 + 2\sqrt{I_1 \cdot I_2 \cdot \cos\ \delta}$$ [8, S. 262] (3)

Wobei I_1 und I_2 die Teilintensitäten der einzelnen Wellen sind, δ die Phasendifferenz in rad ist und I die resultierende Gesamtintensität der Überlagerung ist.Bei Interferenz zweier Wellen gleicher Intensität ergibt sich daraus

$$I = 2 \cdot I_1 + 2 \cdot I_1 \cdot \cos\delta \quad (4)$$

Im Interferenzmaximum ($\delta = 0$) beträgt die Gesamtintensität also das Vierfache der Intensitäten der Einzelstrahlen. Es ist damit möglich, sehr hohe Intensitäten auf sehr kleine Bearbeitungsflächen aufzubringen.

Je nach Anzahl und Position der interferierenden Strahlen zueinander entsteht dabei ein ein- oder zweidimensionales Interferenzmuster.

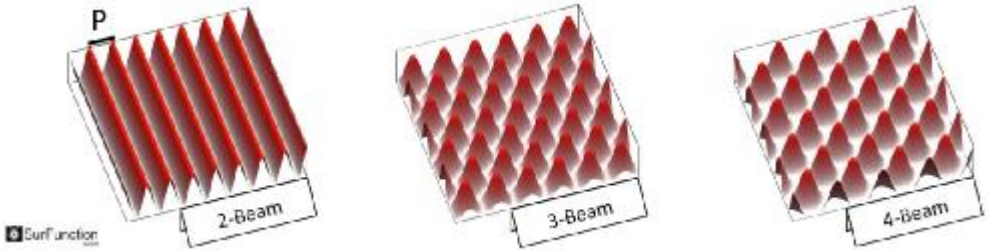

Abbildung 4: Interferenzmuster in Abhängigkeit der Anzahl Strahlen [4]

Die Strukturperiode, also der Abstand der Intensitätsmaxima voneinander, hängt von der Wellenlänge der Strahlung, dem Abstand zwischen Strahlungsquellen und Projektionsebene sowie dem Abstand der Strahlungsquellen zueinander ab. Um die Kohärenz der Strahlungswellen zu garantieren, wird eine einzelne kohärente Strahlungsquelle verwendet und die erzeugte Strahlungswelle mithilfe von optischen Gittern oder diffraktiven optischen Elementen (DOEs) in mehrere, zueinander kohärente Teilwellen aufgeteilt.

2.3.1 Interferenz am optischen Gitter

Optische Gitter lassen sich in zwei Arten unterteilen: Amplitudengitter und Phasengitter. Bei Amplitudengittern wird die Amplitude des Lichts durch das Gitter verändert, also geschwächt oder komplett ausgelöscht. Daher entstehen am Gitter nach dem huygen-fresnelschen Prinzip mehrere Punktquellen, die in den Raum hinein strahlen und somit ein Interferenzmuster über der gesamten bestrahlten Fläche erzeugen, wie in Abbildung 5 zu sehen.

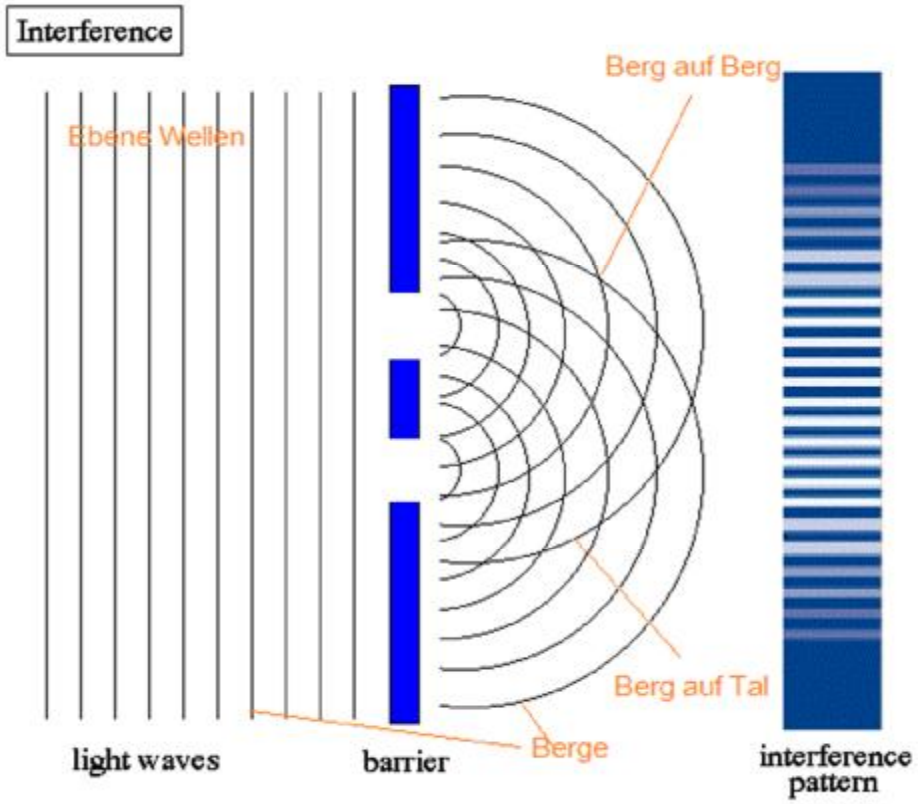

Abbildung 5: Interferenzmuster am Doppelspalt [8]

Ist die Gitterperiode über das Gitter konstant, sind die Abstände der Interferenzmaxima abhängig von der Wellenlänge der Strahlung, der Gitterperiode sowie dem Abstand zwischen Gitter und Schirm. Sowohl Intensität als auch Abstände der Interferenzmaxima sind dabei nicht konstant, sondern vom Abstand zum mittleren Interferenzmaximum abhängig. Sowohl die Intensität der Maxima, als auch der Abstand zwischen ihnen nimmt mit größerem Abstand zur optischen Achse ab.

2.3.2 Interferenz mit diffraktivem optischen Element

Ein diffraktives optisches Element (DOE) ist ein Phasengitter. Dabei wird also nicht die Amplitude des Lichts durch das Gitter verändert, sondern seine Phase. Ein DOE ist eine Glasplatte, auf der eine bestimmte Mikrostruktur aufgebracht wurde. Diese Struktur führt dazu, dass die Strahlung beim Durchlaufen des DOE je nach Position einen unterschiedlichen optischen Weg zurücklegt. Damit wird die Phase der Strahlung nach Durchlaufen des DOE verändert. Die Mikrostruktur des DOE muss keine regelmäßige Struktur mit konstanter Gitterperiode sein.

Durch Anpassung der Mikrostruktur können vielfältige Interferenzmuster erzeugt werden. So kann das DOE zur Strahlformung oder aber zur Strahlaufteilung eingesetzt werden. Bei der Strahlformung wird die Mikrostruktur so gewählt, dass nur das nullte Interferenzmaximum zugelassen wird. Dieses kann in Form, Größe und Intensitäts-verteilung vom Eingangsstrahl abweichen, je nach Mikrostruktur des DOE. Bei der Strahlaufteilung wird mehr als ein Interferenzmaximum zugelassen. Die Abstände der Maxima zueinander sind dabei unabhängig vom Abstand der Maxima zur optischen Achse. Innerhalb einer Ebene ist der Abstand zwei benachbarter Maxima stets konstant. Die Anzahl Teilstrahlen kann genau gesteuert werden, allerdings entstehen immer ungewollte Nebenmaxima. Diese haben zwar eine deutlich geringere Intensität als die gewollten Teilstrahlen, tragen aber dennoch zu Leistungsverlusten der gewünschten Teilstrahlen bei.

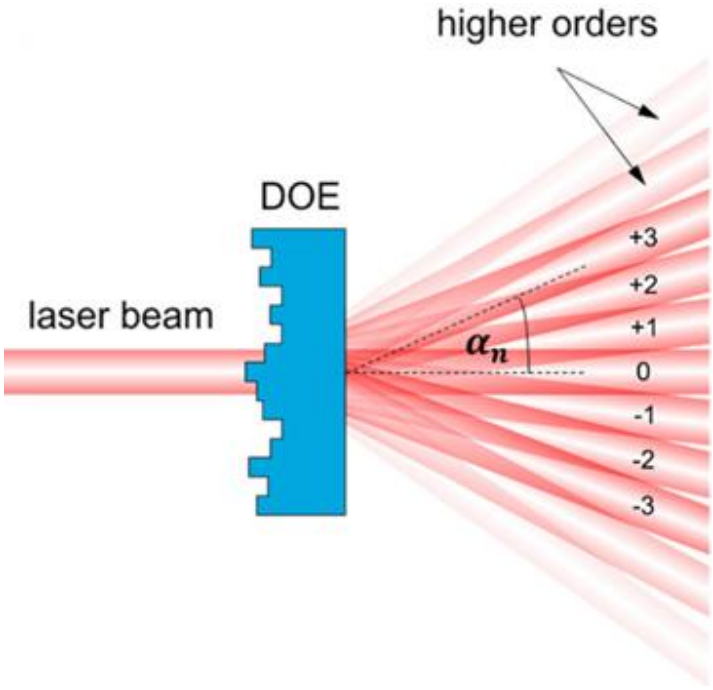

Abbildung 6: DOE als Strahlteiler [9, S. 2]

Die resultierenden Teilstrahlen sind kohärent zueinander. Werden sie also von einer fokussierenden Optik auf einen einzigen Punkt fokussiert, erhält man ein konkretes Interferenzvolumen, in dem die Teilstrahlen aufeinandertreffen.

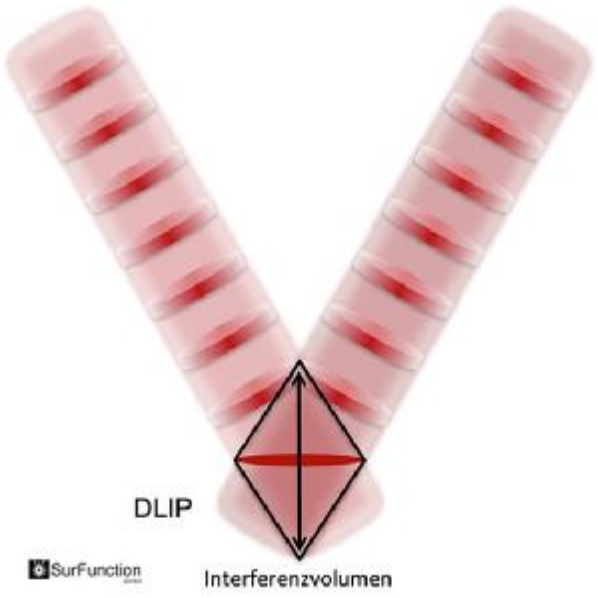

Abbildung 7: Interferenzvolumen DOE [4]

Dabei hängt die resultierende Strukturperiode nur von der Wellenlänge der Laserstrahlung und dem Winkel der Laserstrahlen zueinander ab. Das dabei entstehende Interferenzmuster ist, bei geeigneter Einstellung der Parameter, sehr regelmäßig und kann Strukturperioden im Mikrometer- und Submikrometerbereich erzeugen [4]. Der Einsatz eines DOE und dessen Interferenzmuster zur Oberflächenfunktionalisierung nennt sich DLIP (Direct Laser Interference Patterning).

2.4 DLIP-Modul

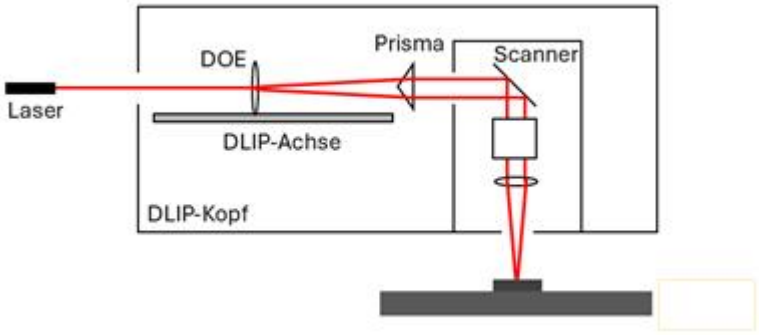

Abbildung 8: Aufbau DLIP-Modul

Ein DLIP-Modul besteht aus einem verfahrbaren DOE, einem Prisma und einem klassischen Galvanometerscanner, wie in Kapitel 2.1 beschrieben. Das DOE teilt den eingehenden Laserstrahl in mehrere Teilstrahlen auf. Die Teilstrahlen werden anschließend durch das Prisma zueinander parallelisiert. Die Parallelstrahlen werden in den Scan-Kopf geleitet und anschließend durch die Optik in die Arbeitsebene fokussiert. In der Arbeitsebene überlagern sich die Teilstrahlen und es kommt zur Entstehung eines Interferenzmusters.

Um die Strukturperiode des Interferenzmusters auf dem Werkstück zu verändern, wird das DOE entlang der optischen Achse verfahren. Das führt dazu, dass die Teilstrahlen nach dem Brechen am Prisma einen größeren Abstand voneinander haben. Durch die optischen Eigenschaften des F-Theta-Objektivs resultiert daraus ein größerer Winkel der Strahlen zueinander in der Fokusebene.

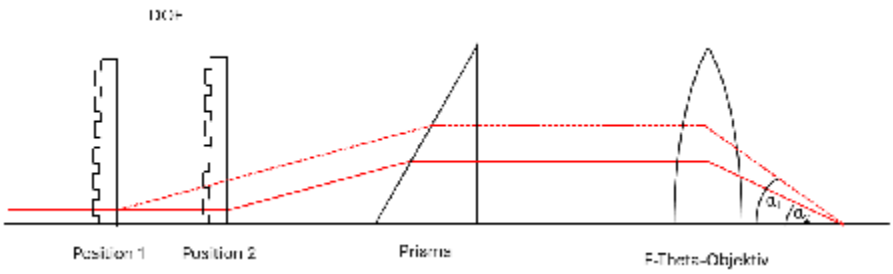

Abbildung 9: Strahlengang durch F-Theta-Objektiv

3 Versuchsaufbau

Der verwendete Versuchsaufbau besitzt die grundlegenden Bestandteile Laserquelle und DLIP-Modul, um die Interferenzstrukturierung auf dem Werkstück zu erzeugen, außerdem eine mechanische Z-Achse, eine Steuerkarte und einen PC. Auf dem PC befinden sich die Programme AcLaser und Fusion Interface.

