

Inhaltsverzeichnis

1	Projektgeschichte und Überblick	1
1.1	Das SAMPΛE Projekt	1
1.2	Die funktionale Sprache SAMPΛE	2
1.3	Die SAMPΛE Programmierumgebung	2
1.4	Die Ausführung von SAMPΛE Programmen	3
1.5	Übersetzer-Architektur und Maschinenmodelle	4
1.6	Aufbau dieser Arbeit	6
2	Einführung	9
2.1	MS - Eine einfache funktionale Sprache	9
2.2	Verzögerte Auswertung und Striktheitsanalyse	14
2.3	Eigenschaften moderner RISC Rechner	17
2.4	Prinzipielle Techniken der MS Implementierung	20
2.4.1	Polymorphismus	21
2.4.2	Automatische Speicherverwaltung und Struktur von Speicherzellen	23
2.4.3	Verzögerte Auswertung	29
2.4.4	Funktionen höherer Ordnung	40
3	Übersetzung von Sprachen mit verzögerter Auswertung	47
3.1	Phaseneinteilung des Übersetzters	47
3.2	Die abstrakte Registermaschine RM	49
3.3	Die abstrakte Syntax von MS	59
3.4	Suspensionsanalyse	62
3.5	RM-Übersetzer	70
3.5.1	RM-Codegeneratoren - die Basis der Übersetzerspezifikation	70
3.5.2	Codegenerator-Kombinatoren	72

3.5.3	Signatur der Übersetzerfunktionen	76
3.5.4	Übersetzen der globalen Definitionen	78
3.5.5	Übersetzen von Ausdrücken	82
3.5.6	Codegenerierung für verzögerte Ausdrücke	95
3.6	Optimierung der Repräsentation verzögerter Ausdrücke	99
3.6.1	Motivation	99
3.6.2	Die optimale Repräsentation von Suspensionen	104
3.6.3	Komplexität des Optimierungsproblems <i>Optimale Sus- pensions-Repräsentation</i>	106
3.6.4	Eine Näherungslösung des Optimierungsproblems	114
3.6.5	Suspensionsanalyse für Multisuspensionen	120
3.6.6	Anpassung des Übersetzers	124
3.6	Generierung von Objektcode am Beispiel des SPARC-Prozessors	99
3.6.1	Aktivieren des Garbage-Collectors und Optimierung von Speicherallokierungen	107
3.6.2	Verminderung der Anzahl der gleichzeitig lebendigen vir- tuellen Register	111
3.6.3	Registerzuteilung	113
4	Speicherverwaltung	157
4.1	Speicherfreigabe für Multisuspensionen	157
4.1.1	Einzelheiten des Two-Space-Copy-Collectors	157
4.1.2	Problematik der Speicherfreigabe für Multisuspensionen	158
4.1.3	Effiziente Speicherfreigabe für Multisuspensionen	159
4.2	Generationenbasierte Speicherbereinigung	166
4.2.1	Gründe und Voraussetzungen für die Anwendung genera- tionsbasierter Techniken	166
4.2.2	Speicherallokierungsverhalten bei Sprachen mit verzöger- ter Auswertung	168
4.2.3	Ein einfacher generationenbasierter Garbage-Collector für Sprachen mit verzögerter Auswertung	171

4.2.4	Evaluierung der Garbage-Collector-Algorithmen	182
5	Testen bei verzögerter Auswertung	189
5.1	Einführung in die Problematik und Diskussion bisheriger Techniken	189
5.2	Das Boxmodell	192
5.3	Implementierung des Testsystems	195
6	Zusammenfassung und Ausblick	199
6.1	Codegenerierung	199
6.2	Speicherverwaltung	200
6.3	Testen	201
	Literaturverzeichnis	202
A	Die funktionale Sprache SAMPAE	215
A.1	Lexikalische Konventionen	215
A.2	Operatoren	216
A.3	Ausdrücke	216
A.4	Definition von Werten	220
A.5	Module	223
A.6	Typen	224
A.6.1	Basistypen	226
A.6.2	Vordefinierte strukturierte Typen	228
A.6.3	Typsynonyme	232
A.6.4	Summentypen	233
A.7	Iterator-Ausdrücke	234
B	Standard SAMPAE-Funktionen	239
C	Befehle des SPARC-Prozessors	245