Aufbau und Test der Elektronik für ein Elektronenstreukoinzidenz-experiment
und
Vergleich der Photoabsorptionsquerschnitte in relativistischer Protonenstreuung mit elektromagnetischen Proben

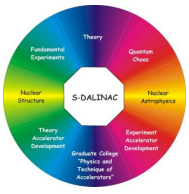
Setup and test of electronics for an electron scattering coincidence experiment and comparison of photoabsorption cross sections in relativistic proton scattering with electromagnetic probes

Master-Thesis von Sergej Bassauer

Juli 2014

TECHNISCHE UNIVERSITÄT DARMSTADT

Fachbereich Physik
Institut für Kernphysik



Gefördert durch die DFG im Rahmen des SFB 634 und NE 679/3-1.

Aufbau und Test der Elektronik für ein Elektronenstreukoinzidenzexperiment
und
Vergleich der Photoabsorptionsquerschnitte in relativistischer Protonenstreuung mit elektromagnetischen Proben
Setup and test of electronics for an electron scattering coincidence experiment and comparison of photoabsorption cross sections in relativistic proton scattering with electromagnetic probes

Vorgelegte Master-Thesis von Sergej Bassauer

1. Gutachten: Prof. Dr. Peter von Neumann-Cosel
2. Gutachten: Dipl.-Phys. Jonny Birkhan

Tag der Einreichung:

Zusammenfassung

Diese Masterarbeit ist in zwei Teile unterteilt. Der erste Teil behandelt den Aufbau und den Test der Elektronik für ein Elektronenstreukoinzidenzexperiment. Am QCLAM-Spektrometer, einem Magnetspektrometer des Instituts für Kernphysik an der TU Darmstadt, werden (e, e')- und $(e, e'x)$-Experimente durchgeführt. Ziel dieser Arbeit war es, die Koinzidenzelektronik für die letzteren zu verwirklichen. Da die zu erwartenden Zählraten sehr gering sind, muss ein möglichst großer Raumwinkel abgedeckt werden. Hierfür stehen 30 Siliziumdetektoren zur Verfügung, die in Koinzidenz mit dem QCLAM-Spektrometer, in dem die Elektronenspektren aufgenommen werden, betrieben werden können. Die Auslese wird mit Hilfe des bei der Gesellschaft für Schwerionenforschung (GSI) entwickelten Multi Branch Systems (MBS) verwirklicht. Im Rahmen dieser Arbeit wurden die Siliziumdetektoren in die Streukammer des QCLAM-Spektrometers eingebaut und es wurde ein Elektronenenergieverlustspektrum aufgenommen. Hierbei wurde der Energieverlust der an einem Kohlenstofftarget gestreuten Elektronen mit Hilfe eines Siliziumdetektors gemessen. So konnte ein Teil des Elektronikaufbaus erfolgreich getestet werden. Des Weiteren wurde die Auslesesoftware erweitert und ein Time-to-Digital-Converter (TDC) wurde programmiert und erfolgreich getestet.

Im zweiten Teil werden Photoabsorptionsquerschnitte in relativistischer Protonenstreuung mit elektromagnetischen Proben verglichen. Hierfür stehen Protonenstreudaten für die Kerne ^{28}Si, ^{40}Ca, ^{48}Ca, ^{96}Mo, ^{120}Sn, ^{144}Sm, ^{154}Sm und ^{208}Pb sowie die entsprechenden Photoabsorptionsquerschnitte zur Verfügung. In dieser Arbeit werden zwei verschiedene Methoden aufgezeigt, mit denen es möglich ist aus Protonenstreudaten Photoabsorptionsquerschnitte zu extrahieren. Eine modellunabhängige Methode, die sogenannte virtuelle Photonenmethode und eine modellabhängige Methode bei der DWBA- und QPM-Rechnungen verwendet werden. Die beiden Methoden werden verglichen und es werden Vor- und Nachteile dieser diskutiert. Es hat sich gezeigt, dass die virtuelle Photonenmethode gute Ergebnisse bei der Extraktion des Photoabsorptionsquerschnitts für schwere Kerne liefert, jedoch bei leichten Kernen versagt. Darüber hinaus wird die Form des Photoabsorptionsquerschnitts gut bei niedrigen Energien beschrieben. Die modellabhängige Methode überschätzt den Photoabsorptionsquerschnitt sowohl für leichte als auch für schwere Kerne, beschreibt aber sehr gut die Form des Photoabsorptionsspektrums in allen Energiebereichen.

Abstract

This master's thesis consists of two parts. In the first part, the setup and test of electronics for an electron scattering coincidence experiment is described. At the QCLAM spectrometer, a magnetic spectrometer of the institute for nuclear physics at TU Darmstadt, (e, e') and $(e, e'x)$ experiments are performed. The aim of this thesis was the realisation of the coincidence electronics for the latter type. Since the expected count rates are very low, a sufficiently large solid angle must be covered. Therefor, 30 silicon detectors are available, which can be operated in coincidence with the QCLAM spectrometer. The data acquisition was realised using the Multi Branch System (MBS), developed at the Gesellschaft für Schwerionenforschung (GSI). In the framework of this thesis the silicon detectors were installed in the scattering chamber of the QCLAM spectrometer and an electron energy loss spectrum was measured. Therefor the energy loss of the electrons which were scatterd off a carbon target was measured using one of the silicon detectors. This way a part of the electronics was tested successfully. Furthermore, the data acquisition software was upgraded and a Time-to-Digital-Converter (TDC) was programmed and tested successfully.

In the second part, photoabsorption cross sections in relativistic proton scattering are compared with electromagnetic probes. For this purpose, proton scattering data for ^{28}Si, ^{40}Ca, ^{48}Ca, ^{96}Mo, ^{120}Sn, ^{144}Sm, ^{154}Sm and ^{208}Pb as well as photoabsorption cross sections are available. This master's thesis shows two different methods, which can be used to extract photoabsorption cross sections from proton scattering data. A model independent so-called Virtual Photon Method and a model dependent method using DWBA and QPM calculations. The two methods are compared in detail. Furthermore assets and drawbacks of each of them are discussed. It became apparent that one obtains good results using the Virtual Photon Method for heavy nuclei. However, the descripton of light nuclei is not so well. In addition the shape of the photoabsorption cross section is well described at lower energies. The model dependent method overestimates the photoabsorption cross sections in both cases, i.e. for light nuclei as well as for heavy nuclei. The shape of the photoabsorption cross section however, is very well described in the whole energy range.

Inhaltsverzeichnis

I Aufbau und Test der Elektronik für ein Elektronenstreukoinzidenz-experiment 9

1 Einleitung 11
1.1 Das QCLAM-Spektrometer 12

2 Theoretische Grundlagen 15
2.1 Halbleiter und das Bändermodel 15
2.2 Energieverlust geladener Teilchen in Materie 16
2.3 Signalerzeugung 18

3 Experimentaufbau 19
3.1 Koinzidenzschaltung 19

4 Messung 23
4.1 Auslesesoftware 23
4.2 Testmessung 23
4.2.1 Test des ADCs 23
4.2.2 Test des TDCs 26

5 Zusammenfassung und Ausblick 27

II Vergleich der Photoabsorptionsquerschnitte in relativistischer Protonen-streuung mit elektromagnetischen Proben 29

6 Einleitung 31

7 Theoretische Grundlagen 33
7.1 Inelastische Protonenstreuung 33
7.1.1 Nukleon-Nukleon-Wechselwirkung 33
7.1.2 Distorted Wave Born Approximation 36
7.1.3 Elektromagnetische Wechselwirkung 37
7.2 Virtuelle Photonenmethode 39

7.3 Quasiteilchen-Phonon-Modell 41

8 Protonenstreuexperimente am RCNP **47**
8.1 Die Spektrometer 47

9 Datenanalyse **51**
9.1 Extraktion von Photoabsorptionsquerschnitten unter Verwendung der virtuellen Photonenmethode 51
9.2 Extraktion von Photoabsorptionsquerschnitten unter Verwendung der modellabhängigen Methode 58
9.3 Vergleich und Diskussion der beiden Methoden 63

10 Zusammenfassung und Ausblick **65**

A Appendix: Teil I **67**
A.1 Durchführung einer Messung mit Hilfe des MBS 67
A.2 Programmcode 68
A.2.1 f_user.c 68
A.2.2 create_mem.h 70
A.2.3 CaenV775.c 70
A.2.4 CaenV775.h 77
A.2.5 CaenV785.c 79
A.2.6 CaenV785.h 86
A.2.7 setup.usf (Zeile 141-146) 88
A.2.8 startup.scom 89
A.2.9 shutdown.scom 89
A.2.10 listmode2plot.c 89

B Appendix: Teil II **95**
B.1 Protonenstreuspektren 95
B.2 Photoabsorptionsspektren 99
B.3 Extrahierte Photoabsorptionsquerschnitte unter Verwendung der virtuellen Photonenmethode 103
B.4 Extrahierte Photoabsorptionsquerschnitte unter Verwendung der modellabhängigen Methode 111

Abbildungsverzeichnis

1.1 Der S-DALINAC mit den Experimentierplätzen. 11
1.2 Aufbau des QCLAM-Spektrometers. 12

2.1 Bändermodell für Metall, Halbleiter und Isolator. 15
2.2 Energieverlust verschiedener geladener Teilchen in Luft. 17
2.3 Durch ein Elektron-Loch-Paar induzierte Pulsform. 18

3.1 Provisorischer Koinzidenzschaltplan. 20

4.1 Spektrum der ADC-Testmessung mit einem Pulsgenerator. 24
4.2 Energieverlustspektrum von am Kohlenstoff gestreuten Elektronen. 25
4.3 Energieverlustspektrum von am Kohlenstoff gestreuten Elektronen mit verschiedener Pulsformzeit. 25
4.4 Elektronenenergieverlustspektrum in Silizium. 25
4.5 Spektrum der TDC-Testmessung mit einem Pulsgenerator. 26

6.1 Die wichtigsten Riesenresonanzen und ihre Interpretationen. 32

7.1 Energieabhängigkeit der Zentralterme der Nukleon-Nukleon-t-Matrix. 35
7.2 Hyperbolische Bahn eines Projektilteilchens in der klassischen Beschreibung. . . . 37
7.3 Differentielle virtuelle Photonenzahlen für ^{40}Ca. 41

8.1 Schematische Darstellung der Experimentiereinrichtung am RCNP. 48

9.1 Gefaltete und ungefaltete differentielle bzw. gemittelte Photonenzahl der E1-Anregung in ^{40}Ca. 53
9.2 Der extrahierte sowie der experimentelle Photoabsorptionsquerschnitt für ^{40}Ca. . 54
9.3 Der extrahierte sowie der experimentelle Photoabsorptionsquerschnitt für ^{120}Sn. . 55
9.4 Vergleich der differentiellen Photonenzahl in semiklassischer bzw. eikonaler Näherung für ^{40}Ca. 56
9.5 Der extrahierte sowie der experimentelle Photoabsorptionsquerschnitt für ^{40}Ca mit Hilfe der eikonalen Näherung. 57
9.6 Normierungsfaktoren, die sich bei der Extraktion des Photoabsorptionsquerschnitts für verschiedene Kerne ergeben. 58
9.7 Das Verhältnis aus differentiellen Wirkungsquerschnitten und B(E1)-Stärken mit $q = 0$ MeV für ^{40}Ca. 60

9.8 Das Verhältnis aus differentiellen Wirkungsquerschnitten und B(E1)-Stärken mit $q = E_x$ für ^{40}Ca. 60
9.9 Das Verhältnis aus differentiellen Wirkungsquerschnitten und B(E1)-Stärken mit $q = 10$ MeV und $q = 20$ MeV für ^{40}Ca. 61
9.10 Das Verhältnis aus differentiellen Wirkungsquerschnitten und der B(E1)-Stärke bei der Schwerpunktsenergie der Riesenresonanz für ^{40}Ca. 61
9.11 Vergleich zwischen den Photoabsorptionsquerschnitten, die mit der modellabhängigen Methode extrahiert wurden und den experimentellen Photoabsorptionsquerschnitten für ^{40}Ca. 62

B.1 Doppeltdifferentieller Wirkungsquerschnitt aus Protonenstreuung für ^{28}Si. 95
B.2 Doppeltdifferentieller Wirkungsquerschnitt aus Protonenstreuung für ^{40}Ca. 95
B.3 Doppeltdifferentieller Wirkungsquerschnitt aus Protonenstreuung für ^{48}Ca. 96
B.4 Doppeltdifferentieller Wirkungsquerschnitt aus Protonenstreuung für ^{96}Mo. 96
B.5 Doppeltdifferentieller Wirkungsquerschnitt aus Protonenstreuung für ^{120}Sn. 97
B.6 Doppeltdifferentieller Wirkungsquerschnitt aus Protonenstreuung für ^{144}Sm. . . . 97
B.7 Doppeltdifferentieller Wirkungsquerschnitt aus Protonenstreuung für ^{154}Sm. . . . 98
B.8 Doppeltdifferentieller Wirkungsquerschnitt aus Protonenstreuung für ^{208}Pb. 98
B.9 Photoabsorptionsquerschnitt für ^{28}Si. 99
B.10 Photoabsorptionsquerschnitt für ^{40}Ca. 99
B.11 Photoabsorptionsquerschnitt für ^{48}Ca. 100
B.12 Photoabsorptionsquerschnitt für ^{96}Mo. 100
B.13 Photoabsorptionsquerschnitt für ^{120}Sn. 101
B.14 Photoabsorptionsquerschnitt für ^{144}Sm. 101
B.15 Photoabsorptionsquerschnitt für ^{154}Sm. 102
B.16 Photoabsorptionsquerschnitt für ^{208}Pb. 102
B.17 Mit virt. Photonenmethode extrahierter Photoabsorptionsquerschnitt für ^{28}Si. . . . 103
B.18 Mit virt. Photonenmethode extrahierter Photoabsorptionsquerschnitt für ^{40}Ca. . . 104
B.19 Mit virt. Photonenmethode extrahierter Photoabsorptionsquerschnitt für ^{48}Ca. . . 105
B.20 Mit virt. Photonenmethode extrahierter Photoabsorptionsquerschnitt für ^{96}Mo. . . 106
B.21 Mit virt. Photonenmethode extrahierter Photoabsorptionsquerschnitt für ^{120}Sn. . . 107
B.22 Mit virt. Photonenmethode extrahierter Photoabsorptionsquerschnitt für ^{144}Sm. . 108
B.23 Mit virt. Photonenmethode extrahierter Photoabsorptionsquerschnitt für ^{154}Sm. . 109
B.24 Mit virt. Photonenmethode extrahierter Photoabsorptionsquerschnitt für ^{208}Pb. . . 110
B.25 Mit modellabh. Methode extrahierter Photoabsorptionsquerschnitt für ^{40}Ca. 111
B.26 Mit modellabh. Methode extrahierter Photoabsorptionsquerschnitt für ^{120}Sn. . . . 112
B.27 Mit modellabh. Methode extrahierter Photoabsorptionsquerschnitt für ^{208}Pb. . . . 113

Tabellenverzeichnis

1.1 Spezifikationen des QCLAM-Spektrometers. 13

8.1 Spezifikationen des Grand Raiden und Large Acceptance Spektrometers. 49

9.1 Auflistung der in dieser Arbeit bearbeiteten Kerne. 51

9.2 Normierungsfaktoren, die sich bei der Extraktion des Photoabsorptionsquerschnitts für die einzelnen Kerne ergeben. 58

Teil I

Aufbau und Test der Elektronik für ein Elektronen-streukoinzidenzexperiment

1 Einleitung

Am supraleitenden Darmstädter Elektronenlinearbeschleuniger (S-DALINAC) am Institut für Kernphysik der TU Darmstadt werden Untersuchungen zur Struktur von Kernen mit Hilfe von inelastischer Elektronenstreuung durchgeführt. Im Jahre 1980 konzipiert und seit 1991 aufgebaut, wird der S-DALINAC im Rahmen von Bachelor-, Master- und Doktorarbeiten ständig weiter entwickelt. In Abb. 1.1 ist der Beschleuniger mit den verschiedenen Experimentierplätzen abgebildet.

Zur Erzeugung der Elektronen wird eine thermionische Elektronenquelle verwendet. Die Elektronen werden dann elektrostatisch auf eine Energie von 250 keV beschleunigt [2]. Alternativ dazu wurde vor kurzem eine Elektronenquelle entwickelt [3], die spinpolarisierte Elektronen erzeugt (②). In der nachfolgenden Chopper-Prebuncher-Sektion wird dem Elektronenstrahl eine 3 GHz Zeitstruktur aufgeprägt und der Strahl kann im sogenannten Injektor-Beschleuniger mit Hilfe von 20-zelligen Niobstrukturen auf 10 MeV beschleunigt werden. Am Injektormessplatz (①) können nun Bremsstrahlungsexperimente mit Elektronenströmen von bis zu 60 μA durchgeführt werden. Der Elektronenstrahl kann aber auch in den Hauptbeschleuniger umgelenkt werden, wo er einen Energiegewinn von bis zu 40 MeV pro Rezirkulation erhält und an einem der beiden Elektronenstreuspektrometer (④, ⑤) für Kernstrukturexperimente verwendet werden kann [2].

Am sogenannten LINTOTT-Spektrometer können (e,e')-Experimente durchgeführt werden. Die Besonderheit des Spektrometers ist hierbei die sehr gute Energieauflösung im sogenannten

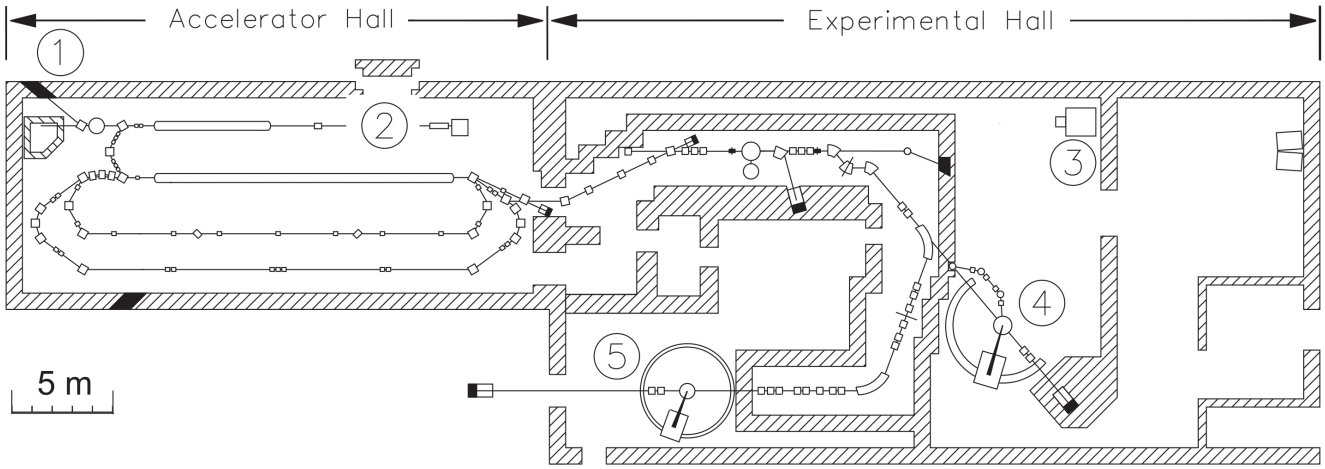


Abbildung 1.1: Der S-DALINAC mit den Experimentierplätzen. ① Bremsstrahlungsexperimente. ② Quelle für spinpolarisierte Elektronen. ③ $(\gamma,\gamma'x)$-Experimente ④ (e,e')- und $(e,e'x)$-Experimente am QCLAM-Spektrometer. ⑤ (e,e')-Experimente am LINTOTT-Spektrometer [1].

Energieverlustmodus. Leider deckt das Spektrometer nur einen kleinen Raumwinkel ab. Ein weiterer Nachteil ist die geringe Impulsakzeptanz [4].

Das QCLAM-Spektrometer (Quadrupol Clamshell) ermöglicht es, (e, e')- und $(e, e'x)$-Experimente unter Streuwinkeln von bis zu 180° durchzuführen. Des Weiteren hat das Spektrometer eine größere Winkel- und Impulsakzeptanz [5]. Da im Rahmen dieser Arbeit nur am QCLAM-Spektrometer gearbeitet wurde, wird auf das LINTOTT nicht weiter eingegangen.

1.1 Das QCLAM-Spektrometer

Das QCLAM-Spektrometer wurde am Institut für Kernphysik der TU Darmstadt entwickelt und aufgebaut. In Abb. 1.2 ist das Spektrometer dargestellt. Es besteht aus einem horizontal fokussierenden Quadrupolmagneten und einem in Form einer Muschel aufgeklappten Dipolmagneten sowie dem Detektorsystem, bestehend aus drei Driftkammern sowie einem Tscherenkowzähler

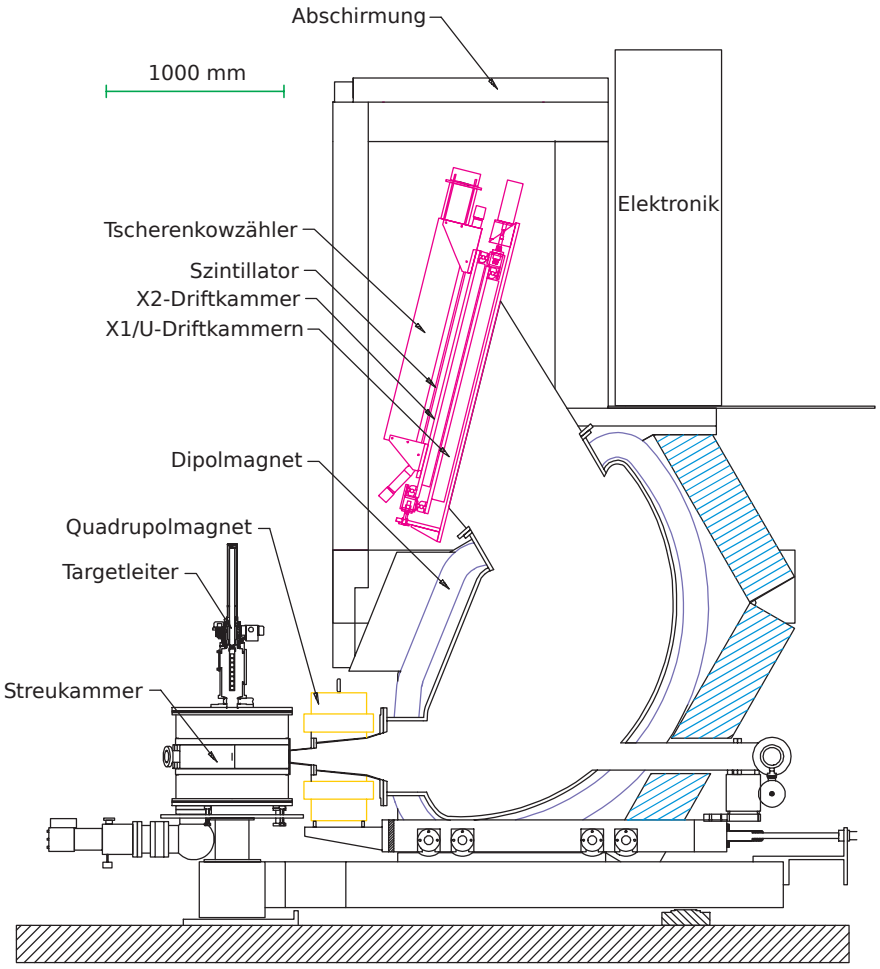


Abbildung 1.2: Aufbau des QCLAM-Spektrometers [6].

und Szintillator als Triggerdetektoren. Nachdem die Elektronen in der Streukammer an einem Target gestreut wurden, passieren sie den Quadrupol- und den Dipolmagneten. Hierbei werden sie nach ihrem Impuls separiert. Mit Hilfe der ortssensitiven Driftkammern kann schließlich die Energie bzw. der Streuwinkel der Elektronen rekonstruiert werden. Das QCLAM zeichnet sich durch die hohe Impulsakzeptanz von 22 % und Raumwinkelakzeptanz von 35 msr aus [5, 7]. Die Spezifikationen des QCLAM-Spektrometers sind in Tab. 1.1 aufgelistet.

Tabelle 1.1: Spezifikationen des QCLAM-Spektrometers [8].

Streuwinkelbereich	25°-155°
Neigung der Fokalebene	37,97°
Impulsakzeptanz	±10%
Impulsauflösung	4350
Raumwinkelakzeptanz	35 msr
Winkelakzeptanz (horizontal)	±100 mrad
Winkelakzeptanz (vertikal)	±100 mrad

2 Theoretische Grundlagen

In diesem Kapitel werden theoretische Grundlagen zu Halbleitern und deren Eigenschaften vermittelt, wobei insbesondere auf Siliziumhalbleiterdetektoren eingegangen wird. Des Weiteren wird das Verhalten von geladenen Teilchen in Materie sowie die Ausbildung des Detektorsignals beschrieben.

2.1 Halbleiter und das Bändermodel

Als Halbleiter werden Materialien bezeichnet, die abhängig von der Temperatur, sowohl die Eigenschaften eines Metalls als auch die Eigenschaften eines Isolators annehmen können. In einem Kristall haben die Atome, die in die Kristallstruktur eingebettet sind, einen definierten Abstand zu einander. Sie befinden sich jedoch so nah bei einander, dass sich die Energieniveaus der Elektronen teilweise überlappen. Diese Energiezustände können zu Bändern zusammengefasst werden. Hierbei unterscheidet man zwischen dem Valenzband und dem Leitungsband. Im Valenzband befinden sich Elektronen, die fest am Atom gebunden sind, in einem Leitungsband hingegen sind die Elektronen frei beweglich. Zwischen den beiden Bändern befindet sich die Bandlücke, die verhindert, dass Elektronen vom Valenzband in das Leitungsband übertreten. Die Größe der Bandlücke bestimmt, ob es sich bei einem Material um Metall, Halbleiter oder Isolator handelt. In Abb. 2.1 wird dies veranschaulicht. Bei einem Metall überlappen sich das Leitungs- und Valenzband, so dass die Elektronen zwischen den beiden Bändern problemlos wechseln können. Beim Anlegen eines externen elektrischen Feldes führt dies zum gerichteten Ladungsstrom. Das andere Extremum stellt der Isolator dar, bei dem die Bandlücke so groß

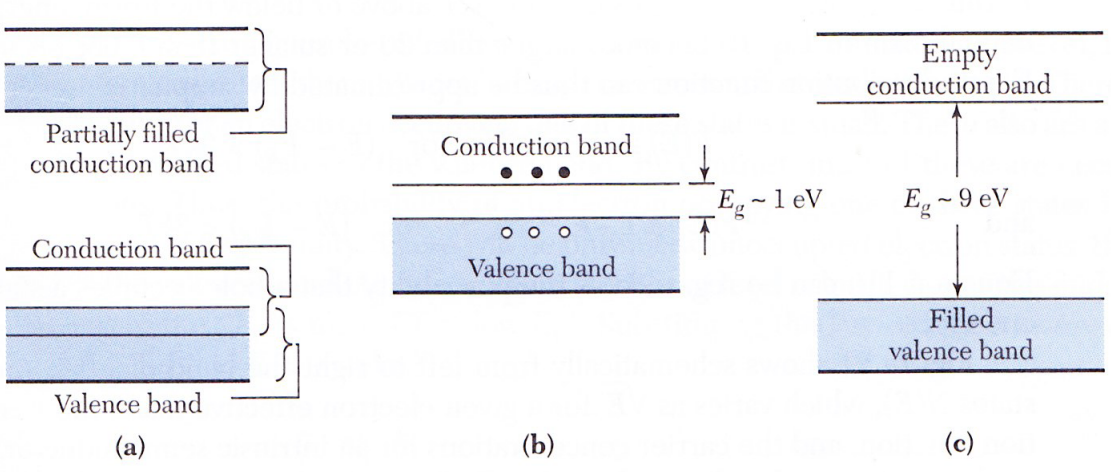


Abbildung 2.1: Bändermodell für Metall (a), Halbleiter (b) und Isolator (c) [9].

ist, dass die Elektronen aus dem Valenzband nicht in das Leitungsband gelangen können, was den Isolator nichtleitend macht. Der Halbleiter bildet einen Übergang zwischen dem Metall und dem Isolator. Die Bandlücke eines Halbleiters ist viel kleiner als die eines Isolators, so dass allein durch die thermische Energie einige Elektronen aus dem Valenzband ins Leitungsband wechseln können. Beim Silizium ist die Bandlücke bei Raumtemperatur 1,1 eV groß [10]. Durch das Anlegen eines elektrischen Feldes ist auch hier ein gerichteter Ladungsstrom möglich. Beim absoluten Nullpunkt der Temperatur befinden sich allerdings alle Elektronen im Valenzband und der Halbleiter verhält sich wie ein Isolator [10, 11].

2.2 Energieverlust geladener Teilchen in Materie

Geladene Teilchen verlieren ihre Energie beim Durchgang durch Materie. Im Wesentlichen sind hierfür zwei Prozesse verantwortlich, inelastische Stöße an Hüllenelektronen und elastische Streuung am Atomkern. Für sehr leichte Teilchen, wie Elektronen und Positronen spielt auch die Bremsstrahlung eine wichtige Rolle. Die erste korrekte quantenmechanische Rechnung für den Energieverlust wurde von Bethe und Bloch durchgeführt. Die Formel hierfür ist gegeben durch [10]

$$-\frac{\mathrm{d}E}{\mathrm{d}x} = 2\pi N_a r_e^2 m_e c^2 \rho \frac{Z z^2}{A\beta^2}\left[\ln\left(\frac{2m_e\gamma^2 v^2 W_{max}}{I^2} - 2\beta^2 - \delta - 2\frac{C}{Z}\right)\right], \tag{2.1}$$

mit

N_a: Avogadrokonstante
m_e: Ruhemasse des Elektrons
ρ: Dichte des Absorbermaterials
z: Ladungszahl des einfallenden Teilchens
β: Geschwindigkeit des einfallenden Teilchens in Einheiten der Lichtgeschwindigkeit
v: Geschwindigkeit des einfallenden Teilchens
I: mittleres Anregungspotential
C: Schalenkorrektur.
r_e: klassischer Elektronenradius
c: Lichtgeschwindigkeit
Z: Ladungszahl des Absorbermaterials
A: Massenzahl des Absorbermaterials
W_{max}: maximaler Energieübertrag bei einem einzelnen Stoß
γ: Lorentzfaktor
δ: Dichtekorrektur

In Abb. 2.2 sieht man den Energieverlust von verschiedenen geladenen Teilchen in Materie. Man kann der Abbildung auch entnehmen, dass für hohe Energien der Energieverlust näherungsweise konstant wird. Dieser spezifische Energieverlust liegt dann bei etwa 2 MeV/g/cm^2 in leichten Materialien. Solche relativistischen Teilchen werden auch MIPs (minimum ionising particles) genannt. Bei niedrigen Energien versagt die Bethe-Bloch-Formel jedoch im Allgemeinen, den Energieverlust korrekt zu beschreiben, da bei niedrigen Energien Ladungseinfangeffekte wichtig werden. Hierbei werden Elektronen vom Projektilteilchen (Ion) solange eingefangen, bis das System zu einem neutralen Atom wird. Die Projektilelektronen werden zudem durch ihre sehr

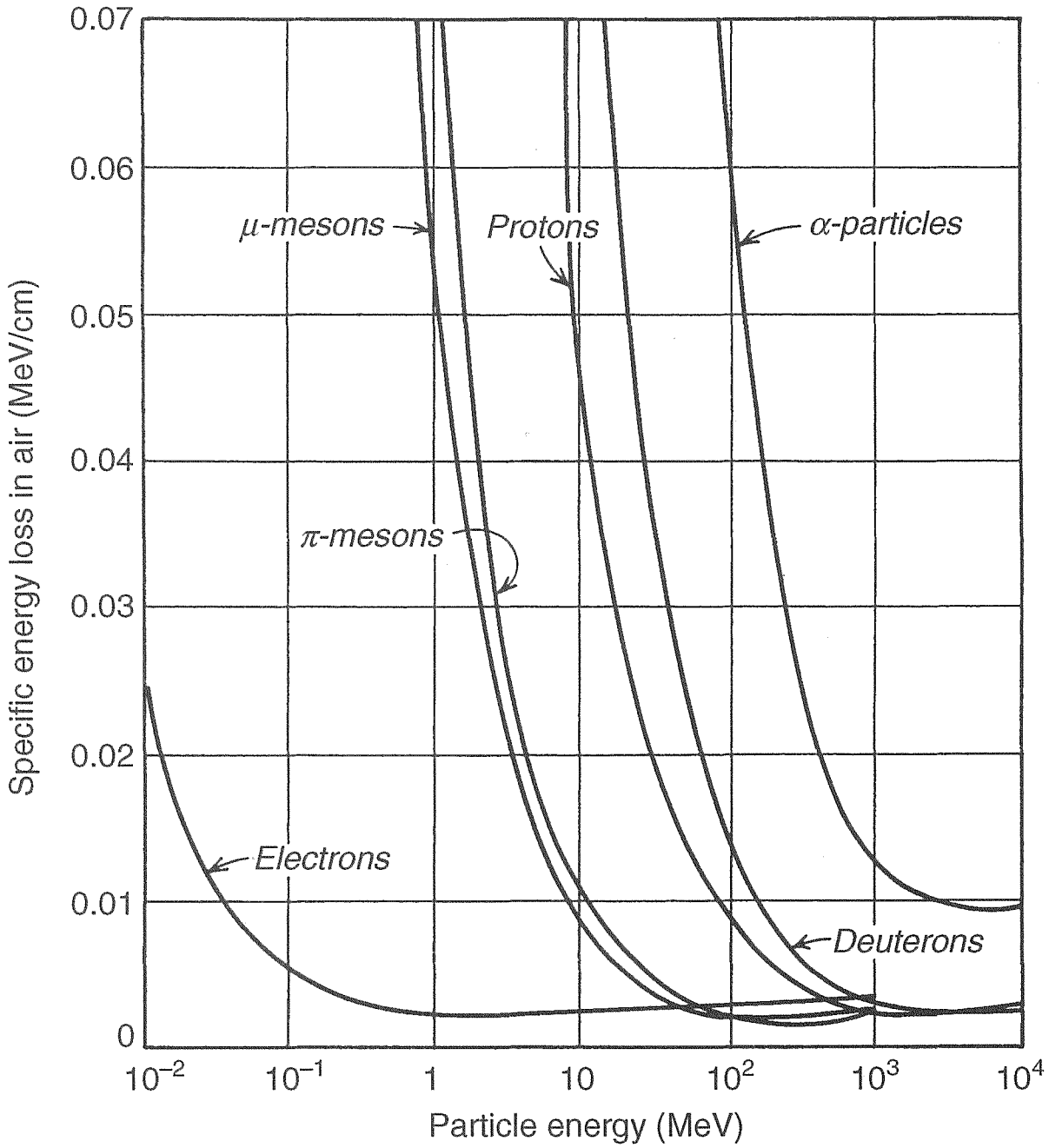


Abbildung 2.2: Energieverlust verschiedener geladener Teilchen in Luft [11].

viel kleinere Masse bei niedrigen Energien viel leichter abgelenkt, so dass die Annahme von geradlinig bewegenden Teilchen nicht mehr erfüllt ist [10, 11].

2.3 Signalerzeugung

Passiert ein geladenes Teilchen einen Halbleiterdetektor, so wird es, wie im letzten Abschnitt beschrieben, Energie an das Detektormaterial abgeben. Ist das Detektormaterial dick genug oder die kinetische Energie des Teilchens niedrig genug, so kann das Teilchen im Detektor auch vollständig gestoppt werden. In beiden Fällen werden im Detektormaterial beim Durchgang des Teilchens Elektron-Loch-Paare erzeugt. Diese induzieren an den Elektroden des Detektors eine Ladung, die für Elektronen bzw. Löcher mit

$$Q_e(t) = \frac{e}{d} x_0 \left[1 - \exp\left(\frac{\mu_e t}{\mu_h}\right)\right] \qquad \text{bzw.} \qquad Q_h(t) = -\frac{e}{d} x_0 \left[1 - \exp\left(\frac{-t}{\tau}\right)\right] \tag{2.2}$$

angegeben werden kann [10]. Hierbei ist d der Abstand zwischen den Elektroden, x_0 der Ort, an dem das Elektron-Loch-Paar entstanden ist, μ_e und μ_h sind die Beweglichkeiten der Elektronen bzw. Löcher, t die Zeit, die das Elektron bzw. das Loch zum erreichen der Elektrode braucht und τ die Anstiegszeit des Signals, die für Silizium im Nanosekundenbereich liegt [10]. In Abb. 2.3 ist die Form des so entstandenen Signals gezeigt.

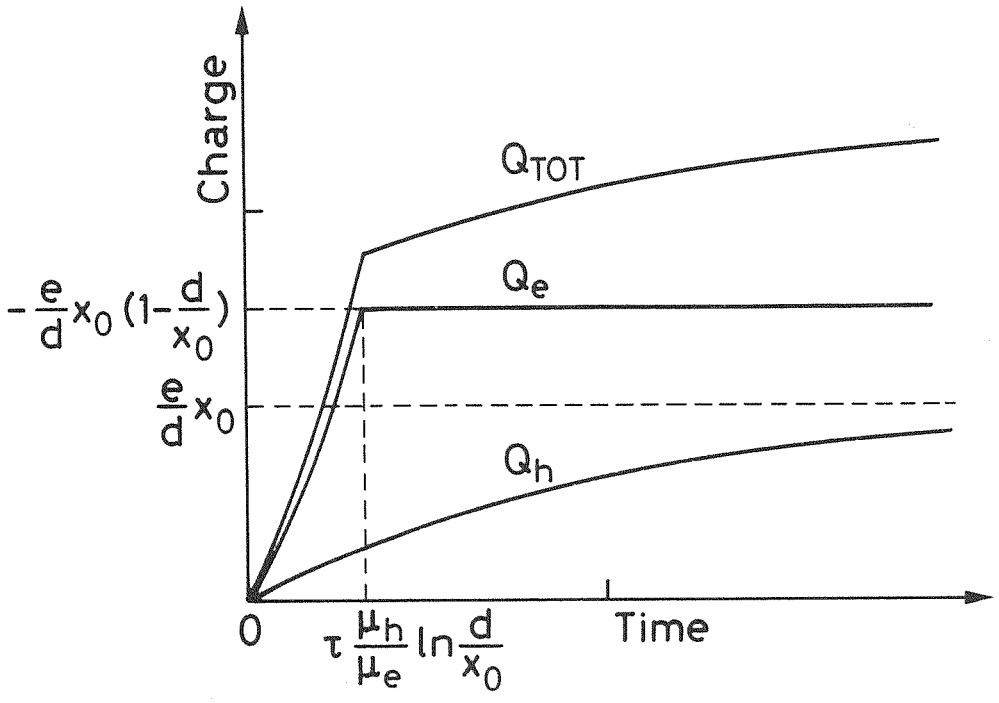


Abbildung 2.3: Durch ein Elektron-Loch-Paar induzierte Pulsform [10].

3 Experimentaufbau

Seit jeher werden am Institut für Kernphysik an der TU Darmstadt Kernstrukturexperimente durchgeführt. Eine besondere Art von Experimenten, aus denen man etwas über die Struktur des Kerns lernen kann, sind Aufbruchexperimente. Hierzu werden leichte Kerne, wie ^{3}He, ^{12}C oder ^{16}O mit Elektronen beschossen, die die Kerne in Protonen, Neutronen und Alphateilchen aufbrechen. Um den Untergrund möglichst klein zu halten, kann man die Aufbruchteilchen in Koinzidenz mit den gestreuten Elektronen detektieren. Da bei den vorgesehenen Experimenten ($(e, e'p)$ und $(e, e'\alpha)$) sehr geringe Zählraten erwartet werden, muss ein möglichst großer Raumwinkel abgedeckt werden. Für einen Testaufbau stehen 30 Siliziumdetektoren zur Verfügung. Jeder Detektor ist $(89{,}54 \pm 0{,}98)\,\mathrm{mm}^2$ groß und 300 µm dick [12]. Durch eine Flugzeitmessung wird eine Teilchentrennung ermöglicht. Außerdem kann man mit diesem Aufbau Winkelkorrelationen bestimmen. Im Rahmen einer Bachelorarbeit [12] wurden die einzelnen Detektoren bereits erfolgreich getestet. Zusätzlich stehen Streifendetektoren zur Verfügung, mit denen eine präzisere Ortsmessung möglich ist. Diese müssen jedoch weiteren Tests unterzogen werden, da sie, wie in der oben genannten Bachelorarbeit beschrieben, einige Probleme bereiteten. Im Rahmen dieser Masterarbeit wurden sie jedoch nicht verwendet.

