Anschluß eines MCP-Detektors an ein Windows NT System

Diplomarbeit

vorgelegt von

Alexander Würz
aus Albstadt-Onstmettingen

Institut für Astronomie und Astrophysik
Abteilung Astronomie

Eberhard-Karls-Universität zu Tübingen

30. August 1999

Im Gedenken an meinen Vater

Inhaltsverzeichnis

Tabellenverzeichnis

Abbildungsverzeichnis

Kapitel 1

Einführung

Seit ihrer Erfindung 1959 in den Bendix Research Laboratories haben Micro Channel Plates (MCP) in vielen Bildsystemen als Bildverstärker und Elektronenvervielfacher Verwendung gefunden. Sie eignen sich besonders für kleine Strahlungsflüsse und bieten dabei eine hohe Zeitauflösung, theoretisch bis in den Nanosekundenbereich. Die Verwendung zusammen mit entsprechenden Photokathoden erlaubt den Einsatz in fast allen Wellenlängenbereichen.

MCP-Detektoren für den optischen Bereich werden seit einigen Jahren am Institut für Astronomie und Astrophysik der Universität Tübingen, Abteilung Astronomie, erfolgreich entwickelt und eingesetzt. Dies begann 1980 und setzt sich bis heute fort. Die daraus gewonnenen Erkenntnisse flossen auch in erfolgreiche Projekte, wie etwa den Bau und Flug des ORFEUS-Teleskops (‚Orbiting and Retrievable Far and Extreme Ultraviolet Spectrometer') auf dem ASTRO-SPAS-Satelliten ein.

Seit der Inbetriebnahme des ersten Detektors (Barnstedt, 1981) wurden in verschiedenen Diplomarbeiten und Dissertationen der Aufbau, die Anodenarten und die Elektronik der optischen MCP-Detektoren weiter entwickelt und auch im astronomischen Einsatz getestet und angewandt. Für die rechnergesteuerte Bildintegration steht jedoch nach wie vor ein DEC PDP 11-System aus der Mitte der 80er Jahre zur Verfügung, dessen weiterer Betrieb aufgrund des Alters zunehmend schwierig wird.

Die Programme der PDP 11/73 sind als Konsolen-Applikationen (Kommandozeilen-Steuerung) aufgebaut und bieten nicht die heute gewohnte Bedienungsfreundlichkeit mit graphischen Oberflächen und Menüsteuerungen. Im neuen Programm, welches mit seiner gekapselten Struktur auch als Grundstein für die Entwicklung eines ganzen Erfassungssystems mit Datenverarbeitungsfunktionen dienen kann, ist dies realisiert. Durch die Verwendung des modernen Betriebssystems Microsoft Windows NT für das neue Datenerfassungssystem ist die weitere Verwendbarkeit der optischen MCP-Detektoren sichergestellt.

Im folgenden werden nun noch ein paar Daten und Vergleiche zwischen MCP- und CCD-Detektoren angeführt bevor in den Kapiteln ausführlich auf die Theorie und Entwicklung des alten und neuen Systems und die Implementierung eingegangen wird. Abgeschlossen wird diese Arbeit mit Vergleichsmessungen der verschiedenen Systeme, der Analyse und einem Ausblick, wie sich die Zukunft des hier erarbeiteten Systems und der optischen Detektoren im allgemeinen darstellen könnte.

1.1 Vergleich MCP – CCD

In Tab. 1.1 sind die wichtigsten Eigenschaften von MCP- und CCD-Detektoren kurz zusammengefaßt. Detektoren können prinzipiell in zwei Kategorien eingeteilt werden: photoemissiv und photokonduktiv. Bei photoemissiven Detektoren muß das Photon genügend Energie haben, um ein Elektron aus einer Photokathode herauszuschlagen, dies sind je nach verwendetem Material einige Elektronenvolt. In einem photokonduktiven Detektor muß das Photon das Elektron mit nur etwa 1 eV an-

	MCP u. Photokathode	CCD
Prinzip	Photoemissive	Photoconductive
Wellenlänge	Kathodenabhängig	3800 – 7500 Å
Kühlung zur		
Untergrundminimierung	bis -10 °C	bis -90 °C
Quantenausbeute (max.)	ca. 30%	$>$ 90%
Dicke der Absorptionsschicht	300 Å	20 μm
Einfluß des Rauschens	auf Ortsbestimmung	Änderung der Intensität

Tabelle 1.1: Vergleichswerte MCP – CCD

regen. Dies entspricht der Energiedifferenz zwischen dem Elektronenniveau und dem Leitungsband. Mit MCP und CCD werden hier zwei gängige Detektorarten aus jeweils einer Sparte miteinander verglichen.

CCD-Detektoren haben ihre höchste Empfindlichkeit im optischen Wellenlängenbereich zwischen 3800 und 7500 Å. Dort bieten sie hohe Quantenausbeuten ($>$ 90%) und sind in der Lage auch hohe Strahlungsflüsse zu verarbeiten. Werden sie in anderen Wellenlängenbereichen verwendet, z.B. im ultravioletten zwischen 500 und 3000 Å, so wirkt sich die hohe Empfindlichkeit im optischen negativ aus. Die ankommenden optischen Photonen erzeugen dann ein starkes Rauschen, in dem die ultravioletten Photonen schlecht nachgewiesen werden können. Eine Verlängerung der Integrationsdauer zur Verbesserung des Signal-Rausch-Verhältnisses (SNR) ist auch nicht immer möglich: Da die Anregungsenergie nur etwa 1 eV beträgt sind CCD-Detektoren auch empfindlich gegen kosmische Strahlung und dies wirkt sich limitierend auf die Integrationsdauer speziell bei satellitengebundenen Systemen aus (Joseph, 1995). Eine andere Methode zur Verbesserung des SNR ist diese sonnenblind zu machen. Sonnenblindheit bedeutet Unempfindlichkeit des Detektors gegen optische Photonen. Dazu müssen dann bei CCDs beispielsweise Alkali-Metall-Filter (Woods Filter) verwendet werden. Die Verwendung von Filtern hat aber auch Nachteile, da zum einen die Intensität an sich verringert wird und zum anderen Filter nicht der Lage sind, den gewünschten Wellenlängenbereich scharf abzutrennen. Es sind kontinuierliche Übergänge vorhanden, die auch immer noch eigentlich abzutrennende Wellenlängen im Detektor zulassen. CCDs weisen darüberhinaus auch ein hohes thermisches Rauschen auf, weshalb sie Kühlung (< -90 °C) und damit erhöhten technischen Aufwand benötigen.

MCPs sind intrinsische UV-Detektoren, d.h. ihre höchste Empfindlichkeit liegt im ultravioletten Wellenlängenbereich (500 bis 3000 Å). Photonen benötigen eine Energie von einigen Elektronenvolt und damit sind optische Photonen nicht in der Lage Elektronen aus dem Bleioxid-Glas herauszuschlagen. Damit ist bei MCPs immer Sonnenblindheit gegeben, also die Verwendung von Filtern mit ihren negativen Eigenschaften zur Abtrennung des optischen Anteils nicht notwendig. Bedingt durch die hohe Anregungsenerige von einigen Elektronenvolt, erlaubt das generell geringe Rauschen auch die Beobachtung von lichtschwache Objekte mit einem guten Signal-Rausch-Verhältnis. Thermisches Rauschen tritt bei den MCP-Detektoren praktisch nicht auf. Möchte man hingegen im optischen Wellenlängenbereich mit MCP-Detektoren messen, so ist man auf die Verwendung von Photokathoden angewiesen. Meist werden dazu S1-Kathoden (AgOCs) oder III–IV-Kathoden (GaAs oder InAsP) verwendet (Haider, 1988). Diese wandeln die optischen Photonen in Elektronen um, die dann in den Mikrokanal-Platten vervielfacht werden. Die Photokathoden haben nur ein sehr geringes thermisches Rauschen und benötigen kaum Kühlung (~ -10 °C).

MCP-Detektoren, gerade auch für den optischen Bereich, sind dann im Vorteil, wenn hohe Zeitauflösungen und gute Abbildungseigenschaften bei kleinen Strahlungsflüssen gefordert sind. Nachteile der Mikrokanal-Platten liegen in der geringen Quantenausbeute und technischen Rahmenbedingungen, die die Einsatzbereiche limitieren (siehe Abschnitt 2.3).

1.2 Einsatzgebiete

Das Einsatzgebiet mit dem sich auch diese Diplomarbeit beschäftigt ist die Verwendung von MCP-Detektoren in optischen Abbildungssystem. In Haider (1988), Diesch (1988) oder Barnstedt (1985) wurden Untersuchungen mit optischen MCP-Detektoren durchgeführt und ausgewertet.

Die Möglichkeit Photonen im ultravioletten Wellenlängenbereich ortsaufgelöst nachzuweisen prädestiniert MCP-Detektoren für den Einsatz in abbildenden Detektoren. So kam diese Technologie bereits in vielen bekannten und erfolgreichen Projekten zum Einsatz.

Ein Projekt, das im Hause mitentwickelt, geplant und geleitet wurde, ist das ORFEUS-Teleskop (Orbiting and Retrievable Far and Extreme Ultraviolet Spectrometer) mit einem Echelle Spektrometer, dessen Detektor in diesem Institut mit MCP-Technologie gebaut wurde. Mikrokanal-Platten finden sich aber auch in Satelliten, wie HEAO-2 (High Energy Astronomical Observatory, siehe Dunbar (1999)) oder ROSAT (ROentgen SATellite), also in der räumlich auflösenden Röntgenastronomie. Weiterführende Informationen zu diesen Projekten finden sich beispielsweise unter Barnstedt (1999).

Zuerst wurden MCPs als Bildverstärkersystem für abbildende Schirme oder CCD Kameras eingesetzt. Die Pioneer 10 und 11 Missionen zu Jupiter und Saturn starteten 1972 und 1973 mit dieser Technik und lieferten 1973 und 1979 die ersten Nahaufnahmen dieser Planeten (Seal, 1999).

Neue Anwendungsgebiete außerhalb der Astronomie finden sich etwa in der Materialforschung mit UV-Strahlung, der Biolumineszenz, also der Untersuchung der UV-Strahlung von lebenden Organismen, der Atomphysik und vielen anderen Gebieten in denen schnelle Abbildungen und hohe Zeitauflösung gefordert sind (Jagutzki et al., 1999).

Im Rahmen dieser Arbeit wurde nun auf der Basis eines handelsüblichen PC-Systems ein Programm mit graphischer Benutzeroberfläche unter Microsoft Windows NT 4.0 entwickelt. Neben dieser verbesserten Benutzerfreundlichkeit bietet das Programm aber auch weitere Vorteile gegenüber dem alten System. Zur Implementierung wurde die gängige Programmiersprache Borland C++ 5.0 verwendet, die in der sogenannten objektorientierten Programmierung mit gekapselten Datenklassen Vorteile auch für die Zukunft bietet. Mit dieser Programmiertechnik ist es etwa möglich das Programm ohne allzu großen Aufwand auf andere Hardware-Komponenten umzustellen ohne den Rest des Programmes ändern zu müssen. Detaillierte Überlegungen dazu finden sich in Kapitel 5. Als Betriebssystem wurde Microsoft Windows NT 4.0 verwendet. Dies bietet als echtes 32 Bit Betriebssystem viele Vorteile was Geschwindigkeit, Betriebs- und Datensicherheit aber auch die Portierbarkeit auf andere Rechner-Hardware-Systeme, wie etwa ein DEC-Alpha-System, anbetrifft.

Kapitel 2

Aufbau eines optischen MCP-Detektors

Im folgenden Kapitel wird der prinzipielle Aufbau eines optischen MCP-Detektors beschrieben. Dieser besteht aus einer Photokathode, zur Umwandlung der Photonen in Elektronen, Mikrokanal-Platten zur Vervielfachung der Elektronen und einer Anode zum ortsaufgelösten Nachweis der entstandenen Elektronenwolke. Diese Anodensignale werden dann noch von einer geeigneten Elektronik registriert und in einem Datenerfassungssystem aufgezeichnet.

2.1 Funktionsweise einer MCP

MCP sind photoemissive Detektoren. Trifft ein Photon mit einer Energie von einigen Elektronenvolt auf die Mikrokanal-Platte, so werden Elektronen aus dem Material ausgelöst. Die angelegte Spannung beschleunigt diese Ladungen und bei erneutem Auftreffen vervielfachen sich diese immer mehr. Es entsteht eine Elektronenwolke, die auf einem Fluoreszenzschirm oder durch eine geeignete Anode nachgewiesen werden kann.

Die folgenden Abschnitte setzen sich im Detail mit dem Aufbau, den Eigenschaften, verschiedenen Anordnungen, Anoden und begrenzenden technischen Eigenheiten der Mikrokanal-Platte auseinander.

2.1.1 MCP-Aufbau

Eine schematische Darstellung des Aufbaus ist in Abb. 2.1 zu sehen. Die Mikrokanal-Platte wird aus Bleioxid-Glas hergestellt und hat etwa $10^4 - 10^7$ parallele Kanäle mit einem Neigungswinkel von bis zu 15°. Die Kanäle haben einen Durchmesser von $8 - 25$ μm (typisch 12,5 μm) und der Öffnungsanteil an der $5 - 50$ cm^2 großen Oberfläche beträgt etwa 60%. Die Dicke der Mikrokanal-Platte liegt im Bereich von $0,5 - 2$ mm. Entsprechend ist das Verhältnis von Durchmesser zu Länge der Kanäle etwa $1 : 40 - 1 : 100$. An den MCP-Oberflächen sind Elektroden angebracht, an denen 1000 V Betriebsspannung anliegen. Die Platte hat einen Widerstand von ca. 10^9 Ω. Um ein homogenes Feld in den Kanälen zu erzeugen wird die Kanaloberfläche in einer $250 - 450$ °C heißen Wasserstoffatmosphäre halbleitend gemacht. Die Elektronen verlassen die Platte mit $10 - 100$ eV und es entstehen durchschnittlich $1,1 - 3,5$ Sekundärelektronen pro Einzelstoß mit der Kanalwand. Der Verstärkungsfaktor der MCP liegt im Bereich von $10^4 - 10^9$. Diese Eigenschaften sind noch einmal tabellarisch in Tab. 2.1 zusammengefaßt.

2.1.2 Eigenschaften

MCPs sind intrinsisch sonnenblinde UV-Detektoren, da die Energie der Photonen im Wellenlängenbereich oberhalb 3000 Å nicht ausreicht, um Elektronen direkt aus den Kanalwänden auszulösen. Um diese Photonen nachweisen zu können, müssen Photokathoden benutzt werden, die die einfallende Photonen in Elektronen umwandeln. Meistens werden transparente S1-Kathoden aus AgOCs

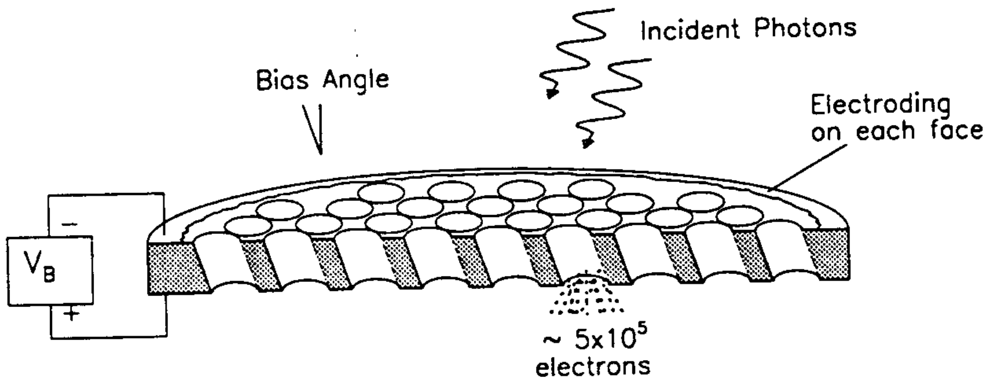

Abbildung 2.1: *Schematischer Aufbau einer Mikrokanal-Platte (MCP). Die Kanäle der MCP sind geneigt, um Ionen-Feedback zu verhindern. Die einfallenden Photonen lösen Elektronen aus, die durch die angelegte Spannung beschleunigt werden und sich an den Kanalwänden vervielfachen. Bild aus Joseph (1995).*

verwendet (Haider, 1988). Die Quantenausbeute beträgt etwa 30% und die Dicke der Absorptionsschicht etwa 300 Å, je nach verwendeter Kathode. Das Kathodenmaterial bestimmt natürlich ebenfalls durch die Abstände seiner Energieniveaus den nachweisbaren Wellenlängenbereich. Eine Kühlung der Photokathode bis etwa -10 °C dient der Untergrundminimierung, d.h. einer Verringerung des Rauschens durch thermische Effekte. Eine weitere Abkühlung bringt keine weiteren Vorteile mehr (Haider, 1988). Die so erzeugten Elektronen werden dann durch die Mikrokanal-Platte verstärkt und durch geeignete Anoden ortsaufgelöst nachgewiesen. Das Rauschen auf den Signalleitungen wirkt sich dabei auf die Genauigkeit der Ortsbestimmung aus, ohne jedoch die Intensität des Signals zu verändern.

2.1.3 Ionen-Feedback

Ab einer MCP-Verstärkung von 10^5 setzt Ionen-Feedback ein. Dabei werden positiv ionisierte Atome zur MCP-Vorderseite hin beschleunigt und lösen dort Elektronen aus. Diese Elektronen werden genauso, wie die die von Photonen ausgelöst werden, beschleunigt und vervielfältigt. Dadurch entsteht eine starke Verfälschung des Signals. Durch eine Kanalneigung von typischerweise 15° kann der Effekt verringert werden, da dann die von Ionen ausgelösten Elektronen nur noch im unteren Teil des Kanals, die von Photonen aber über die ganze Kanallänge vervielfacht werden. Eine weitere Verbesserung bringen gebogene Kanäle (C-Plates, Verstärkung 10^6, Betriebsspannung 1500 – 2000 V) oder die im nachfolgenden Abschnitt beschriebene Chevron-Anordnung.

2.1.4 Chevron-Anordnung

Die Chevron-Anordnung zur Minimierung des Ionen-Feedbacks wird hier ausführlicher beschrieben, da sie in den im Hause verwendeten Detektoren benutzt wird.

Dabei handelt es sich um eine gegenläufige Anordnung (Addition der Neigungswinkel) von zwei oder drei Mikrokanal-Platten hinter der Photokathode. Die Platten haben einen Abstand von etwa 150 μm und die Elektroden sind an einer entsprechenden Spannungskaskade angeschlossen, damit die Ladungen kontinuierlich in Richtung Anode beschleunigt werden. Die Verstärkung der Elektronen aus der Photokathode findet in allen MCP statt und beträgt bis zu 10^9. Die Verstärkung der Elek-

Material	Bleioxid-Glas
Kanalanzahl	$10^4 - 10^7$, gebündelt, parallel
Kanaldurchmesser	$8 - 25\ \mu$m
typisch	$12,5\ \mu$m
Durchmesser:Länge	$1 : 40 - 1 : 100$
Oberfläche	$5 - 50$ cm^2
Öffnungsanteil	bis 60%
Dicke	$0,5$ bis 2 mm
Winkel zur Oberfläche	bis $15°$
Kanaloberfläche	halbleitend
Widerstand	$10^9\ \Omega$
Betriebsspannung	1000 V
Elektronenenergie	$10 - 100$ eV
Verstärkungsfaktor	$10^4 - 10^9$

Tabelle 2.1: Eigenschaften von Mikrokanal-Platten (MCP)

tronen aus positiven Ionen geschieht nur in der letzten MCP und bewegt sich im Bereich 10^5. Damit spielt dieses Signal praktisch keine Rolle mehr (Barnstedt, 1981). Durch die mehrfache Verstärkung findet auch eine Aufweitung der Elektronenwolke statt, die letztlich auf die Anode auftrifft. Diese Aufweitung ist jedoch kein Nachteil, denn die im folgenden beschriebenen Anoden benötigen diese Aufweitung teilweise, um eine gute Ortsauflösung zu erzielen. Auf die Abtastung des entstehenden Impulses wird im Abschnitt 3.2 näher eingegangen.

2.2 Anodenarten

Für die ortsauflösenden Anoden gibt es verschiedene technische Realisierungen, die im folgenden kurz beschrieben sind. Auf die heute verwendete Keil-Streifen-Anode (Wedge-and-Strip-Anode, WSA) wird etwas detaillierter eingegangen.

1. Multi-Anodenfeld: Feld aus $n \times m$ einzelnen, gegeneinander isolierten Elektroden mit jeweils einem eigenen Verstärker- und Zählerkreis. Jede Elektrode entspricht einem Bildpunkt.

2. Drahtgitteranode: Zwei in zwei Ebenen rechtwinklig angeordnete Drahtgitter aus 100 μm dünnen Drähten. Diese sind im Abstand (Mitte-Mitte) des doppelten Drahtdurchmessers parallel zueinander angeordnet. Die Elektronenwolke weitet sich nach dem Austritt aus der MCP auf und trifft auf mehrere Drähte. Die Interpolation der Signale ergibt den Schwerpunkt in x-y-Richtung auf den Bruchteil eines Drahtabstandes genau. Eine Vereinfachung ist, daß benachbarte Drähte mit einem Widerstand verbunden werden und nur noch jeder achte Draht an einen Verstärker angeschlossen wird. Eine geeignete Elektronik macht daraus vier Werte: einen zur Groblokalisierung und drei zur Interpolation.

3. Delay-Line-Anode: Jede Drahtgitteranode verhält sich wie ein kontinuierliches RC-Glied, d.h. ein Impuls wird umso breiter, je länger sein Weg durch die Anode ist. Da der Zeitpunkt des Nulldurchgangs eines Verstärkersignals von der Breite des Eingangsimpulses abhängt, läßt sich aus der Zeitdifferenz je zweier korrelierter Impulse die Ortskoordinate bestimmen. Die andere Möglichkeit ist die Anstiegszeit eines Impulses zu messen, die dann ebenfalls ein Maß für die Entfernung von Auftreffpunkt zu Eckelektrode ist. Diese Art Anoden wurden beispielsweise in den Detektoren des Berkeley Spektrometer im ORFEUS-Teleskop zur Untersuchung des extremen ($400 - 900$ Å) und des fernen UV-Bereichs ($900 - 1150$ Å) eingesetzt (Hurwitz & Bowyer, 1995).

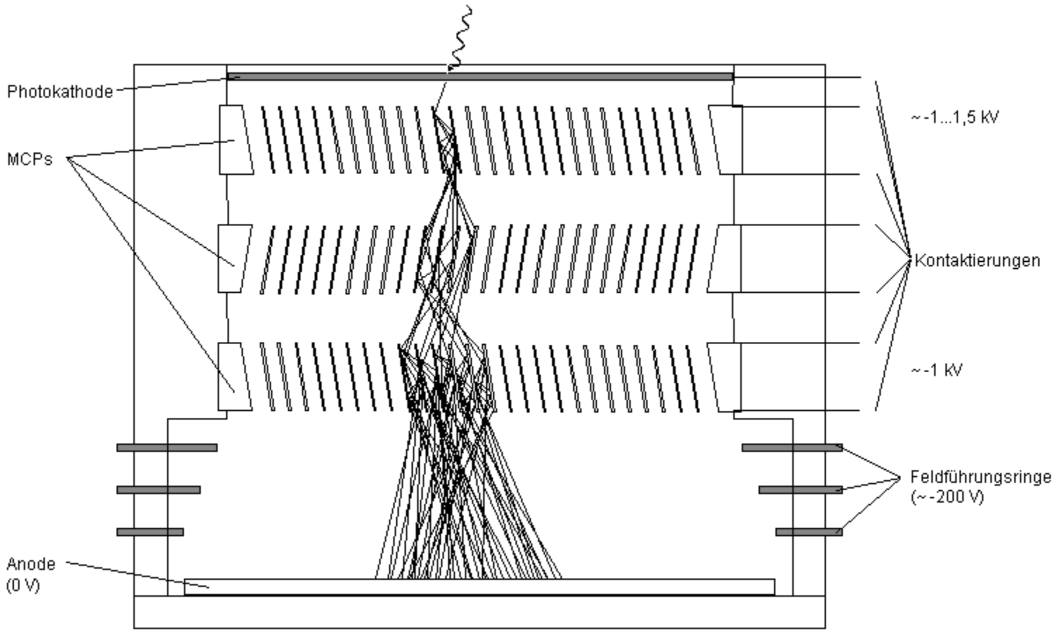

Abbildung 2.2: *Chevron-Anordnung: Hinter der Photokathode befinden sich drei MCP in gegenläufiger Anordnung zur Minimierung des Ionen-Feedback. Die Photonensignale werden in allen MCPs verstärkt (Faktor* 10^9*), während die Ionen dann nur auf die letzte MCP auftreffen (Verstärkung* 10^5*). Die Aufweitung des Impulses ist nicht unerwünscht, da die Keil-Streifen-Anode linear arbeitet, wenn die Elektronenwolke mindestens auf zwei Keil-Streifen-Abschnitte trifft. Bild aus Lindenberger (1992).*

4. Koinzidenzanodenfeld: Hier werden zwei Felder paralleler Leiterbahnen verwendet. Diese sind in zwei gegeneinander isolierten Schichten auf einer Quarzplatte angebracht, und um 90° gegeneinander verdreht. Die Leiter sind so angeschlossen, daß abwechselnd eine Bahn zur Grob- und die nächste zur Feinlokalisierung beiträgt. Notwendig ist, daß jede Wolke mindestens auf zwei Leitungsbahnen auftrifft. Für eine a×b Auflösung benötigt man dann a×b Leiterbahnen in jeder Ebene, die jedoch zu a+b Leitungen zusammengefaßt und herausgeführt werden. Jede Leitung ist an einen eigenen Verstärker- und Diskriminatorkreis angeschlossen. Für eine Anode mit einer Größe von 12×12 mm^2 mit 512×512 Bildpunkten, deren Leiterbahnabstand von Mitte zu Mitte 25 μm in x- und y-Richtung beträgt, werden für jeweils 16 Bahnen zur Grob- und 32 zur Feinlokalisierung $2 \cdot (32 + 16) = 96$ Verstärker- und Diskriminatorenkreise benötigt. Solche Anoden mit noch höherer Auflösung werden beispielsweise in den MAMA-Detektoren (Multi-Anode Microchannel Array) im Space Telescope Imaging Spectrograph (STIS) des Hubble Space Telescope (HST) verwendet (Woodgate, 1997), da sie den Vorteil einer großen Bildfläche bei sehr guter Geometrie bieten.

5. Widerstandsanode: Meist quadratischer Flächenwiderstand an dessen Ecken über vier Elektroden die auftreffenden Signale abfließen können. Aus den Signalen gegenüberliegender Elektroden werden die Koordinaten in x- bzw. y-Richtung gewonnen. Diese Anodenart wurde in den ersten im Hause entwickelten optischen MCP-Detektoren verwendet (Barnstedt, 1981).

6. Keil-Streifen-Anode: Die Keil-Streifen-Anode besteht aus vier gegeneinander isolierten Anodenbereichen, die in Keilen und Streifen angeordnet sind (siehe Abb. 2.3). Die Strukturen sind in der Größenordnung von wenigen μm. Für eine lineare Ortsauflösung ist es notwendig,

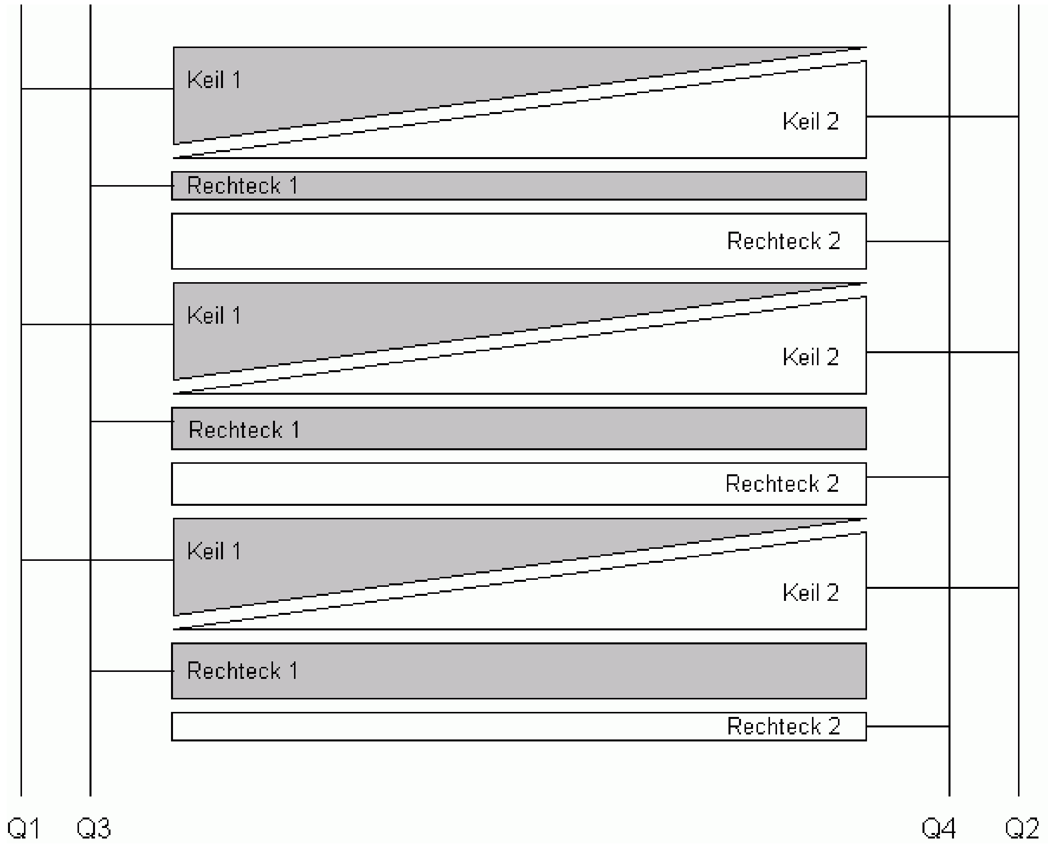

Abbildung 2.3: *Schematische Keil-Streifen-Anordnung: Aus den vier Bereichen werden vier Signale gewonnen, aus denen sich der Ort berechnen läßt. Die wahre Größe der Strukturen auf der Anode liegt im µm-Bereich. Bild aus Lindenberger (1992).*

daß die einfallende Elektronenwolke auf mindestens zwei der Keil-Streifen-Abschnitte auftrifft. Eine Aufweitung der Elektronenlawine, wie sie in der Chevron-Anordnung geschieht, ist hier also sogar von Vorteil. Aus den vier Ladungsverstärkersignalen q_1, q_2, q_3 und q_4 ergibt sich für die Ortskoordinaten:

$$x = \frac{q_1}{q_1 + q_2} \qquad y = \frac{q_3}{q_3 + q_4}$$

Für die Addition und Division der Signale gibt es viele Realisationsmöglichkeiten. Die Pulspositionsanalyse wird später beschrieben. Auf die anderen Verfahren wird nicht näher eingegangen. Der Vorteil der Keil-Streifen-Anode ist eine relativ einfache Elektronik. In den Randbereichen arbeitet die Anode jedoch nicht verzerrungsfrei, da dort die Keile enden und der Teil der Elektronenwolke, der außerhalb der aktiven Fläche auftrifft, verloren geht. Die Keile müßten theoretisch fortgesetzt und der äußere Teile vom Signal wiederum abgezogen werden. Dies ist jedoch nicht möglich.

2.3 Begrenzende Elemente

MCP-Detektoren können nur unter gewissen Rahmenbedingungen sinnvoll betrieben werden. Diese Einschränkungen wirken sich unterschiedlich auf das Ergebnis der Messung aus.

- Sättigung: Die Elektronen innerhalb der entstehenden Wolke besitzen unterschiedliche Impulse, da sie unterschiedliche Beschleunigungsphase durchlaufen haben. Bei hohen Verstärkungen erzeugt die Elektronenlawine am Ende der Kanäle eine hohe Raumladungsdichte. Diese wirkt dem beschleunigenden elektrischen Feld entgegen und verhindert eine weiteres Anwachsen der Lawine, d.h. es tritt eine Sättigung ein. In der Sättigung tritt eine Gaußsche Verteilung auf, deren Maximalwert ein Maß für die Verstärkung der MCP ist. Wird in Sättigung gemessen, können keine Rückschlüsse auf die Energie des einfallenden Photons gezogen werden.

- Nachweiswahrscheinlichkeit: Die Nachweiswahrscheinlichkeit ist abhängig von Art, Energie und Einfallswinkel der Strahlung und vom Verhältnis der Kanalöffnung zur Gesamtfläche. Beim Nachweis von optischer Strahlung spielt das Kathodenmaterial, in dem die Photonen in Elektronen umgewandelt werden, die entscheidende Rolle.

- Untergrundzählrate: Die Untergrundzählrate der MCP liegt bei 1 $\mathrm{cm}^{-2}\mathrm{s}^{-1}$, die bei einem Druck über 10^{-5} mbar ansteigt. Die Untergrundzählrate ist im wesentlichen auf thermische Emission zurückzuführen. Bei Verwendung einer Photokathode mit der MCP steigt die Untergrundzählrate auf $10 - 100$ $\mathrm{cm}^{-2}\mathrm{s}^{-1}$. Dieser Wert setzt eine Untergrenze für die Strahlungsflüsse, die mit den Detektoren noch beobachtet werden können.

- Nachführungsstrom: Die bei der Vervielfachung der auftreffenden Ladung ausgeschlagenen Elektronen müssen durch die Elektroden über das MCP-Material nachgeführt werden. Dieser Nachführungsstrom ist ebenfalls begrenzt und wirkt sich beim Betrieb dahingehend aus, daß die Rate maximal $10 - 100$ Counts $\mathrm{Pixel}^{-1}\mathrm{s}^{-1}$ betragen kann. Befindet sich im Gesichtsfeld des Detektors ein sehr helles Objekt, steigen die Zählraten in einzelnen Pixeln sehr stark an und die Messung wird stark beeinflußt. Beim Betrieb in Spektrographen werden die Spektren eines einzigen Objekts in mehrere Ordnungen zerlegt. Diese werden dann auf den Detektor abgebildet. In diesem Fall treten Zählraten von $10 - 100$ Counts s^{-1} auf der ganzen Detektorfläche auf und nicht in einzelnen Pixeln. Die Begrenzung der Zählrate durch den Nachführungsstrom nimmt also direkten Einfluß auf die mögliche Betriebsart des Detektors und die entsprechende Auswahl der zu beobachtenden Objekte.

- Auflösungsvermögen: Die absolute Untergrenze des Auflösungsvermögens ist die Größe der MCP-Kanäle mit etwa 15 μm. Diese spielt in der Praxis aber keine Rolle. Die eigentliche Begrenzung entsteht durch Rauschen auf den Signalleitungen zwischen Vorverstärkern und der Elektronik. Dies ist maßgebend für die Genauigkeit der Maximalwertbestimmung des gemessenen Impulses und damit für die Genauigkeit der Ortsbestimmung. Weitere Faktoren sind die Anzahl der Bildpunkte pro Bild und der Einfluß statistischer Schwankungen, der durch entsprechend lange Integrationsdauern minimiert werden kann. Die Elektronik des Detektors setzt dementsprechend Grenzen für die möglichen Beobachtungsdauern.

- Signal-Rausch-Verhältnis: MCPs sind relativ unempfindlich gegen kosmische Strahlung, da deren Energie nicht ausreicht, um aus dem Material Elektronen auszuschlagen. Speziell bei satellitengebunden Systemen wird deshalb die Integrationsdauer nicht limitiert, wie dies etwa bei CCD-Detektoren der Fall ist (Joseph, 1995). Dadurch können auch schwache Objekte entsprechend lange beobachtet werden, um ein gutes Signal-Rausch-Verhältnis zu erzielen.

MCP-Detektoren haben Einschränkungen in manchen Bereichen. Wird allerdings eine hohe Zeitauflösung gefordert oder sind nur geringe Strahlungsflüsse vorhanden, so haben sie Vorteile gegenüber anderen abbildenden Detektorsystemen.

