Abstract (deutsch)

Die vorliegende Arbeit befasst sich mit der Untersuchung verschiedener Implementierungen der sogenannten selbst-organisierenden Karten, einem Algorithmus aus dem Bereich der künstlichen neuronalen Netze von Teuvo Kohonen. Der Algorithmus ist, wie sein natürliches Vorbild, inhärent parallel und eignet sich somit für eine Abbildung in paralleler Hardware. Dies ist in vielen Anwendungsgebieten sinnvoll, da hier zum Teil sowohl harte Echtzeitanforderungen als auch Einschränkungen bzgl. des Energiebudgets existieren. Ein Beispiel hierfür ist die Analyse hyperspektraler Aufnahmen von Planetenoberflächen an Bord eines Satelliten oder einer Drohne. Die Untersuchungen gliedern sich in verschiedene Bereiche:

Im Bereich der einzelnen Rechenelemente werden in der Regel hardwarespezifische Anpassungen wie etwa die Diskretisierung der Wertebereiche vorgenommen. Dadurch können an Stelle der in Software gewöhnlich verwendeten Gleitkommaberechnungen einfachere Ganzzahlberechnungen verwandt werden, was den Ressourcenbedarf der einzelnen Recheneinheiten drastisch reduziert. Gleichzeitig wird auch die Funktionalität des Algorithmus verändert: primär reduziert sich die Präzision der Zahldarstellung von Ein- und Ausgangsdaten, sekundäre Effekte entstehen durch die Akkumulation von Veränderungen innerhalb der Berechnungen. Um diese Effekte beurteilen zu können wird im Rahmen dieser Arbeit ein neues Maß erarbeitet, das eine bessere Vergleichbarkeit verschiedener Implementierungen ermöglicht. Dieses wird auf einige Beispieldatensätze angewandt um verschiedene hardwarespezifische Anpassungen hinsichtlich ihrer Auswirkungen zu bewerten.

Auf Architekturebene stellt sich vor Allem die Frage nach der parallelen Implementierung des Algorithmus, denn eine Analyse zeigt effektiv zwei völlig unabhängige Freiheitsgrade, entlang derer parallelisiert werden kann. Um für eine bestimmte Anwendung die richtige Implementierung zu finden, wird auf Basis von exemplarischen Messungen ein Modell für den Ressourcenbedarf (Leistungsaufnahme, Latenz, Fläche) aufgestellt, bei dem unter Anderem Parameter wie die Anzahl der implementierten Rechenelemente und die Art der Parallelisierung frei wählbar sind. Für das oben beschriebene Anwendungsbeispiel wird dann die Pareto-Menge berechnet, deren Mitglieder dann mit Hilfe der sogenannten Ressourceneffizienz bewertet werden. So kann für jede Anwendung eine optimale Implementierung gefunden werden.

Um die so gefundene Implementierung testen zu können werden zunächst zwei verschiedene Werkzeuge vorgestellt, die komfortable Analysen in einem teilautomatisierten Umfeld ermöglicht. Zunächst wird HiLDE vorgestellt, ein Werkzeug das die Kopplung zwischen einer Softwaresimulation (z.B. Matlab/Simulink oder ModelSim) und einer Hardwareimplementierung ermöglicht. Bei den mit HiLDE ermöglichten Tests wird ausschließlich die funktionale Korrektheit einer

Implementierung getestet, Effekte die sich auf den kritischen Pfad beziehen werden außer Acht gelassen. Um auch diese Effekte berücksichtigen zu können wurde HiLDEGART entwickelt, ein Werkzeug das die Beobachtung und Parametrierung einer Implementierung in Echtzeit erlaubt. Da für beide Werkzeuge zahlreiche spezifisch angepasste Hardwarekomponenten benötigt werden, wurde mit der vMAGIC Bibliothek ein Werkzeug für die automatische Quellcodegenerierung für VHDL geschaffen.

Um einen konkreten Vergleich zwischen Hardware- und Softwareimplementierungen herbeizuführen, wurde mit Hilfe dieser Werkzeuge eine prototypische Implementierung erzeugt und vermessen. Hierbei zeigt sich, dass auch für die vergleichsweise kleine und nicht optimal implementierte Variante eine Beschleunigung gegenüber aktuellen Mehrprozessorsystemen erzielt werden kann. Eine Extrapolation für zukünftige Technologien verdeutlicht die prinzipiellen Vorteile der gewählten Vorgehensweise.