INAUGURAL-DISSERTATION
zur
Erlangung der Doktorwürde
der
Naturwissenschaftlich-Mathematischen
Gesamtfakultät
der
Ruprecht-Karls-Universität
Heidelberg

vorgelegt von
Dipl.-Phys. Holger Singpiel
aus Speyer am Rhein

Tag der mündlichen Prüfung: 8. November 2000

Der ATLAS LVL2-Trigger mit FPGA-Prozessoren

Entwicklung, Aufbau und Funktionsnachweis
des hybriden FPGA / CPU-basierten Prozessorsystems ATLANTIS

Gutachter: Prof. Dr. Reinhard Männer
Prof. Dr. Norbert Herrmann

Zusammenfassung

Diese Arbeit beschreibt die Konzeption und Realisierung des hybriden FPGA/CPU-basierten Prozessorsystems ATLANTIS als Triggerprozessor für das geplante ATLAS-Experiment am CERN. Auf der Basis von CompactPCI wird eine enge Verknüpfung zwischen einem Multi-FPGA-System und einer Standard-CPU umgesetzt. Das System ist in der Rechenleistung skalierbar und flexibel nutzbar. Dies wird durch die Aufteilung in spezifische FPGA-Boards für die Algorithmenausführung und I/O-Funktionalität und durch einen integrierten Privat-Bus erreicht. Die Untersuchungen mit dem ATLANTIS-System beziehen sich auf zwei Kernstellen der 2. Triggerstufe (LVL2). Zum einen soll die Ausführung zeitkritischer B-Physik-Triggeralgorithmen beschleunigt werden. Der im Rahmen dieser Arbeit als Funktionsnachweis durchgeführte Benchmark des Full-Scan-TRT-Algorithmus hat gezeigt, daß die Ausführung gegenüber einer Standard-CPU um einen Faktor 5.6 beschleunigt werden kann. Als zweite ATLAS-Anwendung werden mit dem ATLANTIS-System Studien zu den Readout-Systemen durchgeführt. Für Untersuchungen im LVL2-Prototypensystem ist eine dauerhafte Installation des ATLANTIS-Systems am CERN vorgesehen. Der universelle Charakter von ATLANTIS zeigt sich in weiteren Anwendungen, die für das System entwickelt werden und deren Umsetzung im Rahmen dieser Arbeit unterstützt wurde: Das sind Triggeraufgaben bei Experimenten an der GSI/Darmstadt, die beschleunigte Ausführung von 2D/3D-Bildverarbeitungsanwendungen und die Simulation von N-Körper-Systemen in der Astrophysik. Die Anwendungsentwicklung kann mit der standardisierten Hardwarebeschreibungssprache VHDL durchgeführt werden. Alternativ dazu kann die in Mannheim entwickelte Sprache CHDL benutzt werden. Die Entwicklungs-Tools werden durch das ATLANTIS-Betriebssystem ergänzt.

Abstract

This thesis describes the conception and implementation of the hybrid FPGA/CPU based processing system ATLANTIS as trigger processor for the proposed ATLAS experiment at CERN. CompactPCI provides the close coupling of a multi FPGA system and a standard CPU. The system is scalable in computing power and flexible in use due to its partitioning into dedicated FPGA boards for computation, I/O tasks and a private communication. Main focus of the research activities based on the usage of the ATLANTIS system are two areas in the second level trigger (LVL2). First, the acceleration of time critical B physics trigger algorithms is the major aim. The execution of the full scan TRT algorithm on ATLANTIS, which has been used as a demonstrator, results in a speedup of 5.6 compared to a standard CPU. Next, the ATLANTIS system is used as a hardware platform for research work in conjunction with the ATLAS readout systems. For further studies a permanent installation of the ATLANTIS system in the LVL2 application testbed is forseen. The versatility of the system appears in the various applications which are planned or already implemented: These are trigger applications in the field of heavy ions physics experiments at the GSI/Darmstadt, 2D/3D image processing appplications and n-body simulation in astrophysics. Application development for ATLANTIS can be done either by using the standardised hardware description language VHDL. And it is alternatively supported by CHDL, a programming system developed at Mannheim. The ATLANTIS operating system supplements the software tools.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

KAPITEL 1

Einleitung

Zur Zeit wird am Europäischen Kernforschungszentrum CERN der unterirdische Elektron-Positron-Speicherring (LEP[1]) zum *Large Hadron Collider* (LHC) umgebaut. Beim LHC handelt es sich um einen pp-Beschleuniger mit einer Schwerpunktsenergie von 14 TeV. Das entspricht der siebenfachen Schwerpunktsenergie der zur Zeit leistungsfähigsten Beschleunigeranlage Tevatron[2] am Fermilab. Die Inbetriebnahme des LHC ist für das Jahr 2005 vorgesehen. Die hohe Schwerpunktsenergie und die hohe Luminosität, mit der der LHC betrieben werden wird, eröffnet eine große Bandbreite neuen physikalischen Forschungspotentials - von Präzisionsmessungen der Eigenschaften bekannter Teilchen bis hin zur Erforschung neuer Physik.

Insgesamt vier Experimente sind an diesem neuen Teilchenbeschleuniger geplant: *ATLAS* und *CMS* als Universal-Experimente mit dem Schwerpunkt der Erforschung des Ursprungs der Teilchenmassen, des Blicks jenseits des Standardmodells und der Beantwortung einer Reihe weiterer aktueller Fragen der Physik ([ATP94], [Mit00], [CMS94]). Das *LHC-b* Experiment ist zum Studium der B-Physik und der CP-Verletzung konzipiert und optimiert [LHCb95]. *ALICE* ist ein Schwerionen-Physik-Experiment zur Erforschung des Verhaltens von Kernmaterie bei sehr hohen Temperaturen und Dichten, insbesondere des Quark-Gluon-Plasmas [ALI95].

Ein wichtiger Bestandteil bei modernen Hochenergiephysik-Experimenten dieser Größenordnung sind die Triggersysteme. Die grundsätzliche Aufgabe eines Triggersystem besteht darin, aus dem Eventdatenstrom die physikalisch interessanten Ereignisse für die Offline-Auswertung herauszufiltern. Die hohe Energie und die hohe Luminosität von 10^{34} $cm^{-2}s^{-1}$ bei den LHC-Experimenten stellen extreme Ansprüche an das Triggersystem in Bezug auf Rechenleistung und Datenratenverarbeitung. Das für die hier vorliegende Arbeit interessierende ATLAS-Triggersystem besitzt zur Lösung der Aufgabe einen dreistufigen, hierarchischen Aufbau. Es muß insgesamt eine Datenreduktion um einen Faktor 10^6 - 10^7 erreicht werden. Das bedeutet gleichzeitig die Kollisionsfrequenz der Teilchenpakete von 40 MHz auf die Aufzeichnungsrate von 1-10 Hz zu reduzieren, mit der die Events permanent zur weiteren Offline-Auswertung abgespeichert werden. Die Aufgabe der 2. Stufe des

---

[1] LEP: Large Electron Positron Collider.

[2] 2 TeV-$p\bar{p}$-Ring-Beschleuniger am Fermilab in Chicago/USA.

ATLAS-Triggers (LVL2) besteht darin, eine Eventrate von 100 kHz und einen Eingangsdatenstrom von 150 GByte/s verarbeiten zu müssen. Um die höchstmögliche Effizienz zu erreichen, müssen bereits in dieser Stufe vergleichsweise komplexe Algorithmen ausgeführt werden.

1.1 Motivation

Um den gestellten Anforderungen an den ATLAS LVL2-Trigger gerecht zu werden, sind grundsätzlich zwei mögliche Alternativen als Triggerarchitekturen denkbar: Zum einen eine aus sehr vielen (mehrere hundert bis wenige tausend) Knoten bestehende Prozessorfarm. Demgegenüber steht ein kompaktes und gleichzeitig flexibles System, basierend auf Spezialelektronik. Beide Varianten wurden für die Ausführung der LVL2-Triggeraufgabe untersucht.

Mit den Mannheimer Triggerprozessoren, dem Enable-System und dem Enable++-System, wurde ein neues, erfolgreiches FPGA-Prozessorkonzept eingeführt, mit dem die Anforderungen an den ATLAS LVL2-Trigger erfüllt werden können. Im Gegensatz zu herkömmlichen Mikroprozessoren ist das Rechenwerk bei einem FPGA-Prozessor aus reprogrammierbarer Hardware, den FPGAs (Field Programmable Gate Arrays) aufgebaut. Für jeden auszuführenden Algorithmus werden die FPGAs neu programmiert. Damit kombiniert diese neue Rechner-Architektur die Geschwindigkeit einer speziellen Hardwarelösung mit der Flexibilität einer Softwarelösung. Als nachteilig erweisen sich dabei das zur Zeit noch erforderliche Expertenwissen beim Umgang mit solchen Systemen.

Die Vorteile des Farmkonzepts liegen in der erreichbaren Flexibilität - Algorithmen werden auf Standard-CPUs in Software ausgeführt - und in der Möglichkeit überwiegend kommerzielle Komponenten einsetzen zu können. Auf der anderen Seite steht der Nachweis der Realisierbarkeit des notwendigen Kommunikationsnetzwerks für die hohe Zahl an benötigten Prozessorknoten noch aus. Zur Realisierung einer Prozessorfarm in der Größenordnung, wie sie für den ATLAS LVL2-Trigger gefordert wird, sind erhebliche technologische Fortschritte vonnöten. Bis heute konnte noch nicht gezeigt werden, daß eine solche Architektur aufgebaut werden und die hohen Anforderungen des ATLAS LVL2-Triggers erfüllen kann.

Sowohl das FPGA-Prozessor-Konzept als auch das Farmkonzept wurden in den letzten 5 Jahren von verschiedenen internationalen Gruppen als möglicher LVL2-Trigger für ATLAS untersucht. Bereits 1997 hat sich dabei ein wichtiges Ergebnis herauskristallisiert:

Keine Architektur für sich allein genommen kann die gestellten Anforderungen in Bezug auf Rechenleistung, Datenbandbreite, Aufwand und Kosten optimal erfüllen.

Dieses Ergebnis hat dazu geführt, über eine kombinierte, eine hybride Triggerarchitektur nachzudenken. Und zwar sowohl auf Komponentenebene als auch auf Systemebene.

1.2 Zielsetzung

Das Ziel der hier vorliegenden Arbeit knüpft direkt an die im vorherigen Abschnitt beschriebene Ausgangssituation von 1997 an.

Ihm Rahmen dieser Arbeit wurde einerseits an einem Ansatz für einen hybriden ATLAS LVL2-Trigger auf Systemebene mitgearbeitet. Die grundsätzliche Idee besteht dabei darin, eine zusätzliche, rein FPGA-basierte Triggerstufe (LVL2-Prescaler) innerhalb des LVL2-Triggers einzuführen. Durch die Ausführung geeigneter Algorithmen soll eine Reduktion um den Faktor 10 erzielt werden. Die Konsequenz ist eine deutliche Verminderung der Anforderungen an die nachfolgende LVL2-Prozessorfarm in Bezug auf Datenraten und Netzwerkknoten. Die Realisierung einer kommerziellen Farmarchitektur wird so erleichtert bzw. erst durchführbar.

Der Schwerpunkt der Arbeit bildet jedoch die Entwicklung einer hybriden Architektur auf Komponentenebene, bestehend aus Spezialelektronik (FPGAs) und Standand-CPUs. Diese neue Rechner-Architektur soll als Komponente (Netzwerknoten) innerhalb einer Farm-Architektur zur Lösung zeitkritischer und I/O-intensiver Triggeranwendungen eingesetzt werden.

Für die Entwicklung des neuen Systems werden folgende wichtige Ziele gesetzt: Auf der Basis der Erfahrungen mit dem stand-alone FPGA-Prozessor Enable++ und dem erarbeiteten Wissen bei den bisherigen Untersuchungen von Triggeralgorithmen soll eine neuartige Rechner-Architektur entwickelt werden, mit folgenden wichtigen Eigenschaften:

- Die Ausführung von geeigneten ATLAS-Triggeralgorithmen soll durch eine Abbildung und Verteilung auf eine enggekoppelte Architektur aus FPGAs und CPUs beschleunigt werden.
- Das System soll in Rahmen der ATLAS LVL2-Trigger-Architektur sowohl als Farmprozessor als auch im Bereich der Readout-Systeme einsetzbar sein.
- Die transparente Integration der neuen Architektur soll in die jeweilige Umgebung möglich sein.
- Die Arbeit mit dem System soll auch von Nicht-(Hardware-)Experten möglich sein und muß durch eine entsprechende Bediensoftware unterstützt werden.
- Das System soll als Universal-Rechner nicht nur für ATLAS eingesetzt werden können, sondern auch zur Ausführung von zeitkritischen

Applikationen aus anderen Forschungsbereichen und generell als Prototypensystem.

Eingebettet ist die im Rahmen der hier vorliegenden Promotion geleistete Entwicklungsarbeit in zwei Projektphasen der ATLAS-LVL2-Kollaboration: Im Rahmen des Demonstrator-Programms (Ende 1995 - Februar 1998) wurden die drei Ansätze - ein FPGA-basiertes System und zwei Farmkonzepte - im wesentlichen getrennt voneinander entwickelt und untersucht[3]. In der zweiten Hälfte des Demonstrator-Programms wurde die Prescaler-Architektur entwickelt. Im Rahmen des zeitlich nachfolgenden Pilot-Projekts wurde gruppenübergreifend, basierend auf den Ergebnissen der Demonstrator-Programms, an einer gemeinsamen Triggerarchitektur gearbeitet. In diesem Zeitraum ist das ATLANTIS-System entwickelt und eingesetzt worden.

Zur Beschreibung der Entwicklungsarbeiten gliedert sich der Aufbau der hier vorliegenden Dissertation wie folgt:

Im nachfolgenden Kapitel 2 wird eine Einführung zum ATLAS-Experiment gegeben. Es wird das ATLAS-Physik-Programm vorgestellt und der ATLAS-Detektor beschrieben. Das Kapitel schließt mit einer Darstellung des ATLAS-Triggersystems.

Kapitel 3 beinhaltet die Beschreibung der Entwicklungsarbeit für den ATLAS-LVL2-Trigger in den Projektphasen Demonstrator-Programm und Pilot-Projekt. Im Vordergrund steht dabei die Darstellung der Untersuchungen und der Ergebnisse zu den Einsatzmöglichkeiten von FPGA-basierten Systemen. Im Rahmen des Demonstrator-Programms wurde, unter Beteiligung des Autors, ein rein FPGA-basierter Trigger entworfen und untersucht. Aufgrund der Ergebnisse des Demonstrator-Programms wurden im daran anschließenden Pilot-Projekt weitere Arbeiten zum Einsatz der FPGA-Prozessoren auf zwei Teilbereiche des LVL2-Triggers fokussiert. Die Kernstelle für den Einsatz von FPGAs bildet dabei die Beschleunigung von zeitkritischen Triggeralgorithmen in einer Farm-Umgebung bestehend aus Standard-Prozessoren. Aufgrund dieser zentralen Randbedingung wurde die Entwicklung des hybriden FPGA / CPU-basierten Prozessor-Systems ATLANTIS vorgeschlagen. Die FPGA-Resourcen fungieren in diesem Fall als Koprozessor zu den Standard-CPUs (Koprozessorkonzept). Durch eine enge Kopplung von FPGAs und CPUs können Algorithmen grundsätzlich optimal angepaßt und verteilt gerechnet werden. Die Realisierung des ATLANTIS-Systems und der Funktionsnachweis bilden den Schwerpunkt der hier vorliegenden Promotion.

In Kapitel 4 folgt die Beschreibung der Konzeption des hybriden FPGA / CPU-basierten ATLANTIS-Systems. Die grundsätzlichen Eigenschaften des neuen Systems werden durch die darauf auszuführenden Anwendungen vorgegeben. Im Mittelpunkt steht dabei die Anforderung, die ATLAS-Trigger-Applikationen optimal abbilden und ausführen zu können. Andererseits sollen weitere Anwendungen aus anderen Forschungsbereichen sinnvoll implementiert werden können (siehe Kapitel 7). Diesbezüglich werden die Entscheidungskriterien und Analyseverfahren für die

[3]Am Ende der Projektphase (Oktober 1997) gab es bereits den ersten erfolgreichen Integrationstest mit dem damaligen FPGA Prozessor System Enable++ und der ATM-Farm der französischen Saclay-Arbeitsgruppe in Paris[KBC98].

Gesamtarchitektur und für die Wahl der Komponenten des Systems erläutert. Einen großen Raum nimmt dabei die Evaluierung verfügbarer FPGAs ein, da das FPGA den zentralen Baustein darstellt. Weitere wichtige Punkte sind die Wahl der Schnittstellen zur Kopplung der einzelnen Komponenten innerhalb des Systems und mit externen Systemen. Das Kapitel schließt mit einem konzeptionellen Blick auf das neue System. Um ein solches komplexes System erfolgreich entwickeln zu können sind fruchtbare Diskussionen und Anregungen mit kompetenten Kollegen essentiell. Aus diesem Grund sollen an dieser Stelle namentlich Andreas Kugel, Harald Simmler, Klaus Kornmesser und Klaus-Henning Noffz (Silicon Software) genannt werden, ohne deren Unterstützung das Projekt so nicht hätte durchgeführt werden können.

Kapitel 5 beschreibt die Implementierung des ATLANTIS-Systems. Es werden die einzelnen Hardware-System-Komponenten und der Entwicklungsweg beschrieben. Dies beinhaltet nicht nur die Beschreibung des Aufbaus sondern auch die Anwendung des Systems. Die Entwicklung der Hauptkomponenten (ATLANTIS Computing Board, I/O-Board und der Active-Backplane mit dem Privat-Bus) wurde im wesentlichen vom Autor durchgeführt[4]. Ein separater Abschnitt geht auf die Integration von kommerziellen Komponenten ein. Für den Test und die Nutzung der Hardware ist die Verfügbarkeit entsprechender Software-Tools eine notwendige Voraussetzung. Dies umfaßt sowohl die Software zum Testen und zur Programmierung des Systems als auch die Basis-Software zum Betrieb des Systems. Dies Erstellung dieser Software-Tools wurden nicht im Rahmen dieser Promotion sondern in parallel durchgeführten Promotionen für das ATLANTIS-System entwickelt bzw. basieren auf kommerziellen Produkten und wurden für ATLANTIS angepaßt. In einem Überblick wird gezeigt, wie diese Software-Tools (Entwicklungsumgebungen) mit der Hardware zusammen das System anwendbar machen.

Entscheidend für den Nutzen und die Akzeptanz eines Systems wie ATLANTIS ist die praktische Benutzbarkeit. Zum einen muß gezeigt werden, daß der Einsatz von FPGAs im Sinne des Koprozessorkonzepts grundsätzlich mit dem ATLANTIS-System durchgeführt werden kann. Andererseits muß eine signifikante Beschleunigung im Vergleich zur Ausführung auf einem herkömmlichen Farmprozessoren nachgewiesen werden Dazu wurde der zeitkritische ATLAS Full-Scan-TRT-Triggeralgorithmus als hybride Anwendung auf das ATLANTIS-System abgebildet und Messungen durchgeführt. Diese Thematik wird in Kapitel 6 behandelt. Hybrid bedeutet in diesem Zusammenhang, daß die ersten beiden Algorithmenschritte in den FPGAs und die weiteren beiden Algorithmenschritte auf der CPU ausgeführt werden. Der Full-Scan des TRT-Detektors ist das zur Zeit wichtigste Instrument zur Durchführung des B-Physik-Triggers. Untersuchungen zur Ausführung weiterer zeitkritischer B-Physik-Triggeralgorithmen auf FPGAs sind Gegenstand der aktuellen Forschung. Aus diesem Grund wird die Beschreibung der Ausführung des Full-Scan-TRT-Algorithmus' auf ATLANTIS von einem kurzen Abschnitt zu den B-Physik-Studien und zugehörigen Triggeraufgaben eingeleitet.

Das ATLANTIS-System ist grundsätzlich als universelles Rechnersystem vorgesehen. Das bedeutet, daß eine breite Palette von Anwendungen unterstützt werden soll. Konkret wurden vier Forschungsprojekte mitberücksichtigt, deren spezifische

[4]Die Schaltplaneingabe und das Platinendesign wurden von dem Mannheimer Student Erich Krause durchgeführt.

Anforderungen in die Konzeption des Systems mit eingeflossen sind. Das sind innerhalb des ATLAS-Triggers die Untersuchungen für die Readout-Systeme. Hier soll das ATLANTIS-System als Prototypensystem eingesetzt werden. Außerhalb von ATLAS ist das ATLANTIS-System ebenfalls für den Einsatz als Triggerprozessor bei zwei GSI-Experimenten in Darmstadt vorgesehen. Außerdem wird das ATLANTIS-System als Basis-Plattform für die prototypische Implementierung einer Echtzeit-Volumen-Visualisierungsanwendung genutzt. Im Rahmen eines Astronomie-Projekts wird das ATLANTIS-System als Modul für ein kombiniertes System aus ASICs und Standard-CPUs zur Beschleunigung der rechenintensiven Simulation von N-Körper-Systemen untersucht. Eine Darstellung dieser Projekte, an denen der Autor unterstützend mitgewirkt hat, ist Inhalt von Kapitel 7.

KAPITEL 2

Das ATLAS-Experiment

Das hybride FPGA/CPU-basierte Rechnersystem ATLANTIS, dessen Entwicklung, Aufbau und Funktionsnachweis den Schwerpunkt der hier vorliegenden Promotionsarbeit darstellt, wurde in erster Linie als LVL2-Trigger-Prozessor für das ATLAS-Experiment am CERN konzipiert. Darum wird in diesem ersten Kapitel zunächst das ATLAS-Experiment näher beschrieben: Ausgehend vom LHC und dem ATLAS-Physik-Programm wird der ATLAS-Detektor vorgestellt und ein Überblick über das Triggersystem gegeben.

2.1 Physik am Large Hadron Collider (LHC)

Das Standardmodell (SM) als die zur Zeit erfolgreichste und umfassendste Theorie der Teilchenphysik beschreibt die starke, schwache und die elektromagnetische Wechselwirkung elementarer Teilchen. Die Austauschteilchen der drei Wechselwirkungen sind acht masselose Gluonen und ein masseloses Photon für die starke und die elektromagnetische Wechselwirkung, sowie drei massebehaftete Bosonen (W^+, W^- und Z) für die schwache Wechselwirkung. Innerhalb dieser Eichtheorie ist die fermionische Materie in drei Familien organisiert, zu denen jeweils zwei Quarks und zwei Leptonen gezählt werden. Detaillierte Beschreibungen und Diskussionen finden sich in zahlreichen Schriften, unter anderem in [Loh92], [BPh97].

Die Forderung nach Eichinvarianz verbietet es, sowohl explizite Massenterme für die Eichbosonen als auch für die Fermionen in das Modell einzufügen. Die Brechung der elektroschwachen Symmetrie durch die Einführung eines neuen Feldes mit geeignetem Potential löst dieses Problem und führt zu Massentermen für die Eichbosonen und zu einem neuen Teilchen, dem SM-Higgs-Boson. Das Higgs-Boson ist gleichzeitig, nach der Entdeckung des top-Quarks am Tevatron/Fermilab Mitte der

90er Jahre, das einzige Teilchen dessen Nachweis noch aussteht und somit das letzte noch fehlende Glied im SM.

Für die Masse des SM-Higgs-Teilchens läßt sich sowohl eine untere als auch eine obere Schranke angeben. In der letzten Ausbaustufe des LEP für das finale Betriebsjahr 2000 konnte die Strahlenergie auf 104 GeV gesteigert werden. Damit sind die LEP Experimente sensitiv bis zu einer Higgs-Masse von 135 GeV [LEP00]. Für die obere Grenze liefert die Theorie einen Wert von 1 TeV. Der LHC ist in seiner Größe so ausgelegt, daß die Voraussetzungen für die Erzeugung des Higgs gegeben sind[5]. Somit kann das Standardmodell in diesem Punkt bestätigt werden oder die Theorie muß überarbeitet und ergänzt werden. Denn es ist weder gesichert, daß es nur ein Higgs-Boson gibt, noch, daß überhaupt eines notwendig ist.

Ein vielversprechendes theoretisches Szenario jenseits des SM ist die Supersymmetrie (SUSY). Charakteristisch für die SUSY ist, daß jedem Teilchen des Standardmodells ein supersymmetrischer Partner mit Spindifferenz von +-½ zugeordnet wird. Genauer gesagt heißt das, jedes Boson erhält als supersymmetrischen Partner ein Fermion und umgekehrt, wobei alle anderen Eigenschaften für die Teilchenpaare identisch sind (R-Parität ausgenommen). Die supersymmetrischen Partnerteilchen hätten unter diesen Umständen aber schon längst experimentell nachgewiesen werden müssen. Bis heute wurde aber noch kein experimenteller Hinweis für diese Theorie gefunden. Eine Erklärung für die bislang nicht gefundenen „SUSY-Teilchen" ist, daß die SUSY ebenfalls eine gebrochene Symmetrie ist und die Teilchen eine wesentlich höhere Masse als ihre jeweiligen Partner haben. Einige der leichtesten „SUSY-Teilchen" werden mit einer Masse unter 1 TeV erwartet und könnten somit mit dem LHC nachgewiesen werden [Loh92]. Das zur Zeit sehr populäre theoretische Modell der Minimalen Supersymmetrischen Erweiterung des SM (MSSM) fordert die Existenz von 5 Higgs Teilchen (h, H, A, H^+und H^-) und der supersymmetrischen Partner der bekannten Elementarteilchen. Untere Massengrenzen für charginos und sleptons liegen zur Zeit bei 90-100 GeV (LEP) und für squarks' und gluinos im Bereich von 250 GeV (Tevatron) [Fab00].

Im weiteren sollen mit der Inbetriebnahme des LHC neben der Suche nach dem Higgs und den „SUSY-Teilchen" noch eine Reihe anderer Experimente zur genaueren Untersuchung von Teilchen durchgeführt werden. Unter anderem wird der LHC als Top-Quark-Fabrik arbeiten, wodurch ein detailliertes Studium dieses Teilchen möglich wird. Ebenfalls sind Experimente der B-Physik geplant (siehe auch Kapitel 6). Schwerpunkte sind hierbei die Untersuchung der CP-Verletzung im B_d^0 System und die präzise Bestimmung der Winkel der Cabibbo-Kobayashi-Maskawa-Matrix[ATP94].

Weil die Wirkungsquerschnitte bei der Erzeugung dieser schweren Teilchen sehr gering sind, wurde für den LHC eine extrem hohe Strahlluminosität von $1{,}0 * 10^{34}$ cm^{-2} s^{-1} gewählt[6]. Zu diesem Zweck kreisen im LHC zwei Protonenpakete mit jeweils

[5]Ursprünglich war der Bau der Superconducting Supercolliders (SSC) geplant und auch begonnen worden, dessen Hauptaufgabe die Suche nach dem Higgs gewesen wäre. 1993 wurden allerdings vom amerikanischen Kongreß keine weiteren Mittel mehr bewilligt, womit das SSC-Projekt „gestorben" war. Seitdem richten sich die Hoffnungen der Teilchenphysiker einzig und allein auf den LHC [Gil96].

[6] Zum Vergleich: Beim LEP beträgt die Luminosität $2{,}0 * 10^{31}$ cm^{-2} s^{-1}.

einer Energie von 7000 GeV in entgegengesetzter Richtung, die alle 25 ns (das entspricht einer Eventrate von 40 MHz) zur Kollision gebracht werden. Im Mittel kollidieren dabei 18 - 25 Protonenpaare miteinander.

Einer der geplanten Detektoren, der als Allzweckdetektor das gesamte mögliche Entdeckungspotential des LHC abdeckt, ist der ATLAS-Detektor, der im folgenden Abschnitt näher vorgestellt wird.

Obwohl noch mindestens 5 Jahre bis zum ersten Lauf des LHC vergehen werden, sind Planungen für die Collider der übernächsten Generation bereits im vollen Gange. So werden unter anderem Vorstudien für einen neuen e^+e^--Linearbeschleuniger[7] am DESY / Hamburg durchgeführt. Dieser TeV-Beschleuniger wird allerdings frühestens im Jahr 2010 in Betrieb genommen werden können, vorausgesetzt, die für das kommende Jahr geplante Entscheidung für den Bau fällt positiv aus [BSW99].

2.2 Der ATLAS-Detektor

Viele der physikalischen Fragen, die das ATLAS-Experiment klären möchte, bedingen eine extrem hohe Luminosität. Der ATLAS-Detektor ist dahingehend optimiert, daß er diese Anforderung unterstützt und durch ein komplexes Detektorsystem (Abbildung 2.2) möglichst viele Signaturen von Elektronen, γ-Quanten, Myonen, Jets und $E_{T,miss}$[8] liefert. Die Vielzahl der Signaturen zieht eine gewisse Redundanz in den Messergebnissen nach sich, wodurch eine interne Überprüfung der Detektorkomponenten möglich wird. Während der ersten Betriebsjahre soll der LHC bei niedrigerer Luminosität ($1{,}0 * 10^{33}$ cm^{-2} s^{-1}) arbeiten. Deswegen wurde bei der Konzeption des ATLAS-Detektors auch darauf Wert gelegt, komplexere Signaturen, wie z.B. tau-Leptonen zu detektieren [ATP94].

Der Aufbau des ATLAS-Detektors ist in Abbildung 2.1 dargestellt. Typisch für moderne Detektoren ist die zylindersymmetrische Form mit einem schalenförmigen Aufbau. Grundsätzlich besteht der ATLAS-Detektor, von innen nach außen gesehen, aus dem Innendetektor, dem elektromagnetischen und dem hadronischen Kalorimeter und dem Myonenspektrometer. Alle vier Detektorkomponenten sind wiederum aus einer Vielzahl von Subdetektoren aufgebaut.

[7] Das Tesla Projekt, DESY/Hamburg: http://tesla.desy.de/.

[8] $E_{T,miss}$ ist die missing transverse energy.

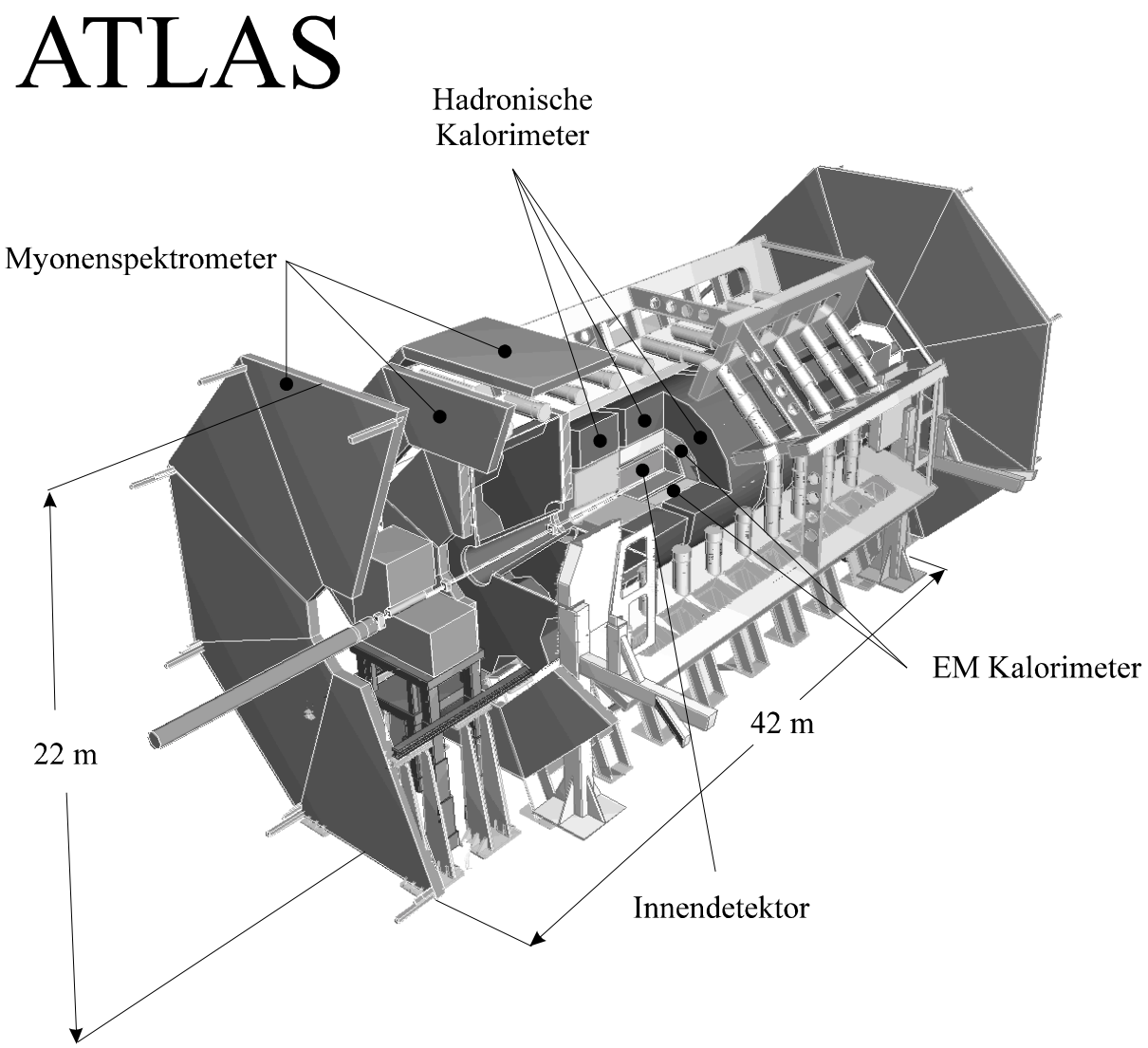


Abbildung 2.1: Der ATLAS-Detektor

Der Innendetektor besteht zum einen aus Halbleiterzählern (in Form von Pixel- und Streifendetektoren), die sich durch eine sehr hohe Ortsauflösung auszeichnen und deswegen direkt um den Wechselwirkungspunkt zum Einsatz kommen. Sie eignen sich besonders zur Impuls- und Vertexbestimmung. Vervollständigt wird das System des Innendetektors durch den Transition Radiation Tracker (TRT), der zwar eine geringere Ortsauflösung als die Halbleiterzähler besitzt, dafür aber eine größere

Anzahl von Punkten pro Spur liefert. Der zylindersymmetrische Teil des Innendetektors wird als Barrel bezeichnet und die beiden äußeren Endstücke als Endcaps (Endkappen). Diese Aufteilung findet sich auch in den anderen drei Subdetektoren wieder. Der markante Unterschied zwischen diesen beiden Bereichen besteht darin, daß im Barrel die Detektoren radial und in den Endcaps scheibenförmig angeordnet sind. Mit dieser Geometrie können die erwarteten Teilchenspuren am effizientesten nachgewiesen werden. Im Bereich des Innendetektors wird zur Impulsbestimmung der geladenen Teilchen mit Hilfe von supraleitenden Spulen ein 2-T-starkes Magnetfeld erzeugt [ATP94].

Nach außen hin folgt dem Innendetekor das EM- und das hadronische Kalorimeter. Im elektromagnetischen Kalorimeter wird die Energie von Elektronen, Positronen und Photonen gemessen. Hierbei wird ausgenutzt, daß diese Teilchen durch sukzessive Bremsstrahlung und Paarbildung eine Kaskade von Sekundärteilchen erzeugen, die ein meßbares Ionisations- oder Lichtsignal liefern. Das hadronische Kalorimeter dient zur Energiemessung von Hadronen. Diese erzeugen in Materie durch eine Reihe von inelastischen Reaktionen überwiegend hadronische Schauer, deren Energie von Sampling-Kalorimetern[9] absorbiert wird [PRS93]. Aufgrund der größeren Ausdehnung der hadronischen Schauer im Vergleich zu den elektromagnetischen nimmt das hadronische Kalorimeter ein größeres Volumen ein.

Den Außenring des ATLAS-Detektors bildet das Myonenspektrometer. Nur die Myonen erreichen aufgrund ihrer außerordentlichen Durchdringungskraft diesen Teil des Detektors und können über Ionisationsreaktionen nachgewiesen werden [ADP99].

Der Detektor hat eine Länge von 42 m und einen Durchmesser von 22 m. Das Gesamtgewicht des Detektors beträgt 7000 to. Bedingt durch die Größe und die hohe Strahlluminosität werden an die Triggerelektronik extrem hohe Anforderungen gestellt. Wie sich das Konzept für das Triggersystem und speziell für den LVL2-Trigger darstellt, wird in den nächsten Abschnitten beschrieben.

2.3 Das ATLAS-Triggersystem

Die grundlegende Aufgabe des Triggersystems besteht darin, aus der Vielzahl der stattfindenden Ereignisse (Events) diejenigen herauszufiltern, deren physikalischer Inhalt vorgegebene Kriterien erfüllt. Das ATLAS-Triggersystem (Abbildung 2.2) unterteilt sich dazu in drei Ebenen (LVL1, LVL2 und Event Filter).

In der ersten Stufe werden vom LVL1-Trigger nur Informationen aus den Kalorimetern und dem Myonenspektrometer für die Triggerentscheidung herangezogen. Die Daten erhält der LVL1 mit der vollen *bunch crossing*-Rate[10] von 40 MHz. Der Datenstrom wird hierbei mit 60 TByte/s abgeschätzt. Die Entscheidungszeit (Latency) ist fest vorgegeben und beträgt 2 µs. Während der LVL1 die für ihn relevanten

[9] Sampling-Kalorimeter bestehen aus alternierenden Ebenen von reinem Absorbermaterial (z.B. Eisen, Uran) und Nachweismaterial (z.B. organischen Szintillator).

[10] Kollisionsfrequenz der Teilchenpakete.

Teildaten prozessiert[11], wird der komplette Datensatz des zugehörigen Events in den Readout-Drivern zwischengespeichert. Liegt eine positive LVL1-Triggerentscheidung vor, werden die Event-Informationen dem LVL2 zugeführt. Von dieser ersten Triggerstufe wird eine Reduktion der *bunch crossing*-Rate von 40 MHz auf ca. 100 kHz gefordert, was einem Unterdrückungsrate von 400 entspricht.

Die Eingangsrate für den LVL2 beträgt maximal 100 kHz. Die Daten werden zunächst aus den RODs in die ROBs übernommen, von wo der LVL2 die für die Triggerentscheidung relevanten Daten abruft. Die Vorgabe für diese Triggerstufe ist eine Reduktion um den Faktor 100. Auch auf dieser Triggerebene wird nicht der vollständige Datensatz zur Entscheidungsfindung verwendet, sondern es werden nur diejenigen Detektorbereiche ausgewertet, in denen physikalisch interessante Ereignisse stattgefunden haben. Die Information, welche Bereiche dies sind, kommt vom LVL1. Dieses Konzept der *regions of interest* (RoI) wurde eingeführt, um die Anforderungen an den LVL2 zu senken. Dadurch, daß der LVL2 nur einen Teil des Detektors auswerten muß, reduziert sich der Eingangsdatenstrom für diese Triggerstufe von 150 GByte/s auf ca. 1-5 GByte/s. Im Fall der B-Physik ist das RoI-Konzept nicht anwendbar. Der LVL1-Trigger ist in diesem Fall sensitiv auf ein 6-GeV-Myon-Signal. Das bedeutet für den LVL2 die Suche von Elektronen- und weiteren Teilchenspuren im gesamten Volumen des Innendetektors (TRT, SCT und Pixel). Sowohl die zu verarbeitende Datenrate als auch die Rechenleistung sind dadurch drastisch erhöht. Dies wird teilweise durch die relativ geringe LVL1-Eventrate für B-Physikereignisse von 6-9 kHz im Betriebsmodus bei geringer Luminosität kompensiert, während dem hauptsächlich die B-Physik-Untersuchungen stattfinden sollen. Der LVL2-Trigger wird ausführlich in Kapitel 3 behandelt, da er für die vorliegende Arbeit von besonderem Interesse ist.

Nachdem ein Event vom LVL2-Trigger akzeptiert worden ist, erhält die dritte und letzte Triggerstufe (Event Builder + Event Filter) den kompletten Datensatz mit einer maximalen Rate von 1 bis 5 kHz. In der dritten Stufe wird eine Reduktion des Eingangsdatenstroms um den Faktor 10 gefordert, woraus sich eine Datenaufzeichnungsrate von 100 bis 150 MByte/s ergibt. Der Event Filter soll als Farm von Standardprozessoren realisiert werden. An dieser Stelle kann schon fast eine vollständige Rekonstruktion des Ereignisses mit Algorithmen durchgeführt werden, die den Offline-Analyse-Algorithmen schon sehr nahe kommen. Bei positiver EF-Entscheidung wird das Event zur weiteren Auswertung permanent auf Datenträgern abgespeichert. Die typische Größe eines Events wird mit ca. 1 - 1,5 MByte abgeschätzt.

[11] Als Hardwarekomponenten kommen hier Spezialprozessoren (ASICs) zum Einsatz.

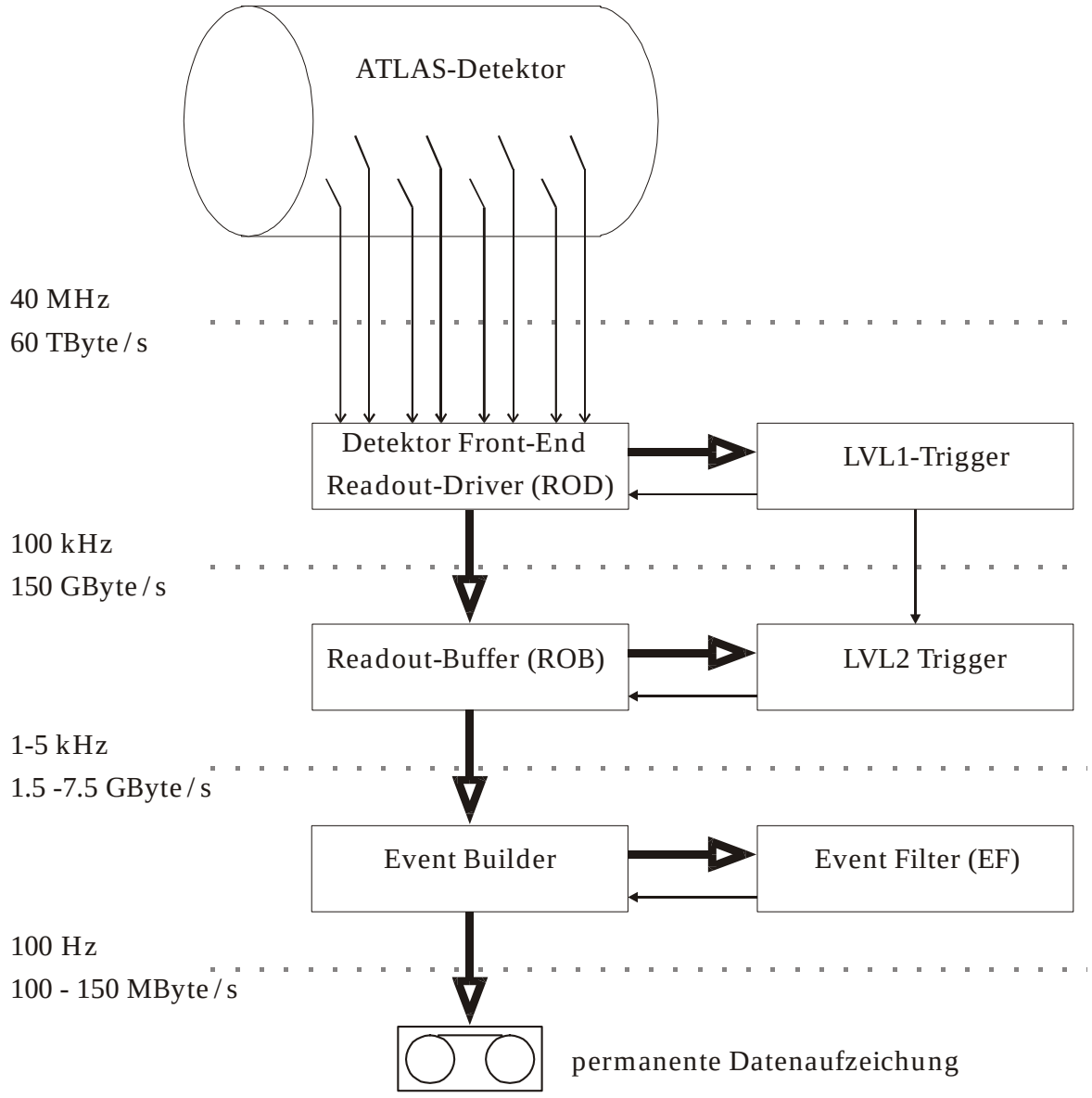


Abbildung 2.2: Die dreistufige Architektur des ATLAS-Triggersystems nach [ATP94]

KAPITEL 3

Der ATLAS LVL2-Trigger mit FPGA-Prozessoren

Im diesem Kapitel wird die Forschungs- und Entwicklungsarbeit für die Realisierung des ATLAS LVL2-Triggers beschrieben. Schwerpunkt ist dabei die Rolle und Einsatzmöglichkeiten von FPGA-basierten Rechnern. Die Darstellung der Untersuchungen folgt in zeitlicher Reihenfolge und beginnt mit der Phase nach Veröffentlichung des *ATLAS Technical Proposals* 1994. Sie beginnt mit der kurzen Diskussion um die beiden anfänglich konkurrierenden Triggerarchitekturen - Prozessorfarm gegen Datenflußrechner (FPGA-Prozessoren). Die Untersuchungen dazu wurden im Rahmen des Demonstrator-Programms durchgeführt. Die Argumente für und gegen die beiden Konzepte und den daraus abgeleiteten Untersuchungen haben die Aktivitäten in der darauf folgenden Projektphase des Pilot-Projekts, in der an einer gemeinsamen LVL2-Triggerarchitektur geforscht wurde, maßgeblich beeinflußt. Innerhalb der Pilot-Projekt-Phase ist das ATLANTIS-System als hybrider FPGA / CPU-basierter Farmprozessor entwickelt worden. Der Funktionsnachweis als leistungsstarker Triggerprozessor wurde durch die erfolgreiche Implementierung des Full-Scan-TRT-Algorithmus' demonstriert.

3.1 Zeitplan und Milestones für die Entwicklung des LVL2-Triggers

Für die Forschungs- und Entwicklungsarbeiten für den ATLAS LVL2-Trigger wurden folgende Projektphasen und Milestones definiert (Abbildung 3.1). Von 1996 bis Anfang 1998 wurden verschiedene Alternativen als LVL2-Triggerarchitekturen im Rahmen des Demonstrator-Programms getrennt voneinander aufgebaut und untersucht. Die Ergebnisse dieser Aktivitäten sind im *ATLAS DAQ, EF, LVL2 and DCS Technical Progress Report* (TPR) festgehalten.

In der nachfolgenden Pilot-Projekt-Phase (Anfang 1998 bis Ende 1999) sollte an einer gemeinsamen Architektur gearbeitet werden. Dabei wurden für die einzelnen Architekturteile alternative Technologien untersucht. Mit der Veröffentlichung des *ATLAS High-Level Triggers, DAQ and DCS Technical Proposal* (TP) Ende 1999 sollte ursprünglich die prinzipielle LVL2-Triggerarchitektur festgelegt werden. Im Rahmen der nachfolgenden Projektphase, die bis Mitte 2001 dauern soll, wird ein LVL2-Prototyp aufgebaut. Für Mitte 2001 ist das Erstellung des *ATLAS HLT Technical Design Reports* (TDR) vorgesehen, in dem sowohl die endgültige LVL2-Triggerarchitektur als auch die verwendeten Technologien definiert werden. Bis zum Start des ATLAS-Experiments im Juli 2005 werden nach und nach die erforderlichen Komponenten beschafft und der vollständige Trigger aufgebaut.

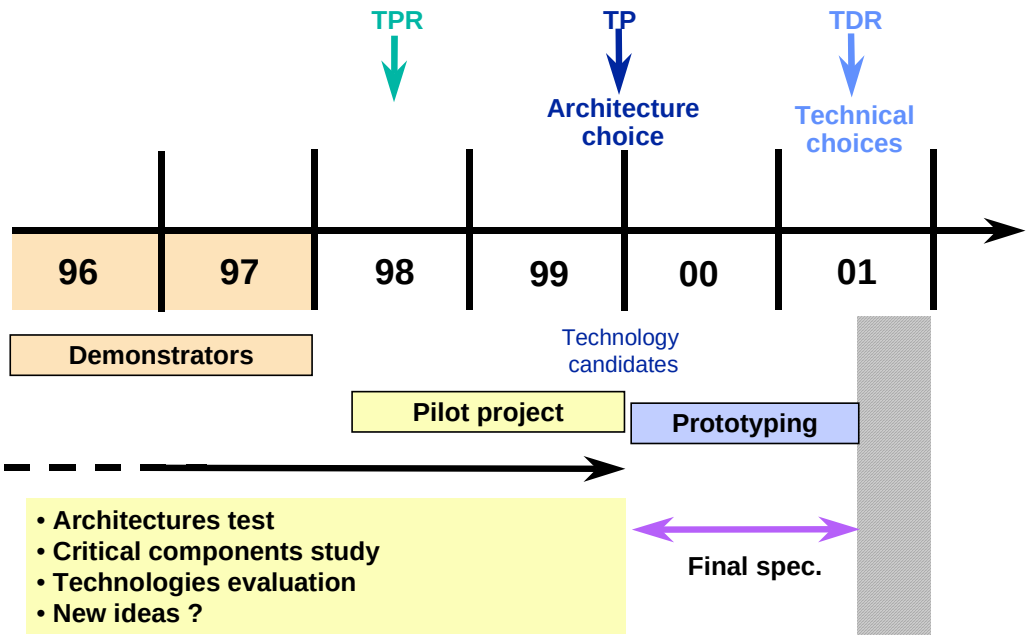


Abbildung 3.1: ATLAS LVL2: Zeitplan und Milestones

3.2 LVL2-Trigger-Konzepte der Anfangsphase

Wie im vorangegangenen Kapitel bereits angesprochen wurde, besteht die Aufgabe des LVL2 grundsätzlich darin, die Eingangseventrate von 100 kHz und damit gleichzeitig den Eingangsdatenstrom von ca. 150 GByte/s um den Faktor 100 zu reduzieren.

Das in [ATP94] definierte Schema sieht dafür ein zweigeteiltes LVL2-Triggersystem vor. Im ersten Schritt werden parallel für sämtliche Detektoren, die in ein Event involviert sind, der abstrakte Informationsinhalt in physikalische Teilchenspuren umgesetzt (Feature Extraction). Im zweiten Schritt (globale Entscheidung) werden zunächst zur Teilchenidentifikation und zur Bestimmung von Impuls und Energie die Spuren aller Subdetektoren, die zu jeweils einer RoI gehören, zusammengesetzt und ausgewertet (RoI Task). Auf Eventebene (Event Task) werden die Teilchen aller RoIs kombiniert, zu dem Event zusammengefaßt und die globale Triggerentscheidung gefällt.

3.2.1 LVL2-Triggerarchitekturen

Für die Realisierung des LVL2-Triggers wurden verschiedene Konzepte vorgeschlagen, die sich in zwei Kategorien einordnen lassen. Das ist auf der einen Seite der Datenfluß- oder datengetriebene Trigger (data driven trigger), der synchron zu der 100 kHz Eingangsrate arbeiten kann und auf der anderen Seite das Farmkonzept, innerhalb dessen die Algorithmen asynchron zu den 100 kHz ausgeführt werden und deshalb ein großer Buffer (Zwischenspeicher) für die Detektordaten benötigt wird. Beiden gemeinsam ist die Erfüllung der Grundanforderung an den LVL2. Diese besteht darin, daß für die verschiedenen Typen von Detektoren nur eine Hardware bzw. nur eine Prozessorarchitektur zum Einsatz kommen soll (Universalität der Hardware). Das bedeutet für die Prozessoren, daß sie ein hohes Maß an Flexibilität aufweisen müssen. Flexibilität heißt in diesem Zusammenhang, daß neben der freien Programmierbarkeit einer solchen Hardware auch die Rechenleistung und die I/O-Bandbreite skalierbar sein sollten, um den unterschiedlichen Anforderungen der Detektoren zu genügen. Im nachfolgenden werden die beiden Konzepte kurz vorgestellt.

3.2.2 Der Datenfluß-Trigger

Die Hardware des **Datenfluß-Triggers** basiert auf sehr leistungsfähigen, FPGA-basierten Prozessoren, die synchron zur Eventrate von 100 kHz arbeiten können. Mit einem solchen Spezialprozessor muß es möglich sein, die Feature Extraction [FEX] für ein RoI innerhalb der geforderten 10 μs auszuführen. Die zu prozessierenden

Daten werden aus dem (vom LVL1 kommenden) 150 GByte/s großen Eventdatenstrom extrahiert, zu RoI-Datensätzen zusammengesetzt und mit einer Datentransferrate von 10-15 GByte/s an die FEX Prozessoren weitergeleitet. Das Sammeln, Zusammensetzen und Weiterleiten der Daten zu den FEX-Prozessoren übernimmt dabei eine ebenfalls sehr leistungsfähiges System: Es wird als *Data Collection Unit* oder auch Router bezeichnet [DKR95]. Die einzelnen Ergebnisse der FEX-Prozessoren werden einem weiteren Prozessor zugeführt, der die globale Triggerentscheidung (Ja/Nein) trifft.

Im Rahmen der EAST/RD11-Kollaboration wurde bereits vor der Demonstrator-Programm-Phase ein Benchmark durchgeführt, dessen Ziel es war, eine geeignete Prozessorarchitektur zu finden. Dazu mußten auf den angetretenen Systemen verschiedene LVL2-Referenzalgorithmen ausgeführt werden. Als einziges System konnte das Mannheimer FPGA-basierte Enable-System alle Anforderungen erfüllen. Dies führte dazu, daß die Untersuchungen mit FPGA-Prozessoren, wie dem Enable-System und einem weiteren FPGA-Prozessor-System DecPeRLe-1, am CERN weitergeführt wurden [BBB93].

3.2.3 Das Farmkonzept

Innerhalb des **Farmkonzepts** sollen Standardprozessoren zum Einsatz kommen, die über eine Hochsprache (z.B. C, C++) programmiert werden. Vorstellbar wäre eine Farm von Workstations, der die RoI-Daten übermittelt werden und die dann die Feature Extraction ausführt. Die Standardprozessoren lassen sich vergleichsweise einfach programmieren, können aber nicht mit der Geschwindigkeit der programmierbaren Spezialprozessoren mithalten. Dies wird dadurch ausgeglichen, daß sehr viele Prozessoren parallel die Algorithmen rechnen. Die Hauptschwierigkeit innerhalb des Farmkonzept liegt daher in der Realisierung eines effektiven Verbindungsnetzwerkes, mit dessen Hilfe die sehr große Anzahl von Prozessoren miteinander kommunizieren kann.

3.2.4 Vergleich der Konzepte

Für einen generellen Vergleich der beiden Konzepte lassen sich drei Hauptkriterien angeben:

- Möglichkeit der Verwendung von kommerziellen Komponenten, da der Einsatz von kommerziellen Produkten innerhalb der ATLAS-Kollaboration stark befürwortet wird
- Flexibilität in der Programmierbarkeit und Skalierbarkeit in Bezug auf Rechenleistung und Datenbandbreite
- Realisierungaufwand, um die geforderte Leistungsfähigkeit zu erreichen

Die Vorteile einer Farmlösung liegen bei diesen Gesichtspunkten klar auf der Hand. Die Programmierung in Software bietet die größtmögliche Flexibilität bei der Implementierung der Algorithmen. Durch den Einsatz von Standard PCs und einer kommerziellen Netzwerktechnologie kann direkt vom technologischen Fortschritt profitiert werden. Über ein entsprechendes Kommunikationsnetzwerk lassen sich fast beliebig Rechenleistung und Datenbandbreite durch Hinzufügen oder Entfernen von Knoten skalieren.

Aber genau hier zeigt sich das große Problem bei Realisierung einer Prozessorfarm, wie sie für den ATLAS LVL2-Trigger gefordert wird. Bis heute konnte noch nicht gezeigt werden, daß es mit einer kommerziellen Netzwerktechnologie möglich ist, unter der geforderten Randbedingungen mehrere hundert bis einige tausend Prozessoren miteinander zu verbinden.

Die Datenflußrechner auf FPGA-Basis konnten bereits ihre Leistungsfähigkeit unter Beweis stellen und demonstrieren, daß sie in der Lage sind, die Anforderungen an den LVL2-Trigger zu erfüllen. Ein wesentlicher Nachteil dieses Konzepts ist, daß für die Programmierung und für den Betrieb solcher Systeme zur Zeit noch Expertenwissen vorausgesetzt wird. Mit dem zunehmenden Fortschritten bei der Entwicklung von effizienten und anwenderfreundlichen Hochsprachen für FPGA-Prozessoren ist die Überwindung dieses Nachteils aber nur noch eine Frage der Zeit. Ein Ansatz war beispielsweise die in Mannheim entwickelte Hochsprache ppC [Zoz97]. Ein weiterer Nachteil (zum damaligen Zeitpunkt) war die Nichtexistenz kommerzieller Systeme. Bis zum Jahr 2000 hat mittlerweile die Zahl der Firmen, die FPGA-basierte Syteme anbieten, deutlich zugenommen. Auch wenn die Eigenschaften, insbesondere die Leistungsfähigkeit dieser kommerziellen Systeme heute noch nicht die LVL2-Anforderungen erfüllen, sind für die kommenden Jahre entsprechende kommerzielle Produkte zu erwarten.

3.3 Das Demonstrator-Programm

Mit dem Demonstrator-Programm wurde ein R&D-Wettbewerb gestartet, um die geeignetste Architektur für den ATLAS LVL2 Trigger zu ermitteln. Von Ende 1995 bis Februar 1998 traten drei verschiedene Konzepte - ein FPGA-basiertes Datenflußrechnersystem und zwei Prozessorfarmen - an, um unabhängig voneinander ihre Eignung als LVL2-Trigger zu demonstrieren.

3.3.1 Ziel des Demonstrator-Programms

Alle drei konkurrierenden Architekturen galten innerhalb der Kollaboration als sich ausschließende Konzepte. Es war daher vorgesehen, für jedes der Konzepte einen, die volle Funktionalität umfassenden, aber aus nur wenigen Komponenten bestehenden Demonstrator aufzubauen (*vertical slice).* Mit dem Ende des

Demonstrator-Programms Anfang 1998 sollte ursprünglich die Entscheidung über den LVL2-Trigger gefällt werden. Im nachfolgenden werden die drei Architekturen näher vorgestellt. Schwerpunkt liegt dabei auf der Darstellung der Mannheimer Arbeiten, die unter dem Synonym Architektur-A durchgeführt wurden.

3.3.2 Architektur-A

Die Architektur-A folgt prinzipiell der Vorgabe des *ATLAS Technical Proposals* von 1994, wie in Abschnitt 3.2 beschrieben [ATP94]. Als Prozessoren werden an der Kernstelle FPGA-Prozessoren eingesetzt. Sie arbeiten als lokale FEX-Prozesssoren mit der vollen LVL2-Eventrate von 100 kHz. Abbildung 3.2 zeigt den grundsätzlichen Aufbau.

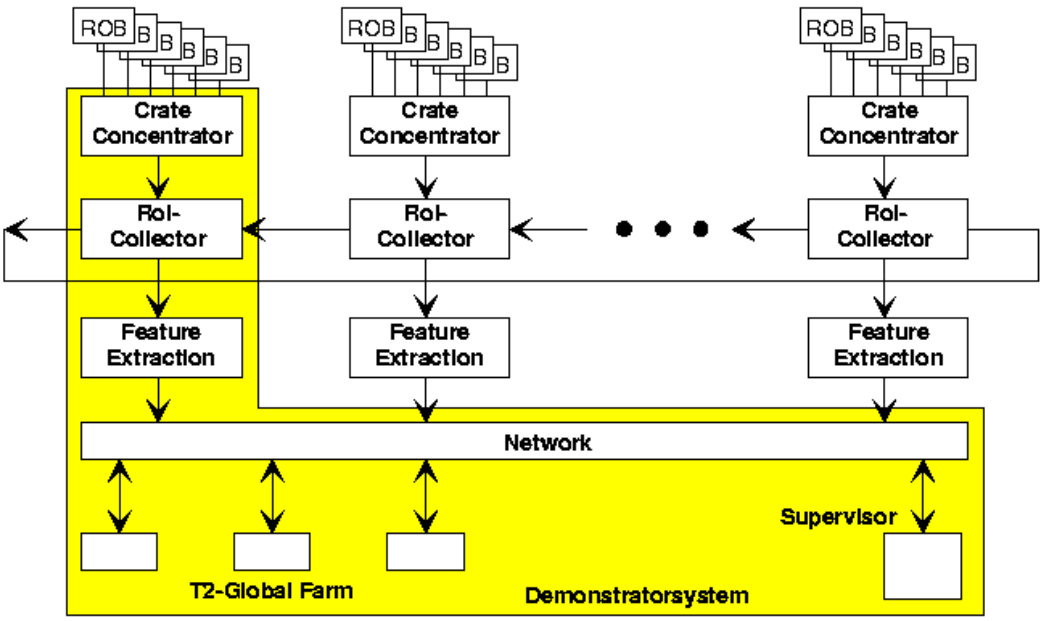


Abbildung 3.2: Demonstrator-A Architektur

Die Architektur-A besteht aus folgenden funktionalen Ebenen:

- Das **ROB**-Interface verbindet die Architektur-A mit den Readout-Buffern (ROBs). Die ROBs - zu Gruppen zusammengefaßt in ROB Crates (CC) - beinhalten Datenvorverarbeitungs-Module und die Schnittstelle zur nächsten hierarchischen Ebene, der RoI-Collection-Ebene
- Die Aufgabe der **RoI Collection** (RoIC) besteht in dem Zusammenfügen von RoI-Fragmenten zu vollständigen RoIs. Eine Kommunikation

zwischen RoIC-Einheiten ist aufgrund der Verteilung der Fragmente notwendig

- In der nachfolgenden **FEX**-Ebene erfolgt die Ausführung der Triggeralgorithmen in den FPGA-Prozessoren
- Die endgültige LVL2-Entscheidung wird in der globalen Prozessorfarm (**T2-Global Farm**) getroffen - basierend auf den Ergebnissen der FEX-Ebene
- Der **Supervisor** besitzt ein Interface zum LVL1 / Event Filter und hat allgemeine Kontrollfunktionen.

Die Architektur-A-Untersuchungen konzentrierten sich auf die ersten drei Ebenen. Die RoIC Komponente ist eine Entwicklung der Universität Jena [DKR95]. Sie ist als Modul des Enable++-Systems ausgeführt und somit direkt in das FEX-System integrierbar. Der FPGA-Prozessor Enable++ ist als FEX-System die zentrale Komponente. In diesem System werden die eigentlichen Triggeralgorithmen zur Teilchenidentifikation ausgeführt. Die Recheneinheit des Enable++-Systems wurde im Rahmen der Promotion von Jozsef Ludvig entwickelt [Lud98]. In dieser Arbeit findet sich eine umfassende Darstellung von ATLAS-Triggeralgorithmen und zu Ausführungsmöglichkeiten auf FPGAs.

Abbildung 3.3 zeigt den Demonstrator-A Meßaufbau. Dieser vergleichsweise kleine Aufbau ist zur Demonstration des Datenflußrechnerkonzepts völlig ausreichend, da keine zusätzliche Funktionalitäten, wie in den anderen beiden Architekturen, benötigt werden. Zum Beispiel muß keine Verwaltung der Farmprozessoren (Resourcen-Allokierung) durch einen Supervisor durchgeführt werden. Als Schnittstelle zum ROB und zur globalen Farm dient ein PC, ausgestattet mit zwei microEnable[12] FPGA-Koprozessoren. Die Datenübertragung erfolgt mittels S-Link[13].

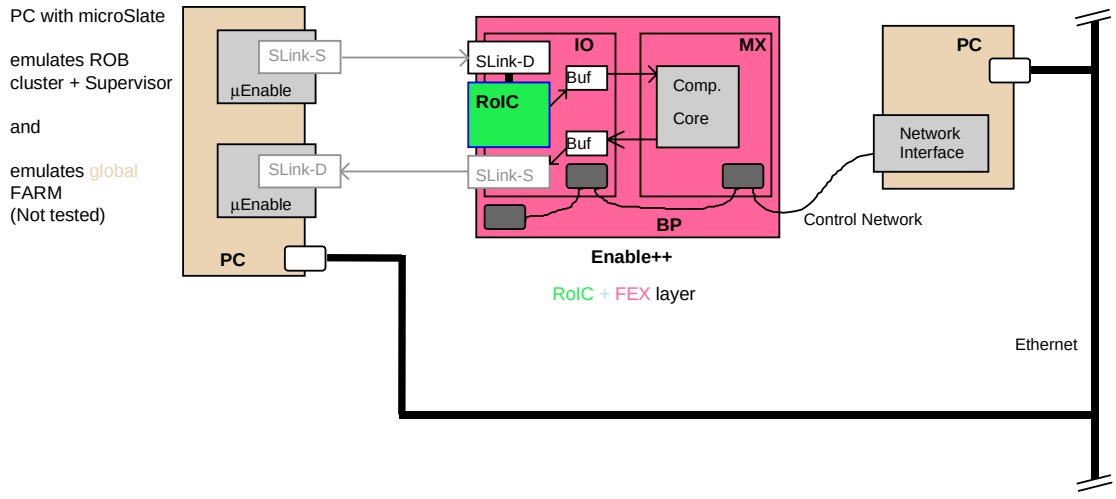


Abbildung 3.3: Demonstrator-A: Setup für Messungen (vertical slice)

[12]microEnable ist ein PCI-Bus basierter FPGA-Koprozessor, ursprünglich entwickelt an der Universität Mannheim und jetzt kommerziell erhältlich (http:///www.silicon-software.de).
[13]vgl. Abschnitt 7.1.3.

Für die **Messungen** wurde als Referenzalgorithmus der TRT-Fex-Algorithmus[14] auf dem Enable++ System ausgeführt. Dieser Algorithmus benutzt als zentrales Element die Hough-Transformation[15]. Grundsätzlich besteht er aus mehreren Schritten - der bei weitem rechenintensivste und zugleich aufwendigste Schritt, die Spurensuche im gesamten Detektorvolumen, wurde auf dem Enable++ System implementiert[16]. Für den Einsatz im Rahmen des hybriden Prescalerkonzepts (Abschnitt 3.3.4) ist diese Implementierung des TRT-FEX-Algorithmus ausreichend effizient. Untersucht wurden sowohl die RoI-geführte Spurensuche als auch der sehr aufwendige Full-Scan des TRT-Detektors, wie er für die B-Physik-Studien im Betrieb bei niedriger bis mittlerer Luminosität gefordert wird.

Die **Ergebnisse der Architektur-A Messungen und Untersuchungen** werden in zwei Dokumenten ausführlich beschrieben ([KBC98], [SDE98]). Darauf aufbauend wurde vom Autor ein Papiermodell für eine hybride LVL2-Architektur[17] mit FPGA-Prozessoren entwickelt [SKK98]. Insgesamt hat das Architektur-A-Projekt die Leistungsfähigkeit und die Vorteile von FPGA-Prozessoren für den Einsatz an kritischen Stellen des ATLAS LVL2-Triggers demonstriert. An dieser Stelle werden die wichtigsten Ergebnisse zusammengefaßt dargestellt:

- FPGA-FEX und RoIC mit 100 kHz
- High-Speed Kommunikation mit S-Link (100 MByte/s pro Link)
- TRT-Eventdatenvorverarbeitung auf FPGAs
- FPGA-Prescaler[18] als FEX-System mit Triggerentscheidung zur Entlastung einer nachfolgenden Prozessorfarm
- Der FPGA-Prescaler ist ein kompaktes System, bestehend aus nur drei verschiedene Hardware-Plattformen, wovon das microEnable-System kommerziell erhältlich ist

Eine wichtige Rolle im Zusammhang mit der LVL2 Feature Extraction (FEX) spielt die **Eventdatenvorverarbeitung**. Mit Datenvorverarbeitung ist in diesem Zusammenhang die Ausführung von Operationen auf die von der Detektorelektronik kommenden Datenformate gemeint. Dies umfaßt typischerweise Nullenunterdrückung, Datenkomprimierung und Formatkonvertierung. Die Ausführung ist im Bereich der Readout-Buffer vorgesehen.

[14]Diese Algorithmus-Version führt, bezogen auf den im Kapitel 6 beschriebenen Full-Scan-TRT-Algorithmus nur den ersten Schritt aus.

[15]Die Hough-Tranformation ist eine zentrale Methode in der Bildverarbeitung. Vereinfacht ausgedrückt werden die Bilddaten (Detektordaten) auf vorberechnete Spurmuster abgebildet. Die Anzahl übereinstimmender Bildpunkte mit den Spurmustern werden für jedes Spurmuster aufsummiert. Die Spurmuster, deren Summe über einer einstellbaren Schwelle liegen, werden als gefundene Spuren interpretiert (siehe auch Abschnitt 6.3).

[16] Die prinzipielle Leistungsfähigkeit der FEX-Algorithmen weiterer ATLAS-Detektoren bei der Ausführung auf FPGA-Prozessoren wurden bereits gezeigt. Eine ausführliche Diskussion findet sich in [Lud98].

[17]Für das Papiermodell wurden die Randbedingungen für die FPGA-basierte Prescaler-Architektur zu Grunde gelegt (vgl. Abschnitt 3.3.4)

[18]Die Beschreibung des FPGA-basierten Prescalers findet sich in Abschnitt 3.3.4.