Kapitel 3

Vorhandenes Datenerfassungssystem

Abb. 3.1 zeigt den prinzipiellen Aufbau des bisherigen Datenerfassungssystems. Dieses wird zum größten Teil weiterverwendet und setzt damit die technischen Rahmenbedingungen für diese Diplomarbeit. Das folgende beschränkt sich auf die wesentlichen Merkmale und weitere Details zu diesen Systemen können in Barnstedt (1981), Barnstedt (1985), Gutekunst (1986) und Lindenberger (1992) nachgelesen werden.

3.1 Aktueller Detektor

Zur Zeit werden Prototypen der Firma Proxitronic verwendet. Diese haben einen kreisförmigen Durchmesser von 25 mm und beinhalten drei MCPs in Chevron-Anordnung. Die Detektoren enthalten als Auslesesystem eine Hochohm-Keramik mit äußerer Keil-Streifen-Anode. Die Hochohm-Keramik ist auf der Innenseite des Detektors aufgebracht und nach außen auf Masse kontaktiert. Die auftreffenden Ladungen bleiben aufgrund der schlechten Leitungseigenschaften der Keramik lange genug erhalten, um kapazitiv auf die außen angebrachte Anode zu koppeln. Je nach Aufgabe kann ein anderer Anodentyp verwendet werden. Keil-Streifen-Anoden eignen sich etwa sehr gut zur ortsaufgelösten Abbildung, während Delay-Line-Anoden eine bessere Zeitauflösung haben. Die verschiedenen Anodentypen mit ihren Einsatzgebieten werden in Abschnitt 2.2 beschrieben. Dieses System hat nun den Vorteil, daß die Anoden problemlos ausgetauscht werden können und so der Detektor flexibel für unterschliedliche Aufgaben einsetzbar ist. In Richtung der äußeren Anoden wird zur Zeit auch technisch weiterentwickelt (Jagutzki et al., 1999). Zur Zeit sind drei dieser Detektoren mit Keil-Streifen-Anoden in zwei Gehäusen am IAAT vorhanden. Detektor #4351 ist in Gehäuse #1 und Detektor #3971 in Gehäuse #2 eingebaut.

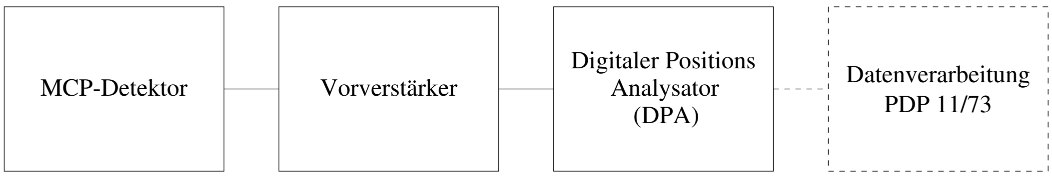

Abbildung 3.1: *Schematischer Aufbau des bisherigen Datenerfassungssystems: Die Abbildung zeigt die Komponenten, die im Datenerfassungssystem vorhanden sind. Das bisherige Datenverarbeitungssystem ist gestrichelt eingezeichnet.*

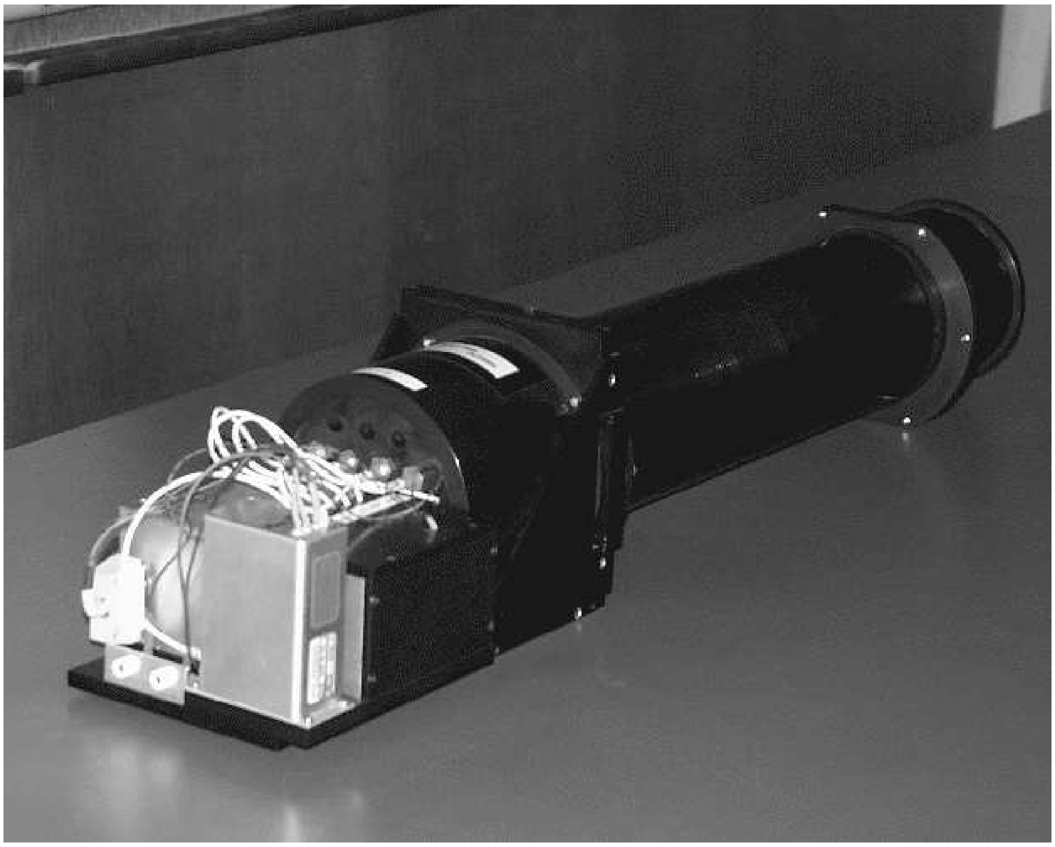

Abbildung 3.2: *Aktueller Proxitronic Detektor: Das Bild zeigt den im Hause verwendeten Detektor der Firma Proxitronic mit Tubus, Vorverstärkerelektronik und der LED-Anordnung zu Testzwecken.*

3.2 Elektronik und Signalformung

Die Elektronik wurde am Anfang so konzipiert, daß das Ausgangssignal der Vorverstärker durch eine zweistufige Impulsformereinheit (Shaper) geschickt wurde. Das daraus resultierende bipolare Signal wurde im Maximum abgetastet und digitalisiert (siehe Abb. 3.3). Der Nachteil in diesem Verfahren ist, daß sich die Unsicherheiten im Zeitpunkt der Maximalwertbestimmung direkt auf die Genauigkeit der Meßwerterfassung auswirkten. Das einmalige Abtasten bot darüberhinaus keine Möglichkeit der Rauschunterdrückung, da der Einfluß des Rauschens nur durch das Vermessen mehrerer Punkte herausgemittelt werden kann. Die Koordinatenberechnung mit nur zwei Werten hat zur Folge, daß Rundungsfehler in der Koordinatenberechnung auftraten, die im Bild deutlich sichtbar waren. Die Totzeit in diesem Meßverfahren betrug etwa 10 μs (siehe Abb. 3.3). Erst nach dem Abklingen des Pulses auf einen Pegel unterhalb der Meßgenauigkeit der Maximalwertabtastung war die Totzeit beendet und der nächste Impuls konnte gemessen werden. Dieses Verfahren hatte zu viele Nachteile und wurde Anfang der 90er Jahre im Rahmen einer Diplomarbeit verbessert (Lindenberger, 1992).

Das neue Verfahren (siehe Abb. 3.4) stellt sich wie folgt dar: Der gravierendste Mangel im alten Verfahren war der Einfluß des Offsets, d.h. eine vertikale Verschiebung des Maximalwertes durch die Elektronik. Es galt also speziell diesen zu beseitigen. Eine Möglichkeit wäre, den bipolaren Impuls zusätzlich im Minimum abzutasten. Dadurch hätte man den Offsetfehler unterdrückt, aber auch gleichzeitig das Signal-Rausch-Verhältnis und damit die Ortsauflösung wesentlich verschlechtert (siehe Abschnitt 2.3). Aus diesem Grund wurde ein anderes Verfahren benötigt. Anstelle der

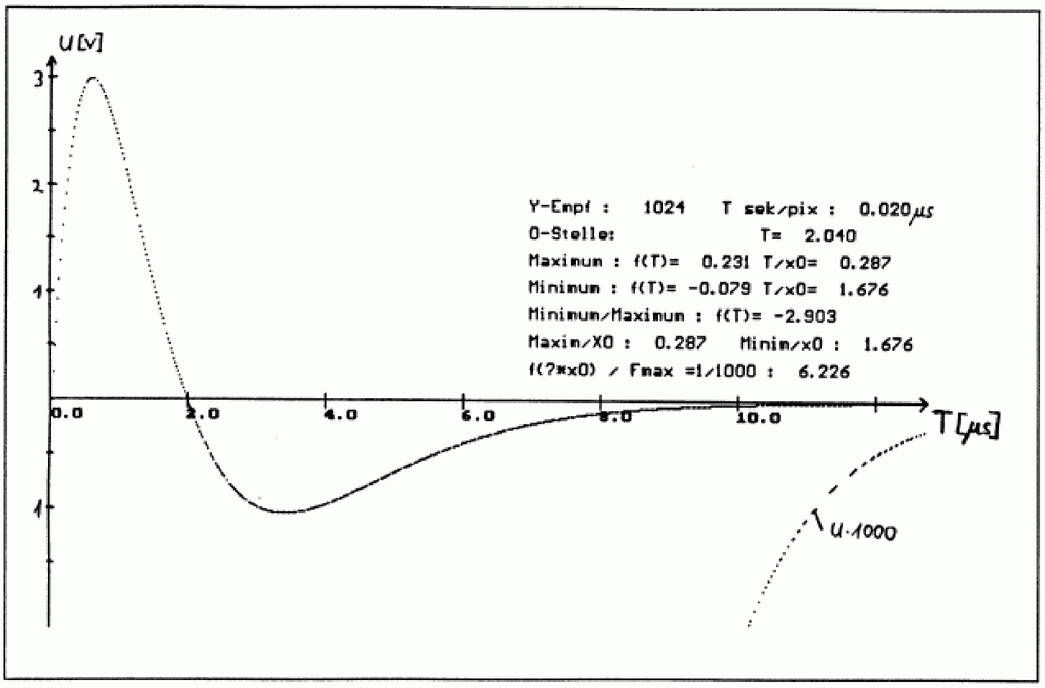

Abbildung 3.3: *Bipolarer Puls am Detektor: Dieses Bild zeigt den Puls am Ausgang der zweistufigen Impulsformereinheit (Shaper). Dieser wird dann im DPA weiterverarbeitet. Bild aus Lindenberger (1992)*

Maximal- und Minimalwertbestimmung wird die Anzahl der Abtastpunkte erhöht, um mit Offsetkorrektur ein besseres Signal-Rausch-Verhältnis zu erhalten. Die Genauigkeit der ADC (Analog-Digital-Wandler), mit Wandlungszeiten von etwa 2 μs, reicht aus, wenn der ganze Dynamikbereich ausgenutzt wird. Dazu muß das Signal so geformt werden, das es möglichst hohe Werte erreicht. Ein Stufensignal (der Pegel bleibt nach dem zu vermessenden Spannungssprung für die Meßdauer konstant) wird mehrmals zur Rauschunterdrückung abgetastet. Durch Summenbildung aus diesen Punkten vergrößern sich die absoluten Beträge, die in die Koordinatenberechnung eingehen und damit wird der Rundungsfehler bei der Division verringert. Optimal wäre ein Signal, das nach dem zu vermessenden Abschnitt schlagartig abklingt und damit die Totzeit minimiert, also ein Rechteckimpuls. Der Offsetfehler kann eliminiert werden, indem nach dem Nutzsignal weiter Abtastungen stattfinden, die mit den übrigen Werten verrechnet werden.

Diese von Lindenberger (1992) entwickelte Elektronik erreicht eine Totzeit kleiner 2 μs und demzufolge Zählraten von 500.000 Counts s^{-1}. Die Genauigkeit der Ortsinformation beträgt 11 – 12 Bit von denen 10 direkt und die restlichen ein oder zwei zur Rundung des Divisionsergebnisses verwendet werden. Diese Ergebnisse spiegeln sich direkt in den Anforderungen an das im Rahmen dieser Arbeit entwickelten Programm wieder (siehe Abschnitt 4.1).

3.3 Digitaler Positions-Analysator

Der Digitale Positions-Analysator (DPA) wurde im Rahmen der Dissertation von J. Barnstedt im Hause konzipiert und gebaut. Hier werden nur kurz die Eigenschaften und Einstellmöglichkeiten des DPA zusammengefaßt. Es gibt zur Zeit zwei dieser Geräte, die sich nicht wesentlich unterscheiden. Die ältere Version ist deutsch beschriftet, die neuere englisch. Die neuere Elektronikbox enthält die etwas schnelleren Analog-Digital-Wandler und liefert geringfügig höhere Datenraten. Ausführlichere

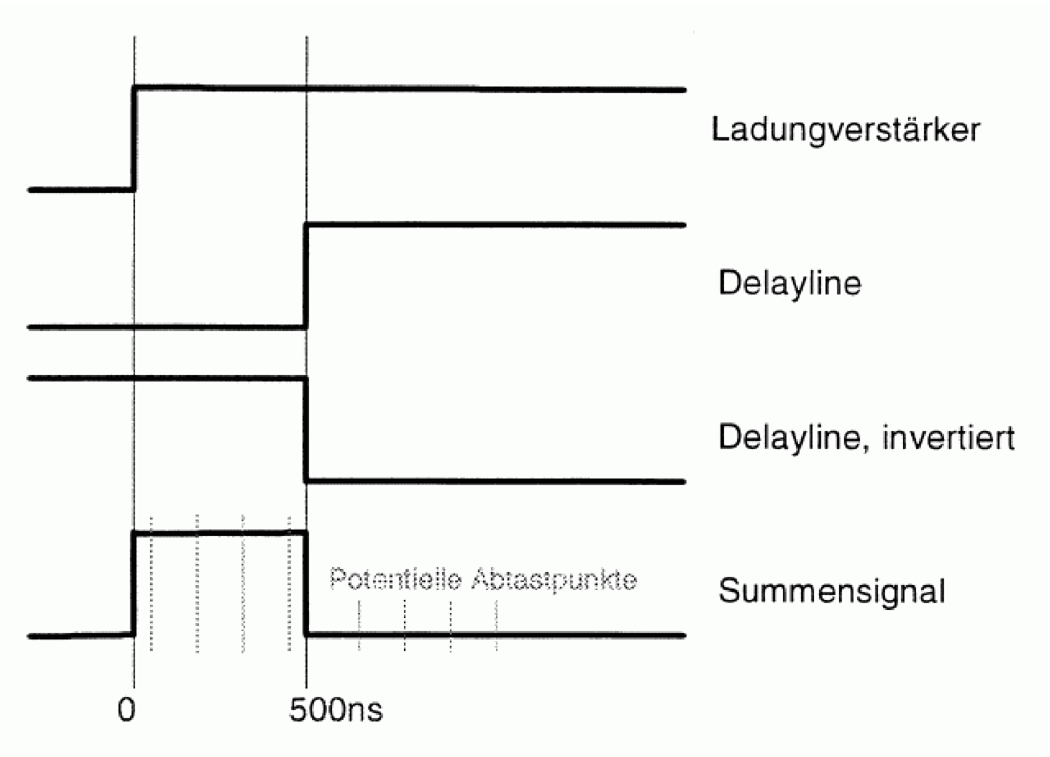

Abbildung 3.4: *Verbesserte Pulsformung: In dieser Abbildung ist das neue Verfahren, das von Lindenberger (1992) entwickelt wurde, graphisch dargestellt. Um die Genauigkeit zu erhöhen werden der zu vermessende Spannungssprung zuerst verzögert, dann invertiert und vom Eingangssignal subtrahiert. Dadurch entsteht ein Rechteckimpuls, der sowohl für die Maximalwertbestimmung wie auch für die Totzeitminimierung optimal ist. Bild aus Lindenberger (1992)*

Darstellungen dazu finden sich in Barnstedt (1985).

Der DPA enthält die Analog-Digital-Wandler und alle digitale Elektronik zur Berechnung der Ortskoordinate der aufgetroffenen Elektronikwolke. Diese Anordnungen befinden sich auf Steckkarten in der Box und bieten weitere Einstellmöglichkeiten, etwa den Dynamik-Bereich der AD-Wandler. Die Box enthält auch die Steuerung des Detektors und der Photokathode. An einem Drehschalter lassen sich die Modi der Datenerfassungsmöglichkeiten der Einzelereignisse wählen:

1. Impulshöhen (z)
2. Ortskoordinaten (xy)
3. Ortskoordinaten mit Zeitmarke (xyt)
4. Ortskoordinaten, Impulshöhen und Zeitmarke (xyzt)
5. Ortskoordinaten über Hardware-Integrations-Interface (HWI)

Am DPA ist auch ein Zeitzähler für die Meßdauer angebracht. Dieser zeigt die Meßdauer totzeitkorrigiert in Sekunden an, d.h. der Zähler wird gestoppt wenn entweder das Datenerfassungssystem keine weiteren Daten abholt oder Totzeiten in der internen Elektronik (AD-Wandler etc.) auftreten.

Als Eingangssignal verwendet der DPA die Vorverstärkersignale des Detektors und als Ausgang Anschlüsse für das 16 Bit Parallel-Interface DRV11J (DEC), das für die ersten vier Modi benutzt

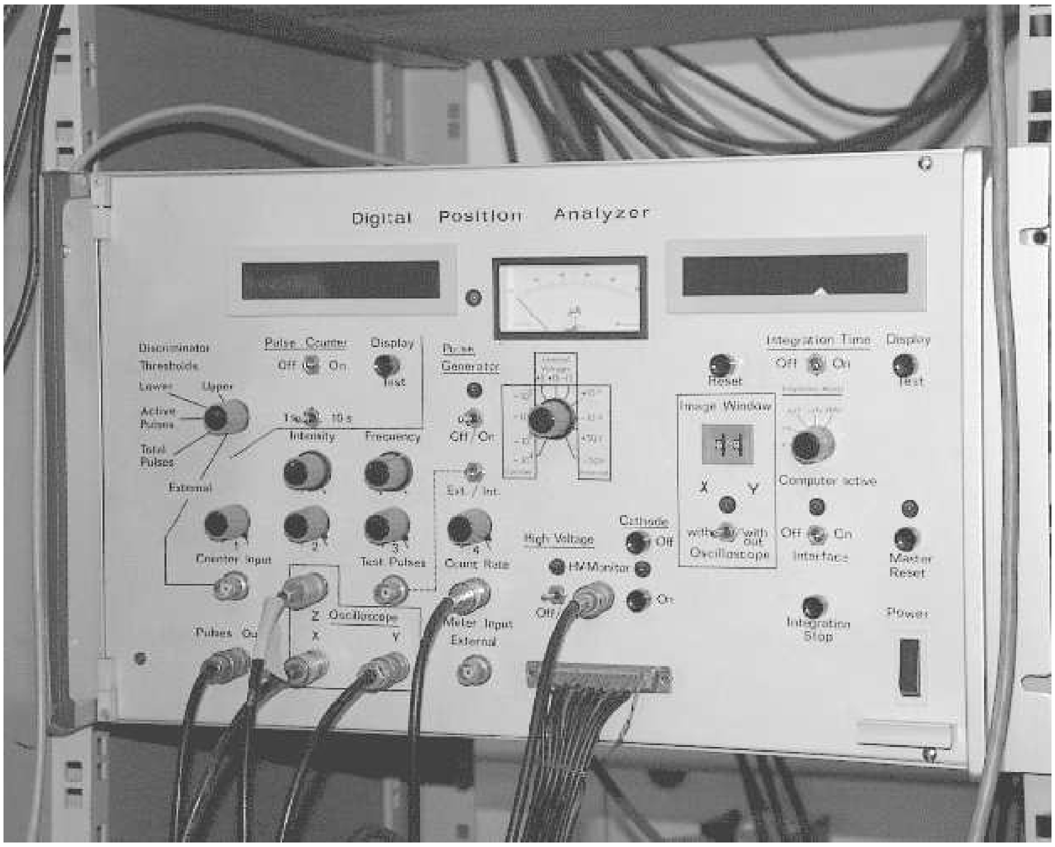

Abbildung 3.5: *Digitaler Positions-Analysator: Das Bild zeigt die aktuellste Version der Elektronikbox. Diese beinhaltet die Analog-Digital-Wandler und die digitale Elektronik zur Positionsberechnung. Auf der rechten Seite befinden sich die Einstellmöglichkeiten für die Meßarten, in der Mitte die Ansteuerung des Detektors und auf der linken Seite die Testelektronik zur Pulssimulation.*

wird, und einen Anschluß über ein im Hause von Barnstedt (1985) entwickeltes Interface für die direkte Hardware-Integration mit der PDP 11/73. Die Elektronik des DPA läßt, mit den derzeit verwendeten Bauteilen, einen Datendurchsatz von ca. 100.000 Counts s^{-1} zu.

Für die ersten vier Betriebsmodi wird im DPA ein Multiplexverfahren verwendet, das speziell auf die Belange der PDP 11/73 zugeschnitten ist und dort eine möglichst schnelle Datenverarbeitung erlaubt (Gutekunst, 1986). In Tab. 3.1 und Tab. 3.2 ist der Aufbau und die Übertragung zusammengefaßt. Bei der PC-Verarbeitung werden diese Daten dann wieder mit Masken und Bitshifts extrahiert und in Koordinaten zurückgerechnet. Das Hardware-Interface liefert reine xy-Koordinaten der Elektronenwolke und schreibt im Betrieb mit der PDP 11/73 direkt in den Bildspeicher des Systems ohne Datentransfer durch ein Programm. Dazu ist das HWI-Interface direkt mit dem Bus des PDP 11/73-Rechners verbunden. In Abschnitt 3.5 und in Kapitel 7 finden sich weitere Überlegungen zum HWI-Modus.

3.4 PDP 11/73

Im Laufe der Jahre wurden für die Datenerfassung verschiedene DEC-PDP 11 Rechner am Institut verwendet, beginnend mit der PDP 11/34 Anfang der 80er Jahre bis zur PDP 11/73 am heutigen Tage.

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1.Wort	WGB	Y11	Y10	Y9	Y8	Y7	Y6	Y5	Y4	T20	T19	Y1	Y0	T18	X1	X0
2.Wort	WGB	T17	T16	Y3	Y2	X11	X10	X9	X8	X7	X6	X5	X4	X3	X2	T15
3.Wort	WGB	T14	T13	T12	T11	T10	T9	T8	T7	T6	T5	T4	T3	T2	T1	T0
4.Wort	WGB	–	Z13	Z12	Z11	Z10	Z9	Z8	Z7	Z6	Z5	Z4	Z3	Z2	Z1	Z0

Tabelle 3.1: Datenworte des DPA

Betriebs-Modus	Reihenfolge
z	Wort 4
xy	Wort 1, Wort 2
xyt	Wort 1, Wort 2, Wort 3
xyzt	Wort 1, Wort 2, Wort 3, Wort 4
Hardware Integration (HWI)	20 Bit xy - Information

Tabelle 3.2: Übertragene Worte im jeweiligen Betriebs-Modus des DPA

Die PDP-Serie war ein Vorgänger der heutigen VAX-Rechner. Als Betriebssystem wurde RSX-11M verwendet, welches schon echten Multi-User- und Multi-Tasking-Betrieb erlaubte. Um die Rechenleistung voll nutzen zu können wurde der Rechner hier aber nur im Single-User-Modus betrieben. Die Multi-Tasking Eigenschaften fanden allerdings von Anfang an Verwendung. Die Benutzeroberfläche der PDP 11/73 ist ähnlich dem DCL-Benutzerinterface des VAX-VMS-Betriebssystems.

Im weiteren wird hauptsächlich die PDP 11/73 beschrieben, da sie heute noch in Betrieb ist, und nun durch das neue System abgelöst wird. Der DEC DJ11-C Prozessor hat ein Taktrate von etwas weniger als 1 MHz. Ein Mikrozyklus dauert ca. 276 ns, was einem Bustakt von etwa 4 MHz entspricht, wobei die verschiedenen Zusatzkarten nach Anmeldung beim System Busmaster werden können. Die PDP 11/73 verfügt über maximal 4 MB Hauptspeicher, der hier auch vorhanden ist. 3 MB wurden als Bildspeicher reserviert (maximal $3 \times 512 \times 1024 \times 16$ Bit) und können vom System nicht für Programme verwendet werden. Die Programme, wie sie auch in Abschnitt 3.5 beschrieben werden, sind in Assembler und Fortran programmiert.

Das Daten-Interface ist 16 Bit DEC DRV11J Parallel DMA Standard, der sich im Laufe der Jahre unter Beibehaltung der wesentlichen Eigenschaften aus dem DEC DRV11C Parallel-Interface entwickelt hat. Dieses wurde für die meisten Integrationsmöglichkeiten des DPA verwendet. Die Datenübertragungsrate dieses Interfaces im Bildintegrations-Modus von 500 kwords s^{-1} ist nach wie vor beachtlich (siehe Tab. 4.2).

Für die Hardware-Integration steht ein von Barnstedt (1981) entwickeltes, eigenes Interface zur Verfügung, welches es erlaubt, daß ohne den Ablauf eines Programmes auf der PDP 11/73 direkt in den 3 MB großen, reservierten Bildspeicher integriert wird. Dieser Zugriff über den Bus ohne die CPU ist am direktesten und damit auch am schnellsten.

Prozessor	DEC DJ11-C
Taktrate im Prozessor	ca. 1 MHz
Taktrate auf dem Bus	ca. 4 MHz
Hauptspeicher	4 MB (3 MB für Bildspeicher)
Betriebssystem	RSX-11M, Multi-User, Multi-Tasking
Interface	16 Bit DMA DRV11J Parallel TTL-Logik
	HWI: Eigene Entwicklung (Barnstedt, 1985)
Übertragungsrate	Modus xy: 500 kwords s^{-1}
DCL-Benutzerinterface	ähnlich VMS

Tabelle 3.3: Eigenschaften des PDP 11/73-Systems

Der Betrieb der PDP 11/73 wird zunehmend schwieriger, da aufgrund des Alters Hardware-Komponenten kaum noch zu bekommen sind. Dies machte die Entwicklung eines alternativen Systems, wie es in dieser Arbeit vorgestellt wird, nötig, um die optischen Detektoren auch weiter betreiben zu können.

3.5 Integrations- und Bearbeitungs-Software

Neben Programmen zur Datensicherung auf Magnetbänder, die hier nicht näher beschrieben werden (Barnstedt, 1985), gibt es im wesentlichen vier Programme zur Datenerfassung und -verarbeitung, die auch bei den Vergleichsmessungen (siehe Kapitel 6) verwendet wurden.

1. PHA: Programm zur Aufnahme von Impulshöhen-Verteilungen (DPA-Modus z oder xyzt) ohne Darstellung während der Integration.
2. INT: Aufnahme von Bildern (DPA-Modus xy, xyt, xyzt oder HWI) mit der möglichen Löschung des Bildspeichers vor Integrationsbeginn.
3. BTR: Programm zur Laufzeit-Darstellung des Bildspeichers auf einem separaten Monitor.
4. BVP: Bildverarbeitungs-Software mit Funktionen etwa zur Bestimmung der Zählrate im Bild, Erstellung von Linien-, Flächen- oder Konturplots oder Graustufenbildern, Summen und Mittelwerte aller Ereignisse und Radialverteilungen in zuvor bestimmten Bildausschnitten. Dieses Programm kann auch parallel zur Bildintegration gestartet werden, um die Integrationsdauer für eine ausreichende Bildintensität abschätzen zu können.

Diese Programme sind in Fortran und Assembler programmiert und erlauben damit eine maximale Ausnutzung der Rechenkapazität des PDP 11/73-Systems. Wird während der Laufzeit der Bildspeicher dargestellt (Programm BTR), so geht die Zählrate im Bild deutlich zurück, wie die Vergleichsmessungen in Kapitel 6 zeigen.

Genauere Angaben zu den Programmen, deren Aufbau und Bedienung finden sich in Barnstedt (1985) ab Seite 45.

Kapitel 4

Problemstellung

Die Hauptaufgabe dieser Diplomarbeit bestand darin, das Datenverarbeitungssystem PDP 11/73 durch ein PC-System zu ersetzen. Dabei wurde Wert darauf gelegt ein handelsübliches System zu verwenden, um gegenüber einer speziellen Entwicklung sowohl Zeit als auch Geld zu sparen. In den folgenden Abschnitten ist zum einen die Anforderung an das neue System aufgelistet und zum anderen die verfügbare neue Hardware angegeben.

4.1 Software-Anforderungen

Die neue Software sollte in der Lage sein die folgenden Anforderungen zu erfüllen. Angaben in Klammern beziehen sich auf die Betriebsmodi des DPA.

1. Geschwindigkeit bis 500.000 Counts s^{-1}
2. Impulshöhen-Integration (z)
3. Bild-Integration (xy)
4. Bild- und Impulshöhen-Integration (xyzt)
5. Datenkollektion in einer Datei (xyt,xyzt)
6. Bilder: 1024×1024 Bildpunkte, 16 Bit Daten-Integration und Darstellung mit 8 Bit Farbtiefe und eigener Farbtabelle und logarithmische oder lineare Umsetzung mit optionaler Maximalwertorientierung unter Verwendung einer schnelle Darstellungsroutine
7. Impulshöhen: Auf die Fenstergröße angepaßte lineare oder doppelt logarithmische, normierte Histogrammdarstellung im eigenen Fenster mit Menüführung
8. Timergesteuerte Aktualisierung der verschiedenen Darstellungen während der Daten-Integration
9. Fensterklasse zur Bildbetrachtung im Bitmap-Format
10. Aufarbeitung der Kollektions-Dateien
11. Auswertungsroutinen für Bild und Daten
12. Fensterverwaltung und Darstellungsoptionen
13. Objektorientierte Programmierung mit Threads

4.2 Verfügbare neue Hardware

Im folgenden werden die technischen Daten des neuen Verarbeitungssystem aufgelistet. Dieses stand zur Verfügung, um die Elektronik des Detektors anzusprechen.

4.2.1 PC-System

Es handelt sich hierbei um ein handelsübliches PC-System mit technisch bereits überholter Ausstattung. Die Daten sind in Tab. 4.1 zusammengestellt.

Prozessor	Intel Pentium
Taktrate	133 MHz
Hauptspeicher	64 MB
Graphikspeicher	2 MB

Tabelle 4.1: Technische Daten des verwendeten PC.

4.2.2 Daten-Interface

Als Daten-Interface ist eine digitale 32 Bit Input-/Output-Karte der Firma National Instruments vorhanden, die bereits für andere Anwendungen im Hause Gebrauch fand. Die Karte verwendet eine Pinbelegung, die mit dem im DPA eingebauten DRV11J Parallel Interface kompatibel ist.

Die Karte wurde Anfang der 90er Jahre für die Verwendung mit 16 MHz IBM kompatiblen Rechnern entwickelt. Die 32 Bit Datenleitungen sind in 4 Gruppen zu je 8 Bit gegliedert0 und die ersten beiden und letzten beiden können zu einer 16 Bit Gruppe konfiguriert werden, für die dann weitere Steuer- und Handshaking-Leitungen zur Verfügung stehen (siehe Anhang B). Die NI AT-DIO32F-Steckkarte hat einen männlichen 50 Pin Stecker mit einer Belegung, die mit dem DEC DRV11J Parallel Interface und den meisten anderen 32 Kanal digital I/O-Geräten verwendet werden kann. Die Daten der Karten sind in Tab. 4.2 zusammengestellt. Der dort erwähnte RTSI-Bus (Real-Time System Integration-Bus) ist eine Entwicklung von National Instruments und dient dazu, mehrere in ein System eingebaute AT-Karten dieser Firma zum Zweck der Synchronisierung miteinander zu verbinden.

Pinbelegung	DEC DRV11J Parallel
Takt	20 MHz
Timing	Onboard Clock oder RTSI-Bus
Bandbreite	32 Bit
Steuerleitungen	8
Masseverbindungen	10
Übertragungsrate	zu 16 MHz IBM Kompatiblen
Typisch	450 kwords
Maximum	660 kwords
DMA	330 kwords
DMA-Kanäle	2
Interrupts	2

Tabelle 4.2: Technische Daten der verwendeten NI AT-DIO32F-Karte

Kapitel 5

Lösung der Problemstellung

Im neuen Datenerfassungssystem wurden alle Komponenten, bis auf die Datenverarbeitung durch die PDP 11/73, beibehalten, d.h. das neue System wird an den Datenausgang des DPA angeschlossen und muß in der Lage sein dessen Formate zu dekodieren. Um den DPA mit der NI AT-DIO32F auf dem Hardware-Niveau verbinden zu können, mußte eine neue Adapterplatine entwickelt werden, die in Abschnitt 5.1 beschrieben wird.

Wie im vorhergehenden Kapitel aufgelistet, stand ein handelsüblicher PC und eine digitale 32 Bit Input-/Output-Karte zur Verfügung. Der PC kann mit verschiedenen Betriebssystemen betrieben werden, die dann jeweils eine andere Implementierung erfordern. Für das neue System wurde auf Microsoft Windows NT 4.0 zurückgegriffen. Dieses moderne 32 Bit Betriebssystem hat einige wesentliche Vorteile, was Geschwindigkeit, Betriebssicherheit und zukünftige Entwicklung anbetrifft. Die Eigenschaften und deren Auswirkungen auf die Programmimplementierung werden im Abschnitt 5.2 erläutert.

Das Windows-Programm selbst wurde in der gängigen Programmiersprache Borland C++ 5.0 geschrieben. Die Realisierung des Programms im einzelnen wird im abschließenden Abschnitt 5.4 dieses Kapitels aufgezeigt.

5.1 Adapterplatine

Zur Verbindung der beiden Systeme mußte eine neue Platine angefertigt werden. Das DRV11J Parallel Interface verwendet einen 37 poligen Sub-D Stecker (Cannon Stecker) nach DIN 41652. Die NI AT-DIO32F hat einen Flachbandkabelanschluß mit 50 Polen in zwei Reihen nach DIN 41651 (Stiftleiste gerade). Um diese korrekt miteinander zu verbinden wurde ein Adapter angefertigt, der entsprechend der Pinbelegung (siehe Anhang B) die Anschlüsse umsetzt.

Zu Testzwecken wurden die Datenleitungen über Schalter auf Masse (GND) verkabelt. Die Konfigurations-Software des Treibers ermöglicht den Test der einzelnen Datenblöcke durch Ein- oder Ausgabe auf Knopfdruck. Um die korrekte Ein- und Ausgabe der Input- und Output-Blöcke zu testen wurde diese Verkabelung erstellt. Sind die Datenleitungen der einzelnen Blöcke frei und nicht auf Masse geschaltet ergeben sich Effekte, die einen Test mit reproduzierbaren Ergebnissen nicht ermöglichen. So kam es vor, daß Eingänge aus dem vorhergehenden Muster noch logisch ‚high‘ (5 V) waren, obwohl beim aktuellen Test an diesen keine Spannung angelegt war. Dieser Effekt trat aber nicht immer auf. Durch die Verbindung auf Masse über einen Schalter können nun einzelne Meßfüße frei geschalten und mit einer logischen Spannung (5 V) versehen werden. Die so erzeugten Muster können fehlerfrei und reproduzierbar eingelesen werden.

Für den Betrieb der Platine und des Meßsystems ist es unerläßlich, daß alle Datenleitungen frei geschaltet sind. Die Schalter sind so verkabelt, daß in der Schalterstellung ‚ON‘ die Datenleitung auf Masse (GND) geschaltet sind. Für den Betrieb müssen sich alle Schalter in der Stellung ‚OFF‘ (Numerierungsseite) befinden. Ist dies nicht der Fall, sind alle Datenleitungen des DPA ohne einen

Widerstand konstant auf Masse (0 V) gelegt und Messungen nicht möglich. Der entstehende Kurzschluß kann darüber hinaus Beschädigungen der Elektronik verursachen.

