Ruprecht-Karls-Universität Heidelberg
Fakultät für Physik und Astronomie
Max - Planck - Institut für Kernphysik

HD-KIP 00-03
HD-ASIC-11-0100

Der ATLAS Level-1 Trigger Präprozessor ASIC

Diplomarbeit

von

Dan Husmann

ASIC-Labor http://wwwasic.kip.uni-heidelberg.de/Asic/
Schröderstraße 90 D-69120 Heidelberg

Fakultät für Physik und Astronomie

Ruprecht-Karls-Universität Heidelberg

Diplomarbeit
im Studiengang Physik

vorgelegt von
Dan Husmann
aus Heidelberg

Januar 2000

Der ATLAS Level-1 Trigger Präprozessor ASIC

Die Diplomarbeit wurde ausgeführt von Dan Husmann am
Kirchhoff-Institut für Physik der Universität Heidelberg
unter der Betreuung von
Herrn Prof. Dr. K. Meier

Inhalt

Diese Arbeit beschreibt zwei Algorithmen, die das Pulsmaximum von ATLAS Level-1 Kalorimeter Signalen dem zugehörigen Bunch-Crossing zuordnen. Der eine Algorithmus ist dafür ausgelegt nicht saturierte Signale zu verarbeiten, während der anderere für saturierte Signale vorgesehen ist. Es soll ein möglichst großer Überlappbereich geschaffen werden, in dem beide Algorithmen mit Erfolg arbeiten. Die Algorithmen sind in digitaler Logik implementiert und arbeiten auf den digitalisierten Werten der Trigger-Tower Elektronik. Um die Effizienz der Algorithmen analysieren zu können, wurde ein Simulationsprogramm entwickelt, das die Algorithmen für saturierte und nicht saturierte Pulse sowohl separat, als auch in ihrem Zusammenspiel simulieren kann. Mit Hilfe dieses Programms wurden verschiedene Parameter für die Algorithmen getestet. Es konnte gezeigt werden, daß die Algorithmen bei der richtigen Wahl der Parameter mit großer Effizienz arbeiten.

Die getesteten Algorithmen wurden mit Hilfe der Hardwarebeschreibungssprache Verilog auf dem sogenannten Präprozessor-ASIC implementiert. Da die Parameter im späteren Betrieb den wahren Pulsformen angepaßt werden müssen, wurden die Schwellen und Koeffizienten der Algorithmen nicht fest, sondern programmierbar implementiert. Erste Tests des PPrAsics auf Code-Ebene wurden erfolgreich durchgeführt.

Abstract

This thesis describes studies to analyze the efficiency of two algorithms that identify the corresponding bunch-crossing in time for ATLAS level-1 calorimeter signals and their implementation on an ASIC. One algorithm is optimized for saturated and the other one for non-saturated signals. The aim is to get a large energy range where both algorithms work reliable. The algorithms are implemented in digital logic and therefore the signals from the trigger-tower electronic have to be digitalized. To test the efficiency of the algorithms software simulation tools were developed. They is able to simulate the algorithms for saturated and non-saturated signals separat or combined. With these simulation tools the algorithms where tested in different configurations of their parameters. It could be shown that for the right choice of parameters a high efficiency can be reached.

The tested algorithms were implemented on the Pre-Processor ASIC using the hardware-description language Verilog. Due to the fact that the right parameters are depending on the final pulse-shape, the implementation of the algorithms is done by using programmable registers. First tests of the PPrAsic based on the Verilog code were done successfully.

Inhaltsverzeichnis

Einleitung

Die immer weiterführende Suche nach neuen genaueren Theorien, die die Zusammensetzung der Materie aus elementaren Bausteinen, sowie die Kräfte, die zwischen diesen wirken, erklären sollen, verlangt nach immer größeren Teilchenbeschleunigern. Diese müssen in der Lage sein, Streuexperimente bei immer höheren Energien durchzuführen. Einer der größten geplanten Teilchenbeschleuniger ist der *Large Hadron Collider (LHC)*, der am europäischen Kernforschungszentrum CERN in Genf in Betrieb genommen werden soll. Er soll es unter anderem möglich machen, das letzte bisher noch nicht entdeckte Teilchen des Standardmodells, das sogenannte *Higgs-Boson*, nachzuweisen.

Die interessanten physikalischen Prozesse, die am *LHC* untersucht werden sollen, sind sehr selten. Will man dennoch physikalisch verwertbare Aussagen erhalten, so müssen sehr viele Reaktionen stattfinden. Aus dieser Menge von Reaktionen müssen dann diejenigen herausgefiltert werden, die die physikalisch interessanten Prozesse beinhalten. Diese Aufgabe übernimmt das *Trigger- System*, das nach Prozessen sucht, die als interessant bestimmt wurden.

Die Anforderungen an die Geschwindigkeit eines solches *Trigger-System* sind extrem hoch, da die gesamte Datenmenge, die während des Entscheidungsprozesses anfällt, zwischengespeichert werden muß. Aus diesem Grund ist es notwendig, schnelle und kompakte Elektronik zu verwenden. Einen Teil dieser Elektronik stellt das Präprozessor-System dar. Eine wichtige Aufgabe dieses Systems ist die Zuordnung der Kalorimetersignale zu denjenigen Teilchenpaket-Kollisionen, aus denen sie entstanden sind (*Bunch-Crossing-Identification (BCID)*. In dieser Diplomarbeit wurden Algorithmen, die diese Aufgabe übernehmen, untersucht. Danach wurden sie auf einem ASIC, der im finalen System eingesetzt werden soll, implementiert.

Die vorliegende Arbeit gliedert sich in ein einleitendes Kapitel, das sich mit der Physik beschäftigt, die am LHC gemacht werden soll und das ATLAS Experiment vorstellt. Danach wird in Kapitel 2 ein kurzer Einblick in die Struktur des ATLAS Trigger-Systems, das sich in drei Stufen gliedert, gegeben. Ein wichtiger Bestandteil der ersten Stufe, das sogenannte *Präprozessor-System*, wird in Kapitel 3 erläutert. Das Kapitel 4 beschäftigt sich mit den Algorithmen des BCID. Danach geht Kapitel 5 auf das Kernstück des Präprozessor-Systems ein, den sogenannten Präprozessor-ASIC, und in Kapitel 6 wird die seine Implementation mit Hilfe der HardwareBeschreibungssprache Verilog dargelegt. Kapitel 7 beschreibt die Simulationen, die gemacht wurden, um die Effizienz der verschiedenen BCID-Algorithmen zu testen und die Richtigkeit der Verilog-Implementation derselben nachzuweisen.

Kapitel 1

Das ATLAS Experiment

1.1 Der Large Hadron Collider am CERN

Der LHC (*Large Hadron Collider*) ist ein Teilchenbeschleuniger, in dem hochenergetische Proton-Proton-Stöße stattfinden sollen. Er wird im Jahre 2005 am Europäischen Kernforschungszentrum CERN in Genf in Betrieb genommen. Um schon vorhandene Infrastruktur nutzen zu können, wird der LHC in den bereits existierenden Tunnel des LEP (*Large Electron Positron Collider*) eingebaut.

In bisherigen LEP-Experimenten wurden Elektronen und Positronen auf einer Kreisbahn im gleichen Magnetfeld gegenläufig beschleunigt. Da die Masse von Elektonen und Positronen nur bei 511 keV/c^2 liegt, wird die maximal erreichbare kinetische Energie der Teilchen begrenzt durch die *Synchrotron-Strahlung*. Diese hängt in der vierten Potenz von der Energie und der Masse der Teilchen ab. Der Energieverlust pro Umlauf für hochrelativistische Teilchen ist in folgender Formel [1] dargestellt.

$$-\Delta E = \frac{4\pi\alpha\hbar c}{3R}\beta^3\frac{E^4}{m^4c^8} \tag{1.1}$$

Um die hierdurch entstehenden Verluste zu reduzieren und so zu höheren Energien vordringen zu können, ist man gezwungen, schwerere Teilchen zu verwenden, denn der Energieverlust der Teilchen ist umgekehrt proportional zur vierten Potenz ihrer Masse. Am LHC werden daher Protonen verwandt,die eine Masse von 938 MeV/c^2 besitzen. Bei gleicher Strahlenergie verlieren diese 10^{13} mal weniger Energie als Elektronen.

Die Protonen werden bis auf eine kinetische Energie von je 7 TeV beschleunigt, bevor sie an ausgewählten Punkten zur Kollision gebracht werden. Somit beträgt die Schwerpunktsenergie, die an diesen Wechselwirkungspunkten zur Verfügung steht, $\sqrt{s} = 14$ TeV. Da der vorhandene LEP-Tunnel benutzt wird, ist der Umfang der Kreisbahn auf 26,7 km festgelegt. Um die Teilchen bei so hohen Energien auf der Kreisbahn zu halten, kann man nicht die bisherigen LEP-Magnetfelder benutzen. Es müssen sehr große Dipolfelder erzeugt werden. Mit Hilfe supraleitender Dipolmagnete, die in einer Umgebung von suprafluidem Helium bei 1,9 K arbeiten, wird ein Dipolfeld mit einer Stärke von 8,3 T erzeugt. Im Unterschied zu Elektron-Positron-Beschleunigern ist die Strahlenergie durch das erreichbare magnetische Feld begrenzt.

Die Protonen am LHC sind in Teilchenpakete aufgeteilt, die in zwei Teilchenstrahlen gegenläufiger Ausrichtung zirkulieren. Jedes Teilchenpaket enthält etwa 10^{11} Teilchen, und über den gesamten Ring sind 2961 solcher Pakete verteilt. Alle 25 ns werden zwei Teilchenpakete im Detektor zur Kollision gebracht. Diesen Vorgang nennt man *Bunch-Crossing (BC)*.

Die Reaktionsrate R, die man am Wechselwirkungspunkt erhält, setzt sich aus zwei Anteilen zusammen. Der eine ist die sogenannte *Luminosität* L. Sie wird vom Beschleuniger bestimmt. Der andere Anteil, der *Wirkungsquerschnitt* σ, ist abhängig von der tatsächlich stattgefundenen Reaktion. Es gilt

$$R = \sigma L. \tag{1.2}$$

Die Luminosität pro Kollision hängt von den Teilchenzahlen n_1, n_2, der Fläche, der zur Kollision gebrachten Teilchenpakete A, und der Zahl der umlaufenden Teilchenpakete pro Zeiteinheit f ab. Hierbei hängt f im Falle eines Ringbeschleunigers von der Zahl der zur Kollision gebrachten Teilchenpakete j, der Geschwindigkeit mit der man die Teilchen umlaufen läßt v und dem Umfang der Kreisbahn U ab ($f = j\frac{v}{U}$). Es gilt [2]

$$L = n_1 n_2 \frac{f}{A} \tag{1.3}$$

Die interessanten Teilchen am LHC werden in inelastischen Streuprozessen erzeugt. Da die zu analysierenden Prozesse sehr selten sind, ist ihr Anteil σ am totalen Wirkungsquerschnitt σ_{tot} ($\sigma_{tot} = \sigma_{el} + \sigma_{inel}$) sehr gering. Er kann bei sehr seltenen Prozessen bis auf einen Bruchteil von $\sigma/\sigma_{tot} \leq 10^{-11}$ absinken. Um genaue Messungen durchführen zu können, benötigt man eine hohe Reaktionsrate, was bei kleinen Wirkungsquerschnitten nur durch eine sehr hohe Luminosität zu erreichen ist. In den ersten drei Jahren soll der LHC mit einer maximalen Luminosität von $10^{33}\mathrm{cm}^{-2}\mathrm{s}^{-1}$ arbeiten, danach soll sie auf $10^{34}\mathrm{cm}^{-2}\mathrm{s}^{-1}$ gesteigert werden.

Um statistische Aussagen über seltene Prozesse mit ausreichender Genauigkeit treffen zu können, ist es notwendig, eine große Anzahl von in einem Zeitintervall T insgesamt stattgefundenen Reaktionen zu erreichen. Ein Maß hierfür ist die integrierte Luminosität, die durch

$$\int_0^T L\,dt \tag{1.4}$$

definiert ist.

1.2 Physik am LHC

Die Grundlage heutiger Hochenergiephysik ist das *Standardmodell* [3]. Es beschreibt die Zusammensetzung der Materie aus elementaren Teilchen und die zwischen diesen wirkenden Kräfte. Auch wenn dieses Modell heutige Teilchenexperimente sehr detailiert beschreiben kann, so bleiben doch einige Dinge ungeklärt. Mit Hilfe der hohen Energien, die am LHC erreicht werden, wird es möglich sein, das Standardmodell in Bereichen zu testen, in denen es bisher noch keiner Prüfung unterzogen wurde. Dies könnte helfen, zu einer Erweiterung des Standardmodells zu gelangen, die es kompletter und theoretisch einheitlicher macht.

Es gibt laut Standardmodell zwei Arten von Elementarteilchen, die *Quarks* und die *Leptonen*. Diese sind ihrerseits in drei *Familien* unterteilt. Jede Familie besteht aus zwei Quarks, einem Lepton und seinem Neutrino (Tabelle 1.1).

	Familien		
Quarks	u (up) d (down)	c (charme) s (strange)	t (top) b (bottom)
Leptonen	e^- ν_e	μ^- ν_μ	τ^- ν_τ

Tabelle 1.1: Die elementaren Teilchen des Standardmodells

Die vier bekannten elementaren Kräfte sind die elektromagnetische, die schwache, die starke Wechselwirkung und die Gravitation. Die Gravitation ist erheblich schwächer als die anderen drei Kräfte und ist nicht Teil des Standardmodells. Die Wechselwirkungen, die auf diesen Kräften beruhen, können als Austausch spezieller Teilchen, der sogenannten Bosonen, erklärt werden (siehe Tabelle 1.2).

Kraft	**Bosonen**	**Symbol**	**Stärke**
Schwach	intermediäre Vektor-Bosonen	$W^{\pm}, Z^0$	$\alpha_{weak} = 0.03$
Electromagnetisch	Photonen	γ	$\alpha_{em} = 1/137$
Stark	Gluonen	g	$\alpha_s \approx 0.12..1$
Gravitation	Graviton		—

Tabelle 1.2: Elementare Kräfte und ihre Austauschteilchen

Die Theorie der Vereinheitlichung von elektromagnetischer und der schwachen Kraft in die elektroschwache Kraft erfordert ein neutrales, skalares Teilchen. Dieses Teilchen, das sogenannte *Higgs-Boson*, ist das einzige Teilchen, dessen Nachweis im Standardmodell noch fehlt. Die Theorie sagt das Higgs-Boson als Produkt des Mechanismus der *spontanen Symmetriebrechung* voraus. Es ist demnach Ursache für das Vorhandensein der Massen der Vektorbosonen $W^{\pm}$ und Z^0. Dies macht die Suche nach dem Higgs-Boson zu einer der wichtigsten Aufgaben der heutigen Hochenergiephysik.

Die theoretisch vorhergesagte Obergrenze für die Higgs-Boson-Masse liegt in der Grössenordnung von 1 TeV [4]. Diese Grenze ist nicht als absolute Grenze zu verstehen, sondern sie signalisiert einen Wert, ab dem es nicht mehr sinnvoll ist, von einem elementaren Higgs-Boson im Rahmen des Standardmodells zu sprechen. Dies läßt sich an der Breite des Higgs-Bosons verdeutlichen. Da diese proportional zur Masse ist, kann sie für ein 1 TeV Higgs-Boson schon in Bereich von 500 GeV liegen. Die experimentelle Untergrenze liegt am Ende der von LEP2 erreichten Energie und somit bei $m_H > 105$ GeV.

Ein weiteres wichtiges Ziel von LHC besteht darin, nach neuer Physik jenseits des Standardmodells zu suchen. Theorien, die die Vereinheitlichung der Elementarkräfte möglich machen, sollen gefunden und untersucht werden. Die sogenannte *Supersymmetrie (SUSY)* fordert für jedes Boson ein dazugehöriges Fermion und umgekehrt. Die Theorie wird dadurch symmetrisch bezüglich sowohl des Fermionen-, als auch des Bosonenspektrums, und ist daher in der Lage, zu einer Vereinheitlichung der Kräfte zu gelangen. Das supersymmetrische Spektrum ist reich an Teilchen. Es enthält Spin-0 Teilchen, wie *Squarks* und *Sleptonen*, und Spin-1/2 Teilchen, wie *Gluino, Charginos* und *Neutralinos*. Das leichteste Teilchen aus dieser Auswahl, das sogenannte *LSP*[1], wird als stabil und schwach wechselwirkend angenommen. Es sollte durch Reaktionen gekennzeichnet sein, bei denen ein großer Teil der transversalen Energie nicht im Detektor deponiert wird.

Es wird erwartet, daß aus der Menge der für die Supersymmetrie geforderten Teilchen einige in einem Massenbereich liegen, der durch den LHC zugänglich gemacht werden könnte.

Der LHC gibt auch die Möglichkeit, bereits nachgewiesene Phänomene genauer zu analysieren. So könnte z.B. das *Top-Quark* intensiver untersucht werden. Außerdem könnte man mit Hilfe der sogennanten *b-Physik*, der Physik der Mesonen, die ein *Bottom-Quark* enthalten, neue Erkenntnisse über die Struktur der *CP-Verletzung* gewinnen.

1.2.1 Experimente am LHC

Um die bisher beschriebene Physik betreiben zu können, sind am LHC vier große Experimente geplant (Abbildung 1.1).

→ ATLAS (*A Toroidal Lhc ApparatuS*)
→ CMS (*Compact Muon Solenoid*)
→ ALICE (*A Large Ion Collider Experiment*)
→ LHC-b (*Large Hadron Collider 'bottom-quarks'*)

ATLAS und CMS sind beides Universaldetektoren, deren Aufgabenbereiche die Suche nach dem Higgs-Boson und die Durchführung schon beschriebener *neuer Physik* sind. Sie unterscheiden sich in ihrem apparativen Aufbau. So unterscheidet sich z.B. der Aufbau der Kalorimeter. ATLAS verwendet für das elektromagnetische Kalorimeter flüssiges Argon, wogegen es bei CMS aus Kristallen aufgebaut ist. Außerdem unterscheiden sich die verwendeten Magnetfelder und die Trigger-Konzepte.

Mit LHC-b versucht man anhand der Analyse von Teilchen, die bei Proton-Proton Streuprozessen entstehen, einen tieferen Einblick in den Mechanismus der *CP-Verletzung* zu bekommen. Dafür eignen sich besonders Teilchen, die ein *bottom-quark* enthalten.

ALICE analysiert Schwerionenkollisionen mit einer Schwerpunktsenergie,die jenseits von 1000 TeV liegt. Der Detektor wird gebaut, um das große physikalische Potential von Nukleon-Nukleon Stößen bei hohen Energien ausnutzen zu können. Er soll die Physik der starken Wechselwirkung bei extrem hohen Energiedichten untersuchen. Es wird erwartet, daß bei so hohen Energien ein neuer Materiezustand erreicht wird, das sogenannte *Quark-Gluon Plasma*. Für das Verständnis von *chiraler Symmetrieerhaltung*[2] und von

[1] engl. lightest supersymmetric particle
[2] Erhaltung der rechtshändigen bzw. linkshändigen Lösung der Dirac-Gleichung.

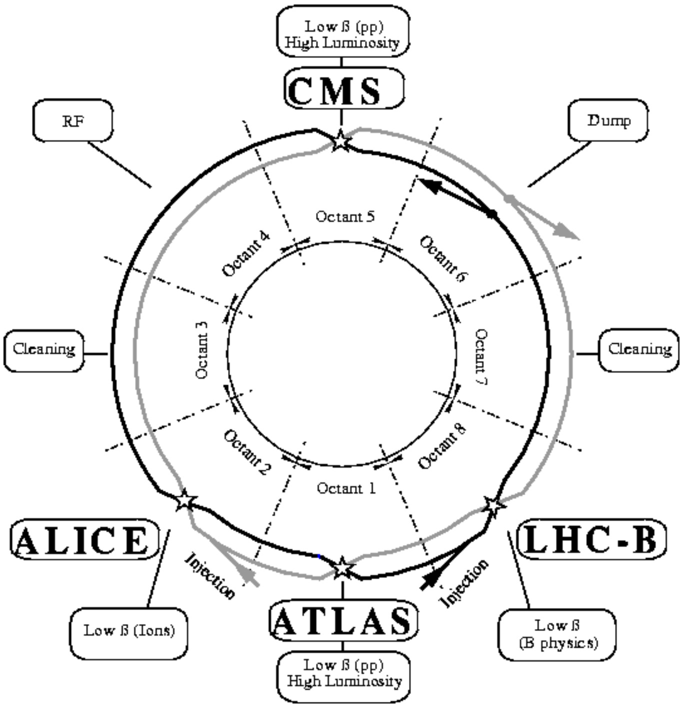

Abbildung 1.1: Standorte der LHC-Experimente [5]

confinement[3] ist die Existenz eines solchen Materiezustandes von entscheidender Bedeutung.

1.2.2 Proton-Proton Streuprozesse

Die Grundlage des ATLAS- sowie des CMS-Experiments bilden Proton-Proton-Streuprozesse, die mit einer Schwerpunktsenergie von bis zu 14 TeV durchgeführt werden. Diese Stöße finden an speziellen Kreuzungspunkten statt, an denen sich die Experimente befinden. Sie sind durch die starke Wechselwirkung bestimmt. Zum physikalischen Verständnis muß daher die *Quantenchromodynamik (QCD)* herangezogen werden.

1.2.3 Quantenchromodynamik

Die Quantenchromodynamik beschreibt die Wechselwirkung zwischen Quarks und Gluonen und erklärt ihr Zusammenwirken bei der Entstehung von *Hadronen*. Sie führt eine neue ladungsartige Eigenschaft für Quarks und Gluonen ein, die sogenannte *Farbe*. Durch den Austausch von Gluonen können Quarks miteinander in Wechselwirkung treten.

Die Tatsache, daß Gluonen selbst Farbladungen tragen, ermöglicht es ihnen, auch untereinander in Wechselwirkung zu treten, was das Verhalten der *starken Kraft* von dem der *elektromagnetischen* unterscheidet. Diese mögliche Wechselwirkung der Gluonen

[3] engl. confinement: die immense Stärke der Quark-Quark Kräfte bei großem Abstand verhindert das Auftreten von freien Quarks

untereinander führt nämlich dazu, daß die Stärke der *starken Kraft* mit zunehmendem Abstand wächst. Die potentielle Energie des *Farbfeldes* wächst bis zu dem Punkt, an dem sie ausreicht, um ein farbloses Farb-Singlett zu erzeugen. Diesen Vorgang bezeichnet man als *Hadronisierung* des Quarks [1]. Nach außen hin bilden sich also immer nur farbneutrale Konfigurationen von Quarks.

1.2.4 Interessante Prozesse der Proton-Proton Streuung

Die Prozesse, die durch die Proton-Proton Streuung ausgelöst werden können, sind von sehr großer Vielfältigkeit. Welche nun im konkreten Fall tatsächlich stattfinden, hängt von drei Faktoren ab, der Energie der Protonen, dem übertragenen Impuls und dem Wirkungsquerschnitt der einzelnen Reaktionen. Die Wechselwirkung kann elastisch oder inelastisch sein.

Bei elastischer Streuung bleibt die Struktur der Protonen unverändert. Außerdem wird die Ablenkung der Flugbahnen der Protonen im allgemeinen nur sehr gering sein. Handelt es sich dagegen um einen inelastischen Streuvorgang, so entstehen durch Hadronisierung zusätzliche Teilchen.

Betrachtet man nun Prozesse mit einem sehr hohen Impulsübertrag, so erkennt man quasi eine Wechselwirkung einzelner Konstituenten innerhalb des Protons, sogenannter Partonen. Die Parton-Parton Wechselwirkungen hadronisieren und erzeugen Jets, welche in einem großen Winkel zur Strahlrichtung aus dem Wechselwirkungspunkt fliegen. Die Flugrichtung dieser Jets hängt stark von dem Impulsübertrag der beteiligten Partonen ab. Stoßen zwei Partonen sehr hart aufeinander, d.h. mit einem sehr großen Impulsübertrag, so erzeugen sie Jets mit zueinander gegensätzlichem Impuls.

Der totale Wirkungsquerschnitt der Proton-Proton Kollisionen am LHC wird durch Prozesse mit geringem Impulsübertrag dominiert, welche für die hier verfolgte Fragestellung physikalisch von sehr geringem Interesse sind. Die Prozesse in denen Jets mit hohem transversalem Impuls erzeugt werden, oder diejenigen, bei denen schwere Quarks beteiligt sind, sind sehr selten. Auch Prozesse bei denen bislang nicht nachgewiesene Teilchen erzeugt werden könnten, wie z.B. das Higgs-Boson oder angeregte Vektorbosonen, werden von der Theorie als sehr selten vorhergesagt. Diese Tatsachen erzwingen eine hohe *Luminosität* und ein Triggerkonzept, das in der Lage ist, die wenigen interessanten Prozesse aus der Menge der Gesamtprozesse herauszufiltern. In Abbildung 1.2 wird der erwartete Wirkungsquerschnitt für einige interessante Prozesse bei voller LHC-Luminosität gezeigt [6].

Bei voller LHC-Energie beträgt der totale Wirkungsquerschnitt für die Proton-Proton Streuung [4] etwa 100 mb. Das bedeutet, daß pro Sekunde etwa 10^9 Proton-Proton-Stöße stattfinden. Da die Teilchenpakete 40.000.000-mal in der Sekunde aufeinandertreffen, ergibt sich eine Rate von durchschnittlich 25 Ereignissen pro Bunch-Crossing. Betrachtet man den in der Abbildung 1.2 dargestellten theoretisch vorhergesagten Wirkungsquerschnitt für die Produktion eines 500 GeV schweren Higgs-Bosons, so erkennt man, daß dieser nur etwa 1 pb beträgt. In einem von 10^{11} Ereignissen könnte also ein Higgs-Boson produziert werden. Wegen dieser geringen Rate ist die hohe LHC Luminosität auf dem

[4] 1 barn = 1 b $=10^{-24}$ cm^2

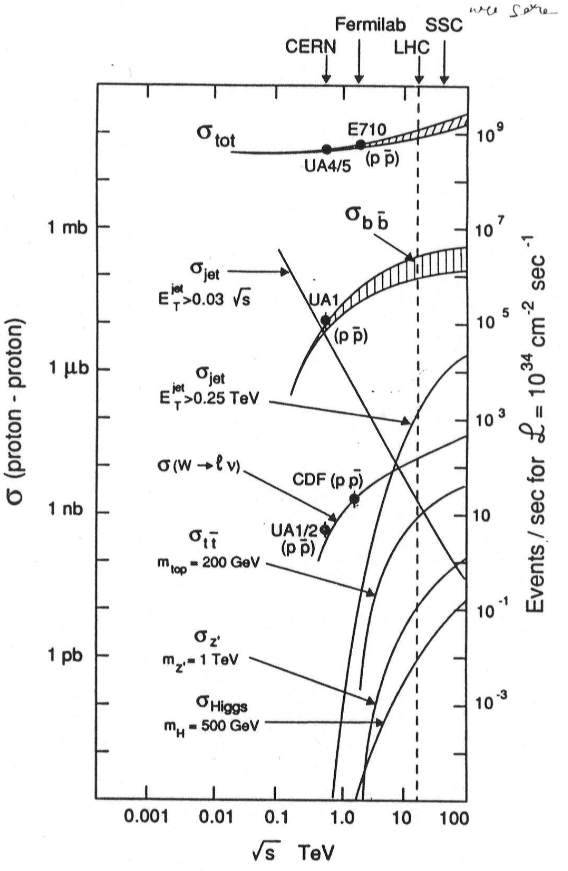

Abbildung 1.2: Wirkungsquerschnitte einiger Proton-Proton Streuprozesse [6]

Weg zu Aussagen mit ausreichender Statistik zwingend erforderlich.

1.2.5 Der Nachweis des Higgs-Bosons

Wie schon früher erwähnt, besteht eine der Hauptaufgaben am LHC im Nachweis des Higgs-Bosons. Es gibt verschiedene Mechanismen, mit denen das Higgs-Boson erzeugt werden kann. Am häufigsten sind die Gluon-Fusion (gg→H) und die W/Z-Fusion (WW→H, ZZ→H). Weniger häufig sind Prozesse, bei denen das Higgs-Boson in Verbindung mit anderen Teilchen erzeugt wird. Dies geschieht entweder mit einem $t\bar{t}$-Paar in den Reaktionen $gg \rightarrow t\bar{t}H$ und $q\bar{q} \rightarrow t\bar{t}H$, oder mit einem W bzw. Z ($q\bar{q} \rightarrow WH$, $q\bar{q} \rightarrow ZH$).