3.1 Koinzidenzschaltung

In Abb. 3.1 ist die angestrebte Verschaltung der Siliziumdetektoren in Koinzidenz mit dem QCLAM-Spektrometer dargestellt. Auf der linken Seite, in Blau eingefärbt, sieht man die Verschaltung der Siliziumdetektoren, die bereits verwirklicht und im Rahmen dieser Arbeit erweitert und getestet wurde. Der rechte Teil, in Schwarz, befindet sich momentan in Entwicklung und ist nur ein provisorischer Schaltplan, da die Elektronik am QCLAM-Spektrometer momentan erneuert wird. Auf jedem Siliziumdetektor ist ein Vorverstärker aufgesteckt, über den der Detektor durch das Hochspannungsversorgungssystem mit etwa 150 V versorgt wird. Die Vorverstärker selbst benötigen 12 V, die sie von einer separaten Spannungsquelle erhalten. Des Weiteren hat jeder Vorverstärker einen Energiesignalausgang, einen Timingsignalausgang und einen Testeingang, mit dem ein Detektorsignal durch Anlegen eines Rechteckpulses simuliert werden kann. Hierbei wird das Timingsingnal bereits 60 ns vor dem Energiesignal erzeugt. Das Timingsignal wird zu einem CFD (Constant Fraction Discriminator) geleitet, der einen negativen Rechteckpuls erzeugt. Von da aus wird das Signal weiter zu einem NIM/ECL-Konverter geleitet. Dieser konvertiert das Signal, welches im NIM-Standard (Nuclear Instrumentation Module) vorliegt, in ein Signal im ECL-Standard (Emitter Coupled Logic). Dies ist nötig da der

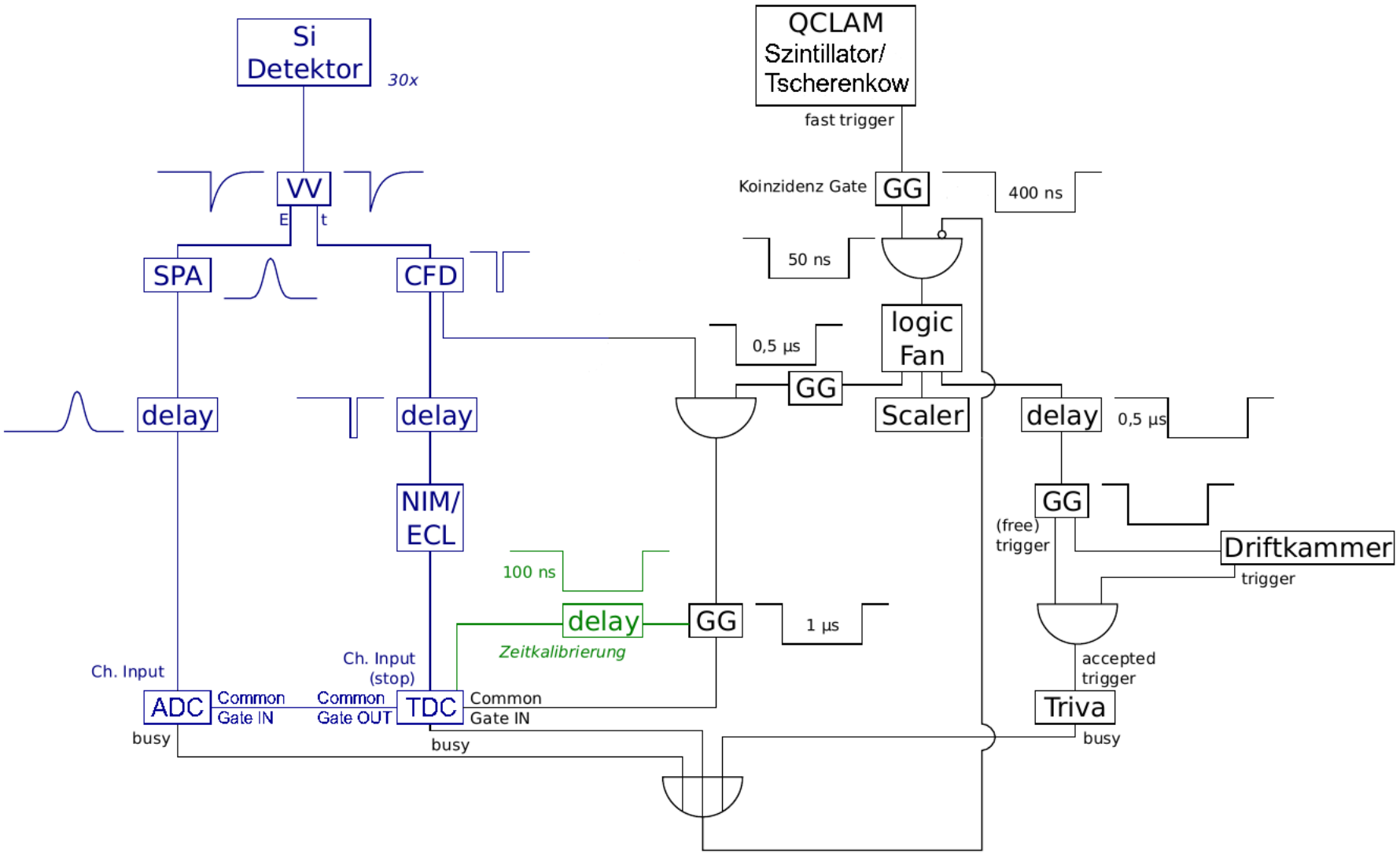


Abbildung 3.1: Provisorischer Koinzidenzschaltplan. Alle Zahlenwerte sind geschätzt [12].

TDC (Time-to-Digital-Converter) am Eingang nur Signale im ECL-Standard akzeptiert. Der TDC digitalisiert die Zeitspanne zwischen einem Start- und Stoppsignal.

Das Energiesignal wird vom Vorverstärker zum SPA (Spectroscopy Amplifier) geleitet und dort verstärkt und in Gaußform ausgegeben. Das gaußförmige Signal wird nun zum ADC (Analog-to-Digital-Converter) geführt, wo die Energieinformationen digitalisiert werden und mit Hilfe des MBS ausgelesen werden können.

Auf der rechten Seite in Abb. 3.1 sieht man die Verschaltung der (alten) Elektronik am QCLAM-Spektrometer. Der Szintillator und der Tscherenkow dienen als Triggerdetektoren, die bei Koinzidenz ein Zeitfenster öffnen. Durch ein UND-Gatter wird überprüft, ob der ADC, TDC und die sogenannte Trivaeinheit, die für die Verarbeitung der Triggersignale verantwortlich ist, betriebsbereit sind. Ist dies der Fall, wird das Signal zu einem Verteiler (Logic Fan) geführt. Von da aus führen drei Signale hinaus. Das erste Signal führt zu einem Zeitfenstergenerator und wird dann durch ein UND-Gatter mit dem Signal des CFD verglichen. Das Signal des UND-Gatters wird dann benutzt um ein Zeitfenster zu öffnen, welches als Trigger für den TDC fungiert. Das zweite Signal des Verteilers führt zu einem Zählermodul (Scaler) der die ankommenden Signale zählt. Das dritte und letzte Signal des Verteilers führt über ein Delaymodul, welches das Signal verzögert, zu einem Zeitfenstergenerator. Nach dem öffnen des Zeitfensters wird das Signal verzweigt. Eines der Signale führt zur Driftkammer und dann weiter als Trigger zu einem UND-Gatter. Dort wird das Signal nun mit dem zweiten Signal des Zeitfenstergenerators verglichen. Das Signal des UND-Gatters wird dann als Trigger für die Trivaeinheit benutzt. In einem ODER-Gatter wird schließlich noch überprüft, ob der ADC, der TDC und die Trivaeinheit betriebsbereit sind. Zwar wird sich vermutlich einiges durch die neue QCLAM-Elektronik ändern, die Grundstruktur dieses Koinzidenzschaltplans dürfte aber bestehen bleiben.

4 Messung

4.1 Auslesesoftware

Zum Auslesen der gemessenen Spektren wird das MBS (Multi Branch System), welches an der GSI entwickelt wurde, verwendet [13]. Das System läuft auf einer RIO3-Einheit [14], einem Ein-Platinen-Computer der im Messraum in das VME-Gerüst (Versa-Module-Eurocard) eingebaut ist und als intelligente Kontrolleinheit dient. Hierbei dient das MBS zum Starten, Stoppen und Speichern der Messungen [12]. Alle nötigen Schritte zum durchführen einer Messung werden in Appendix A.1 erklärt. Die wichtigsten Komponenten der Auslesesoftware sind die Dateien `f_user.c`, `create_mem.h`, `CaenV785.c`, `CaenV785.h`, `CaenV775.c`, `CaenV775.h`, die Konfigurationsdatei `setup.usf` sowie die Skripte `startup.scom`, `shutdown.scom` und das Konvertierungsprogramm `listmode2plot`. Der Quellcode sowie eine kurze Beschreibung dieser Dateien findet sich in Appendix A.2.

4.2 Testmessung

Die einzelnen Siliziumdetektoren wurden bereits von Henno Lauinger im Rahmen seiner Bachelorarbeit [12] auf Funktionalität getestet. Hierzu wurde der ADC programmiert und die Auslese mit Hilfe einer ^{241}Am-Quelle erfolgreich getestet.

Im Rahmen dieser Masterarbeit wurden die Siliziumdetektoren in die QCLAM-Streukammer eingebaut. Da der Elektronikaufbau nicht mehr vorhanden war, musste dieser wieder aufgebaut werden. Hierbei wurde der blaue Teil in Abb. 3.1 leicht abgewandelt zum Testen des ADCs bzw. TDCs verwirklicht.

4.2.1 Test des ADCs

Zum testen des ADCs wurde anstelle der Detektoren zunächst ein Pulsgenerator (Pulser) verwendet. In Abb. 4.1 sieht man ein Spektrum, das bei verschiedenen Spannungen des Pulsers aufgenommen wurde. Hierbei wurde die Spannung nach einer festen Zeit in gleichen Schritten erhöht. Wie man sehen kann, sind die Abstände der einzelnen Peaks konstant, so dass man davon ausgehen kann, dass die Auslese einwandfrei funktioniert. Die Peakhöhe unterscheidet sich leicht, weil die Zeit zwischen den Erhöhungen der Spannung nicht exakt gleich war und die Frequenz des Pulsers recht hoch war.

Während der Strahlzeit im Dezember letzten Jahres konnten die Siliziumdetektoren bei einem Elektronenstreuexperiment an einem ^{12}C-Target getestet werden. Hierbei gaben die gestreuten

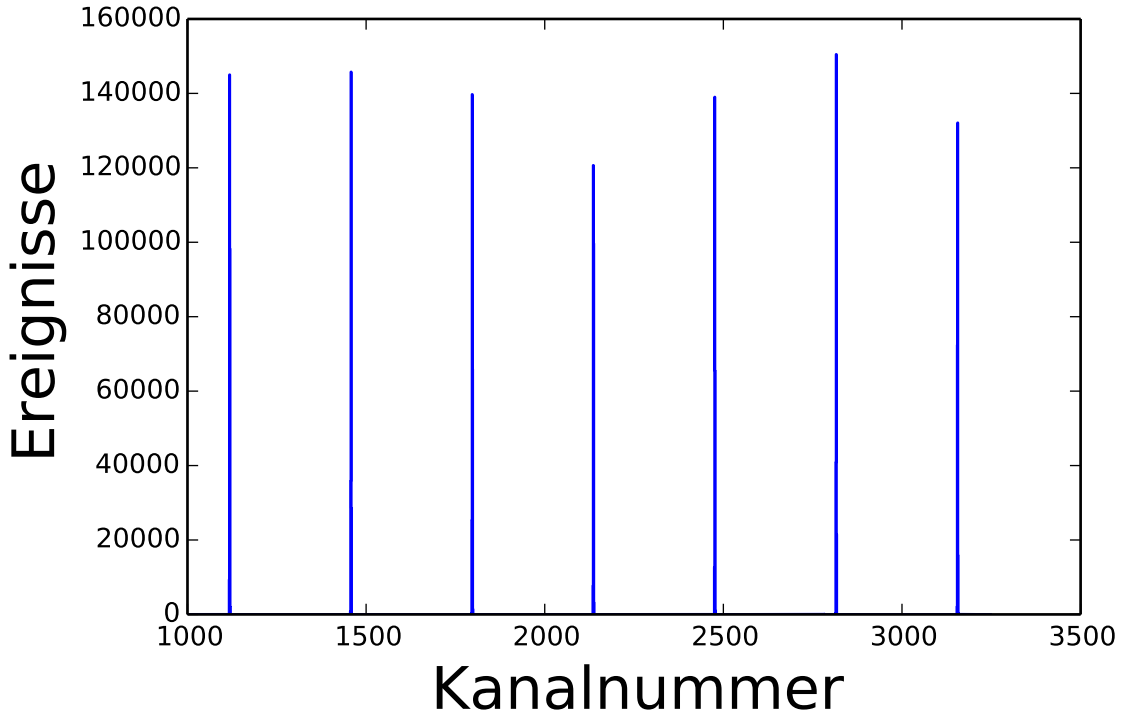


Abbildung 4.1: Spektrum der ADC-Testmessung mit einem Pulsgenerator bei verschiedenen Spannungen.

Elektronen ein Teil ihrer Energie in den Siliziumdetektoren ab. Das resultierende Energieverlustspektrum ist in Abb. 4.2 zu sehen. Im rechten Bild sieht man neben dem Hauptpeak einen zweiten Peak. Es hat sich herausgestellt, dass es sich hierbei um Summenereignisse handelt, d. h. dort gaben zwei Elektronen gleichzeitig ihre Energie ab. Dieser Umstand lässt sich überprüfen, indem man die Pulsformzeit des Hauptverstärkers verändert. Bei einer größeren Pulsformzeit ist zu erwarten, dass der zweite Peak deutlich größer wird. Dies ist tatsächlich der Fall, wie man in Abb. 4.3 sieht. Links ist das Spektrum aus der letzten Abbildung gezeigt. Rechts wurde das Spektrum mit der doppelten Pulsformzeit aufgenommen. Man sieht, dass im rechten Bild der zweite Peak deutlich größer geworden ist. Das bestätigt die Annahme, dass es sich um Summenereignisse handelt.

Zum Vergleich wurden in [15] im letzten Jahr mehrere Energieverlustspektren in 300 µm dickem Silizium aufgenommen. Dies bietet eine gute Möglichkeit um die Funktionalität der Siliziumdetektoren bzw. die Richtigkeit des in Abb. 4.2 gezeigten Spektrums zu überprüfen. Die Spektren sind in Abb. 4.4 gezeigt. Das Spektrum in Abb. 4.2 wurde bei einer Strahlenergie von etwa 75 MeV aufgenommen. Vergleicht man dieses Spektrum mit den Spektren aus Abb. 4.4, so sieht man, dass diese konsistent sind. Zwar wurden die Spektren in Abb. 4.4 bei viel kleineren Energien aufgenommen, man sieht jedoch, dass mit steigender Energie sich lediglich das Maximum zur höheren Energie verschiebt. Dies bestätigt nochmals die einwandfreie Funktionalität der Elektronik bzw. der Auslese.

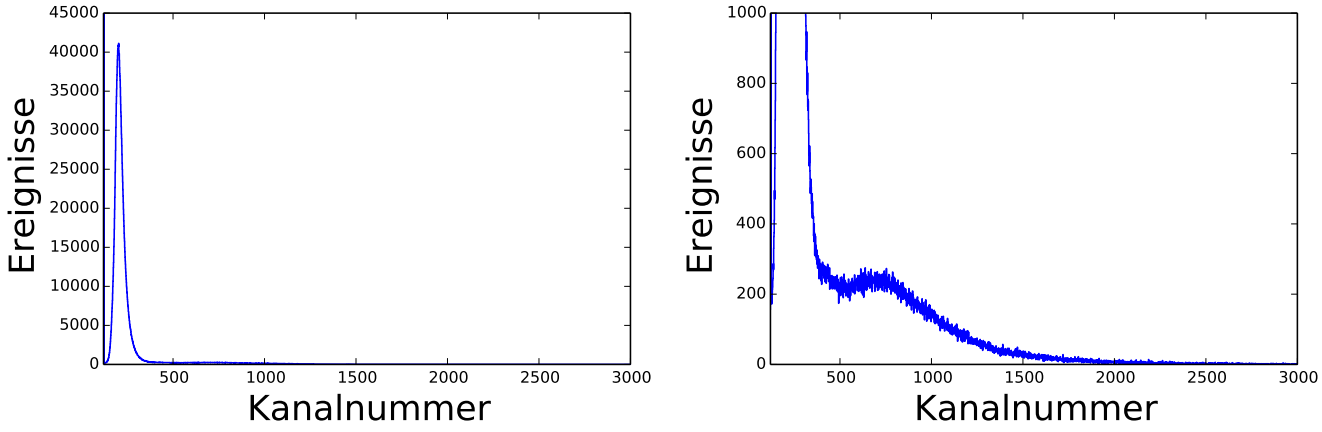


Abbildung 4.2: Energieverlustspektrum von am Kohlenstoff gestreuten Elektronen. Auf der rechten Seite sieht man nach vergrößern des Spektrums einen zweiten Peak.

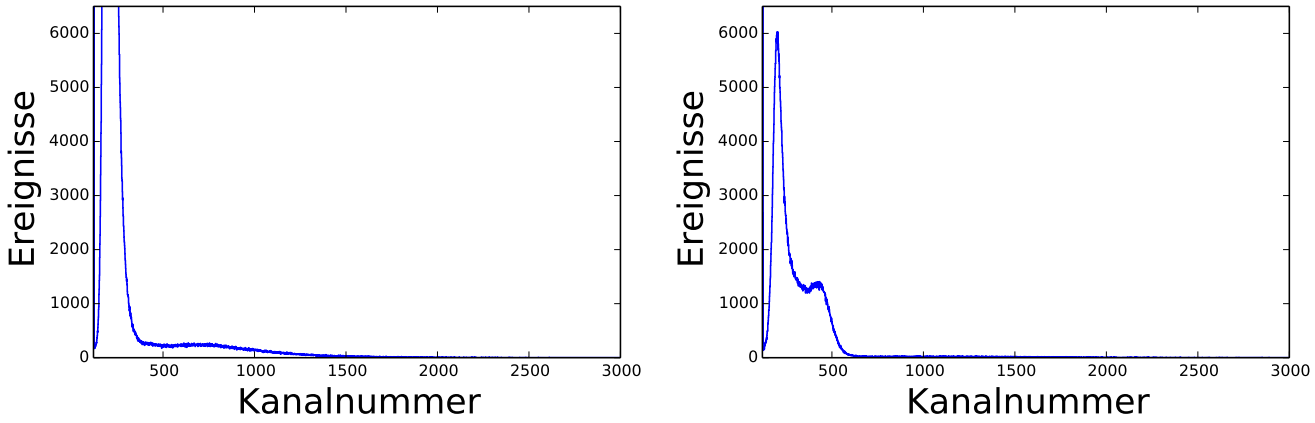


Abbildung 4.3: Energieverlustspektrum von am Kohlenstoff gestreuten Elektronen mit verschiedener Pulsformzeit.

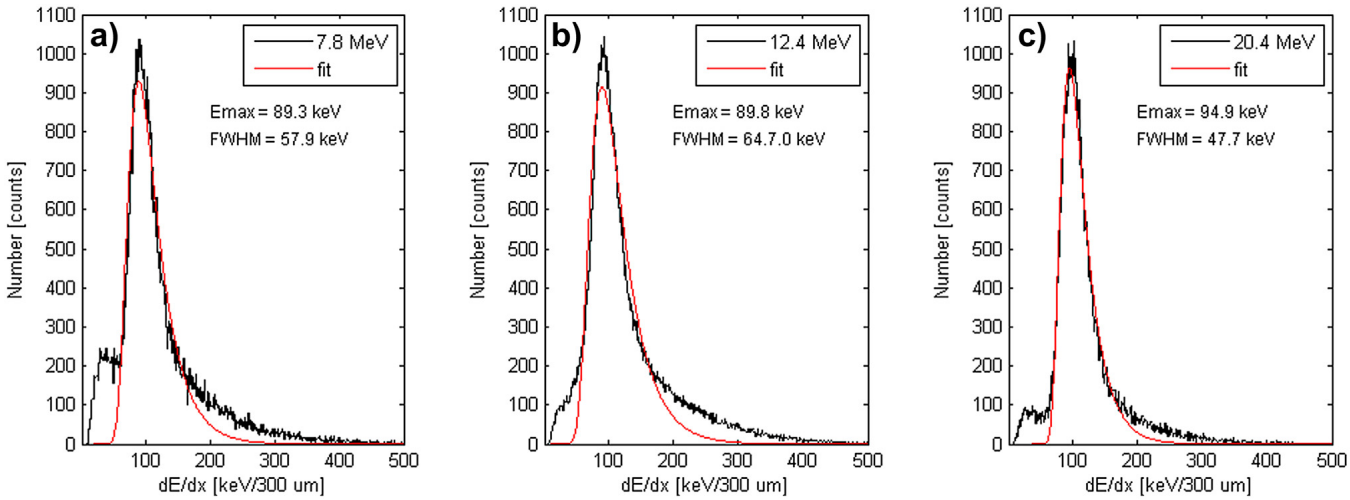


Abbildung 4.4: Elektronenenergieverlustspektrum in Silizium bei einer Elektronenanfangsenergie von a) 7,8 MeV, b) 12,4 MeV und c) 20,4 MeV.

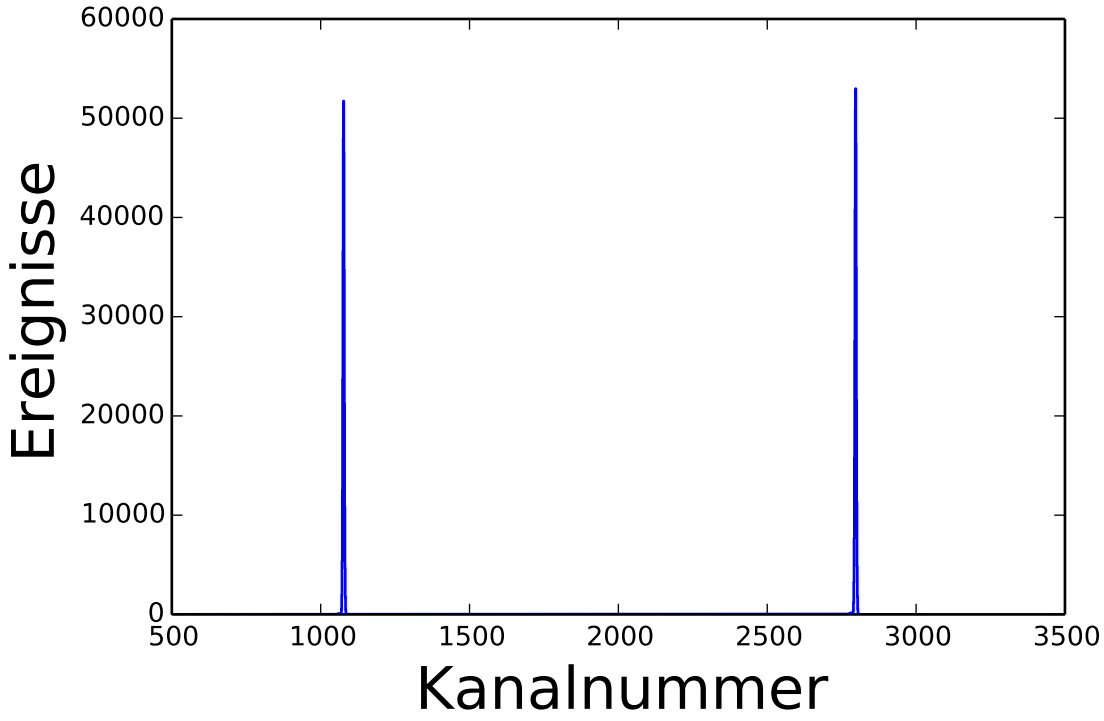


Abbildung 4.5: Spektrum der TDC-Testmessung mit einem Pulsgenerator für eine Zeitspanne von 300 ns und 800 ns.

4.2.2 Test des TDCs

Der TDC wurde im Rahmen dieser Masterarbeit programmiert und ebenfalls getestet. In einer ähnlichen Schaltung, wie beim Test des ADCs wurde auch beim Testen des TDCs ein Pulsgenerator verwendet. Hierbei wurde vom Pulsgenerator ein Startpuls erzeugt sowie zwei Stopppulse. Einer der Stopppulse wurde nach etwa 300 ns nach dem Startpuls erzeugt, ein anderer nach etwa 800 ns. Der Abb. 4.5 kann man entnehmen, dass beide Zeitspannen vom TDC richtig erkannt wurden. Somit verlief auch der Test des TDCs erfolgreich.

5 Zusammenfassung und Ausblick

Im Rahmen dieser Masterarbeit wurden die 30 Siliziumdetektoren in die Streukammer des QCLAM-Spektrometers eingebaut und getestet. Des Weiteren wurde die Elektronik neu verschaltet. Der ADC wurde zuerst mit einem Pulsgenerator, später auch bei einem Elektronenstreuexperiment erfolgreich getestet. Das erhaltene Spektrum wurde mit Elektronenenergieverlustspektren aus einer anderen Arbeit verglichen. Der Vergleich bestätigte die ordnungsgemäße Funktionalität der Elektronik bzw. der Auslese. Des Weiteren wurde ein TDC programmiert und ebenfalls mit einem Pulsgenerator erfolgreich getestet.

In Zukunft müssen alle Siliziumdetektoren im gleichzeitigen Betrieb getestet werden. Dies war bis jetzt nicht möglich, da die Spectroscopy Amplifier hierfür einen Hardwareupgrade brauchten, der nun von der Firma CAEN durchgeführt wurde. Des Weiteren muss die Auslese so programmiert werden, dass die gleichzeitige Auslese des ADCs und TDCs ermöglicht wird. Es besteht weiterhin das Ziel die Detektoren in Koinzidenz mit dem QCLAM-Spektrometer zu betreiben. Die Elektronik wird für das Spektrometer momentan erneuert. Diese muss dann mit der Auslese des Siliziumballs in Einklang gebracht werden. Seit kurzem steht ein Detektoremulator zur Verfügung mit dem auch Koinzidenzexperimente simuliert werden können. Dies bietet eine weitere gute Möglichkeit die Elektronik des Siliziumballs in Koinzidenz zu testen.

Teil II

Vergleich der Photo-absorptionsquerschnitte in relativistischer Protonenstreuung mit elektromagnetischen Proben

6 Einleitung

Riesenresonanzen sind wichtige Untersuchungsobjekte für das Verständnis von kollektiven Phänomenen in Kernen. Baldwin und Klaiber entdeckten die Dipolriesenresonanz (GDR) 1947 bei der Untersuchung von Photodesintegrationsexperimenten [16]. Erste Hinweise auf Riesenresonanzen gab es jedoch bereits 1937 bei Experimenten mit 17 MeV Photonen, die von Bothe und Gentner durchgeführt wurden. Hier wurden Wirkungsquerschnitte an verschiedenen Targets gemessen, die etwa zwei Größenordnungen größer waren als die theoretisch vorhergesagten. Daraus folgerten Bothe und Gentner, dass es in den untersuchten Kernen eine Resonanzabsorption geben muss [17]. In den 1970er Jahren wurden die isoskalare Quadrupolriesenresonanz (GQR) und die isoskalare Monopolriesenresonanz (GMR) experimentell nachgewiesen [18,19]. Nachdem Protonenstrahlen mit hohen Energien verfügbar geworden waren, wurden ab 1980 weitere Arten von Riesenresonanzen entdeckt [20]. Hierbei können diese klassifiziert werden, je nachdem ob es bei der Anregung einen Bahndrehimpulsübertrag ΔL, einen Isospinübertrag ΔT oder einen Spinübertrag ΔS gibt. Wird kein Bahndrehimpuls übertragen, handelt es sich um eine Monopolriesenresonanz. Bei einer Bahndrehimpulsübertragung können höhere Multipolriesenresonanzen angeregt werden, wie z.B. Dipolriesenresonz (ΔL=1), Quadrupolriesenresonanz (ΔL=2) und höhere Ordnungen. Je nach Isospinübertrag führen Protonen und Neutronen Vibrationen bzw. Schwingungen in Phase (ΔT=0) oder außer Phase (ΔT=1) durch. Auch im Spinraum kommt es zu Vibrationen bzw. Schwingungen in Phase (ΔS=0) oder außer Phase (ΔS=1). In Abb. 6.1 sind die wichtigsten Riesenresonanzen dargestellt.

Mit relativistischer Protonenstreuung unter kleinen Winkeln wird bevorzugt die isovektorielle Riesendipolresonanz (IVGDR) angeregt. Das Ziel dieser Masterarbeit ist es nun, aus den vorliegenden Protonenstreudaten Photoabsorptionsquerschnitte zu extrahieren und mit elektromagnetischen Proben zu vergleichen. Hierfür werden zwei verschiedene Methoden verwendet. Die modellunabhängige, sogenannte virtuelle Photonenmethode und eine modellabhängige Methode unter Verwendung von DWBA- und QPM-Rechnungen. Durch den Vergleich der aus den Protonenstreudaten extrahierten Photoabsorptionsquerschnitte mit elektromagnetischen Proben, lässt sich der aus nuklearen Prozessen resultierende Untergrund bestimmen. Allerdings kann man dies nur für absolut extrahierte Photoabsorptionsquerschnitte machen, die bei kleinen Winkeln mit den experimentellen Photoabsorptionsquerschnitten übereinstimmen müssen. In dieser Masterarbeit wird untersucht wie gut die beiden Methoden auf die verschiedenen Kerne anwendbar sind und wo sie ihre Grenzen erreichen.

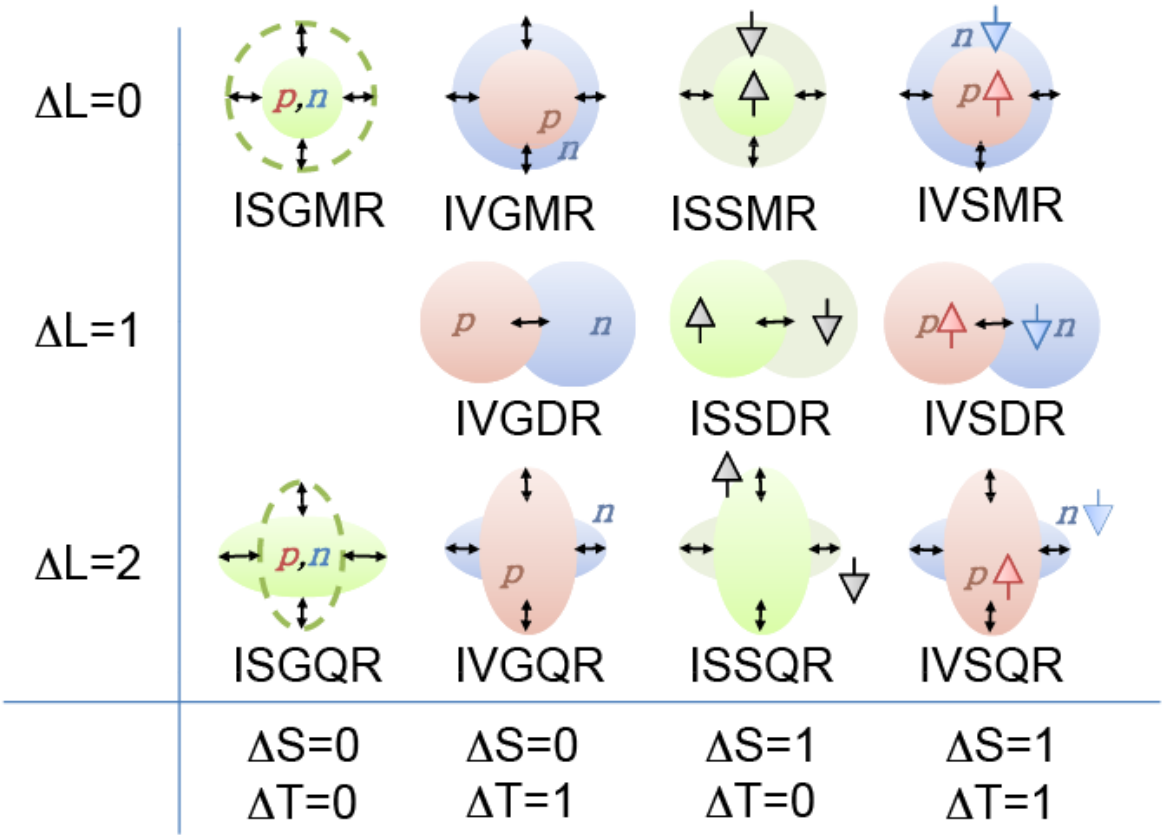


Abbildung 6.1: Die wichtigsten Riesenresonanzen und ihre Interpretationen [21].

7 Theoretische Grundlagen

In diesem Kapitel werden Konzepte und theoretische Modelle vorgestellt, die zur Datenanalyse von Protonenstreudaten benötigt werden. Hierzu zählen die Grundlagen der inelastischen Protonenstreuung, die virtuelle Photonenmethode sowie die Grundlagen der Distorted Wave Born Approximation (DWBA) und des Quasiteilchen-Phonon-Modells (QPM).

7.1 Inelastische Protonenstreuung

Werden Protonen an einem Kern gestreut, so nehmen an diesem Prozess im Wesentlichen zwei Wechselwirkungen teil, nämlich die Nukleon-Nukleon-Wechselwirkung (NN) und die elektromagnetische Wechselwirkung.

Die elektromagnetische Wechselwirkung ist dominant unter kleinen Streuwinkeln und ist für elektromagnetische Anregungen (z.B. Dipolanregungen) verantwortlich. Diese kann klassisch mit Hilfe des Coulombpotentials bzw. semiklassisch durch die virtuelle Photonenmethode beschrieben werden.

Die NN-Wechselwirkung ist unter Vorwärtswinkeln für die Spinflip-Anregungen verantwortlich. Diese kann im Kontext der DWBA mit Hilfe eines effektiven Potentials beschrieben werden.

7.1.1 Nukleon-Nukleon-Wechselwirkung

Die NN-Wechselwirkung bei der inelastischen Streuung von Protonen an einem Kern lässt sich durch die zeitunabhängige Schrödingergleichung beschreiben

$$(H_0 + V)\psi = (H_N + K_0 + V)\psi = E\psi, \tag{7.1}$$

wobei H_0 der Hamiltonoperator des Kerns, K_0 die kinetische Energie des einlaufenden Protons und V das Wechselwirkungspotential zwischen Proton und Kern sind [22]. Die Eigenfunktionen ψ sind durch die Lippmann-Schwinger-Gleichung

$$\psi^{\pm} = \phi^{\pm} + \frac{1}{E - H_0 \pm i\epsilon} V\psi^{\pm} \tag{7.2}$$

gegeben [22]. Hierbei sind $\phi^{\pm}$ die Eigenfunktionen des ungestörten Systems, wobei die Vorzeichen $+$ und $-$ die einlaufenden bzw. die auslaufenden Wellen bezeichnen. Die Übergangswahr-

scheinlichkeit zwischen gestörten und ungestörten Zuständen ist dann durch die Übergangsmatrix

$$T = \langle \phi^- | V | \psi^+ \rangle \tag{7.3}$$

gegeben [22]. Sie steht im folgenden Zusammenhang mit dem differentiellen Wirkungsquerschnitt [22, 23]

$$\frac{\mathrm{d}\sigma}{\mathrm{d}\Omega}(\vec{k}_i, \vec{k}_f) = \left(\frac{\mu_i \mu_f}{2\pi\hbar^2}\right)^2 \frac{|\vec{k}_f|}{|\vec{k}_i|} |T(\vec{k}_f, \vec{k}_i)|^2. \tag{7.4}$$

Hierbei ist $\vec{k}$ der Impuls und μ die reduzierte Masse vor (i) bzw. nach (f) der Streuung. Allerdings gilt die Gleichung in dieser Form nur für Kerne mit Grundzustandsspin $J_i = 0^+$.

Das Potential in Glg. (7.3) lässt sich als Summe der 2-Teilchen-Wechselwirkungen v_n zwischen dem Proton und dem Kernnukleon durch

$$V = \sum_{n=1}^{A} v_n \tag{7.5}$$

beschreiben. In erster Bornscher Näherung ist die Übergangsmatrix T in der gestörten Basis χ durch

$$T \simeq \langle \chi_f^- | \sum_{n=1}^{A} v_n | \chi_i^+ \rangle \simeq \langle \chi_f^- | \sum_{n=1}^{A} t_n | \chi_i^+ \rangle \tag{7.6}$$

gegeben, wobei die 2-Teilchen-Wechselwirkung t_n sich mit Hilfe einer kinematischen Transformation aus der Übergangsmatrix für die Streuung freier Nukleonen gewinnen lässt. Love und Franey [24] haben eine phänomenologische Beschreibung der freien Nukleon-Nukleon-t-Matrix für Einschussenergien von 100 - 800 MeV bestimmt. Die nichtlokale t-Matrix wird hierbei durch einen lokalen Operator repräsentiert. Dieser ist gegeben durch

$$V = V^C(r) + V^{LS}(r)\vec{L} \cdot \vec{S} + V^T(r) S_{12}, \tag{7.7}$$

mit dem Zentralterm $V^C(r)$, dem Spin-Bahn-Term $V^{LS}(r)$ und dem Tensorterm $V^T(r)$. $\vec{L}$ bezeichnet den Drehimpulsoperator, $\vec{S}$ den Spinoperator und S_{12} den Tensoroperator. Der Radialanteil des Zentral-, Spin-Bahn- und Tensorterms wird durch die Summe von Yukawapotentialen ausgedrückt, wobei im Falle des Tensorterms noch mit r^2 multipliziert wird

$$V^C(r) \;=\; \sum_{i=1}^{N_C} V_i^C \frac{\exp(r/R_i)}{r/R_i}, \tag{7.8}$$

$$V^{LS}(r) = \sum_{i=1}^{N_{LS}} V_i^{LS} \frac{\exp(r/R_i)}{r/R_i}, \tag{7.9}$$

$$V^{T}(r) = \sum_{i=1}^{N_{T}} V_i^{T} r^2 \frac{\exp(r/R_i)}{r/R_i}. \tag{7.10}$$

Die Variablen V_i sind hierbei die komplexen Stärkeparameter und R_i die Reichweiteparameter. Die t-Matrix im NN-System ist gegeben durch

$$t_{NN} = \int e^{-i\vec{k}'\cdot\vec{r}} V[1+(-1)^l P] e^{i\vec{k}\cdot\vec{r}} \mathrm{d}^3 r, \tag{7.11}$$

wobei P der Paritätsoperator ist und l den relativen Bahndrehimpuls im NN-System bezeichnet. Bei kleinen Impulsüberträgen ist der Spin-Bahn-Term vernachlässigbar, der skalare spinunabhängige Anteil des Tensorterms ist in gleicher Größenordnung wie der isovektorielle spinabhängige Anteil des Zentralterms. In Abb. 7.1 sind die Zentraltermanteile der Nukleon-Nukleon-t-Matrix bei einem Impulsübertrag von $q = 0$ MeV dargestellt. Hierbei ist t_0^C der isoskalare spinunabhängige Anteil, $t_{\sigma\tau}^C$ der isovektorielle spinabhängige Anteil, t_τ^C der isovektorielle spinunabhängige Anteil und t_σ^C der isoskalare spinabhängige Anteil. Die einzelnen Zusammensetzungen

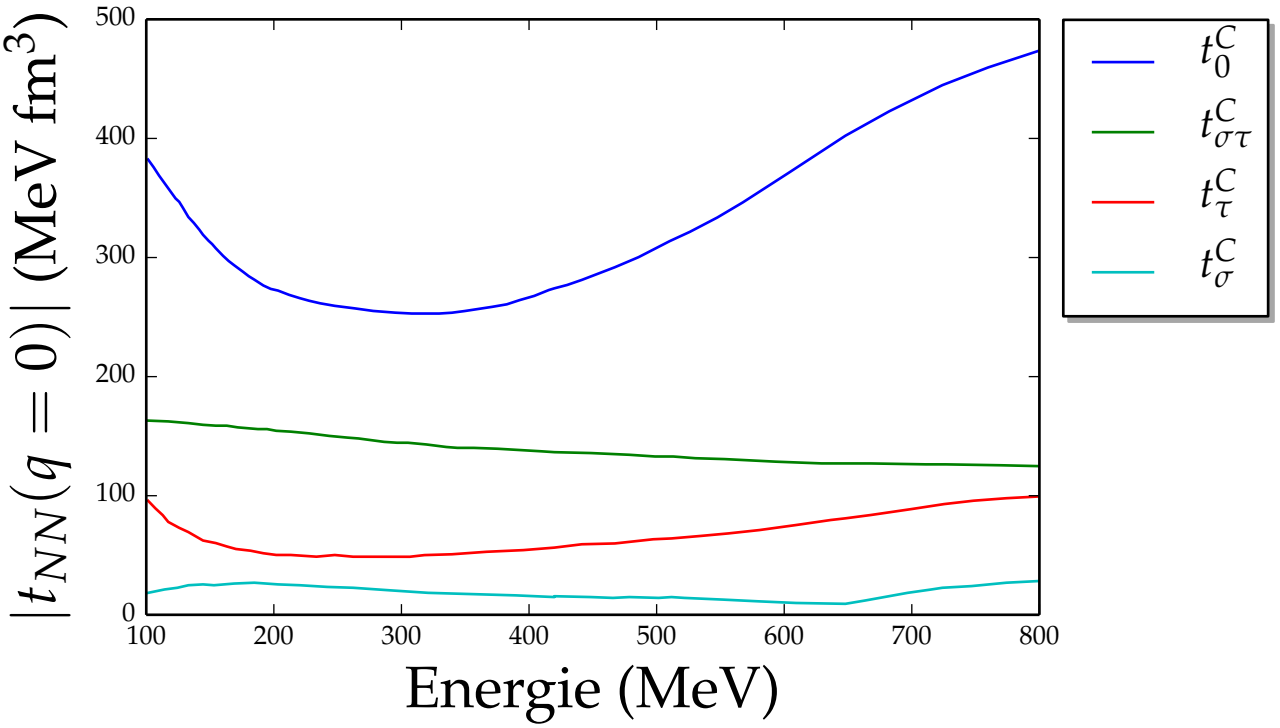


Abbildung 7.1: Energieabhängigkeit der Zentralterme der Nukleon-Nukleon-t-Matrix nach W. G. Love und M. A. Franey bei einem Impulsübertrag von $q = 0$ MeV [24].

der Zentraltermanteile können dem Appendix in [24] entnommen werden. Qualitativ erkennt man in Abb. 7.1, dass bei einer Projektilenergie von 300 MeV der isoskalare spinunabhängige Anteil t_0^C minimal wird. Somit können in diesem Energiebereich spin-isospinflip-Übergänge studiert werden, da $t_{\sigma\tau}^C$ dann relativ groß wird. Die Anteile t_τ^C und t_σ^C hingegen sind über den ganzen Energiebereich relativ klein, wobei t_τ^C zwischen 200 und 300 MeV ein Minimum erreicht.

Die Eigenschaften von t_0^C wurden mit Hilfe elastischer Protonenstreuung in [25–27] untersucht. Die von $t_{\sigma\tau}^C$ und t_τ^C mit Hilfe von (p, n)-Reaktionen in [28–31]. Die Eigenschaften von t_σ^C wurden in [32] diskutiert.

7.1.2 Distorted Wave Born Approximation

Im Rahmen dieser Arbeit wurde das Programm *DWBA07* verwendet [33]. Die Basis dieses Programms bildet hierbei die Distorted Wave Born Approximation (DWBA).