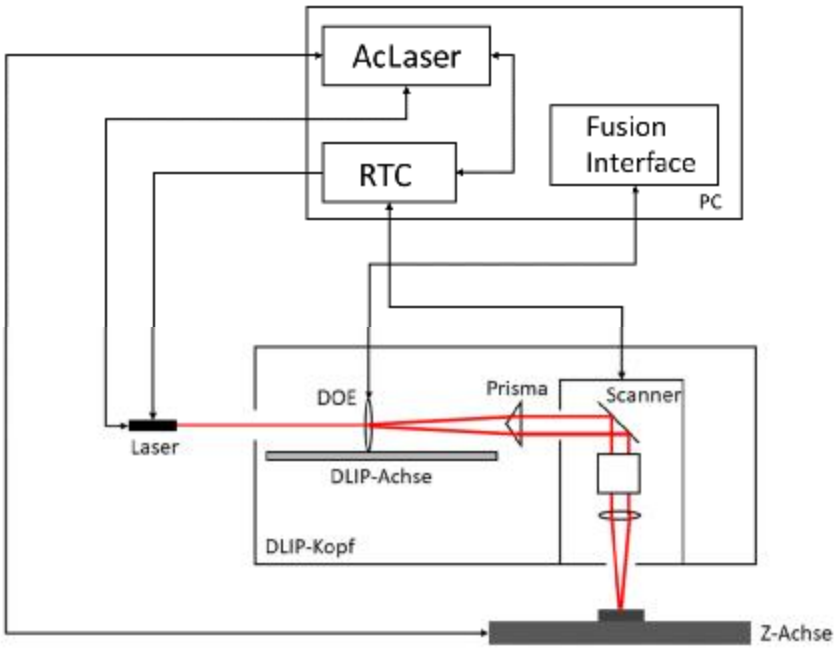

Abbildung 10: Aufbau DLIP-Anlage

3.1 Laserquelle

Eine übliche Laserquelle hat die Parameter Laserleistung, Frequenz und Dauer der Pulse sowie eine Leistungsfreigabe. Diese Parameter können entweder direkt über die elektrische Schnittstelle oder über eine Softwareschnittstelle definiert werden. Elektrische Signale können ein Gate-Signal, ein analoges Signal zur Leistungssteuerung (0 bis 10 V) oder ein Puls-Weiten-Modulationssignal sein. Die elektrische Ansteuerung erfolgt über die Steuerkarte (siehe Kapitel 3.3). So ist eine Synchronisierung zwischen Laser und Steuerkarte möglich.

Zusätzlich stehen Informationen über den Status des Lasers zur Verfügung. Diese werden von AcLaser ausgelesen.

3.2 Interferenzmodul

Der DLIP-Controller erhält von der Fusion Interface-Software den Befehl, eine bestimmte Strukturperiode einzustellen. Aus dem erhaltenen Befehl berechnet dieser dann die Achsposition, zu der das DOE gefahren werden muss und gibt der Achse das Signal zum Verfahren an die berechnete Position. Die Gleichung, mit der die Achsposition berechnet wird, lautet

$$\Delta = \frac{1}{(m_1 + m_2 \cdot x^{m_3 - 1})} \text{ [11]} \tag{5}$$

Dabei ist x die Position des DOE auf der Achse relativ zur Nullposition in Millimetern, Δ ist die Strukturperiode in µm. m_1 bis m_3 sind anlagenspezifische Fitparameter, die je nach Art des DLIP-Moduls, der Anzahl Strahlen und der Brennweite der F-Theta-Linse unterschiedlich sind.

Die Werte für die Parameter sowie die Ober- und Untergrenzen der anfahrbaren Strukturperioden liest der Controller aus der Konfigurationsdatei aus, die mit dem DLIP-Modul vom Hersteller geliefert wird. Diese beinhaltet außerdem die Wellenlänge, für die der Kopf ausgelegt ist, sowie die maximale Strecke, die das DOE auf der Achse verfahren werden kann [10].

Die Achse des DLIP-Moduls hat selbst keinen Encoder, der Controller hat also grundsätzlich keine Information darüber, an welcher Position sich die Achse zu einem bestimmten Zeitpunkt wirklich befindet. Er speichert nur die letzte angefahrene Position. Die Achse muss also nach jedem Neustart des Controllers referenziert werden, damit der Controller die Position der Achse kennt.

Je nach gewünschtem Interferenzmuster kann das DLIP-Modul mit zwei, drei oder vier Teilstrahlen betrieben werden, woraus sich entsprechend andere Fitparameter sowie Grenzen für die möglichen Strukturperioden ergeben. Wenn die Strahlenanzahl von der Fusion-Interface Software an den Controller geschickt wird, lädt dieser die Fitparameter aus seiner Konfiguration.

Die Anzahl Teilstrahlen muss aber mechanisch an der Maschine selbst eingestellt werden, eine Änderung ist also nicht während des Bearbeitungsvorganges möglich.

3.3 Scankopf/Steuerkarte

Der Galvanometerscanner innerhalb des DLIP-Moduls wird nicht vom DLIP-Controller angesteuert, sondern von der Steuerkarte (RTC6-Karte der Firma Scanlab). Diese ist also für die Positionierung des Lasers auf dem Werkstück verantwortlich. Die Bahn, auf der sich der Laser bewegt, wird in der Steuerkarte gespeichert. Um den Laser entlang der Bahn zu bewegen, werden die Spiegel entsprechend ausgelenkt. Dabei werden die Korrekturtabelle und die Matrizen, wie in Kapitel 2.1 beschrieben, auf die Sollposition des Lasers angewandt. So wird immer die richtige Ausgabeposition sichergestellt. Während der Abarbeitung eines Vektors kann immer nur eine Konfiguration, bestehend aus Korrekturtabelle und Matrizen, verwendet werden. Zwischen ausgegebenen Vektoren können aber unterschiedliche Konfigurationen geladen werden. Dafür hat die verwendete Steuerkarte acht Konfigurations-Slots, zwischen denen schnell gewechselt werden kann. Werden mehr als acht verschiedene Konfigurationen benötigt, kann AcLaser weitere Konfigurationen in die Slots schreiben. Das Überschreiben eines Konfigurationsslots benötigt allerdings mehr Zeit als das Wechseln zwischen den Slots.

Weiterhin kann die Steuerkarte den Laser ansteuern. Die Leistung kann dabei über ein Analogsignal zwischen 0 V und 10 V ausgegeben werden. Zusätzlich wird über ein Puls-Weiten-Modulationssignal eingestellt, wann und wie lange der Laser aktiv ist. Die Synchronisierung von gepulsten Lasern mit der Bewegung der Galvanometerspiegel wird ebenfalls von der RTC übernommen.

3.4 Achse für Fokusposition

Die Z-Achse der Maschine ist dafür verantwortlich, das Werkstück entlang der optischen Achse zu verfahren. Ändert sich die Werkstückhöhe oder die optische Konfiguration der Maschine, ergibt sich eine Veränderung der Position der Fokusebene relativ zur Werkstückoberfläche. Um diese Veränderung zu kompensieren, kann die Z-Achse den Werkstückhalter senkrecht zur Fokusebene verfahren. In der AcLaser-Software ist die Z-Achse manuell verfahrbar, auch eine Kompensation der Höhe eines Werkstücks ist möglich. Dafür muss die Werkstückhöhe in AcLaser angegeben werden.

3.5 AcLaser

Die Software AcLaser ist für die zentrale Steuerung der Maschine zuständig. Sie bedient eine Vielzahl von Schnittstellen zu Hardware, die in üblichen ACSYS-Maschinen verwendet wird. Die Grundkonfiguration der Anlage wird ebenfalls mit AcLaser vorgenommen.

Die Software besitzt einen Editor, mit welchem das auszugebende Bearbeitungsmuster komfortabel bearbeitet werden kann. Beispielsweise gibt es Bausteine für Polylinien, Kreise sowie 1D- und 2D-Codes (Barcode, Datamatrixcode, QR-Code).

Eine wichtige Aufgabe im Testaufbau ist die Ansteuerung der Steuerkarte. Die Software lädt die Korrekturtabellen und Matrizen in die Korrekturslots und sendet der Steuerkarte das Bearbeitungsmuster. Sie legt die Leistung und andere Parameter des Lasers fest und sendet diese entweder direkt oder über die Steuerkarte an den Laser. Auch die Fokuskorrektur durch das Verfahren der Z-Achse findet über AcLaser statt. Dafür gibt es eine Funktion, die Fokuskorrektur durchzuführen. Optional kann dabei die Teilehöhe des Werkstücks beachtet werden.

Das Muster, mit dem das Werkstück bearbeitet werden soll, kann ebenfalls in AcLaser erstellt und bearbeitet werden, bevor es an die Steuerkarte geschickt wird. Während der Bearbeitung wird der Fortschritt sowie der Status von Steuerkarte und Laser ausgelesen bzw. überwacht. Im Fehlerfall wird die Bearbeitung abgebrochen.

3.6 Fusion Interface

Abbildung 11: Software Fusion Interface

Die Fusion Interface-Software stellt grundlegende Funktionen zur Ansteuerung des DLIP-Moduls zur Verfügung. Es gibt eine Funktion zum Referenzieren der Achse. Bei Eingabe einer Strukturperiode wird dieser Wert an den DLIP-Controller übermittelt und die Achse an die entsprechende Position gefahren. Allerdings gibt es keine Funktionen zum Lesen bzw. Schreiben der Konfigurationsdatei.

3.7 Arbeitsablauf

Die Bearbeitung eines Werkstücks mit DLIP erfordert einige manuelle Schritte, die vom Anwender ausgeführt werden müssen. Wenn ein Werkstück beispielsweise mit zwei Mustern verschiedener Strukturperioden bearbeitet werden soll, sieht der Arbeitsablauf wie folgt aus:

- AcLaser und Fusion Interface öffnen
- DLIP referenzieren
- Strukturperiode 1 in Fusion Interface einstellen
- In AcLaser wechseln
- Muster 1 in AcLaser laden und Bearbeitung durchführen
- In Fusion Interface wechseln
- Strukturperiode 2 einstellen
- In AcLaser wechseln
- Muster 2 in AcLaser laden und Bearbeitung durchführen

3.8 Folgen der Änderung der Strukturperiode

Bei Ausführung des Beispiels aus 3.7 im Handbetrieb ergaben sich zwei Probleme. Bei Änderung der Strukturperiode ändern sich sowohl die Fokusebene als auch die Feldverzerrung. Die Änderung der Fokusebene ergibt sich daraus, dass beim Verfahren des DOE die Strahlen nicht perfekt parallel in das F-Theta-Objektiv geleitet werden. Deshalb werden sie nicht mehr in der Brennweite des Objektivs fokussiert.

Um die Fokusverschiebung in Abhängigkeit der Strukturperiode festzustellen, wurde eine Messreihe aufgenommen. Dabei wurden für acht Strukturperioden zwischen 6µm und 18µm jeweils mehrere senkrechte Striche nebeneinander auf ein Werkstück aufgebracht. Zwischen jedem Strich wurde die Z-Achse um 0,5 mm verfahren. Die beste Fokusposition wurde anschließend durch das Vermessen der Abtragslinien mit einem Mikroskop bestimmt.

In der besten Fokusposition sind die Interferenzmaxima am größten und schärfsten, je weiter das Werkstück aus dem Fokus gefahren wird, desto unschärfer und kleiner werden die Interferenzmaxima.

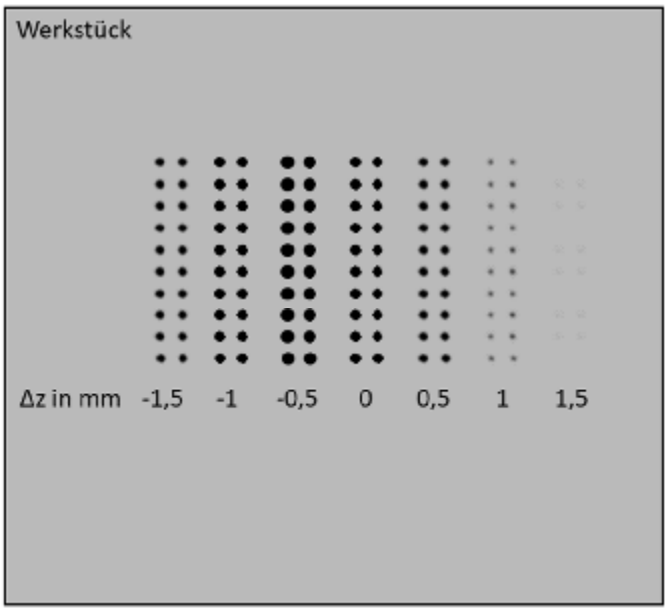

Abbildung 12: Skizze Messreihe bei 8µm Strukturperiode

Die Änderung der Fokusebene in Abhängigkeit der Strukturperiode wurden in Tabelle 1 aufgenommen.

Tabelle 1: Fokusverschiebung in Abhängigkeit der Strukturperiode

Strukturperiode	Fokusverschiebung in mm
6	-1
8	-0,5
10	0
12	0,5
14	1
16	1,5
18	2

Aus den aufgenommenen Werten kann ein linearer Zusammenhang interpoliert werden.

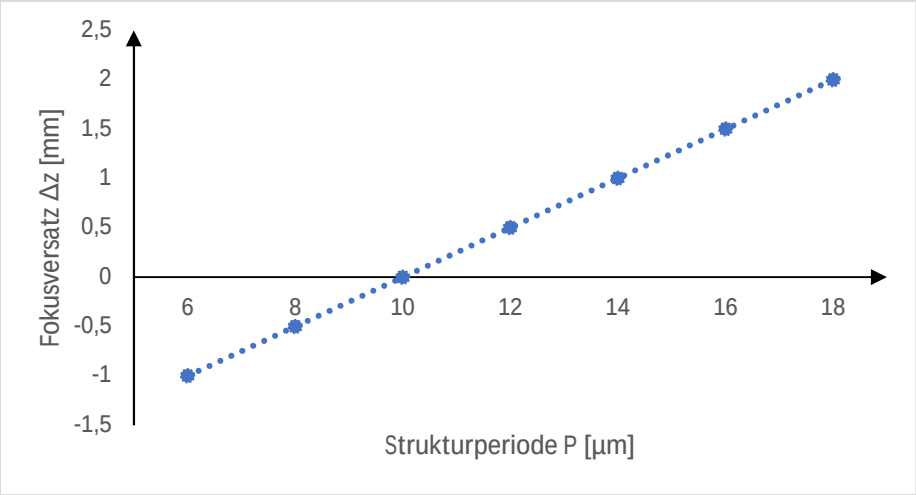

Abbildung 13: Linearer Fit der Messwerte

Die Gleichung der Fokusverschiebung lautet für die aufgenommenen Messwerte

$$\Delta z = 0{,}25\,\frac{mm}{\mu m} \cdot P - 2{,}5\,mm \tag{6}$$

Dabei ist Δz der Fokusversatz relativ zum konfigurierten Laserfokus in Millimeter und P die angefahrene Strukturperiode in Mikrometer. Bei Änderung der optischen Konfiguration der Maschine oder der Anzahl Teilstrahlen, die auf das Werkstück fokussiert werden, kann sich der Zusammenhang zwischen Fokusverschiebung und Strukturperiode ändern. Dieser Zusammenhang muss also nach jeder Änderung des optischen Setups erneut untersucht werden.

Neben der Fokusverschiebung ändert sich auch die Feldverzeichnung bei Änderung der Strukturperiode. Um die Feldverzeichnung für eine bestimmte Strukturperiode zu korrigieren, liegt bereits eine Basis-Korrekturtabelle in der Steuerkarte vor. Für abweichende Strukturperioden müssen eigene Korrekturtabellen erstellt werden. Dafür wurde die zu korrigierende Strukturperiode ausgewählt und die Basis-Korrekturtabelle geladen. Mit diesen Parametern wurde anschließend ein Werkstück mit einem Kontrollraster bearbeitet. Ein Kontrollraster besteht aus mehreren Kreuzen, die in x- und y-Richtung gleichmäßig voneinander entfernt sind und den gesamten Arbeitsbereich abdecken. Im Versuchsaufbau bestand das Raster aus 25 Kreuzen, jeweils 5 pro Zeile und Spalte. Diese wurden über den Arbeitsbereich von 25 mm mal 25 mm gleichmäßig verteilt, hatten also einen Abstand von jeweils 6,25 mm zueinander. Das bearbeitete Werkstück wurde anschließend mit einem Videomessmikroskop vermessen. Dabei wurden die Positionen der Messkreuze auf dem Kontrollraster des Werkstücks aufgenommen und in einer Textdatei ausgegeben (Siehe Anlage 1 und 2). Diese wurde zusammen mit der Basis-Korrekturtabelle in ein Programm des Steuerkartenherstellers geladen, welches aus der Basiskorrektur anhand des Kontrollrasters eine neue Korrekturtabelle speziell für die ausgewählte Strukturperiode erzeugt.