Der Nachweis des Higgs-Bosons geschieht über seine Zerfallsprodukte. Tabelle 1.3 zeigt die bei bestimmten Higgs-Massen dominierenden Zerfallskanäle.

In Abbildung 1.3 ist das Verzweigungsverhältnis der Zerfallskanäle graphisch gegenüber der Masse des Higgs-Bosons aufgetragen. Unterhalb von 120 GeV ist das Higgs-Boson nicht schwer genug, um in zwei W- oder Z-Bosonen zu zerfallen. Hier dominiert der Zerfall in $b\bar{b}$ Paar. Da $b\bar{b}$ Paare bei der Proton-Proton-Steuung auch direkt, d.h. ohne vorherige Produktion des Higgs, erzeugt werden können, benötigt man etwas, das eine

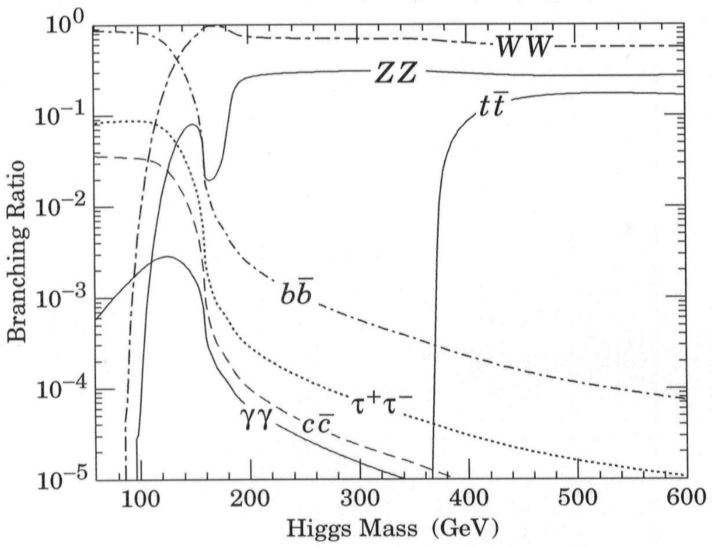

Abbildung 1.3: Verzweigungsverhältnis der Zerfallskanäle des Higgs-Bosons [4]

Zerfallskanäle	Higgs-Masse [GeV]
h $\rightarrow$ bb und H $\rightarrow$ bb	$80 < m_H < 110$
h $\rightarrow \gamma\gamma$ und H $\rightarrow \gamma\gamma$	$95 < m_H < 130$
H $\rightarrow$ ZZ* $\rightarrow$ 4l	$130 < m_H < 700$
H $\rightarrow$ ZZ $\rightarrow$ ll$\nu\nu$ und H $\rightarrow$ WW $\rightarrow$ lνjj	$600 < m_H < 1000$

Tabelle 1.3: mögliche Zerfallskanäle des Higgs-Bosons

Unterscheidung erlaubt. Daher sucht man nur nach solchen Higgs-Bosonen, die gemeinsam mit einem W oder $\mathrm{t\bar{t}}$ produziert wurden. Dieses W oder $\mathrm{t\bar{t}}$-Paar erzeugt ein isoliertes Lepton, das vom Trigger erkannt werden kann.

1.3 Der ATLAS-Detektor am LHC

Eines der zentralen Experimente am LHC ist ATLAS. Es verwendet einen Universaldetektor der in Abbildung 1.4 dargestellt ist.

Er hat eine Höhe von 22 m und wiegt etwa 7000 t. Als Universaldetektor wurde er nicht auf eine spezielle Teilchenreaktion optimiert, sondern er soll in der Lage sein, in einem großen Energiebereich Teilchen und Jets zu identifizieren, die in Proton-Proton Kollisionen entstehen. Außerdem soll er die Energie und den Impuls der Jets möglichst genau messen. Bei einigen Teilchen, wie z.B. bei einzelnen Elektronen oder Myonen, soll neben der Messung der Energie und dem Impuls auch noch die Spur rekonstruiert werden.

Der Detektor muß einige wichtige Voraussetzungen unbedingt erfüllen um seinen Aufgaben gerecht werden zu können:

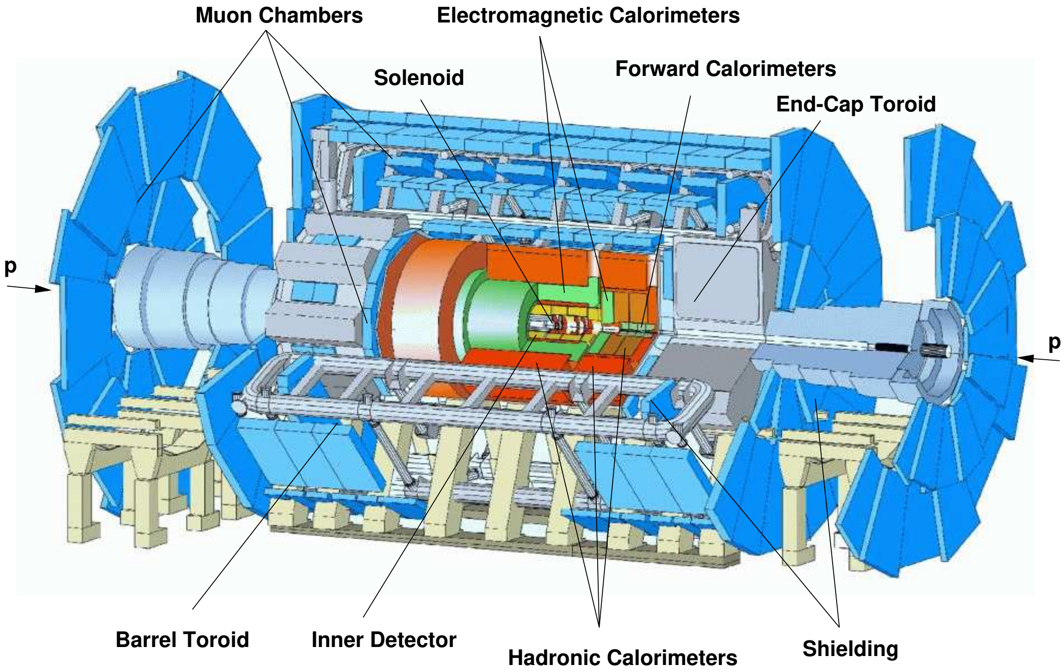

Abbildung 1.4: Der ATLAS-Detektor [7]

- Um Elektronen und Photonen zu identifizieren und ihre Energie zu messen, wird ein sehr fein segmentiertes *elektromagnetisches Kalorimeter* benötigt.
- Die Identifikation von Jets und Hadronen, sowie deren genaue Energiebestimmung fordert ein präzises *hadronisches Kalorimeter.*
- Um *Myonen* mit hohem Transversalimpuls entdecken zu können, wird ein *Myon Spektrometer* benötigt.
- Für die Teilchenidentifikation ist es wichtig die Impulse und die Teilchenspuren zu kennen. Dafür wird ein genaues Tracking-System benötigt.
- Für die einzelnen Aufgaben werden hohe Magnetfelder von einheitlicher Güte benötigt.

Im Inneren, direkt um den Wechselwirkungspunkt, liegt der sogenannte *innere Detektor*[5] (Abbildung 1.5). Er befindet sich in einem etwa 7 m langen Zylinder mit einem Durchmesser von 2,3 m. Seine Aufgabe ist es, Informationen über den Ort durchlaufender Teilchen zu sammeln, um so eine Rekonstruktion der Teilchenspuren zu ermöglichen. Dazu werden Pixeldetektoren, Siliziumstreifenzähler und Drahtkammern verwandt. Der *innere Detektor* ist in ein 2 T starkes *solenoidales*[6] Magnetfeld getaucht, um über die Ablenkung geladener Teilchen deren Impuls bestimmen zu können. Mittels der Daten,

[5] engl. Inner Detector
[6] solenoid bedeutet, daß die magnetischen Feldlinien parallel zum Strahl verlaufen

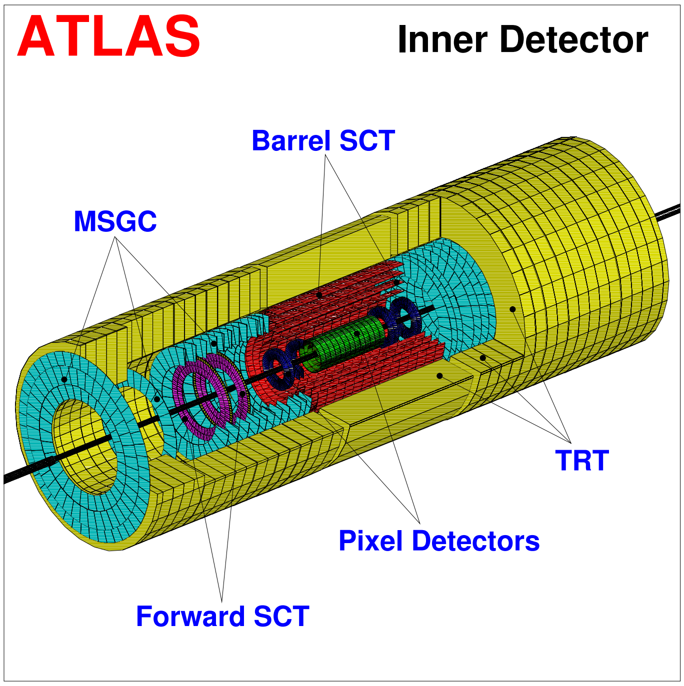

Abbildung 1.5: Der innere ATLAS Detektor [8]

die der innere Detektor liefert, können außerdem Aussagen über die Art der Teilchen und deren Entstehungspunkte, die sogenannten *Vertices* gemacht werden.

Um den inneren Detektor herum befindet sich das elektromagnetische Kalorimeter, das seinerseits von dem hadronischen umschlossen wird.

Da Myonen im Verhältnis zu Elektronen eine wesentlich größere Masse besitzen, und auch nicht stark wechselwirken, können sie den inneren Detektor sowie auch die beiden Kalorimeter relativ ungehindert passieren. Daher befindet sich jenseits der Kalorimeter noch ein Myon-Spektrometer. Um in der Lage zu sein, die Myonenimpulse zu messen, befindet sich dieses in einem toroidalen[7] Magnetfeld.

Das Koordinatensystem von ATLAS benutzt die beiden Winkel eines zylindrischen Koordinatensystems, dessen Ursprung im Wechselwirkungspunkt liegt. Die z-Achse ist die Strahlrichtung. Der Polarwinkel θ wird durch die *Pseudorapidität* η ersetzt, und zusammen mit dem Azimutalwinkel ϕ bildet er das Koordinatensystem. Die Pseudorapidität ist definiert als

$$\eta = -\ln \tan \frac{\theta}{2} \tag{1.5}$$

Sie ist für $\beta = \frac{v}{c} \to 1$ bis auf eine additive Konstante invariant unter Lorentz-Transformationen.

[7]toroidal bedeutet, daß die Feldlinien auf konzentrischen Kreisen um die Strahlrichtung liegen. Die Kreise liegen außerdem in Ebenen senkrecht zur Strahlrichtung

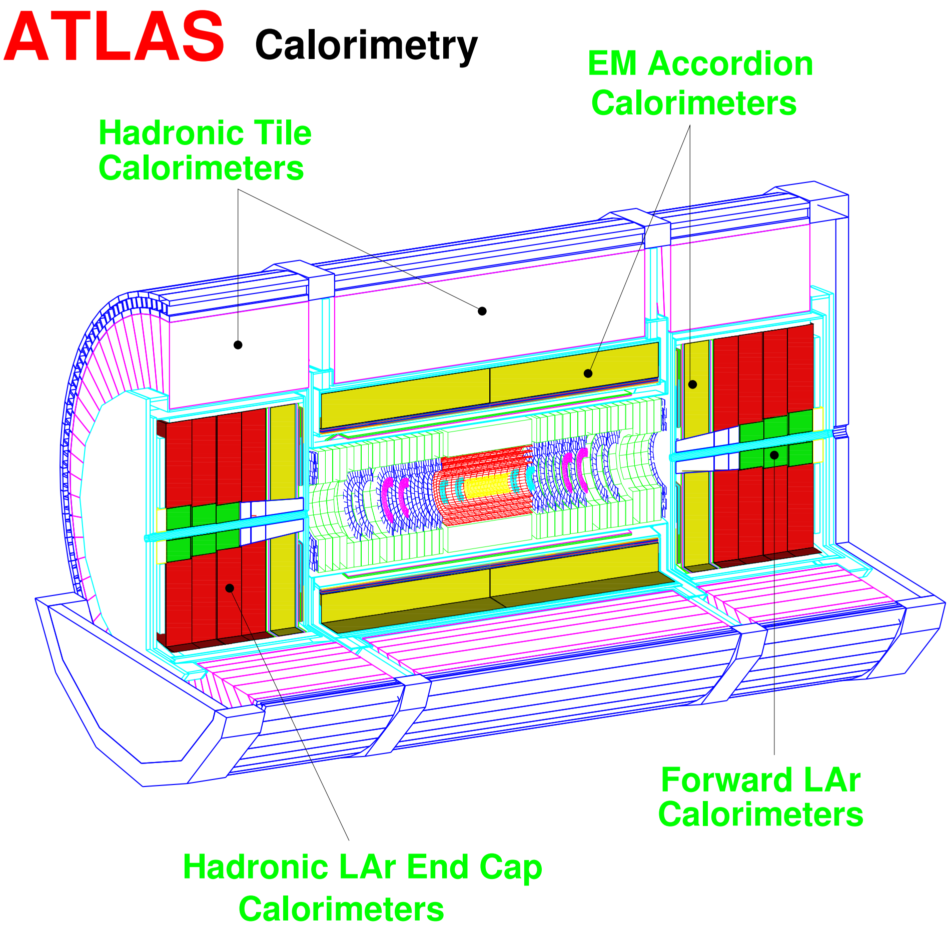

Abbildung 1.6: Das Kalorimetersystem von ATLAS [7]

1.4 Der Aufbau des ATLAS-Kalorimeters

Das Kalorimeter (Abbildung 1.6) spielt für ATLAS eine entscheidende Rolle. Es soll die Energie und Flugrichtung von Photonen, Elektronen, isolierten Hadronen sowie Jets bestimmen und eine genaue Aussage über die dazugehörende fehlende Transversalenergie treffen. Um den hohen Strahlendosen widerstehen zu können und eine hohe Granularität zu erreichen, werden unterschiedliche Techniken für die verschiedenen Aufgaben benutzt. Im Gesamten ist das Kalorimeter aus sich abwechselnden Schichten von Absorber und sensitivem Material aufgebaut, und wird daher als *Sampling*-Kalorimeter bezeichnet. Die Absorberschichten dienen dazu, den durchfliegenden Teilchen Energie zu entziehen, was durch Schauerbildung sekundärer Teilchen erreicht wird. Diese Schauer können dann mittels Ionisation in den sensitiven Bereichen nachgewiesen werden.

1.4.1 Das elektromagnetische Kalorimeter

Das elekromagnetische Kalorimeter benutzt als sensitives Material flüssiges Argon (LAr). Die zur Erzeugung eines Elektron-Ion Paares nötige Energie liegt im flüssigen Argon bei $W_i(LAr) = 24eV$ [9]. Der Grund für die Verwendung von Argon liegt in der hohen Strahlenbelastung, die wegen der hohen Luminosität speziell in Strahlrichtung herrscht. Das flüssige Argon erleidet, im Gegensatz zu vielen anderen sensitiven Materialien, keine Strahlenschäden und ist daher in strahlungsintensiven Regionen besonders nützlich. Als Absorbermaterial wird Blei benutzt, da dies einen hohen Wirkungsquerschnitt für *Bremsstrahlung* und *Paarbildung* besitzt, und diese Mechanismen im elektromagnetischen Teil vorherrschen. Die totale Dicke des elektromagnetischen Kalorimeters beträgt etwa 25 Strahlungslängen im *Faß*[8] und 26 in den *Endkappen*[9]. Damit ist gewährleistet, daß auch hochenergetische Elektronen und Photonen ihre Energie nahezu vollständig im Kalorimeter verlieren. Eine sehr feine Segmentierung der Kalorimeter ist erforderlich, um einzelne Photonen und Elektronen von Jets, die eine ähnliche räumliche Signatur hinterlassen, unterscheiden zu können. Die feine Segmentierung führt zu einer sehr großen Auslesekanalzahl von über 200.000 Auslesekanälen.

1.4.2 Das hadronische Kalorimeter

Die Hadronen besitzen eine wesentlich größere Masse als die Elektronen und verlieren daher im elektromagnetischen Kalorimeter durch Bremsstrahlung nur einen geringen Teil ihrer Energie. Im Gegensatz zu den Elektronen und Photonen ist der für die Energieabnahme dominante Prozeß für die Hadronen die starke Wechselwirkung mit Atomkernen des Absorbermaterials des hadronischen Kalorimeters. Für diesen Prozeß bietet Blei nicht mehr den größten Wirkungsquerschnitt, und daher geht man zu anderen Absorbermaterialien über. In den Bereichen, die in Strahlrichtung liegen, verwendet man Kupfer und Wolfram, im äußeren Bereich dagegen Eisen. Da die Strahlenbelastung aufgrund der Abschirmung durch das elektromagnetische Kalorimeter schon wesentlich reduziert ist, wird flüssiges Argon hier nur noch in den End-Kappen verwandt. In den anderen Bereichen des hadronischen Kalorimeters werden Plastikszintillatoren als sensitives Material eingesetzt. Das hadronische Kalorimeter benötigt im Unterschied zum elektromagnetischen nur gut 20.000 Auslesekanäle.

1.5 Messung der fehlenden Transversalenergie

Der longitudinale Energieanteil der Teilchen, die durch den Stoß erzeugt wurden, enthält keine physikalische verwertbare Aussage, da der Impuls im Schwerpunkt der zusammenstoßenden Partonen in longitudinaler Richtung zum Zeitpunkt des Stoßes nicht bekannt ist. Dies liegt daran, daß der Anteil der Partonen am longitudinalen Gesamtimpuls des jeweiligen Protons in der Regel verschieden ist. Der transversale Impuls der Protonen ist gegenüber dem longitudinalen vernachlässigbar klein.

[8] engl. barrel
[9] engl. end-caps

Das Kalorimeter hat die Aufgabe den transversalen Anteil der Impulse, der durch den Stoß entstandenen Teilchen und Jets, genau zu bestimmen. Mittels dieser Information kann man zu einer wichtigen Meßgröße, der fehlenden Transversalenergie $E_{\mathrm{T}}^{\mathrm{miss}}$, gelangen. Man addiert dafür alle ermittelten Transversalimpulse der Teilchen vektoriell miteinander. Die Summe muß dann aufgrund des Impulserhaltungssatzes Null ergeben. Ist dies nicht der Fall, so haben schwach wechselwirkende Teilchen, wie z.B. *Neutrinos*, ihre Energie nicht im Detektor deponiert, sondern aus diesem hinausgetragen. Aus dem fehlenden Transversalimpuls läßt sich die fehlende Transversalenergie ableiten. Um dies mit ausreichender Genauigkeit tun zu können, ist es notwendig, daß das Kalorimeter den Raumwinkel, in den die Teilchen emittiert werden, möglichst vollständig und genau untersuchen kann. Insgesamt deckt das Kalorimeter einen Bereich von $|\eta| < 4.9$ in Pseudorapidität ab.

Kapitel 2

Der ATLAS Trigger

2.1 Das ATLAS Level-1 Trigger Konzept - Überblick

Die bei ATLAS anfallenden Datenmengen sind so groß, daß eine vollständige *off-line* Analyse, d.h. das Aufzeichnen aller anfallenden Daten und eine anschließende Analyse dieser, nicht mehr möglich ist. Um den physikalischen Inhalt eines Ereignisses, d.h. die Spuren beteiligter Teilchen und Jets, sowie deren Energie, zu bestimmen, ist es jedoch notwendig, den Detektor mit seiner vollen *Granularität* auszulesen. Daher muß eine Auswahl zwischen *interessanten* und *weniger interessanten* Ereignissen getroffen werden. Die Daten der interessanten Ereignisse werden dann komplett ausgelesen, wohingegen die der uninteressanten verworfen werden, und für immer verloren sind. Dies stellt sehr hohe Anforderungen an ein elektronisches Triggersystem, da dies sicherstellen muß, daß keine interessante Physik verloren geht. In Tabelle 2.1 sind die Datenmengen der einzelnen Subdetektoren dargestellt.

Subdetektor	**Kanalzahl**	**Datenvolumen [kByte]**
Pixel Detektor	$1,4 \cdot 10^8$	50
Spurrekonstrunktion	$5,6 \cdot 10^6$	850
Kalorimeter	$2,3 \cdot 10^5$	180
Myon Detektor	$1,3 \cdot 10^6$	200
Gesamt		1280

Tabelle 2.1: Im Detektor pro Bunch-Crossing (alle 25 ns) entstehende Datenvolumina [10]

Die Zeit für ein Bunch-Crossing beträgt 25 ns, d.h. 40.000.000-mal in der Sekunde finden Ereignisse mit einer Datenrate von jeweils 1,28 MByte statt. Dies ergibt eine Datenmenge von etwa 51,2 TByte/s. Die Aufgabe des Triggers besteht nun darin, diese Datenmenge durch die Auswahl physikalisch interessanter Ereignisse auf ein erträgliches Maß zu reduzieren, was eine Datenreduktion um einen Faktor im Bereich von 10^7 bedeutet. Dies muß in möglichst kurzer Zeit geschehen, da während dieser Zeit alle anfallenden Daten zuerst einmal in einer *Pipeline* zwischengespeichert werden müssen. Die Länge dieser Pipeline wird nur durch die Zeit bestimmt, die die erste Stufe des Triggers braucht,

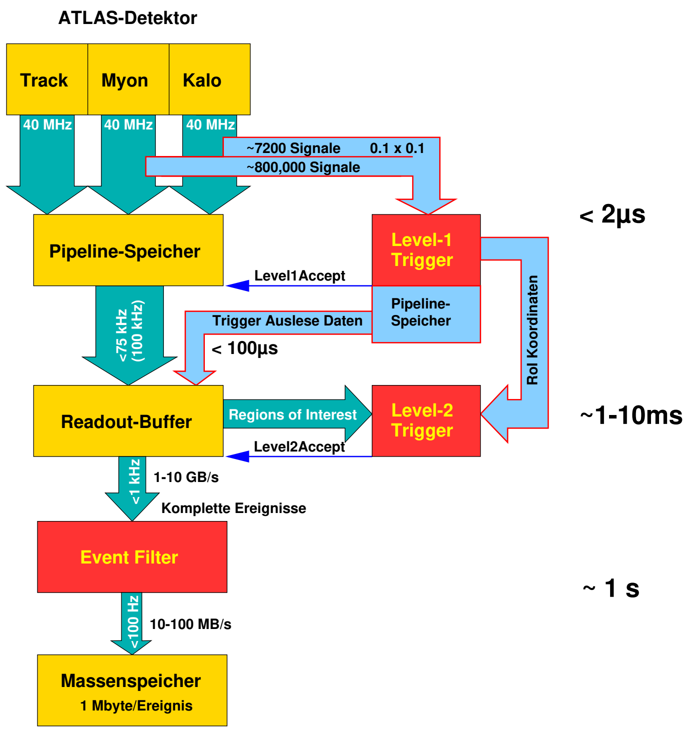

Abbildung 2.1: Detektor-Auslese und Trigger [11]

um eine Entscheidung zu treffen. Bei einer Anzahl von über einer Million Auslesekanälen, ist es einleuchtend, daß jegliche Verlängerung der Pipeline einen enormen finanziellen und technologischen Aufwand nach sich zieht.

Der Weg zur Triggerentscheidung führt über drei Stufen, den Level-1 und den Level-2 Trigger, sowie den Event-Filter. In jeder dieser Stufen wird die Datenmenge reduziert, so daß sich die Zeit, die für die jeweilige Entscheidung zur Verfügung steht, von Stufe zu Stufe erhöht werden kann. Abbildung 2.1 zeigt den gesamten Ausleseweg in Verbindung mit den drei Trigger-Stufen.

2.2 Der Level-1 Trigger

Der Level-1 Trigger ist zuständig für das Entdecken und Auswählen seltener pysikalischer Prozesse. Seine Selektivität ist so hoch, daß er aus der 40 MHz LHC Datenrate eine Rate von 75 kHz erzeugt, wobei auch die Möglichkeit einer Erweiterung auf 100 kHz vorgesehen ist. Das heißt, im Schnitt wird nur eine von etwa 500 Kollisionen für eine weitere Untersuchung ausgewählt.

In dieser Triggerebene müssen während der Bearbeitungszeit die Auslese-Daten des gesamten Detektors mit der vollen Ausleserate von 40 MHz zwischengespeichert werden. Aus diesem Grund ist die zeitliche Beschränkung für den Level-1 Trigger-Algorithmus am stärksten ausgeprägt. Es wäre wünschenswert, wenn die Entscheidung, ob die Daten des Ereignisses aufgezeichnet werden sollen oder nicht, das sogenannte *Level1-Accept-Signal*, schon nach 25 ns vorliegen würde. In diesem Fall würde keine Pipeline für die Datenspeicherung benötiget werden. Dies ist jedoch ein unerfüllbares Ziel, und so wurde dafür eine Zeit vorgesehen, die gerade noch realisierbar ist. Die ausgewählte Zeit beträgt 2 μs, was 80 Bunch-Crossings entspricht. Um diese Zeitvorgabe erfüllen zu können, arbeitet der Level-1 Trigger nicht auf der vollen Detektorgranularität und verwendet nicht die Daten aller Detektoren. Die Granularität des Levep-1 Triggers beträgt 0.1 x 0.1. Dafür werden mehrere Kalorimeterkanäle werden in sogenannten *Trigger-Towern* zusammengefaßt. Dies ergibt eine Gesamtzahl von etwa 7200 zu verarbeitenden analogen Signalen, die die in jedem Trigger-Tower deponierte Energie darstellen.

Die etwa 7200 Signale müssen alle 25 ns ausgelesen und verarbeitet werden. Diese immer noch große Datenmenge und die kurze Zeit, die zur Verfügung steht, um eine Entscheidung zu treffen, machen es notwendig, spezielle Technologien zu verwenden. Daher wird der Level-1 Trigger aus speziell für diesen Anwendungszweck entworfenen integrierten Schaltkreisen, sogenannten ASICs[1], aufgebaut. Damit erreicht man eine hohe Integrationsdichte der elektronischen Komponenten. Der Gebrauch dieser ASICs führt dazu, daß der Level-1 Trigger auch als *Hardware-Trigger* bezeichnet wird.

Der Level-1 Trigger selbst ist wiederum in drei funktionale Blöcke unterteilt, (siehe Abbildung 2.1), den *Kalorimeter-Trigger*, den *Myon-Trigger* und den *zentralen Trigger Prozessor*[2]. Diese werden in den folgenden Abschnitten beschrieben.

2.2.1 Der Kalorimeter-Trigger

Der Kalorimeter-Trigger erhält etwa 7200 vorsummierte Trigger-Tower Signale vom hadronischen und elektromagnetischen Kalorimeter. Dies entspricht einer Granularität von 0,1 sowohl in η, als auch in ϕ Richtung. Seine Aufgaben sind:

- Vorverarbeitung[3] der Eingangssignale
- Die Identifikation von Elektronen und Photonen
- Das Lokalisieren von Hadronen und Taus
- Das Entdecken von Jets
- Die Berechnung globaler Größen: fehlende Transversalenergie (E_T^{miss}) und gesamte Transversalenergie (sum-E_T)
- Die Auslese der Trigger-Daten.

Die Vorverarbeitung der Daten wird von einem *Präprozessor-System* durchgeführt, das am Kirchhoff-Institut für Physik der Universität Heidelberg entwickelt wird.

Elektronen und Photonen werden mittels eines Algorithmus gesucht, der Triggerzellen aufspürt, deren Energieeintrag eine bestimmte gesetzte Schwelle überschreitet. Um auch Teilchen korrekt identifizieren zu können, die ihre Energiedeposition auf benachbarten Zellen verteilt haben, werden immer Paare nebeneinander liegender Zellen betrachtet.