Hauptziele der Datenvorverarbeitung sind, zum einen durch eine Komprimierung die Größe der zu übertragenden Datenpakete zu verkleinern und durch entsprechende Formatierungen ein geeignetes Datenformat für die FEX-Prozessoren zu erzeugen. Im Rahmen des Demonstrator-Programms wurden in Mannheim ein Eventdatenvorverarbeitungsalgorithmus für den TRT-Detektor entwickelt und für die Ausführung auf FPGAs untersucht [BNS97]. Das Datenvolumen des Eventdatensatz konnte um bis zu 30% reduziert werden. Zusätzlich ergeben die Ausführung auf FPGAs gegenüber einer Standard-CPU einen Speedup von 10. Das Format läßt sich gleichermaßen sowohl auf Standard-CPU FEX-Prozessoren als auch auf FPGA-basierten FEX-Prozessoren verarbeiten.

3.3.3 Architektur-B

Im Architektur-B-Modell wird der FEX-Task für jeden Subdetektor von einer lokalen Farm gerechnet. Die Ergebnisse aus diesen lokalen Farmen werden in einer globalen Prozessorfarm kombiniert und daraufhin die LVL2-Triggerentscheidung getroffen. In dieser Architektur veranlaßt der Supervisor die ROBs, die Daten an die lokalen FEX-Prozessoren (LP) zu senden. Nach Ausführung der Feature Extraction werden die Ergebnisse an einem Prozessor der globalen Farm (GP) weitergeleitet, der die Triggerentscheidung fällt. Im Gegensatz zur Architektur-A, in der ebenfalls eine FEX-Ebene Bestandteil der Architektur ist, kann hier die Latenzzeit einige 100 µs oder sogar einige ms betragen.

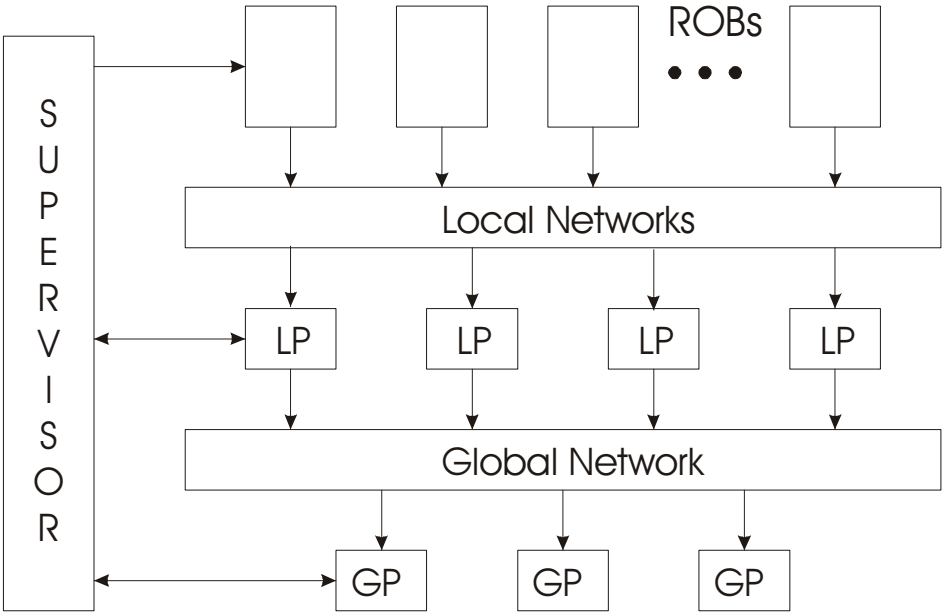


Abbildung 3.4: Architektur-B

Für die Demonstrator wurden zwei verschiedene Netzwerktechnologien getestet: DS-Link und SCI. Insbesondere die Arbeit mit dem MACRAME-System hat wichtige Erkenntnisse für die Entwicklung von Prozessorfarmen geliefert. [ADE98].

3.3.4 Architektur-C

Haupteigenschaft der Architektur C ist, daß nur eine einzige Prozessorfarm für FEX und globale Entscheidung favorisiert wird. Alle Komponenten sind direkt über einen großen Netzwerk-Switch miteinander verbunden. Dabei wird jeweils ein Event von einem Prozessor abgearbeitet. Er führt sowohl die Feature Extraction (Schritt für Schritt für jeden Subdetektor nach einer vorgegebenen Reihenfolge) als auch die globale Entscheidung aus. Um die Netztwerklast und die benötigte Rechenleistung zu reduzieren wird eine sequentielle Triggerstrategie [BCE96] eingeführt. Dabei wird nach jedem einzelnen Zwischenschritt geprüft, ob das Event verworfen werden kann oder weiter untersucht werden muß. Die Funktionalität des Supervisors beschränkt sich auf die Verwaltung der Farmprozessoren, die völlig selbständig arbeiten.

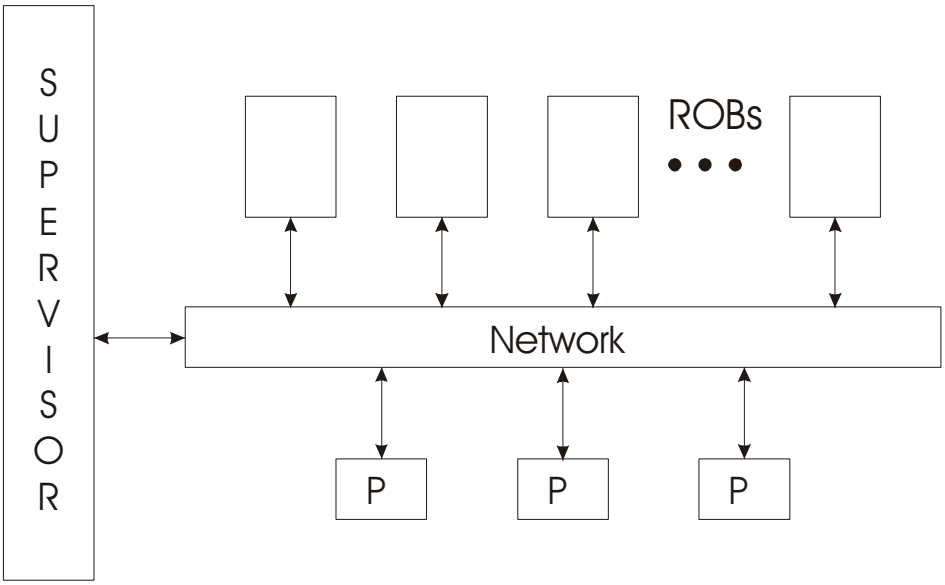


Abbildung 3.5: Architektur-C

Als Netzwerktechnologie wird hier ATM favorisiert. Es wurde ein Demonstrator mit 10 Knoten auf ATM-Basis aufgebaut und gestestet [ADE98].

3.3.5 Ein hybrider LVL2-Ansatz: Der FPGA-Prescaler

Neben dem reinen Architektur-A-Konzept wurde in Mannheim ein hybrides Konzept entwickelt und auf dem Argonne-Trigger / DAQ Meeting im Juni 1997 vorgestellt. Dieses Modell beinhaltet einen FPGA-basierten Prescaler, eine FPGA-basierte Berechnung des Full-Scan-TRTs und eine globale Prozessorfarm (Abbildung 3.6). Wichtig ist hierbei anzumerken, daß sowohl der FPGA-basierte Prescaler als auch die FPGA-unterstützte Ausführung des Full-Scan-TRT die gleiche Hardware benutzen (RoIC / FEX System) [KLL97].

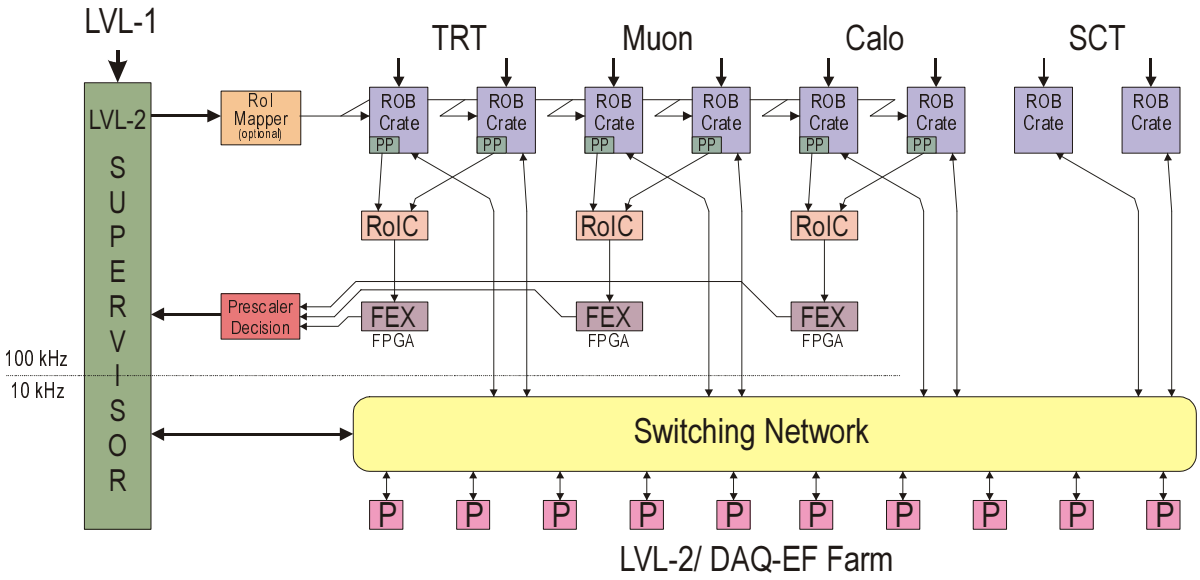


Abbildung 3.6: LVL2-Setup nach dem Prescalerkonzept

Ziel des **Prescalerkonzept**s ist es, durch das Vorschalten eines FPGA-Layers innerhalb des LVL2 eine Reduktion um den Faktor 10 zu erreichen und dadurch die Netzwerklast um 90% zu reduzieren. Damit ergeben sich sowohl Vorteile für den FPGA-Layer als auch für die nachfolgende Prozessorfarm. Für den FPGA-Layer bedeutet das: Die auszuführenden Algorithmen werden vereinfacht[19], die Zahl der zu berücksichtigenden Subdetektoren wird verringert und insgesamt resultiert daraus eine erhöhte Effizienz. Die Prozessorfarm profitiert von der erniedrigten Eventrate von nur noch 10 kHz, durch die reduzierte Netzwerklast und durch die verringerte integrale Rechenleistung - Reduktion der Anzahl der Rechenknoten ([KLL97],[SKK98]). Beide Einheiten - Prescaler und Prozessorfarm - arbeiten völlig unabhängig voneinander. Verbindendes und kontrollierendes Element ist der Supervisor.

Im Vergleich zur ursprünglichen Architektur-A benötigt der FPGA-Prescaler zusätzlich einen RoI-Mapper und eine Prescaler-Entscheidungseinheit. Diese

[19]Für den Full-Scan-TRT-Algorithmus bedeutet dies, daß die Ausführung des ersten Algorithmen-Schritts ausreichend effizient ist (vgl. Abschnitt 6.2).

funktionellen Komponenten lassen sich auf jeweils einem Enable++-System implementieren [SKK98]. Die Readout-Buffer müssen außerdem jeweils eine High-Speed-Schnittstelle zu den RoIC-Systemen zu Verfügung stellen. In [SKK98] ist eine detaillierte Beschreibung zu finden.

Für die B-Physik-Untersuchungen während des LHC-Betriebs bei geringer und mittlerer Luminosität wird der **Full-Scan des gesamten TRT Detektors** gefordert (siehe Kapitel 6). Im Rahmen der hybriden LVL2-Architektur soll der rechenintensivste Schritt dieses Algorithmus' auf dem FPGA-Prozessor Enable++ ausgeführt werden. Der mit dem Full-Scan-TRT beauftragte Farmrechner wird in diesem Szenario durch den FPGA-Prozessor als Koprozessor entlastet.

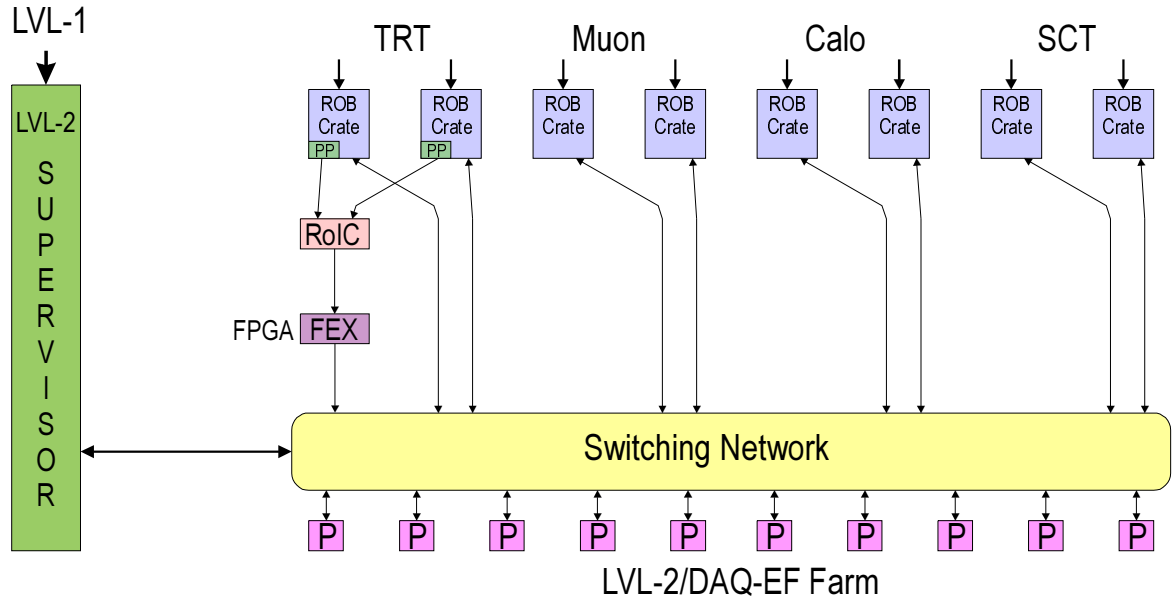


Abbildung 3.7: B-Physik Setup: FPGA-Systeme als Koprozessoren

In Abbildung 3.7 ist die Verwendung eines FPGA-Prozessors als Koprozessor dargestellt. Die Eventdaten, von den TRT-ROBs kommend, werden von den RoICs zusammengefaßt und an den FPGA-basierten FEX-Prozessor weitergeleitet. Die FEX-Ergebnisse werden über das Netzwerk dem zuständigen Farmprozessor zugeführt.

Eine alternative Möglichkeit dazu, wie der FPGA-basierte FEX-Prozessor als Koprozessor eingesetzt werden kann, ist diesen direkt mit einem Farmprozessor zu koppeln. Dabei bekommt der FEX-Prozessor nicht direkt aus den ROBs die Eventdaten zugeführt, sondern erhält diese von einem Farmprozessor.

Da dieses Szenario mit ein Ausgangspunkt für die Entwicklung des neuen hybriden FPGA/CPU-basierten Prozessors ATLANTIS ist, wird diese Konstellation im nachfolgenden Abschnitt separat diskutiert.

3.3.6 Integration von Architektur-A und Architektur-C

Im November 1997 fand der erste Integrationsversuch der konkurrierenden Architekturen A und C in Saclay / Paris statt. Ziel war es, die beiden Ansätze zu verknüpfen, wie es im hybriden Modell für die Ausführung des Full-Scan-TRTs vorgesehen ist. Die Verknüpfung umfaßt neben der essentiellen Hardware-Verbindung auch die wichtige Integration der Softwareumgebungen.

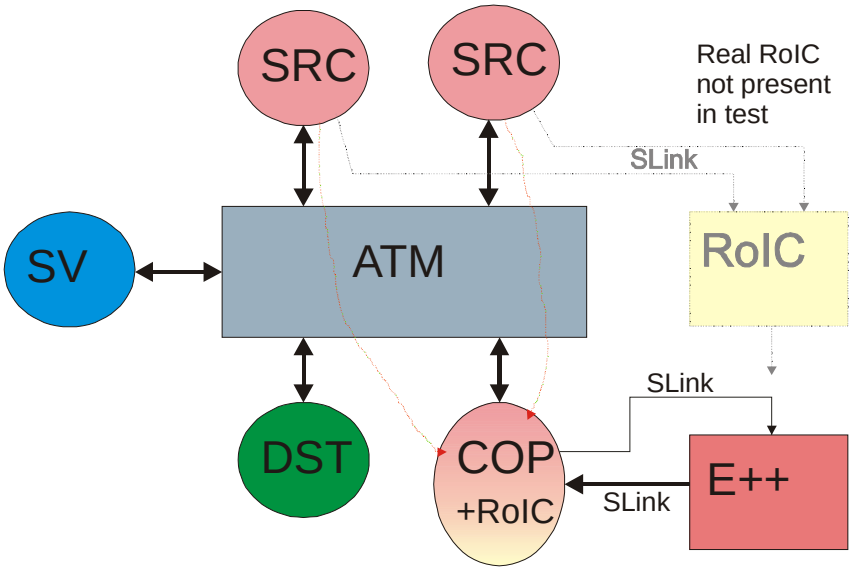


Abbildung 3.8: Integration von Demonstrator-A und Demonstrator-C

Die Verbindung zwischen Farmprozessor und Enable++ wird über die S-Links realisiert. Der vollständige TRT-Eventdatensatz wird vom Farmprozessor (COP) an das Enable++-System (E++) übermittelt. Nach Ausführung des Full-Scan-TRT-Algorithmus' werden die Daten der gefundenen Spuren an den Farmprozessor übertragen, wo weitere Algorithmenschritte ausgeführt werden. In Abbildung 3.8 ist die Situation skizziert.

Die Integration wurde wie beabsichtigt erfolgreich durchgeführt. Die Softwareumgebungen konnten zusammengeführt werden. Ein stabiler Lauf mit 1kHz Eventrate und einer Latency von maximal 800ms wurde nachgewiesen . Die relative große Latency wurde dabei durch den Farmprozessor verursacht [KBC98].

In Abbildung 3.8 ist zusätzlich ein RoIC-System eingezeichnet, das sowohl eine Verbindung zum ROB- als auch zum Enable++-System aufweist. Nicht ohne Grund ist diese Variante, in der die Daten direkt aus den ROBs (hier SRC) über das RoIC System in das Enable++-System geleitet werden, in der Abbildung nur schwach gezeichnet. Dies basiert einerseits auf der wachsenden Zuversicht gegen Ende des Demonstrator-Programms, daß in absehbarer Zeit leistungsfähige Netzwerktechnologien verfügbar sein werden, die die Anforderungen des LVL2 erfüllen. Damit wäre eine zweite, asymmetrische Verbindung und der damit verbundene Aufwand nicht gerechtfertigt.

Andererseits wird dadurch eine politische Entscheidung ausgedrückt: Die Förderung der Jenaer ATLAS-Arbeitsgruppe durch das BMBF wird seit Sommer 1998 nicht mehr fortgesetzt. Die Weiterentwicklung der Jenaer RoIC-Komponente ist somit nicht mehr gegeben. Untersuchungen für die Abbildung der RoIC Funktionalität auf das Enable++-System wurden vom Autor durchgeführt. Auf Grund der weiteren Entwicklung des LVL2 wurde dieses Ziel aber nicht weiter verfolgt. Die vorläufigen Untersuchungsergebnisse bilden aber eine wichtige Grundlage für die Implementierung der ROB-Funktionalität auf das ATLANTIS-System (Kapitel 7).

3.3.7 Ergebnisse des Demonstrator-Programms

Die Aktivitäten der ATLAS LVL2-Kollaboration im Rahmen des Demonstrator-Programms sind insgesamt im *Technical Progress Report* niedergeschrieben. Die wichtigsten Schlußfolgerungen sind nachfolgend aufgelistet [ADE98]:

- **Netztwerktechnologien**

  Mit großer Zuversicht wird erwartet, daß zukünftige Netzwerktechnologien den LVL2-Anforderungen gewachsen sind. Geeignetste Kandidaten sind zur Zeit ATM und SCI.

- **Prozessortechnologien**

  Der Einsatz von Standard RISC / CISC-Prozessoren ist grundsätzlich die erste Wahl, weil sie am einfachsten zu handhaben und sehr flexibel sind. FPGA-Prozessoren haben ihren vorteilhaften Einsatz bei der Ausführung von rechenintensiven Algorithmen wie dem Full-Scan TRT oder den Vorverarbeitungsalgorithmen gezeigt. Die parallele FEX mit FPGA-Prozessoren wurde demonstriert. Diese Architektur soll aber, wegen des deutlich höheren Aufwands, nur dann angewandt werden, wenn mit herkömmlichen Technologien das Problem nicht gelöst werden kann oder andere Komplikationen auftreten[ADE98].

- **Architektur**

  Eine sequentielle Vorgehensweise bei der Eventanalyse wird favorisiert, weil dadurch die Netzwerkdatenbandbreite und die Last auf die Farmprozessoren reduziert werden kann. Es soll nur ein Netzwerk sowohl für Evendatenpakete als auch für Kontrollnachrichten benutzt werden. Der LVL2-Supervisor gibt die gesamte Kontrolle über die Analyse eines Events an einen Farmprozessor ab.

Mit diesen Ergebnissen und Erkenntnissen wurden bereits in der letzten Phase des Demonstrator-Programms die Weichen für die kommenden Aktivitäten der

Mannheimer Arbeitsgruppe im Rahmen des nachfolgenden Pilot-Projekts gestellt. Zwei Schwerpunkte bilden die Hauptlinie der Forschungsarbeit:

1. Konzentration auf den Einsatz von FPGA-Prozessoren als Koprozessoren für die Ausführung von zeitkritischen Algorithmen wie den Full-Scan-TRT

2. Untersuchung von FPGA-Prozessoren als Readout-Buffer-Systeme (ROB)

Das Demonstrator Programm hat insgesamt gezeigt, daß es nicht möglich war, die Architektur und die Technologien getrennt voneinander zu behandeln. Für die nächste Projektphase wurde deshalb vorgeschlagen, nur einige wenige Technologien zu untersuchen und die optimale Architektur darauf aufbauend zu entwickeln.

3.4 Das Pilot-Projekt

Das Ziel des Pilot-Projekts ist es, eine gültige LVL2-Architektur zu entwickeln und dabei die Verwendung geeigneter Technologien für die Implementierung zu untersuchen. Um die Gültigkeit einer Architektur nachzuweisen, muß notwendigerweise mindestens eine Implementierung aufgebaut werden, die die geforderte Leistungsfähigkeit demonstriert und gleichzeitig skalierbar und erschwinglich ist [ATD98].

Basierend auf den Ergebnissen des Demonstrator-Programms geht man mittlerweile davon aus, daß sich der LVL2-Trigger tatsächlich mit kommerziellen Netzwerken[20] und Standard-PCs realisieren läßt[21]. Daraus hat sich eine Architektur abgeleitet, wie sie im nächsten Abschnitt dargestellt ist. Außerdem wurde die Art und Anzahl der in Frage kommenden Komponenten (Hardware und Software) für verschiedene Architekturteile eingeschränkt. Natürlich muß weiterhin Forschungsarbeit, gerade für Schlüsselstellen des LVL2 geleistet werden. Verstärkt muß zusätzlich die Integration des LVL2-Triggers in das gesamte Triggersystem berücksichtigt werden.

[20]Hierbei ist anzumerken, daß zwar kommerzielle Hardware benutzt wird, die in Software geschriebenen Netzwerktreiber aber durch eigene, optimierte ersetzt werden. Muß dadurch bei einem Hardware-Upgrade zu einer leistungsfähigeren Version der Treiberlayer angepaßt werden, ist der Arbeitsaufwand äquivalent zu Hardwareentwicklung und macht den Vorteil des Einsatzes kommerzieller Produkte zunichte.

[21]Zur Ausführung zeitkritischer Algorithmen, wie zum Beispiel dem Full-Scan-TRT, ist der Einsatz von FPGA-Prozessoren weiterhin gefordert.

Unter diesen Voraussetzungen wurde die Arbeit des Pilot-Projekts in drei Hauptgebiete aufgeteilt [ATD98]:

- **Funktionale Komponenten**:
  Supervisor, RoI Builder, ROB-Complex[22], Netzwerke und Prozessoren (speziell FPGA-Prozessoren).

- **Testbeds**:
  Entwicklung der LVL2-Referenz-Software, prototypische Implementierung des gesamten LVL2-Ablaufs und Aufbau von *Application Testbeds* (LVL2-Prototypensysteme) zur Anwendung und zum Test der Software und der funktionalen Komponenten.

- **Systemdesign**:
  Modelling Aktivitäten und Arbeiten zur Integration des LVL2 in das Gesamt-Triggersystem.

3.4.1 Die Hardware-Architektur des Pilot-Projekts

Die Aktivitäten, die im Rahmen des Pilot-Projekts durchgeführt wurden, basieren auf einer gemeinsamen Hardware-Architektur (Abbildung 3.9). Eine wesentliche Änderung im Vergleich zum Beginn der Projektphase hat sich im Zusammenhang mit den FPGA-Prozessoren ergeben. Die FPGA-Prozessoren sind nicht mehr als separate Knoten mit dem Netzwerk verbunden, sondern existieren als Bestandteile der Farmprozessoren in der Funktionalität als Koprozessor. Mit dem ATLANTIS-System wurde im Rahmen der hier vorliegenden Promotion genau ein solches Farmprozessorsystem realisiert (Abschnitt 3.4.3, Kapitel 4 und 5). Nach wie vor wird der Einsatz von kommerziellen Komponenten (Prozessoren, Betriebssysteme, Netzwerktechnologien, etc.) an möglichst vielen Stellen innerhalb der Architektur stark befürwortet. In der nachfolgenden Abbildung 3.9 ist der Aufbau des LVL2-Architektur für das Pilot-Projekt dargestellt:

[22]Der ROB-Complex, wie er in der Hardware-Architektur dargestellt ist, umfaßt mehrere *Input Buffer*, die von einer lokalen Instanz kontrolliert werden und über eine gemeinsame Schnittstelle zum Netzwerk verfügen.

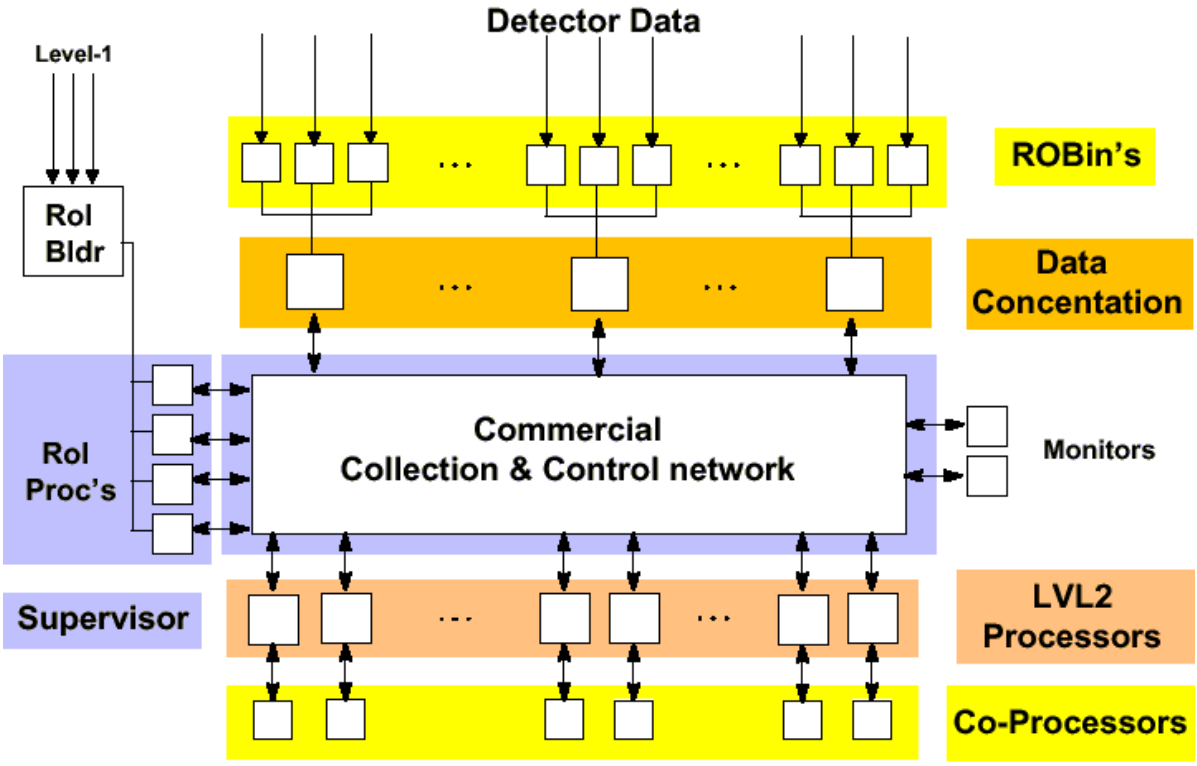


Abbildung 3.9: Die Pilot-Projekt Hardware-Architektur

Vom LVL1-Trigger gelangen die akzeptierten Ereignisse zum RoI-Builder, der in Echtzeit die Parameter der zu untersuchenden RoIs für den LVL2 ermittelt und zu den RoI-Prozessoren schickt. Vom Supervisor wird für jedes Ereignis ein LVL2-Prozessor ausgewählt, der dann von den RoI-Prozessoren alle Informationen zur Ereignisbearbeitung über das Netzwerk erhält.

Parallel dazu werden die Ereignisdaten von den Subdetektoren über Readout-Links an die RobIn's der Readout-Buffer (ROBs) übertragen und dort zwischengespeichert. Die Weitergabe der Daten an die LVL2-FEX-Prozessoren erfolgt nur auf Anfrage des Prozessors selbst. Jeder Prozessor bearbeitet ein Ereignis vollständig, indem er die FEX ausführt und anschließend die globale Triggerentscheidung fällt. Gemäß dem Prinzip des Sequential-Selection [BCE96] werden dazu nacheinander Daten von den Readout-Buffern verschiedener Subdetektoren angefordert und auf der CPU des LVL2-Prozessors selbst oder auf dem angeschlossenen Koprozessor bearbeitet.

Akzeptierte Ereignisse werden in einem speziellen Buffer für die nächste Triggerstufe (Event Filter) abgelegt. Der Abschluß einer Ereignisbearbeitung wird grundsätzlich dem Supervisor mitgeteilt, der bei einer negativen Entscheidung die RobIn's zum Löschen der verworfenen Ereignissen veranlaßt.

Die Aktivitäten innerhalb der Pilot-Projekt-Phase sind ausführlich im *ATLAS HLT DAQ und DCS Technical Proposal* (AHL00) dargestellt und werden zusätzlich durch eine Reihe von Backup-Dokumenten beschrieben. Eine kurze Zusammenfassung der Ergebnisse und die daraus gezogenen Schlüsse für die endgültige ATLAS LVL2-Trigger-Architektur wird im Abschnitt 3.4.5 gegeben.

In den folgenden beiden Abschnitten werden die für die vorliegende Arbeit interessierenden Untersuchungen im Rahmen der Pilot-Projekts diskutiert: Das ist der Einsatz von FPGAs als Koprozessoren für Standard-CPUs und im Bereich der Readout-Buffer.

3.4.2 Der FPGA/CPU-Prozessor ATLANTIS als Farmprozessor

FPGA-Prozessoren haben vor der Entwicklung des ATLANTIS-Systems ihr großes Potential für die Ausführung von ATLAS LVL2-Triggeralgorithmen bereits mehrfach bewiesen ([BBB93], [KMo94], [LLM95]). Mit verschiedenen FPGA-Prozessoren, zuletzt mit dem Enable++-System im Rahmen des Demonstrator-Programms, konnte dies in der Praxis gezeigt werden. Für den endgültige Triggerarchitektur hat sich mittlerweile herauskristalliert, daß FPGA-Prozessoren grundsätzlich eingesetzt werden könnten, daß aber das vorausgesetzte Expertenwissen bei der Handhabung und der zusätzliche Aufwand bei der Integration, den Einsatz als stand-alone Systeme nicht mehr rechtfertigen. Im finalen LVL2-Trigger sollen die FPGAs deshalb nicht stand-alone, sondern als Koprozessoren die kommerziellen Farmprozessoren unterstützen. Und das auf möglichst effiziente und transparente Weise.

Auf der Basis dieser Forderung wurde das ATLANTIS-System als hybrider FPGA/CPU-Prozessor konzipiert (Kapitel 4). Das ATLANTIS-System kann sowohl als stand-alone FPGA-Prozessor als auch als FPGA-basiertes Koprozessorsystem benutzt werden. Durch die Verwendung von CompactPCI als Basis-Bussystem lassen sich kommerziell erhältliche CPU-Karten integrieren. Daneben sind auch kommerzielle Netzwerkkarten, Testsysteme und weitere funktionale Komponenten im CompactPCI-Format erhältlich. Die Konzeption des Systems wird ausführlich in Kapitel 4 dargestellt. Im anschließenden Kapitel 5 folgt dann die detaillierte Beschreibung des Systems.

Im Sinne des Koprozessorkonzept werden auf die FPGA-basierten Baugruppen über die CompactPCI-Bus-Schnittstelle Berechnungen ausgelagert, die entsprechend beschleunigt ausgeführt werden. Durch die Implementierung eines zweiten, unabhängigen Bussystems (Privat-Bus) kann die Rechenleistung des Koprozessors fast beliebig skaliert werden, ohne den PCI-Bus zu belasten. Dazu werden die notwendige Anzahl von FPGA-Baugruppen in einem CompactPCI-Crate parallel betrieben.

Neben der Entwicklung einer geeigneten Hardwarearchitektur, die die Anforderungen für das Koprozessorkonzept optimal unterstützt, spielt eine einfach benutzbare Betriebs- und Programmiersoftware für ein solches spezielles System

eine große Rolle (Akzeptanzerhöhung). Aus diesem Grund wurde gleichzeitig mit der Hardwareentwicklung, die Realisierung einer ATLANTIS Betriebssystemsoftware (AOS) mit höchster Priorität verfolgt. Ein weiteres wichtiges Software-Feature sind die Programmiermöglichkeiten für ein solches System. Mit der Adaption der sogenannten VHDL-Bibliothek wurde für das ATLANTIS-System eine komfortable und einfach zu bedienende Entwicklungsumgebung geschaffen. Die in Mannheim entwickelte C++-basierte Hardwarebeschreibungssprache CHDL, zur Programmierung von FPGA(-Ko)-Prozessoren, wurde parallel dazu ebenfalls an das ATLANTIS-System angepaßt (Abschnitt 5.6).

Die im Rahmen der vorliegenden Arbeit durchgeführten Konzeption und Implementierung der ATLANTIS-Systems wird ausführlich in den beiden folgenden Kapiteln 4 und 5 dargestellt. Die Implementierung und Benchmark des vollständigen Full-TRT-Scan-Algorithmus' als Koprozessoranwendung - ebenfalls Teil dieser Promotion - wird im Kapitel 6 beschrieben.

Als Ergebnis kann bereits an dieser Stelle vorweggenommen werden, daß mit dem ATLANTIS-System das Koprozessorkonzept, wie es für den finalen ATLAS LVL2-Trigger gefordert wird, erfolgreich demonstriert werden konnte. Der Full-Scan-TRT-Algorithmus wurde auf ATLANTIS implementiert und ausgeführt und erreicht prinzipiell die gleiche Effizienz wie die reine CPU-Version des Algorithmus'. Dabei wurden die Algorithmenschritte auf dem dafür geeigneten Teil des ATLANTIS-Systems (FPGA oder CPU) implementiert. Es wurde ein Speedup von 5.6 (bezogen auf alle Algorithmenschritte) gegenüber der reinen CPU-Implementierung erreicht [SHK00]. Die Aussichten, daß und wie dieser Leistungsvorsprung durch eine FPGA-Koprozessorarchitektur noch weiter ausgebaut werden kann, werden in einem separaten Dokument diskutiert [HKL00]. Zusätzlich konnte aber auch der wichtige Integrationsschritt des Systems in die LVL2-Softwareumgebung (Referenz-Software) am CERN durchgeführt werden. Dazu wurde das ATLANTIS-System als Netzwerkknoten im ATM/Fast-Ethernet-Testbed (ATLAS LVL2-Trigger Prototypensystem am CERN) betrieben. Eine detaillierte Darstellung dieser Aktivitäten ist in Kapitel 6 zu finden.

3.4.3 Der ROB-Complex mit FPGAs

Der zweite Bereich, für den der Einsatz von FPGAs für den ATLAS-Trigger untersucht wird, sind die Readout-Buffer (ROB). In den letzten 2 Jahren wurden von 4 Gruppen der LVL2–Kollaboration (neben Mannheim auch NIKHEF, UCL, SACLAY) Prototypen für die Input-Module der ROBs aufgebaut. Weiterhin gibt es die Active-Rob-Complex ArobC-Aktivität von CERN und MSU, bei der ein Multiprozessor mit mehreren PCI-Bussen als Rob-Complex evaluiert wird. Eine ausführlichere Beschreibung der ROB-Untersuchungen in Zusammenhang mit dem ATLANTIS-System finden sich in Abschnitt 7.1.

Die grundsätzliche Aufgabe der ROBs ist es, die Ereignisdaten von den Readout-Drivern (RODs) der einzelnen Detektoren entgegenzunehmen, auf Anfrage die Eventdaten über ein Netzwerk an das Triggersystem (LVL2-Farm und Event Filter-Farm) zu übermitteln und bis zur endgültigen LVL2-Entscheidung zwischen zu speichern. Die Eckwerte, die 1998 im TPR [ADE98] definiert wurden, sind 100MB/s Eingangsdatenrate bei bis zu 100kHz Eingangsfrequenz, Ausgangsdatenrate bis ca. 15MB/s, Speicherdauer mehrere Millisekunden. Die Gesamtzahl der benötigten ROBs beläuft sich auf etwa 1.700. Die gesamte Funktionalität eines ROBs ist in drei Module gegliedert:

- Der RobIn empfängt die Daten über einen optischen/elektrischen Link vom ROD und übernimmt die Speicherung und Verwaltung der Eventfragmente.
- Der RobOut ist das Bindeglied zum LVL2- und Event filter-Netzwerk.
- Der RobController bearbeitet die Anfragen des Triggersystems und steuert die Module RobIn und RobOut.

Gegebenenfalls kann in jeder der drei Stufen noch eine Vorverarbeitung der Daten vorgenommen werden. Die allgemeinere Form eines ROBs, bei der die Anzahl von RobIn- und RobOut-Modulen jeweils auch größer als 1 sein darf, wird als Rob-Complex[23] bezeichnet. In Abbildung 3.9 sind RobController und RobOut zur Funktionsgruppe Data-Concentration zusammengefaßt.

Die Aktivitäten der Mannheimer ROB-Arbeitsgruppe[24] konzentrierten sich innerhalb des Pilot-Projekts auf folgende Gebiete:

- **Aufbau eines RobIn-Prototypen**

  Es konnte bereits während der Demonstrator-Programm-Projektphase durch die Verwendung des kommerziellen FPGA-Koprozessors microEnable demonstriert werden, daß die geforderte Leistung auch mit einfacher, nicht spezialisierter Hardware erreichbar ist (Abschnitt 7.1.1). Auf der Basis dieser

[23] In der Terminologie der DAQ/EF Gruppen entspricht dies einem Readout-Crate.

[24] Die Arbeitsgruppe besteht aus Matthias Müller, Andreas Kugel und dem Autor.

Untersuchungen wurde mit dem Beginn des Pilot-Projekts die Realisierung eines vollständigen ROB-Complex' auf dem ATLANTIS-System begonnen[25] (Abschnitt 7.1.2).

- **Entwicklung einer Punkt-zu-Punkt–Verbindung für ROB-Complex-Testaufbauten**

  Die Verbindung zwischen den ROBs und den Detektoren (RODs) sind innerhalb von ATLAS standardisiert und erfolgen gegenwärtig über S-Link-Schnittstellen[26]. Da die S-Link–Spezifikation aus mechanischen Gründen keine sehr hohe Baugruppendichte erlaubt und die kommerziellen Implementierungen relativ teuer sind, wurde in Mannheim ein sogenannter M-Link[27] entwickelt. Damit können von einem ATLANTIS I/O-Board vier dieser Schnittstellen-Module verwaltet werden - anstelle von nur zwei möglichen, klassischen S-Link-Modulen (Abschnitt 7.1.3).

- **Mitarbeit beim ARobC-Projekt**

  Die microEnable-RobIn-Module, die im Rahmen von ArobC eingesetzt werden, sind weitgehend softwarekompatibel zur ATLANTIS-Hardware. Daher ergeben sich eine Reihe vielversprechender Synergieeffekte zwischen ArobC und dem geplanten Rob-Complex auf ATLANTIS. Bisher wurden von der Mannheimer Gruppe[28] folgende Arbeiten durchgeführt: Spezifikation eines kompatiblen API für ATLANTIS und ArobC (zusammen mit CERN) und die Implementierung der low-level Support- und Interface-Software für WinNT. Desweiteren wurden die grundlegenden RobIn-FPGA-Designs (Firmware) erstellt

3.4.4 Das Pilot-Projekt: Ergebnisse und Ausblick

Zusammengefaßt können folgende Schlußfolgerungen zum Pilot-Projekt gezogen werden:

Die Referenz-Software wurde in verschiedenen Konfigurationen gestestet. Gezeigt wurde, daß auf verschiedenen Netzwerken mit bis zu hundert Knoten die geforderte Leistungsfähigkeit mit kommerzieller Hardware (PCs) und herkömmlichen Betriebssystemen (PC/Linux) erreicht werden kann.

[25] Für die Implementierung der ROB-Funktionaliät dient das ATLANTIS I/O-Board als Basis-Hardware-Plattform. Ein detaillierter Überblick zum ATLANTIS ROB-Complex und eine Darstellung der vorläufigen Ergebnisse sind in Kapitel 7 gegeben.

[26] Die S-Link–Spezifikation (http://www.cern.ch/HSI/s-link) definiert die Mechanik, den Stecker und das Protokoll für eine Punkt-zu-Punkt Verbindung. Die Art der physikalischen Verbindung ist hingegen frei wählbar. Die verbreitetsten Typen sind eine optische Fiber-Channel Implementierung und eine elektrische Implementierung mit parallelen LVDS-Signalen (siehe Abschnitt 7.1.3).

[27] Die M-Link Module sind als Steckkarte für das I/O-Board des ATLANTIS-Systems vorgesehen und werden im Abschnitt 7.1.3 näher beschrieben.

[28] Der Mannheimer Projektbeitrag wird überwiegend von Andreas Kugel geleistet.

Bei den Tests innerhalb des am CERN installierten Testbeds (ATLAS LVL2-Triggerprototypensystem) bildeten ATM, Fast / Gigabit-Ethernet und SCI mit bis zu 48 Netzwerkknoten die Basis-Netzwerktechnologien. Optimierte Hardwarekomponenten wie der Supervisor, der RoI-Builder, das ATLANTIS-System und eine ROB-Complex-Architektur wurden in die Testbeds integriert.

Mit den Untersuchungen zum ROB konnte gezeigt werden, daß mit aktueller Technologie der Aufbau einer ROB-Architektur mit der notwendigen Leistungsfähigkeit bezüglich Datenraten und Datenbandbreiten prinzipiell möglich ist. Die Arbeiten im Rahmen des Pilot-Projekts haben eine Liste mit wichtigen Punkten für die zukünftige ROB-Design-Arbeit hervorgebracht: Erstellung einer detaillierten Anforderungsliste (beinhaltet die UML Beschreibung), Definition der Datenformate, Laufzeitkontrollzustände, Fehlerfälle, Fehlerbehandlung und Timouts, Zustands- und Statistikberichte, Durchführung von realistischen Implementierungen und Messungen.

Ein RoI-Builder-Prototyp wurde entwickelt und zusammen mit einem Supervisor in die ATM- und Ethernet-Testbeds integriert. Die Funktionalität bei der LVL2-Eventrate von 100 kHz konnte dabei nachgewiesen werden. Die Farmprozessoren innerhalb der Testbeds haben unter Verwendung der Referenz-Software gezeigt, daß das Datensammeln für ein RoI eines einzelnen Detektors mit akzeptabler Rate möglich ist. Dabei wurde eine dreistufige Sequential-Selektion-Strategie angewandt und prototypische Algorithmen ausgeführt, wobei von den Prozessoren simulierte Eventdaten von mehreren ROB-Emulatorsystemen angefordert wurden. Diese Arbeiten wurde von der Mannheimer Gruppe indirekt angetrieben und unterstützt, weil diese die einzige Gruppe war, die im Rahmen der Messungen im CERN-Testbed realitätsnah Algorithmen (Full-Scan-TRT) testen wollte (vgl. Abschnitt 6.5.3).

Ausführlich wurden drei Netzwerktechnologien untersucht: ATM, Fast / Gigabit Ethernet und SCI. Dabei haben sich die beiden ersten, kommerziell verfügbaren als die Favoriten herauskristallisiert. Eine endgültige Entscheidung für eine der beiden Technologien wird zum Ende des Pilot-Projekts nicht getroffen[29].

Das für die hier vorliegende Arbeit wichtigste Ergebnis betrifft die Ausführung des TRT-Full-Scan-Algorithmus auf dem ATLANTIS-System. Es wurde eine 5.6-fach schnellere Bearbeitung gegenüber der reinen CPU-Ausführung nachgewiesen. Diese Arbeit hat insgesamt gezeigt, daß mit dem ATLANTIS-System eine transparente Kopplung von leistungsfähigen FPGA-Modulen mit Standard-CPUs für die Ausführung von ATLAS-Triggeralgorithmen erreicht werden kann. Dabei zeigt sich eine deutliche Leistungssteigerung gegenüber der Ausführung auf Standard-CPUs, infolgedessen eine drastische Reduzierung der Größe der favorisierten LVL2-Trigger-Prozessorfarm ermöglicht wird.

Das Aufstellen von Papiermodellen ist die zentrale Methode, um ausgehend von den Untersuchungen und Messungen von Teilsystemen auf das finale System hoch

[29]Die Untersuchungen zur ATM-Netztwerktechnologie wurde von der Pariser Saclay-Gruppe durchgeführt. Im Juni 2000 hat diese Gruppe das Ende der Arbeit in der ATLAS Kollaboration bekannt gegeben. Fraglich ist deshalb, ob, inwiefern und von wem die ATM Tests weitergeführt werden.

zu skalieren. Die Ergebnisse des Pilot-Projekt ergeben folgende Zahlen für das endgültige LVL2-Triggersystem: Die high pt-LVL2-Trigger können bei allen Luminositäten mit ca. 100 bis 200 1000-MIPS-Prozessoren ausgeführt werden. Das maximale Datenvolumen durch das Netzwerk ist 20-40 GBit/s bei geringer Luminosität (inklusive des TRT-Full-Scans für die B-Physik) [AHL00]. Der Full-Scan des Innendetektors erhöht die Rechenleistung und die Datenbandbreiten signifikant. Die Abschätzungen zum Ende des Pilot-Projekts beziffern die benötigte Anzahl von Farmprozessoren (1000MIPS) mit ~770 [AHL00]. Durch den Einsatz von FPGA-Koprozessoren kann die Anzahl auf ~450 reduziert werden! Berücksichtigt man die Abschätzungen basierend auf neueren, angekündigten FPGA-Architekturen, wird die Zahl an benötigten hybriden CPU/FPGA-Systemen als Farmprozessoren nochmals auf nur noch 194 reduziert [HKL00]! Damit wird für die ATLAS LVL2-Prozessorfarm eine Systemgröße in Aussicht gestellt, die, unter den Randbedingungen der ATLAS LVL2-Triggeraufgabe, die Umsetzung des Farmkonzepts realisierbar erscheinen läßt.

An erster Stelle für die zukünftige globale LVL2-Projektarbeit steht die Integration des LVL2-Systems in das globale Triggersystem. Entscheidungen für Komponenten und Technologien haben eine große Überlappung mit denen von DAQ/EF (Standard-PCs als Prozessoren und ATM/Ethernet als Netzwerktechnologien).

Der zukünftige Mannheimer Beitrag wird es sein, den geeigneten FPGA-Koprozessor mit weiteren Algorithmenuntersuchungen zu skizzieren. Aufgrund der Entwicklung des FPGA-Marktes ist es denkbar, daß in den kommenden Jahren ein entsprechend leistungsfähiges System kommerziell zu Verfügung steht. Bezüglich der ROBs steht noch ein Großteil der Forschungsarbeit bevor. Es kann aber als sehr sicher angesehen werden, daß in das finale ROB-System FPGAs integriert sein werden, unabhängig von welcher Arbeitsgruppe der ROB letzt endlich umgesetzt wird. Daher wird die FPGA-Erfahrung Mannheims auch in diesem Sektor die Entwicklung des endgültigen Designs positiv beeinflussen.

KAPITEL 4

Die Konzeption des ATLANTIS-Systems

Das hybride FPGA / CPU-basierte Rechnersystem ATLANTIS soll einerseits als FEX-Prozessor für das ATLAS LVL2-Triggersystem eingesetzt werden. Daraus ergeben sich grundsätzliche, notwendige Randbedingungen für die Konzeption des Systems. Wie im vorangegangen Kapitel bereits diskutiert wurde, steht dabei die Demonstration des Koprozessorkonzepts für die Ausführung zeitkritischer LVL2-Triggeralgorithmen im Vordergrund. Andererseits soll mit ATLANTIS die Eignung eines FPGA-basierten Systems als ROB-Complex untersucht werden. Mit den beiden Mannheimer stand-alone FPGA-Prozessoren Enable und Enable++ und den damit gemachten Erfahrungen werden für die Neuentwicklung grundsätzlich wichtige Designkriterien in Bezug auf den FPGA-Teil des ATLANTIS-Systems vorgegeben. Auf Grund der schnelllebigen Elektronikbranche ist die Evaluierung aktueller FPGA-Architekturen unumgänglich. Für die Umsetzung der engen Kopplung zwischen FPGA und CPU, was eine essentielle Eigenschaft des Systems darstellt, muß die geeignetste Busarchitektur gefunden werden. Auch spielt die Wahl des Formfaktors für die Gesamtsystemarchitektur ein große Rolle. Eine flexible Adaption an eine spezifische Anwendung ist eng mit der Forderung nach Skalierbarkeit und Modularität verbunden. Prinzipiell ist das ATLANTIS-System als universeller FPGA / CPU Prozessor geplant. Deshalb muß der Einfluß von Applikationen auch außerhalb von ATLAS mitberücksichtigt werden. Es existieren konkret geplante Anwendungen aus den Bereichen 2D / 3D-Bildverarbeitung, der Astronomie und Triggeranwendungen bei der GSI / Darmstadt.

Beginnend mit einer Einführung zur neuartigen Rechnerarchitektur der FPGA-Prozessoren werden in diesem Kapitel die Hauptkriterien und Entscheidungen für die Konzeption des ATLANTIS-Systems diskutiert. Ausgangspunkt sind die Anforderungen für die oben genannten Anwendungsgebiete. Eine Auflistung der gestellten Anforderungen und der daraus abgeleiteten Eigenschaften für ein hybrides Prozessorsystem machen, bei Berücksichtigung der großen Zahl an verfügbaren FPGA-basierten Architekturen, eine Neuentwicklung notwendig. Wichtige Entscheidungen betreffen dabei die Wahl des FPGA-Typs, die Integration von Speicherbausteinen, die Definition der Schnittstellen zu externen Systemen und die Wahl der

Verbindungstopologie der internen funktionalen Einheiten. Eng verknüpft ist die Entscheidung für ein Basis-Bussystem mit der Gesamtsystemarchitektur und der Aufteilung des Systems nach funktionalen Baugruppen.

4.1 FPGA-Prozessoren

Mit dem Begriff FPGA-Prozessor ist ganz allgemein eine Baugruppe oder ein System von Baugruppen gemeint, das aus wenigstens einem FPGA besteht und typischerweise Speicherbausteine und I/O-Schnittstellen beinhaltet. Mit einem FPGA-Prozessor lassen sich, je nach Architektur, eine spezielle Anwendung oder eine Klasse von Anwendungen beschleunigt ausführen. Dabei können gegenüber Standardprozessoren Geschwindigkeitssteigerungen von einem Faktor 10 bis 1000 erreicht werden.

Die klassische von-Neumann-Rechnerarchitektur besteht aus einer CPU, Speicher, der Peripherie und einem Bussystem, das die Komponenten miteinander verbindet. Man kann dies auch als einen seriellen Universalrechner bezeichnen. Im Gegensatz dazu ist beim grundsätzlichen Aufbau eines FPGA-Prozessors die CPU durch einen FPGA ersetzt.

Bei der herkömmlichen Rechnerarchitektur wird ein Programm wie folgt abgearbeitet: Während der Programmausführung werden in der CPU vom Steuerwerk aus dem Speicher die Programmbefehle geladen, dekodiert und entsprechende, festverdrahtete Mikroprogramme ausgeführt. Dazu werden Daten aus dem Hauptspeicher in CPU-interne Register geladen, Operationen darauf ausgeführt und die Ergebnisdaten wieder zurück in den Hauptspeicher geschrieben. Ein Programm wird ganz allgemein gesprochen sequentiell ausgeführt.

Der FPGA-Prozessor besitzt durch die Ersetzung der CPU durch einen FPGA eine gänzlich neue Rechnerarchitektur. Operationen werden direkt auf die vom FPGA zur Verfügung gestellten Logikresourcen abgebildet. Im Gegensatz zu herkömmlichen CPUs enthalten FPGAs keine vordefinierten Rechenwerke oder Register. Erst durch die Programmierung der FPGAs werden aus den Grundelementen der FPGA-Logik vergleichbare Einheiten gebildet, wie sie denen der Rechenwerke der Standard-CPUs entsprechen. Dies geschieht sehr flexibel und für jede Anwendung individuell. Die SRAM-basierten FPGAs lassen sie sich beliebig oft umprogrammieren. Innerhalb von Millisekunden kann der FPGA-Prozessor vollständig seine Struktur ändern.

Typischerweise werden FPGA-Prozessoren mit einer maximalen Systemfrequenz von 20 bis 80 MHz getaktet. Das ist eine Größenordnung kleiner wie die zur Zeit schnellsten CPUs von Intel und AMD. Dennoch erreichen sie bei bestimmten Anwendungen Leistungssteigerungen von einem Faktor 10 bis 1000 gegenüber Standard-CPUs. Die Ursache dafür, ist die Möglichkeit Anwendungen in FPGAs massiv parallel und in tiefen Pipelines ausführen zu können. Die klassische CPU arbeitet immer nur einen Befehl gleichzeitig ab.[30] Ein FPGA kann dagegen durch die

[30]Innerhalb superskalarer Architekturen können gleichzeitig mehrere Befehle dekodiert und ausgeführt werden.

Programmierung eine sehr komplexe Struktur annehmen, bei der sich im Extremfall in jedem Takt der Zustand des ganzen FPGAs ändern kann.

Ein Kommunikations-Overhead, wie er bei CPUs durch das notwendige Laden und Schreiben von Befehlen und Daten auftritt, fällt bei FPGAs völlig weg. Die einzelnen Rechenelemente sind direkt miteinander gekoppelt. Das Ergebnis einer Operation kann im darauf folgenden Takt bereits von der nächsten Recheneinheit verarbeitet werden. An dieser Stelle ist das für FPGA-Prozessoranwendungen wichtige Parallelisierungkonzept des Datenflußrechners bzw. der systolischen Architektur zu nennen (vergleiche Abschnitt 3.2.2). Die Daten durchlaufen eine Kette von aufeinanderfolgenden Verarbeitungsstufen innerhalb der FPGAs. Eine systolische Pipeline erzeugt mit jedem Takt ein Ergebnis, wobei eine hochgradige Parallelisierung erreicht werden kann.

Neben der gesteigerten Rechenleistung zeichnen sich entsprechend aufgebaute FPGA-Prozessoren durch extrem hohe Datenbandbreiten von den I/O-Schnittstellen zu den FPGAs hin aus. Auch hier ist nicht vorrangig die Frequenz für die erzielten Bandbreiten verantwortlich, sondern die möglichen Busbreiten. Zum Beispiel besitzt das Enable++-System zwischen den I/O-Modulen und den FPGAs einen Datendurchsatz von 1,6 GByte/s. Erreicht wird dies durch die Implementierung eines 288-Bit-breiten System-Busses.

Hohe Datenbandbreiten lassen sich auch zu den direkt an das FPGA ankopppelbare Speicher-Bausteine erzielen. Dafür verantwortlich ist zum einen der direkte Zugriff und zum andereren der wiederum erreichbare, hohe Parallelisierungsgrad.

Neben den dargestellten Vorteilen der FPGA-Architektur für die Ausführung von spezifischen Anwendungen muß speziell auf eine Schwäche dieser Architektur hingewiesen werden, die mit ein wichtiges Argument für die enge Kopplung von FPGAs und CPUs darstellt. FPGAs eignen sich, bedingt durch den sehr hohen Resourcenbedarf, nur eingeschränkt für die Ausführung von Floating-Point-Arithmetik [LMM98]. Erst neuere Untersuchungen haben gezeigt, daß unter bestimmten Umständen Floating-Point-Operationen doch effizient auf FPGA-Systeme abbildbar sind ([Kub99], vgl. Abschnitt 7.2).

Blickt man in der noch kurzen Geschichte der FPGA-basierten Rechnersysteme an den Anfang zurück, muß an erster Stelle das von der Firma Digital Equipment 1991/1992 entwickelte DECPeRLe-1-System genannt werden [BBo93]. Bei einer Reihe von CERN-Projekten wurde die prinzipielle Eignung dieses Systems getestet. Es wurden sowohl die Ausführung von ATLAS-Triggeralgorithmen als auch Teile der globalen Triggerentscheidung untersucht ([KMo94], [LLM95]). Zusammen mit der Mannheimer Enable Machine konnte das DECPeRLe-1-System bei einem Benchmark-Wettbewerb am CERN als die beiden einzigen Systeme die Berechnung des Referenzalgorithmus' unter den gestellten Bedingungen ausführen.[31]

Als mindestens ebenso wichtig für die richtungsweisende Entwicklung aktueller FPGA-Prozessorsysteme ist das Splash-2-System zu sehen. Der Splash2-FPGA-Prozessor wurde von vornherein als universelles System entwickelt. Anwendungen aus verschiedenen Bereichen, wie zum Beispiel aus dem Bereich Bio-Computing,

[31] Siehe Kapitel 3.1.2.

konnten erfolgreich demonstriert werden. Diese Applikationen sind zusammen mit kritischen Anmerkungen zu Soft- und Hardwareaspekten bei der Entwicklung und Programmierung des Splash-2-Systems in einem Buch [BAK96] ausführlich dokumentiert. Unter anderem wird die Frage der Topologie in den Ausführungen stark betont.[32] Gewisse Ähnlichkeiten des Splash-2-Systems mit dem Mannheimer Enable++-Computing-Board sind hinsichtlich der Struktur festzustellen.

Das Enable++-System, als FPGA-Prozessor der 2. Generation, wurde zwischen 1995 und 1996 als detektorunabhängiger LVL2-Triggerprozessor für das ATLAS-Experiment entwickelt [Lud98]. Ein Enable++-System besteht in seiner Grundeinheit aus vier funktionalen Komponenten: Die eigentlichen Algorithmen werden von der **Recheneinheit** (Computing Board, Abb. 4.1) ausgeführt, während die zu prozessierenden Daten über die **I/O-Einheit** (I/O-Board) in das System gelangen. Die **Kommunikationseinheit** (Active Backplane) stellt die Verbindung von Matrix und I/O-Board dar. Es können (abhängig von der Slotanzahl auf der Backplane) beliebige Kombinationen von Matrix- und I/O-Boards zusammengestellt werden. Die zugehörige **Controllereinheit** besteht aus einer VME-Workstation und einem Transputernetzwerk. Der VME-Bus, der zusätzlich auf der Backplane implementiert ist, macht es möglich, daß das ganze System in einem einzigen Standard-9HU-VME-Crate untergebracht werden kann.

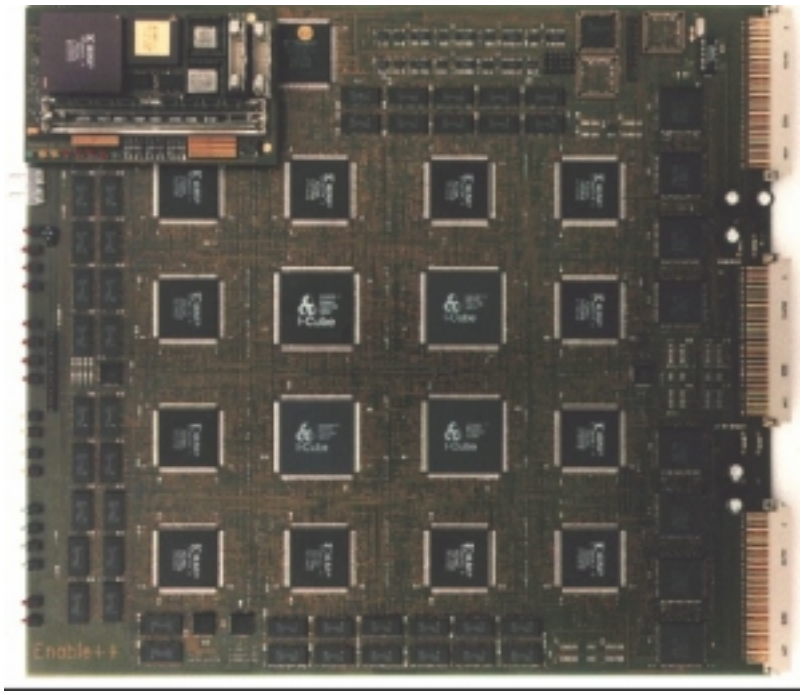


Abbildung 4.1: Das Enable++ Computing-Board mit Controller-Einheit

Das Enable++-System zeichnet sich durch seinen modularen Aufbau aus, was diesen FPGA-Prozessor von anderen abhebt: Die Recheneinheit und die I/O-Einheit

[32]Dazu muß erwähnt werden, daß die ersten FPGA-Prozessoren aus einer Vielzahl von FPGA-Bausteinen aufgebaut sind. Die FPGAs sind zum Teil Matrix-förmig (4x4 und mehr) angeordnet und über einen Interconnect (Crossbar-Switch) untereinander und mit Speicherbausteinen verbunden.

sind auf separaten Platinen untergebracht. Diese Eigenschaft ermöglicht es, sowohl die Rechenleistung als auch die I/O-Bandbreite des Systems skalierbar zu halten. Damit ist eine Anpassung an die verschiedenen Detektoren möglich, wie es für einen LVL2-Trigger im Rahmen des Demonstrator-Programms gefordert wurde. Zusätzlich wurden mit diesem Konzept die Anforderungen von ATLAS bezüglich der Eingangsrate von 100 kHz und der Datentransferrate in der Größenordnung einiger GByte/s erfüllt.

Mittlerweile (Stand Mitte 2000) hat die Anzahl der FPGA-basierten Systeme deutlich zugenommen. Die etablierteste Quelle, die einen Überblick geben kann, ist die von Steven Guccione verwaltete Webseite[33]. Zwei neuere Quellen sind die Webseite der Logic-Jump-Station[34] und die Webseite FPGA-Kochecke[35]. Eine Vollständigkeit bzw. Richtigkeit der dort bereitgestellten Informationen kann allerdings nicht gewährleistet werden. Die Entwickler müssen die Daten zu ihren Systemen selbständig übermitteln und für mögliche Aktualisierungen sorgen. Dennoch wird ein guter Überblick zur Marktsituation gegeben. Auffallend ist der Trend zur Kommerziallisierung von ehemalig akademischen Systemen. Beispiele dafür sind der FPGA-Koprozessor microEnable der Firma Silicon Software und das Spyder-Virtex-X2-System des FZI[36].

Die Vielzahl der zur Zeit existierenden Systeme macht es unmöglich an dieser Stelle eine geordnete Übersicht zu geben. Dennoch lassen sich systematische Gemeinsamkeiten finden. Ein Großteil der Systeme existiert als Single-FPGA-Prozessor-Systeme. Die Systeme sind entweder stand-alone oder als Koprozessor an einen kommerziellen Host ankoppelbar. Die häufig genutzten Bussysteme sind hierbei der PCI-Bus und der ISA-Bus. Die Systeme bestehen meist aus einer einzigen Baugruppe. Typischerweise sind neben dem FPGA noch Speicherbausteine verschiedenen Typs, I/O-Schnittstellen und Universalsteckplätze integriert. Für einige wenige Systeme werden professionelle Entwicklungsumgebungen angeboten.

Prinzipiell ist vor einer Neuentwicklung zu prüfen, ob nicht bereits ein System existiert, welches die gestellten Anforderungen erfüllen kann. Das dies nicht der Fall war, kann bereits an dieser Stelle vorweggenommen werden.

4.2 Anforderungen an das ATLANTIS-System

Das ATLANTIS-System wird in erster Linie als Trigger-Prozessor für die 2. Stufe des ATLAS-Triggersystems entwickelt und soll dort für den Einsatz an zwei Kernstellen untersucht werden: An erster Stelle soll mit dem ATLANTIS-System das

[33]http://www.io.com/~guccione/HW_list.html.
[34]http://www.optimagic.com/boards.html.
[35]http://www.aufzu.de/FPGA/boards.html.
[36]http://www.fzi.de/sim/people/weiss/spyder.html.

Koprozessorkonzept zur beschleunigten Ausführung der rechenintensiven Innendetekor-Full-Scan-Algorithmen demonstriert werden. Zweitens ist das System zu Studien einer FPGA-basierten ROBComplex-Architektur vorgesehen[37]. Die Anforderungen zur optimalen Umsetzung der genannten Anwendungen leiten grundlegend die Konzeptionierung des neuen Systems.

Daneben gibt es weitere konkrete Einsatzgebiete:

1. Zeitgleich mit dem Beginn der Entwicklung des ATLANTIS-Systems wurde die Umsetzung einer 3D-Volume-Rendering-Anwendung für FPGAs begonnen.
2. Das neue System ist nicht nur als Triggerprozessor für ATLAS vorgesehen, sondern bei der Entwicklung wurde auch der Einsatz als Triggerprozessor bei zwei Experimenten an der GSI/Darmstadt berücksichtigt.
3. In der Mannheimer FPGA-Arbeitsgruppe werden FPGAs ganz allgemein für Bildverarbeitung-Applikationen untersucht. Mit dem ATLANTIS-System soll eine leistungsfähige Plattform geschaffen werden, die diese Untersuchungen optimal unterstützt.
4. In einem weiteren Mannheimer Projekt wird die Leistungsfähigkeit von FPGA-Prozessoren zur Unterstützung von Simulationsberechnungen von N-Körper-Systemen in der Astronomie evaluiert. Auch dies findet Berücksichtigung in der Liste der Anforderungen.

Die genannten Anwendungen stellen zunächst eine repräsentative Auswahl dar. Die Ausführung weiterer Applikationen soll generell durch eine flexible Architektur unterstützt werden.

In der nachfolgenden Tabelle sind die Hardware-Anforderungen zusammenfassend aufgelistet: In Spalte 1 ist jeweils die Anwendung beschrieben und Spalte 2 nennt die spezielle Aufgabe (Algorithmenschritt), die in Spalte 3 auf die vier grundlegenden Komponenten (FPGAs, RAM, I/O-Schnittstelle und Interconnect zwischen den FPGAs) abgebildet werden. Spalte 4 beinhaltet die zugehörigen Anforderungen und Vorgaben. Eine ausführliche Darstellung der Anwendungen außerhalb von ATLAS finden sich in Kapitel 7. Diese Kapitel beinhaltet auch eine detaillierte Darstellung der ROB-Complex-Anwendung bei ATLAS. Die in der Tabelle angegeben Daten stammen teilweise aus Implementierungen bzw. Implementierungsanalysen für das Enable++- und für das microEnable-System.

[37]Während der Entwicklung von ATLANTIS wurde von einer Arbeitsgruppe in Israel Interesse bekundet, das System als Prototypensystem für den ATLAS ReadOut-Driver (ROD) einzusetzen. Die Anforderungen des ROD sind vergleichbar mit denen des ROB-Complex' und werden aus diesem Grund an dieser Stelle nicht explizit diskutiert (vgl. Abschnitt 7.1.4).

