TECHNISCHE
UNIVERSITÄT
DARMSTADT

Hardware-Beschleuniger für ein Optisches System zur Drahtdurchmesser-Bestimmung

Bachelorarbeit von

Mario Wetzel

Oktober 2017

Erstgutachter:
Prof. Dr.-Ing. Andreas Koch

Zweitgutachter:
Dr.-Ing. Andreas Engel

Technische Universität Darmstadt
Department of Computer Science
Embedded Systems and Applications Group (ESA)

Hardware-Beschleuniger für ein Optisches System zur Drahtdurchmesser-Bestimmung

Hardware-Acceleration of a Laser Diameter Gauge

Bachelorarbeit von Mario Wetzel
Eingereicht am 09.10.2017
Erstgutachter: Prof. Dr.-Ing. Andreas Koch
Zweitgutachter: Dr.-Ing. Andreas Engel

Eigenständigkeitserklärung

Hiermit versichere ich die vorliegende Bachelorarbeit ohne Hilfe Dritter und nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die Quellen entnommen wurden, sind als solche kenntlich gemacht worden. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen. In der abgegebenen Thesis stimmen die schriftliche und elektronische Fassung überein.

Darmstadt, 9. Oktober 2017

(Mario Wetzel)

Kurzfassung

In dieser Bachelorarbeit wird ein Datenanalyse-Algorithmus beschleunigt, der den Durchmesser eines Drahtes aus dem Beugungsbild bestimmt. Dieser Algorithmus wird von der Firma KJM in einem Messgerät zur Drahtdurchmesserbestimmung eingesetzt. Die Beschleunigung soll durch eine Hardwareimplementierung des Algorithmus verwirklicht werden. Das Ziel davon ist es, das Messgerät mit einer höheren Abtastrate betreiben zu können. Dadurch kann während der Produktion des Drahtes dessen Durchmesser mit einer höheren Abtastrate überwacht werden.

Bevor mit der Implementierung des Datenanalyse-Algorithmus begonnen wird, wird eine Literaturanalyse zu Messverfahren zum Bestimmen des Durchmessers eines Drahtes durchgeführt. Dabei wird das Laser-Scanning Messverfahren vorgestellt, das den Draht mit einem Laser abtastet und aus der Länge des Schattenbereiches den Drahtdurchmesser bestimmt. Die meisten am Markt erhältlichen Messgeräte setzen das Laser-Scanning Messverfahren ein. Die Messgeräte, die einen vergleichbaren minimal messbaren Durchmesser wie das KJM Messgerät haben, erreichen dabei eine Abtastrate von bis zu 2,4 kHz.

Außerdem wird das Messverfahren vorgestellt, welches von KJM verwendet wird. Dieses Messverfahren bestimmt den Drahtdurchmesser aus dem Fraunhoferbeugungsbild.

Bisher wird der Datenanalyse-Algorithmus mit einem NIOS II Softcore Prozessor auf einem Altera Cyclone III Field-Programmable Gate Array (FPGA) ausgeführt.

Bei der Implementierung der Hardwarebeschreibung des Datenanalyse-Algorithmus wird ein streambasierter Ansatz umgesetzt. Dabei wird das Beugungsbild als Datenstrom ausgegeben. Dieser Datenstrom wird von der Hardwareimplementierung zu wenigen Kennwerten aggregiert und aus diesen Daten der Drahtdurchmesser berechnet.

Dadurch kann eine Abtastrate von 11,879 kHz (Standard-Methode) bzw. 6,977 kHz (FFT-Methode) erreicht werden.

Zukünftig könnte man die beiden Methoden kombinieren. Außerdem lässt sich die bisherige Taktfrequenz von 63,02 MHz durch Optimieren der Implementierung noch erhöhen.

Inhaltsverzeichnis

1 Einleitung

1.1 Motivation

Während der Produktion von Drähten ist es wichtig, deren Durchmesser kontinuierlich zu messen. Durch die Überwachung des Drahtdurchmessers kann man sicherstellen, dass die Qualitätsanforderungen an den Draht eingehalten werden. Dabei kann auch das Risiko von Drahtabriss während der Produktion reduziert werden [20].

Da bei der Drahtherstellung der Draht mit einer hohen Geschwindigkeit durch das Messgerät geführt wird, ist es wichtig, dass das Messgerät eine hohe Abtastrate unterstützt. Besonders bei der Herstellung von dünnen Drähten muss der Drahtdurchmesser gleichzeitig auch mit einer hohen Genauigkeit bestimmt werden.

1.2 Aufgabenstellung

Deswegen soll in dieser Bachelorarbeit ein bestehender Algorithmus der Firma KJM GmbH zur Drahtdurchmesserbestimmung beschleunigt werden. Dazu wird der Datenanalyse-Algorithmus durch eine Hardwareimplementierung auf einem FPGA parallelisiert und dadurch beschleunigt. Zur Implementierung der Hardwarebeschreibung des Datenanalyse-Algorithmus ist der in der Programmiersprache C geschriebene Algorithmus von KJM vorgegeben. Das Messgerät von KJM verwendet zurzeit einen Altera Cyclone III FPGA (*EP3C25U256C8N*) zur Berechnung des Drahtdurchmessers. Auf diesem FPGA wird der Datenanalyse-Algorithmus mit einem NIOS II Softcore Prozessor ausgeführt. Der Softcore Prozessor benötigt ca. 25 ms um den Datenanalyse-Algorithmus auszuführen. Somit lässt sich aktuell lediglich eine Messrate von 40 Hz erreichen. Der Sensor würde eine Abtastrate von ca. 3 kHz unterstützen. Da das Messgerät schon das Altera Cyclone III FPGA verwendet, soll die Hardwareimplementierung auch auf diesem FPGA lauffähig sein.

Das Messgerät von KJM erfasst das bei der Beleuchtung des Drahtes entstandene Beugungsbild mit einem CCD. Der Datenanalyse-Algorithmus bestimmt dann aus dem Beugungsbild den Drahtdurchmesser. Zum Auslesen des CCDs ist schon eine

Hardwareimplementierung zum Ansteuern des CCDs und des Analog-Digital-Wandlers (ADC) vorhanden. Des Weiteren ist schon eine allgemeine Implementierung für einen Speicherblock vorhanden, in dem die eingelesenen Daten des CCDs abgelegt werden. Somit setzt die Hardwareimplementierung des Datenanalyse-Algorithmus beim Auslesen dieses Speicherblocks an.

Der CCD-Zeilensensor, der von dem Messgerät verwendet wird, ist der *ILX553B* von Sony. Dieser Sensor hat eine Zeile mit 5150 Pixeln. Der ADC wandelt das analoge Signal des CCDs in ein 10 bit breites digitales Signal um. Die von KJM bereitgestellten Testdateien haben eine Länge von 5000 Pixeln.

Neben der Optimierung des Datenanalyse-Algorithmus soll auch eine Literaturanalyse zu existierenden Messverfahren und Algorithmen für das Vermessen von Drahtdurchmessern durchgeführt werden. Außerdem soll auch eine Marktanalyse zu Messgeräten zum Bestimmen von Drahtdurchmessern angefertigt werden. Weiterhin soll für die Sensoren und die Recheneinheit eine geeignete Auswahl aktueller Hardwarekomponenten erstellt werden. Zur Verifikation der Hardwareimplementierung soll eine Testumgebung zur Simulation der Implementierung aufgesetzt werden.

1.3 Strukturierung der Arbeit

Zu Beginn dieser Bachelorarbeit werden verschiedene Messverfahren zur Drahtdurchmesserbestimmung vorgestellt und miteinander verglichen (Kapitel 2). Des Weiteren wird in Kapitel 3 eine Marktanalyse zu optischen Messgeräten zur Vermessung von Drahtdurchmessern durchgeführt. Dabei werden Messgeräte vorgestellt, die zu dem Messgerät von KJM in Konkurrenz stehen. Anschließend werden in Kapitel 4 Voruntersuchungen zur Implementierung des Algorithmus zur Drahtdurchmesserbestimmung durchgeführt. Dabei wird die Parallelisierbarkeit des Datenanalyse-Algorithmus analysiert. Auf Basis der Ergebnisse dieser Analyse werden verschiedene Implementierungsoptionen dargestellt. Außerdem wird in diesem Kapitel auch ein passendes FPGA und ein Sensor für die Implementierung ausgewählt. In Kapitel 5 wird dann die eigentliche Hardwareimplementierung des Datenanalyse-Algorithmus realisiert. Die Evaluation der Implementierung wird in Kapitel 6 durchgeführt und im Anschluss in Kapitel 7 diskutiert. Abschließend wird in Kapitel 8 die Arbeit kurz zusammengefasst und ein Ausblick auf mögliche Erweiterungen der Implementierung gegeben.

2 Theorie zur Drahtdurchmesserbestimmung

2.1 Beugung

Beugung ist das Phänomen, bei dem man bei der Beleuchtung einer Blende mit kohärentem Licht keine scharfen Schatten beobachten kann, wie man nach der geometrischen Optik erwarten würde. Stattdessen beobachtet man, dass sich die Lichtstrahlen auch in den geometrischen Schatten ausbreiten. Dies lässt sich mit dem Fresnel-Huygens-Prinzip erklären. Danach ist jeder Punkt einer Wellenfront wieder Ausgangspunkt von Elementarwellen, die die gleiche Frequenz wie die ursprüngliche Welle besitzen. Die Amplitude bestimmt sich an den nachfolgenden Punkten durch Überlagerung aller dieser Elementarwellen unter Berücksichtigung der Phase. Wenn sich dabei kein Hindernis im Weg befindet, bilden die Elementarwellen wieder eine neue Wellenfront. Wenn sich jedoch ein Hindernis im Weg befindet, können sich die Wellen so überlagern, dass sie sich gegenseitig verstärken oder sogar vollständig auslöschen. Diese Überlagerung von Wellen nennt man Interferenz.

Wenn man beispielsweise einen Spalt mit kohärentem Licht beleuchtet, lassen sich abhängig von der Breite des Spaltes und der Entfernung der Beobachtungsebene vom Spalt unterschiedliche Beugungsbilder beobachten. Dabei kann man als Faustformel angeben, dass man die Fraunhoferbeugung anwenden kann, wenn der Abstand zwischen Spalt und Beobachtungsschirm l größer als das Quadrat der Breite des Spalts d durch die Wellenlänge λ des Lichts ist. Ansonsten sollte man die Fresnelbeugung anwenden [11].

$$l > \frac{d^2}{\lambda} \tag{2.1}$$

In Abbildung 2.1 ist das Beugungsbild eines 0,1 mm breiten Spaltes im Abstand von 0,1 mm, 1 mm und 1 m dargestellt. Der Spalt wird mit monochromatischem Licht beleuchtet, das eine Wellenlänge von 675 nm hat.

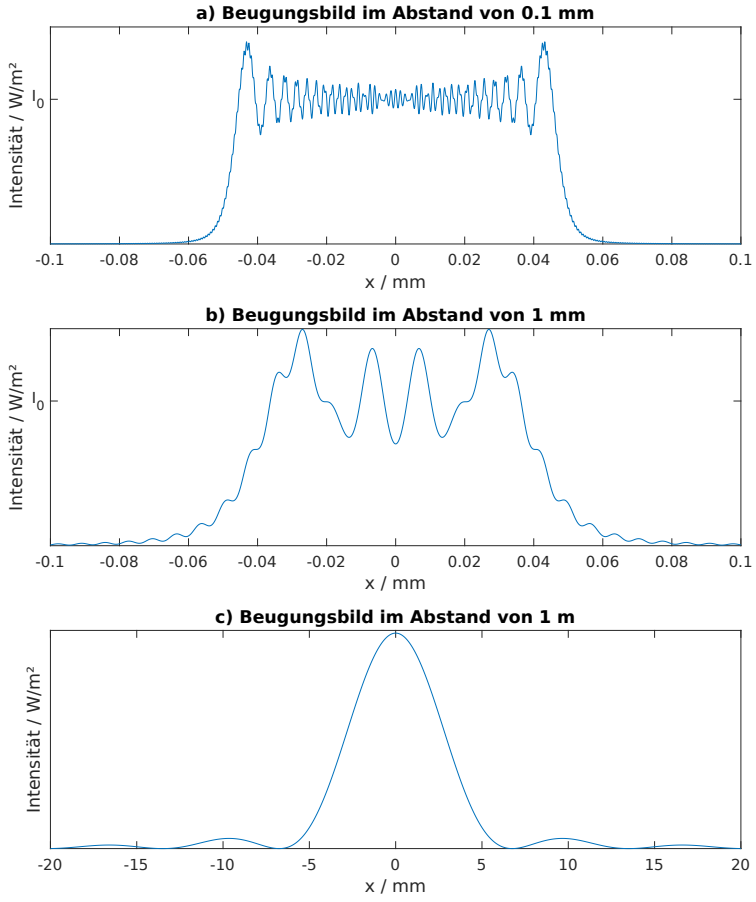

Abbildung 2.1: Beugungsbilder eines 0,1 mm breiten Spalts in den Abständen 0,1 mm, 1 mm und 1 m. Die Wellenlänge des Lichts beträgt 675 nm. (mit Formel 2.7 berechnet)

2.2 Fraunhoferbeugung

Die Fraunhoferbeugung lässt sich als Spezialfall der Fresnelbeugung darstellen. Dazu nimmt man an, dass das Beugungsobjekt relativ klein ist und sich der Abbildungsschirm in nahezu unendlicher Entfernung vom Beugungsobjekt befindet.

Die Intensitätsverteilung bei der Beugung am Einzelspalt lässt sich mit der Formel 2.2 berechnen. Die Herleitung der Formel soll hier nicht gezeigt werden. Diese lässt sich zum Beispiel in [11] nachschlagen. Die Formel der Intensitätsverteilung 2.2 weist jedem Beugungswinkel β eine Beleuchtungsstärke I zu. Hierbei ist I_0 die Intensität des Hauptmaximums, d die Breite des Einzelspaltes, l der Abstand zwischen Spalt und Abbildungsschirm und λ die Wellenlänge des kohärenten monochromatischen

Lichts.

$$I(\beta) = I_0 \left(\frac{\sin z}{z} \right)^2 \tag{2.2}$$

mit

$$z = \frac{d\pi}{\lambda} \sin \beta \tag{2.3}$$

Im Folgenden soll eine anschauliche Herleitung der Nullstellen der Intensitätsverteilung dargestellt werden. Hierfür ist in Abbildung 2.2 die Beugung am Einzelspalt mit der Breite d skizziert. Nach dem Fresnel-Huygens-Prinzip kann man annehmen, dass jeder einzelne Punkt in der Öffnung des Spaltes Lichtstrahlen in alle Richtungen aussendet. Da man bei der Fraunhoferbeugung voraussetzt, dass sich der Abbildungsschirm im Unendlichen befindet, kann man die Lichtstrahlen, die vom Spalt ausgesendet werden und sich auf dem Abbildungsschirm in einem Punkt treffen, als parallele Strahlen annehmen.

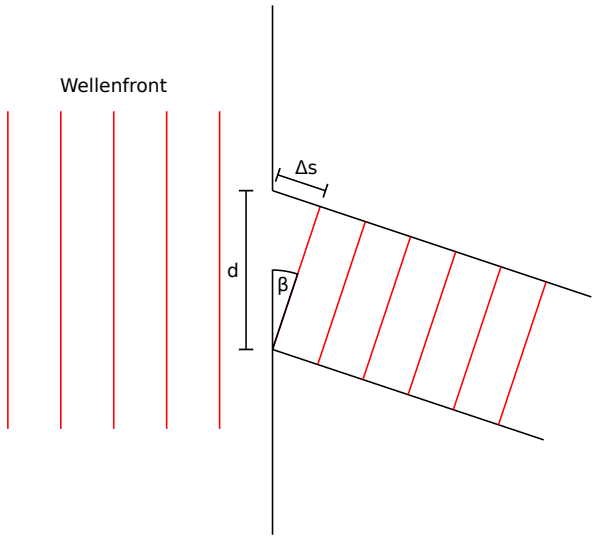

Abbildung 2.2: Skizze zur Herleitung der Formel für die Nullstellen der Bestrahlungsstärke. (frei nach [11] Bild 10.9)

Der Gangunterschied Δs in Abbildung 2.2 ist abhängig von dem Winkel β. Wenn der Gangunterschied Δs zwischen dem oberen und dem unteren Strahl gleich der Wellenlänge λ des Lichts ist, löschen sich der obere Strahl und der Strahl in der Mitte aus, da die beiden Strahlen um $\lambda/2$ verschoben sind. Analog dazu findet man zu jedem Lichtstrahl in der oberen Hälfte einen Lichtstrahl in der unteren Hälfte, die sich gegenseitig auslöschen. Vergrößert man den Winkel weiter, so löschen sich nicht mehr alle Strahlen aus und die Lichtintensität steigt an, bis sie ein Nebenmaximum erreicht. Wenn der Gangunterschied Δs gleich der doppelten Wellenlänge λ ist,

löschen sich die Lichtstrahlen wieder gegenseitig aus und es entstehet ein Minimum.

Allgemein gesagt nimmt die Intensitätsverteilung immer ein Minimum an, wenn der Gangunterschied Δs ein Vielfaches der Wellenlänge λ ist. Somit kann man die Bedingung für ein Minimum in der Intensitätsverteilung wie folgt angeben:

$$d \cdot \sin \beta = \pm i \cdot \lambda \qquad i = 1, 2, 3, ... \tag{2.4}$$

Hierbei ist i die Ordnung des Minimums. Für $i = 0$ ist der Gangunterschied Null und somit kann auch kein Minimum entstehen [11].

2.3 Fresnelbeugung

Auch die Herleitung der Fresnelbeugung soll an dieser Stellen nicht erläutert werden, da man diese beispielsweise in [11] nachschlagen kann. Die Intensitätsverteilung der Fresnelbeugung lässt sich mithilfe der fresnelschen Integrale 2.5 und 2.6 berechnen.

$$C(w) = \int_0^w \cos \frac{\pi w'^2}{2} dw' \tag{2.5}$$

$$S(w) = \int_0^w \sin \frac{\pi w'^2}{2} dw' \tag{2.6}$$

Eine Skizze zur Fresnelbeugung ist in Abbildung 2.3 dargestellt.

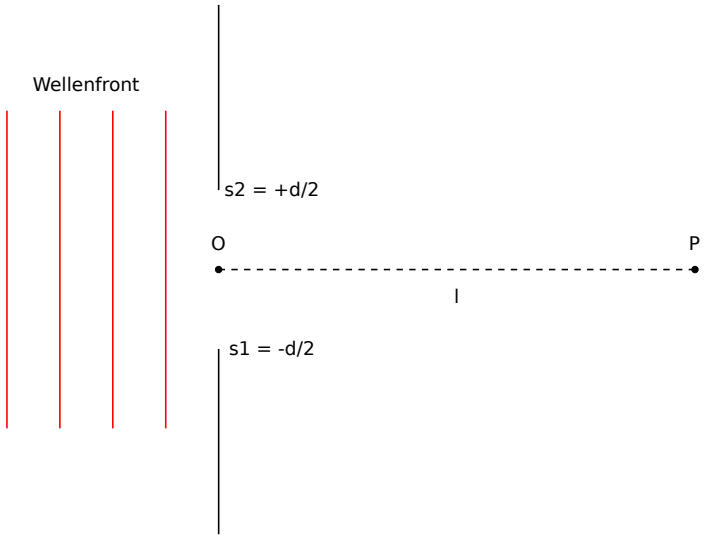

Abbildung 2.3: Skizze zur Fresnelbeugung. (frei nach [11] Bild 10.49)

Die Intensität im zentralen Punkt P lässt sich nach Formel 2.7 berechnen. Hierbei ist l der Abstand zwischen dem Punkt P und dem Spalt mit der Breite d. λ ist

die Wellenlänge des kohärenten monochromatischen Lichts und I_0 ist die Intensität der Wellenfront vor dem Beugungsspalt. j ist die imaginäre Einheit der komplexen Zahlen.

$$I_P = \frac{I_0}{2} \left| [C(u_2) + jS(u_2)] - [C(u_1) + jS(u_1)] \right|^2 \tag{2.7}$$

mit

$$u_1 = s_1 \sqrt{\frac{2}{l \cdot \lambda}} = -\frac{d}{2} \sqrt{\frac{2}{l \cdot \lambda}} \tag{2.8}$$

$$u_2 = s_2 \sqrt{\frac{2}{l \cdot \lambda}} = +\frac{d}{2} \sqrt{\frac{2}{l \cdot \lambda}} \tag{2.9}$$

Um die Berechnung der Intensitätsverteilung zu vereinfachen, verschiebt man nicht den Punkt P auf der Beobachtungsebene sondern den Spalt, also die Punkte s_1 und s_2. Der Fehler den man dabei macht, ist vernachlässigbar klein, wenn die Strecke, über die man den Spalt verschiebt, viel kleiner als der Abstand zwischen Spalt und Beobachtungsschirm ist [11].