[1]Application Specific Integrated Circuit
[2]engl. Central Trigger Processor (CTP)
[3]engl. preprocessing

Um festzustellen, ob es sich um eine lokalisierte Energiedeposition handelt, wie dies für einzelne Elektronen oder Photonen der Fall sein sollte, wird ein Isolationskriterium eingeführt. Dieses betrachtet einen Ring von Zellen um diejenige, in der der Schwellenwert überschritten wurde, und überprüft, ob deren Energiedepositionen unterhalb eines gesetzten niedrigen Wertes sind. Ist dies der Fall, so handelt es sich um eine räumlich isolierte Energiedeposition. Um auszuschließen, daß es sich hierbei um einen Jet handelt, wird ein weiteres Isolationskriterium benutzt. Dazu verfolgt man den Jet bis zum hadronischen Kalorimeter und untersucht dort die dazugehörige Zelle. Ist der Energiegehalt dieser Zelle sehr klein, so ist die Wahrscheinlichkeit hoch, daß es sich nicht um eine hadronisches Ereignis handelt.

Im Falle des Jet-Algorithmus betrachtet man keine einzelnen Zellen, sondern Gruppen von mehreren quadratisch angeordneten Zellen, was zu einer Granularität von 0.2 x 0.2 führt. Dies liegt daran, daß Jets nicht so stark lokalisiert sind wie Elektronen. Es wird dann der Energiegehalt dieses Fensters ermittelt, und gegen einen Schwellwert verglichen. Ist dieser überschritten, so wird ein Jet dieser Schwellenergie gezählt. Als Größe des Fensters kann entweder eine Matrix aus 2 x 2, 3 x 3 oder 4 x 4 Zellen gewählt werden.

Die Schwellen, die auf die transversale Energie gesetzt werden, haben einen sehr großen Einfluß auf die Triggerrate und sind daher mit großer Sorgfalt auszuwählen. Werden sie zu niedrig angesetzt, so schnellt die Triggerrate nach oben, da die Ereignisrate mit sinkender Transversalenergie sehr stark zunimmt. Wählt man dagegen zu hohe Schwellen, so gehen möglicherweise physikalisch interessante Ereignisse verloren.

Der E_T^{miss}-Trigger führt eine vektorielle Summation über die transversalen Impulsanteile aller identifizierten Teilchen und Jets eines Ereignisses durch und berechnet daraus die fehlende Transversalenergie. Dieses Ergebnis wird dann mit einer dafür gesetzten Schwelle verglichen. Für die Güte dieses Verfahrens ist es von entscheidender Bedeutung, welcher η-Bereich vom Detektor abgedeckt wird.

2.2.2 Der Myon-Trigger

Der Myon-Trigger erhält seine Daten zum einen von sogenannten *resistive-plate* Triggerkammern[4] aus der *Faß-Region*[5], die eine Abdeckung von $|\eta| < 1.05$ in Pseudorapidität aufweisen. Zum anderen gelangen Daten von den *thin-gap*[6] Triggerkammern aus der End-Kappen Zone ($1.05 < |\eta| < 2.4$) in den Myon-Trigger. Zusammen kommen etwa 800.000 Signale aus den Myonkammern. Aufgrund dieser großen Zahl ist die Ausleseelektronik gegliedert in einen auf dem Detektor sitzenden, und einen im Überwachungsraum befindlichen Teil. Auf diese Weise kann die Anzahl der optischen Verbindungen vom Detektor zum Überwachungsraum auf 1200 reduziert werden.

2.2.3 Der zentrale Trigger-Prozessor

Die Aufgabe des zentralen Trigger-Prozessor-Systems besteht darin, die Triggerergebnisse aus den Myonkammern und dem elektromagnetischen Kalorimeter zusammenzuführen

[4] engl. resistive-plate trigger chamber (RPC)
[5] engl. barrel-region
[6] engl. thin-gap trigger chambers (TGC)

und daraus ein Gesamtresultat zu generieren. Dafür werden die Ergebnisse der beiden Triggerteile in Objekte eingeteilt. Diese werden dann mittels Boolscher Algebra verknüpft und zu 96 *Items* zusammengefügt. Ein *Item* vereinigt die notwendigen Signaturen eines bestimmten, physikalisch interessanten Prozesses, z.B. eine lokalisierte Energiedeposition von über 15 GeV im elektromagnetischen Kalorimeter in Verbindung mit einem Myon von über 10 GeV. Das Gesamtergebnis wird in einem Bit übertragen, dem sogenannten *Level-1-Accept*-Bit. Dieses Bit signalisiert, ob das Ereignis aus Sicht des Level-1 Triggers aufgezeichnet werden soll. Es wird dann über das sogenannte *TTC-System*[7] den einzelnen Modulen, die es benötigen, wieder zugeführt.

2.3 Der Level-2 Trigger

Der Level-2 Trigger muß die Datenrate von 75 kHz, die er vom Level-1 Trigger erhält, auf eine Rate reduzieren, mit der der Event-Filter zurechtkommt. Diese liegt unter 1 kHz. Der Level-2 Trigger arbeitet im Gegensatz zu Level-1 wieder auf der vollen Granularität, allerdings nur in Gebieten, die er von Level-1 mitgeteilt bekommt, den sogenannten *Regions of Interest (RoI)*. Ein solches *interessantes Gebiet* wird von Level-1 signalisiert durch eine Energieanhäufung im Kalorimeter. Diese ist dann gegeben, wenn zumindest eine der vorher gesetzten Energieschwellen überschritten wird. Zusätzlich wird noch ein Isolationskriterium angewandt, d.h. benachbarte Zellen müssen in ihrem Energiegehalt unter dem der als RoI ausgewählten Zelle liegen. Es wird außerdem ein RoI signalisiert, wenn der Energiegehalt eines Objektes im Myontrigger eine bestimmte Schwelle überschreitet. Als Koordinaten der RoIs werden ihre genaue Position in η und ϕ übertragen. Der Level-2 Trigger entscheidet, welche Daten der jeweiligen Subdetektoren zu welchen RoIs gehören und gibt nur diese an seine Prozessoren weiter.

Die Verarbeitung der Ereignisdaten kann in mehrere Stufen unterteilt werden. In der ersten, der sogenannten *feature extraction*, werden Daten aus einer RoI für einen Subdetektor zusammengenommen und damit physikalische Größen erzeugt. Für das Kalorimeter werden z.B. mehrere Zellen zusammengezogen, um so physikalische Informationen bezogen auf physikalische Teilchen, die die Energie dort deponiert haben, zu erhalten. Im Falle der Spurkammern werden Trefferinformationen in Spurparameter umgerechnet.

Die zweite Stufe, der sogenannte *object builder*, vereinigt für jedes RoI die Werte aller beteiligten Subdetektoren, um so direkte Aussagen über Teilchenart und Teilchenparameter machen zu können. In der dritten Stufe, der sogenannten *trigger type selection*, werden alle Objekte, die in einem Ereignis gefunden wurden, miteinander kombiniert, um so aus einer Liste physikalischer Prozesse eventuell passende auszusuchen und diese Übereinstimmung dann weiterleiten zu können.

2.4 Der Event-Filter

Der Event-Filter reduziert die Datenrate nochmals um einen Faktor 10 von 1 kHz auf 100 Hz und schreibt pro ausgewähltem Ereignis 1.28 MByte auf einen Massenspeicher.

[7]engl. Timing, Trigger and Control (TTC)

Im Gegensatz zum Level-1 und Level-2 Trigger, die zur Verarbeitung Daten aus dem Hauptdatenstrom kopieren während diese dort bis zur Entscheidung weiterhin gespeichert bleiben, ist der Event-Filter Teil der direkten Datenauslese[8]. Während die Ereignisse durch das Event-Filter System verarbeitet werden, speichert dieses zudem die Ereignisdaten in zum System gehörigen Speichern. Die gegenwärtige ATLAS-Strategie besteht darin, eine Software zu entwickeln, die Zugriff auf alle Ereignisdaten sowie auf Daten über die Kalibrierung hat und damit umfassende Aufgaben erfüllen kann. Sie soll in der Lage sein, die Ereignisse möglichst komplett zu rekonstruieren und zu analysieren. Die Algorithmen des Event-Filters sind im Prinzip denen gleich, die später in der *off-line* Analyse verwendet werden. Darüber hinaus soll die Software auch die Möglichkeit haben, Studien über die Kalibrierung durchzuführen und Aussagen über die Güte des Triggers zu treffen. Der endgültige Aufbau des Event-Filters und seine Auswahlkriterien sind noch nicht festgelegt, da es für eine Optimierung am besten ist, Daten aus der Initialisierungsphase des Triggers zu verwenden. Des weiteren steht noch nicht endgültig fest, ob die beiden Stufen Level-2 und Event-Filter tatsächlich getrennt werden müssen, oder ob eine Verbindung der beiden möglich bzw. sinnvoll ist. Da die Algorithmen dieser beiden Stufen im Gegensatz zu Level-1 nicht in Hardware implementiert sind, sondern durch Softwareprogramme realisiert werden, nennt man sie auch *Software-Trigger*.

[8]engl. Data Acquisition (DAQ)

Kapitel 3

Das Präprozessor-System

3.1 Aufbau des Systems

Das Präprozessor-System dient als Bindeglied zwischen den eigentlichen Prozessoren des Level-1 Triggers und den Kalorimetern. Es erhält seine Eingangssignale von den Kalorimetern, führt eine Signalvorverarbeitung durch und schickt seine Daten über schnelle Verbindungen (*links*) zu den nachfolgenden Prozessoren. Abbildung 3.1 zeigt die Einbettung des Präprozessormoduls in die Kalorimeter-Trigger-Architektur.

Das Präprozessor-System besteht aus 128 Präprozessor-Modulen (PPMs), die sich auf 8 *Crates*[1] verteilen. Je zwei *Crates* befinden sich in einem *Rack*[2]. Jedes PPM enthält 16 *MCMs*[3]. Die MCMs bilden die kleinste austauschbare elektronische Einheit des Systems. Sie beinhalten je einen ASIC für die Signalvorverarbeitung, zwei schnelle ADCs[4] für die Digitalisierung der Kalorimetersignale und drei LVDS-Treiber-Chips[5]. Da jeder PPrAsic 4 Kanäle verarbeiten kann, ergibt sich eine Gesamtzahl von 64 Kanälen pro PPM. Jedes PPM kann also 64 Trigger-Tower Signale verarbeiten. Dies führt bei 128 verwendeten PPMs zu einer Gesamtzahl von 8192 möglichen Eingangskanälen.

Außerdem soll jedes PPM noch Eingangsstecker für 64 differentielle analoge Signale, die sich auf der Frontplatte befinden, beinhalten. Auf der Rückseite sollen neben den Steckern für einen *VME-Bus* und für die seriellen *high-speed*-Verbindungen auch die für einen *PipelineBus* angebracht werden. Die Aufgabe des *Pipeline-Bus* besteht in der Verbindung der PPMs mit einem Auslesetreiber. Er stellt einen Auslesebus hoher Bandbreite dar, der auf die speziellen Anforderungen des Präprozessorsystems und der ATLAS-DAQ angepaßt wurde [12]. Des weiteren soll sich auf jedem PPM ein Readout-Merger ASIC (RemAsic) [13] befinden, der die Auslesedaten aller auf dem Modul befindlichen MCMs sammelt und an den *PipelineBus* versendet.

Jedes MCM liefert drei Ausgangssignale. Im Falle des Cluster-Prozessors werden je-

[1] Ein *Crate* ist der mechanische Aufbau, der dazu dient, Steckkarten aufzunehmen. Er stellt die Spannungsversorgung und eine *Backplane* zur Verfügung, die alle Karten elektrisch miteinander verbindet. Für ATLAS werden Crates mit 9 Höheneinheiten (9 U=40 cm), sogenannte 9 U Crates verwandt.

[2] Ein *Rack* ist eine Gestell, das dazu dient einzelne *Crates* aufzunehmen.

[3] Multi-Chip-Modules

[4] Analog Digital Converter

[5] Low Voltage Differential Signalling

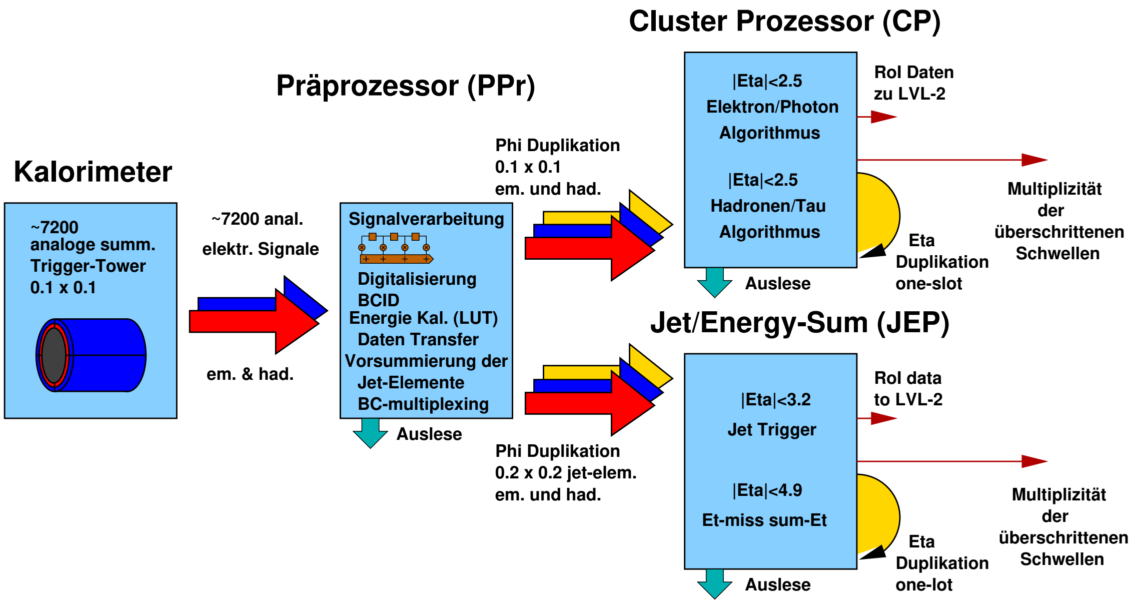

Abbildung 3.1: Überblick über die Einbindung des Präprozessor-Systems in die Level-1 Kalorimeter-Trigger Architektur [11]

weils zwei Ergebnisse der 4 Kanäle auf ein Signal gemultiplext, und daher benötigt man hierfür zwei der drei Signale. Das dritte Signal geht an den Jet-Prozessor. In ihm sind die Ergebnisse aller vier Kanäle in einer Summe zusammengefaßt. Die Zahl der Ausgangsverbindungen wird jedoch noch durch zwei zusätzliche Faktoren bestimmt. Zum einen werden Ergebnisse aus Signalen des Vorwärtskalorimeters und aus Teilen der Endkappen, die eine Pseudorapidität von $|\eta| > 2.5$ besitzen, nur an den Jet-Prozessor geschickt, zum anderen benötigt man zusätzliche Verbindungen, die durch die Architektur des Trigger-Systems bedingt sind. Die Trigger-Matrix wird entlang der ϕ-Richtung in 4 Quadranten unterteilt. Jeder dieser Quadranten wird dann von zwei *Crates* des Präprozessor-Systems repräsentiert. Die Trigger-Algorithmen, die auf bewegten Fenstern beruhen (siehe Abschnitt 2.2.1), benötigen die Daten benachbarter Zellen. An den Grenzen der Quadranten sind für die volle Funktionalität der Algorithmen also Daten von verschiedenen Quadranten nötig. Um zu verhindern, daß die Prozessor-Crates untereinander kommunizieren müssen, werden die Daten an den ϕ-Grenzen der Quadranten verdoppelt und an zwei benachbarte Quadranten der Level-1-Prozessoren verschickt. Für die Versendung der verdoppelten Daten verden zusätzliche Daten-Verbindungen vom Präprozessor zu den nachfolgenden Prozessoren benötigt.

3.2 Funktionalität des Systems

Um alle Trigger-Tower Signale in 8 elektronischen *Crates* unterzubringen, wird ein Präprozessor-Modul benötigt, das 64 Kanäle verarbeitet. Dies erfordert einen hohen Integrationsgrad der einzelnen Komponenten. Abbildung 3.2 zeigt die Zahl der Eingangssignale,

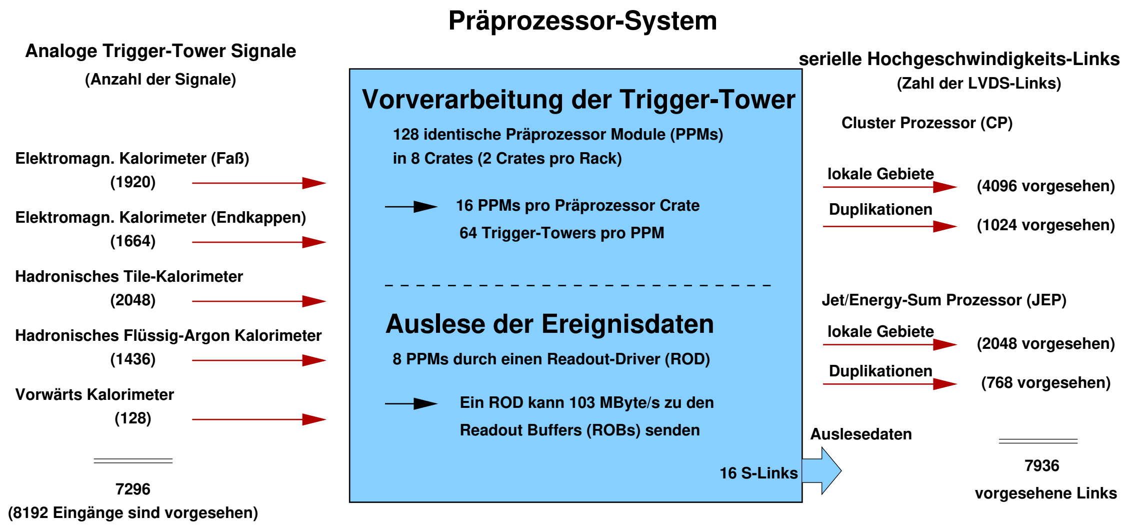

Abbildung 3.2: Überblick über das Präprozessor-System [11]

die Zahl der seriellen Ausgangsverbindungen (*links*) und die Anzahl der zu verwendenden Module bei einer 64-Kanal Lösung.

Insgesamt erhält das Präprozessor-System 7296 analoge Eingangssignale von den verschiedenen ATLAS-Kalorimetern. Diese werden in transversale Energie übersetzt. Für das Flüssig-Argon Kalorimeter sowie das Tile-Kalorimeter geschieht dies in Empfängerstationen, die vor dem Präprozessor-System liegen. Die Gesamtzahl an vorgesehenen Ausgangs-Verbindungen liegt bei 7936. Diese Zahl beinhaltet die Verbindungen, die für die sogenannten ϕ-*Duplikation* vorgesehen sind.

Die Aufgaben, die das Präprozessor-System erfüllen muß, können zu zwei Punkten zusammengefaßt werden:

- **Signalvorverarbeitung:** Die von den Kalorimetern kommenden analogen Signale müssen zuerst digitalisiert werden, damit sie von der nachfolgenden digitalen Logik verarbeitet werden können. Diese ermittelt dann die deponierte transversale Energie, die der ankommenden Puls repräsentiert und findet diejenige Teilchenkollision zu der der Puls gehört. Letzteres wird *Bunch-Crossing-Identification* genannt (siehe Kapitel 4). Diese Signalvorverarbeitung basiert nicht auf der vollen Detektorgranularität, sondern im Fall des Cluster-Prozessors geschieht dies mit einer Granularität von 0,1 x 0,1 für $|\eta| < 2.5$ und für den Jet-Prozessor mit 0,2 x 0,2 für $|\eta| < 4.9$. In beiden Fällen sind die Eingangsdaten getrennt für die elektromagnetischen und die hadronischen Kalorimeter.

- **Auslese der Ereignisdaten:** Um Aussagen darüber treffen zu können, was den Trigger ausgelöst hat, sowie Analysen die Trigger-Qualität betreffend durchführen zu können, werden die Roh-Daten benötigt, die das Präprozessor-System erhält. Diese müssen dem ATLAS DAQ-System zugeführt werden.

Kapitel 4

Ereigniszugehörigkeit der Kalorimetersignale

4.1 Zeitstruktur der Kalorimetersignale

Für jede Kollision der Teilchenpakete am LHC, d.h. alle 25 ns, muß der Level-1-Trigger die Entscheidung treffen, ob ein möglicherweise interessanter physikalischer Prozeß stattgefunden hat. Diese Entscheidung basiert im Falle des Level-1 Kalorimeter-Triggers auf etwa 7200 analogen Signalen, die durch Summation der der Signale der Kalorimeterzellen in sogenannten Trigger-Towern erzeugt worden sind. Da diese Signale in ihrer zeitlichen Ausdehnung größer als 25 ns sind, ist es notwendig herauszufinden, welchem *Bunch-Crossing* das jeweils ankommende Signal zuzuordnen ist. Diesen Vorgang bezeichnet man als die *Bunch-Crossing Identification*.

4.2 Kalorimetersignale und ihre Saturierung

Die Signale, die vom Kalorimeter kommen, können in zwei Gruppen unterteilt werden. Die einen kommen vom Flüssig-Argon-Kalorimeter und weisen einen bipolaren Spannungsverlauf auf, während die anderen, die vom Tile-Kalorimeter kommen, einen unipolaren Spannungsverlauf haben.

Im Falle des Flüssig-Argon-Kalorimeters werden durch die Wechselwirkung hochenergetischer Photonen oder Elektronen mit dem sensitiven Material Elektron-Positron-Paare erzeugt, die ihrerseits einen *Schauer* von Photonen, Elektronen und Positronen erzeugen. Ein weiterer Effekt, der Schauer erzeugt, ist die Bremsstahlung. Die Schauerteilchen erzeugen im flüssigen Argon Ionisationsströme, die direkt zu einem Vorverstärker geführt werden. Danach werden sie differenziert und in einem bipolaren Pulsformer (*Shaper*) weiterverarbeitet. Ein bipolares Signal ist für eine elektrische Übertragung sinnvoll, da sich durch einen Wechselstrom mit gleichen Flächen für jede Richtung die Verschiebung der Nullinie, die sogenannte *Baseline-Verschiebung*, vermeiden läßt. Die Signale werden durch drei Arten von Rauschen beeinflußt:

- **Elektronisches Rauschen:** Dies besteht in erster Linie aus thermischem Rauschen. Die thermische Bewegung der Ladungsträger verursacht statistische Fluktuationen an den Enden des Leiters. Außerdem tragen aktive elektronische Bauteile und Halbleiter zum elektronischen Rauschen bei.

- ***Pile-Up*-Rauschen:** Dies tritt dann auf, wenn zwei Kalorimetersignale miteinander überlappen. Die Häufigkeit und Größe dieser Überlappung hängt von der Rate ab, mit der die Signale erzeugt werden und der Granularität des Detektors. Außerdem wird Sie durch die Länge der Pulse bestimmt, die ihrerseits von der Shaper-Elektronik abhängt.

- **Quantisierungs-Rauschen:** Dies entsteht durch den Rundungsfehler, der bei der Umwandlung der analogen in digitale Daten gemacht wird.

Die Aufgabe des Shapers liegt darin, das Optimum zu finden, bei dem die Summe aus elektrischem und *Pile-Up* Rauschen am geringsten wird. Dies führt zu einer optimalen Peaking-Zeit[1] zwischen $t_p(\delta) = 20$ und $t_p(\delta) = 50\ ns$ (siehe Abbildung 4.1) für die verschiedenen Kalorimeter bei voller Luminosität ($10^{34} cm^{-2} s^{-1}$).

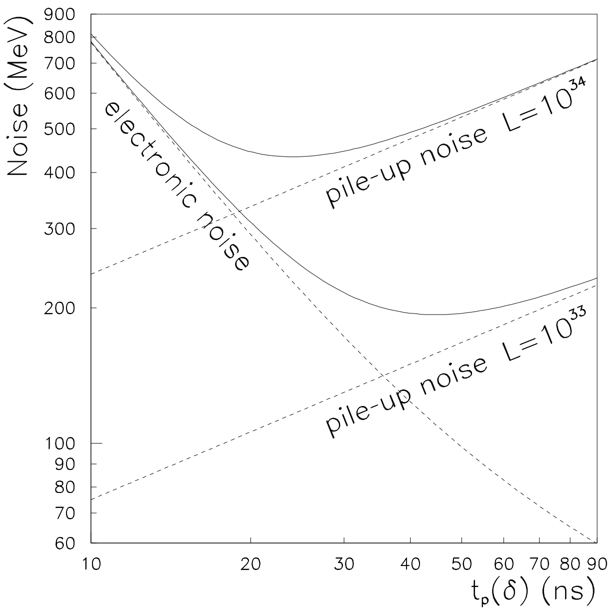

Abbildung 4.1: Abhängigkeit des Elektronik-, Pile-Up- und Gesamtrauschens von der Peaking-Zeit bei niedriger und hoher Luminosität [14]

Das bipolare Shapersignal hat eine Signalbreite von 500 ns. Abbildung 4.2 zeigt das bipolare Shapersignal und den Driftstrom der Ionisation.

Wird die Energie der ankommenden Signale sehr hoch, so führt dies dazu, daß hohe Werte nicht mehr in ihrer eigentlichen Größe übertragen werden. Sie werden dann

[1] engl. peaking time; Zeit, die der Puls benötigt, um von 5% seines Maximums auf 100% anzuwachsen

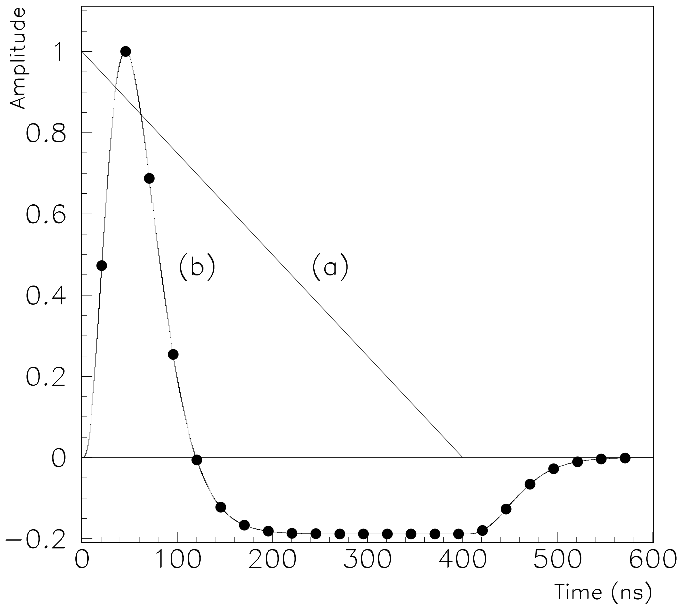

Abbildung 4.2: Driftstrom eines Ionisationskalorimeters (a), und die Antwort eines bipolaren Pulsshapers (b). Die Punkte sind die Abtastpunkte eines 40 MHz *Analog to Digital Converters (ADC)* [14]

nur noch in der maximalen Größe, die das entsprechende Bauteil in der Summierelektronik übermitteln kann, übertragen. Um möglichen Ursachen der Saturierung verstehen zu können, ist es daher notwendig, die Summierkette der Flüssig-Argon-Trigger-Tower-Elektronik genauer anzusehen. Sie ist in folgende drei Stufen unterteilt [15]:

- Ein linearer Mixer, der die Vorverstärkung und den Pulsshaper enthält.
- Das *layer-sum-board*, das die einzelnen Lagen der Kalorimeter aufsummiert.
- Das *tower-builder-board*, das die einzelnen Zellen der Kalorimeter aufsummiert und damit die Granularität von 0.1 x 0.1 erzeugt. Es nimmt dabei eine Anpassung bezüglich des zeitlichen Verlaufs, der Form und der Verstärkung der Signale vor.

In jeder dieser Stufen kann analoge Saturierung auftreten und die Form des saturierten Signals hängt davon ab, welche Stufe die Saturierung verursacht hat. Die Schwelle zur analogen Saturierung lag bei den Simulationen, die den folgenden Graphen zugrundeliegen, bei 3 V. In Abbildung 4.3 sind simulierte Pulsformen dargestellt, die bei einer Energie von 1 TeV am Eingang zum Präprozessor-System aufgezeichnet wurden. Die Simulation der Saturierung wurde mit Hilfe eines PSPICE-Modells [17] für die verschiedenen Stufen der Trigger-Tower Elektronik durchgeführt.

