UNIVERSITATEA POLITEHNICA BUCUREŞTI
Facultatea de Automatică şi Calculatoare
Departamentul Calculatoare

Consideraţii Privind Îmbunătăţirea Performanţei Programelor – Optimizări de Buclă pentru Sisteme DSP

Îndrumător Ştiinţific: Prof. Eng. PHD Nicolae Ţăpuş Doctorand: Anca Burlacu-Zane

ABSTRACT

Teza de doctorat propune dezvoltarea unei platforme reutilizabile de compilator oferind un suport complet pentru auto-vectorizare şi auto-paralelizare, urmărind două direcţii: extinderea unei platforme existente folosită pentru arhitectura StarCore şi dezvoltarea unei noi platforme de dezvoltare pentru multiple arhitecturi. Mai întâi sunt prezentate caracteristicile aplicaţiilor şi ale procesoarelor DSP precum şi modul în care influenţează rezultatele diferitelor tehnici de optimizare, iar acest lucru serveşte următoarelor 2 scopuri: se demonstrează modul în care arhitectura şi tipul de aplicaţii influenţează direcţia aleasă de compilator şi se introduc elementele care trebuie modelate de către un compilator reconfigurabil – pentru obţinerea codului maşină ca şi pentru controlul prin euristici sau chiar pentru activarea/dezactivarea diferitelor optimizări. Apoi se defineşte noua platformă folosindu-se feedback-ul de la cea veche şi se introduc o serie de tehnici de optimizare şi euristici. Principala contribuţie constă într-o serie algoritmi. Majoritatea optimizărilor propuse sunt de nivel înalt, una dintre ele fiind de nivel jos iar domeniile atinse sunt variate. Primul algoritm propune o tehnică de propagare şi reutilizare a cuvintelor cheie după schimbarea scopului cu aplicare directă pentru cuvântul C `restrict` şi analiza de alias. Urmează o serie de algoritmi de buclă, începând cu tehnici de euristică pentru estimarea la nivel înalt a gradului de utilizare a registrelor ca şi pentru folosirea acestei informaţii în diferite transformări precum mutarea de cod invariant în afara buclelor sau vectorizarea acceselor la memorie. O altă euristică propusă este planificatorul la nivel înalt prin care se încearcă estimarea efectului diferitelor variante ale anumitor transformări asupra codului final pentru a se alege cea mai bună soluţie. În vederea vectorizării se introduce şi tehnica transformării acceselor la memorie de tip pointer în accese la memorie de tip vectorial – pentru accese simple sau multi-dimensionale ca şi pentru bucle cu una sau mai multe ieşiri, ce ajută analiza de dependenţe de date. Analiza de inducţie a fost şi ea considerată în 2 variante, şi sunt prezentaţi trei noi algoritmi pentru detectarea modului de adresare modulo, pentru detectarea şi generarea de bucle hardware şi pentru detectarea aliniamentului acceselor la memorie în vederea vectorizării. Un algoritm simplu de analiză de dependenţe de date şi vectorizare la nivel de bloc este de asemenea introdus. În final se prezintă o tehnică de exploatare a pipeline-ului, prin combinarea planificării instrucţiunilor şi a predicării salturilor condiţionate. Toţi algoritmii prezentaţi sunt însoţiţi de rezultate de performanţă pentru diferite suite de teste şi reprezintă o parte importantă din suportul necesar unui compilator reconfigurabil pentru auto-vectorizare şi auto-paralelizare.

Considerations Regarding Improving Program Performance – Compiler Loop Optimizations for DSP Systems

Thesis Supervisor: Prof. Eng. PHD Nicolae Ţăpuş Author: Anca Burlacu-Zane

ABSTRACT

The PHD thesis proposes the development of a reusable compiler with full support for implementing auto-vectorization and paralelization. Two directions are considered – the extension of an already existing compiler for the StarCore architecture and the development of a new multi-architecture framework. The characteristics of DSP applications and processors and the way they influence different compiler techniques help introduce the processor characteristics which must be modeled in order to develop a reconfigurable compiler for both obtaining machine code and controlling optimizations through different heuristics. The new compiler framework is designed using feedback from the limitations of the older one. The main contribution of the thesis consists in a series of algorithms, mostly applied during high level optimizations, with one considering low level optimizations, covering a wide range of techniques. The first algorithm touches alias analysis through a technique of alias property propagation and reuse after variable scope change, with the immediate application for the 'restrict' C keyword. A series of loop algorithms were developed next, consisting in both heuristics and transformations. The first heuristics considers register pressure estimation during high level optimizations. Two techniques use it to control loop invariant code motion and vector memory accesses. The second heuristic was named the high level scheduler. It tries to estimate the effect of different transformations on the final generated machine code and helps in choosing the best solution. Pointer to array is another technique which transforms memory accesses through pointers into array memory accesses, for both multi and single dimension vectors and considering loops with multiple exists. This helps data dependence analysis and in the end vectorization. Induction analysis was also considered – with the implementation of two different techniques. It allowed the development of three new algorithms: modulo addressing generation, hardware loop and loop skip generation as well as memory access alignment detection (also used by vectorization). A simple data dependence algorithm as well as a simple basic block vectorization technique were also developed. The DSP pipeline is also considered through the if-conversion in conjunction with instruction scheduling algorithm. All techniques are accompanied by performance reports on different benchmarks. All work and algorithms helped develop a tool with an important part of the support needed for a fully retargetable compiler aiming auto-vectorization and auto-parallelization.