

Paralleles Rechnen für die Bildverarbeitung auf Cluster-Rechnern und in FPGA

(Prof. Dr. Dietmar Fey)

Algorithmenstudien anhand zellulärer Automaten

Zur Identifikation von Objekten in aus Graustufenbildern erzeugten Binärbildern sollte als Ausgangspunkt die sog. Blob-Analyse dienen. Bei der Blob-Analyse wird das zu den Koordinaten-Achsen parallel ausgerichtete umgebende Rechteck eines Objektes, die sog. ROI (region of interest) bestimmt. Von diesem ausgehend können dann z.B. Flächen bzw. Schwerpunkte des Objektes berechnet werden.

Es geht darum, einen geeigneten parallelen Algorithmus zu entwickeln, der pixel-lokal angewandt für mehrere im Bild befindliche Objekte die entsprechenden Blobs findet. Für diese Aufgabenstellung bietet sich ein zellulärer Automat an. Ein zellulärer Automat ist ein Verband von einzelnen Zellen, die z.B. in einem Feld angeordnet sind. Jede Zelle hat einen bestimmten Zustand, der durch einen Ganzzahlwert charakterisiert ist. Der Zelle entspricht in unserem Falle der Anwendung in der industriellen Bildverarbeitung ein Bildpunkt. Über mehrere Zeitschritte hinweg wird in jedem Zeitschritt für jede Zelle ein neuer Zustand errechnet. Dieser neue Zustand ergibt sich aus der Summe des eigenen Zustandes und der Zustände einiger Zellen in unmittelbarer Nachbarschaft. Diese Summe wird von einem festen Schwellwert abgezogen, was den neuen Zustand der Zelle ergibt.

Mit Hilfe eines frei verfügbaren Simulators für zelluläre Automaten werden mit diesem Simulator mitgelieferte Beispiele analysiert, die unserer gewünschten Anwendung recht nahe kommen. Das Regelsystem dieser Automaten wird in Simulationsstudien analysiert und schließlich für unseren Zweck der Auffindung des Blobs geeignet modifiziert.

Die gefundenen Regeln werden anschließend für Soft-IP-Prozessoren implementiert, welche für den Einsatz in einem FPGA geeignet sind. Für die Entwicklung wird ein Xilinx Spartan-3 Starter Kit Board (XC3S1000) von Digilent verwendet. Es enthält 1Millionen Gatter, **216 Kbits** interner BRAM (Block Random Access Memory) und 1 MB externer SRAM sowie verschiedene Schnittstellen (z.B. RS-232, VGA).

Bei den angesprochenen Soft-IPs handelt es sich um den von dem FPGA-Hersteller Xilinx angebotenen 32-Bit-Risc-Prozessor MicroBlaze und den 8-Bit-Mikrocontroller PicoBlaze, die speziell für die Entwicklung in FPGAs entwickelt wurden. Der VHDL-Quellcode des MicroBlaze ist zwar verschlüsselt, aber er lässt sich entsprechend konfigurieren, um bestimmte Anforderungen zu erfüllen, z.B. erlaubt er das Anlegen von Cache-Speichern sowohl für Instruktionen als auch für Daten. Der VHDL-Quellcode des PicoBlaze ist dagegen frei verfügbar. Der Hauptvorteil dieser beiden Soft-IPs besteht darin, dass sie mit C bzw. mit Assembler programmiert werden können.

Die folgende Abbildung 1 zeigt die mit einem C-Programm für den MicroBlaze im FPGA erzeugte Umsetzung der Blobanalyse. Dargestellt sind auf dem Bildschirm erzeugt Ausgabebilder. Man sieht, dass die Bereiche des Bildes bestimmt werden, in denen sich Objekte befinden. Dazu wird für jedes Objekt das kleinste Rechteck (Blob) bestimmt, das das Objekt vollständig überdeckt. Das begrenzt im weiteren Verlauf den Aufwand der Berechnung, da nur die im Bereich der gefundenen Blobs liegenden Pixel betrachtet werden müssen. Jeder Blob kann durch die Position seiner linken oberen und der rechten unteren Ecke eindeutig charakterisiert werden. Außerdem wird jedem Blob eine eindeutige Nummer zugeordnet.

