Fakultät für Physik und Astronomie
Ruprecht-Karls-Universität Heidelberg

Bachelorarbeit im Studiengang Physik
vorgelegt von

Daniel Dieter Glodeck

geboren in Lich (Deutschland)

2010

Simulation eines L1-Spur-Triggers für ATLAS

Die Bachelorarbeit wurde von Daniel Dieter Glodeck ausgeführt am

Physikalischen Institut der Universität Heidelberg

unter Betreuung von

Herrn Prof. Dr. André Schöning

Zusammenfassung

Im Rahmen dieser Bachelorarbeit soll ein möglicher L1-Spur-Trigger für den ATLAS Detektor bei SLHC untersucht werden. Aufgrund der Erhöhung der Luminosität beim Upgrade von LHC zu SLHC bei gleichbleibender Schwerpunktsenergie ist die Anzahl gleichzeitig erwarteter Ereignisse und somit Spuren im Detektor deutlich erhöht, wodurch ein zusätzlicher L1-Spur-Trigger notwendig wird.

Die Aufgabe des Spur-Triggers ist es innerhalb von 25 ns, also bei einer Kollisionsfrequenz von 40 MHz, Teilchen mit hohem Transversalimpuls zu identifizieren.

In dieser Arbeit soll das Konzept eines solchen L1-Spur-Triggers sowie die Hardware-Anforderungen des Triggers mit einer Simulation untersucht und bereits existierende Theorien zu diesem Trigger mit der Simulation verglichen werden.

Abstract

In the course of this thesis parts of a possible L1-Track-Trigger for ATLAS SLHC will be analysed. Because of the increase in luminosity at the upgrade from LHC to SLHC at constant center of mass energy the number of expected tracks will be strikingly increased. This makes an additional L1-Track-Trigger necessary for ATLAS SLHC.

The Track-Trigger shall detect particles with a high transverse momentum in the time of 25 ns between two collisions.

In this thesis the concept of such a L1-Track-Trigger and the hardware requirements of the trigger will be analysed in a simulation and compared with existing theories concerning the Track-Trigger.

Inhaltsverzeichnis

1 Einleitung **1**
1.1 Der Large Hadron Collider . . . 2
1.2 Ziele des LHC . . . 3
1.3 Der SLHC . . . 4

2 Allgemeine Grundlagen **7**
2.1 Luminosität . . . 7
2.2 Der ATLAS-Detektor . . . 8
2.2.1 Der Innere Detektor . . . 8
2.2.2 Das Magnetsystem . . . 9
2.2.3 Das Kalorimetersystem . . . 9
2.2.4 Das Myonspektrometer . . . 10
2.2.5 Das Triggersystem . . . 11
2.3 Änderungen des ATLAS-Detektors bei SLHC . . . 12
2.3.1 Der Innere Detektor bei SLHC . . . 12
2.3.2 Das Konzept des Spur-Triggers für SLHC . . . 12
2.4 Content Addressable Memory (CAM) . . . 14

3 Grundlagen der Simulation **17**
3.1 Betrachtete Layouts in dieser Simulation . . . 17
3.2 Das Generieren von Ereignissen . . . 18

4 Definition der Ausleseregionen **21**
4.1 Motivation . . . 21
4.2 Die Einteilung in ϕ-Richtung . . . 21
4.3 Die Einteilung in z-Richtung . . . 24

5 Simulation der Ausleseregionen **27**
5.1 Überprüfung der Formel für $\phi_{min}(p_T)$ . . . 27
5.2 Variation der Größe der Regionen auf der Strahlachse in z-Richtung . . . 29
5.3 Design der zum Auslesen genutzten Sensoren und Chips . . . 30

6 Wahl der Parameter $\Delta\phi$ und Δz **33**
6.1 Größe der Regionen in z und ϕ . . . 33
6.1.1 Größe der Regionen in z . . . 33
6.1.2 Größe der Regionen in ϕ . . . 35
6.1.3 Zusammenstellung der Templates für z und ϕ . . . 37
6.2 Duplizierung der Information von einzelnen Chips . . . 39
6.3 Anzahl zu speichernder Templates . . . 41

7 Erstellung der Templates und Speicherung in einer Datenbank **43**
7.1 Darstellung der Templates in der Simulation . . . 44

7.2 Generierung der Templates 44

8 Implementierung des Vergleichs zwischen Pattern und Templates 49
8.1 Der Vergleich in Hardware 49
8.2 Die Mustersuche in der Simulation 50
8.2.1 Sequentielle Verarbeitung der Datenbank 51
8.2.2 Direktzugriff mit Schlüssel auf die Datenbank 51

9 Effizienz und Fake-Rate 53
9.1 Effizienz 53
9.2 Fake-Rate und Selektion der Templates 54

10 Das Doppellagen-Layout 57
10.1 Effizienz 58
10.2 Fake-Rate 60

11 Fazit 63

12 Ausblick auf weitere Fragestellungen 64

1 Einleitung

Mit dem geplanten Upgrade des Large Hadron Collider (LHC) zum Super Large Hadron Collider (SLHC) soll die Design-Luminosität von 10^{34} $\mathrm{cm}^{-2}\mathrm{s}^{-1}$ auf 10^{35} $\mathrm{cm}^{-2}\mathrm{s}^{-1}$ bei gleichbleibender Schwerpunktsenergie von $\sqrt{s} = 14$ TeV verzehnfacht werden. Bei einer Kollision bei SLHC werden dabei bis zu 400 Ereignisse gleichzeitig erwartet. Unter LHC-Bedingungen werden für die Design-Luminosität im Vergleich dazu nur etwa 20 Ereignisse gleichzeitig erwartet. Eine wesentliche Folge davon ist, dass die derzeitigen Trigger-Methoden bei LHC aufgrund der erhöhten Anzahl an gleichzeitigen Ereignissen für SLHC nicht ausreichend sind.

Ziel dieser Arbeit ist es, Teile eines zusätzlichen L1-Spur-Trigger-Systems für den ATLAS Detektor zu simulieren, das nur Informationen aus dem Inneren Detektor zur Entscheidung verwendet. Das System ist als Hardware-Trigger konzipiert, das zusammen mit den L1-Trigger des Kalorimeter- und Myonsystems die erste Selektionsstufe des Triggersystems bildet.

Aufgabe des Spur-Triggers soll es sein, Teilchen mit einem Transversalimpuls von mindestens 10 GeV in einem Ereignis zu erkennen. Um den Entscheidungsprozess, ob und wo eine solche Spur in dem Event vorliegt, so schnell wie möglich zu machen, sollen die Muster der Spuren in den einzelnen Lagen mit vorgefertigten Vergleichsmustern verglichen werden. Auf diese Weise werden aufwendige Berechnungen vermieden werden, die in der kurzen Zeit von etwa 25 ns, die für eine Entscheidung zur Verfügung steht, nicht durchführbar wären.

Es soll unter anderem geprüft werden, ob die bereits existierende theoretischen Annahmen zu einem solchen Spur-Trigger sich in einer Simulation als richtig erweisen und ob die Anforderungen an den Trigger erfüllt werden können.

Für diese Simulation wird auf einer bereits existierenden Simulation des Inneren Detektor von ATLAS und einem später genauer beschriebenen Teilchengenerator für das Simulationsprogramm aufgebaut.

Ziel dieser Arbeit ist die Einteilung der betrachteten Detektorfläche in unabhängige Ausleseregionen und die Analyse von Ereignissen innerhalb dieser Regionen. Mit Hilfe der getroffenen Einteilungen und den vorhandenen Detektordesign insbesondere der Größenangaben und der verwendeten Komponenten werden auch Abschätzungen für die Hardware-Anforderungen an einzelne Komponenten des Triggers durchgeführt. Da das Design des Inneren Detektors für ATLAS bei SLHC zum Zeitpunkt dieser Arbeit jedoch noch nicht endgültig feststeht, können hier Abweichungen von den später tatsächlichen Werten entstehen.

1.1 Der Large Hadron Collider

Der Large Hadron Collider (LHC) ist der größte Beschleunigerring der Welt. Er befindet sich unter der Grenze zwischen Schweiz und Frankreich und hat einen Umfang von etwa 26.7 km [1]. Der Tunnel, in dem sich der LHC heute befindet, ist zwischen den Jahren 1984 und 1989 für den e^+e^- Beschleuniger LEP gebaut worden. Die vier großen Experimente am LHC sind CMS, ATLAS, ALICE und LHCb (siehe Abbildung 1.1). Im Jahre 2000 wurde LEP abgeschaltet und der Tunnel für den Einbau von LHC freigegeben.

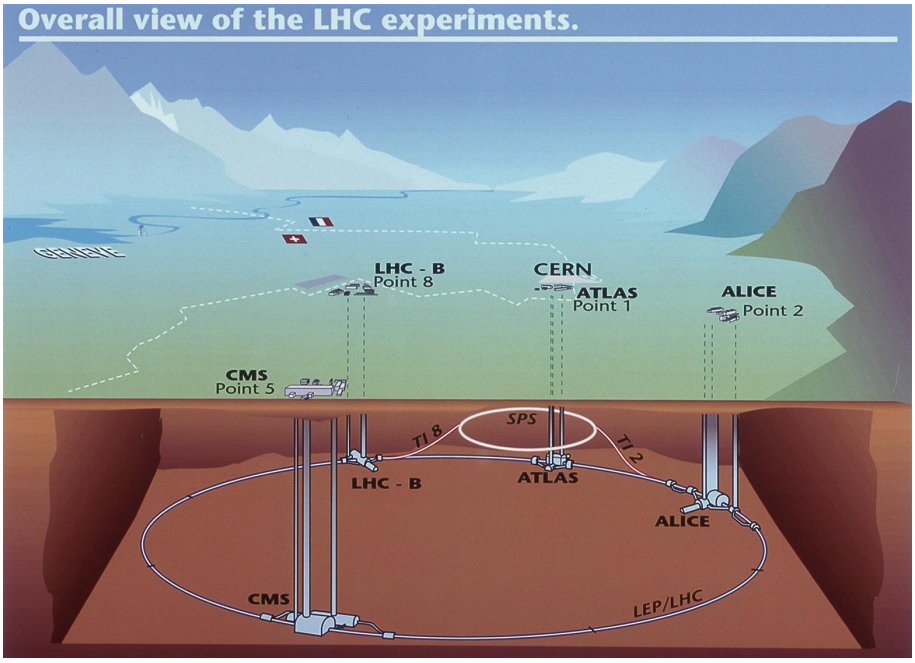

Abbildung 1: Der LHC und seine Experimente (Quelle: CERN).

Der LHC ist ein Proton-Proton-Beschleuniger. In ihm werden Protonenpakete - sogenannte Bunches - beschleunigt und miteinander zur Kollision gebracht. Dafür werden die Teilchenpakete in zwei getrennten Röhren in entgegengesetzten Richtungen beschleunigt. Nur bei den Experimenten werden die Röhren zusammengeführt, so dass die Protonenpakete miteinander kollidieren. Beim LHC werden so viele Pakete in den Ring eingeführt, dass diese mit einer Frequenz von 40 MHz bzw. alle 25 ns kollidieren. Die geplante Schwerpunktsenergie für die Kollisionen bei LHC ist 14 TeV.

1.2 Ziele des LHC

Mit den Experimenten vor LHC konnten bereits viele physikalische Phänomene, die zum Standardmodell der Teilchenphysik führten, bei niedrigen Energien beobachtet und analysiert werden. Manche Phänomene oder Teilchen wie das Higgs-Teilchen werden jedoch erst bei höheren Energien erwartet.

Mit dem LHC soll der Energiebereich, in dem unter anderem das Higgs-Teilchen vorhergesagt wird, analysiert werden. Ein Experiment, das sich an der Suche nach dem Higgs-Teilchen beteiligt, ist das ATLAS Experiment.

Die Suche nach dem Higgs-Teilchen

Mit der Entdeckung des Higgs-Teilchens soll eine entscheidende Lücke des Standardmodells geschlossen werden. Es wird benötigt, um die Massen der Teilchen im Standardmodell zu erklären.

Der Energiebereich, in dem das Higgs-Teilchen zu suchen ist, ist durch theoretische Schranken und frühere Experimente eingegrenzt worden. Durch Experimente am LEP wurde eine untere Grenze $114\,\text{GeV} < m_H$ für die Masse m_H des Higgs-Teilchens bestimmt [2]. Im Bereich von 170 GeV wurden Erkenntnisse durch die Tevatron-Experimente gewonnen. Über die Konsistenz mit dem Standardmodell konnte eine theoretische obere Grenze im Bereich von 800 GeV für die Masse des Higgs-Teilchens bestimmt werden. Mit den Experimenten am LHC soll der Massenbereich $114\,\text{GeV} < m_H < 800\,\text{GeV}$ untersucht werden.

Supersymmetrie

Das Modell der supersymmetrischen Teilchen geht davon aus, dass jedem Standardteilchen ein supersymmetrisches Teilchen zugeordnet ist. Die Massengrenze, ab der diese Teilchen erwartet werden, liegt aufgrund der Untersuchungen am Tevatron-Beschleuniger im Bereich zwischen 300 GeV und 400 GeV [2]. Auch dieser Massenbereich kann mit LHC untersucht werden. Mit diesen deutlich schwereren Teilchen könnte die Dunkle Materie erklärt werden.

1.3 Der SLHC

Durch das geplante LHC-Upgrade soll die Luminosität von 10^{34} $cm^{-2}s^{-1}$ um einen Faktor 10 auf 10^{35} $cm^{-2}s^{-1}$ erhöht werden. Die Möglichkeiten dafür werden in Abschnitt 2.1 vorgestellt. Die Erhöhung der Luminosität ermöglicht eine höhere statistische Signifikanz bei der Untersuchung seltener Prozesse. Diese ist notwendig, wenn man bei LHC etwas findet und näher untersuchen will, aber auch, wenn man mit den Mitteln von LHC nicht in der Lage ist, ein neues schweres Teilchen zu entdecken, und für eine mögliche Entdeckung mehr Daten benötigt werden.

Durch die Erhöhung der Luminosität können bis zu 400 Protonenkollisionen gleichzeitig stattfinden. Dies führt jedoch dazu, dass bei einer Kollision deutlich mehr Spuren ausgewertet werden müssen.

In Abbildung 1.3 werden die Spuren einer (simulierten) Kollision am LHC mit 5 Ereignissen - was einer Luminosität von etwa $0.2 \cdot 10^{34}$ $cm^{-2}s^{-1}$ entspricht - und einer (simulierten) Kollision am Super-LHC mit 400 Ereignissen - was einer Luminosität von etwa $10 \cdot 10^{34}$ $cm^{-2}s^{-1}$ entspricht - gegenübergestellt. Für die Design-Luminosität bei LHC erwartet man etwa 20 Ereignissen. In Abbildung 3 wird eine bei ATLAS aufgenommene Kollision mit 2 rekonstruierten Ereignissen gezeigt.

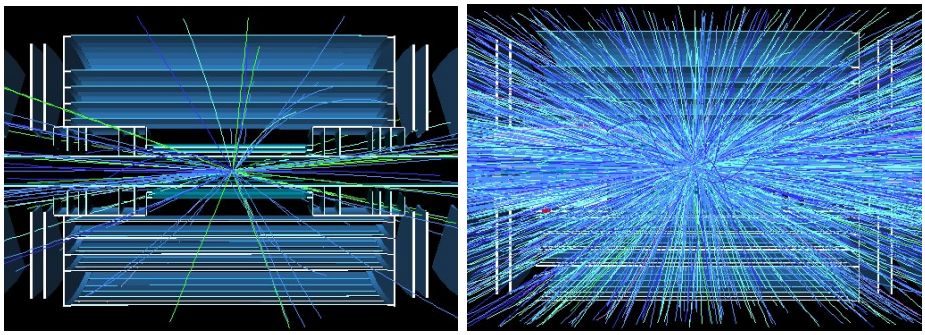

Abbildung 2: Vergleich einer Kollision bei LHC (links) und SLHC (rechts) (simuliert). [3]

Aufgrund der erhöhten Menge an Strahlung im Detektor und der Datenmenge pro Kollision müssen Teile des ATLAS-Detektors neu entwickelt werden. Aus technischen Gründen ist es zum Beispiel notwendig den Inneren Detektor von ATLAS für SLHC neu zu konstruieren (siehe Abschnitt 2.3.1). Auch das Trigger-System muss den neuen Anforderungen angepasst werden.

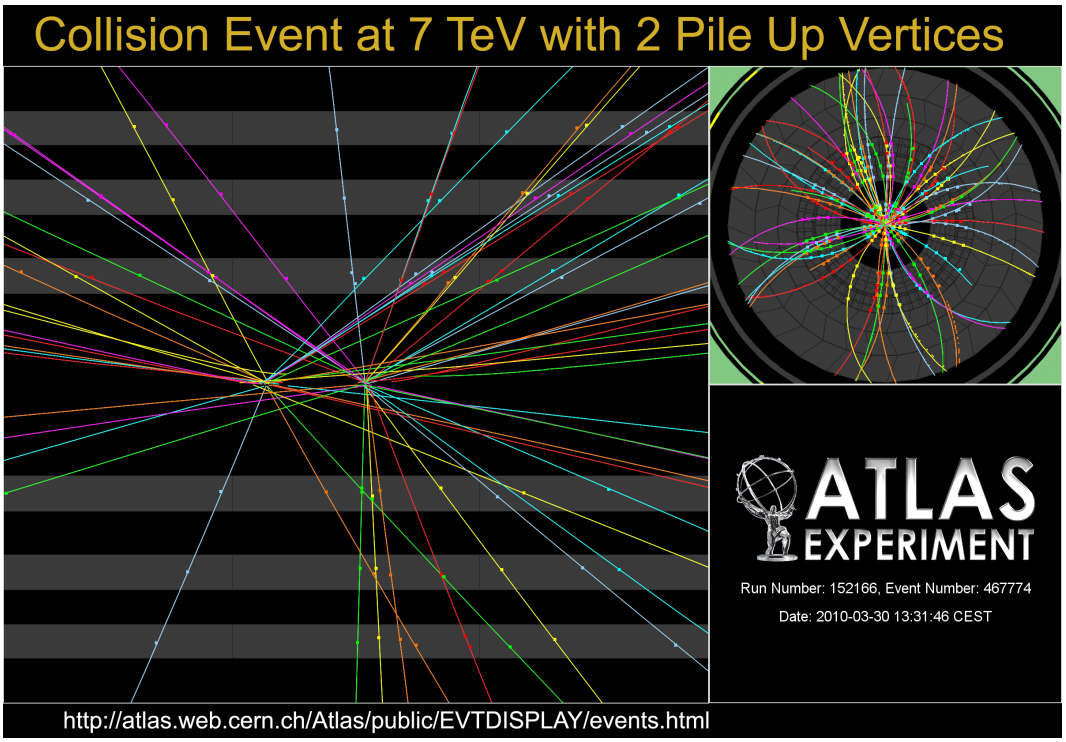

Abbildung 3: Eine Kollision bei ATLAS mit 2 Pileup (Experiment). [4]

2 Allgemeine Grundlagen

2.1 Luminosität

Die Anzahl Ereignisse N pro Zeiteinheit, die bei einem Beschleuniger-Experiment beobachtet werden können, ist abhängig von der Luminosität L am Beschleuniger und dem Wirkungsquerschnitt σ des Prozesses, den man untersuchen will (Gleichung (1)).

$$\frac{dN}{dt} = L \cdot \sigma \tag{1}$$

Die Luminosität hat die Einheit $\mathrm{cm}^{-2}\mathrm{s}^{-1}$.

Sie ist unter anderem abhängig von der Anzahl Protonenpakete (Bunches) n_b sowie der Anzahl Protonen pro Paket N_b und der Umlauffrequenz f_r (Gleichung (2) [2]).

$$L = \frac{f_r \gamma}{4\pi} \frac{N_b^2 n_b}{\epsilon_n \beta^\star} \cdot F \tag{2}$$

ϵ_n bezeichnet die normierte Emmitanz und $\beta^\star$ bezeichnet den Wert der Betatron Funktion am Interaktionspunkt. F ist ein Faktor, der den geometrischen Verlust an Überlappung zwischen zwei Protonenpaketen bei der Kollision unter einem gegebenen Winkel angibt. Es gilt $\mathrm{F} < 1$.

Zur Erhöhung der Luminosität muss man also, wenn die anderen Größen unverändert bleiben, nach Gleichung (2) die Anzahl Protonenpakete n_b, die Anzahl Protonen pro Paket N_b oder die geometrische Überlappung der Pakete erhöhen, oder es muss ϵ_n oder $\beta^\star$ reduziert werden.