5.2 Windows-Betriebssystem

Für die Erstellung und den Betrieb des Programms wurde das Betriebssystem Microsoft Windows NT 4.0 verwendet. Dies hat große Vorteile, was die Datensicherheit, den Betrieb, die Portierbarkeit und die zukünftige Entwicklung anbetrifft. Diese Eigenschaften waren für die Entscheidung dieses System zu verwenden mit ausschlaggebend und werden im folgenden noch näher erläutert.

Die Windows-Programmierung unterscheidet sich wesentlich von der Programmierung unter Microsoft DOS. DOS Algorithmen sind ‚egoistisch' und belegen alle Systemressourcen. Dies bedeutet eine wesentliche Erleichterung der Kommunikation mit der Außenwelt, wie z.B. Tastatureingabe oder Bildschirmausgabe, da für diese Funktionen Standardroutinen vorhanden sind. Die Verarbeitung von Eingaben über fortgeschrittene Techniken, wie z.B. der Maus, ist jedoch mühsamer Kleinarbeit unterworfen.

Ganz anders ist dies bei Microsoft Windows. Dieses Betriebssystem ist vollständig graphisch orientiert. Dabei teilen sich mehrere Programme die Systemressourcen und der Wechsel zwischen mehreren Anwendungen ist kein Problem. Diese Aufteilung erfordert einen erheblichen Mehraufwand in der Verwaltung des Systems, der vom Windows-Manager übernommen wird. Er sorgt für den korrekten Ablauf der Programme, übernimmt die Kommunikation mit der Außenwelt und die Verwaltung der Peripheriegeräte. Windows-Programme können damit weitgehend hardwareunabhängig erstellt werden, und das Betriebssystem sorgt für die korrekte Umsetzung der Programmanforderung auf die jeweils vorhandene Hardware. Dies bringt eine große Verbesserung in der Bedienungsfreundlichkeit, verlangt aber vom Programmierer die Bereitstellung einer korrekten Windows-Manager-Schnittstelle seines Programms.

5.2.1 Entwicklungsstufen

Im Laufe der Jahre durchlief das Microsoft Windows-Betriebssystem verschiedene Entwicklungsstufen, die im folgenden etwas näher beschrieben werden.

a) Windows 3.x ist kein eigenes Betriebssystem, sondern setzt eine graphische Oberfläche auf DOS auf. Dies bedeutet, daß Windows 3.x Anwendungen auf 16 Bit Code beschränkt sind. In dieser Entwicklungsstufe wird noch das kooperative Multi-Tasking verwendet, das im folgenden noch näher erläutert wird (siehe Abschnitt 5.2.2).

b) Windows 95 verhält sich schon wie ein 32 Bit Betriebssystem, ist aber selbst noch in großen Teilen als 16 Bit Code implementiert. Es hat daher die Vorteile einer 32 Bit Umgebung, ist aber ineffizienter und störungsanfälliger als Windows NT.

Windows 95 bringt aber auch noch andere Neuerungen. Neben der Verwendung von preemptivem Multi-Tasking (siehe Abschnitt 5.2.2) werden nun die Anwendungen mit Threads verbunden (siehe Abschnitt 5.2.3). Es werden lange Dateinamen und Standard-Dialogfenster verwendet, um die Anwendungen gleichförmiger und benutzerfreundlicher zu machen. Ein Unterschied zu Windows NT ist, daß sehr zeitintensive Anwendungen Windows 95 aufgrund der teilweisen 16 Bit Implementierung noch immer zum Absturz bringen können.

c) Windows NT ist ein echtes 32 Bit Betriebssystem. Es verbindet die Verbesserungen von Windows 95 mit weiteren entscheidenden Vorteilen, die auch dazu beitrugen ein Windows NT System für die Umsetzung dieser Diplomarbeit zu verwenden. Neben dem FAT-Dateisystem (File Allocation Table) wird ein eigenes NTFS-System (Windows NT File System) verwendet, das mehr Schutz gegen Viren bietet und echten Multi-User-Betrieb erlaubt. Die Speicherverwaltung ist streng nach Sicherheitsaspekten ausgerichtet und Systemabstürze sind selten. Windows NT kann mehrere Prozessoren für sein preemptives Multi-Tasking verwenden und erlaubt die Ausführung von Microsoft DOS-, 16 Bit Windows- und IBM-OS/2-Anwendungen. Dieses System kann aber auch von der Intel-Plattform auf RISC-Rechner, DEC-Alpha oder Motorola PowerPC portiert werden. Windows NT setzt die Hardware-Kapselung umfassend durch und der Zugriff auf die Peripherie-Geräte erfolgt nur noch treibergesteuert (siehe Abschnitt 5.2.7).

5.2.2 Multi-Tasking

Bei Windows Betriebssystemen werden hauptsächlich zwei Multi-Tasking Konzepte verwendet:

1. **Kooperatives Multi-Tasking** wurde von Windows 3.x auf 16 Bit Rechnern verwendet. Wurde Windows aufgerufen, übernahm es die Kontrolle über die CPU und die gestarteten Programme wurden in eine Liste eingetragen. Anhand dieser Liste wurden die Programme abgearbeitet. Jedes aktivierte Programm überprüfte, ob ein Ereignis vorhanden war und arbeitete die entsprechende Funktion als Reaktion darauf ab. Danach wurde die Kontrolle wieder an Windows zurückgegeben. Der wesentlich Nachteil lag darin, daß das Programm die Kontrolle von sich aus wieder abgeben mußte und Anwendungen, die sehr zeitintensiv waren oder in Endlosschleifen gerieten, das System lahmlegten.

2. **Preemptives Multi-Tasking** wird in den 32 Bit Windows-Versionen verwendet und bringt bedeutende Vorteile. Den einzelnen Threads (siehe Abschnitt 5.2.3) werden Zeitscheiben zugeordnet, die die Nutzungsdauer der CPU durch die Anwendung bestimmen. Nach Ablauf der Zeitscheibe wird das Programm unterbrochen, unabhängig von der Beendigung einer Ereignisbehandlung, und das Betriebssystem erhält die Kontrolle zurück. Bei der nächsten Zeitscheibe wird dann die Ereignisbehandlung korrekt weitergeführt. Dies hat den Vorteil, daß die quasi-parallelen Abläufe reibungsloser und flüssiger vonstatten gehen. Der Benutzer oder Programmierer kann einzelnen Anwendungen unterschiedliche Prioritäten zuordnen und damit den Anteil an der CPU-Auslastung steuern. Das Betriebssystem behält jederzeit die Kontrolle und wird nicht mehr von einzelnen Anwendungen blockiert. Eine zeitintensive Anwendung blockiert sich nur noch selbst insofern, als daß während der Abarbeitung des Ereignisses keine weiteren Eingaben mehr möglich sind.

5.2.3 Multi-Threading

Unter Windows 3.x ist jedes Programm ein Task und erhält eigenen Programm- und Datenspeicher. Bei Windows 95 und Windows NT ist jedes Programm mit einem oder mehreren Threads verbunden. Der Hauptthread, der dem Task bei Windows 3.x entspricht, ist in der Lage, weitere Threads zu erzeugen, die vom System wie eigene Anwendungen bei der Zeitscheibenverteilung behandelt werden, da jeder Thread eine eigene Botschaftsverarbeitung erhält (siehe Abschnitt 5.2.6). Dies bietet jeder Anwendung die Möglichkeit zeitintensive Abläufe, wie die Druckerausgabe oder die Graphikberechnung, als Thread abzuspalten und selbst reaktionsfähig zu bleiben.

5.2.4 Speicherverwaltung

Zur Zeit von Microsoft DOS auf Intel 8086 und 8088 Prozessoren konnte 1 MB Speicher adressiert werden. In den IBM-Kompatiblen wurden vom Video-Display und dem ROM (Read Only Memory) basierten System-BIOS (Basic Input Output System) die oberen 384 KB des Adressraumes reserviert. Damit blieben die bekannten 640 KB Speicher für die Programme übrig. Die Adressierung von 1 MB Speicher erfordert eine 20 Bit Adresse. Um dies in 16 Bit Prozessoren zu realisieren, wurde die Unterteilung der Adresse in zwei Bereiche, Segment und Offset, eingeführt. Jedes Segment ist

64 KByte groß und damit passen 16 Segmente in 1 MB. 4 Bit der ersten 16 bestimmen das Segment und die zweiten 16 Bit den Offset. Dieses Prinzip wurde aus Kompatibilitätsgründen bis heute durchgehalten und hat damit die Akzeptanz des Systems wesentlich erhöht.

Windows 95 und Windows NT verfügen beide über eine echte 32 Bit Adressierung und damit entfällt für den Programmierer beispielsweise die Unterscheidung zwischen *near*- und *far*-Zeigern bei der C-Programmierung, denn nun sind alle Zeiger im 32 Bit Format. Genauso gibt es keine Speichermodelle (small, medium, compact, large und huge) mehr, und die Differenzierung von lokalem Heap (Speicher im Datensegment abzüglich globaler Daten und Stack) und globalem Heap (restlicher verfügbarer Speicher) ist ohne Segmentgrenzen unnötig. Es hat aber auch die Konsequenz, daß Datentypen wie *integer* angepaßt wurden, und nun ebenfalls 32 Bit groß sind. Bei der Verwendung älteren Programmcodes kann dies zu Problemen führen, wenn der Programmierer versucht diese an 16 Bit Datentypen, wie beispielsweise *short*, zu übergeben.

Eine der wichtigsten Aufgaben bei der Speicherverwaltung ist der Schutz der Daten der einzelnen Anwendungen. Reservierte Bereiche einzelner Programme dürfen von anderen nicht überschrieben werden. Dazu wird jedem Programm 4 GB virtueller Speicher zugeordnet, wovon 2 GB vom Programmierer genutzt werden können. Dieser virtuelle Speicher hat a priori nichts mit dem physischen Speicher zu tun. Der Inhalt der einzelnen physischen Speicherblöcke ist vom System frei verschiebbar, denn dadurch läßt sich der effektive Schutz der Daten vor Zerstörung ermöglichen.

Da jedes Programm seinen eigenen virtuellen Speicherbereich hat, ist es nun nicht mehr möglich, wie unter DOS oder Windows 3.x, Daten durch festes Programmieren von Speicherbereichen auszutauschen. Unter Windows 95 und Windows NT müssen Daten in die Zwischenablage kopiert werden und die andere Anwendung erhält ein Botschaft (siehe Abschnitt 5.2.6), daß die Daten nun bereitliegen oder sie kontrolliert die Ablage selbständig.

Die Multi-Thread-Programmierung bietet hier wiederum den Vorteil, daß die einzelnen Threads wie eigene Programme bei der CPU-Nutzung behandelt werden, aber alle Threads einer Anwendung auf denselben virtuellen Speicher zugreifen. Damit kann der Datenaustausch innerhalb einer Anwendung bei C-Programmierung durch die Übergabe eines Zeigers auf den Datenbereich realisiert werden.

5.2.5 Instanzierung

Um den physischen Speicher besser ausnutzen zu können, werden von den einzelnen Anwendungen sogenannte Instanzen gebildet.

Beim Start einer Anwendung unter Windows 3.x wurde Programm- und Datenspeicher neu angelegt und zwar unabhängig davon, ob dieses Programm zum zweiten Mal aufgerufen wurden oder nicht. Beim Programmstart unter Windows 95 und Windows NT werden Anwendungen instanziert. Das heißt, beim ersten Start einer Anwendung wird Programm- und Datenspeicher angelegt. Wird dasselbe Programm ein weiteres Mal aufgerufen, so wird ein weitere Instanz gebildet und nur der Datenspeicher, der sich ja vom vorhergehenden unterscheiden kann, neu reserviert. Der Programmspeicherbereich, der für alle Instanzen identisch ist, wird von allen gemeinsam benutzt und ist nur einmal vorhanden. Dieses Prinzip wird auch bei DLL-Dateien (Dynamic Link Libraries) verwendet, die eine Funktionsbibliothek beinhalten und die während der Laufzeit des Programmes eingebunden werden. Benutzen mehrere Programme dieselbe DLL, so befindet sich diese nur einmal im Speicher und die jeweiligen Funktionen können von beiden Anwendungen eingebunden werden.

5.2.6 Kommunikation im System und Ereignisbehandlung

Die Kommunikation der Programme mit dem System und untereinander verläuft über Botschaftsaustausch. Jedem Thread wird eine eigene Botschaftsverarbeitung eingerichtet und jedes Ereignis erzeugt eine Botschaft, die dann in diese Liste eingetragen wird. Erhält der Thread eine Zeitscheibe, so wird die eigene Botschaftsliste kontrolliert und wenn ein Ereignis vorhanden ist, wird dies anhand der zugeordneten Funktion bearbeitet. Wird die Bearbeitung vor Ende der Zeitscheibe erledigt, so wird mit der nächsten Botschaft begonnen oder anderenfalls die Bearbeitung bei der nächsten Zeitscheibe fortgesetzt.

Die eindeutige Zuordnung der Botschaft geschieht über Handles, die in den Parameterlisten der Botschaften übergeben werden. Diese Parameterlisten können auch dazu verwendet werden, um mit derselben Botschaft unterschiedliche Abläufe zu steuern. Handles sind im 32 Bit Betriebssystem 32 Bit Zahlen die jedem Objekt (Speicherbereiche, Gerätekontexte, Fenster, Menüs, etc.) eineindeutig zugeordnet werden und anhand derer sie vom System verwaltet werden können. Dieses Handle-Prinzip ermöglicht auch die im Abschnitt 5.2.4 erwähnte freie Verschiebbarkeit der Speicherbereiche, da diese anhand der Identifikationsnummer immer eindeutig mit einer Anwendung verbunden werden können.

Der Programmierer hat die Möglichkeit für seine Programme eigene Botschaften zu entwerfen. Diese werden beim Programmstart beim System angemeldet und sind dann garantiert einzigartig zur Laufzeit und ermöglichen damit die aufgabenorientierte Kommunikation der Programme oder Threads untereinander. Diese Funktionalität wurde ebenfalls in der Programmierung dieser Anwendung genutzt. Damit lassen sich auf einfache Weise Funktionen in der gekapselten Klassen steuern und das Austauschen solcher Klassen läßt sich durch solche Verfahren ebenfalls vereinfachen. Als Beispiel sei hier die Kommunikation des Hauptprogramms mit den Bitmap-Fensters angeführt. Das Hauptfenster registriert beim System eine Botschaft namens CM_FENSTERNUMMER, die zur Übermittlung des Fensterindex dient. Damit ist die Einzigartigkeit während der Laufzeit sichergestellt. Das Bitmap-Fenster wiederum hat in seiner Antworttabelle dieser Botschaft eine Funktion zugeordnet. Wird das Bitmap-Fenster erzeugt wird ihm anhand der Parameterliste sein Index übergeben und dieser wird dort in eine globale Variable übernommen. Damit kennt das Fenster seinen Index und kann ihn mit entsprechenden anderen Botschaften an das Hauptfenster zurückgeben. Wird etwa ein Bitmap-Fenster vom Benutzer geschlossen, so sendet dieses die Botschaft CM_CLOSEBITMAPWINDOW und in der Parameterliste seinen Index. Damit kann das Hauptfenster, welches in seiner Antworttabelle ein Funktion dieser Botschaft zugeordnet hat, das richtige Fenster von der Oberfläche entfernen und dem Hauptfenster ist stets bekannt welches Fenster in welchem Zustand angezeigt wird. Dies ist exemplarisch für die Kommunikation: Die Klasse wird über eine Botschaft aufgefordert an ihren geschützten Daten und Variablen Änderungen vorzunehmen. Die Daten können nicht direkt von einer übergeordneten Klasse geändert werden und damit ist ein überschaubarer Ablauf gesichert. Soll die Klasse ersetzt werden muß nur in der Antworttabelle ein entsprechende Funktion für die Botschaft vorhanden sein, die die Daten im gleichen Sinne bearbeitet. Wie dies im einzelnen geschieht, ist für alle anderen Klassen, die mit dieser kommunizieren belanglos und darin steckt der große Vorteil der objektorientierten Programmierung.

5.2.7 Hardware-Kommunikation unter Windows NT

Wie bereits erwähnt, kapselt Windows NT die Hardware vollständig ab. Der Zugriff geschieht über sogenannte Kernel Mode Device Drivers. Diese Gerätetreiber sind Software, die die Kommunikation zwischen Betriebssystem und Hardware steuern. Der Treiber verschiebt die Daten zwischen den Peripheriegeräten und dem System. Dazu stellt der Treiber im Fall von Borland C++ 5.0 - Programmierung eine Header-Datei mit Funktionsdefinitionen zur Verfügung, die in das Programm eingebunden werden kann. Diese für den Programmierer verfügbaren Funktionen dienen der Konfiguration und Bedienung der Hardware im ganzen System.

Für die Standardperipherie, wie Bildschirm, Drucker oder Maus, stellt Windows Gerätekontexte bereit, die vom Programmierer genutzt werden können. So geschieht beispielsweise die Bitmap-Darstellung in der eigenen Bitmap-Klasse wie folgt:

1. Im Header des Borland C++ 5.0 Programms wird eine Header-Datei eingebunden, welches die Funktionen zur Gerätebedienung bereitstellt. Programmierern der C-Sprache ist etwa *stdio.h*

ein Begriff, durch das Bildschirm- und Tastaturein- und ausgaben möglich werden. Für die Windows-Anwendung ist dies dann *windows.h*.

2. Es wird ein Zeiger für den Bildschirmgerätekontext erstellt. Anhand dieses Zeigers kann nun der Programmierer verschiedene Funktionen ausführen, die durch den Kontext bereitgestellt werden. In diesem Sinne ist der Gerätekontext eine gekapselte Datenklasse, die der Hardware-Kommunikation dient. So können Eigenschaften der Anzeige abgefragt werden und die Darstellung so dem jeweiligen Bildschirm angepaßt stattfinden.

3. Für das Bitmap wird ein Zeiger auf einen Speicherkontext erstellt, in den das Bitmap von der Festplatte geladen wird. In diesem Kontext können die Bitmap-Eigenschaften durch die Funktionen der Speicherkontextklasse abgefragt werden.

4. Der Bildschirmgerätekontext stellt eine Funktion zur Verfügung, die es erlaubt, die Daten eines Speicherkontextes direkt auf den Bildschirm zu kopieren. Dabei werden der Funktion, die über den Zeiger des Bildschirmgerätekontextes aufgerufen wird, Größe, Position, Art der Darstellung (original, gestreckt oder etwa farbtabelleninvertiert) und eben ein Zeiger auf den Speicherkontext übergeben.

Der Programmierer kann in diesem Sinne weitestgehend hardwareunabhängig programmieren und die Anpassung der Graphik, andere Ausgaben und Hardware-Zugriffe im jeweiligen System geschehen durch den verwendeten Treiber.

Muß wie im vorliegenden Fall, eine spezielle Hardware angesprochen werden, so kann natürlich von Windows kein Gerätekontext zur Verfügung gestellt werden, wie das bei Standardgeräten der Fall ist. Dazu ist der Computer- und Hardware-Markt zu vielfältig und komplex, als das dies mit vernünftigem Aufwand möglich wäre.

Bei der verwendeten NI AT-DIO32F konnte ein aktueller Treiber für Windows NT aus dem Internet geladen werden. Dieser enthält neben der eigentlichen Treiber-Software auch Programme zur Konfiguration der Karte im Windows NT-System, Funktions-Dateien für unterschiedliche Programmiersprachen, Beispielprogramme zu all diesen Sprachen und eine ausführliche Dokumentation.

Der Ablauf ist dann wie folgt:

1. Der Treiber wird im System installiert und die Konfigurations-Software gestartet. In diesem Programm werden Parameter gesetzt, wie DMA-Kanäle und Interrupts, die den Einstellungen auf der Karte entsprechen und der Verwaltung im System dienen. Zusätzlich wird durch das Programm eine Identifikationsnummer für die Karte vergeben über die der Programmierer auf die Karte zugreifen kann. Dieses Programm kann auch dazu verwendet werden mehrere National Instruments Karten mit verschiedenen Treibern in einem System zu verwalten.

2. Zusätzlich zum Standard *windows.h* wird eine weitere Header-Datei eingebunden: *Nidaq.h* (National Instruments Data AQuisition). Dies stellt den Funktionsumfang zur Verfügung.

3. Dem Linker werden zwei weiteren Bibliotheken mit angegeben: *nidex32b.lib*, eine Bibliothek zur Fehlerbehandlung bei 32 Bit Zugriff, und *nidaq32b.lib* für die 32 Bit Input-/Output-Funktionen. Beide Bibliotheken waren im Treiberpaket mit enthalten.

4. Nun geschieht der Hardware-Zugriff nicht über einen Gerätekontextzeiger sondern direkt über die Header-Datei-Funktionen anhand der von der Konfigurations-Software vergebenen Identifikationsnummer der Karte.

Mit diesen Funktionen, die alle die Identifikationsnummer der Karte benötigen, lassen sich dann alle nötigen Einstellungen treffen. Die Karte wird für die jeweilige Aufgabe als Input- oder Output-Device umprogrammiert. Die Handshake-Parameter werden angegeben (ansteigende Flanke und invertierte Polarität) und der Datenpuffer bereitgestellt. Die Daten werden dann über Funktionen eingelesen, die die oben genannten Parameter benötigen und als Ergebnis die eingelesenen Daten in den konfigurierten Puffer schreiben.

Auf die Effizienz und Geschwindigkeit dieses Standardtreibers wird im abschließenden Kapitel 7 noch ausführlicher eingegangen.

5.2.8 Aufbau eines Windows-Programmes

Der Aufbau des Windows-Systems spiegelt sich im Programmaufbau wieder. Der Programmierer ist dafür verantwortlich, daß sein Programm ein korrekte Schnittstelle zum Windows-Manager bietet und damit die Anwendung korrekt verwaltet und ausgeführt werden kann.

Der Aufbau sieht wie folgt aus:

1. Eintrittsfunktion: WinMain() oder OwlMain(). Dieser Funktion werden vom Windows-Manager eine Reihe von Argumenten übergeben. Als erstes erhält jede Instanz ein Handle, anhand dessen sie eindeutig zu identifizieren ist. Das zweite Argument ist ein Handle auf eine mögliche vorherige Instanz. Dann kommen noch eventuelle Kommandozeilen-Argumente und Parameter zur Darstellung des Hauptfensters.

2. Fensterklasse: Innerhalb der Eintrittsfunktion wird die Fensterklasse der Anwendung konfiguriert und registriert. Dabei wird festgelegt, wie das Fenster, das Menü und die Symbole aussehen.

3. Instanz: Vom Hauptfenster wird nun eine Instanz erzeugt. Dies ist die Schnittstelle von der Programminstanz zum Benutzer, denn erst über dieses sichtbare Fenster kann der Benutzer Einfluß auf das Programm nehmen.

4. Message-Loop: Nach der Registrierung der Instanz und der Erzeugung einer sichtbaren Schnittstelle wartet das Programme auf Botschaften des Systems. Dazu durchläuft es fort-während eine Schleife, den sog. Message-Loop. Findet im Fenster einer Anwendung eine Aktion statt, beispielsweise das Anklicken eines Menüpunktes, dann erhält das Fenster eine Botschaft und den Fokus (wird aktiv).

5. Response-Table: Jedes Programm muß eine Antworttabelle haben, in der die aufzurufenden Funktionen als Reaktion zu den entsprechenden Botschaften verzeichnet sind.

Weitere Ausführungen und detaillierte Überlegungen zur Windows-Struktur und Programmierung finden sich etwa in Louis (1996a) oder Petzold (1996).

5.3 Implementierung mit Borland C++ 5.0

Borland C++ 5.0 stellt eine umfangreiche Funktionsbibliothek zur Verfügung. Diese sind in der sogenannten Object Windows Library (OWL) zusammengefaßt. Im folgenden wird eine Zusammenfassung der prinzipiellen Funktionsweise gegeben und dann auf den tatsächlichen Programmaufbau und die Hardware-Kommunikation eingegangen.

5.3.1 Windows API

Diese Abkürzung steht für Application Programming Interface. Diese Bibliothek stellt die grundlegenden Funktionen zur Programmierung von Windows-Anwendungen zur Verfügung. Die Funktionen sind rudimentär und enthalten komplizierte Abfolgen von Befehlen und Parametern, um entsprechende graphische Elemente zu erhalten. Ein direkt mit dem Windows-API erstelltes Programm muß auch alle Standard Windows Botschaften in entsprechenden Antworttabellen verarbeiten. Die direkte Verwendung der API-Funktionen ist im allgemeinen nicht zu empfehlen, da die Komplexität der Befehle und Abfolgen gerne zu Fehlern führt, die schwer zu finden sind oder versteckte Folgen haben, die sich nicht direkt erkennen lassen.

5.3.2 Kapselung und Vererbung

C++ bietet die Möglichkeit objektorientiert zu programmieren. Dies bedeutet, daß die Daten in Klassen geschützt verarbeitet werden und diese Klassen die Bearbeitungsroutinen mit zur Verfügung stellen. Dazu werden die Daten und Funktionen in folgende Arten unterteilt:

1. public: Dieses Element (Variable oder Funktion) kann von jeder übergeordneten Klasse direkt benutzt werden. Hier befinden sich die Funktionen zur Datenverarbeitung.

2. protected: Hier befinden sich die Funktionen und Daten, die nur von klasseneigenen Funktionen benutzt werden können.

Eine neue Klasse kann die als public deklarierten Funktionen einer anderen Klasse erben. Diese geerbten Funktionen werden in der neuen Klasse zu protected Funktionen, können aber entsprechend erweitert werden (überschrieben oder überladen). Überschreiben bedeutet, daß eine Funktion mit gleichem Namen und gleicher Parameterliste in der abgeleiteten Klasse deklariert und neu definiert wird und an die Stelle der geerbten tritt. Bei überladenen Funktionen gibt es mehrere Funktionen gleichen Namens, aber mit unterschiedlichen Parameterlisten. Der Compiler wählt dann anhand der Parameterliste bei der Programmerstellung die jeweils korrekte aus.

Bei dieser Datenkapselung besteht nicht mehr die Möglichkeit direkt auf die Daten zuzugreifen, sondern es werden die von der Klasse mitgelieferten Funktionen verwendet. Damit liefern Klassen eine definierte Schnittstelle zur Kommunikation mit anderen Programmteilen und richtig implementierte Klassen vermindern den Programmieraufwand und die möglichen Fehler bei zukünftigen Projekten.

5.3.3 Object Windows Library

Die Object-Windows-Library (OWL) kapselt Windows-API-Funktionen zu neuen Datenklassen. Diese erleichtern das Programmieren wesentlich und übernehmen viele rudimentären Arbeiten die zur Verwaltung der Klassen notwendig sind. Sie bieten ein Vielzahl von Funktionen zur Datenverarbeitung, ohne aber direkten Zugriff auf die Daten zu erlauben. Damit wird die Fehlerquote erheblich reduziert, allerdings ist dieser indirekte Zugriff in manchen Fällen zu zeitintensiv. Dies ist beispielsweise bei der Graphikdarstellung der Fall. Die Bitmap-Klasse der OWL stellte viele Bearbeitungsfunktionen bereit, aber der Zugriff auf die Daten ist nur über diese möglich. Unter Windows verwenden nur Bitmaps mit einer Farbtiefe von 8 Bit oder weniger eine Farbtabelle. 16 oder 24 Bit Bitmaps arbeiten mit RGB-Werten (Rot, Grün, Blau). Bei RGB-Werten ist die Farbumsetzung der 16 Bit Daten a priori nicht gegeben, da die Umrechnung in drei unabhängige Werte von 0 bis 255 geschehen muß. Um auf diese Weise Farbverläufe zu erstellen, die dem Auge auch als solche erscheinen, ist ein erheblicher Rechenaufwand nötig, der die Darstellung verlangsamt. Für die Umsetzung der Daten mußte eine andere Lösung gefunden werden.

Werden 8 Bit Bitmaps mit Standardklassen der OWL erzeugt, so wird die Systemfarbtabelle des Windows-Betriebssystems verwendet. Diese ist 8 Bit tief, enthält aber keinen Farbverlauf, da dort auch die reservierten Standardfarben etwa für Fensterleisten (blau) oder Warnungen (rot) enthalten sind. Es besteht die Möglichkeit über OWL-Funktionen die Graphikfähigkeiten des Systems abzufragen. Dies geschieht über die Funktion GetDeviceCaps (Get Device Capabilities). Neben der Größe des Bildschirms in Pixeln und anderen Eigenschaften, läßt sich anhand der Systemvariable *SIZEPALETTE* die Anzahl der Farben in der Systempalette erfahren. Ist die Anzeige in der Lage mehr

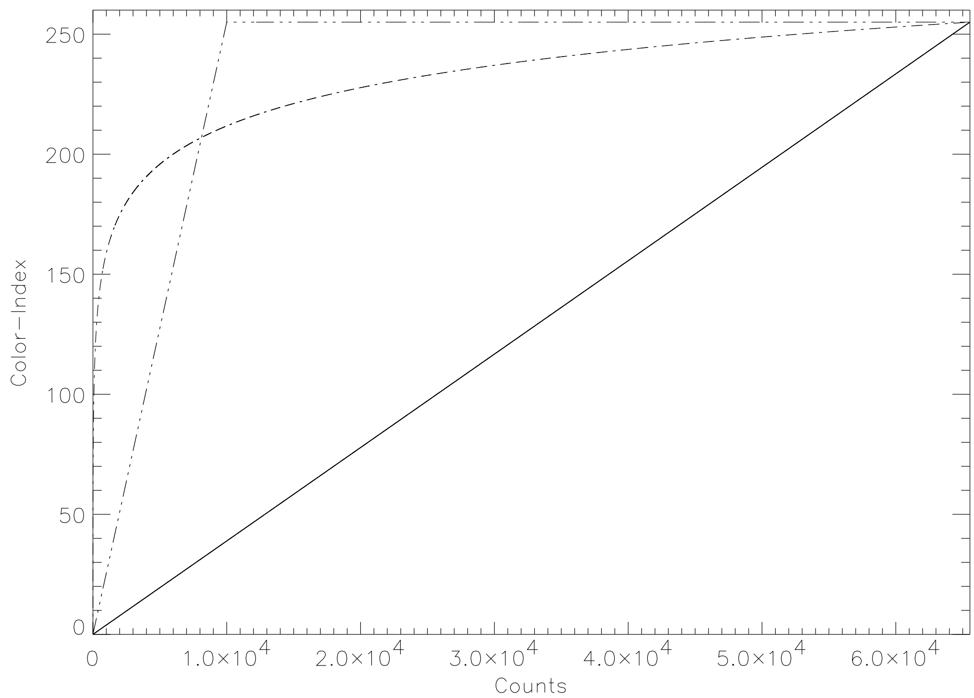

Abbildung 5.1: *Lookup-Tables für die Datenumsetzung: Im Bild sind drei mögliche Umsetzungen der* 16 *Bit Datenwerte in* 8 *Bit Farbwerte angezeigt. Es wurden folgenden Symbole verwendet: — Lineare Umsetzung, -·- Logarithmische Umsetzung und -· · · Lineare Umsetzung mit Maximalwertorientierung bei* 10000 *Counts. Die* 8 *Bit Farbwerte werden dann im Bitmap beispielsweise in eine Schwarz-Weiß-Farbtabelle (RGB-Werte von* 0, 0, 0 *bis* 255, 255, 255*) umgesetzt.*

als 8 Bit Farbtiefe anzuzeigen, so wird dieser Wert Null und es besteht nicht die Möglichkeit auf diese Farbtabelle zuzugreifen. Die Systemvariable *NUMRESERVED* enthält die Anzahl der reservierten Systemfarben (meist 10), die sich auch bei Zugriff auf die Farbtabelle nicht ändern lassen. Das Windows-System erlaubt also nur eine Änderung der Farbtabelle, wenn die Anzahl der dargestellten Farben des Bildschirms oder der Graphikkarte 8 Bit oder geringer ist und dann auch nur der nicht reservierten Farben. Darüberhinaus betrifft die Änderung der Systemfarbtabelle alle anderen Anwendungen im Multi-Tasking, ebenfalls ein nicht erwünschter Effekt. Diese Einschränkungen, die durch die OWL-Standardklassen gegeben sind, sind zu gravierend.

Aus diesen Gründen wurde für die schnelle Darstellung des integrierten Bitmaps auf die gekapselte OWL-Funktion verzichtet und eine rudimentäre Windows-API Funktion verwendet. Es wurde ein eigenes 1024×1024 Punkte großes Array mit 8 Bit Tiefe für das Bitmap eingerichtet. Die integrierten 16 Bit Datenwerte werden dann anhand einer Umsetzungstabelle (Lookup-Table, siehe Abb. 5.1) auf 8 Bit Farbwerte kopiert. Dies erlaubt auch die Verwendung unterschiedlicher Umsetzungen, wie etwa einer linearen oder logarithmischen Tabelle mit optionaler Maximalwertorientierung. Die Größeninformationen des Bitmaps wurden eigenständig berechnet und die 8 Bit Farbtabelle zur Umsetzung der Farbindizes in tatsächliche RGB-Werte wurde speziell erstellt. Dies ermöglicht auch eine menügesteuerte Verwendung verschiedener Farbverläufe, wie etwa Schwarz-Weiß oder Gelb-Rot. Das so erstellte Bitmap wird dann mit der entsprechenden Windows-API Funktion direkt

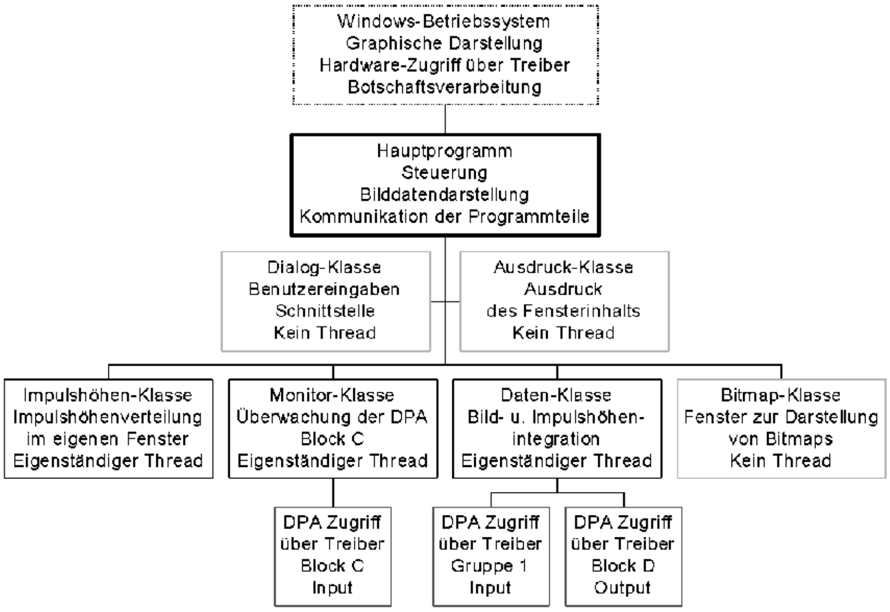

Abbildung 5.2: *Schematischer Aufbau des Datenerfassungsprogramms: In der Darstellung sind die einzelnen Datenklassen und die Punkte des eigentlichen Hardware-Zugriffs gekennzeichnet. Außerdem läßt sich die Thread-Struktur deutlich erkennen.*

in den Bereich des Fensters kopiert. Auf diese Weise konnte sowohl der Benutzerfreundlichkeit als auch der Geschwindigkeit am besten Rechnung getragen werden.

5.4 Programmaufbau

Die in den vorangegangenen Abschnitten erläuterten Eigenschaften des Windows NT-Betriebssystems und der Borland C++ 5.0 -Programmiersprache gehen nun direkt in die Struktur des Programms ein. Der Programmaufbau ist in Abb. 5.2 schematisch dargestellt. In dieser Graphik läßt sich der Aufbau mit gekapselten Klassen gut erkennen. Das Hauptprogramm (Hauptthread) übernimmt die Schnittstelle mit dem Betriebssystem und steuert die anderen Klassen. Diese Klassen sind gekapselt und stellen wiederum definierte Schnittstellen zur Verfügung. Diese Schnittstellen sind entweder als *public* definierte Funktionen der untergeordneten Klasse oder eigene Botschaften. Die *public*-Funktionen können von der übergeordneten Klasse direkt aufgerufen werden, während die Botschaften über die Antworttabellen der untergeordneten Klassen *protected*-Funktionen zugewiesen sind. Dadurch können diese Klassen auch jederzeit in anderen Anwendungen verwendet werden und müssen nicht in jedem Programm aufs neue erstellt werden.