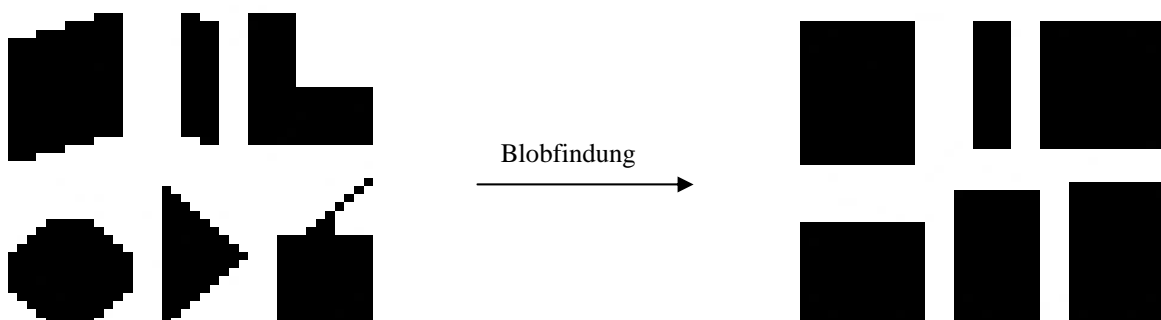


Bild 1: Im FPGA erzeugte Blobs für Objekte auf der Basis in Soft-IPs (MicroBlaze und PicoBlaze) umgesetzter zellulärer Algorithmen

Neben der Blob-Analyse werden weitere auf der Basis lokaler und damit der Vorgehensweise in zellulären Automaten ähnlichen Operatoren für die Bildvorverarbeitung umgesetzt, z.B. für die Kanten- und Eckenerkennung (s. die FPGA-Ausgaben in Bild 2 und 3). Anders ist die Situation für die Bestimmung des Schwerpunktes. Dieser wird nicht mit einem zellularen Ansatz berechnet, sondern mit „klassischen“ Methoden, d.h. aus der Summe aller aufsummierten X- und Y-Koordinaten dividiert durch die Anzahl der Objektpixel. Dies erfolgt im Anschluss an die Blob-Analyse, wobei die Blob-Analyse die Anzahl der dabei zu untersuchenden Pixel einschränkt und sich damit selbst für eine serielle Verarbeitung eine Verbesserung gegenüber dem trivialen Ansatz der pixelweisen Untersuchung aller Bildpixel nacheinander ergibt.

Um ferner die Position eines Objektes eindeutig zu bestimmen, reicht der Schwerpunkt nicht aus. Als eine Möglichkeit wurde deswegen zusätzlich der Winkel zwischen der kürzesten Kante des Objektes und der x-Achse bestimmt. Ausgangspunkt ist dafür zunächst die auf dem Originalbild durchgeführte Kantenerkennung und die danach erfolgte Eckenerkennung. Abschließend werden aus den gefundenen Eckpunkten jedes Objektes die kürzesten Kanten berechnet. Daraus lässt sich der Winkel zur x-Achse bestimmen.