2.4 Messverfahren zur Bestimmung des Drahtdurchmessers

Zur Messung des Durchmessers eines Drahtes existieren in der Industrie hauptsächlich zwei verschiedene Messprinzipien. In beiden Fällen wird der Draht von einer Seite mit monochromatischem Licht beleuchtet, das meist mit einer Laserdiode erzeugt wird. Bei dem ersten Messprinzip wird dann der Drahtdurchmesser aus der Länge des bei der Beleuchtung des Drahtes entstandenen Schattenbereiches bestimmt (Abschnitt 2.4.1). Dagegen wird bei dem anderen Messprinzip der Drahtdurchmesser aus den Minima des bei der Beleuchtung des Drahtes entstandenen Beugungsbildes bestimmt (Abschnitt 2.4.2).

Das Beugungsbild unterscheidet sich dabei für unterschiedliche Beleuchtungsanordnungen (mehr dazu in Abschnitt 2.1). So operiert die Messanordnung aus Abschnitt 2.4.2 im Bereich der Fraunhoferbeugung, während die Anordnung aus Abschnitt 2.4.1 im Bereich der Fresnelbeugung arbeitet.

2.4.1 Drahtdurchmesser aus der Länge des Drahtschattens bestimmen

Der Schattenwurf des seitlich beleuchteten Drahtes wird in der Regel mit einer Photodiode oder einem CCD vermessen.

Länge des Drahtschattens mittels Photodiode bestimmen

Zur Bestimmung der Länge des Schattenbereiches wird der Draht mit einem dünnen Laserstrahl abgetastet. Mit einem Lichtsensor, der beispielsweise eine Photodiode sein kann, wird dann die Zeitdauer gemessen, die die Photodiode nicht beleuchtet wird. Diese Messanordnung ist in Abbildung 2.4 dargestellt.

Dabei wird ein dünner, monochromatischer, kohärenter und kollimierter Laserstrahl auf einen mit konstanter Geschwindigkeit rotierenden Spiegel geworfen. Dieser Spiegel reflektiert den Laserstrahl auf eine Sammellinse. Dadurch tastet der Laserstrahl das Messfeld periodisch ab. Die Laserstrahlen verlaufen dabei immer parallel zueinander durch das Messfeld. Hinter dem Messfeld befindet sich eine weitere Sammellinse. Diese fokussiert die Laserstrahlen, die nicht auf den zu messenden Draht treffen, auf eine Photodiode. Es ergibt sich, dass der Drahtdurchmesser proportional zu der Zeitspanne ist, die die Photodiode nicht beleuchtet wird [10].

Da der Laserstrahl an dem Draht gebeugt wird, entsteht zwischen dem beleuchteten und dem unbeleuchteten Bereich keine scharfe Trennung. Dadurch wird die Genauigkeit der Messung begrenzt [21, 7]. Außerdem muss der Durchmesser des

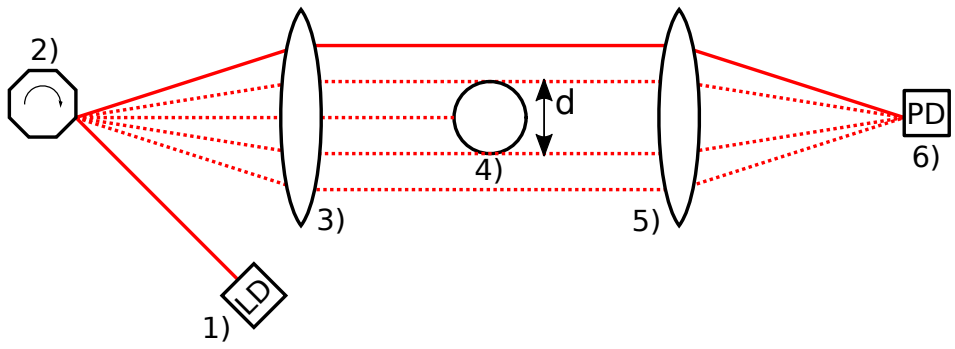

Abbildung 2.4: Messanordnung zur Bestimmung des Drahtdurchmessers aus dem Schattenabbild mithilfe einer Photodiode. (frei nach [10])
1) Laserdiode, 2) rotierender Spiegel, 3) Sammellinse, 4) Messobjekt (Draht mit Durchmesser d), 5) Sammellinse, 6) Photodiode

Laserstrahls kleiner sein als der Durchmesser des zu messenden Drahtes [21]. Des Weiteren können Ungenauigkeiten bei der Messung durch bewegliche Teile entstehen. So kann eine Unwucht im antreibenden Motor die gleichförmige Rotation des Spiegels stören. Zusätzlich kann die Messung durch eine ungenaue Geschwindigkeitsregelung des Motors gestört werden. Auch Staubpartikel, die sich auf den Linsen oder dem Spiegel absetzen, können die Genauigkeit der Messung beeinflussen [12, 7].

Länge des Drahtschattens mittels CCD-Zeilensensor bestimmen

Ein anderes Messverfahren, das in [12] beschrieben wird, bestimmt die Länge des Schattenbereiches nicht mehr mit einer Photodiode, sondern mit einem CCD. Dabei wird der Draht von einer Seite mit einer Laserdiode beleuchtet und das dabei entstandene Bild mithilfe eines CCD-Zeilensensors aufgenommen. Aus diesem Bild lässt sich die Länge des Drahtschattens ermitteln, mit der sich dann der Drahtdurchmesser bestimmen lässt. Dabei muss man allerdings den Abstand des Drahtes vom CCD-Zeilensensor kennen. Dieser Abstand ist entweder konstant, indem der Draht beispielsweise durch Rollen geführt wird, oder der Abstand lässt sich durch das Hinzufügen einer zweiten Messachse bestimmen. Das Messverfahren das zwei Messachsen verwendet, die senkrecht zueinander stehen, ist in Abbildung 2.5 dargestellt. Dabei sind $l_{CCD,x}$ und $l_{CCD,y}$ jeweils die Längen der beiden CCDs, $l_{Schatten,x}$ und $l_{Schatten,y}$ jeweils die Längen der beiden Schattenbereiche und e_x und e_y die Abweichung der Mittelpunkte der Schattenbereiche wenn sich der Draht nicht im Mittelpunkt O befindet. l_1 ist der Abstand zwischen Laserdiode und CCD und l_2 ist der Abstand zwischen Laserdiode und dem Mittelpunkt O.

Wenn sich der Mittelpunkt des Drahtes genau im Mittelpunkt O befindet, lässt

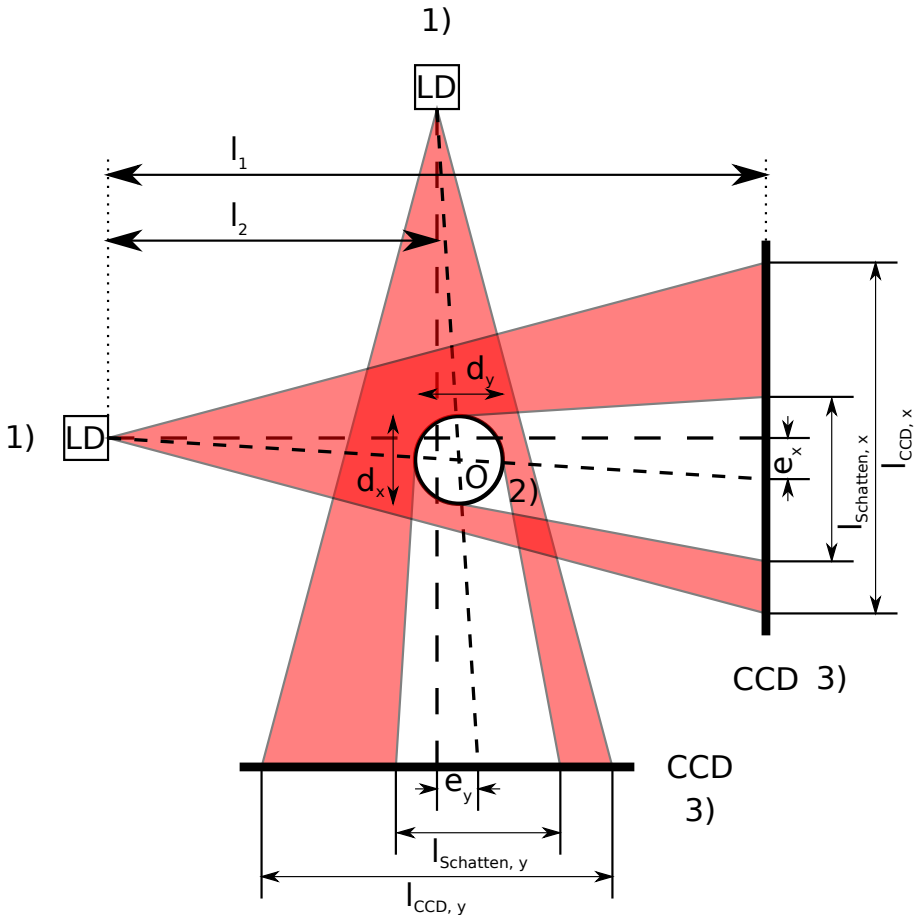

Abbildung 2.5: Messanordnung zur Bestimmung des Drahtdurchmessers aus dem Schattenabbild mithilfe eines CCD-Zeilensensors. (frei nach [12])
1) Laserdiode, 2) Messobjekt (Draht mit Durchmesser d), 3) CCD-Zeilensensor

sich der Drahtdurchmesser d_x bzw. d_y einfach durch Anwendung des Strahlensatzes bestimmen. Hierbei setzt man die Abstände l_1 und l_2 ins Verhältnis zu der Länge des Schattenbereiches $l_{Schatten,x}$ bzw. $l_{Schatten,y}$ und dem Drahtdurchmesser d_x bzw. d_y.

$$d_x = \frac{l_2}{l_1} \cdot l_{Schatten,x} \tag{2.10}$$

$$d_y = \frac{l_2}{l_1} \cdot l_{Schatten,y} \tag{2.11}$$

Wenn sich der Draht aber nicht genau im Mittelpunkt O befindet, muss die Berechnung um die Korrekturfaktoren Δx, Δy, θ_x und θ_y erweitert werden [12].

$$d_x = \frac{l_{Schatten,x} \cdot (l_2 + \Delta y)}{l_1} \cdot \frac{1}{\sqrt{1 + (\frac{l_{Schatten,x}}{2 l_1 \theta_x})^2}} \tag{2.12}$$

$$d_y = \frac{l_{Schatten,y} \cdot (l_2 + \Delta x)}{l_1} \cdot \frac{1}{\sqrt{1 + (\frac{l_{Schatten,y}}{2 l_1 \theta_y})^2}} \tag{2.13}$$

mit

$$\Delta x = \frac{l_2 e_y \cdot (l_1 + e_x)}{l_1^2 - e_x e_y} \tag{2.14}$$

$$\Delta y = \frac{l_2 e_x \cdot (l_1 + e_y)}{l_1^2 - e_x e_y} \tag{2.15}$$

$$\theta_x = \sqrt{1 + (\frac{e_x}{l_1})^2} \tag{2.16}$$

$$\theta_x = \sqrt{1 + (\frac{e_y}{l_1})^2} \tag{2.17}$$

Zur Bestimmung der Länge der Schattenbereiche $l_{Schatten,x}$ und $l_{Schatten,y}$ aus dem Bildsignal des CCD-Zeilensensors müssen die Positionen der beiden Schattenränder bestimmt werden. In den beiden folgenden Abschnitten werden zwei Methoden zur Bestimmung der Position des Schattenrandes vorgestellt.

Im Gegensatz zu dem in Abschnitt 2.4.1 vorgestellten Messverfahren verwendet dieser Ansatz keine beweglichen Bauteile oder optische Linsensysteme. Daher ist dieser Ansatz robuster als das Photodioden-System [12, 7].

Position des Schattenrandes mithilfe einer Ausgleichsgeraden bestimmen

Hong verwendet dazu in [12] einen Algorithmus, der eine Ausgleichsgerade an den Übergängen von dem unbeleuchteten Bereich in den beleuchteten Bereich berechnet. Dabei wird die Fresnelbeugung (Abschnitt 2.3) nach dem in Abbildung 2.6 dargestellten Prinzip vereinfacht. Um den geometrischen Rand des Schattens zu bestimmen, nähert man den Bereich zwischen I_a und I_b mithilfe einer Ausgleichsgeraden an. Wie in Abbildung 2.6 eingezeichnet sind I_a und I_b dabei festgelegt zu

$$I_a = I_L + 0,25(I_H - I_L) \tag{2.18}$$

$$I_b = I_L + 0,75(I_H - I_L) \tag{2.19}$$

(Hinweis: In [12] sind die beiden Faktoren nicht $0,25$ und $0,75$ sondern $2,5$ und $7,5$. Damit I_a und I_b aber zu den entsprechenden Punkten in Abbildung 2.6 passen, wurden diese Faktoren angepasst.)

n_a und n_b sind die entsprechenden Pixel, an denen die Funktion die Werte I_a und I_b annimmt.

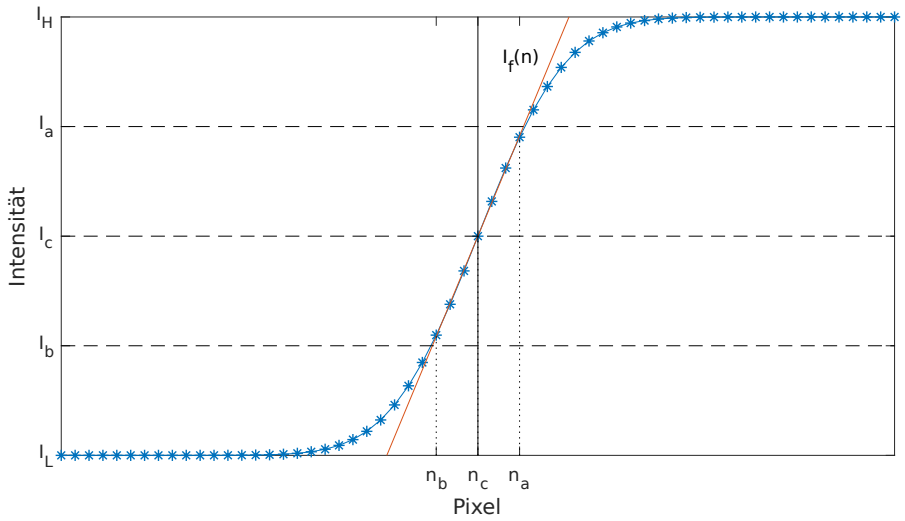

Abbildung 2.6: Bestimmung der Position des Schattenrandes mithilfe einer Ausgleichsgeraden (Rot). (frei nach [12])

Somit lässt sich der Bereich zwischen n_a und n_b mit der Geraden

$$I_f(n) = kn + r \tag{2.20}$$

annähern. Dabei wird die Methode zur Minimierung der Fehlerquadrate angewendet. Somit berechnen sich die Steigung k und der Achsenabschnitt r wie folgt:

$$k = \frac{\sum_{n=n_b}^{n_a} n \cdot \sum_{n=n_b}^{n_a} I(n) - (n_a - n_b + 1) \sum_{n=n_b}^{n_a} [n \cdot I(n)]}{(\sum_{n=n_b}^{n_a} n)^2 - (n_a - n_b + 1) \sum_{n=n_b}^{n_a} n^2} \tag{2.21}$$

$$r = \frac{\sum_{n=n_b}^{n_a} n \cdot \sum_{n=n_b}^{n_a} [n \cdot I(n)] - \sum_{n=n_b}^{n_a} n^2 \sum_{n=n_b}^{n_a} I(n)}{(\sum_{n=n_b}^{n_a} n)^2 - (n_a - n_b + 1) \sum_{n=n_b}^{n_a} n^2} \tag{2.22}$$

Mit dieser Ausgleichsgeraden (2.20) lässt sich dann mit

$$I_c = I_L + 0.5(I_H - I_L) \tag{2.23}$$

die Stelle n_c, die dem geometrischen Rand des Schattens entspricht, bestimmen.

$$n_c = \frac{I_c - r}{k} \tag{2.24}$$

Position des Schattenrandes durch Auswertung der Extremstellen des Beugungsbildes bestimmen

Eine weitere Möglichkeit den Schattenrand zu bestimmen besteht darin, die durch Beugung entstanden Maxima und Minima auszuwerten, wie Chursin in [7] beschreibt. Da der Abstand zwischen Draht und CCD klein ist, muss man zur Auswertung des Beugungsbildes die Fresnelbeugung (Abschnitt 2.3) verwenden. Dabei entsteht bei der Beugung an der Kante eines Objektes auf dem CCD ein unbeleuchteter Bereich und ein beleuchteter Bereich. Dies ist in Abbildung 2.7 dargestellt. Im beleuchteten Bereich entstehen Beugungsmaxima und -minima, wobei der Abstand zwischen zwei benachbarten Extremwerten mit größerem Abstand vom der geometrischen Grenze des Schattens X_t abnimmt. Die Intensität der Extremwerte nähert sich mit zunehmendem Abstand zur geometrischen Grenze des Schattens X_t immer mehr der Intensität I_0 an.

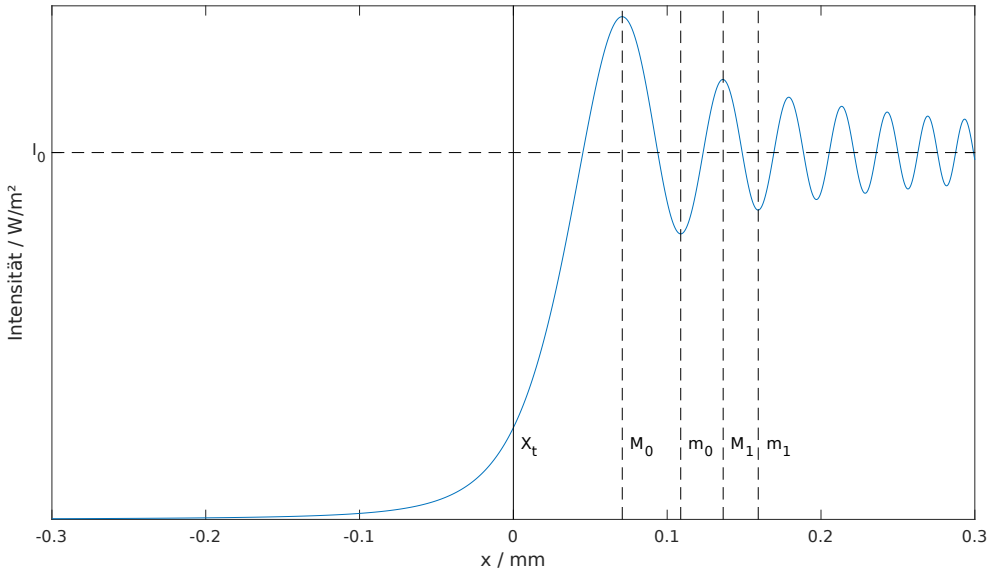

Abbildung 2.7: Beugungsbild, das entsteht, wenn das Licht des Lasers an der Kante eines unendlich langen Hindernisses nach der Fresnelbeugung gebeugt wird. (frei nach [7])
(Abstand Hindernis zu CCD ist 10 mm; $\lambda = 675\,\text{nm}$)
X_t: geometrischer Rand des Schattenbereiches
M_i: Maximum i-ter Ordnung
m_i: Minimum i-ter Ordnung

Der Abstand A_i bzw. a_i zwischen der geometrische Grenze des Schattens X_t und dem Maximum M_i bzw. Minimum m_i der Ordnung i ist nach [7] durch die Formeln 2.25 und 2.26 definiert. Dabei ist l_1 der Abstand zwischen der Laserdiode und dem Draht, l_2 der Abstand zwischen der Laserdiode und dem CCD und λ die Wellenlänge des Lasers.

$$A_i = \sqrt{\frac{\lambda l_1(l_1 - l_2)}{2l_2}(4i + \frac{3}{2})} \qquad i = 0, 1, 2, ... \tag{2.25}$$

$$a_i = \sqrt{\frac{\lambda l_1(l_1 - l_2)}{2l_2}(4i + \frac{7}{2})} \qquad i = 0, 1, 2, ... \tag{2.26}$$

Mit den Abständen A_i und a_i lässt sich dann die Position X_t des Schattenrandes berechnen.