Saturierung im linearen Mixer beeinflußt das Maximum des Pulses nur wenig. Es bleibt weiterin gut lokalisiert und zerfließt nicht, wenn man zu hohen Energien übergeht.

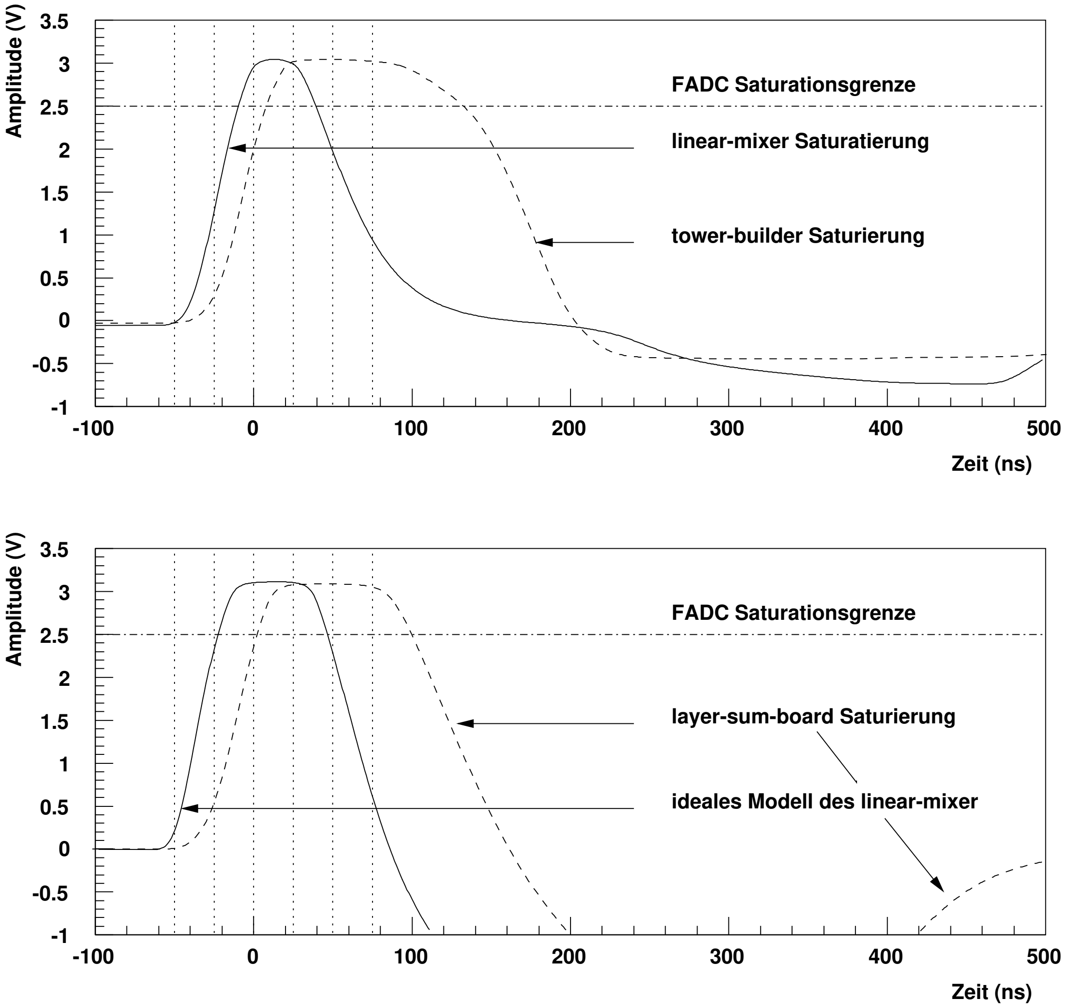

Abbildung 4.3: Pulsformen simulierter, saturierter Trigger-Tower Signale an verschiedenen Stufen der Trigger-Tower Elektronik [18]

Die Gesamtfläche unter der Kurve ist nicht länger Null, da sich der Startpunkt des Unterschwingers mit höheren Energien nach rechts verschiebt.

Tritt die Saturierung im *layer-sum-board* auf, so ergibt dies ein breites, flaches Maximum, das zu höheren Energien hin immer breiter wird. Die Gesamtfläche ist wie beim linearen Mixer von Null verschieden, da der Unterschwinger wegen der symmetrischen Stromversorgung nicht bei 1/5 der positiven Saturierung ebenfalls saturiert, sondern bis -3 V gehen kann.

Saturierung im Tower-Builder verändert die Pulsform, indem sie die fallende Flanke des Pulses zu späteren Zeiten verschiebt.

Die Auswirkungen der analogen Saturierung beeinflussen in erster Linie die fallende Flanke, wohingegen sich die ansteigende Flanke nur wenig ändert. Daher ist es ratsam, daß ein Algorithmus für saturierte Signale auf der ansteigenden Flanke arbeitet. Neben der analogen Saturierung in der Trigger-Tower Elektronik, gibt es auch noch die digitale Saturierung. Für die Digitalisierung soll ein 10-Bit breiter schneller Analog-Digitalwandler, ein sogenannter *Analog Digital Converter (ADC)* mit einem Digitalisierungsbereich von 0 bis 2,5 V verwandt werden. Daher werden Signale, die über 2,5 V liegen, durch den Maximalwert der Digitalisierung, also 3FF (hex), ausgedrückt.

4.3 Das Verfahren zur Bunch-Crossing Identifikation

Die analogen Signale, die von den Kalorimetern kommen, erstrecken sich über mehrere BCs. Die Amplitude des Maximums dieser Pulse repräsentiert die in der entsprechenden Zelle deponierte Energie. Um in der Lage zu sein, die gefundene Energiedeposition dem richtigen Bunch-Crossing zuordnen zu können, ist es notwendig, die zeitliche Lage des Pulses, durch die das dazugehörige BC definiert ist, zu bestimmen. Diesen Vorgang nennt man *Bunch-Crossing Identification (BCID)*.

4.3.1 Voraussetzungen für die Bunch-Crossing Identifikation

Eine wichtige Voraussetzung für die Zuordnung des richtigen BCs ist die Forderung, daß die Pulsform fest und somit unabhängig von der Amplitude, d.h. der Energie, ist. Dies ist jedoch nur für nicht saturierte Signale erfüllt. Im Falle einer Saturierung hängt die Pulsform sowohl von der tatsächlichen Energie der Signale, als auch davon, in welcher Komponente die Saturierung aufgetreten ist, ab. Die Art und Weise wie die Saturierung zustandekommt, hat also Einfluß auf die Signalform. Das bedeutet, daß die Pulsform variiert, je nachdem ob Zellen gleichen oder unterschiedlichen Energiegehalts zur Saturierung führen. Weitere Voraussetzungen sind in Tabelle 4.1 zusammengefaßt.

Voraussetzungen und Parameter	Größe
LAr. pile-up Rauschen	400 MeV
LAr. thermisches Rauschen	217-410 MeV $\eta = 2\text{-}0$
LAr. Gesamtrauschen	450-570 MeV $\eta = 2\text{-}0$
Tile pile-up Rauschen	90 MeV
Tile thermisches Rauschen	35 MeV
Tile Gesamtrauschen	97 MeV
Quantisierungsrauschen	75 MeV
Digitale Saturierungsschwelle	250 GeV (2,5 V;10 bit)
LAr. Analoge Saturierungschwelle	300 GeV (3,0 V)
LAr. Shaper-Zeitkonstante	$15\ ns \pm 0,5\ ns$
LAr. Peaking-Zeit (am Nullpunkt)	51-52 ns
LAr. Peaking-Zeit am Empfänger nach 70 m Kabel	63 ns
Trigger-Tower Zeit-Jitter	in $\pm$3 ns
Zeitversatz für die Trigger-Tower Summierung	$\pm$2,5 ns

Tabelle 4.1: Voraussetzungen und Parameter für die Trigger-Tower Signale des ATLAS Kalorimeters (bei hoher Luminosität) [18]

4.4 BCID für nicht saturierte Kalorimetersignale

Im Falle nicht saturierter Pulse hängt die Pulsform nicht von der Pulshöhe ab, sondern ist für alle Pulse durch die Trigger-Tower Elektronik festgelegt. Das BCID wird im Zusammenspiel von zwei hintereinander gelegenen Modulen erledigt. Zuerst werden die Pulse durch einen Filter geschickt, der Störungen herausfiltern soll. Auf diesen Ergebnissen arbeitet dann ein sogenannter *Peak-Finder*-Algorithmus, dessen Aufgabe es ist, dem Pulsmaximum das richtige Bunch-Crossing zuzuordnen.

4.4.1 Der finite impulse response Filter

Der Aufbau eines *finite impulse response*-Filters (*FIR-Filter*) ist strukturell immer gleich. Die ankommenden Datenwerte werden in einer Pipeline gespeichert, anschließend mit Koeffizienten multipliziert und danach aufsummiert (Abbildung 4.4). Die Wahl der Koeffizienten bestimmt die Filterfunktion. Die Tiefe der Pipeline und die Bit-Breite der Koeffizienten, sowie ihre korrekte Wahl, bestimmen die Qualität des Filters.

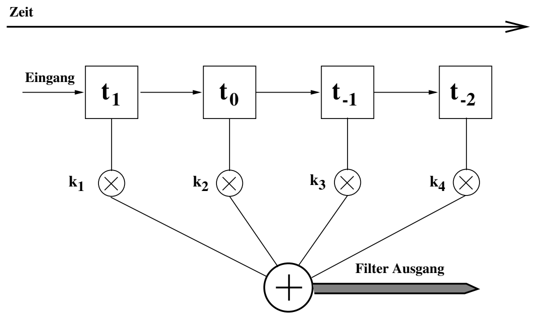

Abbildung 4.4: Aufbau eines FIR-Filters

Der FIR-Filter, der für den ATLAS Präprozessor ausgewählt wurde, besteht aus fünf programmierbaren Koeffizienten, die jeweils 4-Bit breit sind. Eine genauere Beschreibung der eigentlichen Implementation befindet sich in Kapitel 6. Die programmierbaren Koeffizienten erlauben es, den FIR-Filter an die Form der Signale, die aus der Trigger-Tower Elektronik kommen, anzupassen. Die Nutzung des FIR-Filters als einen sogenannten *matched-Filter* ergab in Simulationen die besten Ergebnisse für die anschließende Identifikation des richtigen Bunch-Crossings [21]. Ein *matched-Filter* berechnet die Wahrscheinlichkeit dafür, daß sich ein Puls vorgegebener Position und sein Maximum innerhalb der Filtersamples befindet.

4.4.2 Bestimmung des Pulsmaximums

Um anhand der Filterdaten das richtige Bunch-Crossing identifizieren zu können, benötigt man noch einen Algorithmus, der das Maximum der Filterergebnisse bestimmt. Dieser Algorithmus, der sogenannte *peak-finder*, vergleicht ankommende Werte mit ihren beiden Nachbarn in der zeitlichen Reihenfolge. Für das BCID werden zwei unterschiedliche Bedingungen verwandt:

$$bc_{t_{-1}} < bc_{t_0} \geq bc_{t_1}$$
oder
$$bc_{t_{-1}} < bc_{t_0} > bc_{t_1}.$$

Die erste Bedingung erkennt jeden Peak, auch wenn er sich über zwei BC erstreckt, d.h. in beiden den gleichen Wert aufweist. Die zweite Bedingung erkennt solche Pulse nicht an, da sie kein sauber definiertes Maximum haben.

4.5 BCID für saturierte Kalorimetersignale

Der Algorithmus für saturierte Signale ist zwar nicht abhängig von einer festen Pulsform, aber er benötigt gewisse stabile Parameter, die durch die Trigger-Tower Elektronik vorgegeben sind. Er geht davon aus, daß die ansteigende Flanke der Pulse von der Saturierung wenig beeinflußt wird. Der Grund für die Verwendung der ansteigenden Flanke liegt darin, daß man mit ihrer Hilfe in der Lage ist, den Anfangspunkt des Pulses zu erhalten. Dieser Punkt ist daher so wichtig, weil er der einzige ist, der für alle Pulse einen feste zeitliche Lage hat. Die sogenannte *peaking-time*, d.h. die Zeit, die zwischen dem Startpunkt des Pulses und seinem Maximum liegt, ist unabhängig von der Pulsform, eine Anforderung, die die Elektronik leisten muß.

4.5.1 Der Algorithmus für saturierte Kalorimetersignale

Es gibt nun verschiedene Möglichkeiten, einen solchen Algorithmus in das System zu integrieren. Die eine ist die Verwendung eines analogen Diskriminators, der direkt auf den analogen Signalen arbeitet. Dies wurde in [19] untersucht. Ein solches System benötigt eine zusätzliche Menge an analoger Elektronik, die in das Präprozessor-System integriert werden muß und zusätzliche Kosten verursacht. Daher ist es von Vorteil, wenn ein solcher Algorithmus auf der Basis der digitalisierten Werte arbeitet und damit einfacher Bestandteil des Präprozessor ASICs wird. Die Entwicklung eines solchen Algorithmus, sowie Studien über seine Zuverlässigkeit, wurden am Institut für Hochenergiephysik in Heidelberg durchgeführt [18].

Der Algorithmus stellt zuerst fest, ob ein ankommender Wert saturiert ist (t_{sat}), indem er ihn mit einer vorher gesetzten Schwelle vergleicht (G_{sat}). Überschreitet er diese Schwelle, so gilt das Augenmerk des Algorithmus den zwei Werten, die zeitlich vor ihm eingetroffen sind. Zu diesem Zweck müssen die ankommenden Werte vorher gespeichert worden sein. Abhängig davon welche Schwellen überschritten worden sind, wird entweder t_{sat} oder der folgende t_{sat+1} als das korrekte *Bunch-Crossing* erkannt. Abbildung 4.5 a) zeigt ein Trigger-Tower Signal das gerade saturiert. Die Werte, die vom Algorithmus verwendet werden, sind als Punkte bei t_{sat-2}, t_{sat-1} und t_{sat} eingezeichnet. Der Algorithmus erkennt in diesem Fall t_{sat} als das zugehörige *Bunch-Crossing*. Dies bleibt solange der Fall, bis der Wert bei t_{sat-1} ebenfalls die Saturierungsgrenze erreicht, damit den Algorithmus auslöst, und dadurch selbst zum Wert t_{sat} wird. Einen solchen Fall zeigt Abbildung 4.5 b). In diesem Fall befindet sich der Wert bei t_{sat-1} unterhalb der oberen Schwelle und daher wird korrekterweise t_{sat+1} als das zugehörige *Bunch-Crossing* identifiziert.

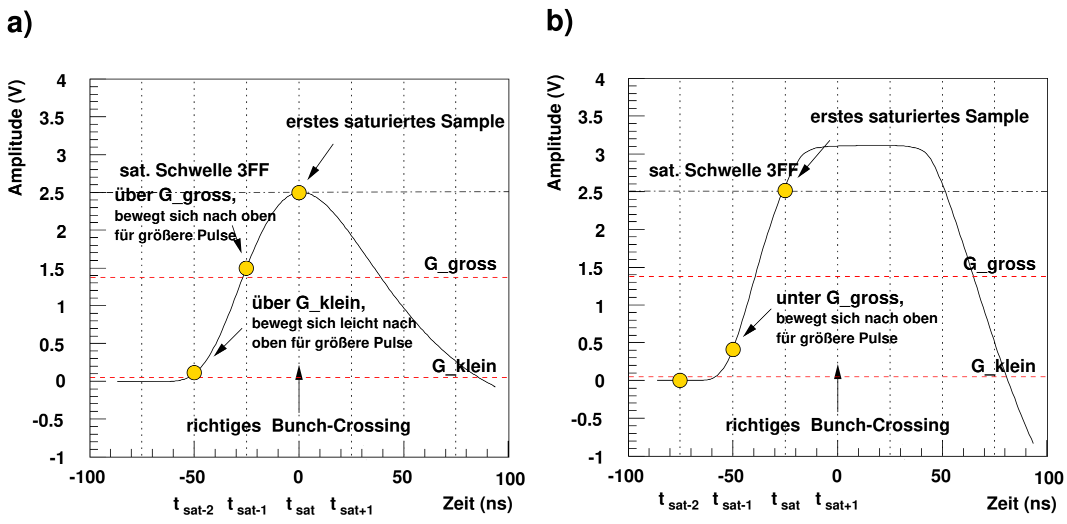

Abbildung 4.5: Arbeitsweise des Algorithmus für unterschiedlich stark saturierte Pulse: a) Puls liegt bei etwa 250 GeV, b) Puls liegt deutlich über der Saturierungsschwelle [18]

Für Pulse noch höherer Energie tritt ein zusätzlicher Effekt auf. Der Wert bei t_{sat-1} kann nicht weiter steigen und die Grenze $G_{groß}$ überschreiten, da die Ausgangsspannung nicht in unendlich kurzer Zeit von 0 auf 2.5 V ansteigen kann. Ein solcher Puls ist in Abbildung 4.6 a) dargestellt. Bis hierher erkennt der Algorithmus das korrekte *Bunch-Crossing*, ohne daß er die zweite kleinere Schwelle benötigt. Betrachtet man aber den Puls aus Abbildung 4.6 a) und berücksichtigt eine geringe Verschiebung desselben nach links, wie sie durch die Elektronik oder die Kabel auftreten kann, so rutscht der erste Wert vor der Saturierung doch über die obere Schwelle $G_{groß}$. Damit würde der Algorithmus fälschlicherweise t_{sat} als das richtige *Bunch-Crossing* identifizieren. Um dies verhindern zu können wurde die zweite Schwelle eingeführt. Sie wird sehr niedrig angesetzt, um den Algorithmus bei Pulsen wie in Abbildung 4.5 a) nicht zu stören. Der Puls der zur falschen Identifikation führte ist von großer Energie, und daher ist seine ansteigende Flanke so steil, daß der Wert bei t_{sat-2} auch die niedrige Schwelle G_{klein} noch unterschreitet.

Tabelle 4.2 zeigt die Logiktabelle des Algorithmus. Von entscheidender Bedeutung für den Erfolg des Algorithmus ist die korrekte Wahl der Schwellen. Sie müssen auf die zu erwartenden Pulsformen optimiert werden.

$t_{sat} \geq G_{sat}$	$t_{sat-2} > G_{klein}$	$t_{sat-1} > G_{groß}$	t_{sat}	t_{sat+1}
0	X	X	0	0
1	1	1	1	0
1	0	1	0	1
1	X	0	0	1

Tabelle 4.2: Logik-Tabelle für den BCID-Algorithmus für saturierte Pulse

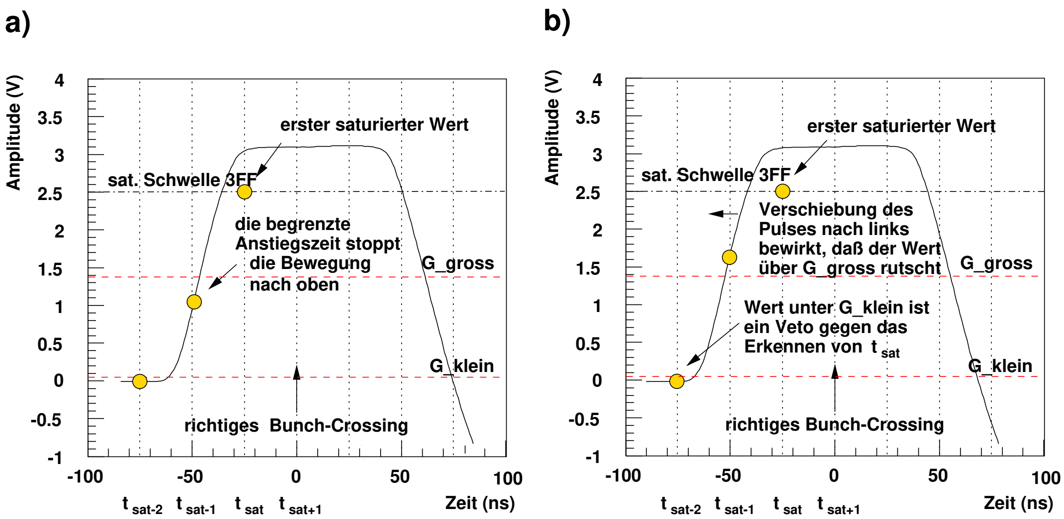

Abbildung 4.6: Der Effekt der endlichen Anstiegszeit der Ausgangsspannung der Operationsverstärker: a) ohne zusätzliche Pulsverschiebung, b) mit zusätzlicher Pulsverschiebung, das Identifizieren des falschen zugehörigen *Bunch-Crossings* wird durch die Schwelle G_{klein} verhindert [18]

Kapitel 5

Der Präprozessor-ASIC

5.1 Die Notwendigkeit eines ASICs

Die Forderung nach schnellen Algorithmen, die für jedes ankommende Kalorimetersignal seine Energie und das zeitlich dazugehörige Ereignis in möglichst kurzer Zeit und mit möglichst geringem Platzverbrauch der verarbeitenden Elektronik bestimmen sollen, macht es notwendig, die Algorithmen in Hardware aufzubauen. Hier standen zwei Möglichkeiten zur Disposition. Zum einen ist es möglich, die Algorithmen in programierbaren Logikbausteinen, sogenannten *Field Programmable Gate Arrays (FPGA)*, zu implementieren, was eine sehr große Flexibilität bietet, da man hier mögliche Veränderungen der Algorithmen leicht verwirklichen kann, indem man die Bausteine neu programmiert. Die andere Möglichkeit besteht darin, die Algorithmen und die dazugehörige Logik direkt auf einem ASIC unterzubringen. Wegen der großen Zahl (7296) von Eingangskanälen ist die Verwendung von FPGAs mit sehr hohen Kosten verbunden. Außerdem ist ein ASIC wesentlich kompakter und kann direkt an die ihn umgebenden Komponenten angepaßt werden. Aus diesen Gründen wurde die Verwendung eines selbst zu entwickelnden ASICs beschlossen. Im Folgenden werden sein struktureller Aufbau und seine Funktionsweise erläutert.

5.2 Überblick über den PPrASIC

Der Präprozessor ASIC ist das Kernstück des Präprozessor-Systems des ATLAS Level-1 Triggers. Er beinhaltet den größten Teil der schnellen Signalverarbeitung, der notwendig ist, um die digitalisierten Daten, die von den Kalorimetern kommen, für den weiteren Triggerpfad vorzubereiten. Abbildung 5.1 zeigt ein Block-Diagramm des PPrASIC. Man erkennt die Aufteilung in zwei Hauptblöcke, den BCID- und den Auslese-Block (*Readout*). Andere wichtige Teile des ASICs sind das serielle Interface, das sowohl für die Kontrolle der internen Register, als auch für die Auslese zuständig ist, die Synchronisationsstufen am Chipeingang, das *Playback*, das *Histogramming*, das *Rate-metering* und das *Bunch-Crossing-Multiplexing*. Die folgenden Abschnitte beschäftigen sich näher mit der Implementation des BCID-Blocks auf dem ASIC. Für die Beschreibung der Funktion und Arbeitsweise der anderen Module des Chips sei auf [23] verwiesen.

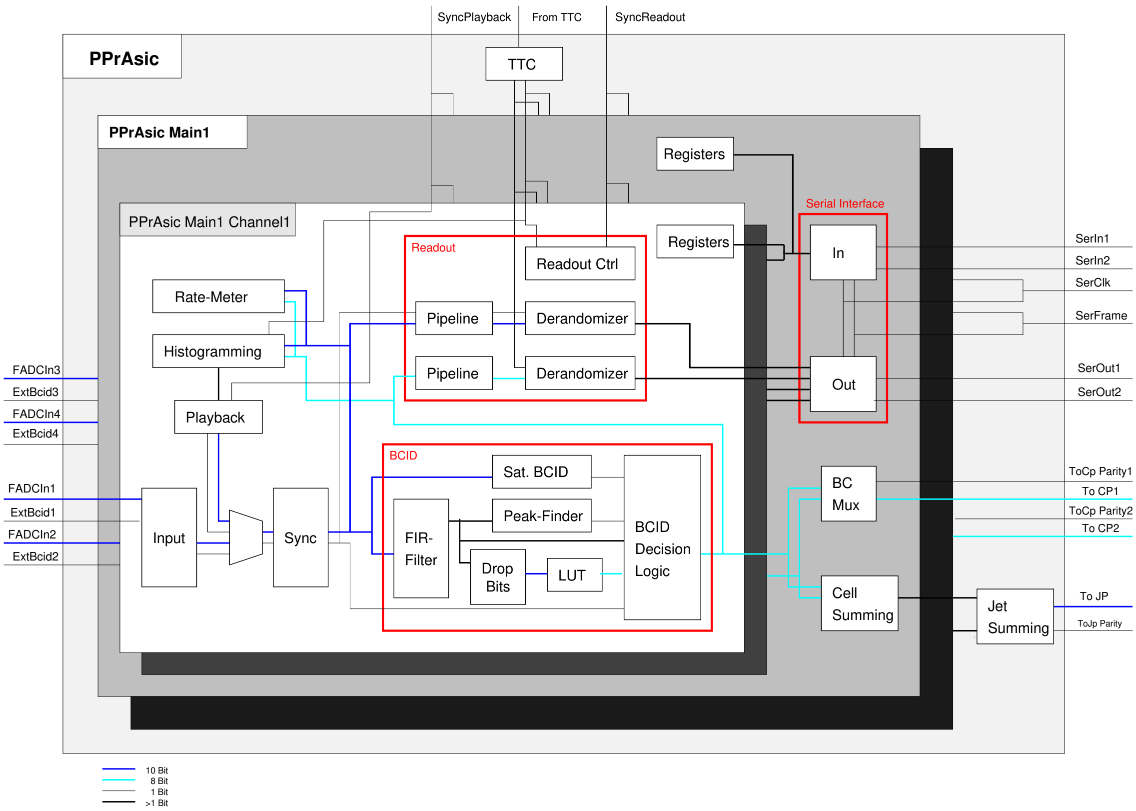

Abbildung 5.1: Block Diagramm des Präprozessor ASIC

5.3 Der Design-Prozeß

In den folgenden Abschnitten soll die Schaltungsentwicklung mit Hilfe der Hardwarebeschreibungssprache *Verilog*, dem Synthesewerkzeug *Synopsys* und dem Layoutwerkzeug *Silicon Ensemble* aus der CAD[1]-Software *Cadence* beschrieben werden. Der Designprozeß läßt sich in drei Stufen unterteilen, wie sie in Abbildung 5.2 dargestellt sind. Jede Stufe verlangt nach ihrer Durchführung eine Überprüfung ihrer Ergebnisse mittels Simulationen. Werden mittels der Simulationen Fehler erkannt, so ist es oftmals notwendig, wieder eine oder mehrere Stufen zurückzugehen, um dort die Ursachen der Fehler zu beheben.

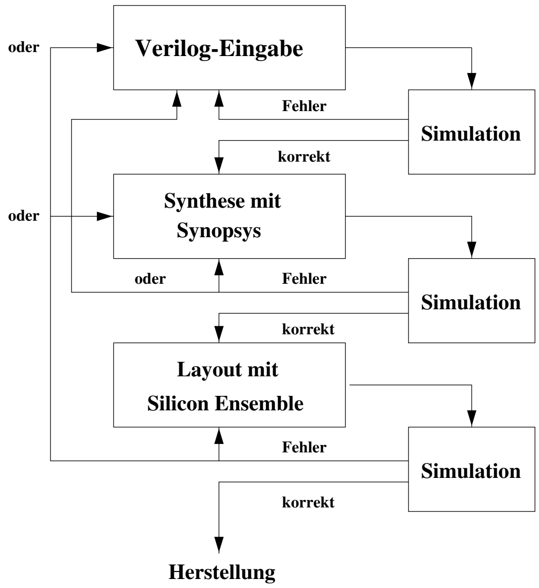

Abbildung 5.2: Der Design-Prozeß

In der ersten Stufe wird die gewünschte Schaltung auf einer abstrakten Ebene anhand ihres Verhaltens beschrieben. Hierbei werden keine konkreten Schaltungselemente, wie Inverter oder Flip-Flops, verwandt. Es wird einzig das gewünschte Verhalten in einer ausgewählten Hardwarebeschreibungssprache, hier Verilog, beschrieben. Diese Beschreibung muß allerdings die Anforderung der Synthetisierbarkeit erfüllen. In der zweiten Stufe, der sogenannten Synthese, wird das durch den Verilog-Code repräsentierte Verhalten automatisch in einen Schaltplan umgewandelt, der die Standardzellen der gewählten ASIC-Technologie benutzt. Mit Hilfe dieses Schaltplans wird dann in der dritten Stufe ein Layout generiert. Layout bedeutet hier eine graphische Darstellung der Anordnung der verschiedenen, für den Prozeß zur Verfügung stehenden, Materialien in x, und y-Koordinaten, so wie sie später auf dem fertigen Chip zu finden sind. Die z-Koordinate wird durch die Auswahl verschiedener Ebenen bestimmt. Das Layout wird aufgebaut, indem Standardzellen, die einer für die ausgewählte Technologie spezifischen Bibliothek entnommen werden, plaziert und anschließend miteinander verbunden werden. Dadurch entsteht dann die gewünschte Schaltung.

