FORTSCHRITT-BERICHTE VDI

Dipl.-Ing. Walter Sebastian Mischo, München

Ein neuronaler Interpolationsspeicher für die lernende Regelung: Konzeptwahl und mikroelektronischer Entwurf

Reihe **9**: Elektronik Nr. **249**

Inhaltsverzeichnis

1 Einführung **1**

Teil I – Strukturfreie Modellbildung **6**

2 Das Grundproblem **7**
2.1 Vorbemerkungen . . . 7
2.2 Zur lernenden Regelung . . . 7
2.2.1 Begriffsbestimmung . . . 7
2.2.2 Eine mögliche Ausführung . . . 8
2.3 Lernende Modellbildung . . . 8
2.4 Mathematischer Rahmen . . . 10
2.4.1 Voraussetzungen . . . 10
2.4.2 Formulierung des Rahmens . . . 11
2.4.3 Zur Unterscheidung verschiedener Verfahren . . . 13
2.4.4 Neuronale oder mathematische Modellierung? . . . 15
2.5 Metamodellierung . . . 16

3 Die Modellierungskonzepte **19**
3.1 CMAC . . . 19
3.1.1 Vorstellung . . . 19
3.1.2 Kritik . . . 20
3.1.3 Beispiele . . . 20
3.2 Neuronale Netze mit Neuronen vom Sigmoid-Typ . . . 22
3.2.1 Vorstellung . . . 22
3.2.2 Kritik . . . 23
3.2.3 Beispiel . . . 24
3.3 Radialbasisfunktions-Netze . . . 25
3.3.1 Vorstellung . . . 25
3.3.2 Kritik . . . 26
3.3.3 Beispiele . . . 27
3.4 Adaptive Fuzzy-Systeme . . . 28
3.4.1 Vorstellung . . . 28
3.4.2 Kritik . . . 32

3.4.3 Beispiel . . . 33
3.5 Mathematisch orientierte interpolierende Speicher . . . 33
3.5.1 Vorstellung . . . 33
3.5.2 Kritik . . . 34
3.5.3 Beispiel . . . 35
3.6 Polynomapproximation . . . 35
3.6.1 Vorstellung . . . 35
3.6.2 Kritik . . . 37
3.6.3 Beispiel . . . 38
3.7 Sparse Distributed Memory . . . 40
3.7.1 Vorstellung . . . 40
3.7.2 Kritik . . . 42
3.8 Selbstorganisierende Karten . . . 44
3.8.1 Vorstellung . . . 44
3.8.2 Kritik . . . 46
3.9 Zusammenfassung und Vergleich . . . 46

4 Das CMAC-Konzept **51**
4.1 Zur Software-Realisierung . . . 52
4.2 Eingangsassoziation . . . 52
4.3 Hash-Coding . . . 53
4.4 Ausgangsberechnung . . . 56
4.5 Training . . . 57
4.5.1 Training durch Fehlerrückverteilung . . . 57
4.5.2 Training mit Zugriffszählern . . . 58

5 Vertiefungen zu CMAC **59**
5.1 Ein neues Assoziationsverfahren . . . 59
5.2 Kontinuierliche rezeptive Felder . . . 60
5.3 Gradientenrichtungen aus CMAC . . . 61
5.4 Weitere Modifikationen . . . 63
5.4.1 Zur Speicheradressierung . . . 63
5.4.2 Trainingsalgorithmen . . . 63
5.5 Zum Konvergenzverhalten von CMAC . . . 64

Teil II – Der Hardware-Entwurf **68**

6 Vorüberlegungen zum Hardwareentwurf HAMS **69**
6.1 Zur Namenswahl . . . 69
6.2 Grundsätzliche Vorbemerkungen . . . 69
6.3 Mikroelektronik für Neuronale Netze . . . 71
6.4 Technologie . . . 71

6.5 Entwurfsmethodik 73
6.6 Potential eines CMAC-Chips 74
6.6.1 Parallelisierung 74
6.6.2 Hash-Coding 75
6.6.3 Die Speichersuche 75
6.6.4 Trainingsalgorithmen 77
6.6.5 Erreichbare Taktfrequenz 77
6.6.6 Erwarteter Gewinn 77
6.7 Anbindung von HAMS an ein Trägersystem 78