2.2 Der ATLAS-Detektor

Der Name ATLAS steht für A Toroidal LHC ApparatuS. Der ATLAS Detektor ist als "general purpose" Experiment unter anderem für die Suche nach dem Higgs-Teilchen sowie anderer unbekannter Teilchen konzipiert. Der Detektor besteht aus einem Inneren Detektor, dem Magnetsystem, dem Kalorimetersystem und einem Myonspektrometer. Der zentrale Detektor ist zylinderförmig aufgebaut.

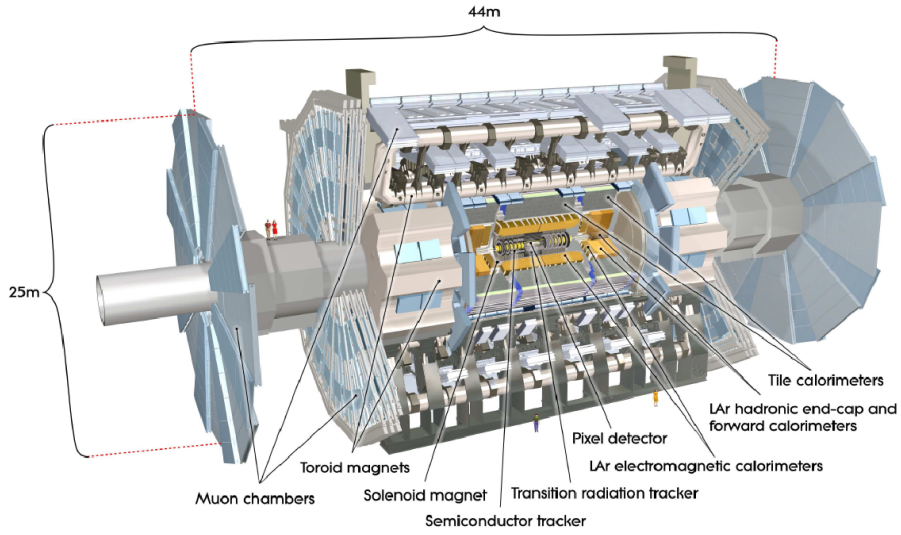

Abbildung 4: Der ATLAS-Detektor. [1]

2.2.1 Der Innere Detektor

Der Innere Detektor dient der Vertex- und Spurrekonstruktion. Durch ein solenoidales Magnetfeld von 2 Tesla werden die Spuren geladenener Teilchen im Magnetfeld gekrümmt, wodurch der transversale Impuls der Teilchen bestimmt werden kann.

Der Innere Detektor besteht aus 3 Einheiten:

- Pixel-Detektor
- Semiconductor Tracker (SCT)
- Transition Radiation Tracker (TRT)

Der Pixel-Detektor, der sich am nächsten an der Strahlachse befindet, besitzt eine hohe Granularität, wodurch präzise Messungen nahe am Wechselwirkungspunkt ermöglicht werden.

Der Semiconductor Tracker besteht aus 8 Lagen Silizium-Streifen-Detektoren. Sie haben eine Auflösung von 16 µm radial zur Strahlachse und eine Auflösung von 580 µm in Strahlrichtung.

Der Transition Radiation Tracker (TRT) ist eine Kombination aus Übergangsstrahlungsdetektor und Driftrohren. Die Driftrohre sind mit Xenongas gefüllt und umgeben von einem speziellen Schaum aus Polyethylen, an dessen Übergang Elektronen elektromagnetische Strahlung im Röntgenbereich abgeben. Diese Strahlen werden im Xenongas absorbiert. Der Transition Radiation Tracker kann somit zu Identifikation von Elektronen genutzt werden, da Elektronen häufiger Übergangsstrahlung emittieren als Pionen gleicher Energie.

Die Anordnung der Detektorlagen ist in Abbildung 2.2.1 rechts gezeigt.

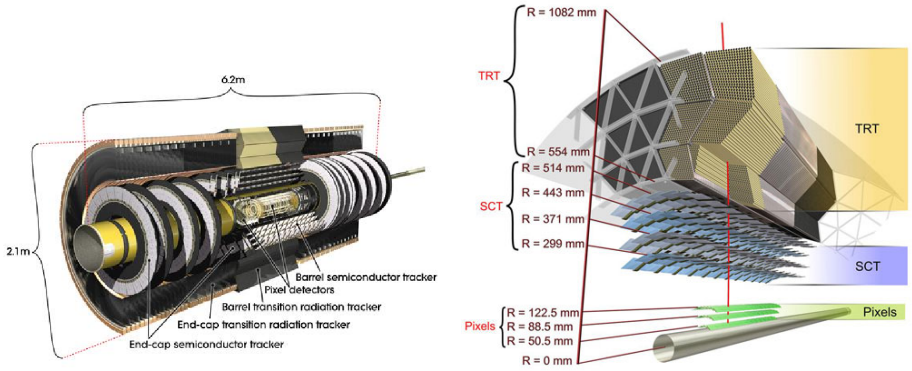

Abbildung 5: Der Innere Detektor. [1]

2.2.2 Das Magnetsystem

Im ATLAS Detektor befinden sich zwei voneinander unabhängige Magnetsysteme.

Das eine Magnetsystem befindet sich innerhalb des Inneren Detektors und hat eine Stärke von 2 Tesla. Es wird zur Bestimmung des transversalen Impulses p_T der Teilchen genutzt.

Das Myonspektrometer befindet sich in einem toroidalen Magnetfeld. Dieses besitzt eine mittlere Stärke von 0.3 Tesla. Es wird zur Vermessung des Impulses der Myonen verwendet.

2.2.3 Das Kalorimetersystem

Das Kalorimetersystem setzt sich aus dem elektromagnetischen und dem hadronischen Kalorimeter zusammen. Sie dienen zur Energiemessung und zur Identifikation der Teilchen. Das elektromagnetische Kalorimeter ist in Verbindung mit den Ergebnissen des

Inneren Detektors für die Identifikation von elektromagnetischen wechselwirkenden Teilchen wie Elektronen und Photonen verantwortlich. Es ist ein Blei-Flüssigargon-Samplingkalorimeter, das akkordeonförmig aufgebaut ist. Auf diese Weise ist eine vollständige Symmetrie in ϕ gegeben. Es ist im Inneren des hadronischen Kalorimeters aufgebaut.

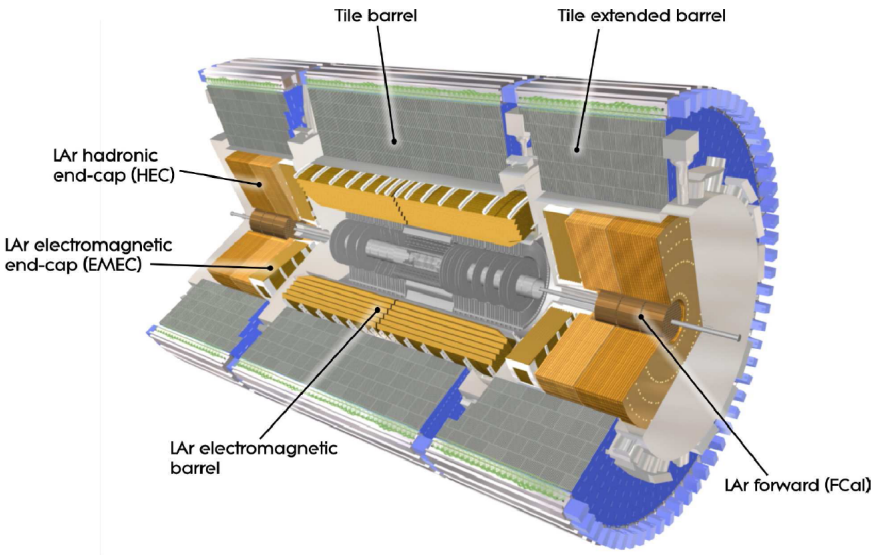

Abbildung 6: Das Kalorimetersystem. [1]

Das hadronische Kalorimeter besteht im Zentralbereich aus einem Eisen-Absorber Kalorimeter mit Plastikszintillatoren als Nachweismedium. In den Endkappenbereichen besteht es aus einem Flüssig-Argon Kalorimeter mit Kupfer-Absorbtionsplatten.

Die Informationen aus den Kalorimetern werden auch für den L1-Trigger verwendet.

2.2.4 Das Myonspektrometer

Aufgrund ihrer geringen Wechselwirkung mit Materie können die meisten Myonen das Kalorimetersystem ungehindert durchdringen.

Das Myonspektrometer bildet die äußerste Einheit des Detektors. Es dient der Impulsmessung von Myonen. Teile der Informationen des Myonspektrometers werden außerdem für den L1-Trigger von ATLAS verwendet. Das Myonspektrometer, das aus Driftrohren besteht, nimmt den größten Raum im ATLAS-Detektor ein.

2.2.5 Das Triggersystem

Das ATLAS Triggersystem bei LHC, dargestellt in Abbildung 2.2.5, besteht aus zwei Trigger-Stufen, wovon die erste als Hardware-Trigger und die zweite als Software-Trigger arbeitet, und einem Event Filter, der ebenfalls als Software-Trigger arbeitet. Ein Hardware-Trigger auf der ersten Stufe ist notwendig, da ein Software-Trigger die Daten nicht mit einer Frequenz von 40 MHz verarbeiten kann und daher die Daten zunächst reduziert werden müssen.

Der L1-Trigger bei ATLAS verwendet hierfür im wesentlichen nur Informationen aus den Kalorimetern (L1Calo) und dem Myonsystem (L1Myon). Seine Aufgabe ist es, die Datenrate auf eine Frequenz von $\sim$ 75 kHz zu reduzieren. Er nutzt bei LHC keine Information aus dem Inneren Detektor.

Der L2-Trigger ist ein Software-Trigger, der die Daten auf eine Frequenz von etwa 1 kHz reduziert und sie dann an den Event-Filter übergibt. Beim L2-Trigger werden Daten aus Regionen genutzt, die nach den Informationen des L1-Triggers von besonderem Interesse sind (Regions of Interest (RoI)).

Der Eventfilter reduziert die Daten dann auf eine Frequenz von etwa 200 Hz, was gespeichert werden kann. Er nutzt die volle Detektorinformation.

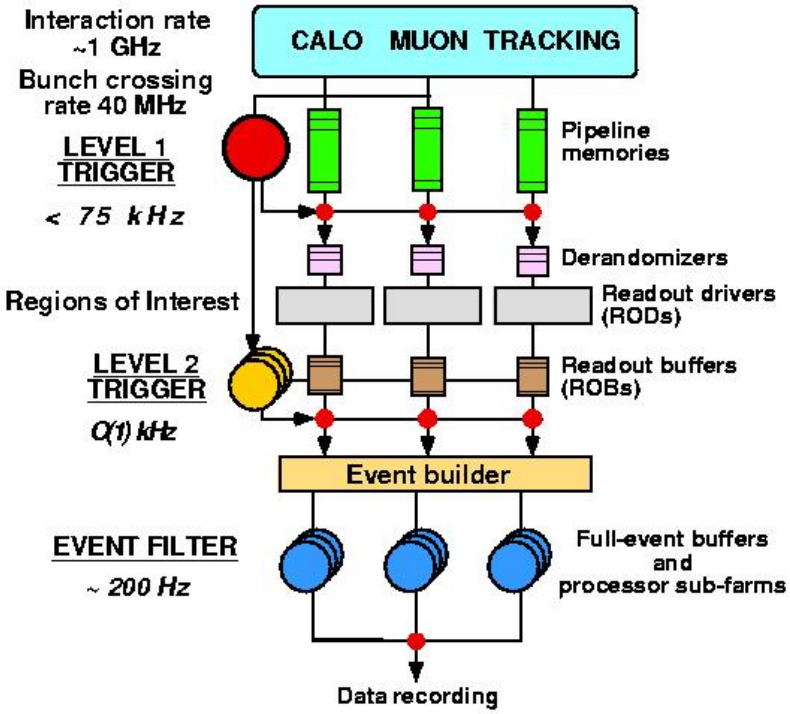

Abbildung 7: Das Triggersystem. [5]

2.3 Änderungen des ATLAS-Detektors bei SLHC

2.3.1 Der Innere Detektor bei SLHC

Für den neuen Inneren Detektor wird in dieser Simulation ein Design mit 4 Silizium-Pixel-Detektorlagen mit den Radien 5, 11, 16 und 21 cm, kurzen Silizium-Streifen-Detektorlagen mit den Radien von 38.0, 50.1 und 62.2 cm und langen Silizium-Streifen-Detektorlagen mit den Radien 74 und 100 cm angenommen. Die Silizium-Streifen-Lagen sollen dabei Doppellagen im Abstand von 4.5 mm sein [6].

Die Abmessungen der Streifen auf den einzelnen Lagen sind:

- Pixel-Lagen: 50 µm x 250 µm
- Kurzstreifen-Lagen: 80 µm x 2.4 cm
- Langstreifen-Lagen: 80 µm x 9.8 cm

Die Doppellagen sollen wie auch beim bisherigen ATLAS-Detektor vor allem der besseren Auflösung in z-Richtung dienen. Das Magnetfeld im Inneren Detektor wird wie auch beim LHC durch einen Solenoid Magneten mit 2 T gegeben sein.

Einen Transition Radiation Tracker soll es im Inneren Detektor bei SLHC voraussichtlich nicht geben.

2.3.2 Das Konzept des Spur-Triggers für SLHC

Der L1-Spur-Trigger soll beim neuen ATLAS Triggersystem an erster Stelle parallel mit L1Calo und L1Myon stehen und nur Informationen aus dem Inneren Detektor nutzen. Da der Spur-Trigger auch bei einer sehr hohen Datenrate arbeiten muss, soll auch er in Hardware implementiert werden.

Ziel des Spur-Triggers ist es, hochenergetische Spuren in Ereignissen zu erkennen und diese Information für die nächsten Trigger-Level bereit zu stellen. In dieser Arbeit wird die Grenze zu einer hochenergetischen Spur auf 10 GeV gesetzt.

Das Konzept des Spur-Triggers ist hierbei:

Im Inneren Detektor wird durch die verschiedenen Spuren, die bei einer Kollision entstehen, ein Treffermuster auf den verschiedenen Lagen erzeugt. Diese Muster werden Pattern genannt. Um schnell entscheiden zu können, ob sich in dem Pattern Trefferkombinationen befinden, die zu einer hochenergetischen Spur gehören, werden im Vorfeld die Muster hochenergetischer Spuren bestimmt und später als Vergleichsmuster genutzt. Diese Vergleichsmuster werden Templates genannt.

Die Templates werden in CAMs (siehe Abschnitt 2.4) gespeichert. Diese ermöglichen es, ein eingehendes Bitmuster direkt mit allen gespeicherten Einträgen zu vergleichen

und so mit hoher Geschwindigkeit den Vergleich des Pattern (bzw. eines Ausschnitts des Pattern) mit allen zugehörigen Templates vorzunehmen.

Das Grundprinzip des Entscheidungsprozesses ist in Abbildung 2.3.2 gezeigt.

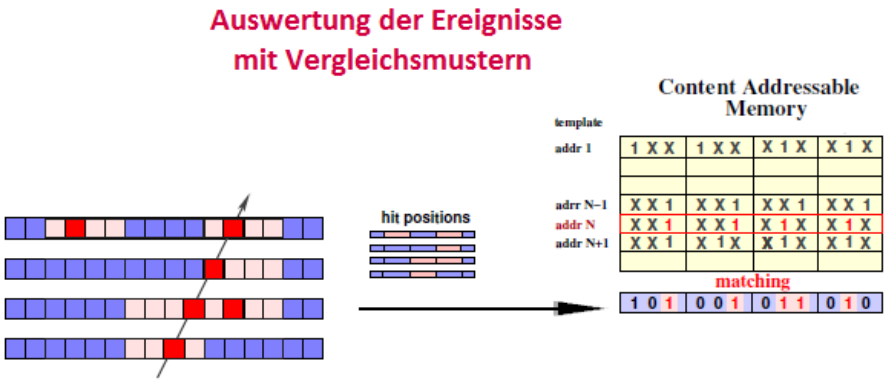

Abbildung 8: Mustersuche mittels CAM. [7] (bearbeitet)

Aufgrund der begrenzten Inputbreite eines CAM, kann nicht die gesamte Information aus dem Inneren Detektor an einen CAM übergeben werden. Da die einzelnen Spuren jedoch nur einen kleinen Raumwinkel im Detektor einnehmen, ist das auch nicht notwendig. Vielmehr ist es sinnvoll, den Inneren Detektor so in Ausleseregionen einzuteilen, dass jede hochenergetische Spur in mindestens einer Ausleseregion vollständig enthalten ist. Die Analyse eines Ereignisses kann dann unabhängig für die einzelnen Ausleseregionen erfolgen, und die Datenmenge für jede Region reduziert sich mit der Anzahl der Ausleseregionen.

2.4 Content Addressable Memory (CAM)

Wenn man ein bestimmtes Bitmuster mit vordefinierten Mustern vergleichen will, bietet sich ein Content Addressable Memory (CAM) zur Speicherung der Vergleichsmuster an. Bei dieser Art des Speichers ist es möglich ein Muster aus dem Detektor mit allen Vergleichsmustern nahezu gleichzeitig zu vergleichen. Die Form des Vergleichs ist ähnlich einem Hashing in einem RAM (Random Accesss Memory). Der Hauptunterschied ist, dass beim Hashing in einem RAM zu jeder möglichen Adresse ein Speicherplatz (mindestens ein Bit) im RAM vorhanden sein muss. Der Speicher, der bei einem RAM benötigt wird, steigt somit exponentiell mit der genutzten Adresslänge [8].

Der Vorteil des CAM ist in diesem Fall, dass je nach Art des CAM Adresslängen im Bereich von 500 Stellen möglich sind. Der einem CAM zur Verfügung stehende Speicher limitiert, wie viele Templates in dem CAM gespeichert werden können. Ist die Zahl der Templates klein gegenüber der Anzahl möglicher Muster, so hat diese Form der Speicherung den großen Vorteil, dass die nicht gültigen Muster nicht im CAM gespeichert werden müssen.

Es gibt zwei verschiedene Arten von CAM-Implementierungen, die hier genutzt werden können. Die eine Art ist, das Prinzip des CAM in ein FPGA einzubetten. In diesem Fall kann man bei Adresslängen von 64 Bit etwa 10000 Templates mit einem FPGA verwalten (z.B. Altera Stratix IV). Der andere Weg sind kommerzielle Network Search Engines (Netlogic), die bei einer Adresslänge von 586 Bit etwa 64000 Templates verwalten können.

Es gibt binäre und tertiäre CAMs. Bei einem binären CAM ist es wie beim RAM nur möglich 1 oder 0 für eine Stelle im Vergleichsmuster zu speichern. Ein tertiärer CAM bietet zusätzlich die Möglichkeit ein X zu speichern, das für 1 oder 0 gleichzeitig steht. Bei einem Vergleich mit einem X erhält man somit immer „wahr“ als Antwort.

In Abbildung 2.4 ist der Vergleich in einem CAM schematisch dargestellt.

Für dieses Beispiel wird von unten das Suchwort (Adresse) 01101 an den CAM angelegt. Über die Suchleitungen (search lines) gelangt es nahezu zeitgleich zu allen gespeicherten Templates im CAM. In den einzelnen Zellen werden alle Stellen des Templates parallel mit den Stellen des Suchworts verglichen. Zu jedem Template gibt es eine Trefferleitung (matchline), die in diesem Bild horizontal verläuft. Diese Leitung ist aktiv, falls alle Zellen des zugehörigen Templates mit dem Suchwort übereinstimmen. Sonst wird die Leitung inaktiv. Alle Trefferleitungen führen zu einem Encoder, der im Falle von mehreren Treffern einen Ausgangsadresse bestimmt. In diesem Beispiel gibt der Encoder immer das (bei Durchnummerierung der Templates) Template mit der kleinsten Nummer zurück. Allgemein benötigt der Encoder eine Funktion, auf der seine Entscheidung beruht.

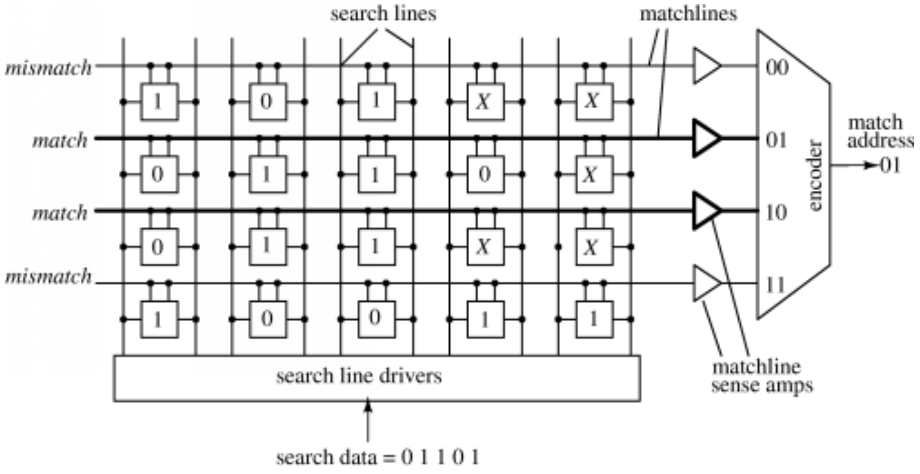

Abbildung 9: Aufbau eines CAM. [9]

Da der Vergleich der Eingangsadresse mit den gespeicherten Templates sowohl für alle Templates als auch für alle Stellen im Template nahezu zeitgleich erfolgt, kann ein solcher Vergleich in sehr kurzer Zeit erfolgen.