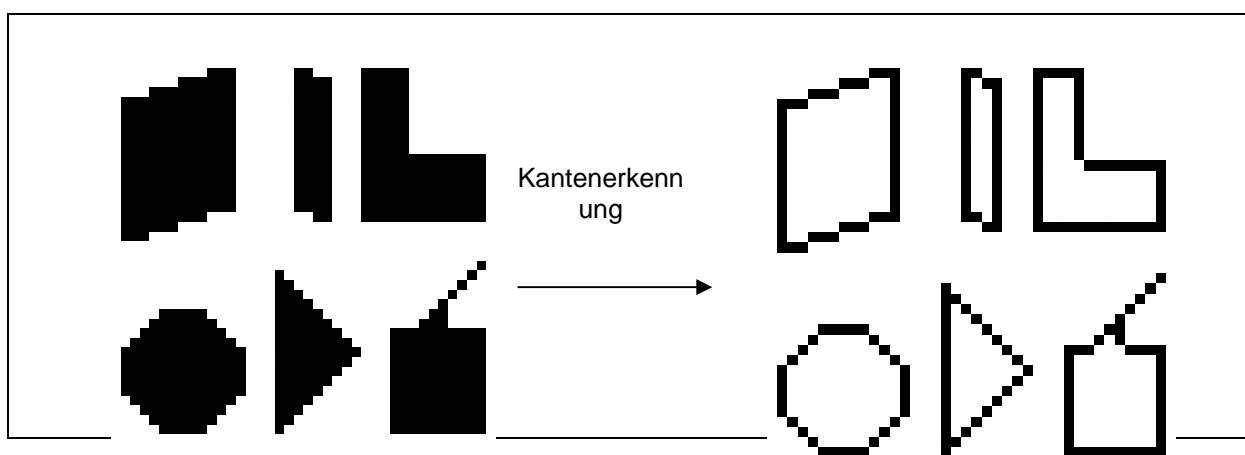


Bild 2: Im FPGA erzeugte Kantenerkennung für Objekte auf der Basis in Soft-IPs (MicroBlaze und PicoBlaze) umgesetzter lokaler Operatoren

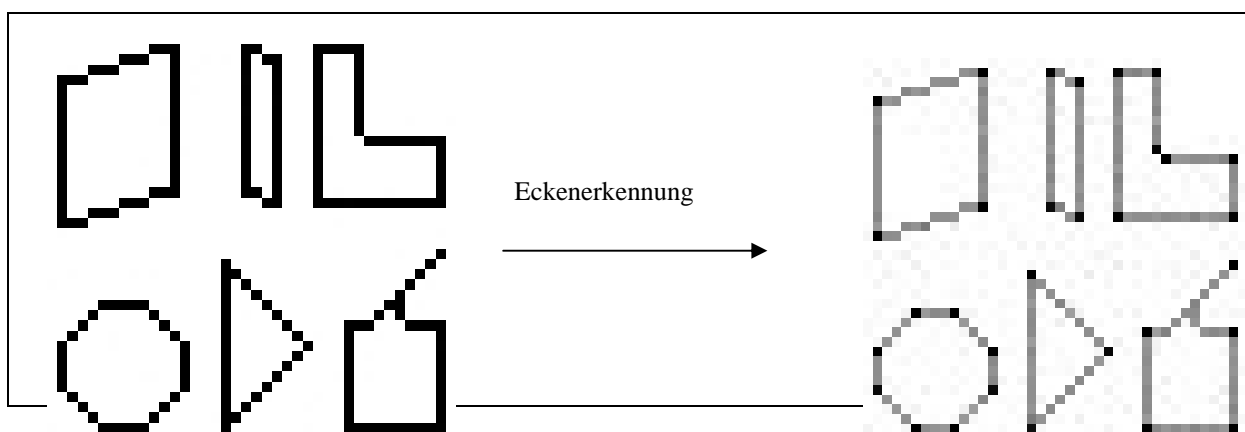


Bild 3: Im FPGA erzeugte Eckenerkennung für Objekte auf der Basis in Soft-IPs (MicroBlaze und PicoBlaze) umgesetzter lokaler Operatoren

Dieser Ansatz ist aber im Vergleich mit der Bestimmung der Momente (s. später) weniger robust, d.h. durch einfache Pixelfehler lässt sich die Anzahl und Position der gefundenen Ecken stark verändern. Alle diese eben aufgezählten Operatoren entsprechen im Prinzip dem Stand der Technik. Sie dienen jedoch als Ausgangsbasis für die behandelte innovative Parallelverarbeitung. Auch diese lassen sich zunächst mit den Soft-IP-Prozessoren MicroBlaze und PicoBlaze durchführen. Wie bereits oben

erwähnt geschieht dies, um in einem Top-Down-Ansatz die Architektur eines 1-Bit-Prozessor-elementes und eines Feldes aus diesen 1-Bit-Prozessorelement zu finden.

Entwurf geeigneter 1-Bit-Prozessorarchitekturen

Um eine schnelle Entwicklung des 1-Bit-Prozessors zu gewährleisten, wird auf den 8-Bit-Mikrocontroller PicoBlaze zurückgegriffen. Da dessen VHDL-Quellcode frei zugänglich ist, kann man die Teile entfernen, die nicht für den 1-Bit-Prozessor benötigt werden.

