// Weierstraß–Institut
für Angewandte Analysis und Stochastik

im Forschungsverbund Berlin e.V.

Report ISSN 0946 – 8838

MAGNUS – Mehrstufige Analyse großer Netzwerke und Systeme

Jürgen Borchardt, Friedrich Grund, Dietmar Horn, Manfred Uhle

Report No. 9

Berlin 1994

Weierstraß–Institut
für Angewandte Analysis und Stochastik

im Forschungsverbund Berlin e.V.

MAGNUS – Mehrstufige Analyse großer Netzwerke und Systeme

Jürgen Borchardt, Friedrich Grund, Dietmar Horn, Manfred Uhle

submitted: 16th August 1994

Weierstraß–Institut
für Angewandte Analysis
und Stochastik
Mohrenstraße 39
D – 10117 Berlin
Germany

Report No. 9
Berlin 1994

Edited by
Weierstraß–Institut für Angewandte Analysis und Stochastik (WIAS)
Mohrenstraße 39
D — 10117 Berlin
Germany

Fax: + 49 30 2004975
e–mail (X.400): c=de;a=d400;p=iaas-berlin;s=preprint
e–mail (Internet): preprint@iaas-berlin.d400.de

INHALTSVERZEICHNIS

1. Einleitung

Elektrische Netzwerke sind Modelle realer technischer Schaltungen, deren Eigenschaften mit den Größen Spannung und Strom beschrieben werden können. Beim rechnergestützten Entwurf integrierter Schaltungen dient die Analyse elektrischer Netzwerke zur Verifikation des elektrischen Verhaltens einer entworfenen Schaltung. Eine Netzwerkanalyse erfordert die numerische Lösung eines Systems von nichtlinearen Algebro-Differentialgleichungen (DAE-System).
Klassische Netzwerkanalyseprogramme verwenden implizite Integrationsverfahren zur Diskretisierung dieses DAE-Systems, modifizierte Newtonverfahren zur Lösung der dabei in jedem diskreten Zeitschritt entstehenden nichtlinearen Systeme und schließlich direkte Verfahren zur Lösung der linearen Systeme unter Verwendung von Techniken für sparsam besetzte Matrizen.

Der in den letzten Jahren ständig gestiegene Integrationsgrad der VLSI-Schaltungen hat dazu geführt, daß immer größere Teilschaltungen elektrisch simuliert, d.h. immer größere DAE-Systeme gelöst werden müssen.
Schon bei Netzwerken mit einigen tausend Transistorenmodellen stoßen klassische Elektriksimulatoren an Leistungsgrenzen. Für eine dynamische Analyse solcher Netzwerke benötigen sie selbst auf leistungsstarken Rechnern oftmals einige Stunden Rechenzeit. Noch größere Schaltungen sind mit ihnen kaum simulierbar.
Um solche Schaltungen simulieren zu können, müssen numerische Verfahren eingesetzt werden, die einen wesentlich geringeren Rechenzeitaufwand erfordern. Eine Möglichkeit dafür ist die Ausnutzung topologischer und schaltungstechnischer Eigenschaften elektrischer Netzwerke.
Eine in diesem Zusammenhang wesentliche Eigenschaft von VLSI-Schaltungen ergibt sich aus ihrer hierarchischen Struktur. Die Modelle solcher Schaltungen bestehen in der Regel aus mehreren Funktionsblöcken, die wiederum hierarchisch aus teilweise identischen Teilstrukturen aufgebaut sind. Einige dieser Funktionsblöcke sind über größere Zeitintervalle elektrisch inaktiv (latent). Darüber hinaus läßt sich, insbesondere bei digitalen MOS-Netzwerken, oftmals apriori der Signalfluß zwischen den Funktionsblöcken beschreiben.

Die in den letzten 15 Jahren entstandenen Netzwerksimulatoren nutzen diese und weitere Eigenschaften bestimmter Schaltungsklassen aus und setzen vereinfachte numerische Verfahren ein [NSV83].
Zum einen wurden dabei Netzwerkanalyseprogramme entwickelt, die die hierarchische Struktur der Netzwerke ausnutzen, indem sie Mehrebenen-Newtonverfahren einsetzen. Diese Simulatoren erfüllen dieselben Genauigkeitsanforderungen wie klassische Analyseprogramme und sind i.a. universell einsetzbar [RSVH79], [ELD82], [FS88], [HS84], [EP85], [GBG85].
Zum anderen wurden solche Netzwerkanalyseprogramme entwickelt, die – ausgehend von einer vereinfachten Modellierung und auf der Grundlage des Signalflußgraphen – Relaxationsverfahren auf dem Niveau der nichtlinearen Systeme [OR70] bzw. auf dem Differentialgleichungsniveau [LRSV82] einsetzen. Diese Simulatoren sind i.a. sehr effektiv auf Netzwerke aus speziellen Schaltungsklassen anwendbar. Die dabei bestimmten Spannungs- und Stromwerte sind teilweise von geringerer Genauigkeit als die Lösungen der klassischen Netzwerkanalyse [HPS87], [CGK75], [Sch87].

Im Netzwerkanalyseprogramm MAGNUS (**M**ehrstufige **A**nalyse **g**roßer **N**etzwerke **u**nd **S**ysteme) werden diese beiden Vorgehensweisen kombiniert. Dabei sind die folgenden Zielstellungen realisiert worden:

(i) Beschreibung sehr großer hierarchischer strukturierter elektrischer Netzwerke durch ein hierarchisches Gleichungssystem;

(ii) Benutzung ein und desselben Gleichungssystems für verschiedene statische und dynamische Analysearten;

(iii) Anwendung qualitativ verschiedener numerischer Analyseverfahren für dieselbe Netzwerkanalyseart;

(iv) Anwendung numerischer und topologischer Partitionierungsverfahren;

(v) Explizite und implizite Darstellungsmöglichkeiten der u–i–Relation der Netzwerkelemente, d.h. hohe Allgemeinheit bei der Wahl der abhängigen und unabhängigen Netzwerkvariablen.

Ausgangspunkt einer Netzwerkanalyse mit dem Programmsystem MAGNUS ist eine hierarchische Beschreibung der Netzwerktopologie und die Definition der Netzwerkelemente. Es wird ein hierarchisch strukturiertes Gleichungssystem verwendet, dessen Gleichungen mit Hilfe der Methode der erweiterten Knotenspannungsanalyse [HRB75] aufgestellt werden. Die Struktur des Gleichungssystems umfaßt die Netzwerkgleichungssysteme der

– Hauptebene,
– Blockebene

und bei Bedarf Netzwerkgleichungssysteme der

– Basisebene.

Diese Hierarchie wird vom Schaltungsentwerfer definiert. Weiterhin hat der Programmnutzer die Möglichkeit, die Blocknetzwerkklemmen bezüglich des Signalflusses [Uhl90b] zu markieren. Entsprechend dieser natürlichen Strukturebenen wird auch das Netzwerkgleichungssystem strukturiert und mit Newton–ähnlichen Mehrebenenverfahren gelöst, wobei auf verschiedenen Ebenen Relaxationsverfahren eingesetzt werden.

Dabei ist wesentlich, daß zur Lösung von Teilsystemen, für die die Relaxationsverfahren nicht hinreichend konvergieren, dynamisch auf die Algorithmen der Mehrebenenverfahren umgeschaltet wird.

Im Programmsystem MAGNUS werden Algorithmen verwendet, die die Besonderheiten von vektorieller Rechentechnik sowohl für die Berechnung von Funktion und Jacobimatrix als auch zur Lösung der linearen Teilgleichungssysteme vorteilhaft ausnutzen, da diese beiden Teilaufgaben bei sehr großen Systemen mehr als 80 Prozent der Rechenzeit erfordern.

Das Programmsystem MAGNUS wurde von J. Borchardt, I. Bremer, F. Grund, D. Horn und M. Uhle implementiert. Es läuft unter VMS auf VAX-Maschinen und umfaßt etwa 500 Subroutinen in FORTRAN77. Unter UNIX wurden ausgewählte Beispiele elektrischer Schaltungen auch auf CRAY-Vektorrechnern simuliert.

Das Netzwerkanalyseprogramm MAGNUS wird von einem Unternehmen und von mehreren wissenschaftlichen Instituten zur Simulation des elektrischen Verhaltens mikroelektronischer Schaltungen eingesetzt. Besonders bewährt hat es sich bei der Verifikation des Entwurfs von dynamischen Speicherschaltkreisen (1– bzw. 4–Mbit–DRAM). Hierbei wurden VLSI–Schaltungen mit mehr als 30 000 Transistoren simuliert.

2. Schaltungsmodellierung durch elektrische Netzwerke

2.1. Netzwerkbegriff.

In der Literatur [EMR77], [KD69], [RSch76] sind verschiedene Ansätze zur Definition des Begriffs elektrisches Netzwerk zu finden. Wir verwenden den Begriff elektrisches Netzwerk als Bezeichnung für Modelle, die dazu dienen, die Eigenschaften technischer Schaltungen mathematisch zu beschreiben. Der Einfachheit halber betrachten wir ein komplexes elektrisches Netzwerk als Zusammenschaltung von Elementarnetzwerken, wobei sowohl für die Zusammenschaltung der Elementarnetzwerke als auch für jedes einzelne Elementarnetzwerk die folgende Netzwerkdefinition (vgl. [RSt76] gelte.