3 Grundlagen der Simulation

3.1 Betrachtete Layouts in dieser Simulation

Wie bereits erwähnt soll ein möglicher L1-Spur-Trigger für ATLAS-SLHC untersucht werden. Dieser Spur-Trigger soll Spuren, die zu einem Teilchen mit einem Impuls über einer vorgegebenen Grenze gehören, in einem Ereignis erkennen und das Ereignis beim Auffinden solcher Spuren als von Interesse kennzeichnen. Für den Spur-Trigger werden zwei Layouts betrachtet.

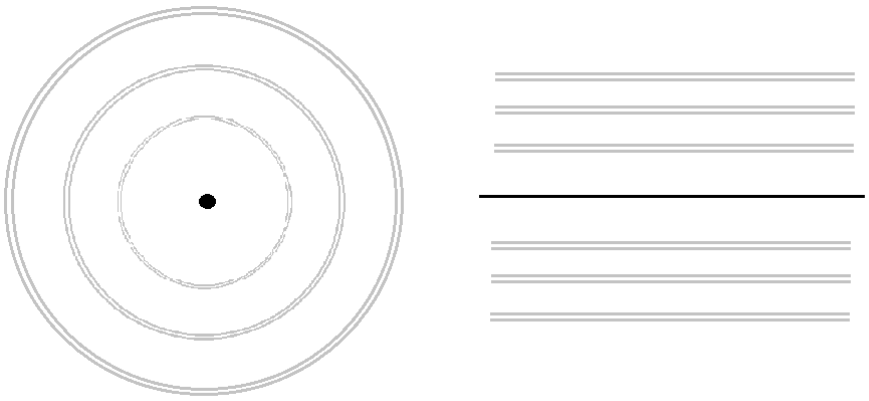

Abbildung 10: Betrachtete Detektorlagen: Die Silizium-Kurzstreifen-Lagen sind grau dargestellt. Die Strahlachse ist schwarz dargestellt (links: r-ϕ-Ebene rechts: r-z-Ebene).

Im ersten Layout werden die 3 Kurzstreifen-Lagen des Inneren Detektors (siehe Abbildung 10) verwendet. Diese werden allerdings nur als Einfachlagen betrachtet. Die zusätzlichen Informationen, die aus den Doppellagen gewonnen werden könnten, werden nicht genutzt. Aufgrund der Nähe der Silizium-Pixel-Lagen zur Strahlachse und der daraus resultierenden hohen Okkupanz auf diesen Lagen sollen diese Lagen nicht für den Spur-Trigger genutzt werden und werden daher auch in dieser Simulation nicht genutzt.

Das zweite Layout, das in Kapitel 10 diskutiert wird, interpretiert jede Doppellage als zwei eigenständige Lagen sodass ein 6-Lagen-Layout entsteht. Im Gegensatz zum 3-Lagen-Layout haben beim 6-Lagen-Layout nicht mehr alle Lagen den gleichen Abstand voneinander. Da diese Eigenschaft der Lagen jedoch bei der Betrachtung des 3-Lagen-Layouts nicht genutzt wird, können die beiden Designs weitgehend gleich behandelt werden.

Für jedes dieser Designs sollen in dieser Arbeit vor allem Effizienz und Fake-Rate untersucht werden. Als Effizienz wird hier bezeichnet, wie häufig eine hochenergetische Spur vom Spur-Trigger als solche erkannt wird. Diese Größe ist sehr wichtig und sollte

so hoch wie möglich sein, wenn der Trigger seine Hauptaufgabe erfüllen soll. Die Fake-Rate gibt an, wie häufig der Trigger pro Ereignis eine Kombination von Treffern auf den Lagen als hochenergetische Spur interpretiert, ohne das tatsächlich eine solche Spur vorlag. Ist diese Zahl zu hoch, wird der Trigger alle Ereignisse als von Interesse kennzeichnen und somit keine Reduktion ermöglichen.

Neben der Bestimmung von Effizienz und Fake-Rate wird für das 3-Lagen-Layout auch analysiert, wie häufig die Information eines Chips pro Ereignis zur Weiterverarbeitung in verschiedenen Ausleseregionen dupliziert werden muss. Dieser Wert ist beim Auslesen über Kabel vor allem von Bedeutung, da in diesem Fall eine Verbindung von einem Auslese-Chip zu jeder Region bestehen muss, die dessen Information für die Analyse benutzt. Im selben Kapitel wird auch die Größe eines Musters für das 3-Lagen-Layout für verschiedene Regionsgrößen und Arten der Zuordnung untersucht. Dieser Wert ist für die in dieser Arbeit durchgeführte Simulation weniger von Bedeutung, ist aber für die Implementierung des Triggers in Hardware wichtig (siehe Abschnitt 2.4).

Eine Größe, die in dieser Arbeit nicht analysiert werden kann, ist das Zeitverhalten des Triggers. Da diese Arbeit auf einer reinen Software-Simulation beruht, kann hiermit das Zeitverhalten in Hardware nicht untersucht werden.

3.2 Das Generieren von Ereignissen

Die Ereignisse, die in dieser Simulation untersucht werden, werden mit einem toy-Monte-Carlo Generator erzeugt. Es wird nur der Innere Detektor ohne Endkappen betrachtet, wobei dieser als zylinderförmig angenommen wird. Beim Generieren der Teilchen wird angenommen, dass die Teilchen in einem Bereich auf der Strahlachse im Detektor entstehen, der die Ausdehnung von 15 cm in der Mitte der Strahlachse des Detektors hat. Die Teilchen sind hierbei auf diesem Bereich gleichverteilt. Die x- und y-Koordinate des Entstehungspunktes der Teilchen sind hierbei auf 0 gesetzt. Die Winkelverteilung der Teilchen in ϕ und θ sind in Abbildung 11 gezeigt.

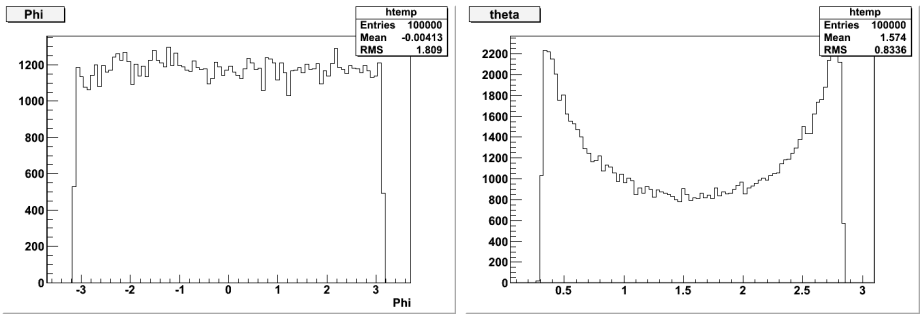

Abbildung 11: Benutzte Winkelverteilung für ϕ (links) und θ (rechts).

Die Verteilung der Teilchen ist flach in ϕ und in der Pseudorapidität η. Daraus ergibt sich für θ die dargestellte Verteilung, die ebenfalls in der Verteilung der Okkupanz für die einzelnen Regionen im Detektor zu erkennen ist.

Die Impulsverteilung der Teilchen wurde nach Pythia 6 für Proton-Proton minimum Bias Kollisionen bei einer Schwerpunktsenergie von 14 TeV gewählt und ist in Abbildung 12 dargestellt.

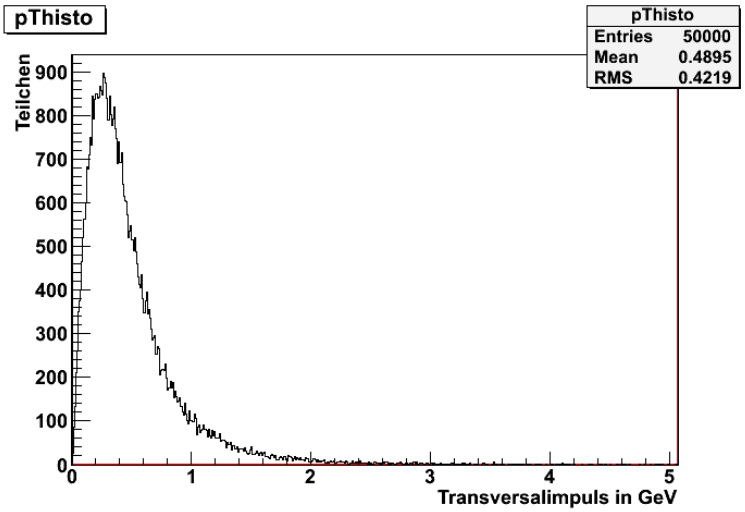

Abbildung 12: Benutzte Transversalimpulsverteilung.

Es gilt zu beachten, dass in dieser Simulation - aufgrund des Radius der innersten Lage - nur Spuren ab einem Impuls von 114 MeV die erste Detektorlage erreichen können. Die Anzahl Spuren, die für ein Ereignis simuliert wurden, wurde so gewählt, dass sie zu der für SLHC erwarteten Okkupanz von etwa 2 % auf der innersten Lage passt.

Bei dieser Simulation erzeugt ein Teilchen auf jeder Lage in genau einer Detektorzelle einen Treffer. Eine Ineffizienz des Detektormaterials wird in dieser Simulation nicht berücksichtigt.

Da in dieser Simulation immer nur eine Region betrachtet wird und die Okkupanz nicht in allen Regionen gleich ist, müssen die Ergebnisse für die Fake-Rate entsprechend der Okkupanz für die anderen Regionen umgerechnet werden. Eine einfache Multiplikation mit der Anzahl der Regionen würde hier zu einem falschen Ergebnis führen.

4 Definition der Ausleseregionen

4.1 Motivation

Pro Kollision müssen mehrere zehntausende Spuren untersucht werden. Da die Trigger-Entscheidung pro Spur immer nur auf Informationen aus einem kleinen lokalen Bereich des Inneren Detektors beruht, ist es sinnvoll, die Analyse auf mehrere Recheneinheiten aufzuteilen, die jeweils nur Informationen aus einem kleinen Teilbereich des Inneren Detektors verwenden. Diese Bereiche, die die Regionen definieren, die von einer Recheneinheit ausgelesen werden, werden im Folgenden als Ausleseregionen bezeichnet.

Ziel des ersten Teils dieser Arbeit ist es, etwa 10000 voneinander unabhängige Ausleseregionen zu definieren, die mit so wenig Überlappung wie möglich auskommen, jedoch insgesamt alle Spuren im Detektor erfassen, die eine positive Triggerentscheidung auslösen sollen, also in dieser Arbeit solche, die einen Transversalimpuls p_T größer als 10 GeV haben. Zur Bestimmung des Transversalimpulses einer Spur muss man die Spurkrümmung in der r-ϕ-Ebene betrachten, die durch das angelegte Magnetfeld bewirkt wird. Da dazu die Information aus allen Detektorlagen benötigt wird, muss sich die Information zu einer Spur aus allen Detektorlagen in einer Ausleseregion befinden. Nur dann kann die Spur in der Region vollständig registriert werden. Bei der Definition der Regionen wird keine Streuung der Teilchen am Detektormaterial bzw. der Teilchen untereinander berücksichtigt.

4.2 Die Einteilung in ϕ-Richtung

In der r-ϕ-Ebene bilden die Detektorlagen in der Simulation konzentrische Kreise. Im Folgenden wird ein einfaches Design mit drei Lagen mit den Radien 38, 50.1 und 62.2 cm betrachtet. Ein Doppellagenlayout, das zu sechs Lagen führt, wird in Kapitel 10 betrachtet. Weitere Designs werden in [7] diskutiert.

Die Grundidee besteht darin, die Ausleseregionen in der r-ϕ-Ebene durch gleich große Winkelbereiche zu definieren. Die Einteilung erinnert dann an eine Torte, die in n gleiche Tortenstücke aufgeteilt wird. Der Winkelbereich eines „Tortenstücks“ wird als $\Delta\phi$ bezeichnet, also $n = 360°/\Delta\phi$. Da der Entstehungsort in der r-ϕ Ebene durch die geringe Ausdehnung eines Teilchenpakets in x- und y-Richtung nahezu als Punktquelle angesehen werden kann, ist die Einteilung hier entsprechend einfach.

Ein Problem ist jedoch, dass die Spuren in dieser Ebene gekrümmt sind und somit auf den verschiedenen Detektorlagen unterschiedliche ϕ-Werte haben. Wenn man garantieren will, dass man dennoch stets die Information zu einer Spur auf allen Lagen in einer Ausleseregion finden kann, ergibt sich direkt eine Bedingung an die minimale Größe der Region in der r-ϕ-Ebene in Abhängigkeit von dem minimalen transversalen Impuls, den man detektieren will.

Die Berechnung dieser minimalen Größe in ϕ kann durch eine einfache geometrische Überlegung und die Beziehung

$$P[\text{ GeV}] = 0.3 \cdot B[\text{ T}] \cdot R[\text{ m}] \tag{3}$$

erfolgen, wobei P der Impuls des Teilchens ist, B das angelegte Magnetfeld und R der Radius der Teilchenbahn.

Die Radien der einzelnen Detektorlagen werden mit r_i bezeichnet, wobei r_1 der Radius der innersten Lage und r_3 der Radius der äußersten Lage ist. Für den minimalen Winkelbereich $\phi_{min}(p_T)$, den eine Region mindestens abdecken muss, damit eine Spur mit dem Transversalimpuls p_T vollständig in einer Region registriert werden kann, gilt:

$$\phi_{min}(p_T) = \arccos\left(\frac{0.3 \cdot B[\text{ T}] \cdot r_1[\text{ m}]}{2 \cdot p_T[\text{ GeV}]}\right) - \arccos\left(\frac{0.3 \cdot B[\text{ T}] \cdot r_3[\text{ m}]}{2 \cdot p_T[\text{ GeV}]}\right) \tag{4}$$

Die Gleichung 4 ergibt sich aus geometrischen Überlegungen gemäß Abbildung 13.

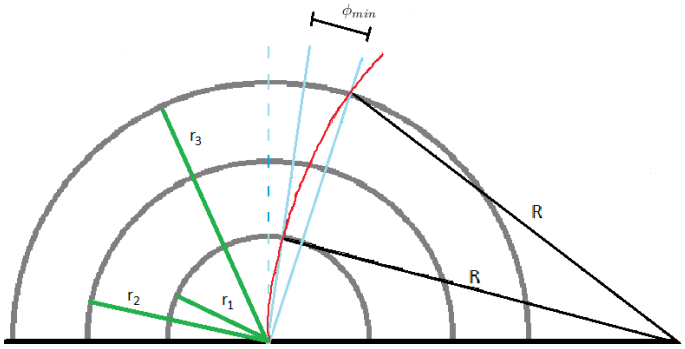

Abbildung 13: Bestimmung des minimalen Winkelbereichs in ϕ.

Da die Schnittpunkte der (rot dargestellten) Spur mit den Detektorlagen alle den Abstand R zum Mittelpunkt des Kreises haben, auf dem die Spurbahn liegt, findet man hier mehrere gleichschenklige Dreiecke. Halbiert man diese Dreiecke an der Symmetrieachse, erhält man Dreiecke mit einem rechten Winkel. Die Winkel in diesen Dreiecken können nun über die Information der Länge der Seiten bestimmt werden.

Da man letztlich nur die Differenz der Winkel benötigt, bei denen die Spur die innerste bzw. die äußerste Lage geschnitten hat, ergibt sich Gleichung 4.

Wird die Größe der Regionen in der r-ϕ Ebene durch den Winkel $\phi_{min}(p_T)$ bestimmt, so verläuft eine Spur mit dem Transversalimpuls p_T ganz in einer Region, falls Spur und Region „günstig“ zueinander liegen, d.h. falls die Spur die innerste Lage am zur Krümmung passenden Randpunkt trifft. Um sicher zu stellen, dass eine Spur stets ganz in mindestens einer der Regionen verläuft, müssen sich die Regionen jedoch mindestens um den Winkel $\phi_{min}(p_T)$ überlappen.

Das Prinzip der Überlappung ist in Abbildung 14 dargestellt.

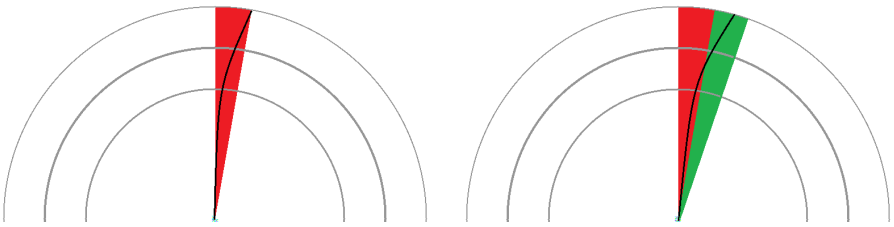

Abbildung 14: Konstruktion der Überlappung in ϕ.

Im linken Teil ist die Region (rot dargestellt) genau so groß, dass die Spur mit dem gewünschten Impuls in die Region passt, im rechten Bild läuft die Spur über den Rand der roten Region hinaus. Nur wenn man die grüne Region als Überlappung zu der roten Region hinzufügt, kann man sicherstellen, dass die Spur immer in der Nachbarregion vollständig registriert wird, wenn sie über den rechten Rand der Region hinausläuft. Der linke Rand würde durch die Region links von der roten Region abgedeckt, die die rote Region als Überlappung nutzen würde.

Es ist natürlich möglich, $\Delta\phi$ größer als $\phi_{min}(p_T)$ zu wählen. In diesem Fall würde es nur am Rand zu einer Überlappung kommen, wogegen der Mittelteil der Region nur zu einer Region gehören würde.

Im Folgenden werden die Regionen immer in den Farben Rot, Grün und Blau dargestellt. Die Überlappung wird dann durch die Farbaddition in den Farben Gelb, Cyan und Magenta dargestellt.

Abbildung 15 zeigt die Einteilung der Regionen in der r-ϕ-Ebene. Im linken Bild wurde $\Delta\phi$ gleich der Größe der Überlappung $\phi_{min}(p_T)$ gesetzt. Man kann erkennen, dass in diesem Fall alle Detektorinformationen von genau zwei Ausleseregionen genutzt werden, da sich die Regionen überall überlappen. Im rechten Bild ist $\Delta\phi$ größer als die Überlappung, wodurch nur die Randbereiche der Regionen von zwei Regionen genutzt werden. Es wurde darauf verzichtet, die Größe der Regionen und Überlappungen maßstabsgerecht darzustellen, da $\phi_{min}(10\ \text{GeV})$ im Bereich von nur etwa 1° liegt.

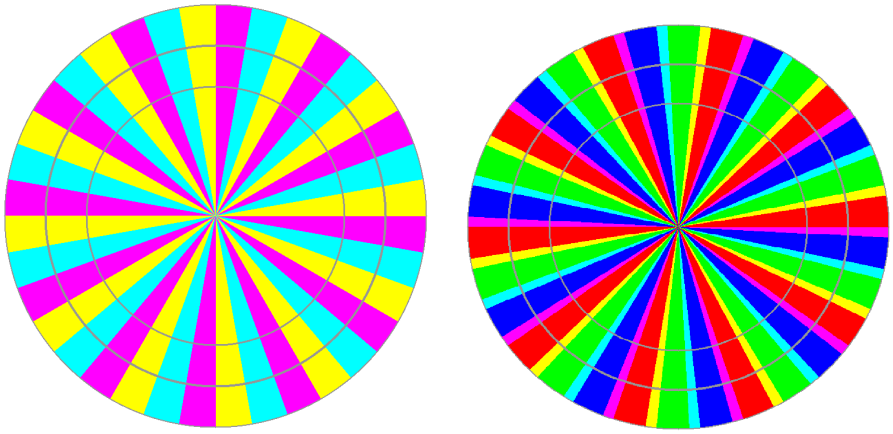

Abbildung 15: Darstellung der Ausleseregionen in ϕ mit Überlappung.

4.3 Die Einteilung in z-Richtung

In der r-z-Ebene ist es von Vorteil, dass hier die Spuren nicht gekrümmt sind, sondern im Idealfall - wenn es nicht zur Streuung an anderen Teilchen oder am Detektormaterial kommt - gerade verlaufen.