In der DWBA wird die Übergangsmatrix $T(\vec{k}_f, \vec{k}_i)$ in einer Basis gestörter Wellenfunktionen $\chi^\pm$ gebildet, die durch die Lippman-Schwinger-Gleichung

$$\chi^\pm = \phi^\pm + \frac{1}{E - H_0 \pm i\epsilon} V_0(\vec{r} - \vec{r}_N)\chi^\pm \tag{7.12}$$

gegeben ist [22]. Hierbei beschreibt $V_0(\vec{r} - \vec{r}_N)$ die Wechselwirkung zwischen Projektil und Kern während des Einlaufens des Projektils. Man erhält eine lokale Darstellung des Potentials, indem man dieses mit der Grundzustandsdichte $\rho_0(\vec{r}_N)$ des Kerns faltet. Das daraus resultierende, sogenannte optische Potential, ist dann gegeben durch [22]

$$U_0(\vec{r}) = \int \rho_0(\vec{r}_N) V_0(\vec{r} - \vec{r}_N) \mathrm{d}^3 r_N. \tag{7.13}$$

Eine andere Herangehensweise zur Bestimmung des optischen Potentials beruht auf der Anpassung an Daten aus elastischer Streuung. Damit erhält man ein phänomenologisches optisches Potential, welches üblicherweise aus einem Real- und Imaginärteil eines Zentral- und Spin-Bahn-Terms besteht. Eine übliche Parametrisierung für den nichtrelativistischen Fall ist durch [22]

$$U(r) = V_C(r) + V_0 f(x_0) + iW_0 f(x_0) - 2(V_{so} + iW_{so}) \frac{1}{r} \frac{\mathrm{d}}{\mathrm{d}r} f(x_{so}) \vec{L} \cdot \vec{\sigma} \tag{7.14}$$

gegeben, mit

$$f(x_i) = \frac{1}{1 + \exp(x_i)} \qquad \text{und} \qquad x_i = \frac{r - r_i A^{1/3}}{a_i}. \tag{7.15}$$

Hierbei ist V_C das Coulomb-Potential, V_0 und W_0 sind die Amplituden des Zentralterms und V_{so} bzw. W_{so} die des Spin-Bahn-Terms. Die Radialabhängigkeit wird durch die Wood-Saxon-Funktionen $f(x_i)$ beschrieben, wobei r_i der Radius und a_i die Oberflächendicke ist. Alle Parameter, d.h. die Amplituden, Radien und Oberflächendicken werden so angepasst, dass die Wirkungsquerschnitte der elastischen Streuung möglichst gut beschrieben werden.

7.1.3 Elektromagnetische Wechselwirkung

Für Stoßparameter, die die Summe aus Projektilradius und Targetkernradius übersteigen, dominiert die elektromagnetische Wechselwirkung. Dies ist bei kleinen Streuwinkeln gegeben, da in diesem Fall die Projektilteilchen kaum abgelenkt werden. Hierbei wird das Projektilteilchen als Punktteilchen betrachtet, welches sich auf einer hyperbolischen Bahn im elektrischen Feld des Targetkerns bewegt. In Abb. 7.2 ist diese Situation veranschaulicht.

Der differentielle Wirkungsquerschnitt ist durch die wohlbekannte Rutherfordsche Streuformel gegeben

$$\left(\frac{\mathrm{d}\sigma}{\mathrm{d}\Omega}\right)_{Ruth} = \left(\frac{1}{4\pi\varepsilon_0}\frac{Z_1 Z_2 e^2}{4E_0}\right)\frac{1}{\sin^4\left(\frac{\theta}{2}\right)}. \tag{7.16}$$

Hierbei ist Z_1 die Ladung des Projektils, Z_2 die Ladung des Targetkerns, e die Elementarladung, ε_0 die Dielektrizitätskonstante, E_0 die Energie des Projektils und θ der Streuwinkel. Die Anregung des Zustandes $|f\rangle$ aus dem Zustand $|i\rangle$ ist dann gegeben durch

$$\left(\frac{\mathrm{d}\sigma}{\mathrm{d}\Omega}\right) = \left(\frac{\mathrm{d}\sigma}{\mathrm{d}\Omega}\right)_{Ruth} P_{i\to f}, \tag{7.17}$$

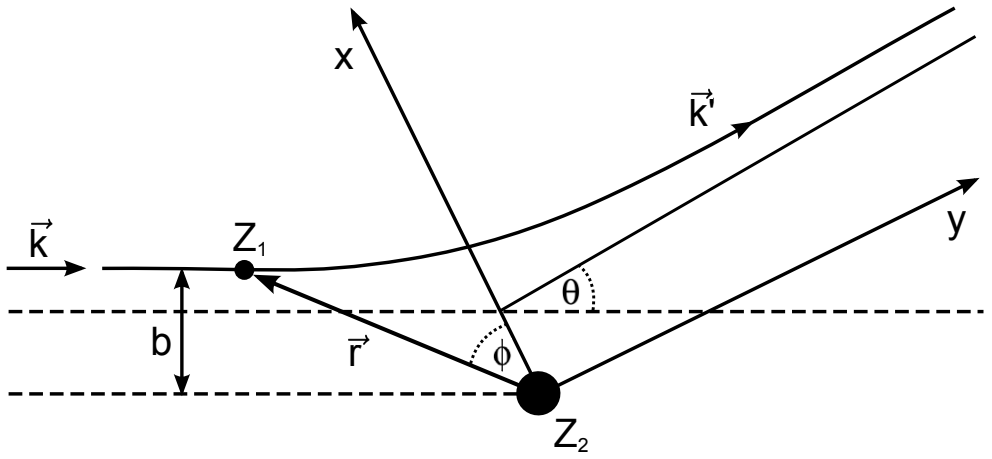


Abbildung 7.2: Hyperbolische Bahn eines Projektilteilchens in der klassischen Beschreibung. Hierbei ist Z_1 die Ladung des Projektils, Z_2 die Ladung des Targetkerns, θ der Streuwinkel, $\vec{k}$ der Impuls, $\vec{r}$ die Position des Projektils und b der Stoßparameter [34].

wobei $P_{i\to f}$ die Wahrscheinlichkeit für die Anregung ist [35]. Mit der Annahme, dass das zeitabhängige elektromagnetische Feld $V(\vec{r}(t))$ des Projektils den Targetkern nur schwach anregt, lässt sich die Wahrscheinlichkeit $P_{i\to f} = |a_{i\to f}|^2$ mit

$$a_{i\to f} = \frac{1}{i\hbar}\int_{-\infty}^{\infty} e^{i\omega t}\langle f|V(\vec{r}(t))|i\rangle \,\mathrm{d}t, \tag{7.18}$$

störungstheoretisch berechnen [35]. Hierbei ist ω die Übergangsfrequenz und $\hbar$ das Planck'sche Wirkungsquantum. Nach der Multipolentwicklung von $V(\vec{r}(t))$, ist die Anregungsamplitude $a_{i\to f}$ gegeben durch [35]

$$a_{i\to f} = i\sum_{\lambda} \chi_{i\to f}^{\pi\lambda} f_\lambda(\xi). \tag{7.19}$$

Die Funktion $f_\lambda(\xi)$ beschreibt die Abhängigkeit des Wirkungsquerschnitts vom Adiabatizitätsparameter, der durch $\xi = \omega b/\gamma v$ gegeben ist. Hierbei ist b der Stoßparameter, v die Geschwindigkeit des Projektils und γ der Lorentzfaktor. Der Parameter $\chi_{i\to f}^{\pi\lambda}$ ist ein Maß für die Stärke der Wechselwirkung. Er ist gegeben durch [35]

$$\chi_{i\to f}^{\pi\lambda} = \frac{Z_1 e\langle f|\mathcal{M}(\pi\lambda\mu)|i\rangle}{\hbar c\, b^\lambda}. \tag{7.20}$$

$\mathcal{M}(\pi\lambda\mu)$ ist das Matrixelement der Multipolmomente, wobei π die Parität, λ die Multipolarität und μ die magnetische Quantenzahl bezeichnet. Die Variable c repräsentiert die Lichtgeschwindigkeit. Der totale Wirkungsquerschnitt kann dann folgendermaßen berechnet werden [35]

$$\sigma = 2\pi\int_{R}^{b_a} |\chi_{i\to f}^{\pi\lambda}|^2 b \,\mathrm{d}b. \tag{7.21}$$

Die untere Integrationsgrenze R ist hierbei die Summe aus Projektil- und Targetkernradius. Die obere Integrationsgrenze b_a ist der Stoßparameter, bei dem der Adiabatizitätsparameter $\xi = 1$ entspricht. Einsetzen von Glg. (7.20) in Glg. (7.21) führt schließlich zum folgenden Ausdruck für den totalen Wirkungsquerschnitt [35]

$$\sigma_{\pi\lambda} \approx \left(\frac{Z_1 e^2}{\hbar c}\right)^2 \frac{B(\pi\lambda, 0\to\lambda)}{e^2 R^{2\lambda}}\pi R^2 \begin{cases} (\lambda-1)^{-1}, & \text{wenn } \lambda \geq 2, \\ 2\ln(b_a/R), & \text{wenn } \lambda = 1. \end{cases} \tag{7.22}$$

Der letzten Gleichung kann man entnehmen, dass eine Messung des Wirkungsquerschnitts die reduzierte Übergangsstärke $B(\pi\lambda)$ liefert. Andererseits lässt sich unter Benutzung theoretischer Übergangsstärken ein modellabhänger Wirkungsquerschnitt berechnen.

7.2 Virtuelle Photonenmethode

Wird eine Ladung an einem Kern gestreut, so kann diese Streuung klassisch durch die Coulombstreuung beschrieben werden. Im Rahmen der Quantenelektrodynamik (QED) kann die Coulombanregung allerdings auch so interpretiert werden, als ob virtuelle Photonen vom Kern absorbiert werden. Diese entsprechen dann der Anzahl realer Photonen, die den gleichen Effekt auf den Kern haben würden. Der Wirkungsquerschnitt für Coulombanregung ist gegeben durch

$$\sigma_{i\to f} = \sum_{\pi\lambda} \int N_{\pi\lambda}(E_\gamma)\sigma_\gamma^{\pi\lambda}(E_\gamma)\frac{\mathrm{d}E_\gamma}{E_\gamma}, \tag{7.23}$$

mit der virtuellen Photonenzahl $N_{\pi\lambda}(E_\gamma)$, dem Photoabsorptionsquerschnitt $\sigma_\gamma^{\pi\lambda}(E_\gamma)$ und der Photonenenergie E_γ [36]. Außerdem lässt sich der Wirkungsquerschnitt durch

$$\sigma_{i\to f} = \left(\frac{Z_1 e^2}{\hbar c}\right)^2 \sum_{\pi\lambda\mu} k^{2(\lambda-1)} \frac{B(\pi\lambda, I_i \to I_f)}{e^2} \left| G_{\pi\lambda\mu}\left(\frac{c}{v}\right)\right|^2 g_\mu(\xi) \tag{7.24}$$

angeben, wobei Z_p die Ladungszahl des Projektils, e die Elementarladung, $\hbar$ das Plancksche Wirkungsquantum, c die Lichtgeschwindigkeit, k die Wellenzahl, v die Geschwindigkeit des Projektils und $B(\pi\lambda, I_i \to I_f)$ die reduzierte Übergangswahrscheinlichkeit ist [35]. Die Funktion $G_{\pi\lambda\mu}$ kann durch assoziierte Legendrepolynome ausgedrückt werden. Explizite Werte dieser Funktion können dem Appendix B aus [35] entnommen werden. Die Funktion $g_\mu(\xi)$ ist durch

$$g_\mu(\xi) = \pi\xi^2 \left(|K_{\mu+1}(\xi)|^2 - |K_\mu(\xi)|^2 - \frac{2\mu}{\xi} K_{\mu+1}(\xi) K_\mu(\xi) \right) \tag{7.25}$$

gegeben [35]. K_μ ist hierbei die modifizierte Besselfunktion μ-ter Ordnung, ξ der Adiabatizitätsparameter und μ die reduzierte Masse. Der Photoabsorptionsquerschnitt ist gegeben durch

$$\sigma_\gamma^{\pi\lambda}(E_\gamma) = \frac{(2\pi)^3(\lambda+1)}{\lambda[(2\lambda+1)!!]^2} \sum_f \rho_f(\epsilon) k^{2\lambda-1} B(\pi\lambda), \tag{7.26}$$

wobei $\rho_f(\epsilon)$ die Zustandsdichte der Endzustände im Targetkern mit der Energie $E_f = E_i + \epsilon$ ist [37]. Durch das Einsetzen der Gl. (7.26) in Gl. (7.23) und Gleichsetzen mit Gl. (7.24) erhält man die virtuelle Photonenzahl

$$N_{\pi\lambda}(E_\gamma) = Z_1^2 \alpha \frac{\lambda[(2\lambda+1)!!]^2}{(2\pi)^3(\lambda+1)} \sum_\mu \left| G_{\pi\lambda\mu}\left(\frac{c}{v}\right)\right|^2 g_\mu(\xi), \tag{7.27}$$

wobei α die Feinstrukturkonstante ist [37]. Für die differentielle virtuelle Photonenzahl der E1-Anregung erhält man mit Hilfe der Werte aus [35] für $G_{\pi\lambda\mu}$ und g_μ

$$\frac{\mathrm{d}N_{E1}}{\mathrm{d}\Omega} = \frac{Z_1^2\alpha}{4\pi^2}\zeta^2\varepsilon^4\left(\frac{c}{\gamma v}\right)^2\left(K_1^2(x) + \frac{1}{\gamma}K_0^2(x)\right), \tag{7.28}$$

mit $\zeta = \omega a/v$ und $a = Z_1 Z_2 e^2/\mu v^2$, wobei μ die reduzierte Masse ist [38]. Der Parameter $\varepsilon = \sin^{-1}(\theta/2)$ ist der Exzentrizitätsparameter. Das Argument der modifizierten Besselfunktionen ist durch

$$x = \left(\frac{\varepsilon\zeta}{\gamma}\right)\cos\left(\frac{\theta}{2}\right) \tag{7.29}$$

gegeben, wobei θ der Streuwinkel ist [38]. Für E2- und M1-Anregungen können die differentiellen virtuellen Photonenzahlen nur näherungsweise mit

$$\frac{\mathrm{d}N_{E2}}{\mathrm{d}\Omega} = \frac{Z_1^2\alpha}{4\pi^2}\left(\frac{c}{v}\right)^4\varepsilon^2\exp\left(-\frac{\pi\zeta}{\gamma}\right)\left(\frac{4}{\gamma^2}(K_1^2 + xK_0K_1 + x^2K_0^2) + x^2\left(2 - \frac{v^2}{c^2}\right)^2 K_1^2\right), \tag{7.30}$$

$$\frac{\mathrm{d}N_{M1}}{\mathrm{d}\Omega} = \frac{Z_1\alpha}{4\pi^2}\left(\frac{\zeta}{\gamma}\right)^2\varepsilon^4\exp\left(-\frac{\zeta\pi}{\gamma}\right)K_1^2 \tag{7.31}$$

angegeben werden, wobei das Argument der Besselfunktion in diesem Falle durch $x = \zeta\varepsilon/\gamma$ gegeben ist [38]. In Abb. 7.3 sieht man als Beispiel die differentielle virtuelle Photonenzahl für E1-, E2- und M1-Anregungen für den Kern ^{40}Ca als Funktion des Streuwinkels bei 19 MeV (Schwerpunktsenergie der Riesenresonanz), berechnet mit Hilfe der Glgn. (7.28, 7.30, 7.31). Nun kann man den doppeltdifferentiellen Wirkungsquerschnitt in Abhängigkeit der differentiellen virtuellen Photonenzahl als

$$\frac{\mathrm{d}^2\sigma}{\mathrm{d}\Omega\mathrm{d}E_\gamma} = \frac{1}{E_\gamma}\sum_{\pi\lambda}\frac{\mathrm{d}N_{\pi\lambda}}{\mathrm{d}\Omega}\sigma_\gamma^{\pi\lambda} \tag{7.32}$$

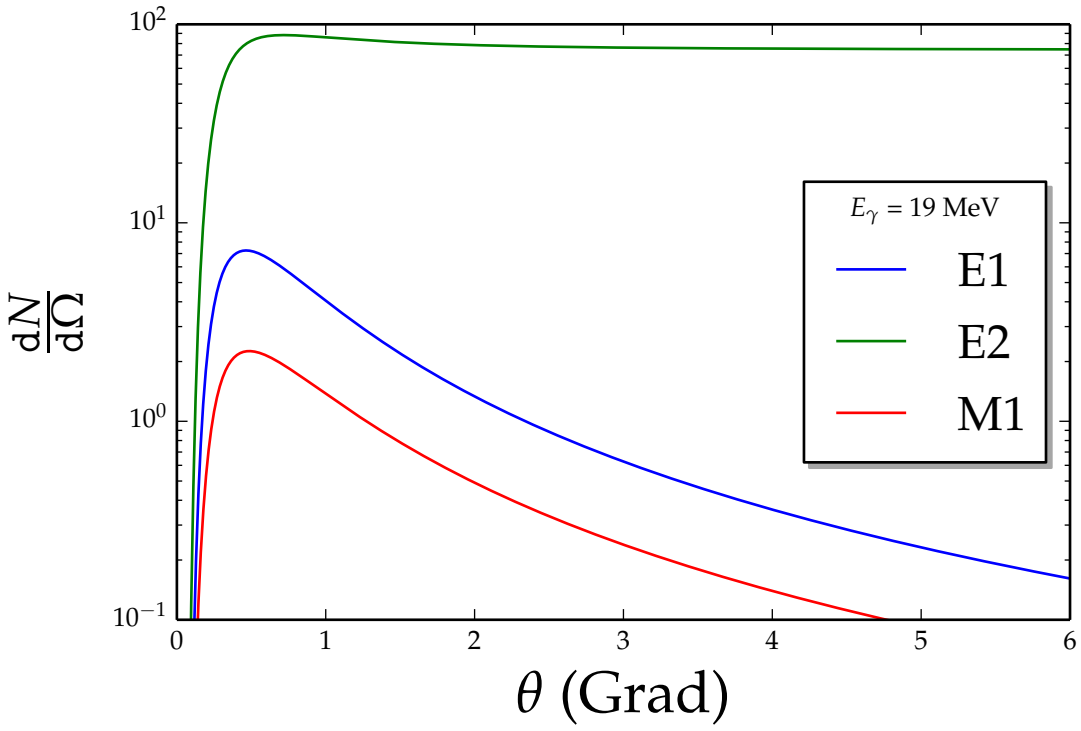


Abbildung 7.3: Differentielle virtuelle Photonenzahlen für Streuwinkel von 0° bis 6° bei einer Anregungsenergie von 19 MeV.

ausdrücken [38]. Diese Gleichung kann nun angewendet werden um aus Photoabsorptionsquerschnitten die entsprechenden doppeltdifferentiellen Wirkungsquerschnitte und umgekehrt zu berechnen. Rechnungen für verschiedene Kerne und die Interpretation der Ergebnisse werden in Kap. 9 diskutiert.

7.3 Quasiteilchen-Phonon-Modell

Im Rahmen dieser Masterarbeit wurden unter anderem DWBA-Rechnungen durchgeführt. Hierfür werden als Eingabeparameter unter anderem reduzierte Übergangswahrscheinlichkeiten benötigt. Eine Möglichkeit diese theoretisch zu berechnen, ist die Anwendung des Quasiteilchen-Phonon-Modells. Das Modell hat sich in der Vergangenheit als sehr erfolgreich bei der Beschreibung kollektiver Anregungen in mittelschweren und schweren Kernen erwiesen [39–42]. Im QPM werden die Kernanregungen durch die Erzeugung von Teilchen-Loch-Paaren beschrieben. Bedingt durch eine ganzzahlige Änderung der Quantenzahlen bei Teilchen-Loch-Übergängen, werden diese durch Erzeugung von Bosonen beschrieben. Derartige Bosonen werden als Phononen bezeichnet. Die einfachsten Übergänge sind 1-Phonon-Übergänge. Im Rahmen dieses Modells können jedoch auch Multi-Phonon-Übergänge miteinbezogen werden, allerdings wird in der Praxis aufgrund der langen Rechenzeit selten über 3-Phonon-Übergänge hinausgegangen. Die QPM-Rechnung lässt sich in vier Schritte unterteilen. Im ersten Schritt werden die

Eigenwerte der Einteilchenenergien und die Wellenfunktionen des Wood-Saxon-Potentials berechnet. Dann wird im zweiten Schritt die kanonische Bogoliubov-Transformation durchgeführt, womit man Teilchenoperatoren zu Quasiteilchenoperatoren konvertiert. Im dritten Schritt wird die Phononbasis konstruiert, die mittels Random-Phase-Approximations (RPA) berechnet wird. Schließlich werden im vierten und letzten Schritt die Quasiteilchen-Phonon-Wechselwirkungen miteinbezogen. Im Folgenden werden die einzelnen Schritte des Quasiteilchen-Phonon-Modells kurz aufgezeigt, ausführliche Darstellungen finden sich in [43, 44].

Betrachtet man Nukleonen, die sich in einem mittleren Feld bewegen und miteinander durch die Restwechselwirkung interagieren, ergibt sich folgender Hamiltonoperator

$$H = H_{s.p.} + H_{pair} + H_{r.i.}, \tag{7.33}$$

wobei hier der Einfachheit halber der Hamiltonoperator für gg-Kerne betrachtet wird.

Der Operator $H_{s.p.}$ beschreibt das mittlere Feld, dem die Protonen (p) und Neutronen (n) ausgesetzt sind. Mit Hilfe der zweiten Quantisierung lässt er sich durch Erzeugungsoperatoren $a^+_{jm\tau}$ und Vernichtungsoperatoren $a_{jm\tau}$ in der Form

$$H_{s.p.} = \sum_{\tau}^{n,p} \sum_{j,m} E_{j\tau} a^+_{jm\tau} a_{jm\tau} \tag{7.34}$$

darstellen, wobei $j \equiv [n, l, j]$ und m die Quantenzahlen der Einteilchenbasis sind. $E_{j\tau}$ ist die Energie des Einteilchenzustands.

Der zweite Operator H_{pair} in Glg. (7.33) berücksichtigt die Paarungsenergie in nichtmagischen Kernen. Er ist gegeben durch

$$H_{pair} = \sum_{\tau}^{n,p} \sum_{j,j'} G^{(0)}_{\tau} \sqrt{(2j+1)(2j'+1)} [a^+_{jm\tau} a^+_{j-m\tau}]_{00} [a_{j'-m'\tau} a_{j'm'\tau}]_{00}, \tag{7.35}$$

mit

$$[a^+_j a^+_{j'}]_{\lambda\mu} = \sum_{m,m'} C^{\lambda\mu}_{jmj'm'} a^+_{jm}\, a^+_{j'm'}. \tag{7.36}$$

Hierbei sind $C^{\lambda\mu}_{jmj'm'}$ die Clebsch-Gordan-Koeffizienten, $G^{(0)}_{\tau}$ ist ein konstantes Matrixelement, welches das Monopol-Feld der Paarungskraft beschreibt. Dieses wird so gewählt, dass die experimentell bekannte Paarungsenergie reproduziert wird.

Die Restwechselwirkung $H_{r.i.}$ wird nach Multipolen entwickelt und in separabler Form folgendermaßen dargestellt

$$H_{r.i.}^{(p-h)} = \sum_{\lambda\mu} \sum_{\tau\rho}^{\pm 1} (\kappa_0^{(\lambda)} + \rho\kappa_1^{(\lambda)}) M_{\lambda\mu}^{+}(\tau) M_{\lambda\mu}(\tau\rho). \tag{7.37}$$

Die Modellparameter $\kappa_i^{(\lambda)}$ legen hierbei die Stärke der isoskalaren bzw. isovektoriellen Restwechselwirkung fest, $\rho = \pm 1$ unterscheidet zwischen isoskalaren und isovektoriellen Übergängen. Für Zustände mit natürlicher Parität hat der Multipoloperator die Form

$$M_{\lambda\mu}^{+}(\tau) = \sum_{j,m,j',m'} \langle jm\tau | i^{\lambda} f_{\lambda}^{\tau}(r) Y_{\lambda\mu}(\Omega) | j'm'\tau \rangle a_{jm\tau}^{+} \, a_{j'm'\tau} \tag{7.38}$$

und für Zustände mit unnatürlicher Parität

$$M_{\lambda\mu}^{+}(\tau) = \sum_{j,m,j',m',lm_1} \langle jm\tau | i^{l} f_{l}^{\tau}(r) [\vec{\sigma} \cdot \vec{Y}_{lm_1}(\Omega)]_{\lambda\mu} | j'm'\tau \rangle a_{jm\tau}^{+} \, a_{j'm'\tau}. \tag{7.39}$$

Die Funktion $f_{\lambda}^{\tau}(r)$ ist der radiale Formfaktor, der in der Praxis als r^{λ} oder als Ableitung des Zentralpotentials des mittleren Feldes $f_{\lambda}^{\tau}(r) = \mathrm{d}U^{\tau}(r)/\mathrm{d}r$ angenommen wird.

Die Lösung des Hamiltonoperators aus Glg. (7.33) erhält man durch Diagonalisierung desselben. Als erstes werden die ersten beiden Terme diagonalisiert. Hierfür wird die kanonische Bogoliubov-Transformation angewendet, mit der man die Teilchenoperatoren zu Quasiteilchenoperatoren transformiert

$$a_{jm\tau}^{+} = u_j \, \alpha_{jm\tau}^{+} + (-1)^{j-m} v_j \alpha_{j-m\tau}. \tag{7.40}$$

Die Größen u_j^2 und v_j^2 entsprechen hierbei den Besetzungswahrscheinlichkeiten der Teilchen bzw. Löcher im Zustand j. Der Grundzustand des Kerns wird als Quasiteilchenvakuum $\alpha_{jm\tau}|\rangle_q \equiv 0$ angenommen. Die Minimierung der Grundzustandsenergie

$$\delta \left\{ {}_q\langle | H_{s.p.} + H_{pair} | \rangle_q + \sum_j \mu_j (u_j^2 + v_j^2 - 1) \right\} = 0, \tag{7.41}$$

mit den Lagrangemultiplikatoren μ_j liefert dann die BCS-Gleichungen [45]. Die Lösung letzterer führt zu den Korrelationsfunktionen $C_\tau = G_\tau^{(0)} \sum_j u_j v_j$ und den chemischen Potentialen λ_τ

für Protonen- und Neutronensysteme. Die Koeffizienten der Bogoliubov-Transformation können dann auf folgende Weise bestimmt werden

$$v_j^2 = \frac{1}{2}\left(1 - \frac{E_{j\tau} - \lambda_\tau}{\epsilon_{j\tau}}\right), \tag{7.42}$$

$$u_j^2 = 1 - v_j^2, \tag{7.43}$$

mit der Quasiteilchenenergie $\epsilon_{j\tau} = \sqrt{C_\tau^2 + (E_{j\tau} - \lambda_\tau)^2}$. Nun können die ersten beiden Hamiltonoperatoren aus Glg. (7.33) in folgender Form geschrieben werden

$$H_{s.p.} + H_{pair} = \sum_\tau^{n,p} \sum_{j,m} \epsilon_{j\tau} \alpha^+_{jm\tau} \alpha_{jm\tau}. \tag{7.44}$$

Die einfachsten angeregten Zustände des Kerns sind 2-Quasiteilchen-Zustände $\alpha^+_{jm\tau}\ \alpha^+_{j'm'\tau}|\rangle_q$, die Teilchen-Loch-Übergängen entsprechen. Die beiden Quasiteilchen sind Fermionen, haben jedoch nach der Kopplung einen ganzzahligen Drehimpuls und können somit durch die Bose-Einstein-Statistik beschrieben werden. Dies erlaubt die Einführung eines bequemeren Operators, nämlich des Phononoperators mit der Multipolarität λ und der Projektion μ

$$Q^+_{\lambda\mu i} = \frac{1}{2} \sum_\tau^{n,p} \sum_{j,j'} \left\{ \psi^{\lambda i}_{jj'\tau} [\alpha^+_{j\tau}\ \alpha^+_{j'\tau}]_{\lambda\mu} - (-1)^{\lambda-\mu} \phi^{\lambda i}_{jj'\tau} [\alpha_{j'\tau} \alpha_{j\tau}]_{\lambda-\mu} \right\}, \tag{7.45}$$

wobei der Index i die Phononen durchzählt, die zur selben Multipolarität koppeln. Die Koeffizienten $\psi^{\lambda i}_{jj'\tau}$ und $\phi^{\lambda i}_{jj'\tau}$ erhält man durch Diagonalisieren des Hamiltonoperators im Raum der 1-Phonon-Zustände $Q^+_{\lambda\mu i}|\rangle_{ph}$. Dies erreicht man durch eine Variationsprozedur

$$\delta \left\{ {}_{ph}\langle | Q_{\lambda\mu i} H Q^+_{\lambda\mu i} | \rangle_{ph} - \frac{\omega_{\lambda i}}{2} \left[\sum_{jj'} \{ (\psi^{\lambda i}_{jj'\tau})^2 - (\phi^{\lambda i}_{jj'\tau})^2 \} - 2 \right] \right\} = 0, \tag{7.46}$$

wobei $\omega_{\lambda i}$ die Energie des i-ten Phonons bezeichnet. Die Prozedur liefert die RPA-Gleichungen, deren Lösung für jede Multipolarität λ^π zu einem 1-Phonon-Spektrum führt.

Bisherige Rechnungen liefern bereits gute Ergebnisse bei der Beschreibung globaler Eigenschaften von Resonanzen. Die energiegewichtete Summenregel (EWSR) und die Schwerpunktsenergie werden gut wiedergegeben. Die Hinzunahme von Multi-Phonon-Konfigurationen hat hierauf kaum Einfluss. Wichtig werden Multi-Phonon-Konfigurationen z. B. bei der Beschrei-

bung des wohlbekannten 1^-_1 Zustands, der eine dominante 2-Phonon-Komponente $[2^+_1 \times 3^-_1]_{1^-}$ besitzt. Auch die experimentell beobachtete Fragmentation der Stärke kann nur durch die Hinzunahme von Multi-Phonon-Konfigurationen erklärt werden [44].

8 Protonenstreuexperimente am RCNP

Die in dieser Arbeit verwendeten Protonenstreuspektren wurden am Research Center for Nuclear Physics (RCNP) in Osaka (Japan) aufgenommen. Die Experimentiereinrichtung ist in Abb. 8.1 gezeigt. Für die Kerne ^{28}Si, ^{40}Ca, ^{48}Ca und ^{144}Sm wurde die Ionenquelle NéoMAFIOS (machine fabriquant ions strippé) [46, 47] verwendet, die unpolarisierte Protonen zur Verfügung stellte. Die Kerne ^{96}Mo, ^{120}Sn, ^{154}Sm und ^{208}Pb wurden hingegen mit polarisierten Protonen aus der High Intensity Polarised Ion Source (HIPIS) [48] beschossen. Nach dem Austritt aus der jeweiligen Quelle, werden die Protonen mit Hilfe des Azimuthally Varying Field (AVF)-Zyklotrons auf 54 MeV beschleunigt. Mit Hilfe eines Ringzyklotrons werden die Protonen dann weiter auf die Zielenergie von 295 MeV beschleunigt und können dann für verschieden Experimente verwendet werden. Unter ES (east south) in Abb. 8.1 können Experimente mit ultrakalten Neutronen durchgeführt werden [49]. Unter EN (east north) werden Experimente mit instabilen Kernen durchgeführt [50]. In der N0-Sektion (north) werden in (p, n)-Experimenten mit Hilfe eines TOF-Aufbaus Spin-Isospin-Anregungen untersucht [51]. Für die (p, p′)-Experimente, die in dieser Arbeit verwendeten Protonenstreuspektren lieferten, wurde der Protonenstrahl in die WS-Sektion (west south) geleitet. Die in Abb. 8.1 eingezeichneten Polarimeter (BLP1 und BLP2) dienten zur Kontrolle der Polarisation in Streuexperimenten, bei denen polarisierte Protonen verwendet wurden. Nachdem die Protonen am Target gestreut wurden, wurden sie in dem hochauflösenden Grand Raiden Magnetspektrometer detektiert [52]. Das sogenannte Large Acceptance Spectrometer (LAS) [53] war für die Überwachung der vertikalen Strahlposition zuständig.

8.1 Die Spektrometer

Bei den Protonenstreuexperimenten wurden zwei Spektrometer benutzt. Das Grand Raiden Spektrometer und das Large Acceptance Spektrometer. Das Grand Raiden Magnetspektrometer [52] hat die Magnetkonfiguration Q1-SX-Q2-D1-MP-D2(+DSR). Hierbei bezeichnet D die Dipolmagnete, Q die Quadrupolmagnete, SX den Sextupolmagneten und MP den Multipolmagneten. DSR bezeichnet einen Dipolmagneten, der zusätzlich für die Spinrotation benutzt werden kann und für die Messung der Polarisation in longitudinaler Richtung der gestreuten Protonen entwickelt wurde. Das Spektrometer hat eine hohe Impulsauflösung von $p/\Delta p = 37075$ und eine Impulsakzeptanz von $\pm 2,5\%$ [52]. Das Detektorsystem des Grand Raiden besteht aus Driftkammern, Plastikszintillatoren, die als Triggerdetektoren fungieren und dem Fokalebenepolarimeter.

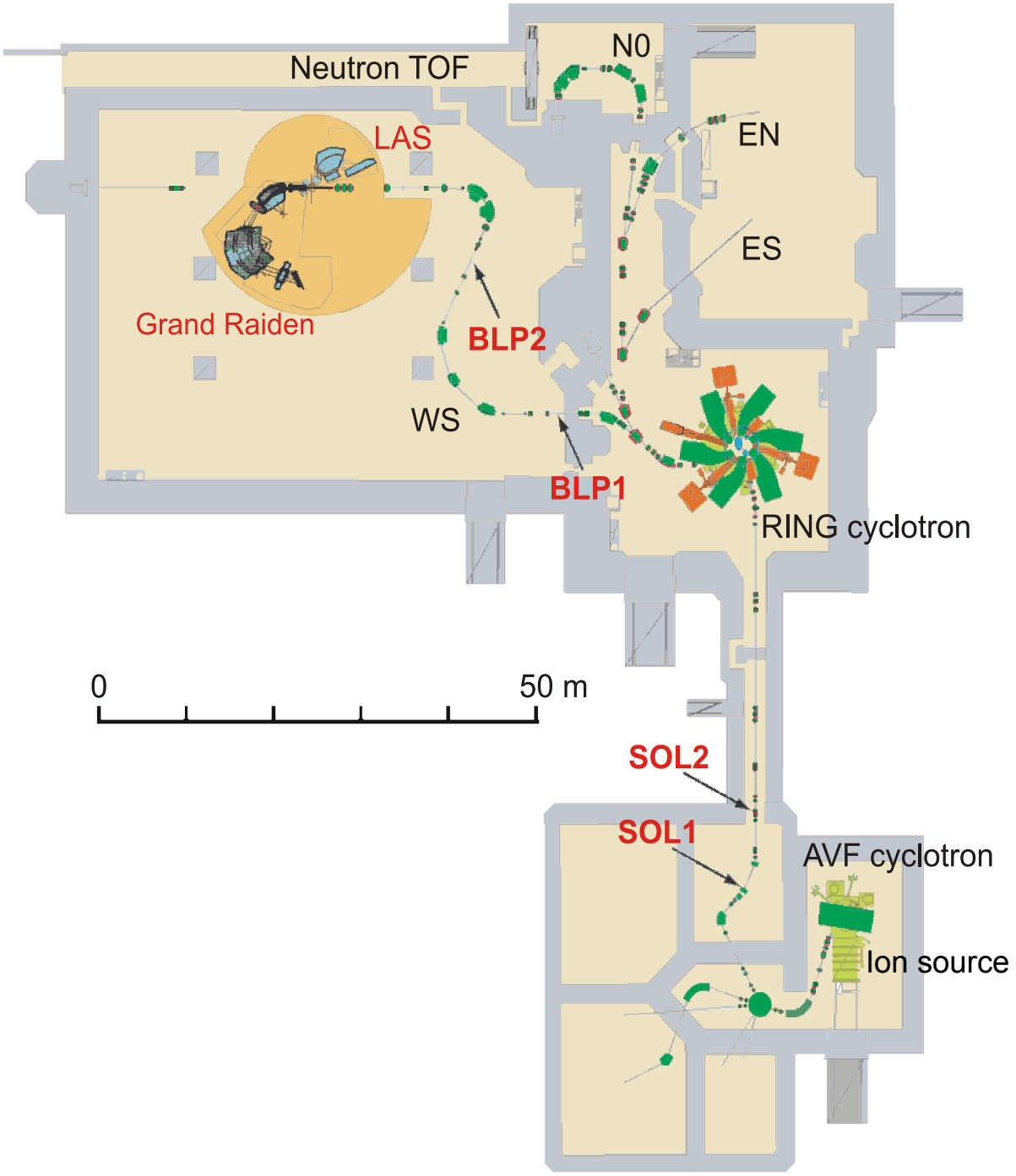


Abbildung 8.1: Schematische Darstellung der Experimentiereinrichtung am RCNP [34].

Das Large Acceptance Spektrometer (LAS) [53] besteht aus einem Quadrupol- und einem Dipolmagneten. Es hat eine hohe Impulsakzeptanz von ±30% und eine Raumwinkelakzeptanz von 20 msr. Das Detektorsystem des Spektrometers besteht aus Vieldrahtdriftkammern und Plastikszintillatoren als Triggerdetektoren. Bei den Protonenstreuexperimenten wurde das Spektrometer benutzt um die Strahlposition in vertikaler Richtung zu überwachen, da diese wichtig für die Kalibrierung der Streuwinkel ist. Die genauen Spezifikationen der beiden Spektrometer sind in Tab. 8.1 zu finden.

Tabelle 8.1: Spezifikationen des Grand Raiden und Large Acceptance Spektrometers [36].

	Grand Raiden	**LAS**
Konfiguration	Q1-SX-Q2-D1-MP-D2(+DSR)	Q-D
Radius der Zentralen Trajektorie	3 m	1,5 m
Maximaler Streuwinkel	180°	70°
Neigung der Fokalebene	45°	57°
Maximale magn. Steifigkeit	5,4 Tm	3,2 Tm
Impulsakzeptanz	±2,5%	±15%
Impulsauflösung	37075	4980
Winkelakzeptanz (horizontal)	±20 mrad	±60 mrad
Winkelakzeptanz (vertikal)	±70 mrad	±100 mrad

9 Datenanalyse

Wie in Kap. 8 beschrieben, wurden die vorliegenden Protonenstreuspektren am RCNP bei einer Strahlenergie von 295 MeV aufgenommen. Der genaue Experimentablauf und die experimentellen Bedingungen sind für jeden einzelnen Kern in [34, 36, 54–59] nachlesbar. Die Spektren liegen als doppeltdifferentielle Wirkungsquerschnitte in 5 keV-Bins vor. Für die Kerne ^{28}Si, ^{40}Ca und ^{48}Ca sind Spektren bei Streuwinkeln von 0,4° bis 14° verfügbar. Da die Coulombanregung unter kleinen Winkeln dominant ist, wurden im Rahmen dieser Arbeit nur die Spektren bei 0,4° verwendet. Die in dieser Arbeit untersuchten Kerne und die dazu gehörigen Streuwinkel sind in Tabelle 9.1 zusammengefasst. Die Streuspektren der einzelnen Kerne können in Appendix B.1 eingesehen werden.

Die Photoabsorptionsquerschnitte wurden der EXFOR-Datenbank (Exchange Format) entnommen [60]. Die experimentellen Daten haben keine feste Binbreite, da die durchschnittliche Binbreite jedoch bei etwa 200 keV liegt, wurden alle Daten mit einer festen Binbreite von 200 keV versehen. Um die Photoabsorptionsquerschnitte zu erhalten, wurden Experimente mit monochromatischen Photonen sowie Bremsstrahlung durchgeführt. Die genaue Vorgehensweise und die experimentellen Bedingungen für die einzelnen Kerne sind in [61–66] beschrieben. Die Spektren der Photoabsorptionsquerschnitte sind in Appendix B.2 dargestellt.

Tabelle 9.1: Auflistung der untersuchten Kerne mit dem jeweiligen Streuwinkelbereich bzw. den dazu gehörigen mittleren Streuwinkeln.

Targetkern	^{28}**Si**	^{40}**Ca**	^{48}**Ca**	^{96}**Mo**	^{120}**Sn**	^{144}**Sm**	^{154}**Sm**	^{208}**Pb**
Streuwinkelbereich	0°-0,7°	0°-0,7°	0°-0,7°	0°-1,1°	0°-2,5°	0°-2,1°	0°-1,7°	0°-2,5°
Mittlerer Streuwinkel	0,4°	0,4°	0,4°	0,6°	1,3°	1,1°	0,9°	1,3°

9.1 Extraktion von Photoabsorptionsquerschnitten unter Verwendung der virtuellen Photonenmethode

Der Photoabsorptionsquerschnitt wurde für alle in Tab. 9.1 aufgeführten Kerne extrahiert. Exemplarisch wird hier die Vorgehensweise für ^{40}Ca aufgezeigt. Die Ergebnisse für alle anderen Kerne können in Appendix B.3 eingesehen werden.

Um den Photoabsorptionsquerschnitt aus den Protonenstreudaten zu extrahieren, kann man die Glg. (7.32) benutzen. Unter der Annahme, dass im Bereich der Riesenresonanz hauptsächlich E1-Anregung stattfindet, vereinfacht sich diese Gleichung zu

$$\frac{\mathrm{d}^2\sigma}{\mathrm{d}\Omega \mathrm{d}E_\gamma} = \frac{1}{E_\gamma}\frac{\mathrm{d}N_{E1}}{\mathrm{d}\Omega}\sigma_\gamma^{E1}. \tag{9.1}$$

Auflösen dieser Gleichung nach σ_γ^{E1} liefert sofort den gewünschten Photoabsorptionsquerschnitt. Allerdings liegen die doppeltdifferentiellen Wirkungsquerschnitte, wie in Tab. 9.1 angegeben, in einem bestimmten Streuwinkelbereich, so dass es sich ein mittlerer Streuwinkel ergibt. Dies hat zur Konsequenz, dass die differentielle virtuelle Photonenzahl über den entsprechenden Streuwinkelbereich gemittelt werden muss. Das geschieht, indem man $\mathrm{d}N_{E1}/\mathrm{d}\Omega$ in Glg. (9.1) durch

$$\overline{N}_{E1} = \frac{\int \frac{\mathrm{d}N_{E1}}{\mathrm{d}\Omega}\mathrm{d}\Omega}{\int \mathrm{d}\Omega} \tag{9.2}$$

ersetzt. Hierbei wird über den vertikalen und horizontalen Winkel integriert. Für ^{40}Ca waren die Winkel $|\theta_h| \leq 0{,}5°$ und $|\phi_v| \leq 0{,}5°$, was gerade dem Streuwinkelbereich von 0°-0,7° entspricht. Des Weiteren wurde die virtuelle Photonenzahl vor der Integration mit einer Gaußfunktion gefaltet. Dadurch wird die endliche experimentelle Winkelauflösung berücksichtigt, die für den vertikalen Winkel $\Delta\theta_h \leq 0{,}1°$ und für den horizontalen Winkel $\Delta\phi_v \leq 0{,}3°$ beträgt [34,36,57]. Zusätzlich wird damit die Singularität, die aufgrund der Coulombstreutheorie bei 0° entsteht, aufgehoben. Ein Vergleich der differentiellen und der gemittelten Photonenzahl der E1-Anregung mit und ohne Faltung ist in Abb. 9.1 dargestellt. Je flacher die Photonenzahl nach dem Maximum abfällt, desto kleiner ist der Faltungseffekt. Bei schweren Kernen hat die Faltung deswegen im Bereich der Riesenresonanz keinen großen Einfluss.