Unter einem **elektrischen Netzwerk** verstehen wir einen zusammenhängenden, orientierten Graphen, dessen Zweigen je zwei Zeitfunktionen zugeordnet sind, die als Zweigspannung und Zweigstrom bezeichnet werden. Die Zweigspannungen und Zweigströme eines elektrischen Netzwerkes müssen einer bestimmten Strom–Spannungs–Relation genügen.

Sind $\mathcal{Z}$ und $\mathcal{K}$ die **Zweig– bzw. Knotenmenge** des elektrischen Netzwerkes $\mathcal{N}$ und setzt man gleiche Orientierung der Zweigspannungen und –ströme voraus, dann bezeichnet

$$\mathcal{G} := (\mathcal{Z}, \mathcal{K}, \mathcal{A}),$$

den **Netzwerkgraphen** und $\mathcal{A} : \mathcal{Z} \to \mathcal{K} \times \mathcal{K}$ die Inzidenzabbildung des Netzwerkgraphen zur Kennzeichnung der positiven Zählrichtung der Zweigspannungen und –ströme.

Mit $U^{\mathcal{Z}}$ und $I^{\mathcal{Z}}$ bezeichnen wir die Menge der Zweigspannungs– bzw. Zweigstromwerte und mit $\mathcal{T}$ den Definitionsbereich der Menge der **Zweigspanungsfunktionen**

$$\mathcal{U} := \left\{ u \mid u : T \to U^{\mathcal{Z}}, T \subset \mathcal{T} \right\}$$

und der **Zweigstromfunktionen**

$$\mathcal{J} := \left\{ i \mid i : T \to I^{\mathcal{Z}}, T \subset \mathcal{T} \right\}.$$

Die Menge $\mathcal{T}$ soll im weiteren als **Zeitachse** interpretiert werden.

Die Mengen $U^{\mathcal{Z}}, I^{\mathcal{Z}}$ und $\mathcal{T}$ sind mit der Struktur eindimensionaler reeller linearer Räume versehen. Die Menge

$$\mathcal{S} \subseteq \mathcal{U} \times \mathcal{J}$$

legt die Grundmenge der im Modellbereich zugelassenen Spannungs– und Stromfunktionen fest. Für unsere weiteren Betrachtungen genügt es, das $\mathcal{T} := \{T\}$ vorausgesetzt wird, wobei T ein abgeschlossenes Intervall der Zeitachse bezeichnen soll. Für die im Modellbereich zulässigen Funktionen $(u, i) \in \mathcal{S}$ werde Riemannintegrierbarkeit und stetige Differenzierbarkeit vorausgesetzt. Mit

$$\mathcal{V} \subseteq \mathcal{S}$$

bezeichnen wir die **Strom–Spannungs–Relation** (u-i-Relation) eines Netzwerkes. Im weiteren existiere stets eine Abbildung $F : \mathcal{S} \to \mathbb{R}^n,\ n \in \mathbb{N}$, so daß

$$\mathcal{V} = \{(u, i) \in \mathcal{S} \mid F(u, i) = 0\}$$

gilt. Dann wird F als **Darstellung der u-i-Relation** bezeichnet.

Die Algorithmen der Netzwerkanalyseprogramme arbeiten nur mit normierten physikalischen Größen. Für **normierte Netzwerke** gilt: $\mathcal{T} = U = I = \mathbb{R}$. Bei der Wahl der Bezugsgrößen sollte stets die Wahl numerischer Paramter, wie zum Beispiel Abbruchschranken berücksichtigt werden, da unter Umständen die Funktionsfähigkeit der numerischen Verfahren beeinträchtigt werden kann. Üblicherweise werden Spannungen in Volt $[V]$, Stromstärken in Milliampere $[mA]$, die Zeit in Nanosekunden $[ns]$ und die Längenangaben in Mikrometer $[\mu m]$ angegeben.

In der Netzwerktheorie ist es üblich, Netzwerke in Klassen einzuteilen. **Netzwerkklassen** können unter anderem nach Eigenschaften der u–i–Relation und nach Eigenschaften der Topologie des Netzwerkes definiert werden.

Der Begriff **n–Pol** definiert eine Klasse von Netzwerken mit sternförmigen Netzwerkgraphen aus n Knoten und $(n-1)$ Zweigen. Ein n–Pol ist ein Repräsentant für eine Netzwerkklasse, die aufgrund von Eigenschaften der Topologie elektrischer Netzwerke gebildet werden kann.

Die folgenden Zweipole sind Repräsentanten für Netzwerkklassen, die aufgrund von Eigenschaften der u–i–Relation $((u(t), i(t)) \in \mathcal{S},\ t \in T)$ definiert werden können:

(i) resistive Netzwerke:

Widerstand:	$u = R \cdot i,$	$R \in \mathbb{R}$
Leitwert:	$i = G \cdot u,$	$G \in \mathbb{R}$
Unabhängige Spannungsquelle:	$u = u^0,$	$u^0 : T \to \mathbb{R}$
Unabhängige Stromquelle:	$i = i^0,$	$i^0 : T \to \mathbb{R}$

(ii) dynamische Netzwerke:

$$\text{Kapazität:} \quad i = C \cdot \dot{u}, \quad C \in \mathbb{R}, \quad \dot{u} := \frac{du}{dt}$$

$$\text{Induktivität:} \quad u = L \cdot \dot{i}, \quad L \in \mathbb{R}, \quad \dot{i} := \frac{di}{dt}$$

Kompliziertere Darstellungen der u–i–Relation dynamischer Netzwerke können in *expliziter* Form, zum Beispiel als Admittanzdarstellungen

$$i = G(u, \dot{u}, t)$$
$$G : \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^n$$

oder in *impliziter* Form

$$0 = F\left(u, \dot{u}, i, \dot{i}, t\right)$$
$$F : \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^n$$

angegeben werden.

2.2. Netzwerkgleichungssystem und Netzwerkanalyse.

Bei der Analyse elektrischer Netzwerke wird die Menge $\mathcal{S}$ als Grundmenge aller im Modellbereich zugelassenen Spannungs– und Stromfunktionen üblicherweise auf eine Teilmenge $\mathcal{S}^{(K)}$ eingeschränkt, da gefordert wird, daß für jede orientierte Masche $(\mathcal{Z}_M^+, \mathcal{Z}_M^-)$ des Netzwerkgraphen alle Zweigspannungsfunktionen dem **Kirchhoffschen Spannungsgesetz**, d.h.

$$\sum_{b \in \mathcal{Z}_M^+} u_b(t) - \sum_{b \in \mathcal{Z}_M^-} u_b(t) = 0, \quad \forall t \in T$$

sowie für jeden orientierten Schnitt $(\mathcal{Z}_S^+, \mathcal{Z}_S^-)$ des Netzwerkgraphen alle Stromfunktionen dem **Kirchhoffschen Stromgesetz**, d.h.

$$\sum_{b \in \mathcal{Z}_S^+} i_b(t) - \sum_{b \in \mathcal{Z}_S^-} i_b(t) = 0, \quad \forall t \in T$$

genügen. Allgemein formuliert besteht die Aufgabe der **Netzwerkanalyse** darin, die Menge

$$\mathcal{L} = \mathcal{V} \cap \mathcal{S}^{(K)}$$

zu bestimmen, d.h. alle Spannungs– und Stromfunktionen zu ermitteln, die auf dem Intervall T der u–i–Relation $\mathcal{V}$ und den Kirchhoffschen Gesetzen genügen.

Wird die u–i–Relation $\mathcal{V}$ durch ein Gleichungssystem beschrieben, dann werden die Elemente von $\mathcal{L}$ durch das Lösen des zugeordneten Netzwerkgleichungssystems bestimmt.
Der Graph eines elektrischen Netzwerkes besitze k Knoten und z Zweige. Dann erhält man aus dem Kirchhoffschen Stromgesetz ein System von $(k-1)$ linear unabhängigen Gleichungen und aus dem Kirchhoffschen Spannungsgesetz ein System von $z-(k-1)$ linear unabhängigen Gleichungen. Bezeichnet man mit K die Knotenmatrix des Kirchhoffschen Stromgesetzes bzw. mit M die Maschenmatrix des Kirchhoffschen Spannungsgesetz, dann gilt:

$$Ki = 0$$

bzw.

$$Mu = 0.$$

Für ein elektrisches Netzwerk mit z Zweigen, k Knoten und der allgemeinen Darstellung der u–i–Relation

$$\begin{aligned} 0 &= V\left(u, \dot{u}, i, \dot{i}, t\right) \\ V &: \mathbb{R}^z \times \mathbb{R}^z \times \mathbb{R}^z \times \mathbb{R}^z \times \mathbb{R} \to \mathbb{R}^z \end{aligned}$$

kann durch

$$\begin{aligned} Ki &= 0 \\ Mu &= 0 \\ 0 &= V\left(u, \dot{u}, i, \dot{i}, t\right) \end{aligned} \tag{2.1}$$

ein allgemeines Gleichungssystem aus $2z$ Gleichungen zugeordnet werden, wobei alle Zweigspannungen u und alle Zweigströme i unabhängige Variable sind. In der Literatur wird das Gleichungssystem (2.1) daher als **Zweigspannungs–Zweigstrom–Gleichungssystem** bezeichnet. Ausgehend von diesem Gleichungssystem können – wie in der Literatur beschrieben – eine Vielzahl von Gleichungssystemen für die Netzwerkanalyse abgeleitet werden [EMR77], wenn man zum Beispiel für die Kirchhoffschen Gesetze äquivalente Formulierungen benutzt oder spezielle Darstellungen für die u–i–Relation des Netzwerkes berücksichtigt.