Hier gibt es dagegen ein anderes Problem, nämlich dass der Bereich, in dem die Kollisionen auf der z-Achse (=Strahlachse) stattfinden, je nach Struktur der Bunches eine Länge von mehreren Zentimetern hat. Dieser Bereich wird als luminoser Bereich bezeichnet, und in dieser Arbeit wird dafür eine Länge von 15 cm angenommen.

Da der Entstehungsort der Spuren in diesem Fall nicht als Punktquelle angesehen werden kann, kann hier das Prinzip der „Tortenstücke“ nicht angewendet werden.

Man muss also überlegen, welche Bedingungen man an die Einteilung der Regionen in der r-z-Ebene stellen soll. Eine sinnvolle Bedingung wäre, dass alle Spuren, die einen Bereich Δz auf der Lage i erreichen können, in einer Region zusammengefasst werden. Für diese Arbeit wird die äußerste Lage als Bezugslage genutzt.

Die Konstruktion erfolgt dann so, dass man vom linken Rand des luminosen Bereichs zum linken Rand des Bereichs Δz auf der äußersten Lage eine Gerade zeichnet und ebenso die rechten Ränder des luminosen Bereichs und des Bereichs Δz verbindet. Bei dieser Konstruktion gibt es keine Überlappung der Ausleseregionen auf der äußersten Lage. In den inneren Lagen ist diese jedoch nicht zu vermeiden.

Durch die Wahl von Δz wird auch direkt die Anzahl Regionen in z festgelegt. Sie ergibt sich als Quotient der Länge der äußersten Lage in z geteilt durch Δz.

In Abbildung 16 ist die Einteilung schematisch dargestellt, wobei hier die Ausdehnung der einzelnen Sensoren in r vergrößert dargestellt wurde. Zur Darstellung wurde auch

hier eine Region rot eingefärbt. Der gelbe Bereich unten im Bild (bei r=0) zeigt den luminosen Bereich auf der Strahlachse. Die drei Lagen darüber zeigen schematisch die Sensoren in in z-Richtung.

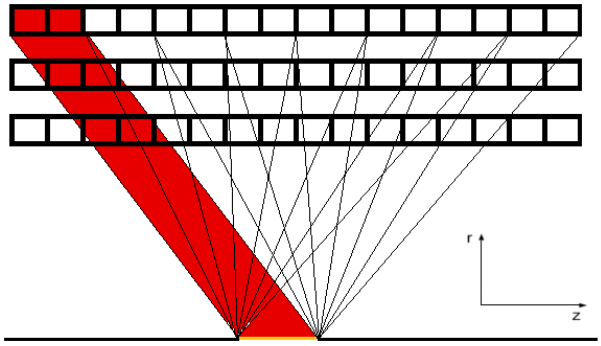

Abbildung 16: Definition der Ausleseregionen in der r-z-Ebene.

Ein Spezialfall dieser Konstruktion ist es, den Bereich Δz genau so groß zu wählen wie den luminosen Bereich. In diesem Fall hat die Region auf allen Detektorlagen die gleiche Ausdehnung in z. Allerdings ist man in diesem Fall nicht mehr in der Lage, die Anzahl Regionen durch Variation von Δz auf die gewünschte Zahl anzupassen.

Die Darstellung der Regionen in der r-z-Ebene und ihre Überlappungen in den inneren Lagen ist in Abbildung 17 farbig dargestellt. Man kann erkennen, dass die Überlappung der Regionen in z zunimmt, je mehr man sich der Strahlachse nähert.

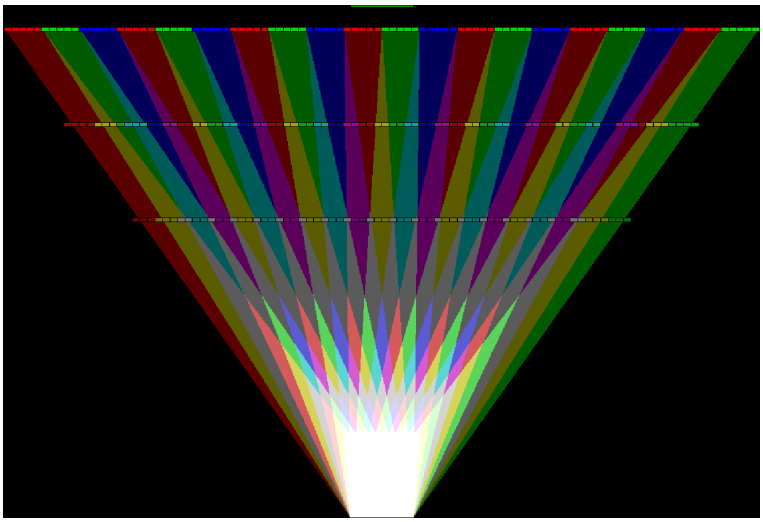

Abbildung 17: Darstellung der Ausleseregionen in der r-z-Ebene.

5 Simulation der Ausleseregionen

Die gewählte Definition der Ausleseregionen wird nun durch eine Computersimulation getestet und die Vorhersagen überprüft. Von besonderer Bedeutung ist hierbei natürlich, ob die erstellte Formel für $\phi_{min}(p_T)$, die die Größe der Überlappung definiert, sich durch die Simulation belegen lässt. Außerdem wird getestet, wie häufig Spuren nicht vollständig in zumindest einer Ausleseregionen registriert werden und wie häufig es vorkommt, dass eine Spur in mehr als einer Region vollständig registriert wird.

5.1 Überprüfung der Formel für $\phi_{min}(p_T)$

In Gleichung 4 wurde die Größe des Überlappungswinkels $\phi_{min}(p_T)$ in Abhängigkeit vom minimalen zu detektierenden transversalen Impuls p_T bestimmt. Diese Beziehung soll nun mit Hilfe einer Computersimulation überprüft werden.

Im ersten Test werden nur Spuren mit einem transversalen Impuls $p_T = 10$ GeV betrachtet und die Größe der Regionen in ϕ variiert. Die Gesamtgröße einer Region in ϕ wird mit ϕ_R bezeichnet.

In diesem Test wird vorausgesetzt, dass sich die Regionen mit der Winkelgröße ϕ_R um $\phi_R/2$ überlappen. Aufgrund der Vorüberlegung erwartet man, dass für $\phi_R < \phi_{min}(10\text{ GeV})$ keine Spuren in einer Region registriert werden.

Für $\phi_R \geq \phi_{min}(10\text{ GeV})$ können Spuren vollständig in einer Region enthalten sein. Sie werden als Treffer gezählt. Die nicht registrierten Spuren werden als verlorene Spuren bezeichnet.

Erst ab $\phi_R/2 = \phi_{min}(10\text{ GeV})$ kann erwartet werden, dass jede Spur vollständig in einer Ausleseregion registriert wird. Wählt man die Überlappung $\phi_R/2$ noch größer, so kommt es immer häufiger dazu, dass Spuren in mehr als einer Ausleseregion vollständig enthalten sind. Diese werden als Mehrfachtreffer gezählt. Das Ergebnis dieser Simulation ist in Abbildung 18 dargestellt.

Im zweiten Test werden Spuren mit unterschiedlichen Transversalimpulsen untersucht und es wird der minimale Überlappungswinkel $\phi_{min}(p_T)$ bestimmt, bei dem erstmals alle Spuren in mindestens einer Ausleseregion vollständig registriert werden. Die Ergebnisse der Simulation sind in Abbildung 19 rot eingetragen, wobei der Wert, der im ersten Test für $p_T = 10$ GeV ermittelt wurde, hier rot eingekreist ist. Die theoretische Kurve nach Gleichung 4 ist grün eingezeichnet.

Man kann erkennen, dass innerhalb der Fehler, die durch die Interpolation der diskreten Werte von ϕ entstehen, die Simulation mit den nach Gleichung 4 erwarteten Werten übereinstimmt.

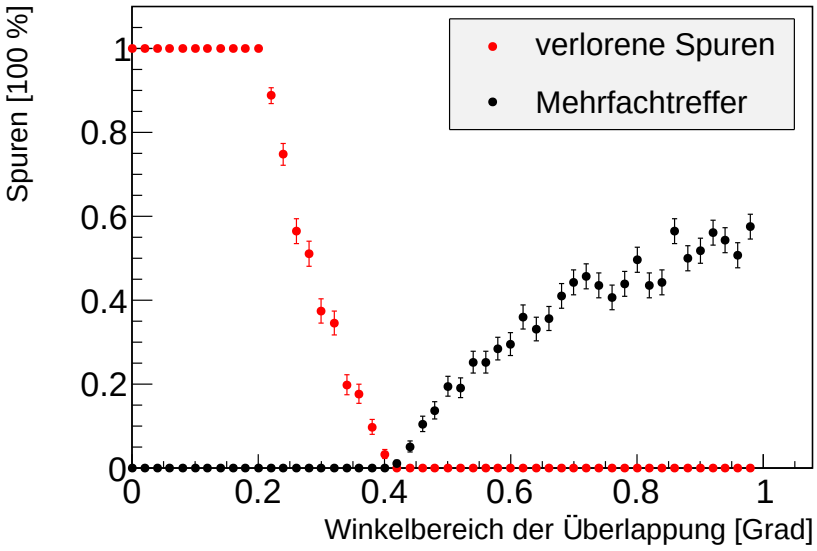

Abbildung 18: Zuordnung der Spuren in Abhängigkeit von Regionsgröße ϕ_R und Überlappungswinkel $\phi_R/2$.

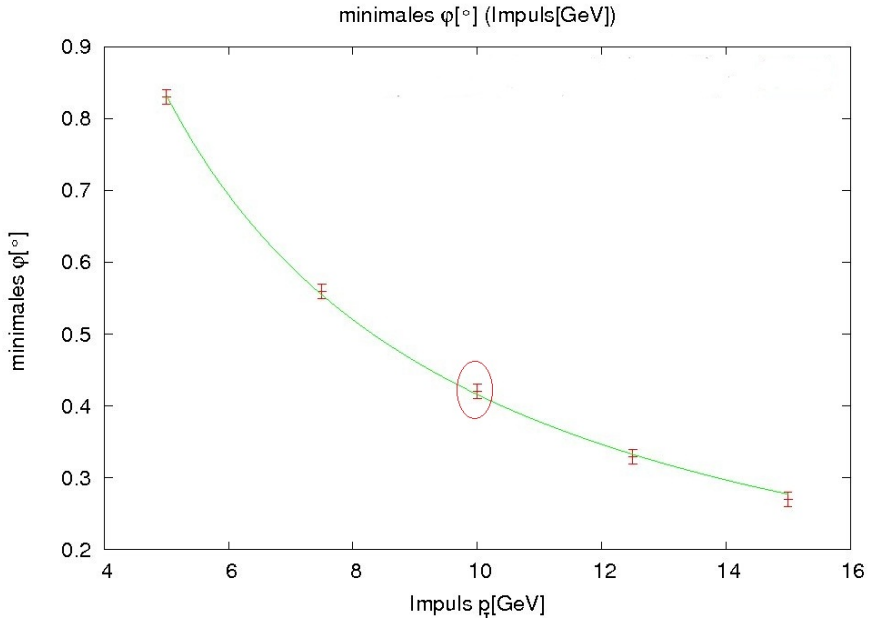

Abbildung 19: Plot von ϕ_{min} in Abhängigkeit von p_T.

Da in dieser Arbeit ein Transversalimpuls von 10 GeV als untere Triggerschwelle genutzt wird, wird im Folgenden ϕ_{min} für $\phi_{min}(10\,\text{GeV}) = 0.42°$ verwendet, wodurch ϕ_{min} konstant wird.

Für die Regionsgröße ϕ_R gilt gemäß Kapitel 4.2:

$$\phi_R = \Delta\phi + \phi_{min} \tag{5}$$

Da die Größe ϕ_{min} bei gewähltem p_T konstant ist, kann die Größe der Region in ϕ - und damit die Anzahl Regionen in ϕ - nur noch über die Größe $\Delta\phi$ verändert werden.

5.2 Variation der Größe der Regionen auf der Strahlachse in z-Richtung

Wie bereits erwähnt, wird in dieser Arbeit eine Länge von 15 cm für den luminosen Bereich angenommen. In Kapitel 4.3 wurden die Regionen so konstruiert, dass ihre Länge auf der Strahlachse - bezeichnet mit D - mit der Länge des luminosen Bereichs - bezeichnet mit l - übereinstimmt. In einem Test wird nun geprüft, welchen Einfluss eine Veränderung von D auf das Wiederfinden einer Spur hat. Man kann erwarten, dass bei D kleiner als 15 cm einige Spuren nicht mehr vollständig in einer Region registriert werden, da sie dann über den Rand einer Region hinaus laufen können. Bei D = 15 cm können nur durch Streuung einige wenige Teilchen so abgelenkt werden, dass ihre Spur nicht vollständig in einer Ausleseregion verläuft. Bei D größer 15 cm sollten alle Spuren vollständig in einer Region enthalten sein. Das Ergebnis der Simulation ist in Abbildung 20 dargestellt. 1 steht hierbei für 100%.

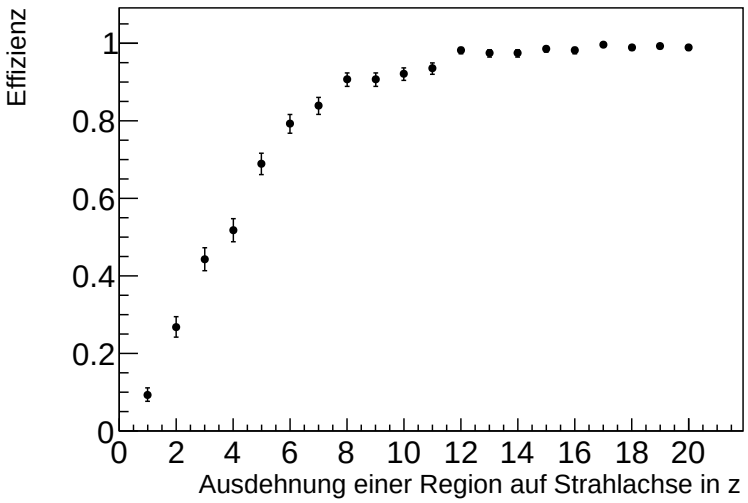

Abbildung 20: Zugeordneten Spuren in Abhängigkeit von D.

Der Test zeigt, dass es ausreichend ist, die Länge einer Region auf der Strahlachse mit der Länge des luminosen Bereichs gleichzusetzen. Daher werden die Ausleseregionen im Folgenden stets so konstruiert.

5.3 Design der zum Auslesen genutzten Sensoren und Chips

Bei der Einteilung der betrachteten Detektorfläche in Ausleseregionen werden die Parameter $\Delta\phi$ und Δz so gewählt, das sie auf der äußersten Lage mit den Bereichen der Auslesesensoren bzw. deren Unterteilungen übereinstimmen.

Die in dieser Simulation genutzten Sensoren sind nach dem Vorbild des ABCD-Chips definiert [10], doch es wurde für die Simulation an einigen Stellen von diesem Vorbild abgewichen. Es werden auf allen betrachteten Detektorlagen baugleiche Sensoren zum Auslesen betrachtet. Sie haben einen quadratischen Aufbau mit einer Kantenlänge von 10 cm. Diese haben in ϕ-Richtung 1280 Silizium-Streifen (Strips). In z-Richtung haben sie einen aktiven Bereich von 9.6 cm. Der restliche Platz in z-Richtung wird für die Auslese-Hardware benötigt. In dieser Simulation wird der inaktive Bereich jedoch nicht betrachtet. Es wird davon ausgegangen, dass die Chips mit ihren aktiven Bereichen aneinander anschließen.

In z-Richtung wird der Sensor in 4 Auslesebereiche aufgeteilt, die jeweils einen Bereich von 2.4 cm in abdecken. Jeder Auslesebereich wird logisch noch einmal in 10 Bereiche in ϕ eingeteilt. Man erhält somit auf einem Auslesesensor 40 Unterteilungen, die in dieser Arbeit als Auslesechips bezeichnet werden. Ein Chip hat somit eine Ausdehnung von 2.4 cm in z-Richtung und 1 cm in ϕ-Richtung. In ϕ-Richtung umfasst ein Chip 128 Silizium-Streifen.

Der schematische Aufbau der Sensoren mit ihrer logischen Unterteilung ist in Abbildung 21 gezeigt.

Die Länge der Detektorlagen in z beträgt auf allen Lagen 240 cm, wodurch in der Simulation ohne inaktive Bereiche 100 Chips pro Lage in z-Richtung angenommen werden. In ϕ-Richtung decken die Chips auf den einzelnen Lagen je nach radialer Entfernung zur Strahlachse unterschiedliche Winkelbereiche ab. Die von einem Chip auf den verschiedenen Lagen abgedeckten Winkelbereiche sind in Tabelle 1 aufgeführt.

Radius [cm]	Anzahl Chips pro Lage in ϕ	Winkelbereich [°]
38.0	240	1.500
50.1	320	1.125
62.2	400	0.900

Tabelle 1: Winkelbereiche der Chips in Abhängigkeit der Lage.

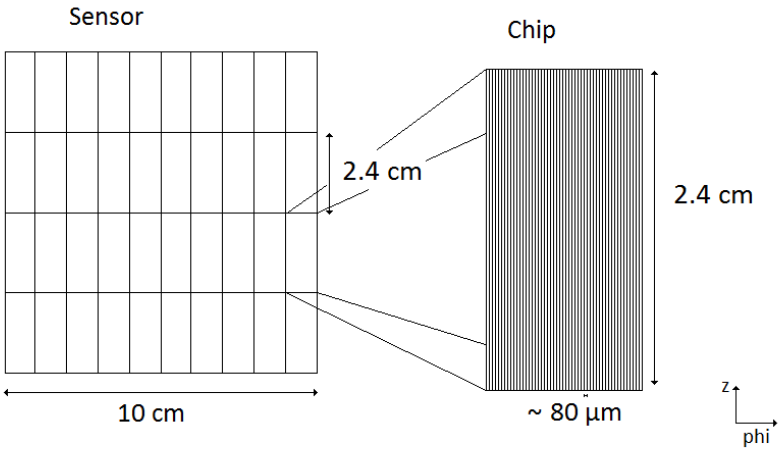

Abbildung 21: Design der Auslesesensoren.

Da der Winkelbereich eines Chips auf der äußersten Lage 0.9° beträgt, sind auf dieser Lage 400 Chips in ϕ-Richtung vorhanden, insgesamt also 40000 Chips auf der äußersten Lage. Wenn die Parameter $\Delta\phi$ und Δz gerade auf die Ausdehnung eines Chips auf der äußersten Lage gesetzt werden, entstehen dadurch 40000 voneinander unabhängige Ausleseregionen. Ein Nachteil von sehr kleinen Regionen ist, dass der Anteil der Überlappungsbereiche sehr hoch ist, deren Informationen in mehreren Regionen verarbeitet werden müssen.

Da auf der anderen Seite mindestens 10000 unabhängige Ausleseregionen definiert werden sollen, um eine schnelle Parallelverarbeitung zu ermöglichen, ergibt sich eine begrenzte Anzahl Möglichkeiten für die Parameter $\Delta\phi$ und Δz, die in Tabelle 2 aufgeführt ist. Die Wahl dieser Parameter wird im nächsten Kapitel noch einmal genauer betrachtet.

$\Delta\phi$	Δz	Anzahl Ausleseregionen
0.9	2.4	40000
1.8	2.4	20000
3.6	2.4	10000
0.9	4.8	20000
1.8	4.8	10000
0.9	9.6	10000

Tabelle 2: Untersuchte Kombinationen der Parameter $\Delta\phi$ und Δz.

6 Wahl der Parameter $\Delta\phi$ und Δz

Im Folgenden soll untersucht werden, welchen Einfluß die Wahl der Parameter $\Delta\phi$ und Δz auf die Anzahl Chips bzw. Strips hat, deren Informationen von einer Region verarbeitet werden müssen. Diese Anzahl hat bei der Implementierung in Hardware eine große Bedeutung, da hier die Informationen aller Strips in Templates gespeichert werden müssen.

Für die Bestimmung dieser Anzahl ist neben der Definition der Region auch von Bedeutung, wie man die Zuordnung zwischen den Sensoren, den Chips und den Strips zu den Regionen wählt. Je nachdem, ob die Zuordnung auf der Ebene der Sensoren, der Chips oder der Strips gemacht wird, ist die Anzahl Strips, die letztlich einer Region zugeordnet werden, unterschiedlich.

Je mehr Regionen die Informationen eines Auslesechips benötigen, desto mehr Verdrahtung wird zwischen den Auslesechips und der Trigger-Hardware benötigt. Eine Alternative zu dieser Verdrahtung ist eine Wireless Auslese der Regionen, an der zum Zeitpunkt dieser Arbeit noch geforscht wird. Der Aspekt der Verdrahtung soll unter anderem in diesem Abschnitt mit untersucht werden.