Algorithmus	**Aufgabe**	**Komponente**	**Anforderung**
ATLAS: TRT	Histogrammierung (Hough-Trans-formation)	RAM	>= 1 Bank, <= 20 Adressen, möglichst viele Datenbits (>= 500 pro Board) Zugriffe rein random, nicht linear => kein SDRAM möglich
		FPGA	Einfache Zähler, Verbrauch pro-portional zur Datenbreite
		I/O	Interner I/O zwischen CPU und FPGA-Teil, Host-I/O für Servicefunktionen. Input: >=1 Straw-Adresse/Takt Output: geringe Bandbreite (Ggf. Separates I/O-Modul)
		Interconnect	Propagation der Straw-Adressen und Ergebnisse
	Max-Finding	FPGA	Einfache lokale Arithmetik (Vergleicher)
		Sonst	Wie bei Histogrammierung
	Gap-Kriterium	FPGA	
		Sonst	Wie bei Histogrammierung
	Spur-Fit, Floating-Point-Arithmetik	FPGAs	Sehr hoher Resourcenbedarf, besser geeignet zur Ausführung auf einer eng gekoppelten CPU
		Sonst	Wie bei Histogrammierung
ATLAS: SCT	Histogrammieren (Hough-Trans-formation)	FPGA	Segmentierte Spuren => mehr Zähler, plus Addierer und Vergleicher
		Sonst	Wie bei TRT
ATLAS: ROBComplex	High-Speed Buffermanagement	RAM	- Min. 3 Bänke unterschiedli-cher Größe. - ca. 1MB für Fragmentdaten - ca. 256k für RoI-Datensätze - ca. 128k Fragment Queue - DPR für paralleles Read/Write
		FPGA	Geringer Resourcenverbrauch
		I/O	Schneller I/O Input und Output über S-Link
		Interconnect	Linear, typischerweise 32-Bit-breite Datenpfade; Interne Kopplung mehrerer RobIns kontrolliert von einem gemeinsamen ROB-Kontroller

2D-Bildverarbeitung	Lineare Filter, Morphologische Filter, Interpolation	RAM	>= 2 Bänke(Quellbild, Zielbild) pro FPGA, möglichst große Bitbreite für hohe Pixelparallelität, i. allg. Datenrate bei Quelle höher als bei Ziel, Zugriffe teilweise sequentiell möglich (SDRAM). Ggf. zusätzliche LUTs für Farbraumkonvertierungen notwendig
		FPGA	Vorwiegend einfache Arithmetik, Vektor-Matrix-Multiplikation, Schwellwertoperationen
		I/O	Extern über variable Module, Anschluss von Kameras, intern zur Host-CPU, hohe Datenraten (einige 10 bis 100 Mbyte/s), Tranfer in/vom loaklen Speicher parallel zu den Rechenoperationen
		Interconnect	Bild wird auf FPGAs verteilt, vorzugsweise mit Overlap => kein oder nur geringer Datenaustausch notwendig
3D-Bildverarbeitung	Visualisierung, Trilineare Interpolation (Lineare Filter + Gradienten-Interpolation), Shading, Stereo Rekonstruktion	RAM	Wie bei 2D-BV, aber: Sehr große Bilddatenblöcke >= 1 GByte => SDRAM mit optimierten Adressmapping notwendig
		Sonst	Wie 2D-Bildverarbeitung
GSI/FOPI		RAM	Vergleichbar zu ATLAS/TRT,
		FPGA	Vergleichbar zu ATLAS/TRT
		I/O	Massive paralleler externer I/O notwendig zur Aufnahme der 960 Komperatorausgänge/FADC (Driftkammer), spezielles I/O-Interface für Signale des Flugzeitdetektors
		Interconnect	Vergleichbar zu ATLAS/TRT
Astronomie	Floating-Point-Arithmetik	RAM	4 Bänke für Quelldatensätze und Ergebnisse, Arithmetische Operationen (1/x, $x^{½}$, x^2) je nach FPGA Resourcen teilweise in zusätzlichem RAM (ca. 20 RAM Bänke)

		FPGA	Sehr hoher Resourcenverbrauch, schnelle FPGAs notwendig
		I/O	Schneller bidirektionaler Host-CPU-I/O notwendig, Transfer in/vom lokalen Speicher parallel zu Rechenoperationen
		Interconnect	Ziemlich komplex, teilweise drei parallele Datenpfade (Datenbusbreite: 20-28bit, Adressbusbreite: 20-28bit)

Tabelle 4.1: Anforderungen verschiedener Anwendungen an die ATLANTIS-Hardware

Aus dem Inhalt der Tabelle 4.1 wird zunächst eines deutlich. Aufgrund der teilweise sehr hohen Ansprüche an das Leistungsvermögen der FPGA-Logik kommt für die zu entwickelnde Hardware-Architektur keine Single-FPGA-Prozessorlösung in Frage, wie sie beispielsweise durch den microEnable-Koprozessor repräsentiert wird. Die Entwicklung eines Multi-FPGA-Prozessorsystems in der Art des Enable++-Systems wird demnach favorisiert. Allerdings haben die Erfahrungen bei der Handhabung von Enable++ gezeigt, daß die gewählten physikalischen Boardmaße (36 cm x 36 cm) einen wesentlichen Einfluß auf die Handhabbarkeit des Systems haben. Insbesondere wurde die Entwicklung des Systems dadurch erschwert. Eine sinnvolle Größe stellt das industrielle 6-HU-Format dar. Dies entspricht einer Platinengröße von ca. 23 cm x 16 cm.

Ein erster Stelle steht die **Wahl des FPGA-Typs**. Zur Frage nach dem idealen FPGA läßt sich feststellen, daß er grundsätzlich sehr komplex sein und gleichzeitig über eine hohe Anzahl von nutzbaren I/O-Pins (Pincount) verfügen muß. Ein hoher Pincount ist essentiell notwendig, um den gleichzeitigen Anschluß von Schnittstellen, Speicher, und weiteren funktionalen Einheiten zu ermöglichen. Auch die Kommunikation zwischen den FPGAs muß eingeplant werden (Interconnect). Sind zur Anwendungsausführung die Logikresourcen mehrerer Baugruppen nötig - was aus dem Anforderungskatalog leicht geschlossen werden kann - ist die Verfügbarkeit eines internen Kommunikationsmechanismus' (im weiteren als Privat-Bus bezeichnet) notwendig. Dies fordert zusätzliche I/O-Pins.

Die Anforderungen an die **Speicherbausteine** sind in Art, Anzahl und Organisation so unterschiedlich, daß die Implementierung eines bestimmten Typs von vornherein ausgeschlossen ist. Es stellt sich also grundsätzlich die Frage, wie eine effiziente Integration von Speicher ins System bewerkstelligt werden kann. Typische Speichertechnologien, die bei FPGA-Prozessoranwendungen eingesetzt werden sind SSRAM[38], SDRAM[39] und DPR[40]. SSRAM erlaubt einen schnellen wahlfreien Zugriff und wird beispielsweise bei LUT-basierten Anwendungen eingesetzt. SDRAM zeichnet sich gegenüber SRAM durch vergleichsweise größere Speicherkapazitäten aus.

[38]SSRAM: Synchronous Static Random Access Memory
[39]SDRAM: Synchronous Dynamic Random Access Memory
[40]DPR: Dual Port Random Access Memory

Als nachteilig stehen dem die langen Zugriffszeiten bei Einzelzugriffen gegenüber[41]. Deshalb wird SDRAM hauptsächlich zur seriellen Speicherung großer Datenblöcke eingesetzt. DPR besitzt als kennzeichnende Eigenschaft, daß Zugriffe auf den Speicher über zwei unabhängige Schnittstellen möglich sind. Dies wird unter anderem dazu benutzt, um eine Entkopplung von Datenströmen bei Schnittstellen zu erreichen. Für die Anwendungsentwicklung für das ATLANTIS-System werden alle beschriebenen Speichertypen benötigt. Allerdings wird die gleichzeitige Implementierung durch die Anzahl von I/O-Pins der FPGAs eingeschränkt und ist grundsätzlich wenig effizient. Unter dieser Randbedingung wurde beschlossen, die Speicherbausteine nicht auf der Basis-Platine zu implementieren, sondern an deren Stelle universelle Steckverbinder vorzusehen. Über diese können dann anwendungsabhängig spezifische Speicher-Module aufgesteckt werden. Weiterhin können dadurch nicht nur dedizierte RAM-Architekturen in das System integriert werden, sondern es wird auch die Integration weiterer Logikbausteine ermöglicht. Alle genannten Speicherarchitekturen werden mittlerweile im ATLANTIS-System eingesetzt. Wie dies in der Praxis umgesetzt wurde, zeigen die Anwendungsbeschreibungen in Kapitel 6 und 7.

Die dritte wichtige Fragestellung bezieht sich auf die Realisierung von Schnittstellen. Im Vordergrund steht die Frage nach einer leistungsfähigen **Verbindung zwischen den FPGA-Modulen und der Host-CPU**. Dieser Schnittstelle fällt die bedeutsame Rolle zu, die Basis für die Umsetzung des hybriden Ansatzes des ATLANTIS-Systems darzustellen. Für die Ausführung einer verteilten Anwendung auf FPGAs und einer CPU ist der schnelle und leistungsfähige Datenaustausch eine essentielle Forderung. Daneben stellt sich die Frage, wie externe Systeme angekoppelt werden können. Aufgrund der unterschiedlichen Anforderungen ist die Implementierung eines bestimmten Schnittstellen-Standards ausgeschlossen. Die dritte wichtige Schnittstellenfunktionalität stellt die Verbindung der FPGAs untereinander dar. Der Interconnect zwischen den FPGAs hängt stark von der Anzahl der FPGA-Bausteine und letzt endlich von der Topologie der Gesamtsystemstruktur ab. Das gleiche gilt für die Schnittstelle zur Kopplung mehrerer FPGA-basierter Systemkomponenten, um die jeweils geforderte Systemkomplexität zu erreichen (Skalierbarkeit).

Im folgenden sind also zwei Schwerpunkte zu diskutieren.

1. Wahl des geeigneten FPGA-Typs
2. Skizzierung der grundsätzlichen System-Architektur inklusive der verschiedenen Schnittstellenvarianten und deren Realisierungsmöglichkeit

[41]Die Designerstellung im Zusammenspiel mit SDRAMs wird weiterhin durch die Einhaltung der notwendigen Refresh-Zyklen verkompliziert.

4.3 FPGAs

Bei der Suche nach dem geeigneten FPGA-Baustein wurden alle im Sommer 1998 auf dem Markt existierenden FPGA-Hersteller berücksichtigt. In einer ausführlichen Analyse wurden auf der Basis vorgegebener Kriterien der Einsatz von FPGA-Bausteinen von insgesamt acht Herstellern untersucht.

Bevor die durchgeführte Evaluierung besprochen wird, soll im folgenden zunächst darauf eingegangen werden, was FPGAs sind und wie grundsätzlich Anwendungen auf diese speziellen Bausteine abgebildet werden. Damit werden gleichzeitig wichtige Begriffe für das Verständnis rund um die FPGAs eingeführt, die im nächsten Abschnitt als Bewertungskriterien benutzt werden.

4.3.1 Architektur von FPGAs

FPGAs sind frei programmierbare Logikbausteine hoher Komplexität. Sie repräsentieren eine noch sehr junge Bausteinfamilie. Erst 1985 wurden die ersten Vertreter dieser Technologie vom Hersteller Xilinx auf den Markt gebracht. Auf einen FPGA lassen sich eine Vielzahl von Logikfunktionen abbilden. Sie sind keine anwenderspezifischen Bausteine, wodurch FPGAs als Massenware gefertigt werden können und dadurch auch relativ preiswert sind. Die prinzipielle Architektur eines FPGAs ist in Abbildung 4.2 dargestellt.

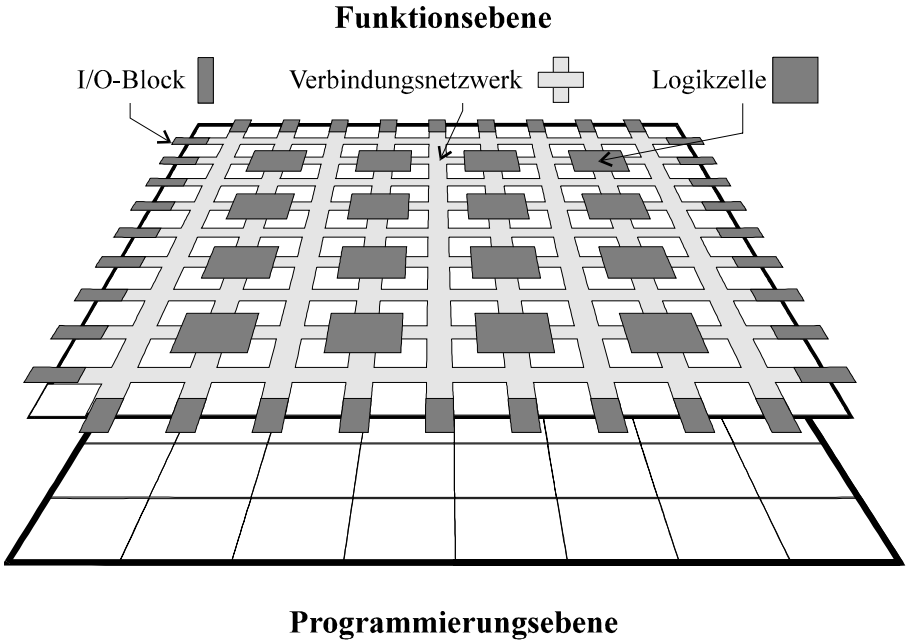


Abbildung 4.2: Prinzipielle Architektur eines FPGAs [Nof95]

Es läßt sich eine Einteilung in zwei Ebenen vornehmen. Die Programmierungsebene ist für den Anwender unsichtbar. In ihr werden die Zustände der Elemente der Funktionsebene physikalisch gespeichert, wobei unterschiedliche Technologien zum Einsatz kommen. Grundsätzlich ist zwischen reversiblen und irreversiblen Programmiertechnologien zu unterscheiden. Bei den reversiblen Technologien werden SRAM-Zellen[42] als Konfigurationsspeicher verwendet - dadurch sind die FPGAs beliebig oft reprogrammierbar. Im irreversiblen Fall wird die Antifuse-Technologie eingesetzt. Durch das Aufschmelzen von Sicherungen wird einmalig und zugleich endgültig die Konfiguration festgelegt. Diese Eigenschaft macht sie für die Verwendung für FPGA-Prozessorsysteme uninteressant und daher werden FPGAs mit dieser Technologie hier nicht weiter betrachtet.

In der Funktionsebene werden die vom Anwender vorgegebenen Applikationen ausgeführt. Die Funktionsebene beinhaltet drei 3 Elemente:

- **Konfigurierbare Logikblöcke (LBs)**:

  Eine quadratische Matrix von LBs bildet das Kernstück des FPGAs. Die LBs beinhalten die programmierbaren Logikresourcen (Abschnitt 4.3.2).

- **I/O-Block (IOB)**:

  Die Anbindung nach außen wird über die IOBs realisiert. Diese I/O-Einheiten können als Eingang oder Ausgang geschaltet oder in einen hochohmigen Zustand gesetzt (getristatet) werden. Prinzipiell sind die I/O-Blöcke der verschiedenen FPGA-Typen ähnlich strukturiert. Unterschiede bestehen in den elektrischen Eigenschaften, wie zum Beispiel der maximalen Stromaufnahme oder der Anzahl der Speicherelemente.

- **FlexiblesVerbindungsnetzwerk**:

  Für die Verdrahtung von IOBs und LBs steht ein mehr oder weniger flexibles Verbindungsnetzwerk zur Verfügung. Es besteht aus einem Gitter von Verbindungskanälen, die zwischen den LBs angeordnet sind. Die Spuren solcher Kanäle können über Schalterelemente mit anderen Spuren, IOBs oder LBs verbunden werden. Die Kreuzungspunkte dieser Kanäle, an denen Schalterelemente sitzen, werden bei den beispielsweise bei Xilinx-FPGAs Switch-Matrizen genannt. Speziell bei den Xilinx-FPGAs gibt es drei Ebenen von Verbindungen. Dazu gehören Spuren von Switch-Matrix zu Switch-Matrix (single length lines), Spuren die eine Matrix überspringen (double length lines) und Spuren, die über den ganzen Chip geführt werden (long lines) [Xil00].

Die logischen Funktionen der LBs, die Zustände der IOBs (Eingang, Ausgang oder getristatet) und die Verbindungsstruktur dieser Blöcke untereinander wird durch ein Konfigurationsprogramm festgelegt. Beim Einschalten werden die

[42]EEPROM-Zellen, die ebenfalls eine Reprogrammierung erlauben würden, werden hauptsächlich bei PLDs eingesetzt. Bislang als einziger Anbieter hat die Firma Gatefield FPGAs mit der EEPROM-Technologie auf den Markt gebracht.

Konfigurationsdaten als *Bitstream* in das FPGA geladen und in den internen statischen Memoryzellen der Programmierungsebene gespeichert. Die Konfiguration kann *in circuit* geschehen, d. h. es wird kein externes Programmiergerät benötigt. Durch die Verwendung von statischen SRAM-Zellen ist das FPGA beliebig oft und sehr schnell reprogrammierbar. Allerdings sind die Programmmierdaten flüchtig und gehen beim Abschalten der Stromversorgung verloren.

Zur Quantifizierung der Komplexität eines FPGAs wird üblicherweise die Maßeinheit Gatter oder auch Gatteräquivalent benutzt. Ein Gatter entspricht einem NAND-Gatter mit zwei Eingängen. Die Komplexitätsangaben der verschiedener Hersteller müssen grundsätzlich kritisch betrachtet werden. Da die Logikblöcke (Abschnitt 4.3.2) nicht aus einzelnen Gattern aufgebaut sind, gibt es für die Umrechnung keine einheitliche Regel. Deshalb lassen sich direkt nur die Angaben eines Herstellers bezüglich der eigenen FPGAs vergleichen.

FPGAs werden mittlerweile mit mehreren hundert I/Os angeboten. Die klassischen *Flat-Pack*-Gehäuse (FP-Gehäuse), bei denen die Pins am Gehäuserand angeordnet sind, sind für eine Pinanzahl dieser Größenordnung nicht mehr geeignet. Der benötigte Umfang und die daraus resultierende Gehäusegröße steht in keinem Verhältnis zum Nutzen. Aus diesem Grund wurde in der nahen Vergangenheit bereits *Pin-Grid-Array*- (PGA) und im zunehmenden Maße *Ball-Grid-Array*-Gehäuse (BGA) eingeführt. Diese dominieren mittlerweile den Markt bei Bausteinen mit einem sehr großen Pincount. Bei diesen beiden Technologien sind die Pinstifte (PGA) bzw. die Lotkugeln (BGA) matrixförmig an der Chipunterseite angebracht. Die Bausteine können daher bei gleicher Pinanzahl kleiner ausfallen, im Vergleich zu herkömmlichen Gehäusetechnologien.

4.3.2 Die konfigurierbaren Logikblöcke (LB)

Die konfigurierbaren Logikblöcke (LB), die herrstellerübergreifend einen typischen Aufbau besitzen, sind innerhalb eines FPGAs in einer Matrix angeordnet. Typischerweise beinhalten die LBs kombinatorische Logikeinheiten, Multiplexer und Register.

In Abbildung 4.3 ist der LB des FPGAs vom Typ XC4005H des Herstellers Xilinx aufgezeichnet. Am Beispiel dieses speziellen Logikblocks soll der typische Aufbau eines LBs veranschaulicht werden. Die LBs dieser Baureihe beinhalten zwei Funktionsgeneratoren (Lookup-Tabellen) mit jeweils vier Eingängen und einen mit drei Eingängen. Damit können Logikfunktionen von bis zu 9 Eingangsvariablen implementiert werden. Außerdem ist eine Verwendung der Lookup-Tabellen (LUT) als kleiner RAM-Block möglich. Durch nachgeschaltete Multiplexer können die Ausgänge auf zwei Registerstufen geleitet oder unregistert aus dem LB herausgeführt werden. Es besteht auch die Möglichkeit, die Funktionsgeneratoren zu umgehen und die Eingänge direkt den beiden Registern zuzuführen (hellgrau unterlegt). Weiterhin sind die Clockleitungen eingezeichnet, wobei für die Register sowohl das Originalsignal als auch das invertierte Signal zur Verfügung steht (dunkelgrau unterlegt).

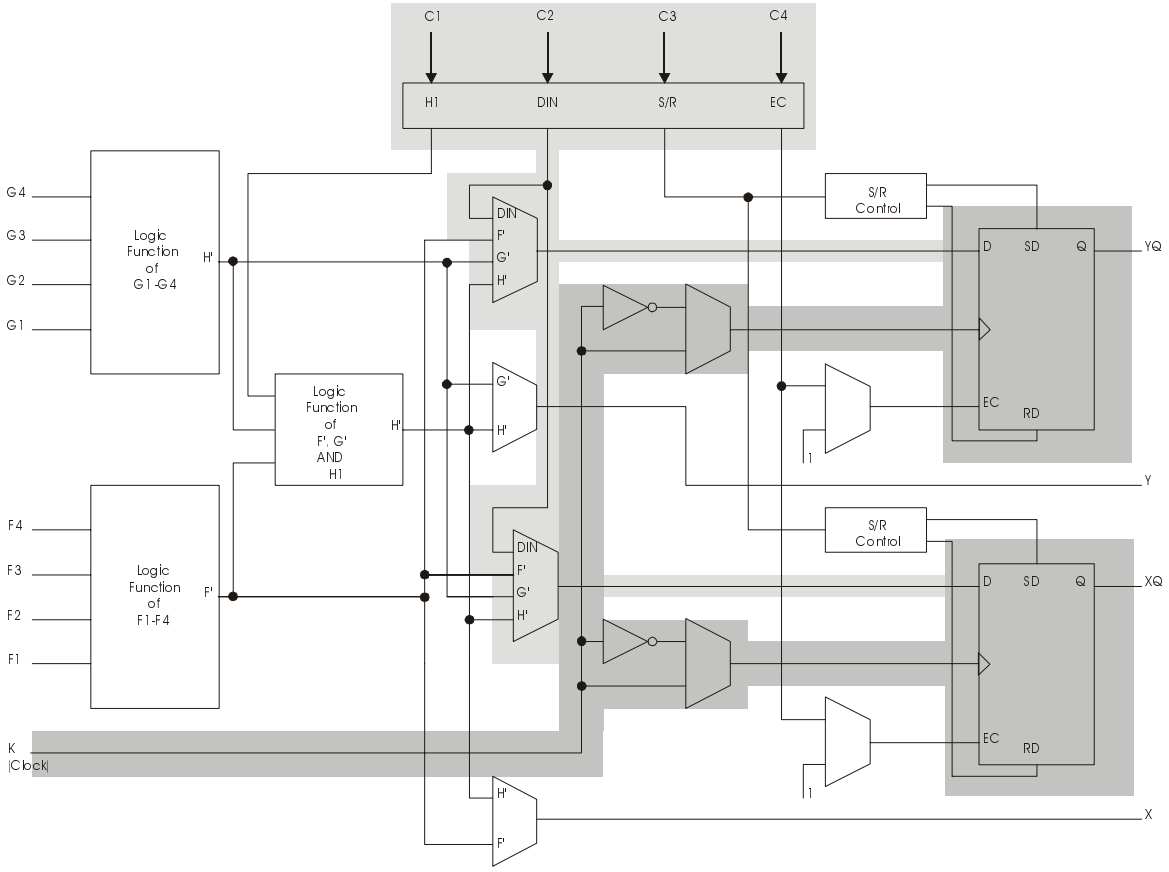


Abbildung 4.3: Der Logikblock des FPGAs XC4005H des Herstellers Xilinx [Xil00]

Grundsätzlich sind die LBs der FPGAs sehr ähnlich aufgebaut. Wesentlich Unterschiede liegen der Granularität der LBs, d.h. in der Komplexität der Logikzelle. Zum Beispiel wird der Logikblock der Xilinx 4k-Serie zu den grobgranularen oder grobkörnigen Logikzellen gerechnet. Aus feingranularen Logikzellen sind beispielsweise die FPGAs des Herstellers Actel aufgebaut (siehe auch Abschnitt 4.3.4). Diese Logikzellen beinhalten nur drei Module, die jeweils eine beliebige kombinatorische Funktion von zwei oder drei Eingängen bilden können.

4.3.3 Programmierung von FPGAs

Der prinzipielle Weg, um eine Applikation auf einen FPGA abzubilden, beginnt mit der mehr oder weniger abstrakten Beschreibung der Schaltung und endet mit der Generierung des Bitstreams, der in den FPGA geladenen wird. In Abbildung 4.4 sind die einzelnen Schritte des FPGA-Designentwicklungablaufs skizziert, der nachfolgend beschrieben wird.

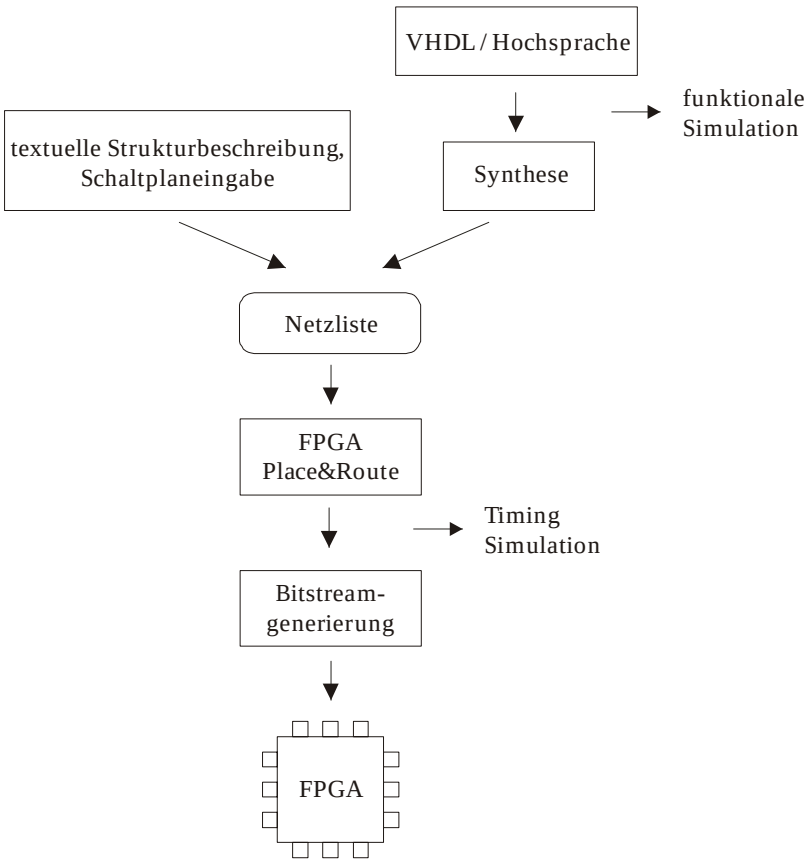


Abbildung 4.4: Ablauf des FPGA-Designentwurfs

Zunächst muß die Schaltung beschrieben werden. Das bezeichnet man auch als Designeingabe. Eine Möglichkeit besteht darin, den Schaltplan in einer spezifischen Software zu generieren (Schaltplaneingabe). Standard-Logikbausteine (Multiplexer, Zähler usw.) werden dem Benutzer hier meist schon als Bibliothekselemente zur komfortablen Schaltplaneingabe zur Verfügung gestellt.

Der beschriebenen visuellen Designeingabe steht die textuelle Eingabe gegenüber. Diese kann auf unterschiedlichen Abstraktionsebenen durchgeführt werden. Auf unterster Ebene werden die Schaltungen in einen strukturellen Darstellung beschrieben. Die Darstellung erfolgt durch die textuelle Beschreibung elementarer logischer Grundfunktionen und deren Verknüpfung untereinander. Dazu zählt auch die Schaltungsbeschreibung mittels logischer Gleichungen und mit Hilfe von State-Machines. Bekannte Sprachen, die die strukturelle Designeingabe unterstützen, sind Abel und Palasm. Auch das in Mannheim entwickelte CHDL, das unter anderem für die Designerstellung für ATLANTIS benutzt werden kann, ist hier einzuordnen (Abschnitt 5.6.3). Diese Sprachen sind relativ weit verbreitet, weil eine direkte Netzlistengenerierung möglich ist und sie eine realitätsnahe, systembezogene Simulation erlauben. Als nachteilig erweist sich die strukturelle Beschreibung bei komplexen Designs. Dies führt zu sehr langen Beschreibungstexten, wobei sehr schnell der Überblick verloren gehen kann.

Die Designeingabe auf der Basis einer Verhaltensbeschreibung bieten kommerzielle Hardwarebeschreibungssprachen wie VHDL und Verilog. Dazu gehört auch

die funktionale Simulation zur Verifikation der Schaltung. Die Schaltungsfunktionalität wird grundsätzlich auf einem höheren Abstraktionsniveau als bei der strukturellen Designeingabe dargestellt. Es sind aber auch gemischte Beschreibungen möglich. In Europa ist VHDL die meist genutzte Sprache zur Beschreibung von FPGA-Designs. Die Beschreibung in VHDL ist Technologie-unabhängig. Deshalb sind sogenannte Synthesewerkzeuge notwendig, die die Abbildung auf die Zielarchitektur durchführen (siehe unten). Das Ergebnis aus der VHDL-Software ist eine Netzliste der Schaltung. Die Anwendung der Sprache VHDL zur Anwendungsentwicklung für das ATLANTIS-System wird ausführlich in Abschnitt 5.6.2 erläutert.

Hardwarebeschreibung auf höchstmöglichen Abstraktionsniveau bieten C-ähnliche Hochsprachen wie HandelC [HLR98], Transmogrifier C [Gal95] und das in Mannheim entwickelte ppC [Zoz97]. HandelC erlaubt zum Beispiel eine sequentielle und parallele Ausführung und benutzt ein Channel-Konzept zur Synchronisation laufender Prozesse. Herkömmliche C-Programme können prinzipiell portiert werden - um die Möglichkeiten bei der Ausführung auf FPGAs effizient auszunutzen, muß der Code allerdings neu geschrieben werden. Auch hier läßt sich die Schaltung mittels einer funktionalen Simulation prüfen. Ein großer Nachteil bei der Hochsprachen-Beschreibung sind die erzielbaren Ergebnisse in Form der Frequenz, mit der der FPGA getaktet werden kann. Ein Vergleich mit der VHDL-Beschreibung führt im Mittel zu einer um einen Faktor 10 niedrigeren Designfrequenz.

Sowohl die Beschreibung in VHDL als auch in einer Hochsprache muß in einem weiteren Schritt auf die Zielarchitektur abgebildet werden (Synthese). Dazu wird die Schaltungslogik in kleine Subschaltungen aufgeteilt und effizient in die LBs der jeweiligen FPGA-Architektur plaziert. Zusätzlich wird von der jeweiligen Synthese-Software die Logik der Schaltung (wenn möglich) durch Reduktion optimiert (z.B. werden Redundanzen beseitigt). Das Ergebnis ist eine Netzliste, die die vollständige Beschreibung der Schaltung beinhaltet.

Danach findet die physikalische[43] Plazierung und Verdrahtung der LBs statt, unter Berücksichtigung der Verbindungsresourcen innerhalb des spezifischen FPGA-Typs (Place&Route). Dabei werden eventuell einzuhaltende Signallaufzeiten und weitere definierte Randbedingungen (constraints) miteinbezogen. Damit sollen bessere Ergebnisse hinsichtlich der maximalen Ausführungsgeschwindigkeit der FPGA-Designs erreicht werden.

Nun können die Konfigurationsdaten in Form des Bitstreams erzeugt werden. Bevor nun die Konfigurationsdaten in den FPGA geladen werden, kann an dieser Stelle eine Timing-Simulation darüber Aufschluß geben, ob das gewünschte Ergebnis erzielt werden konnte.

[43]Bis zu diesem Schritt wurde die Logik auf virtuellen Logikzellen abgebildet.

4.4 FPGA-Evaluierung

Für die FPGA-Evaluierung wurden die FPGA-Familien berücksichtigt, die zum Zeitpunkt der Konzepterstellung (August 1998) verfügbar waren, bzw. deren Verfügbarkeit für die nächsten Monate angekündigt war, aber zumindest Muster bereits bezogen werden konnten. Diese harte Grenze, nur existente Bausteine in die Evaluierung mit einzubeziehen, ist begründet in der Strategie der Hersteller neuere leistungsfähigere Bausteine sehr frühzeitig anzukündigen, um potentielle Kunden zu ködern. Bis zum Erscheinen der FPGAs, wenn dies überhaupt stattfindet, können dann noch Monate bis Jahre vergehen, was natürlich nicht akzeptabel ist. Davon betroffen war die sehr leistungsfähige und wie sich mittlerweile (Sommer 2000) herausgestellt hat, sehr zukunftsweisende Virtex-FPGA-Familie der Firma Xilinx. Für das zeitlich später entwickelte ATLANTIS I/O-Board konnte im Gegensatz zum ATLANTIS Computing-Board diese Familie mitberücksichtigt werden. Eine separate Beschreibung der Virtex-FPGAs findet sich in Abschnitt 4.4.2. Für die FPGA-Evaluierung wurden alle FPGA-Hersteller berücksichtigt, die im Sommer 1998 Bausteine auf dem Markt hatten. Dies sind die Firmen Actel, Altera, Atmel, DynaChip, Lucent, Vantis und Xilinx. Im nächsten Abschnitt sind die evaluierten FPGA-Familien in alphabetischer Reihenfolge zusammenfassend beschrieben.

4.4.1 Übersicht FPGA-Hersteller

Bereits im Oktober 1996 kündigte **Actel** mit der ES-Serie erstmals eine FPGA- Familie an [Act96]. Die ersten FPGAs dieser Familie waren im 1. Quartal 1998 verfügbar. Kennzeichnend für diese FPGA ist die feine Granularität, mit der ein zu 100% automatisches Routing ermöglicht werden soll. Die Verdrahtung der Bausteine geschieht über ein hierarchisch organisiertes Verbindungsnetzwerk. Eine weitere Besonderheit sind die integrierten PLLs (Phase Locked Loops) zur Anpassung der Clockfrequenzphase. Mit bis zu 400.000 Gattern des größten angekündigten FPGAs gehören sie zu den komplexeren Vertretern.

Die Firma **Altera** ist hauptsächlich bekannt als Hersteller von PLDs (Programmable Logic Devices). Mit den Flex-Serie [Alt98] wurden ab 1995 von Altera 3 FPGA-Bausteinfamilien auf den Markt gebracht, die einerseits eine Schaltmatrix-Verdrahtung besitzen wie PLDs. Anderseits sind die Logikblöcke aus Lookup-Tabellen (LUTs) aufgebaut, weshalb sie im allgemeinen als FPGAs bezeichnet werden[44]. Bei der in der Evaluierung berücksichtigten leistungsfähigsten Flex10K(A)-Serie sind jeweils 8 Logikblöcke (LE) zu einem Logic-Array-Block (LAB) zusammengefaßt. Zusätzlich besitzen diese FPGAs Embedded-Logic-Blocks (EABs), die als Logikfunktionen oder als RAM konfiguriert werden können.

Wie Altera ist auch die Firma **Atmel** kein typischer FPGA-Produzent, sondern eher als Hersteller von Mikrokontroller, EEPPROMs, ASICs und PLDs bekannt.

[44] Altera selbst bezeichnet sie als PLDs.

Ende 1997 wurde von Atmel die FPGA-Familie AT40K [Atm97] auf den Markt gebracht, eine Weiterentwicklung der AT6000-Serie. Kennzeichnende Eigenschaften dieser Familie sind die Fähigkeit zur partiellen Rekonfigurierbarkeit im laufenden Betrieb (*Cache Logic*), die feingranularen Logikzellen und das festintegrierte verteilte RAM.

Die Firma **DynaChip** ist als Exot unter den FPGA Herstellern zu sehen. Aufgrund der geringe Komplexität der DynaChip FPGA-Familien [Dyn98] und der speziellen Unterstützung für Signalpegel der ECL-, PECL- und GTL-Logik sowie von TTL-Pegeln, lassen diese Bausteine für den Einsatz als Bus-Koppler am geeignetsten erscheinen. Die relativ groß ausgelegten Logikblöcke sind zu Regionen zusammengefasst. Für die Verbindung von Regionen werden aktive Signalverstärker verwandt. Die Firma Dynchip wurde zwischenzeitlich, im Jahr 1999, von der Firma Xilinx aufgekauft.

Lucent Technologies hat seine Aktivitäten auf dem FPGA-Sektor zunächst als Second-Source mit der Vermarktung von Xilinx-Bausteinen begonnen. Seit 1993 fertigt Lucent eine eigene FPGA-Familie, die 1998 mit Bausteinen der ORCA Series3 -FPGAs [Luc98] die leistungsfähigsten Vertreter stellt. Die Besonderheit der ORCA-FPGAs sind die Erweiterung der grundsätzlich grobgranularen Logikblöcke (PFU) um jeweils einen zugehörigen *Supplemental Logic and Interconnect*-Block (SLIC), wodurch sich effizient PAL-ähnliche Funktionen implementieren lassen. Die Kombination aus PFU und SLIC wird als *Programmable Logic Block* (PLC) bezeichnet. Zusätzlich sind diese Bausteine mit einem Mikroprozessor-Interface, PLLs, und zwei programmierbaren Clockmanager ausgestattet. Neben den reinen FPGA-Bausteinen gibt es Chips (FPSCs), die bereits festverdrahtete Funktionen (z.B. PCI und ATM) enthalten.

Die Firma **Vantis** ist eine Tochterfirma von AMD und deckt seit Jahren das Produktspektrum der programmierbaren Logik-Bausteine (Mach-CPLDs und PALs) ab. Seit 1998 ist Vantis auch im FPGA-Sektor mit der VF1-Familie [Van98] aktiv. Kennzeichnend für diese Bausteine ist die hierarchisch aufgebaute Logikblock-Struktur, wodurch dem Anwender wahlweise eine fein- oder grobkörnige Architektur zur Verfügung steht. Fest integriert sind zusätzlich Dual-Port-RAM-Module.

Als letzter FPGA-Hersteller ist die Firma anzusprechen, die mit Abstand den größten Anteil an der Verbreitung und dem Erfolg der FPGAs hat: **Xilinx**. Die Firma Xilinx hat im Jahr 1985 der ersten FPGA auf den Markt gebracht und ist von Beginn an bis heute der Marktführer. Die 1998 leistungsfähigste Familie ist die 4k-Serie [Xil00]. Für die Evaluierung wurde sowohl der größte Typ der etablierten XL-Serie und der erste Vertreter der neuen XV-Serie berücksichtigt. Xilinx FPGAs der 4k-Serie sind aufgrund der Größe der Logikblöcke[45] (CLBs) den grob-körnigen FPGA-Familien zuzuordnen. Die Konfiguration verschiedener RAM-Typen in die CLBs wird ebenso unterstützt wie der Aufbaus schneller arithmetischer Funktionen, wie zum Beispiel Addierer, Vergleicher und Zähler. Dies wird erreicht durch spezielle Verbindungswege zwischen den CLBs, den sogenannten *Carry Chains*. Diese finden sich auch in den FPGAs der Hersteller Lucent und Altera.

[45] Siehe Abbildung 4.2.

Nach diesem kurzen Überblick folgt im Abschnitt 4.4.4 eine detaillierte Auflistung der Eigenschaften der leistungsfähigsten FPGA-Familien jedes einzelnen Herstellers. Basierend auf einer Reihe von Kriterien, wird der Einsatz der FPGAs für das ATLANTIS-System diskutiert.

4.4.2 Virtex-FPGAs des Hersteller Xilinx

Die neue Virtex FPGA-Familie war zwar zum Zeitpunkt der FPGA-Evaluierung schon angekündigt, wurde aber wegen nicht existierender Muster und der fehlenden Software-Unterstützung für die Designtests nicht mit einbezogen.

Seit dem Sommer 1999 sind diese FPGAs auf dem Markt und setzen in jeder Hinsicht neue Maßstäbe. Mit den Virtex-FPGAs wird erstmals die 1 Million-Gatter-Grenze durchbrochen. Es sind Gehäuse mit über 1000 Pins angekündigt, die zwischen 800 und 1000 User-I/Os ermöglichen. Zum ersten Mal sind im großer Anzahl RAM-Bänke fest integriert. Mit dieser Eigenschaft wird es erstmalig möglich sein, Zwischenergebnisse von Algorithmenschritten effizient im Baustein zwischen zu speichern. Gleichzeitig können im großen Stil LUTs implementiert werden.

Daneben haben die ersten Testdesigns bereits in der Praxis gezeigt, daß im Vegleich zur Xilinx 4k-Serie und der Lucent Series3-FPGAs die Geschwindigkeit von Anwendungen erheblich gesteigert werden kann. Unter anderem wurden Algorithmenschritte des Full-Scan-TRT auf die Virtex-Technologie abgebildet und dabei eine Erhöhung der Ausführungszeit um bis zu 100% festgestellt [Sim99].

Aufgrund dieser Untersuchungsergebnisse - man kann von einer Erweiterung der FPGA-Evaluierung sprechen - die im Jahr 1999 gemacht wurden, ist die Wahl des FPGA-Typs für das ATLANTIS I/O-Board auf den Virtex-Baustein gefallen. An der Argumentation gegen den Einsatz von FPGAs anderer Hersteller, wie sie im Rahmen der ursprünglichen Evaluierung vorgebracht wurden (Abschnitt 4.4.4) hat sich prinzipiell nichts geändert. Gegen die für das ATLANTIS Computing-Board gewählte FPGA-Serie von Lucent - im Vergleich zur neuen Xilinx Virtex-Serie - spricht in erster Linie die geringere Komplexität, das Fehlen der integrierten RAM-Blöcke und die niedrigeren erzielten Designfrequenzen.

4.4.3 Vergleich der FPGA-Architekturen

Um die verschiedenen FPGA-Familien objektiv in Hinblick auf die Eignung für ATLANTIS vergleichen zu können, müssen zunächst die wichtigsten Eigenschaften recherchiert und zusammengestellt werden. Die Auswahl der Eigenschaften basiert auf den Erfahrungen der Mannheimer Gruppe. An der Evaluierung waren neben dem Autor außerdem Harald Simmler, Klaus Kornmesser und Andreas Kugel beteiligt. In diesem Team wurde gemeinsam ein FPGA-Kriterien- und Eigenschaftskatalog ausgearbeitet, die Informationen gesammelt und die endgültige Entscheidung

getroffen. Der Kriterienkatalog leitet sich zum einen aus den Anforderungen der Anwendungen ab. Zum anderen müssen allgemeine Systemeigenschaften mitberücksichtigt werden.

Die Evaluierung erfolgte in zwei Teilen. Im ersten Schritt wurden die leistungsfähigsten FPGA-Familien aller Hersteller auf essentielle Kriterien hin verglichen. Die Eigenschaften der FPGAs sind in der Tabelle 4.2 aufgelistet. Die zugehörige Diskussion wird im nächsten Abschnitt geführt. Im zweiten Schritt wurden für die beiden verbliebenen Kandidaten (Xilinx und Lucent) weitere Praxis-Tests mit Referenzdesigns durchgeführt (Abschnitt 4.4.4).

Die in der Tabelle 4.2 angegebenen Daten beziehen sich entweder auf alle Bausteine einer Familie, wenn es sich um gemeinsame Eigenschaften handelt, bzw. auf den zur Zeit leistungsfähigsten, verfügbaren (!) Vertreter der Familie. Dies wird durch die Angabe des FPGA-Bausteins (jeweils in Klammern) verdeutlicht. Die Eigenschaften sind in logische Blöcke zusammengefaßt. Die Blöcke umfassen Angaben zu:

- **Pinout in Abhängigkeit vom Gehäusetyp** (Pincount): Beeinflußt maßgeblich die Anbindung von Peripherie (Speicher, Schnittstellen und weitere Logik).
- **Versorgungsspannung:** Welche Versorgungsspannungen müssen im System vorhanden sein und inwiefern sind die FPGAs Signal-kompatibel zu anderen Logikbausteinen.
- **Resourcen:** Ein Anhaltspunkt für die Komplexität (Leistungsfähigkeit) der FPGAs.
- **Internes konf. RAM**: Wichtig als Zwischenspeicher bei der Algorithmen-Ausführung, DPR dient zur Entkopplung unterschiedlich getakteter Systembereiche.
- **Bibliotheken:** Das Vorhandensein von vorgefertigten Teillösungen unterstützt und vereinfacht den Algorithmen-Entwicklungsprozeß.
- **Konfiguration:** Art der Schnittstelle und Dauer der Konfiguration. Die Dauer gibt an, wie schnell die Designs gewechselt werden können.
- **Test:** Als essentiell ist die Verfügbarkeit der JTag-Schnittstelle für den Verbindungstest der bestückten Platine zu sehen. Der Readback ist wiederum wichtig für die Entwicklung eines Online-Debuggers und den Multitasking-Betrieb
- **Sonstige Features:** Clock-Einstellungsmöglichkeiten leisten einen Beitrag zur Optimierung der FPGA-Designs.
- **Markt**: Gibt einen Hinweis auf die zu erwartende, längerfristige Verfügbarkeit der FPGAs.

4.4.4 Wahl des FPGA-Typs für ATLANTIS

In diesem Abschnitt wird der Argumentationsgang beschrieben, der zur Wahl des FPGAs für ATLANTIS geführt hat. Es werden nicht im Detail alle Eigenschaften diskutiert, die in der Tabelle 4.2 dargestellt sind. Der Vergleich wird auf die wichtigsten Kriterien begrenzt, die ausschlaggebend für die Wahl des FPGAs waren:

In Bezug auf Komplexität (Leistungsvermögen) und Anzahl verfügbarer I/O-Pins, die mit die wichtigsten Kriterien für die Wahl des geeignetsten FPGA darstellen, kommen nur die FPGA-Familien von Xilinx, Lucent und Altera in Frage. Die anderen Familien haben eine deutlich geringere Komplexität und verfügen auch mit Abstand über weniger I/O-Pins.

Eine weitere wichtige Eigenschaft ist die Verfügbarkeit von internem RAM, entweder festkodiert oder implementierbar. Das interne RAM dient grundlegend dazu, Ergebnisse von Algorithmenschritten direkt im FPGA zwischen speichern zu können. Auch hier sind die zuvor genannten Hersteller führend. Allerdings besteht bei Altera die bedeutsame Einschränkung, daß kein echtes, asynchrones Dual-Port-RAM (DPR) implementiert werden kann. Asynchrones DPR ist sehr wichtig für die Entkopplung von zwei Schnittstellen mit unterschiedlichen oder phasenverschobenen Taktfrequenzen. Dies spielt zum Beispiel eine wichtige Rolle bei der Realisierung von I/O-Schnittstellen. Ebenso ist dies ein essentieller Mechanismus, um grundsätzlich unterschiedlich getaktete Systembereiche über FPGA-interne Resourcen zu entkoppeln. Dazu müssen für jede zu entkoppelnde Schnittstelle mindestens zwei Clocksignale in den FPGA geführt werden können.

Eine wichtige Eigenschaft, um einen statischen Verbindungstest der Boards durchführen zu können ist das Vorhandensein einer JTag-Schnittstelle. In Abschnitt 5.1.6 wird die JTag-Funktionalität in Zusammenhang mit der Entwicklung des ATLANTIS-Systems ausführlich beschrieben. Die an gleicher Stelle in der Tabelle 4.2 aufgelistete Readback-Funktion kann in zweifacher Hinsicht für die Entwicklung und Ausführungen von Anwendungen genutzt werden. Für die Anwendung eines Online-Debuggers ist das Auslesen der Zustände zu einem beliebigen Zeitpunkt aus dem FPGA nötig. Im Fall des Multitaskingbetriebs muß ebenfalls ein Readback des Zustands der FPGA-Designlogik durchführbar sein. Die Informationen werden benötig, um eine vorher unterbrochene Anwendung (Task) bei einem erneuten Taskwechsel wieder restaurieren zu können. Mehr Informationen zu Multitasking auf FPGAs finden sich hier [SLM00]. Eine Eigenschaft, die die Anwendungsentwicklung vereinfacht, ist die Bereitstellung von FPGA-Cores für Standard-Elemente (z.B. Zähler, Vergleicher, etc.). Diese werden üblicherweise direkt vom FPGA-Hersteller angeboten oder sollten zumindest aus anderen Quellen bezogen werden können. Die letztgenannten Eigenschaften werden von allen, der drei in Frage kommenden FPGA-Herstellern, unterstützt.

Weiterhin ist für die FPGA-Wahl mit zu berücksichtigen, wie lange es die Firma schon gibt und ob die Produktlinie schon länger eingeführt ist und dabei die bereits angekündigten Bausteine auch wirklich produziert wurden. Auch hier kann keine negative Wertung bezüglich der drei verbliebenen Kandidaten abgegeben werden.

	Xilinx XC4000XV	**Xilinx XC4000XL**	**Lucent ORCA Series3**
Pinout			
Gehäuse	BGA; PGA	BGA; PGA	BGA
Max. Pinanzahl	560; 559	560; 559	600
Anzahl max. User I/O	432/BGA; 448/PGA	448/BGA; 448/PGA	448
Max. nutzbare User I/O	416 (40125XV)	420 (4085XL)	432 (OT3T125)
Größe [mm^2]	BG560: 45*45,	BG560: 45*45,	45*45
Versorgungs-spannung			
I/O	3.3V, 5V kompatibel	3.3V, 5V kompatibel	3.3V, 5V kompatibel
Core	2.5V	3.3V	3.3V
Ressourcen			
Körnigkeit	grob	grob	grob
Anzahl Logikzellen	4624 CLBs (40125XV)	3136 CLBs (4085XL)	784 PFUs (OT3T125)
Anzahl Register (ohne I/O)	9.248 (40125XV)	6.272 (4085XL)	7.056 (OT3T125)
Anzahl Gatter	125k	85k	186k
Anzahl Userclocks	8	8	4+2
Internes konf. RAM			
Größe [kBit]	140 (40125XV)	85 (4085XL)	100 (OT3T125)
Architektur	SelectRAM	CLB-RAM	PFU-RAM
Max.Address-Bits/Block	5	5	5
Max Daten-Bits/Block	2	2	4
Dual-Port-Memory	Ja, Sync. +Async.	Ja, Sync. +Async.	Ja, Sync. +Async.
Bibliotheken			
Hersteller Core-Tools	Ja	Ja	Ja
Kommerzielle IP-Module	Ja	Ja	Ja
Festkodierte Blöcke	-	-	Ja, PCI+ATM
Konfiguration			
Seriell/Parallel	Ja/Ja	Ja/Ja	Ja/Ja
Konfig. - Frequenz	10 MHz	10 MHz	25 MHz
Konfig. - Zeit	310ms/38,8ms	190ms/23,8ms	34,5ms/4,3ms
Partielle Konfig.	Nein	Nein	Ja
Register Preload	Ja	Ja	Ja
Test			
JTAG/Bound. Scan	Ja	Ja	Ja
Readback	Ja	Ja	Ja
Sonstige Features			
Prog. Clockmanager	Nein	Nein	Ja
Processor Interface	Nein	Nein	Ja
Markt			
Neu am FPGA-Markt?	Nein (seit 1985)	Nein (seit 1985)	Ja (seit ca. 1995)
Serie verfügbar seit	1998 (Ugrade der XLs)	ca. 1995	Ende 1997
FPGA Marktanteil(1998)	57%	57%	9%

Tabelle 4.2: Eigenschaften von

Altera EPF10KA	**Atmel AT40K**	**Actel A65ES**	**DynaChip DL6000**
BGA; PGA	PQFP; BGA	PQFP, BGA; PGA	BGA
600; 599	432; 475	240;391; 432	352
470	352; 384	320; 640; 640	254
406 (EPF10K100A)	128 (AT40K05)	320 (A65ES100)	254 (DL6035)
BGA600 45*45	PQFP208: 30.6x30.6	PQFP240: 34.6x34.6	35*35
3.3V, 5V (MultiVolt-IO)	3.3V, 5V	3.3V, 5V	3.3V
3.3V	3.3V	3.3V	3.3V
grob	fein	fein	grob
LE: 4992; LAB: 624 (EPF10K100A)	256 (AT40K05)	4096 (A65ES100)	1024 (DL6035)
4992 (EPF10K100A)	256 (AT40K05)	4096 (A65ES100)	2048 (DL6035)
100k (EPF10K100A)	5k-10k (AT40K05)	33k-84k (A65ES100)	35k (DL6035)
6	8	4	2
49 (12 EABs)	2	64	32
EAB-RAM	Embedded RAM	Embedded RAM	conf. RAM
11	5	8	5
8	4	8	1
Ja, nur sync.	Ja, sync. + async.	Ja, sync. + async.	Ja, sync. + async.
-	-	-	-
-	-	-	-
EAB	-	-	-
Ja/-	Ja/Ja	Ja/Ja	Ja/Ja
10 MHz	10 MHz	10 MHz	25 MHz
117ms	3,8ms/0,25ms	300ms (ser.+par.)	0,18ms (ser.+par.)
Nein	Nein	Nein	Ja
Nein	Ja	Ja	Ja
Ja	Nein	Nein, aber ActionProbe	Ja
Nein	Ja	Nein	Ja
Nein	Nein	Nein	Nein
Nein	Nein	Nein	Nein
Nein (seit 1992)	Nein (seit 1993)	Nein (seit 1993)	Ja
1.997	Ende 1997	Anfang 1998	Mitte 1998
13%	<5%	16%	<5%

FPGAs verschiedener Hersteller

Zusammenfassend kann nach dem ersten Evaluierungsschritt festgestellt werden, daß für das ATLANTIS-System grundsätzlich nur FPGAs von der Firma Xilinx oder von Lucent in Frage kommen. Dies wird durch ein weiteres Argument gegen die Altera-FPGAs als dritten potentiellen Kandidaten bekräftigt. Bei diesen FPGAs ist die flexible Nutzung dadurch beschränkt, daß die internen Routingführungen vorgegebene Richtungen haben. Bei der Entscheidung wurde auch miteinbezogen, daß es einen beträchtlicher Aufwand darstellt, sich mit einer neuen FPGA-Architektur vertraut zu machen, insbesondere zu lernen, die Programmier-Tools effizient einzusetzen. Das war mit ein entscheidender Grund, die ebenfalls in vielerlei Hinsicht leistungsfähigen Altera-FPGAs nicht in die engere Wahl zu nehmen. Die größte Erfahrung wurde bisher mit den Xilinx-FPGAs gesammelt. Dieser Umstand, zusammen mit den in der Tabelle beschriebenen Eigenschaften, lies sie als sehr geeignet erscheinen. In Bezug auf die Eigenschaften bieten die Lucent-FPGAs in den wichtigen Bereichen überwiegend die größeren Resourcen. Mit den Lucent-FPGAs sind bisher noch keine Erfahrungen gemacht worden, allerdings ist die Architektur sehr ähnlich zu der der Xilinx-FPGAs. Eine große Einarbeitungszeit ist deshalb nicht zu erwarten. Aus diesem Grund gehen die Lucent-FPGAs zusammen mit den Xilinx-Bausteinen in die zweite Evaluierungsrunde.

Bisher (bewußt) übergangen wurde das Kriterium, welche Geschwindigkeiten mit den verschiedenen FPGAs erreicht werden können. Die Hersteller machen hierzu nur Baustein-spezifische Angaben. Diese Angaben dienen allerdings nur dazu, die Geschwindigkeit von FPGAs innerhalb der gleichen Familie eines Hersteller zu charakterisieren. Da ein theoretischer Vergleich nur schwer möglich ist, wurden für die FPGA-Evaluierung Testdesigns definiert, die von den Place&Route-Tools bearbeitet werden mußten. Die Testdesigns sind typische Applikationsteile, wie sie zum Beispiel innerhalb der Implementierung des TRT-Algorithmus' aber auch anderer Anwendungen benutzt werden. Gemessen wurden die erzielten Designfrequenzen und der Resourcenverbrauch.

In diesem Zusammenhang ist auch die Zeit zu beachten, die die FPGA Place&Route-Software benötigt, um ein Design auf den jeweiligen FPGA-Typ abzubilden. Auch diese wurde gemessen. Die Ausführungszeit der Place&Route-Software beeinflußt maßgeblich die Gesamtzeit, in der Änderungen in der Applikation in die Praxis umgesetzt werden können (Turn-Around-Time). In Tabelle 4.3 sind die Ergebnisse dieses Benchmarks dokumentiert. Die Referenzdesigns wurden auf den zum Zeitpunkt der FPGA-Evaluierung komplexesten Bausteine abgebildet, die von der jeweiligen Place&Route-Software unterstützt wurden. Die Angaben beziehen sich deshalb auf den Xilinx-FPGA XC4085XL-3 und Lucent-FPGA Or3T55-5. Die Referenzdesigns sind:

- Funktionsgeneratoren (in den Varianten 8 Bit, 16 Bit und 24 Bit),
  die jeweils eine Addition, Subtraktion, Multiplikation und Division (Integer) beinhalten,
- ein Fließkomma-Multiplizierer (28 Bit),
- Histogrammierer (32 x 8 Bit und 128 x 8 Bit),

- Dual-Port-RAM (16 x 32 Bit),
- Single-Port-RAM (16 x 32Bit).

Natürlich stellen diese Referenzdesigns eine repräsentative Auswahl dar. Dennoch beinhalten sie typische Teilaufgaben aus unterschiedlichen Bereichen, wie sie in der Praxis, bei den für das ATLANTIS-System vorgesehen Anwendungen, auftauchen.

	Xilinx XC4085XL-3		Lucent OR3T55-5	
	Frequenz/#LBs	P&R Zeit	Frequenz/#LBs	P&R Zeit
Funtionsgenerator: 8Bit	29 MHz, 105 CLBs	24 min	33MHz, 25 PLCs	8min
Funtionsgenerator: 16Bit	21 MHz, 347 CLBs	19h	20,4 MHz, 86 PLCs	2h 23min
Funtionsgenerator: 24Bit	Nach 3 Tagen ergebnislos abgebrochen		19 Mhz, 178 PLCs	4h 30min
28bit Fließkomma Multiplizierer	24 MHz, 515 CLBs	3,5h	21,6MHz, 134 PLCs	1h 45min
28bit Fließkomma Multiplizierer mit Core-Tools	47 MHz, 420 CLBs	4,5h	Keine Core-Tool Unterstützung für Multiplizierer	
Histogrammierer 32bit	56MHz, 112 CLBs	40min	52 MHz, 32 PLCs	11min 24sec
Histogrammierer 128bit	32MHz, 844 MHz	11h	41,5 MHz, 228 PLCs	1h 30min
DPRAM	140 Mhz, 18 CLBs	6min	128,5 MHz, 4 PLCs	3min
SPRAM	143 Mhz, 16 CLBs	7min 30sec	133,7 MHz, 4 PLCs	3min 45sec

Tabelle 4.3: Implementierungsergebnisse von FPGA-Referenzdesigns

Analysiert und vergleicht man die Meßergebnisse, stellt man bei den erzielten Designfrequenzen zwischen Lucent und Xilinx nur einen geringfügigen Unterschied fest. Auch der Verbrauch an Logikzellen ist ähnlich. Ein Vergleich läßt sich deshalb ziehen, weil die Logikzelle des Lucent-FPGAs (PLC) in etwa der vierfachen Komplexität einer Xilinx-FPGA Logikzelle (CLB) entspricht.

Ein wesentlicher Unterschied besteht allerdings in der Bearbeitungzeit der Place&Route-Tools. Hier sind bei der Lucent-Software bei allen Referenzdesigns deutlich kürzere Zeiten zu verzeichnen. Zusätzlich wurden von der Lucent-Software ein vorgegebenes Design verarbeitet, an dem die Xilinx-Software gescheitert ist.

Unter der Berücksichtigung der bei der Messung erzielten Ergebnisse und zusätzlich den (weniger objektiven) Angaben zur Leistungsfähigkeit aus den Datenblättern und *Application Notes* ist die Einsatz der Lucent-FPGAs denen des Herstellers Xilinx vorzuziehen.

Als Fazit dieser Evaluierung läßt sich folgendes festhalten: Die leistungsfähigsten FPGA-Technologien aller FPGA-Hersteller, die zum Zeitpunkt der Evaluierung (August 1998) zur Verfügung standen, wurden miteinbezogen. Aufgrund der wichtigsten Entscheidungskriterien sind zum einen die Xilinx-FPGAs als auch die vergleichsweise geringfügig leistungsfähigeren Lucent-FPGAs die geeignetsten für das neue ATLANTIS-System.

Weniger ein technisches Kriterium als ein wirtschaftliches Kriterium sind die Preise sowohl für die Bausteine als auch für die Baustein-spezifische Place&Route-Software. Ein Xilinx-FPGA XC125XV-3 bzw. XC4085XL-3 jeweils im BGA560-Gehäuse kostet DM 3500,- bzw. DM 2000,- (August 1998). Die zugehörige Software-Lizenz (Hochschulsonderpreis) für die Place&Route-Software schlägt mit DM 700,- zu Buche. Der Lucent OR3T125-5 im BGA600-Gehäuse kostet pro Stück DM 820,- (August 1998) und die Lucent Place&Route-Software als 5er-Hochschullizenzpaket umgerechnet DM 3500,-. Bei den Softwarepaketen ist kein großer Preisunterschied auszumachen, eine deutliche Preisdifferenz zu Gunsten von Lucent ist dagegen beim Bauteilpreis festzustellen.

Werden letzt endlich die Möglichkeiten der Place&Route-Software miteinbezogen fällt die Wahl zu Gunsten der Lucent-FPGAs aus. Die Entscheidung für die Lucent-FPGAs wurde in einem abschließenden Meeting von allen Beteiligten der FPGA-Evaluierung einstimmig getroffen.

4.5 Kopplung von FPGA-Modulen und Host-CPU

Für die Kopplung einer Host-CPU mit den FPGA-Resourcen kommen prinzipiell zwei Alternativen in Frage. Die Entwicklung einer stand-alone System-Architektur oder die Verwendung eines kommerziellen Bussystems.

Eigenentwicklung bedeutet einen sehr großen Entwicklungsaufwand, der nicht gerechtfertigt ist, insbesondere wenn man dies im Vergleich zur Verwendung eines kommerziellen Bussystems sieht. Der wesentliche Vorteil dieser Alternative ist eine Reihe von Komponenten inklusive einer Host-CPU kaufen und direkt verwenden zu können. Vom technischen Fortschritt industrieller Entwicklungen kann ohne eigenes Zutun profitiert werden. Die Eigenarbeit beim Aufbau des ATLANTIS-Systems beschränkt sich dadurch auf die Entwicklung der FPGA-basierten Module und die Auswahl von geeigneten kommerziellen Komponenten. Über das industrielle Standard-Bussystem kann die enge Kopplung von CPU und FPGA-Resourcen für die Ausführung von Algorithmen unter dem Koprozessoraspekt erreicht werden. Gleichzeitig erlaubt diese Schnittstelle Zugriffe zur essentiell notwendigen Systemkonfiguration und zur Kontrolle der FPGA-basierten Module. Zusätzlich muß aber die Möglichkeit bestehen, einen Privat-Bus zu integrieren, um eine Skalierbarkeit der FPGA-Resourcen unabhängig vom Systembus zu gewährleisten.

Aus den bisher gemachten Erfahrungen kommen dafür nur zwei Bussysteme in Frage, die in den folgenden beiden Abschnitten vorgestellt werden. Das sind der seit

Jahren etablierte VME-BUS und der seit Mitte der 90er Jahre eingesetzte CompactPCI-BUS.

4.5.1 Der VME-Bus

Der VME-Bus, ursprünglich von der Firma Motorola entwickelt und 1981 erstmals öffentlich vorgestellt, ist ein in der Forschung weit verbreiteter Backplane-Bus (Rückwand-Bus). Der VME-Bus war bewußt und absichtlich von Anfang an gänzlich offen. Als Multiprozessorsystem ausgelegt werden viele gängige (Echtzeit-) Betriebssysteme auf verschiedenen Prozessorarchitekturen unterstützt.

In einem Standard VME-Crate finden nebeneinander bis zu 21 VME-Platinen mit einfacher Breite Platz. In der Höhe variieren die Crates, so daß sowohl Platinen im Einfacheuropakartenformat (100 mm x 160 mm), als auch Platinen mit der doppelten Bauhöhe Verwendung finden. In Abbildung 4.5 ist ein Standard 6-HU-VME-Crate dargestellt. Daneben gibt es noch eine optionale 9-HU-Variante. Dieses Crate kann Platinen mit der dreifachen Bauhöhe einer Europakarte aufnehmen.

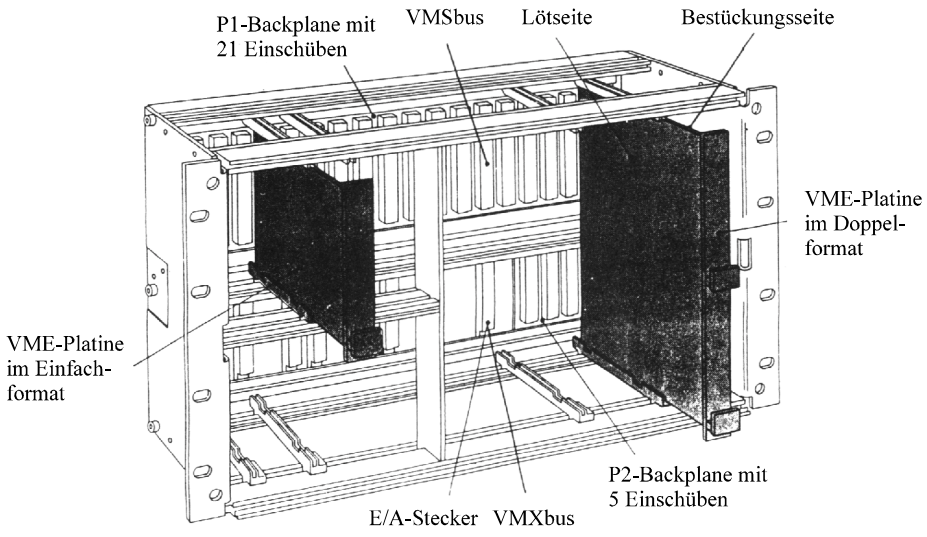


Abbildung 4.5: 6-HU-VME-Crate zur Aufnahme verschiedener VME-Platinen

Der VME-Bus verwendet insgesamt 188 Signalleitungen, wobei der Datenbus eine Breite von 32 Bit aufweist. Mechanisch wird ein Slot durch zwei übereinander angeordnete 96polige VG-Leisten (I/O-Stecker) realisiert. Für benutzerdefinierte Signale sind 64 Pins (die beiden äußeren Reihen) der oberen Stecker reserviert. Die Datenübertragung findet asynchron und ohne Multiplexing (Adress- und Datenleitungen sind getrennt) statt. Zwischen einem Master (Sender) und einem Slave (Empfänger) werden die Daten durch ein Handshaking-Protokoll übermittelt [VME87].

Die maximale Datenübertragungsrate bei der 32-Bit-VME-Version beträgt 40 MByte/s. In der Praxis ist dieser Wert allerdings nicht zu erreichen. Abhängig davon, ob Einzel- oder Blocktransfers durchgeführt werden, beträgt die Transferrate ca. 5 - 15 MByte/s [Wur96]. Seit Januar 1996 ist eine 64-Bit-Erweiterung des VME-Busses VME64 definiert. Bei gleicher Kontaktbelegung sieht diese Spezifikation einen gemultiplexten Daten-/Adress-Bus vor. Dadurch erhöht sich die maximale Datentransferrate auf 80 MByte/s. 1997 wurde der VME64-Extension-Standard als Erweiterung des VME64 spezifiziert. Wichtige Neuerungen sind dabei der neue 160polige Steckverbinder, die dadurch erstmals integrierte 3.3V Spannungsversorgung und die auf 160MByte/s erhöhte Datenübertragungsrate. Die Anzahl der User-I/Os erhöht sich um 141. Mit der Hot-Swapping-Funktionalität wird eine für die Reduktion der Ausfallzeit (Erhöhung der Verfügbarkeit) wichtiger Standard eingeführt. Hot-Swapping besagt, daß während des Betriebs Karten aus dem System entfernt und hinzugefügt werden können [VME97].

Seit 1997 haben gleichzeitig die Aktivitäten zugenommen, die Datenübertragungsrate über den Bus zu erhöhen. Zum Beispiel wurde eine modifizierte VME-Bus Architektur VME320 veröffentlicht, die eine Datenrate von 320 MByte/s ermöglichst. Erreicht wird dies durch ein neues Design der Backplane und einem geänderten Bus-Protokoll[Ryn98].

4.5.2 Der CompactPCI-Bus

Im Desktop-Bereich hat sich der synchrone, prozessorunabhängige PCI-Bus (*Peripheral Component Interconnect*) seit seiner Einführung 1991 als führende Bustechnologie bereits etabliert. Mit einer Datenbandbreite von 132 MByte/s bei 32 Bit und 264 MByte/s bei 64 Bit bei jeweils 33 MHz[46], ist der Peripherie-Bus PCI das Herz nicht nur von Desktop-PCs auf der Basis von Intel Pentium und AMD, sondern vieler moderner Architekturen wie Motorola PowerPC oder Alpha von Digital Equipment [PCI95].

Die Einführung von CompactPCI portiert den PCI-Bus hinsichtlich des Formfaktors, der Steckverbinder, der Mechanik und anderer Eigenschaften in die Welt der Industriebusse. Es sollten die Vorteile von PCI und der dafür vorhandenen breiten Softwarepalette mit den Vorteilen einer robusten 19-Zoll-Technik mit 3-HU-/6-HU-Kartenformat kombiniert werden. Dieses Ziel wurde von der 1994 gegründeten *PCI Industrial Computer Manufacturer Group* (PICMG) verfolgt und mit der Veröffentlichung der CompactPCI-Spezifikation erfüllt [Com97].

Mit CompactPCI steht als Ergebnis heute ein industrietauglicher PCI-Bus zur Verfügung; elektrisch basierend auf Standard-PCI, aber im robusten Einfach- (3 HU) oder Doppeleuropakartenformat (6 HU) mit einer Backplane, die bis zu 8 Steckplätze (Slots) unterstützt. Die postulierten 8 Slots stellen aber keine harte Grenze dar. Mittels PCI-Brücken-Bausteine (PCI-Bridge-Chips) können mehrere CompactPCI-Backplanes effizient und transparent verknüpft werden.

[46]Für die zukünftige CompactPCI -Spezifikation ist die Erhöhung des Systemtakts auf 66 MHz vorgesehen, womit die jeweils doppelten Transferraten erreicht werden können.