2.4.2 Drahtdurchmesser aus Fraunhoferbeugungsbild bestimmen

Durch die Kombination von CCD und Sammellinse lässt sich der Drahtdurchmesser mittels Fraunhofer- statt Fresnelbeugung bestimmen [21, 38]. Dies vereinfacht die Drahtdurchmesserbestimmung, wie nachfolgend gezeigt. Bei der Fraunhoferbeugung muss der Abbildungsschirm (in diesem Fall der CCD-Zeilensensor) unendlich weit von dem Beugungsobjekt (dem Draht) entfernt sein. Da sich dies experimentell nicht realisieren lässt, verwendet man eine Sammellinse mit der Brennweite b hinter dem Draht. Durch diese werden Lichtstrahlen, die das Beugungsobjekt unter gleichem Winkel zur optischen Achse verlassen, also parallel zueinander sind, auf den gleichen Punkt auf dem Abbildungsschirm abgebildet. Der Abbildungsschirm muss sich dabei im Brennpunkt der Sammellinse befinden [11]. Dies entspricht der Bedingung der Fraunhoferbeugung, dass der Abstand zwischen Beugungsobjekt und Abbildungsschirm unendlich groß ist.

Die dazu nötige Messanordnung ist in Abbildung 2.8 dargestellt. β ist dabei der Winkel unter dem das Licht des Lasers gebeugt wird und b ist die Brennweite der Sammellinse.

Das Beugungsbild das man bei dieser Messanordnung beobachten kann, ist in Abbildung 2.9 dargestellt.

Aus den Minima des dabei entstandenen Beugungsbildes lässt sich dann der Drahtdurchmesser bestimmen. Dieses Verfahren ist besonders für sehr dünne Drähte im Bereich von 5 µm bis 100 µm geeignet, da der Beugungseffekt Teil des Messprinzips ist und nicht Quelle von Unschärfe ist [21]. Die Messverfahren aus Abschnitt 2.4.1 könnte man für diesen Messbereich nicht einsetzen.

Gemäß dem Prinzip von Babinet lässt sich die Formel der Fraunhoferbeugung am Draht auf die Beugung am Spalt (siehe Abschnitt 2.2) zurückführen [38, 11]. Die

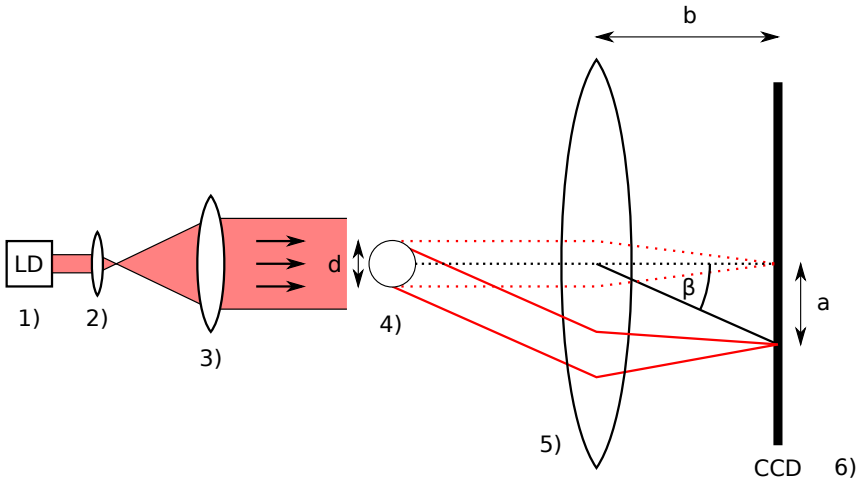

Abbildung 2.8: Messanordnung zur Bestimmung des Drahtdurchmessers aus dem Beugungsbild der Fraunhoferbeugung. (frei nach [21, 38])
1) Laserdiode, 2) Sammellinse, 3) Sammellinse, 4) Messobjekt (Draht mit Durchmesser d), 5) Sammellinse, 6) CCD-Zeilensensor

Intensitätsverteilung $I(\beta)$ des Beugungsbildes lässt sich wie folgt beschreiben:

$$I(\beta) = I_0 \left(\frac{\sin(\frac{d\pi}{\lambda} \sin\beta)}{\frac{d\pi}{\lambda} \sin\beta} \right)^2 \tag{2.27}$$

Hierbei ist I_0 die Intensität des Hauptmaximums, d der Drahtdurchmesser, λ die Wellenlänge des Lasers und β der Beugungswinkel.

Durch Null setzen der Intensitätsverteilung 2.27 ergeben sich folgende Nullstellen, die gleichzeitig auch die Minima der Intensitätsverteilung sind. Hierbei ist i die Ordnung des Minimums. Dabei muss man $i = 0$ ausschließen, da man für $\beta = 0$ durch Null dividiert und die Funktion für $\beta = 0$ gegen I_0 geht.

$$\frac{d \cdot \pi}{\lambda} \sin\beta = \pm i \cdot \pi \qquad i = 1, 2, 3, ... \tag{2.28}$$

$$\Rightarrow d \cdot \sin\beta = \pm i \cdot \lambda \tag{2.29}$$

Da Gleichung 2.29 vom Drahtdurchmesser d abhängig ist, lässt sich aus den Minima des Beugungsbildes der Drahtdurchmesser bestimmen.

Der Beugungswinkel β lässt sich aus dem Abstand a_i des Minimums i-ter Ordnung zum zentralen Maximum und der Brennweite b der Sammellinse nach 2.30 berechnen.

$$\beta = \arctan \frac{a}{b} \tag{2.30}$$

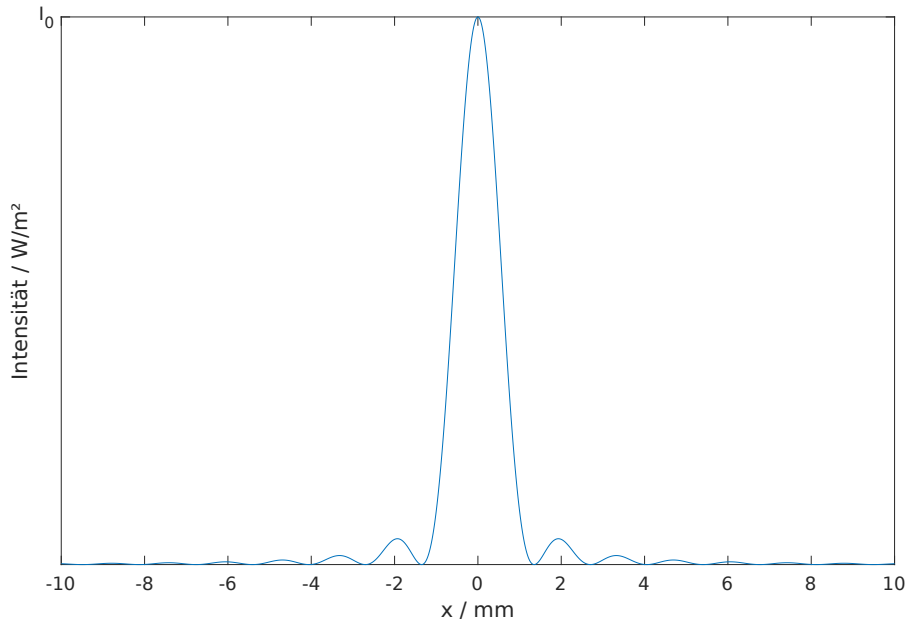

Abbildung 2.9: Beugungsbild, das bei der Fraunhoferbeugung am Draht entsteht. (nach Formel 2.27)
($d = 0{,}05\,\text{mm}$; $b = 0{,}1\,\text{m}$; $\lambda = 675\,\text{nm}$)

Wenn die Brennweite b der Sammellinse deutlich größer ist als der Abstand a_i zwischen einem Minimum und dem zentralen Maximum des Beugungsbildes, lässt sich folgende Kleinwinkelnäherung anwenden:

$$\sin\beta \approx \tan\beta = \frac{a_i}{b} \tag{2.31}$$

Da sich der Abstand zweier Minima gleicher Ordnung a_{+i} und a_{-i} leichter bestimmen lässt als der Abstand eines Minimums vom zentralen Maximum des Beugungsbildes, weil man die Minima besser erkennen kann als das zentrale Maximum, lässt sich Gleichung 2.29 mit der Kleinwinkelnäherung 2.31 umformen zu:

$$d = \frac{2 \cdot i \cdot \lambda}{|a_{+i}| + |a_{-i}|} b \qquad i = 1, 2, 3, ... \tag{2.32}$$

Drahtdurchmesser mithilfe der Diskreten Fourier-Transformation bestimmen

Eine weitere Möglichkeit den Drahtdurchmesser aus dem Beugungsbild 2.9 zu bestimmen, wird von Lüdge in [21] beschrieben. Dabei wird die Schnelle Fourier-Transformation (FFT) angewendet, die eine effiziente Realisierung der Diskreten Fourier-Transformation (DFT) darstellt. Bei diesem Messverfahren wird das Bildsignal

zuerst entrauscht, bevor der Drahtdurchmesser bestimmt wird. Dies ist sinnvoll, da das Signal, das von einem CCD ausgegeben wird, immer einen gewissen Grad an Rauschen enthält und damit das Signal-Rausch-Verhältnis (SNR) verbessert wird.

Bei diesem Algorithmus werden zur Bestimmung des Drahtdurchmessers alle Werte des Beugungssignals verwendet und nicht nur die Position der Minima wie in Abschnitt 2.4.2, um daraus den Drahtdurchmesser zu bestimmen. Dies wird erreicht, indem mithilfe der FFT die dominante Frequenz des Beugungssignals bestimmt wird. Da die Minima des Beugungsbildes für kleine Winkel β periodisch zu dieser Frequenz sind, lässt sich damit der Drahtdurchmesser bestimmen.

Der Algorithmus von Lüdge wird auf einen N Werte breiten Ausschnitt der linken oder rechten Seite des Beugungssignals angewendet. Dieser Ausschnitt des Beugungssignals wird zuerst mit einem adaptiven Tiefpass gefiltert, indem das Signal mit dem Algorithmus der schnellen Fourier-Transformation (FFT) transformiert wird. Das so entstandene komplexe Signal mit Real- und Imaginärteil wird so abgeschnitten, dass es nur noch die ersten $k + v$ Frequenzamplituden enthält und somit das hochfrequente Rauschen abgeschnitten wird. Dabei ist k der die dominante Frequenz repräsentierende Index. v wird so gewählt, dass $k + v$ eine Zweierpotenz ist und das Spektrum von 0 bis $k+v$ so wenig Rauschen wie möglich enthält. Gleichzeitig dürfen aber keine relevanten Frequenzen des Beugungssignals abgeschnitten werden.

Das abgeschnittene Frequenzsignal wird durch die Inverse Schnelle Fourier-Transformation (IFFT) wieder zurücktransformiert. Durch das Abschneiden des Signals im Frequenzbereich enthält das zurücktransformierte Signal nur noch $2(k+v)$ Werte und wird mit Nullen aufgefüllt, sodass es wieder N Werte enthält.

Anschließend wird das Signal wieder mittels FFT transformiert und daraus das spektrale Leistungsdichtespektrum bestimmt. Das Leistungsdichtespektrum hat jetzt durch das vorherige Auffüllen mit Nullen eine höhere Frequenzgenauigkeit. Dadurch entsteht ein ZOOM-Faktor von $N/(2(k + v))$.

Jedoch lässt sich das Auffüllen mit Nullen auch als eine Multiplikation mit einer Rechteckfunktion beschreiben. Wenn die Fouriertransformierte dieser Rechteckfunktion Frequenzanteile größer der halben Abtastrate (Abtasttheorem) besitzt, entsteht eine spektrale Überfaltung (Aliasing). Dadurch kommt es je nach Phase der überfalteten Frequenzanteile zu einer Verschiebung der dominanten Frequenz hin zu einer niedrigeren oder höheren Frequenz. Um diesen Effekt zu kompensieren simuliert man die Durchmesserbestimmung für jeden Drahtdurchmesser im Messbereich mit Beugungssignalen, die mit der Formel 2.27 generiert wurden. Die Ergebnisse werden in einer Lookup-Tabelle (LUT) gespeichert, sodass jeder gemessenen Frequenz ein Drahtdurchmesser zugewiesen wird.

Da das Beugungssignal (2.27) aber keine si-Funktion, sondern eine quadrierte si-Funktion ist, muss beachtet werden, dass das Beugungssignal nach 2.33 einen konstanten Anteil besitzt.

$$I(\beta) = I_0 \cdot \frac{\sin^2(z)}{z^2} = \frac{I_0}{2z^2} \cdot (\cos(2z) + 1) \tag{2.33}$$

mit

$$z = \frac{d\pi}{\lambda} \sin \beta \tag{2.34}$$

In Abbildung 2.10 wurde der Algorithmus auf einen Ausschnitt eines Fraunhoferbeugungsbildes angewendet, zu dem weißes Rauschen hinzugefügt wurde.

Die Sinusfunktion der si-Funktion besitzt eine Frequenz von 740,74 Hz und die Messung ergibt einen Wert von 1465 Hz. Da dieser Wert aber für die quartierte si-Funktion ist, muss man ihn noch durch zwei dividieren und man erhält einen Wert von 732,5 Hz.

2.4.3 Messverfahren des KJM LMD H01B

Das Messgerät *KJM LMD H01B*, das die Grundlage dieser Arbeit ist, wendet das in Abschnitt 2.4.2 beschriebene Messverfahren an. Der Draht wird dabei durch Rollen geführt. Dadurch wird sichergestellt, dass der Draht nicht schwingt und einen konstanten Abstand zum CCD hat.

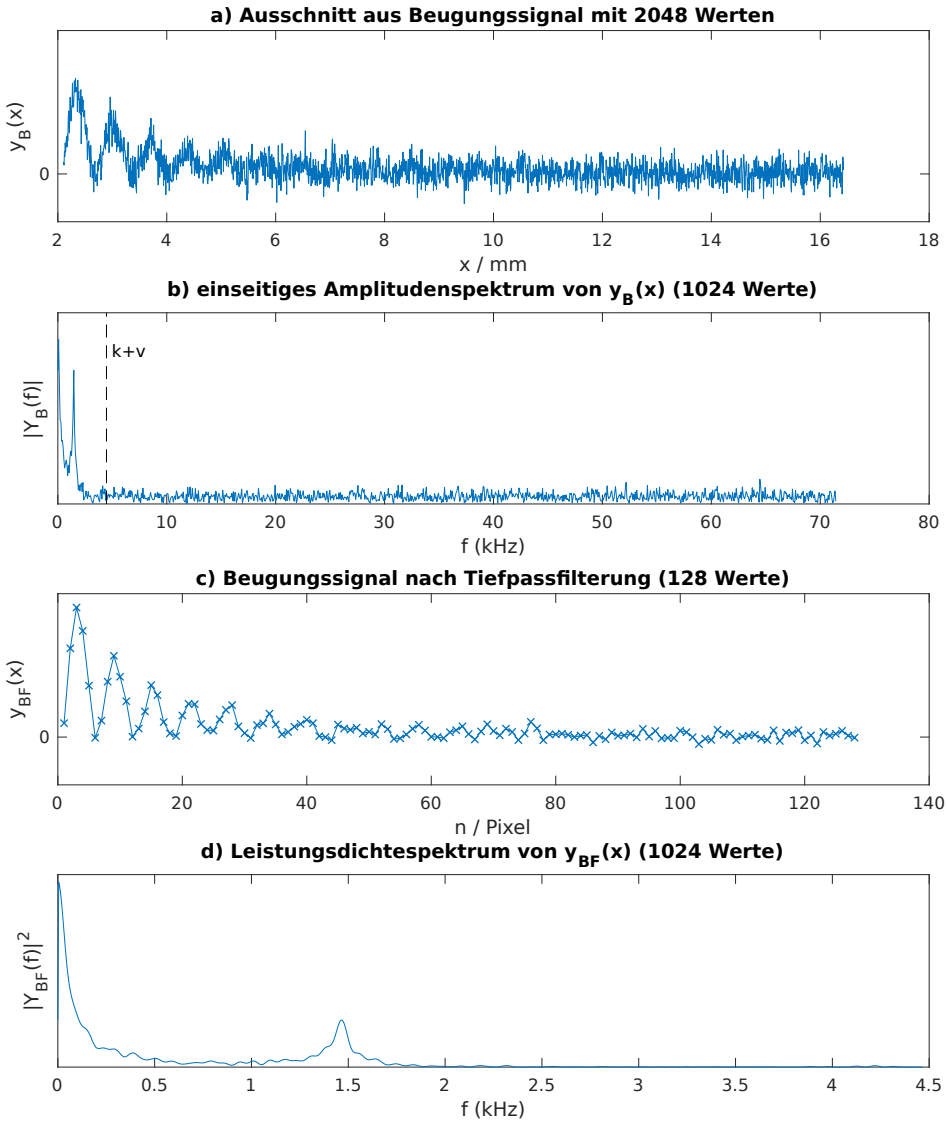

Abbildung 2.10: Entrauschen des Bildsignals und gleichzeitige Bestimmung des Drahtdurchmessers. (frei nach [21])

(a) Ausschnitt aus Fraunhoferbeugungsbild zu dem weißes Rauschen hinzugefügt wurde ($d = 0{,}1\,\text{mm}$; $b = 0{,}1\,\text{m}$; $\lambda = 675\,\text{nm}$)

(b) Amplitudenspektrum von (a) ($k + v = 64$)

(c) Rücktransformation der ersten $k+v$ Werte der Fouriertransformierten (b). Diese wird danach so mit Nullen aufgefüllt, dass sie wieder 2048 Werte enthält.

(d) Leistungsdichtespektrum von (c) mit ZOOM-Faktor $N/(2(k + v)) = 16$

3 Marktanalyse

Es gibt mehrere Hersteller, die Messgeräte anbieten, die den Durchmesser eines Drahtes während der Produktion messen können. Die Messgeräte können in die Produktionslinie integriert werden und der Draht läuft dabei durch das Messgerät. Die Messgeräte überwachen dabei regelmäßig den Durchmesser des Drahtes. Dabei arbeiten die Messgeräte nach einem optischen Messprinzip, da sie so gut in die Produktionslinie integriert werden können, weil der Draht so kontaktlos gemessen werden kann. Es werden Produkte mit unterschiedlichen Messverfahren angeboten. Die meisten Hersteller bieten Messgeräte an, die nach dem Laser-Scanning Messverfahren arbeiten, das in Abschnitt 2.4.1 vorgestellt wurde. Es gibt aber auch Hersteller die Produkte anbieten, die den Drahtdurchmesser mit einem CCD bestimmen. Des Weiteren werden auch Messgeräte angeboten, die mehrere Messachsen verwenden und dabei nicht nur den Durchmesser des Drahtes bestimmen, sondern auch dessen Ovalität. Manche Messgeräte können den Draht auch auf Fehlerstellen überprüfen.