6.1 Größe der Regionen in z und ϕ

Die Größe einer Region in z und ϕ hängt von der Größe des luminosen Bereichs sowie der Anzahl gewünschter Regionen in z ab, ferner vom gewünschten minimalen Transversalimpuls und der Anzahl Regionen in ϕ .

6.1.1 Größe der Regionen in z

Die Größe der Regionen in z kann für die inneren Lagen in Abhängigkeit vom Parameter Δz, der die Größe auf der äußersten Lage beschreibt, bestimmt werden. Die Formel hierfür ist:

$$\delta z(r) = l + \frac{r \cdot (\Delta z - l)}{r_{max}} \tag{6}$$

Dabei ist l die Länge des luminosen Bereichs, r der Radius der betrachteten Lage und $\Delta z = \delta z(r_{max})$. Zur Veranschaulichung dieser Gleichung dient Abbildung 22.

Bei dem betrachteten 3-Lagen-Layout ist r_{max} der Radius der dritten Lage. In Tabelle 3 ist die Breite der Regionen in z auf den einzelnen Lagen für die in Kapitel 5 für Δz zusammengestellten Werte dargestellt.

Da man in z jedoch immer nur ganze Chips auslesen kann, betrachten man nun die Anzahl Chips, im Folgenden mit $N_{chips,z}$ abgekürzt, die man minimal bzw. maximal pro Region in z-Richtung auslesen muss. Die Werte hierfür sind für die einzelnen Lagen

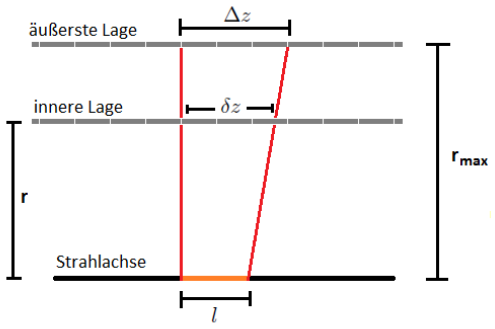

Abbildung 22: Bestimmung der Regionsgröße in z für die inneren Lagen.

	Δz= 2.4 cm	Δz= 4.8 cm	Δz= 9.6 cm
δz_1	7.3 cm	8.8 cm	11.7 cm
δz_2	4.9 cm	6.8 cm	10.7 cm
δz_3	2.4 cm	4.8 cm	9.6 cm

Tabelle 3: Breite einer Region in z auf den 3 Lagen.

in Tabelle 4 dargestellt, wobei die erste Zahl der minimalen und die zweite Zahl der maximalen Anzahl notwendiger Chips entspricht.

	$N_{chips,z,Lage}$		
	Δz= 2.4 cm	Δz= 4.8 cm	Δz= 9.6 cm
Lage 1	4 / 5	4 / 5	5 / 6
Lage 2	3 / 4	3 / 4	5 / 6
Lage 3	1 / 1	2 / 2	4 / 4

Tabelle 4: Anzahl Chips pro Region in z-Richtung für die einzelnen Lagen.

Da die Regionen auf der äußersten Lage genau an die Chips angepasst werden, stimmt da die Anzahl minimal nötiger Chips pro Region in z mit der maximalen Zahl überein. In den inneren Lagen kann es jedoch sein, dass die Regionen in der Mitte eines Chips beginnen und somit einen Chip mehr enthalten als durch die Größe notwendig wäre.

Da für einen einheitlichen Aufbau von Pattern und Templates in den verschiedenen Regionen die Anzahl Chips in allen Regionen gleich sein soll, muss man für jede Region die maximale Anzahl notwendiger Chips wählen. Man erhält in diesem Fall einen Chip mehr in den Lagen 1 und 2.

6.1.2 Größe der Regionen in ϕ

Die Größe einer Region in ϕ ist allein durch den Winkelbereich definiert, den sie abdecken soll. Zu beachten ist, dass wir in ϕ eine Überlappung brauchen, deren Größe durch den minimalen transversalen Impuls gegeben ist, der auf jeden Fall detektiert werden soll. Auch in ϕ werden die Regionen den Auslesechips auf der äußersten betrachteten Detektorlage angepasst. Da die Chips in allen Lagen gleich groß sind, decken sie auf den verschiedenen Lagen unterschiedliche Winkelbereiche ab. Die Winkelbereiche der Chips auf den betrachteten Lagen sind:

	Winkelbereich eines Chips $\phi_{chip,Lage}$ [Grad]	Winkelbereich eines Strips $\phi_{strip,Lage}$ [10^{-3} Grad]	Anzahl Chips pro Lage in ϕ
Lage 1	1.5	11.7	240
Lage 2	1.125	8.8	320
Lage 3	0.9	7.0	400

Tabelle 5: Winkelbereiche der Chips und Strips sowie Anzahl der Chips auf den betrachteten Lagen.

Die Überlappung ϕ_{min} für einen transversalen Impuls von 10 GeV ist gegeben durch 0.42°.

Wenn den Regionen stets ganze Chips zugeordnet werden, werden einer Region deutlich mehr Strips in ϕ zugeordnet als tatsächlich benötigt werden. Das führt dazu, dass bei der Analyse der Region Informationen verarbeitet werden müssen, die für die Trigger-Entscheidung in dieser Region keine Relevanz haben.

Eine Reduktion der Datenmenge kann erfolgen, indem man neben der Zuordnung der Chips zu Regionen auch eine Zuordnung der Strips zu den Regionen durchführt. Die Anzahl Chips bzw. Strips, die in den beiden Fällen ausgelesen werden müssen, sind durch die folgenden Gleichungen beschrieben:

$$N_{chips,\phi} = \lceil \frac{\phi_R}{\phi_{chip,1}} \rceil + \lceil \frac{\phi_R}{\phi_{chip,2}} \rceil + \lceil \frac{\phi_R}{\phi_{chip,3}} \rceil \tag{7}$$

$$N_{strips,\phi} = N_{chips,\phi} \cdot 128 \tag{8}$$

$$N'_{strips,\phi} = \lceil \frac{\phi_R}{\phi_{strip,1}} \rceil + \lceil \frac{\phi_R}{\phi_{strip,2}} \rceil + \lceil \frac{\phi_R}{\phi_{strip,3}} \rceil \tag{9}$$

mit $\phi_R = \Delta\phi + \phi_{min}$ und $\lceil x \rceil$ die Aufrundefunktion [1].

Hierbei ist $N_{strips,\phi}$ die Anzahl der notwendigen Strips pro Region, wenn keine Zuordnung Strips zu Regionen erfolgt, und $N'_{strips,\phi}$ die Anzahl notwendiger Strips, die im

[1] Beispiel: $\lceil 2.1 \rceil = 3$

Falle einer Zuordnung Strips zu Region benötigt werden. Die Anzahl der Chips pro Region ist in beiden Fällen gleich und wird durch die Größe $N_{chips,\phi}$ beschrieben.

Um auch in ϕ-Richtung einheitliche Templates zu erhalten, muss man in den Lagen 1 und 2 jeweils einen weiteren Chip bzw. Strip hinzufügen. In der äußersten Lage werden die Regionen wieder an die Chips angepasst, wodurch hier keine weiteren Chips/Strips notwendig sind.

Die Ergebnisse für die Anzahl zugeordneter Strips pro Region für die beiden Zuordnungen sind in den Tabellen 6 und 7 dargestellt. Auch hier ist der erste Wert der für die minimale Anzahl Strips und der zweite Wert steht für die maximale Anzahl Strips pro Region.

	$N_{strips,\phi}$		
	$\Delta\phi = 0.9°$	$\Delta\phi = 1.8°$	$\Delta\phi = 3.6°$
Lage 1	128 / 256	256 / 384	384 / 512
Lage 2	256 / 384	256 / 384	512 / 640
Lage 3	256 / 256	384 / 384	640 / 640

Tabelle 6: Template-Größen $N_{strips,\phi}$ bei Einteilung auf Chip-Ebene.

	$N'_{strips,\phi}$		
	$\Delta\phi = 0.9°$	$\Delta\phi = 1.8°$	$\Delta\phi = 3.6°$
Lage 1	113 / 114	190 / 191	344 / 345
Lage 2	151 / 152	253 / 254	458 / 459
Lage 3	188 / 188	316 / 316	572 / 572

Tabelle 7: Template-Größen $N'_{strips,\phi}$ bei Einteilung auf Strip-Ebene.

Da es bei der Implementierung in Hardware sein kann, dass die Bitlänge, die in einen CAM eingelesen werden kann, kleiner ist als die Anzahl Strips, die der Region zugeordnet werden, ist es in solchen Fällen notwendig auf einer ersten Stufe mehrere Strips zusammenzufassen, damit diese in einem CAM verarbeitet werden können. Um eine solche Vergröberung der Muster zu ermöglichen, muss die Anzahl Strips pro Region durch die Anzahl Strips, die zusammengefasst werden sollen, teilbar sein. Dazu muss man die Größe $N'_{strips,\phi}$ noch so anpassen, dass sie durch den gewünschten Faktor teilbar ist. Man erhält somit z.B. für eine Vergröberung um den Faktor c:

$$N''_{strips,\phi} = (\lceil N'_{strips,\phi}/c \rceil) \cdot c \tag{10}$$

Die Anzahl Strips pro Region in ϕ für den Vergröberungsfaktor $c = 32$ ist in Tabelle 8 dargestellt. Da hier der notwendigen Anzahl Strips bereits mehr als ein Strip zusätzlich

zugeordnet wurde, gibt es für die hier betrachteten Größen $N''_{strips,\phi}$ keine Unterscheidung mehr zwischen der minimalen und der maximalen Anzahl Strips pro Region in ϕ.

	$N''_{strips,\phi}$		
	$\Delta\phi = 0.9°$	$\Delta\phi = 1.8°$	$\Delta\phi = 3.6°$
Lage 1	128 / 128	192 / 192	352 / 352
Lage 2	160 / 160	256 / 256	480 / 480
Lage 3	192 / 192	320 / 320	576 / 576

Tabelle 8: Template-Größen $N''_{strips,\phi}$ mit Anpassung für Vergröberung um Faktor c = 32.

6.1.3 Zusammenstellung der Templates für z und ϕ

Die Gesamtgröße der Templates ist abhängig davon, welche Zuordnung man zwischen Chips bzw. Strips und den Regionen wählt und welche Werte man für die Parameter $\Delta\phi$ und Δz wählt. Die Anzahl Chips in z für die einzelnen Lagen kann der Tabelle 4 entnommen werden. Diese müssen mit der Anzahl Strips in ϕ (Tabellen 6 - 8) multipliziert werden, um die Gesamtanzahl Strips pro Lage zu erhalten.

Die Anzahl Detektorzellen, die einer Region zugeordnet werden, ergibt sich dann als Summe der Zellen über alle Lagen. Sie ist in Abhängigkeit dieser drei Parameter in Tabelle 9 aufgeführt. Da hier einheitliche Templates betrachtet werden, beruhen die Zahlen auf der maximalen Anzahl Zellen pro Region.

$\Delta\phi$ [Grad]	Verfahren	Δz= 2.4 cm	Δz= 4.8 cm	Δz= 9.6 cm
0.9	$N_{strips,\phi}$	3072	3328	4864
1.8	$N_{strips,\phi}$	3840	4224	6144
3.6	$N_{strips,\phi}$	5760	6400	9472
0.9	$N'_{strips,\phi}$	1366	1554	2348
1.8	$N'_{strips,\phi}$	2287	2603	3934
3.6	$N'_{strips,\phi}$	4133	4705	7112
0.9	$N''_{strips,\phi}$	1472	1664	2496
1.8	$N''_{strips,\phi}$	2304	2624	3968
3.6	$N''_{strips,\phi}$	4256	4832	7296

Tabelle 9: Anzahl Detektorzellen (Strips) pro Region.

Auch wenn die Anzahl genutzter Strips bei den verschiedenen Verfahren der Zuordnung in ϕ unterschiedlich ist, so wird bei allen Verfahren die Information aus der gleichen

Anzahl Chips benötigt. Für die Berechnung der Verdrahtungen der Chips ist es am einfachsten, die Werte $N_{strips,\phi}$ zu benutzen, die auf der Zuordnung ganzer Auslesechips zu den Regionen beruhen. Diese Anzahl Chips pro Region erhält man direkt, indem man die Werte für dieses Verfahren durch die Anzahl Strips pro Chip - also 128 - teilt. Multipliziert man diese Zahl mit der Anzahl Regionen, die aus der Wahl von Δz und $\Delta\phi$ resultiert, so erhält man die Anzahl Verdrahtungen, die für das Auslesen der Chips benötigt werden.

6.2 Duplizierung der Information von einzelnen Chips

Aufgrund der Konstruktion der Auslese-Regionen kommt es zu Überlappungen der Regionen, die es notwendig machen, die Information eines Chip bzw. seiner Strips an verschiedene Auslese-Regionen weiterzuleiten.

Die Anzahl Regionen, an die die Information eines Chips minimal bzw. maximal gesendet werden muss, ist bei der gewählten Konstruktion der Regionen abhängig von den Parametern $\Delta\phi$ und Δz.

Die Anzahl $\overline{I}$ der Regionen in z bzw. in ϕ, an die die Information eines Chips durchschnittlich gesendet werden muss, ist gegeben durch die Anzahl Zuordnungen der Chips zu den Regionen geteilt durch die Anzahl genutzter Chips:

$$\overline{I}_{z,Lage\,i} = \frac{N_{chips,z,i} \cdot N_{Regionen,z}}{c_{z,i}} \tag{11}$$

$$\overline{I}_{\phi,Lage\,i} = \frac{N_{chips,\phi,i} \cdot N_{Regionen,\phi}}{c_{\phi,i}} \tag{12}$$

Dabei ist $c_{z,i}$ die Anzahl Chips in z auf Lage i, deren Informationen von mindestens einer Region genutzt werden. $c_{\phi,i}$ ist die Anzahl aller Chips in ϕ auf Lage i, da in ϕ immer alle Chips mindestens einer Region zugeordnet sind. Chips, deren Information von keiner Region genutzt werden, werden somit nicht in die Durchschnittsbildung einbezogen.

Zur Bestimmung der Größe $c_{z,i}$ wird zunächst mit Gleichung (6) der Bereich $b_{z,i}$ in z auf Lage i bestimmt, der von Regionen genutzt wird:

$$b_{z,i} = l + \frac{r_i \cdot (b_{z,max} - l)}{r_{max}}$$

mit den Bezeichnungen aus Gleichung (6) und der Länge der Äußersten Lage in z $b_{z,max}$ (hier: 240 cm).

Mit der Bezeichnung z_{chip} für die Länge eines Chips in z (hier: 2.4 cm) ist dann:

$$c_{z,i} = 2 \cdot \lceil \frac{b_{z,i}}{2 \cdot z_{chip}} \rceil$$

Die Ergebnisse der Gleichungen (11) und (12) für die bisher betrachteten Werte von Δz und $\Delta\phi$ sind in den Tabellen 10 und 11 aufgeführt. Die vorderen Zahlen beruhen dabei auf der minimalen Anzahl Chips, die für eine Region benötigt werden, die hinteren Zahlen beruhen auf der maximalen Anzahl Chips.

Um die Anzahl Regionen insgesamt zu ermitteln, die die Information eines Chips benötigen, muss man nun die Ergebnisse für z und ϕ multiplizieren. Dazu werden hier die maximalen Werte genutzt (Gleichung (13)). Die maximale Anzahl Vervielfältigungen

	Δz= 2.4 cm	Δz= 4.8 cm	Δz= 9.6 cm
Lage 1	6.25 / 7.81	3.13 / 3.91	1.95 / 2.34
Lage 2	3.66 / 4.88	1.83 / 2.44	1.52 / 1.83
Lage 3	1 /1	1 / 1	1 / 1

Tabelle 10: Anzahl Regionen in z, an die die Information eines durchschnittlich Chips gesendet werden muss $\overline{I}_{z,Lage\,i}$.

	$\Delta\phi = 0.9°$	$\Delta\phi = 1.8°$	$\Delta\phi = 3.6°$
Lage 1	1.67 / 3.33	1.67 / 2.5	1.25 / 1.67
Lage 2	2.5 / 3.75	1.25 / 1.88	1.25 / 1.56
Lage 3	2 / 2	1.5 / 1.5	1.25 / 1.25

Tabelle 11: Anzahl Regionen in ϕ, an die die Information eines durchschnittlich Chips gesendet werden muss $\overline{I}_{\phi,Lage\,i}$.

I_{max} erhält man, indem man die Durchschnittswerte in ϕ und in z einzeln aufrundet (Gleichung (14)).

$$\overline{I}_{Lage\,i} = \overline{I}_{z,Lage\,i} \cdot \overline{I}_{\phi,Lage\,i} \tag{13}$$

$$I_{max,Lage\,i} = \lceil \overline{I}_{z,Lage\,i} \rceil \cdot \lceil \overline{I}_{\phi,Lage\,i} \rceil \tag{14}$$

Die Ergebnisse der Gleichungen (13) und (14) sind in Tabelle 12 zusammengestellt. Die vorderen Werte entsprechen hierbei $\overline{I}_{Lage\,i}$, die hinteren Werte entsprechen $I_{max,Lage\,i}$.

$\Delta\phi$ [Grad]	Lage	Δz= 2.4 cm	Δz= 4.8 cm	Δz= 9.6 cm
0.9	1	26.0 / 32	13.0 / 16	7.8 / 12
1.8	1	19.5 / 24	9.8 / 12	5.9 / 9
3.6	1	13.0 / 16	6.5 / 8	3.9 / 6
0.9	2	18.3 / 20	9.2 / 12	6.9 / 8
1.8	2	9.2 / 10	4.6 / 6	3.4 / 4
3.6	2	7.6 / 10	3.8 / 6	2.9 / 4
0.9	3	2 / 2	2 / 2	2 / 2
1.8	3	1.5 / 2	1.5 / 2	1.5 / 2
3.6	3	1.25 / 2	1.25 / 2	1.25 / 2

Tabelle 12: Anzahl Regionen, die die Information eines Chips im Durchschnitt / maximal nutzen.

Wenn man Tabelle 12 mit Tabelle 2 aus Abschnitt 5.3 gemeinsam betrachtet, stellt man fest, dass eine Einteilung des Detektors in eine höhere Anzahl Ausleseregionen auch zu einer Erhöhung der Duplizierung der Informationen beim Trigger führt. Zusammen mit

den Informationen aus Tabelle 9, die die Abhängigkeit der Anzahl Strips - und somit der Anzahl Einträge in einem Template - von den Parametern $\Delta\phi$ und Δz enthält, können nun diese Parameter festgelegt werden.

Bei der Zuordnung der Chips zu den Regionen wird in dieser Simulation davon ausgegangen, dass nur ganze Chips einer Region zugeordnet werden können ($N_{strips,\phi}$). Da man etwa 10000 unabhängige Ausleseregionen erhalten will, bietet sich die Kombination $\Delta\phi = 1.8$ Grad und $\Delta z = 4.8$ cm zur Regionseinteilung an. Diese Werte werden daher für die weitere Simulation genutzt.

6.3 Anzahl zu speichernder Templates

In Tabelle 13 wird für verschiedene Layouts des Triggers die Anzahl notwendiger Templates für die gesamte Detektorfläche aufgeführt. Man kann erkennen, dass das in diesem Abschnitt betrachtete Layout von den in der Tabelle betrachteten Layouts mit Abstand die wenigsten Templates benötigt. Die Zahlen in den Klammern der Spalte Layout geben an, auf wie vielen Lagen von der Gesamtzahl der betrachteten Lagen die Einträge des Templates mit den Treffern im Pattern übereinstimmen müssen. (5/6) bedeutet z.B., dass 5 von den 6 Einträgen des Templates im Pattern vorkommen müssen. Auf diese Weise berücksichtigt man, dass in der Realität die Detektorlagen nicht zu 100 % effizient sind und somit unter Umständen ein Treffer auf einer Lage fehlt.

Neben der Anzahl Templates ist für die verschiedenen Layouts auch die Anzahl erwarteter falscher Spuren im gesamten Detektor angegeben. Falsche Spuren sind Kombinationen von Treffern im Pattern, die vom Trigger als hochenergetische Spur erkannt werden, obwohl die Treffer von verschiedenen niederenergetischen Spuren stammen. Diese Anzahl ist für das einfache 3-Lagen-Layout sehr groß. Auf die falschen Spuren wird in den Kapiteln 9 und 10 noch genauer eingegangen.