7 Verfeinerungen des Grundkonzepts **80**
7.1 Wahl der Systemparameter 80
7.1.1 Verallgemeinerung 80
7.1.2 Eingangsdimension 80
7.1.3 Zahlenformat, Auflösung 81
7.1.4 Speichergröße 81
7.2 Verarbeitung der Eingangswerte 82
7.2.1 Assoziation 82
7.2.2 Hash-Coding 83
7.2.3 Speichersuche (*linear probing*) 83
7.3 Ausgangswertbildung und Training mit Zählern 83
7.3.1 Ausgangswertbildung 83
7.3.2 Training 83
7.4 Das Training durch Fehler-Rückverteilung 85
7.4.1 Bedingungen für die Vermeidung von Über-/Unterlauf 85
7.4.2 Konvergenzverhalten bei Sättigungsarithmetik 87
7.5 Das Konzept der virtuellen Zugriffseinheit 90
7.6 Zur Speicherverwaltung 91

8 Schnittstellen, Aufbau und Ansteuerung **92**
8.1 Die Schnittstellen 92
8.1.1 Die Modulschnittstelle 93
8.1.2 Die Speicherschnittstelle 94
8.1.3 Die Testschnittstelle 94
8.2 Prinzipieller Aufbau des Chips 94
8.3 Die Ansteuerung 97
8.3.1 Die Ansteuermodi des Chips 97
8.3.2 Das Statuswort 98

9 Zur Synthese von HAMS **99**
9.1 Prinzipien 99

9.2 Hardwarebeschreibung 101
9.3 Übersicht über die HAMS-Synthese 102
9.3.1 Die Entwicklung der VHDL-Spezifikation 102
9.3.2 Die Synthese 102
9.4 Der Entwurfsprozeß an einem Beispiel 106

10 Low-Level-Design und Lastsimulation **111**
10.1 Plazierung und Verdrahtung 111
10.2 Lastsimulation 113

11 Das System zur Echtzeitregelung mit HAMS **116**
11.1 Das HAMS-Modul 116
11.2 Übersicht des Systems 116
11.3 Der Hostprozessor 118
11.3.1 Zur Auswahl 118
11.3.2 Der Einsatz des Prozessors 119
11.3.3 Zusammenfassung der Nutzung der Anschlüsse am DSP 121
11.4 Der Buskoppelbaustein 121

Teil III – Zusammenfassung und Ausblick **123**

12 Zusammenfassung **124**

13 Ausblick **126**

Anhang **128**

A Zur Theorie **129**
A.1 Rechnung zu Abschnitt 5.2 129

B Zum Hardwareentwurf **131**
B.1 Das Hash-Coding-Verfahren 131
B.2 Beweis zu Abschnitt 7.4.1 133
B.3 Beweis zu Abschnitt 7.4.2 140
B.4 Zur Realisierung der Speicherverwaltung 142

C Anmerkungen zur Synthese **144**
C.1 VHDL-Spezifikation 144
C.1.1 Generierung inkorrekter Schaltungen 144
C.1.2 Geschwindigkeit des Simulators 144
C.2 VHDL-Synthese 145
C.2.1 Zum Verdrahtungslastmodell 145
C.2.2 Schaltungstechnisch korrekte aber nicht simulierbare Schaltungsteile . . 146

C.2.3 Taktbaum-Synthese 147
C.2.4 Probleme mit Padanschlüssen 148

D Zu Plazierung, Verdrahtung und Validierung **149**
D.1 Plazierung und Verdrahtung 149
D.1.1 Keine Busdarstellung 149
D.1.2 Schwacher Placement-Algorithmus 149
D.2 Validierung 150
D.3 Zusammenstellung der *physical design*-Prozedur 150

E Zum Systementwurf **155**
E.1 Steckerbelegung des HAMS-Moduls 155

F Zur Charakterisierung des HAMS-ASIC **156**
F.1 Ergebnisse erster Funktionstests 156
F.2 Spezifikation der Testsignale 156
F.3 Das Datenblatt 156
F.3.1 Die Modulschnittstelle 157
F.3.2 Die Speicherschnittstelle 158

Danksagungen **161**

Literaturverzeichnis **162**