Der Hardware-Zugriff findet nur in zwei Klassen (Daten-Klasse und Monitor-Klasse) statt. Die Monitor-Klasse dient im wesentlichen nur der Überwachung des DPA und liest dazu die Funktionsleitungen aus. Diese sind alle auf einem Port der NI AT-DIO32F konfiguriert und enthalten Informationen über den eingestellten Modus des DPA und die ‚Integration-Stop'-Taste. Die Informationen über den Modus werden dann vom Hauptprogramm dazu verwendet, die entsprechenden Integrationsmöglichkeiten im Menü zu steuern. So ist sichergestellt, daß vom Benutzer im Programm keine Modi gewählt werden, die nicht mit dem DPA-Modus vereinbar sind. Die Daten-Klasse setzt die ‚Computer Aktiv'-Statusleitung um den Transfer zu initialisieren und liest dann die Datenleitungen aus. Die Demultiplex-Operationen der Daten und die Verwaltung der Daten-Arrays findet ebenfalls hier statt, um einen schnellen Zugriff während der Integration zu gewährleisten. Durch die Beschränkung des Hardware-Zugriffs auf nur zwei Klassen und die Kapselung dieser Einheiten ist es möglich, mit wenig Aufwand gegebenenfalls eine neue digitale Schnittstellenkarte zu verwenden. In diesem Fall werden die Klassen mit dem NI AT-DIO32F spezifischen Hardware-Zugriff durch neue ersetzt, die dann nur die gleichen Schnittstellen (Steuerungsbotschaften und *public*-Funktionsnamen) zum Hauptprogramm aufweisen müssen. Weitere NI AT-DIO32F-spezifische Einstellungen sind im Hauptprogramm nicht vorhanden. Auf diese Weise lassen sich neue Hardware-Komponenten mit minimalem Aufwand einführen.

Es gibt auch zwei Klassen, die von jeder anderen Klasse verwendet werden können. Dies sind die Dialog- und Ausdruck-Klassen. Diese werden jeweils in einer eigenen Header-Datei zur Verfügung gestellt und erzeugen einen Eingabedialog für die Meßdaten bzw. bieten eine Menüfunktion zum Ausdruck des Fensterinhaltes (Bitmap, integriertes Bild oder Impulshöhen-Verteilung) über den bekannten Windows-Druckerdialog. Die Dialog-Klasse für die Eingabe von Meßkommentaren ist spezifisch auf den optischen Detektor zugeschnitten, aber die Ausdruck-Klasse eignet sich auch für jede zukünftige Windows-Anwendung.

Die Abb. 5.2 zeigt auch die Thread-Struktur des Programmes. Die zeitintensiven Routinen sind als eigene Threads programmiert. Die Monitor-Klasse ist als Thread in der Lage, innerhalb möglichst kurzer Zeit auf den Abbruch der Messung durch Drücken der ‚Integration-Stop'-Taste am DPA zu reagieren. Die Thread-Implementierung bedeutet ja Zuweisung von CPU-Rechenzeit speziell für diese Aufgabe und ohne diese Zuweisung wäre es erfolglos während der Berechnung des neuen Bitmaps der Impulshöhen-Verteilung oder des integrierten Bildes die Messung über den DPA zu beenden. Das Einlesen der Daten geschieht ebenfalls im Thread mit Zuweisung einer erhöhten Priorität während der Integration. Dadurch werden diesem Thread mehr Zeitscheiben zugeordnet und eine schnelle Bearbeitung der Daten sichergestellt. Die Berechnung der Impulshöhen-Darstellung wurde ebenfalls abgespalten und der Hauptthread, der nur die Darstellung des integrierten Bildes übernimmt, bleibt während der Messung reaktionsfähig und erlaubt beispielsweise auch die Beendigung der Integration durch Anklicken des entsprechenden Menüpunktes. Ohne die Thread-Struktur des Programmes wäre es nur mit sehr großer Zeitverzögerung möglich die Messung per Mausklick zu beenden.

Die Bitmap-Klasse, die der Darstellung von Bitmaps in einem eigenen Fenster dient, wurde nicht als Thread implementiert. Während der Integration von Daten ist die Klasse für den Benutzer nicht verfügbar, um die Rechenleistung auf den Hardware-Zugriff zu konzentrieren. Wird sie dann dargestellt, geschieht dies nicht in einer zeitkritischen Phase des Programmablaufs.

Anhand verschiedener Implementierungsversuche hat sich dieser Aufbau als sehr geschickt erwiesen und bietet die meisten Vorteile was Geschwindigkeit und Bedienungsfreundlichkeit anbetrifft. Mit dieser Struktur wurde auch der Grundstein für weitere Entwicklungen des Datenverarbeitungssystems hinsichtlich neuer Hardware und neuen Verarbeitungsaufgaben gelegt. Ausführliche Beschreibungen der Programmfunktionen und Datenklassen finden sich im Anhang unter C und D.

Kapitel 6

Vergleichsmessungen

Das Institut für Astronomie und Astrophysik Tübingen, Abteilung Astronomie, besitzt zur Zeit drei MCP-Detektoren, von denen zwei betriebsbereit in Gehäuse eingebaut sind. Es handelt sich bei allen Detektoren um Prototypen der Firma Proxitronic (Koschützke, 1999) mit kreisförmigem Bilddurchmesser von 25 mm, die drei MCP in Chevron-Anordnung beinhalten. Die Detektoren enthalten eine Bialkali-Kathode mit hoher Blauempfindlichkeit und geringem Dunkelstrom und als Auslesesystem eine Hochohm-Keramik mit äußerer Keil-Streifen-Anode.

Um den Stand der Detektoren, deren Empfindlichkeit und Dunkelströme, sowie die Leistungsfähigkeit der Erfassungssysteme zu dokumentieren wurden Vergleichsmessungen der beiden Detektoren und Dynamikmessungen der Elektronik und der Erfassungssysteme durchgeführt. Dazu wurde die Testanordnung von Diesch (1988) verwendet, die in Abschnitt 6.1 beschrieben und in Abb. 3.2 zu sehen ist. Bereits während der Messungen für diese Arbeit ließen sich Veränderungen an den Detektoren erkennen. An Detektor #3971 traten vermehrt Cold-Spots, Stellen an denen keine Detektorempfindlichkeit mehr vorhanden ist, auf. Aus diesem Grund wurden die gemessenen Daten zu Vergleichszwecken im Anhang A aufgeführt und werden im folgenden dargestellt.

Alle Bildaufnahmen mit dem PC-System sind in 8 Bit Farbtiefe mit einer Weiß-Schwarz-Farbtabelle und Maximalwertorientierung dargestellt (Negativ-Darstellung des Daten-Bitmaps).

6.1 Testaufbau

In Abb. 3.2 ist der Testaufbau dargestellt. Im Vordergrund ist der Detektor mit seiner Hochspannungsversorgung (rechts) und der Vorverstärker-Elektronik für die vier Signale (in der Box links) zu sehen. An das Gehäuse mit Shutter ist ein Tubus lichtdicht angebracht. In der Mitte dieses Tubus befindet sich eine Ringblende um Reflexionen an der Tubusinnenwand zu minimieren. Am Ende ist die in der Diplomarbeit von Diesch (1988) entwickelte LED-Anordnung zur Aufnahme von reproduzierbaren Flatfield-Bildern zu sehen. Diese ist ebenfalls lichtdicht angebracht und enthält Leuchtdioden in verschiedenen optischen und infraroten Wellenlängen. Dieser Aufbau kann auch zur zuverlässigen Bestimmung der relativen Empfindlichkeit des Detektors in Abhängigkeit vom registrierten Auftreffort verwendet werden. Die technischen Daten sind im Anhang A.1.1 aufgeführt.

Um die Messungen vergleichbar zu halten wurde jeweils die Vorverstärkerelektronik, die einer kleinen Box zusammengefaßt ist, ausgetauscht und immer dieselbe verwendet.

Im Rahmen dieser Arbeit wurde die Anordnung, wie sie im Bild zu sehen ist, dazu verwendet, Vergleichsmessungen der Datenerfassungssysteme, Empfindlichkeitsmessungen und Eichkurven der Detektoren zu erstellen.

.

6.2 Vergleich der Detektoren #3971 und #4351

In Tab. 6.1 sind die Messungen zusammengefaßt, die auf den folgenden Seiten zu sehen sind. Alle diese Messungen wurden mit dem PC-System und der neueren DPA-Version durchgeführt. Zuerst wurden die Dynamik-Messungen (siehe Abschnitt 6.4) durchgeführt. Dort wurde aufgrund der Detektorempfindlichkeit die LED LB 5410 Anschluß Blau benutzt. Um vergleichbare Ergebnisse zu erhalten, wurde bei allen anderen Messungen (außer den Empfindlichkeitsmessungen der Detektoren #3971 und #4351, siehe Abschnitt 6.3) ebenfalls die LED LB 5410 Anschluß Blau benutzt.

Datum	Detektor	Art	Meßdauer		Counts	Impulshöhe	FWHM	LED LB 5410
			System [s]	DPA [s]	Gesamt	Max. [V]	[V]	Strom [mA]
17.06.99	#3971	Flatfield	3621,52	2234,008	20.687.667	–	–	0,36
27.07.99	#3971	Flatfield	1742,90	1018,147	9.712.743	1,95	3,48	0,38
27.07.99	#4351	Flatfield	1707,11	1000,330	9.336.504	2,46	2,42	0,35
17.08.99	#3971	Dunkelstrom	5779,21	5000,375	8.979.438	1,17	3,04	0,00
17.08.99	#4351	Dunkelstrom	5144,22	5000,549	1.386.681	0,62	–	0,00

Tabelle 6.1: Daten der PC-Messungen

Flatfield-Aufnahmen

Abb. 6.1 und Abb. 6.2 zeigen Flatfield-Aufnahmen vom 17. Juni 1999 und 27. Juli 1999. Beide Aufnahmen wurden mit dem Detektor #3971, der zur Zeit in Gehäuse #2 eingebaut ist, durchgeführt. In den Bildern sind deutlich sogenannte Cold-Spots zu sehen, an denen keine Detektorempfindlichkeit mehr vorhanden ist. Außerdem ist deutlich zu erkennen, daß innerhalb der kurzen Zeit mehrere Cold-Spots hinzugekommen sind. Bei genauerer Betrachtung zeigt sich auch ein Kreuz, daß horizontal und vertikal durch das Bild läuft. Dies ist auf Rundungsfehler bei der Koordinatenberechnung durch den DPA zurückzuführen.

In Abb. 6.3 ist eine Flatfield-Aufnahme vom 27. Juli 1999 mit Detektor #4351 in Gehäuse #1 zu sehen. In diesem Bild ist im Vergleich zu den Flatfields von Detektor #3971 (Abb. 6.1 und Abb. 6.2) ein gegenteiliger Effekt zu erkennen. Im linken oberen Bildbereich befindet sich ein Hot-Spot an dem ständige Aktivität herrscht. Diese Aktivität ist in diesem Fall auf die Mikrokanal-Platte zurückzuführen. Auf der Anodenstruktur der Keil-Streifen-Anode kann dieser Effekt nicht auftreten und Messungen haben gezeigt, daß der Hot-Spot auch mit abgeschalteter Kathode vorhanden ist.

Die Flatfield-Aufnahmen vom 27. Juli 1999 (Abb. 6.2 und Abb. 6.3) wurden so durchgeführt, daß bei den angelegten Strömen von $0,38$ mA und $0,35$ mA jeweils etwa 10000 Counts s^{-1} vom DPA registriert wurden. Die Meßdauern betragen jeweils etwa 1000 s und die Zählraten sind vergleichbar.

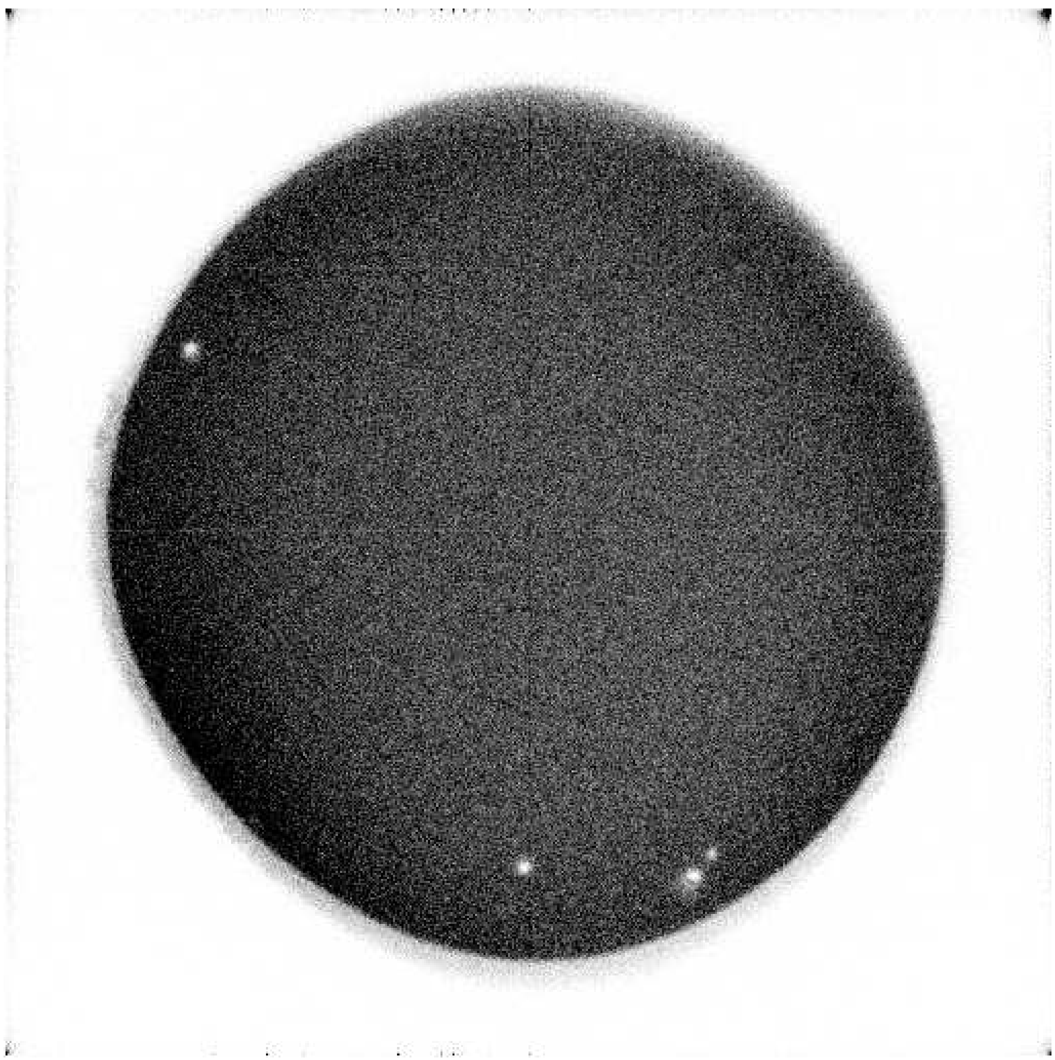

Abbildung 6.1: *Flatfield-Messung des Detektor* #3971 *vom* 17. *Juni* 1999*: Das Bild wurde mit LED LB* 5410 *Anschluß Blau und einem Strom von 0,36 mA aufgenommen. Die Meßdauer betrug* 3621, 52 *s (Windows-Systemuhr) bzw.* 2234, 008 *s (totzeitkorrigiert). Es sind* 20.687.667 *Counts im Bild enthalten. Die Darstellung des Bildes ist in* 8 *Bit Farbtiefe mit einer Weiß-Schwarz-Farbtabelle und Maximalwertorientierung (Negativ-Darstellung des Daten-Bitmaps).*

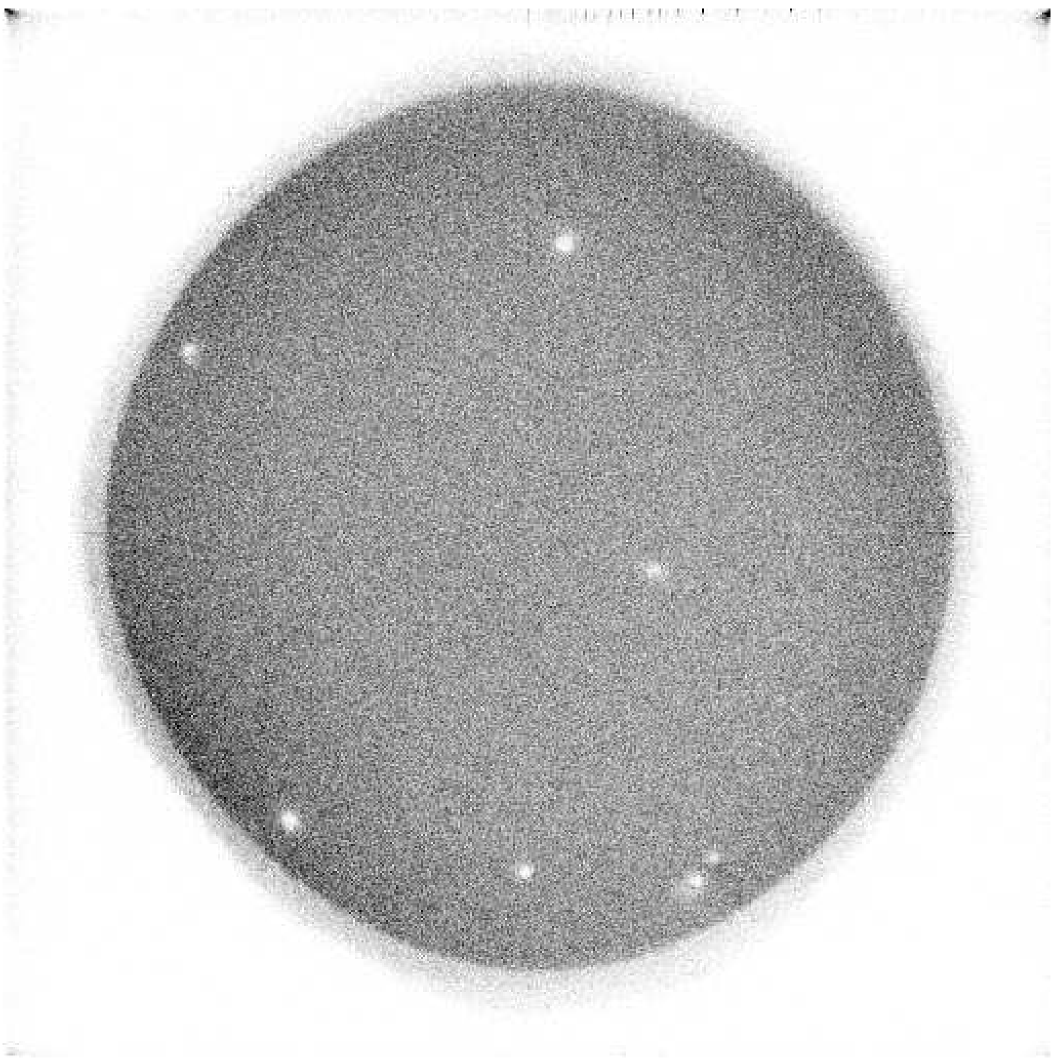

Abbildung 6.2: *Flatfield-Messung des Detektor #*3971 *vom* 27. *Juli* 1999*: Das Bild wurde mit LED LB* 5410 *Anschluß Blau und einem Strom von* 0, 38 *mA aufgenommen. Die Meßdauer betrug* 1742, 90 *s (Windows-Systemuhr) bzw.* 1018, 147 *s (totzeitkorrigiert). Es sind* 9.712.743 *Counts im Bild enthalten. Die Darstellung des Bildes ist in* 8 *Bit Farbtiefe mit einer Weiß-Schwarz-Farbtabelle und Maximalwertorientierung (Negativ-Darstellung des Daten-Bitmaps).*

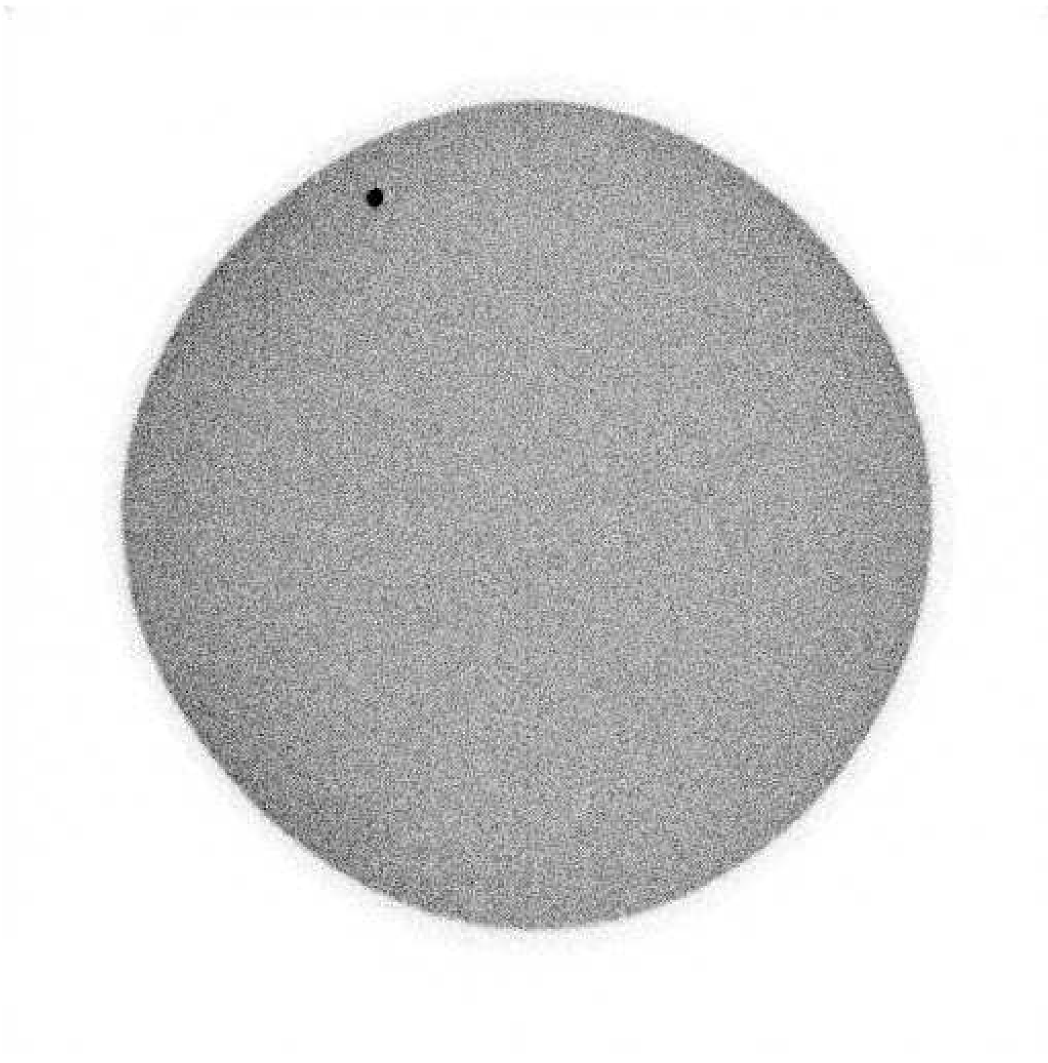

Abbildung 6.3: *Flatfield-Messung des Detektor #4351 vom 27. Juli 1999: Das Bild wurde mit LED LB 5410 Anschluß Blau und einem Strom von 0,35 mA aufgenommen. Die Meßdauer betrug 1707,11 s (Windows-Systemuhr) bzw. 1000,330 s (totzeitkorrigiert). Es sind 9.336.504 Counts im Bild enthalten. Die Darstellung des Bildes ist in 8 Bit Farbtiefe mit einer Weiß-Schwarz-Farbtabelle und Maximalwertorientierung (Negativ-Darstellung des Daten-Bitmaps).*

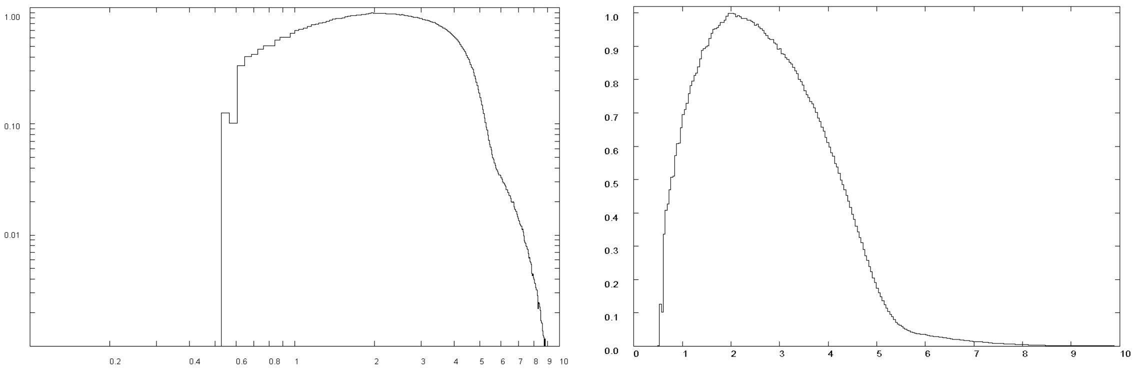

Abbildung 6.4: *Gemessene Impulshöhen-Verteilung des Detektors* #3971*: In den Bildern ist die logarithmische und lineare Impulshöhen-Verteilung der Flatfield-Aufnahme vom* 27*. Juli* 1999 *zu sehen. Das Maximum liegt bei* $1,95$ *V und die Halbwertsbreite der Verteilung bei* $3,48$ *V (*178%*).*

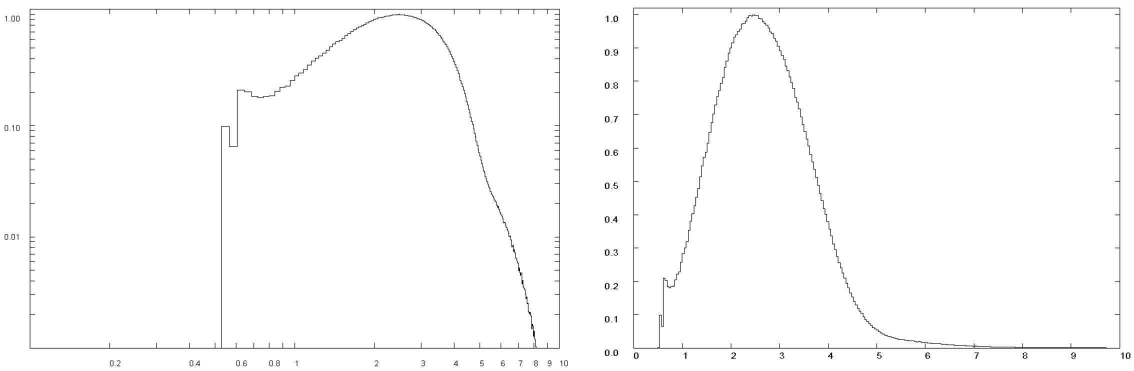

Abbildung 6.5: *Gemessene Impulshöhen-Verteilung des Detektors* #4351*: In den Bildern ist die logarithmische und lineare Impulshöhen-Verteilung der Flatfield-Aufnahme vom* 27*. Juli* 1999 *zu sehen. Das Maximum liegt bei* $2,46$ *V und die Halbwertsbreite der Verteilung bei* $2,42$ *V (*101%*).*

Die Impulshöhen-Verteilungen in Abb. 6.4 und Abb. 6.5 wurden parallel zu den Flatfieldaufnahmen am 27. Juli 1999 gemacht und enthalten dementsprechend gleich viele Impulse. Die Maximalwerte liegen bei $1,95$ V mit einer Halbwertsbreite von $3,48$ V (178%) für Detektor #3971 und bei $2,46$ V mit einer Halbwertsbreite von $2,42$ V (102%) für Detektor #4351.

Ist die Hochspannung hoch genug, so bewirkt die Raumladung der Elektronenwolke am Ende des Kanals eine Begrenzung der Verstärkung. Die von der Mikrokanal-Platte erzeugten Wolken haben somit etwa die gleiche Ladungsanzahl, was durch eine Gaußsche Verteilung wiedergegeben wird. Die Höhe der Verstärkung der Mikrokanal-Platte kann an der Lage des Maximums abgelesen werden. Die Lage der Maxima sollte zwischen 2 und 5 V liegen, um die Zahl der verwertbaren Impulse möglichst groß zu halten. Liegt das Maximum außerhalb dieser Wert, so ergeben sich für die Randbereiche Probleme und zwar insofern als daß entweder sehr kleine Divisoren bei der Koordinatenbestimmung auftreten, oder daß Signale unterdrückt werden, da die AD-Wandler Eingangsspannung auf 10 V beschränkt ist. Die Gaußsche Verteilungsform ist bei Detektor #4351 deutlich zu erkennen. Detektor #3971 zeigt eine verbreiterte Impulsverteilung und liefert dem entsprechend keine so gute Ortsauflösung wie Detektor #4351. Weitere Überlegungen dazu finden sich in Barnstedt (1985) und

Gutekunst (1986).

Beide Detektoren liegen innerhalb des gewünschten Maximalwertbereiches, aber der Detektor #4351 zeigt eine geringere Halbwertsbreite und damit eine gleichmäßigere Verstärkung. Der Detektor #4351 wäre trotz des Hot-Spots bei Messungen dem Detektor #3971, der ja auch einige Cold-Spots zeigt, vorzuziehen.

Dunkelstrom-Aufnahmen

Die Dunkelstrom-Aufnahmen (Abb. 6.6 und Abb. 6.7) vom 17. August 1999 wurden jeweils bei geöffnetem Shutter ohne LED-Strom durchgeführt, da sich bei den anfänglichen Versuchen gezeigt hat, daß bei geschlossenem Shutter die Dunkelstromraten steigen. Dies bedeutet, daß im Detektor schwache Lichtquellen vorhanden sind, deren Photonen am geschlossenen Shutter reflektiert werden. Das Licht ist vermutlich durch Kriechströme oder Feldemission induziertes Fluoreszenzlicht von der Detektor-Keramik.

Die Meßdauer betrug jeweils etwa 5000 Sekunden. Detektor #3971 zeigt eine deutliche höhere Dunkelrate (8.979.438 Counts) gegenüber #4351 (1.386.681 Counts). In Detektor #4351 konzentriert sich der Dunkelstrom auf einen Hot-Spot, der schon in den anderen Messungen deutlich zu sehen war. Dort sind im Maximumpixel in der Meßzeit 5700 Counts registriert worden, während im gesamten Bild nur durchschnittlich 1 Count Pixel^{-1} vorhanden ist. Die durchschnittlichen Dunkelstromraten betragen bei Detektor #3971 $1,71 \cdot 10^{-3}$ Counts Pixel^{-1} s^{-1} und bei Detektor #4351 $2,64 \cdot 10^{-4}$ Counts Pixel^{-1} s^{-1}

Wie bei den Flatfield-Aufnahmen, zeigt sich auch im Dunkelstrombild das Kreuz in der Bildmitte, das durch Rundungsfehler bei der Koordinatenberechnung aufgrund von kleinen Divisoren auftritt. Auch die Cold- und Hot-Spots der einzelnen Detektoren sind im Dunkelstrombild deutlich zu erkennen.