Im Netzwerkanalyseprogramm MAGNUS wurde ein **erweitertes Knotenspannungsgleichungssystem** [HRB75] verwendet. Zur Definition von Knotenspannungen wird aus der Menge der Netzwerkknoten ein **Bezugsknoten** gewählt. Die Potentialdifferenz eines Netzwerkknotens zu diesem Bezugsknoten wird als **Knotenspannung** bezeichnet. Variable dieses Gleichungssystems sind alle Knotenspannungen, soweit diese nicht durch unabhängige Spannungsquellen vorgegeben sind, sowie ausgewählte Zweigströme. Charakteristisch für dieses Gleichungssystem ist, daß ein Teil der u–i–Relation des Netzwerkes für z_1 Zweige durch Gleichungen der Form

$$\begin{aligned} i_1 &= G\left(u, \dot{u}, i_2, \dot{i}_2, t\right) \\ G &: \mathbb{R}^z \times \mathbb{R}^z \times \mathbb{R}^{z_2} \times \mathbb{R}^{z_2} \times \mathbb{R} \to \mathbb{R}^{z_1} \end{aligned}$$

der andere Teil, also $z_2 = z - z_1$ Gleichungen, allgemein nur durch

$$\begin{aligned} 0 &= V\left(u, \dot{u}, i_2, \dot{i}_2, t\right) \\ V &: \mathbb{R}^z \times \mathbb{R}^z \times \mathbb{R}^{z_2} \times \mathbb{R}^{z_2} \times \mathbb{R} \to \mathbb{R}^{z_2} \end{aligned}$$

beschrieben wird.

Mit der gewählten Aufteilung der Zweigmenge folgt für das Kirchhoffsche Stromgesetz

$$K \begin{bmatrix} i_1 \\ i_2 \end{bmatrix} = \begin{bmatrix} K_1 K_2 \end{bmatrix} \begin{bmatrix} i_1 \\ i_2 \end{bmatrix} = 0.$$

Für das Kirchhoffsche Spannungsgesetz wird die äquivalente Form

$$u = K^t u^k$$

verwendet, wobei u^k den Vektor der zu einem Bezugsknoten orientierten Knotenspannungen bezeichnet. Eliminiert man die Zweigspannungen u sowie die Zweigströme i_1, dann erhält man folgendes System aus $(k - 1 + z_2)$ Gleichungen:

$$\begin{aligned} 0 &= \begin{bmatrix} K_1 K_2 \end{bmatrix} \begin{bmatrix} G\left(K^T u^k, K^T \dot{u}^k, i_2, \dot{i}_2, t\right) \\ i_2 \end{bmatrix} \\ 0 &= V\left(K^T u^k, K^T \dot{u}^k, i_2, \dot{i}_2, t\right). \end{aligned} \tag{2.2}$$

Im weiteren werden wir uns ausschließlich mit Netzwerkgleichungssystemen für **konzentrierte dynamische Netzwerke** beschäftigen, d.h., die u–i–Relation wird stets durch eine Momentanwertrelation zwischen Zweigspannung und Zweigstrom dargestellt. Solche Netzwerke werden mathematisch durch Systeme von Algebro–Differentialgleichungen beschrieben (vgl. (2.1), (2.2)), d.h. es ist eine Funktion $x(t), t \in [t_0, t_e]$ mit Werten in $\mathbb{R}^n$ gesucht, die die Anfangswertaufgabe der Gestalt

$$\begin{aligned} 0 &= f(x, \dot{x}, t) \\ f &: \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^n \end{aligned} \tag{2.3}$$

mit $x(t_0) = x^0,\ x^0 \in \mathbb{R}^n$ löst. Das Symbol $\dot{x}$ bezeichnet dabei die zeitliche Ableitung von x. Der Anfangswert x^0 ist in der Regel eine Lösung des nichtlinearen Gleichungssystems

$$f(x, 0, t_0) = 0. \tag{2.4}$$

Je nach Aufgabenstellung unterscheidet man verschiedene Analyseverfahren. Die wichtigsten Verfahren sind die dynamische Analyse (2.3), in der Literatur auch als Transient– oder TR–Analyse bezeichnet, und die statische Analyse (2.4), in der Literatur häufig als Gleichstrom– oder DC–Analyse bezeichnet.

2.3. Klemmenverhalten elektrischer Netzwerke und Makromodellierung.

Der Betriff des Klemmenverhaltens ist einer der zentralen Begriffe der Netzwerktheorie. Anschaulich kann man den Begriff des Klemmenverhaltens mit folgendem Sachverhalt erklären:

Bei einem gegebenen Netzwerk sei eine gewisse Anzahl von Knoten ausgezeichnet. An diesen ausgezeichneten Knoten soll das gegebene Netzwerk mit anderen Netzwerken zusammengeschaltet werden. Damit zerfällt die Knotenmenge $\mathcal{K}$ des Netzwerkes in zwei Klassen: Die Menge der inneren Knoten und die Menge der äußeren Knoten.
Die äußeren Knoten werden auch als **Klemmen** des Netzwerkes bezeichnet. Deshalb werden die Knotenspannungen und –ströme an den äußeren Knoten auch als Klemmenspannungen u^* und Klemmenströme i^* mit $u^* = (u_1^*, u_2^*, \ldots, u_n^*)$ bzw. $i^* = (i_1^*, i_2^*, \ldots, i_n^*)$ bezeichnet.

Das **Klemmenverhalten** eines elektrischen Netzwerkes $\mathcal{N}$ ist durch die Menge aller geordneten Paare von Funktionen von Klemmenspannungen und Klemmenströmen des Netzwerkes $\mathcal{N}$ erklärt, die man erhält, wenn man das Netzwerk $\mathcal{N}$ mit beliebigen anderen Netzwerken beschaltet und das aus dieser Zusammenschaltung resultierende Netzwerk analysiert.

Mit dieser Definition wird der Begriff des Klemmenverhaltens direkt auf den Begriff Netzwerkanalyse zurückgeführt. Eine Abbildung F^* heißt **Darstellung des Klemmenverhaltens**, wenn

$$\mathcal{V}^* = \{(u^*, i^*) \mid F^*(u^*, i^*) = 0\}$$

gilt.
Wir verwenden den Begriff **Makromodell** zur Bezeichnung von Netzwerkmodellen, die als Ersatz für kompliziertere Netzwerke benutzt werden, zum Beispiel für bestimmte schaltungstechnische Varianten oder für Netzwerkmodelle mit vereinfachter Netzwerktopologie bzw. u–i–Relation [RSVH79].

2.4. Beispiel.

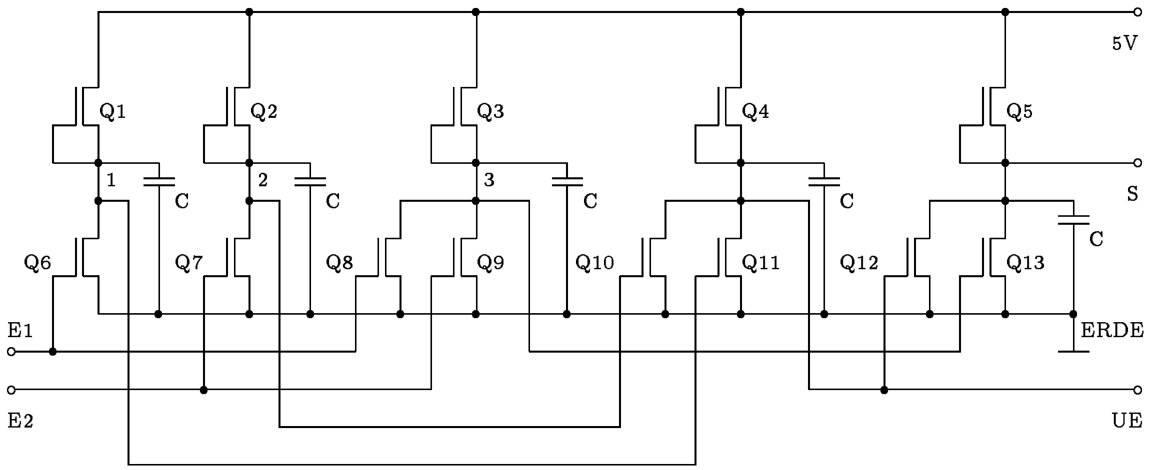


Abbildung 2.1: Schaltung eines Halbadders

Abbildung 2.1 zeigt eine Schaltungsvariante für einen Halbadder. Die Logikgatter sind aus Enhancement– und Depletiontransistoren aufgebaut. Die Knotenkapazitäten symbolisieren die Kapazitäten der Leitbahnen. Ersetzt man jeden Transistor aus Abbildung 2.1 durch das in Abbildung 2.2 eingeführte Modell, erhält man ein Netzwerkdiagramm zur Schaltung des Halbadders.