Layout	Anzahl Templates [Milliarden]	falsche Spuren (Fakes) bei 400 Pile-up
3 shorts (3/3)	3	6000
1 pixel $*$ 3 shorts (4/4)	20	2000
2 pixel $*$ 3 shorts (5/5)	80	1000
5 shorts (5/5)	230	10
5 shorts letzte Lage bei 86 cm (5/5)	100	10
5 shorts letzte Lage bei 86 cm (4/5)	80	1000
3 short Doppellagen (5/6)	230	50
3 short Doppellagen (6 cm Abstand) (5/6)	100	10

Tabelle 13: Anzahl Templates und falsche Spuren für verschiedene Layouts [7]

Bei dem 3-Lagen-Layout müssen die Treffer auf allen Lagen vorhanden sein. Da in dieser Simulation mit dem 3 shorts (3/3) Layout gearbeitet wird, gilt für die Gesamtanzahl Templates:

$$N_{T,0} = 3 \cdot 10^9 \tag{15}$$

Eine Vergröberung der Templates um den Faktor c führt zu einer Veränderung der Anzahl Templates. Die Anzahl Templates mit der vergröberten Struktur ist [7]:

$$N_{T,c} = N_{T,0} \cdot \frac{1}{c^2} \tag{16}$$

Die Anzahl notwendiger Templates, die man für eine Region speichern muss, ist somit gegeben durch:

$$N_{T,c,\#Region} = N_{T,c} \cdot \frac{1}{\#\text{Regionen}} \tag{17}$$

Die Größenordnung der notwendigen Templates ist somit:

$N_{T,0}$	$3 \cdot 10^9$
$N_{T,c=32}$	$3 \cdot 10^6$
$N_{T,c=0,\#Region=10000}$	$3 \cdot 10^5$
$N_{T,c=32,\#Region=10000}$	300

Tabelle 14: Anzahl Templates bei Vergröberung.

Man kann erkennen, dass durch die Vergröberung die Anzahl Templates, die gespeichert werden müssen, deutlich reduziert werden kann. Somit kann mit der Vergröberung der Bitmuster nicht nur das mögliche Problem der Eingangswortlänge für den CAM behoben werden, sondern gleichzeitig auch das Problem, das dadurch entsteht, dass ein CAM nur eine begrenzte Anzahl Templates speichern kann.

7 Erstellung der Templates und Speicherung in einer Datenbank

In diesem Abschnitt wird auf zwei Möglichkeiten der Darstellung der Templates in Hardware und Software eingegangen, und es wird anschließend mit einer der beiden Darstellungen der Aufbau der Datenbank erklärt.

Für die Darstellung der Templates gibt es zwei Möglichkeiten. Die eine ist, dass jedes Bit im Template einen Strip der Ausleseregion repräsentiert. Diese Form der Darstellung heißt „unencoded“. Die zweite Möglichkeit besteht darin, dass jedem Strip der Ausleseregion eine Nummer zugewiesen wird und nur die Nummern der getroffenen Strips weitergegeben werden. Diese Form der Darstellung heißt „encoded“. Die beiden Formen der Darstellung sind in den Abbildungen 23 und 24 dargestellt.

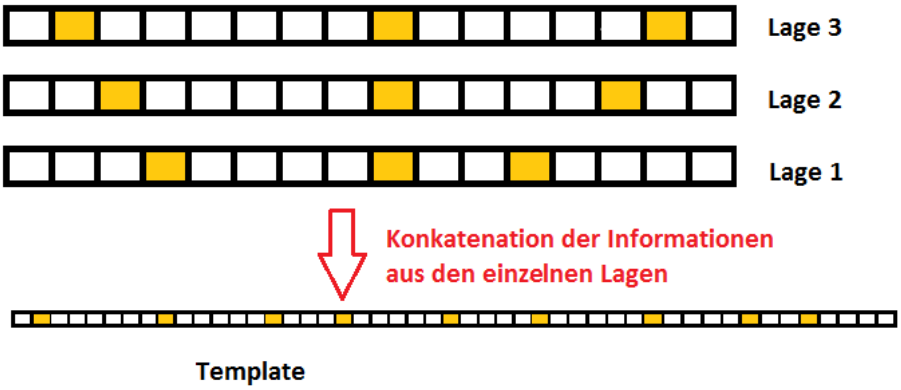

Abbildung 23: Template in der Darstellung als Bitmuster (unencoded).

Abbildung 24: Template bestehend aus Zell-ID's (encoded).

7.1 Darstellung der Templates in der Simulation

Der für den Spur-Trigger in Hardware zur Verfügung stehende tertiäre CAM hat die Möglichkeit, für jede Stelle im Template eine von drei Informationen zu speichern:

- 1: Treffer an dieser Stelle gefordert
- X: don't care (Diese Stelle wird beim Vergleich nicht beachtet)
- 0: kein Treffer an dieser Stelle erlaubt

In einem Template werden die Treffer zu einer Spur mit Einsen gespeichert. Die restlichen Stellen können auf X gesetzt werden, damit das Template immer gültig wird, wenn die Treffer zu der Spur im Pattern vorhanden sind.

Das Einfügen einer 0 im Template kann als Veto-Entscheidung genutzt werden, um ein Template ungültig zu machen, wenn in der Nähe der gesuchten Treffer zusätzliche Treffer sind. Auf diese Weise kann verhindert werden, dass für eine Spur mehrere ähnliche Templates gültig werden und somit Spuren mehrfach registriert werden.

In dieser Simulation werden keine Vetos (0) verwendet. Aus diesem Grund enthält ein Template in der Simulation nur die Informationen 1 oder X. Die Möglichkeiten des tertiären CAMs wurden daher in dieser Simulation nicht ausgenutzt.

Ein CAM arbeitet mit der unencoded Darstellung. Für die Simulation wurde aus Speicherplatzgründen die encoded Form zur Speicherung der Templates in der Datenbank gewählt. Da ein Template nur die Informationen einer Spur enthält, hat jedes Template pro Lage nur an einer Stelle einen Eintrag. Diese Stelle wird über die Zellnummer in z und die Zellnummer in ϕ kodiert. Es ist natürlich möglich, beide Nummern zu einer eindeutigen ID zu verbinden. Die Nummer der Lage wird über die Position der ID im Template kodiert.

7.2 Generierung der Templates

Um die Datenbank mit Templates aufzubauen, gibt es zwei Ansätze:

Der eine ist, alle möglichen Trefferkombinationen zu bilden und mit Hilfe eines Fits zu entscheiden, ob die Trefferkombination zu einem hochenergetischen Teilchen gehört. Der zweite Ansatz ist, Teilchen so lange mit dem gewünschten Transversalimpuls zu generieren und deren Muster in der Datenbank zu speichern, bis keine oder kaum noch neue Muster entstehen. Die Werte aus Tabelle 13 beruhen auf dem Fit-Ansatz.

Der zweite Ansatz wird in dieser Simulation genutzt. Die Teilchen werden so generiert, dass sie nur in der Ausleseregion landen, für die die Templates erstellt werden. Der luminose Bereich wird wie in Kapitel 3 definiert. Für den Impuls wird eine flache Verteilung in der Krümmung der Spuren gewählt. Ihr Minimum ist 0, was einem unendlich hohen Impuls des Teilchens entspricht, und ihr Maximum wird so bestimmt, das es der Spurkrümmung eines Teilchens von 10 GeV entspricht, was die in dieser Simulation

gewählte Triggerschwelle ist. Der Zeitpunkt, bei dem man das Generieren abbricht, hat bei diesem Ansatz Einfluß auf die Effizienz (das Erkennen hochenergetischer Spuren) des Triggers und auf die Datenbankgröße.

Um die Abhängigkeit der Anzahl gespeicherter Templates $N_{gesp.\,Templates}$ von der Anzahl generierter Templates $N_{gen.\,Muster}$ näherungsweise beschreiben zu können, wird zunächst ein vereinfachter Fall betrachtet. Hierfür werden zwei Annahmen getroffen:

1. Die Anzahl unterschiedlicher Templates ist endlich.
2. Alle Templates sind gleich wahrscheinlich.

In diesem Fall gilt für die Anzahl gespeicherter Templates:

$$N_{gesp.\,Templates} = m \cdot (1 - (1 - \frac{1}{m})^{N_{gen.\,Muster}}) \approx m \cdot (1 - e^{-\frac{N_{gen.\,Muster}}{m}}) \qquad (18)$$

Die Variable m steht hierbei für die maximale Anzahl Templates in einer Region. Sie wurde für die Simulation unter Benutzung von Tabelle 13 und einer Abschätzung für die Überlappung auf etwa 380000 geschätzt. Bei der Bestimmung dieses Werts wurde benutzt, dass die Gesamtzahl der Templates nach Tabelle 13 für dieses Layout $3 \cdot 10^9$ beträgt und diese Zahl durch die Anzahl Regionen geteilt werden muss, die in dieser Simulation 10000 beträgt. Hinzu kommt der Anteil, der durch die Überlappungen der Regionen entsteht. Der theoretische Verlauf nach Gleichung (18) ist für diesen Fall in Abbildung 25 zu sehen.

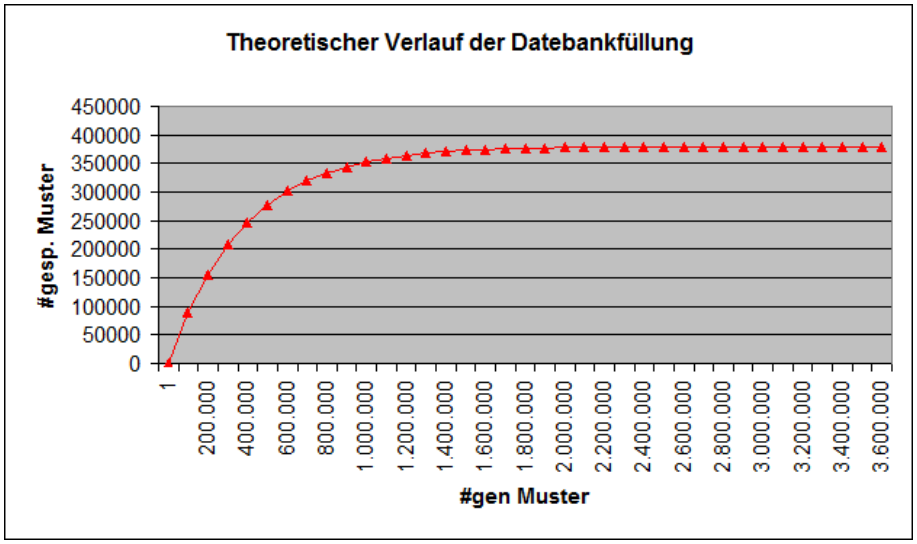

Abbildung 25: Anzahl gespeicherter Templates.

In der Simulation kommen jedoch zwei Effekte hinzu, die den Verlauf der Kurve aus Abbildung 25 leicht verändern.

Zum einen sind nicht alle Templates gleich wahrscheinlich (siehe Abbildung 27), wodurch der Anstieg der Kurve am Anfang etwas langsamer verläuft, da häufiger gleiche Muster generiert werden. Zum anderen ist die Anzahl Templates zwar endlich (aufgrund der endlichen Anzahl Detektorzellen offensichtlich), es können aber durch Streuung der Teilchen am Detektor viele sehr seltene Muster erzeugt werden, durch die die theoretische Maximalzahl an Templates überschritten wird. Aufgrund dieses Effekts hat die Kurve der Simulation nicht eine Asymptote wie die Kurve der Theorie, sondern sie steigt über die Kurve des theoretischen Verlaufs und sogar über den Wert 380000 weiter an. Der Verlauf der Simulation ist in Abbildung 26 dargestellt.

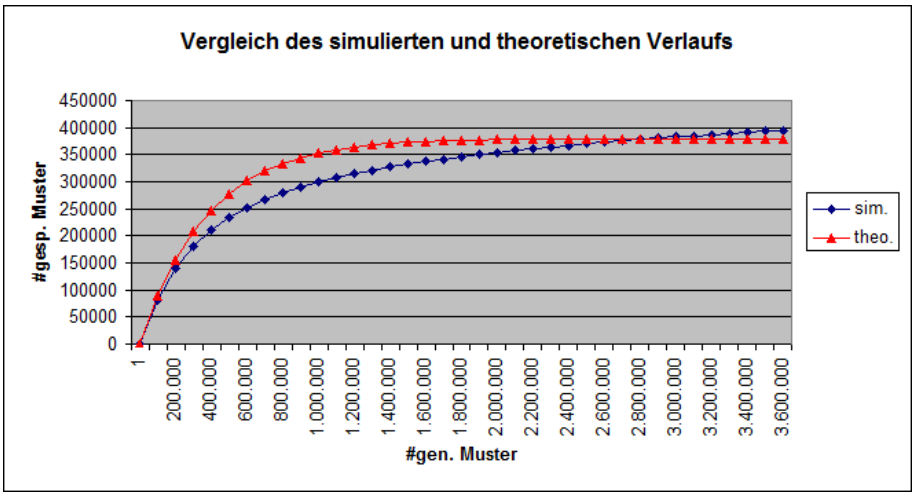

Abbildung 26: Vergleich von Theorie und Simulation.

Man kann erkennen, dass der simulierte Verlauf sich näherungsweise durch Gleichung 18 beschreiben lässt, obwohl die Voraussetzungen nicht vollständig erfüllt sind. Da einige der Templates nur sehr selten von einer gültigen Spur getroffen werden, ist es sinnvoll die Gewichtung der Templates mit zu speichern, um unter Umständen sehr seltene Templates aus der Datenbank zu entfernen. Dieser Schritt spart zum einen Zeit- und Speicher-Ressourcen, reduziert aber auch die Anzahl falscher Spuren, was in Kapitel 9 weiter diskutiert wird.

Um die Gewichtung der Templates betrachten zu können, wird zu jedem Template in der Datenbank noch in einer Variablen gespeichert, wie häufig das Template beim Aufbau der Datenbank erzeugt wurde. Auf diese Weise kann man die Gewichtung der Templates später direkt aus der Datenbank ablesen. Die Variable, die mitzählt, wie häufig das Template generiert wurde, wird Counter genannt.

Die Gewichtung der Templates ist in Abbildung 27 dargestellt. Es wird deutlich, dass es eine sehr große Anzahl Templates gibt, die einen sehr niedrigen Counter haben und vermutlich durch Streuung erzeugt wurden. Diese sehr seltenen Templates müssten

ohne große Verluste in der Effizienz aus der Datenbank entfernt werden können, was in Kapitel 9 untersucht wird. Das Format der Datenbank ist in Abbildung 28 gezeigt.

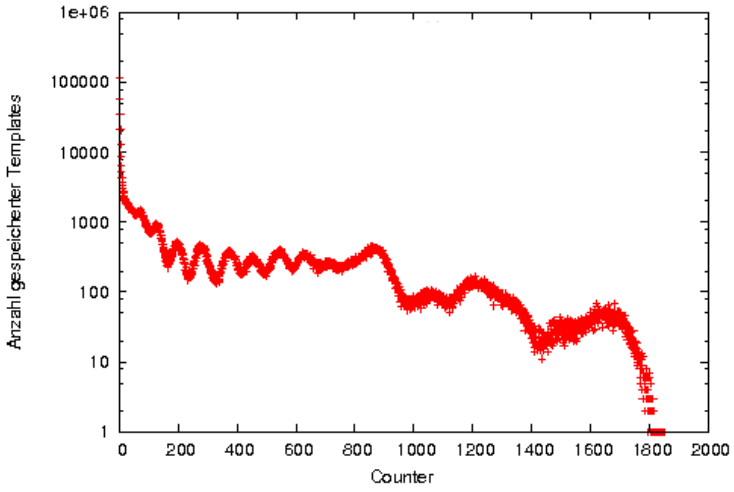

Abbildung 27: Gewichte der Templates in der Datenbank.

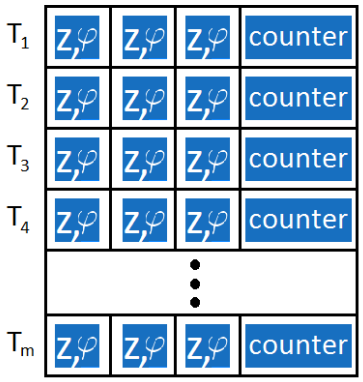

Abbildung 28: Format der Datenbank.

8 Implementierung des Vergleichs zwischen Pattern und Templates

In diesem Kapitel werden verschiedene Wege vorgestellt, wie der Vergleich zwischen Pattern und Templates umgesetzt werden kann. Dabei muss zwischen der Umsetzung in Hardware und Software unterschieden werden, da der Weg, der in Hardware genutzt wird, sich nicht ohne weiteres in Software übertragen lässt. Entscheidende Faktoren in diesem Teil der Simulation sind einerseits das Zeitverhalten der einzelnen Methoden, andererseits die Speicheranforderungen, die diese Methoden stellen. Die in dieser Arbeit gewählte Methode bildet daher einen Kompromiss zwischen dem Zeitverhalten und dem benötigen Speicher, der durch den verfügbaren Speicher limitiert ist.

8.1 Der Vergleich in Hardware

Die Grundidee des Vergleichs ist, nicht suchen zu müssen, ob ein vorliegendes Pattern mit einem Template übereinstimmt, sondern das Pattern als Adresse zu interpretieren und dann nur noch an der Stelle, auf die die Adresse zeigt, prüfen zu müssen, ob es sich um ein gültiges Muster handelt. Dazu müsste dann zu jeder Adresse eine Ja/Nein Information (bool) gespeichert werden, die im Speicher ein Bit benötigt.

Die Länge einer Adresse hängt ab von der Anzahl Detektorzellen, die zu einer Ausleseregion gehören. Diese wurden bereits für die verschiedenen Parameter bestimmt. Bei einer Adresslänge m erhält man maximal 2^m mögliche Adressen.

In Kapitel 6 wurde für verschiedene Regionseinteilungen die Anzahl Strips bestimmt, deren Informationen von einer Region genutzt werden (siehe Tabelle 9). Da diese Werte deutlich über den Werten liegen, die die in Abschnitt 2.4 vorgestellten CAMs verarbeiten können, braucht man noch ein Verfahren, um die Anzahl Bits so zu reduzieren, dass ein CAM sie verarbeiten kann, ohne jedoch notwendigen Informationen zu verlieren.

Ein Weg hierfür ist, das Pattern zunächst um einen Faktor c zu vergröbern (siehe auch Abschnitt 6.1.2 und 6.3), d.h. immer c benachbarte Bits durch eine Oder-Verknüpfung zusammenzufassen. Enthält also mindestens eine der c benachbarten Zellen einen Treffer, so wird als Ergebnis eine 1 geliefert, sonst wird eine 0 zurückgegeben. Die Größe des Pattern wird so um den Faktor c reduziert.

Die Templates im CAM werden ebenfalls in dieser groben Form gespeichert und enthalten eine Adresse zu den feinen Informationen, die dann in einem RAM-Speicher hinterlegt sind. In dem RAM-Speicher müssen dann nur noch die Stellen mit den Treffern wieder verfeinert werden und mit dem Pattern verglichen werden.

Das Grundprinzip ist in Abbilddung 29 gezeigt.

Prinzipiell ist es auch möglich mehrere Verfeinerungsstufen in den CAMs durchzuführen und so einen kaskadenförmigen Aufbau mit den CAMs zu erzeugen.

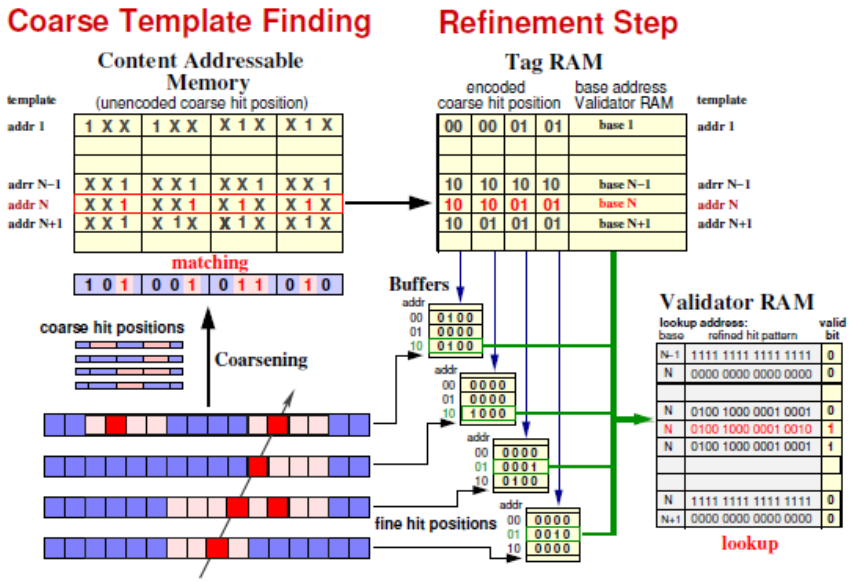

Abbildung 29: Prinzip des Vergleichs mit Vergröberung. [7]

In der Simulation nahm die Anzahl Templates bei einer Vergröberung um den Faktor 2 nicht wie nach Gleichung 16 in Abschnitt 6.3 erwartet um den Faktor 4 ab, sondern nur um etwa 3.5. Diese Abweichung macht sich natürlich vor allem bei großen Vergröberungen bemerkbar, wie zum Beispiel bei dem Faktor 32, der nach Gleichung 16 eine Reduktion um den Faktor 1024 ermöglichen sollte, jedoch in der Simulation nur eine Reduktion der Anzahl Templates um etwa den Faktor 525 bewirkt.