Die somit erzeugte Korrekturtabelle kann in einen Konfigurationsslot der Steuerkarte geladen werden. Wird eine Struktuperiode angefahren, deren dazugehörige Korrekturtabelle in der Steuerkarte gespeichert ist, so kann der entsprechende Konfigurationsslot schnell geladen werden.

3.9 Arbeitsablauf mit Korrektur

Werden Fokusänderung und Feldverzeichnung im Betrieb ebenfalls korrigiert, so ergibt sich folgender Arbeitsablauf:

- AcLaser und Fusion Interface öffnen
- DLIP referenzieren
- Strukturperiode 1 in Fusion Interface einstellen
- In AcLaser wechseln
- Z-Achse in richtigen Fokus für Strukturperiode 1 fahren
- Korrekturslot für Strukturperiode 1 in Steuerkarte auswählen
- Muster 1 in AcLaser laden und Bearbeitung durchführen
- In Fusion Interface wechseln
- Strukturperiode 2 einstellen
- In AcLaser wechseln
- Z-Achse in richtigen Fokus für Strukturperiode 2 fahren
- Korrekturslot für Strukturperiode 2 in Steuerkarte auswählen
- Muster 2 in AcLaser laden und Bearbeitung durchführen

3.10 Nachteile der dezentralen Ansteuerung

Wie in Kapitel 3.9 aufgezeigt, gibt es bei der dezentralen Ansteuerung des DLIP-Moduls eine Vielzahl an Fehlerquellen. Wird die Strukturperiode zwischen zwei Bearbeitungsschritten nicht gewechselt, wird mit einer anderen Struktur bearbeitet als erwartet. Wird die DLIP-Achse vor Benutzung nicht referenziert, entspricht die tatsächliche Strukturperiode nicht dem erwarteten Wert. Wird die Fokuskorrektur nicht durchgeführt, befindet sich die Werkstückoberfläche nur noch teilweise oder gar nicht mehr im Interferenzvolumen.

Damit wird die Bearbeitungsfläche kleiner und die Teilstrahlen außerhalb des Interferenzvolumens bearbeiten das Werkstück unter Umständen ungewollt. Wird nicht der richtige Korrekturslot geladen, wird die Verzeichnung falsch korrigiert und das Bearbeitungsmuster nicht korrekt auf dem Werkstück abgebildet.

Bei wiederholter Fertigung eines Werkstückes müssen die gleichen Parameter vom Anwender eingestellt werden. Diese werden weder von Fusion Interface noch von AcLaser gespeichert.

4 Programm zur automatischen Fertigung

Es soll eine Softwareerweiterung programmiert werden, um die Fertigung mit DLIP zu automatisieren. Die Nachteile aus Kapitel 3.10 sollen so weit wie möglich behoben werden: Die Einstellung und der Wechsel der Strukturperiode, die Referenzierung der DLIP-Achse, Änderung der Steuerkarten-Konfiguration, Verfahren der Z-Achse sowie die Speicherung aller notwendigen Parameter im AcLaser-Layout sollen mit dieser Software realisiert werden.

4.1 Erweiterungsmöglichkeiten der AcLaser-Software

Die AcLaser-Software bietet verschiedene Möglichkeiten zur Erweiterung um kunden- bzw. anlagenspezifische Softwaremodule. Die Art der Erweiterung hängt dabei vom konkreten Einsatzzweck ab. Für die automatische Fertigung mit einem DLIP-Modul soll solch eine Erweiterung für die Verwendung im Layout-Baum entwickelt werden.

4.1.1 Aufbau eines Layouts

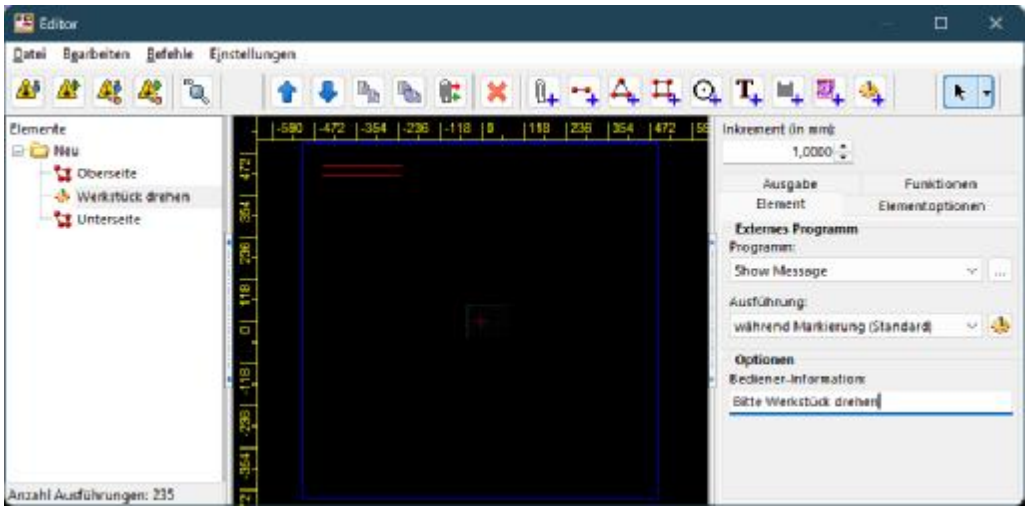

Abbildung 14: Editor AcLaser

Der Editor in AcLaser besteht aus drei Hauptbestandteilen. Auf der linken Seite ist der Layoutbaum. Dieser fungiert als Liste mit den Befehlen, aus welchen das Layout aufgebaut ist.

Der Layoutbaum besteht aus grafischen Elementen und Softwareerweiterungen. Diese werden externe Programme genannt. Die grafischen Elemente können von Linien über Text bis zu Barcodes reichen und werden alle auf der Steuerkarte ausgegeben. Die externen Programme werden durch ein Zahnradsymbol hervorgehoben und werden nicht auf der Steuerkarte, sondern auf dem PC ausgeführt. In der Mitte ist ein Vorschaufeld. In diesem wird eine Vorschau des Musters dargestellt, mit dem das Werkstück markiert wird. Wird ein Element im Layoutbaum ausgewählt, werden die dazugehörigen Parameter auf der rechten Seite angezeigt. Das können für grafische Elemente die Punkte einer Polylinie, Mittelpunkt und Radius eines Kreises oder andere Parameter sein. Für externe Programme hängen die angezeigten Parameter dabei von dessen Aufgabe ab.

Die Elemente im Layoutbaum werden während der Beschriftung von oben nach unten abgearbeitet. Wenn ein grafisches Element bearbeitet wird, wird es direkt in den Speicher der Steuerkarte geschrieben. Falls ein externes Programm als nächstes Element erkannt wird, hält die Steuerkarte ihre Ausführung nach dem Ende des letzten grafischen Elements an und AcLaser führt das externe Programm aus. Nach erfolgreicher Ausführung werden die nächsten Bearbeitungsbefehle wieder mit der Steuerkarte ausgeführt.

Im Beispiel in Abbildung 14 soll ein Werkstück auf Ober- und Unterseite bearbeitet werden. Der Arbeitsbaum dazu besteht aus drei Komponenten: dem ersten Markierbefehl, einem externen Programm und dem zweiten Markierbefehl. Der erste Markierbefehl besteht aus einem waagerechten Strich, der auf der Oberseite des Werkstücks aufgebracht werden soll. Anschließend wird das Programm „Show Message“ ausgeführt. Dieses zeigt eine Meldung an, um dem Anwender zu sagen, dass das Werkstück gedreht werden soll. Der letzte Befehl ist ein weiterer Markierbefehl, der das Werkstück ebenfalls mit einem waagerechten Strich, allerdings auf der Unterseite, bearbeiten soll. Die zu markierenden Striche sind in der Vorschau in der Mitte von Abbildung 14 als rote Linien visualisiert. Das Programm „ShowMessage“ ist angeklickt und seine Parameter werden auf der rechten Seite angezeigt.

4.1.2 Externe Programme

Externe Programme werden beim Abarbeiten des Layoutbaums nicht von der Steuerkarte, sondern von dem PC ausgeführt. Der Code eines externen Programms ist dabei nicht fester Bestandteil von AcLaser. Damit kann AcLaser leicht um kunden- oder anlagenspezifische Softwaremodule erweitert werden.

Ein externes Programm besteht aus bis zu vier Komponenten. Die erste Komponente ist der auszuführende Code selbst. Im obigen Beispiel (siehe Abbildung 14) wird die eingegebene Nachricht angezeigt. Die zweite Komponente ist die Konfiguration des Programms. Diese legt Parameter fest, welche für alle Listenelemente desselben Programms identisch sind. Dazu kann der Port einer seriellen Schnittstelle gehören, über die das Programm mit anderen Geräten kommuniziert, oder aber die Einstellung, welche Parameter veränderbar sind und welche nicht. Wird dasselbe externe Programm im Layout mehrfach aufgerufen, bleibt die Konfiguration für alle Instanzen identisch. Die dritte Komponente beinhaltet die Parameter des externen Programms. Diese können vom Anwender geändert werden und sind Bestandteil des Layouts. Sie werden also in der Layoutdatei gespeichert und sind auch nach dem Laden des Layouts wieder verfügbar. Da sie Bestandteil des Layouts sind, können die Parameter für jedes Element im Layoutbaum unterschiedlich sein. Für das Programm „ShowMessage“ ist die anzuzeigende Nachricht der einzige Parameter. Würde „ShowMessage“ ein zweites Mal im Baum verwendet werden, so könnte die anzuzeigende Nachricht für jedes Element separat eingestellt werden. Die vierte Komponente ist das Panel für die Interaktion mit dem Benutzer. Das externe Programm ist also für die Anzeige geeigneter Steuerelemente zur Bearbeitung der Parameter selbst verantwortlich. Hat es keine Parameter, so ist auch kein Parameter-Panel erforderlich.

4.1.3 Erweiterungsschnittstelle AcLaser

Ein externes Programm ist eine klassische C-Bibliothek. Sie wird als Dynamic Link Library (DLL) in ein spezielles Unterverzeichnis von AcLaser gelegt. Um von AcLaser als gültige Erweiterung erkannt zu werden, muss ein bestimmtes Interface vorhanden sein. Das Interface besteht aus Funktionen, die in fünf Kategorien eingeteilt werden können. Auch wenn Funktionen nicht benötigt werden, müssen sie dennoch implementiert und von der DLL exportiert werden.

Eine Kategorie beinhaltet die Funktionen zum Initialisieren und Finalisieren des Programms. Damit werden Ressourcen beim Laden der Bibliothek angefragt und beim Beenden freigegeben.

Eine zweite Gruppe von Funktionen wird verwendet, um die Erweiterung der Hauptsoftware bekanntzumachen. Mit diesen Funktionen werden ein eindeutiger Bezeichner, ein Anzeigename sowie diverse Fähigkeiten und Eigenschaften an AcLaser übermittelt.

Eine weitere Funktionsgruppe beinhaltet Funktionen zur Anzeige und Konfiguration des Parameter-Panels. Dieses ist Bestandteil des externen Programms, wird jedoch im Zeichenbereich der Kernsoftware angezeigt. Unter anderem sind Funktionen enthalten, welche die Verwendung des Tabulators zum Wechseln der Steuerelemente sicherstellen. Für das Ein- und Ausblenden des Panels existieren ebenfalls Funktionen.

Kategorie vier beinhaltet das Laden und Speichern der Parameter. Wenn Parameter durch Steuerelemente geändert wurden, wird AcLaser über eine Callback-Funktion darüber informiert. Das sorgt dafür, dass die geänderten Parameter ausgelesen und im Baumknoten gespeichert werden. Vor der Ausführung oder beim Anklicken des externen Programms im Baum werden die gespeicherten Parameter an die Steuerelemente geschickt. Somit zeigt das externe Programm stets die richtigen Daten an bzw. verwendet diese auch zur Ausführung.

Schließlich gibt es Funktionen zur Ausführung des Codes. Damit werden bestimmte Aktionen durchgeführt, wenn die Abarbeitung des Layouts den Baumknoten des externen Programmes erreicht. Die Deklarationen der Funktionen sind in Anlage 3 einsehbar.

Für das Interface von externen Programmen ist bei Acsys bereits eine Vorlage vorhanden. Aus dieser wurde ein neues Projekt erzeugt. Damit waren nur noch die Implementierung der erforderlichen Funktionen und das Festlegen der statischen Daten (Name, Eigenschaften, …) notwendig.

Ist die Implementierung des Interface vollständig vorhanden, wird die Erweiterung von AcLaser erkannt und kann geladen werden. Alle geladenen Erweiterungen können in den Layout-Baum eingefügt werden.

Sämtliche konfigurierten Erweiterungen werden beim Start von AcLaser geladen und beim Beenden von AcLaser entladen. Die DLL kann also ein Objekt erzeugen, welches über die gesamte Laufzeit von AcLaser existiert. Dieses Objekt läuft über die gesamte Laufzeit von AcLaser im Hintergrund und kann unabhängig von der Ausführung im Layoutbaum Code abarbeiten. So kann ein externes Programm beispielsweise permanent im Hintergrund die Kommunikation mit einem Gerät überwachen. Der im Hintergrund abgearbeitete Code kann dabei völlig unabhängig von der Aufgabe des Programms im Layoutbaum sein.

4.2 Ansteuerung DLIP-Modul

Um die Strukturperiode des DLIP-Moduls ändern zu können, muss eine Kommunikation mit dessen Controller hergestellt werden. Dafür stellt das Gerät eine Schnittstelle bereit.

4.2.1 Schnittstelle zum Controller

Die Kommunikation mit dem Controller des DLIP-Moduls ist über eine RS232-Schnittstelle möglich. Dabei werden einzelne Befehle an den Controller gesendet. Danach wird auf eine Antwort gewartet. Die meisten Befehle sind dabei im Binärformat und bestehen aus einer Befehlsnummer und einem Parameterfeld im Gleitkommaformat.

Bei erfolgreicher Ausführung des Befehls ist die Antwort des Controllers identisch zum gesendeten Befehl. Wurde ein Befehl zur Abfrage von Parametern gesendet, so wird der Befehlsparameter vom Controller ignoriert. In der Antwort enthält das Parameterfeld den Wert des angefragten Parameters. Einige Befehle können nicht im Binärformat geschickt werden. Bei diesen Befehlen werden sowohl Befehlsname als auch Befehlsparameter in ASCII-Codierung an den Controller gesendet. Für den Betrieb wichtig sind dabei nur die Befehle zum Anfragen und Übermitteln der Konfiguration des Kopfes. Dabei wird im Parameterfeld der Inhalt der Konfigurationsdatei entweder als Befehl oder als Antwort im Klartext gesendet.

Die Achse des DLIP-Moduls muss nach jedem Einschalten referenziert werden. Es gibt allerdings keinen Befehl zur Ermittlung des Referenzierungsstatus. Daher muss das Plugin selbst den Referenzierungsstatus der Achse speichern.

Nach jedem Verbindungsabbruch geht das Plugin davon aus, dass die Achse neu referenziert werden muss.

Bei Änderung des optischen Setups, Austausch der Hardware oder nach einem Firmware-Update muss die Konfiguration erneut in den Controller des DLIP-Modulsgeschrieben werden. Diese kann nicht über Fusion-Interface übertragen werden. Aus diesem Grund muss das Plugin die Konfiguration vom Controller lesen und mit der gewünschten Konfiguration vergleichen. Wird ein Unterschied erkannt, soll die Referenzkonfiguration an den Controller übermittelt werden. Im täglichen Betrieb ist keine Änderung der Konfiguration zu erwarten.