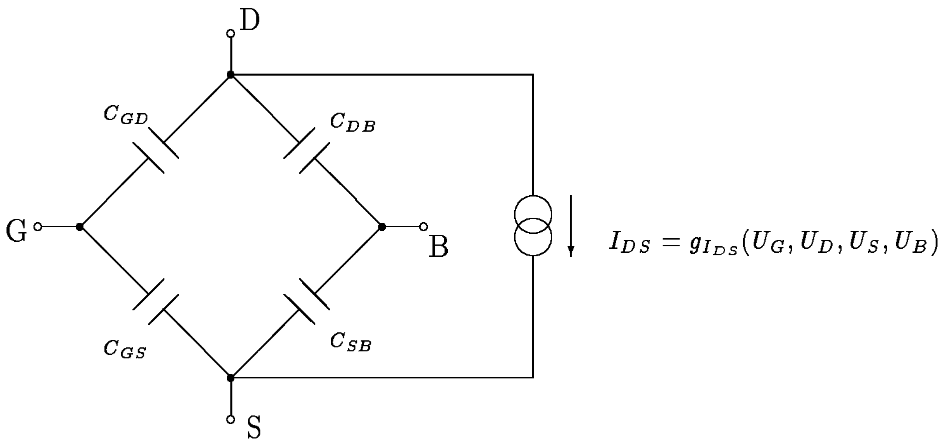


Abbildung 2.2: Diagramm der Transistormodelle

Zur Definition von Knotenspannungen wird der Knoten ERDE als Bezugsknoten des Netzwerkes gewählt. Dann bezeichnet beispielsweise u_2 die Knotenspannung des Knoten 2 bezüglich ERDE. Als Klemmen des Netzwerkes sind die Knoten E1, E2, S und UE ausgezeichnet. Innere Knoten des Netzwerkes sind 1,2,3,4 und ERDE.

Am Knoten 1 ist die Knotenspannung durch das Potential U_0 vorgegeben. Zur Vereinfachung der Schreibweise der u–i–Relation eines Transistormodells $Q_j (j = 1, \ldots, 13)$ wird der folgende Vektor der Klemmenspannungen

$$u_{Qj} := \left(u^G_{Qj}, u^D_{Qj}, u^S_{Qj}, u^B_{Qj} \right)$$

eingeführt und mit

$$\begin{aligned} i^G_{Qj} &:= g_G\left(u_{Qj}, \dot{u}_{Qj}\right) \\ i^D_{Qj} &:= g_D\left(u_{Qj}, \dot{u}_{Qj}\right) \\ i^S_{Qj} &:= g_S\left(u_{Qj}, \dot{u}_{Qj}\right) \end{aligned}$$

unter Beachtung von

$$\begin{aligned} g_G &: \mathbb{R}^2 \times \mathbb{R}^2 \times \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R} \\ g_D &: \mathbb{R}^2 \times \mathbb{R}^2 \times \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R} \\ g_S &: \mathbb{R}^2 \times \mathbb{R}^2 \times \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R} \end{aligned}$$

werden die nichtlinearen Funktionen der Klemmenströme der Transistormodelle (vgl. Abbildung 2.2) definiert. Für die stets zum Bezugsknoten gerichteten Kapazitäten C wird

$$i_C = C\dot{u}$$

als Darstellung der u–i–Relation benutzt, wobei $\dot{u}$ die zeitliche Ableitung der Knotenspannung u bezeichnet. Damit ist es möglich, beim Aufstellen des KIG die Zweigströme bis auf die Klemmenströme $(i_{E1}, i_{E2}, i_{UE}, i_S)$ zu eliminieren und die Gleichungen für das Netzwerk aus Abbildung 2.1 mit den genannten Abkürzungen in folgender Form zu notieren:

$$
\begin{aligned}
0 =& -i_{E1} + g_G\left(u_{Q6}, \dot{u}_{Q6}\right) + g_G\left(u_{Q8}, \dot{u}_{Q8}\right) \\
0 =& -i_{E2} + g_G\left(u_{Q7}, \dot{u}_{Q7}\right) + g_G\left(u_{Q9}, \dot{u}_{Q9}\right) \\
0 =& g_S\left(u_{Q1}, \dot{u}_{Q1}\right) + g_G\left(u_{Q1}, \dot{u}_{Q1}\right) + g_D\left(u_{Q6}, \dot{u}_{Q6}\right) + g_G\left(u_{Q11}, \dot{u}_{Q11}\right) - C\dot{u}_1 \\
0 =& g_S\left(u_{Q2}, \dot{u}_{Q2}\right) + g_G\left(u_{Q2}, \dot{u}_{Q2}\right) + g_D\left(u_{Q7}, \dot{u}_{Q7}\right) + g_G\left(u_{Q10}, \dot{u}_{Q10}\right) - C\dot{u}_2 \\
0 =& g_S\left(u_{Q3}, \dot{u}_{Q3}\right) + g_G\left(u_{Q3}, \dot{u}_{Q3}\right) + g_D\left(u_{Q8}, \dot{u}_{Q8}\right) + g_D\left(u_{Q9}, \dot{u}_{Q9}\right) + \\
& + g_G\left(u_{Q13}, \dot{u}_{Q13}\right) - C\dot{u}_3 \\
0 =& g_S\left(u_{Q5}, \dot{u}_{Q5}\right) + g_G\left(u_{Q5}, \dot{u}_{Q5}\right) + g_D\left(u_{Q12}, \dot{u}_{Q12}\right) + g_D\left(u_{Q13}, \dot{u}_{Q13}\right) - \\
& - C\dot{u}_{UE} - i_{UE} \\
0 =& g_S\left(u_{Q4}, \dot{u}_{Q4}\right) + g_G\left(u_{Q4}, \dot{u}_{Q4}\right) + g_D\left(u_{Q10}, \dot{u}_{Q10}\right) + g_D\left(u_{Q11}, \dot{u}_{Q11}\right) + \\
& + g_G\left(u_{Q12}, \dot{u}_{Q12}\right) - C\dot{u}_S - i_S
\end{aligned}
$$

Das resultierende Gleichungssystem ist ein erweitertes Knotenspannungsgleichungssystem mit 7 Gleichungen. Durch Elimination der Knotenspannungen an den inneren Knoten ist es möglich, eine lokale Approximation für das Klemmenverhalten der Halbadderschaltung der Gestalt

$$
\begin{aligned}
0 &= F^*\left(u_{E1}, i_{E1}, u_{E2}, i_{E2}, u_S, i_S, u_{UE}, i_{UE}\right) \\
F^* &: \mathbb{R}^2 \times \mathbb{R}^2 \times \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}^4.
\end{aligned}
$$

zu bestimmen.

Diese lokale Darstellung für das Klemmenverhalten dient als Darstellung für die u–i–Relation des Makromodells aus Abbildung 2.3.

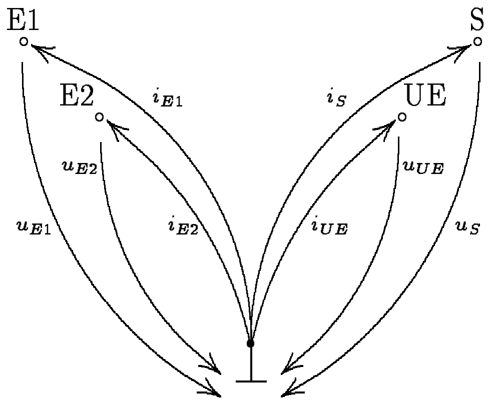


Abbildung 2.3: Makromodell der Schaltung eines Halbadders

3.1. Netzwerkstruktur.

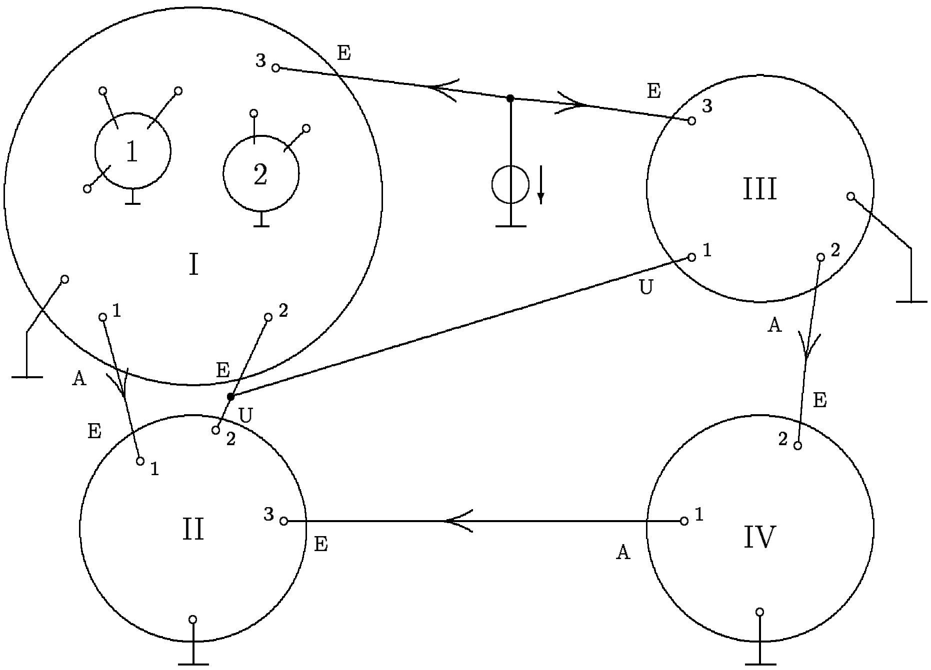
3.1.1. *Hierarchie und Signalfluß.*

Die Eingabemöglichkeit einer hierarchischen Topologie elektrischer Netzwerke ist besonders der Denkweise des Schaltkreisentwerfers beim Entwurf sehr großer Schaltungen angepaßt. Im Netzwerkanalyseprogramm MAGNUS wird ein Gleichungssystem mit einer dreistufigen Hierarchie eingesetzt, die es ermöglicht, ein Netzwerk in eine Basis–, eine Block– und eine Hauptebene zu strukturieren.