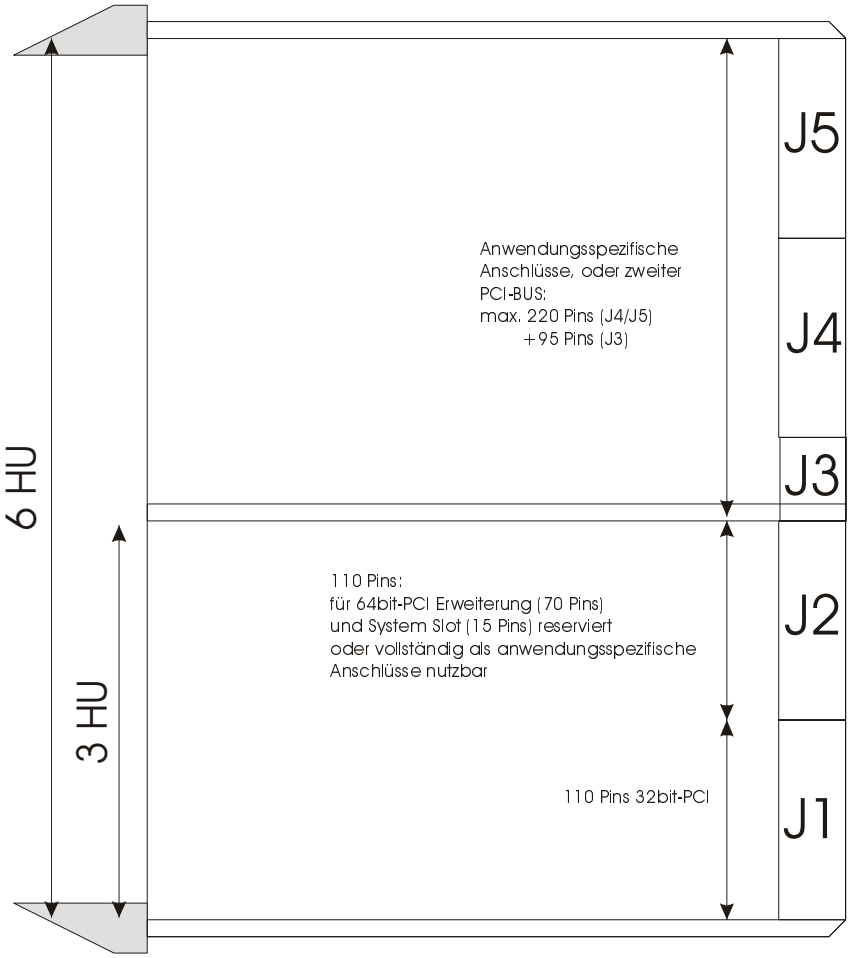


Abbildung 4.6: CompactPCI 6-HU-Board mit Steckerdefinition

In Abbildung 4.6 sind ein 3-HU- bzw. 6-HU-CompactPCI-Board mit zugehöriger Steckerdefinition dargestellt. Ein 3-HU-Board hat demnach zwei 110polige Steckverbinder (J1 und J2), wobei der Stecker J1 standardmäßig mit dem 32-Bit-PCI-Bus-Anschlüssen belegt ist. Bei der 64-Bit-Version werden weitere 70 Pins des J2 belegt, 15 Pins sind für den System-Slot reserviert und 25 Pins des J2 können anwendungsspezifisch definiert werden.

Ein 6-HU-Board ist um die Stecker J3 bis J5 erweitert, die es ermöglichen einen zweiten (32 Bit oder 64 Bit) PCI-Bus zu integrieren oder zusätzlich 315 anwendungsspezifische Pins zur Verfügung zu stellen. Im einem 32-Bit-Umfeld kann damit theoretisch, mit Ausnahme des Systemslots, auf insgesamt 425 benutzerdefinierbare Pins zurückgegriffen werden. Allerdings bedingt dies die Konstruktion einer zusammenhängenden 6-HU-Backplane.

Bei der Verwendung einer kommerziellen 3-HU-Backplane, fällt der Stecker J3 zu Gunsten einer mechanischen Halterung weg. Für die Implementierung des Privat-Busses auf einer zweiten, unabhängigen Backplane verbleiben die Stecker J4 und J5 und somit 220 Pins.

Zusammenfassend sind nachfolgend die wesentlichen Vorteile und wichtigsten Eigenschaften von CompactPCI aufgelistet:

- Prozessorunabhängiger PCI-Bus mit einer Datenrate bis zu 264 MByte/s (zukünftig 512MByte/s), 33 MHz synchrone Taktrate, Burstbetrieb möglich; 5V- und 3.3-V-Technologie werden unterstützt.
- Vorteile für die Softwareerstellung durch die Verfügbarkeit von vielen fertigen Programmen unter den verschiedenen Betriebssystemen für die weit verbreitete PCI-Technologie. Unterstützt werden Windows95/98/NT/2000, Unix, Linux, LynxOS und andere.
- Hochintegrierte Steckverbinder im 2mm Raster ermöglichen die 64 Bit Erweiterung - im Gegensatz zum VME-Bus - bereits bei den sehr kompakten 3-HU-Boards und bieten darüber hinaus noch 25 benutzerdefinierbare Pins. Im 6-HU-Format stehen zusätzlich bis zu 425 benutzerdefinierbare Pins zur Verfügung.
- Hot-Swapping durch voreilende Spannungskontakte ist in der Spezifikation vorgesehen; erlaubt das Einstecken und Ziehen von Modulen im Betrieb
- Von einer Vielzahl von Firmen unterstützter Industriestandard sichert breites Produktspektrum und Langzeitverfügbarkeit.
- Bis zu 8 CompactPCI-Steckplätze; Brücken zu weiteren CompactPCI-Backplanes (für 15 oder mehr Slots in 19-Zoll-Racks) sind realisierbar.

4.5.3 Vergleich und Diskussion

Für die Umsetzung des Konzepts eines hybriden CPU/FPGA-basierten Systems wird ein kommerzielles Basis-Bussystem gesucht, daß einerseits eine direkte Nutzung einer Vielzahl kommerzieller Komponenten u. a. einer *up-to-date* Host-CPU ermöglicht und andererseits die Integration und Skalierbarkeit FPGA-basierter Module einfach und effizient ermöglicht.

Unter diesem Gesichtspunkt sind für die Nutzung eines kommerziellen Bussystems für ATLANTIS folgende Punkte essentiell und müssen erfüllt sein:

- Bereitstellung eines leistungsfähigen Interfaces zwischen Host-CPU und FPGAs zur Unterstützung des Koprozessorkonzepts.
- Möglichkeit der Integration eines leistungsfähigen Privat-Busses, um die FPGA-basierten Module effizient koppeln zu können - Skalierbarkeit der Rechen- und I/O-Leistung.
- Unterstützung von den Betriebssystemen Windows NT und Linux, wie sie für die CERN Aktivitäten und in anderen Bereichen gefordert werden.

- Langfristige Verfügbarkeit kommerzieller Prozessorkarten (Embedded PCs) mit aktueller Technologie.
- Verfügbarkeit von Netzwerkkarten (speziell Ethernet 100MBit, 1GBit und ATM) und weiterer Peripherie, entweder stand-alone oder in die Prozessorkarte integriert.
- Die Integration der FPGA-basierten Module und des Privat-Busses soll mit minimalem Aufwand durchgeführt.

Vergleicht man die Möglichkeiten des VME-Bus und von CompactPCI in Bezug auf die Anforderungsliste, können die genannten Punkte von beiden Systemen gleichermaßen erfüllt werden. Nur bezüglich des letzten Punktes besteht beim VME-Bus ein Mehraufwand, weil für die Integration des Privat-Busses die Entwicklung einer Backplane, die VME-Bus und den Privat-Bus kombiniert unumgänglich ist. Dies liegt daran, weil beim VME-Bus User-I/Os und Systemsignale über den selben physikalischen Steckverbinder geführt sind. Im Gegensatz dazu kann im Fall von CompactPCI eine kommerzielle Systemplatine in der unteren Hälfte des 6-HU-Crates eingesetzt werden und für die obere Hälfte in Eigenentwicklung ein Privat-Bus implementiert werden, für den 220 I/O-Pins zur Verfügung stehen.

Ein weiteres wichtiges Kriterium bei der Wahl des Basis-Bussystems liegt in den bereits gemachten Erfahrungen mit PCI. In der Gruppe wurde bereits 1997 eine Koprozessorkarte auf PCI-Basis entwickelt, mit der unter anderem das TRT-Eventdatenvorverarbeitung für den ATLAS ROB-Task untersucht wurde. Diese Karte wird mittlerweile von der Spin-off-Firma Silicon Software, die aus dem Institut heraus gegründet wurde, kommerziell angeboten. Die Beschäftigung mit dieser Karte und das Know-How, insbesondere bei der Anwendung des PCI-Schnittstelle, stellen einen großes Potential für das ATLANTIS-System dar.

Nicht zuletzt wegen dieser Tatsache ist die Verwendung des CompactPCI dem VME-Bus vorzuziehen. Im Rahmen einer Kooperation wurde deshalb ein Know-How-Austausch bei der Entwicklung des ATLANTIS-Systems und zukünftiger Produkte der Firma vereinbart.

4.6 Das ATLANTIS-Systemkonzept

In diesem letzten Abschnitt des vierten Kapitels wird die Konzeptionierung des ATLANTIS-Systems im Überblick dargestellt.

Die bisherigen Ergebnisse dieses Kapitels können folgendermaßen zusammengefaßt werden: Es wurde für das ATLANTIS-System eine geeigneter FPGA-Baustein und das Basis-Bussystem gefunden. Weiterhin wurde festgestellt, daß Speicherbausteine unterschiedlicher Art in die FPGA-basierten Module integriert werden müssen. Dies kann durch die Verwendung von steckbaren Speicher-Modulen realisiert werden. Andererseits sollen zusätzlich verschiedene I/O-Schnittstellen integrierbar sein (z.B. S-Link/M-Link für die Untersuchung von FPGA-basierten Implementierungen von

ATLAS Readout-Systemen). Auch dies kann über entsprechende I/O-Modulsteckplätze erfolgen.

Grundsätzlich wurde die Entwicklung eines einzigen FPGA-basierten Karte, die die genannten Steckplätze für I/O- , Speicher- und sonstige Logik-Module beinhaltet, in Betracht gezogen. Unter Berücksichtigung der physikalischen Dimensionen des CompactPCI-Formfaktors wurde aber der Aufbau einer einzelnen 6-HU-Karte als FPGA-Modul verworfen. Dies wäre mechanisch und elektrisch nur schwer oder gar nicht zu realisieren gewesen. Mit der Entwicklung zweier FPGA-basierter Karten, eine speziell für I/O-Anwendungen mit entsprechenden I/O-Modulsteckplätzen und eine für die eigentliche Algorithmenausführung mit entsprechenden Möglichkeiten zur flexiblen Integration von Speicher und sonstiger Logik besteht zudem die Möglichkeit, mit der Bauhöhe im Rahmen der Spezifikation eines 1-Slot-Boards zu bleiben. Dies hat den Vorteil in einem Standard-8-Slot-CompactPCI-System sieben solcher Karten inklusive Aufsteckkarten unterzubringen. Der achte Slot ist reserviert für die Host-CPU. In Abbildung 4.7 ist in einer Übersicht die Konzeptionierung des ATLANTIS-Systems skizziert.

Abschließend läßt sich sagen, daß in diesem Kapitel alle relevanten Punkte für die Entwicklung des neuen hybriden FPGA/CPU-basierten ATLANTIS-Systems erarbeitet und diskutiert wurden. Diese Kapitel endet mit dem konzeptionellen Entwurf des Systems unter Berücksichtigung der geplanten Anwendungen. Im folgenden Kapitel 5 wird die Umsetzung des Konzepts und damit die Realisierung des vollständigen ATLANTIS-Systems beschrieben.

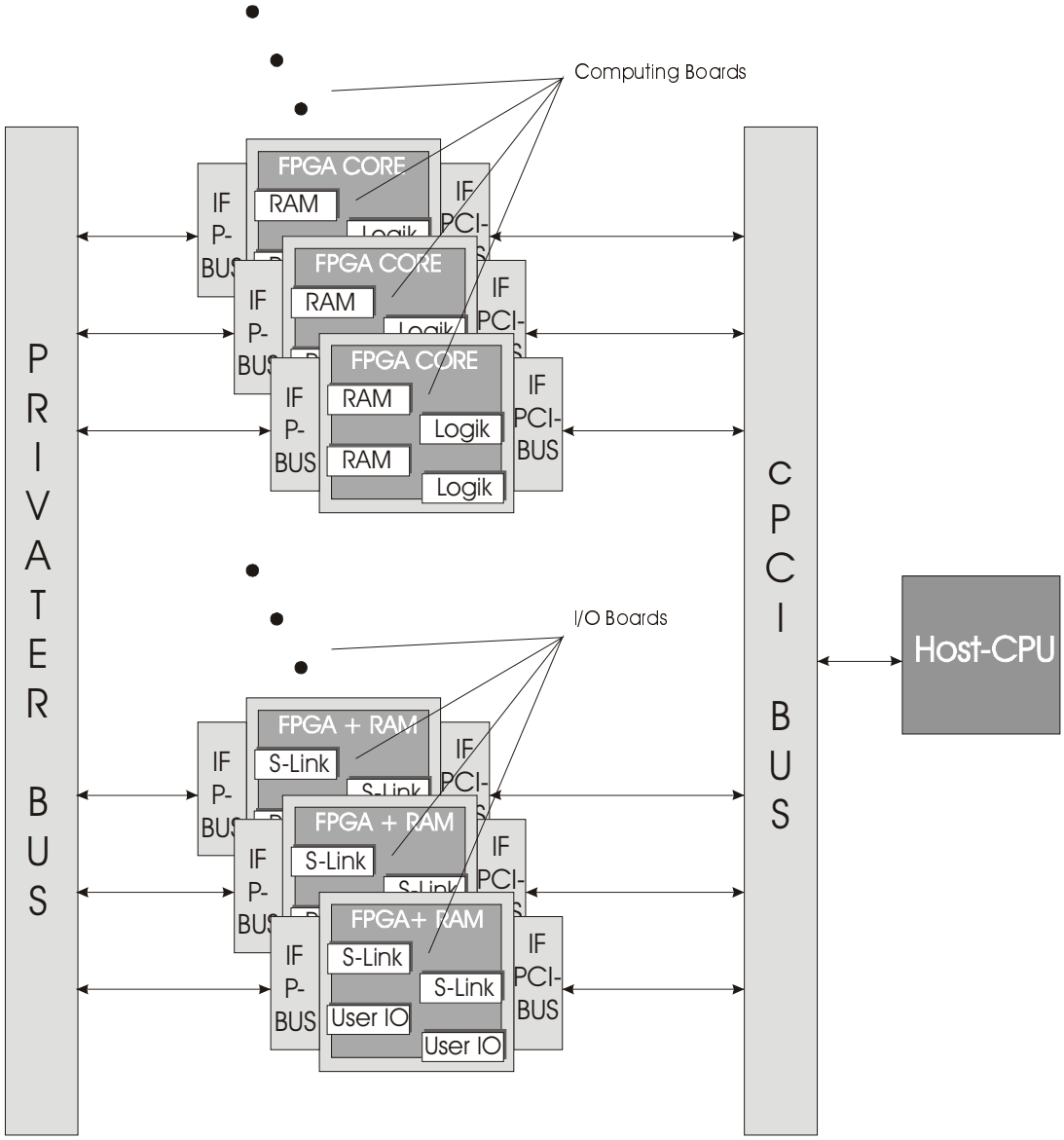


Abbildung 4.7: Das Konzept für das hybride FPGA/CPU-basierte Prozessorsystem ATLANTIS

KAPITEL 5

Implementierung des ATLANTIS-Systems

Im vorangegangen Kapitel wurde das grundlegende Konzept für die Entwicklung des ATLANTIS-Systems auf der Basis der Anforderungen einer Reihe von Anwendungen erarbeitet, die für die Ausführung auf dem neuen System vorgesehen sind. In diesem Kapitel wird die Umsetzung des Konzepts und damit die Realisierung des Systems im einzelnen beschrieben. Als Basis-Bussystem wurde der industrielle CompactPCI-Bus ausgewählt. Als kommerzielle Komponenten können sowohl eine Auswahl von CompactPCI-CPU-Karten als auch die System-Backplane hinzugekauft werden. Auf diese Komponenten wird daher nur am Rande eingegangen. Im Vordergrund stehen in diesem Kapitel die Beschreibung der entwickelten FPGA-basierten Module. Das sind das ATLANTIS Computing-Board, das ATLANTIS I/O-Board und der ATLANTIS Privat-Bus.

Im ersten Abschnitt wird ein Überblick über das Gesamtsystem gegeben, modulübergreifende Eigenschaften besprochen und auf die Entwicklung und den Test der Boards eingegangen. In den weiteren Abschnitten folgen die Detailbeschreibungen der drei Hauptkomponenten des FPGA-basierten Teils von ATLANTIS. Das Kapitel endet mit der Diskussion über die Möglichkeiten von Software-Tools zur Anwendungsentwicklung aber auch zur Konfiguration und Kontrolle des Systems, wie sie für ATLANTIS bereits entwickelt wurden bzw. bisher schon konzipiert worden sind.

5.1 Das ATLANTIS-System im Überblick

Dieser Abschnitt beginnt mit einer Darstellung des Systems im Überblick. Modulübergreifende funktionale Einheiten, wie die implementierte PCI-Schnittstelle und das globale Clocksystemkonzept werden im weiteren vorgestellt. Abgeschlossen wird der Abschnitt mit Bemerkungen zu den einzelnen Entwicklungsschritten - von der Schaltplaneingabe bis zur gefertigten Platine.

5.1.1 Gesamtsystemarchitektur

ATLANTIS wurde als universell verwendbarer, hybrider FPGA/CPU Prozessor auf der Basis von CompactPCI in erster Linie für den Einsatz als Triggerprozessor für das geplante ATLAS-Experiment am CERN konzipiert. Daneben gibt es eine Reihe weiterer konkreter Einsatzgebiete und Anwendungen, deren Ausführung auf dem ATLANTIS-System vorgesehen sind. Der Kernpunkt, um die optimale Adaption eines solchen Systems für verschiedene Anwendungen zu gewährleisten, ist Modularität auf verschiedenen Ebenen. Für ATLANTIS wurde entsprechend des erarbeiteten Konzepts (Kapitel 4) ein dreistufig modularer Aufbau gewählt (Abbildung 5.1).

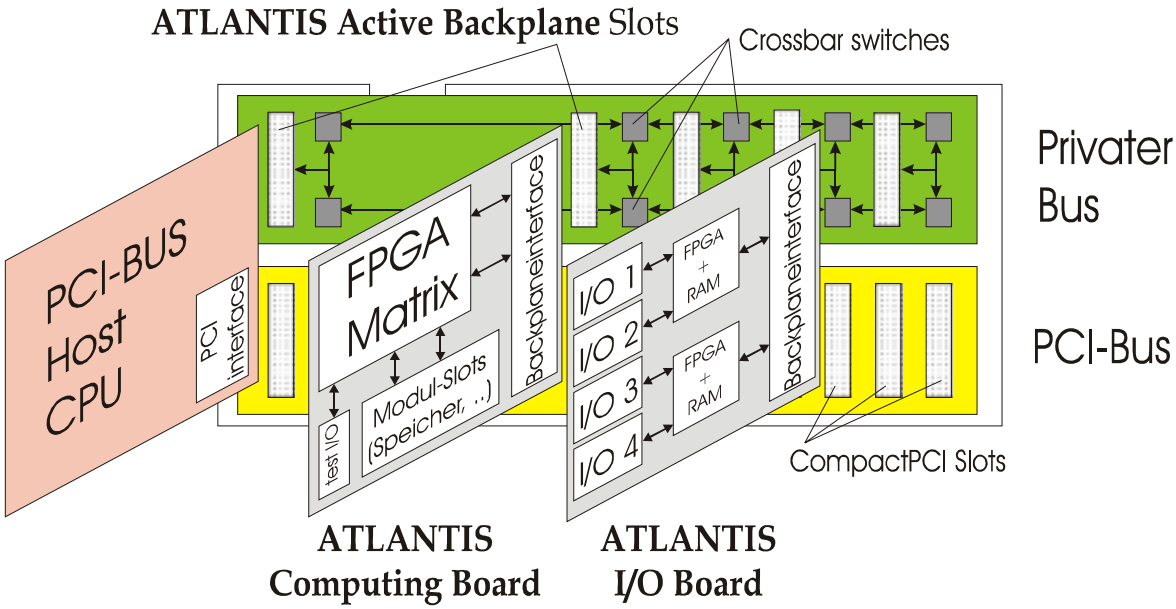


Abbildung 5.1: Das ATLANTIS-System im Überblick

Auf oberster Ebene wird dieses Konzept durch die Aufteilung in die beiden Hauptkomponenten, den FPGA-basierten Teil und die Host-CPU, verwirklicht. Diese Architektur erlaubt es, eine Applikation so aufzuteilen, daß die verschiedenen Teile auf der für sie geeigneten Plattform ausgeführt werden. Mittels PCI-Bus wird die enge Verknüpfung der Applikationsteile erreicht.

Weiterhin gibt es innerhalb des FPGA-basierten Teils eine Aufteilung in drei Module. Ein Modul, das sogenannte ATLANTIS Computing-Board (ACB) beinhaltet den FPGA-Prozessor-Kern und ist für Ausführung der eigentlichen Algorithmen vorgesehen. Ein zweites Modul, das ATLANTIS I/O-Board (AIB) wurde speziell für I/O-intensive Aufgaben entwickelt. Eine beliebige Anzahl[47] und Auswahl aus ACB-

[47]Die maximale Anzahl von Karten wird ausschließlich durch die verfügbare Anzahl von Slots in einem Crate beschränkt.

und AIB-Modulen läßt sich innerhalb von einem CompactPCI-Crate kombinieren. Eine leistungsfähige Kommunikation zwischen ACBs und AIBs ist durch die Implementierung des Privat-Busses auf der ATLANTIS Active-Backplane (AAB) gegeben, das dritte wichtige Modul des FPGA-basierten Teils von ATLANTIS. Hauptvorteil der Verwendung dieses zusätzlichen Kommunikationskanals ist, daß die Rechen- und I/O-Leistung skaliert werden können - und das unabhängig von der Leistungsfähigkeit des PCI-Busses. Die Nutzung des PCI-Busses bleibt voll und ganz für seine eigentliche Aufgabe reserviert, nämlich dem Datenaustausch zwischen den beiden FPGA-basierten Modulen ACB und AIB auf der einen Seite und der Host-CPU auf der anderen Seite.

Die unterste Ebene wird durch den modularen Aufbau der beiden Hauptkomponenten, dem ACB und dem AIB repräsentiert. Das ACB ist mit Modulsteckplätzen ausgestattet, die es erlauben eine beliebige Art und Architektur von zusätzlicher Logik zu integrieren. Im Vordergrund dabei steht die Integrations von den Speicher-Bausteinen. Das AIB verfügt über frei programmierbare Schnittstellen. Diese sind frei konfigurierbar und unabhängig von bestimmten Schnittstellen-Standards.

Innerhalb der folgenden Abschnitte werden die ATLANTIS-Systemkomponenten im einzelnen dargestellt. Zuvor werden wichtige modulübergreifende Implementierungen beschrieben. Das sind das PCI-Interface und das globale Clocksystem-Konzept.

5.1.2 Das PCI-Interface

Zentrales Element für die Kopplung von Host-CPU und FPGA-Resourcen ist der PCI-Bus. Der PCI-Bus wurde ursprünglich zur Entlastung der CPU entwickelt. Es besteht eine Kopplung über einen Bridge-Chip (Chipsatz) zwischen Prozessor-Bus (L2 in Abbildung 5.2) auf der einen und Speicher- und PCI-Bus auf der anderen Seite. Alle Busse sind elektrisch voneinander entkoppelt, so daß unterschiedliche Protokolle und eine separate Taktung ermöglicht wird. Ein Vorteil dieser Architektur ist, daß gleichzeitige, unabhängige Transfers zwischen verschiedenen Komponenten möglich sind [BSC98].

Abbildung 5.2 zeigt schematisch den typischen Aufbau eines PCI-Rechners mit integrierten ATLANTIS-Modulen. Daneben ist die Anbindung der FPGA-Karten (ATLANTIS-Boards) über den PCI-Interface-CHIP (PLX) dargestellt. Zum besseren Verständnis ist zusätzlich der Privat-Bus des ATLANTIS-Systems mit eingezeichnet, der in Abschnitt 5.4 besprochen wird.

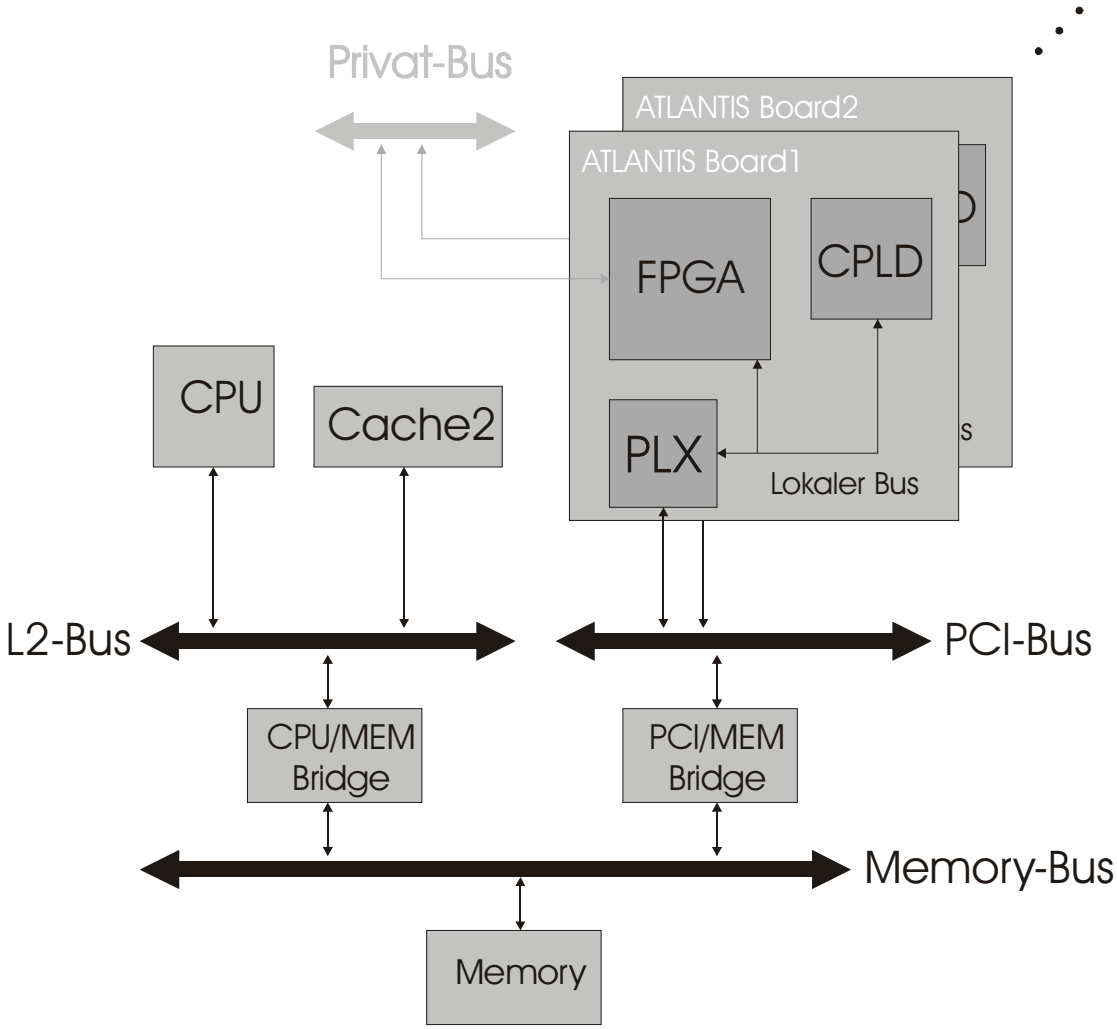


Abbildung 5.2: Schematischer Aufbau eines PCI-Rechners nach [BSC98] mit integrierten ATLANTIS-Modulen

Am PCI-Bus gibt es einen oder mehrere Busmaster und einen oder mehrere Slaves. Da es mehr als einen Master geben kann, ist für die Koordinierung der Zugriffe an zentraler Stelle ein Arbiter vorhanden. Die Datenübertragung erfolgt zwischen einem Initiator (Master) und einem Target (Slave). Alle Transaktionen auf dem Bus bestehen aus einem Adreßzyklus und einem oder mehreren Datenzyklen. Die Übertragung eines aus mehreren Datenwörtern bestehenden Datenpakets wird als Burst-Transfer bezeichnet [PCI95].

Jeder PCI-Bus Teilnehmer benötigt eine lokale PCI-Bus-Kontrolllogik. Dieses sogenannte PCI-Bus-Interface wird von verschiedenen Firmen in Form eines PCI-Schnittstellen-Bausteins angeboten. Daneben sind auch kommerzielle PCI-Schnittstellen als Core-Lösung für FPGAs erhältlich. Für das ACB und AIB

wurde aus mehreren Gründen der Interface-Chip PCI9080 der Firma PLX [PLX98] gewählt:

1. Dieser Chip wird sehr erfolgreich auf dem FPGA-Koprozessor microEnable der Firma Silicon Software eingesetzt. Über den vereinbarten Kooperationsvertrag kann die mehrjährige Erfahrung der Firma bei der Arbeit mit diesem Chip für das ATLANTIS-Projekt genutzt werden.
2. Zu diesem Baustein ist eine sehr gut dokumentierte und umfangreiche Entwicklungsumgebung erhältlich. Insbesondere ist der Quellcode des Treibers verfügbar.
3. Als spezielle Eigenschaft enthält dieser Baustein zwei vollständige DMA-Kanäle, die Grundvoraussetzung für einen effizienten und leistungsfähigen Datentransfer zwischen FPGAs und Host-CPU[48].

Integriert auf den ATLANTIS-Modulen wird der PCI9080 auf der lokalen Seite im sogenannten J-Mode betrieben. D.h. Daten (32-Bit) und Adressen (32-Bit) werden zeitmultiplext auf einem gemeinsamen 32-Bit-breiten physikalischen Bus zu den FPGAs und den CPLDs hin übertragen.

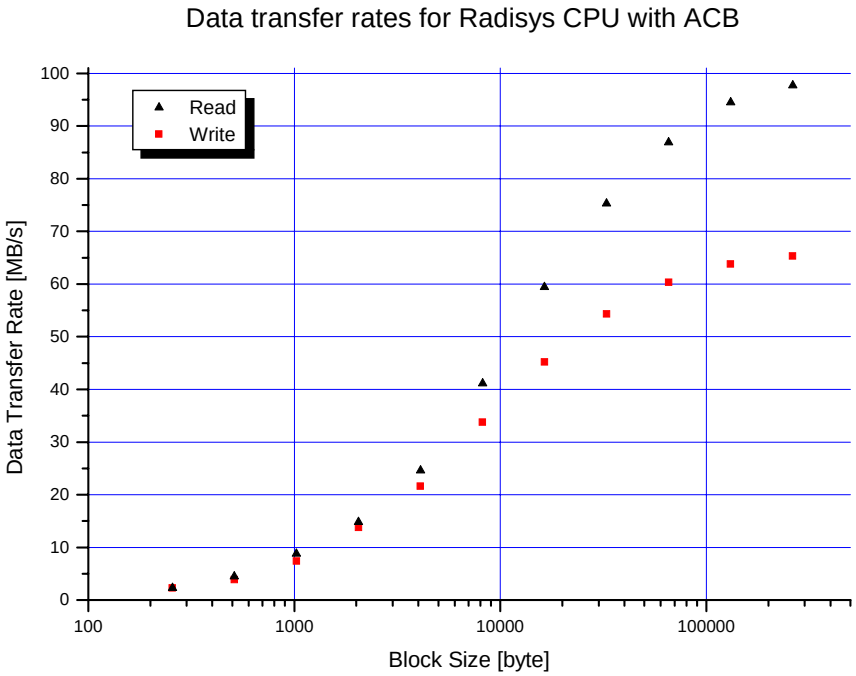
Die Schnittstelle zu den FPGAs wird für die eigentliche Algorithmenausführung genutzt. Wichtig für die hier vorliegende Arbeit sind die erreichbaren Datenraten. Der PCI-Bus (32MHz, 32-Bit) wird grundsätzlich mit 132 MByte/s angegeben, die aber nur theoretisch, bei der Übertragung eines unendlich großen Speicherblocks erreicht werden können. In der Praxis ist daher, abhängig von der Paketlänge, typischerweise eine maximale Datenrate von 110-120 MByte/s möglich. Zuverarbeitende Daten werden aus Performance-Gründen blockweise per DMA[49] in den FPGA geschrieben (Write-DMA) und Ergebnisdaten ebenfalls per DMA zurück in den Host-Rechner übertragen (Read-DMA).

Zur Feststellung der tatsächlich erreichbaren Datenraten wurden verschiedene Messungen durchgeführt. Dazu wurde in dem PCI-Schnittstellen-FPGA des ACBs, neben dem PCI-Slave-Interface ein RAM-Block implementiert und die Übertragungszeiten zum Host-Rechner gemessen. Das Testprogramm, daß auf der Host-CPU ausgeführt wird, schreibt und liest dazu Datenblöcke unterschiedlicher Größe, beginnend von 256 Byte bis 2 MByte. In Abbildung 5.3 ist die mit dem ATLANTIS-System (ACB) gemessene DMA-Datenrate (Read und Write) in Abhängigkeit von der Datenblockgröße dargestellt. Die dargestellten Messungen wurden unter Windows NT 4.0 mit zwei kommerziellen CompactPCI-Host-Prozessorsystemen der Hersteller Radisys (200MHz-Pentium-System mit 128 MByte Hauptspeicher) und SBS (400MHz-Pentium2-System mit 256 MByte Hauptspeicher) durchgeführt. Die Lokal-Bus-Frequenz war auf 40 MHz eingestellt.

---

[48]Für das ATLANTIS-System wurde zunächst der PLX-Chip PCI9054 in Betracht gezogen, weil dieser speziell für den Einsatz in CompactPCI-Systemen entwickelt wurde (z.B.wird Hot-Swapping unterstützt). Im Vergleich zum PCI9080 werden dafür andere wichtige Features, wie z.B. die Bereitstellung von zwei vollwertigen DMA-Kanälen, nicht unterstützt.

[49]DMA: Direct Memory Access.



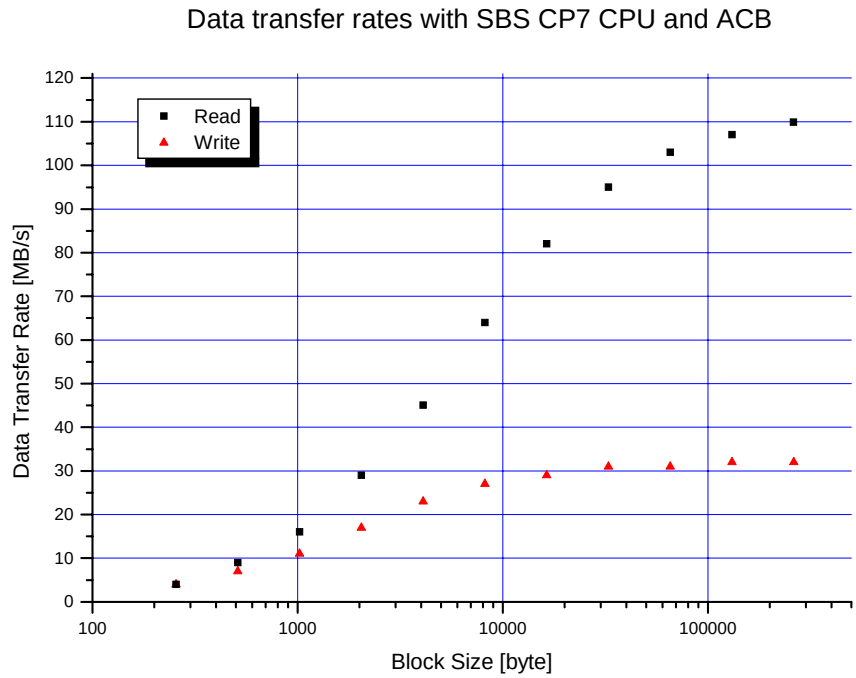


Abbildung 5.3: DMA-Datenraten (Read/Write) in Abhängigkeit von der Datenblockgröße

Die Meßergebnisse lassen sich wie folgt interpretieren. Beide Systemvarianten zeigen eine unterschiedliche DMA-Performance, sowohl beim Read als auch beim Write. Insbesondere die Messungen der Write-Datenraten beim SBS-System zeigen eine sehr schlechte Performance, die in der Größenordnung liegt, die auch mit sequentiellen Einzelzugriffen erreicht werden kann. Aber auch die Write-Datenraten beim Radisys-System liegen unter den Erwartungen. Zu den Read-Datenraten läßt sich sagen, daß auch hier die Datenraten zwar grundsätzlich höher ausfallen, aber trotzdem noch signifikant unter den PCI-Möglichkeiten zurückbleiben.

Zum Vergleich wurden unter den gleichen Randbedingungen die Messungen mit dem kommerziellen microEnable FPGA-Koprozessor durchgeführt. Dazu wurde das microEnble-System mit Hilfe eines PCI-to-CompactPCI-Adapters in die beiden CompactPCI-Crates integriert. Die Messungen mit dem microEnable System haben die schlechte Performance des SBS-Systems beim Read-DMA bestätigt. Gleichzeitig konnten auch die anderen Meßergebnisse reproduziert werden.

Aus Messsungen mit dem microEnable, ausgeführt mit herkömmlichen Desktop-PCs, ist bekannt, daß DMA-Transferraten bis zu 120 MByte/s grundsätzlich möglich sind. Daraus läßt sich schlußfolgern, daß der Grund für die schlechte Performance auf die CompactPCI-Prozessor-Systeme zurückzuführen ist. Aus Rücksprachen mit dem technischen Support der Hersteller konnte die Ursache für die schlechten Raten bisher noch nicht gefunden werden. Gespräche zur Lösung des Problems werden zur Zeit noch geführt[50]. Für die Messungen des Full-Scan-TRT-Algorithmus auf dem ATLANTIS-System wurde aufgrund der hier beschriebenen Messungen das Radisys-System eingesetzt (Kapitel 6).

5.1.3 Konfiguration und Initialisierung

Die Initialisierung und Konfiguration des ATLANTIS-Systems erfolgt ebenfalls über die PCI-Schnittstelle. Auf jedem ACB und AIB ist dafür lokal eine Kontrolllogik in einem CPLD (AIB) bzw. zwei CPLDs (ACB) implementiert. Bei den CPLDs handelt es sich um Bausteine der Firma Xilinx vom Typ XC95144XL und XC95288XL [Xil00]. Von den CPLDs aus kann das System vollständig kontrolliert werden. Hauptaufgaben, die über den CPLD ausgeführt werden, sind dabei:

- Konfiguration des lokalen und globalen Clocksystems
- Konfiguration der FPGAs
- Board-Test per JTag/Boundary Scan (vgl. Abschnitt 5.1.6)
- Konfiguration und Kontrolle der Modul- und I/O-Ports
- Steuerung des Systems im Debugmodus (Einzelschritt und Burstbetrieb)

[50]Dies scheint ein spezielles Problem der beiden derzeit in Mannheim eingesetzten CompactPCI-CPU-Systeme zu sein. Mit einem Demonstrations-Aufbau wurde von der Firma Hartmann gezeigt, daß über 11 CompactPCI-Teil-Systeme hinweg eine Transferrate von über 90 MByte/s erreicht werden kann [Mül99].

- Konfiguration des Privat-Busses (AAB)

Beim Systemstart wird zuerst per Software vom Host-Rechner aus die aktuelle Systemkonfiguration festgestellt. Zu diesem Zweck besitzt jede Komponente des ATLANTIS-Systems ein EEPROM, in dem die Informationen über das jeweilige Modul abgelegt sind. Ist dies geschehen, wird ein Master AIB oder ACB bestimmt, über das die AAB konfiguriert und kontrolliert wird[51]. Daran anschließend folgt eine Grund-Initialisierung und ein grundlegender Funktionstest. Danach ist das System zur Ausführung von Anwendungen bereit. Dazu müssen letzt endlich anwendungsabhängig die FPGA-Designs geladen und das Clocksystem und optional die Modul- und I/O-Ports konfiguriert werden.

Ein interessanter Punkt hierbei ist, mit welcher Geschwindigkeit die FPGAs (re-)konfiguriert werden können. Im schnellsten Konfigurationsmode (8-Bit-parallel) liegt die Konfigurationszeit sowohl für ein ACB als auch für ein AIB in der Größenordnung von mehreren 10 ms. Dies ist deswegen interessant, weil in den letzten Jahren in zunehmenden Maße Multitasking auf rekonfigurierbarer Hardware untersucht wird. Die Grundvoraussetzung für Multitasking auf diesen Plattformen ist die im Vergleich zur Ausführungszeit schnelle Rekonfiguration der Hardware. Dies wurde bereits bei der FPGA-Wahl für ATLANTIS (vgl. Abschnitt 4.4.4) mit berücksichtigt. Weitere Informationen finden sich in [SLM00].

5.1.4 Das ATLANTIS-Clocksystem

Die Konzeption und Realisierung des Clocksystems stellt einen sehr wichtigen und mit den zeitaufwendigsten Teil bei der Entwicklung eines FPGA-Prozessors dar. Vor der Konzeption ist zunächst zu klären, welche maximale Systemfrequenz unterstützt werden soll. Aufgrund der Erfahrungen bei der Synthese von Referenzdesigns für die Lucent Orca FPGAs, sind typischerweise Designfrequenzen in der Größenordnung von maximal 40 MHz bis 50 MHz zu erwarten. Für die mittlerweile verfügbaren Virtex-FPGAs des Herstellers Xilinx sind typischerweise Frequenzen bis 60 MHz möglich [Sim99]. Ausgehend von dieser Abschätzung und unter Berücksichtigung, daß in Zukunft noch schnellere FPGAs verfügbar sein werden, wurde das System auf eine maximale Frequenz von 80 MHz ausgelegt. Damit in der Praxis dieser Systemtakt umgesetzt und angewandt werden kann, müssen die verwendeten Bauteile für diese Frequenz ausgelegt sein. Dies wurde durch Vergleich und Analyse der in den Datenblättern angegeben Timingparameter kontrolliert und bestätigt.

Grundlegende Eigenschaft eines idealen Clocksystems für einen FPGA-Prozessor ist es, eine weitgehend synchrone und phasengleiche Taktung des Gesamtsystems zu unterstützen. Dies vereinfacht ganz allgemein die Anwendungsentwicklung, weil dadurch weniger Randbedingungen bezüglich eines unterschiedlichen Zeitverhaltens von Teilschaltungen zu berücksichtigen sind. Allerdings ist dies in der Praxis

[51] Die Konfigurationslogik für das AAB ist sowohl auf dem ACB als auch auf dem AIB implementiert.

nur durch einen sehr komplexen und aufwendigen Systemaufbau möglich und wie die Arbeit mit Enable++ gezeigt hat, für die optimale Ausführung von Algorithmen nur bedingt notwendig.

Aus diesem Grund wurde für das ATLANTIS-System als Kompromißlösung ein möglichst einfacher und praktisch nutzbarer Aufbau favorisiert, der dennoch grundlegende Voraussetzungen für eine optimale Taktung des Gesamtsystems beinhaltet. Die globale Sicht auf das ATLANTIS-Clocksystem auf Systemebene gibt Abbildung 5.4 wieder.

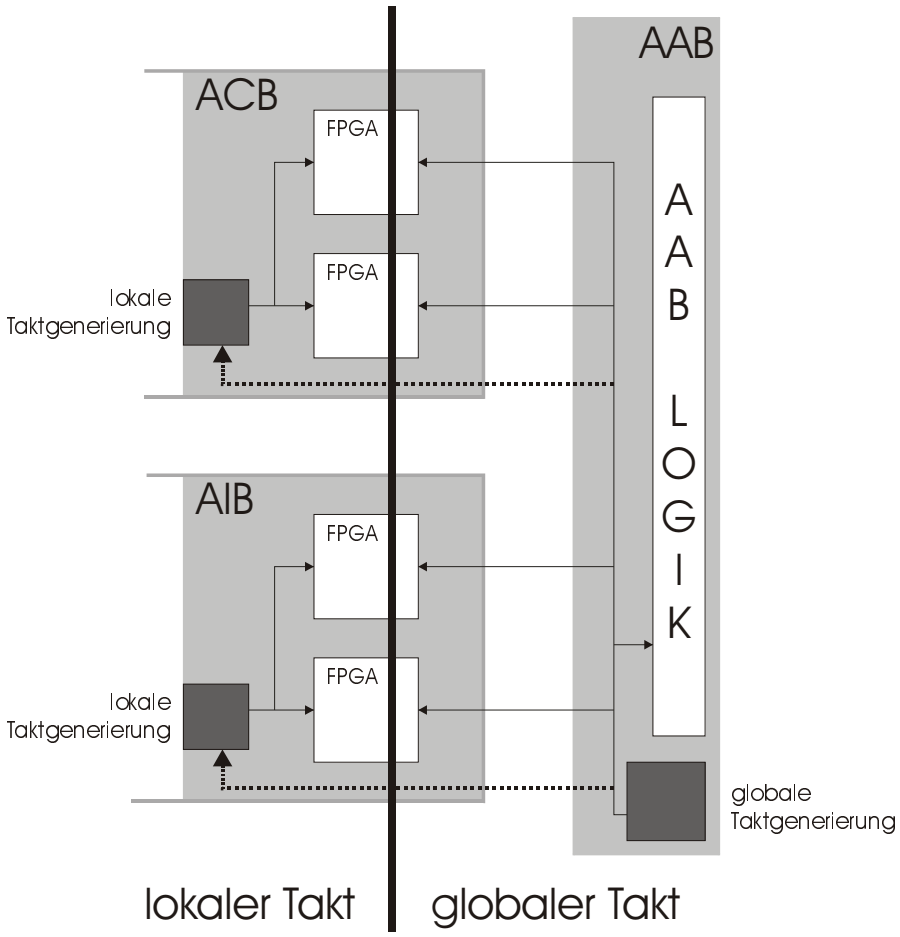


Abbildung 5.4: Das ATLANTIS-Clocksystemkonzept auf globaler Ebene

Die Clockverteilung bezogen auf das Gesamtsystem des FPGA-Teils wurde folgendermaßen gelöst. Es gibt eine globale Taktgenerierung auf dem AAB. Die hier erzeugte Clock wird synchron und phasengleich an die AAB interne Logik und die FPGAs von ACB und AIB, die die Interface-Logik zum Privat-Bus beinhalten, verteilt. Die FPGA-Designs und weitere Board-interne Logik der ACBs und AIBs werden von einer lokal erzeugten Clock mit einem Takt versorgt. Die Entkopplung zwischen beiden Frequenzbereichen wird durch, in die FPGAs zu implementierende, asynchrone FIFOs erreicht.

Vorteil dieser Konstruktion ist, daß einerseits ein Board stand-alone (ohne AAB) zusammen mit einer Host-CPU betrieben werden kann. Andererseits können auf diese Art eine beliebige Anzahl von Boards durch einen einfachen Mechanismus miteinander gekoppelt werden. Zusätzlich kann die AAB-Clock als Quelle des jeweils lokalen Clocksystems dienen. Damit steht systemweit ein frequenzabhängig phasenverschobener aber zumindest synchroner Takt zur Verfügung. Die Besprechung und Darstellung der lokalen Clocksysteme folgt in den entsprechenden Unterabschnitten.

5.1.5 Von der Schaltplaneingabe zur fertigen Platine

Die für die Entwicklung der Platinen notwendige Schaltplaneingabe, der Entwurf des Layouts und das zur Entflechtung des Leiterplatten notwendige Routen wurde unter Windows NT mit der Software Pads bzw. mit dem Specctra-Autorouter durchgeführt.

Die Schaltplaneingabe mit dem Tool PadsLogic gestaltete sich weitgehend umproblematisch und wird vor allem durch eine Reihe vordefinierter Bibliotheks-Modelle für verschiedene Gehäuse-Typen unterstützt. Die nachfolgende Plazierung der Bauteile (Layout) wurde mit PadsPower durchgeführt. Aus den Rücksprachen mit verschiedenen Platinenherstellern ergaben sich die zur Zeit technisch möglichen physikalischen Parameter für Leiterbahnbreiten (150µm - 100µm), Isolationsabstand (150µm - 100µm) und Bohrdurchmesser (0,5mm - 0,2mm, abhängig von der Platinenstärke). Insbesondere die Angaben zu Padsdurchmessern und zur Größe von Durchkontaktierungen und deren physikalisch geeignete Anordnung im Bereich der BGAs waren hilfreich, da bis zu diesem Zeitpunkt am Institut noch keine Erfahrungen mit BGAs gemacht worden waren. Ein wichtiger Hinweis bestand zum Beispiel darin, daß die BGA-Lotkugeln nicht direkt auf Durchkontaktierungen aufgelötet werden sollen, da beim Lötvorgang das Lot durch die Durchkontaktierung hindurch nach unten läuft und dadurch unzuverlässige Kontakte entstehen. Abhilfe schaffen sogenannte Sacklöcher (partielle Durchkontaktierungen) oder die Definition von Pads verbunden mit leicht versetzten Durchkontaktierungen.

Das Routen der Leiterplatten wurde semi-automatisch durchgeführt. Kritische Netze, wie zum Beispiel die Verbindungen zwischen dem CompactPCI-Stecker und dem PLX-Baustein, die nach der CompactPCI-Spezifikation ein bestimmte Länge haben müssen, wurden von Hand geroutet. Außerdem mußte der Fanout[52] der BGAs und der 124poligen Steckverbinder des ACBs von Hand durchgeführt werden, da hier die Fähigkeiten des Specctra-Autorouters an die Grenze gekommen waren.

Die Platinen des ATLANTIS Systems haben einerseits mit 14 Lagen eine beträchliche Größe und eine sehr hohen Bauteildichte, sollen aber andererseits mit bis zu 80

[52]Bei oberflächenmontierten Bauteilen ist zunächst ein sogenannter Fanout-Arbeitsschritt nötig. Dabei werden, normalerweise vom Autorouter, Ringe von Durchkontaktierungen um das Bauelement angeordnet und auf der Außenlage mit den Pads verbunden, um so eine Verbindung der Pads mit allen Signallagen zu schaffen.

MHz betrieben werden können. Um das unvermeidliche Übersprechen von Leiterbahnen zu minimieren, müssen in der Routersoftware entsprechende Randbedingungen angegeben werden. Für beide Boards war das Designziel ein maximales Übersprechen von 500mV zu erzielen, daß auch erreicht werden konnte[53]. Spezielle Randbedingungen wurden außerdem für das Routen der Clocknetze gewählt. Um das Auftreten von Clock-Skews auf funktional zusammenhängenden Clocknetzen zu verhindern, wurden diese für das Routen zu Gruppen zusammengefaßt und auf gleiche Länge definiert. Insgesamt betrugen die Routingaktivitäten pro Board ca. 3 bis 4 Wochen. Das Routen der weniger komplexen Test-Backplane (ATB), konnte innerhalb weniger Stunden durchgeführt werden.

Die Herstellung der Platinen, die Bestückung der BGAs und der Einpresstecker wurde von der Firma CaDUL durchgeführt. Die Bestückung der weiteren Bausteine bei gleichzeitigem Test der Hardware wurde im Institutslabor ausgeführt.

5.1.6 Test der Hardware

Mit der wachsenden Komplexität programmierbarer Logikbausteine und der steigenden Dichte moderner Multilayer-Boards wird es immer schwerer, die Schaltungen auf ihre Funktion hin zu testen. Klassische Methoden, bei denen das Abgreifen der Signale an Pins der ICs unumgänglich ist, können bei den heute sehr populären und weit verbreiteten SMD- und BGA-Gehäuseformen nur noch unter erheblichen Aufwand angewandt werden. Eine mittlerweile etablierte Lösung ist das Testen per JTag / Boundary Scan.

Boundary Scan bezeichnet grundsätzlich ein Protokoll, mit dem interne Knotenpunkte (nodes) eines ICs ausgelesen oder auch in einen bestimmten Zustand gesetzt werden können. Zum Auslesen werden die Zustände der Chip-Register in spezielle Register (Boundary Scan-Zellen) übernommen. Diese sind in einer Kette miteinander verbunden (1-Bit-Shift-Register, DR) und an zwei I/O-Pins angeschlossen. Die Registerinformation kann als *Bitstream* über einen dieser I/O-Pins ausgelesen und ausgewertet werden. Umgekehrt können auf diese Weise (über den anderen Pin) gezielt Daten in die internen Register geschrieben werden. Von der Joint Test Action Group (JTAG) wurde eine standardisierte serielle Schnittstelle nach IEEE 1149.1 eingeführt, mit dem Ziel, eine vereinheitlichte Testumgebung für die unterschiedlichen Logikbausteine bereitzustellen [AKi96].

Alle größeren Bausteine der ATLANTIS-Boards (FPGAs und CPLDs) besitzen diese JTag-Schnittstelle. Die nowendigen statischen Verbindungstests zwischen diesen Bausteinen, die dazu dienen, Löt- und Fertigungsfehler auf der Platine aufzudecken, können also für den Großteil der Netze auf den Platinen durchgeführt werden. Es können die Verbindungen zwischen den FPGAs statisch überprüft werden - auch über den Privat-Bus hinweg und zu den CPLDs hin. Hardwareseitig wurde vorgesehen, daß sowohl durch einen Zugriff über das PCI-Interface als auch über einen eigens dafür integrierten Steckverbinder der JTag-Test durchgeführt

[53]Der gewählte Wert geht auf Ergebnisse von Simulationsuntersuchungen zurück, die von Jozsef Ludvig im Rahmen der Entwicklung des Enable++-Systems gemacht wurden [Lud98].

werden kann. Für die zweite Variante erfolgt der Anschluß mit Hilfe eines speziellen Kabels über den Parallel-Port eines Standard-PCs. Damit läßt sich ein Board-Test durchführen, ohne daß ein CompactPCI-System benötig wird. Eine entsprechende JTag-Software, speziell ausgelegt für das ATLANTIS, wurde von Matthias Müller in Zusammenarbeit mit dem Weizmann-Institut in Israel entwickelt und erfolgreich auf die Boards angewandt. Neben dem JTag-Test sind auf den Boards auch zahlreiche “klassiche” Testmöglichkeiten eingebaut. So können an kritischen Stellen über Test-pins sowohl Clock- als auch Datensignale abgenommen werden. Zusätzlich lassen sich über LEDs der Status von Aktivitäten auf den Boards direkt visuell darstellen.

Beim Test und Aufbau des zuerst entwickelten ACBs hat sich das Debuggen der PCI-Lokal-Bus Schnittstelle als die zeitaufwendigste Arbeit erwiesen. Dies wurde dadurch verursacht, weil für die Analyse der Signale sehr umständlich kleine Drähtchen an die fine-pitch-Bausteine angelötet werden mußten. Für das nachfolgend entwickelte AIB ist diese negative Erfahrung bereits mit in das Platinenlayout eingeflossen. Beim AIB werden alle PCI-Lokal-Bus-Signale über einen Test-Header geführt, mit direkter Anschlußmöglichkeit an einen Logik-Analysator.

5.2 Das ATLANTIS Computing-Board

Das Computing-Board ist das Herzstück des FPGA-basierten Teil des ATLANTIS-Systems (Abbildung im Anhang B). Der FPGA-Prozessorkern besteht aus einer 2x2 FPGA Matrix. Jeder der 4 Lucent Orca-3T125-FPGAs [Luc98] hat eine Komplexität von 186k Gatter, pro ACB stehen also 744k-Gatter-Logikresourcen zur Verfügung. Für die Verschaltung der FPGAs untereinander bzw. nach außen waren zwei wichtige Punkte zu entscheiden:

Frage 1 war, ob die Verwendung dedizierter Interconnect-Bausteine zwischen den FPGAs sinnvoll ist. Damit soll grundsätzlich eine größere Flexibilität bei der Verschaltung der FPGAs untereinander erreicht werden. Die “größere” Flexibilität rechtfertigt den Mehraufwand (zusätzliche Konfigurationslogik für den Interconnect) und den zusätzlichen Flächenbedarf nicht. Für die vier FPGAs sind jeweils 72-Bit-breite Datenpfade zu den beiden nächsten Nachbar-FPGAs vorgesehen, womit ausreichend Signalwege zur Verfügung stehen, die per FPGA-Konfiguration spezifisch genutzt werden können. Damit werden die Anforderungen, der für das ATLANTIS-System geplanten Anwendungen abgedeckt.

Als zweiter Punkt wurde die Notwendigkeit von expliziten Dual-Port-RAM-Bausteinen zur Entkopplung verschieden getakteter Systembereiche diskutiert. Die FPGAs stellen ausreichend Logikresourcen zur Implementierung von DPR zur Verfügung, so daß keine zusätzlichen externen Bausteine benötigt werden.

5.2.1 Die Architektur des Computing-Boards

Bei der Entwicklung der ACB-Architektur wurde grundsätzlich versucht, auf Symmetrien zu achten, weil dies erfahrungsgemäß sowohl den Entwicklungsprozeß als auch die spätere Benutzung des Systems vereinfacht (Abbildung 5.5).

Symmetrie zeigt sich zunächst darin, daß jeder FPGA vier funktionale Schnittstellen (Ports) besitzt:

- zwei 72-Bit-breite Ports, jeweils zum linken und rechten Nachbar-FPGA
- ein 206-Bit-breiter Modul-Port, über den Speicher- und sonstige Module ins ACB integriert werden können
- ein logischer I/O Port, ebenfalls (maximal) 72-Bit-breit

Die vier genannten Ports belegen zusammen 422 I/O-Pins. Der 72-Bit-breite Port zwischen den FPGAs kann von der Verwendung als Höchstleistungsverbindung bis hin zu einer Multi-Kanal-Schnittstelle beliebig genutzt werden. Die Aufteilung und Richtung der Datenpfade wird grundsätzlich durch die Konfiguration der FPGA-I/O-Blöcke festgelegt. Der Modul-Port ist aus zwei *high density*-124-Pin-Steckverbindern aufgebaut. Anwendungsabhängig können beliebig Schnittstellenkarten integriert werden, die vollständig vom FPGA aus kontrollierbar sind. Für die Full-Scan-TRT-Anwendung werden beispielweise SRAM-basierte Speicher-Module verwendet, die als Einzel-Speicher-Bank in der Organisation 512k x 176 aufgebaut sind (vgl. Abschnitt 6.4.3). Das ergibt eine integrale Speicherdichte von 44 MByte SRAM pro ACB. Für den 3D-Volume-Rendering-Algorithmus wurde ein Modul entwickelt, daß mechanisch über alle vier FPGA-Modul-Ports reicht. Dieses Modul ist mit einem weiteren Lucent-FPGA und 128 MByte SDRAM bestückt (Abschnitt 7.4.3).

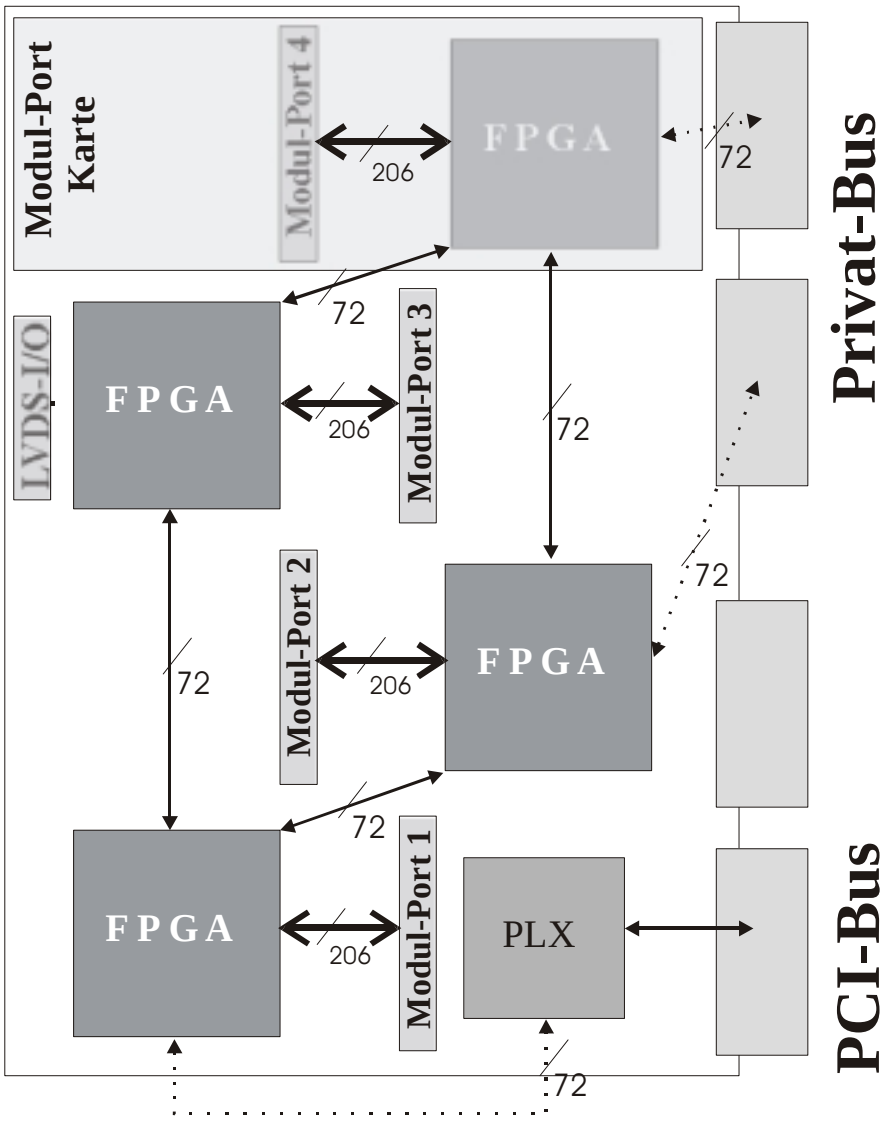


Abbildung 5.5: Schematische Darstellung des Computing Boards (bestückt mit einer Modul-Port-Karte einfacher Größe auf Port 4)

Die logischen I/O-Ports werden von den FPGAs unterschiedlich benutzt. Ein FPGA ist direkt mit dem PLX PCI9080-Interface-Chip verbunden und stellt somit die Schnittstelle zur Host-CPU bereit. Zwei FPGAs verfügen über jeweils ein 72-Bit-breites Interface zum Privat-Bus. Bei 80 MHz Taktfrequenz ergibt dies eine integrale Datentransferrate von 1,44 GByte/s. Der vierte FPGA besitzt als I/O-Port ein zweifaches 28-Bit-LVDS-Interface, das zu Testzwecken eingerichtet wurde. Die LVDS-Ports können zum Anschluß eines externen Test-I/O-Moduls verwendet werden. In erster Linie sind sie dazu gedacht die S-Link/M-Link-Schnittstelle, die im Rahmen der CERN Aktivitäten verwendet wird, anzuschließen (Abschnitt 7.1.3).

5.2.2 Das Clocksystem des Computing-Boards

Bei der Abbildung einer Anwendung auf das ATLANTIS-System muß diese vorab partitioniert und auf die FPGAs verteilt werden. Die Ergebnisse der Place&Route-Software legen letzt endlich die maximale Clockfrequenz fest, mit der ein einzelner FPGA getaktet werden kann. Die bisherigen Erfahrungen mit dem Enable++-System haben gezeigt, daß ein zentraler Systemtakt für die Ausführung einer über mehrere FPGAs verteilten Anwendung völlig ausreichend ist. Typischerweise unterscheiden sich die einzelnen FPGA-Designfrequenzen nur um wenige Prozent. Dies wurde bei der Implementierung des Full-Scan-TRT-Algorithmus' wiederum bestätigt (Kapitel 6). Die durch die Place&Route-Software für die vier FPGA-Designs angegebene Frequenz liegt zwischen 19 MHz und 21 MHz. Daraus ergibt sich der maximale Systemtakt zu 19 MHz, mit dem alle FPGAs von einer Quelle aus mit einem Takt versorgt werden.

Eine bekannte Ausnahme aus der Arbeit mit dem Enable++-System stellen die Zugriffe (Lesen und Schreiben) auf angeschlossene Speicherbausteine dar. Unter bestimmten Umständen kann es sinnvoll sein, zur Verdopplung der Speicherbandbreite, die Speicherbausteine im Verhältnis zum FPGA-Design mit der doppelten Frequenz zu takten. Dieser Fall tritt zum Beispiel ein, wenn das FPGA-Design mit nur einer vergleichsweise niedrigen Frequenz betrieben werden kann, aber doppeltgetaktete Speicherzugriffe die Gesamtperformance erhöhen. In Abbildung 5.6 ist durch das zugehörige Timing-Diagramm die Situation bei einem Lesezugriff dargestellt.

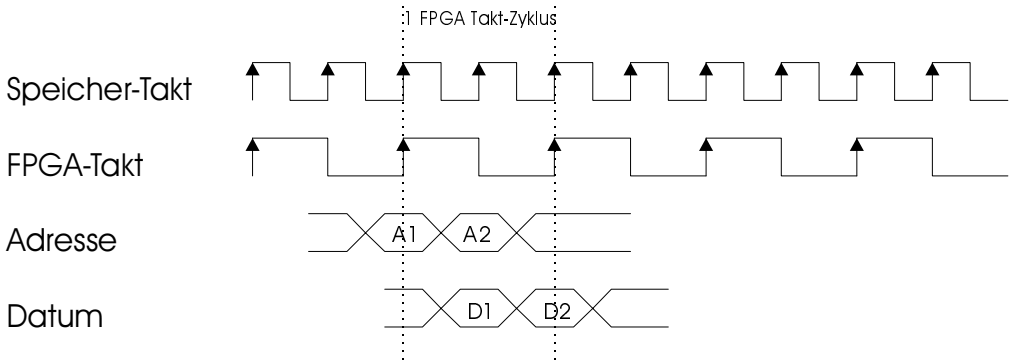


Abbildung 5.6: Lesezugriff auf Speicher mit doppelter Taktfrequnez im Verhältnis zur FPGA-Designfrequenz

Aus der Sicht des FPGAs-Designs werden in jedem Taktzyklus zwei Datenwörter gelesen. Die Speicherbandbreite wird dadurch verdoppelt. Beim ACB wurde dieser Fall mitberücksichtigt.

Unter den genannten Gesichtspunkten, wurde für das Clocksystem des ACBs ein dreigeteilter Aufbau gewählt:

- Lokal-Bus Clocksystem
- System- und Modul-Clock
- LVDS-Interface Clock

Zunächst benötigt der Lokal-Bus einen einstellbaren Clock (Abbildung 5.7, gestrichelte Linie). Für den Lokal-Bus wird die maximale Taktfrequenz durch den PLX-Baustein auf 40 MHz begrenzt. Über den Lokal-Bus werden grundsätzlich die Zugriffe vom PLX PCI-Interface-Baustein zu den CPLDs[54] und zum FPGA hin ausgeführt, der das PCI Lokal-Bus-Interface beinhaltet. Zur Clockgenerierung wird, wie bei den anderen beiden Teilsystemen, der Clockgenerator ICD2053B von Cypress [Cyp95] eingesetzt. Dieser ist über ein serielles Interface vom CPLD aus im Frequenzbereich zwischen 0,3 bis 100 MHz einstellbar, und damit wird die Erzeugung der für das ATLANTIS-System angestrebten systemübergreifenden Taktfrequenz im Bereich zwischen 5 MHz und 80 MHz unterstützt. Der benötigte externe Referenzclock wird von einen 10-MHz-Quarzoszillator geliefert.

Um auf dem ACB die FPGA-Designs und die angeschlossenen Modul-Ports mit den vorgesehenen maximal 80 MHz takten zu können, ist ein zweites Clock-Subsystem nötig und implementiert. Die Taktgenerierung erfolgt wie beim Lokal-Bus. Das generierte Clocksignal wird nicht direkt zu den FPGAs und den Modul-Ports geführt, sondern in den Clock-CPLD eingespeist. Gleichzeitig wird auch das in Abschnitt 5.1.4 angesprochene Clockssignal der AAB an den Clock-CPLD geleitet. Neben der Auswahlmöglichkeit zwischen intern oder extern generierter Clock sind der Hintergrund für das Zwischenschalten des CPLDs, die bereits angesprochenen Debugging-Mechanismen (Abschnitt 5.1.2). Im CPLD ist eine Logik implementiert, die es gestattet, die Anwendung im Einzelschrittbetrieb und Burstmodus zu takten. Aus dem CPLD heraus wird das Clocksignal an drei programmierbare RoboClock PLL-Clockbuffer vom Typ CYB991V [Cyp98a] weitergeleitet. Die Funktionalität dieses Bausteins umfaßt neben einer einstellbaren Taktvervielfachung um feste Faktoren auch einstellbare Phasenverschiebungen [Cyp98b]. Mit dem Einsatz der RoboClock-Bausteine wird durch eine einfache Schaltung eine maximale Flexibilität erreicht. Die zahlreichen Konfigurationsmöglichkeiten sind ausführlich in [Cyp98b] beschrieben. Damit lassen sich die FPGAs-Designs untereinander bzw. im Verhältnis zu den Modul-Port mit unterschiedlichen und phasenverschobenen Taktfrequenzen betreiben. Damit soll die effiziente Integration zusätzlicher Logik über die Modul-Ports unterstützt werden. Beispielweise können Laufzeitunterschiede ausgeglichen werden. Unter anderem kann so die angesprochene Doppeltaktung von Speicherbausteinen auf den Modul-Ports realisiert werden.

Für die Verwendung des Clocksystems im Debugging-Mode ist wichtig, daß sich die PLL-Clockbuffer[55] per externer Konfiguration auch umgehen lassen. Diese und weitere Einstellungen des RoboClocks werden über das Anlegen von

[54]Im Gegensatz zum AIB mußten auf dem ACB 2 CPLDs benutzt werden.

[55]PLL: Phase Locked Loop.

Spannungspegeln an den Konfigurationspins des Bausteins vollzogen. Die Ansteuerung dieser Pins erfolgt vom Clock-CPLD aus. Die Ausgänge zweier Robo-Clocks sind mit jeweils einem FPGA-Clockeingang verbunden (System-Clock1&2). Der dritte RoboClock versorgt die Modul-Ports mit einem Taktsignal (Modul-Clock). Abbildung 5.7 zeigt diesen Sachverhalt; zur besseren Übersicht ist die Clockleitung, die von jedem Modul-Port separat zum FPGA geführt wird (Modul-Eingangs-Clock) in der Abbildung nicht eingezeichnet. Die Frequenz (System- und Modul-Clock) ist im Bereich zwischen 5 MHz und 80 MHz einstellbar.