Im Folgenden wird zuerst das *LMD H01B* Durchmessermessgerät von KJM vorgestellt, das die Grundlage dieser Arbeit ist. Danach werden die Konkurrenzprodukte zu dem KJM Messgerät vorgestellt. Es werden dabei nur Geräte vorgestellt, deren Messbereich vergleichbar mit dem des *LMD H01B* ist. Besonders wurde dabei auf den minimal messbaren Durchmesser geachtet. Abschließend ist in den Tabellen 3.1 und 3.2 eine Übersicht der wichtigsten vorgestellten Messgeräte dargestellt.

3.1 KJM LMD H01B

Das *LMD H01B* von KJM ist das Messgerät, dessen Algorithmus in dieser Arbeit durch eine Hardwareimplementierung auf einem FPGA beschleunigt werden soll. Das Messgerät beleuchtet den Draht mit einem Laser und eine Sammellinse bildet die Laserstrahlen auf ein CCD ab. Dies entspricht der Messanordnung in Abbildung 2.8. Dabei wird der Draht von Rollen geführt, damit sich der Draht immer an einer festen Position in der Messanordnung befindet. Es lassen sich unter anderem Drähte, Glasfasern und Monofile vermessen. Der Messbereich des Durchmessers beträgt 0,02 mm bis 1,8 mm. Eine Besonderheit des *LMD H01B* ist, dass das Messgerät tragbar ist. Es lässt sich somit sowohl in der Produktionslinie als auch außerhalb

z. B. zur Endkontrolle einsetzen. Die aktuell mögliche Messrate beträgt 40 Hz, wobei der CCD-Zeilensensor eine Messrate von ungefähr 3 kHz ermöglichen würde [20].

3.2 Zumbach

Zumbach bietet sowohl Durchmessermessgeräte an, die nach dem Laser-Scanning Messverfahren arbeiten, als auch Messgeräte die ein CCD verwenden.

Die Geräte der ODAC-Serie setzen das Laser-Scanning Messverfahren ein und es lassen sich sowohl opake als auch transparente Materialen vermessen. Es werden Geräte mit einer, zwei und drei Messachsen angeboten. Die Messköpfe der ODAC Serie verwenden eine adaptive Signalverarbeitung. Dabei werden alle für die Genauigkeit relevanten Parameter kontinuierlich überwacht und auch automatisch kompensiert. Dadurch sei eine regelmäßige Kalibrierung nicht nötig. Das *ODAC 2J* verwendet eine Messachse. Der minimal messbare Durchmesser beträgt 0,012 mm und das Messfeld ist 2 mm hoch. Die Messrate beträgt 240 Hz [27]. Das *ODAC 14XY* ist ein zweiachsiges Messgerät, dessen minimal messbarer Durchmesser 0,06 mm in der Standard-Ausführung (*ODAC 14XY-J*) und 0,015 mm in der Mikro Ausführung (*ODAC 14XY-JM*) beträgt. Das Messfeld einer Achse ist in der Standard-Ausführung 16 mm und in der Mikro Ausführung 3 mm hoch. Die Messrate pro Achse beträgt 500 Hz [26]. Das dreiachsige Messgerät *ODAC 13TRIO* ermöglicht einen minimal messbaren Durchmesser von 0,06 mm und das Messfeld einer Achse ist 16 mm hoch. Die Messrate pro Achse beträgt 600 Hz in der Standard-Ausführung und es ist auch noch eine Ausführung mit 1,5 kHz verfügbar [25].

Die MSD-Serie verwendet CCDs um die Position des Schattens zu bestimmen. Die Beleuchtung wird nicht mit einem Laser realisiert, sondern mit verschiedenfarbigen Leuchtdioden (LED) für die einzelnen Messachsen. Das *MSD 25* hat zwei Messachsen und bietet den kleinsten minimal messbaren Durchmesser der Serie von 0,25 mm. Somit sind die Messgeräte dieser Serie nicht für dünne Drähte geeignet. Die Messrate liegt bei 1 kHz [24].

3.3 LaserLinc

Auch von LaserLinc gibt es ein-, zwei- und dreiachsige Messgeräte, die das Laser-Scanning Verfahren anwenden. Es können opake und transparente Materialien vermessen werden. Das *TLAser103* verwendet eine Messachse, bietet einen Messbereich von 0,015 mm bis 2,2 mm und ermöglicht eine Messrate von 1,2 kHz bzw. optional 2,4 kHz [33]. Das Zweiachsen-Messgerät *TLAser203* hat einen Messbereich von 0,04 mm bis 2 mm und eine Messrate von 1,2 kHz pro Achse und optional 2,4 kHz pro Achse [34]. Das *Triton312* bietet drei Messachsen. Dafür beträgt der minimal

messbare Durchmesser nur noch 0,102 mm und die Messrate pro Achse liegt bei 600 Hz. Der maximal messbare Durchmesser beträgt 11,4 mm [35].

3.4 SIKORA

Die Durchmessermessgeräte von SIKORA verwenden pro Messachse ein CCD, das von einem Laser beleuchtet wird. Der Drahtdurchmesser wird mittels Beugungsanalyse aus dem Schattenbild bestimmt. Es werden keine Linsen verwendet, sodass dies der Messanordnung 2.5 entspricht.

Das *LASER 2005 XY* verwendet zwei Messachsen. Der Messbereich liegt bei 0,05 mm bis 5 mm bei einer Messrate von 1,2 kHz [18]. Die Geräte der LASER Series 6000 ermöglichen eine Messrate von bis zu 5 kHz (*LASER 6020 XY*) und ermöglicht damit auch die Erkennung von Knoten und Einschnürungen. Dafür liegt der minimal messbare Durchmesser nur bei 0,2 mm [19]. Beim *TIGER LASER 6010 XY* wurde der Fokus noch stärker auf die Erkennung von Knoten und Einschnürungen gelegt, da dies mit einer Frequenz von 512 kHz durchgeführt wird. Die Durchmesserbestimmung, die einen Messbereich von 0,1 mm bis 10 mm hat, wird aber nur mit 500 Hz durchgeführt [32].

3.5 AEROEL

Die Xactum Messgeräte von AEROEL verwenden ebenfalls das Laser-Scanning Verfahren und können sowohl opake als auch transparente Materialen vermessen. Es werden Messgeräte mit einer und zwei Messachsen angeboten. Auch bei diesen Geräten sei eine regelmäßige Kalibrierung nicht nötig. Das einachsige Messgerät *XLS40/1500/B* ermöglicht einen Messbereich von 0,06 mm bis 38 mm bei einer Messrate von 1,5 kHz [37]. Das *XLS13XY/480/B* ist ein zweiachsiges Messgerät, das einen Messbereich von 0,03 mm bis 3 mm besitzt. Die Messrate liegt bei 480 Hz [36].

Neben den Xactum Messgeräten gibt es noch den *HWS.2 - Handy Wire Scanner*, der ähnlich wie das *LMD H01B* von KJM tragbar ist. Dabei wird der Messvorgang aber durch das Drücken des Auslösers gestartet und soll nur wenige Sekunden dauern. Der Messkopf verwendet zwei Messachsen, die den Draht mit einer LED beleuchten und ein CCD als Sensor einsetzen. Der messbare Durchmesser liegt im Bereich von 0,1 mm bis 6 mm [13].

3.6 Weitere Hersteller

Ansonsten gibt es noch die BETA LaserMike Messgeräte von NDC Technologies und die InteliSENS Messgeräte von Proton Products, die beide das Laser-Scanning Verfahren anwenden. Deren minimal messbarer Durchmesser beträgt aber mindestens 0,1 mm. Deshalb werden die Geräte hier nicht weiter aufgeführt [4, 6, 5, 8, 17, 16].

3.7 Fazit

Beim Anfertigen der Marktanalyse konnte kein Messgerät gefunden werden, welches das gleiche Messverfahren wie das *LMD H01B* von KJM anwendet. Die Konkurrenzgeräte, die den kompletten Messbereich des KJM Messgeräts abdecken, sind das Zumbach *ODAC 2J* und *ODAC 14XY-JM* sowie das LaserLinc *TLAser103*. Diese Geräte wenden das Laser-Scanning Messverfahren an. Von den drei Messgeräten hat das LaserLinc *TLAser103* die größte Messrate mit 1,2 kHz in der Standard-Ausführung und optional 2,4 kHz.

Somit sollte das KJM Messgerät eine Messrate von mindestens 2 kHz erreichen, um einen Vorteil gegenüber den Konkurrenzprodukten zu erlangen. Wenn der Datenanalyse-Algorithmus beschleunigt wird, lässt sich diese Messrate auch erreichen, da der CCD-Zeilensensor eine Abtastrate von ca. 3 kHz unterstützt.

Tabelle 3.1: Übersicht über die wichtigsten vorgestellten Messgeräte der Marktanalyse (Teil 1 von 2)

Hersteller	KJM	Zumbach	LaserLinc
Produkbezeichnung	LMD H01B	ODAC 2J	TLAser103
Messverfahren	Beugungsanalyse	Laser-Scanning	Laser-Scanning
Lichtquelle	Laser	Laser	Laser
Sensor	CCD	Photozelle	Photozelle
Anzahl Messachsen	1	1	1
Messbereich (in mm)	0,02 - 1,8	0,012 - 2 [1]	0,015 - 2,2
Genauigkeit	*keine Angabe*	±0,3 µm [2]	±1,25 µm [3]
Wiederholgenauigkeit	*keine Angabe*	±0,2 µm [4]	±0,025 µm [5]
Auflösung (in µm)	*keine Angabe*	0,1 [6]	0,0025
Messrate (in Hz)	40 (Softcore Prozessor)	240	1200 / 2400 [7]
Besonderheiten	tragbar	Selbstkalibrierungsfunktion	

[1] Die Messfeldhöhe beträgt 2 mm. In der Praxis ist der grösste Objektdurchmesser gleich Messfeldhöhe minus Lageunsicherheit.
[2] für Durchmesser kleiner 1 mm
[3] für Durchmesser kleiner 2,04 mm
[4] $T = 1\,\mathrm{s}$, $\pm 3\,\sigma$
[5] für Durchmesser kleiner 2,04 mm, $T = 2\,\mathrm{s}$
[6] konfigurierbar
[7] optional

Tabelle 3.2: Übersicht über die wichtigsten vorgestellten Messgeräte der Marktanalyse (Teil 2 von 2)

Hersteller	SIKORA	AEROEL
Produkbezeichnung	LASER 2005 XY	XLS13XY/480/B
Messverfahren	Schattenmessung mit Beugungsanalyse	Laser-Scanning
Lichtquelle	Laser	Laser
Sensor	CCD	Photozelle
Anzahl Messachsen	2	2
Messbereich (in mm)	0,05 - 5	0,03 - 3
Genauigkeit	$\pm 0{,}5\,\mu m$ [1]	$\pm 1\,\mu m$ [2]
Wiederholgenauigkeit	$\pm 0{,}1\,\mu m$ [3]	$\pm 0{,}03\,\mu m$ [4]
Auflösung (in µm)	1	10/1/0,1/0,01
Messrate (in Hz)	1200	480
Besonderheiten		Selbstkalibrierungsfunktion

[1] jeweils $\pm 0{,}01\,\%$ des Messwertes
[2] $\pm 2\,\sigma$
[3] $T = 1\,s$, $\pm 3\,\sigma$
[4] für Durchmesser kleiner $0{,}5\,mm$, $T = 1\,s$, $\pm 2\,\sigma$

4 Voruntersuchungen

In diesem Kapitel werden zuerst die Voraussetzungen zur Beschleunigung des Algorithmus zur Bestimmung des Drahtdurchmessers von KJM beschrieben. Anschließend wird auf die Möglichkeiten zur Beschleunigung von Datenverarbeitungsalgorithmen eingegangen. Danach werden verschiedenen Hardware-Architekturen für Datenverarbeitungsalgorithmen dargestellt. Abschließend wird auf Basis der vorherigen Analysen eine geeignete Hardware-Architektur zur Implementierung des Datenanalyse-Algorithmus von KJM ausgewählt.

4.1 Zu Beschleunigender Algorithmus

Das Ziel dieser Arbeit ist es, den Algorithmus von KJM zur Bestimmung des Drahtdurchmessers zu beschleunigen. Dadurch soll eine höhere Abtastrate ermöglicht werden.

Das Messgerät von KJM wendet das in Abschnitt 2.4.2 beschriebene Messverfahren an. Dabei muss der Datenanalyse-Algorithmus das Beugungsbild auswerten, das mit einem CCD-Zeilensensor aufgenommen wird. Dazu bietet der Algorithmus von KJM zwei Möglichkeiten an. Zum einen kann man die Minima des Beugungsbildes auswerten, zum anderen lässt sich der Drahtdurchmesser mittels Fourier-Transformation bestimmen.

Bei der gegenwärtigen Implementierung wurde der Algorithmus in der Programmiersprache C implementiert. Dieser Programmcode wird dann von einem NIOS II Softcore Prozessor auf einem Altera Cyclone III FPGA ausgeführt. Die Ansteuerung des CCD-Sensors wird bereits durch ein entsprechendes FPGA Hardware-Module realisiert.

4.2 Allgemeine Möglichkeiten zur Beschleunigung von Datenverarbeitungsalgorithmen

Ein Datenverarbeitungsalgorithmus lässt sich dadurch beschleunigen, dass man einen höheren Grad an Parallelität erreicht. Dazu lassen sich verschiedene Methoden an-

wenden.

Eine recht einfach umzusetzende Technik ist die Aufteilung des Programmcodes auf mehrere nebenläufige Threads. Dabei muss aber die Hardware auch mehrere Prozessorkerne bereitstellen, damit die Threads auch parallel ausgeführt werden können. Ansonsten werden die Threads nur scheinbar parallel ausgeführt. Des Weiteren dürfen keine Datenabhängigkeiten zwischen den Threads bestehen, da man sie dann nicht gleichzeitig ausführen kann.

Mehr Parallelität lässt sich auch dadurch erreichen, dass man einzelne Instruktionen, die nicht voneinander abhängen gleichzeitigt ausführt.

Ein Algorithmus lässt sich auch dadurch parallelisieren, dass man eine mögliche Datenparallelität der zu verarbeitenden Daten ausgenutzt. Wenn die Operationen, die auf die Eingabedaten angewendet werden, nicht voneinander abhängen, kann man diese parallel ausführen.

Weiterhin kann man die Berechnungen auch durch das Einfügen von Pipeline-Stufen parallelisieren. Dies ist besonders sinnvoll, bei sequenziellen Berechnungen, die sich nicht parallel ausführen lassen und einen Datenstrom verarbeiten. Dabei wird der Grad an Parallelität und der Durchsatz erhöht.

4.3 Hardware-Architekturen für Datenverarbeitungsalgorithmen

Im Folgenden werden verschiedene Hardware-Architekturen zur Implementierung eines Datenverarbeitungsalgorithmus miteinander verglichen. Dabei wird auf eine mögliche Implementierung auf einem Prozessor bzw. einem System-on-Chip (SoC), einem Field-Programmable Gate Array (FPGA) oder einem rekonfigurierbaren System-on-Chip (rSoC) eingegangen.

4.3.1 Prozessor (System-on-Chip)

Ein System-on-Chip vereint einen Mikroprozessor mit weiteren Komponenten wie Speicher, Controller und digitalen Signalprozessoren. Durch die Integration der Komponenten auf einem einzelnen Chip wird die Größe der benötigten Platine verkleinert. Darüber hinaus wird der Datentransfer zwischen den einzelnen Komponenten auf dem SoC beschleunigt und die Kosten werden reduziert. Mit einem Prozessor, der mehrere Kerne besitzt, lassen sich mehrere Datensätze parallel auswerten.

In Prozessoren, die mehrere Pipeline-Stufen besitzen, wird der Durchsatz an Instruktionen dadurch erhöht, dass die Instruktionen aufgeteilt werden und diese Teile parallel abgearbeitet werden. Eine weitere Möglichkeit die Instruktionsparallelität zu erhöhen wird von superskalaren Prozessoren angewendet, indem sich je nach Grad

der Superskalarität eine gewisse Anzahl an Instruktionen parallel ausführen lassen. Dabei müssen die jeweiligen Instruktionen aber unabhängig voneinander ausgeführt werden können.

4.3.2 Field Programmable Gate Array

Ein FPGA besteht aus Logik-Elementen, die über Switch-Matrizen miteinander verbunden werden können. Die Logik-Elemente bestehen dabei aus Lookup-Tabellen (LUTs) in denen die logischen Funktionen realisiert werden und Flip-Flops. Daneben besitzen FPGAs meist weiter Zusatzelemente wie RAM-Blöcke, DSP-Blöcke zur Beschleunigung von beispielsweise Multiplikationen und Taktgeneratoren zur Erzeugung interner Takte aus einem externen Taktsignal. Ein- und Ausgabepins stellen die Verbindung zu externen Bauteilen bereit.

Bei der Programmierung eines FPGAs hat man nicht mehr die Beschränkung, dass einzelne Instruktionen sequenziell auf einem Prozessor ausgeführt werden müssen. Stattdessen programmiert man eine Vielzahl an Logik-Elementen, die dann miteinander verschalten werden. Somit lassen sich Instruktionen aus dem Programmcode, durch eine entsprechende Hardwareimplementierung auf einem FPGA parallel ausführen.

4.3.3 Rekonfigurierbares System-on-Chip

Auf einem rekonfigurierbaren System-on-Chip wird ein Mikroprozessor mit einem FPGA kombiniert. Die rekonfigurierbare Logik des FPGA lässt sich dazu nutzen Teile des Algorithmus zu beschleunigen. Der Nachteil eines rSoC ist, dass zwischen dem Prozessor und dem rekonfigurierbaren Teil ein hoher Kommunikationsaufwand entsteht. Auch der Entwicklungsaufwand kann sich erhöhen, da man sowohl die Software als auch den Hardware-Teil testen muss. Zudem muss man natürlich sowohl den Prozessor als auch den rekonfigurierbaren Teil bezahlen. Dadurch können sich die Materialkosten erhöhen. Wenn der Teil, der in Software ausgeführt wird, klein ist, dann ist es sinnvoller die komplette Implementierung auf einem FPGA zu realisieren.

4.4 Auswahl einer geeigneten Hardware-Architektur

Die bisherige Implementierung verwendet einen Softcore Prozessor, also einen Prozessor, der auf ein FPGA synthetisiert wird. Da Softcore Prozessoren meist deutlich langsamer sind als vergleichbare Hardcore Prozessoren sind, lässt sich der Algorithmus auf einem Hardcore Prozessor schneller ausführen.

Zur Beschleunigung des Datenanalyse-Algorithmus lässt sich die Aufteilung des Programmcodes auf mehrere nebenläufige Threads recht einfach umsetzen. Dies wurde in dem Programmcode von KJM auch gemacht, indem für die Berechnung einer einzelnen CCD-Zeile jeweils ein eigener Thread erzeugt wird.

Für die Bestimmung des Drahtdurchmessers wird aber nicht nur ein Prozessor zur Datenanalyse benötigt, sondern auch Controller zur Ansteuerung des CCDs, ein ADC bzw. die Ansteuerung eines externen ADCs und Peripherie zur Ausgabe des ermittelten Drahtdurchmessers. Dazu wäre ein SoC geeignet, das alle benötigten Komponenten bereits enthält. Dabei ist es aber unwahrscheinlich, dass man ein SoC findet, das einen passenden CCD-Controller integriert hat. Dafür lässt sich dann ein rSoC verwenden. Dabei kann man den CCD-Controller auf dem rekonfigurierbaren Teil implementieren und die Auswertung der CCD-Zeile auf dem Prozessor durchführen. Eine weitere Möglichkeit den rekonfigurierbaren Teil zu verwenden ist, die von dem Datenanalyse-Algorithmus zur Drahtdurchmesserbestimmung verwendete FFT auf dem rekonfigurierbaren Teil zu implementieren.