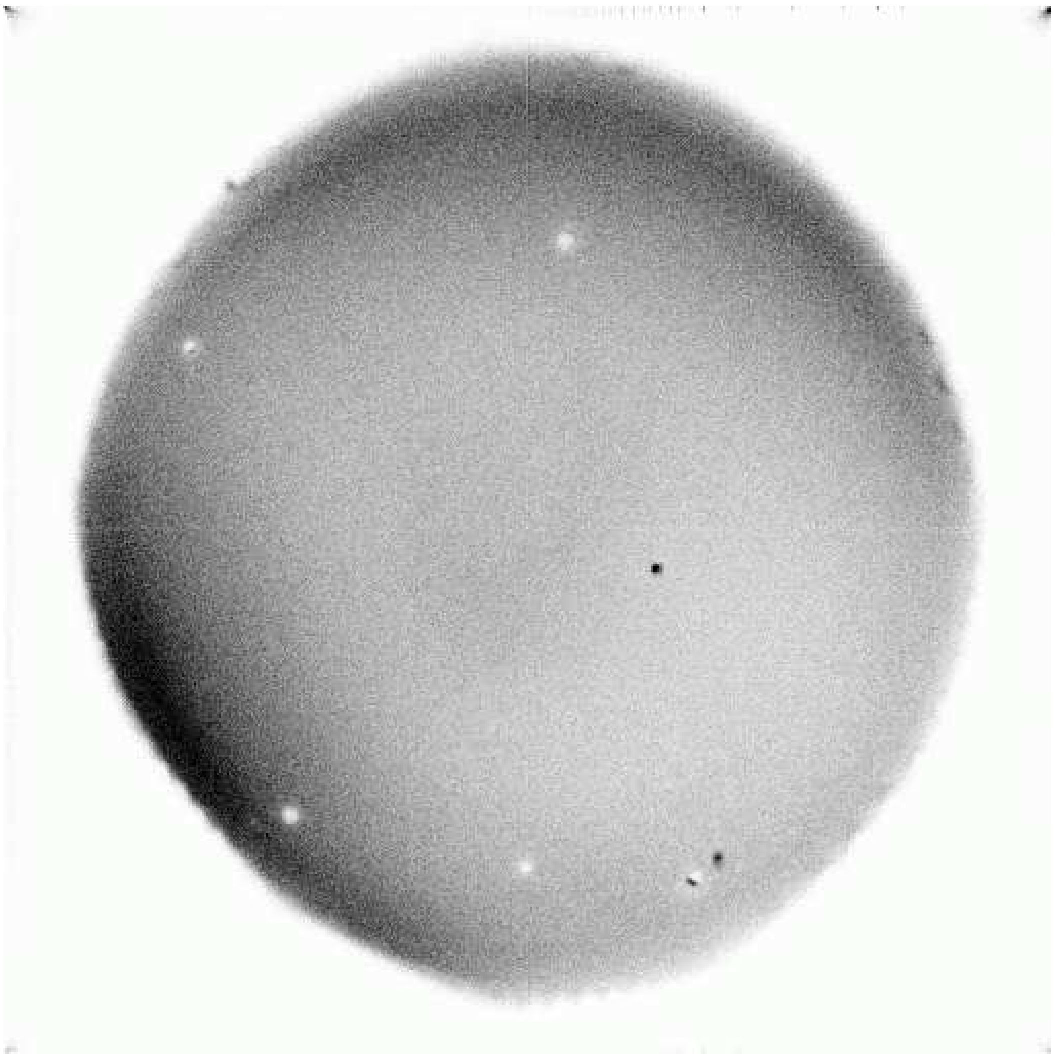

Abbildung 6.6: *Dunkelstrom-Messung des Detektors* #3971 *vom* 17. *August* 1999*: Das Bild wurde ohne LED-Strom mit geöffnetem Shutter aufgenommen. Die Meßdauer betrug* 5779, 21 *s (Windows-Systemuhr) bzw.* 5000, 375 *s (totzeitkorrigiert). Es sind* 8.979.438 *Counts im Bild enthalten. Die Darstellung des Bildes ist in* 8 *Bit Farbtiefe mit einer Weiß-Schwarz-Farbtabelle und Maximalwertorientierung (Negativ-Darstellung des Daten-Bitmaps).*

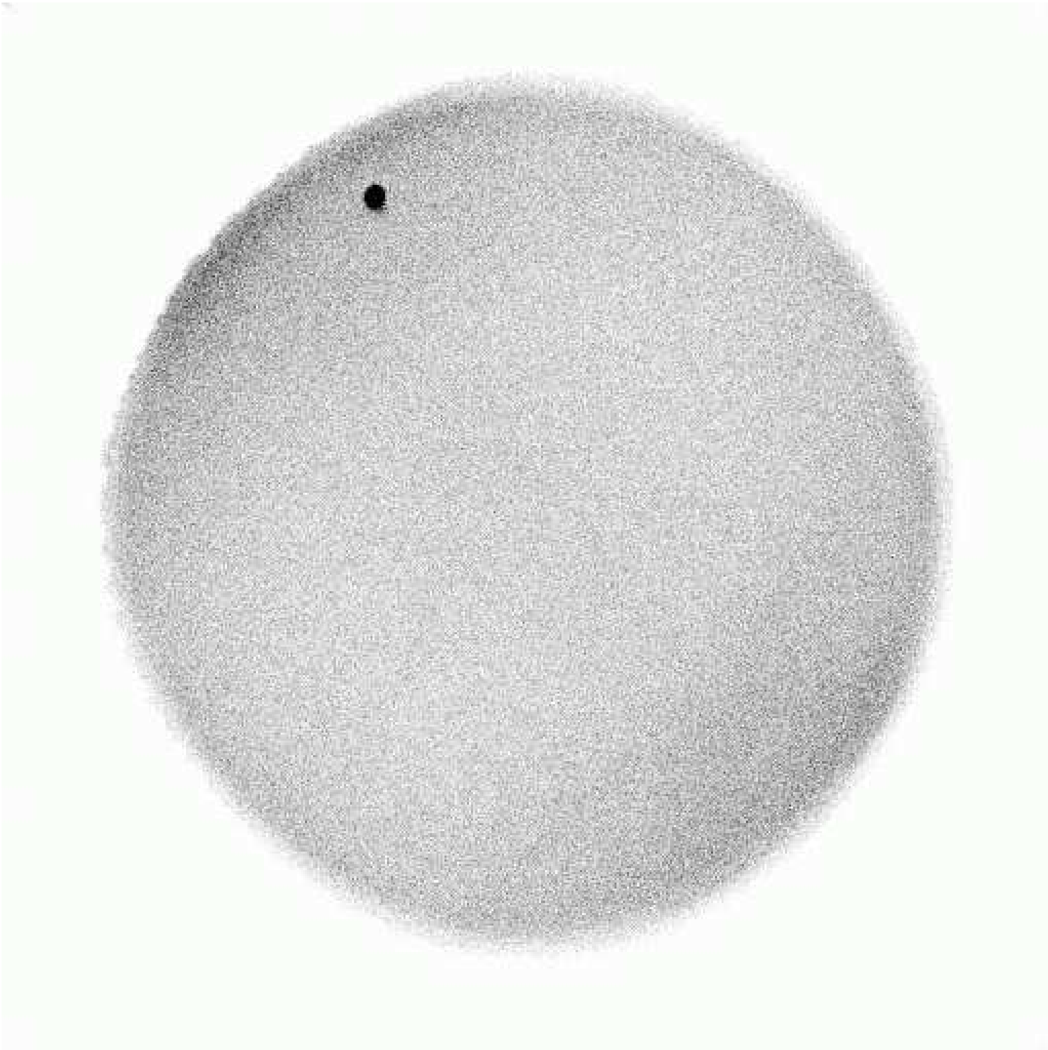

Abbildung 6.7: *Dunkelstrom-Messung des Detektors #*4351 *vom* 17. *August* 1999*: Das Bild wurde ohne LED-Strom mit geöffnetem Shutter aufgenommen. Die Meßdauer betrug* 5144, 22 *s (Windows-Systemuhr) bzw.* 5000, 549 *s (totzeitkorrigiert). Es sind* 1, 386, 681 *Counts im Bild enthalten. Die Darstellung des Bildes ist in* 8 *Bit Farbtiefe mit einer Weiß-Schwarz-Farbtabelle und Maximalwertorientierung (Negativ-Darstellung des Daten-Bitmaps).*

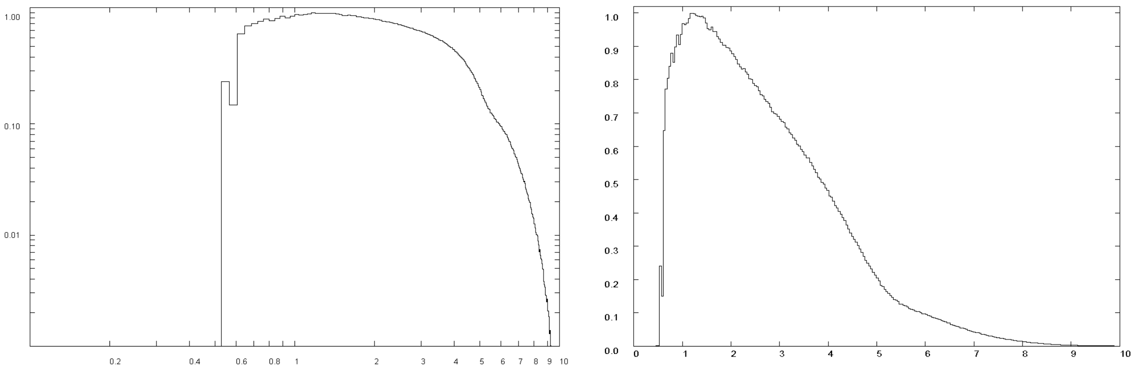

Abbildung 6.8: *Impulshöhen-Verteilung des Dunkelstroms von Detektor #*3971*: Diese Abbildungen zeigen die Impulshöhen-Verteilungen in logarithmischer und linearer Darstellung der Dunkelstrom-Messung Abb. 6.6 vom* 17*. August* 1999*. Die Meßdauer betrug* $5779,21$ *s (Windows-Systemuhr) bzw.* $5000,375$ *s (totzeitkorrigiert). Es sind* $8.979.438$ *Counts im Bild enthalten. Die Maximumsabzisse liegt bei* $1,17$ *V.*

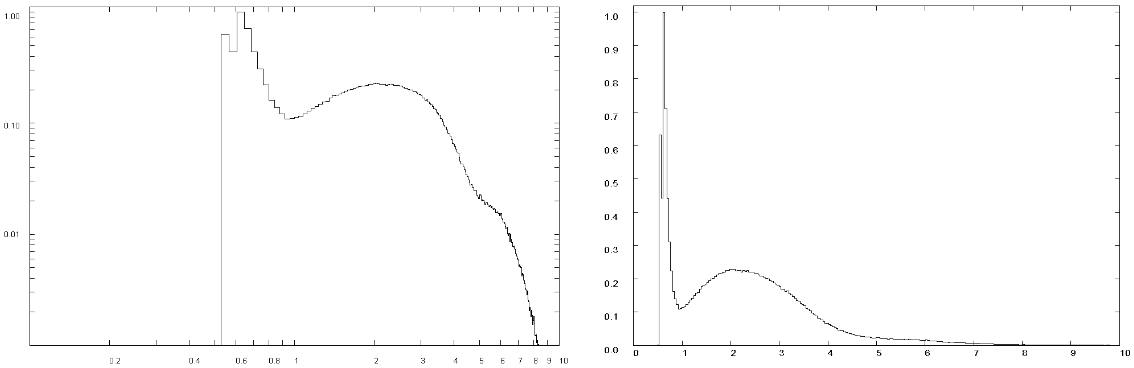

Abbildung 6.9: *Gemessene Impulshöhen-Verteilung des Dunkelstroms von Detektor #*4351*: Diese Abbildungen zeigen die Impulshöhen-Verteilungen in logarithmischer und linearer Darstellung der Dunkelstrom-Messung Abb. 6.7 vom* 17*. August* 1999*. Die Meßdauer betrug* $5144,22$ *s (Windows-Systemuhr) bzw.* $5000,549$ *s (totzeitkorrigiert). Es sind* $1,386,681$ *Counts im Bild enthalten. Die Maximumsabzisse liegt bei* $0,62$ *V.*

Die Maximalwerte der Dunkelströme sind zu kleineren Werten verschoben. Dies ist im Vergleich von Abb. 6.4 und Abb. 6.8 deutlich zu erkennen. Diese Verschiebung läßt sich darauf zurückführen, daß bei Dunkelströmen vermehrt kleine Impulshöhen auftreten und diese dann zunehmend unter die untere Schwelle der Elektronik für zulässige Pulse fallen. In Abb. 6.9 ist die Maximalwertverschiebung zu kleineren Werten bei reinen Dunkelstrommessungen sogar extrem, da der größte Teil des Dunkelstrom im Hot-Spot auftritt und dieser dadurch überproportional in die Impulshöhen-Verteilung eingeht. Die Abszisse des Maximalwerts dieser Verteilung liegt bei $0,62$ V während in der Flatfield-Aufnahme Abb. 6.5 dieses Detektors ein Maximumsabzisse bei $2,46$ V gemessen wurde.

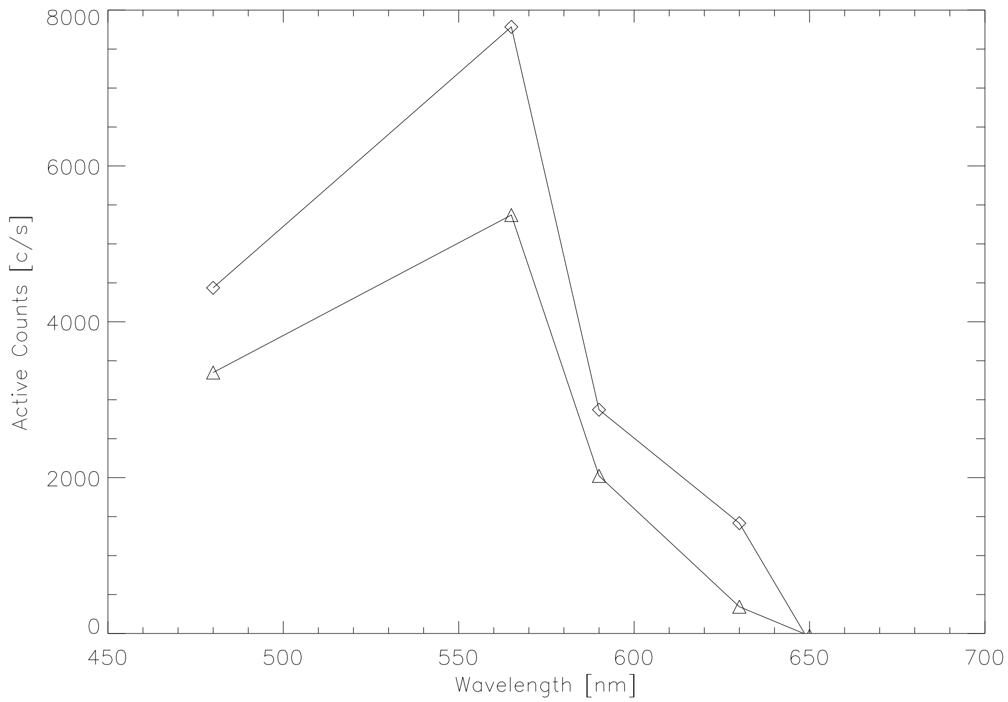

Abbildung 6.10: *Gemessene relative Empfindlichkeit der MCP-Detektoren #3971 und #4351: Für jede LED wurde eine Messung mit 0,20 mA durchgeführt und hier über der Wellenlänge, die dem Intensitätsmaximum der jeweiligen Diode entspricht, aufgetragen.*

6.3 Empfindlichkeitsmessungen der Detektoren #3971 und #4351

Für die Vergleichsmessung der beiden Detektoren (siehe Abschnitt 6.2) wurde nur die LED LB 5410 Anschluß Blau verwendet. Der Testaufbau von Diesch (1988) stellt aber LEDs in unterschiedlichen Wellenlängenbereichen zur Verfügung. In Tab. 6.2 sind die Daten dieser Meßreihe aufgeführt. Die Abb. 6.10 zeigt die Meßwerte bei 0,20 mA, die für jede Diode aufgenommen wurden, über der zentralen Wellenlänge der jeweiligen LED, die Anhang A entnommen werden kann. Detektor #3971 ($\triangle$) zeigt eine geringere Empfindlichkeit als Detektor #4351 ($\diamond$). Bei der LED SFH 480 Anschluß Rot 3, die ihr Intensitätsmaximum bei 890 nm hat, entspricht der Meßwert dem mittleren Dunkelstrom. Im infraroten Wellenlängenbereich besitzen die optischen Detektoren also keine Empfindlichkeit mehr, d.h. die LED SFH 480 Anschluß Rot 3 kann bei künftigen Testmessungen ignoriert werden. Die Messungen zeigen, daß der Detektor #4351 durchgehend eine höhere Empfindlichkeit hat, als Detektor #3971.

		Detektor #4351 Gehäuse #1			Detektor #3971 Gehäuse #2		
Leuchtdiode	Strom [mA]	Dunkelrate [c/s]	Total Pulses [c/s]	Effektive Rate[1] [c/s]	Dunkelrate [c/s]	Total Pulses [c/s]	Effektive Rate[1] [c/s]
LB 5410	0,20	2102,2	6502,2	4436,2	3291,7	6661,4	3348,9
Anschluß	0,50	2116,2	14365,6	12299,6	3325,2	12732,9	9420,4
Blau	1,00	1979,7	26797,9	24731,9	3320,6	22377,2	19064,7
CQY 94	0,20	1871,0	9685,5	7784,8	3340,8	8716,8	5370,5
Anschluß	0,40	1890,9	23065,7	21165,0	3363,6	19148,2	15801,9
Grün	0,60	1940,1	38490,7	36590,0	3334,5	31236,4	27890,1
CQY 96	0,20	2193,7	5102,3	2871,8	3366,1	5351,6	2022,9
Anschluß	0,50	2262,4	15152,6	12922,1	3324,3	12390,6	9061,9
Gelb	1,00	2235,3	37706,2	35475,7	3295,8	29148,7	25820,0
CQX 51	0,20	1881,2	3311,6	1416,4	3394,7	3758,0	343,1
Anschluß	0,80	1940,5	14360,9	12465,7	3415,3	6957,4	3542,5
Rot 1	1,20	1863,9	24880,3	22985,1	3434,6	10371,4	6956,5
CQY 24 Anschluß Rot 2	Diode defekt Kein Stromdurchfluß meßbar						
SFH 480	0,20	2168,0	1939,0	-107,0	3406,5	3347,9	-30,3
Anschluß	1,00	1836,6	2150,9	104,9	3393,1	3306,6	-71,6
Rot 3 IR	1,50	2133,5	2076,9	30,9	3335,1	3301,1	-77,1

Tabelle 6.2: Empfindlichkeitsmessung der Detektoren vom 26. und 27. Juli 1999

1: Dunkelraten der jeweiligen Diode gemittelt und von den Total Pulses subtrahiert.

6.4 Dynamik-Messungen mit beiden Datenerfassungssystemen

Zur Beurteilung der Leistungsfähigkeit sowohl der Elektronik als auch der Erfassungssysteme wurden am 16. Juni 1999 Messungen nur mit dem DPA und dann mit dem PC-System durchgeführt. Am 21. Juni 1999 wurden dann die entsprechenden Messungen mit der PDP 11/73 gemacht. Für alle diese Messungen wurde LED LB 5410 Anschluß Blau, Detektor #3971 und die neuere Version des DPA verwendet. Die Diode LB 5410 Anschluß Blau hat ihre maximale Intensität bei einer Wellenlänge von 480 nm. In diesem Bereich ist die Empfindlichkeit der Detektoren nicht allzu hoch (siehe Abschnitt 6.3). Dementsprechend war zu erwarten, daß die LED-Strom-Werte, die die gewünschten Zählraten von 2000 – 50000 Counts s^{-1} ergeben, nicht zu nahe beieinander liegen. Ansonsten ist die präzise Einstellung über das Drehpotentiometer am Testaufbau nicht gewährleistet. Die Daten sind in Anhang A tabellarisch zusammengefaßt.

In den folgenden Abschnitten werden verschiedene Raten des DPA angegeben. ‚Total Pulses' sind alle Pulse, die von der Elektronik gezählt werden. Im DPA sind zwei Schwellen eingestellt. Der ‚Lower Threshold' ist die untere Schwelle, die alle zu schwachen Impulse ausfiltert. Der ‚Upper Threshold' eliminiert alle Impulse, die über diesem Niveau liegen. Werden von den Total Pulses alle abgezogen, die unterhalb des Lower Threshold und oberhalb des Upper Threshold liegen und alle die, die in die Totzeit fallen, dann bleiben die ‚Active Pulses' übrig, die tatsächlich ins Bild oder die Impulshöhe eingehen.

Der erste Graph (Abb. 6.11) zeigt die aktiven Impulse in den verschiedenen Betriebsmodi des PC-Erfassungssystems über dem LED-Strom. Dazu sind die Total Pulses ohne Computer und die Active Pulses ohne Computer, also die Werte die nur mit dem DPA gemessen wurden, aufgetragen. Der zweite Graph (Abb. 6.12) zeigt die gleichen Werte normiert auf die Total Pulses ohne Computer über den den jeweiligen LED-Strömen entsprechenden Total Pulses ohne Computer.

Die Raten des PC-System fallen gleichmäßig, sprich parallel mit der Zahl der aktiven Pulse. Diese gehen aufgrund der elektronikbedingten Totzeit zurück. Die Bildung eines Maximums, also auch ein Abfall bei kleinen Zählraten, ist dadurch bedingt, daß bei kleinen Raten der Dunkelstrom einen größeren Anteil hat. Diese Impulse sind jedoch zu kleineren Impulshöhen verschoben und fallen daher zu einem großen Teil unter die untere Schwelle der Elektronik (siehe etwa Abb. 6.4 und Abb. 6.8).

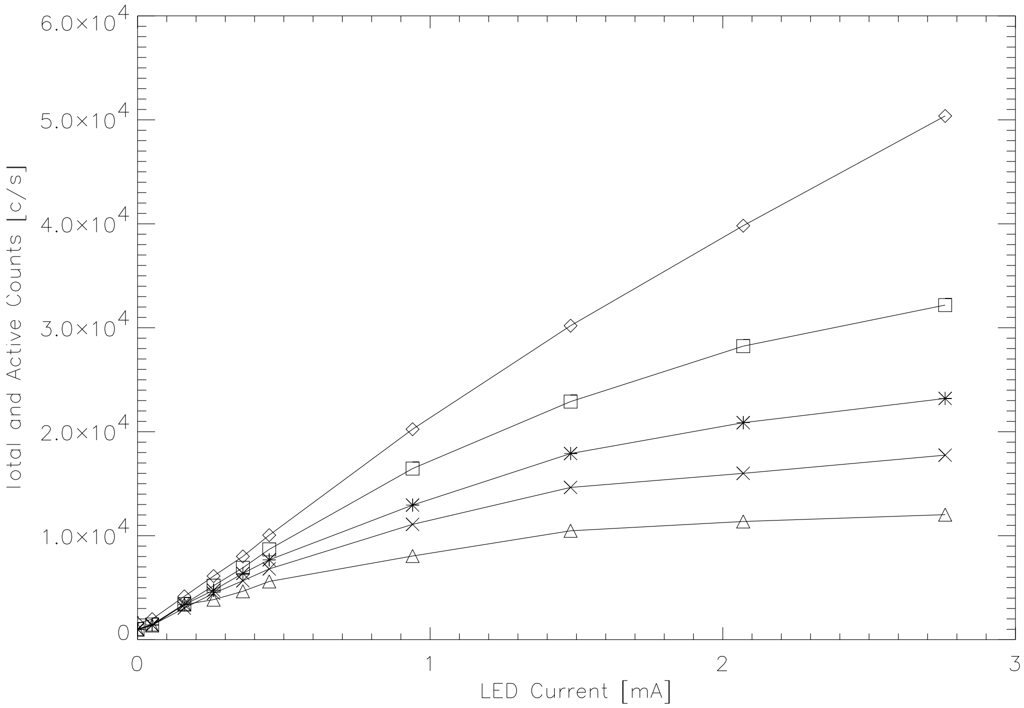

Abbildung 6.11: *Gemessene aktive Zählraten mit dem PC-System: Hier sind die aktiven Zählraten in den Betriebsmodi des DPA über den LED-Strömen zu sehen. Dazu wurden noch die Total Pulses und die Active Pulses ohne Computer eingetragen. Es wurden folgende Symbole verwendet: ◇ Total Pulses, □ Active Pulses ohne Computer, △ Mode xyzt, × Mode xy und ∗ Mode z.*

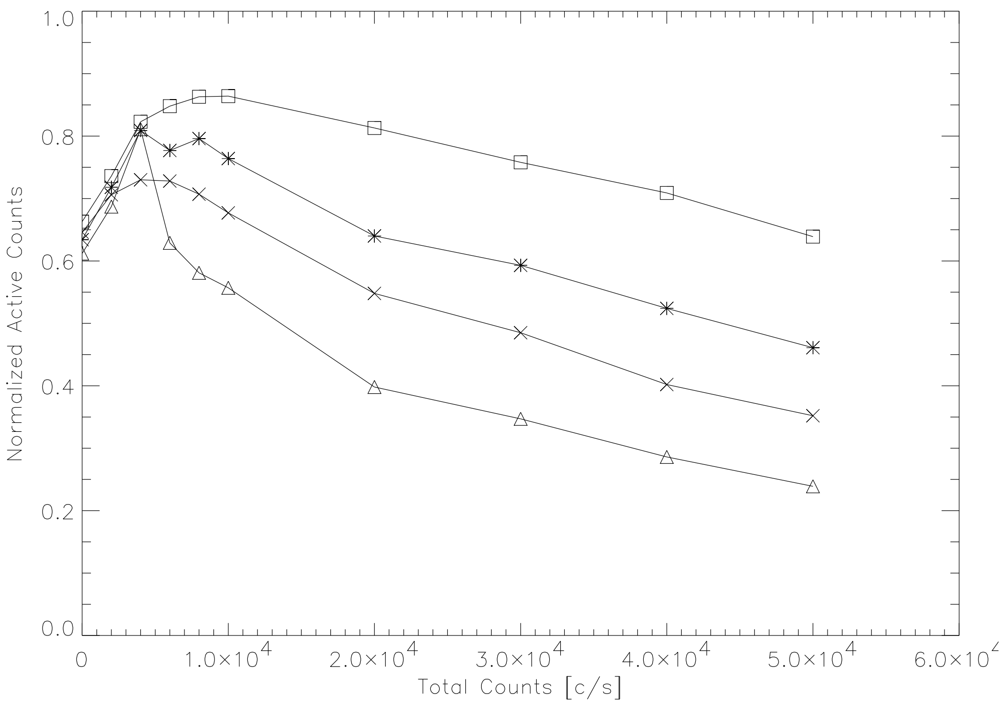

Abbildung 6.12: *Gemessene normierte aktive Zählraten mit dem PC-System: Hier sind die normierten aktiven Zählraten in den Betriebsmodi des DPA über den den jeweiligen LED-Strömen entsprechenden Total Pulses zu sehen. Dazu wurden noch die Active Pulses ohne Computer eingetragen. Es wurden folgende Symbole verwendet:* $\square$ *Active Pulses ohne Computer,* $\triangle$ *Mode xyzt,* $\times$ *Mode xy und* $*$ *Mode z.*

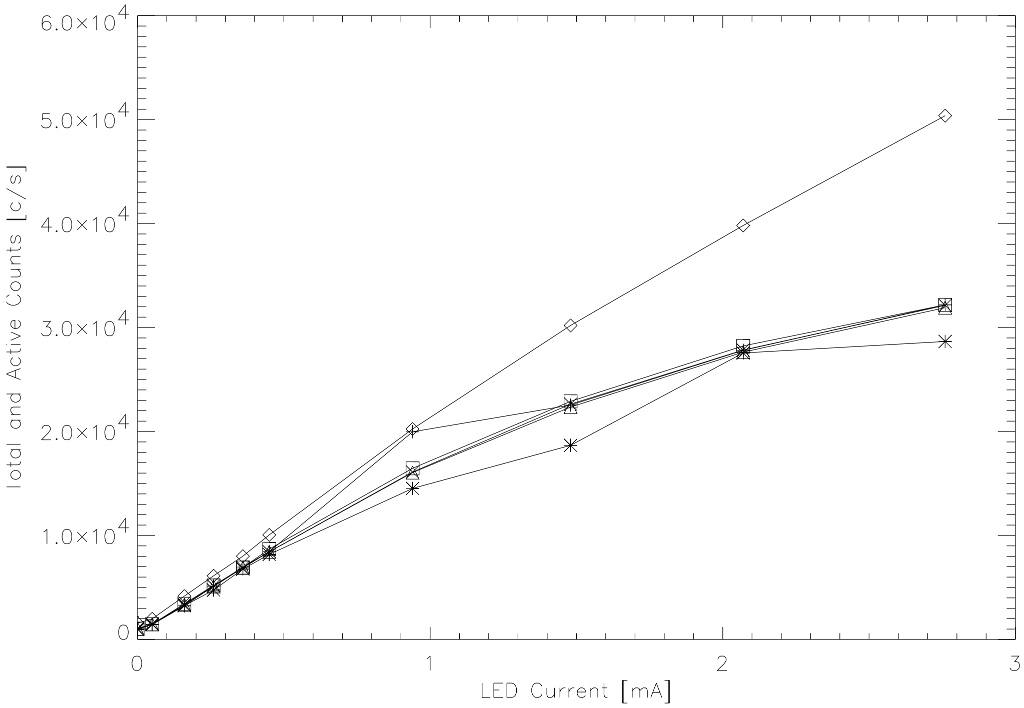

Abbildung 6.13: *Gemessene aktive Zählraten mit dem PDP 11/73-System: Hier sind die aktiven Zählraten in den Betriebsmodi des DPA über den LED-Strömen zu sehen. Dazu wurden noch die Total Pulses und die Active Pulses ohne Computer eingetragen. Es wurden folgende Symbole verwendet: ◇ Total Pulses, □ Active Pulses ohne Computer, △ Mode xy ohne Online-Darstellung, × Mode xy mit Online-Darstellung (BTR), ∗ Mode z ohne BTR und + Mode xy mit Hardware-Interface (HWI) und BTR.*

Die Abbildungen 6.13 und 6.14 zeigen die Meßwerte der PDP 11/73 in analoger Darstellung. Auch hier ist, wie bei der Messungen mit dem PC-System, ein Abfall der Zählraten mit steigenden Total Pulses zu erkennen. Dieses ist ein Effekt der Totzeit innerhalb der DPA-Elektronik. Die Bildung des Maximums und damit der Abfall bei kleinen Zählraten tritt hier in gleicher Weise auf, was die Annahme bestätigt, daß es sich nicht um erfassungssystembedingte Effekte handelt, sondern um den steigenden Anteil des Dunkelstroms und die Unterschreitung der unteren Diskriminatorschwelle in der Elektronik.

Des weiteren zeigt sich, daß die aktiven Zählraten des PDP 11/73-Systems wesentlich näher an den aktiven Raten ohne Computer liegen. Dies ist auf den Interrupt gesteuerten direkten Hardware-Zugriff bei der PDP 11/73 zurückzuführen. Beim Windows NT-System (siehe Abschnitt 5.2.7) ist kein so direkter Zugriff möglich und durch die treibergesteuerte Implementierung geht ein Teil des Leistungsvermögens verloren. Die Abbildungen 6.15 und 6.16, in denen die einzelnen Betriebsmodi des DPA mit jeweils beiden Erfassungssystemen dargestellt sind, zeigen dies ebenfalls. Auf die Möglichkeiten die Leistungsfähigkeit des PC-Systems zu steigern wird noch in Kapitel 7 näher eingegangen.

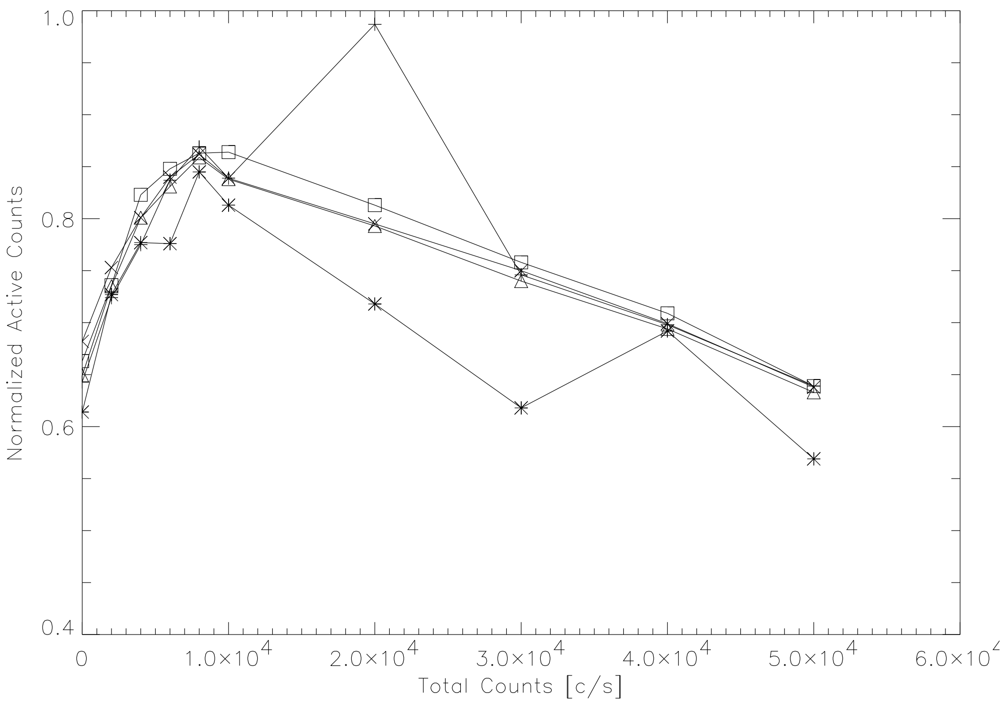

Abbildung 6.14: *Gemessene normierte aktive Zählraten mit dem PDP 11/73-System: Hier sind die normierten aktiven Zählraten in den Betriebsmodi des DPA über den den jeweiligen LED-Strömen entsprechenden Total Pulses zu sehen. Dazu wurden noch die Active Pulses ohne Computer eingetragen. Es wurden folgende Symbole verwendet: $\square$ Active Pulses ohne Computer, $\triangle$ Mode xy ohne Online-Darstellung, $\times$ Mode xy mit Online-Darstellung (BTR), $*$ Mode z ohne BTR und $+$ Mode xy mit Hardware-Integrations-Interface (HWI) und BTR.*

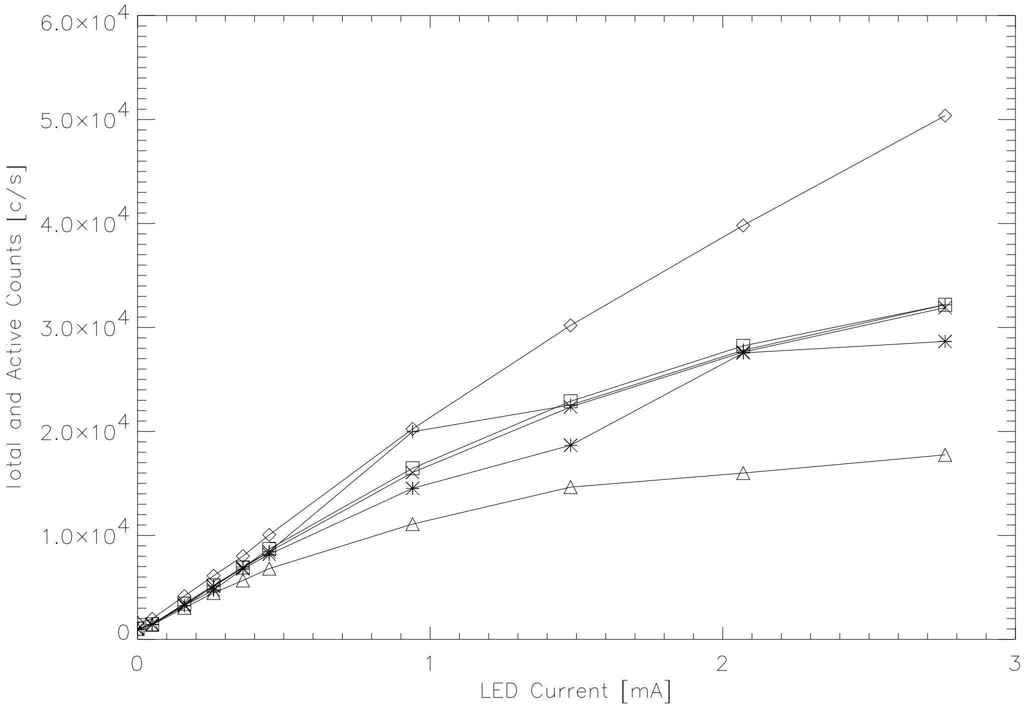

Abbildung 6.15: *Gemessene Zählraten mit den verschiedenen Meßsystemen im Bild-Modus xy: In dieser Graphik sind die aktiven Zählraten der Meßsystem mit DPA-Modus xy zusammengestellt. Zusätzlich sind die Total Counts und Active Counts ohne Computersystem mit eingezeichnet. Es wurden folgende Symbole verwendet: $\diamond$ Total Pulses, $\square$ Active Pulses ohne Computer, $\triangle$ Mode xy mit PC, $\times$ Mode xy mit PDP 11/73 ohne Online-Darstellung (BTR), $*$ Mode xy mit PDP 11/73 und BTR und $+$ Mode xy mit Hardware-Interface (HWI) und BTR.*

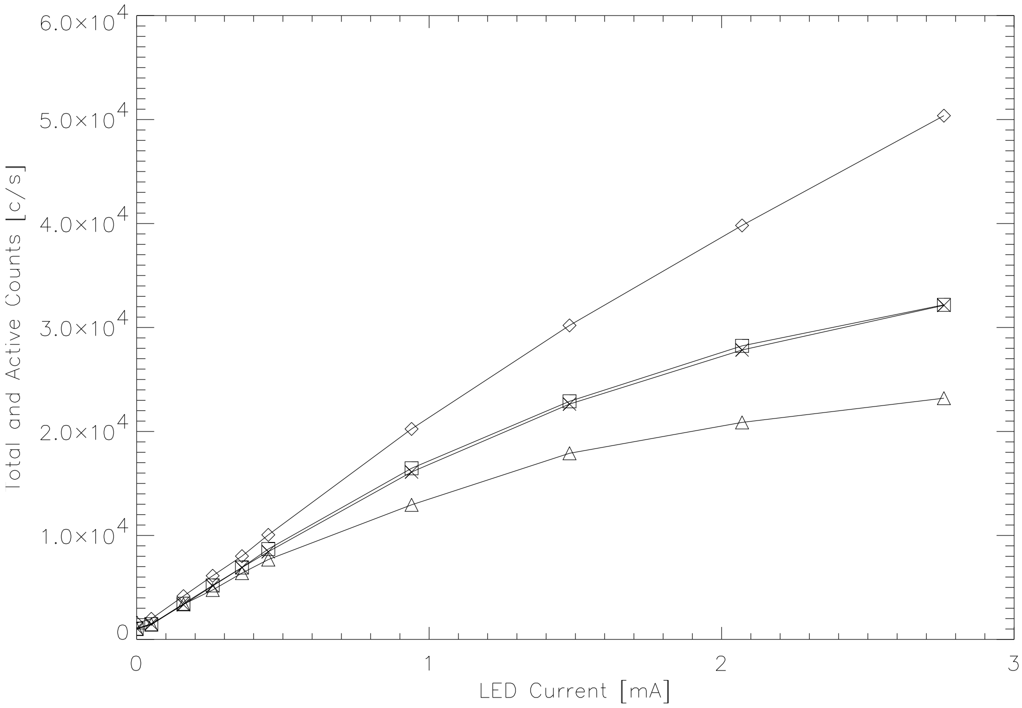

Abbildung 6.16: *Gemessene Zählraten mit den verschiedenen Meßsystemen im Impulshöhen-Modus z: In dieser Graphik sind die aktiven Zählraten der Meßsystem mit DPA-Modus z zusammengestellt. Zusätzlich sind die Total Counts und Active Counts ohne Computersystem mit eingezeichnet. Es wurden folgende Symbole verwendet: $\diamond$ Total Pulses, $\square$ Active Pulses ohne Computer, $\triangle$ Mode z mit PC, $\times$ Mode z mit PDP 11/73.*

6.5 Ergebnisse der Vergleichsmessungen

Die Flatfield-Aufnahmen Abb. 6.2 und Abb. 6.3 der beiden Detektoren haben gezeigt, daß beide fehlerhafte Bereiche in Form von Cold- oder Hot-Spots aufweisen. Bei Detektor #4351 beschränkt sich dies jedoch auf eine einzelne Stelle.