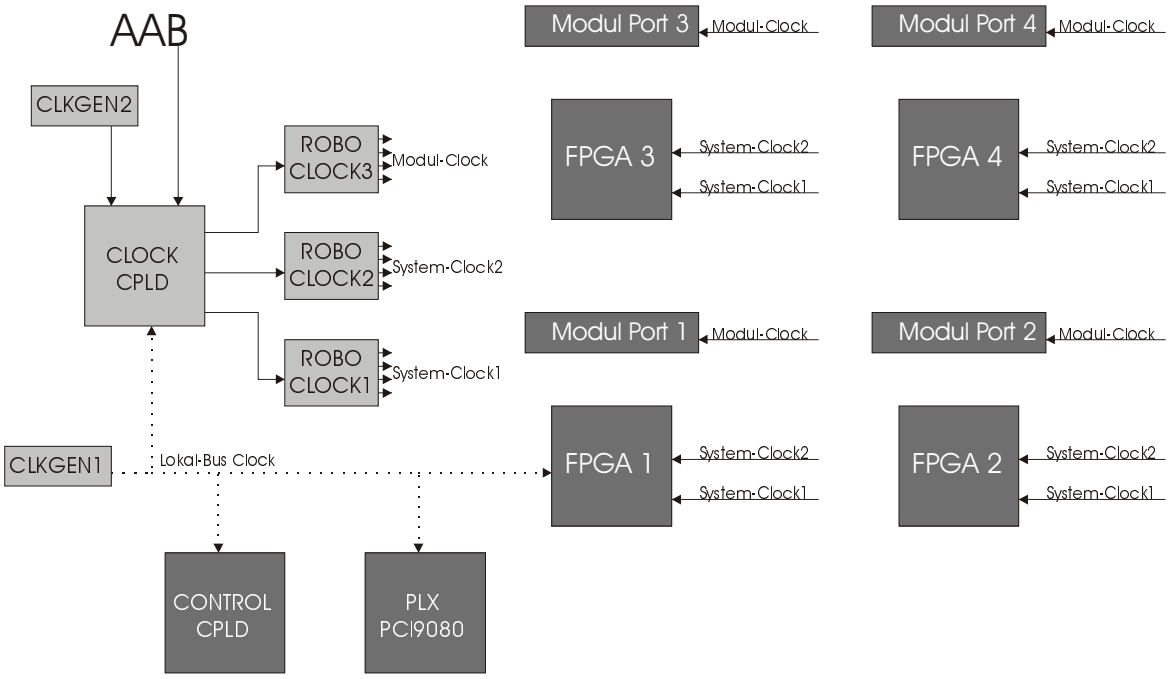


Abbildung 5.7: Blockdiagramm des Computing-Board Clocksystems

Das dritte Clock-Subsystem bezieht sich auf den Test-I/O (LVDS-Interface). Auch hier ist die Clockgenerierung mit Hilfe eines Clockgenerator-Baustein ICD2035B realisiert (ACB als Sender). Das erzeugte Taktsignal wird direkt auf den LVDS-Treiberbaustein geführt. Der Frequenzbereich ist ebenfalls zwischen 5 MHz und 80 MHz einstellbar. In der anderen Richtung wird ein ankommender Clock an das FPGA geleitet (ACB als Empfänger).

5.3 Das ATLANTIS I/O-Board

Die Hauptaufgabe des ATLANTIS I/O-Boards besteht in der Bereitstellung einer leistungsfähigen Schnittstelle zwischen dem ATLANTIS-System und externen Systemen. Die Schnittstelle soll frei konfigurierbar und unabhängig von einem spezifischen Schnittstellen-Standard sein. Die Grundlage dazu bilden der Einsatz von FPGAs u.a. für die Aufnahme der Protokolllogik, DPR-Bausteine (FIFO) zur Entkopplung der Schnittstellen von der Board-Logik und ein Datenspeicher (Buffer) für das

Zwischenspeichern von größeren Datenmengen. Der CompactPCI-Formfaktor schränkt mechanisch die Anzahl möglicher I/O-Ports ein. Die Wahl von vier I/O-Ports ist ein Kompromiß aus größtmöglicher Fläche für die I/O-Module und maximaler Anzahl unabhängiger Schnittstellen. In Abbildung 5.8 ist der Aufbau des I/O-Boards schematisch dargestellt .

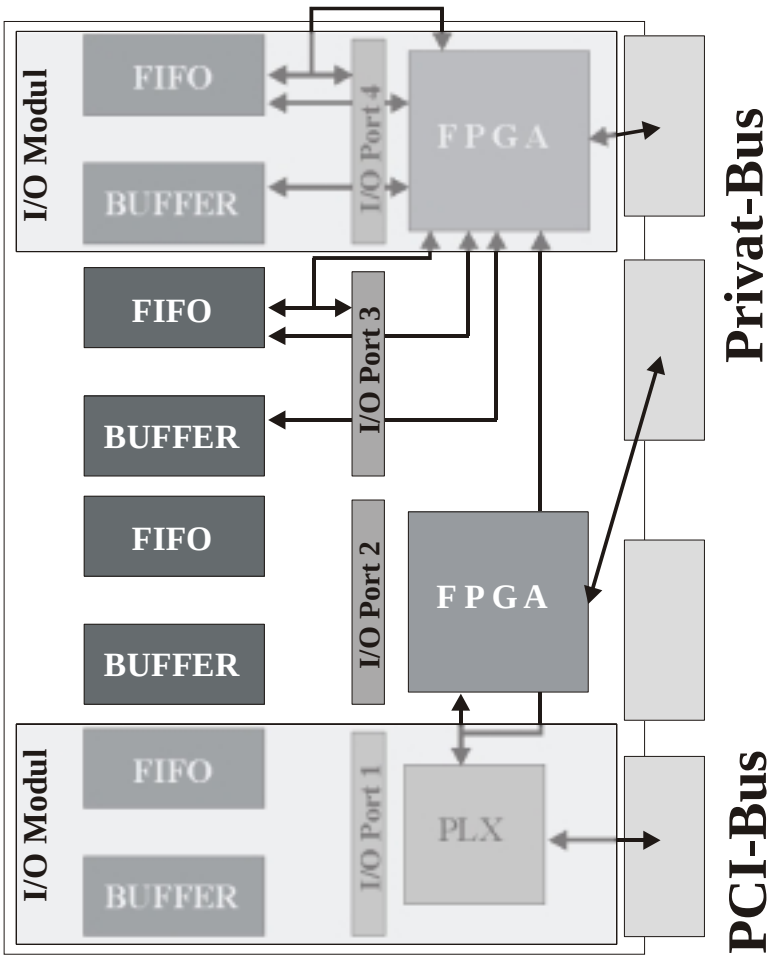


Abbildung 5.8: Schematische Darstellung des I/O-Boards mit bestückten I/O Ports 1 und 4

5.3.1 Architektur des I/O-Boards

In diesem Abschnitt wird die grundlegende Architektur des I/O-Boards beschrieben. Der symmetrische Aufbau ist in Abbildung 5.9 dargestellt. Die zentralen Logikresourcen werden von zwei Xilinx Virtex(-E)-FPGAs im FG680-BGA-Gehäuse bereitgestellt [Xil00]. Dieser Gehäusetyp bietet einerseits den benötigten, sehr hohen Pincount zur Anbindung der Peripherie. Andererseits sind für diesen Gehäusetyp sowohl Bausteine der Virtex-Serie (XCV600-XCV1000) als auch der zukünftigen Virtex-E-Serie (XCV1000E-XCV2000E) erhältlich, bzw. angekündigt. Damit ist der Einsatz zukünftiger, leistungsfähigerer FPGAs aber auch prinzipiell

die Skalierbarkeit der Logikresourcen gewährleistet, ohne daß ein Redesign des AIBs durchgeführt werden muß. In der ersten AIB-Version wird der Virtex-FPGA XCV600 eingesetzt [Xil00]. Für das AIB ergibt sich damit eine FPGA-Logik-Komplexität von 373k Gatter. Zusätzlich beinhalten die FPGAs zusammen 600 kBit festverdrahtetes Block-Ram.

Da im ATLANTIS-System nur standardmäßig 5V und 3,3V Versorgungsspannung zur Verfügung stehen, die FPGAs aber als Core-Spannung 2,5V (Virtex) bzw. 1,8V (Virtex-E) benötigen, wurde im Layout des AIBs die Bestückung von einem passenden Fest-Spannungsregler vorgesehen [Mic99]. Ein weitere Anpassung betrifft die I/O-Blöcke der Virtex-E-FPGAs, die nicht 5V-kompatibel sind. Um dennoch eine Verbindung zum 5V-PLX-PCI9080-Baustein zu schaffen, wurden Pegelshifter-Bausteine [Tex98] dazwischen geschaltet.

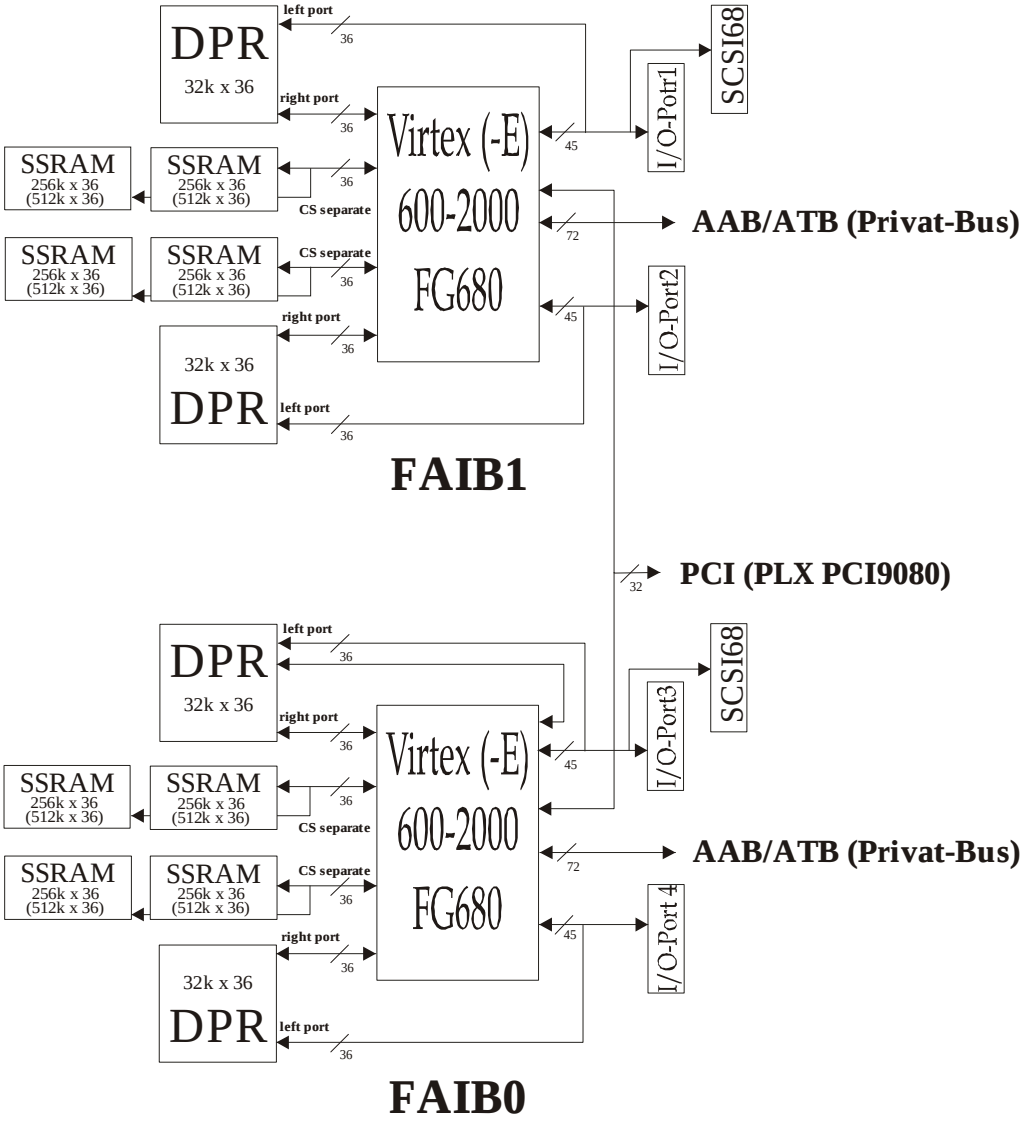


Abbildung 5.9: Blockdiagramm des I/O-Boards

Abbildung 5.9 zeigt die Basis-Architektur des AIBs. Auch beim AIB wurde ein symmetrischer Aufbau gewählt. Beim Design wurden die Anforderungen der ATLAS Readout-Systeme und aber auch der FOPI-Trigger-Anwendung als Vorgabe berücksichtigt. Jeder FPGA (FAIB0 und FAIB1) kontrolliert jeweils zwei I/O-Ports. Ein I/O-Port ist über eine 45-Bit-breite Schnittstelle mit dem FPGA verbunden. Berücksichtigt man die Systemfrequenz von 80 MHz folgt daraus eine maximale Datentranferrate von 450 MByte/s. Die für die ROB-Anwendung zur Integration ins AIB vorgesehene 32-Bit-breite M-Link-Schnittstelle (Abschnitt 7.1.3) unterstützt eine Datenrate von maximal 160 MByte/s und kann somit problemlos integriert werden. Zur Unterstützung der FOPI-Trigger-Anwendung wurde die maximal mögliche Anzahl an Dateneitungen (45) von jeweils einem physikalischen I/O-Port an das FPGA geführt. Das ergibt insgesamt 180 1-Bit-Kanäle bezogen auf ein I/O-Board (vgl. Abschnitt 4.2 und Abschnitt 7.3). Zur Unterstützung eines kontinuierlichen und hohen Datentransfers sind die I/O-Port-Datenleitungen (36 Bit) gleichzeitig auf einen FIFO-ähnlichen Zwischenspeicher geführt. Zum Einsatz kommen DPR-Bausteine vom Typ IDT70V3579S [IDT99] in der Organisation 32k x 36 mit einer Speicherdichte von 144 kByte.

Für das Speichern von größeren Datenblöcken ist jeweils eine Speicherbank pro I/O-Kanal vorhanden. Diese wird gebildet von zwei Samsung SSRAM-Speicher-Bausteinen, wobei die Bestückung flexibel gehalten ist. In der ersten Version des AIBs werden zwei 8-MBit-Bausteine vom Typ KM736V887 in der Organisation 256k x 36 eingesetzt [Sam99a]. Eine Upgrademöglichkeit auf die zukünftigen 16-MBit-Bausteine [Sam99b] sowie die Bestückung mit den neuartigen NtRAM-Bausteinen[56] [Sam99c] wurde hardwareseitig berücksichtigt. Dies ist ohne Umbau der AIB-Platine möglich. Grundvoraussetzung dafür war die weitgehende Pinkompatibilität der verschiedenen Samsung SSRAM-Bausteintypen. Wegen der gemeinsam genutzten Datenleitungen erfolgt der Zugriff auf die beiden Bausteine einer Bank grundsätzlich sequentiell und wird über das CS-Signal kontrolliert. Die Speicherbandbreite beträgt pro Bank 360 MByte/s. In Abschnitt 7.1 ist für die ROB-Anwendung gezeigt, wie die Abbildung auf die AIB-Struktur durchgeführt wird.

Als I/O-Port-Steckverbinder werden nach IEEE-3286 standardisierte 64-Bit-Mezzanine-Stecker verwendet. Zusätzlich ist bei zwei I/O-Ports noch ein 68poliger SCSI-Steckverbinder mit angeschlossen (32-Bit-Datenbus). Diese Datenleitungen zum FPGA werden von jeweils beiden Ports gemeinsam genutzt. Dieser Steckverbinder wird parallel dazu auch auf dem neuen microEnable2-Board von Silicon Software implementiert. Für diese Schnittstelle entwickelte Module können dadurch sowohl im ATLANTIS-System als auch im microEnable2-System genutzt werden.

Das PCI-Lokal-Bus-Interface und die Schnittstelle zur AAB (Privat-Bus) sind funktional äquivalent zu denen auf dem ACB aufgebaut. Der einzige Unterschied besteht darin, daß beim AIB beide FPGAs an den lokalen Bus angeschlossen sind. Analog verhält es sich mit dem Anschluß an den Privat-Bus. Wie beim ACB sind

[56]Nt steht für *No turnaround*. Bei der Verwendung von Bausteinen dieser Technologie müssen im Gegensatz zu herkömmlichen SSRAM-Bausteinen, beim Wechsel zwischen Lese-und Schreibzugriff keine Wartezyklen in Kauf genommen werden. Es wird damit die volle, mögliche Datentransferrate unterstützt.

auch beim AIB zwei 72-Bit-Busse von den FPGAs auf die CompactPCI-Stecker J4 und J5 geführt.

5.3.2 Das Clocksystem des I/O-Boards

Bei der Clocksystementwicklung des I/O-Boards hat sich der Schwerpunkt im Vergleich zum ACB verlagert. Im Vordergrund steht beim AIB die flexible Unterstützung verschiedener Schnittstellen-Standards. Dies wird dadurch erreicht, in dem jeder I/O-Port über eine eigene Clockgenerierungseinheit (Clock-Gen I/O 1-4 in Abbildung 5.10) verfügt. Die Schaltung des Clocksignals auf den Stecker ist einstellbar. Entweder kann das AIB als Clock-Source dienen (Sender-Mode) oder als Clock-Destination (Empfänger-Mode) konfiguriert werden. Die Besonderheit dabei ist, daß physikalisch die gleiche Leitung benutzt wird. Dies ist historisch begründet und hängt mit der Unterstützung des S-Link-Standards zusammen (Abschnitt 7.1.3). Die Schnittstelle, die über die beiden microEnable2-kompatiblen SCSI-Steckverbinder realisiert ist, besitzt im Gegensatz dazu zwei getrennte Clockleitungen.

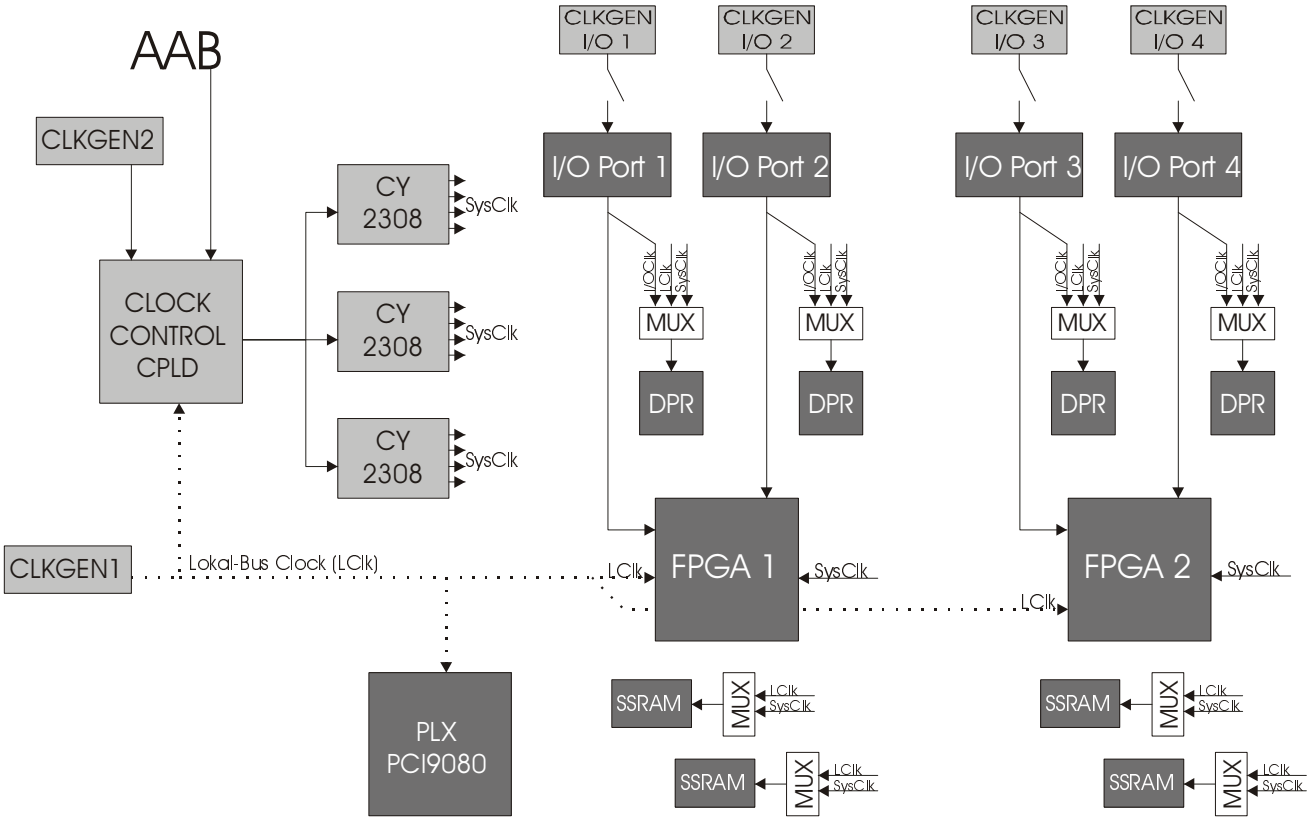


Abbildung 5.10: Blockdiagramm des I/O-Board Clocksystems

Ähnlich wie beim ACB ist der Lokal-Bus-Clock (LClk) implementiert. Alle Teilnehmer des lokalen Busses - der PLX-Baustein, der CPLD und beide FPGAs - müssen mit diesem Takt versorgt werden. Analog zum ACB ist die maximale Frequenz durch den PLX-Baustein auf 40 MHz begrenzt. Zur Bereitstellung der 80-MHz-Systemfrequenz wird auch beim AIB zusätzlich ein unabhängiger Systemtakt (SysClk) erzeugt und ebenfalls (wie beim ACB) zu Debugging-Zwecken über den CPLD geführt. Dieser muß allerdings nicht wie im Fall des ACBs in der Phase einstellbar sein, da alle Bausteine, die mit diesem Clocksignal gespeist werden, auf der Platine festverdrahtet sind und damit unveränderbare elektrische Eigenschaften besitzen. Für die Clockverteilung wird daher ein einfacher PLL-Clock-Buffer des Typs CY2308 von Cypress eingesetzt [Cyp98c]. Der SysClk auf dem AIB läßt sich im Frequenzbereich zwischen 5 MHz und 80 MHz einstellen.

Die Clockverteilung insgesamt sieht wie folgt aus (Abbildung 5.10): Sowohl der LClk als auch der SysClk werden an die FPGAs und an alle Speicherbausteine geführt: Dies wurde deshalb implementiert, damit einerseits eine Anwendung vollständig phasensynchron auf dem AIB ausgeführt werden kann (Lclk mit max. 40 MHz). Aus der Erfahrung beim Aufbau des ACBs hat sich gezeigt, daß diese Eigenschaft die grundlegenden Boardtests stark vereinfacht. Andererseits ist das AIB für einen 80-MHz-Systemtakt spezifiziert. Entsprechend muß diese Frequenz für alle Bauteile verfügbar gemacht werden. Da die Speicherbausteine nur einen Clockeingang besitzen wurden Multiplexer-Bausteine des Typs PI3B3125 [Per00] eingefügt, die über das CPLD gesteuert werden. Damit lassen sich einzelne Speicherbausteine (SSRAM und DPR) anwendungsabhängig, entweder phasensynchron zum LClk oder zum SysClk takten. Bezogen auf den LClk kann das vollständige System - mit Ausnahme der I/O-Ports - synchron und phasengleich mit bis zu 40 MHz getaktet werden. Einen Spezialfall bilden die DPR-Bausteine. Hier ist ein einer der beiden Ports dafür vorgesehen, direkt von einer Schnittstelle Daten entgegenzunehmen bzw. Daten auf die Schnittstelle zu schreiben. Daher muß dieser Port neben dem LClk und dem SysClk auch von jeweils einem I/O-Clock gespeist werden können. Die notwendige Entkopplung zwischen unterschiedlich getakteten Frequenzbereiche erfolgt mittels asynchroner FIFOs in den FPGAs.

5.4 Der ATLANTIS Privat-Bus

Die Entwicklung und der Aufbau einer leistungsfähigen und flexiblen Kommunikationseinheit zur Kopplung FPGA-basierter Module war bereits das Thema der Diplomarbeit des Autors [Sin96]. In dieser Arbeit wurde für das Enable++-System ein synchroner Hochgeschwindigkeitsbus auf FPGA-Basis entwickelt. Viele der damals existierenden Randbedingungen gelten auch heute noch für den ATLANTIS Privat-Bus, so daß das Konzept weitgehend übernommen werden kann.

Das Grundkonzept sieht vor, daß keine durchgehenden Busleitungen (wie z.B. beim VME) die Slots miteinander verbinden, sondern die einzelnen Slots galvanisch durch Registerstufen voneinander getrennt werden. Durch diese gerichtete

Slot-zu-Slot-Verbindung werden die Leitungslängen sehr kurz, wodurch sich bereits gute elektrische Übertragungseigenschaften ergeben und auf eine Terminierung verzichtet werden kann. Der Datentransfer erfolgt synchron von Slot zu Slot zu einem vorgegebenen Systemtakt. Taktraten in der Größenordnung von 80 MHz sind mit der heutigen Elektronik ohne weiteres zu realisieren. Crosstalk und Synchronität der Signale spielen bei diesen kurzen Signalwegen und der hier angegebenen Clockfrequenz auch keine Rolle, so daß die Busbreite entsprechend hoch gewählt werden. Damit sind Datenübertragungsraten im GByte / s-Bereich möglich.

5.4.1 Das Konzept für den ATLANTIS Privat-Bus

Der ATLANTIS Privat-Bus soll prinzipiell als 3-HU-Platine (Active Backplane) für ein herkömmliches CompactPCI-Crate entwickelt und mechanisch über der CompactPCI-Systemplatine in die obere Hälfte eines 6-HU-Crates eingebaut werden. Das bedeutet für ein Standard 8-Slot-CompactPCI-System, daß nur vier Slots für den Privat-Bus zur Verfügung stehen. Jeder zweite Steckverbinder muß aus mechanischen Gründen für die Aufnahme der Verbindungsbausteine und der Bausteine für das Clocksystem ausgespart werden.

Grundsätzlich ist die Frage zu klären, wie die ideale Verbindungstopologie des Bussystems aussieht. Bereits bei der Konzeption des Enable++-Bussystems wurde dies ausführlich diskutiert und hat zum Ergebnis geführt, daß eine T-Topologie die geeignetste ist. Abbildung 5.9 zeigt die T-Topologie für den ATLANTIS Privat-Bus.

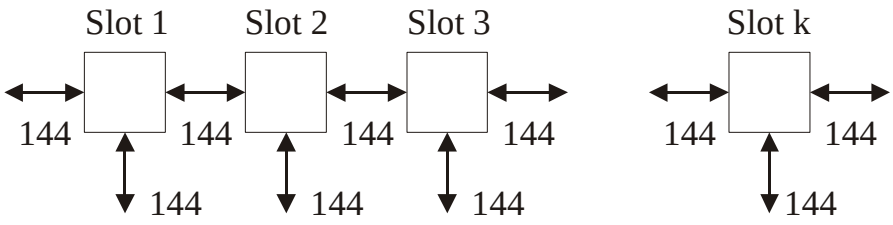


Abbildung 5.11: Verbindungstopologie für den ATLANTIS Privat-Bus

Von einem ACB oder AIB werden vier logische Busse mit je 36 Bit auf einen Slot geführt. Über den T-Topologie können beispielsweise zwei 72-Bit-breite Subbusse vom Steckverbinder über den Verbindungsbaustein je zu dem rechten und linken Nachbar-Slot implementiert werden und parallel dazu kann im Hintergrund über einen weiteren 72-Bit-Bus (über die involvierten Verbindungsbausteine) eine Kommunikation stattfinden. Durch eine entsprechende Konfiguration ist beispielsweise auch ein Broadcast über den gesamten Bus hinweg möglich.

Mechanisch sind zum Aufbau der AAB die Standard-CompactPCI-Steckverbinder vorgesehen. Nach CompactPCI-Spezifikation sind das die Stecker J4 und J5. Damit stehen insgesamt 220 Pins zur Verfügung. Die gewählte Aufteilung sieht vor, daß 144 Pins für Datensignale des Privat-Busses, 10 Pins für den

AAB-Konfigurationsdatenbus, 5 Pins für Clockleitungen und die restlichen 61 Pins als zusätzliche Spannungsversorgungsanschlüsse (+3.3V, GND) genutzt werden.

Bei dem synchronen Hochgeschwindigkeitsbus für das Enable++-System stand mit im Vordergrund, durch die eingesetzten FPGAs als Verbindungsbausteine zwischen den Slots, bereits an dieser Stelle Logikresourcen zur Vorverarbeitung der Daten beim Transfer zwischen der Enable++-I/O-Einheit und der Recheneinheit verwenden zu können. Der Grund dafür war, weil die Komplexität der Logikresourcen auf der I/O-Einheit vergleichsweise gering war und diese im wesentlichen für die Protokolllogik der Schnittstelle benötigt wurde. Beim ATLANTIS-System ist dieser Umstand so nicht mehr gegeben und der Einsatz einfacher Switch-Bausteine (Cross-Bars) ist völlig ausreichend[57].

Die Wahl dieses Verbindungsbausteins ist durch drei Hauptkriterien bestimmt. Mechanisch muß er zwischen zwei CompactPCI-Slots passen. Der integrale Pincount muß durch die T-Topologie 3 x 144 Bit betragen und die 80 MHz Systemfrequenz muß unterstützt werden. Ein Verbindungsbaustein, der alle Kriterien unterstützt ist der IQ64b im TQFP100-Gehäuse[58] von der Firma I-Cube [ICu97]. Jeweils 8 dieser Bausteine werden bei doppelseitiger Bestückung für die konfigurierbare Verschaltung eines Slots benötigt. Ein IQ64b-Baustein kontrolliert einen logischen 18-Bit-Bus. Damit werden pro Baustein 54 I/O-Pins der 64 verfügbaren verwendet (Abbildung 5.12).

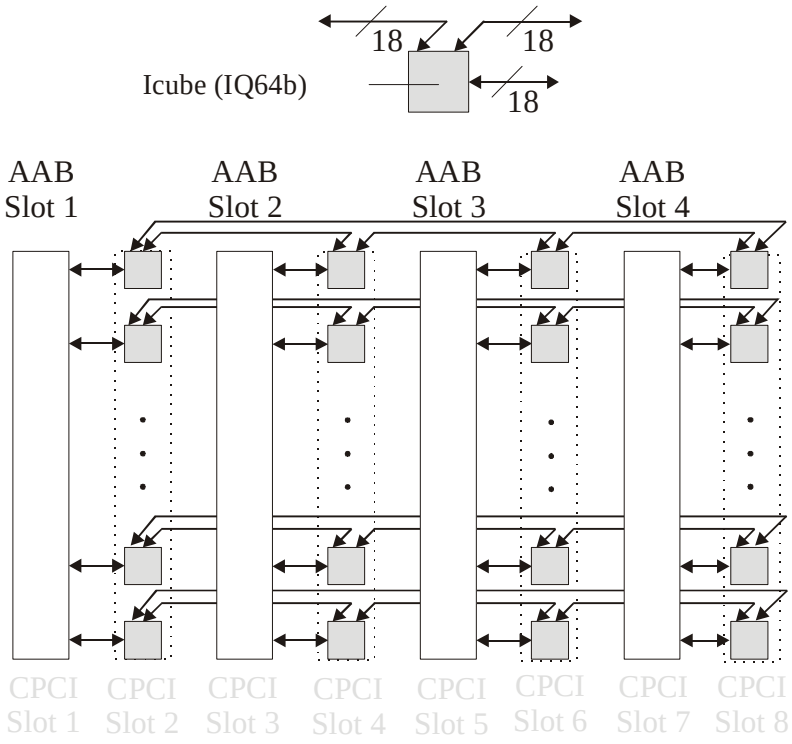


Abbildung 5.12: Schematischer Aufbau des ATLANTIS Privat-Bus (AAB)

[57]Ein Vergleich der Komplexitäten der eingesetzten Bausteine verdeutlicht dies.Im Fall des Enable++-Systems wurden Xilinx-FPGAs vom Typ XC4005H mit einer Komplexität von 5k Gattern eingesetzt. Die Komplexität bei den AIB-FPGAs beträgt 186k Gatter.

[58]TQFP: Thin Quad Flat Pack.

Wichtig ist, daß für die Plazierung der Bausteine des Clocksystems noch genügend Platz auf der Platine zur Verfügung steht. Für die Clockgenerierung ist der im ATLANTIS-System übliche Baustein (Clockgenerator ICD2053B von Cypress) vorgesehen, die Clockverteilung soll durch einfache Clockbuffer realisiert werden.

Es ist zu bemerken, daß durch dieses Konzept beliebig breite Backplanes aufgebaut werden können. Die damit gleichzeitig verbundene notwendige Erweiterung des Systemplatine in der unteren Hälfte des Crates ist durch die Kopplung über PCI-to-PCI-Bridge-Chips möglich. 14-Slot-Systemplatinen sind kommerziell erhältlich.

5.4.2 Die ATLANTIS Test-Backplane

Durch die eingeschränkten Mitarbeiterresourcen und bedingt durch den enggesteckten Projektzeitplan wurde zunächst eine einfache 7-Slot-Test-Backplane (ATB) für ATLANTIS aufgebaut. Der wesentliche Unterschied zur AAB ist, daß anstelle der Crossbar-Switch-Bausteine herkömmliche CompactPCI-Stecker eingesetzt werden. Der Nachteil ist die geringere Flexibilität. Von Vorteil ist, daß so alle verfügbaren sieben Slots (ein Slot wird grundsätzlich von der Prozessorkarte belegt) eines Standard-8-Slotsystems für ACB- und AIB-Baugruppen zur Verfügung stehen. Da ACB und AIB so aufgebaut sind, daß auch mit aufgesteckten Modulkarten mechanisch die Abmessung einer Slotbreite nicht überschritten werden muß[59], können in diesem Fall sieben Karten in einem 8-Slot-CompactPCI-Crate betrieben werden Die Verbindungen zwischen den Steckverbindern sind statisch, die Richtung bleibt aber über die FPGA-Designs von ACB und AIB konfigurierbar. Abbildung 5.13 zeigt schematisch den Aufbau der ATB.

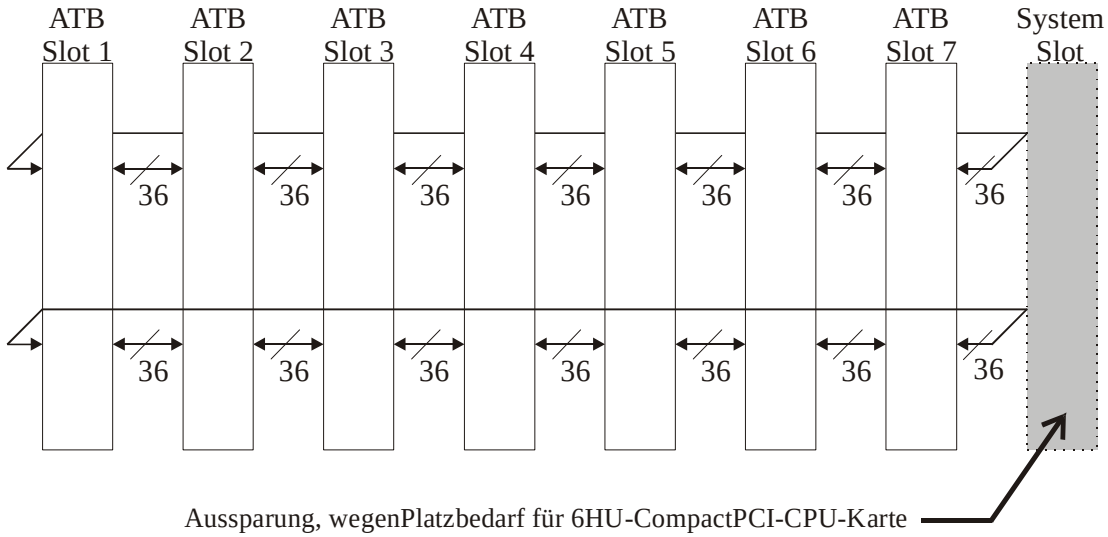


Abbildung 5.13: Schematischer Aufbau der ATLANTIS Test-Backplane für ein 8-Slot-CompactPCI-System

[59]Bisher wurden alle entwickelten Submodule (die Speicher-Module für den TRT, die M-Link-Module für den ROB-Complex und das Erweiterungsmodul für die 3D-Volume-Rendering-Anwendung) in der Bauhöhe so konzipiert, daß die Breite eines Slots nicht überschritten wird.

Es besteht jeweils eine 72-Bit-breite Verbindung (zwei logische 36-Bit-Busse) zwischen zwei benachbarten Slots. Die ATB ist bereits mit einer Clockgenerierungseinheit ausgestattet - wie bei allen anderen Modulen wird auch hier der Clockgenerator ICD2053B von Cypress eingesetzt. Die Frequenz ist einstellbar zwischen 5 MHz und 80 MHz. Die Konfiguration des Clockgenerators erfolgt über ein ACB oder AIB (vgl. Abschnitt 5.1.3). Die Clockleitungen sind so implementiert, daß alle Slots phasengleich mit einem synchronen Takt versorgt werden. Die integrale Datenbandbreite zwischen zwei Slots beträgt 1,44 GByte/s. Eine Foto der ATB findet sich im Anhang B.

5.5 Einsatz von kommerziellen Komponenten

Ein wichtiger Punkt bei der Konzeption des ATLANTIS-Systems war, daß direkt vom technologischen Forschritt in der Industrie profitiert werden kann. Bezogen auf das Host-System war eine grundlegende Voraussetzung, daß kommerzielle Komponenten in ausreichender Anzahl und Variation erhältlich und mit modernster Technologie ausgestattet sind, was im Bereich von CompactCPI erfüllt ist. Im industriellen Sektor ist insgesamt ein wachsendes Interesse für CompactPCI-Produkte zu erkennen. Insbesondere im Telekommunikationssektor sind CompactPCI-Baugruppen weitverbreitet. Es ist anzunehmen, daß die Verfügbarkeit von CompactCPI-Produkten in den kommenden Jahren gegeben ist.

CompactPCI-Baugruppen werden von verschieden Herstellern vielfach einzeln angeboten. Eine gute Übersicht zu CompactPCI-Prozessor-Karten ist auf der OpenSystems-Publishing-Webseite[60] zu finden. Daneben sind aber auch günstige Starter-Kits[61] erhältlich, die üblicherweise ein CompactPCI-Crate, einen Host-Prozessor inklusive Peripherie (Floppy, Festplatte, CD-ROM, etc.), eine System-Backplane und Software umfassen.

Für die Wahl eines CompactPCI-Systems für ATLANTIS sind zwei grundlegende Dinge zu beachten. Die System-Backplane muß eine 3-HU-Backplane sein, damit die ATLANTIS Active-Backplane in die obere Hälfte des 6-HU-Crates mit integriert werden kann. Der CompactPCI-Host-Prozessor darf nicht den CompactPCI-Steckkverbinder J3 beinhalten. Dieser kann bei der Verwendung von zwei separaten 3-HU-Backplanes nicht benutzt werden, weil an dessen Stellen die Führungsschiene zur Halterung der Backplanes verläuft. Häufig wird aber gerade über den J3 die Ansteuerung von Peripheriegeräten durchgeführt.

Bisher wurden bei ATLANTIS zwei kommerzielle Systeme eingesetzt. Beide Systeme basieren auf Standard-Intel-x86-Archituren. In einem System arbeitet ein Pentium-200MHz-MMX-Prozessor (Radisys-System) und in dem anderen ein Celeron-400MHz-Prozessor (SBS-System). Somit ist eine 100prozentige Kompatibilität zu Standard-Desktop-Rechner gewährleistet. Als Betriebssysteme werden

[60] http://www.compactpci-systems.com/.

[61] Für ATLANTIS wurde ein solches Starter-Kit von der Firma SBS-or Industrial Computer GmbH erworben, verfügbar über http://www.sbs.com/sbsor.shtml.

Windows NT und Linux eingesetzt. Dies ermöglicht die Entwicklung von gemeinsam nutzbaren Tools und Gerätetreibern sowohl für den microEnable-Koprozessor und als auch für das ATLANTIS-System.

Die CompactPCI-Prozessorkarten, zusammen mit der angeschlossenen Peripherie, erlauben es - neben der Anwendung selbst - die vollständige ATLANTIS-Entwicklungsumgebung (Abschnitt 5.6) auf dem System auszuführen. ACBs und AIBs können in beliebiger Kombination als Koprozessoren zur CPU eingesetzt werden. Wenn die vollständige Anwendung im FPGA-basierten Teil abläuft, dient die Host-CPU als Konfigurations- und Kontrollsystem.

5.6 Entwicklungsumgebungen für ATLANTIS

Neben einer leistungsfähigen Hardware ist für die praktische Nutzbarkeit, und damit auch für die Akzeptanz eines Systems wie ATLANTIS, die Bedien- und Entwicklungssoftware von entscheidender Bedeutung. Das Ziel beim Entwurf einer solchen Entwicklungsumgebung ist mit herkömmlichen Softwareentwicklungsumgebungen vergleichbar. Die benötigte Software-Werkzeuge für die Entwicklung und den Test von vollständigen Anwendungen sollten im Rahmen einer überschaubaren Oberfläche zusammengefaßt sein. Entscheidend ist, daß durch eine einfache und praktische Nutzbarkeit auch Nicht-Experten ohne großen Einarbeitungsaufwand mit dem System umgehen können. Dies ist mit die Grundvoraussetzung, daß diese leistungsfähige Technologie, Einzug in einen größeren Kreis geeigneter (wissenschaftlicher) Anwendungsgebiete halten kann und das Potential, daß in diesem System steckt, auch wirklich genutzt wird. Die Anwendungen für ATLANTIS bestehen generell aus Programmteilen, die auf der Host-CPU ausgeführt werden und einem oder mehreren Hardware-Designs im FPGA-basierten Teil des Systems. Die Softwareumgebung muß deshalb die Entwicklung einer verteilten Anwendung speziell unterstützen.

Es läßt sich eine Aufteilung der Software-Tools in Werkzeuge für die FPGA-Design-Entwicklung und in die Betriebssystem-Software vornehmen. Für ATLANTIS wird die Konzeption und Entwicklung eines Betriebssystems von Christian Hinkelbein im Rahmen einer Promotion durchgeführt und ist somit Gegenstand der aktuellen Forschung. Daher wird im nächsten Abschnitt nur kurz darauf eingegangen.

Daneben spielen die Möglichkeiten zur Entwicklung der FPGA-Designs in Verbindung mit Programmteilen, die auf der Host-CPU ausgeführt werden eine entscheidende Rolle. Die Applikationsentwicklung für ATLANTIS wird im besonderen Maße für die Verwendung der Hardwarebeschreibungssprache VHDL durch die Existenz von ATLANTIS-spezifischer Bibliotheken und einer standardisierten Simulationsumgebung unterstützt. Auf der anderen Seite bietet die in Mannheim entwickelte Sprache CHDL spezielle Eigenschaften für die Entwicklung von Anwendungen für hybride FPGA/CPU-basierte Architekturen. Die Beschreibung,

wie die Sprachen CHDL und VHDL für die Entwicklung von Anwendungen für ATLANTIS eingesetzt werden können, folgt in zwei separaten Abschnitten.

5.6.1 Das ATLANTIS-Betriebssystem (AOS)

Mit dem ATLANTIS-Betriebssystem (AOS) und mit Betriebssystemen für FPGA-basierte Architekturen ganz allgemein werden analog zu den herkömmlichen Betriebssystemen diejenigen Dienste und Programme bezeichnet, die für das Funktionieren des Systems notwendig. Bei ATLANTIS ist das ergänzende Gegenstück zu dem Teil des AOS, der auf dem Host-Rechner ausgeführt wird, die lokale Kontrollereinheit in Form des bzw. der CPLD-Designs und der zugehörigen Konfigurationsbusse.

Die Basis-Komponenten des AOS sind Initialisierungsfunktionen, Konfigurationsroutinen, Kommunikations- und Debugging-Designs (JTag/Boundary-Scan) und einige weitere (vgl. Abschnitt 5.1.3). Wichtig sind daneben die Bereitstellung von wohldefinierten und dokumentierten Schnittstellen zur Kommunikation mit dem System.

Die Entwicklung des AOS wird objektorientiert in der Sprache C++ durchgeführt. Als Plattformen werden Windows NT und Linux unterstützt. Zum jetzigen Zeitpunkt werden essentielle Dienste zum Betrieb des ATLANTIS-Systems als low-level-Funktionen zur Verfügung gestellt. Bis Mitte 2001 soll das ATLANTIS-Betriebssystem vollständig entwickelt und mit einer graphischen Benutzeroberfläche versehen sein[Hin00].

5.6.2 Die VHDL-Entwicklungsumgebung

Die VHDL-Entwicklungsumgebung *VHDL-Library* [VLi00] wurde von Harald Simmler ursprünglich für das microEnable-System entwickelt[62]. Zwei wesentliche Voraussetzungen haben dazu geführt, daß die Anpassung an das ATLANTIS-System relativ einfach durchgeführt werden konnte. Zum einen wird bei beiden Systemen der PLX-Chip PCI9080 als PCI-Interface-Baustein eingesetzt. Andererseits ist VHDL als Hardwarebeschreibungsprache von einer speziellen FPGA-Architektur unabhängig, so daß die VHDL-Bibliotheken und die sehr leistungsfähige Simulationsumgebung der *VHDL-Library* prinzipiell ohne Änderung eingesetzt werden kann. Lediglich systemspezifische Eigenschaften, wie z.B. der Einsatz von speziellen Speicherarchitekturen oder Modul-Schnittstellen müssen notwendigerweise eingearbeitet werden.

Mit der *VHDL-Library* lassen sich individuelle Applikationen für das ATLANTIS System erstellen. Die Bibliothek umfaßt dazu verschiedene Arten von Elementen:

[62]Der Kooperationsvertrag mit der Firma Silicon Software gestattet die kostenlose Nutzung und Anpassung der *VHDL-library* für ATLANTIS.

1. **Schnittstellen zum PCI-Interface:** Sie erlauben dem Benutzer unterschiedliche Kommunikationsmechanismen zwischen der auf dem Host-Rechner laufenden Applikation und ATLANTIS. Beispiele sind Registerzugriffe, DMA-Transfers oder die Verwendung von Interrupts. Zu jedem VHDL-Element korrespondiert ein Funktionsaufruf aus der ATLANTIS-Softwarebibliothek.
2. **Schnittstellen zu externer Logik.** Damit lassen sich die festintegrierten Speicherbausteine auf dem AIB, aber auch die I/O- bzw. Modulsteckverbinder auf AIB und ACB ansteuern und benutzen.
3. **Interne Kommunikationselemente:** Bei der Entwicklung von Anwendungen gibt es immer wiederkehrende Kommunikationsaufgaben zwischen Teilschaltungen. Dazu werden häufig FIFOs (Entkopplung von Schnittstellen), und interne RAM-Elemente benötigt. Sie werden in der *VHDL-library* als optimierte Module zur Verfügung gestellt.

Der prinzipielle Weg um mit der *VHDL-Library* eine Anwendung zu entwickeln, verläuft in mehreren Schritten (Design-Zyklus):

1. **Auswahl eines geeigneten Kommunikationsmoduls:** Abhängig von der jeweiligen Anwendung kann aus einer vorgegebenen Liste eines von acht Basis-Kommunikationsmodellen mit den benötigten Eigenschaften gewählt werden (Tabelle 5.1).

	Register zugriffe	Speicher-Zugriffe (AIB)	Write DMA-on-Demand[63]	Read DMA-on Demand	Interrupt
Modul 1	ja	ja	nein	nein	nein
Modul 2	ja	ja	nein	nein	ja
Modul 3	ja	ja	ja	nein	nein
Modul 4	ja	ja	ja	nein	ja
Modul 5	ja	ja	nein	ja	nein
Modul 6	ja	ja	nein	ja	ja
Modul 7	ja	ja	ja	ja	nein
Modul 8	ja	ja	ja	ja	ja

Tabelle 5.1: Zuordnung von Kommunikationsmechanismen zu Modulen

2. **Erstellen der individuellen Simulationsumgebung:** Zu jedem Kommunikationsmodul wird ein passender Testbench für die Simulation bereitgestellt, der exakt das Verhalten des ausgewählten Kommunikationsmoduls enthält. Der Quellcode des Anwenders wird an ausgewiesener Stelle eingefügt, wobei weitere Module - sowohl interne Module der ATLANTIS-Bibliothek als auch benutzerdefinierte

[63] Demand on DMA bezeichnet einen Burstzugriff zwischen Host und FPGA, der vom FPGA-Design gesteuert und dabei unterbrochen bzw. verzögert werden kann.

Module - beliebig in die VHDL-Hierarchie eingebunden werden können. Die Erstellung der grundlegenden Simulationsumgebung wird durch ein Software-Tool automatisch durchgeführt.

3. **Test-Vektor-Generator:** Für das Testen des Designs müssen wie bei VHDL üblich Stimuli erzeugt werden. Die *VHDL-Library* unterstützt dies in einfacher Weise. Es werden dazu C-Funktionsaufrufe zur Verfügung gestellt. Die Umsetzung in VHDL-konforme Stimuli erfolgt automatisch vor der Simulation. Das besondere daran ist, daß mit den gleichen Funktionen später auch auf die, in der Realität ablaufende Anwendung zugegriffen werden kann. Die Stimuli können aber auch auf herkömmliche Weise in der dafür vorgesehenen Datei editiert werden.

4. **Simulation:** Die Simulation der Anwendung kann grundsätzlich in jedem konventionellen VHDL-Simulator durchgeführt werden. Damit läßt sich die Schaltung anhand der klassischen Waveformausgabe verifizieren. Als Besonderheit bietet die *VHDL-Library* eine Ausgabe der Simulationsergebnisse in eine Datei, die alternativ zur Überprüfung der korrekten Funktion des Designs benutzt werden kann.

5. **Synthese:** Für die Synthese der Designs werden für die *VHDL-Library*-Module vorsynthetisierte Versionen im Netzlistenformat der jeweiligen FPGA-Technologie bereitgestellt. Es müssen also nur die benutzerdefinierten VHDL-Designs kompiliert werden - die ATLANTIS-Bibliotheksmodule werden als externe Komponenten deklariert. Die erzeugten Netzlisten werden zusammen mit den vorsynthetisierten Netzlisten an die jeweilige Place&Route-Software weitergegeben, wo letzt endlich die Abbildung auf die dedizierte FPGA-Architektur durchgeführt wird und die Konfigurationsdaten in Form einer Bitstreamdatei ausgegeben werden.

Zusammenfassend läßt sich sagen, daß mit der *VHDL-library* ein mächtiges und zugleich auch benutzerfreundliches Design-Werkzeug zur Verfügung steht. Aufbauend auf der in Industrie und Forschung weit verbreiteten Hardwarebeschreibungsprache VHDL macht es das ATLANTIS-System einem großen Benutzerkreis zugänglich. Demgegenüber steht die Sprache CHDL als alternative Möglichkeit, Applikationsentwicklung für das ATLANTIS-System zu betreiben, deren Besonderheit der spezielle Support für Software / Hardware-Codesign ist. Im Vergleich zu VHDL werden grundsätzlich keine kommerziellen Simulatoren und Kompiler und auch keine Synthesetools vorausgesetzt.

5.6.3 Die CHDL-Entwicklungsumgebung

Die CHDL-Entwicklungsumgebung [Kor00], die derzeit im Rahmen einer Promotion von Klaus Kornmesser in der Mannheimer FPGA-Gruppe entwickelt wird, unterstützt im besonderen Maße die Abbildung von Anwendungen auf hybride

FPGA / CPU-basierte Systeme, wie ATLANTIS. CHDL wurde FPGA- und System-unabhängig konzipiert und bereits erfolgreich für Entwicklungsarbeiten im Zusammenhang mit dem microEnable-System eingesetzt. Die gesamte Applikation (FPGA-Design und Ansteuerungssoftware) wird dabei in C++ beschrieben. Die Schnittstelle ist der PCI-Bus. Der Test und das Debuggen der Anwendung geschieht völlig transparent - ohne die Notwendigkeit einer separaten Testumgebung. Die bei der Entwicklung benutzte Zugriffssoftware wird ohne Modifikation beim späteren Betrieb wiederverwendet. Als zur Zeit noch zu den hardwarenahen Programmiersprachen zählend, zeigt CHDL seine Stärke in der effektiven Umsetzung von einfach-strukturierten und massiv-parallelen Schaltungen. Zusammen mit dem speziellen Support für Hardware / Software-Co-Designs ist CHDL eine äußerst adäquate Methode zur Entwicklung hybrider Anwendungen.

CHDL baut auf einer C++-Klassenbibliothek auf, die die der FPGA-Familie zugrunde liegenden, kleinsten Logikelemente (Primitive) enthält. Zusätzlich beinhaltet die Bibliothek Makro-Funktionen, wie Zähler, Addierer, Multiplexer oder Speicher, ist aber auch erweiterbar. Die strukturelle Beschreibung erfolgt durch Instanziierung entsprechender Klassen und der Verknüpfung mittels Standard-Operatoren (=, +, XOR, etc.). In Abbildung 5.14 sind zwei Beispiele - ein 3-Bit-Zähler und ein Zustandsdiagramm - gezeigt.

3-bit counter:

```
ExampleCnt3::ExampleCnt3()
{
 //Declarations
 DFF C0("C0",CLK); //D-Flip-Flop, C0->Bit0
 DFF C1("C1",CLK); //D-Flip-Flop, C1->Bit1
 DFF C2("C2",CLK); //D-Flip-Flop, C2->Bit2
 //Equations
 C0 = !C0;
 C1 = C1^C0;
C2 = C2^(C1&C0;)
}
```

State flow-chart:

```
LABEL("LOOP"); //Define starting point
  BeginState();      //Start a new state
    Dout = 1;        //Define state 1 actions
  EndState();
  BeginState();      //Start a new state
    Dout = 3;        //Define state 3 actions
  EndState();
  BeginState();      //Start a new state
    Dout = 5;        //Define state 5 actions
  EndState();
GOTO("LOOP");  //restart
```

Abbildung 5.14: 2 CHDL Beispieldesigns: 3-Bit-Zähler (links) und Zustandsdiagramm (rechts)

Die Erweiterung von CHDL, um Anwendungen auf einem C++-artigen, höheren Abstraktionsniveau zu beschreiben und damit der Schritt zur Hochsprache, ist der Inhalt aktueller Forschungsarbeit.

Da CHDL nicht nur eine einfache Hardwarebeschreibungssprache ist, sondern auf einer C++-Klassenbibliothek basiert, kann die vollständige Anwendung in einem einzelnen C++-Projekt und sogar in einer einzelnen C++-Datei beschrieben werden. Dies ist eine wesentliche Vereinfachung für die Erstellung hybrider Applikationen.

Neben den FPGA-Basis-Klassen unterstützt CHDL eine Reihe von weiteren Funktionsklassen einschließlich Erweiterungsmöglichkeiten:

- PLX PCI9080-Interface-Chip
- PCI Master / Slave-Schnittstellen

- Verschiedenartige DMA-Kanäle und Interruptfunktionalität
- Verschiedenartige Speicherarchitekturen (DPR, FIFOs und ROM)
- Modulschnittstellen
- Clock-Kontrollfunktionen
- FPGA-Konfiguration einschließlich Laufzeit-Rekonfigurationsunterstützung

CHDL ist Kompiler-unabhängig und wurde bisher mit den Entwicklungsumgebungen von Microsoft (Visual C++ 6.0) und Borland erfolgreich eingesetzt. Während der Ausführung wird zwischen drei Modi unterschieden. Im Simulationsmodus läuft die gesamte Applikation auf der Host-CPU. Es muß keine FPGA-Hardware im System vorhanden sein. Im Implementierungmodus wird die Netzliste für die spezifische Place&Route-Software generiert, das Place&Route-Tool automatisch ausgeführt und damit die Konfigurationsdatendatei erzeugt. Im Ausführungsmodus wird die Konfiguration in die FPGA-Hardware geladen. Die Anwendung kann nun getestet werden, wobei zu diesem Zeitpunkt die verschiedenen Teile parallel auf der Host-CPU und in der FPGA-Hardware ausgeführt werden.

Die bereits durchgeführte Anpassung von CHDL für ATLANTIS beschränkte sich in der Erstellung einer zusätzlichen Klassen-Bibliothek für die im ATLANTIS-System vorkommenden FPGAs (Lucent ORCA-FPGA und Xilinx Virtex-FPGA).

5.7 Zusammenfassung

ATLANTIS ist ein hybrides System, daß die Vorteile eines Multi-FPGA-Prozessorsystems und Standard-CPUs in einer Hardware- und Software-Umgebung vereint. Die Haupteigenschaften der Hardware sind dabei, daß die Architektur skalierbar und sehr flexibel aufgebaut ist. In Hinblick auf die Integration zusätzlicher Hardware-Module (beliebige Speichertypen und -architekturen, I/O-Schnittstellen und sonstige Logik) ist sie sehr offen. Die Entwicklung von Anwendungen als auch der Betrieb des Systems wird durch eine anwenderfreundliche Software, im besonderen Maße auch für Nicht-Experten, möglich gemacht.

Die ATLANTIS Systemkomponenten ACB und ATB sind seit September 1999 bzw. seit Februar 2000 im Einsatz. Das AIB befindet sich kurz vor der Produktionsphase und soll ab Oktober 2000 in Betrieb (Aufbau und Test der Hardware) genommen werden. Für das ACB wurden bereits zwei Modulkarten entwickelt und erfolgreich eingesetzt (Abschnitt 6.4.3 und 7.4.3). Ebenso sind die M-Link-I/O-Module (Abschnitt 7.1.3) für die ROB-Anwendung bereits fertig entwickelt und können zeitgleich mit der Fertigstellung des AIBs zum Einsatz kommen[64].

[64]Der Test der M-Link-Module wurde mit dem microEnable-System durchgeführt.

KAPITEL 6

Ausführung des Full-Scan-TRT-Algorithmus' auf ATLANTIS

Das ATLANTIS-System wurde in erster Linie für die Ausführung von ATLAS LVL2-Triggeralgorithmen entwickelt. Die rechenintensivsten Probleme treten bei Triggeralgorithmen bei der B-Physik auf. Der Hauptgrund dafür ist, daß ein vollständiger Scan mindestens eines der inneren Detektoren (TRT / SCT / Pixel) notwendig ist. Zur Zeit vielversprechend ist die Strategie, mit einem Full-Scan des TRT-Detektors zu beginnen. Die prinzipielle Machbarkeit mit einem FPGA / CPU-basierten System als Farmprozessor diese Aufgabe beschleunigt im Vergleich zu einer reinen CPU-Lösung auszuführen, wurde mit der Implementierung auf das ATLANTIS-System gezeigt.

Dieses Kapitel beginnt mit einem Überblick zur B-Physik-Triggeraufgabe bei ATLAS. Die Erstellung der Software-Version des Full-Scan-TRT-Algorithmus' für Standard-PCs wurde parallel zu der hier vorliegenden Promotionsarbeit von Matthias Sessler am CERN durchgeführt. Der Algorithmus ist hier zusammengefaßt dargestellt, weil er die Grundlage für die Implementierung auf ATLANTIS darstellt. Im Anschluß daran folgt der Schwerpunkt dieses Kapitels, mit der ausführlichen Beschreibung der Adaption des Algorithmus' auf das ATLANTIS-System und Benchmark-Ergebnissen. Das Kapitel endet mit einem Ausblick auf eine mögliche FPGA-Koprozessorarchitektur für das finale ATLAS-Triggersystem im Jahr 2005.

6.1 Motivation

In der Vergangenheit wurde bereits mehrfach das Potential der FPGA-Technologie bei der Ausführung von LVL2-Triggeralgorithmen gezeigt (Kapitel 3). Die Ergebnisse des Demonstrator-Programms haben gezeigt, daß prinzipiell ein rein FPGA-basierter Trigger die Anforderungen an den LVL2-Trigger erfüllen kann - daß ein Großteil der Probleme aber auch mit Standard-Prozessoren gelöst werden kann, die über ein kommerzielles Netzwerk miteinander verbunden sind (Prozessorfarm).

Die zentrale Methode, um abhängig von Architektur, Technologien und Triggerstrategien, auf das endgültige Triggersystem schließen zu können, ist das Aufstellen von Papier- bzw. Computermodellen. Die errechneten Ergebnisse für zu erwartende Daten- und Eventraten, Occupancies und Berechnungszeiten geben Aufschluß darüber, wie die generellen Hardware-Anforderungen aussehen und wo die kritischen Stellen in der Trigger-Archiktektur sind.

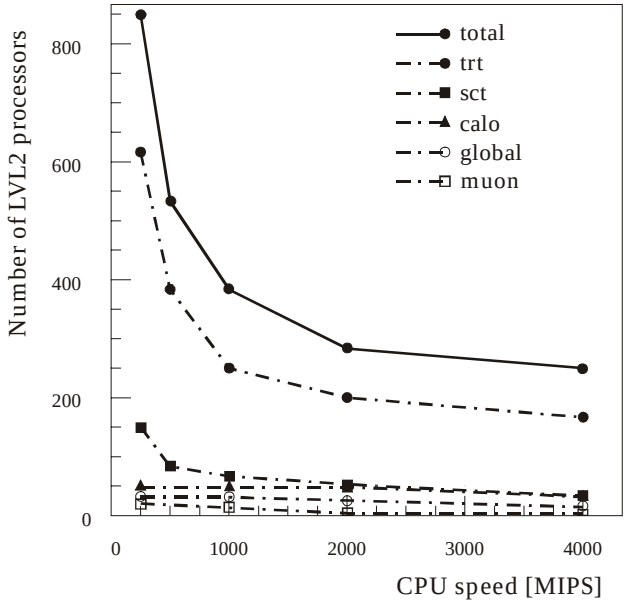


Abbildung 6.1: Anzahl benötigter LVL2-Prozessoren in Abhängigkeit von der Prozessorleistung [ADE98]

Abbildung 6.1 zeigt ein für die hier vorliegende Arbeit wichtiges Resultat der Modelling-Aktivitäten, die zum Abschluß des Demonstrator-Programms in [ADE98] veröffentlicht wurden. Dargestellt ist die Anzahl der benötigten Prozessoren in Abhängigkeit der Prozessorleistung (MIPS[65]) für die auszuführenden Triggeralgorithmen für verschiedene Detektoren. Im Betrieb bei niedriger Luminosität werden mehr als die Hälfte der Prozessorleistung für die im Fall der B-Physik notwendige Ausführung des Full-Scan-TRTs benötigt. Es wird ganz allgemein gezeigt, daß die Zahl benötigten Prozessoren speziell durch die Ausführung der B-Physik-Trigger bei geringer Luminosität bestimmt wird. Neben der aufwendigen Spurensuche im TRT sind auch die Spurverfolgung in den SCT-Detektor und der Full-Scan des Pixel-Detektors dafür mit verantwortlich.

Die daraus abgeleitete Zielsetzung für das nachfolgende Pilot-Projekt war es, eine geeignete Architektur und Technologie zu finden, mit der die Zahl der benötigten Prozessoren verringert werden kann und damit der Einsatz eines technisch-machbaren und zugleich kostengünstigen, kommerziellen Netzwerks im Rahmen des LVL2-Triggers möglich zu machen. Aus Mannheimer Sicht bestand also die Aufgabe darin, in der Praxis zu zeigen, daß durch den Einsatz der FPGA-Technologie in Kombination mit einer Farmarchitektur die Ausführungzeit der B-Physik-Triggeralgorithmen um mehrere Faktoren beschleunigt werden kann. Die Mannheimer

[65]MIPS: Mega Instructions Per Second.

Gruppe hat zur Lösung dieses Problems den Einsatz des hybriden FPGA / CPU-basierten Prozessorsystems ATLANTIS vorgeschlagen. Dieses System soll von außen betrachtet völlig transparent als herkömmlicher Farmrechner in Erscheinung treten und benutzt werden, aber intern FPGA-Resourcen bereitstellen, die die Beschleunigung der dedizierten Algorithmen ermöglicht. Die FPGA-Resourcen werden dabei aus Sicht der CPU als programmierbarer Koprozessor eingesetzt (Koprozessorkonzept). Dieses Kapitel beinhaltet eine detaillierte Darstellung der Implementierung des Full-Scan-TRT-Algorithmus' als Koprozessoranwendung auf ATLANTIS. Die Untersuchungen zur Implementierung weiterer B-Physik-Triggeralgorithmen sind der Gegenstand aktueller Forschungsarbeiten.

6.2 Der B-Physik-Trigger

Mit dem ATLAS-Experiment ist ein breites Spektrum an B-Physik-Untersuchungen geplant. Dies beinhaltet u.a. die Suche nach Hinweisen und Messung der CP-Verletzung im B-Mesonen System. Die B-Physik-Untersuchungen sind schwerpunktmäßig für die ersten Betriebsjahre des LHC vorgesehen - in dieser Phase sind die pileup-Effekte geringer, aufgrund der anfänglich niedriger gewählten Luminosität. Ein für die CP-Verletzung besonders geeigneter Zerfall ist

$$B^0 \rightarrow J/\Psi K_S^0,$$

weil er neben der vorhergesagten starken CP-Asymmetrie, über die Sekundärzerfälle $J/\Psi \rightarrow \mu^-\mu^+$ bzw. $J/\Psi \rightarrow e^-e^+$ und $K_S^0 \rightarrow \pi^+\pi^-$zu einer klaren experimentellen Signatur führt - einem Leptonpaar und einem K_S^0-Zerfall (Abbildung 6.2). Die CP-Verletzung äußert sich in den geringfügig verschiedenen Verzweigungsverhältnissen der Zerfälle $B^0 \rightarrow J/\Psi K_S^0$ und $\bar{B}^0 \rightarrow J/\Psi K_S^0$.

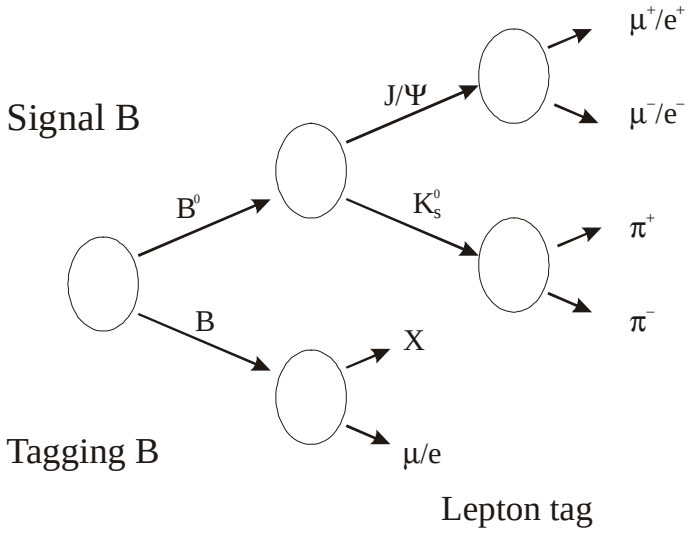


Abbildung 6.2: Experimentelle Signatur des Zerfalls $B^0 \rightarrow J/\Psi K_S^0$

Der B-Physik-Trigger hat die Aufgabe neben dem oben dargestellten Zerfalls-Kanal auf weitere interessierende Kanäle des B-Meson-Zerfalls sensitiv zu sein. Teilchen müsen durch Spurrekonstruktion und Energiemessung identifiziert werden [ADP99].

Ausgangspunkt für den B-Physik-Trigger ist der Hinweis auf mindestens ein Myon in einem Event mit einem Transversalimpuls $p_T > 6$ GeV im Bereich $|\eta| < 2.4$. Diese Information kommt vom LVL1-Trigger und muß zunächst durch den LVL2 bestätigt werden. Die verbleibende Eventfrequenz wird mit ca. 6 -9 kHz angegeben. Der B-Physik-Trigger muß in den akzeptierten Events die geladenen Zerfallsprodukte der verschiedenen möglichen B-Meson-Zerfälle identifizieren. Für diese Aufgabe gibt es keine RoI-Information, was bedeutet, daß das gesamte Detektorvolumen (TRT / SCT / Pixel) zur Spurensuche herangezogen werden muß [ADP99].

Eine essentielle Anforderung an die TRT- und Pixel-Full-Scan-Triggeralgorithmen um die Anzahl von Spurkandidaten und damit die erforderlichen Raten zu erreichen, ist die Suche nach Elektronen mit einem Transversalimpuls $p_T > 0{,}5$ GeV ($J/\Psi(e^+e^-)$-Trigger). Für alle anderen Spurkandidaten liegt die Schwelle bei $p_T > 1{,}5$ GeV. Im Mittel wird erwartet, daß 90 Teilchen pro Event mit einem Transversalimpuls $p_T > 0{,}5$ GeV im Betrieb bei niedriger Luminosität gefunden werden. Die Parameter der gefundenen Spurkandidaten werden an den SCT-Trigger-Algorithmus weitergeleitet. Die geometrischen Informationen dienen als Ausgangspunkt für eine Spurverfolgung in den SCT, um eine weitere Reduktion gefundener Teilchen zu erreichen. Die abschließende Entscheidung des LVL2-Triggers basiert auf den Spurparametern von mindestens zwei oder mehr Spuren. Diese werden dazu benutzt, die invariante Massen zu bestimmen und daraus auf den Zerfall schließen.

Als grundlegender Algorithmus für die Entwicklung von Triggeralgorithmen für den B-Physik-Trigger dient der Offline-Algorithmus xKalman++ [Gav96]. Als Referenzalgorithmus setzt dieser die Maßstäbe in Bezug auf die Effizienz der Mustererkennung des Triggers und damit auch bezüglich der Triggerrate. Dieser Algorithmus, der den gesamten Innendetektor umfaßt, kann nicht direkt als LVL2-Triggeralgorithmus eingesetzt werden, weil die Ausführungszeit weit über einer für die 2. Triggerstufe akzeptablen liegt. Tabelle 6.1 zeigt die Ausführungszeiten für verschiedene untersuchte Algorithmen [Ses00].