Der Datenanalyse-Algorithmus lässt sich mit einem FPGA am besten beschleunigen, da man damit den höchsten Grad an Parallelität erreichen kann. Die Implementierung des Algorithmus lässt sich nach folgendem Ansatz realisieren. Das CCD, das bei der Bestimmung des Drahtdurchmessers verwendet wird, besitzt eine Zeile mit einer gewissen Anzahl an Pixeln. Diese Zeile wird von dem Algorithmus größtenteils in einer Schleife vom Anfang bis zum Ende durchlaufen. Ergo bietet sich für die Implementierung auf einem FPGA die Verarbeitung der CCD-Zeile als Datenstrom an. Mit diesem Ansatz lässt sich auch gut Pipelining anwenden. Dabei kann man die Berechnung parallelisieren, ohne dass sehr viele Ressourcen des FPGAs in Anspruch genommen werden. So wendet der Algorithmus bei der Drahtdurchmesserbestimmung zuerst eine Filterfunktion an. Die gefilterten Daten werden dann differenziert und damit werden dann die Minima des Beugungsbildes gesucht. Wenn man diese Berechnungen so implementiert, dass sie die Daten des CCDs als Datenstrom entgegen genommen werden und auch wieder als Datenstrom ausgeben werden, dann lassen sich diese Berechnungsblöcke aneinanderreihen und somit eine Pipeline bauen. Auch innerhalb eines solchen Blockes ist es sinnvoll Pipelining anzuwenden, da damit die Parallelität erhöht wird. Da die Pixel einer Zeile des CCDs die Pipeline nacheinander durchlaufen, ist die erhöhte Latenz durch mehr Pipeline-Stufen nicht relevant. Eine größere Zahl an Pipeline-Stufen ist auch sinnvoll, um die maximal mögliche Taktfrequenz zu erhöhen. Mit diesem Implementierungsansatz lässt sich auch der Speicherbedarf reduzieren, da neben den Daten, die vom CCD ausgegeben werden, nicht auch noch die gefilterten Daten und die Ableitung davon gespeichert werden muss. Ein geringer Bedarf an Ressourcen des FPGAs ist auch aus wirtschaftlicher Sicht interessant, da sich so vielleicht ein kleineres FPGA verwenden lässt und sich damit die Materialkosten reduzieren lassen. Bei der Implementierung

des Algorithmus lässt sich auch die Datenparallelität des Beugungsbilds ausnutzen. So lässt sich die linke und rechte Seite des Beugungsbildes parallel auswerten.

Je nachdem wie die Ausgabe des Drahtdurchmessers realisiert ist, benötigt man keinen Prozessor mehr. Dabei kommt es darauf an, welche Schnittstellen das Messgerät bereitstellt. Wenn sich diese Schnittstelle einfach auf einem FPGA implementieren lässt, kann man auf den Prozessor des rSoC verzichten. Wenn der Drahtdurchmesser allerdings über ein komplexes Netzwerkprotokoll ausgegeben werden soll, ist es sinnvoll, dies in Software zu realisieren.

4.5 Auswahl aktueller Hardwarekomponenten

Zur Implementierung des Algorithmus zur Drahtdurchmesserbestimmung soll ein FPGA verwendet werden. Im folgenden Abschnitt wird eine Auswahl an FPGAs vorgestellt, die sich dafür eignen. Dabei werden verschieden FPGA-Serien von Intel (ehemals Altera) und Xilinx miteinander verglichen.

Des Weiteren wird auch eine Auswahl geeigneter CCD-Zeilensensoren vorgestellt.

4.5.1 Field-Programmable Gate Array (FPGA)

Zurzeit verwendet KJM ein FPGA der Altera Cyclone III Serie für sein *KJM LMD H01B* Messgerät. Da dieses FPGA nicht mehr aktuell ist und auch keine Funktionen zum Schützen des Designs vor dem Anfertigen unautorisierter Kopien und Reverse Engineering besitzt, soll in Zukunft ein moderneres FPGA verwendet werden.

Bevor jedoch auf alternative FPGAs eingegangen wird, sollen zuerst die Funktionen des derzeit verwendeten *EP3C25U256C8N* der Altera Cyclone III Serie aufgeführt werden. Das Altera *EP3C25* besitzt 24 624 Logik-Elemente und 66 Speicherblöcke. Diese Speicherblöcke besitzen eine Speicherkapazität von jeweils 9 kbit. Außerdem besitzt das FPGA noch 66 eingebettete Multiplizierer. Diese eingebetteten Multiplizierer lassen sich entweder als ein 18 x 18 bit oder zwei 9 x 9 bit Multiplizierer konfigurieren. Der Preis des FPGAs liegt aktuell bei 49,83 € [1].

Um die FPGAs besser vergleichen zu können, wird bei der Auswahl eines geeigneten FPGAs das Model ausgewählt, das eine vergleichbare Anzahl an Logik-Elementen besitzt, wie das zurzeit verwendete Altera *EP3C25* FPGA (ca. 25 000 Logik-Elemente). Die genaue FPGA-Größe, die man für die Implementierung benötigt, kann man erst nach der Synthese der Implementierung bestimmen. Die Synthese muss man mit dem entsprechenden Tool des jeweiligen Herstellers durchführen. Bei der Bestimmung der FPGA-Größe kommt es auch darauf an, ob die Hardwareimplementierung des Datenanalyse-Algorithmus den Softcore Prozessor vollständig ersetzt

[1] Mouser Electronics (25.09.2017)

oder ob dieser weiterhin benötigt wird (z. B. zur Weiterverarbeitung des Drahtdurchmessers).

Beim Vergleichen der FPGAs von Intel und Xilinx muss man jedoch beachten, dass sich diese nicht direkt miteinander vergleichen lassen. Die Logik-Elemente der Intel FPGAs bestehen aus einer Lookup-Tabelle (LUT), die vier Eingänge besitzt und einem programmierbaren Register. Die FPGAs von Xilinx verwenden mehrere konfigurierbare Logik-Blöcke (CLB). Die CLBs bestehen aus jeweils zwei Teilen (slices). Jeder Teil besteht wiederum aus vier LUTs, die sechs Eingänge besitzen, und acht Flip-Flops. Die LUTs mit sechs Eingängen können auch in zwei LUTs mit fünf Eingängen aufgeteilt werden. Damit man die FPGAs, die LUTs mit sechs Eingängen verwenden, mit einfachen FPGAs, die LUTs mit vier Eingängen verwenden, vergleichen kann, gibt Xilinx die Logik-Kapazität des FPGAs auch in Logik-Zellen an. Das Verhältnis zwischen der Anzahl an Logik-Zellen und der Anzahl an LUTs mit sechs Eingängen gibt Xilinx mit 1,6:1 an [1].

Intel

Für die Implementierung der Drahtdurchmesserberechnung kommt von Intel die Cyclone 10 LP Serie und die Max 10 Serie in Betracht.

Cyclone 10 LP

Die Cyclone 10 LP Serie bietet sich für kleine Anwendungen an, bei denen es auf geringe Kosten ankommt. Wie auch der zurzeit verwendete Altera Cyclone III besitzen die FPGAs der Cyclone 10 LP Serie Logik-Elemente mit vier Eingängen, 9 kbit großen Speicherblöcken und 18 x 18 bzw. 9 x 9 Multiplizierern. Der *10CL025* besitzt 24 624 Logik-Elemente, 66 Speicherblöcke und 66 Multiplizierer. Damit ist das FPGA in diesen Bereichen identisch zu dem derzeit verwendeten Altera Cyclone III *EP3C25*. Leider besitzt die Cyclone 10 LP Serie auch keine Kopierschutzmechanismen. In der Konfiguration *10CL025YU256C6G* kostet der FPGA zurzeit 26,78 € [2] und ist damit günstiger als das *EP3C25* [15].

Max 10

Da FPGAs einen flüchtigen Static Random-Access-Memory (SRAM) verwenden, muss die Konfiguration bei jedem Start neu aus einem Flash-Speicher geladen werden. Im Gegensatz zu klassischen FPGAs hat die Max 10 Serie schon einen Flash-Speicher auf dem Chip integriert. Damit ist das Design auch besser gegen unautorisiertes Kopieren geschützt, da es dadurch schwieriger ist den Bitstream beim Programmieren des FPGAs abzugreifen. In dem eingebetteten Flash-Speicher können bei manchen Geräten auch zwei Konfigurationen gespeichert werden.

[2] Mouser Electronics (25.09.2017)

Die Max 10 Serie besitzt aber auch noch zusätzliche Funktionen um das Design gegen Kopieren, Reverse Engineering und fremden Zugriff zu schützen [22]. So unterstützt der Chip die Verschlüsslung nach dem Advanced Encryption Standard (AES) mit 128-bit langen Schlüsseln. Des Weiteren besitzt jeder Max 10 Chip eine 64-bit breite global eindeutige ID. Damit lässt sich sicherstellen, dass das Design nur auf einem autorisierten Chip ausgeführt wird.

Außerdem haben manche Max 10 FPGAs einen Analog-Digital-Wandler (ADC) auf dem Chip integriert. Dieser besitzt eine Bitbreite von 12 bit und unterstützt eine Abtastrate von bis zu 1 MHz. Damit müsste man keinen externen ADC mehr verwenden, um das analoge Signal des CCDs in ein digitales Signal zu wandeln. Leider ist der ADC damit für das verwendete CCD zu langsam.

Auch die Max 10 FPGAs gibt es in einer ähnlichen Ausstattung wie das derzeit verwendete FPGA. So besitzt das *10M25* 25 000 Logik-Elemente, 75 Speicherblöcke und 55 Multiplizierer. Dieses FPGA gibt es dann wiederum in unterschiedlichen Ausstattungsvarianten. So unterstützt das *10M25SCE144C8G* (33,58 € [3]) nur die Kernfunktionen, während das *10M25SAE144C8G* (41,31 € [4]) auch einen ADC besitzt und die Möglichkeit bietet zwei Konfigurationen im Flash-Speicher abzulegen. Diese Konfiguration lässt sich bei dem *10M25SAE144C8G* auch aus der Ferne updaten [23].

Xilinx

Von Xilinx bietet sich für die Implementierung des Algorithmus zur Drahtdurchmesserbestimmung vor allem die Spartan Serie an.

Spartan

Aus der Spartan Reihe von Xilinx könnte man entweder einen FPGA aus der älteren Spartan-6 Serie oder aus der neueren Spartan-7 Serie verwenden. In der Spartan-6 Serie wird aber erst ab dem *XC6SLX75* mit 74 637 Logik-Zellen die Verschlüsselung des Bitstreams unterstützt [28]. Dadurch lässt sich das Design gegen unerwünschtes Kopieren und Reverse Engineering schützen. Aus der Spartan-7 Serie unterstützt jeder FPGA die Verschlüsslung nach AES mit 256-bit langen Schlüsseln [2]. Des Weiteren gibt es in der Spartan-7 Serie auch FPGAs mit einem integrierten Analog-Digital-Wandler (ADC).

Der *XC7S25* besitzt 3650 slices was 23 360 Logik-Zellen entspricht. Außerdem besitzt das FPGA noch 45 Speicherblöcke mit jeweils 36 kbit und 80 DSP-Slices. Diese DSP-Slices bestehen unter anderem aus einem 25 x 18 bit Multiplizierer. Des Weiteren hat das FPGA auch noch einen integrierten ADC. Dieser besitzt eine

[3] Mouser Electronics (25.09.2017)
[4] Mouser Electronics (25.09.2017)

Bitbreite von 12 bit und unterstützt eine Abtastrate von bis zu 1 MHz. Damit ist auch dieser ADC zu langsam für das verwendete CCD.

In der Version *XC7S25-1CSGA324C* kostet das FPGA 35,61 € [5] [3]. Darüber hinaus bietet Xilinx auch einen frei verfügbaren FFT IP Core an, der sich auch in der kostenlosen Vivado HL WebPACK Edition verwenden lässt.

Fazit

Abschließend lässt sich sagen, dass man für die Implementierung der Drahtdurchmesserbestimmung am besten entweder einen Max 10 FPGA oder einen Spartan-7 FPGA verwendet. Beide unterstützen eine Bitstream Verschlüsslung zum Schutz des Designs vor unautorisiertem Kopieren und Reverse Engineering. Die Max 10 FPGAs haben den Vorteil, dass sie schon einen integrierten Flash-Speicher haben und damit kein externer Speicher mehr nötig ist um die Konfiguration zu speichern.

4.5.2 Charge Coupled Device (CCD)

Die Ziele bei der Drahtdurchmesserbestimmung sind, den Drahtdurchmesser mit einer möglichst hohen Abtastrate möglichst genau bestimmen zu können. Das Beugungsbild wird dabei mit einem monochromatischen CCD-Zeilensensor aufgenommen.

Um die Genauigkeit der Drahtdurchmesserbestimmung zu verbessern, benötigt man einen CCD-Zeilensensor mit einer möglichst kleinen Pixelgröße. Da der Bildsensor eine Breite von ungefähr 30 mm haben soll, muss der CCD-Zeilensensor je nach Pixelgröße eine entsprechende Anzahl an Pixeln besitzen.

Das Auslesen der Pixelwerte erfolgt seriell. Man legt eine Taktfrequenz an den CCD-Zeilensensor an und der Sensor gibt der Reihe nach in jedem Takt den Wert eines Pixels aus. Diese Pixelwerte werden als analoges Signal ausgegeben, welches noch in ein digitales Signal gewandelt werden muss. Umso höher die Frequenz ist, mit der sich der CCD-Zeilensensor auslesen lässt, umso größer wird die Abtastrate für eine Zeile.

Die Abtastrate ist aber nicht nur von der Datenrate abhängig, sondern auch von der Belichtungszeit, die benötigt wird, um ein korrekt belichtetes Bild zu erhalten. Somit ist die Datenrate der limitierende Faktor für die Abtastrate. Wenn aber das Bild nach dieser Zeit noch unterbelichtet ist, muss man die Abtastrate verringern, um ein korrekt belichtetes Bild zu erhalten. Die Belichtungszeit lässt sich aber verkürzen, indem man einen Sensor wählt, der eine hohe Empfindlichkeit hat.

Somit sollte man für die Bestimmung des Drahtdurchmessers einen CCD-Zeilensensor wählen, der eine möglichst kleine Pixelgröße und gleichzeitig eine möglichst ho-

[5] Avnet (26.09.2017)

he Empfindlichkeit hat. Dabei muss man jedoch beachten, dass sich dadurch das Rauschen des Sensors erhöhen kann.

In Tabelle 4.1 ist eine Übersicht über die im Folgenden vorgestellten Sensoren dargestellt.

Toshiba TCD1209DG

Die Messanordnung zur Bestimmung des Drahtdurchmessers verwendete ursprünglich den *TCD1209DG* von Toshiba. Dieser Sensor hat 2048 Pixel mit einer Pixelgröße von 14 µm. Die Empfindlichkeit liegt bei ungefähr 31 $\frac{\mathrm{V}}{\mathrm{lx\,s}}$ und das Rauschen wird mit 0,6 mV angegeben. Die Sensorwerte lassen sich mit maximal 20 MHz auslesen [29].

Sony ILX553B

Zukünftig sollte der *ILX553B* von Sony für das Messgerät verwendet werden. Der Sensor hat 5150 Pixel mit einer Größe von 7 µm und einer Empfindlichkeit von 14,8 $\frac{\mathrm{V}}{\mathrm{lx\,s}}$. Durch die kleineren Pixel lässt sich die Genauigkeit der Durchmesserbestimmung verbessern. Die maximale Datenrate liegt bei 16 MHz [14].

Toshiba TCD1711DG und TCD1706DG

Von Toshiba gibt es auch noch zwei Sensoren (*TCD1711DG* und *TCD1706DG*), die noch kleinere Pixel (4,7 µm) besitzen. Dafür hat sich bei den beiden Sensoren das Rauschen gegenüber dem Toshiba *TCD1209DG* von 0,6 mV auf 1,0 mV erhöht. Der *TCD1711DG* hat dabei 7450 Pixel und der *TCD1706DG* 7400 Pixel. Aufgrund der größeren Pixelzahl im Vergleich zu den zuvor vorgestellten Sensoren besitzt der *TCD1711DG* zwei Ausgabekanäle und der *TCD1706DG* vier Ausgabekanäle. Somit lässt sich die benötige Zeit zum Auslesen aller Pixel reduzieren. Die Empfindlichkeit beider Sensoren liegt bei 15 $\frac{\mathrm{V}}{\mathrm{lx\,s}}$. Das Auslesen eines Ausgabekanals ist bei dem *TCD1711DG* mit 30 MHz möglich und bei dem *TCD1706DG* mit 25 MHz [31, 30].

Fazit

Durch die Verwendung des Toshiba *TCD1711DG* oder *TCD1706DG* CCD-Zeilensensors kann man die Genauigkeit der Messung des Drahtdurchmessers erhöhen, weil diese beiden CCDs kleinere Pixel besitzen als das zurzeit verwendete Sony *ILX553B*. Dabei muss man jedoch beachten, dass sich durch die kleineren Pixel das Rauschen vergrößern kann. Dadurch, dass die beiden Toshiba CCDs zwei bzw. vier Ausgabekanäle besitzen und deren Abtastrate höher ist als bei dem Sony CCD, lässt

sich mit diesen beiden CCDs trotz der höheren Pixelzahl die maximal mögliche Abtastrate erhöhen. Dabei wird aber auch das Auslesen des CCDs aufwendiger, da man mehrere Kanäle gleichzeitig in das FPGA einlesen muss. Dabei muss der verwendete ADC auch in der Lage sein, mehrere Kanäle verarbeiten zu können.

Hersteller	Toshiba	Sony	Toshiba	Toshiba
Produkbezeichnung	TCD1209DG	ILX553B	TCD1711DG	TCD1706DG
Pixel	2048	5150	7450	7400
Pixelgröße / µm	14	7	4,7	4,7
Breite / mm	28,6	36,05	35	34,8
Empfindlichkeit (Typ.) / $\frac{\mathrm{V}}{\mathrm{lx\,s}}$	31	14,8	15	15
Datenrate pro Ausleseregister / MHz	20	16	30	25
Ausleseregister	1	1	2	4

Tabelle 4.1: Übersicht über die vorgestellten CCD-Zeilensensoren.

5 Implementierung

Das Ziel dieser Arbeit ist es den von KJM vorgegebenen Algorithmus zur Drahtdurchmesserbestimmung zu beschleunigen. Wie im vorherigen Kapitel 4 bereits diskutiert, lässt sich dies bewerkstelligen, indem man den Algorithmus auf einem FPGA implementiert. Durch die Parallelisierung des in C vorgegebenen sequenziellen Programms von KJM kann die Berechnung des Drahtdurchmessers beschleunigt werden.

Momentan wird ein Altera Cyclone III FPGA genutzt. Deswegen soll die neue Implementierung auch erstmal auf diesem FPGA laufen. Da die Cyclone III Serie von Altera allerdings aus dem Jahre 2007 stammt und somit nicht mehr aktuell ist, soll die Implementierung zukünftig auch auf moderneren FPGAs laufen. Des Weiteren bietet der Cyclone III auch keinen Schutz gegen *Reverse Engineering*. Deswegen wurden in Abschnitt 4.5.1 Vorschläge für andere FPGAs gemacht, auf denen die Implementierung in Zukunft laufen könnte.

Zur Implementierung wurde die Hardwarebeschreibungssprache *VHDL* verwendet. Am Anfang bestand die Überlegung, die Hardwarebeschreibungssprache *Bluespec* zu verwenden. Da *Bluespec* eine High-Level-Hardwarebeschreibungssprache ist, lässt sich damit die Implementierung auf einer höheren Abstraktionsebene beschreiben. Der Vorteil von Bluespec ist, dass durch das Beschreiben auf einer höheren Abstraktionsebene die Steuerlogik vom Bluespec Compiler weitgehend automatisch hergeleitet werden kann. Dadurch reduziert sich der Implementierungsaufwand und die Entwicklungszeit verkürzt sich. Außerdem ist das Testen einer Implementierung in Bluespec einfacher als in einer Hardwarebeschreibungssprache, die auf der Registertransferebene arbeitet. Die Bluespec Simulationsumgebung arbeitet nämlich nicht mit einzelnen Signalleitungen, sondern kann das Design auf einer höheren Ebene verifizieren. Das Bluespec Programm lässt sich dann in Verilog oder VHDL übersetzen. Dies ist nötig, da die Synthesewerkzeuge der FPGA-Hersteller für Verilog bzw. VHDL die beste Tool-Unterstützung anbieten. Da man für *Bluespec* aber eine Lizenz benötigt, wurde stattdessen *VHDL* gewählt. Da die verkauften Stückzahlen des Messgerätes vergleichsweise gering sind, müsste KJM die Lizenzkosten auf den Produktpreis umlegen.