Die Ebene der **Blocknetzwerke** symbolisiert Schaltungsteile, die innerhalb der Schaltung eine bestimmte Funktion erfüllen. Blocknetzwerke können Basisnetzwerke enthalten. Die Aufteilung der Schaltung in Blocknetzwerke sollte unter Beachtung des Signalflusses erfolgen. Wie bei einer Logiksimulation werden dabei die Klemmen der Blocknetzwerke mit den Symbolen E für den Eingang, A für den Ausgang eines Signals und U (unbestimmt) für eine bidirektionale Verbindung markiert.

Die Ebene des **Hauptnetzwerkes** wird durch die Zusammenschaltung der Blocknetzwerke beschrieben. Diese Zusammenschaltung darf nur durch zum Bezugsknoten gerichtete Spannungsquellen ergänzt werden.

Die unterste Ebene – die Ebene der **Basisnetzwerke** – umfaßt Modelle oft auftretender Schaltungsstrukturen. Definiert man ein Teilschaltungsmodells als Basisnetzwerk, sollte man berücksichtigten, daß einer großen Anzahl innerer Knoten – wenn möglich – nur wenig Randknoten gegenüber stehen, um eine hohe Effektivität der eingesetzten numerischen Verfahren gewährleisten zu können. Der Signalfluß bleibt an den Klemmen der Basisnetzwerke unberücksichtigt, d.h., alle Klemmen werden als bidirektionale Verbindungen betrachtet.



Abbilung 3.1: Netzwerkstrukturierung bei allgemeinem Signalfluß zwischen den Blocknetzwerken

Zur Einsparung von Datenspeicher wird für Netzwerke der Basisebene nur eine Darstellung der Beschreibung der Netzwerkgleichungen erzeugt. Ziel ist es, für eine derart allgemeine Netzwerkstrukturierung ein Gleichungssystem abzuleiten, welches sowohl mit Iterationsverfahren vom Gauß–Seidel–Typ als auch mit Mehrebenenverfahren vom Newton–Typ gelöst werden kann (vgl. 4. Abschnitt und [Uhl87]).

3.1.2. *Netzwerkpartitionierung mit Hilfe des Signalflusses.*

Ziel des folgenden Algorithmus [Uhl90b] ist die Überführung des bidirektionalen Signalflusses zwischen Netzwerken der Blockebene in einen unidirektionalen Signalfluß. Ausgangspunkt ist eine allgemeine dreistufige hierarchische Netzwerkstruktur.
Unser Beispiel in Abbildung 3.1 besteht aus vier Blocknetzwerken (I, II, III, IV). Das Blocknetzwerk I enthält zusätzlich die Basisnetzwerke 1 und 2. Die Klemmen der Blocknetzwerke sind mit den Marken E, A und U zur Kennzeichnung von Signaleingängen, –ausgängen sowie bidirektionalen Signalverbindungen versehen.

In Abbildung 3.2 wird die Zurückführung der Netzwerktopologie mit bidirektionalem Signalfluß auf eine Topologie mit unidirektionalem Signalfluß gezeigt, indem ein zusätzliches Blocknetzwerk ergänzt wird, das keine Netzwerkelemente enthält. Die Anzahl der Knoten der Hauptebene, die mit U–markierten Blocknetzwerkklemmen korrespondieren, ergibt die Anzahl der Klemmen des zusätzlichen Blocknetzwerkes. Die Klemmen des zusätzlichen Blocknetzwerkes V werden als Signalausgänge mit A markiert, die korrespondierenden Klemmenmarkierungen der restlichen Blocknetzwerke als Signaleingänge in E–Markierungen geändert.

Mit diesem Vorgehen kann ein beliebiger Signalfluß zwischen Blocknetzwerken stets auf einen unidirektionalen Signalfluß zurückgeführt werden.

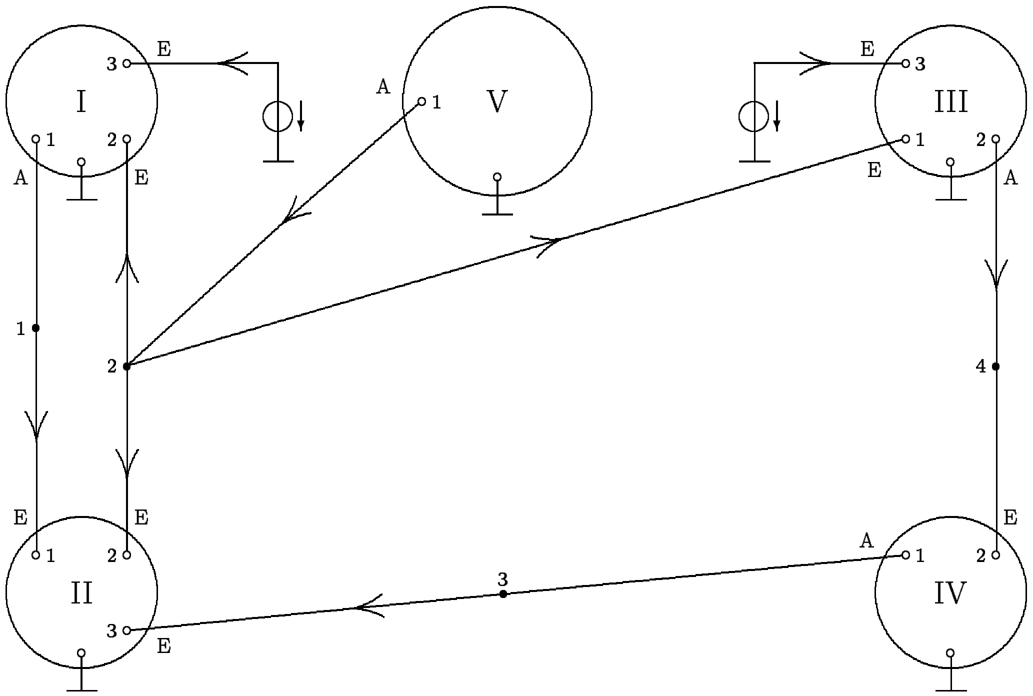


Abbildung 3.2: Überführung der E–A–U–Markierung der Blocknetzwerke in eine E–A–Markierung

Grundsätzlich ist es möglich, für ein hierarchisch beschriebenes Netzwerk ein hierarchisches Gleichungssystem anzugeben. Zur Anwendung effektiver numerischer Lösungsverfahren ist es nötig, das gewonnene hierarchische Gleichungssystem zu entkoppeln. Diese Entkopplung wird im Programmsystem MAGNUS auf der Blockebene durchgeführt. Zur Entkopplung des Netzwerkes wird die in Abbildung 3.3 mit VI gekennzeichnete Hauptnetzwerkstruktur eingeführt. Die Klemmenmarkierung erfolgt dual zu den korrespondierenden Blocknetzwerkklemmen.