[1]Computer Aided Design

5.3.1 Schaltungsentwicklung mittels Hardware-Beschreibungssprachen

Für die Entwicklung des Schaltplans bedient man sich bei großen Schaltungen einer Hardwarebeschreibungssprache, die es erlaubt das Verhalten der Schaltung abstrakt zu beschreiben. Die im Falle des Präprozessor ASIC verwendete Hardwarebeschreibungssprache[2] war Verilog [24]. Mit ihr ist es möglich, eine Hardwarebeschreibung auf drei verschiedenen Ebenen durchzuführen. Entweder bedient man sich der funktionalen Ebene, dann beschreibt man das Verhalten der Schaltung, oder man verwendet die strukturelle Beschreibung, die direkt den Aufbau der Schaltung wiederspiegelt, indem sie Gatter instanziiert und diese miteinander verbindet. Die dazwischenliegende dritte Ebene ist die sogenannte RTL-Ebene[3]. Sie beschreibt die Schaltung anhand von Speicherelementen zwischen denen sich kombinatorische Logik befindet. Sie enthält keine Konstrukte zur Beschreibung von Signalverzögerungszeiten, wie sie in elektronischen Bauelementen auftreten.

Bei der Verwendung einer solchen Programmiersprache sollte darauf geachtet werden, daß die Beschreibung der Schaltung schon unter dem Gesichtspunkt der späteren Synthetisierbarkeit durchgeführt wird. Es ist nämlich durchaus möglich, daß sich eine bestimmte Beschreibung eines Schaltungsverhaltens auf Verilog-Code Ebene simulieren läßt und richtige Ergebnisse erzeugt, sich aber später in der Synthese als nicht synthetisierbar erweist. Bei der Verwendung von sogenannten *Zustandsmaschinen*[4], das sind synchrone Schaltungen, die sich zu jedem Zeitpunkt in einem definierten Zustand befinden, den sie mit jedem Takt wechseln können, kann eine nicht vollständige Beschreibung schnell zu Syntheseproblemen, oder sogar zu einer falschen Synthese führen.

5.3.2 Die Hardware-Beschreibungssprache Verilog

Die Sprache Verilog wurde entwickelt, um digitale Logik entwerfen, beschreiben und simulieren zu können. Sie ermöglicht es, nahezu jede digitale Schaltung mit ihrem Zeitverhalten auf einer abstrakten Ebene zu beschreiben und zu simulieren.

Im Gegensatz zu herkömmlichen prozeduralen, sequentiellen[5] Programmiersprachen muß Verilog die Parallelität von Schaltungselementen nachbilden, denn in einer echten elektronischen Schaltung warten die einzelnen verwendeten Bauelemente nicht aufeinander, sondern bearbeiten anliegende Signale sofort. Auch Schaltungselemente, die untereinander Signale austauschen, können unabhängig voneinander arbeiten.

Abgesehen von diesem grundlegenden Unterschied, sind auch einige Parallelen zu prozeduralen Programmiersprachen wie C zu erkennen. Es gibt in der Verilog-Sprache auch die typischen Sprachelemente, wie Variablen-Deklarationen (*reg, wire*), Zuweisungen (*assign, =*), Operatoren (*+,**), Schleifen (*case*), Bedingungsabfragen (*if*) und Unterprogramme (*module*). Zu diesen kommen einige Elemente, die das parallele und ereignisgesteuerte Ausführen von Anweisungen ermöglichen, sowie Konstruktionen, die spezielle Eigenschaften elektronischer Schaltungen wie Signalverzögerungszeiten nachbilden.

[2]engl. Hardware Description Language (HDL)

[3]engl. register-transfer level

[4]engl. state machines

[5]Sequentielle Programmiersprachen haben eine eindeutig definierte Reihenfolge, in der die Befehle abgearbeitet werden

In Verilog besteht eine grundsätzliche Trennung zwischen zwei Typen logischer Blöcke. Es gibt *kombinatorische* und *sequentielle* Logik-Blöcke. Kombinatorische Blöcke bestehen aus einer Verknüpfung logischer Bausteine ohne Speicherfunktion. In ihnen geschieht alles gleichzeitig, d.h. ein Schaltungselement benutzt immer das momentan an ihm anliegende Signal und ist nicht in der Lage auf das Ergebnis, eines in der logischen Kette vor ihm liegenden Schaltungselements, zu warten. In der reinen Simulation des Verilog-Codes gibt es keine Signallaufzeiten. Somit steht das Ergebnis eines Schaltungselements im selben Moment zur Verfügng, in dem das Eingangssignal angelegt wird. Simuliert man dagegen das Verilog-Modul nach der Synthese, so werden Signallaufzeiten der einzelnen Gatter berücksichtigt. Daher ist es notwendig, schon in der Verilog-Beschreibung einer gewünschten Schaltung an das spätere Syntheseergebnis zu denken. Sequentielle Blöcke sind aus logischen Bausteinen mit einer Speicherfunktion aufgebaut. In diesen werden Signale nur zu bestimmten Zeiten an die nachfolgende Logik weitergegeben.

5.3.3 Das Synthesewerkzeug Synopsys

Um von der Verilog-Beschreibung zu einem Schaltplan zu gelangen, ist es notwendig, einen Syntheseschritt durchzuführen. In diesem Schritt werden die Progammbefehle in Schaltungselemente übersetzt. Diese Schaltungselemente werden einer Bibliothek entnommen, die im Hinblick auf den gewünschten Herstellungsprozeß ausgewählt wird. Eine solche Bibliothek enthält kombinatorische Bausteine, wie AND- und OR-Gatter, sowie sequentielle Logikbausteine, wie *Flip-Flops*. Zu jedem dieser Bausteine sind in der Bibliothek Angaben über intrinsische Laufzeiten, sowie Eingangskapazitäten und Ausgangstreiberstärken gespeichert. Aus diesen Angaben werden dann die Signallaufzeiten berechnet.

Um zu zufriedenstellenden Ergebnissen zu gelangen, ist es notwendig, einige grundlegende Dinge schon bei der Beschreibung der Schaltung zu beachten. Zum einen empfiehlt es sich, die Schaltung in einzelne Module, die jeweils ein spezielles Verhalten repräsentieren, aufzuteilen. Auf diese Weise ist es leichter durch die Synthese auftretende Fehler auf die Code-Ebene zurückzuverfolgen und zu beheben. Zum anderen ist es unbedingt erforderlich, die Schaltung vollständig zu beschreiben. Dies bedeutet, daß jeder Fall, der auftreten kann, genau eine Reaktion der Schaltung nach sich zieht.

5.3.4 Layout mit Silicon-Ensemble

Um aus dem Schaltplan ein Layout zu erhalten, das dann direkt zur Herstellung von Chips verwendet werden kann, ist es notwendig, ein Layoutwerkzeug zu Hilfe zu nehmen. Ein Beispiel für ein solches Werkzeug, das auch zur Entwicklung des PPrASICs verwandt wurde, ist das sogenannte *Silicon Ensemble*, ein Programm, das zum CAD-Paket *Cadence* gehört. Mit ihm lassen sich aus Schaltplänen Chip-Layouts generieren. Dafür werden einer Bibliothek, die speziell für die ausgewählte Technologie und den Prozeß angelegt wurde, Standardzellen entnommen und mit diesen ein Design aufgebaut. Eventuell selbst entworfene Blöcke, bzw. vorgefertigte Speicherblöcke, können in das Layout eingefügt werden. Anschließend werden die verwendeten Zellen elektrisch miteinander verbunden. Diese Arbeit kann von einem automatischen Werkzeug, dem sogenannten *auto-router*, durchgeführt werden, meistens ist jedoch eine gewisse Menge an Handarbeit nicht zu umgehen.

5.4 Flächenanalysen der Multiplizierer des FIR-Filters

In den Vorstudien für den PPrASIC wurden auch Überlegungen über die Bit-Breite der Koeffizienten des FIR-Filters angestellt. Es galt zu analysieren, wie die Größe des Designs von der Bit-Breite abhängt. Dafür wurden die flächenintensivsten Bausteine des FIR-Filters, die Multiplikatoren untersucht. Anhand dieser Untersuchungen wird im folgenden der komplette Design-Prozeß exemplarisch vorgestellt.

5.4.1 Das Verilog-Modul

Zuerst muß ein Verilog-Modul geschrieben werden, das die Funktionalität beschreibt:

```
// $Id: Multiplier4.v,
//
// Multiplier
//
module Multiplier4 (MultiOut,TenIn,FourIn);

output [13:0] MultiOut;

input [9:0] TenIn;
input [3:0] FourIn;

reg [13:0] MultiOut;

  always@(TenIn or FourIn)

     MultiOut = TenIn*FourIn;

endmodule
```

Das Modul wird mit Hilfe des VerilogXL-Interpreters simuliert. Dafür ist es notwendig ein *Stimulus-File* zu schreiben, welches die Testvektoren für die Simulation generiert:

```
// $Id testMultiplier4.v
//
// Test module for Multiplier4.v
//
`timescale 1 ns / 100 ps

module testMultiplier4 ();

  reg [9:0]  TenIn;
```

```
  reg [3:0]  FourIn;
  reg [14:0] Result;
  reg [4:0]  a;
  reg [10:0] b;
  wire [13:0] MultiOut;

  Multiplier4 MULT ( MultiOut, TenIn, FourIn );

// Stimulus
initial
    begin
    TenIn=-1;
    FourIn=-1;
    for (a=0 ; a<17 ; a=a+1)
      begin
        FourIn=FourIn+1;
        for (b=0 ; b<1025 ; b=b+1)
          begin
           TenIn=TenIn+1;
           #35
           Result = TenIn * FourIn;
           if (Result != MultiOut)
             begin
              $display("Output  ",MultiOut,"  FourIn  ",FourIn,
               "  TenIn  ",TenIn);
              $display("Falsch");
             end
          end
      end
    end

// Write data for waveform display
  initial begin
    $shm_open("waves.shm");
    $shm_probe("AS");
  end

endmodule
```

Im vorliegenden Fall ist es möglich, alle Kombinationen von Eingangssignalen zu simulieren und mit dem gewünschten Ergebnis zu vergleichen. Im Regelfall eines kompletten ASIC-Designs ist dies jedoch nicht mehr machbar, daher hängt in diesem Fall die Qualität der Simulationen unmittelbar mit der Wahl der Testvektoren zusammen. Der zeitliche Verlauf aller Signale wird in ein File (*waves.shm*) geschrieben und kann mit Hilfe eines Programms, dem sogenannten *waveform display*, angeschaut werden. Dieses Programm bietet die Möglichkeit, sich den zeitlichen Verlauf jedes verwendeten Signals graphisch anzusehen. Damit lassen sich eventuelle Fehler leicht zu ihrem Ausgangspunkt zurückverfolgen.

5.4.2 Die Synthese des Moduls

Nach erfolgreicher Simulation des Verilog-Codes wird ein Schaltplan generiert (siehe Abbildung 5.3). Dafür steht der *design_analyzer* von Synopsys zur Verfügung. Mit seiner Hilfe wird zuerst das Verilog-Modul eingelesen. Danach muß eine *library* angegeben werden, die die verfügbaren Zellen des gewünschten Prozesses enthält. Jetzt kann das Design synthetisiert werden und so entsteht der Schaltplan.

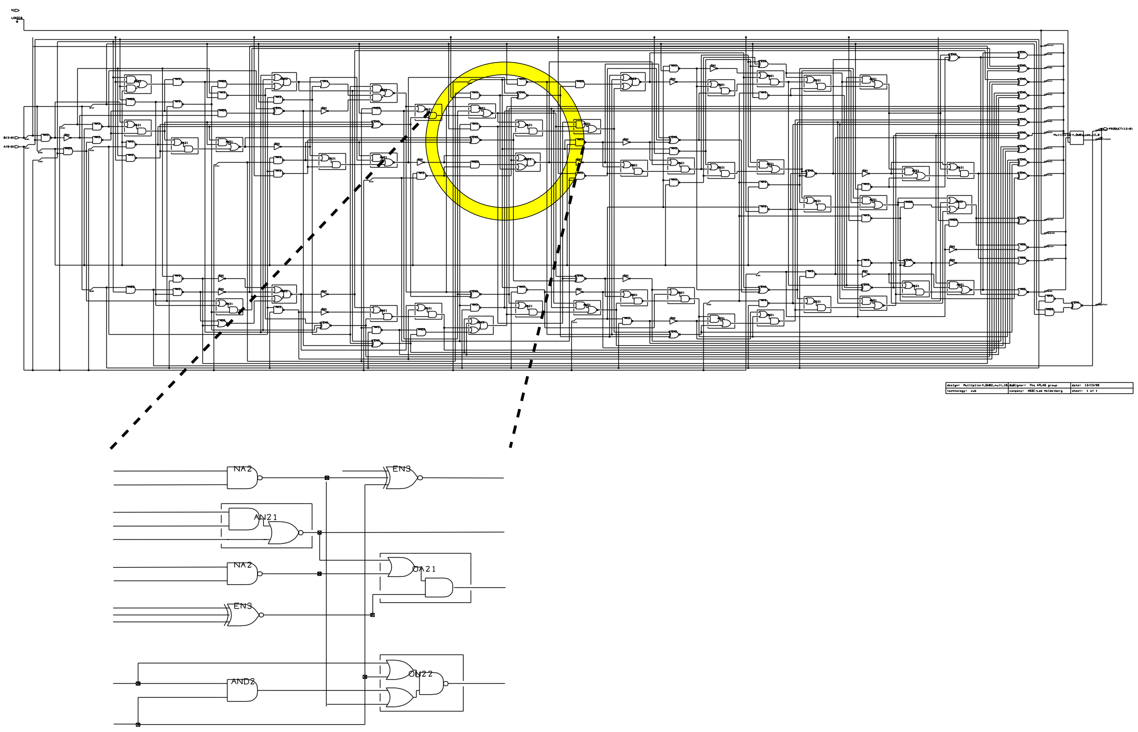

Abbildung 5.3: Schaltplan eines 4 x 10 Bit Multiplizierers

Diesen Schaltplan kann man auf die Zieltechnologie abbilden und man erhält dann ein File, das die Information über alle verwendeten Zellen und ihre Verknüpfung untereinander beinhaltet, die sogenannte Verilog-Netzliste. Diese wiederum läßt sich mit demselben *Stimulus-File* testen, das schon für die Verilog-Beschreibung verwandt wurde.

5.4.3 Die Erstellung des Layouts

Nachdem die Korrektheit der Netzliste festgestellt wurde, kann nun aus ihr ein Layout generiert werden. Dazu bedient man sich des Programm-Paketes *Cadence*. Zuerst wird die Netzliste importiert, wobei schon eine Bibliothek ausgewählt sein muß, die sogenannte *Standardzellen*, d.h. schon fertig gelayoutete Blöcke, die eine bestimmte Funktionalität aufweisen (AND, NOR, usw.), enthält. Nachdem alle Standardzellen, die notwendig sind, um den Schaltplan nachzubilden, eingelesen wurden, beginnt die eigentliche Layouttätigkeit. Zuerst müssen die Standardzellen plaziert werden. Diese Plazierung hat natürlich Einfluß auf die Größe des Designs, da durch sie die Länge der Verbindungen zwischen den einzelnen Standardzellen festgelegt wird. Außerdem können durch eine ungünstige Plazierung einzelne Pfade des Designs unnötig lang, und somit langsam werden, was die Funktionalität der gesamten Schaltung beeinträchtigen kann. Aus diesen Gründen kommt der richtigen Plazierung eine große Bedeutung zu. Da schon vermeintlich kleine Designs aus einer Vielzahl von Standardzellen aufgebaut sind, verwendet man ein Programm, das die Zellen automatisch in einer günstigen Art und Weise anordnet, den sogenannten *placer*. Danach werden die Zellen mittels eines Werkzeugs, dem sogenannten *auto-router*, automatisch miteinander verbunden.

5.4.4 Die Flächenabhängigkeit der Multiplizierer von der Bitbreite ihrer Koeffizienten

Um die Flächenabhängigkeit des Layouts eines Multiplizierers von der Bitbreite seiner Koeffizienten bestimmen zu können, wurden Layouts von 2x10-bit, 3x10-bit, 4x10-bit, 5x10-bit, 6x10-bit und 8x10-bit Multiplizierern wie oben beschrieben generiert. Dafür wurden Standardzellen des 0.6 μ-Prozeß der Firma AMS[6] verwendet. Dann wurden die jeweiligen Flächen der Layouts über die Bit-Breite der Koeffizienten aufgetragen (Abbildung 5.4). Außerdem wurden die Flächen, die der *design_analyzer* angegeben hat, mit in die Graphik aufgenommen.

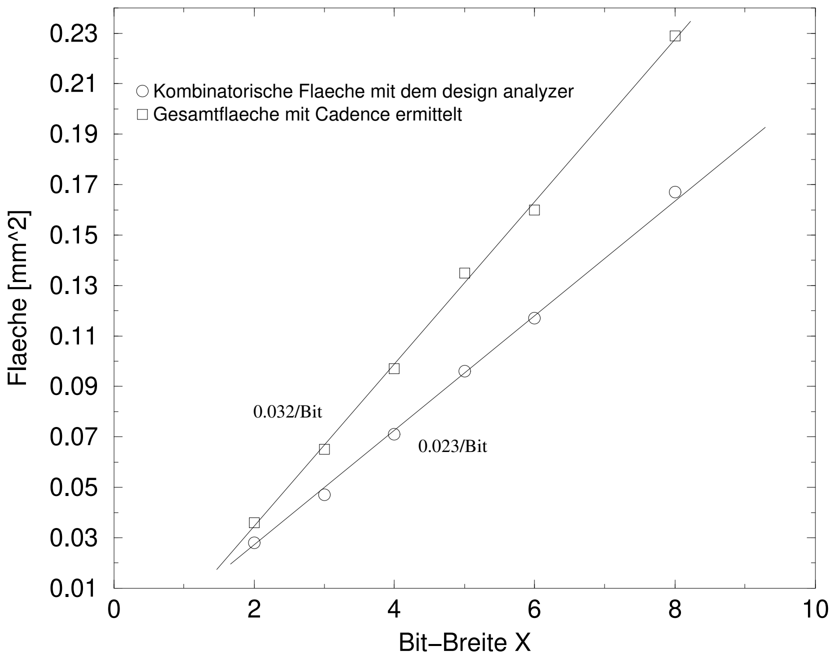

Abbildung 5.4: Simulierte Chipflächen von 10 x X-Bit Multiplizierern

[6] Austrian Micro-Systems

Wie man sieht, ist das Anstiegsverhalten der beiden Kurven gut linear. In der unteren Geraden ist nur die reine Fläche der Standardzellen dargestellt. In der von *Cadence* ermittelten Gesamtfläche des Layouts ist neben der Fläche, die die Standardzellen benötigen, auch noch eine Fläche vorhanden, die den Platz, der für die elektrische Verdrahtung der Standardzellen benötigt wird, wiederspiegelt. Diese Fläche steigt mit zunehmender Größe des Layouts an. Es ist auch zu erwarten, daß für mehr Standardzellen auch mehr verbindende Leitungen und somit mehr Fläche notwendig ist. Betrachtet man das Verhältnis von Gesamtfläche zur Fläche der Standardzellen, so stellt man fest, daß dieses um einen konstant bleibenden Mittelwert gruppiert ist. Dieser liegt bei etwa 1,36, d.h. etwa ein Viertel der Gesamtgröße des Design wird für die Verdrahtung der Standardzellen gebraucht.

Kapitel 6

Die Implementation des BCID-Blocks

6.1 Struktur der Implementation

Die Implementation des BCID-Blocks weist eine hierarchische Ordnung auf. Abbildung 6.1 zeigt einen Überblick über die Struktur des BCID-Blocks. Jedem Kasten entspricht ein Modul, dessen Verilog-Code eine bestimmte Funktionsweise beschreibt. Der hierarchische Aufbau ermöglicht eine leichtere Verfolgung der einzelnen Signale durch das Design. Im folgenden werden die Aufgaben der einzelnen Module erläutert.

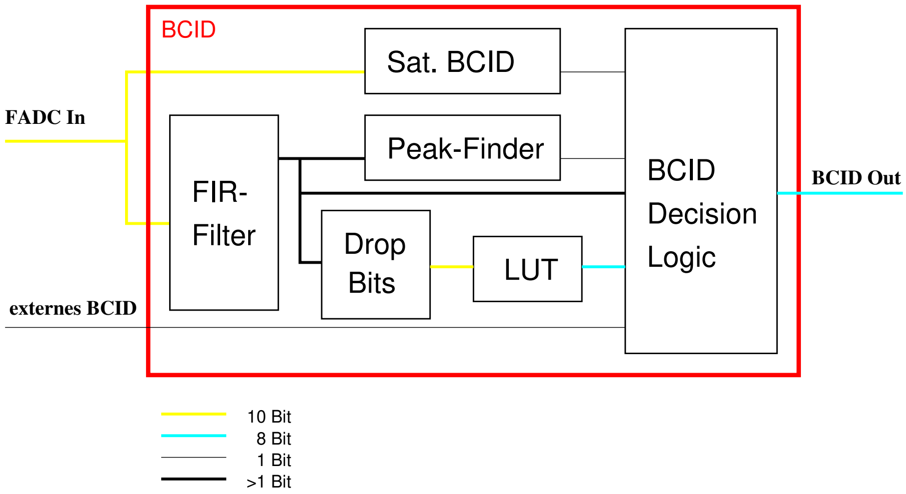

Abbildung 6.1: Der BCID-Block des PPrAsic

6.1.1 Die Implementation für saturierte Pulse

Der Algorithmus für saturierte Pulse wird in der Verilog-Implementation durch das (*Sat. BCID*)-Modul repräsentiert. Die Aufgabe dieses Moduls ist die richtige Bestimmung der

Bunch-Crossing Zugehörigkeit für saturierte Pulse. Die Struktur der dafür implementierten Logik zeigt Abbildung 6.2.

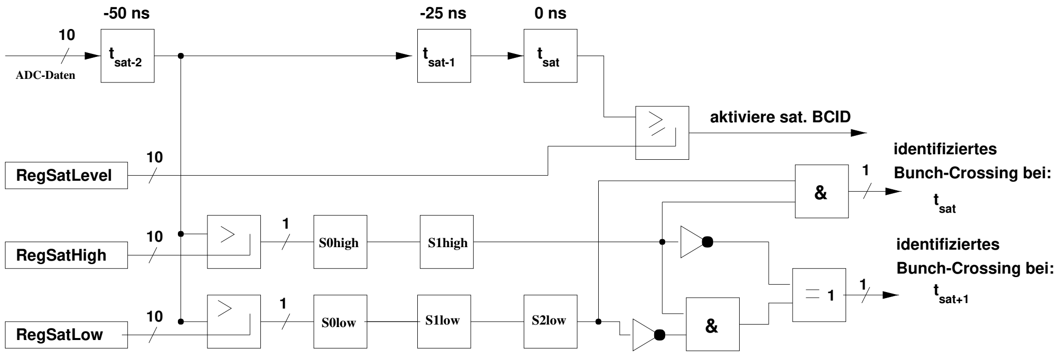

Abbildung 6.2: Die Implementation des saturierten Algorithmus

Der Algorithmus benutzt drei programmierbare Register:
→ *RegSatLevel*
→ *RegSatHigh*
→ *RegSatLow*

Das Register *RegSatLevel* setzt eine Schwelle für die Saturierung, d.h. ankommende Pulswerte, die diesen Wert erreichen oder überschreiten, aktivieren den Algorithmus. Mit Hilfe dieser Schwelle ist es somit möglich, die Schwelle für die Saturierung, die normalerweise auf dem größtmöglichen Wert, also 255, liegt, künstlich herabzusetzen. Diese Option könnte dazu benutzt werden, den Überlappbereich der Algorithmen für saturierte und nicht saturierte Pulse auszudehnen. Nach seiner Aktivierung wird der Algorithmus einmal ausgeführt und erst durch das Unterschreiten der Schwelle durch einen nachfolgenden Wert wird er wieder in einen Zustand versetzt, aus dem heraus er ein weiteres Mal aktiviert werden kann.

Der aktivierte Algorithmus analysiert die steigende Flanke des Pulses, um auf diese Weise den Ursprung des Pulses zu ermitteln. Denn dieser ist der einzige feste Anhaltspunkt, von dem aus das Pulsmaximum errechnet werden kann. Der Algorithmus vergleicht daher die beiden Werte, die vor dem saturierten Wert angekommen sind, mit vorbestimmten Schwellen.

6.1.2 Die FIR-Filter Realisierung

Der Algorithmus für nicht saturierte Pulse setzt sich aus zwei Modulen zusammen. Das erste Modul ist der FIR-Filter. Er ist dafür zuständig, das Rauschen abzumildern und die Pulse dahingehend zu verändern, daß der nachfolgende Peak-Finder keine Schwierigkeiten bei der Feststellung der Position des Pulsmaximums hat.

Der FIR-Filter besteht der Idee nach aus einer Pipeline von 5 aufeinanderfolgenden Registern, in denen die ankommenden 10-Bit breiten Werte gespeichert werden. Anschliessend sollen sie dann mit fünf 4-Bit breiten, vorher bestimmten, Koeffizienten multipliziert und zu einem 16-Bit breiten Ausgangssignal aufsummiert werden (siehe Abbildung 6.3).

Die fünf Koeffizienten können von außen über die serielle Schnittstelle in den Chip geladen werden. Sie müssen sorgfältig ausgewählt und auf die Pulsform optimiert werden. Angepaßt an die Aufgaben des FIR-Filters, sind die 4 Bit der Koeffizienten unterschiedlich aufgeteilt. Die drei mittleren Koeffizienten können nur positive Werte annehmen und erstrecken sich daher von 0 bis 15. Der erste und der fünfte Koeffizient kann sowohl als positive, als auch als negative Zahl interpretiert werden. Dies geschieht in der Zweierkomplementdarstellung mit der Einschränkung, daß der Wert -8 als 0 interpretiert wird, und daher der Wertebereich nur von -7 bis +7 reicht. Dies liegt darin begründet, daß die Multiplikation in allen Fällen auf zwei Taktzyklen verteilt wird. Bei den Koeffizienten 2-4 werden im ersten Taktzyklus zwei 2x10-Bit Multiplikationen durchgeführt und diese dann im nachfolgenden aufsummiert. Bei den Koeffizienten 1 und 5 verarbeitet der erste Taktzyklus eine 3x10-Bit Multiplikation und im zweiten werden die Ergebnisse im Falle eines negativen Koeffizienten dann in ihre Zweierkomplemente umgewandelt.