Algorithmus	**Ausführungszeit**
XKalman++ (TRT&SCT/Pixel)	11.2 s
Pixel Scan & SCT Kalman	220 ms
TRT LUT Hough & SCT/Pixel Kalman	250 ms

Tabelle 6.1: Ausführungszeiten für verschiedene B-Physik FEX-Algorithmen [Ses00]

Ein Vergleich der verschiedenen Algorithmen bezüglich der Ausführungszeit und der Effizienz ergibt, daß die Spurensuche im TRT mit anschließender Spurverfolgung in den SCT als sehr geeignet für die Ausführung als B-Physik-Trigger-Algorithmus erscheint. Die Implementierung des TRT-Full-Scan-Algorithmus' (TRT-LUT-Hough) für Standard-Prozessoren wurde von Matthias Sessler im Rahmen seiner Promotion durchgeführt. Ein ausführliche Darstellung ist in der

Dissertation von Matthias Sessler [Ses00] zu finden. Dieser fundamentale Algorithmus für den B-Physik-Trigger wurde in der hier vorliegenden Arbeit als Referenzalgorithmus für die Adaption auf das ATLANTIS-System benutzt. Im folgenden Abschnitt wird in einer kurzen Zusammenfassung der Software-Algorithmus dargestellt. Im weiteren Verlauf des Kapitels wird ausführlich die Implementierung auf dem ATLANTIS-System beschrieben.

6.3 Die Full-Scan-TRT-Implementierung für Standard-CPUs

Die Entwicklung des FEX-Algorithmus *TRT-LUT-Hough* für den ATLAS LVL2-Trigger ist der zentrale Teil der Promotionsarbeit von Matthias Sessler. Eine ausführliche Diskussion findet sich in [Ses00]. An dieser Stelle werden in einer Zusammenfassung die für die FPGA-Implementierung relevanten Sachverhalte wiedergegeben. Der Algorithmus wurde für den Einsatz sowohl als RoI-geführter high-p_T-Scan im TRT als auch für den Full-Scan des TRT-Detektors entwickelt. Die Entwicklung erfolgte in der Sprache C++ unter Linux. Der Algorithmus berücksichtigt die Möglichkeit einer Adaption auf FPGAs. Dies wird dadurch erreicht, weil für den ersten, rechenintensivsten Algorithmenschritt (die grundlegende Spurensuche) die Hough-Transformations-Methode angewandt wird, die bereits in früheren FPGA-Implementierungen ihre besondere Eignung gezeigt hat [SDE98]. In diesem Fall hat die Anwendung dieser Methode umgekehrt zu einer bedeutsamen Beschleunigung der Softwareversion des Algorithmus' geführt [Ses00].

Für den hier interessierenden TRT-Full-Scan wird eine Spurensuche im gesamten Volumen des TRT-Detektors gefordert. Der *TRT-LUT-Hough*-Algorithmus ist so konstruiert, daß die beiden TRT-Barrelhälften und die beiden Endkappen getrennt bearbeitet werden. Eine Berücksichtigung des Übergangsbereichs zwischen Barrelhälften und Endkappen zur Erhöhung der Mustererkennungseffizienz steht noch aus. Die grundlegende Algorithmenstruktur, die weitgehend unabhängig von den Detektorteilen Barrel und Endkappen ist, umfaßt die folgenden Schritte[66]:

- **Grundlegende Spurensuche:** Benutzt eine LUT-basierte Hough-Transformation mit Histogrammierung zur Identifikation potentieller Spurkandidaten
- **Lokale Maximum-Suche:** Selektiert potentielle Spurkandidaten und eliminiert mehrfach gefundene Spuren
- **Spur-Splitting:** Entfernt versehentlich zugeordnete Hits zu einer Spur und trennt fälschlicherweise kombinierte Teilspuren
- **Abschließende Selektion und Spur-Fit:** Selektiert abschließend die potentiellen Spurkandidaten.

[66] Aufgelistet sind die Algorithmen-Schritte für den TRT-Barrel. Im Unterschied dazu beinhaltet der Algorithmus für die Endkappen einen zusätzlichen Schritt zur erweiterten Maximum-Suche [Ses00].

Die prinzipielle Arbeitsweise des Algorithmus' wird nachfolgend anhand der TRT-Barrels erklärt. Die Überlegungen zu den Endkappen sind analog, nur liegt eine andere Detektorgeometrie zugrunde. Details finden sich in [Ses00].

Der TRT-Barrel besteht aus 52544 Driftkammern (Straws), die beidseitig ausgelesen werden (linker und rechter Barrel). Die Straws sind auf ringförmigen Ebenen um die Strahlachse angeordnet. Der Abstand der innersten Ebene zur Strahlachse beträgt r = 56 cm, der der äußersten r = 103 cm. Der Abstand der Straws, die selbst einen Durchmesser von 4 Millimetern haben, beträgt auf einer Ebene im Mittel 6,8 Millimeter - ihre Anzahl nimmt von der innersten der 73 Ebenen (480) bis zur äußersten (928) kontinuierlich zu. Die Position eines Straws wird geometrisch durch einen Winkel ϕ um die Strahlachse und einen Abstand r als Abstand zur Strahlachse definiert. Die Aufgabe der einzelnen Algorithmenschritte ist wie folgt:

Grundlegende Spurensuche: Der erste Algorithmenschritt führt eine grundlegende Spurensuche im gesamten Detektorvolumen durch. Dazu wird eine LUT-basierte Hough-Tranformation ausgeführt. Die Hough-Transformation ist eine Standard-Methode in der Bildverarbeitung, mit der die Erkennug globaler Muster in einem abstrakten Bildraum durch die Erkennung lokaler Muster in einem transformierten Parameterraum erreicht wird. Im hier vorliegenden Fall werden mit der Hough-Transformationsmethode die aktiven Straws (Hits) des (r,ϕ)-Raums auf den Teilchenspurraum (ϕ, $1/p_T$) abgebildet. Vereinfacht ausgedrückt werden für jedes vordefinierte Spurmuster die Anzahl der aktiven Straws aufsummiert, durch die die Spur gebildet wird. Die benötigte LUT besteht aus 96000 (ϕ x 1 / pT = 1200 x 80) vordefinierten Spurmustern. Dabei werden für jeden Straw bis zu maximal 130 korrespondierende Spurmuster (bins) abgespeichert.

Während der Ausführung führt das Anlegen jedes aktiven Straws (Hits) an die LUT dazu, daß die korrespondierenden Adressen der Spurmuster zurückgegeben werden. Für die entsprechenden Spurmuster werden mit dieser Information zugeordnete Zähler erhöht. Das Ergebnis ist ein Histogramm im (ϕ, $1/p_T$)-Raum. Potentielle Spurkandidaten sind diejenigen Spurmuster, deren Zählerinhalt (Histogrammzelle) am Ende dieses Algorithmenschrittes über einer festen Schwelle (14) liegt. Abbildung 6.3 zeigt das Histogrammierergebnis für ein einzelnes 3-GeV-Myon im TRT-Barrel.

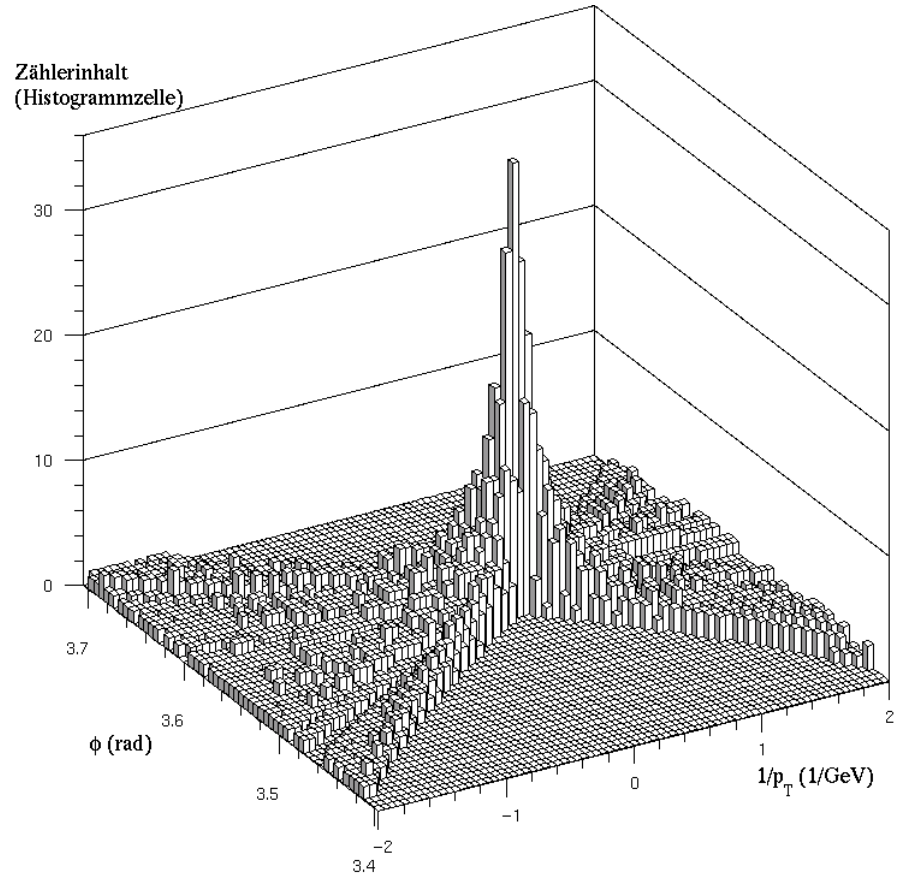


Abbildung 6.3: Histogrammierergebnis für ein einzelnes 3-GeV-Myon im TRT-Barrel [Ses00]

Lokale Maximumsuche: Die Motivation für diesen Algorithmenschritt ist, daß viele lokale Maxima im (ϕ, $1/p_T$)-Raum von weiteren Einträgen mit einem Zählerstand über der Schwelle in direkter Nachbarschaft ($1/p_T$, ϕ) umgeben sind (siehe Abbildung 6.3). Der Grund dafür ist in der Definition der Spurmuster zu suchen. Benachbarte Spurmuster besitzen einen Überlapp von 30% bis 50% in $1/p_T$ und ϕ, um eine Erhöhung der Effizienz bei der Spurensuche zu erreichen. Ziel dieses Schrittes ist, lokale Maxima zu finden und im Histogramm direkt benachbarte, über dem Schwellwert liegende Kandidaten zu entfernen. Allein mit der lokalen Maximumsuche wird eine Reduktion der potentiellen Spurkandidaten um den Faktor 10 erzielt.

Spur-Splitting: In diesem Schritt werden die zu den verbliebenen Spurkandidaten zugehörigen Hits analysiert. Dazu wird eine zweite LUT benötigt, in der für jedes Spurmuster die zugehörigen Straws aufgelistet sind. Zusätzlich wird einmal pro Event eine Hash-Tabelle generiert, in der für jeden Straw abgespeichert wird, ob er getroffen wurde (aktiv = 1) oder nicht (inaktiv = 0). Das Kriterium für Track-Splitting ist folgendermaßen definiert: Beinhaltet ein Spurkandidat in >= 9 aufeinanderfolgenden Ebenen keinen Hit wird er aufgeteilt in zwei separate Spurkandidaten. Daraufhin wird wieder das Schwellwert-Kriterium (mindestens 14 Hits) aus Schritt 2 angewendet. Liegen beide Kandidaten über der Schwelle, wird derjenige auf den

äußeren Radien verworfen. Die Reduktion des Spur-Splittings liegt bei ungefähr Faktor 2.

Abschließende Selektion und Spur-Fit: Aufgrund der mechanischen Anordnung der Driftröhren in z-Richtung (Strahlrichtung) kann zu einer weiteren Reduktion der Spuranzahl der kleinste Radius der Hits eines Spurmusters benutzt werden. Damit werden in erster Linie Spurkandidaten, deren Startpunkt auf einer der äußersten Ebenen liegt als Hintergrund-Rauschen bewertet und verworfen. Hierdurch wird eine weitere Reduktion um den Faktor 1.2 erreicht. Für den abschließenden Spur-Fit wird ebenfalls eine Reduktion um den Faktor 1.2 angegeben. Für die Ausführung des Spur-Fits wird ein Polynom dritten Grades benutzt. Eine Korrektur dritten Grades wird deshalb benötigt, weil die low-p_T-Spuren nicht als gerade Linie approximiert werden können. Es ist anzumerken, daß keine Driftzeitinformation für den Fit verwendet wird. Nach dem Fit wird nochmals das Schwellwert-Kriterium angewandt.

Interessant, insbesondere zum Vergleich mit der FPGA-Implementierung, sind die Ausführungszeiten der einzelnen Algorithmenschritte auf einer Standard-CPU. In Tabelle 6.2 sind die Zeiten für die Ausführung auf einem 300-MHz-PentiumII-Prozessor aufgelistet - dabei handelt es sich um B-Physik-Events mit pile-up bei geringer Luminosität [Ses00].

Algorithmenschritt	**Barrel**	**Endkappen**
Grundlegende Spurensuche	35 ms	96 ms
Lokale Maximum-Suche	7 ms	13 ms
Spur-Splitting	3 ms	52 ms
Erweiterte Maximum-Suche	-	8 ms
Abschließende Selektion und Spur-Fit	2 ms	5 ms
Vollständiger Algorithmus	47 ms	174 ms

Tabelle 6.2: Full-Scan-TRT-Ausführungszeiten (C++-Version) 300-MHz-PentiumII-CPU [Ses00]

6.4 Die Full-Scan-TRT-Implementierung auf ATLANTIS

Das Ziel der Implementierung des Full-Scan-TRT-Algorithmus auf dem ATLANTIS-System ist, die Software-Version so zu adaptieren, daß ein bedeutsame Verkürzung der Ausführungszeit nachgewiesen wird. Dabei soll die Qualität der Triggereffizienz im Vergleich zur reinen CPU-Implementierung erhalten bleiben. Eine Reihe von Fragen zu Funktionalitätsstudien und Implementierungsideen wurden sowohl bezüglich der CPU- als auch der ATLANTIS-Version von Matthias Sessler und dem Autor diskutiert und beantwortet. An der nachfolgend dargestellten Adaption und Ausführung auf ATLANTIS waren maßgeblich der Autor und Harald Simmler beteiligt.

6.4.1 Überblick zur Implementierung

Für den Benchmark auf dem ATLANTIS-System wurde der TRT-Barrel-Algorithmus gewählt. Der wesentliche Unterschied zwischen einer Barrel- und Endkappen-Implementierung besteht im Aufbau der jeweiligen LUT - die Algorithmenstruktur ist davon nicht beeinflußt. Der Barrel ist logisch in zwei symmetrische Hälften aufgeteilt. Die im folgenden gemachten Angaben beziehen sich jeweils auf eine Barrelhälfte, ausgenommen es wird explizit der gesamte Barrel genannt.

Aus einer ersten Abschätzung ergibt sich, daß unter Berücksichtigung der wichtigsten Randbedingung die ersten beiden Schritte des TRT-CPU-Algorithmus' am geeignetsten für die Ausführung auf dem FPGA-basierten Teil des ATLANTIS-Systems sind - nämlich auf einem ATLANTIS Computing-Board. Hauptgründe dafür sind der FPGA-Resourcenbedarf und gleichzeitig die erreichbare Beschleunigung. Die ersten beiden Schritte beanspruchen 89% der Gesamtausführungszeit der reinen CPU-Lösung. Das Ziel ist es, die Ausführung von Schritt 1 und 2 um mehr als einen Faktor 10 zu beschleunigen. Die weiteren beiden Schritte sind im CPU-Teil des ATLANTIS-Systems auszuführen. Der Beitrag aus der Beschleunigung der weiteren beiden Schritte in Bezug auf den FPGA-Implementierung rechtfertigt den unverhältnismäßig hohen zusätzlichen Aufwand nicht, beziehungsweise eine weitere Beschleunigung ist mit der vorhandenen FPGA-Technologie nicht effizient möglich. Das liegt an den beschränkten Logikresourcen der zur Zeit eingesetzten FPGAs und wird hauptsächlich durch den hohen Resourcenbedarf der für den letzten Schritt benötigten Floating-Point-Arithmetik verursacht (vgl. Abschnitt 7.2.2). Die für die kommenden ein bis zwei Jahre angekündigten neueren, leistungsfähigeren FPGAs beinhalten signifikant erhöhte Logikresourcen und machen zumindest die zusätzliche Ausführung des dritten Algorithmen-Schrittes in FPGAs effizient möglich. Die Skizzierung einer Alternativ-Implementierung des TRT-Algorithmus auf der Basis zukünftiger FPGA-Technologie findet sich in [HKL00].

6.4.2 Abbildung auf das ATLANTIS Computing-Board

Für die Abbildung des Algorithmus' werden die Logikresourcen eines ACBs zu Grunde gelegt. Die ersten beiden Algorithmenschritte müssen an die geänderte Hardware-Architektur angepaßt werden. Auch bei der FPGA-Implementierung wird eine Hough-Transformation mittels Histogrammier-Methode und eine lokale Maximum-Suche durchgeführt - allerdings basierend auf einer in der Organisation geänderten LUT. Dabei ist jedem Spurmuster ein Daten-Bit zugeordnet. Die unteren 16-Adress-Bit (2^{16} = 64k-Adressraum für 52544 Straws) werden für die Kodierung der Straws benutzt. Auf einem ACB kann somit in einem Takt eine aktiver Straw auf maximal 712 Spurmuster abgebildet werden, was zur Inkrementierung zugeordneter Histogrammierkanäle genutzt wird. Um die notwendige Auflösung zu erreichen und damit die geforderte Anzahl von Spurmustern zu unterstützen, werden zur

Wahl weiterer Spurmusterräume die oberen drei Adress-Bit eingesetzt (vgl. Abschnitt 6.4.5).

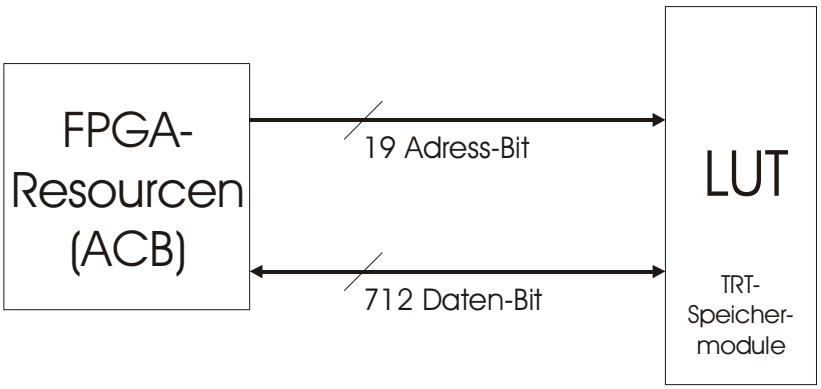


Abbildung 6.4: Skizzierung der Architektur aus FPGAs und Speicher-Modulen

Zur Speicherung der Spurmuster in der LUT wurden spezielle Speicher-Module entwickelt (Abschnitt 6.4.3). Mit dieser Architektur aus FPGAs und Speicher-Modulen kann der besondere Vorteil der massiven Parallelisierbarkeit und der Implementierung einer tiefen Pipeline bei der Ausführung auf FPGAs ausgenutzt werden. Der Adressraum hat eine Größe von 512k und die integrale Daten-Bit-Breite beträgt 712 (Abbildung 6.4).

6.4.3 Das TRT-Speicher-Modul

Das TRT-Speicher-Modul wurde in Mannheim spezifiziert und am Weizmann-Institut in Rehovot / Israel entwickelt und aufgebaut. Die mechanischen Ausmaße sind so gewählt, daß jeweils ein Modul-Port auf dem ACB belegt wird. Somit kann ein ACB mit vier TRT-Speicher-Modulen bestückt werden.

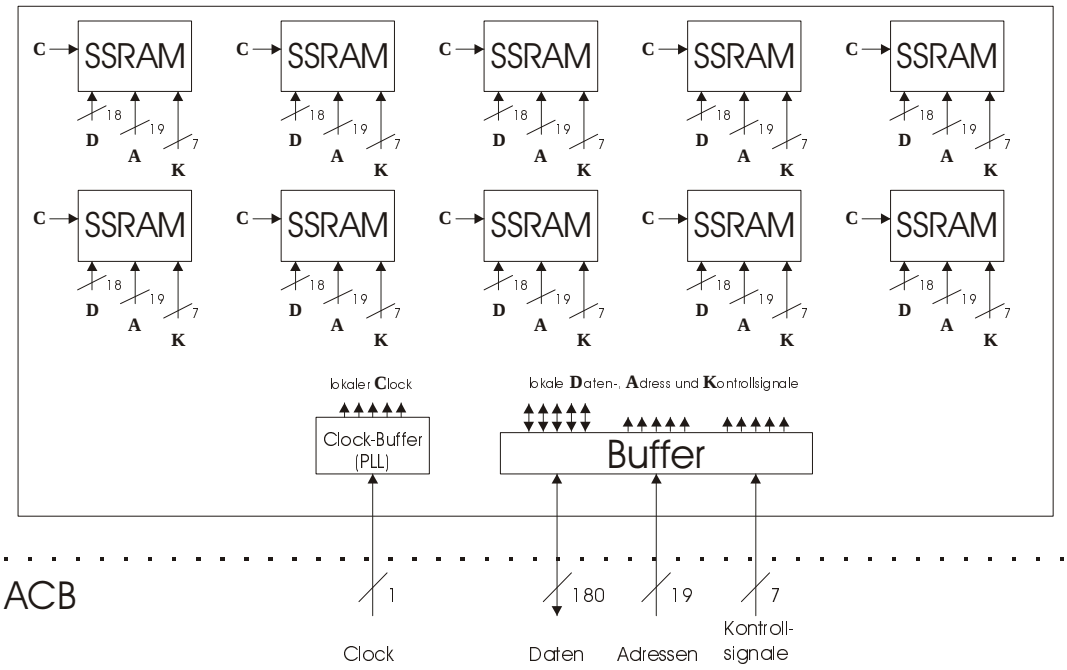


Abbildung 6.5: Blockdiagramm des TRT-Speicher-Moduls

Die Platine ist mit 10 SSRAM-Bausteinen des Typs KM718V989 [Sam98] mit der Organisation 512k x 18 bestückt, die zusammen eine Speicherdichte von 44 MByte aufweisen. Die Organisation eines Speicher-Moduls ist 512k x 180 - der Zugriff auf alle Daten-Bits erfolgt parallel (Abbildung 6.5). Allerdings können nur 178 der 180 Bit vom ACB aus benutzt werden, weil versehentlich 2-Daten-Bits FPGA-Pins zugeordnet wurden, die nur als Ausgang geschaltet werden können. Zur Verteilung der Kontroll- und Clocksignale werden Buffer-Bausteine vom Typ SN74ALB16244DL [Tex97] und der Clockbuffer-Baustein CY2308 [Cyp98c] von Cypress eingesetzt [Kug99]. Eine Photo der Platine findet sich in Anhang B.

Für den Test und für das später für die Algorithmus-Ausführung notwendige Laden der LUT wurden die Speicherzugriff-FPGA-Designs in der Sprache CHDL (Abschnitt 5.6.3) beschrieben und erzeugt. Die Entwicklung und der Test der Designs zusammen mit den TRT-Speicher-Modulen wurden von Klaus Kornmesser und dem Autor durchgeführt.

6.4.4 Basis-Parameter

Die ATLANTIS-Implementierung und die CPU-Implementierung unterscheiden sich im Besonderen bei der Ausführung des ersten Algorithmenschritts. Die CPU bildet jeden Hit, auf die dem Straw zugehörigen Spurkandidaten ab und erhöht die korrespondierenden Histogrammzähler. Der Prozeß ist völlig sequentiell und die Ausführungszeit skaliert linear mit der Anzahl der Hits und der Anzahl der gesuchten

Spurmuster. Für die ATLANTIS-Implementierung wurde ein Ansatz gewählt, der im Besonderen den hohen Parallelisierungsgrad durch den Einsatz von FPGAs ausnutzt. Mit der in den TRT-Speicher-Modulen geladenen LUT können bis zu 712 Histogrammierzähler pro ACB gleichzeitig inkrementiert werden. Die aktuelle Implementierung verwendet nur 594, auf Grund der beschränkten Logikresourcen des Lucent-OR3T125-FPGAs[67]. Um die geforderte Anzahl der Spurmuster zu erreichen, werden acht Durchläufe (Passes) durchgeführt, während derer die gleichen Hits auf acht verschiedene Spurmusterräume abgebildet werden.

Eine wichtige Tatsache ist, daß ein einzelner Hit nur zu Spurmustern in einem dedizierten ϕ-Suchbereich zugehörig ist. Dem wird durch die Einführung des Konzepts der Pseudo-RoIs Rechnung getragen. Die Einführung von Pseudo-RoIs heißt, den Barrel in logische Bereiche in ϕ zu unterteilen. Die Wahl der Größe dieser Bereiche hängt von der Form der Spurmuster ab. Die kleinstmögliche Bereichsgröße in ϕ wird eingeschränkt durch die Spurmuster mit dem kleinsten p_T und damit dem kleinsten Krümmungsradius. Um die über den Grenzbereich von zwei Pseudo-RoIs verlaufenden Spurmuster nicht zu verlieren, werden die Hits aus dem Überlappungsbereich der linken und rechten Nachbar-Pseudo-RoIs mitberücksichtigt. Abbildung 6.6 verdeutlicht diesen Sachverhalt.

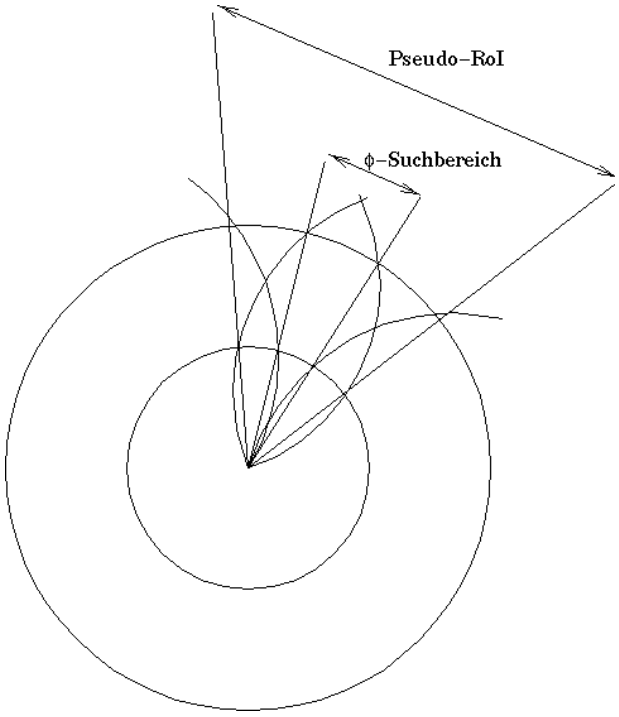


Abbildung 6.6: Definition von Pseudo-RoIs

Die Einführung der Pseudo-RoIs führt zu einem bedeutsamen Vorteil für die Implementierung auf FPGAs. Der TRT-Barrel-Detektor besitzt grundsätzlich eine 32fache Symmetrie. Durch die Einteilung in Pseudo-RoIs unter Ausnutzung der Symmetrie

[67]Ursprünglich waren in dem BGA600-Gehäuse noch komplexere FPGA-Typen (Or3T165 und Or3T225) angekündigt, mit denen alle 704 Speicher-Bits hätten ausgenutzt werden können. Diese wurden von Lucent aber wieder abgekündigt.

kann bei der Berechnung der Spurmuster und der Generierung der LUT deren Größe bis um einen Faktor 32 reduziert werden. Zur Zeit wird bei am CERN verwendeten Simulations-Softwaretools die Symmetrie noch nicht ausgenutzt. Die Konsequenz ist, daß ein entsprechend erhöhter Speicherbedarf für die LUT benötigt wird. Für die FPGA-Messungen wurde die Symmetrie mit der Einführung der Pseudo-RoIs bereits berücksichtigt.

In einer TRT-Barrelhälfte beträgt die Anzahl der Straws und damit gleichzeitig die Anzahl der Auslesekanäle 52544. Es wird vorausgesetzt, daß die TRT-Eventdaten vorverarbeitet, nach Pseudo-RoIs gruppiert und Nullen-unterdrückt sind und daß jeder aktive Straw (Hit) durch eine 16-Bit-Straw-Adresse repräsentiert wird[68]. Bei einer mittleren Occupancy von 4% beträgt die Anzahl der Hits pro Event 2048. Die Größe eines Eventdatenpakets beträgt somit 4 kByte. Im Mittel werden durch die ersten beiden Algorithmenschritte 25 potentielle Spurkandidaten gefunden. Diese werden durch ein 20-Bit-Wort charakterisiert. Die Übertragung über den PCI-Bus erfolgt im 32-Bit-Format pro Spurkandidat. Bezogen auf die angegebenen Parametern ergibt sich eine mittlere Zahl von 100 Spurmuster zu den ein einzelner Straw zugehörig ist. Die grundlegenden Parameter der aktuellen Implementierung sind in der nachfolgenden Tabelle 6.3 zusammengefaßt. Weitere Erklärungen zu dem Tabelleninhalt finden sich in nächsten Abschnitt.

Parameter	**Wert**	**Kommentar**
Größe der Suchraums/Auflösung	76.000	950 ϕ x 80 p_T *(siehe Fußnote[69])*
Anzahl Pseudo-RoIs	16	beinhaltet 4750 Spurmuster
Parallele Histogrammierkanäle	594	27 Basis-Module mit jeweils 22 Zählern
Anzahl Durchläufe/P.-RoI	8	passes
Überlappungsfaktor (ÜF)	2	Aktiver P.-RoI + 50% Nachbar-P.-RoIs
Hits pro Event	2.048	
Eingangsdatenvolumen/Event	64 kByte	Hits x 2 Byte x passes x ÜF
Spurkandidaten pro Event	25	
Ausgangsdatenvolumen/Event	100 Byte	

Tabelle 6.3: Parameter für die Full-Scan-TRT-Implementierung auf ATLANTIS (Barrelhälfte)

[68]Die gemachten Annahmen beruhen auf den gleichen Voraussetzungen, wie sie in der offiziellen LVL2-Reference-Software gemacht werden.

[69]Die verwendete Anzahl von 76000 Spurmustern ist der optimale Wert bezogen auf die nutzbaren Resourcen eines ACBs. Die reduzierte Auflösung im Vergleich mit der CPU-Version ist akzeptabel. In [Ses00] wird angegeben, daß eine Halbierung der Auflösung zu einem Effizienzverlust von nur wenigen Prozent führt. Mit der Einteilung in 16 Pseudo-RoIs und 8 Durchgängen pro Pseudo-RoI können insgesamt 76032 Spurmuster berücksichtigt werden. Es werden mit der angegeben Auflösung aber tatsächlich nur 76000 Teilchenspuren gesucht.

6.4.5 Details der FPGA-Algorithmus-Implementierung

Die Struktur und der Ablauf der FPGA-Implementierung sind eng miteinander verknüpft und werden deshalb in diesem Abschnitt zusammen beschrieben. Der FPGA-Algorithmus besitzt grundsätzlich einen modularen Aufbau. Das Basis-Modul besteht aus einer Histogrammier- und einer Maximum-Einheit und wird durch ein nachgeschaltetes FIFO ergänzt (Abbildung 6.7). Für die Abbildung auf das ACB werden insgesamt 27 der Basis-Module in die FPGAs implementiert.

Der grundsätzliche Algorithmenablauf teilt sich für jeweils ein zu bearbeitendes Event in die Spurensuche in jedem der 16 Pseudo-RoIs auf, die sequentiell abgearbeitet werden. Für jedes Pseudo-RoI werden die Hitdaten in 8 Durchläufen auf 8 verschiedene Spurmusterräume abgebildet und die lokale Maximumsuche durchgeführt. Die gefundenen Spuren werden in einem Ausgangs-FIFO gesammelt und erst nach Abarbeitung des vollständigen Events über den PCI-Bus (zur Weiterverarbeitung auf der CPU) in den Arbeitsspeicher zurückgeschrieben.

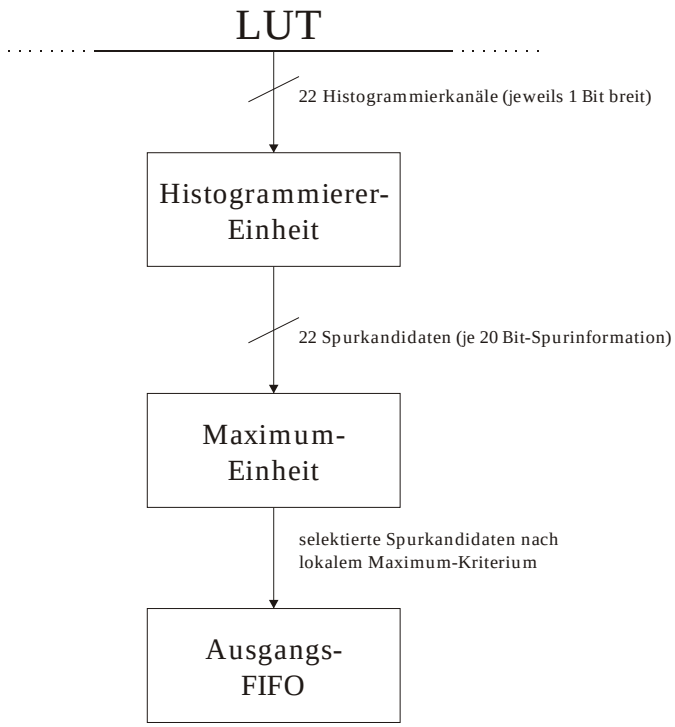


Abbildung 6.7: TRT-Basis-Modul bestehend aus Histogrammier-, Maximum- und FIFO-Einheit

Die Basis-Module werden von jeweils 22 Daten-Bits aus den TRT-Speicher-Modulen gespeist. Das bedeutet, daß jedes Modul gleichzeitig 22 Spurmusterkanäle in der Histogrammier-Einheit bearbeitet. Ist die Histogrammierphase für einen Durchgang abgeschlossen, werden die Informationen zu den 22 Spurmustern an die Maximum-Einheit übertragen. Das 20-Bit-Spurformat besteht aus der Hit-Anzahl (counter-value), einer Modul-Spur-ID, einer Modul-ID, einer FPGA-ID und einen Schwellwert-Flag (threshold-bit).

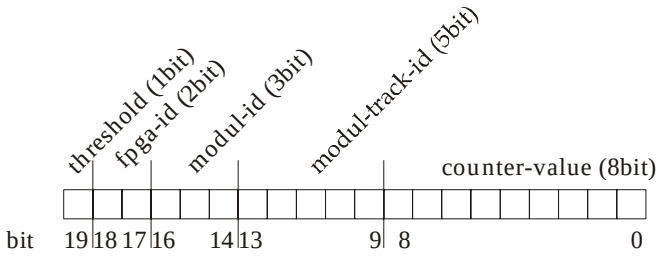


Abbildung 6.8: Spurmuster-Datenformat

Die Maximum-Einheit wird nach jedem der insgesamt 8 Durchläufe aktiviert, und zwar insofern, daß die Informationen zu den 22 Spurmustern parallel von der Histogrammier-Einheit übernommen werden. Nach dem dritten bis zum achten Durchlauf wird parallel auf alle Spuren ein 3x3-Maximum-Filter-Operation angewandt. Ziel dieses Filters ist es, jede Spur hinsichtlich ihrer Hit-Anzahl mit ihren 8 nächsten Nachbarn zu vergleichen. Das Ergebnis ist, daß die jeweilige Spur entweder die höchste Hit-Anzahl aufweist und damit als lokales Maximum markiert wird (Schwellwert-Flag = 1) oder einer der acht nächsten Nachbarn besitzt mehr Hits und die jeweilige Spur wird als Minimum klassifiziert (Schwellwert-Flag = 0). Bei gleicher Hit-Anzahl wird die Spur auch als Maximum markiert.

Von der Maximum-Einheit werden alle Spuren mit gesetztem Schwellwert-Flag ins FIFO übertragen. Die FIFOs der Basis-Einheiten sind untereinander verbunden. Am Ende jeder Pseudo-RoI-Phase werden die Spurendaten aller FIFOs ausgelesen und in ein globales FIFO übertragen, aus dem am Ende der Eventphase die Spuren über den PCI-Bus in der Hauptspeicher der CPU übertragen werden. Auf diese Spurdaten werden von der CPU die letzten beiden Algorithmenschritte ausgeführt.

Die Steuerung der Eventverarbeitung teilt sich auf in die Kontrolle auf Eventebene und auf Pseudo-RoI-Ebene. Die sequentielle Verarbeitung der 16 Pseudo-RoIs wird mit Hilfe eines 4-Bit-Zählers kontrolliert. Der vergleichsweise komplexere Ablauf einer Pseudo-RoI-Phase wird durch eine Zustandsmaschine (FSM) gesteuert, die in Abbildung 6.9 dargestellt ist. Nach einem RESET befindet sich die FSM im Zustand *IDLE*. Bei der Übertragung der Hitdaten für jedes Pseudo-RoI wird durch das erste übertragene Byte die Anzahl der Hitdaten angegeben. Die Anzahl wird im Zustand *READ Length* dekodiert und daraufhin der erste Durchlauf gestartet. Dabei werden wie oben beschrieben das Histogrammieren und die lokale Maximum-Suche durchgeführt. Dies wiederholt sich bis zum 8. Durchlauf. Danach werden im Zustand *OUTPUTDATA* die sich in den lokalen FIFOs befindlichen Spuren in das Ausgangs-FIFO geschrieben und wieder in den Zustand *IDLE* zurückgesprungen. Sind alle 16 RoIs abgearbeitet, wird der Inhalt des Ausgangs-FIFOs in den PC-Speicher übertragen und das Event ist fertig bearbeitet.

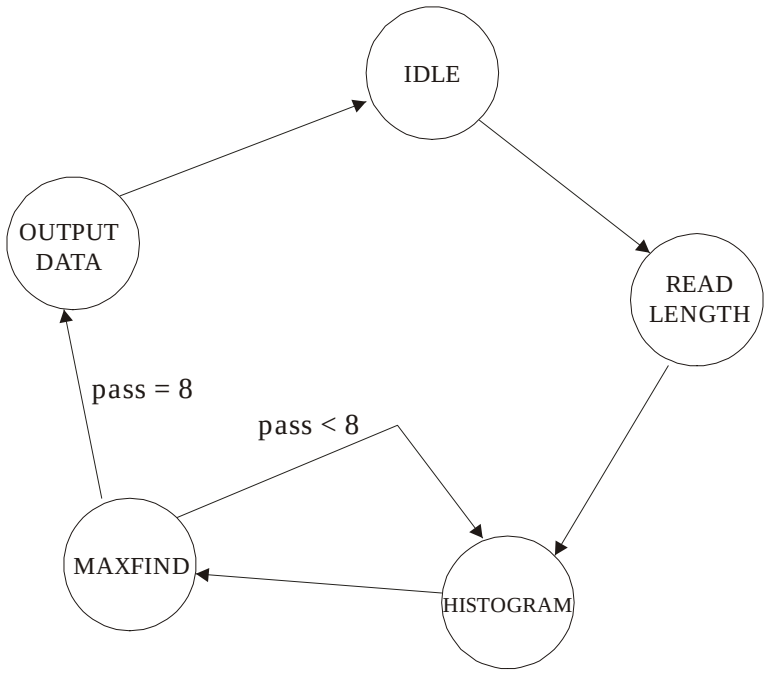


Abbildung 6.9: Zustandsmaschine (FSM) zur Ablauf-Steuerung eine Pseudo-RoI-Bearbeitung

Ein prinzipieller Nachteil bei der Definition von Modulen ist die Beschränkung auf die 22 x 8 großen Spurmusterbereiche. Dadurch, daß immer nur ein abgegrenzter Spurmusterraum von der Maximum-Einheit bearbeitet wird, ist die Reduktionsrate schlechter als bei der CPU-Implementierung. Der Grund ist, daß der 3x3 Filter über die Randbereiche nicht angewandt werden kann. Dies hat zur Folge, daß die nachfolgenden Algorithmenschritte eine größere Reduktion leisten müssen. Durch einen einfachen Mechanismus kann die Reduktionsrate auch bei der jetzigen FPGA-Implementierung deutlich verbessert werden. Der geänderte Aufbau der LUT und die dazu notwendige Erweiterung der Maximum-Einheit kann so gewählt werden, daß die Randspuren in allen Modul-Einheiten überlappen. Die zusätzlich benötigten Speicherresourcen belaufen sich auf ca. 10%, wenn in diesem Fall in einer Basis-Einheit 20 anstatt 22 Spurmuster bearbeitet werden. Um die Überlappung von Spuren aus aufeinander folgenden Durchläufen zu erreichen müssen die Ergebnisse aus den vorangegangenen zwei Durchläufen in der Maximum-Einheit zwischengespeichert werden. Die dazu benötigte Schaltung führt nur zu einer unwesentlichen Erhöhung der Logikresourcen in den FPGAs.

6.4.6 Erstellung der FPGA-Designs

Für die Entwicklung der FPGA-Designs wurde die Sprache VHDL benutzt. Die Umsetzung des TRT auf ATLANTIS (von der VHDL-Beschreibung bis zu den Messungen) wurden in einer gemeinschaftlichen Arbeit vom Autor und von Harald

Simmler durchgeführt. Die Entwicklung wurde unter Verwendung der *VHDL-library* (Abschnitt 5.6.2) in sofern unterstützt, daß das vollständige PCI-Lokal-Bus-Interface bereits vorgegeben war. Es konnte also direkt mit der Erstellung des TRT-Designs begonnen und diese durch simulierte Zugriffe über das PCI-Interface getestet werden. Zusätzlich wurde noch ein Simulationsmodell der TRT-Speicher-Module entwickelt. Damit ist eine vollständige Simulation des TRT-Algorithmus' auf einem ACB möglich.

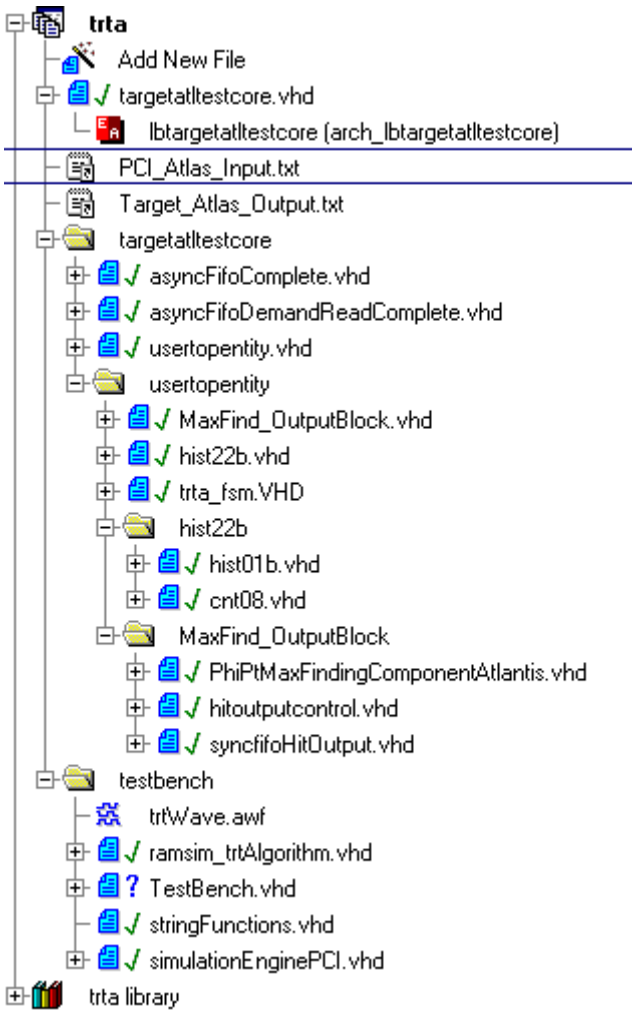


Abbildung 6.10: Datei-Struktur der TRT-VHDL-Designs

Abbildung 6.10 zeigt die Datei-Struktur der VHDL-Designs. Die Beschreibung des TRT-Algorithmus' ist hierarchisch gegliedert. In der obersten Ebene wird in der Datei targetatltestcore.vhd die Schnittstelle zwischen Algorithmus und PCI-Interface zur Verfügung gestellt. Der Algorithmus selbst ist dreifach gegliedert in die Histogrammier-Einheit (hist22b.vhd), die Maximum-Einheit (MaxFind_OutputBlock.vhd) und eine Kontroll-Einheit (trta_fsm.vhd). Über die Datei usertopentity.vhd sind die drei Einheiten miteinander gekoppelt. Die genannten drei Unter-Einheiten verfügen teilweise noch über eine weitere Hierarchieebene.

Für die Simulation wurden für das RAM-Simulations-Modell eigene Test-Eventdaten erzeugt, die später auch für die Messungen auf dem Board benutzt wurden. Die Designs wurden mit dem VHDL-Kompiler *Active CAD* der Firma Aldec simuliert und getestet. Für die Design-Synthese wurde das Tool *Synplify* der Firma Synplicity benutzt. Der abschließende Place&Route-Prozeß wurde in der Software *Foundry* des FPGA-Herstellers Lucent ausgeführt.

6.5 Benchmark des Full-Scan-TRT-Algorithmus'

Für die Messungen wird die Existenz der LUT mit den vordefinierten Spurmustern essentiel vorausgesetzt. Die LUT muß an die Speicherstruktur (TRT-Speicher-Modul) und an die Algorithmenstruktur angepaßt werden. Dies leistet das von Matthias Sessler entwickelte Programm LUT-GEN, daß abhängig von der Detektor-Geometrie und der gewünschten Auflösung in ϕ und p_T, parameter-gesteuert, die gewünschte LUT erzeugt. Da für die aktuelle FPGA-Messung die Detektor-Symmetrie vorausgesetzt wird, mußten eigene Test-Eventdaten erzeugt werden, woraus die LUT generiert wurde.

Die maximale Frequenz, mit der die FPGA-Designs ausgeführt werden können, wird von der Place&Route-Software mit 19 MHz angegeben und in der Praxis bestätigt. Die Datenübertragung über den PCI-Lokal-Bus wird mit der maximalen Frequenz von 40 MHz durchgeführt. Die Entkopplung wird mittels asynchroner FIFOs realisiert (sind Bestandteil des PCI-Interface-Logik im FPGA).

Aus Sicht der Host-CPU werden die Hitdatenpakete als Burst-Transfer auf das ACB übertragen und die Ergebnisse (gefundene Spuren) ebenfalls als Burst-Transfer wieder zurückgelesen. In beiden Fällen wird der DMA-Modus der PLX-PCI9080-Bausteins benutzt, womit ein direkter Transfer vom ACB zum Host-Speicher (Read) und umgekehrt vom Hostspeicher zum ACB (Write) durchgeführt wird. Das Performance-Problem bei DMA-Übertragungen wurde bereits in Abschnitt 5.1.2 diskutiert.

6.5.1 Software Voraussetzungen und Randbedingungen

Alle Messungen (CPU-Version und ATLANTIS-Version) wurde unter dem Betriebssystem Windows NT und sofern möglich, unter Verwendung der LVL2-Trigger-Referenz-Software Version 1.51 ausgeführt. Teilweise wurden auf Grund des Zustands der Referenz-Software für Windows NT Messungen unter Verwendung selbstentwickelter Testumgebungen durchgeführt[70]. Zur Vereinfachung der Konfiguration wurde eine stand-alone Applikation (TRTFEX) benutzt, extrahiert aus der

[70]Bei den Integrationstests am CERN wurde die Erfahrung gemacht, daß die Entwicklung der Referenz-Software für Linux im Vergleich zu Windows NT deutlich stärker und besser unterstützt wird. Als Konsequenz daraus, wird, bezogen auf künftige Mannheimer Aktivitäten, eine Prioritätsverschiebung von Windows NT zu Linux stattfinden müssen.

Referenz-Softwareumgebung und die nur den TRT-Full-Scan-Algorithmus rechnet. Eine Datei mit ca. 300 simuliertem Events bildete die Basis für die Messungen für die CPU-Ausführung. Alle Zeitmessungen wurden mit hoher Präzision auf unterster Algorithmenebene ausgeführt.

Die Integration des ATLANTIS Systems in die Referenz-Software wird über Schnittstellen des AOS (Abschnitt 5.6.1) bewerkstelligt. Die Hauptarbeit bei der Einbettung des ATLANTIS-Systems in die Referenz-Softwareumgebung wurde von Christian Hinkelbein geleistet, dem Entwickler des AOS. Die Messungen des FPGA-Algorithmus' werden durch Standard-Funktionsaufrufe des AOS sehr gut unterstützt. Die wichtigsten Funktionen u.a. zum Laden der FPGA-Designs, zur Einstellung des Board-Frequenz, zum Beschreiben und Lesen der TRT-Speicher-Module und zur Messung von Algorithmen-Ausführungzeiten, sind bereits fester Bestandteil der sich im Aufbau befindlichen Software. Unter dem AOS können inzwischen mehrere ACBs gleichzeitig angesteuert und kontrolliert werden.

6.5.2 Durchführung und Ergebnis

In diesem Abschnitt werden die Full-Scan-TRT-Messungen auf dem ATLANTIS-System beschrieben und die Ergebnisse dargestellt. Für den Vergleich mit der reinen CPU-Implementierung werden die in [Ses00] angegebenen Ausführungszeiten benutzt. Diese Messungen wurden unter der Referenz-Software Version 1.51 mit einem 300-MHz-PentiumII--System ausgeführt.

Die in dieser Arbeit angegeben Resultate für die Ausführungszeiten auf dem ATLANTIS-System basieren auf Einzelmessungen für den FPGA-Teil und den CPU-Teil. Der Grund dafür ist, daß die für das ATLANTIS-System benutzen Datenformate sich von denen in der Referenz-Software benutzten Datenformate unterscheiden. Die Definition einer gemeinsamen, allgemeinen Schnittstelle wird gegenwärtig diskutiert und umgesetzt. Aufgrund der unterschiedlichen Formate sind die Ausführungszeiten der hybriden FPGA / CPU-Implementierung zusammengesetzt aus Einzelmessungen von Ausführungen auf den getrennten Systemen. Es ist wichtig anzumerken, daß dies auf die Ergebnisse der Messungen keinen Einfluß hat, weil die Algorithmus-Grenze klar definiert sind und deshalb die Algorithmenteile exakt ausgemessen werden können.

Der Transfer der Eventdaten zum ACB überlappt mit der Ausführung des Histogrammierers. Dadurch erklärt sich, daß die effektive Zeit für die Ausführung beider Schritte nicht die Summe der Einzelschritte sondern das Maximum von beiden ist. Die Zeit für das FPGA-Histogrammieren wird fast ausschließlich durch die Transferdauer der Hitdatenübertragung bestimmt[71]. Daneben wird auch der Großteil der lokalen Maximum-Suche bereits während der Übertragung der Hitdaten ausgeführt. Die in der Tabelle angegeben Zeiten für das Histogrammieren und die lokale Maximumsuche sind nicht separat meßbar und beruhen auf Berechnungen.

[71]Die Latenzzeit für das Durchleiten der Hit-Daten durch die FPGAs berechnet sich aus den 4 benötigten Taktzyklen und der Taktfrequenz von 19 Mhz zu 210 ns und ist deswegen, bezogen auf die Gesamtzeit des Algorithmenschrittes vernachlässigbar.

Die Messungen des FPGA-Teils wurden auf einem 200-MHz-Pentium-System (Radisys-System) basierten ATLANTIS-System unter Windows NT ausgeführt. Die DMA-Transferraten auf diesem System sind besser als die auf dem zweiten verfügbaren 400-MHz-PentiumII-Prozessorsystem-basierten ATLANTIS (vgl. Abschnitt 5.1.2). Den Meßaufbau zeigt die Abbildung 6.11. Zusammen mit der Host-CPU werden zwei ACBs parallel betrieben. Ein ACB ist mit jeweils vier TRT-Speicher-Modulen bestückt.

Abbildung 6.11: TRT-Meßaufbau: CompactPCI-System mit zwei parallel betriebenen ACBs

Die Ergebnisse der Messungen sind in der Tabelle 6.4 angegeben. Die angegebenen Zeiten für die Ausführung auf dem FPGA-Teil sind die Mittelwerte basierend auf der Ausführung von mehreren 1000 Eventdurchläufen. Die Vergleichszahlen sind Angaben aus der Dissertation von Matthias Sessler [Ses00][72].

[72] Angesichts der aktuellen, im Massenmarkt erhältlichen 1-GHz-CPUs (z.B. AMD Athlon) wirkt die Verwendung von Vergleichszahlen basierend auf einer PentiumII-300 MHz CPU nicht angemessen. Objektiv gesehen gehörte aber zum Zeitpunkt der Fertigstellung des ACBs im Sommer 1999 die oben genannte CPU bezogen auf das Preis-/Leistungs-Verhältnis zu den High-End-CPUs.

Beschreibung	Platt-form	Zeit [ms]	Kommentar
Nur CPU-Ausführung			
Histogrammierung	CPU	17,5	Gemessen mit PentiumII-300 MHz, aus [Ses00]
Lokale Maximum-Suche	CPU	3,5	,,
Spur-Splitting	CPU	1,5	,,
Finale Selektion und Spur-Fit	CPU	1	,,
Gesamtzeit: Nur CPU-Ausführung		**23,5**	
Hybride FPGA/CPU-Ausführung			
Event-Transfer zum ACB	CPCI	1,9	64 kByte Hit-Daten[73], DMA
Histogrammierung	FPGA	1,85	Überlappung mit Event-Transfer
Lokale Maximum-Suche	FPGA	0,15	Überlappung mit Event-Transfer
Ergebnis-Transfer ins Host-System	CPCI	0,1	
Histogr. + lok. Maximum-Suche	FPGA	2	
Gesamtzeit: Hybride Ausführung	hybrid	**4,5**	

Tabelle 6.4: Meßergebnisse der Full-Scan-TRT-Algorithmusausführung (Barrelhälfte)

Die Beschleunigung des vollständigen Algorithmus' bezogen auf eine TRT-Barrelhälfte beträgt 5.2. Vergleicht man die ersten beiden Schritte miteinander ergibt sich ein Beschleunigungsfaktor von 10.5, was über dem angestrebten Ziel von einem Faktor 10 liegt.

Neben den Messungen mit einem ACB wurde auch die parallele Ausführung auf zwei Boards untersucht (Ergebnisse in Tabelle 6.5). Der gleichzeitige Einsatz von zwei ACBs unter der Verwendung von nicht-blockierenden DMA-Transfers wirkt sich in einer weiteren Reduktion der Ausführungszeit aus. Diese Konstellation erlaubt es, mit zwei ACBs den vollständigen TRT-Barrel schneller zu rechnen, im Vergleich zur sequentiellen Ausführung von den beiden Barrelhälften auf einem Board. Der Beschleunigungsfaktor des gesamten Full-Scan-TRTs bezogen auf den vollständigen Barrel, errechnet sich zu 5.6. Die Ausführung der ersten beiden Schritte wird durch die Ausführung auf FPGAs in diesem Fall sogar um den Faktor 12.5 beschleunigt.

[73]Grundsätzlich besteht eine Hit-Datum aus einem 16-Bit-Wort. Die Übertragung der Hitdaten erfolgt komprimiert - es werden jeweils zwei 16-Bit-Hitdaten in einem 32-Bit-PCI-Datenwort übertragen. Dadurch ist die Größe des Nutzdatenpakets gleich der Größe des Gesamtdatenpakets, nämlich 64 kByte.

Beschreibung	Platt-form	Zeit [ms]	Kommentar
Nur CPU-Ausführung			
Histogrammierung + lok. Max.	CPU	42	Gemessen mit PentiumII-300 MHz, aus [Ses00]
Gesamtzeit: Nur CPU-Ausführung	CPU	**47**	
Hybride FPGA/CPU-Ausführung			
Histogrammierung + lok. Max.	FPGA	4	1 ACB, sequentiell
Histogrammierung + lok. Max.	FPGA	3,35	2 ACBs, parallel
Gesamtzeit: Hybride Ausführung	hybrid	**8,35**	

Tabelle 6.5: Meßergebnisse der Full-Scan-TRT-Algorithmusausführung (Barrel)

6.5.3 Integration von ATLANTIS ins CERN-Testbed

Während zweier Aufenthalte am CERN konnte im September und im November 1999 die erfolgreiche Integration des ATLANTIS-Systems in das LVL2-Pilot-Projekt ATM/Ethernet-Applikation-Testbed (ATB) durchgeführt werden. Dies umfaßte sowohl die Hardware- als auch die Software-Integration. Es wurde gezeigt, daß das ATLANTIS-System zum integrierten Bestandteil des ATBs gemacht werden kann, ohne daß die Referenz-Softwareumgebung modifiziert werden muß. Aus der Sicht der Referenz-Software verhält sich das ATLANTIS-System wie ein herkömmlicher Farmprozessor. Der Zugriff auf die FPGA-Resourcen des ATLANTIS-Systems ist unabhängig von der Verwendung weiterer installierter Hardware-Komponenten, wie zum Beispiel den ATM-Netzwerkkarten von Saclay[74] oder den Standard-Ethernet-Netzwerkkarten, die beide gleichzeitig installiert waren und für den von den ROB kommenden Eventdatentransfer genutzt wurden. Weder die zusätzliche Softwareschicht für Konfiguration und Datentransfer von und zu den FPGAs, noch der Gerätetreiber für den Betrieb der ACBs unter Windows NT hatten einen negativen Einfluß auf die Referenz-Softwareumgebung. Im weiteren konnte während des Integrationstests gezeigt werden, daß die Algorithmenqualität von der reinen CPU-Version und der hybriden FPGA/CPU-Version identisch sind[75]. Zu diesem Zeitpunkt (November 1999) konnten mit dem ATLANTIS-System aufgrund sowohl des Zustands der Algorithmus-Implementierung als auch der CERN-Testbed- und ATLANTIS-Softwareumgebung keine Ausführungszeiten gemessen werden.

[74] Arbeitsgruppe aus Paris/Frankreich.

[75] Die Ergebnisse basieren auf einer in der Größe reduzierten LUT (32-Bit-Daten-Interface), zeigen aber, daß prinzipiell die gleiche Qualität erreicht werden kann.

6.5.4 Verbesserungspotential der aktuellen FPGA-Implementierung

Obwohl mit der dargestellten Implementierung auf ATLANTIS ein bedeutsamer Speedup beim Full-Scan-TRT gegenüber der reinen CPU-Version erzielt werden konnte, kann durch verschiedene Verbesserungen eine weitere Leistungssteigerung erreicht werden. Die vielversprechendsten Ansätze sind nachfolgend aufgelistet:

1. **Lokaler Hitdatenspeicher**: Wie aus den Zahlen deutlich wird, wird die Ausführungzeit im wesentlichen von der Übertragungszeit der Hitdaten zum ACB dominiert. Um dies zu reduzieren ist die Verwendung eines lokalen Hitdatenspeichers denkbar. Bezogen auf die derzeitige Implementierung bedeutet dies, eine Reduktion der zu übertragenden Daten um den Faktor 16. Dadurch wird die Eventbearbeitung nicht länger von der PCI-Performance definiert und weitere Verbesserungen sind möglich (siehe Punkt 3)
2. **Symmetrie**: Die Detektor-Symmetrie im Barrel wird derzeit weder von den CERN-Simulations-Tools noch von der Referenz-Software berücksichtigt. Die Anwendung der Symmetrie reduziert bei der aktuellen FPGA-Implementierung den benötigten Adressraum für die LUT um den Faktor 16 und damit um den gleichen Faktor den Speicherbedarf. Dies wurde bei den durchgeführten Messungen bereits vorausgesetzt.
3. **Design-Frequenz**: Zur Zeit liegt die maximale Designfrequenz für die Full-Scan-TRT bei 19 MHz. Grundsätzlich unterstützt das ATLANTIS-System Designfrequenzen bis 80 MHz. Existierende, einzelne TRT-FPGA-Designs laufen bereits mit 26 MHz. Die bisherigen Erfahrungen lassen die optimistische aber realistische Erwartung auf Designs mit 30 MHz zu.

Die Umsetzung der genannten Verbesserungsmöglichkeiten sind der Gegenstand aktueller Forschungsarbeiten. Der Schwerpunkt liegt dabei bei der Einführung des lokalen Hitdatenspeichers durch die Verwendung von drei ACBs für den gesamten Barrel und unter Einbeziehung des Privat-Bus. Dabei könnte gleichzeitig, ohne größeren Aufwand, die Größe des Spurmusterraums der FPGA-Implementierung an die Größe der CPU-Version (1200 ϕ x 80 p_T) angepaßt werden.

6.5.5 Fazit der Full-Scan-TRT-Implementierung auf ATLANTIS

Interpretiert und legt man die Ergebnisse seit dem ersten Integrationtests im September 1999 zu Grunde, führt dies zu den nachfolgenden fünf wesentlichen Ergebnissen und Schlußfolgerungen:

- Der vollständige TRT-Full-Scan-Algorithmus (Barrel) wurde auf das ATLANTIS-System abgebildet. Die beiden rechenintensivsten Schritte wurden in die FPGAs implementiert.
- Die aktuellen Zahlen für die Ausführungszeit zeigen einen signifikanten Speedup der hybriden FPGA / CPU-basierten Ausführung gegenüber der reinen CPU-Version (Faktor 5.6).
- Zeitkritische und strukturierte Algorithmen wie der Full-Scan-TRT können durch die Verwendung von FPGA-basierter Technologie effizient ausgeführt werden. Dabei wird der Performance-Gewinn durch eine völlig transparente Integration der exotischen FPGA-Technologie erreicht, wie mit dem ATLANTIS-System gezeigt wurde.
- Die für das Jahr 2001 angekündigte FPGA-Technologie läßt den Aufbau noch leistungsfähigerer Koprozessorarchitekturen erwarten. Die Hochrechnungen aus [HKL00] führen dazu, von einem Single-FPGA-System auszugehen, mit dem jeder Farmprozessor ausgestattet wird.
- Als wesentliche Konsequenz, ist die Reduktion der Größe der LVL2-Prozessorfarm um den Faktor 2-5 [ADE00] durch den Einsatz der zukünftigen FPGA-Koprozessorarchitektur zu erwarten.

6.6 Neue Wege der Algorithmen-Implementierung

Die Umsetzungsmöglichkeiten von Triggeralgorithmen auf FPGAs wird stark durch die Komplexität und die Architektur der FPGA-Bausteine bestimmt. Komplexitätszuwächse und Architekturänderungen bei FPGAs sind im Vergleich zu Standard-CPUs in kurzfristigeren Abständen und in einem zum Teil extremeren Ausmaß zu beobachten. Ein bedeutsames Feature in der neuen Virtex-FPGA-Bausteinfamilie wurde Ende 1998 von der Firma Xilinx angekündigt[76]. Als Novum werden bei dieser Familie zum ersten Mal eine größere Anzahl von festverdrahteten internen Speicherblöcken bereitgestellt.

Mit dieser innovativen Architektur sind völlig neue Möglichkeiten bei der Algorithmen-Entwicklung denkbar. Insbesondere eröffnen sich für die Ausführung des Full-Scan-TRT-Algorithmus' neue, effizientere Möglichkeiten. Die Ergebnisse der vorläufigen Studien finden sich in [HKL00]. Dabei handelt es sich überwiegend um theoretische Überlegungen, die durch partielle Simulationen gestützt werden. Da

[76]Seit Mitte 1999 sind die FPGAs der Virtex-Familie für den Massenmarkt verfügbar.

dies nur *einen* möglichen neuen Weg für zukünftige Algorithmenausführungen darstellt, soll auf Details an dieser Stelle nicht eingegangen. Ein wesentliches Ergebnis der Untersuchungen ist, daß die Ausführungszeit des Full-Scan-TRT-Algorithmus' weiter reduziert werden kann [HKL00]. Außerdem werden durch die stark an der CPU-Version orientierten Implementierungsidee die parallele Weiterentwicklung beider Versionen erleichtert und grundsätzlich der Effizienzvergleich vereinfacht.

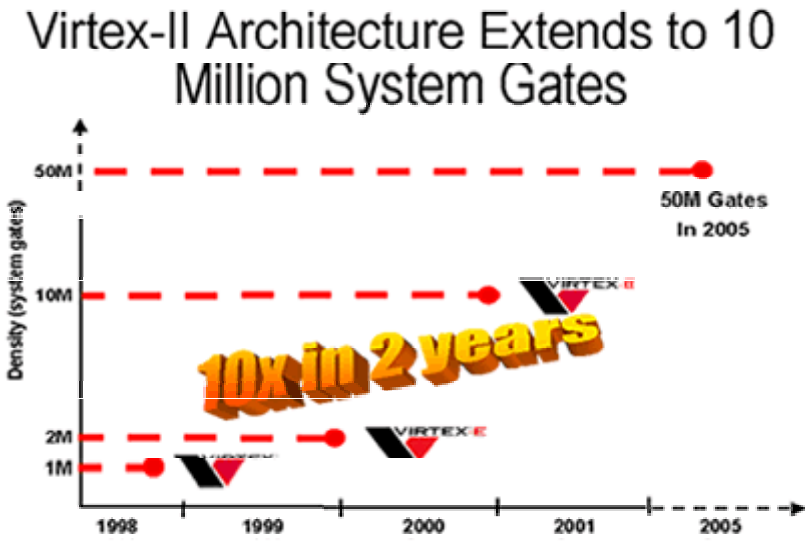


Abbildung 6.12: Road-Map der Xilinx Virtex-FPGA-Familie[77]

Diesen Studien zu Grunde gelegt ist der Einsatz des für das Jahr 2001 angekündigten Virtex-FPGAs XCV3200E [Xil00]. In Abbildung 6.12 ist die von der Firma Xilinx veröffentliche Road-Map bis zum Jahr 2005 dargestellt. Bereits für das Jahr 2000 ist die VirtexII-Familie angekündigt. Diese besitzt eine dreifach höhere Komplexität im Vergleich zur Virtex-E-Familie. Im Jahr 2005 sollen Bausteine auf den Markt kommen, die eine nochmals um einen Faktor 5 erhöhte Komplexität besitzen. Dies soll demonstrieren, daß die Leistungsfähigkeit zukünftiger FPGAs auch in naher Zukunft noch deutlich anwachsen wird und daher die endgültige Konzeption und der Aufbau der finalen Koprozessorarchitektur frühestens zwei Jahre vor Experimentbeginn festgelegt werden bzw. beginnen sollte. Zu diesem Zeitpunkt sollte aber die Grundstruktur der auszuführenden Algorithmen feststehen.

[77] http://www.xilinx.com/products/virtex/ss_vir2.htm.

6.7 Gesamtbewertung und Ausblick

Aus den im diesen Kapitel dargestellten Arbeiten lassen sich zwei wesentliche globale Ergebnisse ableiten:

1. Mit der Ausführung des Full-Scan-TRT-Algorithmus' auf dem ATLANTIS-System konnte zum Ende der Pilot-Projekt-Phase in der Praxis gezeigt werden, daß mit einer hybriden FPGA/CPU-basierten Architektur ein wesentlicher Beitrag bei der Umsetzung der favorisierten Farmarchitektur für den ATLAS LVL2-Trigger geleistet wird. Der Ausblick auf noch leistungsfähigere zukünftige FPGA-Technologien untermauern dieses Ergebnis.
2. Die Verfügbarkeit der für die kommenden Jahre angekündigten FPGA-Architekturen ermöglicht neue und noch effizientere Wege zur Implementierung von Triggeralgorithmen. Das ATLANTIS-System ist bereits auf die Integration zukünftigen FPGA-Technologien vorbereitet. Damit kann das ATLANTIS-System bis zur Entwicklung oder Beschaffung eines endgültigen FPGA-Koprozessors für ATLAS im Jahre 2002/2003 als Prototypensystem genutzt werden.

Berücksichtigt man den wachsenden Markt der FPGA-Koprozessoren, besteht die Möglichkeit, daß innerhalb der nächsten Jahre ein geeignetes System kommerziell erworben werden kann. Die wichtigste Arbeit in der verbleibenden Projektphase bis zur Entscheidung für ein System, ist daher ausführliche Studien zu verschiedenen B-Physik-Triggeralgorithmen auf FPGAs durchzuführen. Damit soll eine breite Basis an Informationen gesammelt werden, um die richtige Wahl für den geeignetsten FPGA-basierten Koprozessor für die Ausführung der ATLAS LVL2-B-Physik-Triggeralgorithmen treffen zu können.

KAPITEL 7

Der universelle Charakter des ATLANTIS-Systems

Das ATLANTIS-System wurde vorrangig zur Ausführung zeitkritischer und I/O-intensiver Aufgaben des ATLAS LVL2-Triggers konzipiert und aufgebaut. Neben den Aktivitäten für ATLAS werden in der Mannheimer FPGA-Gruppe weitere Projekte durchgeführt, denen alle gemeinsam ist, aufbauend auf reprogrammierbarer Hardware den optimalen Lösungsweg zu entwickeln und umzusetzen. Unter dieser Voraussetzung war es für die Konzeption des ATLANTIS-Systems grundlegend, die speziellen Anforderungen dieser Projekte mit zu berücksichtigen (vgl. Kapitel 4). Im diesen Kapitel werden die Projekte in einer Übersicht vorgestellt und die Rolle des ATLANTIS-Systems bei der Ausführung der Applikationen aufgezeigt. Die Projekte im einzelnen sind Studien zu den ATLAS Trigger-Readout-Buffern und Readout-Drivern, ein Astronomie-Anwendung zur Simulation von N-Körper-Systemen, die Lösung der Triggerprobleme bei zwei Experimenten an der GSI/Darmstadt und die Umsetzung einer Volumenvisualisierungsanwendung. In dieser Reihenfolge wird in den nachfolgenden 4 Abschnitten auf die einzelnen Projekte eingegangen. Weiterhin werden in der Mannheimer FPGA-Arbeitgruppe 2D-Bildverarbeitunganwendungen studiert, die bisher auf dem kommerziellen FPGA-Koprozessor microEnable von Silicon Software ausgeführt werden. Es ist geplant, das große Potential von ATLANTIS zukünftig auch für diese Anwendungen einzusetzen.