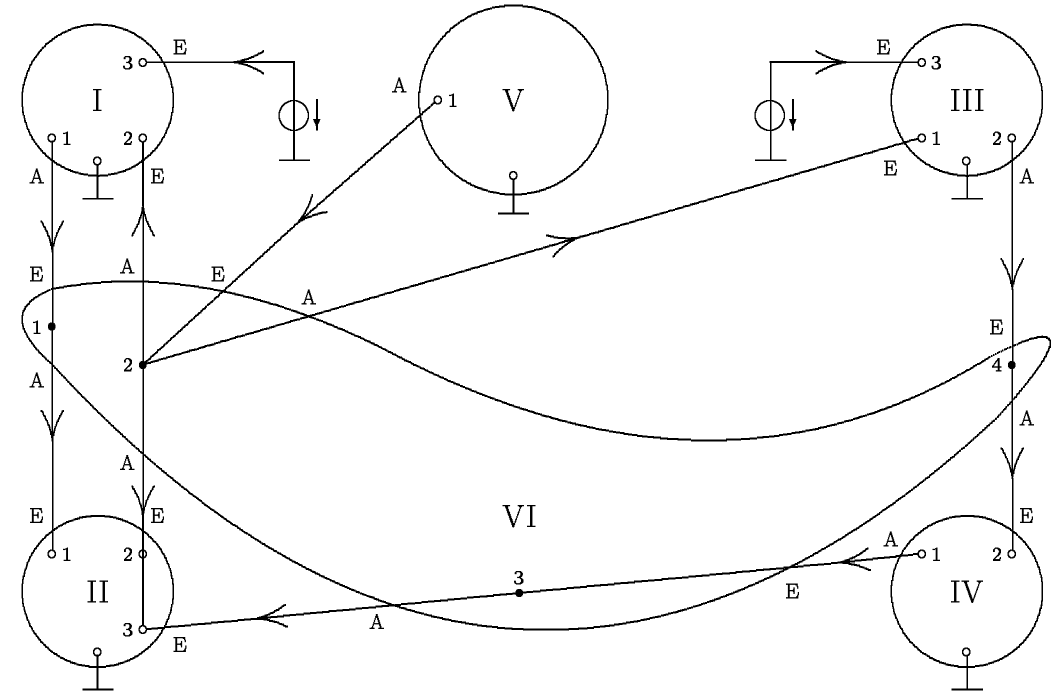


Abbildung 3.3: Einführung und Markierung der Hauptnetzwerkstruktur

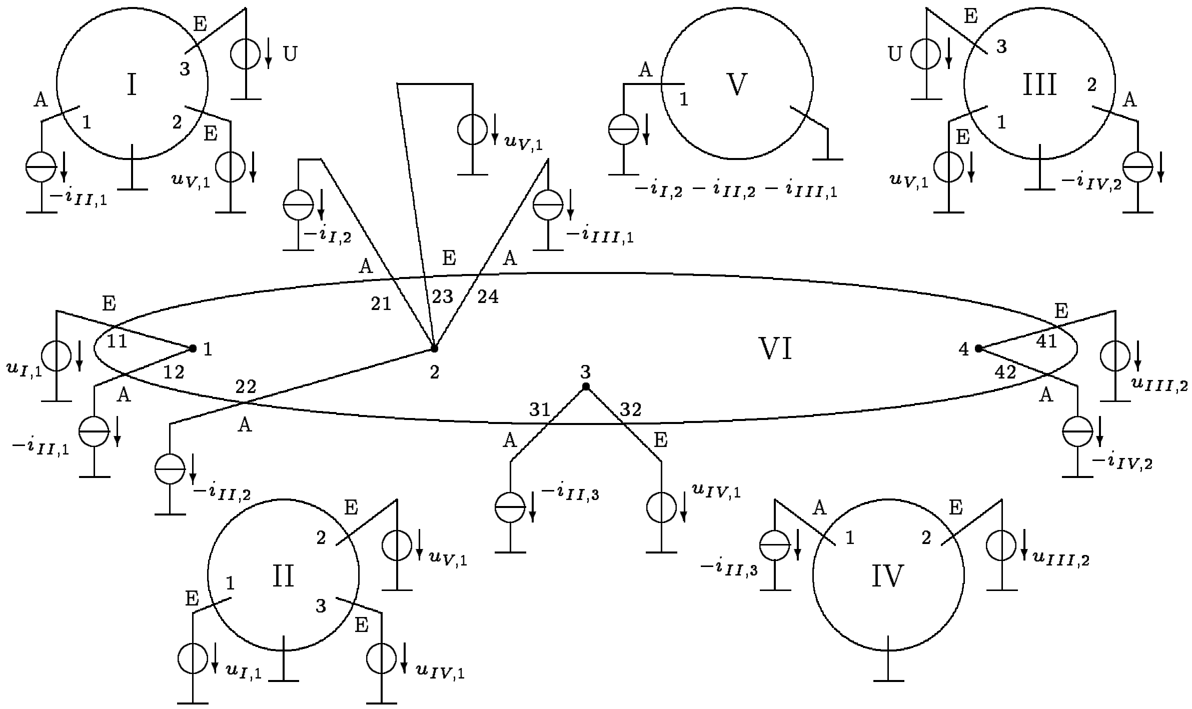


Abbildung 3.4: Netzwerkinterpretation des Entkopplungsansatzes

Ein Netzwerk mit unidirektionalem Signalfluß zwischen den Netzwerken der Blockebene läßt sich dadurch entkoppeln, daß an E–markierten Klemmen die Spannung mit Hilfe gesteuerter Spannungsquellen und an A–markierten Klemmen der Strom mit Hilfe gesteuerter Stromquellen vorgeschrieben wird. Weist man nun den eingeführten Netzwerkelementen als Darstellung der u–i–Relation die Klemmenspannung bzw. den Klemmenstrom der angeschlossenen Blocknetzwerkklemmen zu, dann kann man daraus direkt das in Abbildung 3.4 dargestellte Netzwerkdiagramm ableiten. Dabei erfolgt die Darstellung der u–i–Relation der gesteuerten Stromquellen mit Hilfe der u–i–Relationen der an den jeweiligen Hauptnetzwerkknoten angeschlossenen Netzwerkelemente.

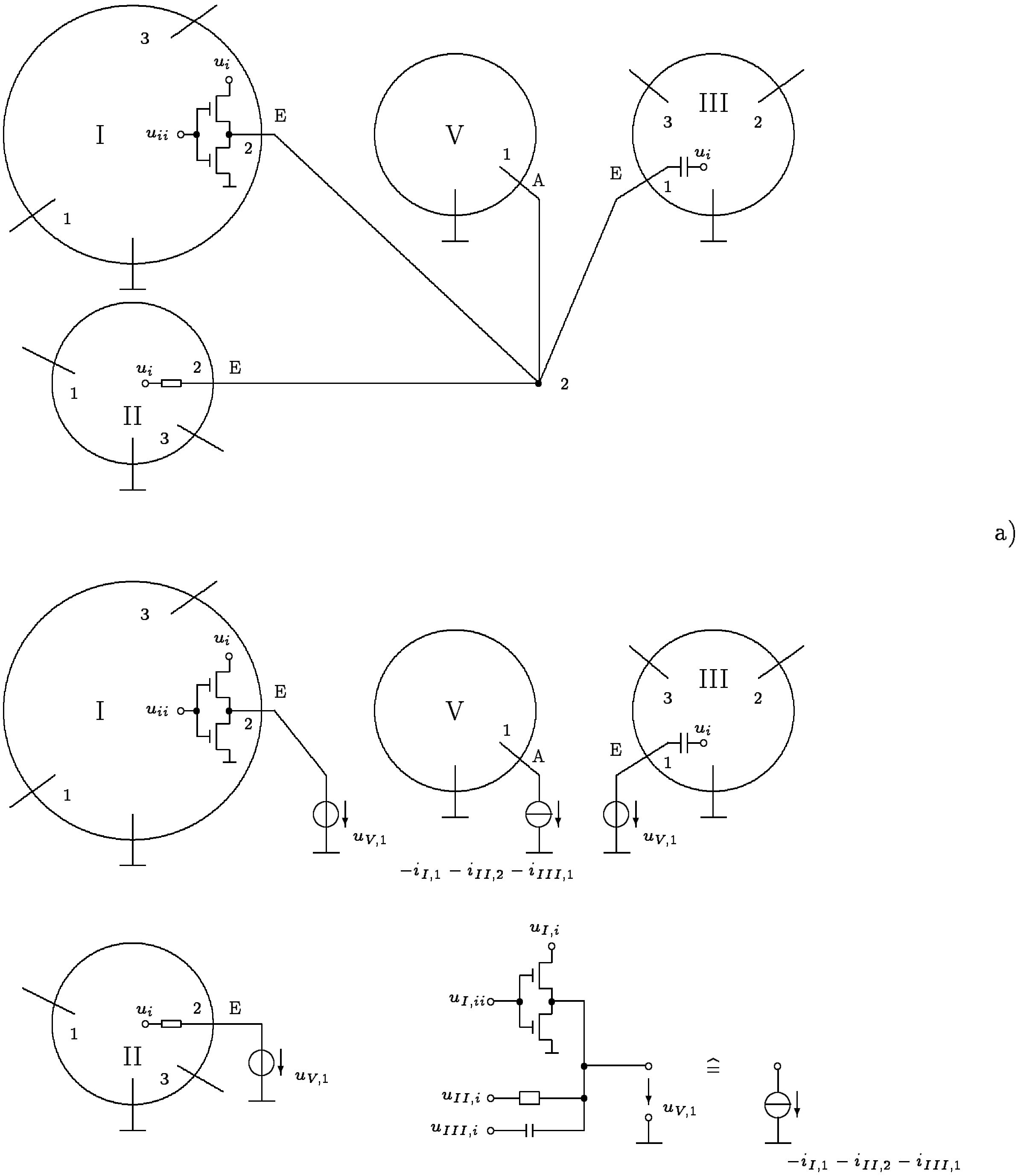


Abbildung 3.5: Approximation von Klemmenspannungen und -strömen

In Abbildung 3.5 wird diese Methode der Approximation des Klemmenstromes am Beispiel der Klemme 1 des Blocknetzwerkes V dargestellt. Auf diese Art und Weise wird der Klemmenstrom $i_{V,1}$ durch eine nichtlineare Funktion $i_{V,1} = g(u_{V,1})$, $g : \mathbb{R} \rightarrow \mathbb{R}$, der Klemmenspannung $u_{V,1}$ angenähert. Diese Schritte sind in Form eines automatisch ablaufenden Partitionierungsalgorithmus im Netzwerkanalyseprogramm MAGNUS implementiert und ermöglichen das Aufstellen eines allgemeinen strukturierten Netzwerkgleichungssystems, auf das neben Mehrebenen–Newtonverfahren auch Gauß–Seidel–Verfahren zur Lösung angewendet werden können.