8.2 Die Mustersuche in der Simulation

In der Simulation wird statt der CAMs eine Datenbank [11] benutzt. Außerdem wird ohne eine Vergröberung der Muster gearbeitet. Das Interpretieren des Pattern als Adresse ist bei den hier benötigten Adresslängen in der Software-Simulation nicht möglich. Daher wird hier statt dessen die Suche in einer Datenbank genutzt, obwohl das die Simulation natürlich deutlich langsamer macht. Bis auf das Zeitverhalten sind die Ergebnisse jedoch mit dem Verfahren in Hardware ohne Vergröberung vergleichbar.

Für die Implementierung der Mustersuche in der Simulation wurden zwei Verfahren gewählt, die je nach Eingabemenge und Fragestellung ihre Vorteile haben. Die Verfahren werden in den nächsten Abschnitten diskutiert werden.

8.2.1 Sequentielle Verarbeitung der Datenbank

Das nahe liegendste Verfahren, ein Pattern mit den Templates zu vergleichen, ist, einmal die Datenbank sequenziell zu lesen und für jedes Template zu fragen, ob alle markierten Zellen des Templates auch im Pattern getroffen wurden. Das Verfahren ist in Abbildung 30 dargestellt. In dieser Simulation wurden dabei alle Templates der

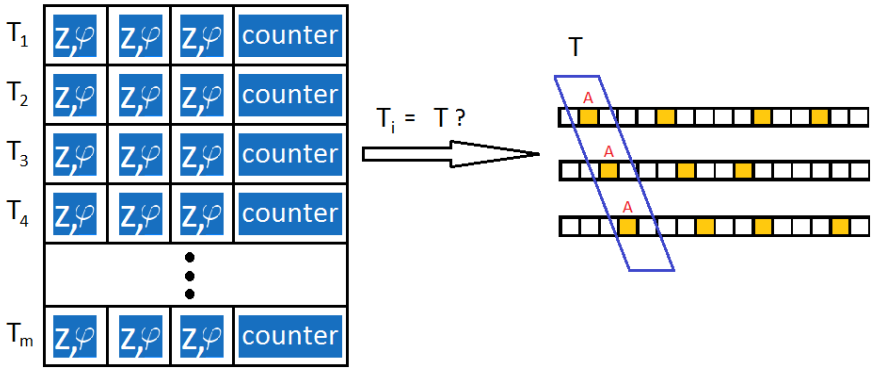

Abbildung 30: Vergleich mit Durchlaufen der Datenbank.

Datenbank sequentiell gelesen und für jedes Template ein Vergleich mit allen Treffern im Pattern durchgeführt. Die Laufzeit hängt in diesem Fall sowohl von der Anzahl Templates m in der Datenbank ab als auch von der Anzahl Treffer k pro Lage im Pattern. Die Anzahl Treffer ist hierbei in jeder Lage näherungsweise gleich. Insgesamt gilt in jeder Region $m \gg k$. Die Laufzeit ist:

$$L = O(m \cdot k) \tag{19}$$

Es ist möglich, durch eine andere Form der Speicherung der Einträge des Pattern - z.B. in einem Array - das Durchlaufen der Treffer zu vermeiden. In diesem Fall geht der Faktor k nicht mehr in die Laufzeit ein. Bei den in der Simulation festgelegten Bedingungen ist die Anzahl Treffer pro Lage in einer Region im Bereich von 5 - 8.

8.2.2 Direktzugriff mit Schlüssel auf die Datenbank

Das zweite Verfahren ist, zunächst alle Kombinationen von jeweils einem Treffer pro Lage im Pattern zu bilden und dann diese Kombinationen in der Datenbank direkt zu suchen. Die Anzahl möglicher Kombinationen ist hierbei im Fall von 3 Lagen $O(k^3)$, bei n Lagen $O(k^n)$. Die Suche nach den Mustern in der Datenbank kann mit O(log m) erfolgen. Die Gesamtlaufzeit bei diesem Verfahren ist somit:

$$L = O(log(m) \cdot k^n) \tag{20}$$

Das Verfahren ist in Abbildung 31 dargestellt.

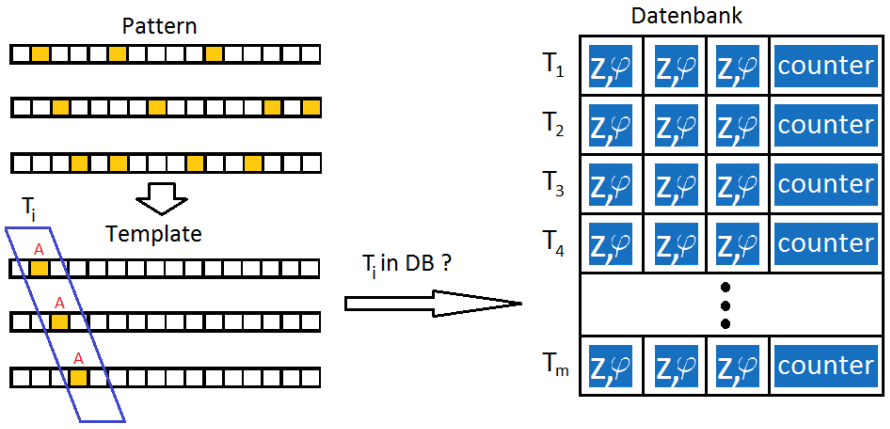

Abbildung 31: Zugriff mit Kombinationen aus dem Pattern.

Für wenige Lagen und wenige Treffer pro Region kann sich dieses Verfahren gegen das zuvor beschriebene Verfahren durchsetzen. Vor allem bei Tests, bei denen immer nur eine Spur erzeugt und in der Datenbank gesucht wird, ist dieses Verfahren deutlich schneller als das erste Verfahren. Das Zeitverhalten in Abhängigkeit von der Anzahl Spuren pro Ereignis ist in Abbildung 32 gezeigt.

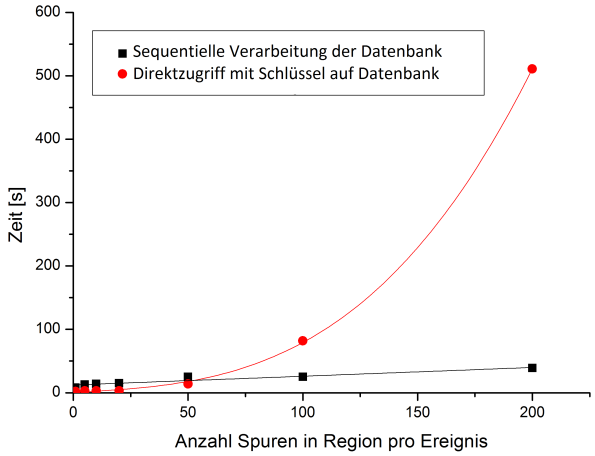

Abbildung 32: Zeitverhalten der beiden Verfahren bei 3 Lagen in Abhängigkeit von der Anzahl Spuren.

9 Effizienz und Fake-Rate

9.1 Effizienz

Als Effizienz E bezeichnet man, wie viele hochenergetische Spuren von dem Spur-Trigger als solche erkannt werden.

$$E = \frac{\text{Anzahl gefundener hochenergetischer Spuren}}{\text{Anzahl vorhandener hochenergetischer Spuren}} \tag{21}$$

Die Effizienz steigt natürlich mit der Anzahl gespeicherter Templates in der Datenbank und somit auch mit der Anzahl generierter Muster für die Datenbank. Es ist zu erwarten, dass die Effizienz nahe an die 100 % gehen wird, wenn man genug Muster generiert.

In Abbildung 33 ist die Effizienz des Spur-Triggers für unterschiedliche Datenbankgrößen dargestellt. Dabei enthält aufgrund der Art ihrer Generierung eine Datenbank mit mehr Mustern alle Muster der kleineren Datenbanken. Es zeigt sich, dass die Effizienz bei allen betrachteten Datenbanken über 99 % liegt und mit steigender Datenbankgröße wie erwartet zunimmt.

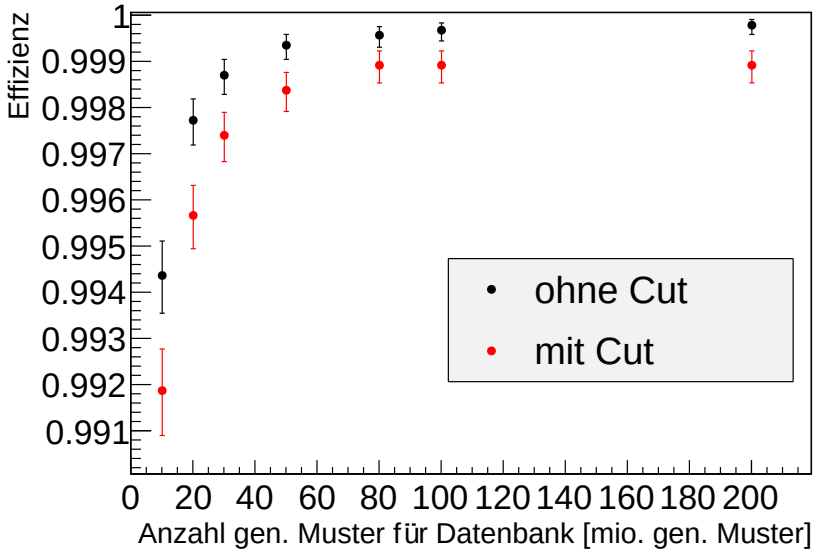

Abbildung 33: Effizienz mit und ohne Selektion der Templates.

Für die Bestimmung der Effizienz mit Cut wurden - wie in Abschnitt 9.2 beschrieben - alle Templates aus der Datenbank entfernt, die einen Counter kleiner als 2.5 % des durchschnittlichen Counters haben.

Um zu zeigen, dass die Effizienz nicht nur in einem bestimmten Impulsintervall sehr hoch ist, wurde die Effizienz für die Datenbank mit 200 Millionen generierten Mustern noch einmal in Abhängigkeit vom Transversalimpuls bestimmt. Das Ergebnis ist in Tabelle 15 aufgeführt. Der Endwert von 600 GeV wurde mit Hilfe einer Abschätzung der Spurkrümmung der Teilchen und der Strip-Breite in ϕ bestimmt. Ab diesem Wert sollten sich die Treffer-Kombinationen, die die Teilchen auf den betrachteten Lagen hinterlassen, nicht mehr wesentlich ändern.

Transversalimpuls [GeV]	1-5	5-8	8-10	10-15	15-20	20-300	300-600
Effizienz [100%] ohne Cut	0	0,005	0,2	0,9995	0,9994	0,9996	0,9998
Effizienz [100%] mit Cut	0	0	0,11	0,9979	0,9967	0,9982	0,9989

Tabelle 15: Effizienz in Abhängigkeit vom Transversalimpuls.

Es wird deutlich, dass ab dem für das Generieren genutzten minimalen Impuls von 10 GeV die Effizienz in den betrachteten Impulsintervallen immer in dem erwarteten hohen Bereich liegt, es also keine Bereiche über 10 GeV gibt, in denen die Teilchen besonders schlecht detektiert werden können.

9.2 Fake-Rate und Selektion der Templates

Als Fakes oder falsche Spuren bezeichnet man Kombinationen von Treffern verschiedener Spuren, die zu einem Muster einer hochenergetischen Spur passen und somit fälschlich eine positive Trigger-Entscheidung bewirken. Die Anzahl Fakes F ist abhängig von der Okkupanz auf den einzelnen Detektorlagen und von der Anzahl Templates T in der Datenbank (siehe Gleichung (22)). Dabei ist die Okkupanz occ_i auf der i-ten Detektorlage gegeben durch den Quotienten der Anzahl Treffer auf der i-ten Lage geteilt durch die Anzahl Detektorzellen (Strips) der i-ten Lage.

$$F = T \cdot \prod_{i=1}^{Anzahl\ Lagen} occ_i \tag{22}$$

Die Fake-Rate ist, wie zuvor die Effizienz, in Abhängigkeit von der Größe der Datenbank in Abbildung 34 dargestellt. Die Werte nach Gleichung (34) werden dabei als theoretische Werte bezeichnet.

Da die Fake-Rate stark von der Anzahl Templates in der Datenbank abhängt, kann sie gesenkt werden, indem man alle Templates mit einem sehr niedrigen Gewicht (Counter) aus der Datenbank zu entfernt. Da die Counter der seltenen Templates auch mit der Größe der Datenbank wachsen, wurde für die Selektion ein Wert gewählt, der abhängig von dem durchschnittlichen Gewicht ist. Für die Simulation wurden alle Templates aussortiert, deren Counter unterhalb von 2.5 % vom durchschnittlichen Counter

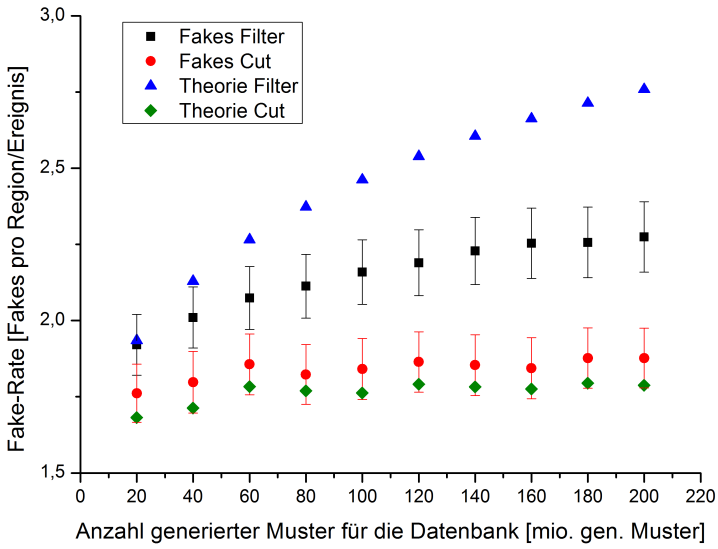

Abbildung 34: Fake-Rate mit und ohne Selektion der Templates.

ist. Der Einfluß dieser Auswahl (Cut) ist ebenfalls in Abbildung 34 dargestellt. Man kann erkennen, dass der Einfluß der Selektion auf die Fake-Rate deutlich größer ist als auf die Effizienz, deren Änderung in Abbildung 33 dargestellt wurde.

Die Fehler bei den simulierten Werten sind die statistischen Fehler des Mittelwerts. Da der Fehler der simulierten Werte nur aus dem Fehler der Okkupanz kommen kann und dieser sehr klein ist, sind diese Fehler vernachlässigbar klein. Man kann erkennen, dass bei den Kurven, die als Filter nur verwenden, dass alle Templates in der Datenbank zu der Ausleseregion gehören, die simulierten Werte von den theoretischen Werten stark abweichen. Das hat folgenden Grund: Die Templates in der Datenbank sind nicht willkürlich, sondern beschreiben alle eine ganz bestimmte Art von Mustern. Da sich ab einem bestimmten Punkt die neuen Templates nur noch minimal von den bisher gespeicherten unterscheiden, steigt danach die Fake-Rate nur noch wenig an. Da Gleichung (22) nur auf Kombinatorik beruht und keine Strukturen in den Templates und bei den Treffermustern im Detektor berücksichtigt, kann die theoretische Vorhersage nach Gleichung (22) diesen Effekt nicht erfassen.

Bei den Templates mit dem Cut beschreibt die theoretische Vorhersage im Rahmen der statistischen Fehler die simulierten Werte sehr gut. Es ist jedoch zu erkennen, dass man eine systematische Verschiebung zwischen der theoretischen Vorhersage und den simulierten Werten hat. Die simulierten Fake-Raten liegen hier immer etwas über den Werten der Theorie.

Insgesamt zeigt sich jedoch, dass sich die Fake-Rate für dieses Layout trotz geringer

Abweichungen relativ gut durch Gleichung (22) beschreiben lässt.

Die Okkupanz, die die für die Bestimmung der Fake-Rate generierten Ereignisse in den einzelnen Regionen haben, ist in Abbildung 35 links dargestellt. Hier wird deutlich, dass die Okkupanz stark von z abhängt. Die Nummer einer Region setzt sich nach Gleichung (23) aus der Nummer in ϕ und der Nummer in z zusammen.

$$Regionsnummer = (N_{Regionen,\phi} \cdot Regionsnummer\,in\,z) + Regionsnummer\,in\,\phi \tag{23}$$

Die in dieser Simulation verwendete Region hat die Nummer 5210, was dem z-Bereich von 4.8 cm - 9.6 cm und dem ϕ-Bereich von 17.58° - 19.8° entspricht.

Die Okkupanz für einen festen z-Bereich sollte aufgrund der flachen Verteilung der Teilchen in ϕ beim Generieren unabhängig von ϕ sein. Dies wurde für den hier betrachteten z-Bereich noch einmal überprüft und das Ergebnis wurde in Abbildung 35 rechts dargestellt. Man kann erkennen, dass die Werte für die Regionen in ϕ um einen festen Wert streuen. Die globale Okkupanz auf der innersten Lage liegt bei etwa 2.05 %.

Da die Regionen aus geometrischen Gründen nur einen Teil der Chips auf der innersten Lage benutzen und die äußersten Chips der Lage von keiner Region genutzt werden, ist der Mittelwert lokalen Okkupanz der einzelnen Regionen auf der innersten Lage bei etwa 2.36 %.

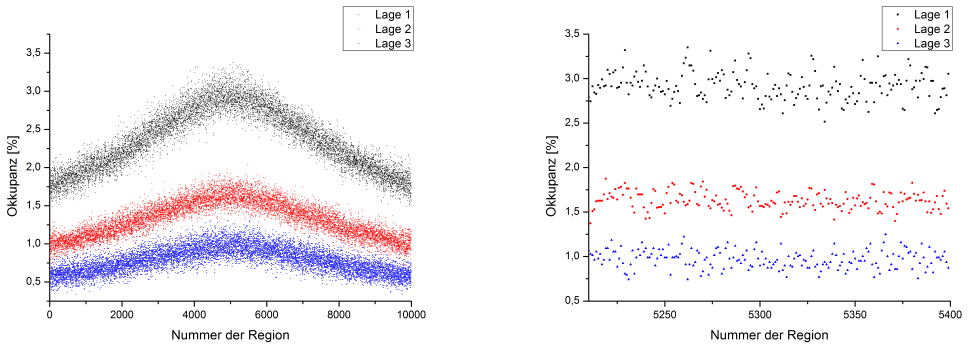

Abbildung 35: Okkupanz für die verschiedenen Regionen (links: alle Regionen rechts: Regionen mit festem z-Bereich).

10 Das Doppellagen-Layout

Zum Abschluß dieser Arbeit werden noch Teile der Auswertung für das Einzellagen-Layout (3-Lagen-Layout) für das Doppellagen-Layout durchgeführt.

Man betrachtet in diesem Fall die Doppellagen als zwei getrennte Lagen und erhält somit 6 Lagen für die Analyse. Da bei diesem Layout jeweils zwei Lagen nahe beieinander liegen, ist die Okkupanz dieser beiden Lagen ähnlich.

Aufgrund von Formel 22 aus Kapitel 9 folgt somit direkt, dass aufgrund der Multiplikation der Okkupanzen, die alle in einem Bereich bis etwa 2 % liegen, die Fake-Rate deutlich kleiner werden sollte. Die Effizienz hängt wie in Kapitel 9 beschrieben hauptsächlich davon ab, wie viele Templates man in der Datenbank speichert bzw. wie lange man nach möglichen gültigen Mustern sucht. Es ist also zu erwarten, dass die Effizienz bei diesem Layout ebenfalls Werte über 99 % annehmen kann.

Im Unterschied zu dem vorher betrachteten 3-Lagen-Layout werden hier zwei Möglichkeiten zur Auswertung betrachtet. Zum einen kann man wie bei dem Einzellagen-Layout verlangen, dass die Einträge eines Templates auf allen Lagen im Pattern vorkommen müssen. Wenn man hier jedoch bereits berücksichtigen will, dass die Detektorlagen in der Realität nicht zu 100 % effizient sind, würde man mit einer solchen Forderung bereits viele Spuren verlieren. Um zumindest die Spuren erkennen zu können, die auf 5 Lagen einen Treffer hinterlassen haben, muss man die Forderung so stellen, dass auch beim Vergleich mit den Templates nur 5 Treffer verlangt werden, oder man muss sechs Datenbanken aufbauen, die jeweils die Muster für 5 Lagen enthalten, und diese durchsuchen. Im zweiten Fall würde jedoch eine Spur, die auf allen Lagen einen Treffer hinterlassen hat, von sechs Datenbanken als gültig erkannt und somit mehrfach gezählt werden. Auch ähnliche Muster würden bei der Bedingung, dass nur die Einträge von 5 der 6 Lagen des Templates mit den Treffern des Pattern übereinstimmen müssen, viel häufiger getroffen, wodurch Spuren mehrfach gezählt würden (siehe Abbildung 36).