7.1 Evaluierung FPGA-basierter ATLAS Readout-Systeme

Neben dem Einsatz als hybrider Triggerprozessor innerhalb der LVL2-Farmarchitektur, dient die ATLANTIS Hardware-Plattform auch als Prototypensystem zur Evaluierung der ATLAS Readout-Systeme. Die Funktionsbeschreibung des zweigeteilten Readout-Systems bestehend aus Readout-Drivern (RODs) und Readout-Buffern (ROBs) ist bereits in Kapitel 3 in Zusammenhang mit der

Darstellung der Trigger-Architekur erfolgt (Abschnitt 3.4.3). An dieser Stelle soll in detaillierterer Form auf die Untersuchungen, basierend auf dem ATLANTIS-System eingegangen werden. Die Entwicklung des FPGA-basierten ROB-Complex' werden von Matthias Müller im Rahmen einer Promotion durchgeführt. Unterstützung erfährt er hierbei von Andreas Kugel und dem Autor. Eine wichtige Vorarbeit mit der Implementierung der RobIn-Funktionalität auf das microEnable-System wurde von Rüdiger Rissmann in Rahmen einer vom Autor mitbetreuten Diplomarbeit geleistet.

Beginnend mit den Ergebnissen der RobIn-Realisierung wird die Entwicklung des ROB-Complex' auf ATLANTIS beschrieben. Ein wichtige Rolle spielen dabei die neuentwickelten M-Link-Module, die separat beschrieben werden. Die israelische Gruppe unter Leitung von Dr. Lorne Levinson[78], die für die Entwicklung der Readout-Driver für den Myonendetektor mit verantwortlich zeichnet, ist an der Verwendung der ATLANTIS-Systems als Prototypensystem interessiert. Die Beschreibung und Status dieser sich in der Anfangsphase befindlichen Aktivität folgt im abschließenden Teilabschnitt.

7.1.1 Entwicklung eines RobIn-Prototyps

Die Aufgabe eines RobIn-Moduls besteht darin, die Eventfragmente von einem Readout-Driver-Modul entgegen zu nehmen und zu speichern. Auf Anforderung vom LVL2 oder Event Filter müssen die Fragmente weitergeleitet bzw. gelöscht werden. Die Zugriffe auf das RobIn-Modul werden von einem ROB-Controller gesteuert (Abschnitt 3.4.3).

Der generelle Aufbau des RobIn-Moduls, wie es in der Arbeit von Rüdiger Rissmann entwickelt und untersucht wurde, ist in Abbildung 7.1 gezeigt. Die Basis-Hardware-Plattform bildet der PCI-basierte microEnable FPGA-Koprozessor. Die RobIn-Modul-Struktur ist in vier Teile gegliedert und übernimmt spezifische Aufgaben:

(1) Über die **S-Link-Schnittstelle** (Abschnitt 7.1.3) werden die vom ROD-Modul gesendeten Daten entgegengenommen. Die S-Link-Funktionalität ist dabei in einer Standard-Aufsteckkarte integriert - die Verbindung zu den microEnable Logikresourcen (1 Xilinx-FPGA der 4k-Serie) erfolgt über ein 32-Bit-breites Interface.

(2) Die **Eingangssteuerungs-Logik** nimmt die Fragmente entgegen (S-Link-Empfänger-Design), speichert sie im lokalen Speicher ab und übernimmt die Verwaltung (Buffermanagement).

[78]Weizmann-Institut, Rehovot, Israel.

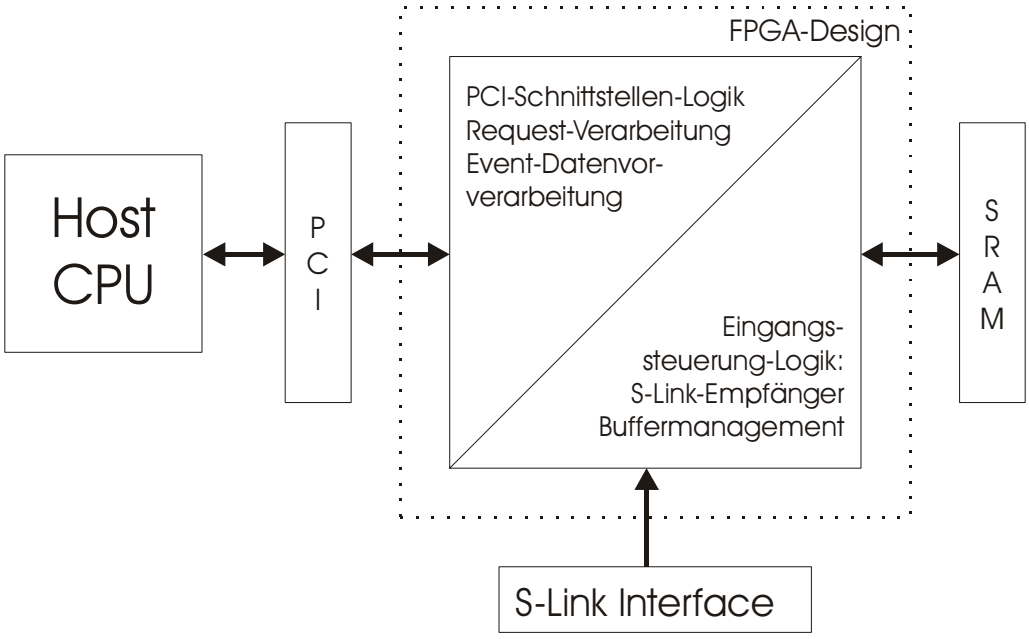


Abbildung 7.1: Das microEnable-basierte RobIn-Modul [Ris98]

(3) Als **Speicher** für die Eventfragmente wird das microEnable onboard-Ram (2 MByte SRAM) benutzt. Auf Anfrage von LVL2/Event Filter werden durch den ROB-Controller (Host-CPU) über den PCI-Bus Eventdaten aus dem Speicher angefordert.

(4) **Die Schnittstellen-Logik zum PCI** (PCI-Interface-Logik) beginnt daraufhin, die angeforderten Daten ebenfalls über den PCI aus dem lokalen Speicher auf den Host-Rechner zu übertragen. Diese werden dort über eine dedizierte Netzwerkschnittstelle an den LVL2/Event Filter weitergeleitet [Ris98].

Als wichtiges Novum, wurde in dieser Arbeit zum ersten Mal innerhalb der LVL2-Kollaboration in der Praxis gezeigt, daß eine effiziente Vorverarbeitung von Eventdaten bereits auf der ROB-Ebene *on-the-fly* möglich ist. Dies wurde durch die Implementierung eines TRT-Datenvorverarbeitung-Algorithmus' [BNS97] demonstriert. Dabei wird das unbehandelte Detektor-Datenformat in ein Adress-Format aktiver Straws (Hits) umgewandelt, wie es vom TRT-Trigger-Algorithmus zur FEX benötigt wird. Durch weitere Eventdatenvorverarbeitungsschritte (Formatkonvertierung und Nullen-Unterdrückung) wird zusätzlich eine signifikante Reduzierung zu übertragender Daten erreicht. Gesamtziel ist dabei, die Netzwerklast auf die propagierte LVL2-Prozesssorfarm drastisch zu reduzieren und stellt somit einen wichtigen Teilaspekt bei der Umsetzung der geplanten LVL2-Architektur dar.

Mit der Implementierung des RobIn-Moduls auf dem microEnable-System wurden folgende wichtige Ergebnisse erzielt: Es wurde gezeigt, daß mit der FPGA-Technologie die geforderte Eingangs-Eventrate pro ROD-Link von 100 MByte/s verarbeitet werden kann. Ebenso wurde demonstriert, daß einschließlich TRT-Eventdatenvorverarbeitung eine Eventdatenrate von 10 kHz[79] zur Host-CPU über den PCI-Bus erreicht werden kann [Ris98].

[79]Der Wert bezieht sich auf Eventfragmentdatenpakete von 1kByte Größe.

7.1.2 Das ATLANTIS ROB-Complex-Szenario

Es ist eine essentielle Forderung, daß die Readout-Buffer gruppiert in einer kompakten Hardware umgesetzt werden müssen (ROB-Complex). Ein einfacher Grund dafür sind die zu erwartenden mechanischen Ausmaße der rund 1700 Einzel-ROBs des gesamten Detektors. Zusätzlich ist dadurch die Möglichkeit gegeben, bereits auf ROB-Ebene, ROB-Modul-übergreifend Vorverarbeitungsschritte ausführen zu können, um die Datenraten zu reduzieren und den Datentransfer insgesamt effizienter zu machen. Dazu gehört u. a. die im vorherigen Abschnitt angesprochene TRT-Datenvorverarbeitung, aber auch das Clustering der SCT/PIXEL-Subdetektoren oder das Aufsummieren von Kalorimeterzelleninhalte.

Basierend auf dem ATLANTIS-System schlägt die Mannheimer Gruppe für die Umsetzung ein kompaktes ROB-Complex-System in einem Standard-14-Slot-CompactPCI-Crate vor [MSH00], in dem theoretisch bis zu 56 RobIn-Module verwaltet werden können (Abbildung 7.2).

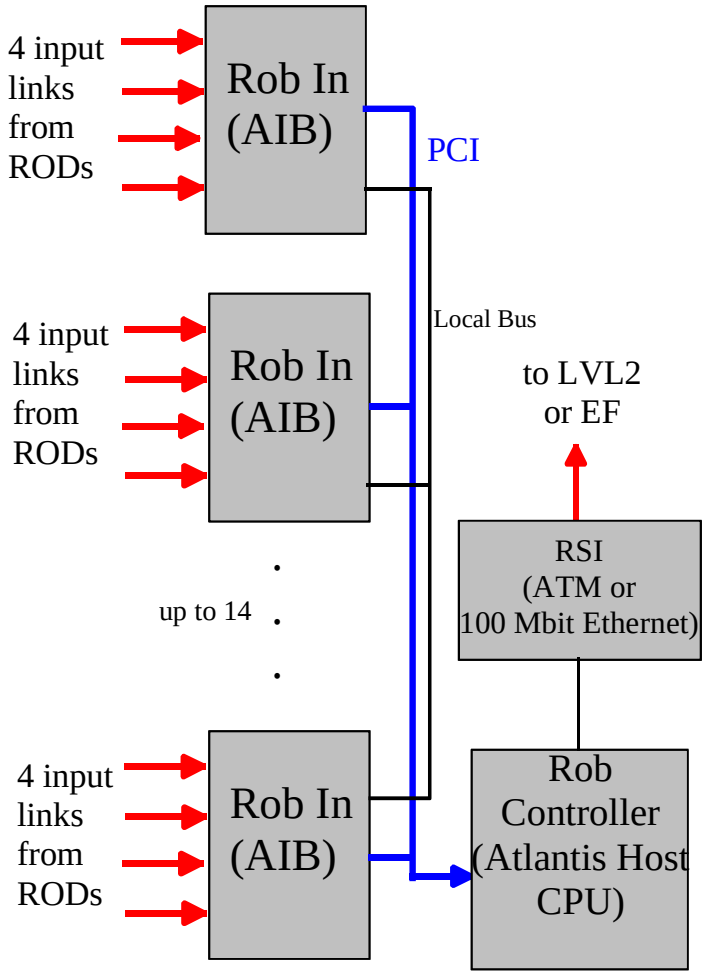


Abbildung 7.2: Der ATLANTIS-basierte ROB-Complex

Für den ATLANTIS-basierten ROB-Complex können in der Praxis bis zu 13 AIBs in einem Crate parallel betrieben werden. Ein Slot ist für den Host-Rechner reserviert. Jedes AIB kontrolliert vier RobIn-Module. Über den ATLANTIS Privat-Bus werden die RobIn-Module der einzelnen AIBs miteinander verbunden und können so sehr leistungsfähig (bis zu 1,4 GByte/s) einen internen Datenaustausch durchführen, ohne den PCI-Bus zu belasten. Als gemeinsamer ROB-Controller für alle RobIns dient der kommerzielle Host-Prozessor. Über diesen werden, wie beim microEnable-RobIn, die Kommunikation mit dem LVL2 und dem Event Filter durchgeführt. Diese funktionale Schnittstelle wird als RSI bezeichnet (ROB-to-Switch-Interface). Als Netzwerk-Technologien werden zur Zeit ATM und Ethernet (100MBit/1GBit) eingesetzt.

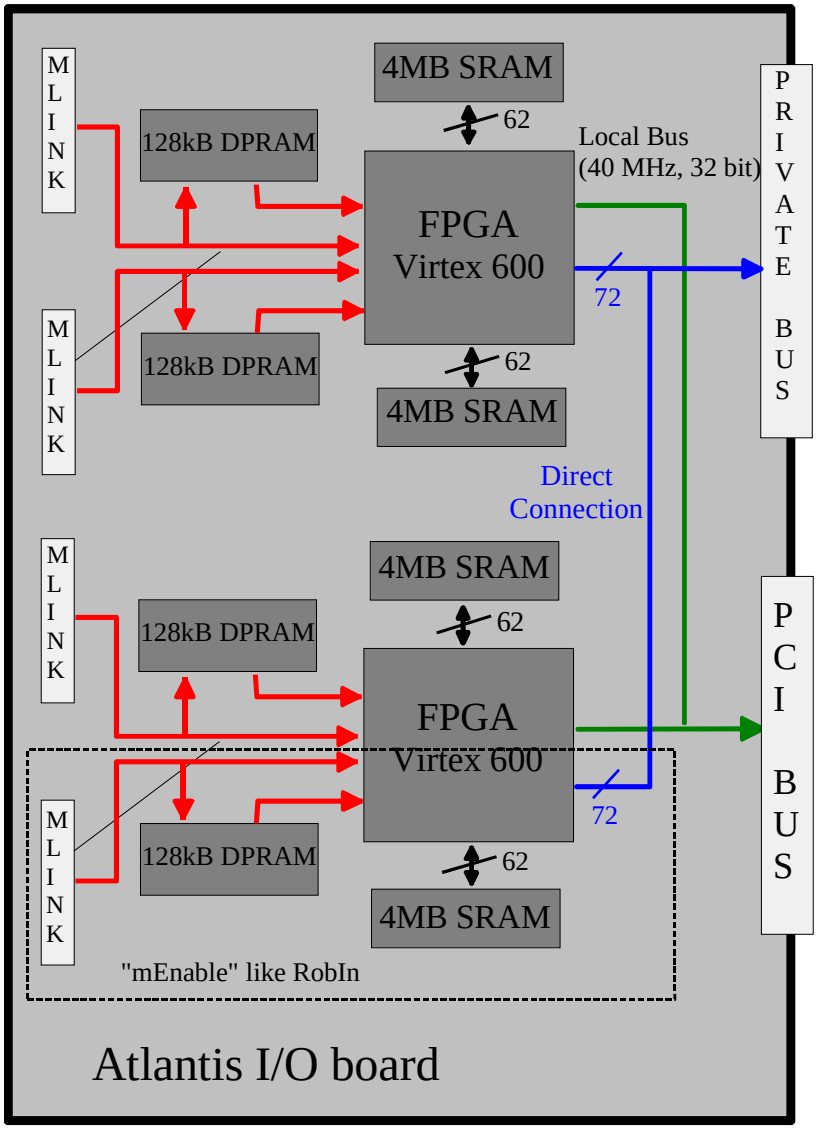


Abbildung 7.3: Das ATLANTIS I/O-Board als Vierfach-RobIn

Abbildung 7.3 zeigt ein AIB mit vier RobIn-Modulen. Diese bestehen grundlegend aus jeweils einer passiven M-Link-Adapter-Karte (Abschnitt 7.1.3). Der M-Link

erlaubt eine Taktfrequenz von 40 MHz und ermöglicht eine Datenrate von 160 MByte/s. Zwei M-Link-Schnittstellen-Module werden jeweils von einem FPGA aus kontrolliert. Zur Sicherstellung der permanten Annahme von Eventfragmentdaten werden die auf dem ATLANTIS I/O-Board festverdrahteten, externen DPR-Bausteine benutzt. Die stellt eine Verbesserung im Vergleich zur RobIn-Lösung auf dem microEnable System dar. Dort mußte eine Teil des SRAMs als DPR simuliert werden, wodurch der Zugriff auf die Hälfte der Systemfrequenz limitiert war.

Pro RobIn-Kanal auf dem AIB werden weiterhin 4-MByte-SRAM zum Speichern der Eventdaten zur Verfügung gestellt. Bei einer durchschnittlichen Eventfragmentgröße von 1 kByte lassen sich unter Berücksichtigung der 100 kHz LVL2-Eventeingangsrate im Mittel parallel 100k Events über einen Zeitraum von 40 ms speichern. Diese Zeitspanne ist ausreichend lang, um ein Speichern der Daten zu garantieren, bis das zugehörige Event entweder verworfen oder zur endgültigen Bearbeitung zum Event Filter weitergeleitet wird [Ver00].

Die Untersuchungen zum ATLANTIS ROB-Complex sind Gegenstand aktueller Forschung und werden von Matthias Müller durchgeführt. Ein Ziel ist es, auf Grund von Papier- und Computermodellen eine sinnvolle Gruppierung der RobIn-Module unterschiedlicher Detektoren zu finden. Dabei sind die Möglichkeiten der Eventdatenvorverarbeitung mit einzubeziehen. Die typischen Eingangseventraten pro RobIn liegen abhängig vom Subdetektor zwischen 8,4 MByte/s und 137,4 MByte/s (Tabelle 7.1) und können, wie die microEnable-Untersuchungen gezeigt haben, leicht verarbeitet werden [Mül00].

Sub-Detektor	**Anzahl RobIns**	**Datenrate pro RobIn** (high luminosity)	**Datenrate pro RobIn** (low luminosity)
TRT	256	77,4 MByte/s	58,7 MByte/s
SCT	92	122,4 MByte/s	21,15 MByte/s
Pixel	84	62,4 MByte/s	8,4 MByte/s
EM Kalorimeter	760	137,4 MByte/s	137,4 MByte/s
Had. Kalorimeter	98	137,4 MByte/s	137,4 MByte/s
Präz. Myonen-Detektor	192	62,4 MByte/s	62,4 MByte/s
Trig. Myonen-Detektor	48	30,9 MByte/s	30,9 MByte/s

Tabelle 7.1: RoBIn-Eingangsdatenraten [Mül00]

In Tabelle 7.2 sind die modellierten Ausgangsdatenraten zum LVL2 und Event Filter aufgelistet [Mül00]. Für alle angegeben Zahlen ist eine LVL2-Eventfrequenz von 75 kHz angenommen. Die mögliche Gruppierung einer bestimmten Anzahl von RobIns in einem Rob-Complex wird grundsätzlich von der Leistungsfähigkeit des PCI-Busses beschränkt. Zur Zeit werden diverse Ansätze untersucht, um den Datentransfer über den PCI-Bus zu verbessern. Dazu gehören u.a. die große Anzahl vieler kleiner Datenpakete zu wenigen größeren Datenpaketen zusammzufassen, die wesentlich effizienter übertragen werden können (vgl. Abschnitt 5.1.2). Die Berücksichtigung von Vorverarbeitungsschritten kann grundsätzlich zu einer Reduktion

der Raten führen. Um den über den PCI-Bus notwendigen Datentransfer zusätzlich zu reduzieren, soll in verstärktem Maße der Privat-Bus mit berücksichtigt und eingesetzt werden.

Sub-Detektor	**Anzahl RobIns**	**Datenrate pro RobIn LVL-2 + EF** (high luminosity)	**Datenrate pro RobIn LVL-2 + EF** (low luminosity)
TRT	256	2,5 MByte/s	9,7 MByte/s
SCT	92	4,8 MByte/s	3,8 MByte/s
Pixel	84	2,9 MByte/s	2,5 MByte/s
EM Kalorimeter	760	6,2 MByte/s	5,6 MByte/s
Had. Kalorimeter	98	6,5 MByte/s	7,3 MByte/s
Präz. Myonen-Detektor	192	1,8 MByte/s	2,3 MByte/s
Trig. Myonen-Detektor	48	1,0 MByte/s	1,5 MByte/s

Tabelle 7.2: Ausgangsdatenraten zum LVL2 und Event Filter [Mül00]

Die Zielsetzung für die weitere Arbeit in Zusammenhang mit dem ATLANTIS ROB-Complex wird sein, verschiedene Vorverarbeitungs-Algorithmen für FPGAs zu untersuchen, um daraus reale Anforderungen ableiten und die endgültige RobIn-Konzentration pro Detektor-Typ angegeben zu können. Dabei ist es ein wichtiger Teilbereich, eine effiziente Nutzung des PCI-Busses zu erreichen. Neben der Erstellung von Papiermodellen sind die Voraussetzungen für Praxistests durch die Existenz des ATLANTIS-Systems und der M-Link-Module gegeben.

7.1.3 Die M-Link-Schnittstelle als Variante des S-Link-Standards

Bereits Mitte der 90er Jahre wurde das S-Link-Konzept als standardisierte Schnittstelle für das ATLAS Readout-System eingeführt, um die Readout-Driver (Detektor-Front-End) mit den Readout-Buffern zu verbinden. Die S-Link-Schnittstelle wurde aus der Motivation heraus eingeführt, frühzeitig einen Standard zu definieren, der die Entwicklung einer allgemein-gültigen, Detektor-unabhängigen ROB-Architektur zuläßt. Andererseits durfte, bezogen auf die LHC-Zeitskala, die Schnittstellendefinition nicht von einem spezifischen, physikalischen Übertragungsweg abhängig gemacht werden, um für zukünftige, leistungsfähigere Technologien offen zu sein [MBi96].

Dies wurde bei der S-Link-Definition so gelöst, daß nur das Schnittstellen-Protokoll (Sender- und Empfänger) definiert ist, die physikalische Implementierung dagegen nicht spezifiziert wird. Für den Benutzer ist der S-Link ein einfacher FIFO-ähnlicher Daten-Link, der zusätzlich nur Fehlerkorrektur, einen Rückkanal zur Datenflußkontrolle und einen Selbsttest beinhaltet [BMB96]. Als physikalische

Verbindung haben sich mittlerweile eine optische Fiber-Channel-Implementierung und eine elektrische Implementierung mit parallelen LVDS[80]-Signalen durchgesetzt.

Da die S-Link-Spezifikation aus mechanischen Gründen keine sehr hohe Baugruppendichte erlaubt und die kommerziellen Implementierungen relativ teuer sind, wurde in Mannheim der sogenannter M-Link[81] entwickelt. Dafür wurden die mechanischen Vorgaben der S-Link-Spezifikation so modifiziert, daß nun 4 statt 3 Link-Module auf ein ATLANTIS I/O-Board montiert werden können. Das Protokoll und die elektrischen Eigenschaften wurden so gewählt, daß jeweils als Gegenstück zu einem Sender- oder Empfänger-M-Link auch die elektrische S-Link-Implementierung gewählt werden kann. Abbildung 7.4 zeigt den S-Link und den M-Link im Vergleich.

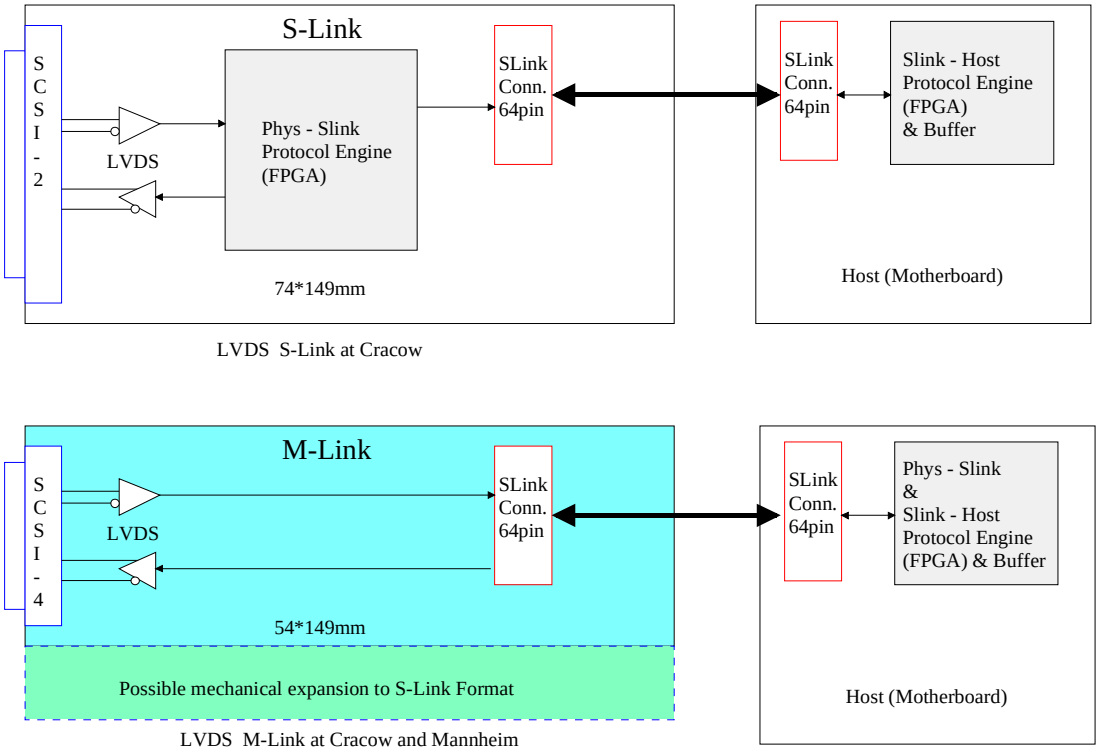


Abbildung 7.4: S-Link- (oben) und M-Link-Schnittstelle (unten) im Vergleich

Zwei wesentliche Dinge unterscheiden den M-Link von der elektrischen S-Link-Implementierung. Die M-Link-Platine ist 20 mm schmaler und das LVDS-Protokoll (Phys. S-Link Protocol Engine) ist nicht als lokale Logik, wie beim S-Link, direkt auf der Platine implementiert. Beim M-Link muß das LVDS-Protokoll in die Logikresourcen der Basis-Platine implementiert werden. Im Fall des AIB wird das Protokoll entsprechend in die FPGAs programmiert. Für die M-Link-Module wurde ein SCSI-Stecker höherer Dichte aber gleicher Pin-Anzahl eingesetzt (68poliger SCSI4-Steckverbinder anstatt des SCSI2-Steckverbinders). Der 64polige S-Link-Connector wird bei beiden Implementierungen benutzt.

[80]LVDS: Low Voltage Differential Signal.

[81]M-Link: Mannheim-Link, die Idee dazu stammt von Andreas Kugel.

Die Datenbusbreite ist 32 Bit. Der M-Link kann mit bis zu 40 MHz getaktet werden. Daraus errechnet sich die maximale Datentransferrate zu 160 Mbyte/s. Die M-Link-Module wurden in gemeinschaftlicher Arbeit von der Mannheimer Gruppe zusammen mit der polnischen Gruppe in Krakow entwickelt, die bereits das elektrische S-Link-Modul entwickelt und aufgebaut hat[82]. Die Abbildung eines M-Link-Moduls findet sich in Anhang B.

7.1.4 ATLANTIS als ROD-Prototypensystem

Bereits zum Zeitpunkt des Aufbaus des ATLANTIS-Systems wurde von einer israelischen Gruppe Interesse für den Einsatz des Systems als ROD-Prototypensystem bekundet. Die von Dr. Lorne Levinson geführte Gruppe zeichnet verantwortlich für die Entwicklung der Readout-Driver-Systeme (RODs) der Myonen-Endkappen-Trigger-Kammern (TGCs). Die ROD-Systeme bilden die Schnittstelle zwischen der Detektor-Frontend-Elektronik und den ROBs. Vom LVL1-Trigger akzeptierte Ereignisse werden vom ROD direkt an den ROB weitergeleitet. Bis zur LVL1-Entscheidung werden die Eventdaten in einer Pipeline-artigen Speicherstruktur im ROD zwischengespeichert. Generell kann die Funktionalität der ATLAS Readout-Driver-Systeme als Untermenge der ROB-Funktionalität gesehen werden. Gemeinsam ist beiden die Anforderung an die Verarbeitung hoher Eingangs- und Ausgangsdatenraten (Größenordnung 100 MByte/s pro Link) bei gleichzeitiger Datenformatkonvertierung. Im ROD-Bereich beschränkt sich die Bearbeitung der Daten auf relativ einfache Umformungen. Angewandt wird die Konvertierung von einer Board/Kanal-Nummer zu geometrischen Koordinaten (r, φ) und einer Kammer-ID mittels einer LUT [Lev00].

Im Mai 2000 wurde in einem ersten ausführlichen Meeting die Anwendung des AIBs für den ROD-Prototypen diskutiert[83]. Eine wesentliche Randbedingung ist, daß standardisierte optische S-Link-Varianten eingesetzt werden sollen. Aus mechanischen Gründen können diese Module nicht direkt auf das AIB aufgesteckt werden. Darum wurde beschlossen, einen Adapter zu entwickeln, mit dem zwei optische S-Links mit zwei AIB I/O-Ports verbunden werden. Die verbleibenden beide AIB I/O-Ports werden über den Adapter kurz geschlossen. Daraus ergibt sich für ein einzelnes AIB eine zusätzliche 45-Bit-Breite Direkt-Verbindung zwischen den beiden FPGAs (Abbildung 7.5).

[82]Die Beschreibung des elektrischen S-Links ist verfügbar über http://www.ifj.edu.pl/~iwanski/e-slink/uman.html.

[83]Das konstituierende Meeting für die geplante Kooperation wurde im Mai 2000 am CERN abgehalten - unter Beteiligung von Dr. Lorne Levinson und dem Autor.

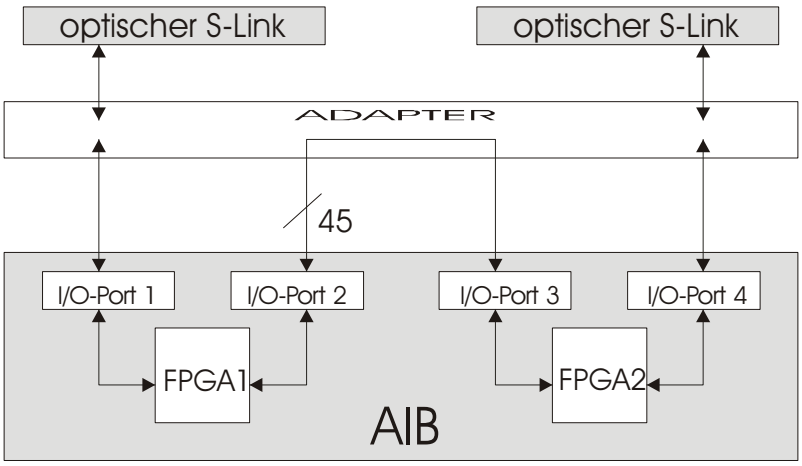


Abbildung 7.5: Integration von zwei optischen S-Link-Modulen in ein AIB mittels Adapter

Die Adapterkarte soll mit Mannheimer Unterstützung in Israel entwickelt werden. Der Zeitplan sieht vor, daß bis Ende 2000 das AIB-basierte ROD-Prototypensystem in Betrieb genommen werden soll.

7.2 Das Astronomie-Projekt AHA-GRAPE

In diesem interdisziplinären Forschungsprojekt aus Astrophysik und Informatik soll der Spezial-Rechner AHA-GRAPE[84] entwickelt und angewendet werden. Dieses System soll dazu eingesetzt werden, um astrophysikalische Teilchensimulationen effizient ausführen zu können. Mit dem AHA-GRAPE werden die bereits erfolgreich eingesetzten GRAPE-Spezialrechner (ASICs, entwickelt an der Universität von Tokyo in Japan, [Mak97]) mit rekonfigurierbaren Systemen bestehend aus FPGAs (ATLANTIS) kombiniert. Der neue hybride Rechner findet insbesondere Anwendung bei der Simulation der Entstehung und Entwicklung von Kugelsternhaufen und dichten Galaxienkernen. Insgesamt sollen das System für eine Reihe von astrophysikalischen Teilprojekte zur Dynamik von Sternsystemen und interstellarer Materie genutzt werden.

7.2.1 Simulationsberechnungen in der Astrophysik mit AHA-GRAPE

Das hier zu entwickelnde Rechnersystem (AHA-GRAPE) wird für eine weitaus größere Anzahl von Anwendungen interessant als die bisherigen japanischen GRAPE-Rechner, die effizient nur für das rein gravitative N-Körperproblem mit Punktmassen einsetzbar waren. Mit dem AHA-GRAPE-System werden diese in einen optimalen Rechner integriert, der insbesondere für N-Körper-Methoden zur

[84] AHA-GRAPE: **A**daptive **H**ydrodynamics **A**rchitecture **GRA**vity **P**ip**E.**

Lösung von gasdynamischen Gleichungen (“smoothed particle hydrodynamics”), aber auch für effiziente Hybridcodes zur Lösung des reinen N-Körper-Problems flexibel einsetzbar sind.

Im klassischen N-Körperproblem teilt sich die Rechenzeit in zwei Anteile O(N) (Bewegung der Teilchen) und $O(N^2)$ (Berechnung aller wechselweisen Gravitationskräfte) auf. Dabei wird mit typischen Teilchenzahlen von $N = 10^4$ bis 10^7 gerechnet. Der erste Teil läuft auf einem normalen Arbeitsplatzrechner ab (dem sogenannten Host-Rechner), und der zweite Teil, wird mit einer hohen Effizienz auf Spezialhardware (GRAPE) implementiert. Der Teil, der auf dem GRAPE-System ausgeführt dominiert den Rechenaufwand um mehrere Größenordnungen, so daß mit dieser Aufteilung ein hoher Speedup bezogen auf die Gesamtberechnung ermöglicht wird. Die Verwendung komplizierterer Programme (u.a. SPH) zur zusätzlichen Berücksichtigung nichtgravitativer Kräfte, verursacht erhebliche Leistungsverluste, da diese nur wenig effizient auf dem Host-Rechner ausgeführt werden können. Durch den SPH kommt ein weiterer Berechnungsterm der Ordnung $O(NN_n)$ hinzu. N_n ist die Zahl der direkten Nachbarteilchen zu jedem Teilchen, die für den SPH-Ansatz berücksichtigt werden müssen und ist typischerweise 50. Die Ausführung des $O(NN_n)$-Terms kann nicht auf dem GRAPE-System durchgeführt werden. Ziel ist es demnach, den $O(NN_n)$ skalierenden Teil der Rechnung auf einem flexiblen, speziell angepaßten Rechner zu implementieren, der in der Rechengeschwindigkeit zwischen GRAPE und dem Host-Rechner liegt. Dies soll durch den Einsatz von FPGA-basierter Hardware erreicht werden. Mit der optimalen Aufteilung der Anwendung auf ein dreigeteiltes hybrides System (FPGA, GRAPE und Host-Rechner) soll die maximal mögliche Performance erzielt werden [KKM99].

7.2.2 Vorstudien auf dem Enable++-System

Um die grundsätzliche Eignung FPGA-basierter Systeme festzustellen, wurde ein wichtiger Teil des SPH-Algorithmus’ bereits 1998 für die Implementierung auf dem Enable++-System untersucht [Kub99]. Dabei wurde die Ausführung der 1. Schrittes des SPH-Algorithmus auf ein Enable++-System abgebildet: Es wird für alle Teilchen i abhängig von den Nachbarn j die Dichte ρ berechnet. Dieser Teil enthält bereits ein Drittel aller Floating-Point-Operationen des gesamten SPH-Algorithmus’. Der Algorithmenteil wurde in der Hardwarebeschreibungssprache VHDL formuliert, simuliert und abschließend synthetisiert (Abbildung 7.6 zeigt das Code-Fragment). Details finden sich in [Kub99].

```
DO i = 1 , Ni
  DO j = 1 , Nj
    (1) rij = ri - rj
    (2) rij = | rij |
    (3) hij = ( hi + hj ) / 2                    //Smoothing Length
    (4) 1 / hij
    (5) W(rij , hij)                             //Glättungsfunktion
    (6) ρi = ρi + mjW (rij , hij) / hij^3        //Dichteberechnung
  END DO
END DO
```

Abbildung 7.6: SPH-Algorithmus Code-Fragment

Im Rahmen der Diplomarbeit von Torsten Kuberka, die vom Autor mit betreut wurde, konnten folgende wichtige Resultate erzielt werden: Für den implementierten Algorithmenteil wurden 15 der 16 FPGAs des Enable++-Systems benutzt, unter massiver Verwendung der Verschaltungsmöglichkeiten der FPGAs untereinander (Abbildung 7.7 vermittelt einen Eindruck der Umsetzung). Für die Implementierung wurde ein reduziertes 28-Bit-Floating-Point-Format benutzt: 1 Vorzeichen-Bit, 7 Bit Exponent und 20 Bit Mantisse. Die implementierte Ausführungs-Pipeline ist 6-stufig, wobei mit jedem Takt ein Ergebnis produziert wird. Verschiedene Operationen (Wurzel, Kehrwert und Division) werden durch die Verwendung von LUTs realisiert[85]. Die Gesamtleistung bei einem maximalen System-Takt von 13 MHz ergibt 208 MFlops[86]. Bei der möglichen Bestückung des Enable++-Systems mit den leistungsfähigeren Xilinx-FPGAs vom Typ XC4036-3 anstelle des Typs XC4013-5 hätte die Ausführung 2fach parallelisiert werden und die Leistung insgesamt auf 1024 MFlops erhöht werden können. Paraller I/O-Transfer ist ebenfalls realisierbar mit 52 MByte/s bzw. 128 MByte/s (bei 2fach parallelisierter Ausführung) auf der Eingangsseite und einigen wenigen MByte/s auf der Ausgangsseite [Kub99].

Die beschleunigte Ausführung von Algorithmen auf FPGAs, die schwerpunktsmäßig auf Floating-Point-Berechnungen aufbauen, wurde in der Vergangenheit als wenig effizient eruiert [LMM98]. Mit diesen neueren Studien [Kub99] konnte als wichtiges Ergebnis gezeigt werden, daß die Komplexität von Multi-FPGA-Prozessor-Systemen mittlerweile einen Grad erreicht, um Floating-Point-Arithmetik wesentlich effizienter und leistungsfähiger auf FPGAs umzusetzen. Insbesondere Addierer und Multiplizierer lassen sich ausreichend effizient aufbauen. Dem gegenüber sind Funktionen wie Division, Wurzel und Kehrwert sinnvoller mit Hilfe von LUTs zu realisieren [Kub99].

[85] In diesem Zusammenhang heißt die Verwendung von LUTs, daß Ergebnisse von Funktionen für einen bestimmten Wertebereich vorberechnet und in einen Speicher (LUT) abgelegt werden. Die Genauigkeit der Berechnungen hängt dabei von der Größe LUT (Speicherdichte) und des gewählten Wertebereichs ab.

[86] MFlops: Mega Floating Point Operations Per Second.

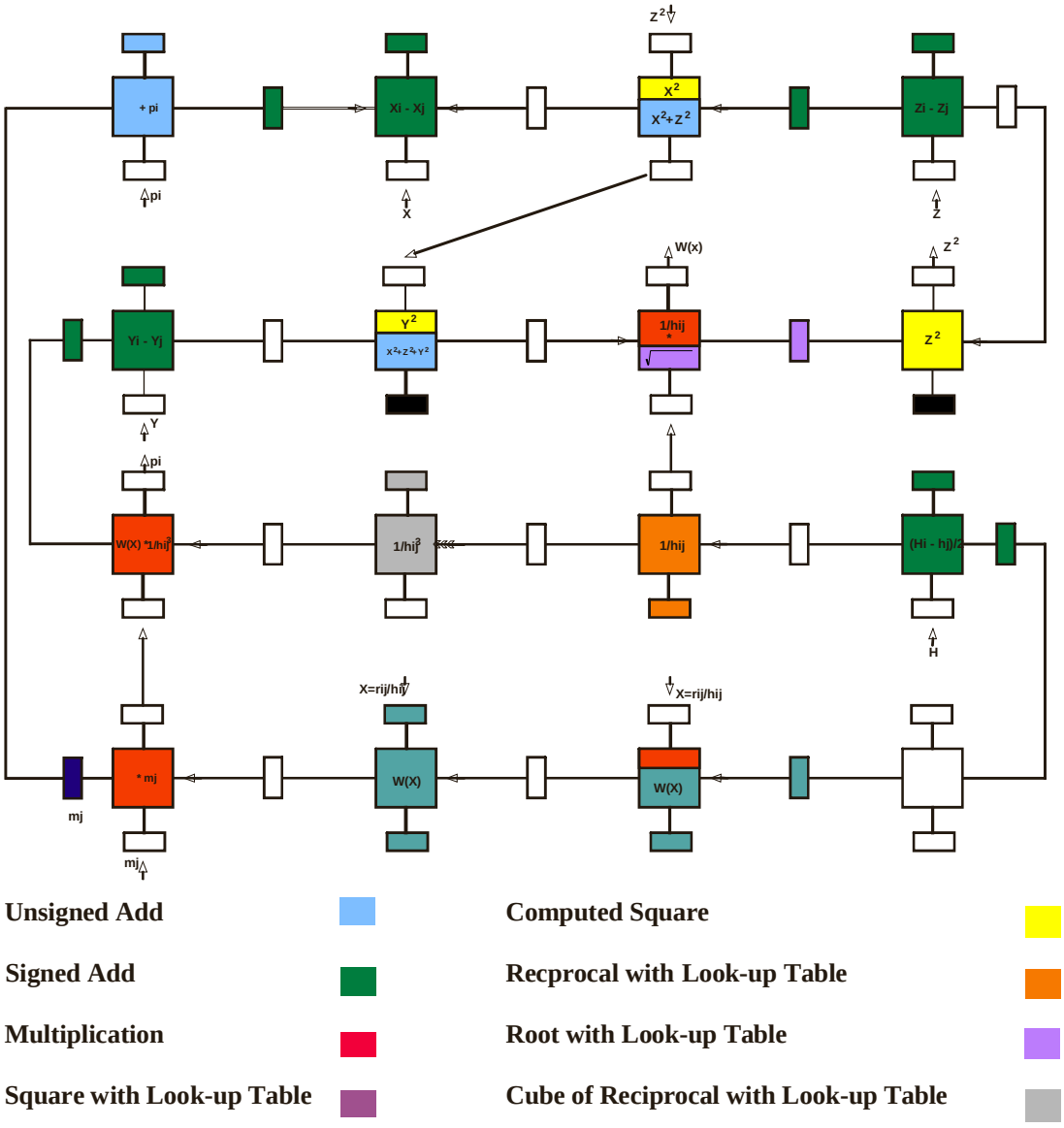


Abbildung 7.7: Plazierung des SPH-Teilalgorithmus und Datenfluß auf dem Enable++-System

Als Gesamtfazit kann bezüglich dieser Arbeiten kann festgehalten werden, daß der Einsatz von FPGAs prinzipiell die Ausführung des SPH-Algorithmus' durch einen signifikanten Performance-Gewinn gegenüber der Ausführung auf einer Standard-CPU unterstützt. Dies hat dazu geführt, daß sich mittlerweile Gerhard Lienhart im Rahmen seiner Promotion seit Februar 2000 mit der Implementierung des vollständigen SPH-Algorithmus' auf das noch leistungsfähigere ATLANTIS-System beschäftigt.

7.2.3 Ausblick auf den Einsatz von ATLANTIS

Abbildung 7.8 zeigt das AHA-GRAPE-System und die geplante Integration des ATLANTIS-Systems. Sowohl das ATLANTIS Computing-Board als auch das GRAPE-Board werden über den PCI-Bus von einer Linux-basierten Host-Workstation aus kontrolliert. Dieses hierarchische System ist für die Ausführung einer Vielzahl von astrophysikalischen N-Körper-Simulationsberechnungen geeignet. Als Formfaktor bietet sich der für das ATLANTIS-System gewählte CompactPCI an. Hier können durch Benutzung eines passiven PCI-zu-CompactPCI-Adapters völlig transparent herkömmliche PCI-Karten - wie das neue, PCI-basierte GRAPE-6-System - integriert werden. Ebenfalls in der Abbildung sind die Leistungsdaten für die Submodule in der Einheit GFlop[87] angegeben. Die Host-Workstation wird mit 0,1 GFlop bewertet, ein ACB mit 1,5 GFlop und das GRAPE-Board mit 50 GFlop.

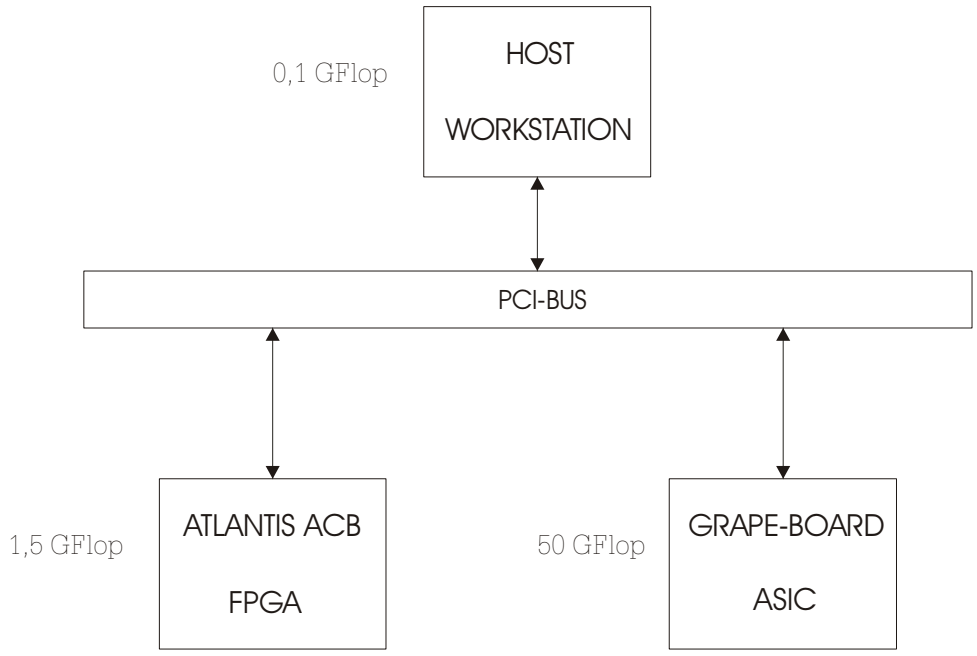


Abbildung 7.8: Skizzierung der geplanten Hardware-Konfiguration des AHA-GRAPE

Das Ziel der Mannheimer Projekt-Arbeit in den Jahren von 2000 bis 2002 ist es, den vollständigen SPH-Algorithmus auf dem ATLANTIS-System zu implementieren. Es soll am Aufbau des AHA-GRAPE-Prototypensystem mitgewirkt werden. Es wird abgeschätzt, daß durch die Integration der FPGA-basierten ATLANTIS-Module die Performance um bis zu einem Faktor 10 gegenüber der herkömmlichen Lösung gesteigert werden kann. Dieser Abschätzung ist eine Teilchenzahl von 10^6 zu Grunde gelegt [KKM99].

[87] GFlop: Giga Floating Point Operations Per Second.

7.3 Umsetzung von Triggersystemen bei GSI-Experimenten

Innerhalb dieses Projekts sollen die Einsatzmöglichkeiten der in Mannheim entwickelten FPGA-Prozessoren als Triggersysteme bei verschiedenen GSI-Experimenten untersucht werden. Zwei besonders erfolgversprechende Einsatzgebiete für das ATLANTIS-System als Triggerprozessor sind das Online-Spurerkennungsaufgabe im Tracking-System des FOPI-Detektors und die Online-Geschwindigkeitsmessung schwerer Ionen mit RICH-Detektoren.

7.3.1 Die Online-Geschwindigkeitsmessung mit RICH-Detektoren

Ring-abbildende Cherenkov-Detektoren (RICH) stellen eine attraktive Alternative zu herkömmlichen Flugzeitdetektorsystemen dar und haben in vielen Experimenten der Hochenergiephysik Anwendung zur Teilchenidentifizierung gefunden. Die neue Methode des Einsatzes von RICH-Detektoren zur Geschwindigkeitsbestimmung für schwere Ionen im SIS-Energiebereich (1-2 AGeV) konnte am Fragmentseparator FRS bereits erfolgreich demonstriert werden. Für geplante Experimente mit radioaktiven Strahlen sind jedoch noch erhebliche Verbesserungen notwendig, um bei gleichzeitig starkem Untergrund leichter Fragmente die interessierenden seltenen Isotope eindeutig identifizieren und ihre Geschwindigkeit mit hoher Genauigkeit (10^{-4}) messen zu können.

In dem bereits existierenden HIRICH (Heavy Ion RICH) Detektor wird das Cherenkovlicht in einem flüssigen C6F14-Radiator von den durchfliegenden Fragmenten bei E = 600 - 1000 A*MeV erzeugt. Wird der Radiator im Durchfluß betrieben, können VUV-Licht absorbierende Strahlenschäden durch hohen Teilchenuntergrund vermieden werden. Das aus dem Radiator ausgetretene Licht wird von einem planaren VUV-Spiegel aus dem Teilchenstrahl heraus und in einen zweidimensional ortsempfindlichen Photonendetektor reflektiert. Die VUV-Photonen lösen in einer mit CsJ beschichteten Kathode freie Photoelektronen aus, die in einer Vieldrahtproportionalkammer (MWPC) mit Kathodenpad-Auslese einzeln nachgewiesen werden.

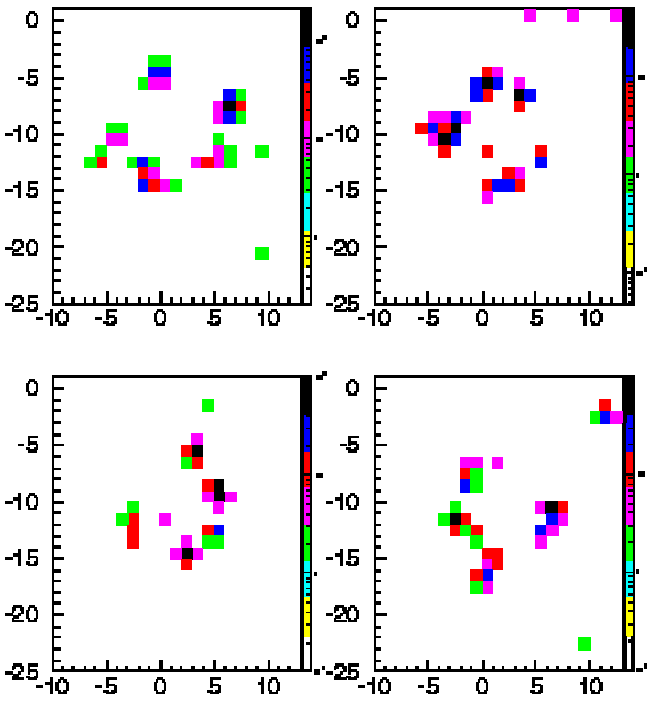


Abbildung 7.9: Einzelne Cherenkovringe für 430 A MEV ^{12}C-Ionen bei Durchtritt durch einen 2mm MgF_2-Radiator

Der Photonendetektor stellt daher zusammen mit der Ausleseelektronik die unter den gegebenen Anforderungen technologisch anspruchsvollste Komponente des RICH-Systems dar. Im Jahr 1997 durchgeführte Messungen mit Fragmenten haben eine Geschwindigkeitsauflösung $<5*10^{-4}$ ergeben [BDH98].

Für den effektiven Einsatz in Experimenten sind jedoch noch weitere Entwicklungsarbeiten notwendig, insbesondere im Bereich der Ausleseelektronik und der schnellen online-Ringerkennung und Ringvermessung. Im Gegensatz zu anderen RICH-Anwendungen sind beim Fragmentnachweis Ringe variabler Größe und unterschiedlicher Anzahl ansprechender Pads zu bearbeiten. Daher ist ein erhöhter dynamischer Bereich der analogen Signalauslese sowie eine völlig andere detektorseitige Datenvorverarbeitung zur schnellen Ringvermessung notwendig.

Die in der Offline-Analyse zur Ringvermessung verwendeten Algorithmen sind zur schnellen Online-Analyse aufgrund der großen Rechenzeiten nicht zu gebrauchen. Aus bereits in Mannheim durchgeführten Projekten und Voruntersuchungen zur Implementierung von Ringerkennungsalgorithmen, kann auf das große Potential FPGA-basierter Systeme, in Bezug auf die Lösung derartiger Probleme geschlossen werden [GBM90].

Zusätzlich zu der Möglichkeit, die eigentliche Ringerkennung und Ringvermessung effizient mit der ATLANTIS-Hardware durchführen zu können, ist die Anbindung des ATLANTIS-Systems an die vorhandene Ausleseelektronik des HIRICH-Detektors in einfacher Weise realisierbar. Dazu muß lediglich eine passive Adapterkarte entwickelt werden - die auf das ATLANTIS I/O-Board aufgesteckt wird - um die beiden Systeme physikalisch miteinander verbinden zu können.

Desweiteren muß das Schnittstellen-Protokoll, wie es durch die Ausleseelektronik vorgegeben wird, in die FPGAs des AIBs implementiert werden.

Ursprünglich war geplant, weitere Vorstudien bereits seit 1998/99 auf der Enable++-Plattform durchzuführen [BDH98], unter Berücksichtigung der möglichen Portierung auf das neue ATLANTIS-System. Aufgrund der beantragten, aber nicht genehmigten Doktorandenstelle zur Bearbeitung dieser Aufgabe, mußte dieses Projekt aber zu Gunsten der Lösung der Triggeraufgabe beim FOPI-Experiment zurückgestellt werden.

7.3.2 Die Triggeraufgabe beim FOPI-Experiment

Ziel des FOPI-Experiments ist es, Informationen über den Zustand von Kernmaterie unter hohen Drücken und Temperaturen zu erhalten. Solche Kernzustände werden in zentralen Kern-Kern-Stößen bei relativistischen Energien erreicht. Um Rückschlüsse auf diese Hochdichtephase machen zu können, werden die zahlreichen Kernbruchstücke, in die die Kerne infolge der explosionsartigen Expansionsphase aufbrechen, untersucht. Zentrales Instrument ist der 4π- oder FOPI-Detektor, in dem alle emittierten Teilchen im gesamten Raumwinkel ausgemessen werden.

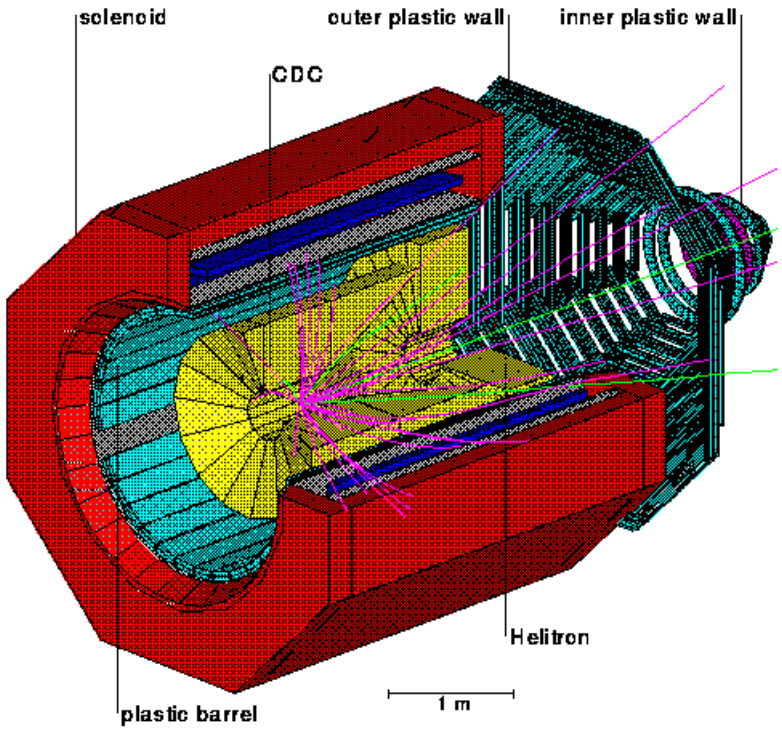


Abbildung 7.10: Der FOPI-Detektor

Insbesondere von den sehr selten produzierten seltsamen Teilchen und speziell von den negativ geladenen Kaonen erwartet man Erkenntnisse über das hadronische Medium und fundamentale Symmetrien der QCD bei hoher Dichte. Diese interessanten Sonden werden allerdings nur mit einer Wahrscheinlichkeit von $2x10^{-4}$/Ereignis produziert, so daß man vor dem Problem steht, eine große Anzahl von Events ($>10^8$) zu archivieren, und/oder eine zuverlässige Triggerentscheidung *online* zu treffen. Letzteres ist insbesondere attraktiv, da 4π-Experimente normalerweise Datenraten-limitiert sind.

Vergleicht man einen der bisher verwendeten Algorithmen zur Spurfindung im Trackingsystem (CDC) des FOPI-Detektors mit dem erfolgreich implementierten Full-Scan-TRT-Algorithmus (1. Schritt) für das ATLAS-Experiment am CERN, ergeben sich viele Gemeinsamkeiten. So stellt die Hough-Transformation bei beiden Algorithmen die zentrale Methode dar. Berücksichtigt man, daß die Randbedingungen für den Full-Scan-TRT-Algorithmus (Eventrate 6-9 kHz, 200 Tracks/Ereignis) vergleichbar sind mit dem FOPI-Algorithmus (Eventrate 10 kHz, 100 Tracks/Ereignis), läßt dies eine erfolgreiche Bearbeitung des FOPI-Triggerproblems auf ATLANTIS-System erwarten. Zusätzlich zum Tracking ist für die FOPI-Applikation allerdings noch das Matching mit der Flugzeitinformation nötig, um die erforderliche Sensitivität (1 aus 10^5 Tracks) zum Erkennen seltsamer Teilchen zu erreichen [BDH98].

Ursprünglich waren die FOPI-Untersuchungen im Zusammenhang mit dem Enable++-System geplant [BDH98]. Mittlerweile steht mit dem ATLANTIS-System der leistungsfähigere FPGA-Prozessorsystem zur Verfügung und soll deshalb als Hardware-Plattform eingesetzt werden. Ein wichtiger Punkt ist die Integration des ATLANTIS-Systems in den geplanten Neuaufbau des Front-End Read-Out/Data-Reduction-Systems. Zur Zeit wird der Ausbau des zentralen Flugzeitdetektors in FOPI und ein Upgrade der Front-End-Elektronik in der letzten Phase ausgeführt. Mit der Verfügbarkeit eines schnellen FPGA-Triggerprozessors soll das bereits bekannte Problem erhöhter Datenraten gelöst werden, die mit dem Upgrade der Elektronik verbunden sind.

Die Hauptzielsetzung im Rahmen der Mitarbeit der Mannheimer Arbeitsgruppe ist, das Triggerproblem innerhalb des FOPI-Experiments im Detail zu studieren sowie mit der Vorerfahrung aus der Implementierung der ATLAS-Triggeralgorithmen eine prototypische Lösung zu entwickeln und auf dem ATLANTIS-System zu implementieren und zu verifizieren.

7.3.3 Status der Arbeiten für den FOPI-Trigger

Die Entwicklung des Triggers für das FOPI-Experiment wird von Mannheim aus seit Mai 1998 von Oliver Brosch im Rahmen einer Promotion durchgeführt. In Zusammenarbeit mit der Heidelberger Arbeitsgruppe[88] wurden zunächst zwei wichtige Dinge geklärt. Erstens wurde entschieden, die Datennahme für das ATLANTIS-

[88]Projektpartner in Heidelberg ist die Gruppe von Prof. Norbert Herrmann u.a. mit Marc Stockmeier

System über den Komperatorausgang der FADCs[89] des Readout-Systems zu realisieren. Dazu müssen die 960 bzw. 480 Auslesekanäle über das ATLANTIS I/O-Board ins System geführt werden[90]. Das notwendige Sampling der Kanäle zur Erzeugung des Eventbildes kann über aktive Adapterkarten erreicht werden, die über das ATLANTIS I/O-Board ins System gelangen. Pro I/O-Board können 180 1-Bit-Datenkanäle verarbeitet werden. Zweitens wurde das FOPI-Triggerproblem im Detail spezifiziert und die Beschreibung des Hardware-orientierten Algorithmus' in C++ begonnen, um vereinfacht Effizienzuntersuchungen durchführen zu können. Die Softwarearbeiten am Triggeralgorithmus werden durch ein Detektor-Visualisierungsprogramm ergänzt, mit dem Events graphisch dargestellt werden können.

Für die Kopplung der aus der Hough-Transformation erzielten Tracking-Information mit der essentiel wichtigen Flugzeit-Information zur Teilchenidentifikation werden zur Zeit Studien durchgeführt. Neben der Erweiterung des Algorithmus durch Einbeziehung der Flugzeitdaten wurde bereits ein geeignetes Verbindungschema zwischen Flugzeit-Detektor und dem ATLANTIS-System erarbeitet. Die Definition der physikalischen Schnittstelle steht noch aus.

Die bisherigen Arbeiten haben zu folgenden wesentlichen Ergebnissen geführt: Der FOPI Trigger-Algorithmus wurde in parametrisierter Form in Software beschrieben und durch Variation der Parameter in Bezug auf Geschwindigkeit (Ausführungszeit), Qualität (Triggereffizienz) und Resourcenverbrauch (erforderliche Anzahl von ACBs) untersucht [Bro00]. Die Grundstruktur entspricht dabei dem in Abschnitt 6.3 beschriebenen 1. Schritt des Full-Scan-TRT-Algorihtmus'. Vorläufige Ergebnisse sind in Tabelle 7.2 zusammengefaßt.

Anzahl Spurmuster (LUT)	Anzahl Hits	Anzahl ACBs	Effizienz	*Fake*[91]-Spuren
922.000	1.500	31	93%	32%
100.000	3.000	7	56%	30%
57.000	1.500	2	42%	16%

Tabelle 7.3: Vorläufige Ergebnisse des FOPI-Triggers [Bro00]

Für die in der Tabelle angegeben Werte werden folgende Grundannahmen gemacht. Mit jedem ACB lassen sich in einem Takt 712 (4 x 178) Spurmuster vergleichen. Zur Speicherung der LUT können die TRT-Speicher-Module benutzt werden (Abschnitt 6.4.3). Die Systemtaktfrequenz wird mit 50 MHz angenommen, was durchaus realistisch ist. Jedes Event besteht im Mittel aus 3000 bzw. 1500 Hits[92]. Der Schwellwert ist

[89]FADC: Flash Analog Digital Converter.

[90]Der FOPI-Tracking-Detektor (CDC) besteht aus 16 Sektoren, wobei jeder Sektor 60 Zähldrähte beinhaltet. Daraus ergibt die Zahl für die Auslesekanäle zu 960, bzw. zu 480, wenn nur jeder zweite Zähldraht ausgelesen wird.

[91]Mit *Fake*-Spuren werden die im Eventbild als Spur interpretierten Objekte bezeichnet, die zwar die Kriterien eines Spurmusters erfüllen, tatsächlich aber nur durch Überlagerungen von anderen Teilchenspuren zustande kommen oder auf Rauschen basieren.

[92]Die Hit-Anzahl ist angegeben in Abhängigkeit von der Anzahl ausgelesener Zähldrähte (960 bzw.

auf 14 eingestellt[93]. Generelle Randbedingung ist, daß die Ausführungszeit maximal 100 µs betragen darf [Bro00].

Folgende Schlußfolgerungen können gezogen werden: Unter der Annahme, ein ATLANTIS-System in der Größenordnung von 2-8 ACBs für den FOPI-Trigger einzusetzen, liegt die Effizienz für gefundene Spuren zur Zeit bei maximal 50% - 60%. Um die Effizienz auf die geforderten 90% zu erhöhen und gleichzeitig die Zahl fälschlicherweise gefundener Spuren (*Fake*-Spuren) zu reduzieren, wurden diverse Mechanismen evaluiert. Hierzu wurden bereits verschiedene Filteroperationen untersucht, beziehungsweise ist es geplant, spezifische p_T-Spektren gesuchter Teilchen mit einzubeziehen. Ein weiteres wichtiges Mittel um die Qualität zu beeinflussen, ist die Variation des Schwellwerts. Die Studien hierzu sind Teil aktueller Untersuchungen [Bro00].

Als Ausblick bleibt zu sagen, daß in naher Zukunft zwei Hauptziele erreicht werden müssen. Es müssen die vorhandenen Mechanismen getestet und neue Methoden gefunden und entwickelt werden, um die Qualität des Triggers zu erhöhen. Daneben ist weitere Arbeit zu leisten, damit die physikalische Integration des ATLANTIS-Systems in das neue Readout-System bzw. die Kopplung an die neuen Flugzeitdetektoren durchgeführt werden kann.

7.4 Hardware-beschleunigte Volumen-Visualisierung

In diesem Abschnitt wird ein Mannheimer Projekt vorgestellt, in dem, basierend auf der Hardware-Plattform des ATLANTIS-Systems, ein algorithmisch-optimiertes Volume-Rendering entwickelt wird. Diese Arbeit wird von Bernd Vettermann im Rahmen einer Promotion durchgeführt. Die Implementierung benutzt ein ATLANTIS Computing-Board, bestückt mit einem speziell dafür entwickelten Sub-Modul (VGE-Modul) mit Zusatz-Logik (Abschnitt 7.4.3).

7.4.1 Einführung und Überblick

Die Volumen-Visualisierungs-Methode ist eine sehr leistungsfähige Anwendung zur Darstellung von 3D-Bilddaten aus verschiedensten Quellen. Beispiele dafür sind die aus der Medizin bekannten CT- und NMR-Aufnahmen oder auch die zunehmenden Anwendungsgebiete in der Unterhaltungsindustrie. Eine Software-basierte Volumen-Visualisierung steht mittlerweile für Standard-PCs zur Verfügung. Die sinnvolle und effiziente Ausführung in Software - auch auf Highend-PCs - wird dabei durch die Größe der Datensätze eingeschränkt. Um Bilddatensätze in der

480).

[93]Mit der Wahl des Schwellwert wird angegeben, wie viele der Hits in einem Spurmuster vorhanden sein müssen, damit dieses als Spur erkannt wird.

Größenordnung 256^3 Raumpunkte in Echtzeit darzustellen, ist der Einsatz einer dedizierten Hardware unumgänglich.

In der Vergangenheit wurden bereits verschiedene Hardware-Systeme zur Beschleunigung einer Volumen-Visualisierung vorgeschlagen. Davon wurden aber nur wenige Systeme in die Realität umgesetzt. Darunter sind das Mannheimer Virim System [GPR94] und das VolumePro-System von Mitsubishi [Mit99]. Das neue Mannheimer Projekt setzt bezüglich einer Volumen-Visualisierung in Hardware verglichen mit vorhandenen Systeme neue Maßstäbe. Durch die Abbildung eines vorab optimierten Algorithmus' in Hardware, können mehr als ein Faktor 10 der Logikresourcen gegenüber herkömmlichen Lösungen eingespart werden. Gleichzeitig kann die Visualisierung-Rate um mehr als Faktor 10 gesteigert werden.

7.4.2 Der VGE-Algorithmus

Im Vergleich zur klassischen Hardware-Implementierung beinhaltet die VGE-Methode zwei wesentliche Optimierungen: Space-Leaping und Early-Ray-Termination. Space-Leaping bedeutet, daß leere Teil-Datenvolumina erkannt und nicht weiter berücksichtigt werden. Dazu wird eine Distanz-Transformation angewandt, mit deren Hilfe die Abstände zwischen nicht-leeren Voxeln vorberechnet werden. Mit der zweiten Optimierung Early-Ray-Termination ist das frühzeitige beenden von Lichstrahlberechnungen auf weitere Sample-Punkte gemeint, wenn die Intensität einen (vom Benutzer einstellbaren) Schwellwert unterschreitet. Weitere Details finden sich in [VHM99].

Der VGE-Algorithmus basiert grundsätzlich auf einer klassischen Volumen-Visualisierungs-Pipeline. Der vollständige Datensatz wird vor der Bearbeitung in Teilvolumina (sub-cubes) zerlegt, die separat bearbeitet und am Ende wieder zusammengesetzt werden. Die klassische Bearbeitung läuft in drei Phasen ab (Abbildung 7.11).

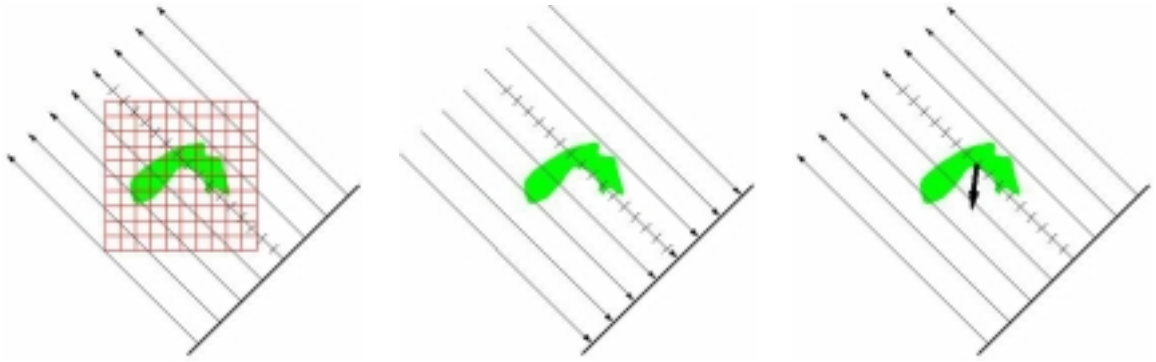


Abbildung 7.11: Die drei Phasen des Volume-Renderings (Sampling, Shading und Compositing)

In der ersten Phase wird vom Beobachter aus durch jeden Pixel des resultierenden Bildes (Projektionsebene) ein Lichtstrahl zum simulierenden Datenvolumen gesendet. Entlang der Lichtstrahlen werden in gleichmäßigen Abständen Sample-Punkte erzeugt (Sampling). Genauer gesagt, die Position der Sample-Punkte im Datenvolumen wird berechnet und eine trilineare Interpolation benachbarter Raumpunkte (Voxel) durchgeführt. In einer zweiten Phase (Shading) wird für jeden Sample-Punkt die Reflexion bestimmt. Hierbei wird unter anderem auch die Absorption des Lichts beim Durchgang durch das Volumen berücksichtigt. Abschließend werden in der letzten Phase (Compositing) die Reflexionen der Sample-Punkte entlang eines Lichtstrahls in Richtung des Beobachters aufsummiert. Das Ergebnis entspricht dem Helligkeitswert des Pixels, durch den der Lichttrahl lief.