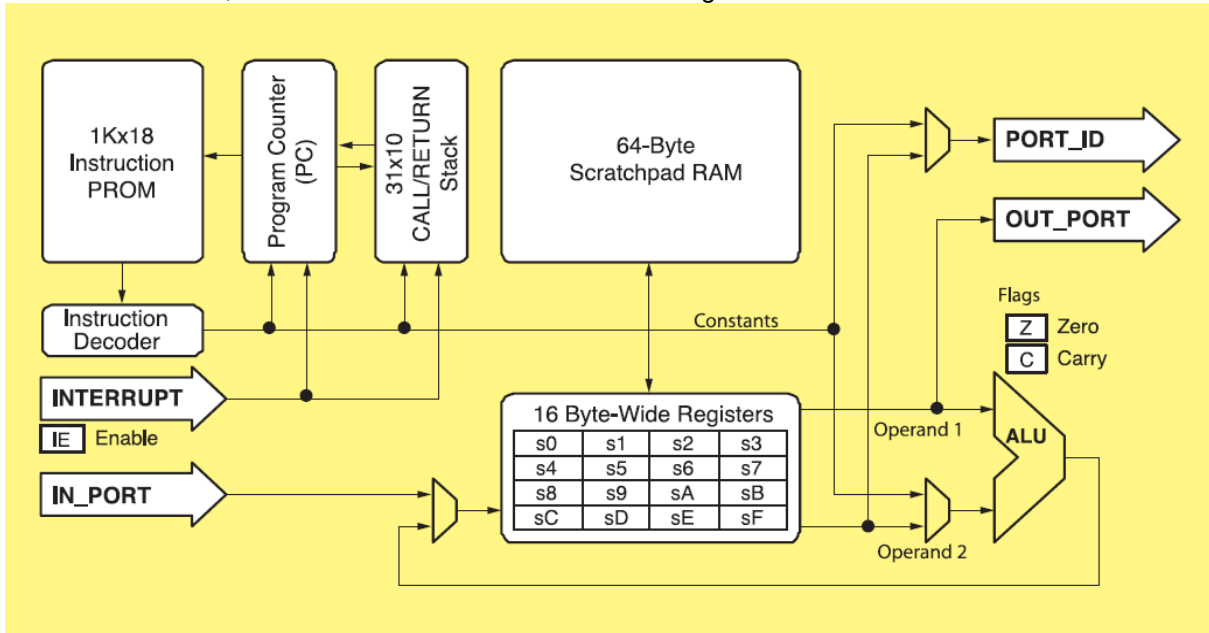


Bild 4: Aufbau des 8-Bit-Mikrocontroller PicoBlaze

Da der PicoBlaze mit 8-Bit rechnet, aber nur 1-Bit benötigt wird, kann man hier viel Logik einsparen. Außerdem ist der 64-Byte große Scratchpad Speicher unnötig, da die 16 Register vollkommen ausreichend sind. Außerdem wird später im Prozessorfeld eine zentrale Steuereinheit existieren, so dass die einzelnen Prozessorelemente keinen separate Programmzähler und keinen CALL/RETURN Stack benötigen. Insgesamt lässt sich der PicoBlaze so auf weniger als ein Zehntel seiner ursprünglichen Größe schrumpfen.

Entwurf FPGA-kompatibles SIMD-Prozessorfeld

Die Bildverarbeitung wird durch die Verwendung mehrerer MicroBlazes parallelisiert. Das Bild wird dabei schichtweise auf die einzelnen MicroBlazes verteilt. Einer von ihnen fungiert als Master, die anderen als Slaves. Alle bekommen einen Teil des Bildes zugeordnet und der Master ist außerdem noch für die Auswertung und Ausgabe der Ergebnisse zuständig. Zusätzlich besitzt jeder MicroBlaze noch einen Rand, der die Bildinformationen der Nachbarn enthält, die für die Berechnungen