4.2.2 Hintergrund-Thread zur Gerätekommunikation

Um während der gesamten Laufzeit von AcLaser die Kommunikation mit dem DLIP-Controller aufrechtzuerhalten, wird beim Laden der DLL ein Thread erzeugt, welcher erst beim Entladen des Plugins wieder beendet wird. Dieser Thread läuft also die ganze Zeit im Hintergrund. Er ist für jegliche Kommunikation mit dem Controller verantwortlich. Er initialisiert zu Beginn der Kommunikation die Schnittstelle, referenziert die DLIP-Achse und liest ihre Grenzwerte aus. Anschließend überwacht der Thread permanent die Verbindung zum Gerät. Bei Verbindungsabbruch (Gerät neu gestartet, Verbindungsfehler) wird die Initialisierung neu durchgeführt.

Somit wird sichergestellt, dass die zwischengespeicherten Werte im Thread immer mit dem Zustand des Controllers übereinstimmen.

Auch eine Funktion zum Anfahren einer bestimmten Strukturperiode ist im Thread vorhanden. Das ist die Funktion, die bei Ausführung des Plugins im Layout-Baum gestartet wird. Diese wird im Kontext des Ausführthreads in AcLaser aufgerufen. Der Zugriff auf Ressourcen muss also threadsicher ausgeführt werden. Um gleichzeitige Zugriffe auf Ressourcen von unterschiedlichen Threads zu verhindern, werden diese von einem Mutex geschützt. Wenn ein Thread auf eine Ressource zugreifen will, versucht er mit einem Timeout den Mutex anzufordern. Wenn dies gelingt, kann auf die Ressource zugegriffen werden. Der Mutex bleibt für diese Zeit gesperrt. Nachdem die Ressource gelesen oder geschrieben wurde, wird der Mutex wieder freigegeben. Ab dann können andere Threads wieder auf die Ressource zugreifen. Die Zeit, in welcher der Mutex vom Thread angefordert bleibt, sollte so kurz wie möglich sein. Ansonsten könnten andere Threads eine Zeitüberschreitung beim Warten auf den Mutex erhalten. Ist das Anfordern des Mutex erfolglos, weil ein anderer Thread die Ressource gerade verwendet, so muss die Aktion entweder später wiederholt werden oder zu einem Fehler führen.

4.2.3 Zustandsautomat

Für die Ansteuerung des DLIP-Controllers wird ein Zustandsautomat benutzt. Dieser besteht aus diskreten Zuständen, die jeweils mit verschiedenen Aktionen und Folgezuständen verbunden sind. Der Zustandsautomat ist Teil des Hintergrund-Threads, der beim Laden des Plugins erzeugt wurde. Der Hintergrund-Thread überprüft also die Bedingungen für den aktuellen Zustand. Ist eine Bedingung erfüllt, wird die zugehörige Aktion ausgeführt und der Zustand gewechselt. Die vollständige Struktur des Zustandsautomaten ist in Anlage 4 zu sehen. Zur Verdeutlichung der Funktionsweise werden Ausschnitte daraus in Tabelle 2 bis 5 dargestellt.

Tabelle 2: Verbindungsaufbau

Zustand	Bedingung	Aktion	Neuer Zustand
Offline	Port soll geöffnet sein und Port konnte geöffnet werden	keine	Init
	Port soll geöffnet sein und Port konnte nicht geöffnet werden	keine	Disconnect
	Port soll geschlossen sein	Port schließen	Offline
WaitForOnline	Wartezeit bis zum Neuverbinden abgelaufen	keine	Init
	Port soll geschlossen sein	Port schließen	Offline
Disconnect	Wartezeit für Disconnect abgelaufen	keine	Offline
	Port soll geschlossen sein	Port schließen	Offline
Init	Senden des Init-Befehls erfolgreich	keine	WaitInit
	Senden des Init-Befehls fehlgeschlagen	keine	Offline
	Port soll geschlossen sein	Port schließen	Offline
WaitInit	Antwort auf Init-Befehl erhalten	keine	Initialized
	Zeitüberschreitung beim Warten auf Antwort	keine	WaitForOnline
	Port soll geschlossen sein	Port schließen	Offline

Der Zustandsautomat beginnt beim Laden des Plugins im Zustand „Offline“. Ist die Bedingung „Öffnen der Schnittstelle erlaubt“ wahr, wird versucht, den Port zu öffnen. Ist dies erfolgreich, wechselt der Zustandsautomat vom Zustand "Offline" in den Zustand „Init“. Kann in folgenden Zuständen ein Befehl nicht gesendet werden, wird davon ausgegangen, dass ein Problem mit der seriellen Schnittstelle vorliegt. In solchen Fällen wechselt der Zustandsautomat immer in den Zustand „Offline“, um eine erneute Öffnung des Ports zu versuchen. Die einzige Ausnahme stellt dabei der Zustand „Offline“ selbst dar, wenn in diesem Zustand der Port nicht geöffnet werden kann, wird in den Zustand „Disconnect“ gewechselt. Dort wartet der Thread auf den Ablauf einer Wartezeit, bevor er wieder in den Zustand „Offline“ wechselt. Diese Wartezeit ist wichtig, damit in dem Fall, dass der Port nicht geöffnet werden kann, das Logfile nicht mit unzähligen Verbindungsaufbauversuchen überfüllt wird.

Im Zustand „Init“ wird der Initialisierungsbefehl versucht an den Controller zu senden. Ist das Senden erfolgreich, wird in „WaitForInit“ gewechselt. Dort wird für eine bestimmte Zeit auf die Antwort des Controllers gewartet. Erhält der Thread innerhalb der Wartezeit eine gültige Antwort, so wechselt er in den Zustand „Initialized“. Tritt in folgenden Zuständen ein Timeout beim Warten auf die Antwort des Controllers auf, wird davon ausgegangen, dass die Kommunikation unterbrochen wurde. Die Kommunikation könnte unterbrochen worden sein, weil der Controller abgeschaltet wurde, nicht mehr angeschlossen ist oder eine elektromagnetische Störung vorliegt. Da bei Abschaltung des Controllers eine erneute Initialisierung vorgenommen werden muss, wird immer von einem Neustart des Controllers ausgegangen und nach jedem Kommunikationsabbruch erneut initialisiert.

In dem Fall wird immer in den Zustand „WaitForOnline“ gewechselt, um den Controller erneut zu initialisieren. In diesem wartet der Thread, ähnlich wie in „Disconnect“ eine bestimmte Zeit ab, bevor er in den Zustand „Init“ wechselt, um den Controller neu zu initialisieren. Auch hier ist die Wartezeit wichtig, damit das Logfile, solange das Problem nicht behoben ist, nicht mit unzähligen Verbindungsaufbauversuchen überfüllt wird.

Tabelle 3: Konfigurationsvergleich und Referenzierung

Initialized	Konfigurationsdaten definiert	Konfiguration anfordern	WaitForConfig
	Konfigurationsdaten nicht definiert	keine	Home
	Port soll geschlossen sein	Port schließen	Offline
WaitForConfig	Konfiguration erhalten und diese entspricht Soll-Konfiguration	keine	Home
	Konfiguration erhalten und diese entspricht nicht Soll-Konfiguration	keine	DifferentConfig
	Zeitüberschreitung beim Warten auf Konfiguration	keine	WaitForOnline
	Port soll geschlossen sein	Port schließen	Offline
DifferentConfig	Senden der Konfiguration erfolgreich	keine	WaitConfigSent
	Senden der Konfiguration fehlgeschlagen	keine	Offline
	Port soll geschlossen sein	Port schließen	WaitForOnline
WaitConfigSent	Bestätigung der Konfiguration erhalten	keine	Home
	Zeitüberschreitung beim Warten auf Antwort	keine	WaitForOnline
	Port soll geschlossen sein	Port schließen	Offline
Home	Senden des Home-Befehls erfolgreich	keine	WaitHome
	Senden des Home-Befehls fehlgeschlagen	keine	Offline
	Port soll geschlossen sein	Port schließen	Offline
WaitHome	Antwort Homing erfolgreich erhalten	Wertzähler rücksetzen	AskValues
	Zeitüberschreitung beim Warten auf Antwort	keine	WaitForOnline
	Port soll geschlossen sein	Port schließen	Offline

Wurde der Controller erfolgreich initialisiert, wird überprüft, ob eine Vergleichskonfiguration auf dem PC vorhanden ist. Ist dies nicht der Fall, wird der Vergleich der Konfigurationen übersprungen und direkt in den Zustand „Home“ gewechselt. Liegt aber eine Vergleichskonfiguration vor, so wird der Befehl zur Anforderung der Konfiguration des Controllers gesendet und in den Zustand „WaitForConfig“ gewechselt.

In diesem Zustand wird auf die Antwort des Controllers gewartet und die erhaltene Konfiguration anschließend mit der Vergleichsdatei verglichen. Stimmen beide miteinander überein, wechselt der Zustandsautomat direkt in den Zustand „Home“. Sind die Konfigurationen allerdings nicht identisch, wird in den Zustand „DifferentConfig“ gewechselt. Dort wird der Befehl zum Überschreiben der Konfiguration zusammen mit der Vergleichsdatei des PCs gesendet und in „WaitConfigSend“ gewechselt. In diesem Zustand wird schlussendlich auf die Bestätigung des Controllers gewartet und anschließend in „Home“ gewechselt.

Im Zustand „Home“ wird der Befehl zur Referenzierung der Achse an den Controller gesendet und in den Zustand „WaitHome“ gewechselt. In diesem wird eine bestimmte Zeit auf die Bestätigung des Controllers gewartet und bei Erhalt der Antwort in den Zustand „AskValues“ gewechselt und der Wertzähler zurückgesetzt.

Tabelle 4: Abfragen der Grenzwerte

AskValues	Senden der Anfrage des Wertes entsprechend Zähler erfolgreich	keine	WaitValues
	Senden der Anfrage des Wertes entsprechend Zähler fehlgeschlagen	keine	Offline
	Port soll geschlossen sein	Port schließen	WaitForOnline
WaitValues	Antwort auf Werteanfrage erfolgreich erhalten und nicht letzter Wert	Wert speichern, Wertzähler erhöhen	AskValues
	Antwort auf Werteanfrage erfolgreich erhalten und letzter Wert	Wert speichern	Ready
	Zeitüberschreitung beim Warten auf Antwort	keine	WaitForOnline
	Port soll geschlossen sein	Port schließen	Offline

Im Zustand „AskValues“ werden die Grenzwerte der Strukturperioden für alle Strahlenanzahlen abgefragt. Um sechs verschiedene Werte mit nur zwei Zuständen nacheinander abzufragen, wird eine Variable verwendet, die als Wertzähler fungiert. Je nach Wert der Variable wird ein anderer Befehl an den Controller gesendet. Nach Senden des Befehls wird in den Zustand „WaitValues“ gewechselt.

Wenn mit dem angefragten Wert geantwortet wurde, wird der Wertzähler um 1 erhöht und wieder in „AskValues“ gewechselt, bis der Wert 6 erreicht. Dieser Wert bedeutet, dass alle sechs Grenzwerte abgefragt wurden. Dann wird in den Zustand „Ready“ gewechselt.

Da bei Verbindungsabbrüchen die Grenzwerte erneut abgefragt werden müssen, wird der Wertezähler nach der Referenzierung der Achse immer auf 1 gesetzt.

Tabelle 5: Bewegung und Verbindungskontrolle

Ready	Fahrbefehl aktiv und konnte gesendet werden	keine	WaitMove
	Fahrbefehl aktiv und konnte nicht gesendet werden	keine	Offline
	Wartezeit für Verbindungstest abgelaufen und Testbefehl konnte gesendet werden	keine	ConnectControl
	Wartezeit für Verbindungstest abgelaufen und Testbefehl konnte nicht gesendet werden	keine	Offline
	Port soll geschlossen sein	Port schließen	Offline
ConnectControl	Antwort Testbefehl erfolgreich erhalten	keine	Ready
	Zeitüberschreitung beim Warten auf Antwort	keine	WaitForOnline
	Port soll geschlossen sein	Port schließen	Offline
WaitMove	Antwort Fahren erfolgreich erhalten	keine	Ready
	Zeitüberschreitung beim Warten auf Antwort	keine	WaitForOnline
	Port soll geschlossen sein	Port schließen	Offline

Der Zustand „Ready“ ist der einzige Zustand, in dem Verfahrbefehle aus dem Hauptthread verarbeitet werden. Ist ein Fahrbefehl aktiv, wird dieser an den Controller gesendet und nach erfolgreichem Senden in den Zustand „WaitMove“ gewechselt. Der Controller antwortet auf einen Bewegungsbefehl erst, wenn die Bewegung abgeschlossen ist. Der Zustand „WaitMove“ wartet also eine bestimmte Zeit auf die Antwort des Controllers auf den Bewegungsbefehl. Wie in den anderen Zuständen auch, werden damit Verbindungsabbrüche erkannt.

Außerdem wird damit das Senden eines weiteren Verfahrbefehls unterdrückt, solange sich die Achse noch in Bewegung befindet. Erst nach Erhalt der Antwort wechselt der Zustandsautomat zurück in den Zustand „Ready“.

Befindet sich der Zustandsautomat eine Zeit lang im Zustand „Ready“, ohne dass ein Verfahrbefehl aktiv ist, so wird ein Testbefehl an den Controller gesendet, um bei längerer Inaktivität die Verbindung zum Controller zu testen. Anschließend wird in den Zustand „ConnectControl“ gewechselt. In diesem Zustand wird auf die Antwort des Controllers auf den Testbefehl gewartet. Wird der Befehl innerhalb der festgelegten Wartezeit erhalten, wird wieder in den Zustand „Ready“ gewechselt. Tritt beim Warten auf die Antwort eine Zeitüberschreitung auf, wechselt der Zustandsautomat, wie bei allen Wartezuständen, in den Zustand „WaitForOnline“, um den Controller erneut zu initialisieren.

In allen Zuständen des Automats wird geprüft, ob die Verbindung noch geöffnet sein darf. Ist das nicht der Fall, so wird der Port geschlossen und in den Zustand „Offline“ gewechselt. Das Schließen der Verbindung wird beispielsweise beim Finalisieren der Bibliothek angefordert.

4.2.4 Absolutes und relatives Fahren

Im DLIP-Plugin soll die Auswahl der Strukturperiode wahlweise absolut oder relativ angegeben werden. Um zu vermeiden, dass ein Fahrbefehl an den Controller gesendet wird, der über die Grenzen der Strukturperioden hinausgeht, prüft das Plugin, ob die Zielposition innerhalb des gültigen Bereichs liegt. Dazu muss der absolute Wert bekannt sein. Dieser wird einfach aus der letzten angefahrenen Strukturperiode und dem relativen Versatz berechnet. Da der letzte angefahrene Wert vom Controller nicht gelesen werden kann, speichert das Plugin diesen Wert nach jeder erfolgreichen Bewegung ab. Dies hat zur Folge, dass relative Bewegungen erst nach mindestens einer absoluten Bewegung möglich sind.

Wie in Kapitel 3.8 ausgeführt, wird auch für die Berechnung der Z-Achskorrektur sowie der Auswahl der besten Korrekturdatei die absolute Strukturperiode benötigt.

4.2.5 Protokollierung

Sämtliche Zustandswechsel, Warnungen und Fehlermeldungen des Plugins können in einem dafür angelegten Logfile gespeichert werden (siehe Anlage 5 und 6). Je nach Vorliebe des Anwenders kann zwischen vier Log-Stufen entschieden werden.