Die Treffer des Pattern sind in Abbildung 36 blau dargestellt. In den Templates sind die Einträge, die mit den Treffern im Pattern übereinstimmen grün markiert. Auf den Lagen im Template, auf denen der Vergleich fehlschlägt, ist der Eintrag des Templates schwarz markiert und der Treffer vom Pattern rot dargestellt. Neben Template 2, das auf allen Lagen mit den Einträgen des Pattern übereinstimmt, werden auch die anderen für gültig befunden, da nur 5 von 6 Treffern des Pattern mit dem Template übereinstimmen müssen. Eine mögliche Bedingung an den Encoder ist, dass die Templates mit möglichst vielen Übereinstimmungen gegenüber den anderen bevorzugt werden (siehe Abschnitt 2.4).

Da in der Simulation immer ein Treffer auf allen Lagen vorhanden ist bedeutet das, dass bei der Betrachtung der Effizienz untersucht werden muss, wie häufig man mit der 6/6-Bedingung bereits ein Treffer in der Datenbank findet. Eine Untersuchung mit der 5/6-Bedingung würde hier kein sinnvolles Ergebnis liefern, da in diesem Fall die

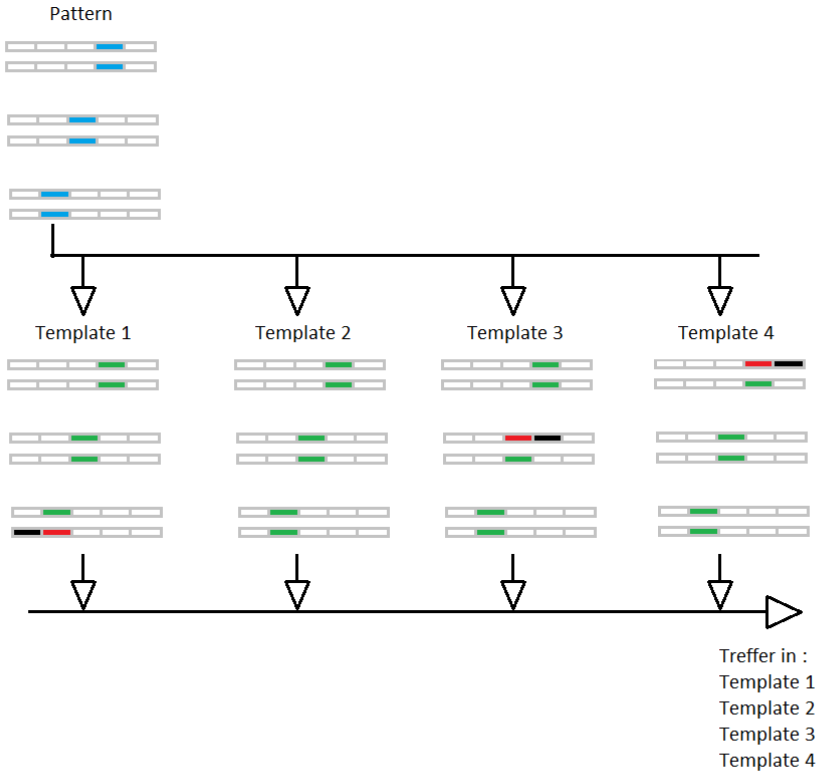

Abbildung 36: Lookup beim Doppellagenlayout mit Bedingung 5 aus 6.

Spuren mehrere Treffer auslösen und das eigentliche Ziel dieser Bedingung, nämlich Ineffizienzen des Detektormaterials auszugleichen, nicht untersucht werden kann.

Für die Fake-Rate werden zunächst alle Spuren betrachtet, die mit der 5/6-Bedingung zugewiesen werden, da hier die Anzahl falscher Spuren deutlich höher ist als bei der 6/6-Bedingung.

10.1 Effizienz

Wie in Kapitel 9 wird auch hier die Effizienz des Spur-Triggers in Abhängigkeit der Anzahl generierter Muster für die Datenbankerstellung bestimmt. Im Rahmen dieser Arbeit konnten für die Datenbank nur bis zu 300 Millionen Muster erzeugt werden, so dass Effizienz etwas niedriger ausfällt als beim 3-Lagen-Layout. Das liegt daran, dass beim 6-Lagen-Layout im Vergleich deutlich mehr Templates benötigt werden als beim 3-Lagen-Layout. Es wurde aber auch hier eine Effizienz über 99 % erreicht. Bei den Datenbanken wurde darauf geachtet, dass keine Templates in ihnen enthalten sind, die

nicht zu der betrachteten Ausleseregion gehören. Ein Cut wie in Kapitel 9 beschrieben wurde bei diesem Layout nicht verwendet. Die Effizienz für das Doppellagen-Layout ist in Abbildung 37 dargestellt.

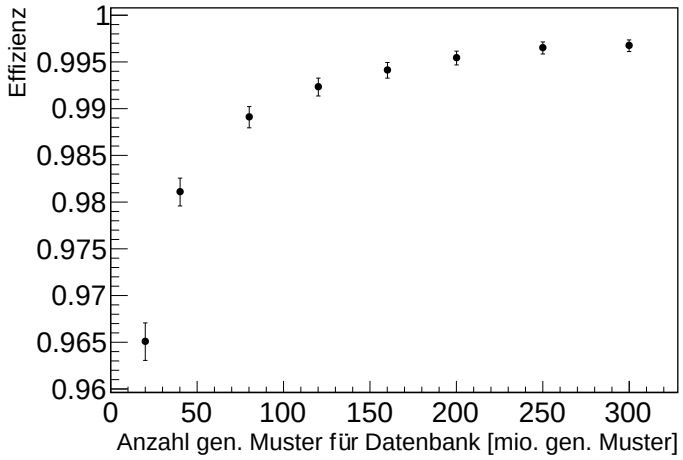

Abbildung 37: Effizienz für das Doppellagenlayout.

Man kann erkennen, dass auch hier die Effizienz den Erwartungen entspricht und sehr stark von der Anzahl Muster abhängt, die zum Aufbau der Datenbank generiert werden. Die Anzahl Templates, die in den verwendeten Datenbanken gespeichert sind, ist in Tabelle 16 zusammengestellt. Sie hat gegenüber dem Einzellagen-Layout deutlich - etwa um den Faktor 10 - zugenommen.

Anzahl gen. Muster für Datenbank	Anzahl Templates in Datenbank
20.000.000	2.050.131
40.000.000	2.588.501
60.000.000	2.929.762
80.000.000	3.181.702
100.000.000	3.382.748
120.000.000	3.551.636
140.000.000	3.696.970
160.000.000	3.825.561
180.000.000	3.941.099
200.000.000	4.045.534
250.000.000	4.272.745
300.000.000	4.462.226

Tabelle 16: Gespeicherte Templates in den Datenbanken.

10.2 Fake-Rate

Auch die Bestimmung der Fake-Rate wird nur mit den Datenbanken aus dem Abschnitt 10.1 durchgeführt. In Rahmen dieser Arbeit konnte aus Zeitgründen auch hier keine Analyse des Einflusses der Selektion der Templates in der Datenbank über den Counter (Cut) durchgeführt werden. Es ist jedoch zu erwarten, dass die Fake-Rate bei diesem Layout deutlich geringer ist als bei dem zuvor betrachteten Layout (siehe Tabelle 13).

Zur Bestimmung der Fake-Rate wird verlangt, dass 5 der 6 Einträge der Templates im Pattern aktiv sein müssen. Bei dieser Bedingung ist die Fake-Rate höher als wenn man verlangt, dass alle Einträge übereinstimmen. Die theoretische Fake-Rate kann in diesem Fall bestimmt werden, indem man annimmt, dass man sechs mal die Kombinationen für 5 Lagen bestimmt. Dabei wird immer eine andere Lage und damit ihre Okkupanz weggelassen. Die Ergebnisse werden dann aufsummiert. Die Formel für die Fake-Rate bei 5 aus 6 Lagen ist in Gleichung 24 dargestellt.

$$F = T \cdot \sum_{l=1}^{Anzahl\ Lagen} \left(\prod_{i=1, i \neq l}^{Anzahl\ Lagen} occ_i \right) \tag{24}$$

Der Vergleich der simulierten Fake-Rate mit Gleichung 24 ist in Abbildung 38 dargestellt.

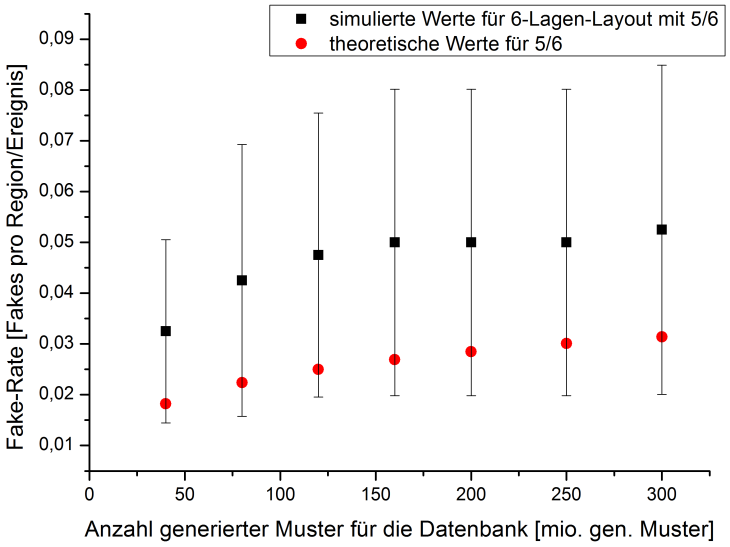

Abbildung 38: Fake-Rate für Doppellagen (5/6).

Im Rahmen der Fehler stimmen die simulierten Werte gut mit der theoretischen Vorhersage nach Gleichung 24 überein. Die simulierten Werte liegen jedoch immer etwas über den theoretischen Werten. Eine Ursache hierfür ist, dass bei dieser Simulation nicht mit Vetos gearbeitet wurde. Aus diesem Grund führt eine Kombination im Pattern, die auf allen sechs Lagen mit einem Template übereinstimmt, dazu, dass bei der 5/6-Bedingung mehrere ähnliche Templates mit dem Pattern übereinstimmen. Für die mit 200 Millionen Mustern generierte Datenbank wurden 400 Ereignisse analysiert und Verteilung der Fakes auf die Ereignisse in Abbildung 39 dargestellt.

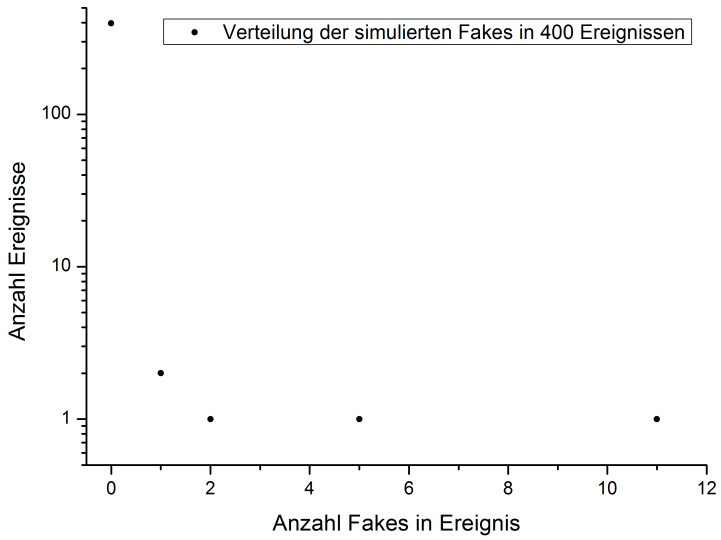

Abbildung 39: Verteilung der Fakes auf die Ereignisse.

Es ist zu erkennen, dass nur in fünf von 400 Ereignissen Fakes zu finden sind, in einem Ereignis allerdings sogar 11 Fakes gleichzeitig. Bei diesem Ereignis ist eine Trefferkombination entstanden, die mit einem Template in allen sechs Einträgen übereinstimmt. Zusammen mit in der Nähe liegenden anderen Treffern im Pattern entstand so diese hohe Anzahl von Fake-Spuren.

Die Okkupanzverteilung für die Regionen auf den sechs Lagen bei den hier betrachteten Ereignissen ist in Abbildung 40 dargestellt. Da immer zwei Lagen sehr nah beieinander liegen, ist die Okkupanz auf diesen Lagen sehr ähnlich.

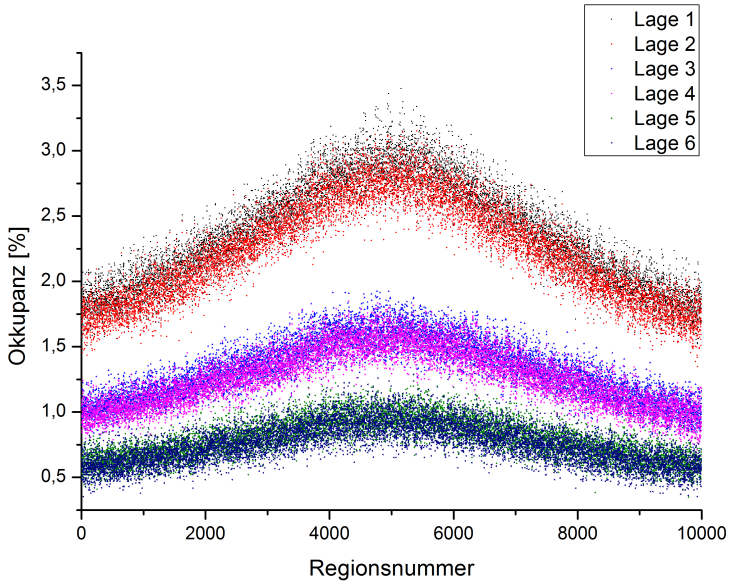

Abbildung 40: Okkupanz auf den betrachteten Detektorlagen.

11 Fazit

In dieser Arbeit wurden bestimmte Aspekte eines L1-Spur-Triggers für ATLAS unter den bei SLHC erwarteten Bedingungen simuliert. Durch die Erhöhung der Luminosität von LHC um den Faktor 10 auf $10^{35}\ \mathrm{cm}^{-2}\mathrm{s}^{-1}$ werden deutlich mehr Spuren auszuwerten sein. Die Erwartung für die Anzahl Ereignisse pro Kollision liegt bei etwa 400, wobei die Kollisionen alle 25 ns stattfinden. Aufgrund dieser erhöhten Datenmenge kann ein L1-Spur-Trigger für ATLAS SLHC notwendig sein. Zur Zeit von LHC verfügt ATLAS noch nicht über einen Spur-Trigger.

Für den L1-Spur-Trigger wurden in dieser Arbeit zwei verschiedene Layouts betrachtet, die unterschiedlich viele Informationen aus dem Inneren Detektor verwenden. Bei beiden Layouts wurden jedoch nur Informationen aus den Silizium-Streifen-Lagen des Inneren Detektors verwendet.

Der betrachtete Teil des Inneren Detektors wurde für die Analyse zunächst in viele voneinander unabhängige Analyseregionen aufgeteilt, die sich aus geometrischen Gründen jedoch überlappen müssen. Für diese Analyseregionen wurde die Überlappung in Abhängigkeit vom Transversalimpuls, auf den getriggert werden soll, bestimmt. Ferner wurde für verschiedene Einteilungen des betrachteten Detektors die Anzahl Verdrahtungen der Auslesechips sowie die Anzahl Bits, die pro Region verarbeitet werden müssen, bestimmt.

Schließlich wurden für beide Layouts Effizienz und Fake-Rate für eine Region analysiert. Es zeigte sich, dass die Fake-Rate bei einem 3-Lagen-Layout so hoch ist, dass der Trigger ohne zusätzliche Selektion der Informationen aus dem Detektor hier nur einen geringen Erfolg haben kann. Man könnte zu diesem Zweck mit einer gesonderten Filteranalyse versuchen Treffer von niederenergetischen Spuren im Pattern zu erkennen und diese nicht an den Trigger zu übermitteln.

Bei dem betrachteten 6-Lagen-Layout ist die Fake-Rate dagegen deutlich geringer. Im Rahmen der statistischen Fehler haben die in der Simulation ermittelten Fake-Raten sich bei beiden Layouts gut durch theoretische Abschätzungen beschreiben lassen. Die Effizienz lag aufgrund der Art der Erstellung der Templates für beide Layouts deutlich über 99 %.

12 Ausblick auf weitere Fragestellungen

In dieser Simulation wurden die Möglichkeiten, die ein tertiärer CAM bietet, nicht genutzt. Eine wichtige Fragestellung ist somit, ob die Fake-Raten für die betrachteten Layouts noch weiter gesenkt werden können, wenn man Nullen als Veto-Entscheidungen in die Templates einbaut. Vor allem beim 5/6-Doppellagen-Layout können hier Nullen die Mehrfachzählung von Spuren durch ähnliche Templates reduzieren.

Bei Benutzung von Veto-Entscheidungen muss auch die Form der Speicherung der Templates, wie sie in dieser Simulation genutzt wurde, überarbeitet werden, da es in diesem Fall sinnvoll ist die Templates unencoded zu speichern.

Ein weiterer Ansatz ist, die Informationen, die aus dem Inneren Detektor kommen, mit einem ergänzenden Algorithmus zunächst zu filtern, um Treffer von niederenergetischen Spuren zu verwerfen, bevor die Informationen an den hier untersuchten Trigger weitergegeben werden.

Da in dieser Simulation keinerlei Aussage über die Zeit getroffen werden konnte, die ein so konzipierter L1-Spur-Trigger benötigt, sollte auch das Zeitverhalten des Triggers untersucht werden.

Literatur

[1] A. Breskin, R. Voss, The CERN Large Hadron Collider: Accelerator and Experiments Volume 1 (2009), 2008 JINST 3 S08001, 2008 JINST 3 S08003

[2] Michelangelo L. Mangano, The super-LHC (2009), http://arxiv.org/abs/0910.0030

[3] http://atlas.web.cern.ch/Atlas/GROUPS/UPGRADES (aufgerufen 24 Juli 2010)

[4] The ATLAS Experiment at CERN, http://atlas.ch

[5] ATLAS collaboration, Detector and physics performance technical design report, CERN/LHCC/99-014 http://cdsweb.cern.ch/record/391176

[6] R. L. Bates, Upgrading the ATLAS barrel tracker for the super-LHC, Nucl. Instrum. Methods Phys. Res., Sect. A 607 (2009) 24-26

[7] S. Schmitt, A. Schöning, A Fast Track Trigger for ATLAS at Super-LHC (2010), Vortrag, Workshop on Intelligent Trackers 2010, 3–5 Februar 2010, Lawrence Berkeley National Laboratory, Berkeley, USA

[8] A. J. McAuley and P. Francis, Fast Routing Table Lookup Using CAMs, in Proceedings of the Conference on Computer Communications (IEEE Infocom), (San Francisco), vol. 3, pp. 1382-1391, March/April 1993

[9] K. Pagiamtzis, Content-Addressable Memory Introduction, http://www.pagiamtzis.com/cam/camintro.html (aufgerufen 18 Juli 2010)

[10] ABCD3T Chip Specification Version V1.2, July 24, 2000, http://scipp.ucsc.edu/groups/atlas/elect-doc/abcd3t_spec.pdf

[11] Konstantin Knizhnik, http://www.garret.ru/gigabase.html, Version 3.74.

Danksagung

Als erstes möchte ich mich bei Herrn Prof. Dr. André Schöning für die Anregung zu dieser interessanten Arbeit und den Einblick in seine Arbeitsgruppe sowie für seine Unterstützung bei der Erstellung dieser Arbeit bedanken.

Bei Frau Prof. Dr. Stephanie Hansmann-Menzemer möchte ich mich für ihr Interesse an dieser Arbeit bedanken und für ihre Bereitschaft, als zweite Prüferin die Arbeit zu bewerten.

Außerdem danke ich meinem Betreuer Sebastian Schmitt für die gute Betreuung bei der Ausführung dieser Arbeit und seine Bereitschaft, bei Fragen und Problemen immer zur Verfügung zu stehen.

Und schließlich möchte ich mich auch bei meinen Eltern für das Korrekturlesen dieser Arbeit bedanken.

Erklärung

Ich versichere, dass ich diese Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Heidelberg, den 25. Juli 2010 Unterschrift: _____