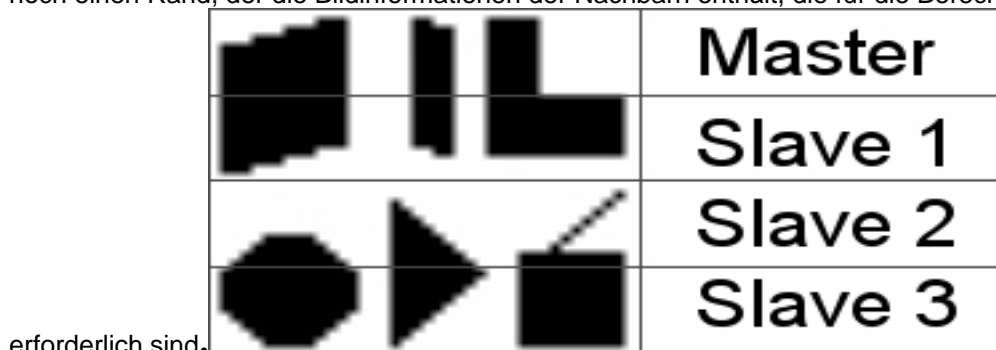


Bild 5: Aufteilung des Bildes auf 4 MicroBlazes

Alle MicroBlazes greifen gemeinsam über einen OPB (Open Peripheral Bus) auf den externen Speicher zu, in dem das Bild gespeichert ist. Untereinander kommunizieren die MicroBlazes mittels

FSL (Fast Simplex Link) Bussen. Diese ermöglichen eine schnelle unidirektionale Datenübermittlung innerhalb von zwei Takten. Die MicroBlazes sind miteinander in einer Ringstruktur verknüpft. Somit können auch der Master und der letzte Slave kommunizieren. Das ist z.B. erforderlich um die MicroBlazes zu synchronisieren.

Während der Bildverarbeitung führt jede MicroBlaze die in Kapitel 3 beschriebenen Operationen lokal auf seinem Bildteil aus. Nach jedem Berechnungsschritt werden dann, falls erforderlich, Informationen zwischen den MicroBlazes ausgetauscht.

Nach den Bildvorverarbeitungsoperationen (Erosion, Dilatation) werden nur die geänderten Pixel an den Rändern ausgetauscht. Bei der Blobanalyse bestimmt jeder MicroBlaze die Blobs auf seinem Bildausschnitt selbständig. Danach hat jeder MicroBlaze seine eigene Nummerierung der Blobs. Diese wird anschließend korrigiert so dass sie mit der bei einer seriellen Verarbeitung auftretenden Nummerierung übereinstimmt.

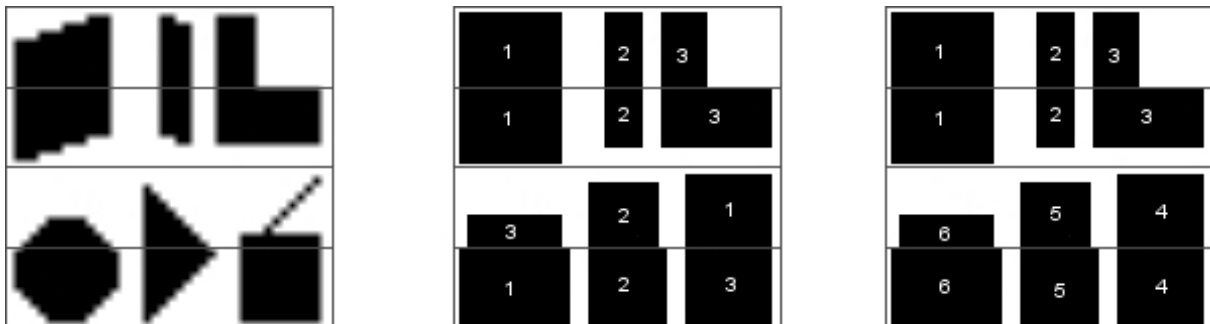


Bild 6: Blobanalyse bei Parallelverarbeitung durch 4 MicroBlazes

Aufgrund der genaueren Blobanalyse lässt sich durch die Parallelisierung auch Aufwand sparen, da die einzelnen Blobs der 4 MicroBlazes zusammen weniger Pixel umfassen als die Blobs die bei der seriellen Abarbeitung entstehen (Vergleich Bild 1).