Der Photoabsorptionsquerschnitt kann nun unter Verwendung von Glgn. (9.1) und (9.2) berechnet werden. Das Ergebnis ist in Abb. 9.2 dargestellt, wobei hier der experimentelle Photoabsorptionsquerschnitt zum Vergleich ebenfalls aufgetragen ist. Der Abbildung kann man entnehmen, dass der extrahierte Photoabsorptionsquerschnitt nicht mit dem experimentellen übereinstimmt. Der berechnete Photoabsorptionsquerschnitt ist um etwa einen Faktor 3,4 kleiner als der experimentelle. In der zweiten Abbildung kann man jedoch sehen, dass die Form des extrahierten Spektrums, zumindest für kleine Energien, recht gut wiedergegeben wird. Die Diskrepanz bei hohen Energien hat zwei Ursachen. Einerseits liegen im Protonenstreuspektrum auch Anregungen höherer Multipole vor, denen nicht Rechnung getragen wird. Andererseits verschiebt sich das Maximum der differentiellen Photonenzahl bei höheren Energien zu höheren Winkeln, so dass die semiklassische Beschreibung bei kleinen Winkeln wegen des starken Abfalls der differentiellen Photonenzahl nicht mehr zufriedenstellend ist. Für schwere Kerne

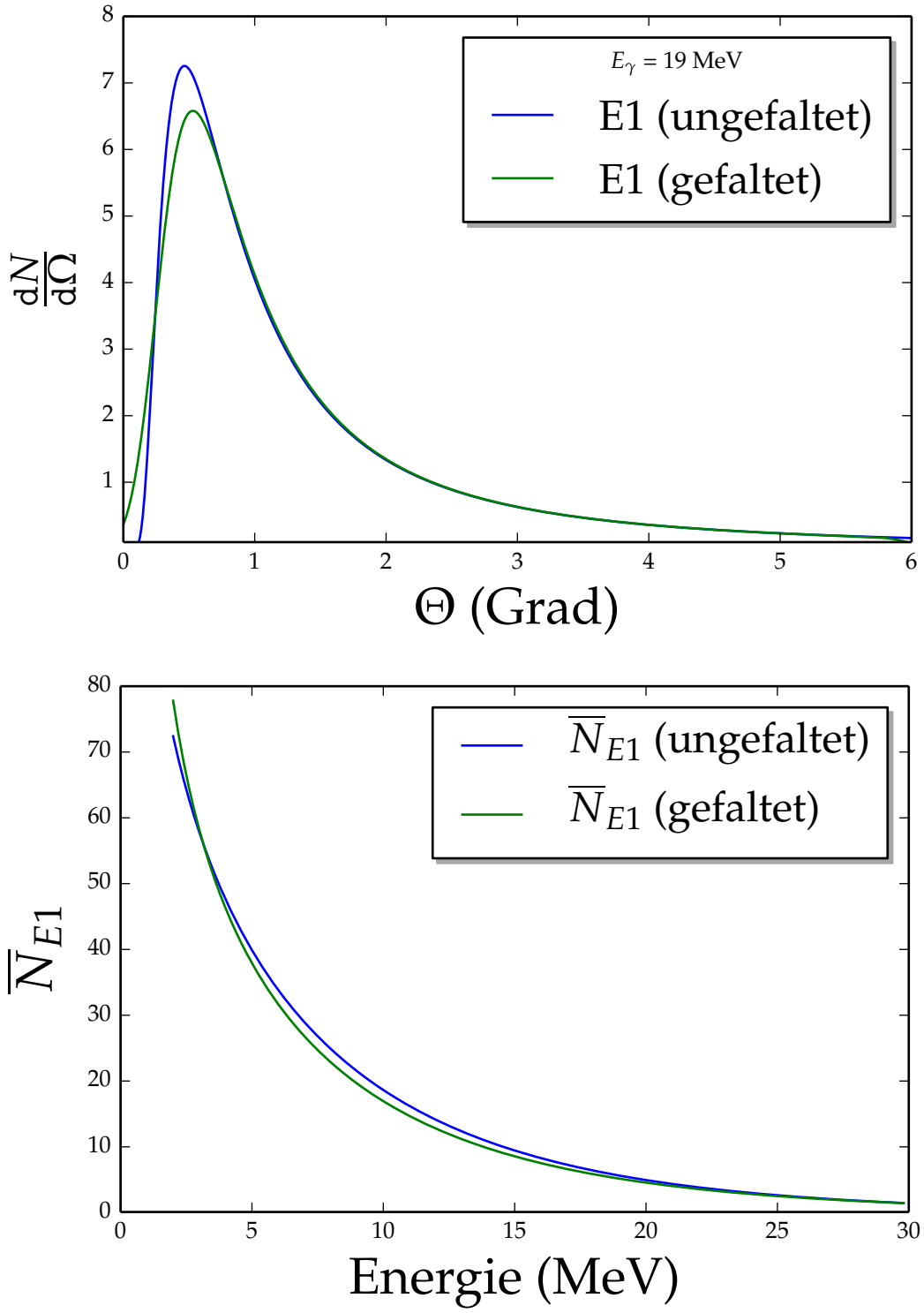


Abbildung 9.1: Gefaltete und ungefaltete differentielle bzw. gemittelte Photonenzahl der E1-Anregung in ^{40}Ca.

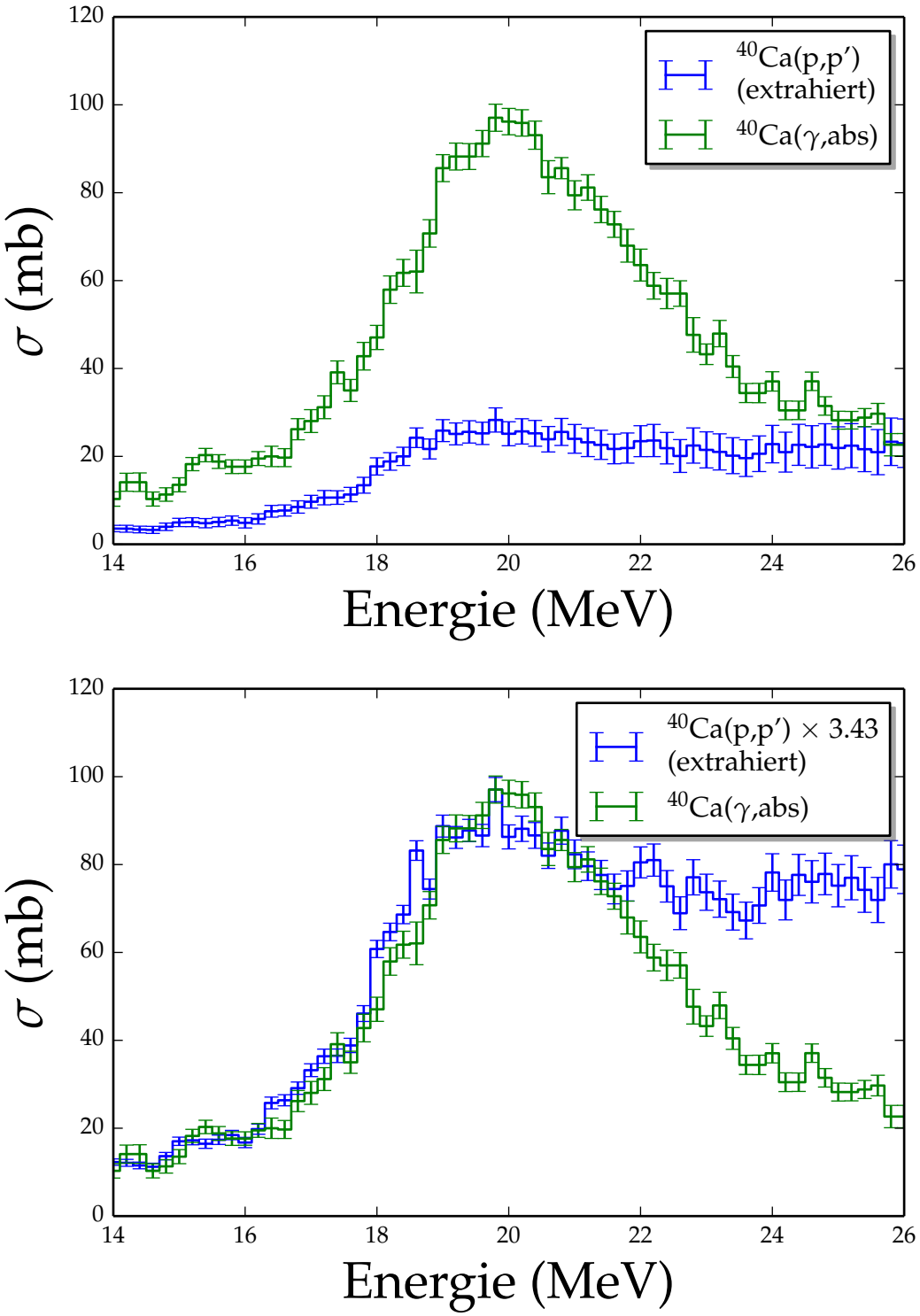


Abbildung 9.2: Der extrahierte sowie der experimentelle Photoabsorptionsquerschnitt für ^{40}Ca. Zum besseren Vergleich sind beide Datensätze in 200 keV-Bins aufgetragen. Im zweiten Bild wurde der extrahierte Photoabsorptionsquerschnitt mit einem Faktor von 3,43 multipliziert.

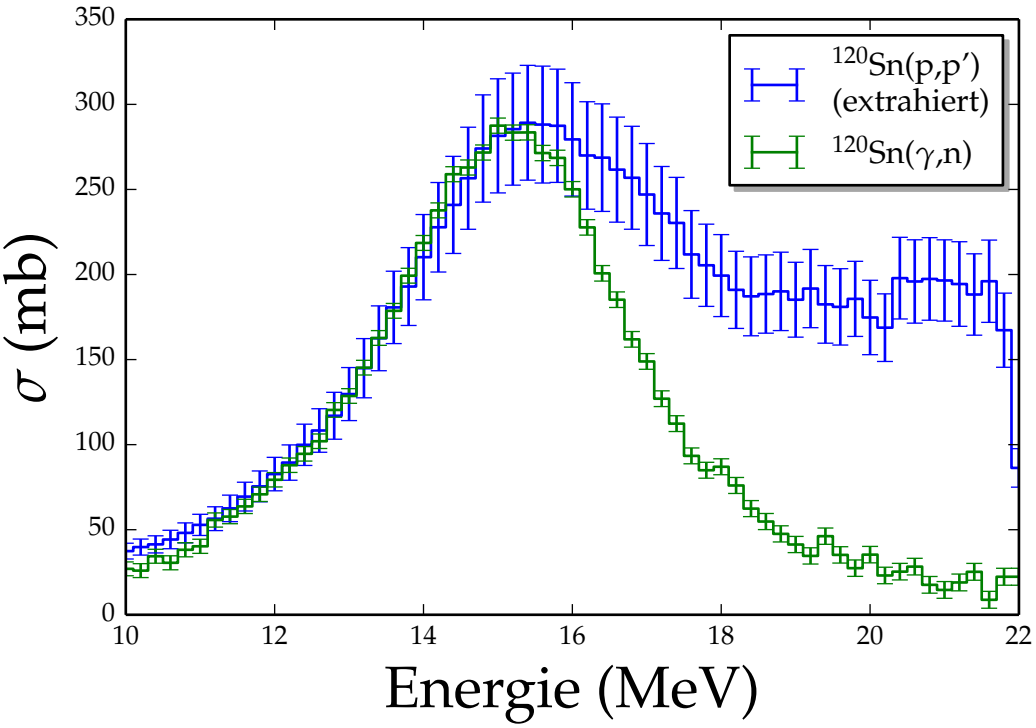


Abbildung 9.3: Der extrahierte sowie der experimentelle Photoabsorptionsquerschnitt für ^{120}Sn. Zum besseren Vergleich sind beide Datensätze in 200 keV-Bins aufgetragen.

wird der extrahierte Photoabsorptionsquerschnitt im Allgemeinen deutlich besser wiedergegeben. In Abb. 9.3 ist der extrahierte bzw. der experimentelle Photoabsorptionsquerschnitt für ^{120}Sn dargestellt. Es ist zu sehen, dass bei niedrigen Energien sowohl die Form, als auch die absolute Höhe des Spektrums gut wiedergegeben wird. Auch hier gibt es bei höheren Energien eine Abweichung, die auf die Anregungen höherer Multipole zurückgeht. Die Tatsache, dass schwere Kerne im Allgemeinen besser durch die virtuelle Photonenmethode beschrieben werden wird klar, wenn man sich den Sommerfeldparameter anschaut. Dieser ist durch $\eta = Z_1 Z_2 \alpha/\beta$ gegeben und ist ein Maß dafür, ob eine semiklassische Rechnung gerechtfertigt ist [67]. Ist der Sommerfeldparameter $\eta \gg 1$, so kann semiklassisch gerechnet werden, ansonsten müssen Quanteneffekte berücksichtigt werden. Für ^{40}Ca beträgt der Sommerfeldparameter $\eta = 0{,}22$, für ^{208}Pb liegt er bei $\eta = 0{,}92$. Man sieht also, dass streng genommen noch nicht mal für schwere Kerne eine semiklassische Rechnung angemessen ist. Dies ist vor allem darauf zurückzuführen, dass es sich beim Projektilteilchen um ein Proton handelt, dessen Ladung im Sommerfeldparameter kaum ins Gewicht fällt. Um die Form, aber auch die absolute Höhe des Spektrums, insbesondere bei leichten Kernen richtig zu beschreiben, muss also quantenmechanisch gerechnet werden.

Erste quantenmechanische Rechnungen wurden bereits für ^{40}Ca und ^{208}Pb durchgeführt. Hierfür wird die eikonale Näherung benutzt, wie in [68] beschrieben. Dabei werden relativistische Effekte sowie Retardierungseffekte berücksichtigt. Die differentielle Photonenzahl im Vergleich

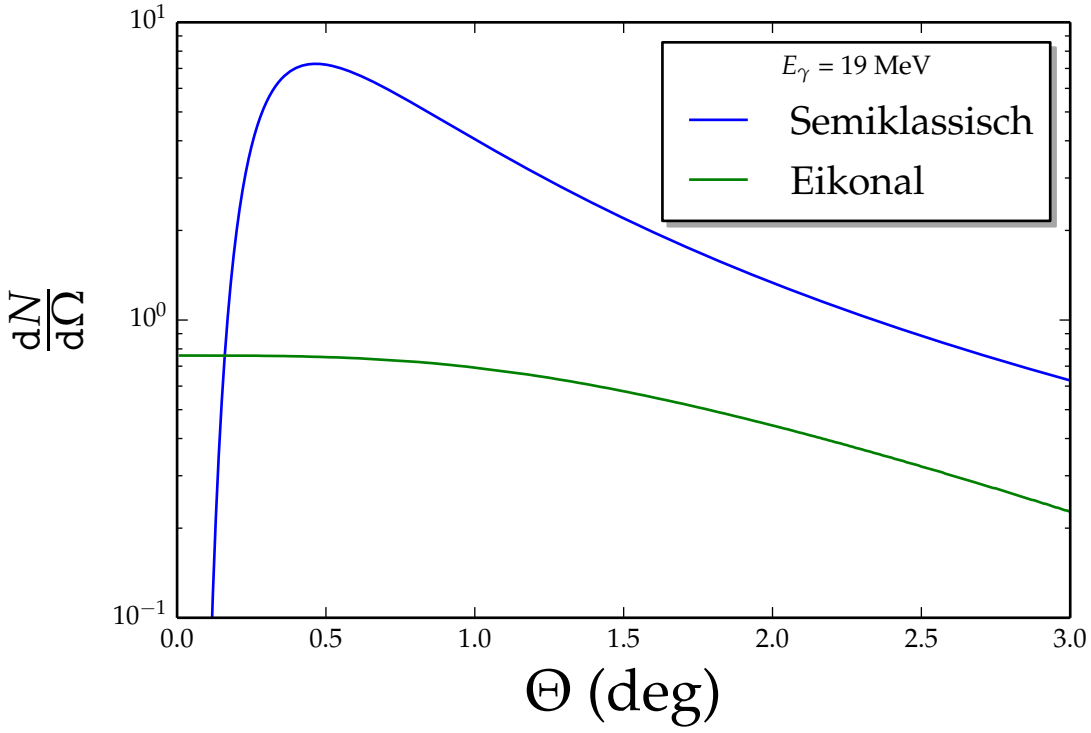


Abbildung 9.4: Vergleich der differentiellen Photonenzahl in semiklassischer bzw. eikonaler Näherung für ^{40}Ca.

zwischen eikonaler und semiklassischer Rechnung ist für die E1-Anregung in Abb. 9.4 dargestellt. Der Abbildung kann man entnehmen, dass bei der eikonalen Näherung die Singularität bei 0° verschwindet. Zudem ist die differentielle Photonenzahl über den ganzen Winkelbereich, insbesondere aber bei kleinen Winkeln, deutlich kleiner als bei der semiklassischen Rechnung. Der extrahierte Photoabsorptionsquerschnitt mit Hilfe der eikonalen Näherung ist in Abb. 9.5 zu sehen. Man sieht, dass in diesem Fall der extrahierte Photoabsorptionsquerschnitt überbestimmt wird, so dass der Normierungsfaktor nun kleiner als eins ist. Der Grund hierfür ist zur Zeit nicht bekannt. Eine mögliche Ursache könnte in der Wahl der Parameter für das optische Potential liegen, welches für die eikonale Rechnung benötigt wird. Die eikonale Rechnung für ^{208}Pb liefert jedoch die selben Ergebnisse, wie die semiklassische.

Die extrahierten Photoabsorptionsquerschnitte für alle anderen Kerne finden sich in Appendix B.2. In Abb. 9.6 und Tab. 9.2 sieht man die Normierungsfaktoren für alle bearbeiteten Kerne. Hier ist nochmals die große Diskrepanz zwischen dem extrahierten und dem experimentellen Photoabsorptionsquerschnitt bei leichten Kernen zu erkennen. Für schwere Kerne werden die Daten recht gut wiedergegeben. ^{154}Sm ist hierbei ein Sonderfall, da es sich hier um einen stark deformierten Kern handelt. Der Normierungsfaktor ist zudem davon abhängig, auf welchen der zwei Peaks in der Riesenresonanz des ^{154}Sm-Kerns normiert wird. Eine Verbesserung würde hier eine Multipolentfaltungsanalyse liefern. Besonders diskrepant ist das Ergebnis für ^{208}Pb. Hier ergibt sich ein viel zu hoher Photoabsorptionsquerschnitt. In [36] wurde für diesen

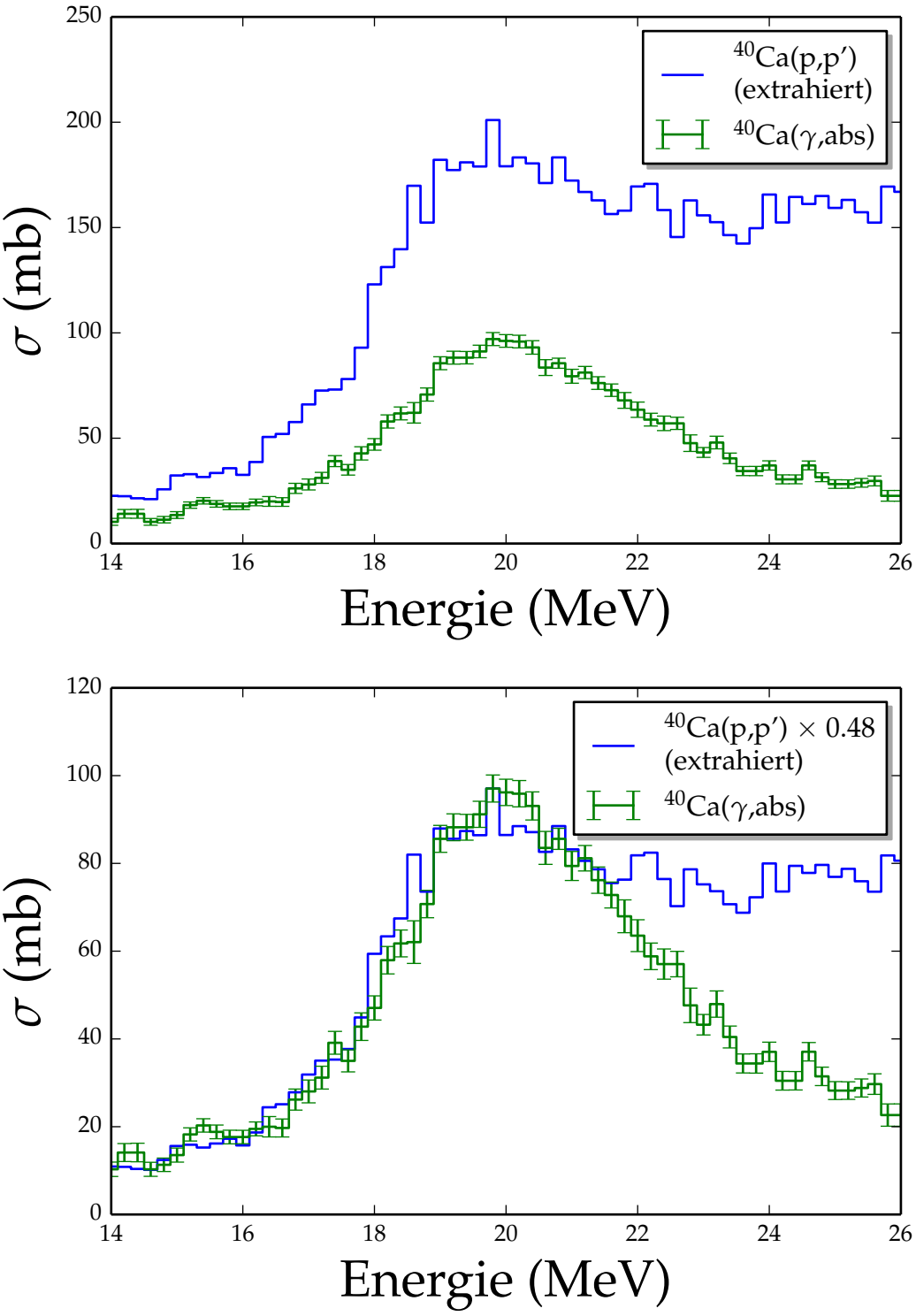


Abbildung 9.5: Der extrahierte sowie der experimentelle Photoabsorptionsquerschnitt für ^{40}Ca mit Hilfe der eikonalen Näherung. Zum besseren Vergleich sind beide Datensätze in 200 keV-Bins aufgetragen. Im zweiten Bild wurde der extrahierte Photoabsorptionsquerschnitt mit einem Faktor von 0,48 multipliziert.

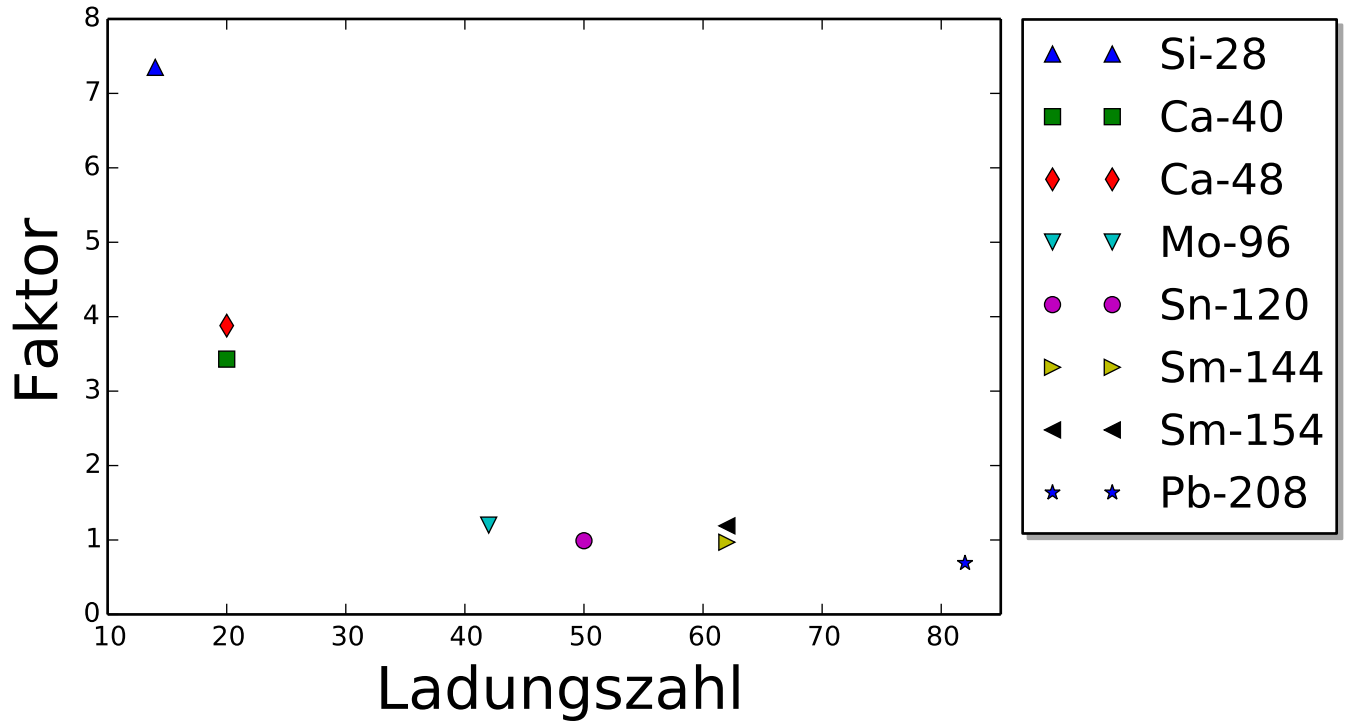


Abbildung 9.6: Normierungsfaktoren, die sich bei der Extraktion des Photoabsorptionsquerschnitts für verschiedene Kerne ergeben.

Kern der Photoabsorptionsquerschnitt bereits extrahiert, mit exzellenter Übereinstimmung mit experimentellen Daten. Die Ursache für diese Diskrepanz ist nicht bekannt.

Tabelle 9.2: Normierungsfaktoren, die sich bei der Extraktion des Photoabsorptionsquerschnitts für die einzelnen Kerne ergeben.

Targetkern	^{28}**Si**	^{40}**Ca**	^{48}**Ca**	^{96}**Mo**	^{120}**Sn**	^{144}**Sm**	^{154}**Sm**	^{208}**Pb**
Normierungsfaktor	7,35	3,43	3,88	1,20	0,99	0,97	1,19	0,69

9.2 Extraktion von Photoabsorptionsquerschnitten unter Verwendung der modellabhängigen Methode

Angesichts der Probleme bei der Extraktion der Photoabsorptionsquerschnitte, die im letzten Abschnitt beschrieben wurden, wurde eine alternative modellabhängige Methode für die Extraktion entwickelt. Hierzu wurde das Programm DWBA07 [33] verwendet, wobei nur die Coulombwechselwirkung berücksichtigt wurde. Als Eingabeparameter werden unter anderem B(E1)-Stärken benötigt. Diese wurden mit Hilfe des QPM berechnet. Da diese Methode deutlich aufwändiger ist als die virtuelle Photonenmethode, wurden die Photoabsorptionsquerschnitte

nur für die Kerne ^{40}Ca, ^{120}Sn und ^{208}Pb extrahiert. Die Idee dieser Methode zeigt sich in der folgenden Gleichung

$$\frac{\left(\frac{\mathrm{d}\sigma}{\mathrm{d}\Omega}\right)_{exp}}{B(E1)_{exp}} = \frac{\left(\frac{\mathrm{d}\sigma}{\mathrm{d}\Omega}\right)_{DWBA}}{B(E1)_{QPM}}. \tag{9.3}$$

Das Verhältnis aus dem theoretischen differentiellen Wirkungsquerschnitt und der theoretischen B(E1)-Stärke aus der QPM-Rechnung muss dem experimentellen Verhältnis entsprechen [69]. Umstellen der Gleichung liefert

$$B(E1)_{exp} = \left(\frac{\mathrm{d}\sigma}{\mathrm{d}\Omega}\right)_{exp} \frac{B(E1)_{QPM}}{\left(\frac{\mathrm{d}\sigma}{\mathrm{d}\Omega}\right)_{DWBA}}. \tag{9.4}$$

Da die Protonenstreudaten als doppeltdifferentielle Wirkungsquerschnitte vorliegen, ergibt sich für die letzte Gleichung

$$\left(\frac{\mathrm{d}B(E1)}{\mathrm{d}E}\right)_{exp} = \left(\frac{\mathrm{d}\sigma}{\mathrm{d}\Omega \mathrm{d}E}\right)_{exp} \frac{B(E1)_{QPM}}{\left(\frac{\mathrm{d}\sigma}{\mathrm{d}\Omega}\right)_{DWBA}}. \tag{9.5}$$

Die Photoabsorptionsquerschnitte können dann mit Hilfe von

$$\sigma_{\gamma}^{E1} = \frac{16\pi^3}{9} \left(\frac{\mathrm{d}B(E1)}{\mathrm{d}E}\right)_{exp} \frac{E_x}{\hbar c} \tag{9.6}$$

berechnet werden [70]. In Abb. 9.7 wird das Verhältnis aus den differentiellen Wirkungsquerschnitten aus DWBA und den B(E1)-Stärken aus QPM für einen Impulsübertrag von $q = 0$ MeV gezeigt. Wie man sieht reproduziert der *DWBA07*-Code recht gut die Proportionalität des differentiellen Wirkungsquerschnitts zu den B(E1)-Stärken bis etwa 3°. Man beachte auch den starken Abfall des differentiellen Wirkungsquerschnitts bei kleinen Winkeln in Übereinstimmung mit dem semiklassischen Modell. Allerdings kann man der Abb. 9.8 entnehmen, dass für einen endlichen Impulsübertrag die Proportionalität nicht so gut reproduziert wird. Das Problem lässt sich beheben, indem man dem Impulsübertrag einen festen Wert zuweist. Abb. 9.9 zeigt, dass die Proportionalität nun wiederhergestellt wird. Hierbei wurde im linken Bild $q = 10$ MeV gesetzt und im rechten Bild $q = 20$ MeV. In folgenden Rechnungen wurde der Impulsübertrag der Schwerpunktsenergie der Riesenresonanz gleichgesetzt und für die Berechnung der Proportionalität wurde der B(E1)-Wert bei dieser Energie benutzt. Wie in Abb. 9.10 dargestellt, wird auch hier die Proportionalität gut reproduziert. Die experimentellen differentiellen B(E1)-Werte können nun mit Glg. (9.5) aus den Protonenstreudaten berechnet werden. Nach dem Einsetzen dieser B(E1)-Werte in Glg. (9.6) erhält man schließlich die Photoabsorptionsquer-

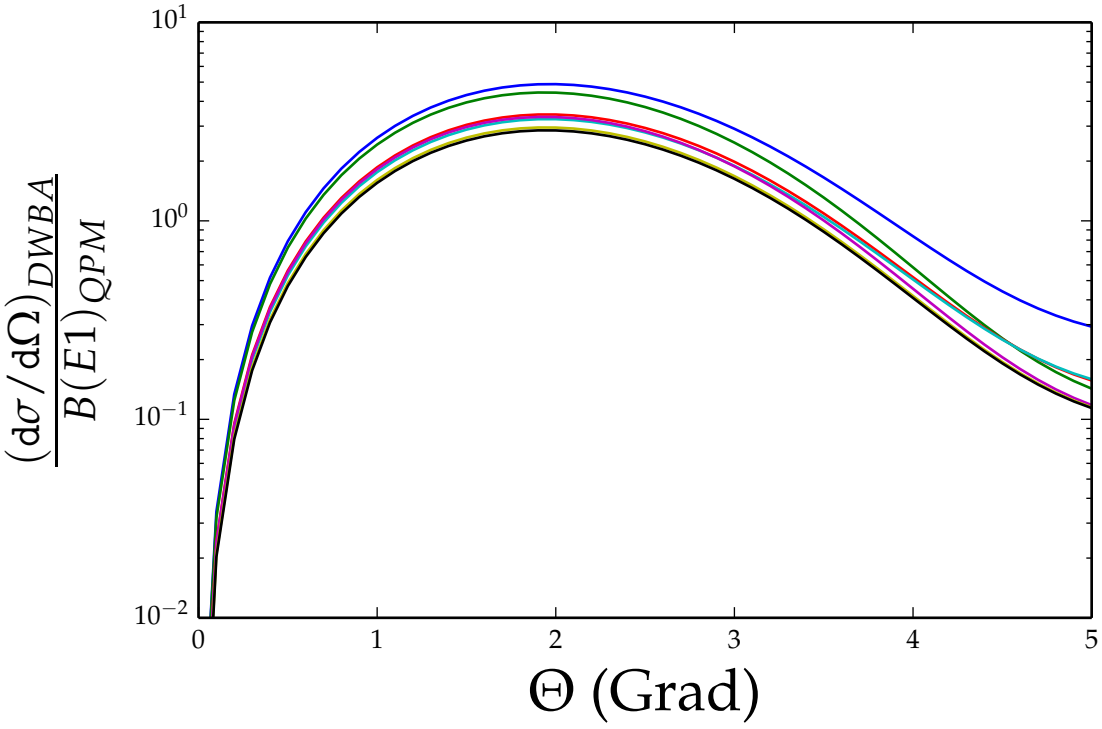


Abbildung 9.7: Das Verhältnis aus differentiellen Wirkungsquerschnitten und B(E1)-Stärken in Abhängigkeit vom Streuwinkel mit $q = 0$ MeV und für alle B(E1) $\geq 10^{-1}$ e^2 fm^2 für den Kern ^{40}Ca.

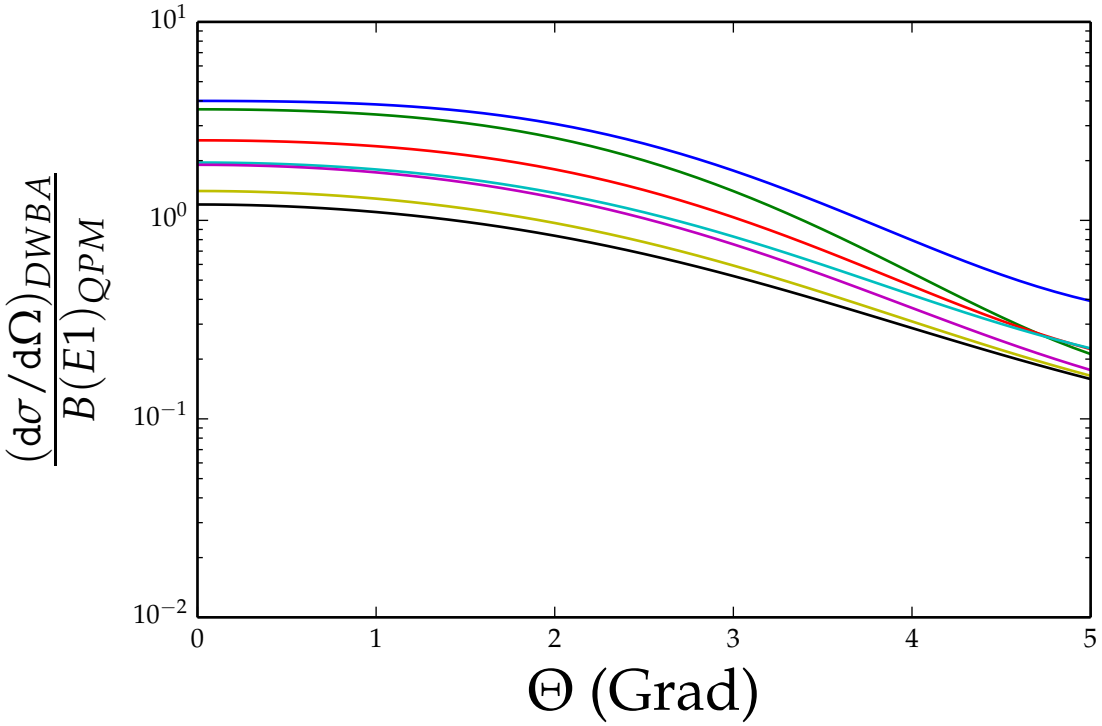


Abbildung 9.8: Das Verhältnis aus differentiellen Wirkungsquerschnitten und B(E1)-Stärken in Abhängigkeit vom Streuwinkel mit $q = E_x$ und für alle B(E1) $\geq 10^{-1}$ e^2 fm^2 für den Kern ^{40}Ca.

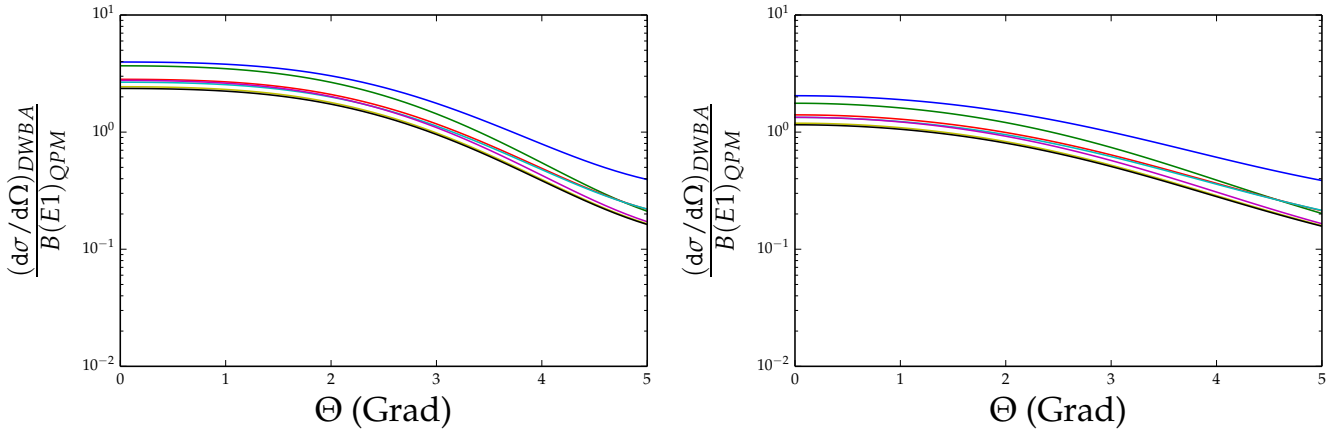


Abbildung 9.9: Das Verhältnis aus differentiellen Wirkungsquerschnitten und B(E1)-Stärken in Abhängigkeit vom Streuwinkel mit $q = 10$ MeV (links) und $q = 20$ MeV (rechts) für alle B(E1) $\geq 10^{-1}$ e^2 fm^2 für den Kern ^{40}Ca.

schnitte. In Abb. 9.11 sind die so extrahierten Photoabsorptionsquerschnitte für ^{40}Ca gezeigt. Die Photoabsorptionsquerschnitte für ^{120}Sn und ^{208}Pb finden sich in Appendix B.4. Man sieht, dass auch mit dieser Methode keine absolute Übereinstimmung zwischen den experimentellen und den extrahierten Photoabsorptionsquerschnitten erreicht wird. Im Gegensatz zu der virtuellen Photonenmethode werden die hier extrahierten Photoabsorptionsquerschnitte überschätzt. Die Form wird allerdings auch für hohe Anregungsenergien gut wiedergegeben. Auf

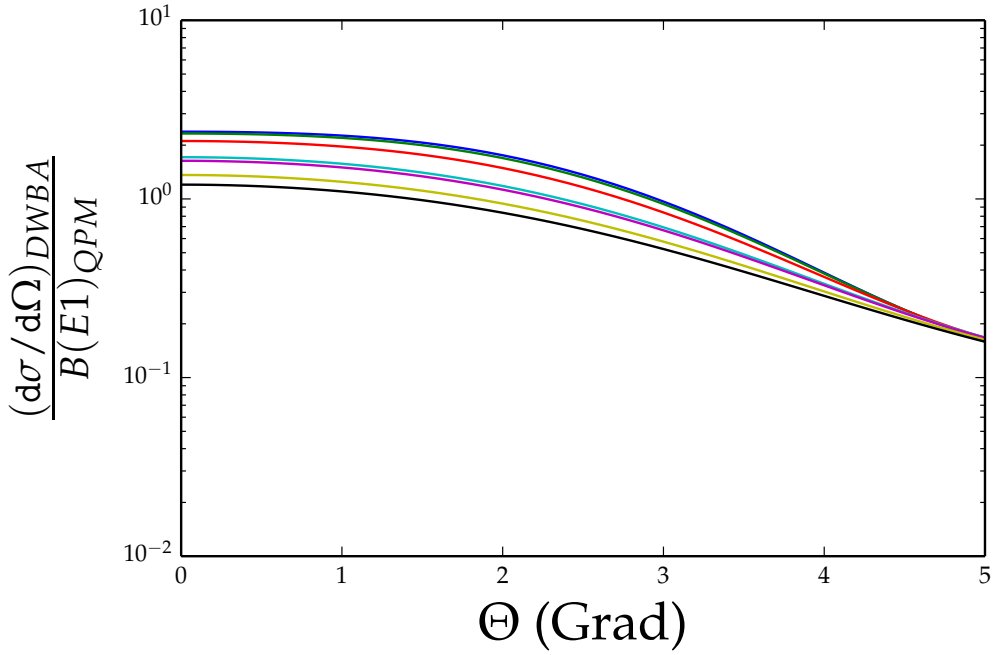


Abbildung 9.10: Das Verhältnis aus differentiellen Wirkungsquerschnitten und der B(E1)-Stärke. Der Impulsübertrag wurde der Schwerpunktsenergie gleichgesetzt, die für ^{40}Ca bei etwa 19 MeV liegt.