Bei Logstufe 1 werden nur Fehlermeldungen ins Logfile geschrieben, Warnungen und Informationen zu Zustandsübergängen werden ignoriert. Bei Logstufe 2 werden zusätzlich zu den Fehlermeldungen auch Warnhinweise gespeichert. Bei Logstufe 3 werden alle Fehlermeldungen, Warnhinweise und Zustandsänderungen aufgezeichnet. Soll kein Logfile erstellt werden, so wird Logstufe 0 ausgewählt.

Durch die Protokollierung sind Fehleranalysen an bereits ausgelieferten Maschinen leicht durchführbar. Die Fehlerquelle kann aus dem Logfile ermittelt werden. Ist beispielsweise die serielle Schnittstelle falsch konfiguriert, so wird eine entsprechende Fehlermeldung beim Öffnen der Schnittstelle im Protokoll zu finden sein. Ist das DLIP-Modul ausgeschalten oder das Kabel nicht angeschlossen, finden sich im Protokoll Kommandos mit Zeitüberschreitungsfehler. Durch Protokollierung der ausgelesenen Grenzwerte können auch Strukturperioden außerhalb des gültigen Bereichs erkannt werden.

4.2.6 Probebetrieb

Das Plugin wurde auf seine Funktion sowie seine Fehlerresistenz im Probebetrieb getestet. Die Öffnung des Ports, Initialisierung des Controllers, Referenzierung der Achse sowie regelmäßige Verbindungskontrolle im „Ready“-Zustand wurden alle wie erwartet ausgeführt. Bewegungsbefehle aus dem Haupt-Thread werden nur im Zustand „Ready“ verarbeitet und an den Controller gesendet.

Auf getestete Störungen der Kommunikation reagierte das Plugin wie erwartet. Wurde die Kommunikation mit dem geöffneten Port durch Abziehen des USB-zu-seriell-Konverters gestört, so wurde in den Zustand „Offline“ gewechselt und eine erneute Öffnung des Ports versucht. Nachdem der Konverter wieder angesteckt wurde, erfolgte ein neuer Verbindungsaufbau mit automatischer Initialisierung (inklusive Referenzierung, Lesen der Grenzwerte, …).

Wurde der Controller an irgendeiner Stelle während des Betriebs ausgeschaltet, wechselte das Plugin in den Zustand „Init“ und sendete regelmäßig den Initialisierungsbefehl, bis der Controller wieder angeschaltet war. Das gleiche Verhalten zeigte sich durch Abziehen und Anstecken des Verbindungskabels.

Ein Kommunikations-/Verbindungsabbruch führte also nicht zu einem Absturz des Plugins oder gar AcLaser, sondern wurde im Hintergrund behandelt.

Sämtliche Zustandsänderungen bzw. Fehlerbehandlungen wurden im Protokoll mit Logstufe 3 festgehalten. Das korrekte Verhalten des Zustandsautomats konnte so nachgewiesen werden.

4.3 Ansteuerung der Z-Achse

Da die Fokusverschiebung im untersuchten Wertebereich eine lineare Abhängigkeit von der Strukturperiode aufweist (siehe Kapitel 3.8), wird die Fokusverschiebung anhand von zwei Stützstellen berechnet. Diese werden in der Windows-Registrierdatenbank (ab hier Registry) gespeichert und bestehen jeweils aus einer Strukturperiode und der dazugehörigen Fokusposition relativ zum eigentlichen Fokus. Aus den beiden Wertepaaren berechnet das Plugin die Korrekturgerade der Form

$$\Delta z = m \cdot P + n \tag{7}$$

Dabei ist Δz die Fokusposition relativ zum konfigurierten Fokus beim Einrichten der Anlage und P die Strukturperiode, analog zu Gleichung 6 (Kapitel 3.8). Wenn das Plugin einen Verfahrbefehl verarbeitet, wird parallel zum Fahren des Controllers die Fokusposition berechnet und der entsprechende Verfahrbefehl an die Z-Achse der Maschine gesendet. Die Z-Achse der Maschine und die Achse des DLIP-Moduls fahren somit gleichzeitig. Das Plugin wartet in seiner Ausführfunktion auf das Erreichen der Zielposition beider Achsen. Schlägt das Verfahren der DLIP-Achse fehl, so wird die Bewegung der Z-Achse abgebrochen.

Für die Z-Achskorrektur gibt es drei Möglichkeiten. Die Korrektur kann deaktiviert werden, beispielsweise um die Stützstellen für die Korrekturfunktion zu ermitteln. Ist die Korrektur aktiviert, kann ausgewählt werden, ob die Teilehöhe berücksichtigt werden soll oder nicht.

4.4 Feldkorrektur

Wie in Kapitel 3.8 beschrieben, verändert sich je nach angefahrener Strukturperiode die Feldverzeichnung. Um diese für verschiedene Strukturperioden zu korrigieren, müssen verschiedene Korrekturtabellen auf die Bewegungsbefehle der RTC-Karte angewendet werden.

Dafür besitzt die RTC acht Korrekturslots, die jeweils eine Korrekturtabelle enthalten. Die Korrekturtabellen werden beim Einrichten der Maschine ermittelt und in die Konfiguration von AcLaser eingetragen. AcLaser lädt die Konfigurationen beim Start in die entsprechenden Slots der RTC. Das Plugin speichert, welche Strukturperiode beim Erzeugen der Korrekturtabelle der jeweiligen Slots verwendet wurde. Wird ein Bewegungsbefehl verarbeitet, sendet das Plugin nicht nur einen Bewegungsbefehl an die DLIP- und Z-Achsen, sondern ermittelt auch, welcher Korrekturslot die Feldverzeichnung bei der angefahrenen Strukturperiode am besten korrigiert. Dafür vergleicht es die anzufahrende Strukturperiode mit den Strukturperioden, die zu jedem Slot gehören. Der Korrekturslot, dessen Strukturperiode die kleinste Differenz zur angefahrenen besitzt, wird vom Plugin als der Beste ausgewählt. Für die Suche werden nur die Korrekturslots verwendet, für welche in der Registry eine gültige Strukturperiode zugeordnet wurde. Anschließend wird ein Befehl an die RTC geschickt, um den ermittelten Korrekturslot für folgende Grafikbefehle festzulegen. Um die Feldverzerrung aufzunehmen bzw. Korrekturtabellen zu ermitteln, kann die automatische Korrektur der Feldverzeichnung im Parameterpanel deaktiviert werden.

Erst wenn die Bewegung der DLIP-Achse, die Korrektur der Fokusverschiebung und das Laden des besten Korrekturslots abgeschlossen sind, wird die Ausführ-Funktion des Plugins beendet. Da die Ausführung im Markiermodul auf das Ende der Funktion wartet, befinden sich alle Achsen an der korrekten Position, wenn die nächsten Grafikbefehle an die RTC geendet werden.

4.5 Plugin-Steuerelemente

Alle Parameter, die vom Anwender einstellbar sein sollen, sind im Parameter-Panel des Plugins steuerbar.

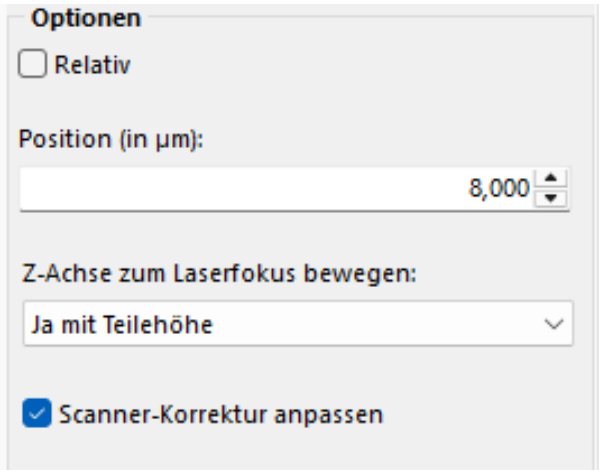

Abbildung 15: Panel des Plugins

Um das relative und absolute Verfahren einzustellen, gibt es eine Checkbox mit der Bezeichnung „Relativ“. Wird diese angeklickt, wird das relative Fahren aktiviert, wenn es nicht angeklickt ist, wird absolut gefahren.

Für die Einstellung der angefahrenen Strukturperiode (bzw. der Strukturperioden-änderung im Fall des relativen Fahrens) ist ein SpinEdit mit der Überschrift „Position (in µm):“ vorhanden. Vor dem Start der Bewegung wird die absolute Strukturperiode mit den Grenzen der konfigurierten Anzahl Strahlen validiert. Wird der gültige Bereich überschritten, so wird der Fahrbefehl abgelehnt.

Auch die Korrektur der Fokusverschiebung sowie Feldverzerrung sind im Panel aktivierbar. Für die Korrektur der Fokusverschiebung ist eine ComboBox vorhanden. Die drei Auswahlmöglichkeiten sind: keine Korrektur, Korrektur unter Berücksichtigung der Teilehöhe und Korrektur ohne Berücksichtigung der Teilehöhe. Für die Korrektur der Feldverzeichnung ist eine weitere Checkbox mit dem Label „Scanner-Korrektur anpassen“ vorhanden. Wird die Checkbox geklickt (standardmäßig der Fall), so wird die Feldkorrektur automatisch vorgenommen. Ist die CheckBox nicht angeklickt, geschieht dies nicht.

5. Zusammenfassung und Ausblick

Im Rahmen dieses Bachelorprojekts wurde erfolgreich ein Plugin zur Ansteuerung des DLIP-Moduls programmiert. Die Nachteile und Fehleranfälligkeit der manuellen Bedienung mit Fusion Interface wurden vollständig beseitigt.

Das Plugin übernimmt die Initialisierung des Kopfes und die Referenzierung der DLIP-Achse im Hintergrund. Die Zeit, die der Bediener benötigt, das zu fertigende Layout auszuwählen, genügt in der Regel, um im Hintergrund die Verbindung zum DLIP-Controller aufzubauen und diesen zu initialisieren bzw. zu referenzieren. Somit kann das Plugin und damit das DLIP-Modul sofort nach dem Laden des Layouts verwendet werden. Im Fehlerfall wird der Wiederaufbau der Kommunikation im Hintergrund vorgenommen. Dazu sind keine manuellen Arbeitsschritte des Bedieners erforderlich.

Die für die Validierung der anzufahrenden Strukturperiode verwendeten Grenzen werden automatisch ausgelesen. Damit werden Fehlpositionierungen verhindert.

Der mathematische Zusammenhang von Fokusverschiebung und Strukturperiode erlaubt die automatische Korrektur über den gesamten Arbeitsbereich des DLIP-Moduls. Die Feldverzeichnung kann mit der verwendeten Hardware für acht feste Strukturperioden korrigiert werden. Andere Werte verursachen weiterhin optische Fehler. Diese könnten zukünftig genauer untersucht werden. Falls ein mathematischer Zusammenhang zwischen Feldverzeichnung und Strukturperiode ermittelt werden kann, könnte die Korrektur auf den gesamten Arbeitsbereich ausgedehnt werden.

Im Rahmen dieses Bachelorprojekts wurde die Bearbeitung eines Werkstücks mit DLIP von einer Machbarkeitsstudie hin zu einer echten Kundenlösung entwickelt. Die Software hat Kundentauglichkeit erreicht und kann in einer zukünftigen Anlage sofort verwendet werden.

Literaturverzeichnis

[1] ACSYS Lasertechnik GmbH: Lasersysteme - Made in Germany. Unter: https://acsyslaser.com/, Zugriff am 2025-08-27, 13:40.

[2] ACSYS Lasertechnik GmbH: Laser-Verfahren: Strukturieren. Unter: https://acsyslaser.com/laser-verfahren/strukturieren/, Zugriff am 2025-08-27, 13:50

[3] Hauschwitz, P.; D. Jochcová; R. Jagdeesh; D. Rostohar; J. Brajer; J. Kopeček; M. Cimrman; M. Smrž; T. Mocek; A. Lucianetti: Towards rapid large-scale LIPSS fabrication by 4-beam ps DLIP. In: Optics & Laser Technology Volume 133 (2021), Nr. 14

[4] SurFunction GmbH: ELIPSYS & Direct Laser Interference Patterning. Unter https://surfunction.com/en/technology/laser_interference_dlip/, Zugriff am 2025-09-02, 10:33

[5] Sill Optics GmbH: f-Theta Objektive. Unter https://www.silloptics.de/produkte/sill-technikon/laser-optik/f-theta-objektive, Zugriff am 2025-09-01, 13:12

[6] Scanlab GmbH: Glossar: F-Theta-Objektiv. Unter https://www.scanlab.de/de/service/glossar/f-theta-objektiv, Zugriff am 2025-08-29, 16:10

[7] Scanlab GmbH, Installation und Inbetriebnahme RTC6 PCIe-Karte, RTC6 Ethernet-Karte, Revision 1.1.3 de-DE, 2025.

[8] Pedrotti, F.; L. Pedrotti; W. Bausch; H-Schmidt: Optik für Ingenieure; Grundlagen. 3. Auflage. Berlin, Heidelberg: Springer, 2005.

[9] Haupt, K.: Doppelspalt, einfaches Modell. Unter https://physikkursq3lichtundquanten.blogspot.com/p/doppelspalt.html, Zugriff am 2025-09-02, 11:25

[10] Hofmann, O.; J. Stollenwerk; P. Loosen: Design of multi-beam optics for high throughput parallel processing. In Journal of Laser Applications 32 (2019), Nr. 1, Seiten 012005-1 bis 021005-8

[11] Internes Dokument der Firma Fusion Bionic, klassifiziert.

[12] Khatsevich, T. B. A.: Telecentric F-Theta-Lenses for Scanning Systems. In Optoelectronics, Instrumentation and Data Processing 58 (2022), Nr. 3, S. 241-249.