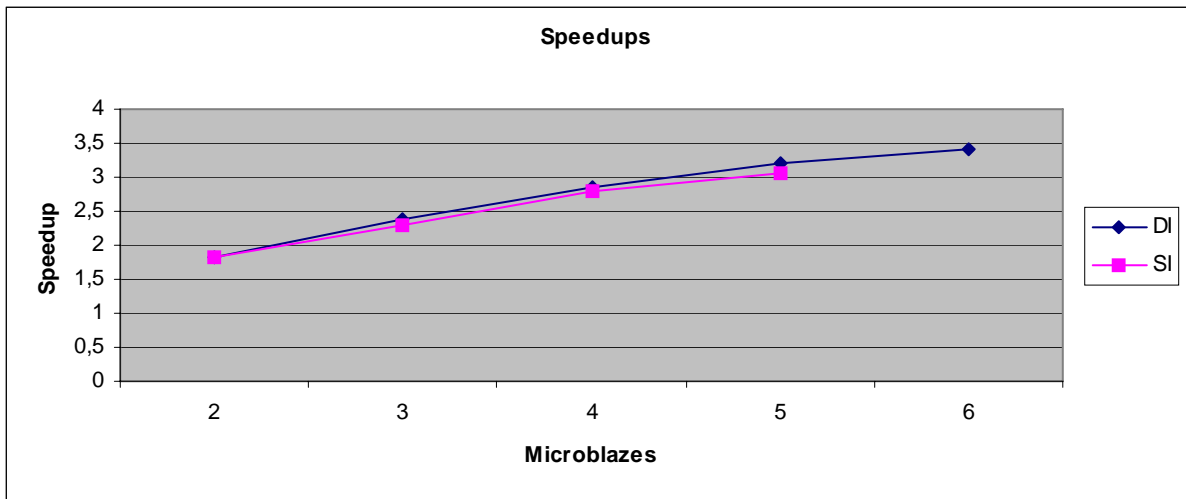
Anschließend bestimmen alle MicroBlazes die Schwerpunkte und Pixelzahlen ihrer Blobs. Die Ergebnisse werden auf dem Master gesammelt. Als letztes findet die Eckenerkennung statt. Diese kann jeder MicroBlaze wieder selbständig ausführen. Die sich ergebenden Ecken werden auf dem Master gesammelt, der dann die kürzesten Kanten bestimmt.

Der beschriebene Ansatz stellt ein klassisches SIMD Verfahren dar.

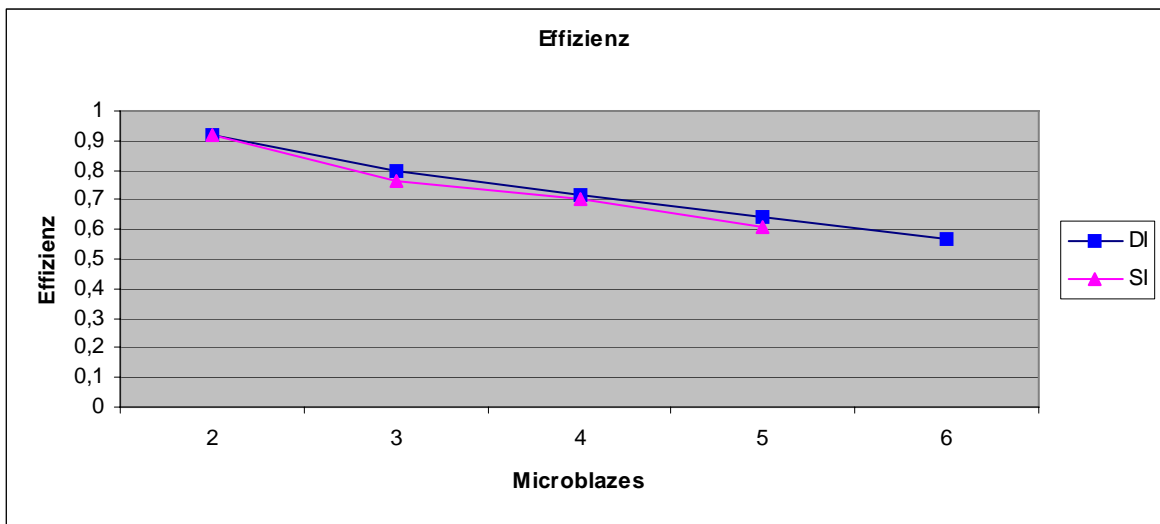
Dieses wird nun auf verschiedene Weisen umgesetzt. Zuerst wird zwischen „Distributed Instruction“ (DI) und „Shared Instruction“ (SI) unterschieden. Bei DI besitzt jeder Slave seinen eigenen BRAM mit Instruktionen auf den er mittels LMB (Local Memory Bus) zugreift. Bei SI dagegen werden die Instruktion der Slaves in einem gemeinsamen BRAM gespeichert. Auf diesen greifen alle Slaves mittels OPB zu. Um den Zugriff nicht zum Flaschenhals werden zu lassen, besitzt jeder Slave einen Instruktionscache. Trotzdem erreichen die Speed-up Werte von SI nicht die von DI.