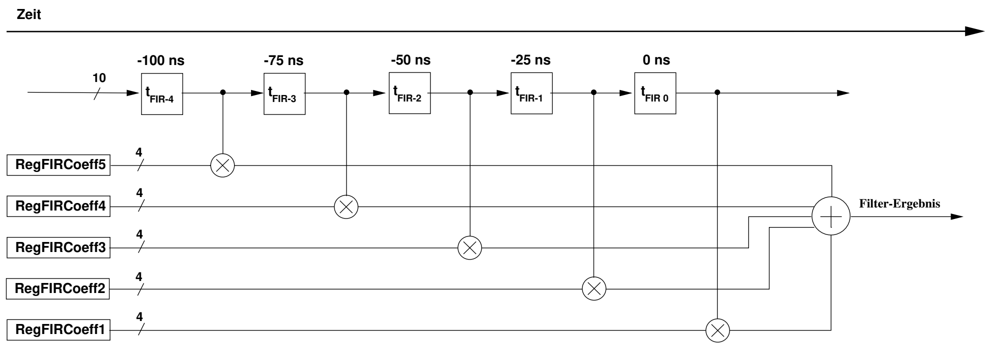

Abbildung 6.3: Die logische Arbeitsweise des FIR-Filters

Die Anforderung an den Trigger, seine Aufgaben in möglichst wenig *Takt-Zyklen* zu bewältigen, d.h. mit einer möglichst geringen *Latenzzeit*[1], gilt natürlich auch hier. Da der FIR-Filter mit seinen 10 x 4-Bit Multiplikationen und der anschließenden 4-stufigen Addition zu den zeitkritischsten Komponenten auf dem ASIC gehört, wurde hier nicht nach Fläche bzw. Gattern optimiert, sondern ein funktioneller Aufbau überlegt, der die Operation mit möglichst geringem logischen Aufwand durchführen kann und daher schnell ist. Die Implementation eines FIR-Filters auf dem direkten Weg, wie sie in Abbildung 6.3 zu sehen ist, speichert zuerst die ankommenden Daten in einem Schieberegister. Bei fünf Koeffizienten benötigt man daher zwei weitere Taktzyklen nach dem zu berechnenden Datum, damit alle für die Operation benötigten Daten vorliegen. In diesen zwei Taktzyklen geschieht nichts, außer dem Weiterwandern der Daten im Schieberegister, d.h die 25 ns, die für einen Takt-Zyklus zur Verfügung stehen, werden nicht ausgenutzt. Danach werden die fünf Werte mit ihren Koeffizienten multipliziert und anschließend die Zwischenergebnisse aufsummiert.

Um die notwendigen logischen Operationen günstiger auf die Gesamtzahl der benötigten Taktzyklen zu verteilen, wurde eine strukturell neue Implementation des FIR-Filters überlegt. In dieser neuen Implementation werden die Daten nicht erst in ein Schiebere-

[1] engl. latency

gister geführt, sondern werden sofort einer logischen Verarbeitung zugeführt. Im Gegensatz zur früheren Idee wird jedes ankommende Datum sofort mit den fünf Koeffizienten multipliziert. Die Ergebnisse werden anschließend mit denen vom vorherigen Taktzyklus verknüpft und zwischengespeichert. Die Verknüpfung gestaltet sich folgendermaßen:

- Datum x Koeffizient 5 $\rightarrow$ Das Ergebnis wird in einem Register *FilterInReg5* gespeichert.
- Datum x Koeffizient 4 $\rightarrow$ Zum Ergebnis wird der Inhalt des *FilterInReg5*-Registers, das im vorigen Taktzyklus gefüllt wurde, addiert, und das Ganze wird dann im Register *FilterInReg4* gespeichert.
- Datum x Koeffizient 3 $\rightarrow$ Zum Ergebnis wird der Inhalt des *FilterInReg4*-Registers, das im vorigen Taktzyklus gefüllt wurde, addiert, und das Ganze wird dann im Register *FilterInReg3* gespeichert.
- Datum x Koeffizient 2 $\rightarrow$ Zum Ergebnis wird der Inhalt des *FilterInReg3*-Registers, das im vorigen Taktzyklus gefüllt wurde, addiert, und das Ganze wird dann im Register *FilterInReg2* gespeichert.
- Datum x Koeffizient 1 $\rightarrow$ Zum Ergebnis wird der Inhalt des *FilterInReg2*-Registers, das im vorigen Taktzyklus gefüllt wurde, addiert. Dieses Resultat stellt dann das FIR-Filter Ergebnis dar.

Der strukturelle Aufbau dieser Implementation wird in Abbildung 6.4 gezeigt.

Der Vorteil dieser Implementation liegt darin, daß die vier Schritte der Addition der Ergebnisse aus den fünf Multiplikationen, nicht alle an einem Stück erfolgen müssen, sondern auf mehrere Taktzyklen aufgeteilt werden, wobei man auch diejenigen Taktzyklen ausnutzt, die vorher allein dafür benutzt wurden, die Daten in der Pipeline weiterzuschieben, um gleichzeitig die Daten von fünf zeitlich aufeinanderfolgenden Bunch-Crossings zur Verfügung zu haben. Für die vierstufige Addition, wie sie in der ersten FIR-Filter Implementation erfolgt, benötigte man einen ganzen Taktzyklus. In der neuen optimierten Version ist es möglich, die letzte Stufe der Addition mit der zweiten Hälfte der Multiplikation in einem Taktzyklus durchzuführen. Damit kann ein Taktzyklus für den FIR-Filter Algorithmus eingespart werden. Abbildung 6.5 zeigt den Vergleich der Latenzzeiten der beiden Implementationen.

6.1.3 Die Implementation des Peak-Finders

Der zweite Teil des Algorithmus für nicht saturierte Pulse wird durch den sogenannten *Peak-Finder* gebildet. Seine Aufgabe besteht darin herauszufinden, in welchem Bunch-Crossing das Maximum der durch den FIR-Filter vorverarbeiteten Pulse liegt. Dafür vergleicht er jeden ankommenden Wert mit seinem zeitlichen Vorgänger und Nachfolger. Für diesen Vergleich kann von außen eine von zwei Bedingungen ausgewählt werden:

$$bc_{t_{-1}} < bc_{t_0} \geq bc_{t_1}$$
$$\text{oder}$$
$$bc_{t_{-1}} < bc_{t_0} > bc_{t_1}.$$

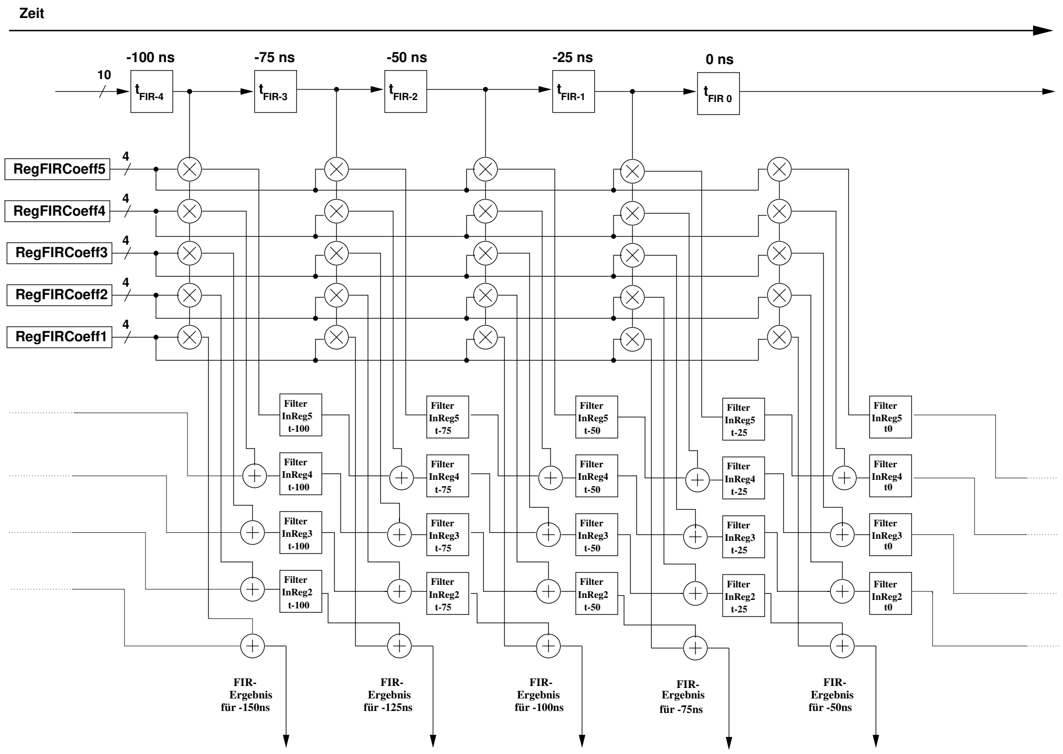

Abbildung 6.4: Die geschwindigkeits-optimierte Implementation des FIR-Filters

Wird die ausgewählte Bedingung erfüllt, so geht das Ausgangsbit des Peak-Finders, das ein gefundenes Pulsmaximum signalisiert, auf Eins, ansonsten geht es auf Null.

6.1.4 Die Look-Up-Table

Die Look-Up-Table, die für die Feinkalibrierung der Energie zuständig ist, besteht aus einem Speicherblock mit 1024 Speicherzellen. In jeder dieser Zellen ist ein 8-Bit Wert gespeichert, der auf den Ausgang gegeben wird, falls der Eingangswert der Addressierung der Speicherzelle entspricht. Auf dem PPrAsic ist dieser Speicherblock durch einen von AMS[2] vorgefertigten Block realisiert.

6.1.5 Das Bindeglied zwischen FIR-Filter und LUT

Da der in der Look-Up-Table verwendete Speicherblock 1024 Zellen besitzt, und somit nur Eingangswerte bis zu einer Breite von maximal 10 Bit verarbeiten kann, die Ausgangswerte des FIR-Filters jedoch eine Breite von 16 Bit haben, ist es notwendig, ein Bindeglied zu verwenden, welches die 16-Bit breiten Werte auf 10 Bit reduziert. Dieses Bindeglied ist in der Abbildung 6.1 als *Drop-Bits*-Block bezeichnet. Die Reduktion der Bitbreite kann auf drei unterschiedliche Weisen geschehen:

→ Die obersten 6 Bit werden abgeschnitten.

[2]austrian micro-systems

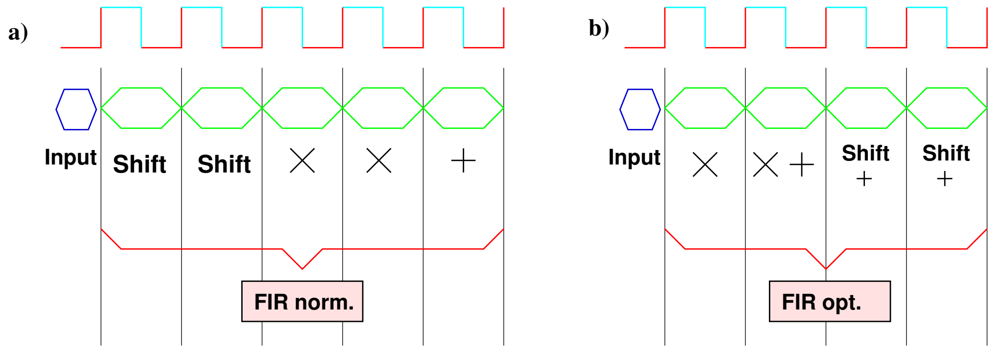

Abbildung 6.5: Die Latenzzeiten der beiden FIR-Filter Implementationen: a) die nicht geschwindigkeits-optimierte Version; b) die optimierte Version

→ Die untersten 6 Bit werden abgeschnitten.

→ Es werden sowohl unten als auch oben Bits abgeschnitten.

Das Abschneiden der obersten Bits kann die Daten verfälschen und bewirken, daß bestimmte hohe Energien ganz verschwinden. Dies kann unter keinen Umständen sinnvoll sein, da der Trigger gerade auf hohe Energien ansprechen soll.

Schneidet man die untersten Bits ab, so vermindert man die Auflösung der Werte, die aus dem FIR-Filter kommen und verliert niedrige Energien.

Wie schon erwähnt, ist der Verlust von hohen Energien nicht akzeptabel. Die Ausgangsbreite von 16 Bit des FIR-Filters ist jedoch dadurch bestimmt, wie groß die Werte werden können, die in diesem entstehen. Für die meisten Konfigurationen der Koeffizienten können keine Werte entstehen, die die vollen 16 Bit ausfüllen. So liegt beispielsweise der Maximalwert, der bei einer Konfiguration der Koeffizienten von -1, 0, 5, 0, -1 entstehen kann, bei $1023 * 5 = 5115$. Dieser Wert braucht in der Binärdarstellung nicht mehr als 13 Bit (8192), und daher kann man bei einer solchen Einstellung des FIR-Filters die obersten 3 Bit abschneiden ohne Daten zu verlieren oder sie zu verfälschen.

Aus diesem Grund wurde die dritte Lösung gewählt, bei der sowohl oben, als auch unten Bits wegeschnitten werden. Um diese Option mit maximaler Effektivität nutzen zu können, ist es notwendig für jede Konfiguration an FIR-Filter Koeffizienten, die man wählt, denjenigen Wert zu berechnen, der maximal auftreten kann. Dies muß nicht auf dem Chip geschehen, da die Koeffizienten auch nicht intern geändert werden. Zu jedem Satz Koeffizienten wird ein dazugehöriges Bit, das sogenannte *Startbit*, in den Chip geladen. Dieses Startbit bezeichnet dasjenige Bit, von dem an das 10-Bit Datenwort aus den 16 Bit ausgeschnitten wird (siehe Abbildung 6.6).

6.1.6 Das Entscheidungsmodul

Ein weiteres Modul des BCID-Blocks ist die sogenannte *BCID Decision-Logic*. Sie analysiert die Ergebnisse der einzelnen Algorithmen und entscheidet, wo sich die Pulsmaxima befinden. Diese Analyse beruht auf den Ergebnissen der Algorithmen und auf den Pulsenergien, die durch die FIR-Filter-Ausgangswerte repräsentiert werden. Damit ist der Energiebereich auf 16-Bit Breite festgelegt. Dieser wird nun mittels zweier Schwellen

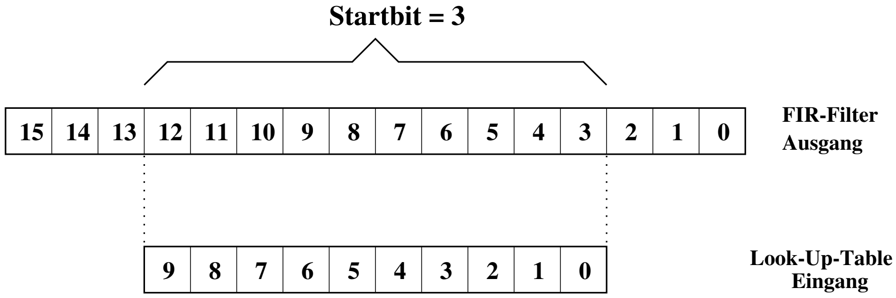

Abbildung 6.6: Der Mechanismus der Bitreduktion von 16 Bit FIR-Filterausgang auf 10 Bit Look-Up-Table Eingang

(*RegEnergyLevelLow, RegEnergyLevelHigh*), die von außen gesetzt werden können, in drei Bereiche unterteilt. Die beiden Schwellen haben eine Auflösung von 10 Bit, und müssen daher zuerst auf 16 Bit erweitert werden. Dies geschieht durch einfaches Anhängen von 6 auf Null gesetzten Bits an das untere Ende der Werte. Die eine Schwelle dient als untere Grenze für den obersten Bereich und als obere für den mittleren Bereich. Die andere hat die Funktion einer oberen Grenze für den unteren und einer unteren Grenze für den mittleren Bereich. Für jeden dieser Bereiche wird nun definiert, bei welcher Kombination an Ergebnissen der einzelnen Algorithmen ein Pulsmaximum als entdeckt gilt und bei welcher nicht. Drei Algorithmen ergeben 8 unterschiedliche Kombinationen, und so repräsentiert eine 8-Bit breite Zahl (*RegBcidDecision*) die Antworten auf alle Kombinationen. Tabelle 6.1 zeigt wie sich diese Zahl zusammensetzt. Es gibt für jeden Bereich eine solche Zahl, die von außen gesetzt wird.

BCID nicht saturiert	**BCID saturiert**	**externes BCID**	**Bit Nr.**
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

Tabelle 6.1: Die Tabelle zeigt für jede mögliche Kombination der Ergebnisse der Algorithmen das dazugehörige Bit in der *RegBcidDecision*-Zahl

Der Weg, auf dem das Modul zu einer Entscheidung gelangt, ist folgendermaßen gegliedert:

- Die Ergebnisse der Algorithmen werden zu einer Zahl zusammengefaßt, die angibt, welches Bit aus der *RegBcidDecision*-Zahl für die Entscheidung herangezogen werden muß. Ergeben z.B. die beiden internen Algorithmen eine 1, und der externe eine 0, so wird das 6. Bit zur Entscheidung herangezogen.

- Der FIR-Filter Ausgangswert wird mit den beiden Schwellen verglichen, und so der Energiebereich, in den der Wert gehört, festgelegt.
- Danach wird das entsprechende Bit der zu diesem Energiebereich gehörenden *Reg-BcidDecision*-Zahl eingelesen. Ist dieses Bit eine 1 wird eine positive Entscheidung getroffen, ansonsten eine negative.
- Im Falle einer positiven Entscheidung wird der zu dem eingelesenen FIR-Filter Ausgangswert gehörende Ausgangswert der Look-Up-Table nach außen gegeben, ansonsten eine Null.

6.2 Funktionale Tests mit dem Verilog-Interpreter Verilog-XL

Alle Module wurden einzeln auf ihre korrekte Funktionsweise getestet. Dafür wurden Testprogramme verwandt, wie sie in 5.4 beschrieben wurden. Bei manchen Modulen, wie z.B. dem *Drop-Bits*-Block, war es noch möglich, alle möglichen Einstellungen (7) mit allen möglichen Eingangswerten (2^{16}) zu testen ($2^{16} * 7 = 458.752$). Bei dem FIR-Filter stößt man dagegen schon auf Werte, die einen vollständigen Test nicht mehr sinnvoll erscheinen lassen ($2^{10} * 2^{10} * 2^{10} * 2^{10} * 2^{10} * 16^5 > 10^{21}$). In diesen Fällen wurden die realistisch zu erwartenden Konfigurationen (-1)-0-4-0-(-1) mit realistisch zu erwartenden Pulsen simuliert. Außerdem wurden Extremfälle, wie z.B. 7-15-15-15-7 oder (-7)-0-0-0-(-7), und eine große Anzahl von zufällig ausgewählten Konfigurationen getestet. Dies geschah teilweise mit Hilfe von automatischen Programmen, die die Ergebnisse der Simulation des Verilog-Codes mit dem Verilog-XL Interpreter mit denen, die aus der Ptolemy-Simulation des Verilog-Codes entstanden sind, verglichen. Eine genauere Erläuterung der verwendeten automatischen Simulationsprogramme findet sich im *PPrAsic Design Guide* [22].

6.2.1 Analyse mit dem Programm Sim-Wave

Eine weitere Methode die Module zu testen und gefundene Fehler zu lokalisieren, besteht in der Analyse der Ergebnisse mittels eines Hilfsprogramms. Dieses Programm (*Sim-Wave*) bietet die Möglichkeit, sich den internen Signalverlauf anzusehen. Dies geschieht mit Hilfe eines sogenannten *waveform-displays*, das für jedes Signal oder für Gruppen von Signalen (Busse), seinen zeitlichen Verlauf anzeigt. Abbildung 6.7 zeigt eine Bildschirmansicht dieses Programms am Beispiel des FIR-Filters.

Man erkennt als Eingangssignale in den FIR-Filter die 40 MHz *Clock*, die die sequentiellen Bauteile steuert, das *FilterIn*-Signal, das die ankommenden Testdaten repräsentiert, und die Koeffizienten des FIR-FIlters (*RegFIRCoeff1-5*). Als Zwischenergebnis, d.h. als internes Signal des Moduls wird das Signal *FIRCoeff3Out* angezeigt. Es zeigt das Ergebnis der Multiplikation der Eingangsdaten mit dem 3. Koeffizienten an. Als Endergebnis werden zwei Signale dargestellt. Das eine ist das Ergebnis der Implementation des FIR-Filters (*FilterOut*), und das andere ist das Ergebnis einer Vergleichsoperation, die direkt im Testmodul durchgeführt wurde (*CorrectOut2*). Wie man sieht, stimmen beide Ergebnisse überein.

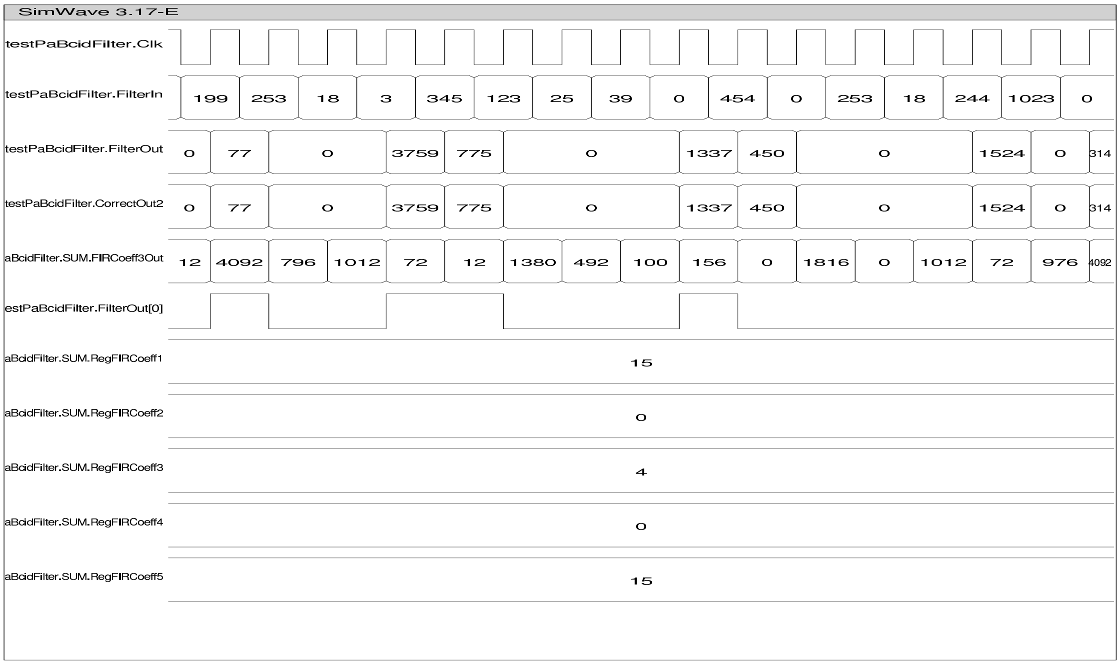

Abbildung 6.7: Zeitliche Struktur einiger FIR-Filter Signale

6.3 Synthese und Layout

Neben dem BCID-Block mußten noch andere Funktionsblöcke des PPrAsic, wie z.B Readout und Histogramming, in Verilog-Code verwirklicht und anschließend simuliert werden. Danach konnte der komplette Chip synthetisiert werden.

Im folgenden werden einige technischen Daten und Design-Parameter der finalen Version der PPrAsics vorgestellt. Für detailliertere Informationen sei auf das PPrAsic-Manual verwiesen [23].

- Die Anzahl der Kanäle, die pro PPrAsic verarbeitet werden können, beträgt vier.
- Die vier Kanäle sind in zwei Gruppen von jeweils 2 Kanälen unterteilt. Es gibt zwei serielle *Interfaces*. Dies ist vorgesehen, um die Möglichkeit offenzuhalten, zwischen *daisy-chaining* und unabhängigem Zugriff auf die beiden Gruppen wählen zu können. Soll *daisy-chaining* verwendet werden, so verbindet man den Ausgang des einen Interfaces mit dem Eingang des anderen. Damit reduziert man die Anzahl der nach außen gehenden Leitungen der seriellen Schnittstelle um zwei. Allerdings geht dadurch auch Bandbreite verloren.
- Die Zahl der *Pads* beträgt 113. Für *JTAG*[3] sind 5 und für die Spannungsversorgung 14 *Pads* vorgesehen. *JTAG* ist eine Testlogik, die die interne Chiplogik umgibt. Mit ihrer Hilfe ist es möglich Testvektoren in den Chip zu laden und auszulesen ohne die serielle Schnittstelle nutzen zu müssen. Außerdem bietet sie die die Möglichkeit

[3]engl. Joint Test Action Group

Tests auf Board-Ebene durchzuführen, für die Komponenten auf dem Board, die ein JTAG-Interface besitzen.

- Da die Fläche der PPrAsic nicht durch seine *Pads*, sondern durch die Fläche, die die Logik beansprucht, bestimmt ist, lassen sich die *Pads* so anordnen, daß der Chip später sowohl *Flip-Chip*[4] als auch drahtgebondet werden kann.

- Als Prozeßtechnologie wurde der 0.6μ CMOS[5] Prozeß der Firma AMS ausgewählt. Die Versorgungsspannung soll 3.3 V betragen.

Abbildung 6.8 zeigt das Layout des PPrAsic in seiner 4-Kanal-Version.

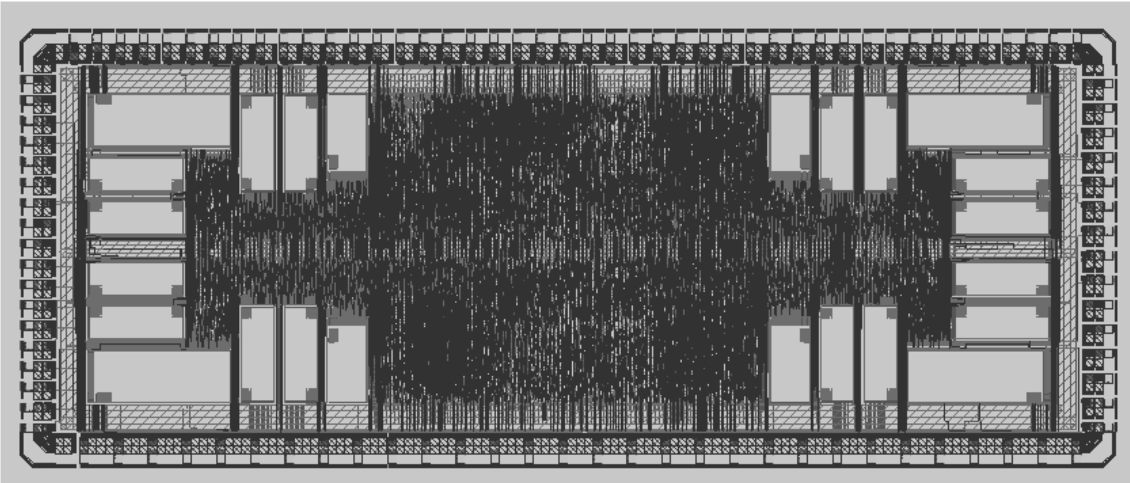

Abbildung 6.8: Layout des PPrAsics

Die Zahl der Standardzellen liegt bei etwa 34000. Die Gesamtfläche des PPrAsic beträgt etwa 68 mm^2, davon sind 14.12 mm^2 von den vorgefertigten Speicherbausteinen von AMS belegt. Insgesamt sind 24 solcher Speicherbausteine vorhanden. Die vier großen, die in den Ecken des Designs liegen, werden für die Look-Up-Tables der einzelnen Kanäle gebraucht. Sie besitzen 1024 10-Bit breite Speicherzellen. Die anderen Speicherbausteine werden als Auslesespeicher für das *Histogramming* und die *Playback*-Daten benötigt.

6.4 Die Gesamtlatenzzeit

Eine der wichtigsten Anforderungen an den PPrAsic ist die Verarbeitung der Signale in möglichst kurzer Zeit, d.h. in möglichst wenig Taktzyklen. Im *Technical Design Report* wurde für die gesamte Kette der Signalverarbeitung auf dem PPrAsic eine Zahl von 10 Taktzyklen für den Datenpfad zum Cluster-Prozessor und von 12 für den Datenpfad zum

[4]Bei der *Flip-Chip* Technologie werden die Chips überkopf direkt auf das Trägersubstrat aufgeklebt. Dafür werden auf dem Substrat an Stellen, auf denen ein *Pad* zu liegen kommen soll,Lötkugeln aufgebracht, die dann den Kontakt mit dem Chip herstellen

[5]Complementary Metal Oxide Semiconductor

Jet-Prozessor gefordert. Abbildung 6.9 zeigt die Anzahl der Taktzyklen, die der PPrAsic für die Signalverarbeitung benötigt und vergleicht diese mit den im TDR geforderten Werten. Die untere Hälfte des Bildes zeigt die Bildschirmdarstellung nach einer Simulation des Verilog-Codes.