[13] Baumann, R.; T. Rauscher; C.I. Bernäcker; C.Zwahr; T. Weißgärber; L. Röntzsch, A. F. Lasagni: Laser Structuring of Open Cell Metal Foams for Micro Scale Surface Enlargment. In JLMN-Journal of Laser micro/Nanoengineering 15 (2020), Nr. 2, S. 132-138

Verzeichnis der Anlagen

Anlage 1: Positionsanalyse des Testrasters bei 6 µm

Machine Serial Number: SKL2252417 Page: 1

Ablauf-Name	Los Nr.#	Datum & Zeit
25x25_0°_KT.RTN		2 Thu Jul 3 11:25:31 2025

Merkmal	Einheit	Nennwert	Istwert	Toleranzen	Abweichung	Abw/Graf
Schritt 7						
X Position	mm		-012.47652			
Y Position	mm		-012.33817			
Schritt 8						
X Position	mm		-006.27154			
Y Position	mm		-012.35077			
Schritt 9						
X Position	mm		-000.05002			
Y Position	mm		-012.36753			
Schritt 10						
X Position	mm		+006.16810			
Y Position	mm		-012.40989			
Schritt 11						
X Position	mm		+012.37900			
Y Position	mm		-012.44291			
Schritt 12						
X Position	mm		-012.45245			
Y Position	mm		-006.14742			
Schritt 13						
X Position	mm		-006.24160			
Y Position	mm		-006.17489			
Schritt 14						
X Position	mm		-000.02770			
Y Position	mm		-006.20734			
Schritt 15						

X Position	mm	+006.18970
Y Position	mm	-006.21870

Schritt 16

X Position	mm	+012.40039
Y Position	mm	-006.24222

Schritt 17

X Position	mm	-012.44166
Y Position	mm	+000.03400

Schritt 18

X Position	mm	-006.20257
Y Position	mm	+000.01771

Merkmal	Einheit	Nennwert	Istwert	Toleranzen	Abweichung	Abw/Graf
Schritt 19						
X Position	mm		+000.00049			
Y Position	mm		+000.00051			
Schritt 20						
X Position	mm		+006.22449			
Y Position	mm		-000.01419			
Schritt 21						
X Position	mm		+012.42705			
Y Position	mm		-000.05292			
Schritt 22						
X Position	mm		-012.38768			
Y Position	mm		+006.24997			
Schritt 23						
X Position	mm		-006.16855			
Y Position	mm		+006.23221			
Schritt 24						
X Position	mm		+000.04339			
Y Position	mm		+006.19971			
Schritt 25						
X Position	mm		+006.23826			
Y Position	mm		+006.16981			
Schritt 26						
X Position	mm		+012.43780			
Y Position	mm		+006.13251			
Schritt 27						
X Position	mm		-012.36127			
Y Position	mm		+012.40575			
Schritt 28						
X Position	mm		-006.14407			

```
Y Position    mm            +012.38495

Schritt 29
X Position    mm            +000.07353
Y Position    mm            +012.36759

Schritt 30
X Position    mm            +006.27338
Y Position    mm            +012.33931

Schritt 31
X Position    mm            +012.47670
Y Position    mm            +012.31210
--------------------------------------------------------------------------------
================================================================================
```

Anlage 2: Positionsanalyse des Testrasters bei 18 µm

Machine Serial Number: SKL2252417 Page: 1

Ablauf-Name	Los Nr.#	Datum & Zeit
25x25_0°_KT.RTN		1 Thu Jul 3 11:22:02 2025

Merkmal	Einheit	Nennwert	Istwert	Toleranzen	Abweichung	Abw/Graf
Schritt 7						
X Position	mm		-012.55753			
Y Position	mm		-012.25113			
Schritt 8						
X Position	mm		-006.29973			
Y Position	mm		-012.27601			
Schritt 9						
X Position	mm		-000.05300			
Y Position	mm		-012.30415			
Schritt 10						
X Position	mm		+006.19881			
Y Position	mm		-012.32890			
Schritt 11						
X Position	mm		+012.42309			
Y Position	mm		-012.35454			
Schritt 12						
X Position	mm		-012.51925			
Y Position	mm		-006.09606			
Schritt 13						
X Position	mm		-006.27463			
Y Position	mm		-006.13016			
Schritt 14						
X Position	mm		-000.02284			
Y Position	mm		-006.15641			

Schritt 15		
X Position	mm	+006.20388
Y Position	mm	-006.18560
Schritt 16		
X Position	mm	+012.44916
Y Position	mm	-006.21221
Schritt 17		
X Position	mm	-012.48000
Y Position	mm	+000.05748
Schritt 18		
X Position	mm	-006.23754
Y Position	mm	+000.01543

Merkmal	Einheit	Nennwert	Istwert	Toleranzen	Abweichung	Abw/Graf
Schritt 19						
X Position	mm		+000.00002			
Y Position	mm		+000.00100			
Schritt 20						
X Position	mm		+006.24327			
Y Position	mm		-000.02806			
Schritt 21						
X Position	mm		+012.48602			
Y Position	mm		-000.05661			
Schritt 22						
X Position	mm		-012.43904			
Y Position	mm		+006.19798			
Schritt 23						
X Position	mm		-006.20024			
Y Position	mm		+006.17052			
Schritt 24						
X Position	mm		+000.04556			
Y Position	mm		+006.14340			
Schritt 25						
X Position	mm		+006.27822			
Y Position	mm		+006.10957			
Schritt 26						
X Position	mm		+012.51407			
Y Position	mm		+006.08467			
Schritt 27						
X Position	mm		-012.38916			
Y Position	mm		+012.33429			
Schritt 28						
X Position	mm		-006.16197			

```
Y Position    mm            +012.30158

Schritt 29
X Position    mm            +000.08399
Y Position    mm            +012.28096

Schritt 30
X Position    mm            +006.30224
Y Position    mm            +012.22794

Schritt 31
X Position    mm            +012.53448
Y Position    mm            +012.19712
--------------------------------------------------------------------------------
================================================================================
```

Anlage 3: Deklarationen der Plugin-Funktionen

```
type
  // functions for tree plugins
  // calling convention is stdcall

  // function to get information about external code
  TAcExternalCodeEventGetInfoA = procedure(Info : PAcExternalCodeInfo); stdcall;
  TAcExternalCodeEventGetInfoW = procedure(Info : PAcExternalCodeInfoW); stdcall;

  // function to get capabilities of external code
  // fills structure of type PAcExternalCodeCapabilities in version 1
  TAcExternalCodeEventGetCapabilities = function(RequestedVersion : Integer;
          Capabilities : Pointer) : Boolean; stdcall;

  // function to set new size of area for configuration window
  TAcExternalCodeEventSetParamPanelOptionsA = procedure(ParentWindow : THandle;
          Visible : Boolean; Width : Integer; Height : Integer; Language :
          PAnsiChar); stdcall;

  TAcExternalCodeEventSetParamPanelOptionsW = procedure(ParentWindow : THandle;
          Visible : Boolean; Width : Integer; Height : Integer; Language :
          PWideChar); stdcall;

  // function to request first and last control of frame
  TAcExternalCodeEventGetParamPanelControls = procedure(FirstControl : PHandle;
          LastControl : PHandle; MainHandle : PHandle); stdcall;

  // function to set new properties of spinedits
  TAcExternalCodeEventSetSpinEditProperties = procedure(Decimals : Byte;
          RemoveTrailingZeros : Boolean); stdcall;

  // function to set callback for onchange event
  TAcExternalCodeEventSetCallbackOnChange = procedure(lParam : AcNativeInt;
          Callback : TAcExternalCodeEventOnChanged); stdcall;

  // function to set options
  TAcExternalCodeEventSetOptions = function(Buffer : Pointer; BufferSize :
          Integer) : Boolean; stdcall;

  // function to get options
  TAcExternalCodeEventGetOptions = function(Buffer : Pointer; BufferSize :
          Integer; UsedBytes : PInteger) : Boolean; stdcall;

  // function to get default options
  TAcExternalCodeEventGetDefaultOptions = function(Buffer : Pointer; BufferSize :
          Integer; UsedBytes : PInteger) : Boolean; stdcall;

  // function to set group and object index and execution mode
  // is called directly before execution
  TAcExternalCodeEventSetExecutionDetails = procedure(GroupIndex : Integer;
          ObjectIndex : Integer; ExecutionMode : TAcExternalCodeExecutionMode);
          stdcall;

  // function to execute code
  TAcExternalCodeEventMain = function : Boolean; stdcall;

  // main function in script with error handling
  TAcExternalCodeEventMainExA = function(const ErrorMessage : PAnsiChar; const
          MaxChars : DWord) : Boolean; stdcall;
  TAcExternalCodeEventMainExW = function(const ErrorMessage : PWideChar; const
          MaxChars : DWord) : Boolean; stdcall;
```

Anlage 4: Zustandsautomat

Zustand	Bedingung	Aktion	Neuer Zustand
Offline	Port soll geöffnet sein und Port konnte geöffnet werden	keine	Init
	Port soll geöffnet sein und Port konnte nicht geöffnet werden	keine	Disconnect
	Port soll geschlossen sein	Port schließen	Offline
WaitForOnline	Wartezeit bis zum Neuverbinden abgelaufen	keine	Init
	Port soll geschlossen sein	Port schließen	Offline
Disconnect	Wartezeit für Disconnect abgelaufen	keine	Offline
	Port soll geschlossen sein	Port schließen	Offline
Init	Senden des Init-Befehls erfolgreich	keine	WaitInit
	Senden des Init-Befehls fehlgeschlagen	keine	Offline
	Port soll geschlossen sein	Port schließen	Offline
WaitInit	Antwort auf Init-Befehl erhalten	keine	Initialized
	Zeitüberschreitung beim Warten auf Antwort	keine	WaitForOnline
	Port soll geschlossen sein	Port schließen	Offline
Initialized	Konfigurationsdaten definiert	Konfiguration anfordern	WaitForConfig
	Konfigurationsdaten nicht definiert	keine	Home
	Port soll geschlossen sein	Port schließen	Offline
WaitForConfig	Konfiguration erhalten und diese entspricht Soll-Konfiguration	keine	Home
	Konfiguration erhalten und diese entspricht nicht Soll-Konfiguration	keine	DifferentConfig
	Zeitüberschreitung beim Warten auf Konfiguration	keine	WaitForOnline
	Port soll geschlossen sein	Port schließen	Offline
DifferentConfig	Senden der Konfiguration erfolgreich	keine	WaitConfigSent
	Senden der Konfiguration fehlgeschlagen	keine	Offline
	Port soll geschlossen sein	Port schließen	WaitForOnline
WaitConfigSent	Bestätigung der Konfiguration erhalten	keine	Home

	Zeitüberschreitung beim Warten auf Antwort	keine	WaitForOnline
	Port soll geschlossen sein	Port schließen	Offline
Home	Senden des Home-Befehls erfolgreich	keine	WaitHome
	Senden des Home-Befehls fehlgeschlagen	keine	Offline
	Port soll geschlossen sein	Port schließen	Offline
WaitHome	Antwort Homing erfolgreich erhalten	Wertzähler rücksetzen	AskValues
	Zeitüberschreitung beim Warten auf Antwort	keine	WaitForOnline
	Port soll geschlossen sein	Port schließen	Offline
AskValues	Senden der Anfrage des Wertes entsprechend Zähler erfolgreich	keine	WaitValues
	Senden der Anfrage des Wertes entsprechend Zähler fehlgeschlagen	keine	Offline
	Port soll geschlossen sein	Port schließen	WaitForOnline
WaitValues	Antwort auf Werteanfrage erfolgreich erhalten und nicht letzter Wert	Wert speichern, Wertzähler erhöhen	AskValues
	Antwort auf Werteanfrage erfolgreich erhalten und letzter Wert	Wert speichern	Ready
	Zeitüberschreitung beim Warten auf Antwort	keine	WaitForOnline
	Port soll geschlossen sein	Port schließen	Offline
Ready	Fahrbefehl aktiv und konnte gesendet werden	keine	WaitMove
	Fahrbefehl aktiv und konnte nicht gesendet werden	keine	Offline
	Wartezeit für Verbindungstest abgelaufen und Testbefehl konnte gesendet werden	keine	ConnectControl
	Wartezeit für Verbindungstest abgelaufen und Testbefehl konnte nicht gesendet werden	keine	Offline
	Port soll geschlossen sein	Port schließen	Offline
ConnectControl	Antwort Testbefehl erfolgreich erhalten	keine	Ready
	Zeitüberschreitung beim Warten auf Antwort	keine	WaitForOnline
	Port soll geschlossen sein	Port schließen	Offline

WaitMove	Antwort Fahren erfolgreich erhalten	keine	Ready
	Zeitüberschreitung beim Warten auf Antwort	keine	WaitForOnline
	Port soll geschlossen sein	Port schließen	Offline

Anlage 5: Beispiel Logfile

```
14.09.2025 16:01:50.323 ------------------------------------------------------------
14.09.2025 16:01:50.323 logfile AcDLIPInterface#1.log opened (OPEN_ALWAYS).
14.09.2025 16:01:50.323 ------------------------------------------------------------
14.09.2025 16:01:51.113 HINT Opened serial port [4]
14.09.2025 16:01:51.113 HINT State changed from [Offline] to [Init]
14.09.2025 16:01:51.119 HINT Sending binary command: Command=[1] Value=[0,000]
14.09.2025 16:01:51.119 HINT State changed from [Init] to [WaitInit]
14.09.2025 16:01:53.277 HINT Received Data: #1 #162 #140 #230 ?
14.09.2025 16:01:53.921 HINT Received binary command: Command=[1] Value=[1,801]
14.09.2025 16:01:53.921 HINT State changed from [WaitInit] to [Initialized]
14.09.2025 16:01:53.928 HINT State changed from [Initialized] to [Home]
14.09.2025 16:01:53.934 HINT Sending binary command: Command=[2] Value=[0,000]
14.09.2025 16:01:53.934 HINT State changed from [Home] to [WaitHome]
14.09.2025 16:01:54.597 HINT Received Data: #2 #0 #0 #0 #0
14.09.2025 16:01:55.167 HINT Received binary command: Command=[2] Value=[0,000]
14.09.2025 16:01:55.167 HINT State changed from [WaitHome] to [AskValues]
14.09.2025 16:01:55.382 HINT Sending binary command: Command=[62] Value=[0,000]
14.09.2025 16:01:55.537 HINT State changed from [AskValues] to [WaitValues]
14.09.2025 16:01:55.548 HINT Received Data: >#0 #0 `@
14.09.2025 16:01:56.061 HINT Received binary command: Command=[62] Value=[3,500]
14.09.2025 16:01:56.061 HINT State changed from [WaitValues] to [AskValues]
14.09.2025 16:01:56.067 HINT Sending binary command: Command=[63] Value=[0,000]
14.09.2025 16:01:56.067 HINT State changed from [AskValues] to [WaitValues]
14.09.2025 16:01:56.085 HINT Received Data: ?#0 #0 PA
14.09.2025 16:01:56.588 HINT Received binary command: Command=[63] Value=[13,000]
14.09.2025 16:01:56.588 HINT State changed from [WaitValues] to [AskValues]
14.09.2025 16:01:56.595 HINT Sending binary command: Command=[64] Value=[0,000]
14.09.2025 16:01:56.595 HINT State changed from [AskValues] to [WaitValues]
14.09.2025 16:01:56.601 HINT Received Data: @#0 #0 #160 @
14.09.2025 16:01:57.106 HINT Received binary command: Command=[64] Value=[5,000]
14.09.2025 16:01:57.106 HINT State changed from [WaitValues] to [AskValues]
14.09.2025 16:01:57.112 HINT Sending binary command: Command=[65] Value=[0,000]
14.09.2025 16:01:57.112 HINT State changed from [AskValues] to [WaitValues]
14.09.2025 16:01:57.125 HINT Received Data: A#0 #0 #144 A
14.09.2025 16:01:57.634 HINT Received binary command: Command=[65] Value=[18,000]
14.09.2025 16:01:57.634 HINT State changed from [WaitValues] to [AskValues]
```