Das Messgerät von KJM verwendet das Messverfahren, das in Abschnitt 2.4.2 beschrieben wurde. Bei der Anwendung dieses Messverfahrens wird der Drahtdurch-

messer aus den Minima des Beugungsbildes bestimmt. Des Weiteren lässt sich wie in Abschnitt 2.4.2 beschrieben auch die Diskrete Fourier-Transformation anwenden. Das Programm von KJM unterstützt beide Methoden, wobei die Minima-Auswertung als Standard-Methode bezeichnet wird und die Fourier-Transformation-Auswertung als FFT-Methode bezeichnet wird.

In Abbildung 5.1 ist eine Übersicht über die verwendeten Funktionsblöcke dargestellt. Darin ist auch der Datenfluss der Implementierung abgebildet. Auf die Darstellung des Kontrollflusses wurde zur besseren Übersicht verzichtet.

Bei der Implementierung der Standard-Methode wird als erstes ein Filterblock benutzt. Das gefilterte Signal wird dann differenziert um aus dieser Ableitung die Nullstellen bzw. die Minima zu bestimmen. Dieser Vorgang läuft komplett streambasiert ab. Das heißt, dass das Beugungsbild als Datenstrom Pixel für Pixel eingelesen wird. Der Funktionsblock, der die Minima bestimmt, gibt dann die Positionen aus, an denen ein Minimum gefunden wurde. Daraus berechnet ein weiterer Funktionsblock den Drahtdurchmesser.

Die FFT-Methode transformiert das Beugungssignal anders als in Abschnitt 2.4.2 nur einmal in den Frequenzbereich und wird auch nicht wieder in den Zeitbereich zurücktransformiert. Stattdessen wird direkt aus dem transformierten Beugungssignal der Höchstwert k (wie in Abschnitt 2.4.2) bestimmt und damit dann die Drahtdurchmesserberechnung durchgeführt.

Im Folgenden soll auf die einzelnen Funktionsblöcke genauer eingegangen werden. Dabei kann jedoch nicht die genaue Implementierung dargestellt werden, da die KJM-Implementierung vertraulich behandelt werden muss und bei genauer Darstellung zu viel von dem ursprünglichen Programmcode offengelegt werden würde.

5.1 CCD- und ADC-Controller

Damit man den Drahtdurchmesser aus dem gemessenen Beugungsbild bestimmen kann, müssen zuerst die Messdaten von dem CCD eingelesen werden. Zur Ansteuerung des CCD benötigt man einen Controller. Die Signale, die das CCD ausgibt sind allerdings noch nicht digital sondern analog. Aus diesem Grund muss das analoge Signal des CCDs erst in einen digitalen Datenstrom gewandelt werden. Dazu wird ein externer Analog-Digital-Wandler (ADC) verwendet. Auch dieses Bauteil wird mit einem Controller angesteuert. Das digitale Signal, das von dem ADC eingelesen wird, ist 10 Bit breit. Die Länge einer CCD-Zeile beträgt ungefähr 5000 Pixel.

Konkret verwendet das Messgerät das *ILX553B* CCD von Sony und den *VSP2560* ADC von Texas Instruments. Für beide Komponenten wurde eine Controller-Implementierung und Simulationsmodelle bereitgestellt.

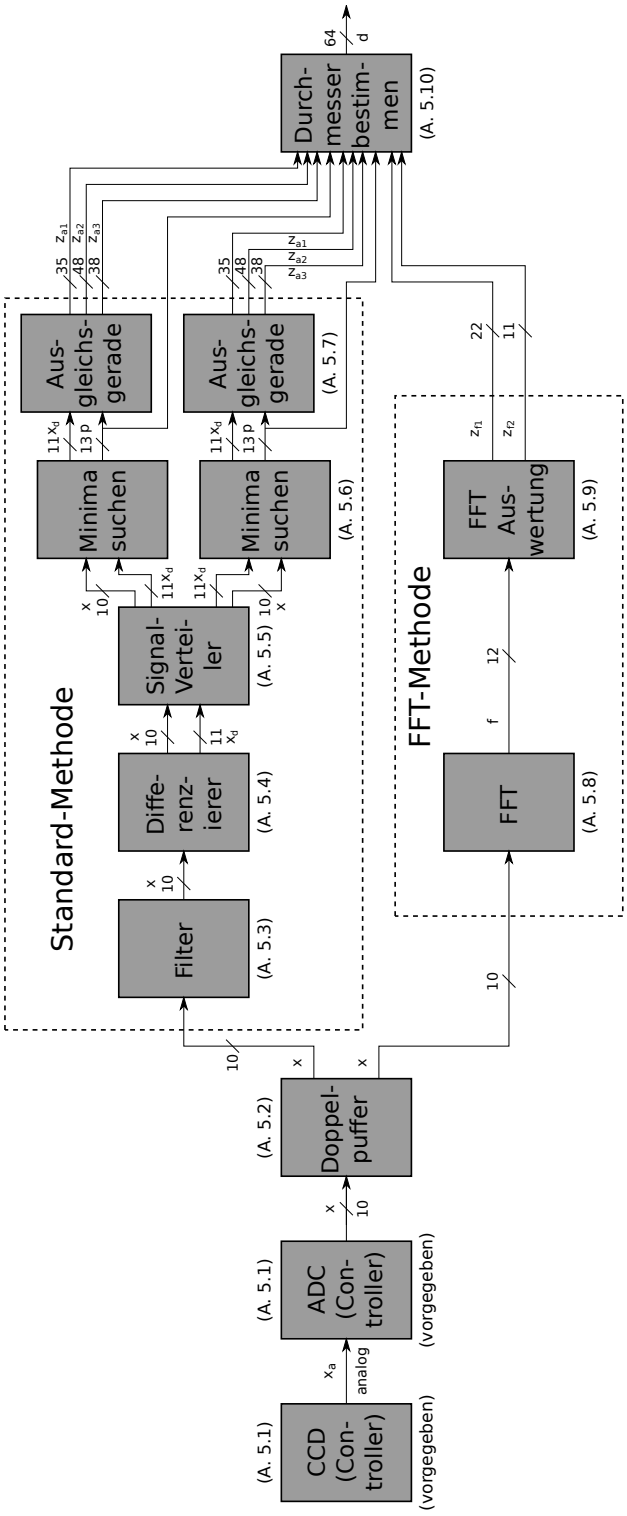

Abbildung 5.1: Übersicht über die bei der Implementierung verwendeten Funktionsblöcke. Der Datenfluss zwischen den Funktionsblöcken wird durch Pfeile dargestellt.
x: Beugungsbild, x_d: Ableitung Beugungsbild, p: Position Minimum, z_a: Zwischenwerte Ausgleichsgerade, f: Fourier-Transformation, z_f: Zwischenwerte FFT Auswertung, d: Drahtdurchmesser

5.2 Doppelpuffer

Der Doppelpuffer ist nötig, da das Beugungsbild seriell von einem CCD-Zeilensensor eingelesen wird. Das Einlesen einer vollständigen Zeile vom CCD-Sensor ist mit einer Abtastrate von ungefähr 3 kHz möglich. Durch den Doppelpuffer kann man während dem Einlesen des neuen Beugungsbildes bereits das vorherige Beugungsbild auswerten. Durch diesen Vorgang wird die Berechnung stark beschleunigt, da nach dem Bestimmen des Drahtdurchmessers aus dem Beugungsbild, nicht erst darauf gewartet werden muss, dass das neue Beugungsbild eingelesen wird. Stattdessen lässt sich durch ein einfaches Umschalten direkt mit der Auswertung des parallel eingelesenen neuen Beugungsbilds beginnen. Mit dem Umschalten muss man jedoch warten, bis das neue Beugungsbild vollständig eingelesen ist. Dabei wurde der Doppelpuffer so implementiert, dass er aus zwei getrennten RAM-Blöcken besteht. Zum Speichern der beiden 5000 x 10 bit breiten Bilder benötigt man 12 der 9 kbit großen Speicherblöcke des Altera Cyclone III FPGAs. Auch wenn man nur einen RAM-Block mit getrennten Adressbereichen verwenden würde, würde man 12 Speicherblöcke benötigen. Durch einen Multiplexer lässt sich dann das neue Beugungsbild entweder in den einen oder den anderen RAM-Block einlesen. Währenddessen lässt sich aus dem anderen RAM-Block das zuvor eingelesene Beugungsbild auslesen. Weil der darauffolgende Algorithmus das Beugungsbild als Stream verarbeitet, muss der Doppelpuffer in der Lage sein, einen solchen Stream auszugeben. Zu diesem Zweck wird ein Counter verwendet, der von einer Startadresse entweder hoch oder runter zählt. Bei der Standard-Methode wird das ganze Beugungsbild als ein Datenstrom ausgegeben, während bei der FFT-Methode die linke und die rechte Seite des Beugungsbildes getrennt ausgegeben wird. Bei der Ausgabe der linken Seite des Beugungsbildes muss der Counter von der Startadresse runter zählen, damit das Beugungsbild von innen nach außen gelesen wird. Bei der rechten Seite ist das dann umgekehrt.

Bei der Beschreibung von Hardware muss neben dem eigentlichen Datensignal auch immer noch ein Kontrollsignal (valid-Signal) ausgegeben werden. Dieses Signal gibt an, ob die aktuell ausgegebenen Daten gültig sind.

Zur Auswertung des Beugungsbildes wird dann entweder die Standard-Methode oder die FFT-Methode verwendet. Da der FFT-Algorithmus nur Daten, deren Länge eine Zweierpotenz ist, verarbeiten kann, muss der Puffer sicherstellen, dass bei der FFT-Methode die Länge des Streams die entsprechende Länge für die FFT hat. Der verwendete FFT-Core von Altera benutzt das Avalon-ST Protokoll zur Kommunikation mit anderen Blöcken. Dazu muss der Puffer neben dem valid-Signal noch ein start-of-packet-Signal und end-of-packet-Signal ausgegeben. Das start-of-packet-Signal wird high gesetzt, wenn das erst Element des Streams valid ist und das end-of-packet-Signal wird high gesetzt, wenn das letzte Element des Streams valid ist. Ansonsten sind beide Signale low.

5.3 Filter

Zur Auswertung des Beugungsbildes mit der Standard-Methode wird das Bildsignal des CCDs zuerst mit einem Funktionsblock gefiltert. Die Filterfunktion des Programms von KJM kann das Signal mit unterschiedlichen Filterparametern filtern. Bei der Hardwarebeschreibung der Filterfunktion muss dieser Parameter während das Reset-Signal low ist (also ein Reset stattfindet) gesetzt werden und danach konstant anliegen. Das Zurücksetzen des Funktionsblocks und das Setzen der Parameter wird von der globalen Steuerlogik übernommen.

Im vorgegebenen C Programmcode wird das CCD-Bildsignal der Filterfunktion als Array übergeben. Die Filterfunktion iteriert dann über das Array und filtert dabei das Signal. Die gefilterten Daten werden dann in ein anderes Array geschrieben. Dadurch, dass die Filterfunktion über das Array mit dem Beugungsbild iteriert, lässt sich diese Funktion gut in Hardware umsetzen, indem die Daten des CCDs Pixel für Pixel als Datenstrom entgegengenommen werden. Dies entspricht dem Iterieren über das Bildsignal mit einer for-Schleife im C Programmcode. Das Problem dabei ist, dass in der for-Schleife nicht nur auf das aktuelle Element i zugegriffen wird, sondern auch auf Elemente die im Array vor dem aktuellen Element i liegen. Dabei wird aber nur auf Elemente zugegriffen, die einen konstanten Abstand p zum aktuellen Element i haben. Dieser Abstand ist abhängig vom Filterparameter, der für das Filtern einer CCD-Zeile konstant ist. Deswegen kann man gut eine FIFO verwenden, um die entsprechende Anzahl an Elementen zu puffern. Wie in Abbildung 5.2 gezeigt, kann man damit auf das Element an der aktuellen Stelle i und auf das Element $i - p$ zugreifen. Zum Implementieren der FIFO wurde der *scfifo* Block von Altera verwendet. Diese FIFO wird dann beim Synthetisieren in einen Speicherblock des FPGAs implementiert. Da die FIFO allerdings nur eine begrenzte Anzahl an Elementen aufnehmen kann, muss vor dem Synthetisieren der Hardwarebeschreibung auf das FPGA die Größe der FIFO festgelegt werden. Somit muss man vor dem Synthetisieren auch einen maximal möglichen Filterparameter festlegen.

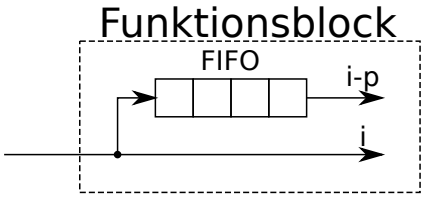

Abbildung 5.2: Puffer in Filterfunktionsblock.

Der Filterfunktionsblock benötig auch noch eine Steuerlogik, die abhängig davon ob am Eingang gültige Daten anliegen (valid-Signal high) und der Anzahl an Elementen in der FIFO die entsprechenden Operation der Filterfunktion ausführt. Auf

die genauen Operationen der Filterfunktion wird aber aus Geheimhaltungsgründen nicht eingegangen.

Bei der Anwendung der Filterfunktion verkürzt sich die Länge der ausgegeben Daten im Vergleich zu der Länge des Datenstroms am Eingang. Dabei fallen am Anfang und am Ende des Datenstroms die Hälfte des Filterparameters Werte weg. Damit die Länge des Datenstroms für die nachfolgenden Funktionsblöcke konstant bleibt, wird der ausgegebene Datenstrom am Anfang und Ende mit Nullen aufgefüllt. Das valid-Signal bleibt dabei immer für 5000 Werte gesetzt und hat somit die gleiche Länge wie am Eingang.

Die Daten der Filterfunktion müssen abschließend noch durch einen festgelegten Wert dividiert werden. Dazu wurde ein *lpm_divide* Funktionsblock von Altera verwendet. Dieser Funktionsblock wendet eine Ganzzahldivision an. Dabei entsteht aber kein Genauigkeitsverlust im Vergleich zur Referenzimplementierung von KJM, da diese hier ebenfalls eine Ganzzahldivision verwendet. Wenn der Dividierer auf dem kritischen Pfad des Designs liegt, lässt sich durch Hinzufügen von Pipeline-Stufen die maximal mögliche Taktfrequenz der Implementierung erhöhen. Dabei wird auch gleichzeitig der Grad an Parallelität erhöht, da der Dividierer während der Verarbeitung eines Beugungsbildes in jedem Takt einen neuen Eingabewert bekommt.

5.4 Differenzierer

Die Implementierung des Differenzierers ist ähnlich zu der Implementierung der Filterfunktion (Abschnitt 5.3). Der Differenzierer bekommt als Eingang die gefilterten Daten des Filterfunktionsblocks und einen Parameter fürs Differenzieren. Der Parameter wird, wie schon bei dem Filterfunktionsblock, beim Zurücksetzen des Funktionsblocks gesetzt. Wie in Abbildung 5.2 schon für den Filterfunktionsblock gezeigt, verwendet auch der Differenzierer eine FIFO, um auf die Elemente i und $i - p$ des Datenstroms zuzugreifen. Der Abstand p ist abhängig vom Parameter des Differenzierers.

Der Differenzierer gibt dann sowohl die Ableitung des eingegebenen Datenstroms, als auch den ursprünglichen Datenstrom aus. Des Weiteren wird auch hier wieder ein valid-Signal ausgegeben, das angibt, ob die Daten am Ausgang gültig sind. Auch hier ist die Länge des Datenstroms am Ausgang gleich der Länge des Datenstroms am Eingang, da der Ausgangsdatenstrom am Anfang und Ende mit Nullen aufgefüllt wird. Das valid-Signal bleibt für 5000 Takte gesetzt.

5.5 Signal-Verteiler

Nachdem man die Ableitung des gefilterten Beugungsbildes bestimmt hat, werden mithilfe der Ableitung die Minima des Beugungsbildes bestimmt. Dies wird im Programmcode von KJM getrennt für die linke und die rechte Seite des Beugungsbildes erledigt. Das Suchen nach Minima wird dabei im Beugungsbild von innen nach außen durchgeführt. Bei der rechten Seite des Beugungsbildes ist dies kein Problem, da der Datenstrom schon die richtige Reihenfolge hat. Bei der linken Seite des Beugungsbildes muss jedoch die Richtung des Datenstroms umgedreht werden. Dazu wird ein Puffer benötigt, in den die linke Seite des Beugungsbildes eingelesen wird und dann in der anderen Reihenfolge ausgegeben wird. Eine weitere Möglichkeit besteht darin, das Beugungsbild für die beiden Seiten getrennt von innen nach außen aus dem Doppelpuffer auszugeben. Dazu benötigt man die Filterfunktion und den Differenzierer zweimal, nämlich für die linke und die rechte Seite. Da diese beiden Funktionsblöcke nicht viele Hardwareressourcen benötigen, spielt dies für den Verbrach an Hardwareressourcen keine große Rolle. Da in dem Programmcode von KJM das Beugungsbild aber am Stück gefiltert wird, wurde dies aus Gründen der bitgenauen Vergleichbarkeit zur Referenzimplementierung nicht gemacht.

5.6 Minima suchen

Dieser Funktionsblock dient dazu, aus der Ableitung des Beugungsbildes die Nullstellen zu finden und damit die Positionen der Minima zu bestimmen. Auch beim Suchen der Minima kommt wieder eine FIFO zum Einsatz, um auf Elemente vor und nach der aktuellen Position i zuzugreifen. Im Gegensatz zu der Filterfunktion und dem Differenzierer wird aber sowohl auf die Elemente $i-p$ als auch $i+p$ zugegriffen. Deswegen verwendet man hierfür zwei FIFOs. Der eingegebene Datenstrom wird in die erste FIFO geleitet. Wenn diese FIFO die entsprechende Anzahl an Elementen enthält, gibt die FIFO die Elemente in eine zweite FIFO aus. Dieser Vorgang ist in Abbildung 5.3 dargestellt.

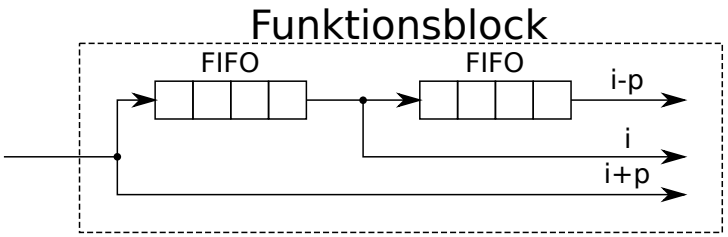

Abbildung 5.3: Puffer in Funktionsblock "Minima suchen".

Die Ausgabe dieses Funktionsblockes ist ein Signal, das anzeigt, ob im aktuellen Takt ein Minimum gefunden wurde oder nicht. Das Positionssignal zeigt an, an welcher Position das Minimum gefunden wurde. Die Position wird bestimmt, indem in jedem Takt, in dem ein gültiges Eingangssignal anliegt, ein Counter hochgezählt wird. Außerdem wird noch die eingegebene Ableitung zusammen mit dem entsprechenden valid-Signal an den Ausgang weitergeleitet. Die Ableitung wird ausgegeben, weil es einen weiteren Funktionsblock gibt, der die Positionen der Minima genauer annähert (siehe Abschnitt 5.7). Wenn der Funktionsblock die Suche nach Minima beendet hat, wird das finished-Signal high gesetzt.