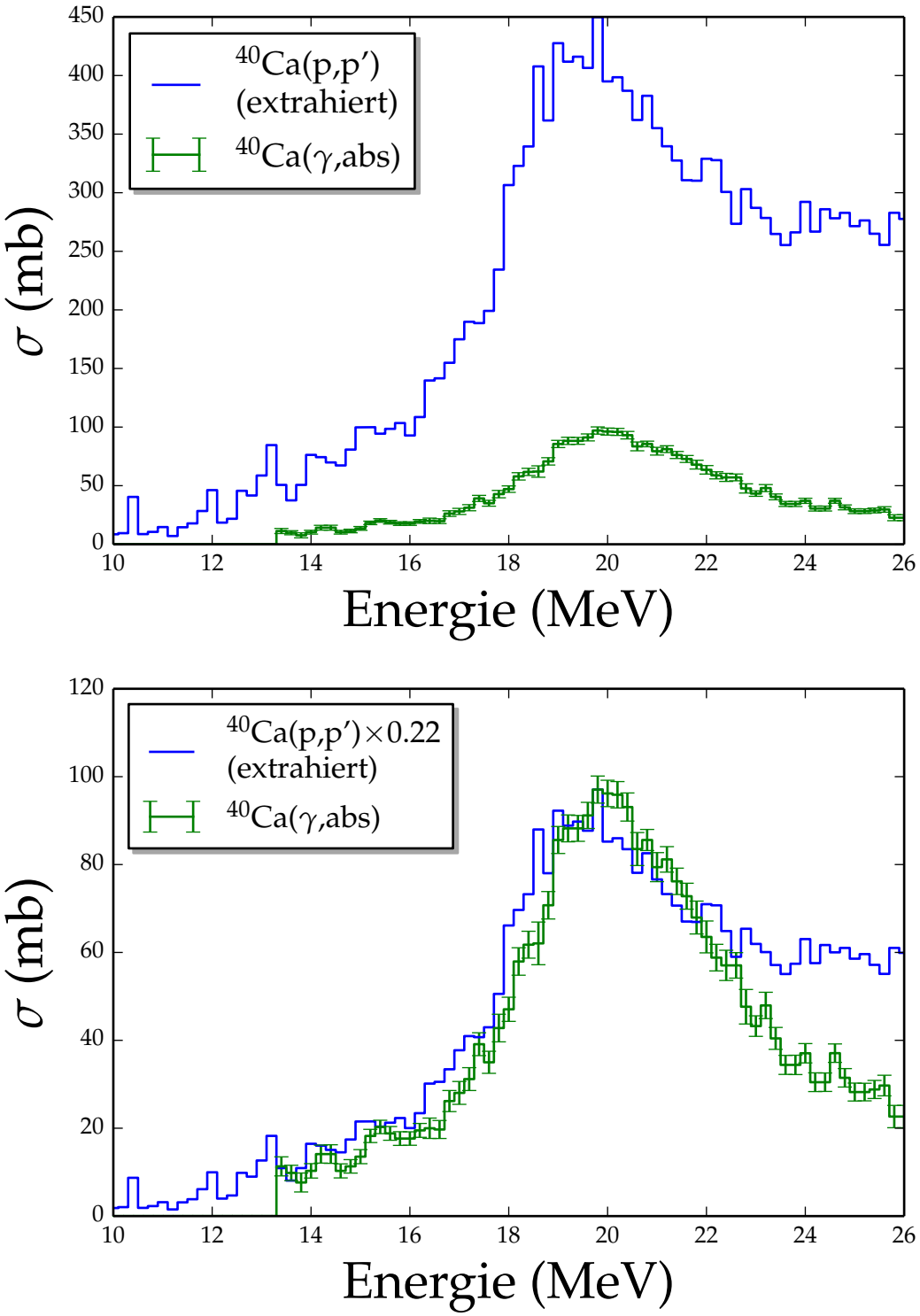


Abbildung 9.11: Vergleich zwischen den Photoabsorptionsquerschnitten, die nach der im Text beschriebenen Methode extrahiert wurden und den experimentellen Photoabsorptionsquerschnitten für ^{40}Ca. Im unteren Bild wurden die extrahierten Photoabsorptionsquerschnitte auf die experimentellen normiert.

Grund der hohen Komplexität des *DWBA07*-Codes kann nicht (kurzzeitig) überprüft werden, wie die Coulombstreuung dort behandelt wird. Daher kann die Ursache für die Überschätzung des Photoabsorptionsquerschnitts im Moment nicht festgestellt werden.

9.3 Vergleich und Diskussion der beiden Methoden

Wie man in den letzten beiden Abschnitten gesehen hat, kann man weder mit der virtuellen Photonenmethode noch mit der modellabhängigen Methode die Photoabsorptionsquerschnitte richtig wiedergeben. Zwar liefert die virtuelle Photonenmethode zufriedenstellende Ergebnisse für schwere Kerne, versagt jedoch bei leichten Kernen. Während bei der virtuellen Photonenmethode die Photoabsorptionsquerschnitte unterschätzt werden, überschätzt die modellabhängige Methode diese. Die Form wird bei beiden Methoden im niedrigen Energiebereich gut wiedergegeben. Die modellabhängige Methode liefert auch gute Ergebnisse im hohen Energiebereich. Wie man in Abschnitt 9.1 gesehen hat, liefert die semiklassische Beschreibung der virtuellen Photonenzahlen keine guten Ergebnisse bei hohen Energien und kleinen Winkeln, da dort eine Singularität auftaucht. Wie gezeigt, kann dieses Problem mit Hilfe der eikonalen Näherung gelöst werden. Allerdings wird auch bei der eikonalen Rechnung der absolute Photoabsorptionsquerschnitt bei leichten Kernen nicht richtig reproduziert. Der Grund hierfür könnte in der falschen Wahl der Parameter für das optische Potential liegen. Ansonsten lässt sich die virtuelle Photonenmethode gut für schwere Kerne bei niedrigen Anregungsenergien anwenden. Da die modellabhängige Methode sowohl für leichte, als auch für schwere Kerne eine gute Beschreibung der Form liefert, lässt sie sich, falls man nur an der Form des Spektrums interessiert ist, für eine Abschätzung des Verlaufs des Photoabsorptionsquerschnitts nutzen.

10 Zusammenfassung und Ausblick

Im zweiten Teil dieser Masterarbeit wurde vorgestellt, wie Photoabsorptionsquerschnitte aus doppeltdifferentiellen Wirkungsquerschnitten aus Protonenstreuung extrahiert werden können. Hierfür wurden zwei verschiedene Methoden benutzt, die virtuelle Photonenmethode und eine modellabhängige Methode unter Verwendung von DWBA- und QPM-Rechnungen. Die Photoabsorptionsquerschnitte für die Kerne ^{28}Si, ^{40}Ca, ^{48}Ca, ^{96}Mo, ^{120}Sn, ^{144}Sm, ^{154}Sm und ^{208}Pb wurden extrahiert, wobei mit Hilfe der modellabhängigen Methode nur die Kerne ^{40}Ca, ^{120}Sn und ^{208}Pb bearbeitet wurden. Es zeigte sich jedoch, dass die virtuelle Photonenmethode nur für schwere Kerne und vor allem im niedrigen Energiebereich gut anwendbar ist. Eine Erweiterung auf die leichten Kerne ist möglicherweise mit Hilfe der eikonalen Näherung möglich. Wie im letzten Kapitel gezeigt wurde, wird hiermit der Photoabsorptionsquerschnitt jedoch überschätzt. Der Grund hierfür ist im Moment Gegenstand der Untersuchung. Die modellabhängige Methode reproduziert recht gut die Form des Spektrums, überschätzt das Spektrum jedoch deutlich sowohl bei leichten als auch bei schweren Kernen. Sie stellt jedoch eine gute Möglichkeit dar um die Form des Photoabsorptionsquerschnitts abzuschätzen.

Man könnte auch den umgekehrten Weg gehen und aus Photoabsorptionsquerschnitten doppeltdifferentielle Wirkungsquerschnitte berechnen. Da mit den beiden Methoden nur der Coulombanteil in der Riesenresonanz berücksichtigt wird, hätte man somit eine Möglichkeit den nuklearen Untergrund, der bei der Protonenstreuung vor allem bei großen Streuwinkeln auftritt, zu bestimmen. Führt man diese Rechnungen für mehrere Kerne durch, könnte man dann versuchen eine Funktion zum Beschreiben des nuklearen Untergrunds in Abhängigkeit vom Targetkern, Streuwinkel und Anregungsenergie zu finden.

A Appendix: Teil I

A.1 Durchführung einer Messung mit Hilfe des MBS

Um eine Messung durchzuführen, loggt man sich zuerst mit einem Terminal auf dem labr-Server mit *ssh qclam@labr* ein. Das Passwort ist mit dem Benutzernamen identisch. Von da aus kann man sich nun per Telnet mit *telnet r3-1* direkt mit der RIO3-Einheit verbinden. Auch hier lautet der Benutzername bzw. das Passwort *qclam*. Nun betritt man das Verzeichnis *sergej* mit *cd sergej*. Je nach Wahl des Speicherorts für die Messdaten bei der letzten Messung kann es passieren, dass sich in diesem Verzeichnis eine Datei namens *data.lmd* befindet. Dies ist die Datei in die die Messdaten geschrieben werden. Diese sollte gesichert und dann gelöscht werden, da das MBS bei jeder Messung eine neue Datei mit diesem Namen öffnet und es zu Fehlern kommt, falls so eine Datei schon vorhanden ist. Nun kann das MBS mit dem Befehl *mbs* gestartet werden. Nach dem Start ändert sich das Anforderungszeichen zu *mbs>*. Nun führt man das Skript `startup.scom` mit *@startup* aus. Alle Vorbereitungen wurden hiermit abgeschlossen und die Messung kann mit *start acquisition* gestartet werden. Beendet wird die Messung, indem man das Skript `shutdown.scom` mit *@shutdown* ausführt. Möglich wäre auch *stop acquisition*, aber dann müssten noch eine Reihe anderer Befehle, unter anderem *close file*, manuell ausgeführt werden um die Messung ordnungsgemäß zu beenden. Die Messdaten sind nun in der Datei *data.lmd* abgespeichert. Hierbei enthält die Datei auch Informationen über die Adresse des ADCs/TDCs, Headerdaten und andere. Um daraus alleine die Messergebnisse zu extrahieren, kann man nun das Programm `listmode2plot` benutzen. Die Messergebnisse werden dann in eine Textdatei extrahiert.

A.2 Programmcode

A.2.1 f_user.c

In der Datei `f_user.c` sind die Adressen des ADCs bzw. des TDCs eingetragen. Die Module werden in dieser Datei initialisiert und die meisten Einstellungen, wie z.B Zero Suppression, Soft Reset, Overflow Suppression und andere werden hier vorgenommen. Zudem wurde die Verarbeitung der Trigger in dieser Datei programmiert.

```
1  /* N. Kurz, GSI, 12-Feb-1999 */
2  /*H. Lauinger, Sep-2012*/
3  /*S. Bassauer, Okt-2013*/
4
5  #include <smem.h>
6
7  #include "stdio.h"
8  #include "s_veshe.h"
9
10 #include "modules/CaenV785.h"
11 #include "modules/CaenV775.h"
12 #include "create_mem.h"
13
14 /*****************************************************************************/
15
16 /*
17  * all pointer which are used for read/write operations in the functions
18  * f_user_init and f_user_readout must be defined here as static variables
19  */
20
21 static CaenV785_module adc1;
22 static CaenV775_module tdc1;
23 int trigger_count = 0;
24
25 /*****************************************************************************/
26
27 int f_user_get_virt_ptr (long  *pl_loc_hwacc, long  pl_rem_cam[])
28 {
29     /*
30      * create virtual pointer to be used in f_user_init and f_user_readout
31      */
32     //Make memory segments for static memory mapping.
33     create_mem();
34     pl_loc_hwacc = (long*) (pl_a32);
35
36     adc1 = CaenV785_module_init(0x0000, pl_loc_hwacc);
37     tdc1 = CaenV775_module_init(0x0001, pl_loc_hwacc);
38 }
39
40 /*****************************************************************************/
41
42 int f_user_init (unsigned char   bh_crate_nr,
43                  long           *pl_loc_hwacc,
44                  long           *pl_rem_cam,
45                  long           *pl_stat)
46
47 {
48     int i = 0;
49     switch (bh_crate_nr)
50     {
51         case 0:
```

```
52 /********************** ADC Configuration ************************************/
53             printf("ADC Firmware = %d\n", adc1.reg->firmware_revision);
54             printf("ADC Address: %x\n", adc1);
55             printf("ADC Register Address: %x\n", adc1.reg);
56             printf("Initialise ADC.\n");
57             CaenV785_SoftReset(&adc1);
58             CaenV785_StandardSetup(&adc1);
59             CaenV785_DisplayStoredThresholds(&adc1);
60             CaenV785_ZeroSuppressionOn(&adc1);
61             CaenV785_OverflowSuppressionOff(&adc1);
62             CaenV785_PrintInfo(&adc1);
63
64 /********************** TDC Configuration ************************************/
65             printf("TDC Firmware = %d\n", tdc1.reg->firmware_revision);
66             printf("TDC Address: %x\n", tdc1);
67             printf("TDC Register Address: %x\n", tdc1.reg);
68             printf("Initialise TDC.\n");
69             CaenV775_SoftReset(&tdc1);
70             CaenV775_StandardSetup(&tdc1);
71             CaenV775_DisplayStoredThresholds(&tdc1);
72             //save space
73 //          for (i = 0; i < 32; i++) {
74 //              set Threshold to 54*2 = 100 (Step Threshold defaults 1)
75 //              CaenV775_SetThreshold(&tdc1, i, 85);
76 //          }
77             CaenV775_DisplayStoredThresholds(&tdc1);
78             CaenV775_ZeroSuppressionOn(&tdc1);
79             CaenV775_OverflowSuppressionOff(&tdc1);
80             break;
81         default:
82             break;
83     }
84     return 1;
85 }
86
87 /*****************************************************************************/
88
89 int f_user_readout (unsigned char   bh_trig_typ,
90                     unsigned char   bh_crate_nr,
91                     register long  *pl_loc_hwacc,
92                     register long  *pl_rem_cam,
93                     long           *pl_dat,
94                     s_veshe        *ps_veshe,
95                     long           *l_se_read_len,
96                     long           *l_read_stat)
97 {
98     *l_se_read_len = 0;
99     switch (bh_crate_nr)
100    {
101        case 0:
102
103            switch (bh_trig_typ)
104            {
105                case 14:
106                case 15:
107                case 1:
108                    //BLT mode (or one of the other modes) must be implemented
109                    //to operate two modules at the same time!
110                    *l_se_read_len += CaenV785_Readout(&adc1, pl_dat);
111                    *l_se_read_len += CaenV775_Readout(&tdc1, pl_dat);
112                    trigger_count++;
113                    if (trigger_count % 1000 == 0 || bh_trig_typ > 7) {
114                      printf("received %i trigger\n", trigger_count);
115                    }
116                    break;
117                case 2:
```

```
                case 12:
                case 13:
                    break;
                default:
                    break;
            }
            break;
        default:
            break;
    }

    return 1;
}

/*****************************************************************************/
```

A.2.2 create_mem.h

Diese Headerdatei ist für die Reservierung des Modulspeicherbereichs verantwortlich.

```
// create_mem.h used for MBS with RIO3
// creates memory segment for static memory mapping of VME modules
// 20060603 A. Shevchenko shevart@rocketmail.com

#define BASE_ADDR_A16  0x4e000000 /*  A16 short adress space on RIO3 */
#define BASE_ADDR_A32  0x50000000 /*  A32 short adress space on RIO3 */

unsigned long vme_base; // base adress of a segment
unsigned char *pl_a32;  // pointer to memory for A32 access
unsigned char *pl_a16;  // pointer to memory for A16 access

void create_mem()
{

  vme_base = BASE_ADDR_A32; // static mapping region for A32 access on RIO3
  pl_a32 = (unsigned char *) smem_create("vme_a32", (char *)vme_base,
         (unsigned int)0x10000000, SM_READ | SM_WRITE);
  if(pl_a32 == NULL)
  {
      printf("Could not create section for A32 access. base=%lx\n",vme_base);
      fflush(stdout);
  }

  vme_base = BASE_ADDR_A16; // static mapping region for A16 access on RIO3
  pl_a16 = (unsigned char *) smem_create("vme_a16", (char *)vme_base,
         (unsigned int)0x10000, SM_READ | SM_WRITE);
  if(pl_a16 == NULL)
  {
      printf("Could not create section for A16 access. base=%lx\n",vme_base);
      fflush(stdout);
  }

}
```

A.2.3 CaenV775.c

In der Datei `CaenV775.c` werden erweiterte Einstellungen des TDCs vorgenommen und die einzelnen Register angesteuert. Die Bedeutung der einzelnen Register kann hierbei dem Handbuch

des Moduls entnommen werden [71]. Die Datei enthält auch die Auslesefunktion sowie einige Funktionen zur Überprüfung der einzelnen Register.

```
#include "CaenV775.h"

#include <stdio.h>
#include <string.h>

int CaenV775_event_count = 0, CaenV775_busy_count = 0, CaenV775_empty_count = 0, CaenV775_invalid_count = 0;

// calculate module base address and return a module-structure
CaenV775_module CaenV775_module_init(uint16_t address, long *pl_loc_hwacc)
{
    long module_base_addr = (long)pl_loc_hwacc + address*CAEN_V775_OFFSET;
    struct CaenV775_module module =
    {
        (uint32_t *) ( module_base_addr ), // output_buffer
        (struct CaenV775_reg*) ( module_base_addr + CAEN_V775_REG_OFFSET )  // reg
    };

    return module;
}


// Read the Output Buffer of CAEN TDC V775
int CaenV775_Readout(CaenV775_module *module, long *mbs_data)
{
  long i, temp, loop_count = 0;
  long num_data_words, lw_type;
  long *mbs_data_start = mbs_data;

  //if(!(module->reg->status_reg_1 & 0x04)) // module not busy
  if (CaenV775_CheckBufferStatus(module) != 0) // meb not empty
  {

    if (CaenV775_CheckBusyStatus(module) == 1 && CaenV775_CheckBufferStatus(module) != 2) // module is busy
    {
//      *mbs_data++ = 0x05000000 + (module->reg->status_reg_1 & 0x01FF);  // module busy
      //printf("Busy!\n");
      CaenV775_busy_count++;
      if (CaenV775_busy_count % 500 == 0) {
    printf("counted %i trigger while busy (and meb not full)\n", CaenV775_busy_count);
      }
    }
    else
    {
      while (CaenV775_CheckDreadyStatus(module) == 1 && loop_count < 34)
      {
    loop_count++;

    // read the first longword
    temp = *module->output_buffer; //automatic pointer increment!

    // extract longword "type" (0=valid data, 2=header, 4=end of block, 6=invalid data)
    lw_type = (temp >> 24) & 0x7;

    if (lw_type == 2)
    {
      // extract number of data longwords
      num_data_words = (temp >> 8) & 0x3F;
      *mbs_data++ = temp;
//    printf("wrote header of %i data longwords\n", num_data_words);
      // read and pass on the data
      for(i=0 ; i < num_data_words; i++)
      {
```

```
63          *mbs_data++ = *module->output_buffer;
64          /*    printf("\tch: %d un: %d ov: %d -> val: %d\n",
65           *      (*(mbs_data-1) >> 16) & 0x1F,
66           *      (*(mbs_data-1) >> 13) & 0x1 ,
67           *      (*(mbs_data-1) >> 12) & 0x1 ,
68           *      (mbs_data-1) & 0xFFF
69           *      ); */
70          CaenV775_event_count++;
71          if (CaenV775_event_count % 10000 == 0) {
72            printf("counted %i valid subevents above threshold\n", CaenV775_event_count);
73          }
74        }
75        // read the EOB longword (GEO and "event counter" info)
76        *mbs_data++ = *module->output_buffer;
77        //    printf("wrote EOB\n");
78      }
79      else  // most likely no valid data (lwtype was 6)
80          {
81 //     printf("header expected: %X\n", temp);
82        CaenV775_invalid_count++;
83        if (CaenV775_invalid_count % 1000 == 0) {
84          printf("counted %i events without header\n", CaenV775_invalid_count);
85        }
86      //save space
87        *mbs_data++ = temp;
88          }
89        }  // while DReady
90      }
91      // clear memory to prevent memory overflow
92      //    CaenV775_ClearData(module);
93    } // meb not empty
94    else
95    {
96 //     *mbs_data++ = 0x07000000 + (module->reg->status_reg_2 & 0x00FF);  // meb empty
97      CaenV775_empty_count++;
98      if (CaenV775_empty_count % 500 == 0) {
99        printf("counted %i trigger while empty meb\n", CaenV775_empty_count);
100     }
101   }
102
103   // return number of bytes read
104   return (char*)mbs_data - (char*)mbs_data_start;
105 }
106
107 // module setup
108 void CaenV775_SoftReset(CaenV775_module *module)
109 {
110     module->reg->bit_set_1 = 0x80; //invoke software reset
111     module->reg->bit_clear_1 = 0x80; //reset to normal mode
112     printf("Software reset done!\n");
113 }
114
115 void CaenV775_StandardSetup(CaenV775_module *module)
116 {
117     CaenV775_ClearData(module);
118
119     //deactivate memory test
120     module->reg->bit_clear_2 = 0x1;
121
122     module->reg->event_counter_reset = 1;
123
124     //module->reg->bit_clear_2 = 0x307b; // 0011 0000 0111 1011
125     //module->reg->bit_set_2 = 0x4980;   // 0100 1001 1000 0000
126
127     CaenV775_SlidingScaleOn(module);
128     CaenV775_StepThresholdOn(module);
```

```
129     CaenV775_ZeroSuppressionOff(module);
130     CaenV775_OverflowSuppressionOn(module);
131     CaenV775_EnableAllChannels(module);
132     CaenV775_ResetThresholds(module);
133 
134     printf("Standard setup done!\n");
135 }
136 
137 void CaenV775_SlidingScaleOn(CaenV775_module *module) {
138   // enable sliding scale + write 1 in unused bit
139   module->reg->bit_set_2 = 0x0080;
140 }
141 
142 void CaenV775_SlidingScaleOff(CaenV775_module *module) {
143   // disable sliding scale + write 0 in unused bit
144   module->reg->bit_clear_2 = 0x0080;
145 }
146 
147 void CaenV775_StepThresholdOn(CaenV775_module *module)
148 {
149     module->reg->bit_set_2 = 0x100;
150 }
151 
152 void CaenV775_StepThresholdOff(CaenV775_module *module)
153 {
154     module->reg->bit_clear_2 = 0x100;
155 }
156 
157 
158 
159 void CaenV775_ZeroSuppressionOn(CaenV775_module *module)
160 {
161     module->reg->bit_clear_2 = 0x10;  /* clear Bit 4 */
162 }
163 
164 
165 void CaenV775_ZeroSuppressionOff(CaenV775_module *module)
166 {
167     module->reg->bit_set_2 = 0x10;    /* set Bit 4 */
168 }
169 
170 
171 void CaenV775_EnableAllChannels(CaenV775_module *module)
172 {
173     int i;
174 
175     for (i=0; i<32; i++)
176     {
177         module->reg->thresholds[i] = module->reg->thresholds[i] & 0xfeff;   /* clear kill bit(8) */
178     }
179 }
180 
181 
182 void CaenV775_DisableChannels(CaenV775_module *module, unsigned long mask)
183 {
184     // ----- disable chosen channels
185     //-----------------------------------------------
186 
187     int i;
188     unsigned long ch_bit=1;
189 
190     for (i=0; i<32; i++)
191     {
192         if (ch_bit & mask)
193         {
194             module->reg->thresholds[i] = module->reg->thresholds[i] | 0x0100;   /* set kill bit(8) */
```

```
195         }
196         ch_bit=2*ch_bit;
197     }
198 }
199
200
201 void CaenV775_SetThreshold(CaenV775_module *module, int n, int val)
202 {
203     // Set the threshold value [val] for the
204     // n_th channel
205     //   !!! be careful not to overwrite the "kill" bit (8)!
206     //--------------------------------------
207     short l_val;
208
209     if (n>= 0 && n < 32 && val <= 255)
210     {
211         l_val = ((module->reg->thresholds[n] >> 8) << 8) + val;
212         module->reg->thresholds[n] = l_val;
213     }
214 }
215
216 void CaenV775_ResetThresholds(CaenV775_module *module)
217 {
218     //  Reset/Initialize the V775 thresholds
219     //---------------------------------------------
220
221     int i;
222
223     for(i=0 ; i < 32 ; i++)
224     {
225         CaenV775_SetThreshold(module, i, 0) ;
226     }
227 }
228
229
230 void CaenV775_DisplayStoredThresholds(CaenV775_module *module)
231 {
232   // Display thresholds stored in V775
233   //-------------------------------------
234
235     int i;
236     char dummy[10];
237
238
239     printf("\n Channel/Threshold/Kill-bit -> \n") ;
240     for(i=0 ; i < 32; i++)
241     printf(" %d/%d/%d ",i,module->reg->thresholds[i] & 0xff,
242                         (module->reg->thresholds[i] & 0x100)>>8) ;
243
244     printf("\n");
245     fflush(stdout);
246
247
248   //F_ERROR(ERR__MSG_INFO,0,"Channel/Threshold/Kill-bit ->",l_pr_mask);
249   //sprintf(c_line," ");
250     for(i=0;i<32;i++)
251     {
252     //sprintf(dummy," %02d/%02d/%d ",i,module->reg->thresholds[i] & 0xff,
253     //                  (module->reg->thresholds[i] & 0x100)>>8) ;
254     //strcat(c_line,dummy);
255     //if (strlen(c_line) >= 55)
256         {
257       //F_ERROR(ERR__MSG_INFO,0,c_line,l_pr_mask);
258     //  sprintf(c_line," ");
259         }
260     }
```

```
261
262     //strcat(c_line,".");
263     //F_ERROR(ERR__MSG_INFO,0,c_line,l_pr_mask);
264
265     return ;
266 }
267
268
269 void CaenV775_SetFileThresholds(CaenV775_module *module)
270 {
271     // Read file "thresholds.V775"
272     // and Set the treshold value [val] for the
273     // n_th channel
274     //--------------------------------------
275
276     FILE *fp;
277     int  i,ival;
278     char txt[50];
279
280     // ----- Open the thresholds file -----
281
282     //sprintf(txt,"./caen/thresh_V775_%d.dat",card);
283     if ((fp=fopen(txt,"r")) != NULL)
284     {
285 //    printf("\nFound file %s -> Read/Set ........", txt) ;
286
287     //sprintf(c_line,"Found file %s -> Read/Set ...", txt);
288     //F_ERROR(ERR__MSG_INFO,0,c_line,l_pr_mask);
289
290     // ----- Get/Set the values -----
291     //while(fgets(txt, 50, fp) != NULL)
292     {
293         //if (strchr("!@#$%*",txt[0]) != NULL || strlen(txt) < 2) continue ;
294         //sscanf(txt, "%d %d ", &i, &ival) ;
295         //CaenV775_SetThreshold(card, i, ival) ;
296     }
297
298     //fclose(fp) ;
299 //    printf("....Ok! done\n") ;
300     }
301     else
302 //    printf("\n*** file %s NOT-Found --> No-Action ***\n", txt) ;
303     {
304         //sprintf(c_line,"file %s NOT found --> No Action!", txt);
305         //F_ERROR(ERR__MSG_INFO,0,c_line,l_pr_mask);
306     }
307 }
308
309
310 int CaenV775_CheckDreadyStatus(CaenV775_module *module)
311 {
312     //  this function checks Status Register 1
313     //  to see if the DREADY bit (#0) is set or not
314     int dready;
315
316     dready = module->reg->status_reg_1 & 0x1;
317     return dready ;
318 }
319
320
321 void CaenV775_ClearData(CaenV775_module *module)
322 {
323     //  this function clears data, read & write pointers,
324     //  event counter and QAC sections...
325     //  by toggling bit 2 of register 2
326
```

```
327     module->reg->bit_set_2 = 0x4;
328     module->reg->bit_clear_2 = 0x4;
329 }
330 
331 
332 int CaenV775_CheckBufferStatus(CaenV775_module *module)
333 {
334     //  this function checks Status Register 2
335     //  to see if the output buffer contains data
336     //  return val: 0: buffer empty, 2: buffer full, 1: not empty - not full
337 
338     int l_val,
339     buffstat;
340 
341     l_val = module->reg->status_reg_2 ;
342     if (l_val & 0x2)        /* buffer empty */
343         buffstat=0 ;
344     else
345     {
346         if (l_val & 0x4)      /* buffer full */
347             buffstat=2 ;
348         else
349             buffstat=1 ;        /* something between empty and full */
350     }
351 
352     return buffstat ;
353 }
354 
355 
356 int CaenV775_CheckBusyStatus(CaenV775_module *module)
357 {
358   // checks BUSY bit in status register 1 (#2) if module is busy (1) or not (0)
359   return ((module->reg->status_reg_1 >> 2) & 0x1);
360 }
361 
362 void CaenV775_OverflowSuppressionOn(CaenV775_module *module)
363 {
364     module->reg->bit_clear_2 = 0x8;  /* clear Bit 3 */
365 }
366 
367 
368 void CaenV775_OverflowSuppressionOff(CaenV775_module *module)
369 {
370     module->reg->bit_set_2 = 0x8;    /* set Bit 3 */
371 }
372 
373 int CaenV775_Firmware(CaenV775_module *module)
374 {
375     return module->reg->firmware_revision;
376 }
377 
378 void CaenV775_PrintInfo(CaenV775_module *module)
379 {
380    // display information on the status of the module
381    int i;
382    printf("\n*********************************** Info Caen V775 TDC ************************************ \n") ;
383    for( i=0 ; i < 32; i++){
384       if ( ((module->reg->thresholds[i] & 0x100)>>8) == 0) printf("Channel %d    Threshold  %d  \n",i,module->reg->
              thresholds[i] & 0xff) ;
385    }
386    if ( (module->reg->bit_set_2 & 0x100)==0 ) printf("Threshold resolution is x16 \n") ;
387    if ( (module->reg->bit_set_2 & 0x100)==0x100 ) printf("Threshold resolution is x2 \n") ;
388    if ( (module->reg->bit_set_2 & 0x10)==0 ) printf("Zero suppression is on \n") ;
389    if ( (module->reg->bit_set_2 & 0x10)==0x10 ) printf("Zero suppression is off \n") ;
390    if ( (module->reg->bit_set_2 & 0x800)==0 ) printf("Automatic increment of the readout pointer is off  \n") ;
391    if ( (module->reg->bit_set_2 & 0x800)==0x800 ) printf("Automatic increment of the readout pointer is on  \n") ;
```

```
392     if ( (module->reg->bit_set_2 & 0x8)==0 ) printf("Overflow suppression is on \n") ;
393     if ( (module->reg->bit_set_2 & 0x8)==0x8 ) printf("Overflow suppression is off \n");
394     if ( (module->reg->bit_set_2 & 0x400)==0 ) printf("Common start mode \n");
395     if ( (module->reg->bit_set_2 & 0x400)==0x400 ) printf("Common stop mode \n");
396     if ( (module->reg->bit_set_2 & 0x80)==0 ) printf("Sliding scale is off  \n") ;
397     if ( (module->reg->bit_set_2 & 0x80)==0x80 ) printf("Sliding scale is on  \n") ;
398     if ( &(module->reg->thresholds[31])== (volatile int16_t*) 0xf00110be ) printf("Address Map is ok \n") ;
399        else printf("Error: Address Map is wrong \n");
400     printf("********************************************************************************************** \n") ;
401     printf("\n");
402     fflush(stdout);
403     return ;
404 }
```

A.2.4 CaenV775.h

In der zugehörigen Headerdatei `CaenV775.h` werden die Registerkarten erstellt, sowie einige Funktionen zum Einstellen der Module definiert.

```
1  #ifndef CAEN_V775_H
2  #define CAEN_V775_H
3
4  #include <inttypes.h>
5
6  /* caen VME ADC V775 */
7
8  #define CAEN_OFFSET 0x10000
9  #define CAEN_V775_OFFSET 0x10000
10 #define CAEN_V775_REG_OFFSET 0x1000
11
12 #define V767__HEADER_TYPE_SHIFT 20
13 #define V767__HEADER_TYPE_MASK 0x6
14 #define CAEN_V775__OUTPUT_BUFFER_GEO_N_WORD_COUNT   8
15 #define CAEN_V775__OUTPUT_BUFFER_WORD_COUNT         0x3F
16 #define CAEN_V775__OUTPUT_BUFFER_GEO_N_WORD_COUNT   8
17 #define CAEN_V775__OUTPUT_BUFFER_WORD_COUNT         0x3F
18
19 /* Address map of module. See module manual for details. */
20 struct CaenV775_reg
21 {
22   volatile int16_t firmware_revision;
23   volatile int16_t geo_addr;
24   volatile int16_t mcst_addr;
25
26   volatile int16_t bit_set_1;
27   volatile int16_t bit_clear_1;
28
29   volatile int16_t interrupt_level;
30   volatile int16_t interrupt_vector;
31
32   volatile int16_t status_reg_1;
33   volatile int16_t control_reg_1;
34
35   volatile int16_t ader_h;
36   volatile int16_t ader_l;
37   volatile int16_t single_shot_reset;
38
39   volatile int16_t dummy0;
40
41   volatile int16_t mcst_ctrl;
42
43   volatile int16_t dummy1[2];
```

```
44
45    volatile int16_t event_trig_reg;
46    volatile int16_t status_reg_2;
47
48    volatile int16_t event_counter_l;
49    volatile int16_t event_counter_h;
50    volatile int16_t inc_event;
51    volatile int16_t inc_offset;
52
53    volatile int16_t load_test_reg;
54    volatile int16_t fclr_window;
55
56    volatile int16_t dummy2;
57
58    volatile int16_t bit_set_2;
59    volatile int16_t bit_clear_2;
60
61    volatile int16_t w_mem_test_addr;
62    volatile int16_t mem_test_wh;
63    volatile int16_t mem_test_wl;
64
65    volatile int16_t crate_select;
66    volatile int16_t test_event_write;
67    volatile int16_t event_counter_reset;
68
69    volatile int16_t dummy3[15];
70
71    volatile int16_t full_scale_range;   /* TDC only */
72
73    volatile int16_t dummy4;
74
75    volatile int16_t r_test_addr;
76
77    volatile int16_t dummy5;
78
79    volatile int16_t sw_comm;
80    volatile int16_t slide_const;
81
82    volatile int16_t dummy6[2];
83
84    volatile int16_t aad;
85    volatile int16_t bad;
86
87    volatile int16_t dummy7[6];
88
89    volatile int16_t thresholds[32];
90 };
91
92
93 struct CaenV775_module
94 {
95     volatile uint32_t *output_buffer;
96     volatile struct CaenV775_reg *reg;
97 };
98
99 typedef struct CaenV775_module CaenV775_module;
100
101
102 /**
103     Initialize module pointers.
104     'address' is the number represented by the 4 rotary
105     switches on the piggy-back boards.
106 */
107 CaenV775_module CaenV775_module_init(uint16_t address, long *pl_loc_hwacc);
108
109
```

```
110 /** @brief readout the output buffer of one card and copy it to the mbs system
111     returns number of read words
112 **/
113 int CaenV775_Readout(CaenV775_module *module, long *mbs_data);
114
115 // this is the interface to the module
116 void CaenV775_SoftReset (CaenV775_module *module);
117 void CaenV775_StandardSetup(CaenV775_module *module);
118
119 void CaenV775_SlidingScaleOn(CaenV775_module *module);
120 void CaenV775_SlidingScaleOff(CaenV775_module *module);
121
122 void CaenV775_StepThresholdOn(CaenV775_module *module);
123 void CaenV775_StepThresholdOff(CaenV775_module *module);
124
125 void CaenV775_ZeroSuppressionOn(CaenV775_module *module);
126 void CaenV775_ZeroSuppressionOff(CaenV775_module *module);
127
128 void CaenV775_EnableAllChannels(CaenV775_module *module);
129 void CaenV775_DisableChannels(CaenV775_module *module, unsigned long mask);
130
131 void CaenV775_SetThreshold(CaenV775_module *module, int n, int val);
132 void CaenV775_ResetThresholds(CaenV775_module *module);
133 void CaenV775_DisplayStoredThresholds(CaenV775_module *module);
134
135 void CaenV775_SetFileThresholds(CaenV775_module *module);
136
137 int CaenV775_CheckDreadyStatus(CaenV775_module *module);
138 void CaenV775_ClearData(CaenV775_module *module);
139 int CaenV775_CheckBufferStatus(CaenV775_module *module);
140 int CaenV775_CheckBusyStatus(CaenV775_module *module);
141
142 void CaenV775_OverflowSuppressionOn(CaenV775_module *module);
143 void CaenV775_OverflowSuppressionOff(CaenV775_module *module);
144
145 int CaenV775_Firmware(CaenV775_module *module);
146
147 void CaenV775_PrintInfo(CaenV775_module *module);
148
149
150 #endif
```