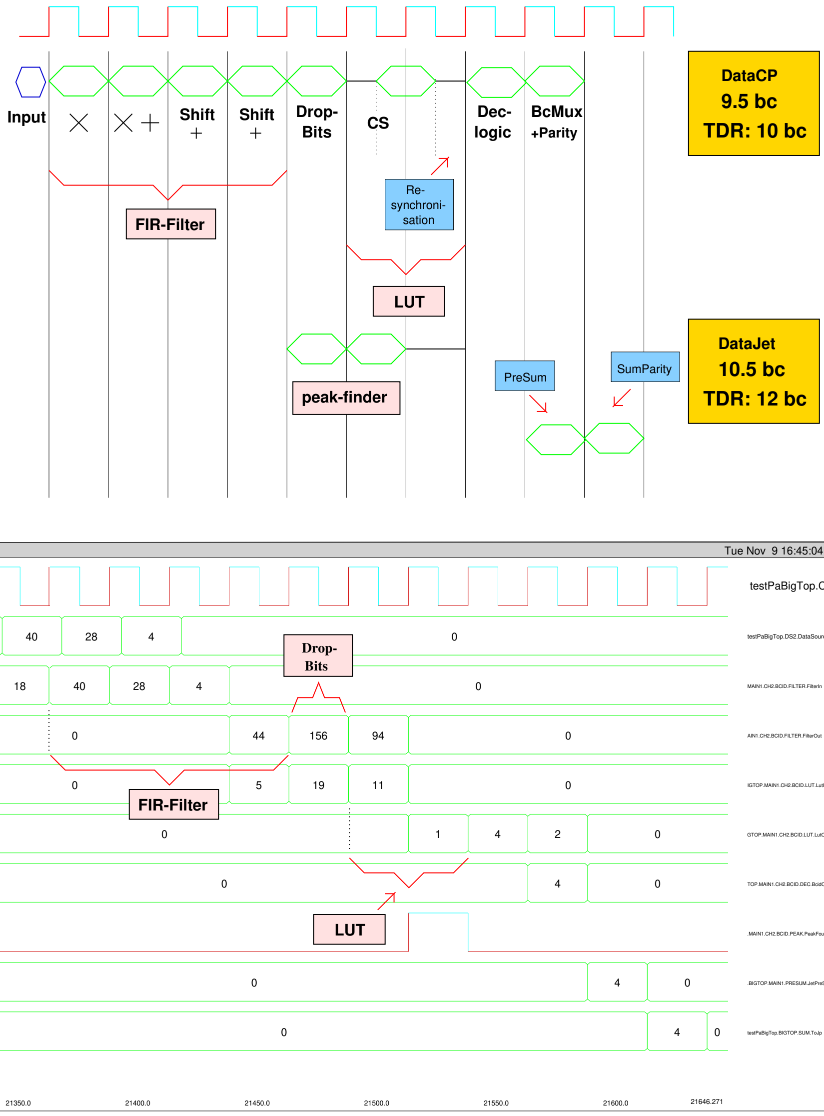

Abbildung 6.9: Vergleich der für die Signalverarbeitung auf dem PPrAsic wirklich benötigten Taktzyklen mit den Angaben des TDR

Die Anzahl der Taktzyklen, die der PPrAsic in der vorgestellten Version benötigt, um aus den Eingangsdaten das Bitwort zu erzeugen, das an den Cluster-Prozessor geht, beträgt 9.5. Damit liegt sie unter der geforderten Maximaldauer von 10 Taktzyklen. Im Falle des Datenpfades, der zum Jet-Prozessor geht, sieht dies noch günstiger aus. Der PPrAsic benötigt hier 10.5 Taktzyklen und bleibt damit 1.5 Taktzyklen unter der geforderten Zahl.

Kapitel 7

Simulationen des BCID und seiner Implementation

7.1 Der Simulationsaufbau und seine Ziele

Um in der Lage zu sein, das BCID mit den beiden Algorithmen vollständig zu simulieren, und Aussagen über seine Effizienz treffen zu können, wurde eine Simulationsumgebung geschaffen, die alle Komponenten des BCIDs beinhaltet. Dafür wurden zwei Programme ausgewählt, die eine gute Umgebung für eine umfangreiche Simulation boten. Um die Simulation mit realistischen Eingangssignalen durchführen zu können, wurde der *Mikro Sim PSPICE-V8* Simulator dazu verwandt, die Elektronik zu simulieren, die die analogen Pulse für das Präprozessor-System erzeugt. Um den Algorithmus auf einer abstrakten funktionalen Ebene beschreiben zu können, wurde das Progammpaket PTOLEMY 0.7.1 [16] ausgewählt, das an der Berkeley Universität in Kalifornien entwickelt wurde. Mit Hilfe dieser beiden Progamme war es möglich das BCID in vielen möglichen Konfigurationen und mit realistischen Eingangspulsen zu simulieren.

Da die PTOLEMY-Simulation des BCID vollkommen unabhängig vom Verilog-Code des BCID-Blocks ist, bietet sie außerdem die Möglichkeit die korrekte Funktionsweise des in Verilog geschriebenen BCID-Blocks mit seinen beiden Algorithmen zu testen. Damit wurde sie zu einem nützlichen Werkzeug bei der Entdeckung von Fehlern in dem BCID beschreibenden Verilog-Code.

7.2 Simulation der analogen Eingangsdaten mit Hilfe des Schaltungssimulators PSPICE

Für die durchzuführenden Simulationen wurden zuerst Eingangsdaten gebraucht. Hier schien es vorteilhaft, schon möglichst realitätsnahe Pulse zu verwenden. Daher wurde das sogenannte PSPICE-Programm dazu benutzt, um die Elektronik, die auch die wirklichen Pulse durchlaufen müssen, nachzubilden. Dazu wurde ein Schaltplan verwandt, der die Elektronik der Trigger-Tower Auslesekette des Flüssig-Argon Kalorimeters bei ATLAS darstellt [17]. Er beinhaltet ein Modell des *linear mixer*, des *layer sum board*, der *tower-builder*, der Empfänger-Station und ein ideales Modell für ein 70 m Kabel. Das Eingangssignal für diese Schaltungssimulation war ein dreieckiger Flüssig-Argon *Drift-Strom* mit einer *Drift-Zeit* von $t_{dr} = 400$ *ns* und einem Strom von 3.2 μA/GeV. Die erwartete Zeit bis zum Pulsmaximum liegt bei dieser Schaltung zwischen 51 und 52 ns nach dem tower-builder und bei etwa 63 ns am Ausgang der Empfänger-Station.

Mit dieser Schaltungssimulation wurden Pulse mit einer transversalen Energie zwischen 1 GeV und 10 TeV erzeugt. Die Schrittweiten betrugen 1 GeV zwischen 1 und 399 GeV, 5 GeV zwischen 400 und 995 GeV und 25 GeV zwischen 1 und 10 TeV. Die gelieferten Ergebnisse wurden mittels eines *Perl-Scripts* auf ein für PTOLEMY handhabbares Format gebracht.

7.3 Das Programm-Paket PTOLEMY

PTOLEMY ist ein umfangreiches und flexibles Programmpaket, mit dem Prototypen von Schaltungssystemen aufgebaut werden können. Es simuliert nicht die physikalischen Eigenschaften einzelner elektronischer Bauteile, sondern nur ihr Gesamtverhalten, welches nach außen erkennbar wird.

PTOLEMY beinhaltet eine graphische Oberfläche, mit der auf Datenfluß basierende Signalverarbeitung durchgeführt werden kann. Es verbindet Simulationen, die auf einem synchronen Datenfluß basieren (*synchronous data flow, SDF*), mit solchen, die durch diskrete Ereignisse gesteuert werden (*discrete event, DE*).

Das Programm ist hierarchisch aufgebaut. In der obersten Ebene steht die lauffähige Simulation eines abgeschlossenen Systems, das sogenannte *Universe*. Dieses kann eine Vielzahl funktionaler Blöcke beinhalten. Diese Blöcke können entweder einzelne alleinstehende Programme, sogenannte *stars* sein, oder sie repräsentieren ein Unterprogramm, eine sogenannte *galaxy*, das seinerseits aus einzelnen funktionalen Blöcken aufgebaut ist. Jede *galaxy* und jedes *universe* muß in eine sogenannte *domain* eingebettet sein. Diese Domänen repräsentieren eine bestimmte Umgebung, in der die Module arbeiten. So gibt es beispielsweise Domänen, die die Umgebung für eine Simulation, die auf einem synchronen Datenfluß basiert (*SDF*), bereitstellt. Es stehen, für jede Domäne separat, eine große Anzahl vorgefertigter Module (*stars*) zur Verfügung, die über eine graphische Oberfläche miteinander verbunden und so zu Simulationsprogrammen zusammengefügt werden können. Das Programm unterstützt jedoch auch das Erzeugen von eigenen *stars*, die in C++ geschrieben und dann in die Simulation integriert werden können. Diese Methode wurde beim Aufbau der Simulation für den BCID-Block überwiegend benutzt.

Ziel der Strategie, die beim Aufbau des Simulationsprogramms angewandt wurde, war es, eine vollständige Parallelität zwischen PTOLEMY- und Verilog-Simulation zu erreichen. Daher wurden alle Funktionsblöcke, die in der Verliog-Beschreibung als Module bezeichnet wurden, mit Hilfe von C++ Programmen nachgebildet. Diese wurden dann als eigenständige *stars* in die PTOLEMY-Simulation eingefügt. Abbildung 7.1 zeigt einen Überblick über das Simulationsprogramm, das für den Test des BCID aufgebaut wurde. Sie zeigt die Bildschirmdarstellung, wie sie sich einem Benutzer darstellt.

In dieser Abbildung existieren zwei unterschiedliche Gruppen von Modulen. Die Blökke, die durch einen Stern symbolisiert werden, stehen für ausführbare, in C++ geschrie-

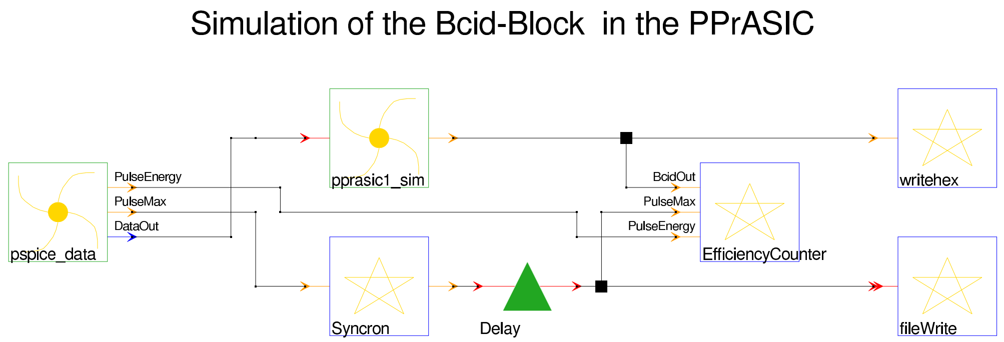

Abbildung 7.1: Überblick über das zur Simulation des BCID-Blocks auf dem PPrASIC entwickelte Programm

bene Programme, deren Funktionalität den Modulen, die als Verilog-Code vorliegen, entspricht (*stars*). Die andere Gruppe repräsentiert Unterprogramme, die ihrerseits aus den vorher erklärten Blöcken aufgebaut sind (*galaxies*). Im folgenden werden die Aufgaben dieser Blöcke näher erläutert.

7.3.1 Der Block für die Dateneinlese

Dieser Block (*pspice-data*) übernimmt die Aufgabe Pulsdaten, die vorher mittels *PSPICE* erzeugt wurden, einzulesen. Danach fügt er Rauschen hinzu und, falls erwünscht, verschiebt er die Zeitstruktur der Pulse um einen geringen Betrag. Der Betrag dieser Verschiebung wurde einer Gaußverteilung mit einer Standardabweichung von 1 ns und einer Begrenzung der Maximalwerte auf 2 bzw. 4 ns entnommen. Dieses sollte unerwartete Verschiebungen der Signale aufgrund von kurzzeitigen nicht kalibrierbaren Effekten simulieren. Die Pulsverschiebung und das Hinzufügen des Rauschens wird durch Unterprogramme erledigt. In Abbildung 7.2, die die innere Struktur dieses Blocks zeigt, sind diese Unterprogramme dargestellt.

Der Block *Data1Source* liest die von PSPICE generierten analogen Pulsdaten ein. Jeder Puls hat eine Breite von 1200 ns, d.h. 1200 Werte müssen pro Puls eingelesen werden. Um vergleichen zu können, ob der simulierte BCID-Block das richtige Bunch-Crossing identifiziert hat, ist noch die Information über die tatsächliche Lage des Pulsmaximums notwendig. Diese wird durch die PSPICE-Simulation festgelegt und ist daher auch für Pulse mit unterschiedlicher Energie gleich. Die Analyse der erzeugten Pulse ergab, daß bei jedem Signal der Maximalwert nach 487 ns erreicht wird. Um in der Lage zu sein, die Ergebnisse der Simulationen gegenüber der Pulsenergie auftragen zu können, werden noch die Länge der Pulse (1200 ns), die Anzahl der im Datenfile stehenden Pulse und die zu den Pulsen gehörende Energie eingelesen.

Der Block *ReadFile* ist dafür zuständig, die analogen Pulsdaten mit Rauschen zu versehen und die Verschiebung der Zeit-Struktur vorzunehmen. Die Verschiebung der Zeitstruktur erfogt auf die oben beschriebene Weise. Für das Rauschen liest er dafür gene-

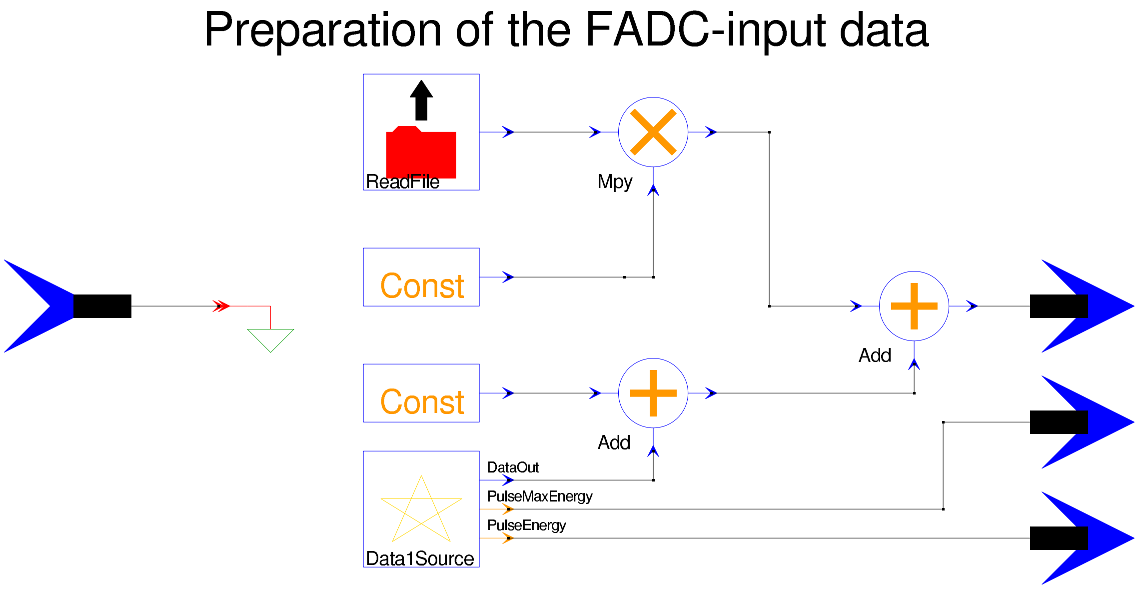

Abbildung 7.2: Überblick über den pspice_data Block, der die analogen Pulsdaten einließt, und sie für die Simulation bereitstellt

rierte Werte ein. Das Rauschen wurde erzeugt, wie in [19] beschrieben. Es beinhaltet zum einen das Pile-Up-Spektrum, das von *ATRIG*[1] bei hoher Luminosität simuliert wurde. *ATRIG* [20] ist ein Programmpaket, das Simulationen der Level-1 Trigger Elektronik und Menue-Auswahl, sowie von Teilen, der in der Entwicklung befindlichen Level-2 Algorithmen erlaubt. Zum anderen integriert es das Spektrum des erwarteten Rauschens der Flüssig-Argon Kalorimeter Elektronik, die von einem viergradigen Tiefpass mit einer Abschneidefrequenz von 20 MHz abgeschlossen wird. Abbildung 7.3 zeigt die Amplitudenverteilung dieses Rauschens.

7.3.2 Der Simulationsblock des BCID

Dieser Block (*pprasic1_sim*) repräsentiert die komplette Simulation des BCID-Blocks auf dem Präprozessor ASIC. Er erhält die Eingangsdaten von dem oben beschriebenen Block (*pspice_data*). Zuerst werden diese Daten mittels eines Modells eines idealen schnellen Digital-Analog-Wandlers[2] digitalisiert. Die Ergebnisse werden einerseits in ein Datenfile geschrieben, das Ausgangspunkt für die Simulation des Verilog-Codes mittels des *Verilog-XL interpreters* (siehe Abschnitt 6.2) ist. Andererseits stehen sie für die PTOLEMY-Simulation des BCID zur Verfügung. Abbildung 7.4 zeigt den strukturellen Aufbau dieses Simulationsblocks.

[1] Atlas TRIGger Simulation Package
[2] engl. Flash Analog Digital Converter (FADC)

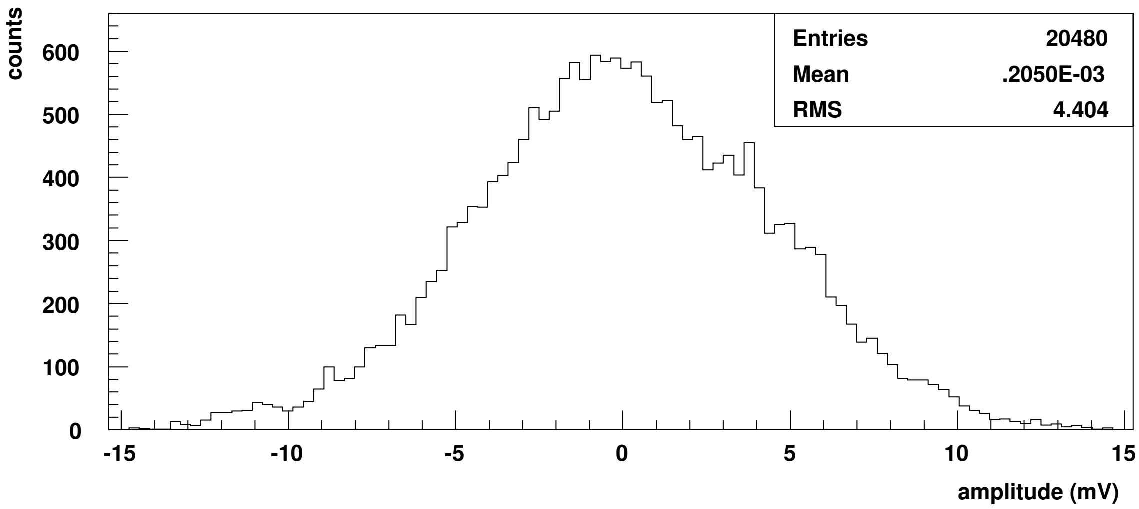

Abbildung 7.3: Erwartete Ampitudenverteilung des Rauschens der Flüssig-Argon Kalorimeter Elektronik einschließlich des pile-up Rauschen, das vom ATRIG bei hoher Luminosität genommen wurde. Das RMS-Rauschen beträgt 4,4 mV (440 MeV) [19]

7.3.3 Der Synchronisationsblock

Dieser Block (*Syncron*) ist dafür zuständig, die wirkliche Position des Pulsmaximums, die ihm auf 1 ns genau übermittelt wird, auf das Zeitraster von 25 ns, das der BCID Algorithmus für seine Entscheidung benutzt, abzubilden, um in der Lage zu sein, einen direkten Vergleich zwischen wahrem und dem vom BCID-Algorithmus bestimmten Zeitpunkt des Pulsmaximum anstellen zu können.

7.3.4 Der Block für die Effizienzmessung

Dieser Block (*Efficiency Counter*) ist dafür zuständig, die Ergebnisse der PTOLEMY-Simulation, d.h. den Zeitpunkt des von ihr entdeckten Pulsmaximums, mit dem wirklichen zu vergleichen und die Ergebnisse zu speichern, bzw. in einem Graphen darzustellen. Damit ist es möglich die beiden Algorithmen für saturierte und nicht saturierte Pulse in unterschiedlichen Modifikationen zu simulieren und auf ihre Effizienz zu untersuchen.

7.4 Ergebnisse der PTOLEMY-Simulationen des BCID

Mit Hilfe der PTOLEMY-Simulation ist es möglich, den kompletten BCID-Block mit seinen beiden Algorithmen für saturierte und nicht saturierte Signale und der dazugehörigen Entscheidungslogik zu simulieren. Diese Funktionalität wurde für verschiedene Zwecke eingesetzt:

- Zum einem wurde mit Hilfe der PTOLEMY-Simulation die Effizenz des BCIDs sowohl für saturierte, als auch für nicht saturierte Pulse analysiert. Dafür wurden die

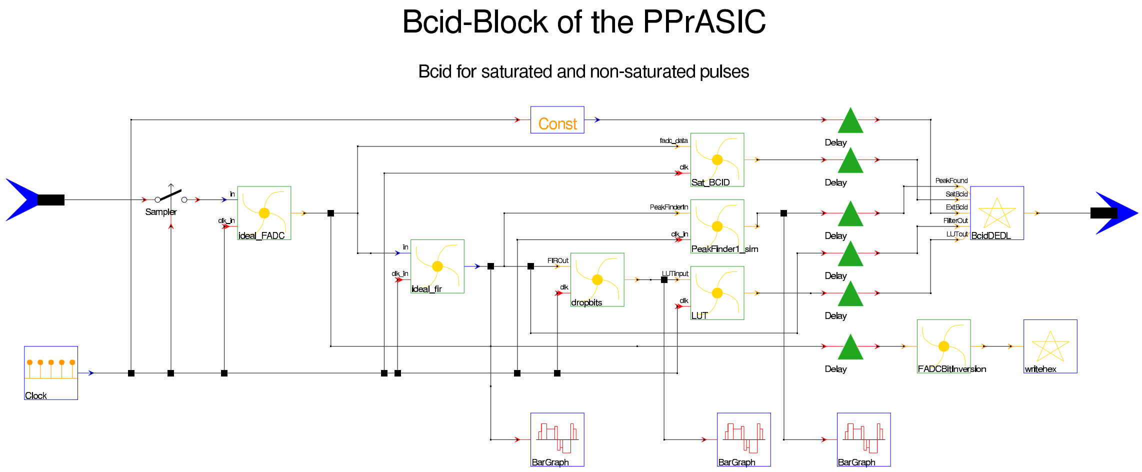

Abbildung 7.4: Überblick über die innere Struktur des Ptolemy-Simulationsblocks, der die Verilog-Implementation des BCID repräsentiert

Entscheidungen des BCIDs, abhängig von unterschiedlichen Einstellungen der in den einzelnen Stufen des BCIDs verwendeten Werte, untersucht.

- Das zweite wichtige Einsatzfeld der PTOLEMY-Simulation war die automatische Überprüfung des Verilog-Codes.

7.4.1 Die Analysen zur Effizienz der Algorithmen

Die folgenden Graphen zeigen Ergebnisse mehrerer PTOLEMY-Simulationen, die durchgeführt wurden, um die Effizienz der Algorithmen abhängig von den einstellbaren Werten zu testen. Alle Simulationen verwendeten als Eingangsdaten simulierte Pulsformen, die mittels der schon erwähnten PSPICE-Simulation für Energien zwischen 0.5 GeV und knapp über 8 TeV generiert wurden. Die Pulse wurden mit dem in Abschnitt 7.3.1 beschriebenen Rauschen in der Grßenordnung von 500 MeV behaftet. Außerdem wurden sie um zufällige Werte, die einer Gaußverteilung mit einer Standardabweichung von 1 ns entnommen wurden, verschoben. Die Werte wurden auf ± 4 ns beschränkt.

Abbildung 7.5 zeigt das Ergebnis einer Simulation des Algorithmus für nicht saturierte Signale. Der FIR-Filter wurde in diesem Fall abgeschaltet, was durch die FIR-Filter Koeffizienten 0-0-1-0-0 erreicht wurde. Als Bedingung für den *Peak-Finder* wurde die erste Bedingung ($bc_{t_{-1}} < bc_{t_0} \geq bc_{t_1}$) gewählt. Wie man erkennt, steigt die Effizienz nicht sofort an, so daß erst ab 21 GeV jedes Pulsmaximum dem richtigen *Bunch-Crossing* zugeordnet wird. Kurz nach der Saturierung, die bei 256 GeV einsetzt, erkennt man einen sehr raschen Abfall der Effizienz, die bei Werten über 275 GeV stabil auf Null bleibt. Bei saturierten Pulsen ergibt sich kein eindeutiges Pulsmaximum mehr, sondern es entsteht ein Plateau, von dem der Peak-Finder-Algorithmus immer die vordere Flanke als Maximum identifiziert. Da die vordere Flanke aber mit zunehmender Energie der Pulse immer steiler wird, liegt der erste saturierte Wert schon vor dem eigentlichen Pulsmaximum und daher wird dieses zu früh identifiziert.

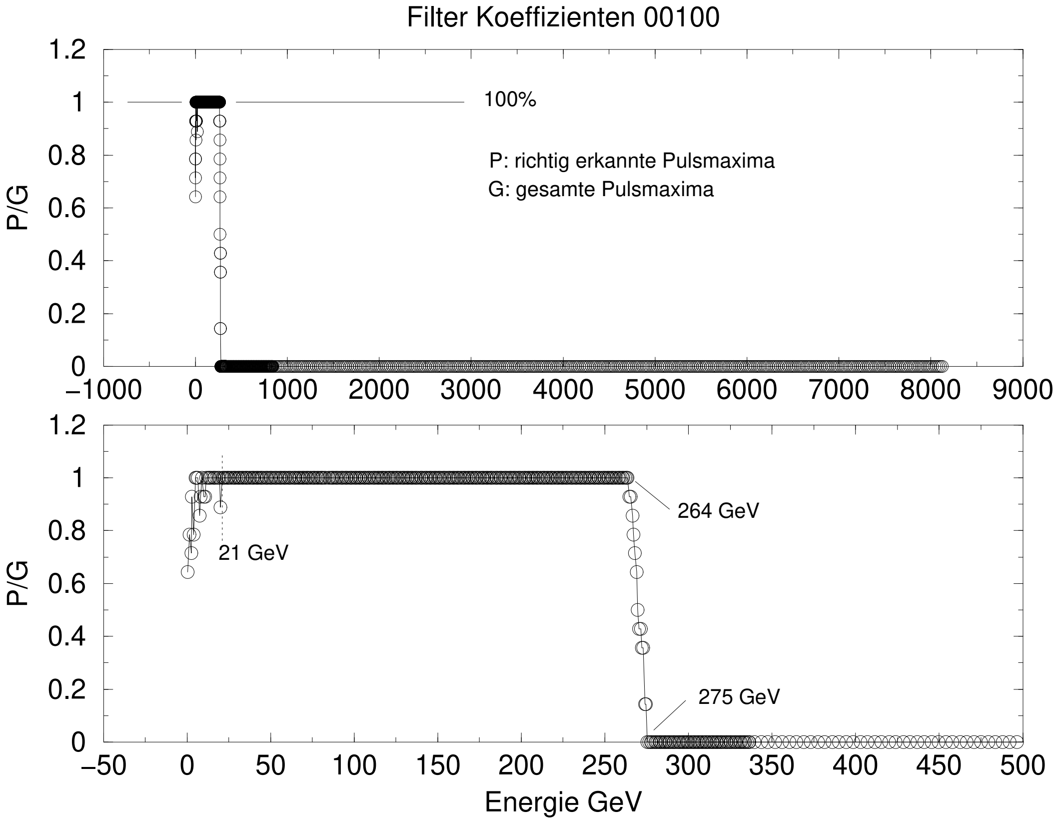

Abbildung 7.5: Effizienz des Algorithmus für nicht saturierte Signale bei einem auf Durchlaß geschalteten FIR-Filter (00100). Die untere Hälfte der Abbildung zeigt einen vergrößerten Ausschnitt der oberen Hälfte.

Wählt man nun die Koeffizienten des FIR-Filters so, daß die Pulse geschärft werden und dadurch wieder ein definiertes Pulsmaximum erzeugt wird, so erwartet man, daß der Wirkungsbereich des Peak-Finder-Algorithmus bis zu Werten, die schon in der Saturierung liegen, erweitert werden kann. Das Ergebnis einer Simulation, bei der eine solche FIR-Filter Konfiguration ((-1)-0-4-0-(-1)) verwandt wurde, zeigt Abbildung 7.6.