Die parallel dazu integrierten Impulshöhen-Verteilungen Abb. 6.4 und Abb. 6.5 ergaben bei Detektor #4351 die erwartete Gaußsche Verteilung. Der Maximalwert, und damit die Verstärkung der MCP, von $2,46$ V liegt innerhalb der gewünschten Grenzen von 2 bis 5 V. Die Halbwertsbreite von 101% (typischerweise $100 - 180\%$) ergibt eine gute Ortsauflösung dieses Detektors. Detektor #3971 mit einer Maximalwertabszisse von $1,95$ V und einer FWHM (Full Width Half Maximum) von 178% liefert nicht ganz so gute Ergebnisse.

Die Dunkelstrom-Messungen ergaben Raten bei Detektor #3971 von $1,71 \cdot 10^{-3}$ Counts Pixel^{-1} s^{-1} und bei Detektor #4351 von $2,64 \cdot 10^{-4}$ Counts Pixel^{-1} s^{-1}. Der Dunkelstrom konzentriert sich bei Detektor #4351 in einem Hotspot, in dem im Maximumpixel in der Meßzeit 5700 Counts registriert worden, während im gesamten Bild nur durchschnittlich 1 Count Pixel^{-1} vorhanden ist. Bei Detektor #3971 treten mehrere Cold- und Hot-Spots auf.

Beide Detektoren zeigen im Infraroten bei der LED SFH 480 Anschluß Rot 3 (Maximale Intensität bei $\lambda = 890$ nm) keine Empfindlichkeit mehr. Diese kann in zukünftigen Messungen vernachlässigt werden. Beide Detektoren haben ihr Maximum bei der LED CQY 96 Anschluß Gelb. Diese hat ihre maximale Intensität bei einer Wellenlänge von 590 nm. Die Messungen zeigten auch eine durchgängig höhere Empfindlichkeit des Detektors #4351.

Diese Vergleichsmessungen haben gezeigt, daß Detektor #4351 eindeutig vorzuziehen ist. Er hat nur eine schadhafte Stelle in Form eines Hot-Spots, liefert eine sehr gute Ortsauflösung, ist durchgängig empfindlicher und hat einen geringeren Dunkelstrom.

Die Vorverstärkerelektronik von Lindenberger (1992) ist in der Lage ca. 500.000 Counts s^{-1} zu verarbeiten. Der DPA mit seinen derzeitigen Bauteilen läßt aber nur etwa 100.000 Counts s^{-1} zu. Das neue wie das alte System sind in der Lage diese Anzahl zu bearbeiten. Aufgrund der direkteren Zugriffe ist die PDP 11/73 im Moment in der Lage noch geringere verarbeitungsbedingte Totzeiten zu liefern. Die Möglichkeiten zur Steigerung der Zählraten im neuen PC-System werden im folgenden Kapitel 7 näher beschrieben.

Kapitel 7

Ausblick und Zusammenfassung

Das neue Windows-Programm bietet gegenüber der alten Datenerfassung eine ganze Reihe von Vorteilen. Die alten Kommandozeilen-Steuerung der PDP 11/73 wurde durch eine vollständig graphische Oberfläche ersetzt, die dem Benutzer eine gewohnte und komfortable Steuerung der Anwendung ermöglicht. Um diese Umgebung zu schaffen, wurden im Programm viele Standardelemente verwendet, die der Computernutzer aus anderen Windows-Paketen, wie etwa Microsoft-Office, kennt. Die Funktionalität des alten Programms wurde dabei weitgehend übernommen und um einige Funktionen erweitert. Durch die objektorientierte und gekapselte Struktur und die Verwendung einer modernen Programmiersprache, wie Borland C++ 5.0 , ist es auch kein Problem, weitere Bearbeitungs- oder Integrationsfunktionen dem Programm hinzuzufügen. In den Anhängen C und D sind die Programmfunktionen und die Spezifikationen und Schnittstellen der Datenklassen ausführlich beschrieben. Jede dieser Klassen ist in sich geschlossen und kann jederzeit durch eine neue ersetzt werden, die für die bisherige Funktionalität die gleiche Schnittstelle zum Hauptprogramm bereitstellt. Dadurch ist es auch möglich die Vorteile des Windows NT-Betriebssystems voll zu nutzen. Bei Verwendung anderer Hardware, wie etwa der Input-/Output-Karte, müssen nur die beiden Datenklassen, die den Zugriff durchführen, durch entsprechend neue mit gleicher Ansteuerung durch das Hauptprogramm ersetzt werden. Das neue Programm legt mit seiner Struktur auch die Basis für ein komplettes Programmpaket zur Aufzeichnung und Auswertung von Daten, die mit den optischen MCP-Detektoren gemacht werden können und stellt auch Basisklassen zur Verfügung, die in Windows-basierten Programmen anderer Projekte Verwendung finden können.

Es wurde das Microsoft Windows NT 4.0 Betriebssystem verwendet, das ebenfalls Vorteile hat. Die Microsoft-Betriebssysteme sind weitestgehend abwärtskompatibel, so daß dieses Programm, eventuell mit einem aktualisierten Treiber, auf weiteren Entwicklungsstufen des Windows NT-Systems problemlos weiter betrieben werden kann. Auch die Portierbarkeit auf andere Systeme, wie etwa ein DEC-Alpha-System, bietet den Vorteil, nicht an ein System gebunden zu sein. Ist dort Windows NT und ein entsprechender Treiber installiert, so setzt das Betriebssystem die Software-Befehle entsprechend um, und der Betrieb ist kein Problem. Windows NT erlaubt auch Multi-Prozessor-Maschinen für sein Multi-Tasking und setzt somit der technischen Entwicklung auch in Zukunft kaum Grenzen. Diese Eigenschaften und die damit verbundene Zukunftssicherheit waren mit ausschlaggebend für die Verwendung des Windows NT-Systems.

Die Software-Anforderungen aus Abschnitt 4.1 wurden bis auf die Geschwindigkeitanforderung alle umgesetzt. Die Vergleichsmessungen haben gezeigt, daß die aktiven Zählraten des PDP 11/73-Systems, die sich nahe an den aktiven Zählraten ohne Erfassungssystem befinden, noch nicht ganz erreicht werden. Dies hat verschiedene Gründe.

Ein großer Teil der höheren Rechnergeschwindigkeit wird durch die graphische Oberfläche aufgebraucht. Im Moment wird ein Intel Pentium Prozessor mit einem Boardtakt von 133 MHz und 64 MB Hauptspeicher verwendet. Dieses System ist bereits überholt und ein Intel Pentium II mit mehr Hauptspeicher und einer entsprechend guten Graphikkarte kann hier sicher Abhilfe schaffen.

Auch die NI AT-DIO32F ist nicht neuerster Stand der Technik. Sie wurde Anfang der 90er Jahre für die Datenaquisition in 16 MHz IBM-Kompatiblen mit entsprechenden Bauteilen entwickelt. Die typische Übertragungsrate von 450 kwords s^{-1} fällt noch hinter das 16 Bit DRV11J Interface der PDP 11/73 mit 500 kwords s^{-1} zurück. Da diese Komponenten über kurz oder lang ausgetauscht werden müssen, wurde auf die gekapselte Struktur des Programms besonderen Wert gelegt. Aber auch mit dieser Hardware kann die Zählrate des Programms noch gesteigert werden.

Im Moment werden die Daten blockweise mit einer Größe von 1000 Worten eingelesen und abgearbeitet. Die Implementierung einer doppelt gepufferten Einleseroutine dürfte die Zählraten noch etwas steigern. Dabei wird der erste Teil des Blocks (500 Worte) eingelesen und während der zweite Teil (die nächsten 500 Worte) von der Karte selbständig gelesen werden, arbeitet die CPU die ersten 500 Worte ab. Dabei muß nur sichergestellt sein, daß die Abarbeitung schneller geschieht als das Einlesen des nächsten halben Blockes, sonst kommt es zu Datenverlusten. Im speziellen Fall dürfte das aber kein Problem sein, da die Abarbeitung sich im wesentlichen auf logische Bitoperationen und Bitshifts beschränkt, die in der CPU praktisch mit der internen Taktung ablaufen.

Des weiteren wird im Moment nur über die Datenleitungen mit Multiplexverfahren des DPA gelesen. Das im Hause entwickelte Hardware-Integrations-Interface liefert dagegen 20 Bit direkte Ortskoordinaten, welche nicht weiter bearbeitet werden müssen. Die Implementierung dieses Modus könnte ebenfalls noch ein Erhöhung der PC-Zählrate bewirken, da sich bei den PDP 11/73-Messungen gezeigt hat, daß dieser Modus unter Verzicht auf Impulshöhen- und Zeitinformation auch dort die besseren Ergebnisse liefert.

Bei der Beschreibung des Windows NT-Betriebssystems in Abschnitt 5.2 wurde daraufhingewiesen, daß ein direkter Zugriff auf die Hardware nicht möglich ist, sondern Treiber verwendet werden müssen. Dies wurde in Abschnitt 5.2.7 weiter vertieft. In diesem Bereich liegt wohl die beste Möglichkeit Geschwindigkeit zu gewinnen. Im Moment wird ein Standardtreiber verwendet, der einen sehr großen Befehlsumfang zur Verfügung stellt. Für die grundsätzliche Entwicklung des neuen Programms war dies, auch dank der sehr guten Dokumentation, ausreichend. Um das System aber auszureizen, müßte ein spezieller Kernel Mode Device Treiber für genau diese Anwendung entwickelt werden. Kernel Mode bedeutet privilegierten Zugriff auf den Speicher und direkten Zugriff auf die Schnittstellen-Karte. Die von Microsoft empfohlene und unterstützte Sprache für die Entwicklung ist C, nicht C++. Die Treiber müssen eine bestimmte Struktur beinhalten und von einem speziellen Compiler, wie etwa dem Windows NT Device Driver Kit (DDK) von Microsoft, abgearbeitet werden. Dies ist sicherlich aufwendig und verlangt viel Einarbeitungszeit. Es ist daher zu überlegen, ob dies mit der jetzigen Karte NI AT-DIO32F und ihren Übertragungsraten sinnvoll ist, oder ob die Entwicklung nicht mit der Anschaffung einer neuen, schnellen Karte einhergehen sollte. Wird dieser Schritt aber getan, speziell mit einer besseren Steckkarte, so kann mit Sicherheit ein sehr erheblicher Geschwindigkeitsgewinn erzielt werden. Weitere Anhaltspunkte zur Treiberentwicklung finden sich in etwa Hanrahan (1999).

Durch das im Rahmen dieser Diplomarbeit erstellte Programm ist der weitere Betrieb der MCP-Detektoren am Institut für Astronomie und Astrophysik der Universität Tübingen, Abteilung Astronomie, auch in Zukunft sichergestellt. Das Programm bietet die Grundlage für weitere Entwicklungen und kann den kommenden technischen und bearbeitungstechnischen Anforderungen mit wenig Aufwand angepaßt werden.

Die Vergleichsmessung der Detektoren ergab die eindeutig besseren Werte bei Detektor #4351. Dieser hat den geringeren Dunkelstrom, der sich zudem in den wenigen Pixeln eines einzelnen Hot-Spots konzentriert, und durchgängig eine höhere Empfindlichkeit bei allen gemessenen Wellenlängen. Die Impulshöhen-Verteilung des Detektors #4351 ist erwartungsgemäß Gaußförmig und die geringe Halbwertsbreite spricht für die gute Ortsauflösung des Detektors. Die Messungen mit den Detektoren und dem LED-Array dokumentieren den Stand der optischen Detektoren am Institut und können zukünftig als Referenz und für die Beurteilung der alterungsbedingten Veränderungen an den Detektoren dienen.

Literaturverzeichnis

Barnstedt J., 1981, Diplomarbeit, Universität Tübingen

Barnstedt J., 1985, Dissertation, Universität Tübingen

Barnstedt J., 1999, Institut fü Astronomie und Astrophysik, Universität Tübingen, ORFEUS Homepage, available at `http://astro.uni-tuebingen.de/groups/orfeus`

Diesch C., 1988, Diplomarbeit, Universität Tübingen

digital, 1984, Computer Special Systems, Alternate Buffer DMA Interfaces

Dunbar B., 1999, National Aeronautics & Space Administration (NASA) Homepage, available at `http://www.nasa.gov`

Ezzell B., 1993, Window NT 3.1 Programming, Ziff-Davies Press

Gutekunst M., 1986, Dissertation, Universität Tübingen

Haider P., 1988, Diplomarbeit, Universität Tübingen

Hanrahan J.E., 1999, Kernel Mode Systems Homepage, available at `http://www.cmkml.com/`

Hurwitz M., Bowyer C.S., 1995, Proc. SPIE 2517, 116

Jagutzki O., Barnstedt J., Spillmann U., et al., 1999, In: Proc. SPIE., to be published

Joseph C.L., 1995, Supplement to Ap&SS (Experimental Astronomy) 6, 97

Koschützke O., 1999, Proxitronic Homepage, available at `http://www.proxitronic.de`

Lindenberger A., 1992, Diplomarbeit, Universität Tübingen

Louis D., 1996a, Borland C++ 5 Das Kompendium, Markt & Technik, Buch- und Software-Verlag

Louis D., 1996b, Schnell-Übersicht Borland C++ 5, Markt & Technik, Buch- und Software-Verlag

National Instruments, 1991, AT-DIO-32F User Manual

Petzold C., 1996, Programming Windows 95, Microsoft Press

Polidan R., 1997, The EUV Spectrograph Homepage, available at `http://snoopy/gsfc.nasa.gov/$\sim$orfeus2/detailedEUV.html`

Seal D., 1999, Jet Propulsion Laboratory (JPL), Mission and Spacecraft Library, available at `http://leonardo.jpl.nasa.gov/msl/QuickLooks/pioneer10QL.html`

Woodgate B., 1997, American Astronomical Society Meeting, 190, #42.05

Anhang A

Meßprotokolle

In den nachfolgenden Tabellen sind alle Meßergebnisse und Auswertungsdaten zusammengefaßt. Tabellen aus dem laufenden Text sind hier teilweise der Vollständigkeit halber nochmals aufgeführt.

1.1 Relative Empfindlichkeit der Detektoren

Messung mit dem älteren Digitalen Positions-Analysator (deutsche Beschriftung).

		Detektor #4351 Gehäuse #1			Detektor #3971 Gehäuse #2		
Leuchtdiode	Strom [mA]	Dunkelrate [c/s]	Total Pulses [c/s]	Effektive Rate[1] [c/s]	Dunkelrate [c/s]	Total Pulses [c/s]	Effektive Rate[1] [c/s]
LB 5410	0,20	2102,2	6502,2	4436,2	3291,7	6661,4	3348,9
Anschluß	0,50	2116,2	14365,6	12299,6	3325,2	12732,9	9420,4
Blau	1,00	1979,7	26797,9	24731,9	3320,6	22377,2	19064,7
CQY 94	0,20	1871,0	9685,5	7784,8	3340,8	8716,8	5370,5
Anschluß	0,40	1890,9	23065,7	21165,0	3363,6	19148,2	15801,9
Grün	0,60	1940,1	38490,7	36590,0	3334,5	31236,4	27890,1
CQY 96	0,20	2193,7	5102,3	2871,8	3366,1	5351,6	2022,9
Anschluß	0,50	2262,4	15152,6	12922,1	3324,3	12390,6	9061,9
Gelb	1,00	2235,3	37706,2	35475,7	3295,8	29148,7	25820,0
CQX 51	0,20	1881,2	3311,6	1416,4	3394,7	3758,0	343,1
Anschluß	0,80	1940,5	14360,9	12465,7	3415,3	6957,4	3542,5
Rot 1	1,20	1863,9	24880,3	22985,1	3434,6	10371,4	6956,5
CQY 24 Anschluß Rot 2	Diode defekt Kein Stromdurchfluß meßbar						
SFH 480	0,20	2168,0	1939,0	-107,0	3406,5	3347,9	-30,3
Anschluß	1,00	1836,6	2150,9	104,9	3393,1	3306,6	-71,6
Rot 3 IR	1,50	2133,5	2076,9	30,9	3335,1	3301,1	-77,1

Tabelle A.1: Empfindlichkeitsmessung der Detektoren vom 26. und 27. Juli 1999

1: Dunkelraten der jeweiligen Diode gemittelt und von den Total Pulses subtrahiert.

LED	Anschluß	Hersteller	Max. Intensität [nm]	FWHM [nm]	Intensität [W/cm^2]	Bemerkung
LB 5410	Blau	Siemens	480	80	$6,17 \cdot 10^{-8}$	1
CQY 94	Grün	Valvo	565	30	$1,77 \cdot 10^{-6}$	1
CQY 96	Gelb	Valvo	590	40	$1,34 \cdot 10^{-6}$	1
CQX 51	Rot 1	Valvo	630	45		
CQY 24	Rot 2	Valvo	650	20	$1,60 \cdot 10^{-6}$	1, defekt
SFH 480	Rot 3	Siemens	890	80	$5,60 \cdot 10^{-5}$	2

Tabelle A.2: Technische Daten der LED-Anordnung, aus Diesch (1988)

1: Maximaler Diodenstrom 20 mA

2: Maximaler Diodenstrom 2 mA

1.2 Dynamik-Messungen

Zu allen folgenden Messungen wurde die Diode LB 5410 Anschluß Blau und der DPA neueren Datums (englische Beschriftung) verwendet. Die Normierung bezieht sich auf die Total Pulses des jeweiligen LED-Stroms. Diese Messungen wurden am 16. Juni 1999 durchgeführt.

1.2.1 Messung ohne Computersysteme

LED-Strom [mA]	Mode xyzt [c/s]	Mode xy [c/s]	Mode z [c/s]	Mittelwert [c/s]
0,00	1602,1	1562,8	1510,1	1558,3
0,05	1585,7	1578,8	1573,8	1579,4
0,16	1551,2	1559,3	1551,6	1554,0
0,26	1560,7	1570,9	1580,2	1570,6
0,36	1612,1	1610,8	1607,6	1610,2
0,45	1630,0	1669,3	1663,4	1654,2
0,94	1523,7	1486,2	1491,0	1500,3
1,48	1582,2	1566,0	1540,8	1563,0
2,07	1548,0	1530,6	1506,0	1528,2
2,76	1525,2	1515,1	1540,7	1527,0

Tabelle A.3: Dunkelstromraten in den DPA-Modi und Mittelwerte

LED-Strom [mA]	Mode xyzt [c/s]	Mode xy [c/s]	Mode z [c/s]	Mittelwert [c/s] (**T**)
0,00	1599,8	1566,2	1527,3	1564,4
0,05	1950,8	1992,2	1996,6	1979,9
0,16	4024,8	4264,4	4220,6	4169,9
0,26	6098,6	6141,5	6133,1	6124,4
0,36	7986,6	8013,8	8035,8	8012,1
0,45	10085,8	10050,2	10028,0	10054,7
0,94	20216,3	20261,3	20248,9	20242,2
1,48	30262,5	30216,9	30121,0	30200,1
2,07	39745,8	39822,1	39880,9	39816,3
2,76	50352,5	50351,5	50417,6	50373,9

Tabelle A.4: Total Pulses in den DPA-Modi und Mittelwerte

LED-Strom [mA]	Mode xyzt [c/s]	Mode xy [c/s]	Mode z [c/s]	Mittelwert [c/s] (**A**)	Normierter Mittelwert (A/T)
0,00	1060,8	1044,0	1004,9	1036,6	0,663
0,05	1453,5	1447,5	1470,9	1457,3	0,736
0,16	3351,1	3481,2	3460,1	3430,8	0,823
0,26	5191,1	5192,8	5196,9	5193,6	0,848
0,36	6860,8	6889,6	6999,4	6916,6	0,863
0,45	8683,4	8687,0	8699,3	8689,9	0,864
0,94	16452,4	16433,4	16497,7	16461,2	0,813
1,48	22958,4	22816,2	22933,9	22902,8	0,758
2,07	27939,0	28587,0	28177,4	28234,5	0,709
2,76	32137,4	32193,8	32223,2	32184,8	0,639

Tabelle A.5: Active Pulses in den DPA-Modi und Mittelwerte

1.2.2 Messung mit PC-System

Gemessen wurde bei 10 verschiedenen LED-Stromwerten. Die erste Messung wurde bei geöffnetem Shutter ohne LED-Strom durchgeführt, um den Dunkelstrom zu bestimmen. Die anderen Ströme entsprechen etwa 2000, 4000, 6000, 8000, 10000, 20000, 30000, 40000 und 50000 Counts s^{-1}. Von der PC Seite wurden im jeweiligen DPA-Modus dann parallel Bild und Impulshöhe (xyzt), nur Bild (xy) und nur Impulshöhe (z) integriert.

Die Normierung bezieht sich auf den Mittelwert der Total Pulses (**T**) des jeweiligen LED-Stroms aus der Messung ohne Computersysteme. Die PC-Messungen mit der NI AT-DIO32F wurden 16. Juni 1999 durchgeführt.

LED-Strom [mA]	Mode xyzt [c/s] (**B**)	Mode xyzt Normiert (B/T)	Mode xy [c/s] (**C**)	Mode xy Normiert (C/T)	Mode z [c/s] (**D**)	Mode z Normiert (D/T)
0,00	958,1	0,612	1010,6	0,646	991,6	0,634
0,05	1360,2	0,687	1397,2	0,706	1421,3	0,718
0,16	3380,6	0,811	3042,7	0,730	3372,5	0,809
0,26	3849,9	0,629	4459,8	0,728	4757,0	0,777
0,36	4654,8	0,581	5666,5	0,707	6378,1	0,796
0,45	5600,0	0,557	6807,3	0,677	7681,7	0,764
0,94	8049,2	0,398	11092,3	0,548	12946,7	0,640
1,48	10477,0	0,347	14642,9	0,485	17903,1	0,593
2,07	11369,3	0,286	16000,8	0,402	20876,1	0,524
2,76	12025,9	0,239	17750,2	0,352	23200,3	0,461

Tabelle A.6: Active Pulses mit PC-System in den Modi des DPA

Die Systemzeit des Windows NT-Systems wird mit der Meßdauer des DPA verglichen. Die DPA-Zeitanzeige bleibt stehen, wenn vom Meßsystem keine Daten abgeholt werden oder innerhalb der Elektronik, etwa bei den AD-Wandlern, Totzeiten auftreten. Dadurch ergibt sich ein Maß für die vom System bedingte Totzeit. Fehlgänge der Windows-Systemuhr, die bei hoher Auslastung der CPU (Central Processing Unit) auftreten können, wurden hier nicht berücksichtigt.

LED	Total	Zeiten im Mode xyzt			Zeiten im Mode xy			Zeiten im Mode z		
Strom [mA]	Pulses [c/s] (**T**)	DPA [s]	System [s]	Totzeit [%]	DPA [s]	System [s]	Totzeit [%]	DPA [s]	System [s]	Totzeit [%]
0,00	1564,4	60,379	66,41	9,08	60,288	63,73	5,40	60,444	62,95	3,98
0,05	1979,9	60,383	68,08	11,31	60,276	64,32	6,29	60,370	63,67	5,18
0,16	4169,9	60,260	84,40	28,60	59,963	68,49	12,45	60,143	66,22	9,18
0,26	6124,4	60,261	89,70	32,82	60,317	75,12	19,71	60,548	71,54	15,36
0,36	8012,1	60,240	97,04	37,92	60,361	79,87	24,43	60,615	72,40	16,28
0,45	10054,7	61,229	106,28	42,39	60,302	85,77	29,69	60,285	76,14	20,82
0,94	20242,2	60,345	153,39	60,66	60,443	111,08	45,59	60,208	94,90	36,56
1,48	30200,1	60,230	208,78	71,15	60,177	137,83	56,34	60,206	109,67	45,10
2,07	39816,3	60,090	249,25	75,89	60,104	169,69	64,58	60,161	131,16	54,13
2,76	50373,9	60,115	308,03	80,48	60,102	205,18	70,71	60,138	152,59	60,59

Tabelle A.7: Totzeiten in den verschiedenen Meßmodi des PC und des DPA

1.2.3 Messung mit PDP 11/73

Die Normierung bezieht sich auf die Total Pulses des jeweiligen LED-Stroms aus der Messung ohne Computersysteme. Die Messungen mit der PDP 11/73 wurden am 21. Juni 1999 durchgeführt.

LED-Strom [mA]	Mode xy [c/s]	Mode xy Normiert	Mode z [c/s]	Mode z Normiert
0,00	1014,8	0,649	1066,2	0,682
0,05	1455,8	0,735	1490,7	0,753
0,16	3339,1	0,801	3340,4	0,801
0,26	5089,0	0,831	5144,6	0,840
0,36	6884,8	0,859	6909,7	0,862
0,45	8428,4	0,838	8433,6	0,839
0,94	16050,0	0,793	16100,0	0,795
1,48	22352,0	0,740	22637,0	0,750
2,07	27635,0	0,694	27836,0	0,699
2,76	31911,0	0,633	32131,0	0,638

Tabelle A.8: Active Pulses mit PDP 11/73 ohne Online-Darstellung

LED-Strom [mA]	Mode xy [c/s]	Normiert
0,00	959,9	0,614
0,05	1440,0	0,727
0,16	3241,6	0,777
0,26	4749,9	0,776
0,36	6766,6	0,845
0,45	8177,0	0,813
0,94	14527,0	0,718
1,48	18671,0	0,618
2,07	27556,0	0,692
2,76	28657,0	0,569

Tabelle A.9: Active Pulses mit PDP 11/73 mit Online-Darstellung (BTR)

LED-Strom [mA]	Mode xy [c/s]	Normiert
0,00	1006,0	0,643
0,05	1433,3	0,724
0,16	3232,8	0,775
0,26	5127,6	0,837
0,36	6966,3	0,869
0,45	8440,6	0,839
0,94	19973,0	0,987
1,48	22543,0	0,746
2,07	27811,0	0,698
2,76	32185,0	0,639

Tabelle A.10: Active Pulses mit PDP 11/73 Hardware-Integrations-Interface (HWI) und BTR

Anhang B

Adapterplatine

Zum Anschluß des DPA an das neue PC-System wurde eine Adapterplatine entwickelt und hergestellt.

2.1 Pinbelegungen und Verbindungen der Systeme

In den folgenden Tabellen sind Pinbelegungen und die entsprechenden Blockkonfigurationen der NI AT-DIO32F angegeben.

Nr.	DPA-DRQ Pin		Belegung	NI AT-DIO32F Pin		Belegung	Kabelfarbe
1	1		DAT15	45		DIOB7	blau
2		20	DAT14		46	DIOB6	blau
3	2		DAT13	43		DIOB5	blau
4		21	DAT12	49		DIOB4	blau
5	3		DAT11		48	DIOB3	blau
6		22	DAT10		44	DIOB2	blau
7	4		DAT09		50	DIOB1	blau
8		23	DAT08	47		DIOB0	blau
9	5		DAT07	41		DIOA7	rot
10		24	DAT06		36	DIOA6	rot
11	6		DAT05		42	DIOA5	rot
12		25	DAT04	35		DIOA4	rot
13	7		DAT03		40	DIOA3	rot
14		26	DAT02		38	DIOA2	rot
15	8		DAT01	39		DIOA1	rot
16		27	DAT00	37		DIOA0	rot
17	9		GND				
18		28	Frei				
19	10		GND				
20		29	DAT AVAIL	33		REQ1	grün weiß
21	11		GND				
22		30	ON LINE		10	DIOC7	weiß
23	12		GND				
24		31	DIR IN		4	DIOD0	weiß
25	13		GND				
26		32	Frei				
27	14		GND				
28		33	Frei				
29	15		GND				
30		34	DAT ACCEPT	27		ACK1	grün weiß
31	16		GND				
32		35	Frei				
33	17		GND				
34		36	FNCT 3	13		DIOC2	braun
35	18		FNCT 2		12	DIOC1	braun
36		37	FNCT 1		14	DIOC0	braun
37	19		Frei				

Tabelle B.1: Pinbelegung und Umsetzung des DPA-DRV11J-Interface auf den PC-Anschluß

Der Digitale Positions-Analysator hat Steckverbindungen mit Sub-D Stecker nach DIN 41652 mit 37 Polen. Ausgehend von diesen ist die Umsetzung auf die 50 polige Stiftleiste gerade nach DIN 41651 angegeben. Die verschobenen Nummern geben die jeweilige Polleiste an. Die Masseverbindungen (GND) wurden über eine Masseleiste auf der Platine realisiert und sind nicht einzeln verkabelt. Die Masseverbindungen und weitere freie Pins der NI AT-DIO32F sind in der zweiten Tabelle angegeben. Zu Testzwecken sind die Datenleitungen über Schalter mit schwarzen Kabeln auf Masse gelötet und mit Meßfüßen versehen. Dadurch ist es möglich über die mit dem Treiber gelieferte Test-Software der NI AT-DIO32F entsprechende Muster auf den Datenleitungen zu erzeugen und einzulesen.

NI AT-DIO32F Pin	Belegung
17	GND
19	GND
21	GND
23	GND
25	GND
26	GND
28	GND
30	GND
32	GND
34	GND
1	Frei
2	Frei
3	Frei
5	Frei
6	Frei
7	Frei
8	Frei
9	Frei
11	Frei
15	Frei
16	Frei
18	Frei
20	Frei
22	Frei
24	Frei
29	Frei
31	Frei

Tabelle B.2: Weitere Pinbelegung der NI AT-DIO32F.

Achtung: Für den Betrieb der Platine und des Meßsystems ist es unerläßlich, daß alle Datenleitungen frei geschaltet sind. Die Schalter sind so verkabelt, daß in der Schalterstellung ‚ON' die Datenleitung auf Masse (GND) geschaltet sind. Für den Betrieb müssen sich alle Schalter in der Stellung ‚OFF' (Numerierungsseite) befinden.

2.2 Konfiguration der NI AT-DIO32F

Die folgende Tabelle enthält die Konfiguration der Datenports der NI AT-DIO32F.

Konfiguration	Art	Richtung
Gruppe 1 (A,B)	16 Datenbits	PC-Input
ACK1, REQ1	Hardware-Handshake	automatisch
Block C	Mode-Drehschalter und Integration Stop	PC-Input
Block D	Computer aktiv	PC-Output

Tabelle B.3: Blockkonfiguration der NI AT-DIO32F.

Anhang C

Programmfunktionen

In den folgenden Abschnitten werden die einzelnen Menüpunkte des Programms und ihre Funktionalität erklärt.

3.1 Menü des Hauptprogramms

Beim Programmstart und wenn keine Datenerfassung stattfindet, wird dieses Menü im Hauptfenster angezeigt.

3.1.1 Datei

1. *Daten Laden*: Ein Dialog-Fenster im Windows-Explorer Stil und alle im aktuellen Verzeichnis vorhandenen Header-Dateien der Messungen werden angezeigt. Der Benutzer kann über Maus- oder Tastatureingabe das Verzeichnis und die Datei auswählen und diese wird geladen und angezeigt. Es wird immer das Bild und die Impulshöhen-Verteilung angezeigt. Bei einer reinen Impulshöhen-Messung erscheint das Bild im Hauptfenster komplett schwarz, wohingegen bei einer reinen Bildmessung keine Datenlinie im Rahmen der gewählten Impulshöhen-Darstellung zu erkennen ist (diese deckt sich mit der Achse).

2. *Daten Speichern*: Das selbstdefinierte Dialog-Fenster für die Eingabe der Meßdaten (Name, Kommentar, Bemerkung und Meßdauer) wird angezeigt. Wird ein bereits existierender Name eingegeben, so werden diese Dateien überschrieben. Wird ein neuer Name angegeben, so werden die neuen Dateien erzeugt. Nach dem Speichern wird eine Erfolgs- oder Fehlermeldung in einer Message-Box ausgegeben.

3. *Bitmap Speichern*: Ein Dialog-Fenster im Windows-Explorer Stil und alle im aktuellen Verzeichnis vorhandenen Bitmap-Dateien werden angezeigt. Der Benutzer kann über Maus- oder Tastatureingabe das Verzeichnis und die Datei auswählen. Wird eine bereits existierende Datei gewählt, wird diese ohne Warnung überschrieben, andernfalls wird eine neue Datei erzeugt. Das Bitmap wird dabei mit einer Größe von 1024×1024 Bildpunkten erzeugt, unabhängig von der aktuellen Darstellungsgröße auf dem Bildschirm. Es wird der aktuelle angezeigte Farbverlauf (Schwarz-Weiß oder Gelb-Rot) und die aktuelle Farbumsetzung (linear oder logarithmisch jeweils mit optionaler Maximalwertorientierung) auch für das Bitmap verwendet, so daß die Darstellung im Bitmap, von der Größe abgesehen, mit der Darstellung auf dem Bildschirm identisch ist.

4. *Programm Beenden*: Das Programm wird beendet. Wurden nach einer Messung die Daten nicht gesichert, so gehen diese ohne Warnung beim Programmende verloren. Die Erzeugung neuer Daten und das Abspeichern ungesicherter Daten wird nicht protokolliert.

3.1.2 Daten

Die Integrationsmöglichkeiten werden ja nach Modus des DPA wählbar angezeigt. Die Modi des DPA sind nach ihrer Position an der Hardware numeriert. 1: Impulshöhe z, 2: Ortskoordinate xy, 3: Ortskoordinate mit Zeitmarke xyt, 4: Ortskoordinate, Impulshöhe und Zeitmarke xyzt. Ist der DPA nicht im entsprechenden Modus oder ausgeschalten, sind die Menüpunkte grau unterlegt und nicht wählbar. Nach Auswahl des Menüpunktes erscheint der selbst erstellte Meßdialog. Darin kann ein Dateiname, ein messungsspezifischer Kommentar, ein Bemerkung, die separat gespeichert (*‚Comment.dat‘*) wird, und in jedem weiteren Meßdialog bis zur Änderung so erscheint, und die Meßdauer laut DPA angegeben werden. Mit Tastenfeld ‚OK‘ wird die Messung gestartet und mit Tastenfeld ‚Abbruch‘ abgebrochen. Wird die Messung abgebrochen, erscheint wieder das Hauptfenster mit dem normalen Menü (siehe Anhang C.3.1) im Anfangszustand. Wird die Messung durchgeführt, wird auf das Integrationsmenü (siehe Anhang C.3.3) umgeschalten. Es erscheint eine Message-Box mit dem DPA-Modus und dem gewählten Integrations-Modus. An dieser Stelle kann der DPA-Modus noch geändert werden. Die Änderung des Integrations-Modus ist nicht mehr möglich. Durch das Tastenfeld ‚OK‘ wird die Messung gestartet. Nach dem Ende der Messung, entweder durch Menüwahl oder Drücken der Tasten ‚Integration-Stop‘ am DPA wird die Messung beendet. Es erscheint wieder der Meßdialog und es kann der endgültige Dateiname und die Meßdauer laut DPA eingegeben werden. Wird dieser Dialog durch das Tastenfeld ‚Abbruch‘ beendet, werden die Daten nicht gespeichert und der Benutzer muß durch den Menüpunkt ‚Daten Speichern‘ die Daten selbst sichern. Eine entsprechende Message-Box weist darauf nochmals hin. Bei Programmende ohne eigene Speicherung gehen die Daten sonst verloren.

Während der Integration werden die angezeigten Bilder und Impulshöhen-Verteilungen alle 10 Sekunden aktualisiert.

Die Impulshöhen-Verteilung wird normiert dargestellt. Zu diesem Zweck wird das Maximum in der Impulshöhen-Verteilung bestimmt. Ist ein Integrations-Modus des PC gewählt, in dem die Impulshöhe während der Messung angezeigt wird, so laufen während der Berechnung der normierten Impulshöhen-Verteilung weitere Daten auf und es kann vorkommen, daß dadurch Werte entstehen, die größer als eins sind und über den Rahmen der gewählten Darstellung hinaus angezeigt werden. Dabei wird die Verteilung stets mit richtigen Proportionen, aber eventuell nicht übereinstimmend mit der Abszissenachse, angezeigt. Dieser Effekt tritt speziell am Anfang einer Messung auf, wenn die Counts in den einzelnen Kanälen noch gering sind und relativiert sich mit zunehmender Meßdauer. Nach Ende einer Messung und bei geladenen Daten ist die Darstellung fehlerfrei.