A.2.5 CaenV785.c

Analog zum TDC werden in dieser Datei erweiterte Einstellungen des ADCs vorgenommen und die einzelnen Register angesteuert. Auch hier kann die Bedeutung der Register im Handbuch eingesehen werden [72].

```
1  #include "CaenV785.h"
2
3  #include <stdio.h>
4  #include <string.h>
5
6  //#include "adc_785_debug_tools.h"
7
8  int CaenV785_event_count = 0, CaenV785_busy_count = 0, CaenV785_empty_count = 0, CaenV785_invalid_count = 0;
9
10
11 // calculate module base address and return a module-structure
12 CaenV785_module CaenV785_module_init(uint16_t address, long *pl_loc_hwacc)
13 {
```

```
14    long module_base_addr = (long)pl_loc_hwacc + address*CAEN_V785_OFFSET;
15
16    struct CaenV785_module module =
17    {
18      (uint32_t *) ( module_base_addr ), // output_buffer
19      (struct CaenV785_reg*) ( module_base_addr + CAEN_V785_REG_OFFSET )  // reg
20    };
21
22    return module;
23  }
24
25
26  // Read the Output Buffer of CAEN ADC V785
27  int CaenV785_Readout(CaenV785_module *module, long *mbs_data)
28  {
29    long i, temp, loop_count = 0;
30    long num_data_words, lw_type;
31    long *mbs_data_start = mbs_data;
32
33
34    //if(!(module->reg->status_reg_1 & 0x04)) // module not busy
35    if (CaenV785_CheckBufferStatus(module) != 0) // meb not empty
36    {
37
38      if (CaenV785_CheckBusyStatus(module) == 1 && CaenV785_CheckBufferStatus(module) != 2) // module is busy
39      {
40  //      *mbs_data++ = 0x05000000 + (module->reg->status_reg_1 & 0x01FF);  // module busy
41        CaenV785_busy_count++;
42        if (CaenV785_busy_count % 500 == 0) {
43      printf("counted %i trigger while busy (and meb not full)\n", CaenV785_busy_count);
44        }
45      }
46      else
47      {
48        while (CaenV785_CheckDreadyStatus(module) == 1 && loop_count < 34)
49        {
50      loop_count++;
51
52      // read the first longword
53      temp = *module->output_buffer; // for some reason, the output_buffer pointer must not be incremented !!
54
55      // extract longword "type" (2=header,6=invalid data)
56      lw_type = (temp >> 24) & 0x7;
57
58      if (lw_type == 2)
59      {
60        // extract number of data longwords
61        num_data_words = (temp >> 8) & 0x3F;
62        *mbs_data++ = temp;
63  //    printf("wrote header of %i data longwords\n", num_data_words);
64        // read and pass on the data
65        for(i=0 ; i < num_data_words; i++)
66        {
67          *mbs_data++ = *module->output_buffer;
68          /*    printf("\tch: %d un: %d ov: %d -> val: %d\n",
69           *      (*(mbs_data-1) >> 16) & 0x1F,
70           *      (*(mbs_data-1) >> 13) & 0x1 ,
71           *      (*(mbs_data-1) >> 12) & 0x1 ,
72           *      (mbs_data-1) & 0xFFF
73           *      ); */
74          CaenV785_event_count++;
75          if (CaenV785_event_count % 10000 == 0) {
76            printf("counted %i valid subevents above threshold\n", CaenV785_event_count);
77          }
78        }
79        // read the EOB longword (GEO and "event counter" info)
```

```
80        *mbs_data++ = *module->output_buffer;
81        //    printf("wrote EOB\n");
82      }
83      else  // most likely no valid data (lwtype was 6)
84          {
85  //    printf("header expected: %X\n", temp);
86        CaenV785_invalid_count++;
87        if (CaenV785_invalid_count % 1000 == 0) {
88          printf("counted %i events without header\n", CaenV785_invalid_count);
89        }
90      //save space
91        *mbs_data++ = temp;
92          }
93        }  // while DReady
94      }
95      // clear memory to prevent memory overflow
96      //    CaenV785_ClearData(module);
97    } // meb not empty
98    else
99    {
100 //    *mbs_data++ = 0x07000000 + (module->reg->status_reg_2 & 0x00FF);  // meb empty
101     CaenV785_empty_count++;
102     if (CaenV785_empty_count % 500 == 0) {
103       printf("counted %i trigger while empty meb\n", CaenV785_empty_count);
104     }
105   }
106
107   // return number of bytes read
108   return (char*)mbs_data - (char*)mbs_data_start;
109 }
110
111
112 // module setup
113 void CaenV785_SoftReset (CaenV785_module *module)
114 {
115   module->reg->bit_set_1 = 0x80;
116   module->reg->bit_clear_1 = 0x80;
117   printf("software reset done\n");
118 }
119
120 void CaenV785_StandardSetup(CaenV785_module *module)
121 {
122   CaenV785_ClearData(module);
123
124   // deactivate mem test
125   module->reg->bit_clear_2 = 0x1;
126
127   module->reg->event_counter_reset = 1;
128
129   //    module->reg->bit_clear_2 = 0x307b; // 0011 0000 0111 1011
130   //    module->reg->bit_set_2 = 0x4980;   // 0100 1001 1000 0000
131
132   CaenV785_SlidingScaleOn(module);
133   CaenV785_StepThresholdOn(module);
134   CaenV785_ZeroSuppressionOff(module);
135   CaenV785_OverflowSuppressionOn(module);
136   CaenV785_EnableAllChannels(module);
137   CaenV785_ResetThresholds(module);
138
139
140   printf("standard setup done\n");
141 }
142
143 void CaenV785_SlidingScaleOn(CaenV785_module *module) {
144   // enable sliding scale + write 1 in unused bit
145   module->reg->bit_set_2 = 0x00a0;
```

```
146 }
147
148 void CaenV785_SlidingScaleOff(CaenV785_module *module) {
149   // disable sliding scale + write 0 in unused bit
150   module->reg->bit_clear_2 = 0x00a0;
151 }
152
153 void CaenV785_StepThresholdOn(CaenV785_module *module)
154 {
155   // low threshold values
156   // threschold value = set threshold * 2
157   module->reg->bit_set_2 = 0x100;
158 }
159
160 void CaenV785_StepThresholdOff(CaenV785_module *module)
161 {
162   // low threshold values
163   // threschold value = set threshold * 16
164   module->reg->bit_clear_2 = 0x100;
165 }
166
167
168
169 void CaenV785_ZeroSuppressionOn(CaenV785_module *module)
170 {
171   module->reg->bit_clear_2 = 0x10;  /* clear Bit 4 */
172 }
173
174
175 void CaenV785_ZeroSuppressionOff(CaenV785_module *module)
176 {
177   module->reg->bit_set_2 = 0x10;    /* set Bit 4 */
178 }
179
180
181 void CaenV785_EnableAllChannels(CaenV785_module *module)
182 {
183   int i;
184
185   for (i=0; i<32; i++)
186   {
187     module->reg->thresholds[i] = module->reg->thresholds[i] & 0xfeff;   /* clear kill bit(8) */
188   }
189 }
190
191
192 void CaenV785_DisableChannels(CaenV785_module *module, unsigned long mask)
193 {
194   // ----- disable chosen channels
195   //-------------------------------------------
196
197   int i;
198   unsigned long ch_bit=1;
199
200   for (i=0; i<32; i++)
201   {
202     if (ch_bit & mask)
203     {
204       module->reg->thresholds[i] = module->reg->thresholds[i] | 0x0100;   /* set kill bit(8) */
205     }
206     ch_bit=2*ch_bit;
207   }
208 }
209
210
211 void CaenV785_SetKillThreshold(CaenV785_module *module, int channel) {
```

```
212   // don't store values under the channel's threshold in MEB
213   module->reg->thresholds[channel] = (module->reg->thresholds[channel] & 0xfeff) + 0x0100;
214 }
215
216
217 void CaenV785_SetSaveThreshold(CaenV785_module *module, int channel) {
218   // store values under the channel's threshold in MEB
219   module->reg->thresholds[channel] = module->reg->thresholds[channel] & 0xfeff;
220 }
221
222
223 void CaenV785_SetThreshold(CaenV785_module *module, int n, int val)
224 {
225   // Set the threshold value [val] for the
226   // n_th channel
227   //   !!! be careful not to overwrite the "kill" bit (8)!
228   //--------------------------------------
229   short l_val;
230
231   if (n>= 0 && n < 32 && val <= 255)
232   {
233     l_val = ((module->reg->thresholds[n] >> 8) << 8) + val;
234     module->reg->thresholds[n] = l_val;
235   }
236 }
237
238 void CaenV785_ResetThresholds(CaenV785_module *module)
239 {
240   //  Reset/Initialize the V785 thresholds
241   //---------------------------------------------
242
243   int i;
244
245   for(i=0 ; i < 32 ; i++)
246   {
247     CaenV785_SetThreshold(module, i, 0) ;
248   }
249 }
250
251
252 void CaenV785_DisplayStoredThresholds(CaenV785_module *module)
253 {
254   // Display thresholds stored in V785
255   //--------------------------------------
256
257   int i;
258   char dummy[10];
259
260
261   printf("\n Channel/Threshold/Kill-bit -> \n") ;
262   for(i=0 ; i < 32; i++)
263     printf(" %d/%d/%d ",i,module->reg->thresholds[i] & 0xff,
264        (module->reg->thresholds[i] & 0x100)>>8) ;
265
266        printf("\n");
267        fflush(stdout);
268
269
270        //F_ERROR(ERR__MSG_INFO,0,"Channel/Threshold/Kill-bit ->",l_pr_mask);
271        //sprintf(c_line," ");
272        for(i=0;i<32;i++)
273        {
274          //sprintf(dummy," %02d/%02d/%d ",i,module->reg->thresholds[i] & 0xff,
275          //                 (module->reg->thresholds[i] & 0x100)>>8) ;
276          //strcat(c_line,dummy);
277          //if (strlen(c_line) >= 55)
```

```
278           {
279             //F_ERROR(ERR__MSG_INFO,0,c_line,l_pr_mask);
280             //  sprintf(c_line," ");
281           }
282         }
283 
284         //strcat(c_line,".");
285         //F_ERROR(ERR__MSG_INFO,0,c_line,l_pr_mask);
286 
287         return ;
288 }
289 
290 
291 void CaenV785_SetFileThresholds(CaenV785_module *module)
292 {
293   // Read file "thresholds.v785"
294   // and Set the treshold value [val] for the
295   // n_th channel
296   //--------------------------------------
297 
298   FILE *fp;
299   int  i,ival;
300   char txt[50];
301 
302   // ----- Open the thresholds file -----
303 
304   //sprintf(txt,"./caen/thresh_v785_%d.dat",card);
305   if ((fp=fopen(txt,"r")) != NULL)
306   {
307     //    printf("\nFound file %s -> Read/Set ........", txt) ;
308 
309     //sprintf(c_line,"Found file %s -> Read/Set ...", txt);
310     //F_ERROR(ERR__MSG_INFO,0,c_line,l_pr_mask);
311 
312     // ----- Get/Set the values -----
313     //while(fgets(txt, 50, fp) != NULL)
314     {
315       //if (strchr("!@#$%*",txt[0]) != NULL || strlen(txt) < 2) continue ;
316       //sscanf(txt, "%d %d ", &i, &ival) ;
317       //CaenV785_SetThreshold(card, i, ival) ;
318     }
319 
320     //fclose(fp) ;
321     //    printf("....Ok! done\n") ;
322   }
323   else
324     //    printf("\n*** file %s NOT-Found --> No-Action ***\n", txt) ;
325   {
326     //sprintf(c_line,"file %s NOT found --> No Action!", txt);
327     //F_ERROR(ERR__MSG_INFO,0,c_line,l_pr_mask);
328   }
329 }
330 
331 
332 int CaenV785_CheckDreadyStatus(CaenV785_module *module)
333 {
334   //  this function checks Status Register 1
335   //  to see if the DREADY bit (#0) is set or not
336   int dready;
337 
338   dready = module->reg->status_reg_1 & 0x1;
339   return dready ;
340 }
341 
342 
343 void CaenV785_ClearData(CaenV785_module *module)
```

```
344 {
345   //  this function clears data, read & write pointers,
346   //  event counter and QAC sections...
347   //  by toggling bit 2 of register 2
348
349   module->reg->bit_set_2 = 0x4;
350   module->reg->bit_clear_2 = 0x4;
351 }
352
353
354 int CaenV785_CheckBufferStatus(CaenV785_module *module)
355 {
356   //  this function checks Status Register 2
357   //  to see if the output buffer contains data
358   //  return val: 0: buffer empty, 2: buffer full, 1: not empty - not full
359
360   int l_val,
361   buffstat;
362
363   l_val = module->reg->status_reg_2 ;
364   if (l_val & 0x2)        /* buffer empty */
365     buffstat=0 ;
366   else
367   {
368     if (l_val & 0x4)      /* buffer full */
369       buffstat=2 ;
370     else
371       buffstat=1 ;        /* something between empty and full */
372   }
373
374   return buffstat ;
375 }
376
377
378 int CaenV785_CheckBusyStatus(CaenV785_module *module)
379 {
380   // checks BUSY bit in status register 1 (#2) if module is busy (1) or not (0)
381   return ((module->reg->status_reg_1 >> 2) & 0x1);
382 }
383
384
385 void CaenV785_OverflowSuppressionOn(CaenV785_module *module)
386 {
387   module->reg->bit_clear_2 = 0x8;  /* clear Bit 3 */
388 }
389
390
391 void CaenV785_OverflowSuppressionOff(CaenV785_module *module)
392 {
393   module->reg->bit_set_2 = 0x8;    /* set Bit 3 */
394 }
395
396
397 void CaenV785_PrintInfo(CaenV785_module *module)
398 {
399    // display information on the status of the module
400    int i;
401    printf("\n************************************ Info Caen V785 ADC ************************************ \n") ;
402    for( i=0 ; i < 32; i++){
403       if ( ((module->reg->thresholds[i] & 0x100)>>8) == 0) printf("Channel %d    Threshold  %d  \n",i,module->reg->
             thresholds[i] & 0xff) ;
404    }
405    if ( (module->reg->bit_set_2 & 0x100)==0 ) printf("Threshold resolution is x16 \n") ;
406    if ( (module->reg->bit_set_2 & 0x100)==0x100 ) printf("Threshold resolution is x2 \n") ;
407    if ( (module->reg->bit_set_2 & 0x10)==0 ) printf("Zero suppression is on \n") ;
408    if ( (module->reg->bit_set_2 & 0x10)==0x10 ) printf("Zero suppression is off \n") ;
```

```
409     if ( (module->reg->bit_set_2 & 0x800)==0 ) printf("Automatic increment of the readout pointer is off  \n") ;
410     if ( (module->reg->bit_set_2 & 0x800)==0x800 ) printf("Automatic increment of the readout pointer is on  \n") ;
411     if ( (module->reg->bit_set_2 & 0x8)==0 ) printf("Overflow suppression is on \n") ;
412     if ( (module->reg->bit_set_2 & 0x8)==0x8 ) printf("Overflow suppression is off \n");
413  //  if ( (module->reg->bit_set_2 & 0x400)==0 ) printf("Common start mode \n");
414  //  if ( (module->reg->bit_set_2 & 0x400)==0x400 ) printf("Common stop mode \n");
415     if ( (module->reg->bit_set_2 & 0x80)==0 ) printf("Sliding scale is off  \n") ;
416     if ( (module->reg->bit_set_2 & 0x80)==0x80 ) printf("Sliding scale is on  \n") ;
417 //    if ( &(module->reg->thresholds[31])== (volatile int16_t*) 0xf00110be ) printf("Address Map is ok \n") ;
418 //       else printf("Error: Address Map is wrong \n");
419     printf("********************************************************************************************** \n") ;
420     printf("\n");
421     fflush(stdout);
422     return ;
423 }
```

A.2.6 CaenV785.h

Die zum TDC analoge Headerdatei des ADCs.

```
1  #ifndef CAEN_V785_H
2  #define CAEN_V785_H
3
4  #include <inttypes.h>
5
6  /* caen VME ADC V785 */
7
8  #define CAEN_OFFSET 0x10000
9  #define CAEN_V785_OFFSET 0x10000
10 #define CAEN_V785_REG_OFFSET 0x1000
11
12 #define V767__HEADER_TYPE_SHIFT 20
13 #define V767__HEADER_TYPE_MASK 0x6
14 #define CAEN_V775__OUTPUT_BUFFER_GEO_N_WORD_COUNT   8
15 #define CAEN_V775__OUTPUT_BUFFER_WORD_COUNT         0x3F
16 #define CAEN_V785__OUTPUT_BUFFER_GEO_N_WORD_COUNT   8
17 #define CAEN_V785__OUTPUT_BUFFER_WORD_COUNT         0x3F
18
19 /* Address map of module. See module manual for details. */
20 struct CaenV785_reg
21 {
22   volatile int16_t firmware_revision;
23   volatile int16_t geo_addr;
24   volatile int16_t mcst_addr;
25
26   volatile int16_t bit_set_1;
27   volatile int16_t bit_clear_1;
28
29   volatile int16_t interrupt_level;
30   volatile int16_t interrupt_vector;
31
32   volatile int16_t status_reg_1;
33   volatile int16_t control_reg_1;
34
35   volatile int16_t ader_h;
36   volatile int16_t ader_l;
37   volatile int16_t single_shot_reset;
38
39   volatile int16_t dummy0;
40
41   volatile int16_t mcst_ctrl;
42
```

```
43    volatile int16_t dummy1[2];
44
45    volatile int16_t event_trig_reg;
46    volatile int16_t status_reg_2;
47
48    volatile int16_t event_counter_l;
49    volatile int16_t event_counter_h;
50    volatile int16_t inc_event;
51    volatile int16_t inc_offset;
52
53    volatile int16_t load_test_reg;
54    volatile int16_t fclr_window;
55
56    volatile int16_t dummy2;
57
58    volatile int16_t bit_set_2;
59    volatile int16_t bit_clear_2;
60
61    volatile int16_t w_mem_test_addr;
62    volatile int16_t mem_test_wh;
63    volatile int16_t mem_test_wl;
64
65    volatile int16_t crate_select;
66    volatile int16_t test_event_write;
67    volatile int16_t event_counter_reset;
68
69    volatile int16_t dummy3[15];
70
71    volatile int16_t full_scale_range;   /* TDC only */
72
73    volatile int16_t dummy4;
74
75    volatile int16_t r_test_addr;
76
77    volatile int16_t dummy5;
78
79    volatile int16_t sw_comm;
80    volatile int16_t slide_const;
81
82    volatile int16_t dummy6[2];
83
84    volatile int16_t aad;
85    volatile int16_t bad;
86
87    volatile int16_t dummy7[6];
88
89    volatile int16_t thresholds[32];
90 };
91
92
93 struct CaenV785_module
94 {
95     volatile uint32_t *output_buffer;
96     volatile struct CaenV785_reg *reg;
97 };
98
99 typedef struct CaenV785_module CaenV785_module;
100
101
102 /**
103     Initialize module pointers.
104     'address' is the number represented by the 4 rotary
105     switches on the piggy-back boards.
106 */
107 CaenV785_module CaenV785_module_init(uint16_t address, long *pl_loc_hwacc);
108
```

```
109
110 /** @brief readout the output buffer of one card and copy it to the mbs system
111  * writes
112  *  0x01
113  *  0x03
114  *  0x05 if module busy and meb not full
115  *  0x07 if meb empty
116     returns number of read words
117 **/
118 int CaenV785_Readout(CaenV785_module *module, long *mbs_data);
119
120 // this is the interface to the module
121 void CaenV785_SoftReset (CaenV785_module *module);
122 void CaenV785_StandardSetup(CaenV785_module *module);
123
124 void CaenV785_SlidingScaleOn(CaenV785_module *module);
125 void CaenV785_SlidingScaleOff(CaenV785_module *module);
126
127 void CaenV785_StepThresholdOn(CaenV785_module *module);
128 void CaenV785_StepThresholdOff(CaenV785_module *module);
129
130 void CaenV785_ZeroSuppressionOn(CaenV785_module *module);
131 void CaenV785_ZeroSuppressionOff(CaenV785_module *module);
132
133 void CaenV785_EnableAllChannels(CaenV785_module *module);
134 void CaenV785_DisableChannels(CaenV785_module *module, unsigned long mask);
135
136 void CaenV785_SetThreshold(CaenV785_module *module, int n, int val);
137 void CaenV785_ResetThresholds(CaenV785_module *module);
138 void CaenV785_DisplayStoredThresholds(CaenV785_module *module);
139
140 void CaenV785_SetFileThresholds(CaenV785_module *module);
141
142 int CaenV785_CheckDreadyStatus(CaenV785_module *module);
143 void CaenV785_ClearData(CaenV785_module *module);
144 int CaenV785_CheckBufferStatus(CaenV785_module *module);
145 int CaenV785_CheckBusyStatus(CaenV785_module *module);
146
147 void CaenV785_OverflowSuppressionOn(CaenV785_module *module);
148 void CaenV785_OverflowSuppressionOff(CaenV785_module *module);
149
150 void CaenV785_PrintInfo(CaenV785_module *module);
151
152 #endif
```

A.2.7 setup.usf (Zeile 141-146)

Die Konfigurationsdatei `setup.usf` wird unter anderem dazu verwendet um die Trivaeinheit zu konfigurieren. Hier kann das Zeitfenster nach erhalten eines Triggers näher spezifiziert werden, aber auch verschiedene Einstellungen am gesamten MBS vorgenommen werden. In diesem Ausschnitt sieht man die aktuelle Einstellung des Zeitfensters, die bei 13,4 µs liegt.

```
1 #-------------------------------------------------------------------------
2 #                 crate. nr: 0     1     2     3     4     5     6     7
3       TRIG_CVT           = (134,   0,    0,    0,    0,    0,    0,    0,-
4 #                 crate. nr: 8     9     10    11    12    13    14    15
5                            0,    0,    0,    0,    0,    0,    0,    0),-
6 #*************************************************************************
```

A.2.8 startup.scom

Die Skriptdatei `startup.scom` fasst einige MBS-Befehle zusammen, die für das ordnungsgemäße Starten einer Messung benötigt werden. Unter anderem wird hier die Datei `setup.usf` geladen, der Transport der Messdaten eingerichtet sowie der Speicherort und Speichername festgelegt. Die Messung selbst muss separat mit dem Befehl `start acquisition` gestartet werden.

```
start task m_util
load setup setup.usf
set trig_mod
start task m_read_meb "./m_read_meb"
start task m_collector
start task m_transport
start task m_daq_rate
open file "/d/d1/qclam/sergej/data.lmd" -disk
```

A.2.9 shutdown.scom

Die Datei `shutdown.scom` ist entsprechend für das ordnungsgemäße Beenden der Messung verantwortlich. Im Gegensatz zum Starten der Messung, muss der Stoppbefehl nicht separat eingegeben werden, so dass das Ausführen dieses Skripts zum ordnungsgemäßen Beenden der Messung ausreichend ist.

```

stop acq
close file
stop task m_transport
stop task m_collector
stop task m_read_meb
stop task m_util
```

A.2.10 listmode2plot.c

Das Programm `listmode2plot` wandelt die Rohdaten der Messung in Textdateien um, die die Kanäle und die gemessene Anzahl der Ereignisse enthalten. Bei längeren Messungen kann die Umwandlung bis zu einer Stunde dauern. Allerdings wird nach jeweils 100 000 gelesenen Ereignissen eine Textdatei mit den bis dahin konvertierten Daten ausgegeben. Somit erhält man eine Möglichkeit sich einen ersten Eindruck von der Messung zu verschaffen [12].

```
/*
 * listmode2bit.c
 *
 * converts MBS listmode file to a file with pure data
 * for testing purposes
 *
 * 08/2012 Henno Lauinger
 *
 */
```

```
10
11 #include <stdio.h>
12 #include <stdlib.h>
13 #include <math.h>
14 #include <string.h>
15
16 #include "typedefs.h"
17 #include "s_filhe_swap.h"
18 #include "s_bufhe_swap.h"
19 #include "s_ve10_1.h"
20 #include "s_ves10_1.h"
21 #include "s_evhe_swap.h"
22 #include "f_evt.h"
23
24 #include "histogram_structs.h"
25 #include "adc_785.h"
26 #include "adc_785_debug_tools.h"
27
28
29 static struct energy_count* energy_hist[32];
30 static struct energy_count* energy_hist_all;
31 unsigned int evt_counter = 0;
32 unsigned int evt_k_counter = 0;
33 static char* file_out_name = "plotfiles/output.dat";
34
35
36 int readListmodeFile(char* filename) {
37   s_evt_channel *input_channel;
38   s_filhe *file_header;
39   s_bufhe *buffer_header;
40
41   int32_t *event_data;
42   int events_read = 0;
43
44   // initialize the input channel
45   input_channel = f_evt_control();
46
47
48     /*+   first argument of f_evt_get_open()    : Type of server:         */
49     /*-               GETEVT__FILE   : Input from file                    */
50     /*-               GETEVT__STREAM : Input from MBS stream server       */
51     /*-               GETEVT__TRANS  : Input from MBS transport           */
52     /*-               GETEVT__EVENT  : Input from MBS event server        */
53     /*-               GETEVT__REVSERV: Input from remote event server     */
54     //   second argument of f_evt_get_open()    : name of server (for ex. "r4-4" in case of online analysis or "filename
           .lmd" in case of reading data from file)
55     int32_t result = f_evt_get_open(GETEVT__FILE, filename, input_channel, (CHARS**) (&file_header), 1, 0);
56
57     printf("result = %i\n", result);
58
59     if (result != GETEVT__SUCCESS)
60       return 1;
61
62     if (file_header != 0)
63     {
64       printf("filhe_dlen : %i\n", file_header->filhe_dlen);
65       printf("filhe_file : %s\n", file_header->filhe_file);
66       printf("filhe_user : %s\n", file_header->filhe_user);
67     }
68
69     printf("\n");
70
71
72     for(;;)
73     {
74       result = f_evt_get_event(input_channel, &event_data, (INTS4**)(&buffer_header));
```

```
75        if (result != GETEVT__SUCCESS)
76      break;
77
78        ++events_read;
79        // uncomment the following lines to output the "raw data and header info from the event"
80        /*            std::cout << std::endl << "processing event number " << std::dec << events_read << std::endl;
81        *
82        *     std::cout << "f_evt_type(...) output: " << std::endl;
83        *     f_evt_type(buffer_header, (s_evhe*)event_data, -1, 0,1,0);
84        *     std::cout << "-------------------------------------------------" << std::endl;*/
85
86        int sub;
87        for (sub = 1; result != GETEVT__NOMORE; ++sub)
88        {
89      s_ves10_1 *subevent_header;
90      int32_t *data;
91      int32_t length;
92
93      result = f_evt_get_subevent((s_ve10_1*)event_data, sub, (int32_t**)&subevent_header, &data, &length);
94
95      if (result == GETEVT__SUCCESS)
96      {
97        // do something with the subevent data
98        output(data, length);
99      }
100       }
101
102     }
103     printf(" read %i events \n", events_read);
104
105     f_evt_get_close(input_channel);
106 }
107
108
109
110 int output(int32_t *data, size_t length) {
111   // write data to struct
112   int i;
113   for (i = 0; i < length; i++){
114     int32_t type = getType(*data);
115     if (type == type_data) {
116 /*      addNewEnergyNode(getData(*data), energy_hist[getChannel(*data)]);
117       addNewEnergyNode(getData(*data), energy_hist_all);*/
118       addEnergySorted(getData(*data), energy_hist[getChannel(*data)]);
119       addEnergySorted(getData(*data), energy_hist_all);
120     }
121     else if (type != type_header && type != type_eob) {
122       char* string = (char*) malloc(33);
123       toBinaryString(*data, string);
124       printf("no data (%i): %i/%i\t%s\n", type, i+1, (int) length, string);
125       free(string);
126     }
127     data++;
128   }
129
130
131   if (evt_counter++ >= 100000) {
132     evt_k_counter++;
133     fprintf(stderr, "writing first %i events\n", evt_k_counter * 100000);
134     writeData();
135     fprintf(stderr, "done\n");
136     evt_counter = 0;
137   }
138
139   return 0;
140 }
```

```
141
142
143
144 int writeData() {
145
146   // open files
147   FILE* f_out = fopen(file_out_name, "wb");
148   if (f_out == NULL) {
149     fprintf(stderr, "couldn't open output file\n");
150     return 1;
151   }
152   fprintf(f_out, "# energy\tcount\r\n");
153
154   FILE* f_ch_out[32];
155   int32_t i;
156   char* f_ch_name = (char*) malloc(19);
157   for (i = 0; i < 32; i++) {
158     sprintf(f_ch_name, "plotfiles/ch%i.dat", i);
159     f_ch_out[i] = fopen(f_ch_name, "wb");
160       if (f_out == NULL) {
161       fprintf(stderr, "couldn't open output file\n");
162       return 1;
163     }
164     fprintf(f_ch_out[i], "# energy\tcount\r\n");
165   }
166
167   // write memory to plottable files
168
169   // combined
170 /*  printEnergyList(energy_hist_all);
171   energy_hist_all = getFirstEnergyNode(sortEnergyNodes(energy_hist_all));
172   fprintf(stderr, "2\n");*/
173   energy_hist_all = getFirstEnergyNode(energy_hist_all);
174   fprintf(f_out, "%i\t%i\r\n", energy_hist_all->energy, energy_hist_all->count);
175   while (energy_hist_all->next_energy != NULL) {
176     energy_hist_all = energy_hist_all->next_energy;
177     fprintf(f_out, "%i\t%i\r\n", energy_hist_all->energy, energy_hist_all->count);
178   }
179   fclose(f_out);
180
181   // seperate channels
182   for (i = 0; i < 32; i++) {
183 //    energy_hist[i] = getFirstEnergyNode(sortEnergyNodes(energy_hist[i]));
184     energy_hist[i] = getFirstEnergyNode(energy_hist[i]);
185     fprintf(f_ch_out[i], "%i\t%i\r\n", energy_hist[i]->energy, energy_hist[i]->count);
186     while (energy_hist[i]->next_energy != NULL) {
187       energy_hist[i] = energy_hist[i]->next_energy;
188       fprintf(f_ch_out[i], "%i\t%i\r\n", energy_hist[i]->energy, energy_hist[i]->count);
189     }
190     fclose(f_ch_out[i]);
191   }
192
193
194   return 0;
195 }
196
197
198
199 int main(int argc, char *argv[]) {
200
201   // usage
202   if (argc != 2) {
203     printf("usage: \n\t%s <listmode file>\n", argv[0]);
204     printf(
205       "\t\t converts file.lmd to file.bit\n");
206     return 1;
```

```
207   }
208
209   // get input file name and generate output file name
210   int name_length = (int) strlen(argv[1]);
211   char file_name[name_length];
212   char file_base[name_length];
213   char file_ext[4];
214   file_out_name = (char*) malloc(name_length + 10);
215   strcpy(file_name, argv[1]);
216   strncpy(file_ext, file_name+name_length-3, 3);
217   strncpy(file_base, file_name, name_length-4);
218   file_base[name_length-4] = '\0';
219 //  file_base[sizeof(file_base)-1] = '\0';
220
221   strcpy(file_out_name, "plotfiles/");
222   strcat(file_out_name, file_base);
223   strcat(file_out_name, ".dat");
224
225   printf("in:\t%s\nbase:\t%s\next:\t%s\nout:\t%s\n", file_name, file_base, file_ext, file_out_name);
226
227
228
229   // initialize histogram_structs and open output files
230
231   energy_hist_all = newEnergyNode(0);
232   int i = 0;
233   for (i = 0; i < 32; i++) {
234     energy_hist[i] = newEnergyNode(0);
235   }
236
237   // read listmode file
238
239   readListmodeFile(file_name);
240
241   //write all data
242
243   writeData();
244
245
246 //  fclose(f_in);
247   return 0;
248 }
```

B Appendix: Teil II

B.1 Protonenstreuspektren

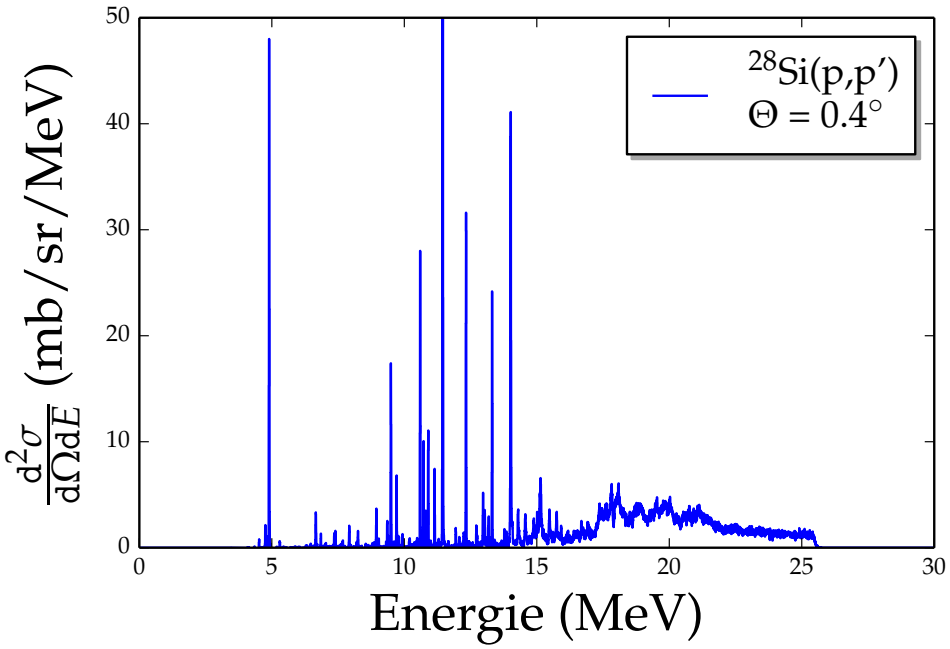


Abbildung B.1: Doppeltdifferentieller Wirkungsquerschnitt aus Protonenstreuung für ^{28}Si.

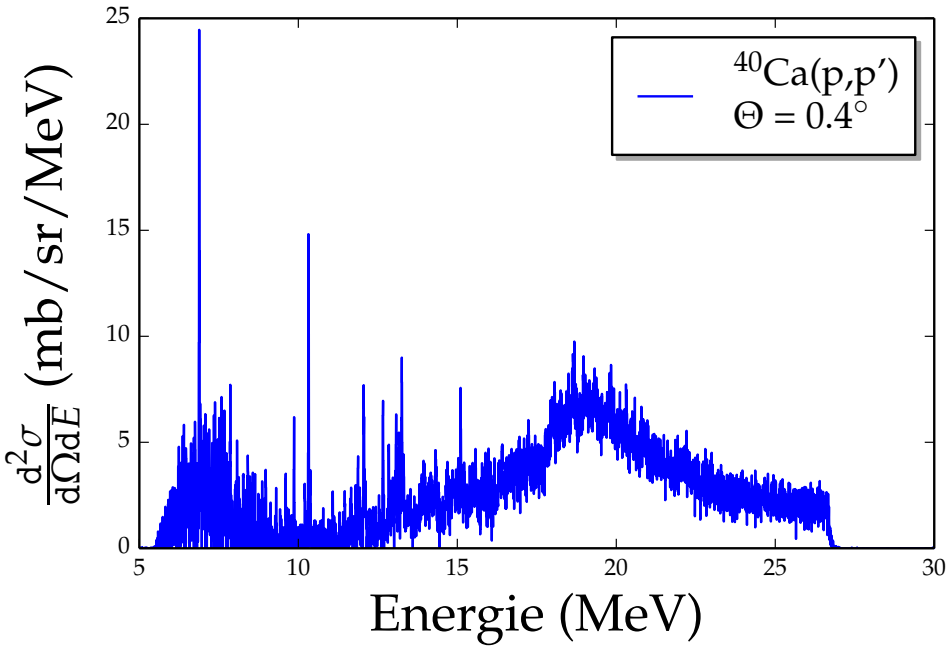


Abbildung B.2: Doppeltdifferentieller Wirkungsquerschnitt aus Protonenstreuung für ^{40}Ca.

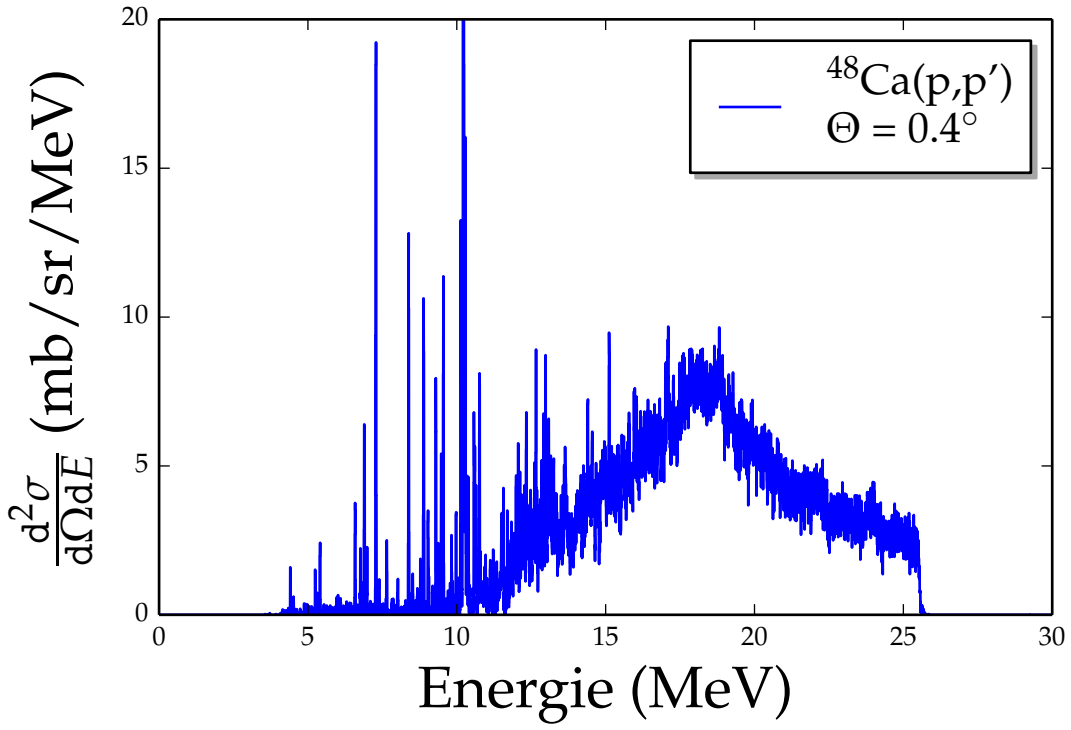


Abbildung B.3: Doppeltdifferentieller Wirkungsquerschnitt aus Protonenstreuung für ^{48}Ca.

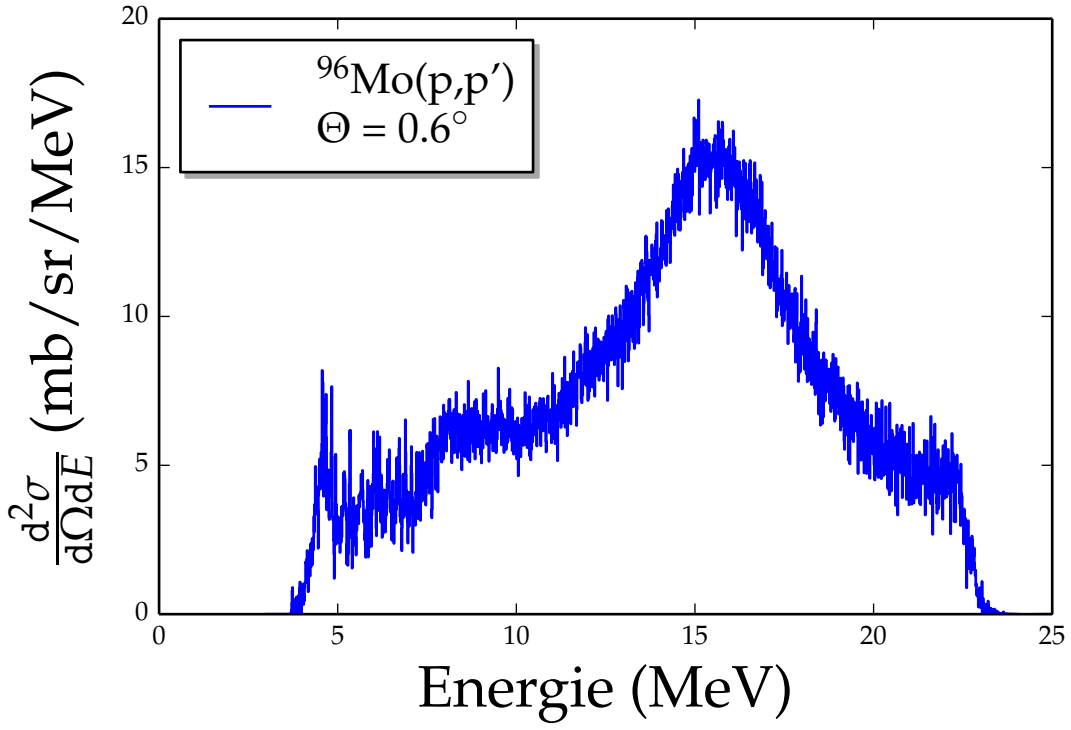


Abbildung B.4: Doppeltdifferentieller Wirkungsquerschnitt aus Protonenstreuung für ^{96}Mo.

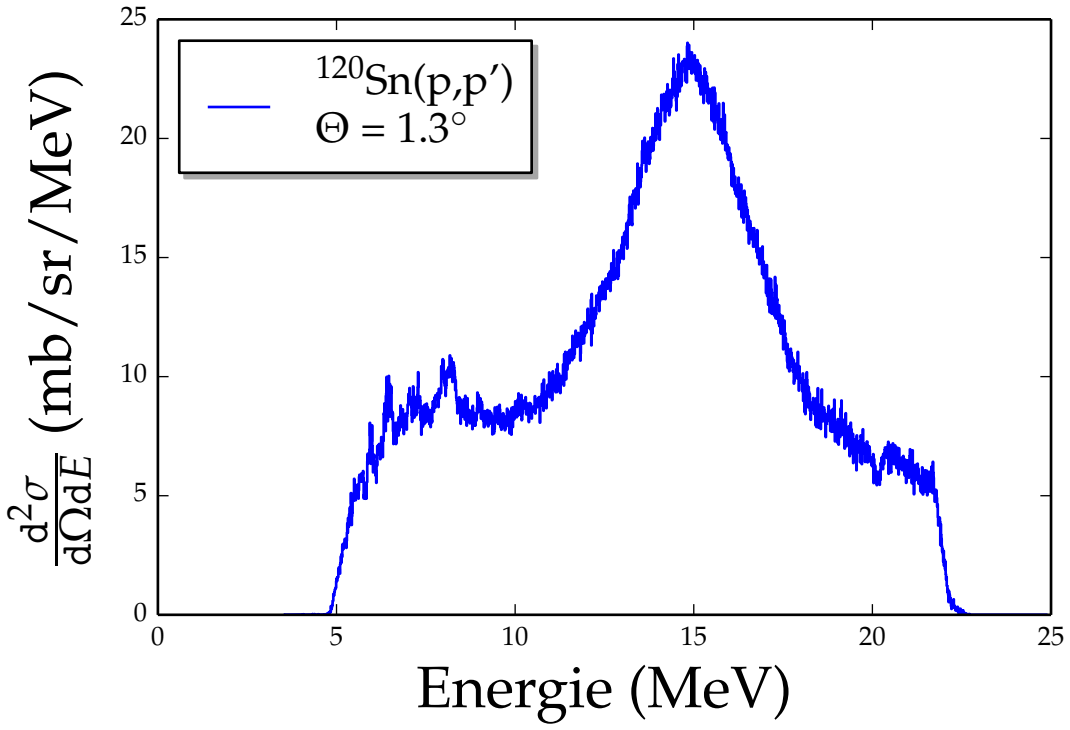


Abbildung B.5: Doppeltdifferentieller Wirkungsquerschnitt aus Protonenstreuung für ^{120}Sn.

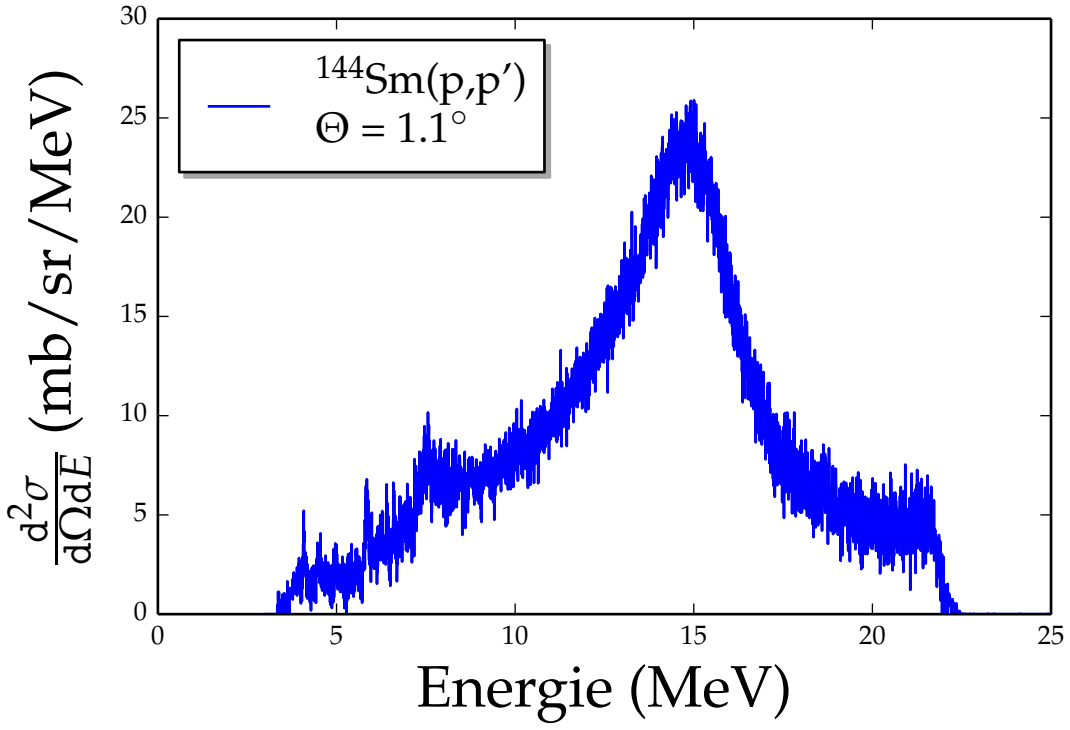


Abbildung B.6: Doppeltdifferentieller Wirkungsquerschnitt aus Protonenstreuung für ^{144}Sm.

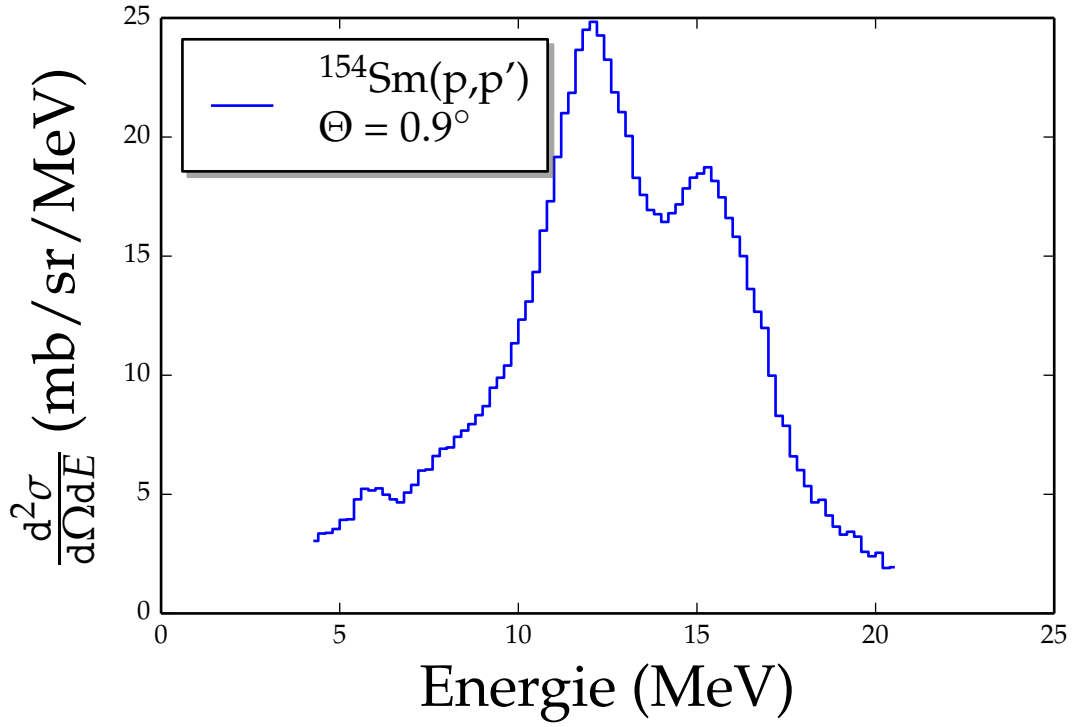


Abbildung B.7: Doppeltdifferentieller Wirkungsquerschnitt aus Protonenstreuung für ^{154}Sm. Dies ist nicht das Originalspektrum, sondern das aus der Multipolentfaltungsanalyse erhaltene Spektrum (E1-Anteil).