Der Effizienz des Algorithmus erreicht bei einer solchen Konfiguration schon bei 9.5 GeV ihr Maximum. Das sie dies nicht noch früher erreicht, liegt an der zufälligen Verschiebung der Pulse. Bei einer solchen Verschiebung müssen die FIR-Filter Koeffizienten jedem einzelnen Puls angepaßt werden Man erkennt, daß der Algorithmus bis hin zu sehr hohen Energien, die längst im Saturationsbereich liegen, verläßlich arbeitet. . Ziel war es auch den Überlappbereich von saturiertem und nicht saturiertem Algorithmus so groß wie möglich zu machen, um eine hohe Sicherheit für die richtige *Bunch-Crossing-Identification* zu erreichen.

Analysiert man den Algorithmus für saturierte Pulse, so erkennt man die Abhängigkeit seiner Effizienz von den Schwellen, die er verwendet. Die Schwellen werden auf die 10-Bit breiten digitalen Eingangswerte gesetzt, und können sich daher im Rahmen von 0 bis 1023

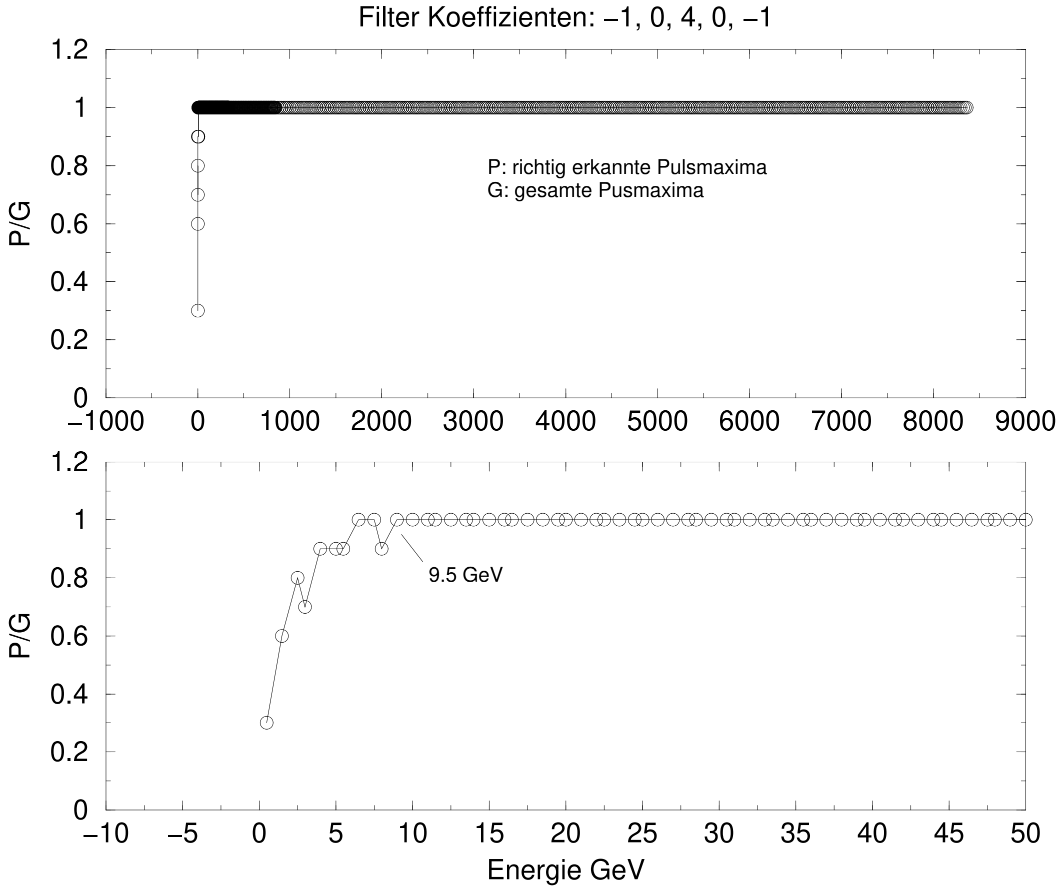

Abbildung 7.6: Effizienz des Algorithmus für nicht saturierte Signale bei FIR-Filter Koeffizienten von -1, 0,4,0,-1. Die untere Hälfte der Abbildung zeigt einen vergrößerten Ausschnitt der oberen Hälfte.

bewegen.

Abbildung 7.7 zeigt die Effizienz des Algorithmus bei einer unteren Schwelle von 500 und einer oberen von 900. Man erkennt, daß der Algorithmus erst wesentlich nach der eigentlichen Saturierung anfängt richtige *Bunch-Crossings* zu identifizieren. Erst ab 410.5 GeV steigt die Effizienz langsam an und erreicht 100% bei 517.5 GeV.

Die untere Schwelle ist dafür da, um bei sehr hohen Pulsen zu verhindern, daß der Algorithmus ein zu frühes *Bunch-Crossing* als das richtige identifiziert. Bei Pulsen von geringer Energie wird sie eigentlich nicht benötigt. Hier sollte sie immer überschritten werden. Daher liegt es nahe, daß man als erstes diese Schwelle auf einen sehr niedrigen Wert absenkt, um in der Lage zu sein auch das Maximum kleinerer Pulse richtig zu erkennen. Abbildung 7.8 zeigt die erreicht Effizienz bei einer unteren Schwelle von 10 und bei einer beibehaltenen oberen Schwelle von 900.

Man erkennt, daß der Algorithmus nun schon bei Pulsen kleinerer Energien anfängt das richtige *Bunch Crossing* zu identifizieren, und seine Effizienz auch wesentlich schneller

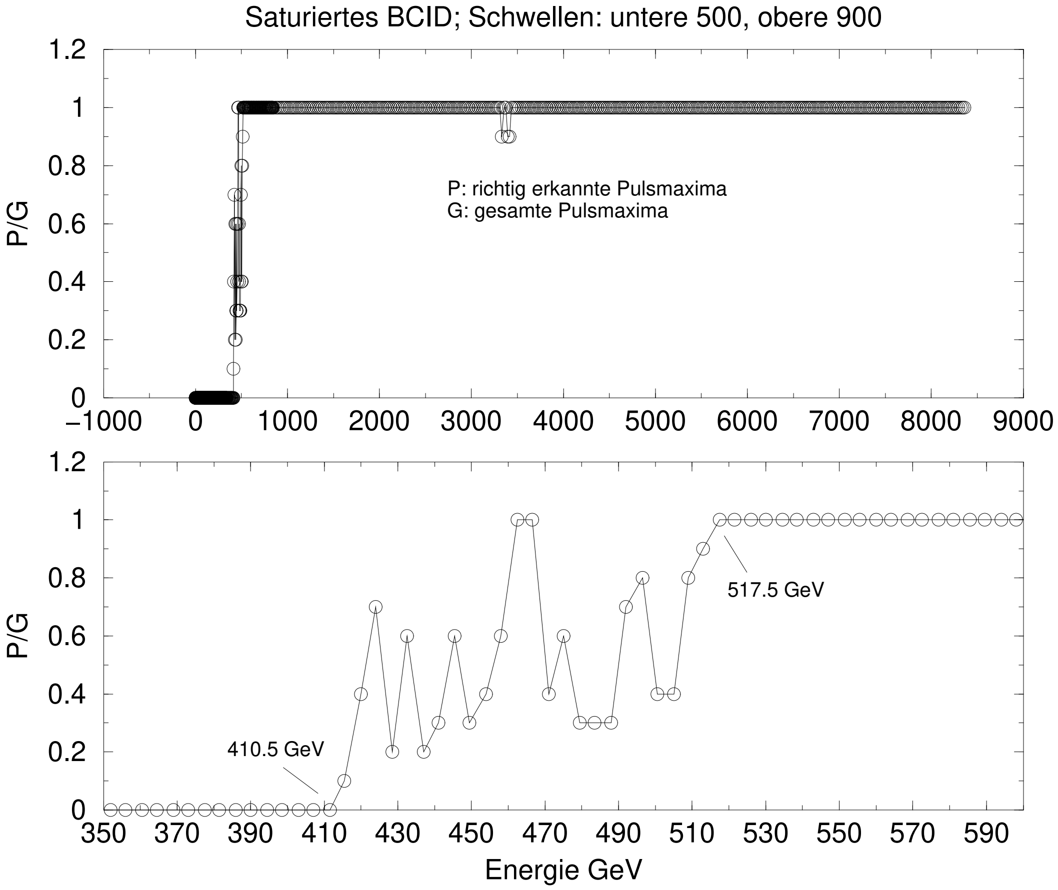

Abbildung 7.7: Effizienz des Algorithmus für saturierte Signale bei einer unteren Schwelle bei 500 und einer oberen bei 900. Die untere Hälfte der Abbildung zeigt einen vergrößerten Ausschnitt der oberen Hälfte.

auf den Maximalwert steigt. Allerdings fängt er mit 329.5 GeV noch erheblich später als die Saturationsgrenze an, effizient zu werden.

Betrachtet man nur Pulse mit hohen Energien, so sollte die obere Schwelle möglichst hoch gewählt werden, um ein Fehlverhalten des Algorithmus auszuschließen. Für Pulse niedriger Energien sieht dies allerding anders aus. Denn schon der Puls, der gerade die Saturierungsschwelle überschreitet, sollte richtig erkannt werden. Dafür ist es notwendig das der sat_{-1}-Wert dieses Pulses oberhalb der oberen Schwelle liegt. Abbildung 7.9 zeigt daher die Effizienz des Algorithmus bei einer unteren Schwelle von 10 und einer oberen von 500.

Wie man erkennt arbeitet der Algorithmus bei einer solchen Wahl der Schwellen ab 259.5 GeV mit einer Effizienz von 100%.

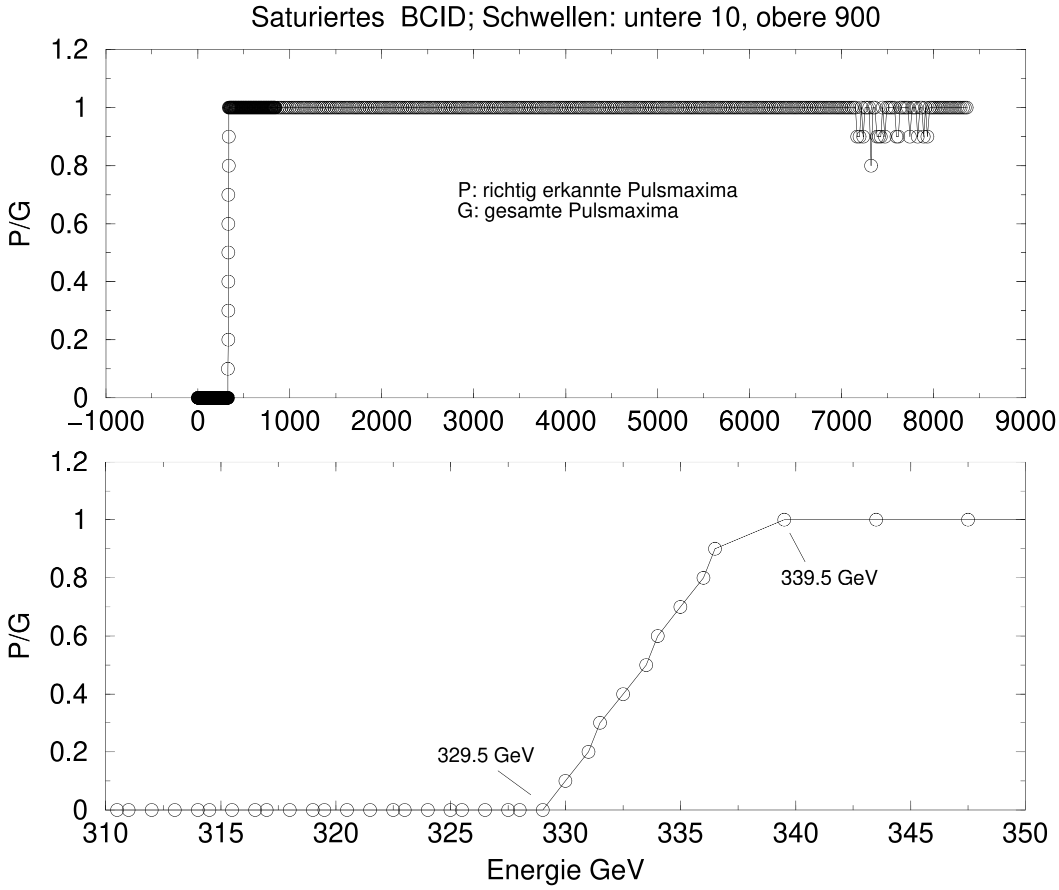

Abbildung 7.8: Effizienz des Algorithmus für saturierte Signale bei einer unteren Schwelle bei 10 und einer oberen bei 900. Die untere Hälfte der Abbildung zeigt einen vergrößerten Ausschnitt der oberen Hälfte.

7.4.2 Die Überprüfung des Verilog-Codes

Mit Hilfe des PTOLEMY-Simulationsaufbaus ist es außerdem möglich, die korrekte Arbeitsweise des geschriebenen Verilog-Codes zu überprüfen. Um dies in vielen möglichen Konfigurationen tun zu können, wurde ein System entwickelt, das dem Benutzer die Möglichkeit gibt, relativ unkompliziert neue Konfigurationen sowohl in den ASIC, als auch in die PTOLEMY-Simulation zu laden. Das System basiert auf dem Zusammenspiel mehrerer *Perl*-Skripte. Ein Hauptprogramm liest aus einem Konfigurationsfile die gewünschte Konfiguration ein und generiert daraus zum einen das Stimulus-File für die Verilog-Simulation, das über die serielle Schnittstelle die ausgewählten Registerwerte an die richtige Stelle bringt. Zum anderen schreibt es auch die Konfigurationsfiles heraus, die die PTOLEMY-Simulation verwendet. Auf diese Weise erlangt man rasch notwendige Übereinstimmung der Parameter für die beiden Simulationen.

Um die Ergebnisse der Simulationen automatisch miteinander vergleichen zu können, wurde ein Verilog-Modul geschrieben. Dieses liest die Ergebnisse der beiden Simulationen

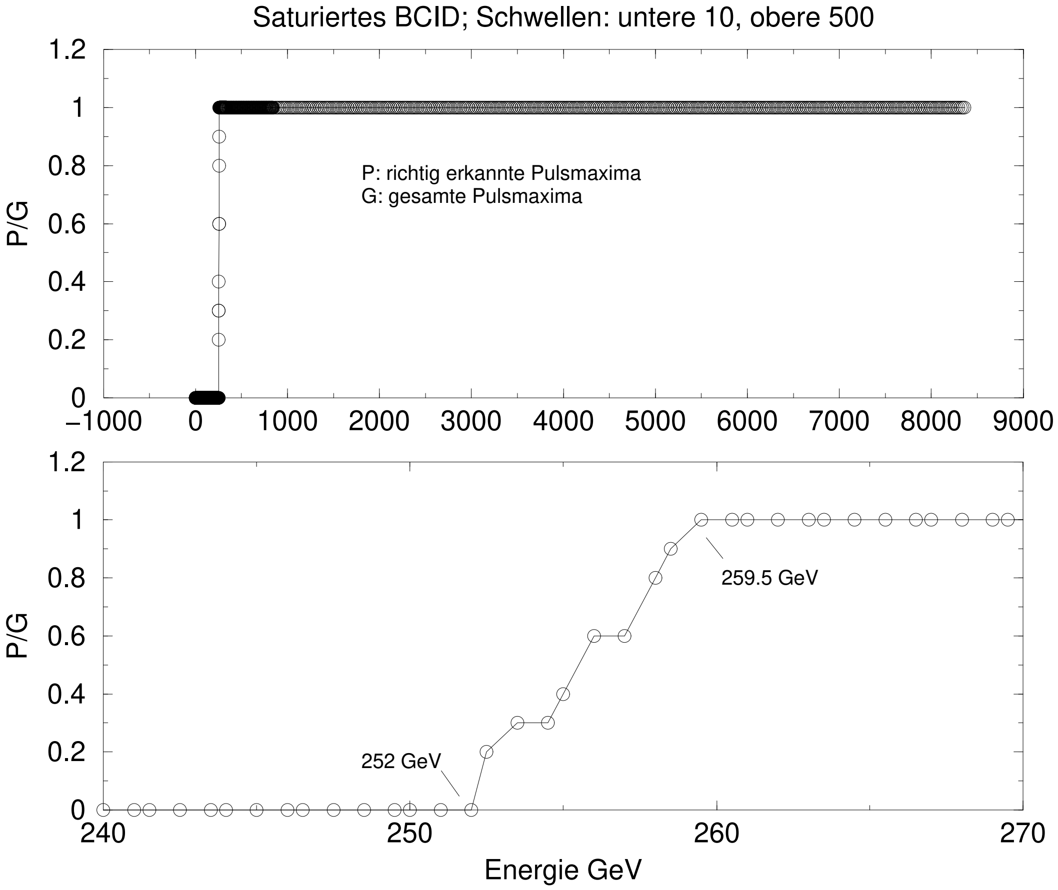

Abbildung 7.9: Effizienz des Algorithmus für saturierte Signale bei einer unteren Schwelle bei 10 und einer oberen bei 500. Die untere Hälfte der Abbildung zeigt einen vergrößerten Ausschnitt der oberen Hälfte.

ein, vergleicht sie und speichert das Resultat. Damit war es möglich mehrere Konfigurationen vorher auszuwählen und dann die Simulationen an einem Stück durchführen zu lassen. Für genauere Informationen über die Arbeitsweise und das Zusammenspiel der verwendeten *Perl*-Skripte sei auf den *PPrAsic Design Guide* [22] verwiesen.

Es wurde eine Liste erstellt, die alle notwendigen Simulationen beinhaltet. Für alle Punkte auf der Liste für die ein automatischer Test sinnvoll schien, wurde ein Programm geschrieben, das diese nacheinander abarbeitet und die Ergebnisse zusammenfaßt. Jeder Test wurde mit Pulsen der Energien 1 bis etwa 8500 GeV, die mit Hilfe des PSPICE-Simulators erzeugt wurden, durchgeführt. Im folgenden ist ein Ausschnitt aus dem Ergebnisfile abgedruckt:

```
Test -- 1 --
->RegChannelBoth
FIRCoeff1 -1
```

```
FIRCoeff3 4
FIRCoeff5 -1
SatLow 0
SatHigh 0
BcidDecision1 0x0f0
BcidDecision2 0x0f0
BcidDecision3 0x0f0
BypassLut=on(1)

Wrong1 :     0
Right1 :   890
Wrong2 :     0
Right2 :   885
```

Zuerst werden die Konfigurationen die in den Chip geladen wurden, und anschließend die Ergebnisse angezeigt. Auf den Verilog-Code bezogen verliefen alle Simulationen der finalen Version positiv.

Der Simulationen der Implementation des PPrAsic sind noch nicht abgeschlossen. Bisher wurde die korrekte Arbeitsweise des PPrAsic auf der Basis des Verilog-Codes gezeigt. Es verbleibt noch die vollständige Simulation des synthetisierten Codes, also des Schaltplans, mit den Informationen über Signallaufzeiten der verwendeten Gatter, und die des daraus resultierenden fertigen Layouts.

Zusammenfassung und Ausblick

Ziel dieser Diplomarbeit war es, einen wichtigen Bestandteil des Präprozessor-Systems des ATLAS Level-1 Triggers, den sogenannten PPrAsic mitzuentwickeln. Die Aufgaben dieses ASICs bestehen in der *Bunch-Crossing-Identification*, der Kalibration der Kalorimeterdaten auf transversale Energie und Einstellung der Nullinie, sowie der Übertragung der Daten an die nachfolgende Trigger-Logik. Er verfügt über einen *Playback*-Modus, in dem es möglich ist, selbst generierte Testdaten als Eingangssignale zu verwenden. Außerdem ist er in der Lage, sowohl die Eingangsdaten, als auch die dazugehörigen Ergebnisdaten für jeden Kanal auszulesen. Dies kann dazu verwandt werden, die Funktionalität des Triggers zu analysieren. Dafür sind auf dem ASIC zusätzliche Speicherbausteine vorhanden, in die Histogramme geschrieben werden können.

Die Hauptaufgabe des PPrAsic besteht darin, die zuvor digitalisierten Signale der Kalorimeter dem richtigen *Bunch-Crossing* zuzuorden. Dafür sollen auf dem ASIC zwei Algorithmen implementiert werden. Um diese auf ihre Effizienz hin untersuchen zu können, wurde ein eigenständiges Simulationsprogramm geschrieben. Mit seiner Hilfe konnte das BCID zum ersten Mal vollständig simuliert werden.

Die Implementation des PPrAsic erfolgte mit Hilfe der Hardwarebeschreibungssprache Verilog. Aus dem Verilog-Code wurde ein Schaltplan synthetisiert, und mit diesem anschließend ein Layout generiert.

Bei der Überprüfung der korrekten Arbeitsweise des Verilog-Codes konnte neben der Analyse des zeitlichen Verhaltens der Signale mit Hilfe eines Programms, das dieses graphisch anzeigt, auch das oben beschriebene Simulationsprogramm eingesetzt werden. Da es vollständig unabhängig von Verilog aufgebaut ist, brauchte man nur seine Ergebnisse mit denen des Verilog-XL Simulators zu vergleichen, um eventuelle Fehler im Code entdecken zu können. Auf diese Weise kann man die Fehlersuche schneller, zuverlässiger und effizienter durchführen.

Nachdem der Verilog-Code umfangreich getestet und aufgetretene Fehler behoben worden waren, konnte die Synthese erfolgen. Die synthetisierte Version mit integriertem JTAG wurde zuerst auf ihre Funktionalität hin überprüft. Dabei wurden noch keine Informationen über die Zeitstruktur der Schaltung, d.h über Signallaufzeiten der Gatter, oder der elektrischen Verbindungen zwischen diesen, in die Simulationen integriert.

Als nächster Schritt folgt die Überprüfung des Layouts und des Schaltplans mit den vollständigen Daten über Signallaufzeiten. Nach erfolgreichem Abschluß dieser Tests, kann der PPrAsic submittiert werden.

Damit verbleibt als weitere Aufgabe für die Zukunft, den fertigen ASIC zu testen und das Präprozessor-System aufzubauen.

Literaturverzeichnis

[1] siehe z.B. B. Povh, K. Rith, C. Scholz, F. Zetsche, *Teilchen und Kerne*, Springer-Verlag, 1994

[2] siehe z.B. C. Berger, *Teilchenphysik*, Springer-Verlag, 1992

[3] D. Griffiths, *Elementarteilchenphysik*, Akademie-Verlag, 1996

[4] R.M. Barnett et al., Physical Review D54 1, (1996)

[5] The LHC Study Group, *The Large Hadron Collider, Conceptual Design Report*, 1995, CERN/AC/95-05(LHC), http://www.cern.ch/CERN/LHC/YellowBook95/LHC95/LHC95.html

[6] European Committee for Future Accelerators, *Large Hadron Collider Workshop, Proceedings, Volume I*, 1990, CERN 90-10

[7] ATLAS Homepage, *Atlas Figures*, http://atlasinfo.cern.ch/Atlas/ATLASFIGS/atlasfigures.html

[8] ATLAS Homepage, *Atlas Figures*, http://atlasinfo.cern.ch/Atlas/ATLASFIGS/ATLASPIC/atlaspic_withgif.html

[9] K. Kleinknecht, *Detektoren für Teilchenstrahlung*, B.G. Teubner Stuttgart, 1992

[10] ATLAS, *Technical Proposal*, CERN, 1994, CERN/LHCC/94-43

[11] nach U. Pfeiffer, *Dissertation*, *A Compact Pre-Processor System for the ATLAS Level-1 Calorimeter Trigger*, Heidelberg, HD-IHEP 99-11

[12] C. Schumacher, *The Readout Bus of the ATLAS Level-1 Calorimeter Trigger Pre-Processor*, Fifth Workshop on Electronics for LHC Experiments, Snowmass, 20-24 September 1999 http://wwwasic.kip.uni-heidelberg.de/atlas/publications.html

[13] Readout Merger ASIC (RemAsic), *Readout Merger ASIC -User and Reference Manual*, Internes Papier, Universität Heidelberg, 18. März 1998 http://wwwasic.kip.uni-heidelberg.de/atlas/docs.html

[14] Technical Proposal for a general purpose pp experiment at the LHC at CERN, CERN/LHCC/94-43, 15. Dezember 1994

[15] ATLAS : liquid argon technical design report, *ATLAS Collaboration*, CERN-LHCC-96-41 ATLAS-TDR-2, 1996
http://atlasinfo.cern.ch/Atlas/GROUPS/LIQARGON/TDR/ps.html

[16] Ptolemy Project, *University of California Berkeley*,
http://www.ptolemy.eecs.berkeley.edu:80

[17] B. Cleland, *PSPICE model of the ATLAS LAr. analog trigger tower chain*, Bill Cleland, private communication, University of Pittsburgh, Juli 1998

[18] nach U. Pfeiffer, W. Hötzel, *Bunch-Crossing Identification for saturated calorimeter signals*, ATLAS note, 19. April 1999

[19] ATLAS First-Level Trigger Technical Design Report, *ATLAS Level-1 Trigger Group*, ATLAS TDR-12, CERN/LHCC/98-14, 24 Juni 1998

[20] Atlas TRIGger Simulation Package, *Atrig librarian/release coordinators.*
http://www.cern.ch/Atlas/GROUPS/DAQTRIG/ATRIG/index.html

[21] I. P. Brawn, *Bunch Crossing Identification for the ATLAS Level-1 Calorimeter Trigger*, Thesis, Faculty of Science University of Birmingham, England Juni 1996

[22] D. Husmann, M. Keller, K. Mahboubi, C. Schumacher, *Pre-Processor Asic Design Guide*, Universität Heidelberg, ASIC-Labor 1999
http://wwwasic.kip.uni-heidelberg.de/atlas/docs/index.html

[23] D. Husmann, M. Keller, K. Mahboubi, C. Schumacher, *Pre-Processor Asic User and Reference Manual*, Universität Heidelberg, ASIC-Labor 1999
http://wwwasic.kip.uni-heidelberg.de/atlas/docs/index.html

[24] Cadence Design Systems, Inc., *Verilog-XL Reference*, Online-Dokumentation, 1995

Danksagung

Abschließend möchte ich mich noch bei all denen bedanken, die mich während meiner Diplomarbeit so tatkräftig unterstützt haben.

Prof. Meier ermöglichte es mir, diese Arbeit überhaupt durchzuführen.

Prof. Kluge übernahm freundlicherweise die Zweitkorrektur der Arbeit.

Cornelius Schumacher half mir, einen raschen Einstieg in die Arbeit zu finden und betreute mich während des ganzen Jahres auf hervorragende Weise. Am Ende las er meine Arbeit noch eimal gründlich Korrektur und wies mich auf so manche Ungenauigkeit hin.

Die *fünf-Uhr*-Tasse Tee mit Till Tippel war immer eine angenehme Sache und ermöglichte das abendliche Arbeiten.

Ullrich Pfeiffer half mir in vielen Gesprächen den Hintergrund des ATLAS-Projektes verstehen zu lernen und war mir eine große Hilfe beim Umgang mit PSPICE und PTOLEMY.

Michael Keller war stets ein freundlicher und geduldiger Helfer in der Not, wenn ich mich zum wiederholten Male in den Fäden des UNIX-Netzes verfangen hatte.

Johannes Schemmel war immer ein interessanter Gesprächspartner bei den traditionellen Mensa-Gängen und stand mir auch bei so manchem Problem mit Rat zur Seite.

Torsten Maucher befriedigte stets meine Neugier und erzählte bereitwillig über die Fortschritte in seinem Projekt.

Außerdem möchte ich noch den *Dienstags-Meetings* mit all ihren Mitgliedern danken. Die regelmäßigen Treffen waren mir eine große Hilfe und ich danke Herrn Dr. Hanke und Klaus Schmitt für einige sehr informative Gespräche.

Zuletzt möchte ich noch den neuen Mitgliedern der ATLAS-Gruppe, namentlich den Stelzer-Brüdern Bernd und Oliver, sowie Volker Schatz und dem gesamten ASIC-Labor für die außergewöhnlich angenehme Arbeitsatmosphäre und die stetige Hilfsbereitschaft danken.