14.09.2025 16:01:57.640 HINT Sending binary command: Command=[66] Value=[0,000]
14.09.2025 16:01:57.640 HINT State changed from [AskValues] to [WaitValues]
14.09.2025 16:01:57.653 HINT Received Data: B#0 #0 #160 @
14.09.2025 16:01:58.168 HINT Received binary command: Command=[66] Value=[5,000]
14.09.2025 16:01:58.168 HINT State changed from [WaitValues] to [AskValues]
14.09.2025 16:01:58.174 HINT Sending binary command: Command=[67] Value=[0,000]
14.09.2025 16:01:58.174 HINT State changed from [AskValues] to [WaitValues]
14.09.2025 16:01:58.186 HINT Received Data: C#0 #0 #144 A
14.09.2025 16:01:58.699 HINT Received binary command: Command=[67] Value=[18,000]
14.09.2025 16:01:58.699 HINT State changed from [WaitValues] to [Ready]
14.09.2025 16:01:59.714 HINT Sending binary command: Command=[62] Value=[0,000]
14.09.2025 16:01:59.714 HINT State changed from [Ready] to [ConnectControl]
14.09.2025 16:01:59.727 HINT Received Data: >#0 #0 `@
14.09.2025 16:02:00.245 HINT Received binary command: Command=[62] Value=[3,500]
14.09.2025 16:02:00.245 HINT State changed from [ConnectControl] to [Ready]
14.09.2025 16:02:01.259 HINT Sending binary command: Command=[62] Value=[0,000]
14.09.2025 16:02:01.259 HINT State changed from [Ready] to [ConnectControl]
14.09.2025 16:02:01.273 HINT Received Data: >#0 #0 `@
14.09.2025 16:02:01.776 HINT Received binary command: Command=[62] Value=[3,500]
14.09.2025 16:02:01.776 HINT State changed from [ConnectControl] to [Ready]
14.09.2025 16:02:02.793 HINT Sending binary command: Command=[62] Value=[0,000]
14.09.2025 16:02:02.793 HINT State changed from [Ready] to [ConnectControl]
14.09.2025 16:02:02.806 HINT Received Data: >#0 #0 `@
14.09.2025 16:02:03.322 HINT Received binary command: Command=[62] Value=[3,500]
14.09.2025 16:02:03.322 HINT State changed from [ConnectControl] to [Ready]
14.09.2025 16:02:04.338 HINT Sending binary command: Command=[62] Value=[0,000]
14.09.2025 16:02:04.338 HINT State changed from [Ready] to [ConnectControl]
14.09.2025 16:02:04.344 HINT Received Data: >#0 #0 `@
14.09.2025 16:02:04.853 HINT Received binary command: Command=[62] Value=[3,500]
14.09.2025 16:02:04.853 HINT State changed from [ConnectControl] to [Ready]
14.09.2025 16:02:05.868 HINT Sending binary command: Command=[62] Value=[0,000]
14.09.2025 16:02:05.868 HINT State changed from [Ready] to [ConnectControl]
14.09.2025 16:02:05.875 HINT Received Data: >#0 #0 `@
14.09.2025 16:02:06.387 HINT Received binary command: Command=[62] Value=[3,500]
14.09.2025 16:02:06.387 HINT State changed from [ConnectControl] to [Ready]
14.09.2025 16:02:07.403 HINT Sending binary command: Command=[62] Value=[0,000]
14.09.2025 16:02:07.403 HINT State changed from [Ready] to [ConnectControl]
14.09.2025 16:02:07.409 HINT Received Data: >#0 #0 `@

14.09.2025 16:02:07.920 HINT Received binary command: Command=[62] Value=[3,500]
14.09.2025 16:02:07.920 HINT State changed from [ConnectControl] to [Ready]
14.09.2025 16:02:08.932 HINT Sending binary command: Command=[62] Value=[0,000]
14.09.2025 16:02:08.932 HINT State changed from [Ready] to [ConnectControl]
14.09.2025 16:02:08.945 HINT Received Data: >#0 #0 `@
14.09.2025 16:02:09.452 HINT Received binary command: Command=[62] Value=[3,500]
14.09.2025 16:02:09.452 HINT State changed from [ConnectControl] to [Ready]
14.09.2025 16:02:10.464 HINT Sending binary command: Command=[62] Value=[0,000]
14.09.2025 16:02:10.464 HINT State changed from [Ready] to [ConnectControl]
14.09.2025 16:02:10.477 HINT Received Data: >#0 #0 `@
14.09.2025 16:02:10.979 HINT Received binary command: Command=[62] Value=[3,500]
14.09.2025 16:02:10.979 HINT State changed from [ConnectControl] to [Ready]
14.09.2025 16:02:11.997 HINT Sending binary command: Command=[62] Value=[0,000]
14.09.2025 16:02:11.997 HINT State changed from [Ready] to [ConnectControl]
14.09.2025 16:02:12.009 HINT Received Data: >#0 #0 `@
14.09.2025 16:02:12.528 HINT Received binary command: Command=[62] Value=[3,500]
14.09.2025 16:02:12.528 HINT State changed from [ConnectControl] to [Ready]
14.09.2025 16:02:13.144 HINT Sending binary command: Command=[8] Value=[8,000]
14.09.2025 16:02:13.144 HINT State changed from [Ready] to [WaitMove]
14.09.2025 16:02:15.220 HINT Received Data: #8 #0 #0 #0 A
14.09.2025 16:02:15.728 HINT Received binary command: Command=[8] Value=[8,000]
14.09.2025 16:02:15.728 HINT State changed from [WaitMove] to [Ready]
14.09.2025 16:02:15.728 HINT completed movement
14.09.2025 16:02:15.734 HINT Sending binary command: Command=[62] Value=[0,000]
14.09.2025 16:02:15.734 HINT State changed from [Ready] to [ConnectControl]
14.09.2025 16:02:15.753 HINT Received Data: >#0 #0 `@
14.09.2025 16:02:16.262 HINT Received binary command: Command=[62] Value=[3,500]
14.09.2025 16:02:16.262 HINT State changed from [ConnectControl] to [Ready]
14.09.2025 16:02:17.276 HINT Sending binary command: Command=[62] Value=[0,000]
14.09.2025 16:02:17.276 HINT State changed from [Ready] to [ConnectControl]
14.09.2025 16:02:17.281 HINT Received Data: >#0 #0 `@
14.09.2025 16:02:17.796 HINT Received binary command: Command=[62] Value=[3,500]
14.09.2025 16:02:17.796 HINT State changed from [ConnectControl] to [Ready]
14.09.2025 16:02:18.806 HINT Sending binary command: Command=[62] Value=[0,000]
14.09.2025 16:02:18.806 HINT State changed from [Ready] to [ConnectControl]
14.09.2025 16:02:18.812 HINT Received Data: >#0 #0 `@
14.09.2025 16:02:19.323 HINT Received binary command: Command=[62] Value=[3,500]
14.09.2025 16:02:19.323 HINT State changed from [ConnectControl] to [Ready]

```
14.09.2025 16:02:20.593 HINT Sending binary command: Command=[62] Value=[0,000]
14.09.2025 16:02:20.593 HINT State changed from [Ready] to [ConnectControl]
14.09.2025 16:02:20.926 HINT Received Data: >#0 #0 `@
14.09.2025 16:02:21.432 HINT Received binary command: Command=[62] Value=[3,500]
14.09.2025 16:02:21.432 HINT State changed from [ConnectControl] to [Ready]
14.09.2025 16:02:21.471 ------------------------------------------------------------
14.09.2025 16:02:21.471 logfile AcDLIPInterface#1.log closed.
14.09.2025 16:02:21.471 ------------------------------------------------------------
```

Anlage 6: Beispiel Logfile Fehlerfall

```
14.09.2025 16:04:20.573 ------------------------------------------------------------
14.09.2025 16:04:20.573 logfile AcDLIPInterface#1.log opened (OPEN_ALWAYS).
14.09.2025 16:04:20.573 ------------------------------------------------------------
14.09.2025 16:04:21.486 HINT Opened serial port [4]
14.09.2025 16:04:21.486 HINT State changed from [Offline] to [Init]
14.09.2025 16:04:21.492 HINT Sending binary command: Command=[1] Value=[0,000]
14.09.2025 16:04:21.492 HINT State changed from [Init] to [WaitInit]
14.09.2025 16:04:23.513 HINT Received Data: #1 #162 #140 #230 ?
14.09.2025 16:04:24.028 HINT Received binary command: Command=[1] Value=[1,801]
14.09.2025 16:04:24.028 HINT State changed from [WaitInit] to [Initialized]
14.09.2025 16:04:24.034 HINT State changed from [Initialized] to [Home]
14.09.2025 16:04:24.041 HINT Sending binary command: Command=[2] Value=[0,000]
14.09.2025 16:04:24.041 HINT State changed from [Home] to [WaitHome]
14.09.2025 16:04:24.612 HINT Received Data: #2 #0 #0 #0 #0
14.09.2025 16:04:25.124 HINT Received binary command: Command=[2] Value=[0,000]
14.09.2025 16:04:25.124 HINT State changed from [WaitHome] to [AskValues]
14.09.2025 16:04:25.130 HINT Sending binary command: Command=[62] Value=[0,000]
14.09.2025 16:04:25.130 HINT State changed from [AskValues] to [WaitValues]
14.09.2025 16:04:25.139 HINT Received Data: >#0 #0 `@
14.09.2025 16:04:25.656 HINT Received binary command: Command=[62] Value=[3,500]
14.09.2025 16:04:25.656 HINT State changed from [WaitValues] to [AskValues]
14.09.2025 16:04:25.662 HINT Sending binary command: Command=[63] Value=[0,000]
14.09.2025 16:04:25.662 HINT State changed from [AskValues] to [WaitValues]
14.09.2025 16:04:25.667 HINT Received Data: ?#0 #0 PA
14.09.2025 16:04:26.306 HINT Received binary command: Command=[63] Value=[13,000]
14.09.2025 16:04:26.470 HINT State changed from [WaitValues] to [AskValues]
14.09.2025 16:04:26.629 HINT Sending binary command: Command=[64] Value=[0,000]
14.09.2025 16:04:26.791 HINT State changed from [AskValues] to [WaitValues]
14.09.2025 16:04:26.801 HINT Received Data: @#0 #0 #160 @
14.09.2025 16:04:27.505 HINT Received binary command: Command=[64] Value=[5,000]
14.09.2025 16:04:27.505 HINT State changed from [WaitValues] to [AskValues]
14.09.2025 16:04:27.518 HINT Sending binary command: Command=[65] Value=[0,000]
14.09.2025 16:04:27.530 HINT State changed from [AskValues] to [WaitValues]
14.09.2025 16:04:27.536 HINT Received Data: A#0 #0 #144 A
14.09.2025 16:04:28.043 HINT Received binary command: Command=[65] Value=[18,000]
14.09.2025 16:04:28.043 HINT State changed from [WaitValues] to [AskValues]
```

```
14.09.2025 16:04:28.049 HINT Sending binary command: Command=[66] Value=[0,000]
14.09.2025 16:04:28.049 HINT State changed from [AskValues] to [WaitValues]
14.09.2025 16:04:28.062 HINT Received Data: B#0 #0 #160 @
14.09.2025 16:04:28.572 HINT Received binary command: Command=[66] Value=[5,000]
14.09.2025 16:04:28.572 HINT State changed from [WaitValues] to [AskValues]
14.09.2025 16:04:28.578 HINT Sending binary command: Command=[67] Value=[0,000]
14.09.2025 16:04:28.578 HINT State changed from [AskValues] to [WaitValues]
14.09.2025 16:04:28.596 HINT Received Data: C#0 #0 #144 A
14.09.2025 16:04:29.103 HINT Received binary command: Command=[67] Value=[18,000]
14.09.2025 16:04:29.103 HINT State changed from [WaitValues] to [Ready]
14.09.2025 16:04:30.125 HINT Sending binary command: Command=[62] Value=[0,000]
14.09.2025 16:04:30.125 HINT State changed from [Ready] to [ConnectControl]
14.09.2025 16:04:30.137 HINT Received Data: >#0 #0 `@
14.09.2025 16:04:30.653 HINT Received binary command: Command=[62] Value=[3,500]
14.09.2025 16:04:30.653 HINT State changed from [ConnectControl] to [Ready]
14.09.2025 16:04:31.667 HINT Sending binary command: Command=[62] Value=[0,000]
14.09.2025 16:04:31.667 HINT State changed from [Ready] to [ConnectControl]
14.09.2025 16:04:31.679 HINT Received Data: >#0 #0 `@
14.09.2025 16:04:32.184 HINT Received binary command: Command=[62] Value=[3,500]
14.09.2025 16:04:32.184 HINT State changed from [ConnectControl] to [Ready]
14.09.2025 16:04:33.200 HINT Sending binary command: Command=[62] Value=[0,000]
14.09.2025 16:04:33.200 HINT State changed from [Ready] to [ConnectControl]
14.09.2025 16:04:33.218 HINT Received Data: >#0 #0 `@
14.09.2025 16:04:33.729 HINT Received binary command: Command=[62] Value=[3,500]
14.09.2025 16:04:33.729 HINT State changed from [ConnectControl] to [Ready]
14.09.2025 16:04:34.746 HINT Sending binary command: Command=[62] Value=[0,000]
14.09.2025 16:04:34.746 HINT State changed from [Ready] to [ConnectControl]
14.09.2025 16:04:34.753 HINT Received Data: >#0 #0 `@
14.09.2025 16:04:35.264 HINT Received binary command: Command=[62] Value=[3,500]
14.09.2025 16:04:35.264 HINT State changed from [ConnectControl] to [Ready]
14.09.2025 16:04:36.277 HINT Sending binary command: Command=[62] Value=[0,000]
14.09.2025 16:04:36.277 HINT State changed from [Ready] to [ConnectControl]
14.09.2025 16:04:36.289 HINT Received Data: >#0 #0 `@
14.09.2025 16:04:36.792 HINT Received binary command: Command=[62] Value=[3,500]
14.09.2025 16:04:36.792 HINT State changed from [ConnectControl] to [Ready]
14.09.2025 16:04:37.808 HINT Sending binary command: Command=[62] Value=[0,000]
14.09.2025 16:04:37.808 HINT State changed from [Ready] to [ConnectControl]
14.09.2025 16:04:37.820 HINT Received Data: >#0 #0 `@
```

```
14.09.2025 16:04:38.323 HINT Received binary command: Command=[62] Value=[3,500]
14.09.2025 16:04:38.323 HINT State changed from [ConnectControl] to [Ready]
14.09.2025 16:04:39.340 HINT Sending binary command: Command=[62] Value=[0,000]
14.09.2025 16:04:39.340 HINT State changed from [Ready] to [ConnectControl]
14.09.2025 16:04:39.346 HINT Received Data: >#0 #0 `@
14.09.2025 16:04:39.852 HINT Received binary command: Command=[62] Value=[3,500]
14.09.2025 16:04:39.852 HINT State changed from [ConnectControl] to [Ready]
14.09.2025 16:04:40.873 HINT Sending binary command: Command=[62] Value=[0,000]
14.09.2025 16:04:40.873 HINT State changed from [Ready] to [ConnectControl]
14.09.2025 16:04:40.879 HINT Received Data: >#0 #0 `@
14.09.2025 16:04:41.388 HINT Received binary command: Command=[62] Value=[3,500]
14.09.2025 16:04:41.388 HINT State changed from [ConnectControl] to [Ready]
14.09.2025 16:04:42.400 HINT Sending binary command: Command=[62] Value=[0,000]
14.09.2025 16:04:42.400 HINT State changed from [Ready] to [ConnectControl]
14.09.2025 16:04:42.406 HINT Received Data: >#0 #0 `@
14.09.2025 16:04:42.915 HINT Received binary command: Command=[62] Value=[3,500]
14.09.2025 16:04:42.915 HINT State changed from [ConnectControl] to [Ready]
14.09.2025 16:04:43.935 HINT Sending binary command: Command=[62] Value=[0,000]
14.09.2025 16:04:43.935 HINT State changed from [Ready] to [ConnectControl]
14.09.2025 16:04:43.947 HINT Received Data: >#0 #0 `@
14.09.2025 16:04:44.462 HINT Received binary command: Command=[62] Value=[3,500]
14.09.2025 16:04:44.462 HINT State changed from [ConnectControl] to [Ready]
14.09.2025 16:04:45.480 HINT Sending binary command: Command=[62] Value=[0,000]
14.09.2025 16:04:45.480 HINT State changed from [Ready] to [ConnectControl]
14.09.2025 16:04:45.486 HINT Received Data: >#0 #0 `@
14.09.2025 16:04:45.997 HINT Received binary command: Command=[62] Value=[3,500]
14.09.2025 16:04:45.997 HINT State changed from [ConnectControl] to [Ready]
14.09.2025 16:04:47.011 HINT Sending binary command: Command=[62] Value=[0,000]
14.09.2025 16:04:47.011 HINT State changed from [Ready] to [ConnectControl]
14.09.2025 16:04:48.025 HINT State changed from [ConnectControl] to [Wait-
ForOnline]
14.09.2025 16:04:49.044 HINT State changed from [WaitForOnline] to [Init]
14.09.2025 16:04:49.050 HINT Sending binary command: Command=[1] Value=[0,000]
14.09.2025 16:04:49.551 ERROR Failed to send data: Timeout während Operation
14.09.2025 16:04:49.551 HINT State changed from [Init] to [Offline]
14.09.2025 16:04:49.557 HINT Closed serial port
14.09.2025 16:04:49.557 HINT Opened serial port [4]
14.09.2025 16:04:49.557 HINT State changed from [Offline] to [Init]
```