Des weiteren werden verschiedene Ansätze der Verteilung der Rechenlast untersucht. Bei DI werden das Bild und die Verarbeitungsoperationen gleichmäßig auf alle MicroBlazes verteilt. Das lässt sich natürlich auch anders machen, z.B. in einer Pipeline. Dabei führen nicht alle MicroBlazes alle Bildverarbeitungsoperationen durch. Eine Aufteilung aus drei MicroBlazes sähe beispielsweise so aus, das der erste MicroBlaze das Einlesen des Bildes und die Vorverarbeitung, der zweite MicroBlaze die Blobanalyse und der dritte MicroBlaze die Objekt- und Positionserkennung übernimmt. Eine gleichmäßige Lastverteilung lässt sich aber nur schwer realisieren, da die Bildverarbeitungsoperationen unterschiedlich aufwendig sind. Deswegen wird der obige Ansatz mit dem Pipelining vermischt. D.h. ein MicroBlaze liest das Bild vom externen Speicher ein und sendet es an eine Gruppe von MicroBlazes. Diese übernehmen dann entweder alle restlichen Bildverarbeitungsoperationen (1_X) oder es schließt sich ein weiterer MicroBlaze an (1_X_1), der die Lagebestimmung ausführt.

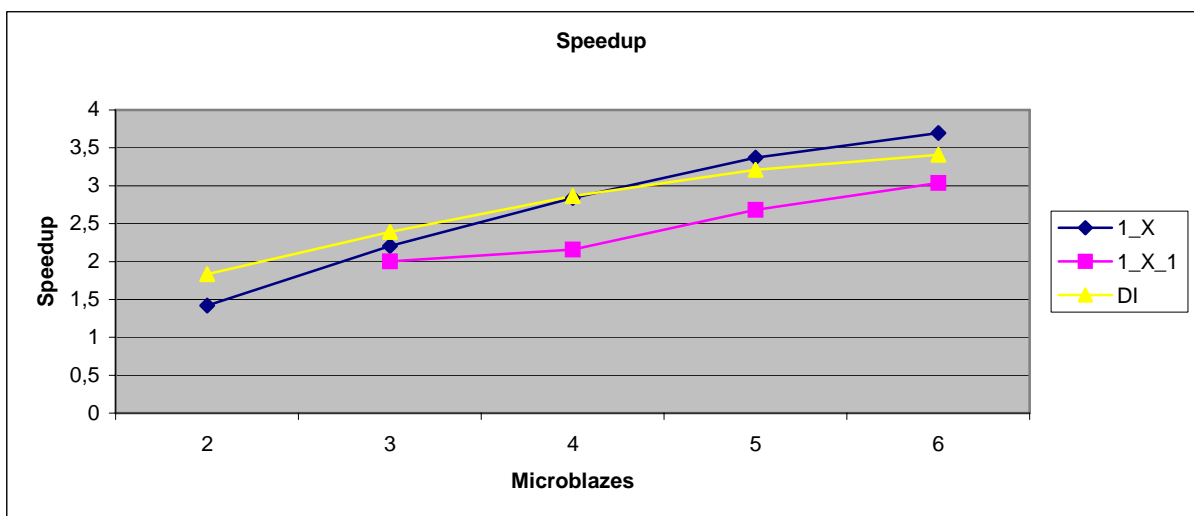
Das folgende Diagramm stellt die Speed-up Werte für verschiedene Anzahlen von MicroBlazes dar. Dabei wird zwischen Distributed Instruction (DI) und Shared Instruction (SI) unterschieden.