3.1.3. *Netzwerk– und Modellbeschreibung.*

Elektrische Netzwerke werden im Programmsystem MAGNUS mit der Eingabesprache MSPICE beschrieben [BBG93]. MSPICE enthält alle wesentlichen Komponenten der Eingabesprache des Netzwerkanalyseprogrammes SPICE. MSPICE stellt eine Erweiterung der SPICE–Eingabesprache dar. Sie ermöglicht die für MAGNUS notwendige Beschreibung der Hierarchie des Netzwerkes einschließlich des Signalflusses. Die Hierarchie von Teilnetzwerken löst der MSPICE–Compiler bis auf jene auf, die als **Stufennetzwerke** gekennzeichnet wurden.
Alle Netzwerke der Block– und Basisebene sind Stufennetzwerke und werden dadurch gekennzeichnet, daß die Klemmen als Eingänge (E), Ausgänge (A) und als unbestimmt (U) bezüglich des Signalflusses gekennzeichnet werden. Andere hierarchische Strukturen von Teilnetzwerken werden ohne Zuordnung einer Signalflußrichtung notiert und von MSPICE aufgelöst.
Die Darstellung der Modelle wurde so geändert, daß auch beliebige Nutzermodelle einbezogen werden können. Die Steueranweisungen wurden an das Netzwerkanalyseprogramm MAGNUS angepaßt. Ein MSPICE–Text besteht aus der Titelanweisung, Elementanweisungen, Modellanweisungen und Steueranweisungen [BBG93]. Die letzte Anweisung ist die .END–Anweisung.

Die Elementanweisungen beschreiben die Netzwerkelemente, aus denen das Netzwerk besteht. Es können Widerstände (R), lineare Kapazitäten (C), lineare Induktivitäten (L), unabhängige Spannungsquellen (V), unabhängige Stromquellen (I), Halbleiterbauelemente (D, Q, J, M) und Teilnetzwerke (X) sein.
Die Modellanweisungen (.MODEL) dienen zur Spezifikation der benutzten Halbleiterbauelemente. Neben den Standard–Modellen im Programmsystem MAGNUS sind hier auch zusätzliche Nutzer–Modelle angebbar.

a) Teilnetzwerke

Gleiche mehrfach auftretende Netzwerkstrukturen bzw. Block– oder Basisnetzwerke werden als Teilnetzwerke notiert und können dann an den entsprechenden Stellen in das Netzwerk eingebettet werden. Es ergibt sich der prinzipielle Aufbau:

> **. SUBCKT** *subname ausknoten*
>
> Beschreibung des Teilnetzwerkes
>
> **. ENDS** *subname.*

Die Einbettung in das Netzwerk erfolgt durch die X–Anweisung:

> **X***name anknoten subname.*

subname bezeichnet das einzubettende Teilnetzwerk und *name* ist der Einbettungsplatzname des Teilnetzwerkes. *anknoten* steht für die Liste der Knoten, die mit der im Teilnetzwerkkopf definierten Liste der Klemmenknoten (*ausknoten*) identifiziert werden.

Das eingebettete Teilnetzwerk *subname* wird wie ein Pseudoschaltelement behandelt. Die Einbettung von Teilnetzwerken kann verschachtelt sein, sie darf jedoch nicht rekursiv sein. Alle Block– und Basisnetzwerke müssen in MSPICE als Stufennetzwerke gekennzeichnet werden. Stufennetzwerke werden vom MSPICE–Compiler nicht eingebettet. Der Simulationsmodus Elektrik oder Timing [BBG89a] wird in der Kopfzeile der Stufennetzwerkdefinition durch die Angabe /E bzw. /T unmittelbar hinter *subname* festgelegt. Fehlt diese Angabe, wird der Elektrik–Modus benutzt.

b) Schaltelemente

Mit MSPICE können verschiedene passive Schaltelemente (Widerstand, lineare Kapazität, lineare Induktivität) beschrieben werden. Die entsprechenden Elementanweisungen beginnen mit einem Buchstaben, der den Typ des Schaltelements kennzeichnet. Bezeichnet *kn_a* den Startknoten und *kn_b* den Zielknoten eines Zweipols, so notiert man zum Beispiel mit den folgenden Anweisungen die Netzwerkelemente:

- Widerstand: **R***name kn_a kn_b wert,*
- lineare Kapazität: **C***name kn_a kn_b wert,*
- lineare Induktivität: **L***name kn_a kn_b wert.*

Weiter können ideale unabhängige Spannungs– und Stromquellen in MSPICE notiert werden:

- Spannungsquelle: **V***name kn_a kn_b zeitverhalten,*
- Stromquelle: **I***name kn_a kn_b zeitverhalten.*

Für die zeitlich konstante Quelle wird bei *zeitverhalten* nur ein Wert angegeben. Das Zeitverhalten veränderlicher Quellen kann durch eine Anzahl von Funktionen beschrieben werden, wie Pulsquellen durch PULSE oder PLS, Sinusquellen durch SIN oder M_SIN, Exponentialquellen durch EXP, Polygonquellen durch PWL oder LINPOL. Nutzerspezifische Funktionen können eingebunden werden. Dazu sind eine Funktionsbeschreibung und eine entsprechende Funktionsroutine notwendig [BBG93].

c)Halbleiterbaulemente

In MSPICE können Modelle für vier verschiedene Halbleitertypen notiert werden. Die entsprechenden Anweisungen beginnen wie bei den Schaltelementen mit einem Buchstaben als Kennzeichen:

- Diode:
  D*name kn_a kn_b mname pname1 = pwert1 ...,*
- Bipolartransistor:
  Q*name kn_c kn_b kn_e <kn_s> mname pname1 = pwert1 ...,*
- Sperrschicht–Feldeffekttransistor:
  J*name kn_d kn_g kn_s mname pname1 = pname2 ...,*
- MOS–Feldeffekttransistor:
  M*name kn_d kn_g kn_s kn_b mname pname1 = pname2 ... .*

In diesen Elementanweisungen sind neben der Verschaltung die in der Schaltung verwendeten Modellnamen und eventuell Parameterveränderungen notiert. Die Zuordnung der Modellnamen *mname* zu den im Simulator eingebauten Modellen und deren Parameterisierung wird in zugeordneten Modell–Anweisungen beschrieben:

.MODEL *mname modell* <*pname1* = *pwert1 ...*>.

Unter *modell* muß der Zuordnungsname des vom Simulator zu verwendeten Modells stehen. In MSPICE und MAGNUS sind 4 MOS–Standardmodelle eingebaut:

- n–MOS Modelle MOSE und MOSD,
- CMOS–Modelle TN und TP.

Weitere Modelle können vom Nutzer eingebunden werden. Dazu sind die Modellbeschreibung und die entsprechenden Modellroutinen nötig [BBG90]. Für ein und denselben Modellnamen können in der zugeordneten Modellbeschreibung mehrere Modellroutinen durch den LEVEL–Parameter zugeordnet werden. Dadurch ist eine Simulation der gleichen Modelle mit unterschiedlicher Genauigkeit möglich.

d) Steueranweisungen

Mit diesen Anweisungen wird die Simulation gesteuert. Mit der .OPTIONS–Anweisung können Parametergrößen festgelegt bzw. verändert werden. Der Nutzer kann damit in den Programmablauf eingreifen, um zum Beispiel Konvergenzschwierigkeiten zu beseitigen, höhere Genauigkeiten zu erreichen oder Standardwerte zu verändern.

Mit der .OP–Anweisung kann der Ablauf der Berechnung des Gleichstrom–Arbeitspunktes beeinflußt werden. Die Berechnung kann durch eine dynamische Anfangswertberechnung mit dem Relaxationsverfahren (Standardverfahren) bzw. mit dem Mehrebenenverfahren erfolgen. Eine Anfangswertberechnung mit der Vorgabe der Anfangswerte mit Hilfe der .NODESET–Anweisung ist ebenfalls möglich.

Zur Analyse von Einschwingvorgängen bei linearen bzw. nichtlinearen elektrischen Netzwerken Schaltungen berechnet das Programmsystem MAGNUS schrittweise die Größe der Netzwerkvariablen als Funktion der Zeit. In der .TRAN–Anweisung können das verwendete Verfahren, das Zeitintervall für die Berechnung und die Schrittweite für die Ausgabe der Ergebniswerte angegeben werden. Als Verfahren können ein Relaxationsverfahren und ein Mehrebenenenverfahren benutzt werden.

Mit Hilfe der .LEVEL–Anweisung kann die benutzte Approximation der u–i–Relation von Modellen gesteuert werden. Je Stufennetzwerk und Modell kann eine Levelstufe festgelegt werden. Dadurch können verschiedene Varianten der u–i–Relation für Modelle mit gleichem Namen je Block– bzw. Basisnetzwerk gewählt werden.

Während der Simulation werden Werte für die in der .OUTLIST–Anweisung spezifizierten Netzwerkgrößen (Knotenspannungen, Zweigspannungen, Leistungen, Ströme) in eine Ausgabedatei geschrieben. Während bzw. nach der Analyse kann diese Ausgabedatei ausgewertet werden.