Die Suche nach Minima wird zurzeit abgebrochen, wenn das erste Minimum gefunden wurde. Der Programmcode von KJM kann allerdings auch mehrere Minima bestimmen. Dies ist aber momentan noch nicht implementiert.

5.7 Ausgleichsgerade

Der Programmcode von KJM nähert die Nullstellen der gefundenen Minima mit einer Ausgleichsgeraden an um eine präzisierte Nullstelle zu erhalten. Dazu wird für die zuvor gefundene Nullstelle (siehe Abschnitt 5.6) in diesem Funktionsblock eine Ausgleichsgerade berechnet. Für die Berechnung der Ausgleichsgeraden werden die Punkte links und rechts der Nullstelle der Ableitung verwendet.

Damit man die Ausgleichsgerade direkt aus dem Datenstrom bestimmen kann, müssen die Werte der Ableitung wieder in einer FIFO zwischengespeichert werden. Dies ist nötig, weil man, wenn ein Minimum gefunden wurde, im Datenstrom nur noch auf die Werte rechts der Nullstelle zugreifen kann. Um auch auf die Werte links der Nullstelle zugreifen zu können, muss man den Datenstrom der Ableitung in der FIFO zwischenspeichern.

Zur Berechnung der Ausgleichsgerade müssen auch Werte dividiert werden. Um bei dieser Division gewährleisten zu können, dass die Berechnung genau genug ist, muss man an dieser Stelle Gleitkommazahlen verwenden. Um die gleiche Genauigkeit wie die Referenzimplementierung zu erreichen, wird die Berechnung mit doppelter Genauigkeit durchgeführt (Exponent 11 bit; Mantisse 52 bit). Die Division mit Gleitkommazahlen ist aber auf einem FPGA recht aufwendig und benötigt auch viele Ressourcen des FPGAs. So benötigt die Division mit doppelter Genauigkeit 1313 Logik-Zellen, 916 Register, drei 9 kbit große Speicherblöcke und 44 9 x 9 bit Multiplizierer. Da man den Funktionsblock zur Berechnung der Ausgleichsgerade zweimal benötigt, wenn man die linke und die rechte Seite des Beugungsbildes parallel auswertet, ist es sinnvoll die Division in den Funktionsblock zur Drahtdurchmesserbestimmung auszulagern. Dann kann man nämlich die Division der linken und rechten Seite nacheinander mit demselben Funktionsblock berechnen. Dies hat auch den Vorteil, dass man dann auch die Berechnung des Draht-

durchmessers in Gleitkommaarithmetik durchführen kann.

5.8 FFT

Zur Drahtdurchmesserbestimmung mittels Fourier-Transformation muss das Beugungssignal zunächst in den Frequenzbereich transformiert werden. Dazu wird von KJM die Schnelle Fourier-Transformation (FFT) angewendet. Die FFT lässt sich auf FPGAs gut parallelisieren, weil die Berechnungen in einer Baumstruktur durchgeführt werden. Zur Implementierung der FFT auf dem FPGA wird der FFT IP Core von Altera verwendet. Im Gegensatz zu den anderen Funktionsblöcken von Altera benötigt man für den FFT IP Core eine Lizenz, um diesen einsetzen zu können. In der kostenpflichtigen Subscription Edition der Quartus II Synthesesoftware ist die Lizenz für den FFT IP Core bereits enthalten. Mit der freien Quartus II Web Edition ist die Simulation diese Funktionsblocks aber auch ohne Lizenz möglich. Für den Einsatz im Endprodukt muss dann aber eine separate Lizenz erworben werden. Als Alternative könnte man auch einen lizenzfreien FFT-Block verwenden. Die frei verfügbaren FFT-Blöcke lassen sich jedoch meist nicht so gut parametrisieren und sind auch nicht auf ein spezielles FPGA optimiert. Dadurch lassen sie sich aber besser auf ein anderes FPGA portieren. Der FFT IP Core von Altera bietet verschiedene Parameter an, um den FFT-Block nach den entsprechenden Bedürfnissen anzupassen. So lassen sich damit FFT-Blöcke generieren, die eine feste Länge an Daten verarbeiten können oder auch FFT-Blöcke deren Datenlänge zur Laufzeit angepasst werden kann. Da die Länge des CCD-Zeilensensors konstant ist, benötigt man die Möglichkeit zur Anpassung der Datenlänge zur Laufzeit nicht.

Der FFT-Block mit fester Datenlänge verwendet eine Radix-4-FFT. Wenn die Datenlänge der FFT eine Zweierpotenz mit ungeradem Exponenten ist, wird in der letzten Stufe eine Radix-2-FFT angewendet. Eine weitere Möglichkeit zur Anpassung des FFT IP Core besteht darin, die Anzahl der parallelen Berechnungen einer FFT-Engine zu wählen. Bei der Quad-Output-FFT-Engine werden alle Berechnungen der FFT-Engine parallel ausgeführt, während bei der Single-Output-FFT-Engine nur eine Berechnung ausgeführt wird. Deswegen benötigt die Single-Output-FFT-Engine nur einen komplexen Multiplizierer, während die Quad-Output-FFT-Engine drei komplexe Multiplizierer benötigt. Dafür benötigt die Quad-Output-FFT-Engine natürlich auch weniger Taktzyklen für die Berechnung. Des Weiteren lassen sich entweder eine, zwei oder vier parallele FFT-Engines auswählen.

Die Zahlendarstellung des FFT IP Core von Altera wird mit einer *Block-floating-point* Darstellung realisiert. Bei dieser Darstellung wird für die Transformation ein gemeinsamer Exponent verwendet. Im Gegensatz zu einer *fixed-point* Berechnung wird dadurch die benötigte Datenbreite reduziert. Gegenüber einer *floating-point* Berechnung wird damit auch die Anzahl der benötigten Hardwareressourcen

reduziert [9].

Da der FFT-Block viele Ressourcen des FPGAs benötigt, wird die linke und rechte Seite des Beugungsbildes nicht parallel transformiert, sondern nacheinander. Dadurch benötigt man nur noch einen FFT-Block und keine zwei FFT-Blöcke mehr. Für die Implementierung wurde der FFT IP Core so konfiguriert, dass er zwei Quad-Output-FFT-Engines verwendet. Die Transformationslänge beträgt 2018 Werte und die Datenbreite der Ein- und Ausgabekanäle beträgt 12 bit. Dies stellt für die Anwendung einen guten Kompromiss zwischen den benötigten Hardwareressourcen und der für die Berechnung benötigten Takte dar. Die Berechnung des Drahtdurchmessers darf dabei nur so lange dauern, wie das CCD zum Ausgeben einer Zeile benötigt.

5.9 FFT Auswertung

Zur Bestimmung des Drahtdurchmessers aus der Fourier-Transformation des Beugungsbildes muss das transformierte Beugungsbild ausgewertet werden. Dieser Funktionsblock wertet das transformierten Beugungsbildes aus, indem das Maximum k im Frequenzbereich (siehe Abschnitt 2.4.2) gesucht wird. Dazu wird der von dem FFT-Block (siehe Abschnitt 5.8) ausgegebene Datenstrom schrittweise analysiert wird. Hierfür speichert der Funktionsblock das Maximum der bisher analysierten Daten in einem Register und vergleicht, ob die neuen Daten größer oder gleich dem bisherigen Maximalwert sind. Wenn dies der Fall ist, wird der Maximalwert aktualisiert. Die Position des Maximums wird noch genauer bestimmt, indem auch die Werte links und rechts davon miteinbezogen werden. Die Zwischenergebnisse werden an den Funktionsblock zum Bestimmen des Drahtdurchmessers weitergeleitet.

5.10 Drahtdurchmesser bestimmen

Der Funktionsblock zum Bestimmen des Drahtdurchmesser kann entweder aus den Ausgabewerten der Standard-Methode den Drahtdurchmesser bestimmen, oder aus den Ergebnissen der FFT-Methode. Wenn der Drahtdurchmesser nach der Standard-Methode berechnet wird, benötigt der Funktionsblock als Eingabewerte die Positionen der gefundenen Minima und die Zwischenergebnisse der Ausgleichsgeraden. Diese Daten bekommt der Funktionsblock sowohl von dem Teil der die linke Seite des Beugungsbildes ausgewertet hat, als auch von dem Teil der die rechte Seite des Beugungsbildes ausgewertet hat. Zurzeit verwendet der Funktionsblock nur die beiden inneren Minima zur Berechnung des Drahtdurchmessers. Der Programmcode von KJM kann auch noch mehrere Minima in die Berechnung einbeziehen, aber dies ist aktuell in der Hardwarebeschreibung noch nicht implementiert. Auch bei der Drahtdurchmesserberechnung mit der FFT-Methode werden die Ergebnisse

der Auswertung der linken und rechten Seite des Beugungsbildes eingegeben. Da die FFT-Methode die linke und rechte Beugungsbildseite aber nicht parallel berechnet, werden diese Daten nacheinander eingegeben.

Auch bei der folgenden Beschreibung des Funktionsblocks zur Drahtdurchmesserbestimmung wird aus Gründen der Geheimhaltung nicht auf die genaue Berechnung eingegangen. Stattdessen soll hier auf die Problemstellungen, die sich bei der Implementierung dieses Funktionsblockes ergaben, eingegangen werden.

Um die Berechnung des Drahtdurchmessers mit der nötigen Genauigkeit durchführen zu können, muss man die Berechnungen in Gleitkommaarithmetik durchführen. In der Referenzimplementierung werden dafür Gleitkommazahlen mit doppelter Genauigkeit verwendet. Da Gleitkommaberechnungen auf einem FPGA aber vergleichsweise viele Ressourcen benötigen, ist es sinnvoll die nötigen Berechnungsblöcke nur einmal zu erzeugen und dann für mehrere Berechnungen zu verwenden. Zur Berechnung des Drahtdurchmessers wurde ein Funktionsblock zum Dividieren und ein Funktionsblock zum Addieren bzw. Subtrahieren verwendet. Auch hier wurden die entsprechenden Funktionsblöcke *altfp_div* und *altfp_add_sub* von Altera verwendet. Zum Konvertieren der Zahlen von *integer* nach *floating-point* bzw. von *floating-point* nach *fixed-point* wurde jeweils ein *altfp_convert* Block verwendet. Um die Blöcke leicht wiederverwenden zu können (operator folding), werden an den Eingängen Register erzeugt, in die man die Eingabewerte schreibt.

Wenn die Operatoren aber jeweils nur einmal vorhanden sind, müssen alle Berechnungen, die denselben Funktionsblock verwenden, nacheinander ausgeführt werden. An dieser Stelle ist dies aber nicht gravierend, da man nur noch wenige Berechnungen durchführen muss und diese Berechnungen auch nur einmal pro CCD-Zeile ausgeführt werden müssen.

Zur sequenziellen Ausführung der Berechnungen wird ein endlicher Automat (FSM) benötigt. Die verwendeten Berechnungsblöcke besitzen eine unterschiedliche Anzahl an Pipeline-Stufen, sodass die Ergebnisse nach unterschiedlichen Latenzzeiten vorliegen. Der *altfp_convert* Funktionsblock hat eine Latenz von 6 Takten. Die Latenz des *altfp_div* und *altfp_add_sub* Funktionsblocks wurde so eingestellt, dass sie 10 bzw. 7 Takte beträgt. Somit muss die FSM nach der entsprechenden Anzahl an Takten die Ergebnisse von einem Berechnungsblock an die Eingänge eines anderen Berechnungsblock weiterleiten. Dieser Vorgang ist in Codeausschnitt 5.4 dargestellt.

Bei der Berechnung des Drahtdurchmessers müssen häufig die gleichen Berechnungsschritte nacheinander durchgeführt werden. Bei der Implementierung erzeugt man deswegen für jeden Berechnungsabschnitt einen Modul, das man wiederverwenden kann. In dieses Modul kann man dann über einen Multiplexer entweder die Zwischenergebnisse aus der Analyse der linken oder rechten Seite des Beugungsbildes eingeben.

Der Codeausschnitt 5.4 stellt exemplarisch die Division zweier ganzzahliger Werte

dar. Des Weiteren zeigt der Codeausschnitt auch, wie man einen Modul sowohl für Berechnungen der linken oder rechten Seite des Beugungsbildes nutzen kann.

In Zeile 1 und 3 wird abhängig davon, welche Seite gerade ausgewertet wird, die Daten der linken oder rechten Seite an das Modul weitergeleitet. Wenn die Steuerlogik das Signal *s_div_start* 1 setzt, werden die Berechnungen gestartet (Zeile 9). Gleichzeitig muss die Steuerlogik den Startzustand *STATE_STEP_0* setzen. In Zustand *STATE_STEP_0* wird der ganzzahlige Wert *s_div_dataa* an den Funktionsblock zum Konvertieren in eine Gleitkommazahl angelegt. Im darauffolgenden Zustand wird dies mit *s_div_datab* gemacht. Die Konvertierung dauert 6 Takte. Deswegen werden in Zustand *STATE_STEP_7* und *STATE_STEP_8* die konvertierten Werte an den Dividierer weitergeleitet. Da Ergebnis der Division liegt in Zustand *STATE_STEP_23* an und kann in das Register *s_div_result* geschrieben werden. Dabei wird über das Signal *s_div_finished* angezeigt, dass die Berechnung abgeschlossen ist.

Das Ergebnis der Drahtdurchmesserbestimmung wird dann als Gleitkommazahl mit doppelter Genauigkeit ausgegeben. Wenn man den Drahtdurchmesser über eine Benutzerschnittstelle (z. B. ein Display) ausgeben will, könnte man den Drahtdurchmesser zuvor auch in die *fixed-point* Darstellung konvertieren. Dadurch kann die Benutzerschnittstelle vereinfacht werden, da diese dann keine Gleitkommazahlen verarbeiten können muss. Über ein finished-Signal wird der globalen Steuerlogik mitgeteilt, dass die Berechnung abgeschlossen ist.

5.11 Globale Steuerlogik

Zur Steuerung der Berechnung des Drahtdurchmessers wird eine globale Steuerlogik verwendet. Mit dieser Steuerlogik wird der Berechnungsvorgang gestartet, sobald der Einlesevorgang der Daten vom CCD abgeschlossen ist. Zu Beginn sind noch keine Daten eingelesen worden und die Datenanalyse darf erst gestartet werden wenn eine CCD-Zeile eingelesen wurde. Beim Start der Datenanalyse müssen auch die dazu nötigten Parameter gesetzt werden. Des Weiteren müssen auch die einzelnen Funktionsblöcke vor der Analyse einer neuen Zeile zurückgesetzt werden. Je nachdem welche Analyse-Methode verwendet werden soll werden entweder die Funktionsblöcke der Standard-Methode oder der FFT-Methode aktiviert. Welche Methode ausgeführt werden soll, lässt sich zur Laufzeit von außen durch ein Eingangssignal bestimmen.

```
1   s_div_dataa <= s_dataa_left when s_side = '0'
2                  else s_dataa_right;
3   s_div_datab <= s_datab_left when s_side = '0'
4                  else s_datab_right;
5
6   p_div : process(clk)
7   begin
8     if rising_edge(clk) then
9       if s_div_start = '1' then
10        case state_step is
11          when STATE_STEP_0 =>
12            s_int2float_in <= std_logic_vector(s_div_dataa);
13            state_step     <= state_step + 1;
14          when STATE_STEP_1 =>
15            s_int2float_in <= std_logic_vector(s_div_datab);
16            state_step     <= state_step + 1;
17          when STATE_STEP_7 =>
18            s_floatdiv_dataa <= s_int2float_out;
19            state_step       <= state_step + 1;
20          when STATE_STEP_8 =>
21            s_floatdiv_datab <= s_int2float_out;
22            state_step       <= state_step + 1;
23          when STATE_STEP_23 =>
24            s_div_result   <= s_floatdiv_result;
25            s_div_finished <= '1';
26          when others =>
27            state_step <= state_step + 1;
28        end case;
29      end if;
30    end if;
31  end process;
```

Abbildung 5.4: Codeausschnitt zur Division zweier ganzzahliger Werte mit Gleitkommaarithmetik.

6 Evaluation

Die Evaluation der Hardwareimplementierung wird anhand der folgenden Kriterien durchgeführt. Als erstes wird die funktionale Verifikation durchgeführt, indem die Genauigkeit der Ergebnisse der Hardwareimplementierung mit der Referenzimplementierung verglichen wird. Danach wird die Performanz der Hardwareimplementierung evaluiert. Dazu wird sowohl die Anzahl der Takte als auch die maximal mögliche Taktfrequenz der Implementierung ausgewertet. Abschließend wird der Ressourcenverbrauch der Implementierung bestimmt. Das Ziel-FPGA ist das Altera Cyclone III *EP3C25U256C8N*. Dieses FPGA hat 24 624 Logik-Elemente, 66 9 kbit Speicherblöcke und 132 eingebettete 9 x 9 bit Multiplizierer.

Zum Verifizieren der Implementierung wurden mehrere Testbenches verwendet. Es wurde eine Testbench zum Testen der Standard-Methode erstellt und eine Testbench zum Testen der Standard- und FFT-Methode. Beide Testbenches laden die Simulationsdaten aus einer Testdatei. Die Beugungsbilder der Testdateien bestehen aus 5000 Werten und wurden von KJM bereitgestellt. Diese Beugungsbilder sind in Abbildung 6.1 dargestellt. Die Testbench die nur die Standard-Methode testet, lädt die Simulationsdaten direkt in den Funktionsblock zum Filtern der Daten (Abschnitt 5.3). Außerdem wird mit dieser Testbench nicht der Funktionsblock zum Berechnen des Drahtdurchmessers (Abschnitt 5.10) getestet. Die andere Testbench, die sowohl die Standard- als auch die FFT-Methode ausführen kann, lädt die Daten einer Zeile erst in den Doppelpuffer (Abschnitt 5.2) und startet anschließend die Datenanalyse. Je nachdem welche Methode in der Testbench ausgewählt wurde, wird der Drahtdurchmesser entweder mit der Standard-Methode oder mit der FFT-Methode bestimmt. Des Weiteren wurde auch die vorgegebene Testbench verwendet, um die Datenanalyse zusammen mit den Simulationsmodellen des CCDs und des ADCs zu testen.

6.1 Genauigkeit

Ein Kriterium für die Implementierung ist die Genauigkeit des bestimmten Drahtdurchmessers im Vergleich zur Referenzimplementierung.