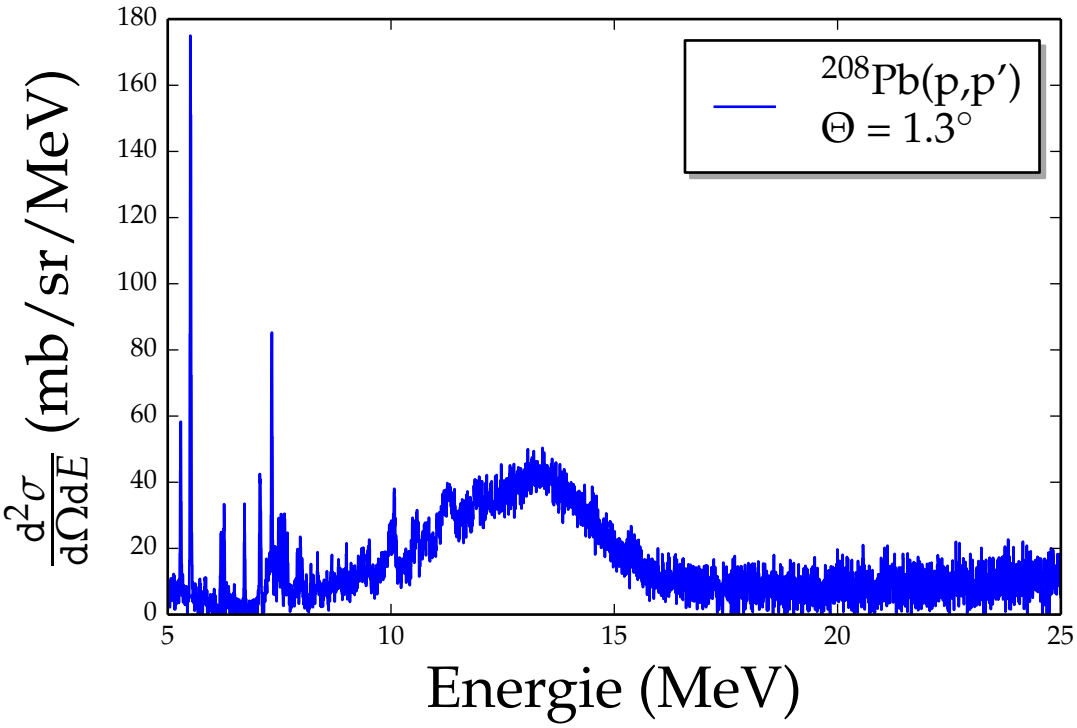


Abbildung B.8: Doppeltdifferentieller Wirkungsquerschnitt aus Protonenstreuung für ^{208}Pb.

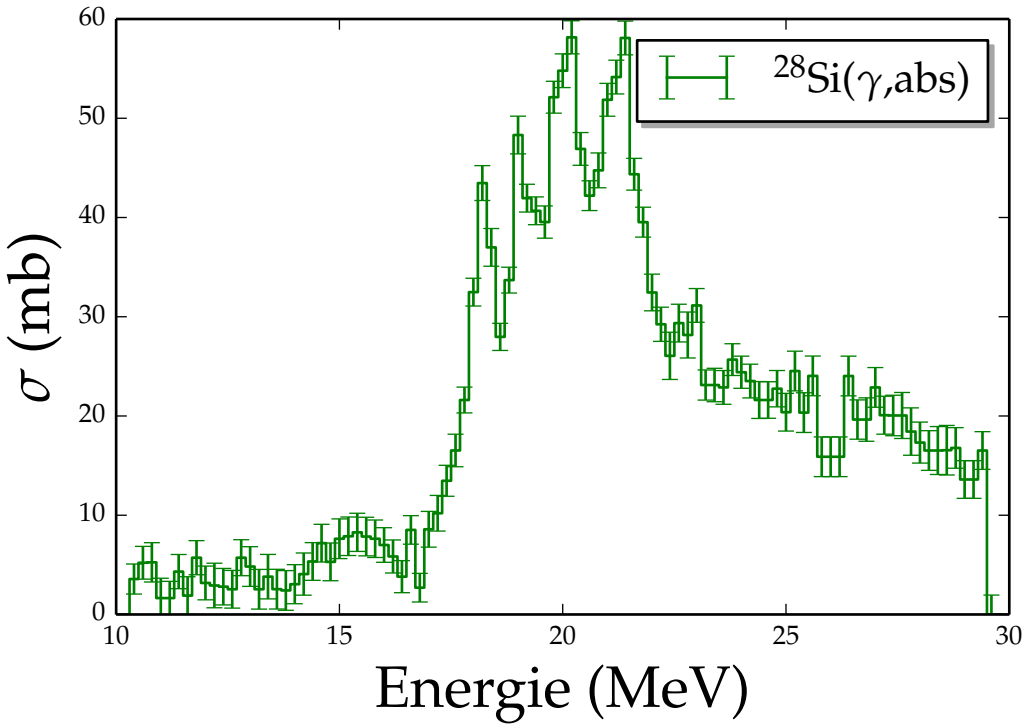


Abbildung B.9: Photoabsorptionsquerschnitt für ^{28}Si.

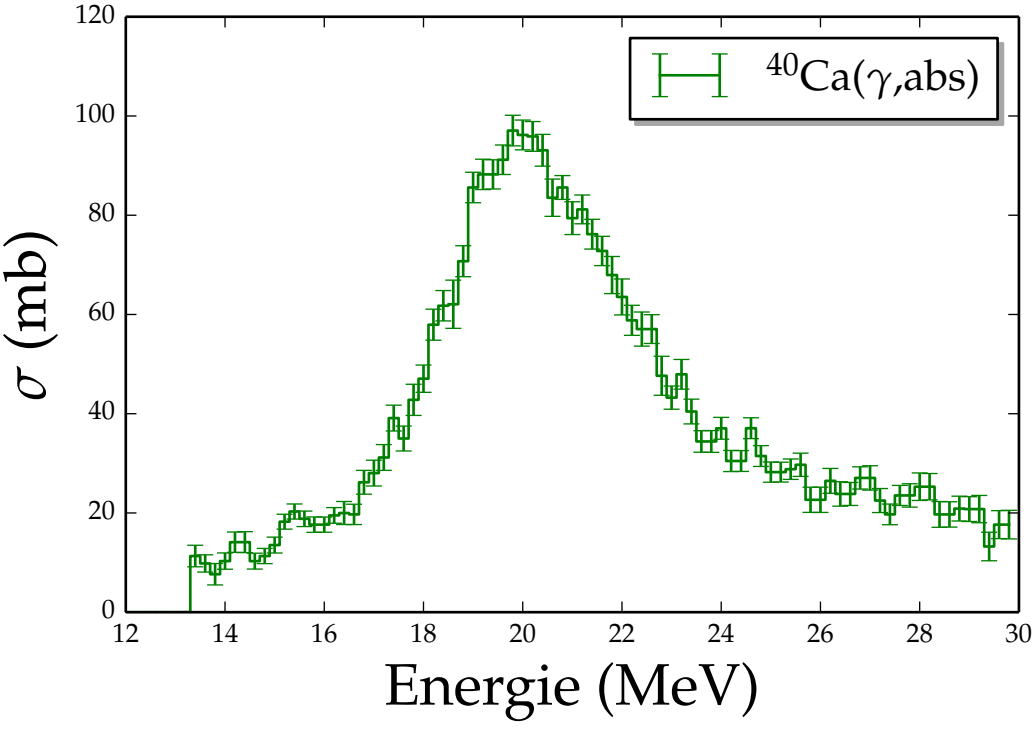


Abbildung B.10: Photoabsorptionsquerschnitt für ^{40}Ca.

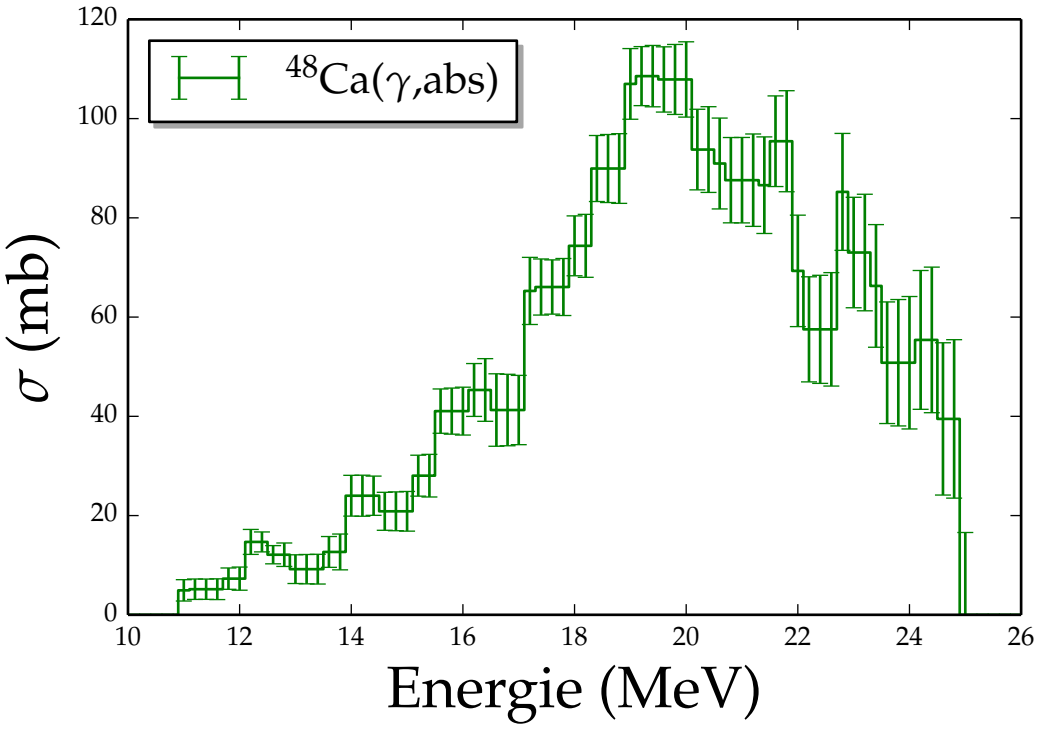


Abbildung B.11: Photoabsorptionsquerschnitt für ^{48}Ca.

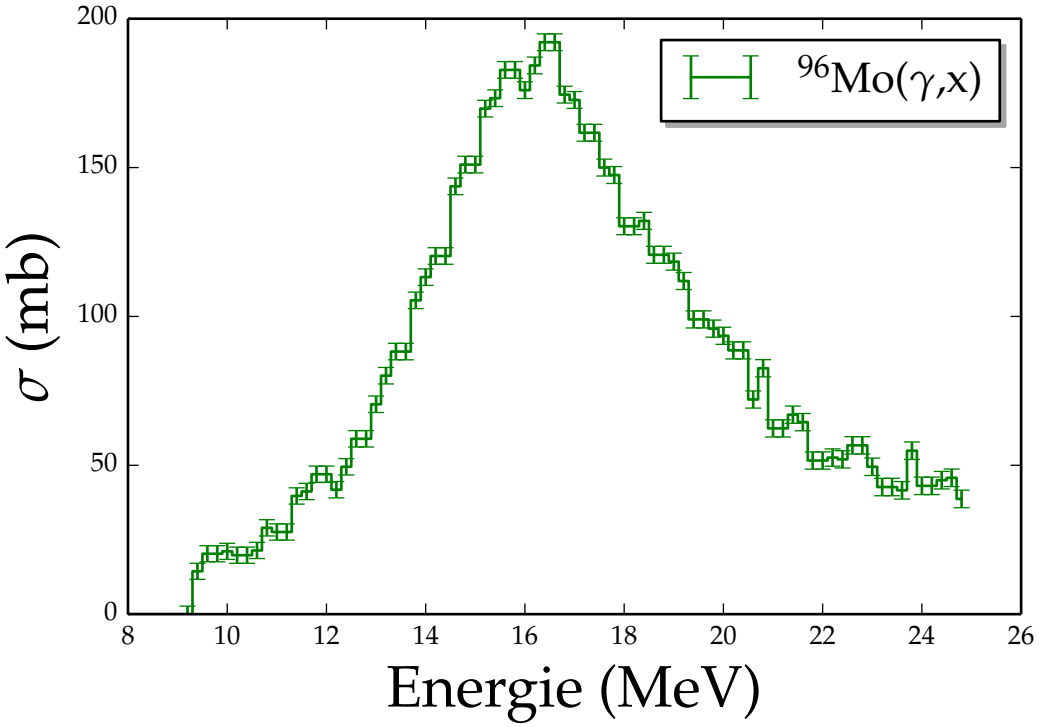


Abbildung B.12: Photoabsorptionsquerschnitt für ^{96}Mo.

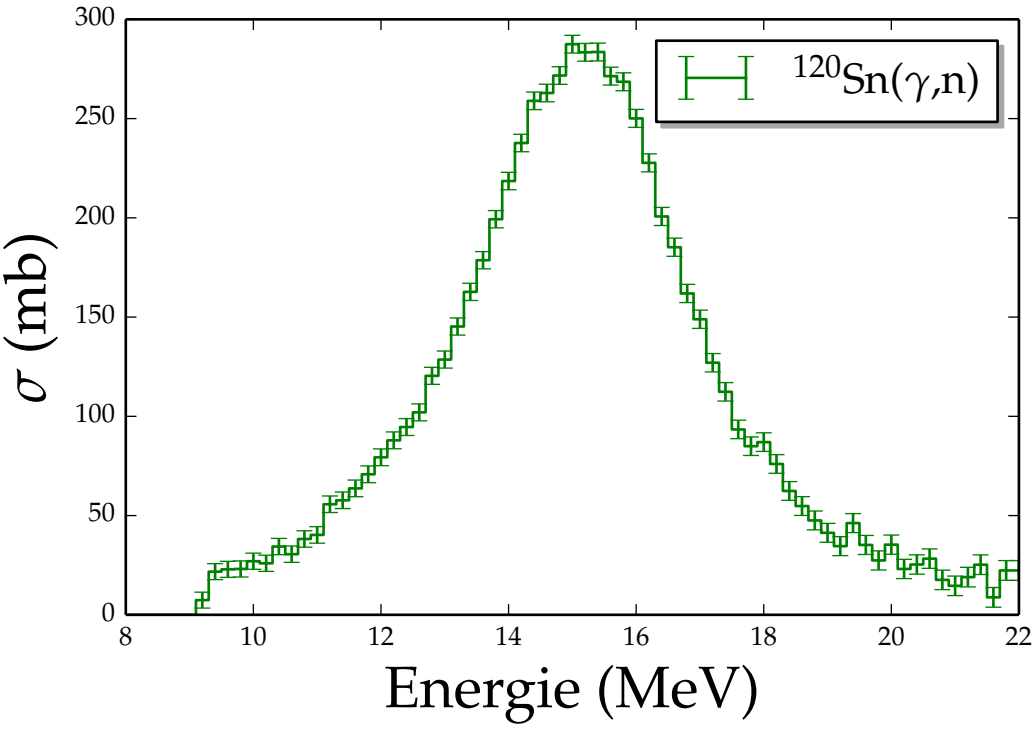


Abbildung B.13: Photoabsorptionsquerschnitt für ^{120}Sn.

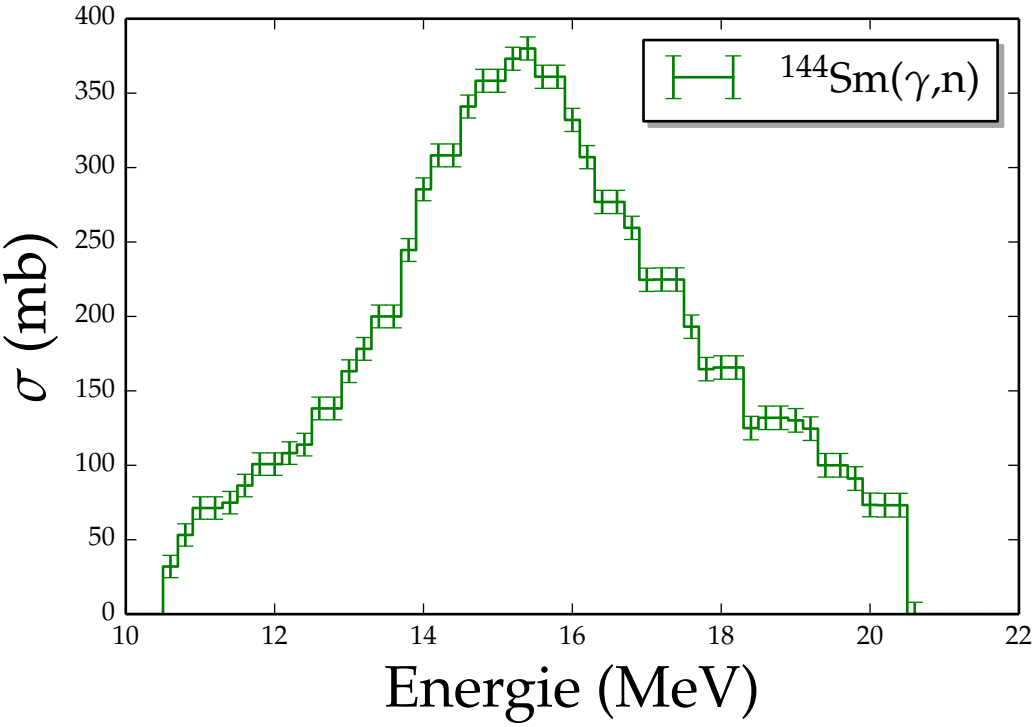


Abbildung B.14: Photoabsorptionsquerschnitt für ^{144}Sm.

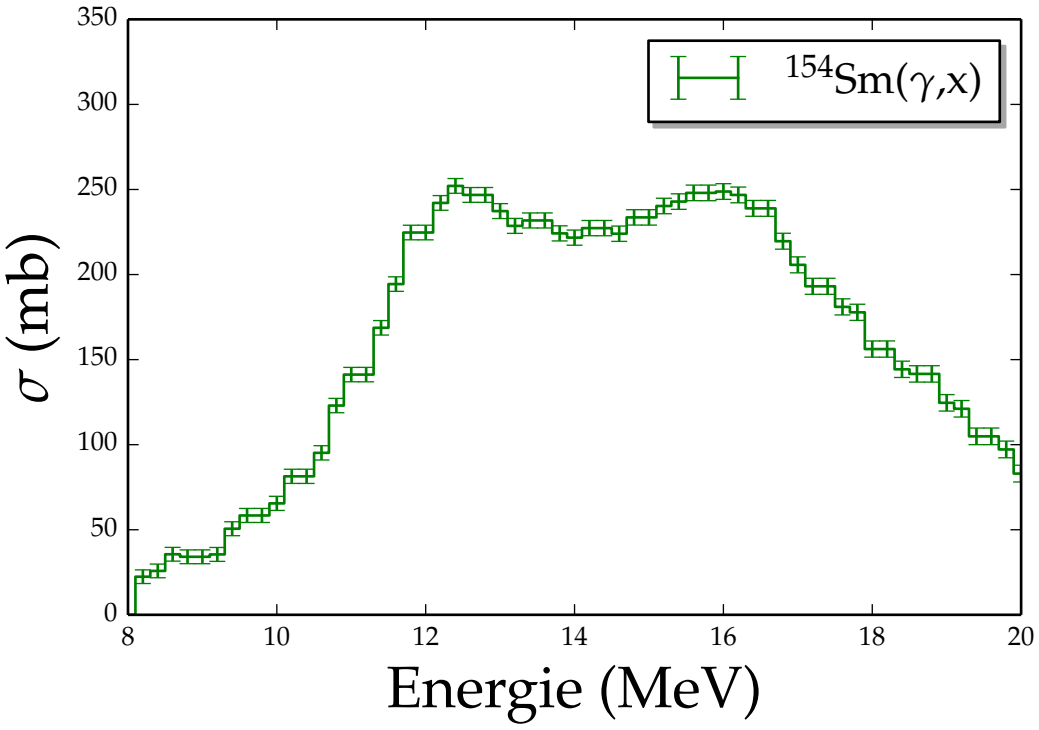


Abbildung B.15: Photoabsorptionsquerschnitt für ^{154}Sm.

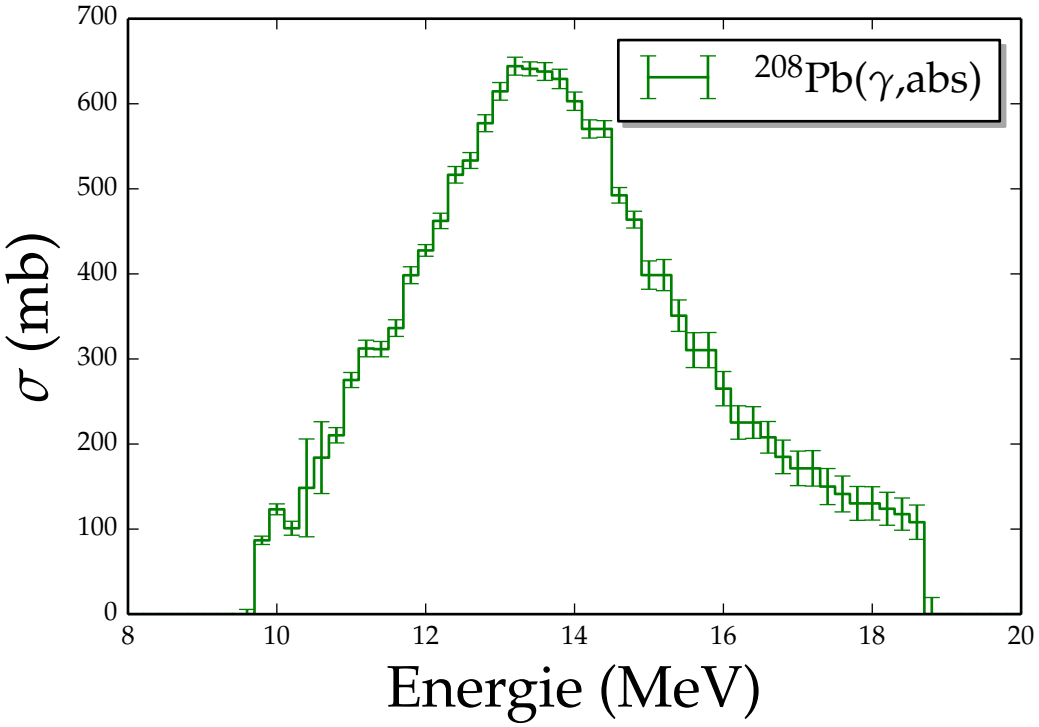


Abbildung B.16: Photoabsorptionsquerschnitt für ^{208}Pb.

B.3 Extrahierte Photoabsorptionsquerschnitte unter Verwendung der virtuellen Photonenmethode

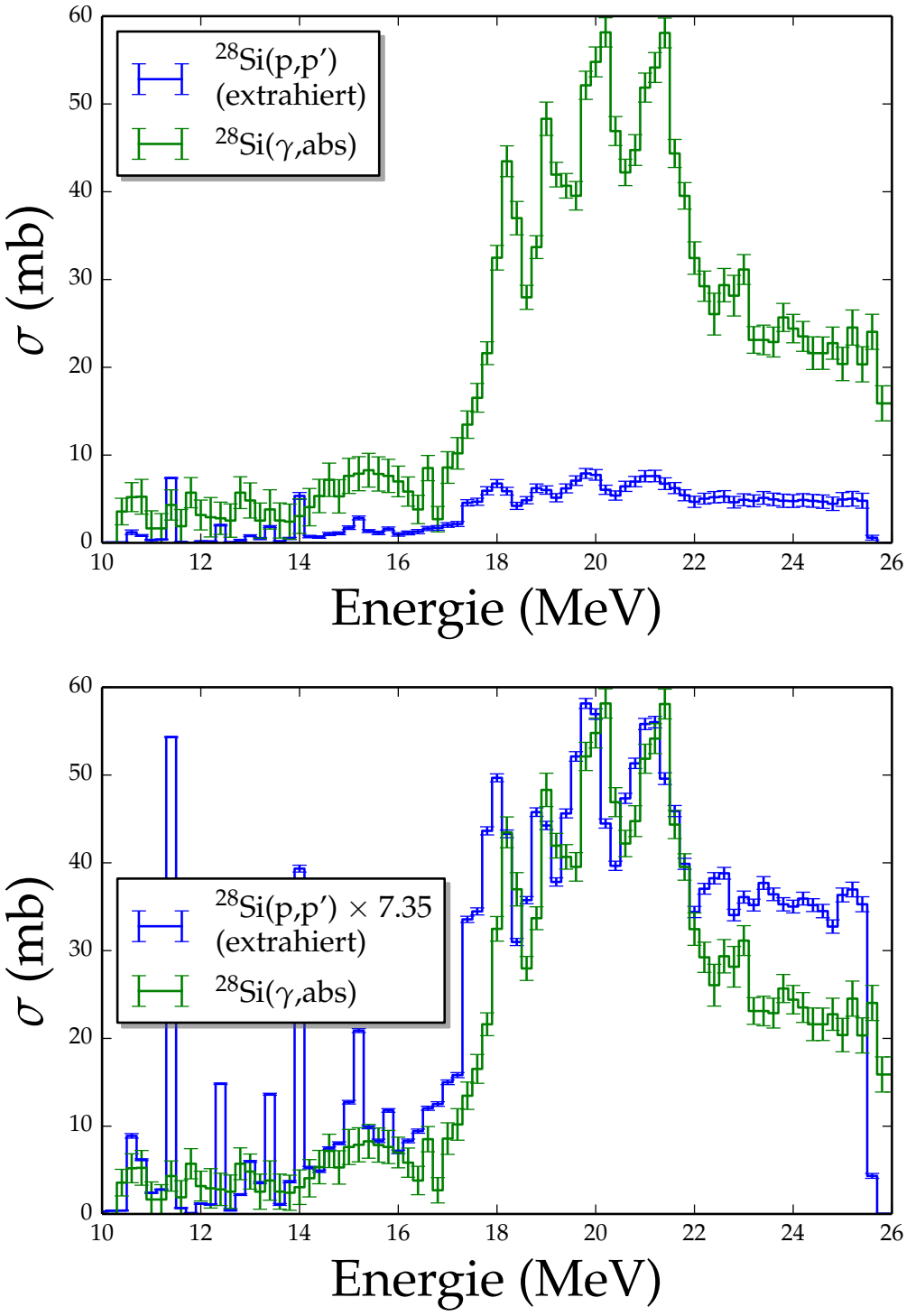


Abbildung B.17: Extrahierter bzw. experimenteller Photoabsorptionsquerschnitt für ^{28}Si. Im unteren Bild wurde der extrahierte Photoabsorptionsquerschnitt mit einem Faktor von 7,35 multipliziert.

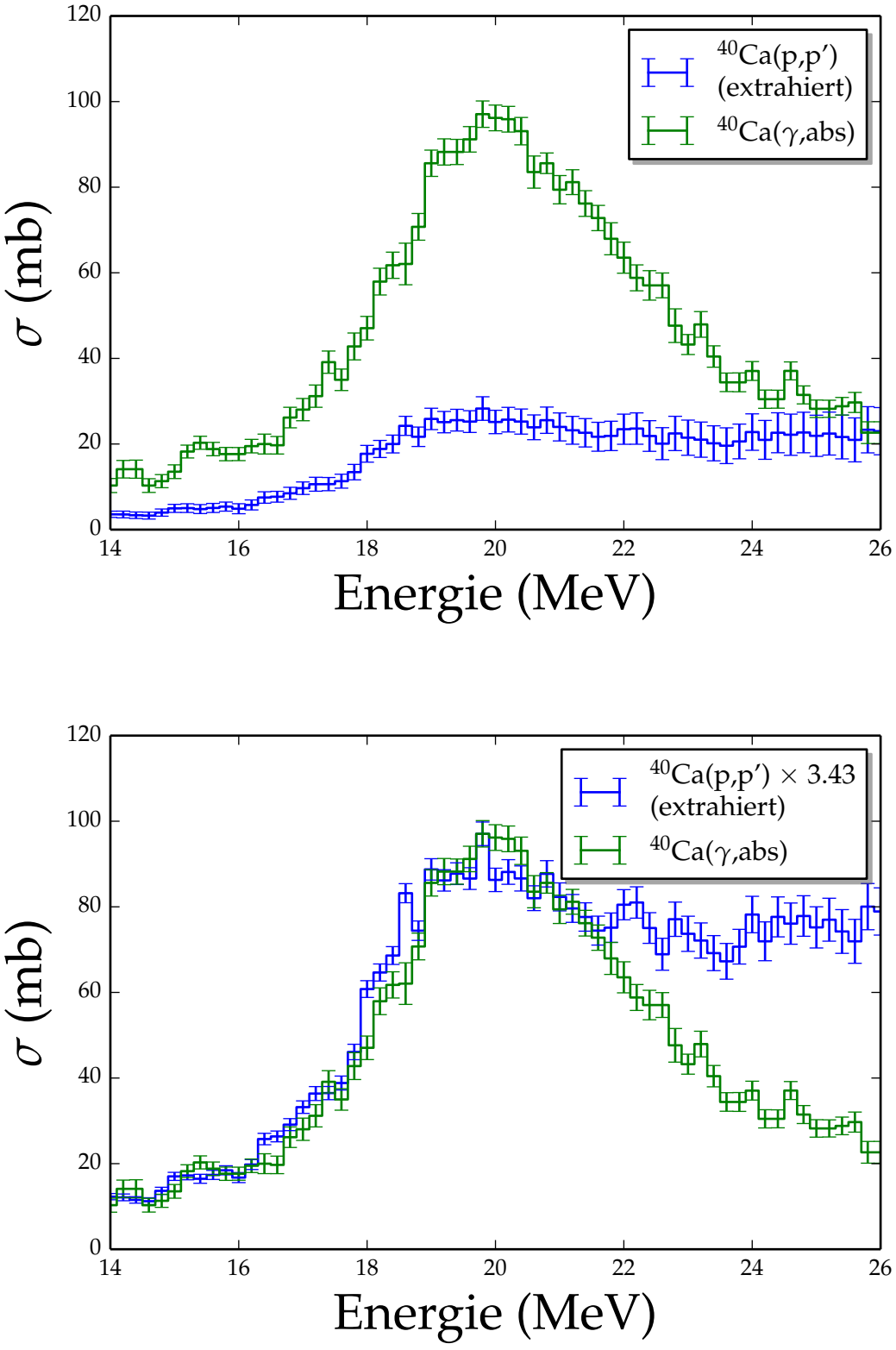


Abbildung B.18: Extrahierter bzw. experimenteller Photoabsorptionsquerschnitt für ^{40}Ca. Im unteren Bild wurde der extrahierte Photoabsorptionsquerschnitt mit einem Faktor von 3,43 multipliziert.

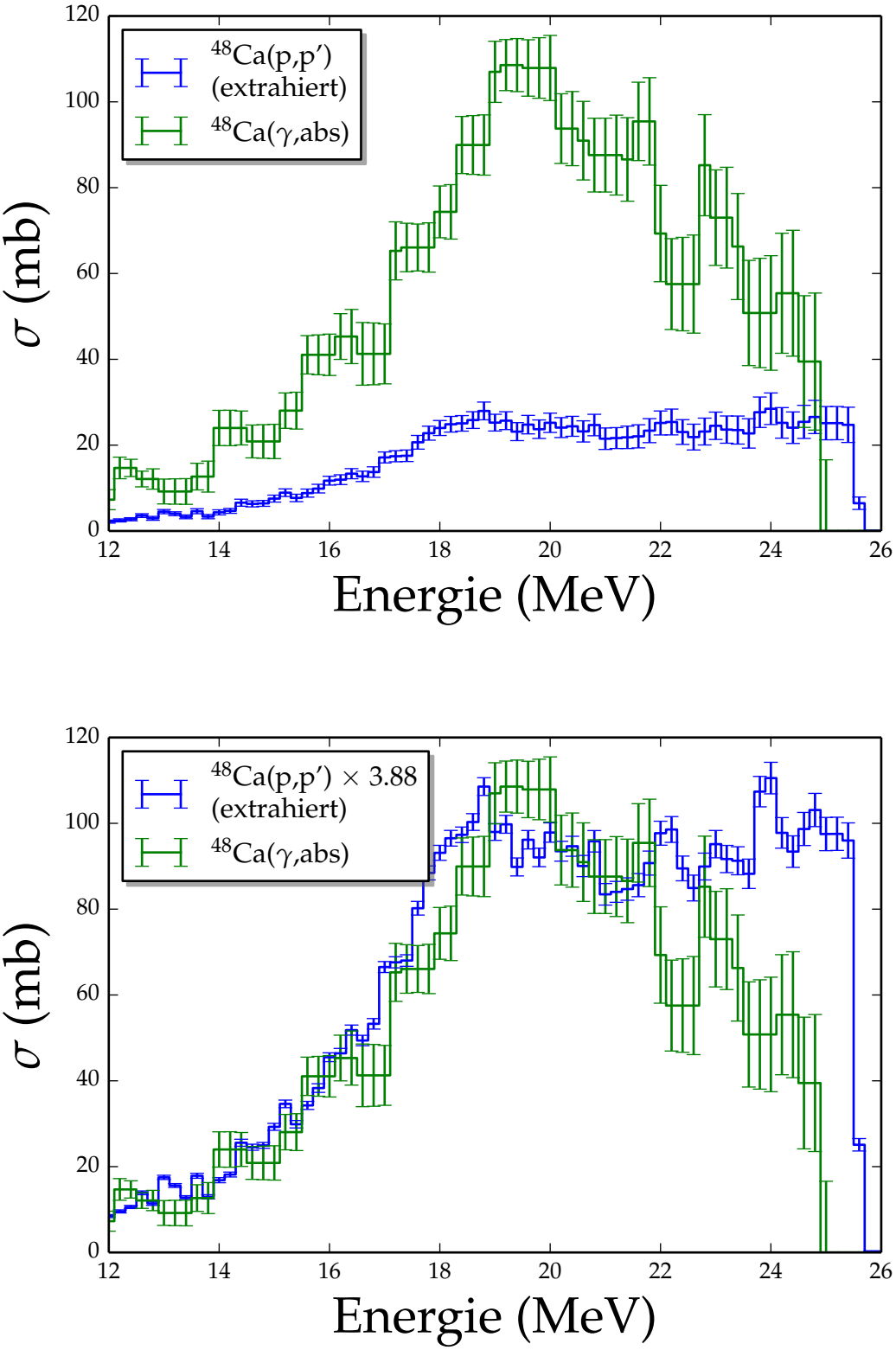
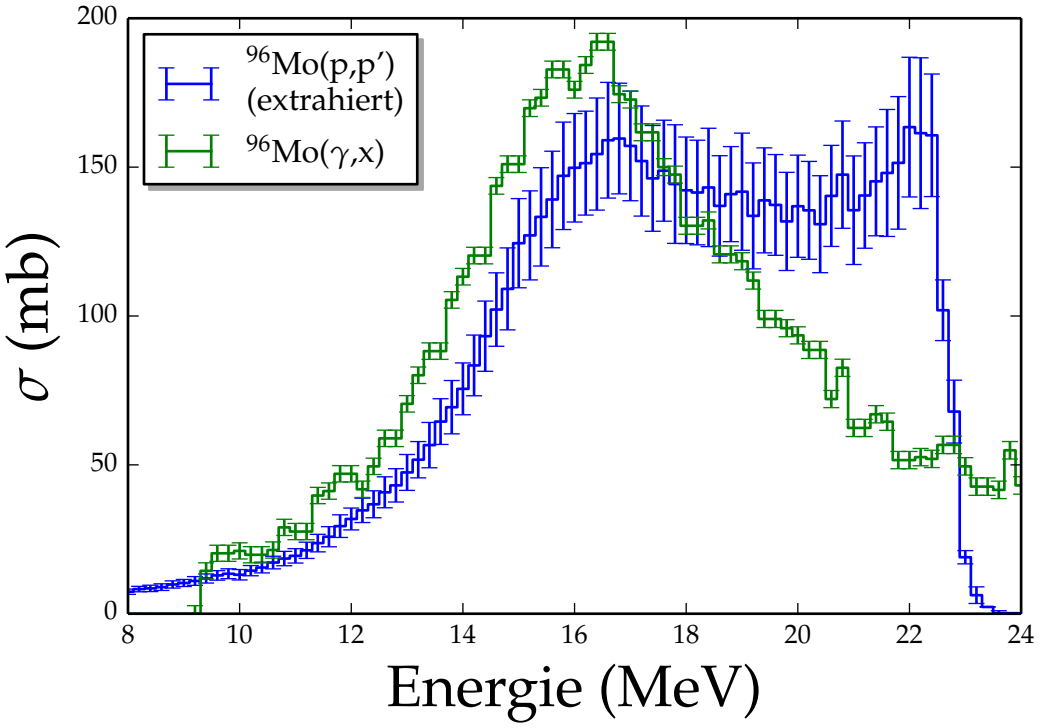


Abbildung B.19: Extrahierter bzw. experimenteller Photoabsorptionsquerschnitt für ^{48}Ca. Im unteren Bild wurde der extrahierte Photoabsorptionsquerschnitt mit einem Faktor von 3,88 multipliziert.



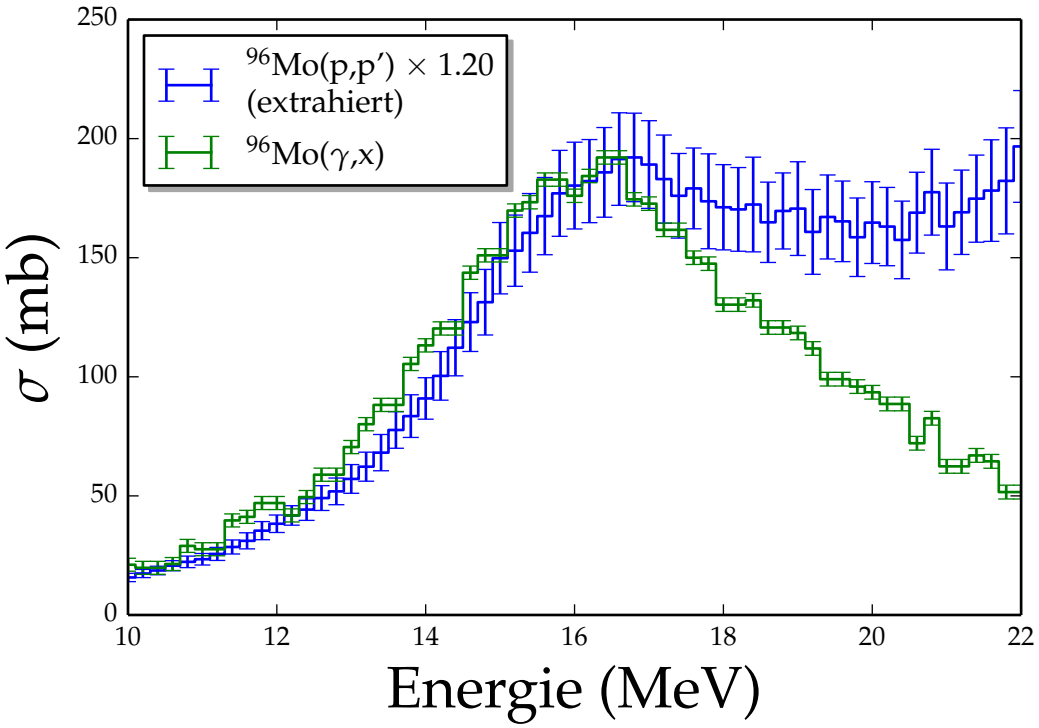
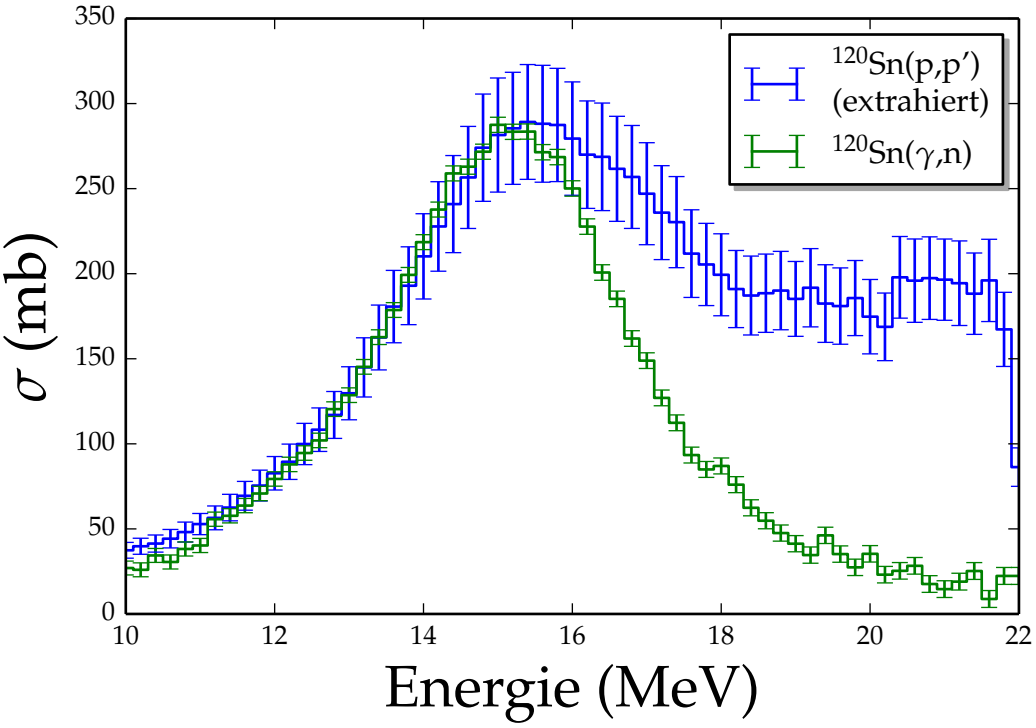


Abbildung B.20: Extrahierter bzw. experimenteller Photoabsorptionsquerschnitt für ^{96}Mo. Im unteren Bild wurde der extrahierte Photoabsorptionsquerschnitt mit einem Faktor von 1,20 multipliziert.



