Instruction Scheduling

Serial Assembly Optimizer for a VLIW Processor

Submitted by
Usman Sajeel Haider
(992CS137)

Supervised by
Mr. Jehanzeb Ahmed

A report submitted to the department of Computer Science,
Bahria Institute of Management and Computer Sciences, Islamabad.
In partial fulfillment of requirement for the degree of BCS

Department of Computer Sciences,
Bahria Institute of Management and Computer Sciences, Islamabad.
University of Peshawar, Peshawar.

Abstract

We present a *Serial Assemble Optimizer (SOA)* for the Media Engine (ME-2), being developed in Communications Enabling Technology (CET). The SOA works in collaboration with the already coded module that takes the C code and converts it into the serial assembly, referred hereon as Front-end Serial Assembly Generator (FSAG). CET has only implemented a minimal functionality prototype of FSAG, whereas in actuality the GNU C compiler is being used as the serial code generator on the backend. Our project scope is the Serial Assembly Optimizer (SAO), which takes the serial code generated by the FSAG and makes use of advanced optimization techniques to generate a parallel, and optimized code.

Table of Contents

Abstract 3

Acknowledgements 4

Table Of Contents 5

Section one *9*

Introduction 9

- I- Hand Optimization Techniques are not Scalable 10
- II- Hand-Optimized Code is not Portable 10
- III- Related Work 11

Chapter 1 12

Introduction To The Dissertation 12

- 1.1) Purpose of Dissertation 12
- 1.2) Scope of Dissertation 12
- 1.3) Layout of the Dissertation 13

Chapter 2 14

Project Description 14

- 2.1) Description 14
- 2.2) Features 15
  - 2.2.1) Modular Approach 15
  - 2.2.2) Front-end Serial Assembly Generator 15
  - 2.2.3) Serial Assembly Optimizer 15
- 2.3) Life Cycle Model 16
  - 2.3.1) PHASES 16
  - 2.3.2) FSAG 17
  - 2.3.3) SAO 17

Section two *18*

Background Knowledge 18

Chapter 3 19

ME-2 Architecture 19

- 3.1) Introduction 19
- 3.2) VLIW Architecture 20
  - 3.2.1) Terminology 22
  - 3.2.2) Principles Behind VLIWs 23
    - 3.2.2.1) Datapaths 23
    - 3.2.2.2) Pipelines 23
    - 3.2.2.3) Functional units 24
- 3.3) ME-2 Architecture 25
  - 3.3.1) Central Processing Unit (CPU) 26
  - 3.3.2) Internal Memory 27
  - 3.3.3) TXP/RXP Pipeline 27
- 3.4) Instruction Set Overview 27
  - 3.4.1) Instruction Types 27

3.4.1.1) AGU Instructions 28
3.4.1.2) DataPath Instructions 28
3.4.2) Registers 28
3.4.3) Addressing Modes 28
3.4.4) Use of data pointer registers 29
3.4.5) Execution Block Packet Composition 29
3.4.6) VLIW Grouping Restrictions 30
3.4.7) Looping restrictions 31
3.4.8) Conditional Execution 32
3.4.9) Latencies 32

Chapter 4 34

Optimizations Techniques 34

4.1) Parallelism in Programs 34
4.1.1) Coarse-grain parallelism 34
4.1.2) Fine-grain of Instruction Level Parallelism 35
4.2) Types of Optimizations 35
4.2.1) Classical Optimizations 36
4.2.2) Superscalar Optimizations 36
4.2.3) Multiprocessor Optimizations 37
4.3) Dependence Analysis 38
4.3.1) Resource Dependencies 38
4.3.2) Control Dependencies 38
4.3.3) Data Dependencies 39
4.3.4) Dependence Graphs 40
4.4) VLIW Compilers 41
4.5) Optimization Techniques For a VLIW Compilers 42
4.5.1) Trace Scheduling 42
4.5.2) Software Pipelining 42
4.5.3) Loop Unrolling 60
4.5.4) Register Scheduling 63

Section three 68

Project Specifications 72

Chapter 5 74

Analysis and Design Specification 74

5.1) Environmental Model 74
5.1.1) Statement of Purpose 74
5.1.2) Context Diagram 75
5.2) Data Flow Diagrams 76
5.3) Process Specification 78
5.3.1) Dependence Analysis (2.1) 78
5.3.2) Loop Unrolling (2.2) 78
5.3.3) Software Pipelining and Scheduling (2.3) 78
5.3.4) Low-Level Optimization (1.2.3) 79
5.3.5) Code Generation (1.2.4) 79
5.4) Use Case Diagram 80
5.4.1) Actor—Programmer 81
5.4.2) Use Case Description 81
5.4.2.1) Serial Assembly Optimization 81
5.5) Class Relationship Collaborators 82
5.6) Class Relationship Diagram 86
5.7) Sequence Diagrams 87
5.7.1) Serial Assembly Optimization 87
5.8) Class—Attributes, Methods 88

Section four — *97*

Results and Conclusion — **97**

I- Results — 97

II- Analysis and Conclusions — 100

III- Future Recommendations — 101

Appendix A — *103*

Serial Assembly Format — **103**

A-1) Register Allocation — 104

A-2) For Loops — 104

A-3) If else and Predicated Execution — 106

A-4) Auto Correlation — 106

Index — **112**

References — *114*