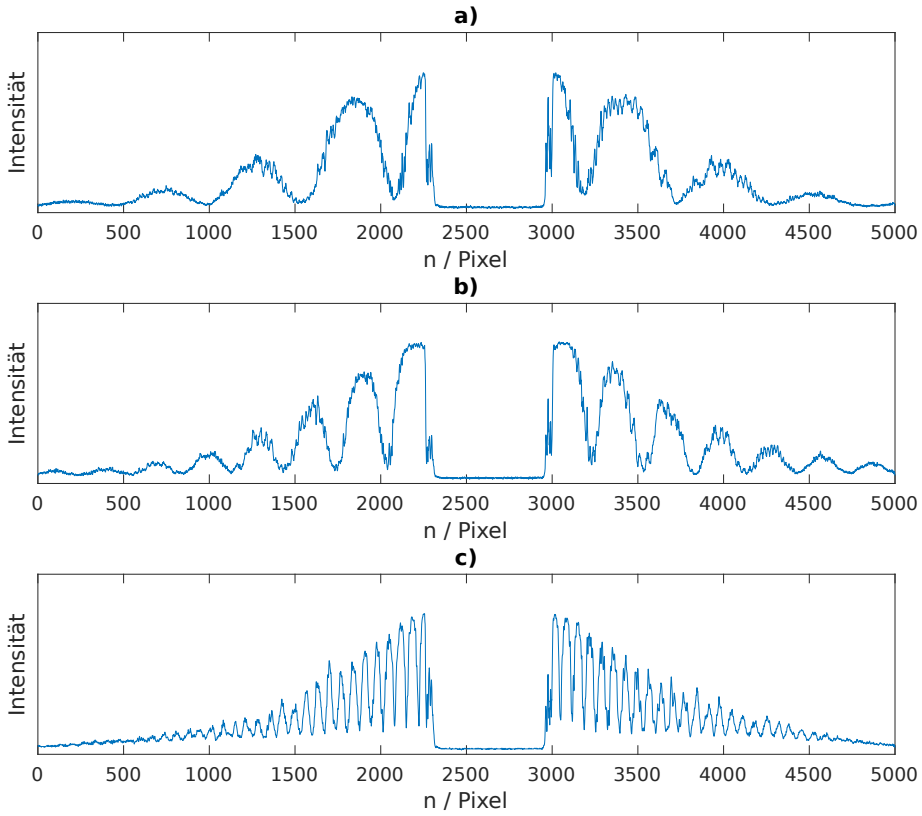

Abbildung 6.1: Beugungsbilder der Testdateien. (von KJM)

6.1.1 Standard-Methode

Die Standard-Methode analysiert das Beugungsbild mit Ganzzahlarithmetik. Dabei entsteht allerdings kein Genauigkeitsverlust, da dies in der Referenzimplementierung ebenfalls mit Ganzzahlarithmetik durchgeführt wird. Die endgültige Berechnung des Drahtdurchmessers aus den zuvor gewonnenen Daten wird in dem Programmcode von KJM mit Gleitkommazahlen mit doppelter Genauigkeit durchgeführt. Auch in der Hardwareimplementierung wird die Berechnung des Drahtdurchmessers mit Gleitkommaarithmetik mit doppelter Genauigkeit durchgeführt. Der leichte Genauigkeitsverlust der dabei entsteht, wird dadurch verursacht, dass die Reihenfolge der Rechenoperationen geändert wurde, um Berechnungen mit zuvor bekannten Konstanten vor der Synthese durchzuführen. So ergab die Auswertung der Testdatei (a) mit der Standard-Methode einen Durchmesser von 13,883 58 µm, während die Referenzimplementierung einen Durchmesser von 13,882 99 µm ermittelte.

6.1.2 FFT-Methode

Der FFT IP Core von Altera, der bei der Auswertung nach der FFT-Methode zum Einsatz kommt, führt die Berechnungen nicht wie bei der Softwareimplementierung mit Gleitkommazahlen mit doppelter Genauigkeit aus, sondern verwendet eine *Block-floating-point*-Arithmetik. Dies führt bei der Auswertung des transformierten Beugungsbildes zu einem Genauigkeitsverlust. So ergab die Simulation der Implementierung mit der Testdatei (b) einen Durchmesser von 27,124 05 µm, während die Referenzimplementierung einen Durchmesser von 27,089 57 µm ermittelte. Für die Testdatei (c) lag das Ergebnis der Hardwareimplementierung bei 111,534 56 µm und das der Referenzimplementierung bei 111,547 61 µm.

6.2 Berechnungsdauer

Zur Analyse der Berechnungsdauer muss zum einen die Anzahl der benötigten Takte und zum anderen die maximal mögliche Taktfrequenz der Implementierung bestimmt werden. Um die Anzahl der benötigten Takte zu bestimmen, wurde die oben dargestellte Testbench in der *ModelSim-Altera Starter Edition* (Version 10.1d) Simulationsumgebung ausgeführt. Aus dem Signalverlauf der Simulation lässt sich dann die Anzahl der Takte bestimmen, die die Implementierung für die Drahtdurchmesserbestimmung benötigt.

Die maximale Taktfrequenz f_{max} wurde mit dem *TimeQuest Timing Analyzer* (Version 13.1.4) bestimmt. Alle Funktionsblöcke der Implementierung werden mit der gleichen Taktfrequenz betrieben. Diese Taktfrequenz kann man beim Betreiben des FPGAs mit 1,2 V bei einer Temperatur von 85 °C bis zu 63,02 MHz eingestellt werden. Der kritische Pfad liegt bei dem Funktionsblock zum Berechnen der Ausgleichsgerade (Abschnitt 5.7), da dort eine Multiplikation und Addition in einem Takt durchgeführt wird.

Da die Implementierung zwei verschiedene Methoden zum Bestimmen des Drahtdurchmessers anbietet, muss man die benötigte Taktanzahl auch für beide Methoden getrennt bestimmen.

6.2.1 Standard-Methode

Die Standard-Methode bestimmt den Drahtdurchmesser aus den Positionen der Minima des Beugungsbildes. Aktuell verwendet die Standard-Methode nur die beiden inneren Minima für die Berechnung des Drahtdurchmessers. Wenn die beiden Minima gefunden wurden, wird Datenanalyse abgebrochen und die Drahtdurchmesserberechnung gestartet. Deswegen hängt die Anzahl der benötigten Takte von der Position der Minima ab. Im schlechtesten Fall benötigt die Standard-Methode 5305 Takte.

Für die Testdatei (a) von KJM werden 3475 Takte benötigt. Somit lässt sich für diese Testdatei der Datenanalyse-Algorithmus mit einer Abtastrate von 18,135 kHz auführen. Für den Worst Case lässt sich eine Abtastrate von 11,879 kHz erreichen.

6.2.2 FFT-Methode

Die Taktzahl der FFT-Methode ist konstant. Die Implementierung benötigt immer 9032 Takte, um mit der FFT-Methode aus dem Beugungsbild den Drahtdurchmesser zu bestimmen. Demnach kann die Auswertung der CCD-Daten (ohne Einlesen der CCD-Zeile) mit einer Abtastrate von 6,977 kHz durchgeführt werden.

6.3 Ressourcenverbrauch

Der Ressourcenverbrauch wurde mit dem Synthesetool *Quartus II Web Edition* (Version 13.1.4) bestimmt. Die Implementierung verwendet zusammen mit dem CCD- und ADC-Controller 16 016 Logik-Elemente von den 24 624 Logik-Elementen des Altera Cyclone III *EP3C25*. Davon werden 13 656 kombinatorische Logik-Funktionen und 9715 Register benötigt. Von den 66 9 kbit großen Speicherblöcken werden 55 verwendet. Die 132 9-bit Multiplizierer werden vollständig ausgelastet.

7 Diskussion

Im vorherigen Kapitel 6 wurde gezeigt, dass sich mit der Hardwareimplementierung der Standard-Methode eine Abtastrate von 11,879 kHz erreichen lässt und mit der FFT-Methode in Abtastrate von 6,977 kHz. Wenn in dem Programmcode von KJM nur die Standard-Methode ausgeführt werden soll, wird diese allerdings zweimal pro CCD-Zeile mit unterschiedlichen Parametern ausgeführt. Dies ist in der Hardwareimplementierung noch nicht realisiert. Dadurch würde sich die Abtastrate auf 5,939 kHz halbieren.

Der Programmcode von KJM bietet auch die Möglichkeit, die Standard-Methode mit der FFT-Methode zu kombinieren. Bei dieser Spezial-Methode wird zuerst die FFT-Methode ausgeführt und auf Basis der Ergebnisse der FFT-Methode wird dann einmal die Standard-Methode ausgeführt. Dies ist in der Hardwareimplementierung noch nicht realisiert. Wenn man dies noch implementieren würde, müsste man die globale Steuerlogik so anpassen, dass sie zuerst die Standard-Methode und danach die FFT-Methode ausführt. Somit ließe sich mit dieser Spezial-Methode eine Abtastrate von 4,395 kHz erreichen.

Die Hardwareimplementierung der Standard-Methode verwendet zur Bestimmung des Drahtdurchmessers nur die beiden inneren Minima des Beugungsbildes. Wenn man dies noch implementieren würde, würde sich die Berechnungsdauer etwas verlängern. Die Berechnungsdauer der Auswertung des Beugungsbildes würde sich dabei für den Worst Case nicht verlängern, aber die Berechnung des Drahtdurchmessers würde sich verlängern, da dabei mehr Kennwerte der vorherigen Datenanalyse für die Berechnung verwendet werden müssen.

Die Taktfrequenz der Hardwareimplementierung ließe sich durch Optimieren der Implementierung noch erhöhen. Dadurch ließe sich dann auch ein Sensor verwenden, der eine höhere Abtastrate unterstützt oder mehr Pixel besitzt. So könnte man beispielsweise den *TCD1706DG* CCD-Zeilensensor von Toshiba verwenden, der 7400 Pixel hat und dadurch, dass er vier Ausgabekanäle besitzt, eine Abtastrate von bis zu 13 kHz ermöglicht. Die Taktfrequenz kann dadurch erhöht werden, dass man bei den Berechnungen, die auf dem kritischen Pfad liegen, noch zusätzliche Pipeline-Stufen einfügt. Die Möglichkeit zur Optimierung der Taktfrequenz der Implementierung wird allerdings durch die maximale Taktfrequenz des *floating-point* Dividierers beschränkt. Ab 61 Pipeline-Stufen benötigt der Dividierer so viele Hard-

wareressourcen, dass er sich nicht mehr für das aktuell von KJM eingesetzte FPGA synthetisieren ließe. Um die komplette Implementierung mit einer höheren Taktfrequenz betrieben zu können, müssen allerdings auch noch bei anderen Berechnungen (besonders bei der Berechnung der Ausgleichsgeraden) weitere Pipeline-Stufen eingefügt werden.

Im Vergleich zu der ursprünglichen Implementierung auf einem Softcore Prozessor konnte die Abtastrate deutlich erhöht werden. Somit ist der Datenanalyse-Algorithmus jetzt schneller, als der aktuell verwendete CCD-Zeilensensor für das Auslesen einer Zeile benötigt. Auch im Vergleich zu den Konkurrenzprodukten hat man einen Vorteil, da die meisten vergleichbaren Produkte deutlich langsamer sind.

Die Analyse des Ressourcenverbrauchs ergab, dass sich die Hardwareimplementierung in das vorgegebene Altera Cyclone III FPGA (*EP3C25*) synthetisieren lässt. Eine kleinere Version des FPGAs lässt sich jedoch nicht auswählen, da die kleinere Version nur 15 408 Logik-Elemente besitzt. Die Hardwareimplementierung benötigt aber 16 083 Logik-Elemente. Außerdem besitzt die kleinere Version des FPGAs auch weniger eingebettete Multiplizierer.

8 Zusammenfassung und Ausblick

Das Ziel dieser Bachelorarbeit war die Beschleunigung der Drahtdurchmesserbestimmung des Messgerätes von KJM. Dabei sollte der Datenanalyse-Algorithmus beschleunigt werden, damit er mit der gleichen Abtastrate wie der CCD-Zeilensensor betrieben werden kann. Dieses Ziel wurde durch die Hardwareimplementierung des Datenanalyse-Algorithmus erreicht.

Die Hardwareimplementierung bietet dabei aber noch nicht die gleiche Funktionalität wie die Softwareimplementierung von KJM. So kann die Standard-Methode bisher nur die beiden inneren Minima zur Bestimmung des Drahtdurchmessers verwenden. Außerdem lässt sich die Standard-Methode noch nicht zweimal pro CCD-Zeile mit unterschiedlichen Parametern ausführen. Des Weiteren ist die Spezial-Methode, die die Standard-Methode mit der FFT-Methode kombiniert noch nicht implementiert. Darüber hinaus wird in dem Programmcode von KJM an manchen Stellen eine Bereichsprüfung durchgeführt, um zu überprüfen, ob die Ergebnisse in einem gewissen Bereich liegen und damit gültig sind.

Wenn man CCD-Zeilensensoren verwenden will, die eine höhere Abtastrate als die 3 kHz des aktuellen CCD-Zeilensensors unterstützen oder mehr als 5000 Pixel besitzen, kann man die Hardwareimplementierung noch weiter optimieren. Dazu gibt es wie in Kapitel 7 gezeigt noch Potential.

Im Vergleich zu den Konkurrenzgeräten lässt sich mit der Hardwareimplementierung eine höhere Abtastrate erreichen. Gleichzeitig verwendet das KJM Messgerät ein Messverfahren, das für die Messung von dünnen Drähten besser geeignet ist, als das Laser-Scanning Messverfahren, das von der Konkurrenz eingesetzt wird.

Durch das Verwenden eines aktuelleren FPGAs kann das Design auch gegen Kopieren und Reverse Engineering geschützt werden. Dazu bietet sich der Max 10 FPGA von Intel an. Wenn man ein FPGA von Altera bzw. Intel einsetzt, hat dies auch den Vorteil, dass man die verwendeten Altera Cores nicht durch entsprechende Cores eines anderen Herstellers ersetzen müsste und sich das Design somit ohne großen Aufwand für das neuere FPGA synthetisieren lässt.

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

FPGA Field-Programmable Gate Array

CCD Charge Coupled Device

ADC Analog-to-Digital-Converter

FFT Schnelle Fourier-Transformation

IFFT Inverse Schnelle Fourier-Transformation

DFT Diskrete Fourier-Transformation

SNR Signal-Rausch-Verhältnis

LED Leuchtdiode

LUT Lookup-Tabelle

AES Advanced Encryption Standard

IP intellectual property

SoC System-on-Chip

rSoC rekonfigurierbares System-on-Chip

DSP Digitaler Signalprozessor

RAM Random-Access-Memory

SRAM Static Random-Access-Memory

CLB Configurable Logic Block

FSM Finite-State Machine

Literatur

[1] *7 Series FPGAs Configurable Logic Block User Guide*. Xilinx. Sep. 2016.

[2] *7 Series FPGAs Configuration User Guide*. Xilinx. Sep. 2016.

[3] *7 Series FPGAs Data Sheet: Overview*. Xilinx. Aug. 2017.

[4] *AccuScan 1000: Präzise, vielseitige Durchmesser-Messungen*. `http://www.betalasermike.com/beta-lasermike-downloads/diam_accuScan1000_de.pdf`; abgerufen am 29.09.2017. Beta LaserMike. Aug. 2002.

[5] *ACCUSCAN 6012: The industry's first four-axis diameter and ovality gauge for products up to 12 mm*. `http://www.betalasermike.com/beta-lasermike-downloads/AccuScan6012.pdf`; abgerufen am 29.09.2017. NDC Technologies. Feb. 2015.

[6] *AccuScan-Serie für qualitäts- und kostenbewusste Hersteller*. `http://www.betalasermike.com/beta-lasermike-downloads/cc_accuScan_de.pdf`; abgerufen am 29.09.2017. Beta LaserMike. Sep. 2013.

[7] Y. Chursin und E. Fedorov. "Methods of resolution enhancement of laser diameter measuring instruments". In: 2014.

[8] *DIAMETER GAUGE - Super Fast InteliSENS DG Series*. `http://www.protonproducts.com/wp-content/uploads/2013/11/DG-k-brochure-20131128.pdf`; abgerufen am 29.09.2017. Proton Products.

[9] *FFT IP Core*. Intel. Jan. 2017.

[10] X. Gao, T. Gao, W. Ding und Z. Gong. "High Precision Contactless Object-diameter measurement using Laser light source". In: 2009.

[11] E. Hecht. *Optik*. München: Oldenbourg, 2014.

[12] Z. Hong, W. Xuan und W. Rui. "High Speed On Line Measurement of Digital Wire Outer Diameter With Laser and CCD Technology". In: 2003.

[13] *HWS.2 Der erste tragbare optische Mikrometer zur Messung von Drähten und Kabel!* http://www.aeroel.it/images/pdf/datasheet/hws2_de.pdf; abgerufen am 29.09.2017. AEROEL. Sep. 2017.

[14] *ILX553B*. Sony.

[15] *Intel Cyclone 10 LP Device Overview*. Intel. Mai 2017.

[16] *InteliSENS BG Series Benchtop Diameter Gauges*. http://www.protonproducts.com/wp-content/uploads/2014/06/Proton_Products_BG_Benchtop_gauges_Eng.pdf; abgerufen am 29.09.2017. Proton Products.

[17] *InteliSENS DG1000 Series*. http://www.protonproducts.com/wp-content/uploads/2013/08/dg-series-datasheet1.pdf; abgerufen am 29.09.2017. Proton Products. Jan. 2009.

[18] *LASER Series 2000 Effiziente Durchmessermessköpfe für Kabelproduktionslinien*. https://sikora.net/wp-content/uploads/LASER_Series_2000_dk.pdf; abgerufen am 29.09.2017. SIKORA. Jan. 2017.

[19] *LASER Series 6000 High-End-Durchmessermessköpfe für Kabelproduktionslinien*. https://sikora.net/wp-content/uploads/2016/04/LASER_Series_6000_DE_DK.pdf; abgerufen am 29.09.2017. SIKORA. Apr. 2016.

[20] *LMD H01B / H02B*. Website. https://www.kjm-ag.eu/deutsch/durchmesserme%C3%9Fger%C3%A4te/lmd-h01b-h02b/; abgerufen am 27.09.2017. KJM GmbH.

[21] W. Lüdge und A. Lüdge. "HIGH-RESOLUTION DISTANCE MEASUREMENT OF LASER-INDUCED DIFFRACTION SIGNALS BY DIGITAL SIGNAL PROCESSING". In: 1993.

[22] *MAX 10 FPGA Configuration User Guide*. Intel. Juli 2017.

[23] *MAX 10 FPGA Device Overview*. Intel. Feb. 2017.

[24] *MSD 25/50/100/200*. http://blog.zumbach.com/de/wp-content/uploads/sites/2/2013/11/MSD_MSD.002.0001.D.pdf; abgerufen am 29.09.2017. Zumbach. Jan. 2013.

[25] *ODAC 13TRIO*. http://www.zumbach.com/pdf/Literature_DE/Catalogs/ODAC/ODAC13TRIO_ODAC.007.0313.DE.pdf; abgerufen am 29.09.2017. Zumbach. März 2016.

[26] *ODAC 14XY*. http://www.zumbach.com/pdf/Literature_DE/Catalogs/ODAC/ODAC14XY_ODAC.007.2014.DE.pdf; abgerufen am 29.09.2017. Zumbach. März 2016.

[27] *ODAC 2J/16J*. http://www.zumbach.com/pdf/Literature_DE/Catalogs/ODAC/ODAC2_16_ODAC.007.0007.DE.pdf; abgerufen am 29.09.2017. Zumbach. Feb. 2016.

[28] *Spartan-6 FPGA Configuration User Guide*. Xilinx. März 2017.

[29] *TCD1209DG*. TOSHIBA. Feb. 2004.

[30] *TCD1706DG*. TOSHIBA. Mai 2012.

[31] *TCD1711DG*. TOSHIBA. Juli 2004.

[32] *TIGER LASER 6010 XY ONLINE DURCHMESSERMESSUNG UND OBERFLÄCHENINSPEKTION BEI DER KABELPRODUKTION*. https://sikora.net/wp-content/uploads/2016/02/TIGER_LASER_6010_XY.pdf; abgerufen am 29.09.2017. SIKORA. Apr. 2014.

[33] *TLAser103 Specifications and Drawings*. http://laserlinc.com/specs/TLAser103_specs.html; abgerufen am 29.09.2017. LaserLinc.

[34] *TLAser203 Specifications and Drawings*. http://laserlinc.com/specs/TLAser203_specs.html; abgerufen am 29.09.2017. LaserLinc.

[35] *Triton312 Specifications and Drawings*. http://laserlinc.com/specs/Triton312_specs.html; abgerufen am 29.09.2017. LaserLinc.

[36] *XACTUM Lasermikrometer für höchstpräzise Durchmessermessung - XLS13XY - XLS35XY für Zweiachsenmessungen*. http://www.aeroel.it/images/pdf/datasheet/xls_xy_de.pdf; abgerufen am 29.09.2017. AEROEL. Nov. 2014.

[37] *XACTUM Lasermikrometer für höchstpräzise Durchmessermessung - XLS40 - XLS80 - XLS150 für Einachsenmessungen*. http://www.aeroel.it/images/pdf/datasheet/xls_x_de.pdf; abgerufen am 29.09.2017. AEROEL. Mai 2013.

[38] Z. Zhang, Y. Su, C. Lu, Y. Hao und X. Zhang. "Denoising of Fraunhofer diffraction signal based on wavelet analysis". In: 2010.