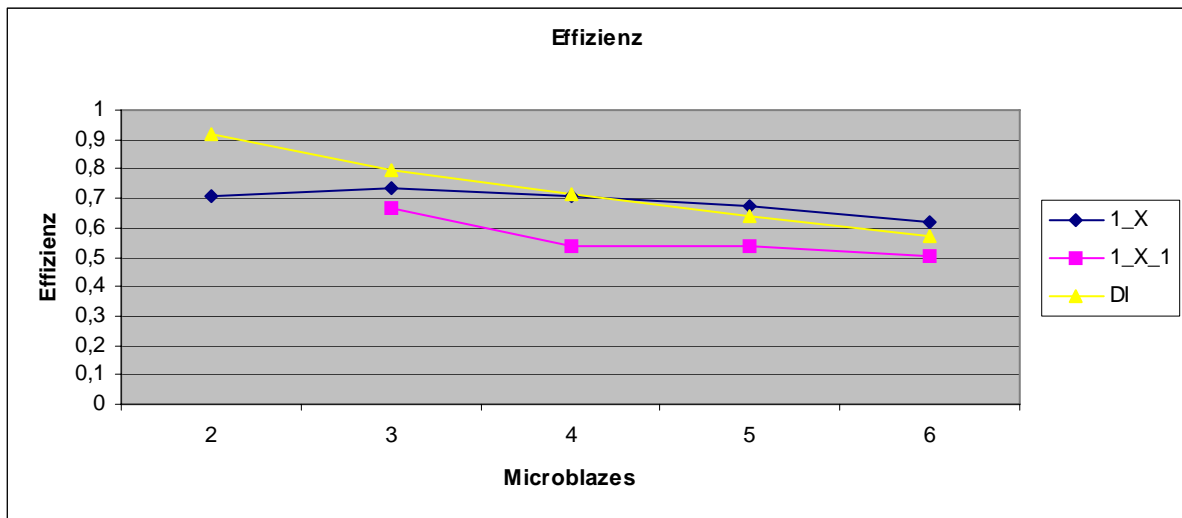
Die Speed-up Werte steigen mit zunehmender Zahl von MicroBlazes zwar an, dafür sinkt die Effizienz, wie man folgendem Diagramm entnehmen kann.



Die Speed-up Werte für das Pipelining sehen ähnlich aus.



In diesem Fall profitiert das Pipelining davon, dass der erste MicroBlaze nur für das Lesen des Bildes aus dem externen Speicher zuständig ist.



Zusammenfassend lässt sich somit Folgendes festhalten.

In den bisher erfolgten Arbeitsschritten wurde vorgestellt, wie man Bildverarbeitung parallel auf einem FPGA mit MicroBlaze- und PicoBlaze-Soft-IPs durchführen kann. Dabei wurden verschiedene Ansätze getestet: MIMD, SIMD und Pipelining. Zwar wurden befriedigende Speedup Werte erreicht, aber nur auf unrealistisch kleinen Bildern wird die geforderte Antwortzeit eingehalten. Um diese Bedingung doch zu erfüllen, könnte man leistungsfähigere FPGA Board verwenden. Damit kann die Rechenleistung durch zusätzliche Soft-IPs erhöhen.

Wir wählen jedoch einen anderen Weg, der bereits bei geringer dimensionierten FPGAs bessere Leistungen liefern wird. Dieser beruht darauf, die universell einsetzbaren Soft-IPs durch speziell auf die benötigten Bildverarbeitungsoperationen zugeschnittene Prozessoren zu ersetzen. Diese würden erheblich kleiner als die Micro- und PicoBlazes ausfallen und somit eine höhere Anzahl von Prozessoren pro Board erlauben. Als Ausgangspunkt um solche einfacheren Prozessoren zu erzielen, wählen wir die frei verfügbare VHDL-Beschreibung des PicoBlaze. Diese werden von vielen für die 8-Bit-Architektur des PicoBlaze notwendigen Struktur-Elementen befreit; herauskommt eine VHDL-Beschreibung für eine 1-Bit-Architektur, die für weitere Erprobungen für den Einsatz für die Berechnung von Momenten erprobt werden.