In Abbildung 7.12 ist der gesamte Ablauf der VGE-Algorithmus dargestellt. Die Hauptänderung, bezogen auf die klassische Pipeline-Struktur betrifft die Einführung der Ray-Queue. Es handelt sich hierbei um ein Schieberegister, in dem in jeder Stufe die Adresse zu den Parametern unterschiedlicher Lichstrahlen abgelegt sind. Für jeden neu zu berechnenden Sample-Punkt eines Lichstrahls, wird der Lichstrahl in Abhängigkeit vom Abstand zum Betrachter in die Ray-Queue einsortiert. Durch die Ray-Queue wird erreicht, daß die Daten speicherkoherent ausgelesen und in jeder Stufe der Pipeline zur gleichen Zeit unterschiedliche Lichtstrahlen bearbeitet werden. Da Datenabhängigkeiten nur entlang eines Lichtstrahls bestehen, tritt der Fall, daß auf Daten gewartet wird (Datenhazard) und somit die Pipeline anhält oder neu geladen werden muß, praktisch nie auf. Zusätzlich werden die Lichtstrahlen, wenn die Intensität unter den Schwellwert fällt, automatisch gelöscht. Mit der neuen Methode wird die Effizienz des Gesamtsystems im Vergleich zur klassischen Methode um 90% gesteigert [VHM99].

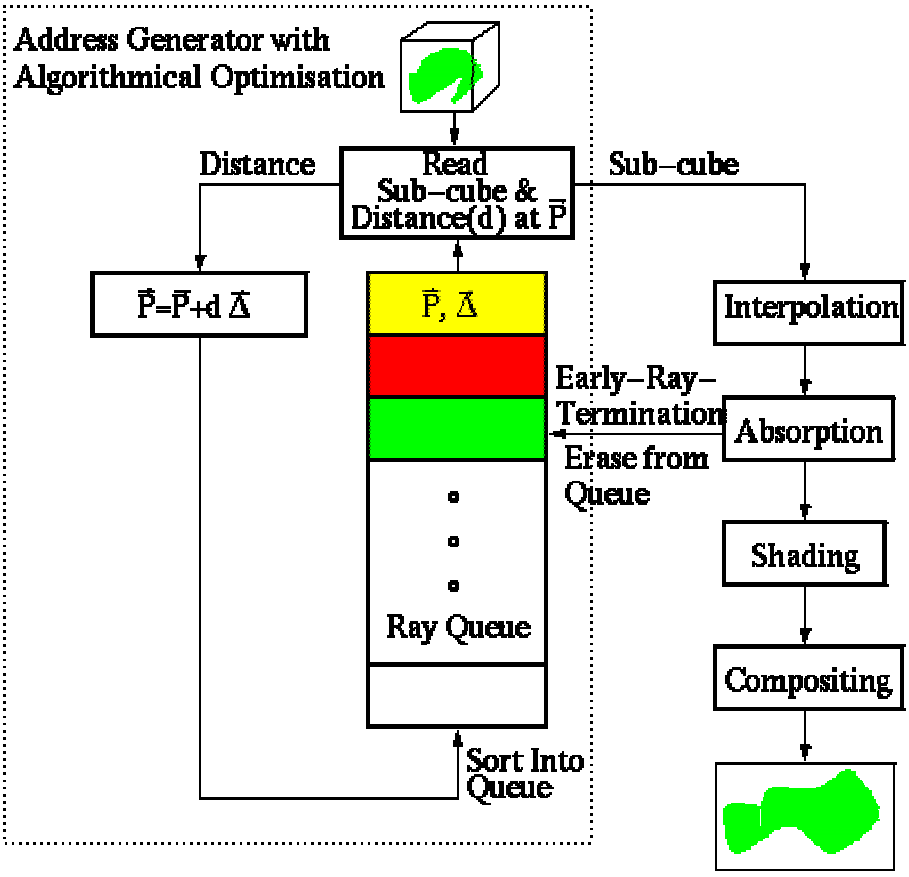


Abbildung 7.12: Volumen-Visualisierungs-Pipeline nach der VGE-Methode

7.4.3 Implementierung und Ergebnis

Für die Implementierung des VGE-Algorithmus werden die Hardware-Resourcen eines ACBs benutzt. Zusätzlich wurde ein Sub-Board (VGE-Modul) entwickelt, daß hauptsächlich dazu dient, den Bilddatensatz zu speichern. Dazu befinden sich auf dem VGE-Modul 128-MByte-SDRAM. Es ist aufgeteilt in acht Unterblöcke, für einen parallelen Zugriff zu den acht nächsten Nachbarn während der Interpolations-Berechnung eines Sample-Punktes. Jeder Block besteht aus zwei 64-MBit-SDRAM-Bausteinen des Typs KM416S8030 von Samsung [Sam99d]. Das gesamte Speichervolumen reicht aus, um einen 512^3-Voxel großen Datensatz speichern zu können, wobei jedes Voxel 16-Bit-breit kodiert werden kann. Neben dem SDRAM zur Speicherung des Datensatzes, sind zusätzlich LUTs für die Klassifikation, das Shading und die Frame-Buffer vorhanden.

Abbildung 7.13 zeigt das vollständige Blockschaltbild des Systems. F1 bis F5 bezeichnet die FPGAs[94]. Die Aufteilung der einzelnen Einheiten auf die FPGAs wird aus dem Blockschaltbild ersichtlich. Für die Ausführung der Distanz-Transformation muß das FPGA F3 jeweils rekonfiguriert werden. Die Rekonfiguration benötigt 34

[94]Vier der FPGAs befinden sich auf dem ACB. Der fünfte, ebenfalls ein Lucent FPGA des Typs Or3T125, wurde mit auf das VGE-Modul implementiert.

ms, während die Distanz-Transformation für den gesamten 512^3 großen Datensatz in 4 s ausgeführt wird. Hierbei durchläuft ein nicht-linearer Filter zweimal das gesamte Daten-Volumen. Auf weitere Einzelheiten, insbesondere der detaillierten Arbeitsweise der Hardwarekomponenten soll hier nicht eingegangen werden. Eine ausführliche Darstellung findet sich in [VHM99]. Die Entwicklung des VGE-Moduls wurde von Bernd Vettermann durchgeführt und vom Autor unterstützt.

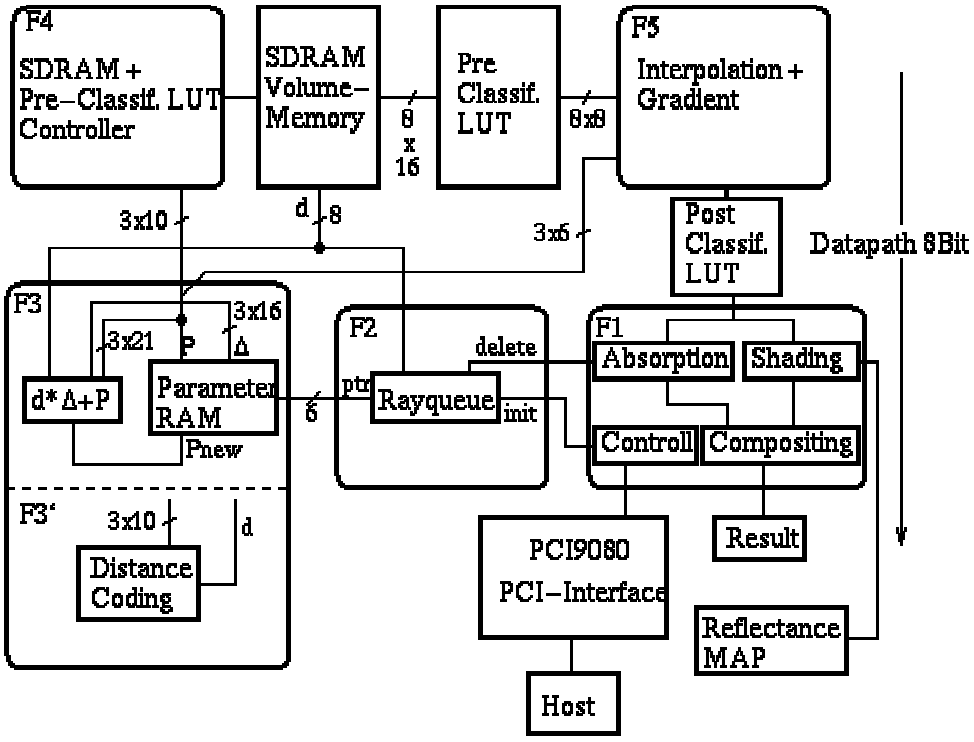


Abbildung 7.13: Blockschaltbild des VGE-Algorithmus', abgebildet auf ein ACB und das VGE-Modul

Die vorläufigen Ergebnisse diese Projekts lassen sich folgendermaßen zusammenfassen: Das gesamte System wurde bis Ende 1999 aufgebaut und getestet. Prinzipiell kann ein 1024^3 großer Datenwürfel mit einer Frequenz von 4 Hz dargestellt werden[95]. Die Ausführungsgeschwindigkeit ist zur Zeit bei einer maximal möglichen Systemfrequenz von 25 MHz eingeschränkt. Für die detaillierte Simulation wurde ein 256x256x128-Voxel-großer Datensatz benutzt. Im Mittel konnte eine Effizienzsteigerung von 90% bis 97% im Vergleich zu herkömmlichen Lösungen erreicht werden. Die erzielte Reduktion der zu berechnenden Musterpunkte beträgt 10% bis 15% bei Bilddaten mit wenigen, lichtundurchlässigen Objekten bzw. 25% bis 40% bei semi-transparenten Bilddaten. Die genannten Ergebnisse führen zu einer Visualisierungsrate von 20 Hz bei semi-transparenten Datensätzen und 138 Hz bei undurchsichtigen Objekten und paralleler Projektion. Die perspektivische Ansicht reduziert die Visualisierungsgeschwindigkeit um einen Faktor 2.

Insgesamt hat das Projekt gezeigt, daß die FPGA-basierte Hardware-Implementierung eines vorab optimierten Algorithmus' effizient umgesetzt werden

[95]Hierbei sind 100 MHz Systemfrequenz zu Grunde gelegt.

kann. Daneben stellt dieser Lösungsweg eine kostengünstigere und zudem leistungsfähigere und flexiblere Alternative im Vergleich zu dem kommerziellen Volume-Pro-Chip von Mitsubishi dar ([VHM99],[Vet00]). Unter Berücksichtigung der in Zukunft verfügbaren FPGA-Technologien wird die Abbildung des gesamten Algorithmus' in ein einzelnes FPGA möglich werden.

7.5 Zusammenfassung

In diesem Kapitel wurden vier Anwendungen vorgestellt, für deren Ausführung eine FPGA-basierte Plattform favorisiert wird. Die spezifischen Anforderungen dieser Applikationen sind mit in die Konzeption des ATLANTIS-Systems eingeflossen (Kapitel 4). Mit den geplanten bzw. teilweise bereits durchgeführten Implementierungen wurde die Eignung und die Funktionalität von ATLANTIS als universelles System in der Praxis nachgewiesen.

Bei den prototypischen Studien zu den ATLAS Readout-Systemen steht der Einsatz der AIBs im Vordergrund. Die AIBs zeichnen sich dabei durch die Verarbeitung hoher Datentransferraten aus, was durch die Skalierbarkeit der I/O-Einheiten zusätzlich unterstützt wird. Die Rolle der Kontroller-Einheit übernimmt dabei die Host-CPU. Die Kopplung an die Außenwelt wird über die am CERN weit verbreitete, standardisierte S-Link-Schnittstelle bereitgestellt. Eine spezielle mechanische Version dieses Standards wurde für das AIB entwickelt, um eine höhere Baugruppendichte zu erreichen.

Studien aus der Vergangenheit haben gezeigt, daß die Ausführung von Floating-Point-Arithmetik auf FPGAs wenig effizient ist, wegen des vergleichsweise hohen Resourcenbedarfs und der erzielten niedrigen Design-Frequenzen. Neue Untersuchungen auf dem Enable++-System haben gezeigt, daß Multi-FPGA-Systeme mittlerweile eine Komplexität erreichen, die umgerechnet zu einer Leistungsfähigkeit in der Größenordung von einigen GFLOPs führt. Auf dieser Basis soll ATLANTIS als Teilsystem in einem kombinierten System aus ASICs, FPGAs und Standard-CPUs die Simulation von N-Körper-Systemen mit mehr als 10^5 Partikeln unterstützen.

Für den FOPI-Trigger werden alle drei FPGA-basierten Komponenten des ATLANTIS-Systems zum Einsatz kommen. Über die AIBs gelangen die Detektordaten ins System und erfahren dabei eine Vorverarbeitung auf spezifischen I/O-Modulen. Die Weiterleitung der Daten erfolgt über den Privat-Bus zu den ACBs. Dort wird die eigentliche Triggeranwendung ausgeführt unter Verwendung von aufgesteckten Speicher-Modulen. Die Ergebnisdaten (Triggerentscheidung) werden wiederum über den Privat-Bus und ein AIB an das Datenaufnahme-System zurückgeleitet.

Für die Ausführung von Volumen-Visualisierungs-Algorithmen, zur Darstellung hochaufgelöster 3D-Bilder mit einer akzeptablen Rate, sind Hardware-basierte Systeme feste Voraussetzung. Der VGE-Algorithmus zeichnet sich gegenüber anderen Hardware-basierten Algorithmen durch eine optimierte Struktur aus, die zu

einer Einsparung von Logikresourcen führt und insgesamt eine effizientere Ausführung zur Folge hat. Dieser Algorithmus wurde auf einem ACB implementiert. Für die Speicherung der Bilddaten und zur Ausführung weiterer Algorithmenteile wurde ein auf das ACB aufsteckbares Submodul entwickelt und bereits eingesetzt.

KAPITEL 8

Ergebnis und Ausblick

In der vorliegenden Arbeit wurde eine hybride Rechnerarchitektur, bestehend aus FPGAs und einer Standard-CPU entworfen und realisiert. Das Haupteinsatzgebiet für das Rechnersystem ATLANTIS ist das LVL2-Triggersystem für das ATLAS-Experiment am CERN. Hierbei sollen sowohl zeitkritische Triggeralgorithmen als auch I/O-intensive Aufgaben im Bereich der Readout-Systeme für die beschleunigte und effiziente Ausführung auf dem ATLANTIS-System untersucht werden.

In der Vergangenheit konnte der vorteilhafte Einsatz der FPGA-Technologie für Teilaufgaben im Triggerbereich mehrfach demonstriert werden. Andererseits haben auch Standard-CPUs gezeigt, daß sie im Vergleich zu Speziallösungen in vielen Bereichen des Triggersystems sinnvoll und ausreichend effizient eingesetzt werden können. Für eine bestimmte Klasse von Problemen hat sich gezeigt, daß der kombinierte Einsatz von Spezialelektronik (FPGAs) und Standard-CPUs den geeigneten Lösungsweg darstellen würde. Aus dieser Motivation heraus wurde das ATLANTIS-System aufgebaut, in dem FPGAs und eine Standard-CPU über das kommerzielle Bussystem CompactPCI eng gekoppelt sind. Der Einsatz einer kommerziellen Host-CPU bietet noch einen weiteren Vorteil. Aus der Sicht eines Anwenders kann das ATLANTIS als herkömmlicher PC mit integrierten FPGA-Prozesssor gesehen werden. Es können herkömmliche Software-Tools, insbesondere unter den weit verbreiteten Standardbetriebssystemen Linux und Windows NT auf dem ATLANTIS System ausgeführt werden. Die Host-CPU wird also in doppelter Funktion genutzt - sie ist sowohl zur Ausführung von Algorithmenteilen als auch als Kontroller einsetzbar.

Eine weitere Motivation war es mit dem System nicht nur dedizierte Anwendungen des ATLAS Triggersystems ausführen können, sondern eine universelles System zu entwickeln. Deshalb wurden bei der Konzeptentwicklung des Systems eine Reihe weiterer konkreter Anwendungsgebiete mitberücksichtigt. Dabei handelt es sich um zeitkritische Problemstellungen aus der Astronomie, bei Triggeranwendungen in der Schwerionenphysik und im Bereich der 2D/3D-Bildverarbeitung. Dies stellt nur eine Auswahl von Anwendungen dar, die konkret für die Ausführung auf ATLANTIS geplant sind, bzw. bereits ausgeführt werden. Die flexible Architektur des ATLANTIS-Systems schränkt grundsätzlich die Untersuchung und Ausführung weiterer Anwendungen nicht ein. Denkbar ist es beispielsweise, das ATLANTIS-System

als Prototypensystem für die Entwicklung von Anwendungen aus dem Bereichen Kryptographie oder der Bio-Informatik zu nutzen.

Die Akzeptanz und damit die tatsächliche Anwendung eines Spezialsystems wie ATLANTIS wird stark durch die praktische Bedienbarkeit bestimmt. Es war ein großes Anliegen des Autors, ein leistungsfähiges System zu schaffen, daß gleichzeitig auch für Nicht-Experten, ohne lange Einarbeitungszeit genutzt werden kann. Eine wichtige Rolle spielen in diesem Zusammenhang die Software-Tools, die parallel zum Aufbau des ATLANTIS-Systems, aber nicht im Rahmen dieser Promotion entwickelt bzw. adaptiert wurden. Damit ist sowohl die Programmierumgebung als auch das Betriebssystem für die hybride Architektur gemeint. Bezüglich der Programmierumgebung können Anwendungen für das ATLANTIS-System sowohl mit der weit verbreiteten Sprache VHDL als auch mit der in Mannheim entstehenden Sprache CHDL entwickelt werden. Bei beiden Ansätzen wird der Entwickler durch eine komfortable Simulationsumgebung und durch vorgefertigte, immer wieder auftauchende, Teillösungen unterstützt. Der Anwender braucht sich in diesem Fall nicht um essentielle Dinge zu kümmern, die typischerweise von jeder Anwendung vorausgesetzt werden, sondern kann sich voll und ganz auf die Abbildung seines speziellen Problems auf ATLANTIS konzentrieren. Ein Beispiel für eine immer benötigte funktionale Komponente ist das PCI-Interface zwischen FPGA-Teil und Host-CPU. Das Betriebsystem für ATLANTIS hat ganz allgemein die Aufgabe einen Satz von Basisfunktionen zur Verfügung zu stellen, mit denen das System kontrolliert werden kann. Dazu gehören unter anderem die Funktionen, mit deren Hilfe der Zugriff auf die FPGA-Designs ermöglicht wird aber auch Funktionen zur Konfiguration der FPGA-basierten Baugruppen (Konfiguration der FPGAs, des Clocksystems unter weiterer funktionaler Systembereiche). Der vollständige Entwicklungsprozeß einer Anwendung wird prinzipiell auf dem ATLANTIS-System ermöglicht. Die Zugriffe auf die FPGAs können als herkömmliche Funktionsaufrufe aus einem C/C++-Programm umgesetzt werden, wodurch für den Anwender die Nutzung der FPGA-Resourcen völlig transparent wird. Der Aufbau des ATLANTIS-Betriebssystems befindet sich noch im Anfangsstadium. Rudimentäre Funktionen sind bereits implementiert, so daß die Arbeit mit dem System durchgeführt werden kann. Um das System einem größeren Anwenderkreis vereinfacht nutzbar zu machen, steht unter anderem die Erstellung der obligatorischen graphischen Benutzeroberfläche noch aus. In einer gemeinsamen Umgebung sollten sämtliche Software-Tools vereint und ausgeführt werden können.

Ganz allgemein wird die Entwicklung von Hardware-Systemen (wie auch bei ATLANTIS), die auf aktueller Technologie basieren, permanent von einem entscheidenden Faktor begleitet: *Zeit*. Ganz konkret heißt das, daß unter Umständen die Entwicklung eines Systems so lange dauert (aus welchen Gründen auch immer), daß der besondere Nutzen durch die größere Leistungsfähigkeit von neueren (kommerziellen) Systemen oder auch nur Komponenten kompensiert bzw. sogar übertroffen wird. Um dies zu vermeiden muß bei der Entwicklung einer solchen Hardware-Architektur darauf geachtet werden, daß zukünftige Technologien in einem möglichst hohen Maße integrierbar sind, ohne daß ein komplett neues System aufgebaut werden muß (Upgrade ohne Redesign). Bei ATLANTIS wurde dies auf verschiedenen Ebenen mitberücksichtigt und umgesetzt. Zunächst wurde mit CompactPCI ein

kommerzielles und auf dem Markt wachsendes Basis-Bussystem gewählt. Damit ist eine längerfristige Verfügbarkeit aktueller Host-Prozessoren gewährleistet. Auf Boardebene wurden, bezüglich der Erweiterungsmöglichkeit auf leistungsfähigere Komponenten, verschiedene Dinge beachtet. Zunächst sind die festverdrahteten FPGA-Bausteine zu betrachten. Die Upgrademöglichkeiten werden hierbei durch die zukünftige Verfügbarkeit leistungsfähigerer Bausteine in dem gewählten Gehäusetyp bestimmt. Bezogen auf die Lucent-FPGAs des ACBs konnte dieses Ziel nicht erreicht werden. Das liegt daran, daß die für den Gehäusetyp angekündigten, leistungsfähigeren Bausteine wieder abgekündigt wurden. Umgekehrt läßt die Kombination aus Architektur und Gehäusetypen der Xilinx-FPGAs es zu, daß bereits FPGA-Bausteine der nächsten Generation für den Einsatz auf dem AIB berücksichtigt werden konnten. Auch wurden auf dem AIB die festverdrahteten Speicherbausteine so angeschlossen, daß auch hier die kommende Generation noch eingesetzt werden kann. Zu guter letzt sind über die Modul- und I/O-Schnittstellen von ACB und AIB generell neue Technologien integrierbar. Insgesamt kann an dieser Stelle festgehalten werden, daß das ATLANTIS-System auch in den nächsten Jahren noch als leistungsfähiges Prototypensystem eingesetzt werden kann, ohne daß ein Redesign der FPGA-basierten Komponenten nötig ist.

Für die Wahl der Kernkomponente des ATLANTIS-Systems, dem FPGA, wurde eine ausführliche Evaluierung durchgeführt, um den idealen Baustein zu finden. Nach Berücksichtigung einer Reihe von wichtiger Kriterien fiel die Wahl auf die Lucent ORCA-Series3-FPGAs. Zum Zeitpunkt der Systemkonzeption war dies auch in jeder Hinsicht gerechtfertigt. Durch die neue Virtex-FPGA-Architektur der Firma Xilinx, die seit 1999 auf dem Massenmarkt verfügbar ist, wurde eine neue innovative FPGA-Architektur zur Verfügung gestellt. In einer erweiterten Evaluierung wurden zur Bestückung des AIBs die Virtex-FPAGs ausgewählt und damit den Lucent-FPGAs vorgezogen. Dies hat sich auch im nach hinein als richtig erwiesen. Denn unter anderem haben sich bei der Arbeit mit der Lucent Place&Route-Software im Laufe der Zeit einige Mängel herausgestellt, die trotz intensiven Kontakts mit dem Lucent-Support bis heute nicht alle beseitigt werden konnten. Als Fazit kann bezüglich dieser Sache gesagt werden, daß in einer vergleichbaren Situation der Systemaufbau eher verzögert werden sollte. Im Fall des ATLANTIS-Systems war dies nicht möglich, weil durch den ATLAS LVL2-Projekt-Zeitplan eine harte Grenze für die Demonstration des Full-Scan-TRT-Algorithmus' vorgegeben war. Aus jetziger Sicht kann gesagt werden, daß mittelfristig die Firma Xilinx mit den FPGA-Typen der Virtex-Familie den Markt der komplexesten FPGAs beherrschen wird. Dies kann auch bezüglich der Programmier-Software und dem generellen Support so unterstrichen werden. Es ist eher als unwahrscheinlich anzusehen, daß ein anderer FPGA-Hersteller in den kommenden Jahren als ernsthafter Konkurrent komplexere und leistungsfähigere FPGAs auf den Markt bringt. Für die Entwicklung oder die Anschaffung des zukünftigen FPGA-Koprozessorsystems kann prognostiziert werden, daß als zentraler Baustein ein Xilinx Virtex-FPGA erste Wahl sein muß und wird.

Mit der Wahl der Systemarchitektur zum Aufbau eines flexiblen, hybriden Systems konnten bis jetzt schon zwei Anwendungen in der Praxis zeigen, daß der richtige Weg eingeschlagen wurde. Aber auch die theoretischen Überlegungen zur Implementierung weiterer geplanter Anwendungen bestätigen, daß die

Systemarchitektur leistungsfähig und flexibel genug ist, um optimal an eine bestimmte Applikation angepaßt zu werden. Dies drückt sich beispielsweise darin aus, daß mit dem ACB nur durch Austausch der Modul-Karten und der FPGA-Designs innerhalb weniger Minuten von einer Triggeranwendung zu einer Volumenvisualisierung gewechselt werden kann. Andererseits können beispielsweise für die N-Körper-Simulationen in der Astronomie bereits vorhandene ASIC-basierte Baugruppen für den PCI-Bus integriert werden. Als weiteres Beispiel zeigt die flexible I/O-Schnittstelle des AIBs, daß sowohl Daten für die ATLAS Readout-Systeme als auch für den FOPI-Trigger über ein und dieselbe physikalische Schnittstelle, nur durch den Austausch einer einfachen Schnittstellenkarte, in das System übertragen werden können. Die entsprechende Protokolllogik wird in die FPGAs programmiert.

Bis jetzt (Stand September 2000) wurden zwei ATLANTIS-Systeme in der Praxis eingesetzt. Die Systeme bestehen aus einem kommerziellen Hostprozessor-System und jeweils einem ACB und einer ATB, die den Privat-Bus beinhaltet. Die Fertigstellung des AIBs steht kurz vor Abschluß. Durch die zeitlich nachfolgende Entwicklung des AIBs konnten Erfahrungen, die beim Aufbau und Test des ACBs gemacht wurden, bereits in das AIB-Design mit einfließen. Zum Beispiel hat das mühsame Debuggen der PCI-Lokalbus-Schnittstelle auf dem ACB dazu geführt, daß auf dem AIB dieses Interface zum direkten Anschluß an einen Logik-Analysator ausgeführt wurde. Das AIB wurde bereits vollständig geroutet. Kleine Änderungen im Platinendesign werden zur Zeit noch durchgeführt. Die Produktion des Boards ist für Oktober geplant. Nach den bisherigen gemachten Erfahrungen, beim Aufbau und Test des ACBs, kann das AIB bis spätestens Ende des Jahres eingesetzt werden.

Bisher konnten auf dem ATLANTIS-System schon zwei Anwendungen umgesetzt werden. Für die Ausführung des Full-Scan-TRT-Algorithmus für den ATLAS LVL2-Trigger wurden zwei ACBs, bestückt mit jeweils vier TRT-Speicher-Modulen in einem System parallel betrieben. Für die Volumenvisualisierungsanwendung wurde ein ACB und ein speziell entwickeltes Aufsatzmodul eingesetzt. Die Programmierung wurde sowohl in VHDL und als auch in CHDL ausgeführt. Für den Full-Scan-TRT wurden die eigentlichen Algorithmen-Designs in VHDL entwickelt. Die Zugriffsdesigns für die TRT-Speicher-Module zum Testen aber auch zum Laden der LUT wurden in CHDL formuliert. Damit wurde in der Praxis gezeigt, daß beide Ansätze zur Designentwicklung für ATLANTIS simultan genutzt werden können. Die Entwicklung der Volumenvisualisierungs-Anwendung wurde ausschließlich unter Verwendung von VHDL durchgeführt.

Die erfolgreiche Ausführung des im Rahmen dieser Promotion adaptierten Full-Scan-TRT-Triggeralgorithmus' auf dem ATLANTIS-System hat zwei Dinge gezeigt: Grundsätzlich war dies der Funktionsnachweis für die Verwirklichung des ATLANTIS-Systemkonzepts. Damit wurde die Anwendbarkeit und das Funktionieren der Hardware (ACB) als auch die praktische Nutzbarkeit der Software-Tools gezeigt. Bezüglich des ATLAS LVL2-Triggers konnte nachgewiesen werden, daß durch die Anwendung des Koprozessorkonzepts eine signifikante Reduktion der benötigten Farmprozessoren möglich ist. Dies basiert auf der um einen Faktor 5.6 beschleunigten Ausführung des Full-Scan-TRT-Algorithmus gegenüber einer rein CPU-basierten Ausführung. Vergleicht man den auf den FPGAs gerechneten Teil des

Full-Scan-TRTs mit der CPU-Ausführung ergibt sich sogar ein Beschleunigungsfaktor von 12.5. Die offiziellen Zahlen, basierend auf den Meßergebnissen mit dem ATLANTIS-Systems, wurden im März 2000 im *ATLAS High-LEVEL TRIGGERS, DAQ AND DCS TECHNICAL PROPOSAL* veröffentlicht [AHL00]. Durch den Einsatz von FPGA-Prozessoren als Koprozessoren zur Beschleunigung von B-Physik-Triggeralgorithmen reduziert sich die benötigte Anzahl der Farmprozessoren von ~770 auf nur noch ~450. Unter Berücksichtigung von für das Jahr 2001 angekündigten FPGA-Architekturen, kann diese Zahl nochmals auf nur noch 194 Farmprozessoren reduziert werden, die für die endgültige ATLAS LVL2-Triggerfarm benötigt werden. Mit einer Anzahl von unter 200 Netzwerkknoten wird eine Gesamtnetzwerkgröße in Aussicht gestellt, die unter den LVL2-Trigger-Randbedingungen eine Farmarchitektur realisierbar erscheinen läßt. Gleichzeitig führt dies unter Umständen auch zu einer kostengünstigeren Gesamtlösung. Die Kosten für einen FPGA-Koprozessor für das Jahr 2003 können mit ca. 4000,- DM abgeschätzt werden. Man muß dabei berücksichtigen, daß nicht nur die Anzahl der Farmprozessoren reduziert wird, sondern auch die Netzwerk-Infrastruktur kleiner ausfällt.

Als Ausblick bleibt an dieser Stelle festzuhalten, daß das ATLANTIS-System dauerhaft am CERN im LVL2-Trigger-Prototypensystem installiert werden soll. Dies wird dazu dienen, weitere geeignete ATLAS LVL2-Triggeralgorithmen im Umfeld einer Prozessorfarm zu untersuchen. Die Untersuchung weiterer B-Physik-Triggeralgorithmen für FPGA-basierte Systeme soll von Mannheimer Seite aus fortgeführt werden. Zu Zeit werden neben dem Full-Scan-TRT-Algorithmus auch der Pixel-Scan mit anschließender Spurverfolgung in den SCT favorisiert. Als Alternative soll auch die Spurverfolgung vom TRT aus in den SCT evaluiert werden.

Als Gesamtfazit aus der Arbeit mit dem ATLANTIS-System kann folgender Schluß gezogen werden. Es wurde eine neuartige Rechnerarchitektur entwickelt, die die einfache und effiziente Kopplung von Standard-CPUs und einem Multi-FPGA-Prozessorsystem erlaubt. Sowohl Hardware- als auch Softwareaspekte wurden berücksichtigt. Die Umsetzung des Konzepts wurde erfolgreich durchgeführt. Das System wurde bereits in der Praxis getestet. Es muß an dieser Stelle nochmals betont werden, daß die gestellte Aufgabe, die in der hier vorliegenden Arbeit beschrieben wird, im vollen Umfang niemals von einer Person in dem vorgegebenen Zeitraum alleine hätte durchgeführt werden können. Zusätzlich war aufgrund des propagierten universellen Charakter des Systems indirekt ein größerer Personenkreis in die Entwicklung des Systems miteinbezogen. Mitentscheidend für die Umsetzung des ATLANTIS-Systemkonzepts war das erfolgreiche Arbeiten im Team. Ein wichtiges Fazit ist, daß die erfolgreiche und schnelle Bearbeitung von Teilaufgaben im besonderen auf eine gute Teamarbeit zurückzuführen sind. Dabei standen Diskussions- und Dialogfähigkeit im Vordergrund.

Ein Blick in die weitere Zukunft zeigt, daß das ATLANTIS-System, mit dem Grundgedanken FPGA-Resourcen und eine Standard-CPU effizient zu koppeln, nur ein Zwischenschritt beim Übergang von einer hybriden Architektur auf Systemebene zu einer hybriden Architektur auf Chipebene darstellt. Dies wird untermauert durch die folgende Pressemeldung. Im Juli 2000 wurde offiziell bekannt gegeben, daß die Firma Xilinx und der Prozessorhersteller IBM eine Kooperation eingehen wollen. Ziel ist es, die PowerPC-Prozessoren von IBM mit FPGA-Resourcen auf

Chipebene zu koppeln. Damit sind die Weichen für moderne Architekturen gestellt. Bezogen auf diese Arbeit läßt sich der Blick in die Zukunft für kommende Triggersysteme in einem Wort deuten: Trigger-on-a-Chip. Dieser Ausdruck stellt im Moment vielleicht noch eine wenig greifbare Zukunftsaussicht dar - die Umsetzung von Triggerproblemen auf eine solche Architektur (in nicht allzu ferner Zeit) liegt aber durchaus im Bereich des möglichen.

Anhang A

Das ATLANTIS-System: Technische Spezifikation

Das ATLANTIS-System basiert auf dem CompactPCI-Bus und verwendet den 6-HU-Formfakor. ATLANTIS baut auf einem Standard-6-HU-CompactPCI-Crate auf und beinhaltet fünf funktionsspezifische Baugruppen: Das ATLANTIS Computing-Board (ACB), das ATLANTIS I/O-Board (AIB), den ATLANTIS Privat-Bus (3-HU-Format), ein kommerzielles CompactPCI-Prozessorsystem und eine 3-HU-CompactPCI-Systembusplatine.

ATLANTIS - Globale technische Eigenschaften	
Betriebssystem	Windows NT 4.0 und Linux
PCI-Datenbusbreite	32 Bit
PCI-Taktfrequenz	33 MHz
Übertragungsleistung PCI (Theorie)	132 MByte/s
Übertragungsleistung PCI (gemessen)	110 MByte/s (Read) / 65 MByte/s (Write)
Betriebsfrequenzbereich (FPGA-Teil)	5 MHz - 80 MHz

Tabelle A.1: ATLANTIS: Globale technische Eigenschaften

ATLANTIS Computing-Board: Technische Eigenschaften	
FPGA-Typ	Lucent Or3T125
FPGA-Anzahl	4
integrale FPGA-Komplexität	744k Logik-Gatter
Datenbusbreite zwischen den FPGAs (Ring-Topologie)	72 Bit
Privat-Bus-Interface (Datenbusbreite)	2 x 72 Bit über 2 der 4 FPGAs
Anzahl Modul-Ports	1 Modul-Port pro FPGA
Datenbusrreite pro Modul-Port	206 Bit
I/O-Interface	2 x 28-Bit-LVDS (S-Link, M-Link-kompatibel)

Tabelle A.2: ATLANTIS Computing-Board: Technische Eigenschaften

ATLANTIS I/O-Board: Technische Eigenschaften	
FPGA-Typ	Virtex XCV600 - XCV2000E (variable Bestückung)
FPGA-Anzahl	2
Integrale FPGA-Komplexität	372k - 1036k Logik-Gatter
Block-RAM pro FPGA	295 kBit - 655 kBit
Privat-Bus-Interface (Datenbusbreite)	72 Bit pro FPGA
Anzahl Dual-Port-RAM-Bänke	4
Organisation Dual-Port-RAM-Bank	32k x 36
Integrale Speicherdichte Dual-Port-RAM	576 kByte
Anzahl SSRAM-Bänke	4
Organisation SSRAM-Bank	2 x 256k/512k x 36 (sequent. Zugriff)
Integrale Speicherdichte SSRAM	8 MByte - 16 MByte (variable Bestückung)
Anzahl unabhängiger I/O-Ports	4
Datenbusbreite pro I/O-Port	45 Bit
I/O-Port Typ	64poliger Steckverbinder nach IEEE-3286
2 microEnable2-kompatible I/O-Ports (mechanisch), Datenbus überlappt mit zwei der vier I/O-Ports	68poliger SCSI-3-Steckverbinder, jeweils ein 32-Bit-breiter Datenbus

Tabelle A.3: ATLANTIS I/O-Board: Technische Eigenschaften

ATLANTIS Privat-Bus: Technische Eigenschaften	
Anzahl Slots	variabel (eingeschränkt nur durch Crate-Breite)
Art der Steckverbinder	Standard-2mm-pitch-CompactPCI-Steckverbinder nach IEC1076-4-101
Datenbus-Breite pro Slot	144 Bit
maximale Datentransferrate pro Slot	1.44 GByte/s
Ausführung als ATB (existiert als 8-Slot-Version)	festverdrahteter 72-Bit-Datenbus; Verbindung zu jeweils zu linken und rechten Nachbar-Slot
Ausführung als AAB (geplant)	144-Bit-Datenbus, konfigurierbar über Cross-Bar-Switch

Tabelle A.4: ATLANTIS Privat-Bus: Technische Eigenschaften

Anhang B

Bilder von ATLANTIS-Systemkomponenten

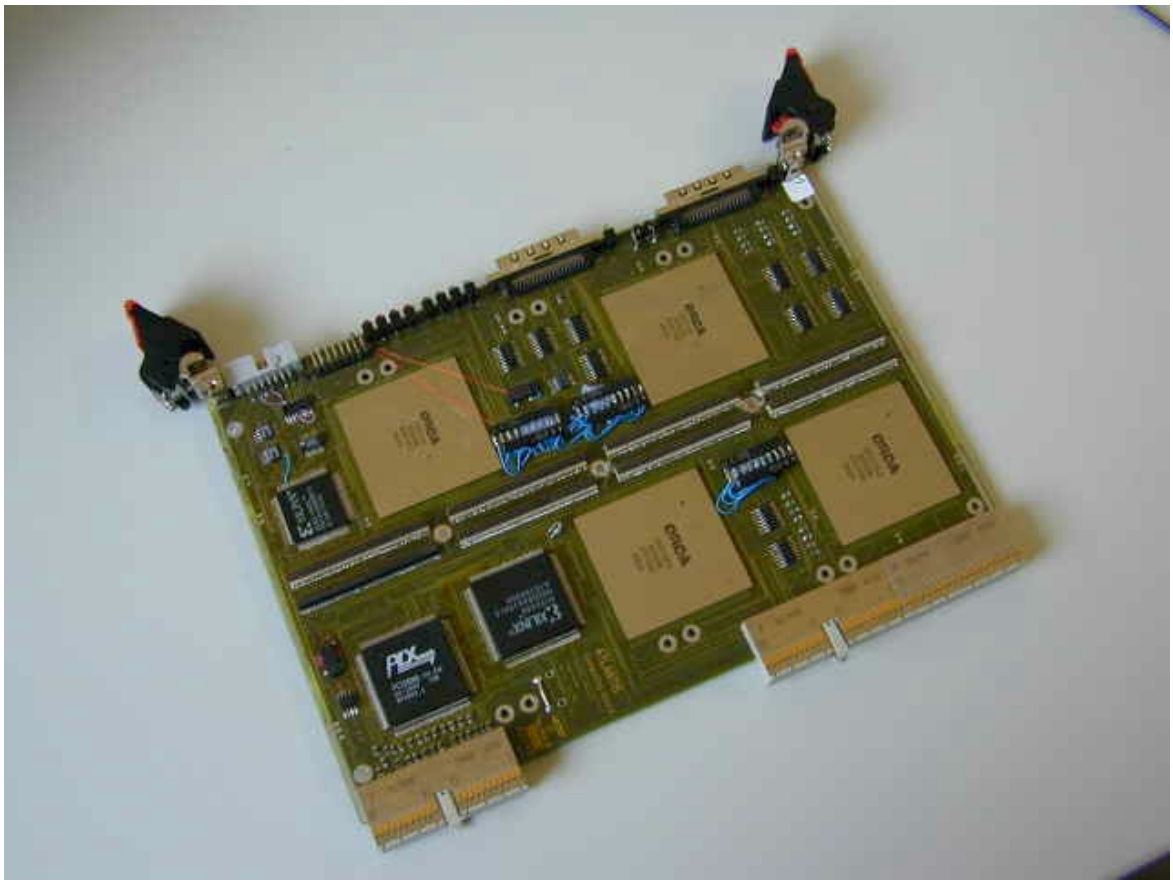


Abbildung B.1: Foto des ATLANTIS Computing-Boards

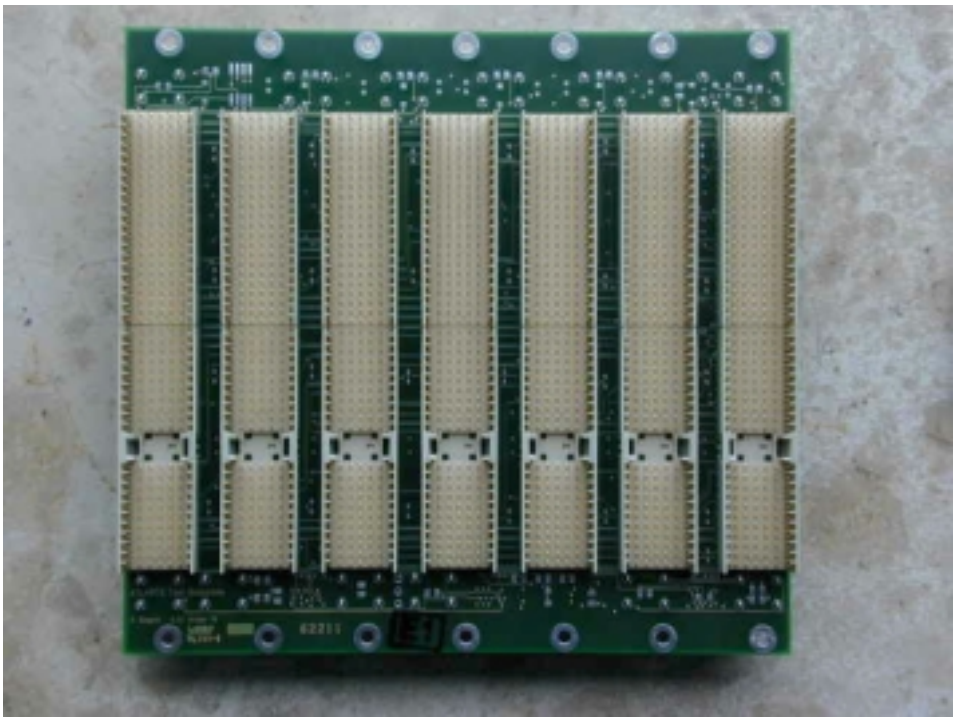


Abbildung B.2: Foto der ATLANTIS Test-Backplane

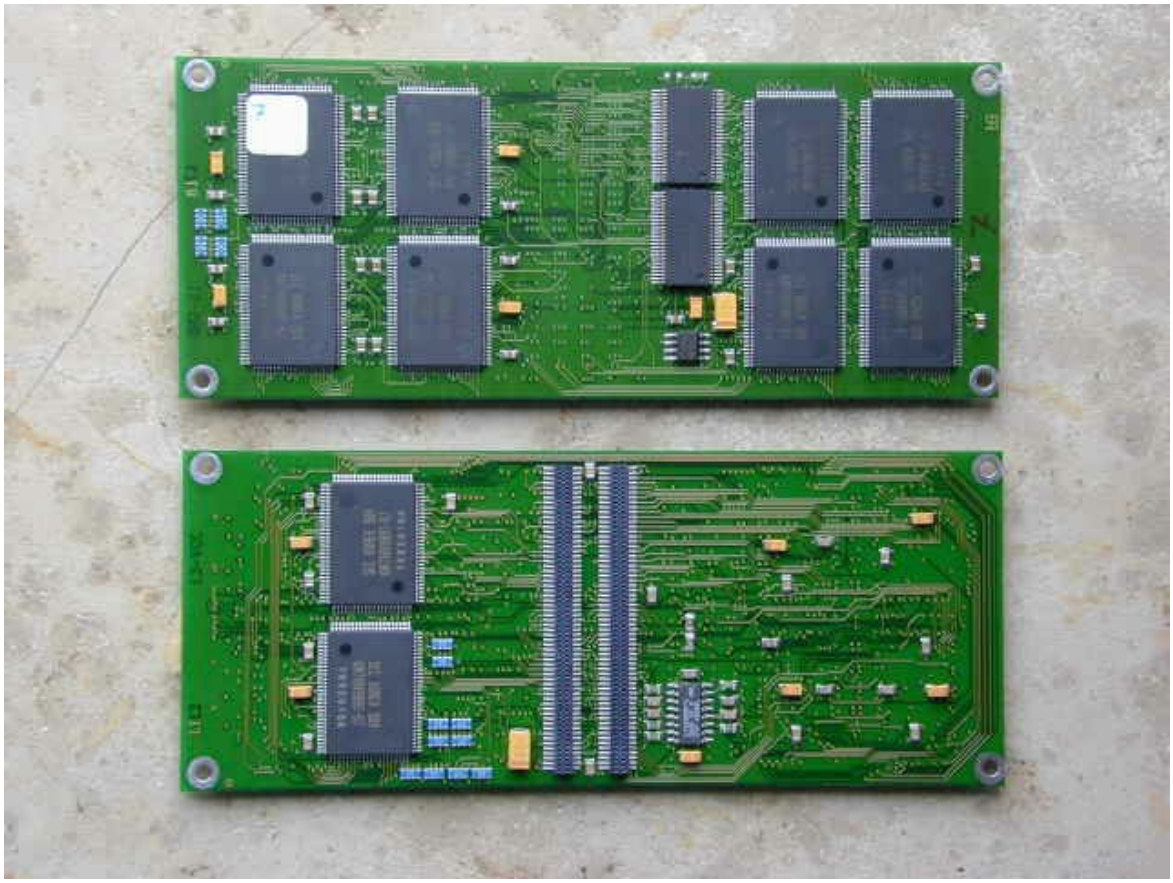


Abbildung A.3: Foto des TRT-Speicher-Moduls (Vorder- und Rückseite)

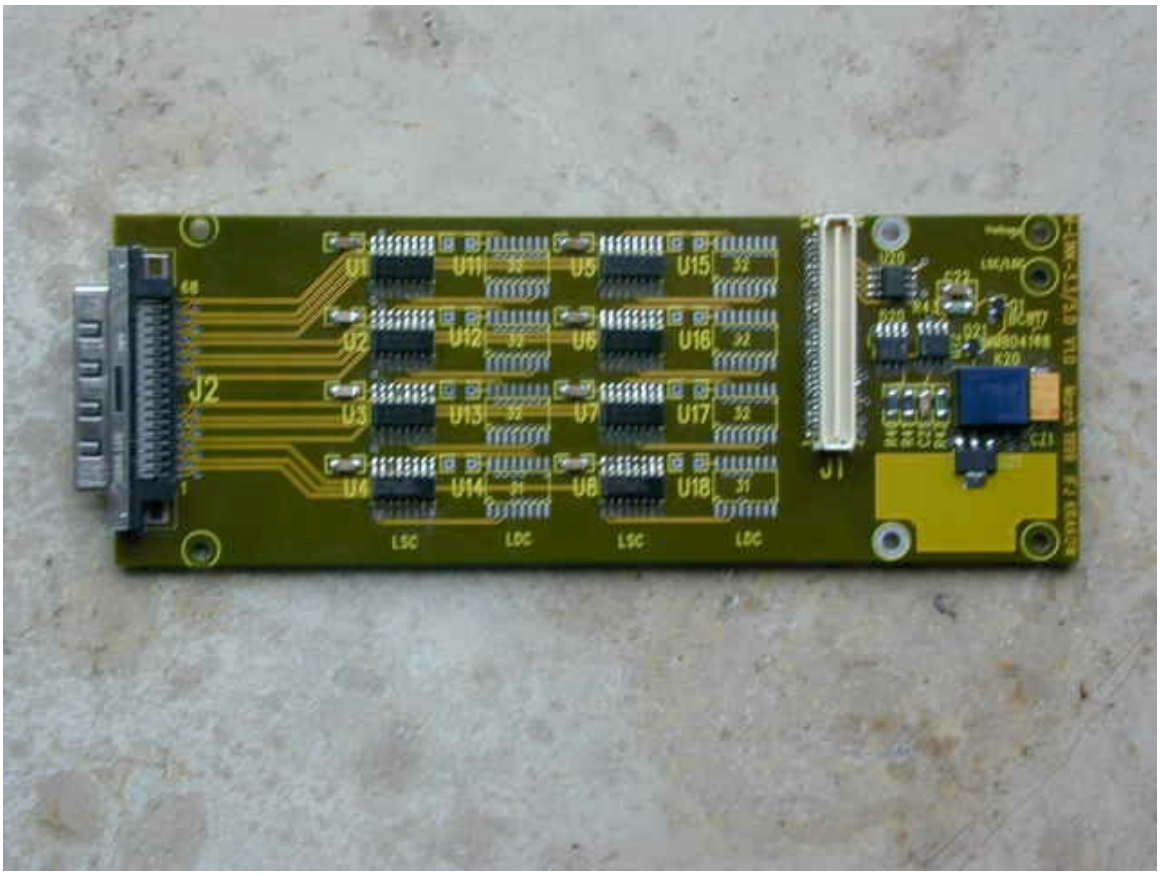


Abbildung B.4: Foto des M-Link-Schnittstellen-Moduls

Literaturverzeichnis

[Act96] *Actel's Reprogrammable SPGAs - Preliminary Advance Information,* Actel Corporation, 1996

[ADE98] *ATLAS DAQ, EF, LVL2 and DCS Technical Progress Report,* ATLAS Trigger / DAQ Community, CERN / LHCC 98-16, CERN, 1998

[ADP99] *ATLAS Detector and Physics Performance Technical Design Report,* ATLAS Collaboration, CERN / LHCC / 99-14, CERN, 1999

[AHL00] *ATLAS High-Level Triggers, DAQ and DCS Technical Proposal,* ATLAS Trigger / DAQ Community, CERN / LHCC 2000-17, CERN, 2000

[AKi96] A. Auer und R. Kimmelmann: *Schaltungstest mit Boundary Scan,* Hüthig Verlag, 1996

[ALI95] A Large Ion Collider Experiment - *Technical Proposal, ALICE Collaboration*, CERN / LHCC 95-71, 1995

[Alt98] *Altera Flex 10K/10KA Embedded Programmable Logic Familiy*, Altera Corporation, 1998

[ATD98] *ATLAS Trigger-DAQ: Level-2 Pilot Project*, ATL-DAQ-98-118, ATLAS Note, CERN, 1998

[Atm97] *AT40K FPGAs,* ATMEL, 1997

[ATP94] *ATLAS Technical Proposal for a General-Purpose pp Experiment at the LARGE HADRON COLLIDER at CERN,* CERN / LHC / 94-43, LHCC / P2, 1994

[BAK96] D. A. Buell, J.M. Arnold, W.J. Kleinfelder: *Splash2: FPGAs in a custom computing machine,* IEEE Computer Society Press, 1996

[BBB93] J. Badier et al.: *Evaluating Parallel Architectures for Two Real-Time Applications with 100 kHz Repitition Time*, in IEEE Transaction on Nuclear Science, Vol. 40, No 1, Feb 1993

[BBE00] John Baines et al.: *B-Physics Event Selection for the ATLAS High Level Trigger,* ATL-DAQ-2000-031, CERN, 2000

[BBH99] J. Baines R. Bock, C. Hinkelbein, A. Kugel, R. Männer, M. Müller, M. Sessler, H. Simmler, H. Singpiel, M. Smizanska: *Global Pattern Recognition for B-Physics in the ATLAS Trigger,* ATL-DAQ-99-012, ATLAS Note, CERN, 1999

[BBo93] P. Bertin und P. Boucard. *DecPeRLe-1, Hardware Programmers Manual*, DEC-PRL, DECPeRLe-1 documentation, 1993

[BCE96] J. Bystricky et al.: *A Sequential Processing Strategy For The ATLAS Event Selection*, Proc. Nuclear Science Symposium and Conference in Anaheim, CA, USA, 1996

[BDH98] O. Brosch et al.: *The FPGA Processor Enable++ as a Trigger System for FOPI and Hirich*, GSI Scientific Report 1998 176, GSI, 1998

[BFV99] R. Bock et al.: *The active ROB complex*, ATL-DAQ-99-020, ATLAS Note, CERN, 1999

[BMB96] Owen Boyle, Robert McLaren & Eric van der Bij: *The S-LINK Interface Specification*, CERN, 1996, verfügbar über http://www.cern.ch/HSI

[BNS97] R. Bock et al.: *TRT Preprocessing Algorithm, Implementations and Benchmarks*, ATLAS Note, ATL-DAQ-97-066, CERN, 1997

[BPh97] Vernon D. Barger, Roger J.N. Phillips: *Collider Physics - updated edition*, Addison-Wesley Publishing Company, 1997

[Bro00] Oliver Brosch, *private Mitteilung*, Mannheim, 2000

[BSC98] Hans-Joachim Blank, Hermann Strass, Roland Chochoiek: *PCI in der Industrie*, Markt&Technik Buch- und Software-Verlag, 1998

[BSW99] R. Brinkmann, P. Schmüsser, B.H. Wiik: *TESLA*, in Physik in unserer Zeit, 30. Jahrgang, März, 1999

[CMS94] *The Compact Muon Soleonid - Technical Proposal*, CMS Collaboration, CERN/LHCC 94-38, 1994

[Com97] *CompactPCI Local Bus Specifikation 2.0, Revision 2.1*, PICMG, 1997, verfügbar unter http://www.picmg.org

[Cyp95] *Cypress ICD2053B Programmable Clockgenerator Data Sheet*, Cypress Semiconductor Corporation, USA, 1995

[Cyp98a] *Cypress CY7B991V 3.3V RoboClock Low Voltage Programmable Skew Clock Buffer Data Sheet*, Cypress Semiconductor Corporation, USA, 1998

[Cyp98b] *Cypress - Everything You Need to Know About CY7B991/2 (RoboClock) and the RoboClock Family Application Note*, Cypress Semiconductor Corporation, USA, 1998

[Cyp98c] *Cypress CY2308 3.3V Zero Delay Buffer Data Sheet*, Cypress Semiconductor Corporation, USA, 1998

[DKR95] V. Dörsing, P. Kammel, A. Reinsch: *A Data Collection and Preprocessing Unit for the LVL-2 Trigger of ATLAS*, East internal note, EAST Note 95-02, 1995

[Dyn98] *DL6000 Family Fast Field Programmable Gate Array*, Dynachip, 1998

[Fab00] Fabiola Gianotti: *Collider Physics: LHC*, ATL-CONF-2000-001, CERN, 2000

[Gal95] Dave Galloway: *Transmogrifier C Hardware Description Language and Compiler for FPGAs*, Proc. IEEE Symposium on FPGAs for Custom Computing Machines (FCCM), Napa Valley, CA, USA, 1995

[Gav96] I. Gavrilenko: *Description of Global Pattern Recognition Programm (xKalman)*, ATLAS-INDET-note-165, CERN, 1996

[GBM90] Gläß J., Baur R., Männer R.: *Programmable Trigger for Electron Pairs in Ring Image Cherenkov Counters*, IEEE Tr. Nucl. Sci. NS-37, 2 (1990) 241-247, 1990

[GPR94] T. Günther et al.: VIRIM: *A Massively Parallel Processor for Real-Time Volume Visualization in Medicine,* W.Straßer, Proc. 9th Eurographics Workshop on Graphics Hardware, Oslo, Norwegen, 1994

[Gri96] D. J. Griffiths: *Einführung in die Elementarteilchenphysik,* Akad. Verlag, Berlin, 1996

[Hin00] C. Hinkelbein, *private Mitteilung*, Mannheim, 2000

[HKL00] C. Hinkelbein et al.: *Prospects of FPGAs for the ATLAS LVL2 Trigger*, ATL-DAQ-2000-006, ATLAS Note, CERN, 2000

[HKM99] C. Hinkelbein, A. Kugel, R. Männer, M. Müller, M. Sessler, H. Simmler, H. Singpiel: *Pattern Recognition Algorithms on FPGAs and CPUs for the ATLAS LVL2 Trigger*, Proc. IEEE Real-Time Conference, Santa Fe, NM, USA, 1999

[HLR98] *Handel-C Language Reference Manual*, Embedded Solutions, 1998, verfügbar über http://www.embedded-solutions.ltd.uk/corp_page.htm

[Hor93] P. Horowitz, W. Hill: *The Art of Electronics, 2nd ed.,* Cambridge University Press, London, 1993

[HVS99] J. Hesser, B. Vettermann, H. Singpiel, A. Kugel, R. Männer: *Implementation of Algorithmically Optimized Volume Rendering on FPGA Hardware,* Proc. IEEE Visualization '99, San Francisco, CA, USA, 1999

[ICu97] *I-Cube IQ Family Data Sheet*, I-Cube Inc., USA, 1997

[IDT99] *IDT 70V3579S High-Speed 3.3V 32K x 36 Synchronous Pipelined Dual-Port Static RAM, Integrated Device Technology Inc., USA, 1999*

[KBD98] Klaus Kornmesser et al.: *Simulating FPGA-Coprocessors using the FPGA Development System CHDL,* Proc. International Conference on Parallel Architectures and Compilation Techniques 78-82, Paris, Frankreich, 1998

[KDL98] A. Kugel et al.: *microEnable - A Reconfigurable FPGA coprocessor*, Proc. 4th Workshop on Electronics for LHC Experiments 402-406, Rom, Italien, 1998

[KKL98] A. Kugel et al.: *50 kHz Pattern Recognition on a Large FPGA Processor*, Proc. IEEE Symposium on Field-Programmable Custom Computing Machines 262-263, Napa Valley, CA, USA, 1998

[KBC98] A. Kugel et al.: *ATLAS Level-2 Trigger Demonstrator-A Activity Report Part 1: Overview and Summary*, ATL-DAQ-98-085, ATLAS Note, CERN, 1998

[KKM99] T. Kuberka, A. Kugel, R. Männer, H. Singpiel, R. Spurzem, R. Klessen: *AHA-GRAPE: Adaptive Hydrodynamic Architecture – GRAvity PipE,* Proc. Int´l Conf. on Parallel and Distributed Processing Techniques and Applications '99 417, vol 3, Las Vegas, USA, 1999

[KLL97] A. Kugel et al.: *A Hybrid Approach for the ATLAS Level-2 Trigger*, Proc. 3rd Workshop on Electronics for LHC Experiments, London, UK, 1997

[KMo94] W. Krischer, L. Moll: *Implementation of pattern recognition algorithms for the Si trakker on DecPeRLe-1*, EAST Note 94-21, CERN, 1994

[Kor00] Klaus Kornmesser: *CHDL: Eine C++-basierte Hardwarebeschreibungssprache für FPGAs und PLDs,* Online-Benutzerhandbuch, Mannheim, 2000

[Kub99] T. Kuberka: *Leistungsanalyse von Gleitkommaarithmetik auf FPGA-Prozessoren an Hand der Implementierung des SPH-Algorithmus (Schritt 1)* , Diplomarbeit, Universität Mannheim, 1998

[Kug99] Andreas Kugel: *ATLANTIS Memory Board TrtMem Specification Version 1.5,* Mannheim, 1999

[LEP00] *LEP's last lap sprint: LEP achieves record energy levels,* CERN Courier Volume 40, Number 5, June 2000, CERN, 2000

[Lev00] L. Levinson: *private Mitteilung*, CERN, 2000

[LHC95] *The Large Hadron Collider, Conceptual Design Report*, LHC Study group, CERN / AC 95-05, 1995

[LHCb95] *Letter of Intent for a Dedicated LHC Collider Beauty Experiment for Precision Measurements of CP-Violation,* LHCb Collaboration, CERN / LHCC95-5, 1995

[LLM95] L. Lundheim, I. C. Legrand, L. Moll: *A Neural Network for the Global second Level Trigger - A Real-time implementation on DecPeRLe-1*, ATL-DAQ-95-044, CERN, 1995

[LMM98] W. Ligon et al.: *A Re-evaluation of the Practicality of Floating-Point Operations on FPGAs,* Proc. IEEE Symposium on FPGAs for Custom Computing Machines

[Loh92] E. Lohrmann: *Hochenergiephysik,* Teubner Studienbücher, 4. überarbeitete und erweiterte Auflage, Stuttgart, 1992

[LRG97] Level-2 Requirements group: *ATLAS Level-2 Trigger User Requirements Document,* ATL-DAQ-97-079, ATLAS Note, CERN, 1997

[Luc98] *Lucent Orca Series 3C and 3T Field-Programmable Gate Arrays Data Book*, Lucent Technologies, USA, 1998

[Lud98] J. Ludvig: *Enable++: Ein universeller FPGA-Triggerprozessor für das ATLAS-Experiment*, Dissertation, Ruprecht-Karls-Universität Heidelberg, 1998

[Mak97] J. Makino et al.: *GRAPE-4: A Massively Parallel Special Purpose Computer for Collisional N-Body Simulations*, Astrophysical Journal, Vol 480, 1999

[MBe99] *MicroEnable Benutzerhandbuch*, Silicon Software GmbH, Mannheim, 1999, verfügbar über http: / / www.silicon-software.de

[MBi96] R. A. McLaren et al.: *An Application of S-LINK, a Data Link Interface Specification, in the ATLAS Readout System,* Publ. of Second International Data Acquisition Workshop (DAQ96) on Networked Data Acquisition systems, RCNP Osaka, JAPAN, 1996

[Mic99] *Micrel MIC39500/39501 5A Low-Voltage Low-Dropout Regulator Data Sheet*, Micrel Inc., USA, 1999

[Mit99] *VolumePro, a PCI based volume rendering coprocessor by Mitsubishi Electronics America Inc.*, RTVIZ, 1999, verfügbar über http: / / www.rtviz.com

[Mit00] V.A. Mitsou: *Search for new Physics with ATLAS at the LHC,* ATL-CONF-2000-002, CERN, 2000

[MSH00] M. Müller et al.: *ATLANTIS - a modular, hybrid FPGA/CPU processor for the ATLAS Readout Systems,* acc. for publ. in Proc. of 6th Workshop on Electronics for LHC Experiments, Krakow, Polen, 2000

[Mül00] H. Müller: *Hartmann Elektronik*, private Mitteilung, 1999

[Mül00] Matthias Müller: *ROB Scenario on ATLANTIS*, private Mitteilung, Mannheim, 2000

[Nof95] K.-H. Noffz: *Ein FPGA-Prozessor als 2nd-Level-Trigger für ATLAS*, Dissertation, Ruprecht-Karls-Universität Heidelberg, 1995

[PCI95] *PCI Local Bus Specifikation 2.0, Revision 2.1,* PCI Special Interest Group, 1995, verfügbar über http://www.pcisig.com

[Per00] *Pericom PI3B3125 3.3V 4-Bit 2-Port Bus-Switch Data Sheet,* Pericom Semiconductor Corporation, USA, 2000

[PLX98] *PLX PCI 9080 Data Book Version 1.05, PLX Technology Inc., USA*, 1998

[PRS93] B. Povh, K. Rith, C. Scholz, F. Zetsche: *Teilchen und Kerne,* Vorlesungsskript, Universität Heidelberg, 1993

[Ris99] Rüdiger Rissmann: *Implementierung von Vorverarbeitungsalgorithmen für den ATLAS Level 2 Trigger auf dem FPGA-Prozessor microEnable,* Diplomarbeit, Ruprecht-Karls-Universität Heidelberg, 1999

[Ryn98] John Rynearson: VME320/2eSST Protocol, Vita Journal October, Vita, 1998, verfügbar unter http://www.vita.org

[Sam98] *Samsung KM718V989 512Kx18 Synchronous SRAM Data Sheet*, Samsung Electronics CO., USA, 1998

[Sam99a] *Samsung KM736V887 256Kx36 Synchronous SRAM Data Sheet*, Samsung Electronics CO., USA, 1999

[Sam99b] *Samsung KM736V987 512Kx36 Synchronous SRAM Data Sheet*, Samsung Electronics CO., USA, 1999

[Sam99c] *Samsung KM736V947 512Kx36 Flow-Trough NtRAM Data Sheet*, Samsung Electronics CO., USA, 1999

[Sam99d] *Samsung KM416S8030BT: 2M x 16Bit x 4 Banks Synchronous DRAM Data Sheet*, Samsung Electronics CO., USA, 1999

[SDE98] H. Simmler et al.: *Demonstrator Results Architecture A, ATL-DAQ-98-084, ATLAS Note,* CERN, 1998

[Ses00] M. Sessler: *Algorithmen auf CPUs und FPGAs für den ATLAS LVL2 Trigger,* Dissertation, Ruprecht-Karls-Universität, Heidelberg, 2000

[Sim99] H. Simmler, *private Mitteilung*, Mannheim, 1999

[Sin96] H. Singpiel: *Entwicklung des synchronen Hochgeschwindigkeits-Verbindungsnetzwerks für den Second Level Trigger-Prozessor Enable++,* Diplomarbeit, Ruprecht-Karls-Universität Heidelberg, 1996

[SLM00] H. Simmler et al.: *Multitasking on FPGA coprocessors,* acc. for publ. in Proc. 10th International Conference on Field Programmable Logic and Applications, Villach, Österreich, 2000

[SHH99] H. Singpiel et al.: *ATLANTIS - A Hybrid Approach Combining the Power of FPGAs and RISC Processors based on CompactPCI,* Proc. International Symposium on Field Programmable Gate Arrays 245, Monterey, CA, USA, 1999

[SHK00] H. Singpiel et al.: *LVL2 Full TRT Scan FEX algorithm for B-Physics performed on the hybrid FPGA/CPU processor system ATLANTIS: Measurements results,* ATL-DAQ-2000-12, ATLAS Note, CERN, 2000

[SKK98] H. Singpiel et al.: *ATLAS Level-2 Trigger Demonstrator-A Activity Report Part 3: Paper Model,* ATL-DAQ-1998-101, ATLAS Note, CERN, 1998

[Tex97] *Texas Instruments SN74ALB16244DL 16-Bit Buffer/Driver with 3-State Outputs Data Sheet,* Texas Instruments Incorporated, USA, 1997

[Tex98] *Texas Instruments SN74CBTD16210 20-BIT FET Bus Switch with Level Shifting Data Sheet,* Texas Instruments Incorporated, USA, 1998

[Van98] Vantis VF1 Field Programmable Gate Array, Vantis Corporation

[Ver00] Jos Vermeulen: *Paper modelling of the ATLAS LVL2 trigger system,* verfügbar über http://www.nikhef.nl/pub/experiments/atlas/daq/modelling.html

[Vet00] Bernd Vettermann, *private Mitteilung,* Mannheim, 2000

[VHM99] B. Vettermann, J. Hesser, R. Männer: *Solving the Hazard Problem for Algorithmically Optimized Real-Time Volume-Rendering,* Proc. International Workshop on Volume Graphics, Swansea, UK, 1999

[Vli00] *VHDL library, User Manual,* Silicon Software GmbH, Mannheim, 2000

[VME87] *VMEbus Specifications, VMEbus International Trade Association,* ANSI/IEEE STD 1014-1987, 1987

[VME97] *VME64 Extensions, VMEbus International Trade Association,* ANSI/VITA 1.1-1997, 1997, verfügbar über http://www.vita.com

[Wur96] A. Wurz: *Private Mitteilung,* Mannheim, 1996

[Xil00] *The Programmable Logic Data Book,* Xilinx, CA, USA, 2000

[Zoz97] Ralf Zoz: *Eine Hochsprachen-Programmierumgebung für FPGA-Prozessoren,* Dissertation, Ruprecht-Karls-Universität Heidelberg, 1997

Danksagung

An dieser Stelle möchte ich denjenigen danken, die dazu beigetragen haben, daß ich meine Promotion habe durchführen können.

Bei Herrn Prof. Dr. Männer möchte ich mich dafür bedanken, daß ich meine Promotion an seinem Lehrstuhl durchführen konnte. Aufgrund der vielfältigen Aktivitäten an diesem Lehrstuhl war es mir möglich, durch die aktive Mitarbeit an Projekten auch außerhalb meines Spezialgebiets meinen Horizont zu erweitern.

Einen wesentlichen Beitrag zum Gelingen meiner Arbeit haben Harald Simmler und Andreas Kugel geleistet, die sich während der Promotionsphase mit mir das Arbeitszimmer teilten. Für diese Unterstützung bedanke ich mich. Matthias Sessler danke ich für die ständige Diskussionsbereitschaft in Zusammenhang mit Fragen zur Physik, zu ATLAS und zu den Triggeralgorithmen.

Insgesamt möchte ich mich bei den Mitgliedern meiner FPGA-Arbeitsgruppe für die gute Zusammenarbeit bedanken. Zu erwähnen ist das gute Arbeitsklima in der Gruppe aber auch innerhalb des gesamten Lehrstuhls, das sich sehr positiv auf die Durchführung meiner Arbeit ausgewirkt hat.

Bei den Geschäftsführern der Firma Silicon Software Dr. Klaus-Henning Noffz und Dr. Ralf Lay bedanke ich mich für den geleisteten Know-How-Transfer und die immer vorhandene Gesprächsbereitschaft.

Andrea Seeger danke ich für die vorbildliche logistische Betreuung.

Für die sehr gute Unterstützung möchte ich mich bei meiner Familie bedanken, die mir in vielerlei Hinsicht immer einen wertvollen Rückhalt gegeben hat.

Bedanken möchte ich mich abschließend bei allen, die mich in positiver Art und Weise bei meiner Arbeit unterstützt haben, die ich aber an dieser Stelle nicht namentlich nennen kann.