14.09.2025 16:04:49.563 HINT Sending binary command: Command=[1] Value=[0,000]
14.09.2025 16:04:50.064 ERROR Failed to send data: Timeout während Operation
14.09.2025 16:04:50.064 HINT State changed from [Init] to [Offline]
14.09.2025 16:04:50.070 HINT Closed serial port
14.09.2025 16:04:50.070 HINT Opened serial port [4]
14.09.2025 16:04:50.070 HINT State changed from [Offline] to [Init]
14.09.2025 16:04:50.076 HINT Sending binary command: Command=[1] Value=[0,000]
14.09.2025 16:04:50.576 ERROR Failed to send data: Timeout während Operation
14.09.2025 16:04:50.576 HINT State changed from [Init] to [Offline]
14.09.2025 16:04:50.582 HINT Closed serial port
14.09.2025 16:04:50.582 HINT Opened serial port [4]
14.09.2025 16:04:50.582 HINT State changed from [Offline] to [Init]
14.09.2025 16:04:50.589 HINT Sending binary command: Command=[1] Value=[0,000]
14.09.2025 16:04:51.090 ERROR Failed to send data: Timeout während Operation
14.09.2025 16:04:51.090 HINT State changed from [Init] to [Offline]
14.09.2025 16:04:51.096 HINT Closed serial port
14.09.2025 16:04:51.096 HINT Opened serial port [4]
14.09.2025 16:04:51.096 HINT State changed from [Offline] to [Init]
14.09.2025 16:04:51.103 HINT Sending binary command: Command=[1] Value=[0,000]
14.09.2025 16:04:51.604 ERROR Failed to send data: Timeout während Operation
14.09.2025 16:04:51.604 HINT State changed from [Init] to [Offline]
14.09.2025 16:04:51.610 HINT Closed serial port
14.09.2025 16:04:51.610 HINT Opened serial port [4]
14.09.2025 16:04:51.610 HINT State changed from [Offline] to [Init]
14.09.2025 16:04:51.617 HINT Sending binary command: Command=[1] Value=[0,000]
14.09.2025 16:04:52.117 ERROR Failed to send data: Timeout während Operation
14.09.2025 16:04:52.117 HINT State changed from [Init] to [Offline]
14.09.2025 16:04:52.123 HINT Closed serial port
14.09.2025 16:04:52.123 HINT Opened serial port [4]
14.09.2025 16:04:52.123 HINT State changed from [Offline] to [Init]
14.09.2025 16:04:52.129 HINT Sending binary command: Command=[1] Value=[0,000]
14.09.2025 16:04:52.631 ERROR Failed to send data: Timeout während Operation
14.09.2025 16:04:52.631 HINT State changed from [Init] to [Offline]
14.09.2025 16:04:52.637 HINT Closed serial port
14.09.2025 16:04:52.637 HINT Opened serial port [4]
14.09.2025 16:04:52.637 HINT State changed from [Offline] to [Init]
14.09.2025 16:04:52.643 HINT Sending binary command: Command=[1] Value=[0,000]
14.09.2025 16:04:53.144 ERROR Failed to send data: Timeout während Operation

```
14.09.2025 16:04:53.144 HINT State changed from [Init] to [Offline]
14.09.2025 16:04:53.150 HINT Closed serial port
14.09.2025 16:04:53.150 HINT Opened serial port [4]
14.09.2025 16:04:53.150 HINT State changed from [Offline] to [Init]
14.09.2025 16:04:53.157 HINT Sending binary command: Command=[1] Value=[0,000]
14.09.2025 16:04:53.657 ERROR Failed to send data: Timeout während Operation
14.09.2025 16:04:53.657 HINT State changed from [Init] to [Offline]
14.09.2025 16:04:53.663 HINT Closed serial port
14.09.2025 16:04:53.663 HINT Opened serial port [4]
14.09.2025 16:04:53.663 HINT State changed from [Offline] to [Init]
14.09.2025 16:04:53.669 HINT Sending binary command: Command=[1] Value=[0,000]
14.09.2025 16:04:54.170 ERROR Failed to send data: Timeout während Operation
14.09.2025 16:04:54.170 HINT State changed from [Init] to [Offline]
14.09.2025 16:04:54.177 HINT Closed serial port
14.09.2025 16:04:54.177 HINT Opened serial port [4]
14.09.2025 16:04:54.177 HINT State changed from [Offline] to [Init]
14.09.2025 16:04:54.183 HINT Sending binary command: Command=[1] Value=[0,000]
14.09.2025 16:04:54.684 ERROR Failed to send data: Timeout während Operation
14.09.2025 16:04:54.684 HINT State changed from [Init] to [Offline]
14.09.2025 16:04:54.690 HINT Closed serial port
14.09.2025 16:04:54.690 HINT Opened serial port [4]
14.09.2025 16:04:54.690 HINT State changed from [Offline] to [Init]
14.09.2025 16:04:54.697 HINT Sending binary command: Command=[1] Value=[0,000]
14.09.2025 16:04:55.197 ERROR Failed to send data: Timeout während Operation
14.09.2025 16:04:55.197 HINT State changed from [Init] to [Offline]
14.09.2025 16:04:55.203 HINT Closed serial port
14.09.2025 16:04:55.203 HINT Opened serial port [4]
14.09.2025 16:04:55.203 HINT State changed from [Offline] to [Init]
14.09.2025 16:04:55.209 HINT Sending binary command: Command=[1] Value=[0,000]
14.09.2025 16:04:55.662 HINT State changed from [Init] to [WaitInit]
14.09.2025 16:04:58.665 HINT State changed from [WaitInit] to [WaitForOnline]
14.09.2025 16:04:59.686 HINT State changed from [WaitForOnline] to [Init]
14.09.2025 16:04:59.692 HINT Sending binary command: Command=[1] Value=[0,000]
14.09.2025 16:04:59.692 HINT State changed from [Init] to [WaitInit]
14.09.2025 16:05:01.711 HINT Received Data: #1 #162 #140 #230 ?
14.09.2025 16:05:02.216 HINT Received binary command: Command=[1] Value=[1,801]
14.09.2025 16:05:02.216 HINT State changed from [WaitInit] to [Initialized]
14.09.2025 16:05:02.222 HINT State changed from [Initialized] to [Home]
```

14.09.2025 16:05:02.228 HINT Sending binary command: Command=[2] Value=[0,000]
14.09.2025 16:05:02.228 HINT State changed from [Home] to [WaitHome]
14.09.2025 16:05:02.754 HINT Received Data: #2 #0 #0 #0 #0
14.09.2025 16:05:03.261 HINT Received binary command: Command=[2] Value=[0,000]
14.09.2025 16:05:03.261 HINT State changed from [WaitHome] to [AskValues]
14.09.2025 16:05:03.267 HINT Sending binary command: Command=[62] Value=[0,000]
14.09.2025 16:05:03.267 HINT State changed from [AskValues] to [WaitValues]
14.09.2025 16:05:03.274 HINT Received Data: >#0 #0 `@
14.09.2025 16:05:03.793 HINT Received binary command: Command=[62] Value=[3,500]
14.09.2025 16:05:03.793 HINT State changed from [WaitValues] to [AskValues]
14.09.2025 16:05:03.799 HINT Sending binary command: Command=[63] Value=[0,000]
14.09.2025 16:05:03.799 HINT State changed from [AskValues] to [WaitValues]
14.09.2025 16:05:03.806 HINT Received Data: ?#0 #0 PA
14.09.2025 16:05:04.326 HINT Received binary command: Command=[63] Value=[13,000]
14.09.2025 16:05:04.326 HINT State changed from [WaitValues] to [AskValues]
14.09.2025 16:05:04.332 HINT Sending binary command: Command=[64] Value=[0,000]
14.09.2025 16:05:04.332 HINT State changed from [AskValues] to [WaitValues]
14.09.2025 16:05:04.339 HINT Received Data: @#0 #0 #160 @
14.09.2025 16:05:04.855 HINT Received binary command: Command=[64] Value=[5,000]
14.09.2025 16:05:04.855 HINT State changed from [WaitValues] to [AskValues]
14.09.2025 16:05:04.861 HINT Sending binary command: Command=[65] Value=[0,000]
14.09.2025 16:05:04.861 HINT State changed from [AskValues] to [WaitValues]
14.09.2025 16:05:04.873 HINT Received Data: A#0 #0 #144 A
14.09.2025 16:05:05.384 HINT Received binary command: Command=[65] Value=[18,000]
14.09.2025 16:05:05.384 HINT State changed from [WaitValues] to [AskValues]
14.09.2025 16:05:05.390 HINT Sending binary command: Command=[66] Value=[0,000]
14.09.2025 16:05:05.390 HINT State changed from [AskValues] to [WaitValues]
14.09.2025 16:05:05.402 HINT Received Data: B#0 #0 #160 @
14.09.2025 16:05:05.918 HINT Received binary command: Command=[66] Value=[5,000]
14.09.2025 16:05:05.918 HINT State changed from [WaitValues] to [AskValues]
14.09.2025 16:05:05.924 HINT Sending binary command: Command=[67] Value=[0,000]
14.09.2025 16:05:05.924 HINT State changed from [AskValues] to [WaitValues]
14.09.2025 16:05:05.936 HINT Received Data: C#0 #0 #144 A
14.09.2025 16:05:06.451 HINT Received binary command: Command=[67] Value=[18,000]
14.09.2025 16:05:06.451 HINT State changed from [WaitValues] to [Ready]
14.09.2025 16:05:07.464 HINT Sending binary command: Command=[62] Value=[0,000]
14.09.2025 16:05:07.464 HINT State changed from [Ready] to [ConnectControl]
14.09.2025 16:05:07.470 HINT Received Data: >#0 #0 `@

```
14.09.2025 16:05:07.979 HINT Received binary command: Command=[62] Value=[3,500]
14.09.2025 16:05:07.979 HINT State changed from [ConnectControl] to [Ready]
14.09.2025 16:05:08.997 HINT Sending binary command: Command=[62] Value=[0,000]
14.09.2025 16:05:08.997 HINT State changed from [Ready] to [ConnectControl]
14.09.2025 16:05:09.009 HINT Received Data: >#0 #0 `@
14.09.2025 16:05:09.525 HINT Received binary command: Command=[62] Value=[3,500]
14.09.2025 16:05:09.525 HINT State changed from [ConnectControl] to [Ready]
14.09.2025 16:05:10.543 HINT Sending binary command: Command=[62] Value=[0,000]
14.09.2025 16:05:10.543 HINT State changed from [Ready] to [ConnectControl]
14.09.2025 16:05:10.556 HINT Received Data: >#0 #0 `@
14.09.2025 16:05:11.076 HINT Received binary command: Command=[62] Value=[3,500]
14.09.2025 16:05:11.076 HINT State changed from [ConnectControl] to [Ready]
14.09.2025 16:05:12.091 HINT Sending binary command: Command=[62] Value=[0,000]
14.09.2025 16:05:12.091 HINT State changed from [Ready] to [ConnectControl]
14.09.2025 16:05:12.103 HINT Received Data: >#0 #0 `@
14.09.2025 16:05:12.623 HINT Received binary command: Command=[62] Value=[3,500]
14.09.2025 16:05:12.623 HINT State changed from [ConnectControl] to [Ready]
14.09.2025 16:05:13.634 HINT Sending binary command: Command=[62] Value=[0,000]
14.09.2025 16:05:13.634 HINT State changed from [Ready] to [ConnectControl]
14.09.2025 16:05:13.640 HINT Received Data: >#0 #0 `@
14.09.2025 16:05:14.153 HINT Received binary command: Command=[62] Value=[3,500]
14.09.2025 16:05:14.153 HINT State changed from [ConnectControl] to [Ready]
14.09.2025 16:05:15.169 HINT Sending binary command: Command=[62] Value=[0,000]
14.09.2025 16:05:15.169 HINT State changed from [Ready] to [ConnectControl]
14.09.2025 16:05:15.175 HINT Received Data: >#0 #0 `@
14.09.2025 16:05:15.680 HINT Received binary command: Command=[62] Value=[3,500]
14.09.2025 16:05:15.680 HINT State changed from [ConnectControl] to [Ready]
14.09.2025 16:05:16.805 HINT Sending binary command: Command=[62] Value=[0,000]
14.09.2025 16:05:16.805 HINT State changed from [Ready] to [ConnectControl]
14.09.2025 16:05:16.956 HINT Received Data: >#0 #0 `@
14.09.2025 16:05:17.313 ------------------------------------------------------------
14.09.2025 16:05:17.313 logfile AcDLIPInterface#1.log closed.
14.09.2025 16:05:17.313 ------------------------------------------------------------
```

WHZ Westsächsische Hochschule Zwickau
Hochschule für Mobilität

Selbstständigkeitserklärung

Hiermit versichere ich, dass ich die vorliegende Arbeit in allen Teilen selbstständig angefertigt und keine anderen als die in der Arbeit angegebenen Quellen und Hilfsmittel benutzt habe, und dass die Arbeit in gleicher oder ähnlicher Form in noch keiner anderen Prüfung vorgelegen hat. Mir ist bewusst, dass ich Autor/in der vorliegenden Arbeit bin und volle Verantwortung für den Text trage. ☑

Ich erkläre, dass ich wörtlich oder sinngemäß aus anderen Werken – dazu gehören auch Internetquellen – übernommene Inhalte als solche kenntlich gemacht und die entsprechenden Quellen angegeben habe. ☑

Mir ist bewusst, dass meine Arbeit auf Plagiate überprüft werden kann. Mir ist bekannt, dass es sich bei der Abgabe eines Plagiats um ein schweres akademisches Fehlverhalten handelt und dass Täuschungen nach der für mich gültigen Prüfungsordnung geahndet werden. ☑

Zusätzlich versichere ich, dass ich auf künstlicher Intelligenz (KI) basierende Werkzeuge nur in Absprache mit den Prüfern verwendet habe. Dabei stand meine eigene geistige Leistung im Vordergrund, und ich habe jederzeit den Prozess steuernd bearbeitet. ☑

Diese Werkzeuge habe ich im Quellenverzeichnis in der Rubrik „Übersicht verwendeter Hilfsmittel" mit ihrem Produktnamen und einer Übersicht des im Rahmen dieser Prüfungs-/Studienarbeit genutzten Funktionsumfangs unter Angabe der Textstelle in der Arbeit vollständig aufgeführt. ☑

Ich versichere, dass ich keine KI-basierten Tools verwendet habe, deren Nutzung die Prüfer explizit schriftlich ausgeschlossen haben. Ich bin mir bewusst, dass die Verwendung von Texten oder anderen Inhalten und Produkten, die durch KI-basierte Tools generiert wurden, keine Garantie für deren Qualität darstellt. ☑

Ich verantworte die Übernahme jeglicher von mir verwendeter maschinell generierter Passagen vollumfänglich selbst und trage die Verantwortung für eventuell durch die KI generierte fehlerhafte oder verzerrte Inhalte, fehlerhafte Referenzen, Verstöße gegen das Datenschutz- und Urheberrecht oder Plagiate. ☑

Zwickau, 22.09.2025 J. Heinrich

Ort, Datum Unterschrift