1. *Bild integrieren (2,3,4)*: In den Modi 2,3 und 4 werden nur die Ortskoordinaten ausgewertet und in einem 1024×1024 Pixel großen Bitmap online angezeigt. Alle anderen Daten, die je nach Modus mit übertragen werden, bleiben unberücksichtigt.

2. *Impulshöhe integrieren (1,4)*: In den Modi 1 und 4 werden nur die Impulshöhen ausgewertet und die Verteilung in einem eigenen Fenster online angezeigt. Alle anderen Daten, die im Modus 4 mit übertragen werden, bleiben unberücksichtigt.

3. *Vollintegration (4)*: Im Modus 4 werden die Ortskoordinaten im einem 1024×1024 Pixel großen Bild und die Impulshöhen in einem eigenen Fenster online integriert und dargestellt. Die gleichzeitig übertragene Zeitmarke wird nicht berücksichtigt.

4. *Integration in File (3,4)*: Es erscheint ein eigener Dialog zur Integration in einer Datei. Darin kann der Dateiname, der Kommentar und eine Bemerkung angegeben werden. Die Messung wird nach Anzeige des DPA und Integrations-Modus durch Drücken des Tastenfeldes ‚OK‘ gestartet. Während der Messung wird das Fortschreiten durch Anzeige der gemessenen Blöcke angegeben. Die Datei-Integration macht nur Sinn mit Zeitmarke, also im Modus 3 und 4. Es werden die Daten roh, wie sie eingelesen werden, in eine Datei geschrieben. Nach dem

Ende der Messung wird der Meßdialog wieder angezeigt und die DPA Meßdauer kann eingetragen werden. Wird der Dialog mit dem Tastenfeld ‚Abbruch' beendet, gehen die Daten insofern verloren als daß keine Header-Datei mit den Meßinformationen erzeugt wird und die Integrations-Datei von der Aufarbeitungsroutine nicht gelesen werden kann. Wird der Dialog mit Tastenfeld ‚OK' beendet, wird der DPA Modus mitvermerkt und die Daten können später unter Angabe der Start- und Stopzeitmarke ausgewertet werden.

5. *Integrationsfile aufarbeiten*: Es erscheint ein Dialog im Windows-Explorer Stil und der Benutzer kann durch Maus- oder Tastatureingabe das Verzeichnis und die Datei anwählen. Es wird dann die erste und letzte Zeitmarke der Messung angezeigt und der Benutzer kann die gewünschten Start- und Stopzeitmarken und den gewünschten Dateinamen angeben. Danach startet die Aufarbeitung und es werden die entsprechenden Daten-Dateien mit dem eingegeben Namen angelegt. Wird ein bereits existierender Dateiname angegeben, so wird die alte Datei ohne Warnung durch die neue ersetzt und die alten Daten gehen verloren.

6. *Farbverlauf Schwarz-Weiß*: Für die Umsetzung des Farbindex wird eine Schwarz-Weiß-Farbtabelle im Bitmap verwendet. Dabei wird der Index 0 in die RGB-Werte $0, 0, 0$ (Schwarz) und der Index 255 entsprechend in die RGB-Werte $255, 255, 255$ (Weiß) umgesetzt. Die gewählte Darstellung wird durch einen Check ($\surd$) im Menü angezeigt. Die Umsetzung der Datenwerte in den Farbindex bestimmt die gewählte Darstellung und die optionale Maximalwertorientierung.

7. *Farbverlauf Gelb-Rot*: Für die Umsetzung des Farbindex wird eine Gelb-Rot-Farbtabelle im Bitmap verwendet. Dabei wird der Index 0 in die RGB-Werte $255, 255, 0$ (Gelb) und der Index 255 entsprechend in die RGB-Werte $255, 0, 0$ (Rot) umgesetzt. Die gewählte Darstellung wird durch einen Check ($\surd$) im Menü angezeigt. Die Umsetzung der Datenwerte in den Farbindex bestimmt die gewählte Darstellung und die optionale Maximalwertorientierung.

8. *Maximalwert orientiert*: Wird die Maximalwertorientierung gewählt, so wird bei jeder Aktualisierung bzw. Anzeige des Bildes eine neue Farbumsetzung berechnet. Dabei wird das Array nach dem Maximalwert durchsucht. Diesem Maximalwert wird der Farbindex 255 zugeordnet. Während der Messung kann sich im Laufe der Maximalwertbestimmung bzw. der Anzeige des neuen Bildes der Maximalwert ändern. Aus diesem Grund wird allen Werten, die größer als der aktuelle Maximalwert sind, ebenfalls der Farbindex 255 zugeordnet. Entsprechend der gewählten Farbumsetzung werden dann die Daten zwischen 0 und dem Maximalwert den Farbindizes von 0 bis 255 entweder linear oder logarithmisch zugeordnet. Ist die Maximalwertorientierung gewählt, so ist der Menüpunkt mit einem Check ($\surd$) versehen. Ist die Maximalwertorientierung abgeschaltet, so wird die Farbumsetzung mit dem absoluten Maximalwert von 65535 gerechnet.

9. *Lineare Farbumsetzung*: Zwischen 0 dem Maximalwert (aktuell oder absolut, je nach optionaler Maximalwertorientierung) wird eine lineare Umsetzung zwischen den 16 Bit Datenwerten und den 8 Bit Farbindizes berechnet. Zusätzlich wird auf ganze Zahlen abgeschnitten. Die gewählte Farbumsetzung wird durch einen Check ($\surd$) gekennzeichnet.

10. *Logarithmische Farbumsetzung*: Zwischen 0 dem Maximalwert (aktuell oder absolut, je nach optionaler Maximalwertorientierung) wird eine logarithmische (zur Basis 10) Umsetzung zwischen den 16 Bit Datenwerten und den 8 Bit Farbindizes berechnet. Die Formel dafür lautet dann:
$$Index = 255 \cdot \frac{\log(Datenwert)}{\log(Maximalwert)}.$$
Zusätzlich wird auf ganze Zahlen abgeschnitten. Die gewählte Farbumsetzung wird durch einen Check ($\surd$) gekennzeichnet.

11. *Impulshöhen-Verteilung*: Wurde das Impulshöhen-Fenster geschlossen, entweder durch das $\times$-Tastenfeld in der rechten oberen Ecke oder den entsprechenden Menüpunkt, so kann es durch diesen Menüpunkt wieder dargestellt werden. Bei Programmbeginn wird die Impulshöhen-Verteilung doppelt-logarithmisch angezeigt. Wurde die Darstellung im Programmablauf verändert, so wird bei der erneuten Anzeige, auch eventuell mit anderen Daten, die letzte gewählte Art dargestellt. Das Impulshöhen-Fenster ist ein Kindfenster des Hauptfensters und bleibt im Vordergrund relativ zum Hauptfenster und erscheint auch nicht auf der Taskleiste. Wird das Hauptfenster minimiert, so wird das Impulshöhen-Fenster mit minimiert. Bei der Wiederherstellung des Hauptfensters wird das Impulshöhen-Fenster wieder angezeigt, unabhängig davon, ob es zuvor minimiert war oder nicht. Wenn es geschlossen wurde, wird es nicht wieder hergestellt. Wurde eine reine Bildmessung durchgeführt, so erscheint im Impulshöhen-Fenster der Rahmen der gewählten Darstellung, ohne eine erkennbare Datenlinie (diese deckt sich mit der Achse).

12. *Löschen*: Die Daten werden gelöscht und ein eventuell vorhandenes Impulshöhen-Fenster geschlossen. Das Hauptfenster wird wieder in der Anfangsgröße angezeigt. Wurden die Daten nach einer vorangegangenen neuen Messung nicht gespeichert, so gehen sie verloren.

13. *Dateninfo*: Diese Message-Box wird nach dem Laden, nach Beendigung einer Messung oder durch Wahl dieses Menüpunktes angezeigt. Es werden folgende Daten angezeigt bzw. berechnet: Dateiname, Kommentar, Bemerkung, Integrationsstart laut Windows-Systemuhr, Integrationsstop laut Windows-Systemuhr, Integrationsdauer als Differenzwert der beiden vorangegangenen Angaben, totzeitkorrigierte Meßdauer (wenn diese bei Meßende im Dialog angegeben wurde), Anzahl der Counts im Array, Counts pro Sekunde unter Verwendung der Systemzeit (nicht der DPA Meßdauer), Maximalwert in Counts und Stellenangabe (Array-Nummer und x,y-Koordinate) und der Durchschnittswert im Array, also die Anzahl der Counts im Array dividiert durch die Anzahl der Bildpunkte ($1024 \times 1024 = 1.048.576$). Wurde eine reine Impulshöhen-Messung durchgeführt, so ist die Anzahl der Counts im Array 0 und alle daraus berechneten Werte, wie die Counts pro Sekunde, der Maximalwert und die durchschnittlichen Counts, ebenfalls. Bei einer reinen Impulshöhen-Messung lassen sich die Anzahl der Counts in der Impulshöhen-Verteilung und die Ordinate des Maximalwertes über den Menüpunkt ‚Dateninfo' im Impulshöhen-Fenster abrufen. Wurde die Impulshöhen-Verteilung und das Bild parallel gemessen, so müssen bei einer fehlerfreien Messung, die Counts im Bildarray und in der Impulshöhen-Verteilung übereinstimmen. Diese werden separat bestimmt.

3.1.3 Anzeigen

1. *Originalgröße*: Die gemessenen Daten werden im Originalformat (1024×1024 Bildpunkte) angezeigt. Ist das Fenster kleiner als das Bild, werden automatisch Bildlaufleisten eingeblendet, mit denen der entsprechende Bildausschnitt verschoben werden kann. Ist die Fit-Option aktiviert, so wird das Bild durch diesen Menüpunkt auf die Originalgröße gebracht. Würde dadurch das Fenster größer als der Bildschirmbereich, so wird das Fenster maximiert und das Bild auf diese Fenstergröße gefittet und entspricht dann unter Umständen nicht der Originalgröße. Eine eins zu eins Pixeldarstellung des Bildes auf dem Bildschirm ist nur ohne Fit-Option durch den Menüpunkt ‚Originalgröße' möglich.

2. *Fit*: Durch Auswahl dieser Option wird das angezeigte Bitmap auf die Fenstergröße eingepaßt ohne auf die Proportionen des Original Rücksicht zu nehmen. Die Option kann nur bei geladenen oder neu gemessenen Daten gewählt werden, ansonsten ist sie grau unterlegt und nicht aktivierbar. Ist die Option gewählt, so ist der Menüpunkt durch einen Check ($\surd$) gekennzeichnet. Werden die vorhandenen Daten gelöscht oder durch andere (geladen oder gemessen) ersetzt so wird die Fit-Option durch das Programm wieder zurückgesetzt. Wird durch den

Menüpunkt ‚Originalgröße‘ das Bitmap auf die ursprüngliche Größe zurückgesetzt und würde das Fenster dadurch größer als der Bildschirm, so wird das Fenster maximiert und das Bitmap auf diese Fenstergröße eingepaßt, was unter Umständen nicht der Originalgröße entspricht. Eine eins zu eins Pixeldarstellung des Bildes auf dem Bildschirm ist nur ohne Fit-Option durch den Menüpunkt ‚Originalgröße‘ möglich. Wird das Bild mit gesetzter Fit-Option stark verkleinert, so kann es aufgrund der Standard-Windows-Umsetzungsroutine der Pixelfarben zu Verzeichnungen des Bildes kommen.

3. *Bildinfo*: Es wird ein Message-Box mit Information über das Bild angezeigt: Dateiname, Originalgröße und Farbtiefe der Darstellung werden angegeben. Ist die Fit-Option gesetzt, wird auch die aktuelle Größe des Bildes angezeigt.

3.1.4 Fenster

Diese Menüpunkte beziehen sich auf die Steuerung der Bitmap-Fenster und das Hauptfenster. Alle Bitmap-Fenster werden zu Programmbeginn erzeugt, aber nicht angezeigt. Während der Integration werden diese Fenster tatsächlich komplett entfernt, um Ressourcen zu sparen, und nach Ende der Messung wieder neu erstellt. Ansonsten bewirkt ein Schließen nur ein Löschen des angezeigten Bitmaps, ein Rücksetzen der Überschrift und Löschen des sichtbaren Fensters von der Bildschirmoberfläche, ohne aber das eigentliche Windows-Objekt zu entfernen. Alle Bitmap-Fenster sind Kindfenster des Hauptfensters und sind immer oberhalb des Hauptfensters angeordnet. Die Bitmap-Fenster erscheinen dadurch auch nicht auf der Taskleiste des Systems. Das Hauptfenster bleibt immer im Hintergrund und kann nicht vor die Bitmap-Fenster geholt werden. Beim Minimieren des Hauptfensters werden auch alle Bitmap-Fenster mit minimiert. Bei der Wiederherstellung des Hauptfensters werden dann wieder alle geöffneten Bitmap-Fenster normal angezeigt, unabhängig davon, ob sie zuvor minimiert waren oder nicht. Das Impulshöhen-Fenster ist ein genauso ein Kindfenster, wie die Bitmap-Fenster, und diese können sich gegenseitig überlappen und überdecken.

1. *Fenster Öffnen*: Durch Anwahl dieses Menüpunktes wird ein neues Bitmap-Fenster geöffnet und angezeigt. Es können bis zu 5 Bitmap-Fenster geöffnet werden. Danach erscheint eine Message-Box, mit dem Hinweis, daß keine weiteren Bitmap-Fenster mehr geöffnet werden können.

2. *Alle Anordnen*: Diese Funktion bezieht sich auf das Hauptfenster und die Bitmap-Fenster, nicht auf ein eventuell angezeigtes Impulshöhen-Fenster. Dies bleibt von der Funktion unberührt. Es wird die Anzahl der Bitmap-Fenster bestimmt und das Hauptfenster und die Bitmap-Fenster so angeordnet und in der Größe geändert, daß alle angezeigten Fenster ganz zu sehen sind und der Bildschirmbereich optimal genutzt wird, ohne die Taskleiste zu verdecken. Eventuell gesetzte Fit-Optionen in den einzelnen Fenstern bleiben erhalten. Sind diese Optionen nicht gesetzt, so werden Bildlaufleisten angezeigt, wenn das angezeigte Bild größer ist als die neue Fenstergröße.

3. *Alle Kaskadieren*: Das Hauptfenster und alle angezeigten Bitmap-Fenster werden auf die Anfangsgröße gebracht und nacheinander, um die Höhe der Überschriftenleiste in Höhe und Breite, beginnend in der linken oberen Bildschirmecke, gegeneinander versetzt angezeigt. Eventuell gesetzte Fit-Optionen in den einzelnen Fenstern bleiben erhalten. Sind diese Optionen nicht gesetzt, so werden Bildlaufleisten angezeigt, wenn das angezeigte Bild größer ist als die neue Fenstergröße. Ein eventuell angezeigtes Impulshöhen-Fenster bleibt davon unberührt.

4. *Alle Minimieren*: Alle angezeigten Bitmap-Fenster werden minimiert und nacheinander links unten, oberhalb der Taskleiste, angeordnet. Ein eventuell angezeigtes Impulshöhen-Fenster bleibt davon unberührt. Das Hauptfenster wird gleichzeitig maximiert und die Anzeige entsprechend angepaßt.

5. *Alle Schließen*: Alle angezeigten Bitmap-Fenster werden von der Bildschirmoberfläche entfernt. Dabei wird der angezeigte Inhalt gelöscht und die Fensterüberschrift zurückgesetzt. Das eigentliche Fensterobjekt wird nicht entfernt, unabhängig von der Methode des Schließens.

3.1.5 Hilfe

1. *Programminfo*: Es wird eine Message-Box mit Informationen über das Programm angezeigt: „Dieses Programm wurde im Rahmen einer Diplomarbeit am Institut für Astronomie und Astrophysik, Abt. Astronomie, der Universität Tübingen entwickelt. Alexander Würz, 1999“.

3.2 Menü des Bitmap-Fensters

3.2.1 Datei

1. *Laden*: Ein Dialog-Fenster im Windows-Explorer Stil und alle im aktuellen Verzeichnis vorhandenen Bitmap-Dateien werden angezeigt. Der Benutzer kann über Maus- oder Tastatureingabe das Verzeichnis und die Datei auswählen und diese wird geladen und in Originalgröße angezeigt. Wird das Fenster dabei größer als der verfügbare Bildschirmbereich, so wird das Fenster maximiert und Bildlaufleisten angezeigt, mit denen der Bildausschnitt verschoben werden kann.

2. *Speichern*: Ein Dialog-Fenster im Windows-Explorer Stil und alle im aktuellen Verzeichnis vorhandenen Bitmap-Dateien werden angezeigt. Der Benutzer kann über Maus- oder Tastatureingabe das Verzeichnis und die Datei auswählen. Wird ein bereits vorhandener Name gewählt, so wird dieses Bitmap ohne Warnung überschrieben. Wird ein neuer Name angegeben, so wird das aktuelle Bitmap in Originalgröße unter diesem Namen abgelegt.

3. *Drucken*: Es erscheint der gewohnte Druckerdialog, in dem die entsprechenden Drucker im Netzwerk ausgewählt werden können. Der Fensterinhalt wird dann größenangepaßt auf dem jeweiligen Drucker mit den im Dialog gewählten Optionen ausgegeben.

4. *Schließen*: Das Bitmap-Fenster wird von der Bildschirmoberfläche entfernt. Dabei wird der angezeigte Inhalt gelöscht und die Fensterüberschrift zurückgesetzt. Das eigentliche Fensterobjekt wird nicht entfernt, unabhängig von der Methode des Schließens.

3.2.2 Bild

1. *Originalgröße*: Die gemessenen Daten werden im Originalformat (1024$\times$1024 Bildpunkte) angezeigt. Ist das Fenster kleiner als das Bild, werden automatisch Bildlaufleisten eingeblendet, mit denen der entsprechende Bildausschnitt verschoben werden kann. Ist die Fit-Option aktiviert, so wird das Bild durch diesen Menüpunkt auf die Originalgröße gebracht. Würde dadurch das Fenster größer als der Bildschirmbereich, so wird das Fenster maximiert und das Bild auf diese Fenstergröße gefittet und entspricht dann unter Umständen nicht der Originalgröße. Eine eins zu eins Pixeldarstellung des Bildes auf dem Bildschirm ist nur ohne Fit-Option durch den Menüpunkt ‚Originalgröße‘ möglich.

2. *Fit*: Durch Auswahl dieser Option wird das angezeigte Bitmap auf die Fenstergröße eingepaßt. Die Option kann nur bei geladenen oder neu gemessenen Daten gewählt werden, ansonsten ist sie grau unterlegt und nicht aktivierbar. Ist die Option gewählt, so ist der Menüpunkt durch einen Check ($\surd$) gekennzeichnet. Werden die vorhandenen Daten gelöscht oder durch andere (geladen oder gemessen) ersetzt so wird die Fit-Option durch das Programm wieder zurückgesetzt. Wird durch den Menüpunkt ‚Originalgröße‘das Bitmap auf die ursprüngliche Größe zurückgesetzt und würde das Fenster dadurch größer als der Bildschirm, so wird das

Fenster maximiert und das Bitmap auf diese Fenstergröße eingepaßt, was unter Umständen nicht der Originalgröße entspricht. Eine eins zu eins Pixeldarstellung des Bildes auf dem Bildschirm ist nur ohne Fit-Option durch den Menüpunkt ‚Originalgröße' möglich. Wird das Bild mit gesetzter Fit-Option stark verkleinert, so kann es aufgrund der Standard-Windows-Umsetzungsroutine der Pixelfarben zu Verzeichnungen des Bildes kommen.

3. *Löschen*: Das angezeigt Bitmap wird gelöscht, das Fenster wird wieder auf Anfangsgröße gebracht und die Fensterüberschrift wird zurückgesetzt.

4. *Bildinfo*: Es wird ein Message-Box mit Information über das Bild angezeigt: Dateiname, Originalgröße und Farbtiefe der Darstellung werden angegeben. Ist die Fit-Option gesetzt, wird auch die aktuelle Größe des Bildes angezeigt.

3.2.3 Fenster

Bitmap-Fenster können nur über das Hauptfenster geöffnet werden. Da die Fenster nur von der Bildschirmoberfläche entfernt werden und nicht das Objekt an sich gelöscht wird, ist das Schließen nicht eigentlich ein solches. Die Kindfenster löschen ihren Inhalt, setzen ihre Überschrift zurück und versenden an das Hauptfenster eine Botschaft, welches daraufhin das Bitmap-Fenster nur von der Oberfläche entfernt.

1. *Alle Anordnen*: Diese Funktion bezieht sich auf das Hauptfenster und die Bitmap-Fenster, nicht auf ein eventuell angezeigtes Impulshöhen-Fenster. Dies bleibt von der Funktion unberührt. Es wird die Anzahl der Bitmap-Fenster bestimmt und das Hauptfenster und die Bitmap-Fenster so angeordnet und in der Größe geändert, daß alle angezeigten Fenster ganz zu sehen sind und der Bildschirmbereich optimal genutzt wird, ohne die Taskleiste zu verdecken. Eventuell gesetzte Fit-Optionen in den einzelnen Fenstern bleiben erhalten. Sind diese Optionen nicht gesetzt, so werden Bildlaufleisten angezeigt, wenn das angezeigte Bild größer ist als die neue Fenstergröße.

2. *Alle Kaskadieren*: Das Hauptfenster und alle angezeigten Bitmap-Fenster werden auf die Anfangsgröße gebracht und nacheinander, um die Höhe der Überschriftenleiste in Höhe und Breite, beginnend in der linken oberen Bildschirmecke, gegeneinander versetzt angezeigt. Eventuell gesetzte Fit-Optionen in den einzelnen Fenstern bleiben erhalten. Sind diese Optionen nicht gesetzt, so werden Bildlaufleisten angezeigt, wenn das angezeigte Bild größer ist als die neue Fenstergröße. Ein eventuell angezeigtes Impulshöhen-Fenster bleibt davon unberührt.

3. *Alle Minimieren*: Alle angezeigten Bitmap-Fenster werden minimiert und nacheinander links unten, oberhalb der Taskleiste, angeordnet. Ein eventuell angezeigtes Impulshöhen-Fenster bleibt davon unberührt. Das Hauptfenster wird gleichzeitig maximiert und die Anzeige entsprechend angepaßt.

4. *Alle Schließen*: Alle angezeigten Bitmap-Fenster werden von der Bildschirmoberfläche entfernt. Dabei wird der angezeigte Inhalt gelöscht und die Fensterüberschrift zurückgesetzt. Das eigentliche Fensterobjekt wird nicht entfernt, unabhängig von der Methode des Schließens.

3.3 Menü des Hauptprogramms während der Integration

Dieses Menü wird anstelle des Hauptmenüs (siehe Anhang C.3.1) während der Datenerfassung angezeigt.

3.3.1 Einstellungen

1. *Integration beenden*: Die Integration wird mit Warnton beendet. Es erscheint der Meßdialog und es kann der endgültige Dateiname und die DPA-Meßdauer angegeben werden. Wird der Dialog mit ‚OK‘ beendet, werden die Daten gespeichert, oder bei Angabe eines bereits existierenden Dateinamens diese Datei überschrieben. Es erscheint eine Erfolgs- oder Fehlermeldung über das Speichen der Daten. Wird der Dialog mit ‚OK‘ beendet, so werden die Daten nicht gespeichert und gehen beim Programmende verloren. Darauf wird in einer Message-Box nochmals hingewiesen. Dann wird das Bild und eventuell die Impulshöhen-Verteilung erneut berechnet und angezeigt. Es erscheint die Message-Box mit den Dateninformationen, die ansonsten über das Hauptmenü oder das hier im weiteren beschriebene Integrationsmenü aufgerufen werden kann.

2. *Farbverlauf Schwarz-Weiß*: Für die Umsetzung des Farbindex wird eine Schwarz-Weiß-Farbtabelle im Bitmap verwendet. Dabei wird der Index 0 in die RGB-Werte 0,0,0 (Schwarz) und der Index 255 entsprechend in die RGB-Werte 255,255,255 (Weiß) umgesetzt. Die gewählte Darstellung wird durch einen Check ($\surd$) im Menü angezeigt. Die Umsetzung der Datenwerte in den Farbindex bestimmt die gewählte Darstellung und die optionale Maximalwertorientierung.

3. *Farbverlauf Gelb-Rot*: Für die Umsetzung des Farbindex wird eine Gelb-Rot-Farbtabelle im Bitmap verwendet. Dabei wird der Index 0 in die RGB-Werte 255,255,0 (Gelb) und der Index 255 entsprechend in die RGB-Werte 255,0,0 (Rot) umgesetzt. Die gewählte Darstellung wird durch einen Check ($\surd$) im Menü angezeigt. Die Umsetzung der Datenwerte in den Farbindex bestimmt die gewählte Darstellung und die optionale Maximalwertorientierung.

4. *Maximalwert orientiert*: Wird die Maximalwertorientierung gewählt, so wird bei jeder Aktualisierung bzw. Anzeige des Bildes eine neue Farbumsetzung berechnet. Dabei wird das Array nach dem Maximalwert durchsucht. Diesem Maximalwert wird der Farbindex 255 zugeordnet. Während der Messung kann sich im Laufe der Maximalwertbestimmung bzw. der Anzeige des neuen Bildes der Maximalwert ändern. Aus diesem Grund wird allen Werten, die größer als der aktuelle Maximalwert sind, ebenfalls der Farbindex 255 zugeordnet. Entsprechend der gewählten Farbumsetzung werden dann die Daten zwischen 0 und dem Maximalwert den Farbindizes von 0 bis 255 entweder linear oder logarithmisch zugeordnet. Ist die Maximalwertorientierung gewählt, so ist der Menüpunkt mit einem Check ($\surd$) versehen. Ist die Maximalwertorientierung abgeschaltet, so wird die Farbumsetzung mit dem absoluten Maximalwert von 65535 gerechnet.

5. *Lineare Farbumsetzung*: Zwischen 0 dem Maximalwert (aktuell oder absolut, je nach optionaler Maximalwertorientierung) wird eine lineare Umsetzung zwischen den 16 Bit Datenwerten und den 8 Bit Farbindizes berechnet. Die gewählte Farbumsetzung wird durch einen Check ($\surd$) gekennzeichnet.

6. *Logarithmische Farbumsetzung*: Zwischen 0 dem Maximalwert (aktuell oder absolut, je nach optionaler Maximalwertorientierung) wird eine logarithmische (zur Basis 10) Umsetzung zwischen den 16 Bit Datenwerten und den 8 Bit Farbindizes berechnet. Die Formel dafür lautet dann:
$$Index = 255 \cdot \frac{\log(Datenwert)}{\log(Maximalwert)}.$$
Zusätzlich wird auf ganze Zahlen abgeschnitten. Die gewählte Farbumsetzung wird durch einen Check ($\surd$) gekennzeichnet.

7. *Impulshöhen-Verteilung*: Wurde das Impulshöhen-Fenster geschlossen, entweder durch das $\times$-Tastenfeld in der rechten oberen Ecke oder den entsprechenden Menüpunkt, so kann

es durch diesen Menüpunkt wieder dargestellt werden. Bei Programmbeginn wird die Impulshöhen-Verteilung doppelt-logarithmisch angezeigt. Wurde die Darstellung im Programablauf verändert, so wird bei der erneuten Anzeige, auch eventuell mit anderen Daten, die letzte gewählte Art dargestellt. Das Impulshöhen-Fenster ist ein Kindfenster des Hauptfensters und bleibt im Vordergrund relativ zum Hauptfenster und erscheint auch nicht auf der Taskleiste. Wird das Hauptfenster minimiert, so wird das Impulshöhen-Fenster mit minimiert. Bei der Wiederherstellung des Hauptfensters wird das Impulshöhen-Fenster wieder angezeigt, unabhängig davon, ob es zuvor minimiert war oder nicht. Wenn es geschlossen wurde, wird es nicht wieder hergestellt. Wurde eine reine Bildmessung durchgeführt, so erscheint im Impulshöhen-Fenster der Rahmen der gewählten Darstellung, ohne eine erkennbare Datenlinie (diese deckt sich mit der Achse).

8. *Originalgröße*: Die gemessenen Daten werden im Originalformat (1024$\times$1024 Bildpunkte) angezeigt. Ist das Fenster kleiner als das Bild, werden automatisch Bildlaufleisten eingeblendet, mit denen der entsprechende Bildausschnitt verschoben werden kann. Ist die Fit-Option aktiviert, so wird das Bild durch diesen Menüpunkt auf die Originalgröße gebracht. Würde dadurch das Fenster größer als der Bildschirmbereich, so wird das Fenster maximiert und das Bild auf diese Fenstergröße gefittet und entspricht dann unter Umständen nicht der Originalgröße. Eine eins zu eins Pixeldarstellung des Bildes auf dem Bildschirm ist nur ohne Fit-Option durch den Menüpunkt ‚Originalgröße‘ möglich.

9. *Fit*: Durch Auswahl dieser Option wird das angezeigte Bitmap auf die Fenstergröße eingepaßt. Die Option kann nur bei geladenen oder neu gemessenen Daten gewählt werden, ansonsten ist sie grau unterlegt und nicht aktivierbar. Ist die Option gewählt, so ist der Menüpunkt durch einen Check ($\sqrt{}$) gekennzeichnet. Werden die vorhandenen Daten gelöscht oder durch andere (geladen oder gemessen) ersetzt so wird die Fit-Option durch das Programm wieder zurückgesetzt. Wird durch den Menüpunkt ‚Originalgröße‘ das Bitmap auf die ursprüngliche Größe zurückgesetzt und würde das Fenster dadurch größer als der Bildschirm, so wird das Fenster maximiert und das Bitmap auf diese Fenstergröße eingepaßt, was unter Umständen nicht der Originalgröße entspricht. Eine eins zu eins Pixeldarstellung des Bildes auf dem Bildschirm ist nur ohne Fit-Option durch den Menüpunkt ‚Originalgröße‘ möglich. Wird das Bild mit gesetzter Fit-Option stark verkleinert, so kann es aufgrund der Standard-Windows-Umsetzungsroutine der Pixelfarben zu Verzeichnungen des Bildes kommen.

10. *Dateninfo*: Diese Message-Box wird nach dem Laden, nach Beendigung einer Messung oder durch Wahl dieses Menüpunktes angezeigt. Es werden folgende Daten angezeigt bzw. berechnet: Dateiname, Kommentar, Bemerkung, Integrationsstart laut Windows-Systemuhr, Integrationsstop laut Windows-Systemuhr, Integrationsdauer als Differenzwert der beiden vorangegangenen Angaben, totzeitkorrigierte Meßdauer (wenn diese bei Meßende im Dialog angegeben wurde), Anzahl der Counts im Array, Counts pro Sekunde unter Verwendung der Systemzeit (nicht der DPA-Meßdauer), Maximalwert in Counts und Stellenangabe (Array-Nummer und x,y-Koordinate) und der Durchschnittswert im Array, also die Anzahl der Counts im Array dividiert durch die Anzahl der Bildpunkte (1024$\times$1024=1048576). Wurde eine reine Impulshöhen-Messung durchgeführt, so ist die Anzahl der Counts im Array 0 und alle daraus berechneten Werte, wie die Counts pro Sekunde, der Maximalwert und die durchschnittlichen Counts, ebenfalls. Bei einer reinen Impulshöhen-Messung lassen sich die Anzahl der Counts in der Impulshöhen-Verteilung und die Ordinate des Maximalwertes über den Menüpunkt ‚Dateninfo‘ im Impulshöhen-Fenster abrufen. Wurde die Impulshöhen-Verteilung und das Bild parallel gemessen, so müssen bei einer fehlerfreien Messung, die Counts im Bildarray und in der Impulshöhen-Verteilung übereinstimmen. Diese werden separat bestimmt.

3.4 Menü des Impulshöhen-Fensters

3.4.1 Darstellung

1. *Bitmap Speichern*: Ein Dialog-Fenster im Windows-Explorer Stil und alle im aktuellen Verzeichnis vorhandenen Bitmap-Dateien werden angezeigt. Der Benutzer kann über Maus- oder Tastatureingabe das Verzeichnis und die Datei auswählen. Wird eine bereits existierende Datei gewählt, wird diese ohne Warnung überschrieben, andernfalls wird eine neue Datei erzeugt. Da das Bitmap für die jeweilige Fenstergröße neu berechnet wird, um die Auflösungsvorteile zu nutzen, was bei einer einfachen Vergrößerung durch Streckung nicht der Fall ist, wird das Bitmap in der aktuell angezeigten Größe und im aktuellen Darstellungs-Modus mit 24 Bit Farbtiefe abgespeichert.

2. *Lineare Darstellung*: In diesem Modus wird die Impulshöhen-Verteilung mit einer linearen Skala dargestellt. Entlang der x-Achse ist der Bereich der AD-Wandler von 0 bis 10 Volt aufgetragen. Die Abszisse zeigt die auf den Maximalwert der Verteilung normierte Impulshöhe. Die Impulshöhen-Verteilung sollte einer Gaußschen Verteilung entsprechen (siehe Abschnitt 6.2). Dies läßt sich in dieser Darstellung erkennen.

3. *Logarithmische Darstellung*: In diesem Modus wird die Impulshöhen-Verteilung in einer doppelt logarithmischen Darstellung angezeigt. Entlang der x-Achse ist der AD-Wandler Bereich von $0,01$ bis 10 V aufgetragen. Die Abszisse zeigt die auf den Maximalwert der Verteilung normierte Impulshöhe oberhalb von $0,001$. Die Lage des Maximums der Impulshöhen-Verteilung ist ein Maß für die Verstärkung des Detektors (siehe Abschnitt 6.2). Wird diese geändert verschiebt sich demzufolge der Maximalwert und auch die Form der Verteilung in der linearen Darstellung. Damit ist ein Vergleich stark erschwert. In der doppelt logarithmischen Darstellung hingegen bleibt trotz der Maximalwertverschiebung die Form der Verteilung erhalten und bei einer Änderung des Arbeitspunktes kann die Funktionalität des Detektors anhand der Impulshöhen-Verteilung kontrolliert werden.

4. *Dateninfo*: Es wird eine Message-Box mit der Anzahl der Counts und dem Ordinatenwert des Maximalwertes angezeigt. Wurde eine reine Impulshöhen-Verteilung gemessen, so ist dies die einzige Möglichkeit, die Zahl der Counts in der Messung anzuzeigen. Bei einer simultanen Messung eines Bildes und einer Impulshöhen-Verteilung sollte dieser Wert, bei einer fehlerfreien Integration, mit der Anzahl der Counts im Bild identisch sein. Die Lage des Maximalwertes gibt Auskunft über die Verstärkung des Detektors (siehe Abschnitt 6.2).

5. *Schließen*: Durch Schließen des Impulshöhen-Fensters wird dies nur von der Oberfläche entfernt. Das Objekt an sich wird nicht gelöscht. Durch die Menüpunkte ‚Impulshöhen-Verteilung' im Haupt- oder Integrationsmenü läßt es sich wieder anzeigen.

Anhang D

Datenklassen

Im folgenden wird die Struktur und der Aufbau der einzelnen Datenklassen näher beschrieben. Zum Programm gehören die folgenden Komponenten, wobei die Unterpunkte die Stelle der Einbindung bezeichnen. Bei der Einbindung muß auf die Reihenfolge geachtet werden, die im Quellcode des Programms angegeben ist, da manche Unterklassen auf andere zugreifen und diese dann bereits deklariert sein müssen. Aufgrund des Umfangs von etwa 6000 Zeilen eigenem Programmcode wurde auf die Auflistung verzichtet und nur die Struktur mit den wichtigsten Details der Schnittstellen angegeben. Ausführliche Kommentare zu jeder Codezeile, den Funktionen mit ihren Parametern und den Variablen sind im Programmquellcode enthalten. Grundstrukturen und Standardfunktionen finden sich beispielsweise in Louis (1996a), Louis (1996b), Ezzell (1993) oder Petzold (1996) zu finden sind.