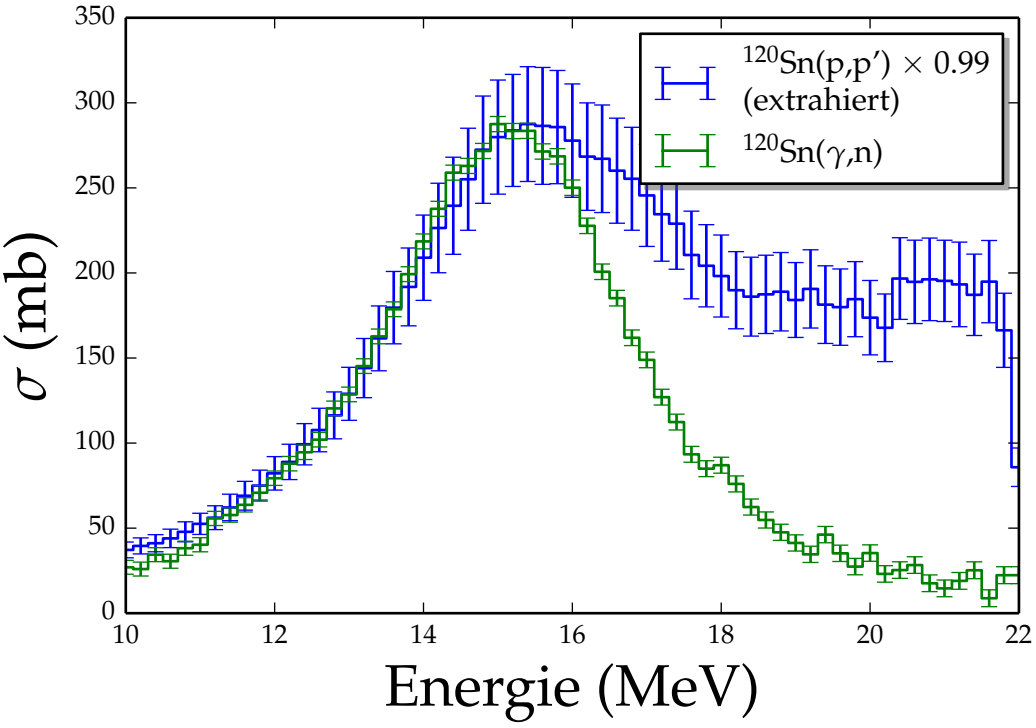


Abbildung B.21: Extrahierter bzw. experimenteller Photoabsorptionsquerschnitt für ^{120}Sn. Im unteren Bild wurde der extrahierte Photoabsorptionsquerschnitt mit einem Faktor von 0,99 multipliziert.

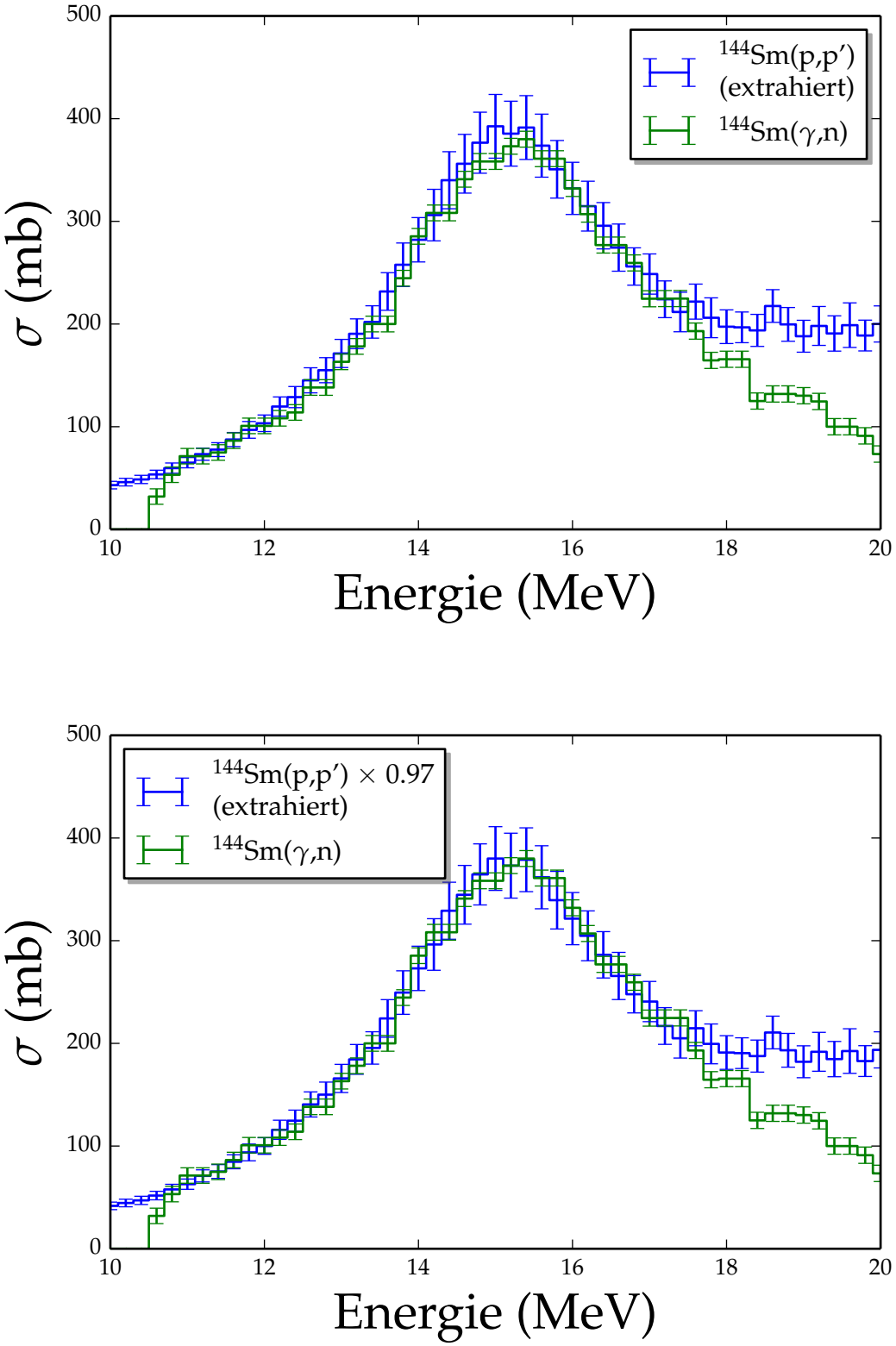


Abbildung B.22: Extrahierter bzw. experimenteller Photoabsorptionsquerschnitt für ^{144}Sm. Im unteren Bild wurde der extrahierte Photoabsorptionsquerschnitt mit einem Faktor von 0,97 multipliziert.

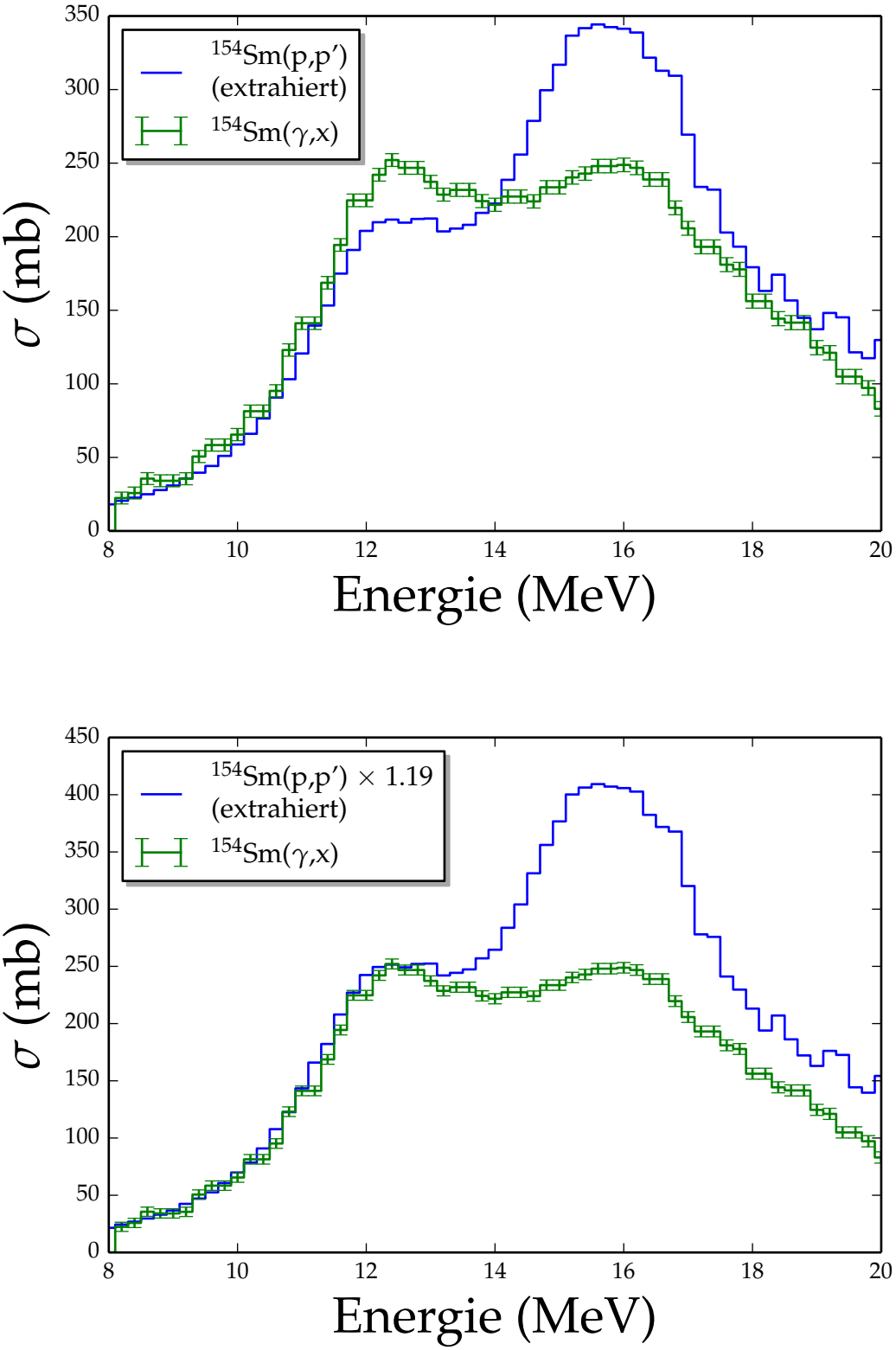


Abbildung B.23: Extrahierter bzw. experimenteller Photoabsorptionsquerschnitt für ^{154}Sm. Im unteren Bild wurde der extrahierte Photoabsorptionsquerschnitt mit einem Faktor von 1,19 multipliziert.

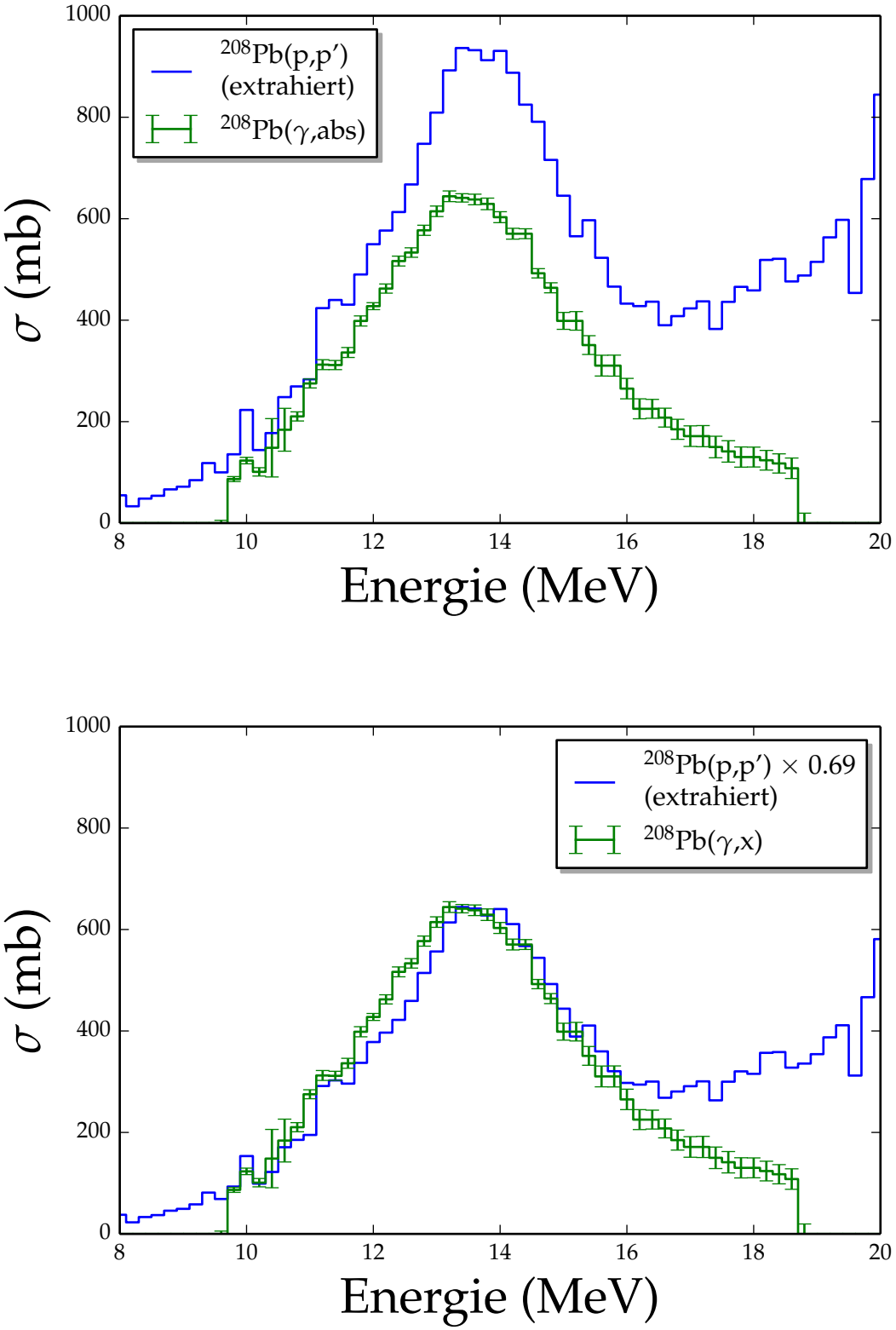


Abbildung B.24: Extrahierter bzw. experimenteller Photoabsorptionsquerschnitt für ^{208}Pb. Im unteren Bild wurde der extrahierte Photoabsorptionsquerschnitt mit einem Faktor von 0,69 multipliziert.

B.4 Extrahierte Photoabsorptionsquerschnitte unter Verwendung der modellabhängigen Methode

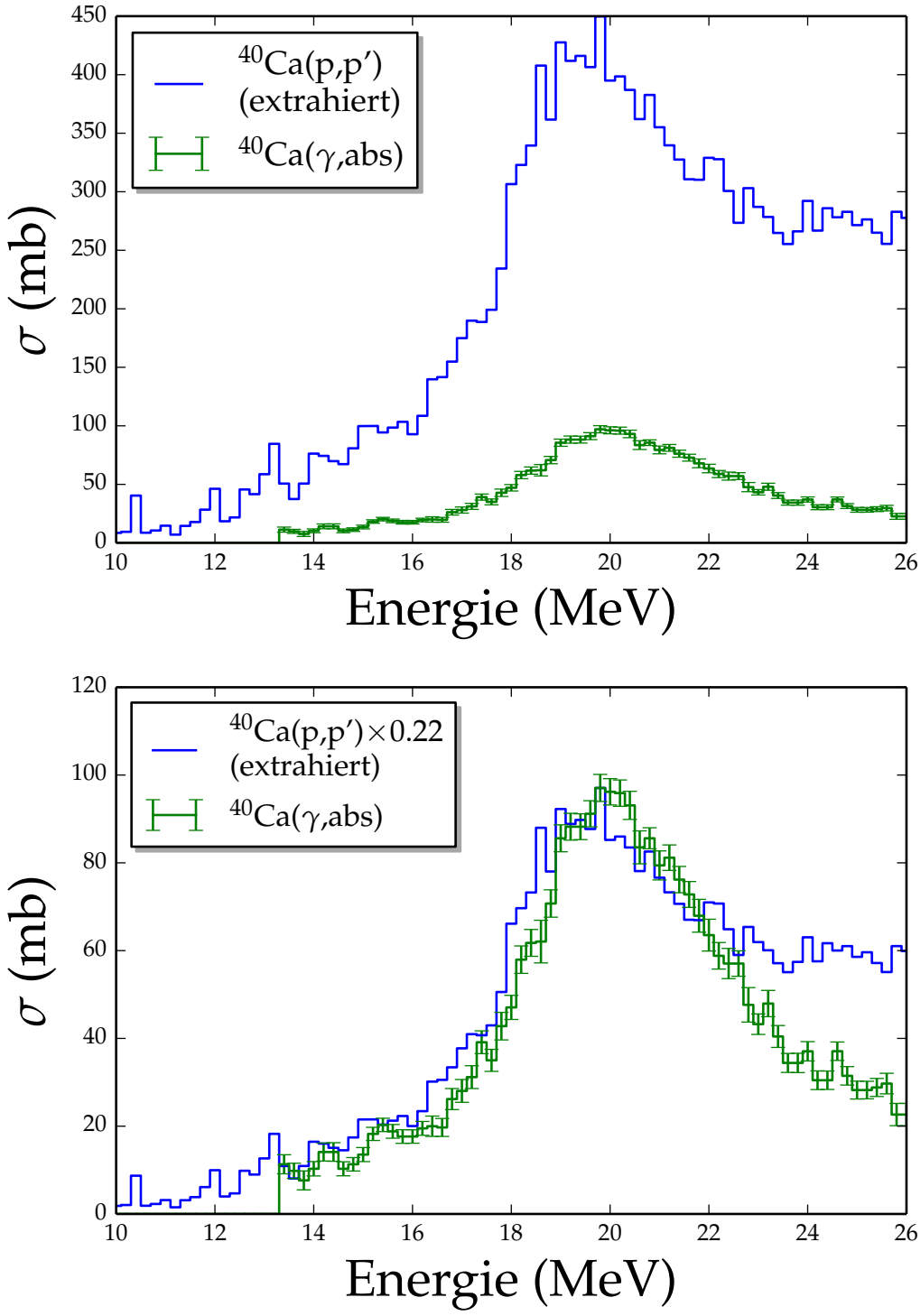


Abbildung B.25: Extrahierter bzw. experimenteller Photoabsorptionsquerschnitt für ^{40}Ca. Im unteren Bild wurde der extrahierte Photoabsorptionsquerschnitt mit einem Faktor von 0,22 multipliziert.

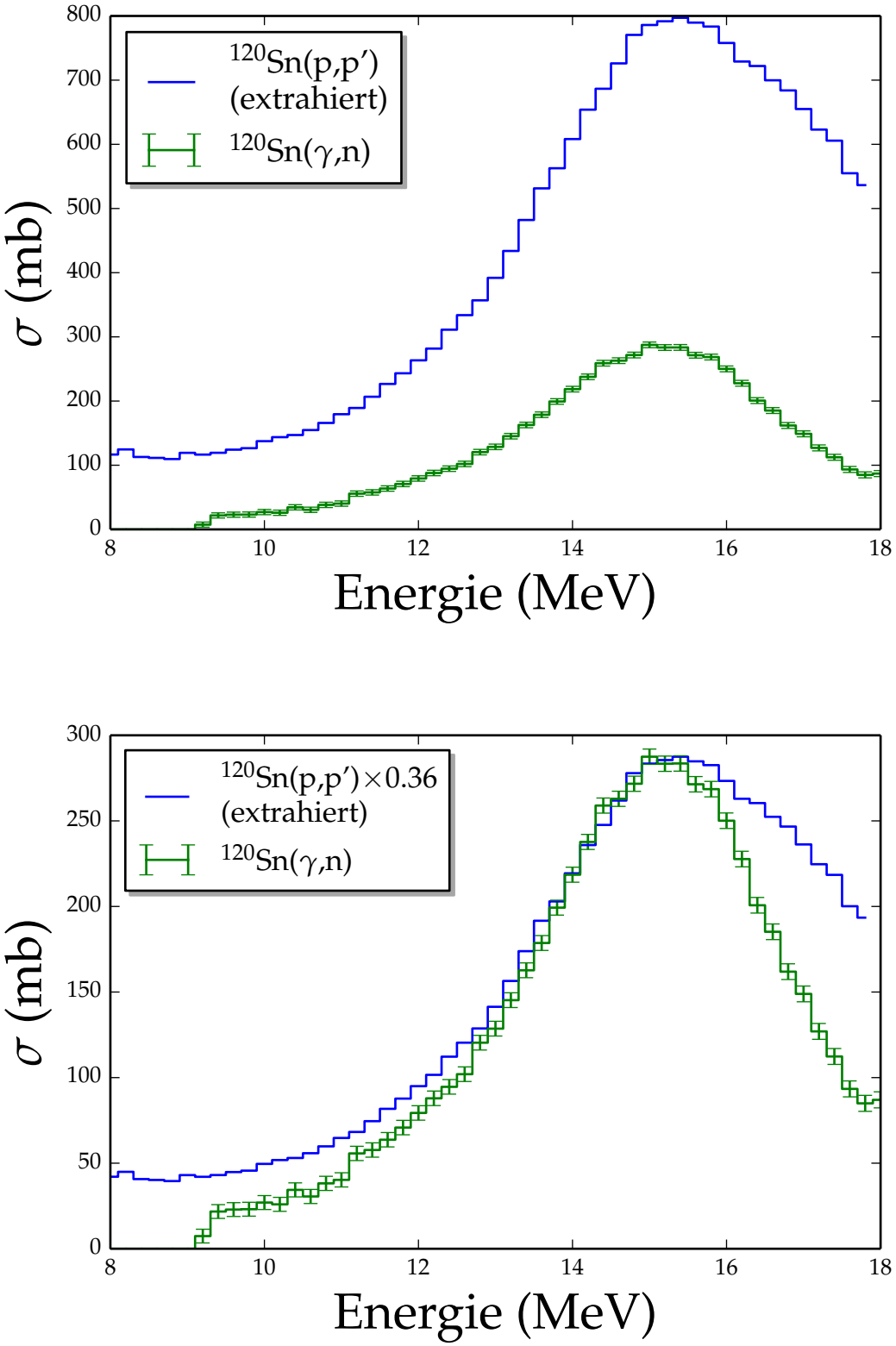
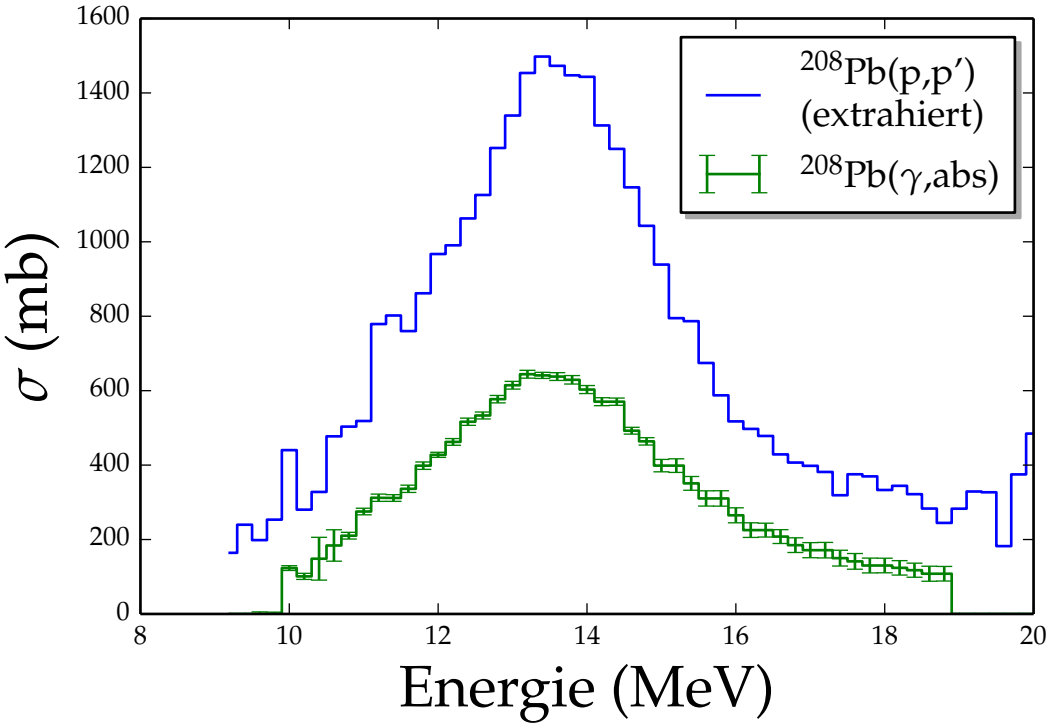


Abbildung B.26: Extrahierter bzw. experimenteller Photoabsorptionsquerschnitt für ^{120}Sn. Im unteren Bild wurde der extrahierte Photoabsorptionsquerschnitt mit einem Faktor von 0,36 multipliziert.



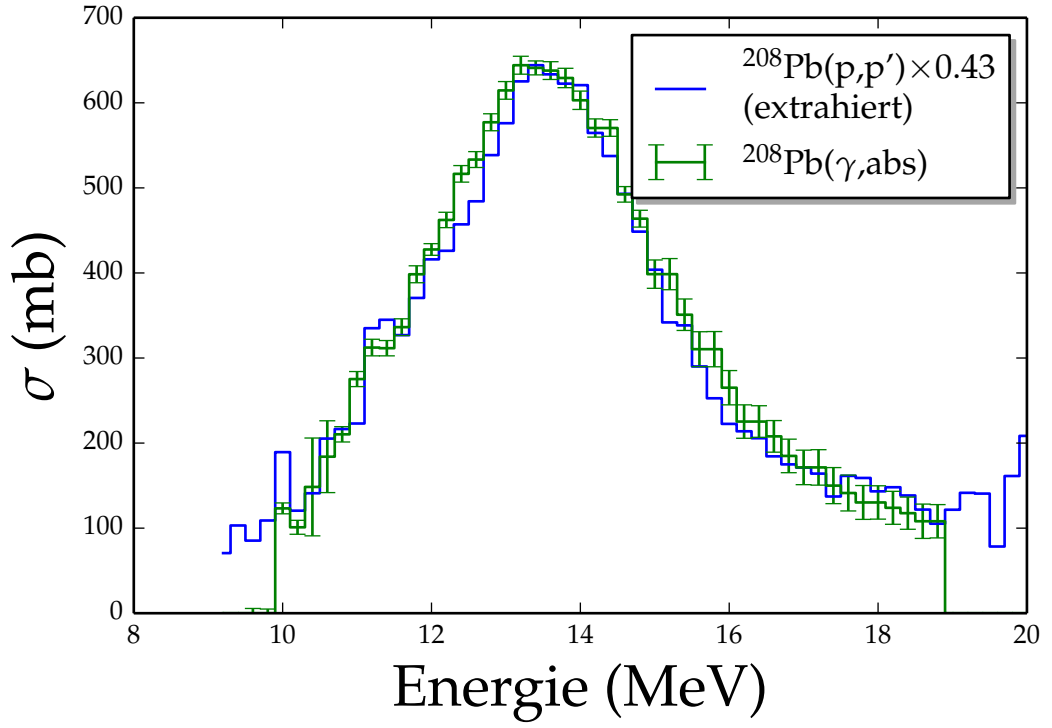


Abbildung B.27: Extrahierter bzw. experimenteller Photoabsorptionsquerschnitt für ^{208}Pb. Im unteren Bild wurde der extrahierte Photoabsorptionsquerschnitt mit einem Faktor von 0,43 multipliziert.

Literaturverzeichnis

[1] O. Burda, *Nature of Mixed-Symmetry 2^+ States in ^{94}Mo from High-Resolution Electron and Proton Scattering and Line Shape of the First Excited $1/2^+$ State in ^{9}Be*, Dissertation, D17, TU Darmstadt (2007).

[2] A. W. Lenhardt, *Entwicklung eines Si-Mikrostreifendetektors für das 169°-Spektrometer am S-DALINAC*, Dissertation, D17, TU Darmstadt (2004).

[3] Y. Fritzsche, *Aufbau und Inbetriebnahme einer Quelle polarisierter Elektronen am supraleitenden Darmstädter Elektronenlinearbeschleuniger S-DALINAC*, Dissertation, D17, TU Darmstadt (2011).

[4] O. Burda und A. Krugmann, *Performing Electron Scattering Experiments at the LINTOTT Spectrometer at the S-DALINAC* (2009).

[5] W. Schmidt, *Messung kinematischer Koinzidenzen an Protonen und Entwicklung eines Computerprogramms zur Magnetfeldsteuerung am QCLAM-Spektrometer*, Diplomarbeit, TU Darmstadt (1992).

[6] K.-D. Hummel, *Entwicklung, Aufbau und Inbetriebnahme eines Vieldrahtdriftkammer-Detektorsystems für das QCLAM-Spektrometer am supraleitenden Darmstädter Elektronenbeschleuniger S-DALINAC*, Dissertation, D17, TU Darmstadt (1992).

[7] S. Dieterich, *Kalibrierung des QCLAM Spektrometers am S-DALINAC*, Diplomarbeit, TU Darmstadt (1998).

[8] M. Knirsch, *Konzeption, Aufbau und Erprobung eines hochauflösenden QCLAM-Elektronenspektrometers mit großem Raumwinkel und hoher Impulsakzeptanz am Elektronenbeschleuniger S-DALINAC*, Dissertation, D17, TU Darmstadt (1991).

[9] S. M. Sze, *Semiconductor Devices: Physics and Technology*, John Wiley & Sons, New York (1985).

[10] W. R. Leo, *Techniques for Nuclear and Particle Physics Experiments*, Springer Verlag, Berlin (1994).

[11] G. F. Knoll, *Radiation Detection and Measurement*, John Wiley & Sons, New York (2010).

[12] H. Lauinger, *Aufbau und Inbetriebnahme des Silizium Detektor-Balls am QCLAM-Spektrometer*, Bachelorarbeit, TU Darmstadt (2012).

[13] N. Kurz und H. Essel, *Multi Branch System* (Aufruf am: 13.06.2014), http://www-win.gsi.de/daq.

[14] CES, *Ein-Platinen-Computer RIO3* (Aufruf am: 07.10.2013), http://http://www.ces.ch/board-products/vme-single-board-computers.

[15] C. Granja, P. Krist, D. Chvatil, J. Solc, S. Pospisil, J. Jakubek, und L. Opalka, Rad. Meas. **59** (2013) 245 .

[16] G. C. Baldwin und G. S. Klaiber, Phys. Rev. **71** (1947) 3.

[17] W. Bothe und W. Gentner, Zeitschrift für Physik **106** (1937) 236.

[18] R. Pitthan und T. Walcher, Phys. Lett. B **36** (1971) 563 .

[19] M. Lewis und F. Bertrand, Nucl. Phys. A **196** (1972) 337 .

[20] J. Speth, Editor, *Electric and Magnetic Giant Resonances in Nuclei*, World Scientific, Singapore (1998).

[21] H. Sakai, *Search for the β^+ IVSM Resonance via (^{3}H,^{3}He) Reactions at 900 MeV.* (2010), Workshopvortrag, unveröffentlicht.

[22] K. Amos, P. Dortmans, H. von Geramb, S. Karataglidis, und J. Raynal, Adv. Nucl. Phys. **25** (2000) 275.

[23] G. R. Satchler, *Introduction to Nuclear Reactions*, The Macmillan Press Ltd., London (1980).

[24] W. G. Love und M. A. Franey, Phys. Rev. C **24** (1981) 1073.

[25] B. Clark, R. Mercer, und P. Schwandt, Phys. Lett. B **122** (1983) 211 .

[26] J. R. Shepard, J. A. McNeil, und S. J. Wallace, Phys. Rev. Lett. **50** (1983) 1443.

[27] A. Rahbar, B. Aas, E. Bleszynski, M. Bleszynski, M. Haji-Saeid, G. J. Igo, F. Irom, G. Pauletta, A. T. M. Wang, J. B. McClelland, J. F. Amann, T. A. Carey, W. D. Cornelius, M. Barlett, G. W. Hoffmann, C. Glashausser, S. Nanda, und M. M. Gazzaly, Phys. Rev. Lett. **47** (1981) 1811.

[28] T. N. Taddeucci, J. Rapaport, D. E. Bainum, C. D. Goodman, C. C. Foster, C. Gaarde, J. Larsen, C. A. Goulding, D. J. Horen, T. Masterson, und E. Sugarbaker, Phys. Rev. C **25** (1982) 1094.

[29] T. Taddeucci, C. Goulding, T. Carey, R. Byrd, C. Goodman, C. Gaarde, J. Larsen, D. Horen, J. Rapaport, und E. Sugarbaker, Nucl. Phys. A **469** (1987) 125 .

[30] C. D. Goodman, C. A. Goulding, M. B. Greenfield, J. Rapaport, D. E. Bainum, C. C. Foster, W. G. Love, und F. Petrovich, Phys. Rev. Lett. **44** (1980) 1755.

[31] W. G. Love, K. Nakayama, und M. A. Franey, Phys. Rev. Lett. **59** (1987) 1401.

[32] A. Tamii, *Polarization transfer observables for proton inelastic Scattering from ^{12}C at zero degrees*, Dissertation, Kyoto University (1999).

[33] J. Raynal, *DWBA-Code*, NEA Data Service NEA1209/08.

[34] D. Martin, *Investigation of the reaction $^{144}Sm(p,p')$ under extreme forward angles*, Bachelorarbeit, TU Darmstadt (2011).

[35] A. Winther und K. Alder, Nucl. Phys. A **319** (1979) 518 .

[36] I. Poltoratska, *Complete dipole response in ^{208}Pb from high-resolution polarized proton scattering at 0°*, Dissertation, D 17, TU Darmstadt (2011).

[37] C. Bertulani und G. Baur, Nucl. Phys. A **442** (1985) 739 .

[38] C. A. Bertulani und G. Baur, Phys. Rep. **163** (1987) 299.

[39] N. Ryezayeva, T. Hartmann, Y. Kalmykov, H. Lenske, P. von Neumann-Cosel, V. Y. Ponomarev, A. Richter, A. Shevchenko, S. Volz, und J. Wambach, Phys. Rev. Lett. **89** (2002) 272502.

[40] C. Walz, H. Fujita, A. Krugmann, P. von Neumann-Cosel, N. Pietralla, V. Y. Ponomarev, A. Scheikh-Obeid, und J. Wambach, Phys. Rev. Lett. **106** (2011) 062501.

[41] B. Reitz, A. van den Berg, D. Frekers, F. Hofmann, M. de Huu, Y. Kalmykov, H. Lenske, P. von Neumann-Cosel, V. Ponomarev, S. Rakers, A. Richter, G. Schrieder, K. Schweda, J. Wambach, und H. Wörtche, Phys. Lett. B **532** (2002) 179 .

[42] I. Poltoratska, P. von Neumann-Cosel, A. Tamii, T. Adachi, C. A. Bertulani, J. Carter, M. Dozono, H. Fujita, K. Fujita, Y. Fujita, K. Hatanaka, M. Itoh, T. Kawabata, Y. Kalmykov, A. M. Krumbholz, E. Litvinova, H. Matsubara, K. Nakanishi, R. Neveling, H. Okamura, H. J. Ong, B. Özel-Tashenov, V. Y. Ponomarev, A. Richter, B. Rubio, H. Sakaguchi, Y. Sakemi, Y. Sasamoto, Y. Shimbara, Y. Shimizu, F. D. Smit, T. Suzuki, Y. Tameshige, J. Wambach, M. Yosoi, und J. Zenihiro, Phys. Rev. C **85** (2012) 041304.

[43] V. G. Soloviev, *Theory of Atomic Nuclei: Quasiparticles and Phonons*, IOP Publishing Ltd, Bristol (1992).

[44] C. Bertulani und V. Ponomarev, Phys. Rep. **321** (1999) 139 .

[45] J. Bardeen, L. N. Cooper, und J. R. Schrieffer, Phys. Rev. **108** (1957) 1175.

[46] M. Tanaka *et al.*, *Annual Report*, Technical report, RCNP (1991), 215.

[47] R. Geller, *Electron Cyclotron Resonance Ion Sources and ECR Plasmas*, IOP Publishing Ltd, Bristol (1996).

[48] K. Hatanaka, K. Takahisa, H. Tamura, M. Sato, und I. Miura, Nucl. Instr. Meth. A **384** (1997) 575 .

[49] Y. Masuda, T. Kitagaki, K. Hatanaka, M. Higuchi, S. Ishimoto, Y. Kiyanagi, K. Morimoto, S. Muto, und M. Yoshimura, Phys. Rev. Lett. **89** (2002) 284801.

[50] T. Shimoda, H. Miyatake, und S. Morinobu, Nucl. Instr. Meth. B **70** (1992) 320 .

[51] H. Sakai, H. Okamura, H. Otsu, T. Wakasa, S. Ishida, N. Sakamoto, T. Uesaka, Y. Satou, S. Fujita, und K. Hatanaka, Nucl. Instr. Meth. A **369** (1996) 120 .

[52] M. Fujiwara, H. Akimune, I. Daito, H. Fujimura, Y. Fujita, K. Hatanaka, H. Ikegami, I. Katayama, K. Nagayama, N. Matsuoka, S. Morinobu, T. Noro, M. Yoshimura, K. Sakaguchi, Y. Sakemi, A. Tamii, und M. Yosoi, Nucl. Instr. Meth. A **422** (1999) 484.

[53] N. Matsuoka, K. Hatanaka, S. Morinobu, T. Noro, A. Okihana, und K. Sagara, *Annual Report*, Technical report, RCNP (1991), 186.

[54] H. Matsubara, *Study of M1 Quenching in ^{28}Si by a (p,p') Measurement at zero-degrees*, Masterarbeit, Osaka University (2006).

[55] H. Matsubara, *Isoscalar and isovector spin-M1 transitions from the even-region $N = Z$ nuclei across the sd-shell region*, Dissertation, Osaka University (2010).

[56] J. Birkhan, Dissertation, D17, TU Darmstadt (2014), in Vorbereitung.

[57] D. Martin, *Gamma Strength Function of $^{96}Mo(p,p')$: A Test of the Axel-Brink Hypothesis*, Masterarbeit, TU Darmstadt (2013).

[58] A. M. Krumbholz, Dissertation, D17, TU Darmstadt (2014), in Vorbereitung.

[59] A. Krugmann, Dissertation, D17, TU Darmstadt (2014), in Vorbereitung.

[60] *EXFOR-Datenbank* (Aufruf am: 07.10.2013), `www-nds.iaea.org/exfor/exfor.htm`.

[61] J. Ahrens, H. Borchert, K. Czock, H. Eppler, H. Gimm, H. Gundrum, M. Kröning, P. Riehn, G. S. Ram, A. Zieger, und B. Ziegler, Nucl. Phys. A **251** (1975) 479 .

[62] G. J. O'Keefe, M. N. Thompson, Y. I. Assafiri, R. E. Pywell, und K. Shoda, Nucl. Phys. A **227** (1974) 427.

[63] H. Beil, R. Bergère, P. Carlos, A. Leprêtre, A. de Miniac, und A. Veyssière, Nucl. Phys. A **469** (1987) 239.

[64] A. Leprêtre, H. Beil, R. Bergère, P. Carlos, A. D. Miniac, A. Veyssière, und K. Kernbach, Nucl. Phys. A **219** (1974) 39 .

[65] P. Carlos, H. Beil, R. Bergère, A. Leprêtre, A. D. Miniac, und A. Veyssière, Nucl. Phys. A **225** (1974) 171 .

[66] K. Schelhaas, J. Henneberg, M. Sanzone-Arenhövel, N. Wieloch-Laufenberg, U. Zurmühl, B. Ziegler, M. Schumacher, und F. Wolf, Nucl. Phys. A **489** (1988) 189 .

[67] K. Alder, A. Bohr, T. Huus, B. Mottelson, und A. Winther, Rev. Mod. Phys. **28** (1956) 432.

[68] C. Bertulani und A. Nathan, Nucl. Phys. A **554** (1993) 158 .

[69] V. Y. Ponomarev, *Mündliche Mitteilung* (2014).

[70] T. Chapuran, R. Vodhanel, und M. K. Brussel, Phys. Rev. C **22** (1980) 1420.

[71] CAEN, *TDC-Manual* (Aufruf am: 13.06.2014), `http://www.caen.it/csite/CaenProd.jsp?parent=11&idmod=35`.

[72] CAEN, *ADC-Manual* (Aufruf am: 13.06.2014), `http://www.caen.it/csite/CaenProd.jsp?parent=11&idmod=37`.

Danksagung

Zuallererst möchte ich mich bei Herrn Prof. Dr. Peter von Neumann-Cosel bedanken, der mir die Gelegenheit gab an einem so interessanten Thema zu arbeiten und mir stets für Fragen und Diskussion zur Verfügung stand.

Ich bedanke mich bei meinem Betreuer Dipl.-Phys. Jonny Birkhan, der mir immer offen für Fragen stand und mir sowohl bei der Datenanalyse als auch beim Verständnis der Elektronik sehr weiter geholfen hat.

Ich Danke auch Dr. Vladimir Yu. Ponomarev für die QPM-Rechnungen und eine sehr gute Einführung zum *DWBA*-Code.

Insbesondere danke ich auch Andreas Krugmann, M.Sc., Anna Maria Krumbholz, M.Sc. und Dirk Martin, M.Sc. für die vielen anregenden Diskussionen rund um die Protonenstreuung.

Ich möchte mich auch bei Dipl.-Phys. Simela Aslanidou bedanken, durch die ich sehr viel über das QCLAM-Spektrometer gelernt habe.

Ein großer Dank gebührt auch den Kollegen aus Japan, insbesondere Prof. Dr. Atsushi Tamii und Dr. Hiroaki Matsubara, die mir ermöglichten die Protonenspektren auszuwerten.

Ich danke allen meinen Bürokollegen, die mich sowohl unterstützt, als auch zu einem angenehmen Arbeitsklima beigetragen haben.

Schließlich möchte ich mich bei meiner Familie bedanken, ohne deren Unterstützung diese Arbeit nicht möglich gewesen wäre.

Erklärung zur Master-Thesis

Hiermit versichere ich, die vorliegende Master-Thesis ohne Hilfe Dritter nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die aus Quellen entnommen wurden, sind als solche kenntlich gemacht. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Darmstadt, den 6. Juli 2014

(Sergej Bassauer)