Die *kursiv* geschriebenen Datenklassen führen den eigentlichen Hardware-Zugriff durch. Es werden teilweise noch weitere Standard-Windows-Dateien eingebunden. Diese sind alle mit ihrem jeweiligen Verwendungszweck in der Datei Erfassung.cpp angegeben.

- Erfassung.exe
  1. Nidex32b.lib
  2. Nidaq32b.lib
  3. Erfassung.rc
     (a) Ait_logo.bmp
     (b) Ait.ico
  4. Erfassung.cpp: Client-Klasse.
     (a) Erfassung.def
     (b) Ausdruck.h: Ausdruck-Klasse.
     (c) *Portc.h*: PortC-Klasse.
     (d) Bitmap.h: MyBitmapWindow-Klasse.
     (e) Dialog.h: MeinDialog-Klasse.
     (f) *Datenarray.h*: Datenarray-Klasse.
     (g) Imp-win.h: MyImpulsWindow-Klasse.

Die hier angeführte Struktur findet sich in den folgenden Abschnitten wieder.

Alle Kontroll-Variablen jeder Fensterklasse sind in einer Kontroll-Struktur zusammengefaßt. Diese ist global definiert und somit für jede Klassenfunktion erreichbar. In der jeweiligen Funktion werden die Parameter gesetzt und dann mit Hilfe von Invalidate() eine WM_Paint-Botschaft verschickt. Damit wird die Paint()-Routine der jeweiligen Fensterklasse aufgerufen und dort wird anhand der gesetzten Parameter die geeignete Funktion aufgerufen. Auf diese Art ist sicher gestellt, daß immer die gewählte Darstellung angezeigt wird, auch wenn der Benutzer das Fenster minimiert und wiederherstellt oder verschiebt und gleichzeitig nicht in jeder Routine die gleichen Abfragen und Aufrufe programmiert sind. Die Paint()-Routine ist eine Standard-Funktionen die entsprechend überschrieben und erweitert wurde.

Zur Kommunikation zwischen den einzelnen Datenklassen wurden teilweise eigne Windows-Botschaften erstellt. Dies läuft nach folgendem Prinzip ab: Die versendende Klasse deklariert eine Variable vom Typ *integer* mit dem Namen der Botschaft und meldet sie dann bei System über die Funktion *RegisterWindowMessage(Botschaftsname)* an. Die empfangende Klasse hat dem Namen der Botschaft in der Antworttabelle über *EV_REGISTERED(Botschaftsname,Funktion)* eine Funktion zugeordnet, die die entsprechende Aktion dann ausführt. Auf diese Art wird die empfangenden Klasse aufgefordert ihre geschützten Daten zu ändern. Jede Klasse ist dazu nur selbst in der Lage und damit ist eine geregelter Ablauf gesichert. Wird die empfangende Klasse ausgetauscht, so muß die neue nur ebenfalls eine sinngemäße Botschaftszuordnung zu einer Funktion haben. Wie die Umsetzung dann im einzelnen innerhalb der Klasse geschieht, ist für alle anderen Klassen bedeutungslos. Darin liegt der große Vorteil der objektorientierten Programmierung.

4.1 nidex32b.lib

Diese Bibliothek wurde mit dem Treiber der NI AT-DIO32F mitgeliefert. Sie beinhaltet Routinen zur Fehlererkennung und -behandlung.

4.2 nidaq32b.lib

Diese Bibliothek wurde mit dem Treiber der NI AT-DIO32F mitgeliefert. Sie beinhaltet Routinen zum konfigurierten Zugriff auf die NI AT-DIO32F.

4.3 Erfassung.rc

Diese Ressourcen-Datei enthält die Deklaration und Definition der Menüs, der verwendeten Bitmaps und der Dialoge.

Menüzuordnung:

1. MENU_1: Menü des Hauptfensters.
2. MENU_2: Menü des Bitmap-Fensters.
3. MENU_3: Menü des Hauptfensters während der Integration.
4. MENU_4: Menü des Impulshöhen-Fensters.

Bitmapzuordnung:

1. STARTBILD: Bitmap des Bitmap-Fenster, wenn kein anderes Bild dargestellt wird: ait_logo.bmp.
2. MEIN_ICON: Icon des Programm im Windows-Explorer: ait.ico.

Dialogzuordnung:

1. DIALOG_1: Meßdialog vor und nach der Integration.
2. DIALOG_2: Dialog für die Aufarbeitungsroutine der Datei-Integration.

4.3.1 Ait_logo.bmp

Datei mit dem Bitmap des AIT-Logos. Dieses wird in den Bitmap-Fenstern (Bitmap.h) angezeigt, wenn kein anderes Bild geladen ist.

4.3.2 Ait.ico

Datei mit einem Teil des AIT-Logos im Icon-Format. Dieses wird im Windows-Explorer und auf dem Desktop als Programmicon verwendet.

4.4 Erfassung.cpp

Alle anderen Datenklassen sind dieser untergeordnet und die gesamte Verwaltung der Abläufe geschieht innerhalb dieses Programmcodes. Hier sind alle Header-Dateien in entsprechender Reihenfolge angegeben. Diese rufen teilweise weitere auf, die dann aber entweder Standard sind oder mit dem Treiber für die Ein-/Ausgabe-Karte geliefert wurden. Es gibt keine andere Stelle an der Header-Dateien eingebunden wurden. Die folgenden Standard-Header-Dateien werden am Anfang des Quellcodes eingelesen:

1. windows.h: Grundlagen für jedes Windows-Programm, wie etwa Botschaftsverarbeitung.
2. owl\framewin.h: TFrameWindow für das Rahmenfenster.
3. owl\applicat.h: TApplication für die Anwendungsklasse.
4. owl\owlpch.h: Standard-Routinen für die Object Windows Library (OWL).
5. classlib\thread.h: TThread für die Thread-Programmierung.
6. owl\opensave.h: TFileOpenDialog für das Laden und Speichern von Dateien. Standard Ein- und Ausgabe. (fopen etc.)
7. errno.h: File-Handling, wie etwa Fehlerbehandlung beim Öffnen der Dateien.
8. fcntl.h, sys/stat.h: Deklaration der Öffnungsmodi bei binärem Datei-Zugriff.
9. stdio.h, io.h, iostream.h: Ein- und Ausgaberoutinen.
10. string.h: Verarbeitung von Strings.
11. owl\scroller.h: TScroller zur Verwendung von Bildlaufleisten.
12. classlib\date.h, classlib\time.h: TDate zur Datums- und TTime zur Meßdauerberechnung.
13. math.h: Berechnung von Funktionen für Farbverläufe.
14. owl\dialog.h: Header-Dateien für die Verwendung von Dialogboxen.
15. owl\edit.h: Editierbare Dialogfelder.
16. owl\validate.h: Überprüfung der Eingaben im Dialog zur Fehlerminimierung. Damit werden beispielsweise Sonderzeicheneingaben im Dateinamensfeld verhindert.
17. owl\printer.h, owl\dc.h: Drucker-Zugriff.
18. owl\printer.rc: Ressourcen-Datei für den Standard-Windows-Druckerdialog.

19. nidaqex.h: Zugriff auf National Instruments AT-DIO32F Karte. Diese Datei wurde mit dem Treiberpaket geliefert.

In dieser Klasse wird das Bitmap, mit allen Unterfunktionen, wie der Farbtabelle oder den Lookup-Tables, die untergeordneten Fenster und die Kommunikation mit dem System verwaltet. Zur Kommunikation wurden einige eigene Botschaften erstellt. Von der Client-Klasse, abgeleitet von TWindow und eigentliches Hauptprogramm, werden die folgenden Botschaften verstanden:

- CM_HIDEBITMAPWINDOW wird gesendet von der Datenarray-Klasse mit dem Index des Bitmap-Fenster. Dieses wird darauf von der Oberfläche entfernt.
- CM_ARRANGEBITMAPWINDOW wird gesendet von der Datenarray-Klasse. Daraufhin werden alle angezeigten Bitmap-Fenster und das Hauptfenster größenangepaßt neben- und untereinander dargestellt.
- CM_CASCADEBITMAPWINDOW wird gesendet von der Datenarray-Klasse. Daraufhin werden alle angezeigten Bitmap-Fenster und das Hauptfenster in der Startgröße, um die Höhe der Überschriftenleiste gegeneinander in Breite und Höhe verschoben, angezeigt.
- CM_MINIMIZEBITMAPWINDOW wird gesendet von der Datenarray-Klasse. Daraufhin werden alle angezeigten Bitmap-Fenster minimiert und das Hauptfenster maximiert.
- CM_CLOSEBITMAPWINDOW wird gesendet von der Datenarray-Klasse. Daraufhin wird an alle angezeigten Bitmap-Fenster eine CM_DELETEBITMAP Botschaft (siehe Anhang D.4.4) gesendet, worauf diese ihr angezeigtes Bitmap löschen und wieder das AIT-Logo anzeigen. Dann werden alle Bitmap-Fenster von der Oberfläche entfernt.
- CM_HARDWARESTOP wird gesendet von der PortC-Datenklassen, wenn die ‚Integrations-Stop'-Taste am DPA gedrückt wird. Daraufhin wird durch Aufruf der IntegrationStop()-Funktion die Datenerfassung beendet.
- CM_HIDEIMPULSWINDOW wird gesendet von der MyImpulsWindow-Klasse wenn der Benutzer das Fenster schließt. Daraufhin wird das Fenster nur von der Oberfläche entfernt, das Windows-Objekt selbst aber nicht gelöscht.

Gesendet werden die folgenden Botschaften:

- CM_FENSTERNUMMER wird nach dem Erzeugen an die MyBitmapWindow-Klasse gesendet. In der Parameterliste wird der Index übergeben, so daß das Bitmap-Fenster bei Versenden der CM_HIDEBITMAPWINDOW diesen wieder zurückgeben kann und das richtige Bitmap-Fenster von der Oberfläche entfernt wird.
- CM_DELETEBITMAP wird nach dem Erhalt einer CM_CLOSEBITMAPWINDOW Botschaft an alle angezeigten Bitmap-Fenster geschickt, die daraufhin ihre angezeigten Bitmaps löschen und wieder das AIT-Logo anzeigen.

Öffentliche Funktionen sind in dieser Klasse nicht vorhanden, da sie das Hauptprogramm darstellt und die Unterklasse nicht direkt darauf zugreifen.

Alle Datenarrays werden in *Datenarray.h* verwaltet. Um die Bilddaten in Farben umsetzten zu können müssen diese an die Client-Klasse übergeben werden. Das geschieht mit Hilfe der öffentlichen MyBitmapWindow-Klassenfunktion xy-zeiger(), die einen Zeiger auf den Anfang des Arrays zurück gibt.

Das gleiche Prinzip wird mit der MyImpulsWindow-Klasse angewandt. Das Array wird ebenfalls in *Datenarray.h* verwaltet, um einen möglichst direkten Zugriff auf das Array während der Integration zu haben. Die MyBitmapWindow-Klasse wird zuerst initialisiert und dann kann über die

MyBitmapWindow-Klassenfunktion z-zeiger() der Speicherbereich abgefragt werden. Dieser wird bei der Erzeugung des Impulshöhen-Fensters an die Klasse übergeben und diese kann über den Zeiger auf die Impulshöhen-Daten zugreifen und diese als Bitmap anzeigen.

Die timergesteuerte Aktualisierung der Darstellungen sowohl des Bildes als auch der Impulshöhen-Verteilung geschieht in dieser Klasse. Dazu werden entsprechend Funktionen in der eigenen Client-Klasse und in der MyImpulsWindow-Klasse aufgerufen.

4.4.1 Erfassung.def

Hier werden alle Programmkonstanten definiert. Das sind die Puffergrößen für das Einlesen der Daten, die Masken für die Demultiplex-Operationen der Datenworte, Größen der Fenster, Größen und Farben der Bitmaps und Erweiterungen der erzeugten Daten-Dateien.

- BUFFERSIZE=2000: Größe des gesamten Datenpuffers.
- ITEMS=1000: Anzahl der einzulesenden Datensätze.
- stopmask=$0x80$: Bit für die ‚Integrations-Stop'-Taste am DPA.
- funcmask=$0x07$: Eingestellter Integrations-Modus am DPA.
- wgbmask=$0x8000$: Maske für das Wortgruppenbit (WGB).
- ymask1=$0x7F98$: y-Werte aus Wort 1.
- xmask1=$0x0003$: x-Werte aus Wort 1.
- tmask1T18=$0x0004$: t-Werte aus Wort 1.
- tmask1T20T19=$0x0060$: t-Werte aus Wort 1.
- ymask2=$0x1800$: y-Werte aus Wort 2.
- xmask2=$0x07FE$: x-Werte aus Wort 2.
- tmask2T15=$0x0001$: t-Werte aus Wort 2.
- tmask2T17T16=$0x6000$: t-Werte aus Wort 2.
- tmask3=$0x7FFF$: t-Werte aus Wort 3.
- zmask4=$0x3FFF$: z-Werte aus Wort 4.
- STARTBREITE=400: Gesamte Breite des Fensters beim Programmstart.
- STARTHOEHE=300: Gesamte Höhe des Fensters beim Programmstart.
- RAHMENBREITE=8: Gesamte Rahmenbreite des Fensters auf beiden Seiten.
- RAHMENHOEHE=46: Höhe des Fensterrahmens an der Ober- und Unterseite inklusive Fensterüberschrift und Menüleiste.
- STATUSHOEHE=0: Es wird im Fenster keine Statuszeile verwendet.
- PROGRAMMNAME=‚Datenerfassung': Name des Programms.
- DATENNAME=‚Untitled': Name für Dateien, wenn kein eigenen Name angegeben wird.
- BITMAPBREITE=1024: Breite in Pixeln.

- BITMAPHOEHE=1024: Höhe in Pixeln.
- BILDPUNKTE=BITMAPBREITE*BITMAPHOEHE: Größe des Bildarrays.
- FARBANZAHL=256: Anzahl der Farben für das selbsterstellte Bitmap.
- Bitmap-Fenster=5: Anzahl der gleichzeitig aufrufbaren Bitmap-Fenstern.
- HEADERFILE=‚.hdr': Daten der Messung als einfacher Text.
- BILDFILE=‚.bil': Bilddaten im Binärformat ohne Header.
- IMPULSFILE=‚.phv': Impulshöhen-Daten im Binärformat.
- DATENFILE=‚.int': Integrations-Datei bei direkter Datei-Integration.

4.4.2 Ausdruck.h

Diese Klasse stellt den bekannten Druckerdialog von Microsoft Windows zur Verfügung. Damit läßt sich der aktuell angezeigt Bildinhalt in aktueller Größe auf dem ausgewählten Drucker ausgeben. Momentan wird ist diese Klasse das Bitmap-Fenster (MyBitmapWindow-Klasse) eingebunden. Von dieser Klasse werden keine Botschaften versendet oder empfangen, da die Klasse von TPrintOut (Standard Drucker Klasse der OWL) abgeleitet ist und nicht von einer Fensterklasse. Damit hat sie auch keine eigene Botschaftsverarbeitung.

4.4.3 Portc.h

Diese Klasse ist als eigenständiger Thread implementiert und überwacht den DPA. Der Thread wird bei Programmstart erzeugt, aber noch nicht gestartet. Die Klasse erhält bei der Erzeugung einen Zeiger auf das Hauptfenster und kann dadurch Botschaften an das Hauptfenster versenden. Es können keine Botschaften empfangen werden, da die Klasse nicht von einer Fensterklasse abgeleitet ist und damit keine eigene Botschaftsverarbeitung hat.

Es wird folgende Konfiguration der NI AT-DIO32F vorgenommen:

- Port C wird als Input-Port ohne Handshaking (imode=0) konfiguriert. Es werden immer alle 8 Bit eingelesen.
- Von den 8 Linien werden 4 benutzt. 3 Linien (C0-C2) ergeben die 3 Bit des Modus-Drehschalter (gefiltert aus den gesamten 8 Bit mit der Maske *funcmask*, siehe Anhang D.4.4.1) und 1 Bit (C7) überwacht die ‚Integration-Stop'-Taste des DPA (gefiltert aus den gesamten 8 Bit mit der Maske *stopmask*).

Die Statusleitungen des DPA sind über die Adapterplatine auf Port C gelegt. Dieser ist hier als Input konfiguriert. Die einzige eigene öffentliche Funktion ist *return_mode()*. Wird diese vom Hauptprogramm aufgerufen, so wird der Drehschalter des DPA abgefragt, durch die Maske *funcmask* (siehe Anhang D.4.4.1) der Modus bestimmt und dieser an das Hauptprogramm zurückgegeben. Entsprechend des eingestellten Modus können dann im Hauptprogramm nur die Integrationsmodi gestartet werden, die mit dem gewählten Betriebs-Modus des DPA übereinstimmen.

Wird im Hauptprogramm die Datenerfassung begonnen, so wird auch der Thread durch Aufruf der überschriebenen öffentlichen Standard-Funktion *Run()* gestartet. Dieser überwacht nun eigenständig die ‚Integration-Stop'-Taste des DPA durch Abfrage der Statusleitung und Extraktion durch die Maske *stopmask* (siehe Anhang D.4.4.1). Wird die Taste gedrückt, so versendet der Thread die Botschaft CM_HARDWARESTOP an das Hauptprogramm und die Integration wird daraufhin beendet. Der Thread wird dann angehalten und die Klasse gelöscht, da Threads die einmal gestartet wurden nicht nochmal begonnen werden können. Nach dem Löschen wird ein neuer Thread erzeugt, aber nicht gestartet. So ist die ständige Überwachung des DPA und wiederholte Messungen möglich.

Bei Veränderung der Hardware, etwa dem Einsatz einer neuen Input-/Output-Karte anstelle der NI AT-DIO32F, muß diese Klasse durch eine neue ersetzt werden, die die gleiche Schnittstelle und Botschaftsverarbeitung bietet.

4.4.4 Bitmap.h

Diese Klasse ist eine eigenständige Fensterklasse mit eigener Botschaftsverarbeitung. Wie oben erwähnt, wird zur Steuerung des Fensters ebenfalls eine Kontrollstruktur verwendet, die alle wichtigen Parameter des Fensters enthält. Wie im Hauptfenster (*Erfassung.cpp*), werden von den einzelnen Unterfunktionen nur dieses Parameter gesetzt und dann die *Paint()*-Routine aufgerufene, die anhand dieser Parameter die Darstellung umsetzt. Durch einbinden in eine Applikation (TApplication) kann daraus ein eigenes Programm gemacht werden. Die Klasse ist zur Darstellung von Bitmaps gedacht. Dies geschieht in Originalgröße mit eventuelle Bildlaufleisten oder angepaßt auf die Fenstergröße. Es wird die Printer-Klasse aus *Ausdruck.h* eingebunden, um die angezeigten Bitmaps ausdrucken zu können.

Da die Klasse hier als Unterklasse verwendet wird, wurden eigene Botschaften zur Steuerung hinzugefügt.

Es können folgende Botschaften empfangen werden:

- CM_FENSTERNUMMER: Bei Erzeugung dieser Klasse wird in der Parameterliste dieser Botschaft der Fensterindex des Hauptprogramms an die Klasse übergeben. Dieser wird global gespeichert und mit entsprechenden Botschaften wieder an das Hauptprogramm zurückgegeben.

- CM_DELETEBITMAP: Bei Empfang dieser Botschaft, die durch das Hauptfenster gesendet wird, löscht die Klasse das aktuell angezeigte Bitmap, ändert die Größe des Fensters auf die Anfangswerte und zeigt das AIT-Logo (aitlogo.bmp) an.

Folgende Botschaften werden von dieser Klasse an das Hauptfenster gesendet:

- CM_HIDEBITMAPWINDOW: Beim Schließen des Fensters durch den Benutzer wird diese Botschaft mit dem Fensterindex in der Parameterliste an das Hauptprogramm gesendet. Daraufhin wird das Fenster durch das Hauptprogramm von der Oberfläche entfernt, ohne daß das Objekt an sich gelöscht wird.

- CM_ARRANGEBITMAPWINDOW: Diese Botschaft wird an das Hauptfenster gesendet, woraufhin alle angezeigten Bitmap-Fenster und das Hauptfenster in der Größe so angepaßt werden, daß sie in 2 Reihen untereinander und in entsprechender Anzahl neben einander auf dem Bildschirm angezeigt werden können.

- CM_CASCADEBITMAPWINDOW: Diese Botschaft wird an das Hauptfenster gesendet, woraufhin alle angezeigten Bitmap-Fenster und das Hauptfenster in Anfangsgröße, um die Höhe der Überschriftenleiste in Höhe und Breite gegeneinander verschoben angezeigt werden.

- CM_MINIMIZEBITMAPWINDOW: Diese Botschaft wird an das Hauptfenster gesendet, woraufhin alle angezeigten Bitmap-Fenster minimiert werden und das Hauptfenster maximiert angezeigt wird.

- CM_CLOSEBITMAPWINDOW: Diese Botschaft wird an das Hauptfenster gesendet, woraufhin alle angezeigten Bitmap-Fenster vom Hauptfenster die CM_DELETEBITMAP-Botschaft (siehe oben) erhalten. Danach werden dann alle angezeigten Bitmap-Fenster von der Oberfläche gelöscht.

4.4.5 Dialog.h

Diese Klasse stellt den Meßdialog vor und nach der Datenerfassung und den Dialog für Datei-Integration zur Verfügung. Die Dialog-Klasse ist ein eigenes Fenster und kann prinzipiell auch als Hauptfenster einer Anwendung dienen. Hier wird sie nur als Unterklasse verwendet und hat keine eigenen Botschaftsverarbeitung.

Der Datenaustausch der eingegebenen Parameter mit dem Hauptprogramm erfolgt über eine global definierte Struktur *transferpuffer* vom selbst erstellten Typ *TTransferBuffer* für den Meßdialog. Diese ist in dieser Klasse deklariert und definiert. Die Daten werden vom Dialog in diese Struktur übernommen und bleiben auch nach dem Beenden des Dialogfensters für das Hauptprogramm erhalten. Für den Datei-Integrations-Dialog ist der Ablauf entsprechend.

4.4.6 Datenarray.h

Diese Datenarray-Klasse ist als eigenständiger Thread implementiert und steuert den Datenaustausch mit dem DPA. Der Thread wird bei Programmstart erzeugt, aber noch nicht gestartet. Die Klasse erhält bei der Erzeugung einen Zeiger auf das Hauptfenster und kann dadurch Botschaften an das Hauptfenster versenden. Es können keine Botschaften empfangen werden, da die Klasse nicht von einer Fensterklasse abgeleitet ist und damit keine eigene Botschaftsverarbeitung hat.

In diesem Thread werden die Daten eingelesen und je nach Integrations-Modus mit den entsprechenden Multiplex-Masken dekodiert. Hier sind auch die eigentlichen Datenarrays in denen die eingelesenen Informationen gespeichert werden.

Es gibt die folgenden öffentlichen Funktionen und Variablen:

- SetIntegrationMode(dpamode,integrationsmode): Mit dieser Funktion werden vom Hauptprogramm der Modus des DPA und der vom Benutzer gewählte Integrations-Modus gesetzt. Der DPA-Modus wird von der *PortC*-Datenklasse eingelesen und an das Hauptprogramm übergeben (siehe Anhang D.4.4.3). Nach diesen Parametern wird dann die Datenerfassung durchgeführt.

- Datenarray_laden(char* filename): Diese Funktion lädt alle Datensätze der aktuellen Messung und erzeugten die entsprechende Header-Datei. Als Parameter wird der Dateiname übergeben, der zuvor durch eine Standard-FileOpenSaveDialog im Explorer-Stil über die Tastatur oder per Mausklick vom Benutzer ausgewählt wurde.

- Datenarray_speichern(char* filename): Diese Funktion speichert alle Datensätze der aktuellen Messung und erzeugten die entsprechende Header-Datei. Als Parameter wird der Dateiname übergeben, der zuvor durch eine Standard-FileOpenSaveDialog im Explorer-Stil über die Tastatur oder per Mausklick vom Benutzer ausgewählt wurde.

- daten_init(): Die vorhandene Datensätze werden gelöscht, ohne daß der Thread an sich entfernt wird. Diese Funktion wird zum Löschen der Daten verwendet, wenn der Benutzer den entsprechenden Menüpunkt ‚Daten Löschen' ausgewählt hat, oder wenn Daten von der Platte neu geladen werden.

- daten_start_datum(): Diese Funktion wird bei Start der Integration vom Hauptprogramm aufgerufen. Dann wird die Systemzeit des Windows-Betriebssystems ausgelesen und gespeichert. Diese wird dann verwendet, um die Integrationsdauer laut der Systemzeit zu berechnen.

- daten_stop_datum(): Diese Funktion wird bei Stop der Integration vom Hauptprogramm aufgerufen. Dann wird die Systemzeit des Windows-Betriebssystems ausgelesen und gespeichert. Diese wird dann verwendet, um die Integrationsdauer laut der Systemzeit zu berechnen.

- daten_statistics(): Diese Funktion führt die rudimentären Auswertungen der Datensätze durch. Wenn im Hauptprogramm der Menüpunkt ‚Dateninfo' gewählt wird, wird zuerst diese Funktion ausgeführt. Die Auswertung kann nur innerhalb der Klasse geschehen, da die Daten an sich geschützt sind. Um auf die Ergebnisse zugreifen zu können, werden diese als Strings in den folgenden Char-Arrays abgelegt, die dann von ‚Dateninfo ' in einer Message-Box dargestellt werden:
  - start_date: Startdatum der Integration im Format dd.mm.yy hh.mm.ss.
  - stop_date: Stopdatum der Integration im Format dd.mm.yy hh.mm.ss.
  - integration_dauer: Berechnete Differenz zwischen Start- und Stopdatum der Integration. Dazu werden beiden Daten julianische Daten umgerechnet und dann subtrahiert. Das Zwischenergebnis in Tagen wird in Hundertstelsekunden umgerechnet, da die Systemzeit ebenfalls Hundertstelsekunden angibt. Dann wird die Differenz der Uhrzeit gebildet und das Ergebnis der Tage damit korrigiert. Das Endergebnis wird dann in Sekunden mit zwei Nachkommastellen angegeben.
  - integration_counts: Zahl der Counts im Bildarray.
  - integration_rate: Zahl der Counts im Bildarray dividiert durch die Integrationsdauer laut Systemuhr in Sekunden.
  - integration_mean: Zahl der Counts im Bildarray dividiert durch die Anzahl der Bildpixel.
  - integration_correct_time: Vom Benutzer eingegebene Integrationsdauer laut DPA-Anzeige. Diese Zeit liegt nur als String vor und wird zu keinen Berechnungen bei den rudimentären Auswertungen verwendet.
  - global_comment: Vom Benutzer in den Meßdialog eingegebener Kommentar. Dieser wird in der Datei *‚comment.dat'* gespeichert und bleibt erhalten, bis der Benutzer ihn ändert oder die Datei durch den Benutzer gelöscht wird. Dann wird der String ‚Kein globaler Kommentar' angezeigt.
  - special_comment: Vom Benutzer in den Meßdialog eingegebene Bemerkung. Diese wird nur mit der aktuellen Messung gespeichert und dient der Identifikation dieser Daten in einer Meßreihe.
- xy_zeiger(): Diese Funktion liefert einen Zeiger auf die Bilddaten an das Hauptprogramm zurück. Dieser wird dann zur Berechnung des Bitmaps der Bilddaten verwendet.
- z_zeiger(): Diese Funktion liefert einen Zeiger auf die Impulshöhen-Daten an das Hauptprogramm zurück. Die Klasse wird vor der MyImpulsWindow *Imp-win.h* erzeugt, so daß dieser Zeiger bereits vorhanden ist. Er wird dann vom Hauptprogramm bei der Erzeugung der MyImpulsWindow-Klasse an diese weitergegeben und wird dann zur Berechnung des Bitmaps für die Impulshöhen-Verteilung verwendet.

Botschaftsempfang oder -versendung findet in dieser Datenklasse nicht statt.

Wird im Hauptprogramm die Datenerfassung begonnen, so wird auch der Thread durch Aufruf der überschriebenen öffentlichen Standard-Funktion *Run()* gestartet. Der Thread setzt dann das ‚Computer aktiv'-Steuerbit und beginnt die Datenwort einzulesen. Dann werden sie je nach Modus unterschiedlich dekodiert und in die jeweiligen Arrays geschrieben.

Wird die Integration beendet, so wird der Thread angehalten und das ‚Computer aktiv'-Steuerbit deaktiviert. Der Thread wird jedoch nicht gelöscht, da sonst die Daten verloren gehen würden. Vor dem Start einer neuen Messung wird der Thread dann entfernt und neu erzeugt. Dadurch sind alle Daten und Parameter zurückgesetzt und eine neue Messung kann beginnen.

Zum durchgeführten Hardware-Zugriff wird folgende Konfiguration der NI AT-DIO32F vorgenommen:

- Port D wird als Output-Port konfiguriert. Eine Linie (D0) dieser acht dient zum Setzen des ‚Computer aktiv'-Signals. Nur wenn dieses gesetzt ist, können Daten eingelesen werden. Diese Signal ist statisch und wird zu Anfang der Integrations gesetzt und bis zum Ende aufrecht erhalten, d.h. es findet kein Handshaking auf diese Leitungen statt (imodeD=0). Beim Integrationsende wird D0 wieder logisch ‚low' also elektrisch ‚high' (5 V) gesetzt. Diese Steuersignale sind in der TTL-Logik ebenfalls invertiert.

- Port A und B werden zur Input-Gruppe 1 mit Handshaking konfiguriert. Über diese 16 Datenleitungen werden die Datenworte eingelesen. Datenleitungen sind in TTL-Logik nicht invertiert, d.h. logisch ‚low' entspricht elektrisch ‚low' (0 V).

- Das Handshaking für Gruppe 1 hat folgende Eigenschaften:

  - iSignal=1: Protokoll Modus: Gepulstes Acknowledge Handshaking, d.h. auf ein Request des DPA folgt nach der iAckDelayTime ein Einlesen der Daten und dann ein Acknowledge Signal des PCs, der ein Deaktivieren des Requests zur Folge hat. Nach der Deaktivierung geht das Acknowledge Signal wieder logisch ‚low'und das System wartet auf den nächsten Request des DPA.
  - iEdge=0: Triggering auf führende Kante (Leading Edge). Da die Steuersignale invertiert sind, entspricht dies einem Abfall der Spannung von 5 auf 0 V.
  - iReqPol=1: Handshake Request Polarität invertiert: logisch ‚high' entspricht elektrisch ‚low' (0 V). Dies ist eine Eigenschaft der TTL-Logik, nach dessen Standard der DPA aufgebaut ist.
  - iAckPol=1: Handshake Acknowledge Polarität invertiert: logisch ‚high' entspricht elektrisch ‚low' (0 V). Dies ist eine Eigenschaft der TTL-Logik, nach dessen Standard der DPA aufgebaut ist.
  - iAckDelayTime=0: Verzögerungszeit von 50 ns bis zum Auslesen der Daten nach einem Request.

- Vor dem Löschen des Threads werden die Ports wieder dekonfiguriert, damit beim erneuten Erzeugen der Datenarray-Klasse keine Probleme bei der wiederholten Konfiguration auftreten.

Bei Veränderung der Hardware, etwa dem Einsatz einer neuen Input-/Output-Karte anstelle der NI AT-DIO32F, muß diese Klasse durch eine neue ersetzt werden, die die gleiche Schnittstelle und Botschaftsverarbeitung bietet.

4.4.7 Imp-win.h

Diese MyBitmapWindow-Klasse ist eine eigenständige Fensterklasse mit eigener Botschaftsverarbeitung und einer Kontrollstruktur für die Fenster- und Darstellungsparameter. Diese Klasse wird beim Programmstart nach der Datenarray-Klasse erzeugt und erhält einen Zeiger auf dessen Impulshöhen-Array und einen Zeiger auf das Hauptprogramm. Der Zeiger auf das Impulshöhen-Array wird zur Berechnung des Impulshöhen-Verteilungs-Bitmaps verwendet. Die Klasse hat eine öffentliche Funktion *ShowDistribution()*, die eine neue Berechnung und Darstellung der Impulshöhen-Verteilung bewirkt, indem die entsprechenden Parameter gesetzt werden und dann durch Aufruf der *Paint()*-Funktion das Fenster neu gezeichnet wird. Diese Verfahrensweise ist in allen erstellten Fensterklassen gleich, erleichtert die Programmierung und verringert den Programmumfang. Während der Integration wird die regelmäßige Aktualisierung (10 s) durch einen Timer im Hauptprogramm durch diesen Funktionsaufruf gesteuert.

Die Klasse kann eine Botschaft an das Hauptfenster senden. Die Botschaft lautet CM_HIDE-IMPULSWINDOW und wird gesendet, wenn der Benutzer das Impulshöhen-Fenster schließt. Das

Fenster wird dann von der Oberfläche entfernt, das Objekt an sich aber nicht gelöscht. Das Fenster empfängt keine Botschaften.

Während des Programmablaufs wird z.B. vor einer neuen Messung die Datenarray-Klasse gelöscht und neu erzeugt. Dadurch wird der Zeiger auf die Impulshöhen-Daten verändert und die MyImpulsWindow-Klasse muß ebenfalls gelöscht und neu erzeugt werden.

Wurde nur ein Bild integriert, so sind keine Impulshöhen-Daten vorhanden und es wird scheinbar eine leere Impulshöhen-Verteilung angezeigt. Tatsächlich zeigt der Zeiger aber auf ein mit dem Wert 0 gefülltes Array und die Umsetzung erzeugt eine Linie, die mit der Abszissen-Achse zusammenfällt und nicht als solche zu erkennen ist.

Zu Umsetzung der Daten wurde eine Funktion implementiert, die zuerst die Größe des Fensters bestimmt. Den Eckpunkten werden Werte (Benutzerkoordinaten) zugeordnet und diese anhand der Fenstergröße in Bitmap-Koordinaten umgerechnet. Dann werden die vorhandenen Daten auf den Maximalwert normiert und in eine Histogrammdarstellung umgewandelt, so daß sich der eigentliche Meßwert in der Mitte der jeweiligen Säule befindet. Diese Daten werden dann im Bitmap angezeigt.

Index

Danksagungen

Mein Dank gilt Herrn Prof. Dr. Klaus Werner für die Vergabe und Begutachtung dieser Diplomarbeit.

Herrn Dr. Jürgen Barnstedt danke ich herzlich für die hervorragende Betreuung und den Rat in technischen Fragen zu den MCP-Detektoren und den von ihm entwickelten Hard- und Software-Systemen sowie für die Entwicklung der Aufgabenstellung dieser Diplomarbeit an sich.

Herrn Dr. Norbert Kappelmann möchte ich danken für die Betreuung und Unterstützung in allgemeinen Fragen während der gesamten Erstellung der Diplomarbeit.

Allen Mitgliedern der ORFEUS-Arbeitsgruppe und des Instituts danke ich für das sehr angenehme Arbeitsklima, für vielfältige Hilfeleistungen und zahlreiche anregende Diskussionen.

Für die immer freundliche, prompte und kompentente Unterstützung bei der Erstellung der neuen Hardware möchte ich der Elektronikwerkstatt danken, im besonderen Herrn Olaf Luz.

Des weiteren gilt mein Dank all denen, die durch freundschaftlichen Rat zum Gelingen dieser Arbeit beigetragen haben.

Abschließend möchte ich mich besonders bei meinen Eltern bedanken, die es mir durch Ihre vielfältige Unterstützung über die Jahre hinweg erst ermöglicht haben das Studium und damit auch diese Diplomarbeit zu vollenden.

- Diese Arbeit wurde mit LaTeX–2ε geschrieben.
- Die Graphiken mit IDL (Interactive Data Language),Microsoft-PowerPoint bzw. mit xfig unter HP-UX erstellt.
- Die Programmierung der vorgestellten Software erfolgte vollständig mit dem Entwicklungssystem Borland C++ 5.0 unter Microsoft Windows NT 4.0
- In dieser Arbeit werden an verschiedenen Stellen Soft- und Hardware-Bezeichnungen verwendet, die in den meisten Fällen auch eingetragene Warenzeichen sind und als solche den gesetzlichen Bestimmungen unterliegen.

Erklärung:

Hiermit erkläre ich gemäß der Diplom-Prüfungsordnung Physik vom 9. Juni 1993 §18 Abs. 7, daß ich die hier vorliegende Arbeit selbständig verfaßt und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.