Um bei mehrfacher Simulation eines Netzwerkes die Berechnung der Anfangslösung einzusparen, sind in der MAGNUS–Steuerung spezielle Anweisungen vorgesehen. Sie wirken in der Reihenfolge der Steueranweisungen auf den unmittelbar vorher berechneten Kennlinienpunkt. Die .STORE–Anweisung speichert alle für eine Weiterrechnung von diesem Punkt aus notwendigen Größen des Netzwerkes. Durch die .LOAD–Anweisung werden diese gespeicherten Daten wieder eingelesen.

3.1.4. *Beispiel.*

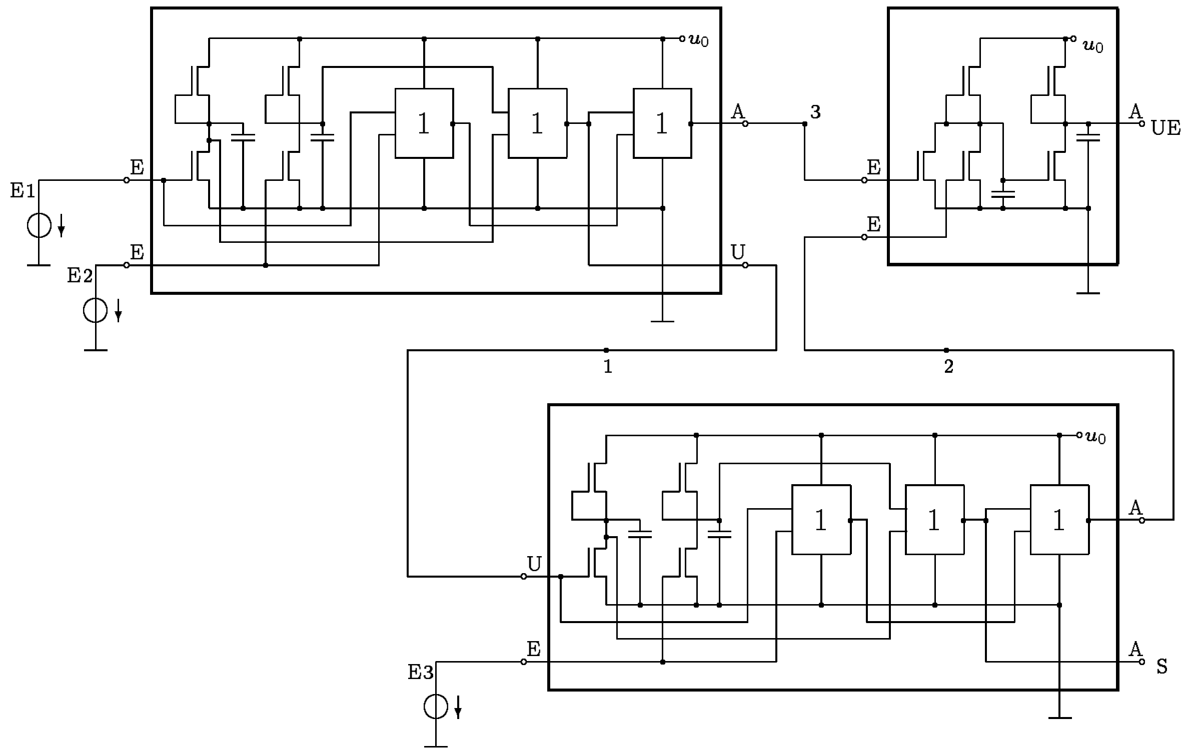


Abbildung 3.6: Schaltungsbeispiel Volladder

In Abbildung 3.6 ist eine einfache Volladderschaltung unter Verwendung von INVERTER– und NOR–Gattern dargestellt. Es wurde eine 3stufige Hierarchie mit drei Blocknetzwerken und einem Basisnetzwerk gewählt.

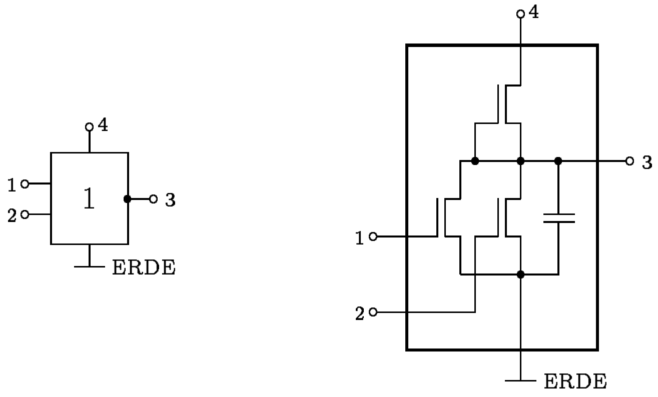


Abbildung 3.7: Basisnetzwerk NOR–Gatter

Als Basisnetzwerk dient das NOR–Gatter aus Abbildung 3.7. Die Klemmen der Halbadderschaltungen wurden bezüglich eines willkürlich vorgegebenen Signalflusses mit den Marken E, A und U markiert. Die Notation der Schaltung im MSPICE–Format ist aus Abbildung 3.8 ersichtlich. Das Beispiel wurde in eine Hierarchie von Teilnetzwerken zerlegt.

```
Beispiel fuer Report, Volladder
* Hauptnetzwerk
XB1 1 2 4 5 BHA1
XB2 5 3 6 7 BHA2
XB3 4 6 8 BOR
V1 1 0 PWL(TAB1 -1000 5 100 5 101 0 1000 0)
V2 2 0 PWL(TAB2 -1000 0 50 0 51 5 150 5 151 0 1000 0)
V3 3 0 PWL(TAB3 -1000 5 1000 5)
* Blocknetzwerk Halbadder 1
.SUBCKT BHA1 E(E1 E2) A(UE) U(S)
XS1 E1 E2 UE S UHA
.ENDS BHA1
* Blocknetzwerk Halbadder 2
.SUBCKT BHA2 U(E1) E(E2) A(UE S)
XS1 E1 E2 UE S UHA
.ENDS BHA2
* Teilnetzwerk Halbadder
.SUBCKT UHA E1 E2 UE S
XS1 E1 2 1 INV
XS2 E2 3 1 INV
XS3 E1 E2 4 NOR
XS4 3 2 UE NOR
XS5 UE 4 S NOR
VU 1 0 5.
.ENDS UHA
* Blocknetzwerk Orgatter
.SUBCKT BOR E(E1 E2) A(A)
MQ1 A1 1 A1 0 MOSD
MQ2 E1 A1 0 0 MOSE
MQ3 E2 A1 0 0 MOSE
XS2 A1 A 1 INV
VU 1 0 5.
.ENDS BOR
* Teilnetzwerk Inverter
.SUBCKT INV E A US
MQ1 A US A 0 MOSD
MQ2 E A 0 0 MOSE
C A 0 0.05
.ENDS INV
* Basisnetzwerk Norgatter
.SUBCKT NOR U(E1 E2 A)
MQ1 A 1 A 0 MOSD
MQ2 E1 A 0 0 MOSE
MQ3 E2 A 0 0 MOSE
CC A 0 0.05
V1 1 0 5.
.ENDS NOR
* Transistoren
* ENHANCEMENT-TRANSISTOR
.MODEL MOSE MOSE (L=8 B=60 UET0=0.48)
* DEPLETION-TRANSISTOR
.MODEL MOSD MOSD (B=8 L=8)
.OPTIONS MEMORY=800
.OUTLIST V(* *.*)
.OPTIONS NITN=5 EPSD=0.05 HMAX=100 XPMAX=1.E-4
.OP $T
.STORE
.OPTIONS CPUZYK=0.1 PROT=1
.TRAN T 200 0 1
.LOAD
.TRAN M 200 0 1
.END
```

Abbildung 3.8: MSPICE–Eingabetext zur Schaltung aus Abbildung 3.6

Die Netzwerke BHA1, BHA2, BOR und NOR sind als Stufennetzwerke ausgezeichnet. Die Teilnetzwerke INV und UHA werden durch den MSPICE–Compiler aufgelöst und in die durch die Stufennetzwerke definierte Netzwerkstruktur eingegliedert.

Der MSPICE–Text enthält weiterhin eine Reihe von Anweisungen für das Netzwerkanalyseprogramm MAGNUS zur Auswahl und Steuerung der Analysearten sowie zur Definition der verwendeten Transistormodelle.

Nach einer DC–Analyse (.OP \$T) wird der berechnete Arbeitspunkt gespeichert (.STORE). Die nachfolgende dynamische Analyse wird zunächst mit dem Standardverfahren (.TRAN T) im Intervall [0ns, 200ns] durchgeführt und danach mit einem Mehrebenen–Newtonverfahren (.TRAN M) wiederholt, wobei zuvor die Lösung des berechneten Arbeitspunktes als Anfangswert durch den Befehl .LOAD von einer Datei wieder hergestellt wurde.

Vor Erzeugung der Netzwerkgleichungen überführt ein in Abschnitt 3.2.2. beschriebener Algorithmus den bidirektionalen Signalfluß zwischen den Halbadder–Blocknetzwerken in einen unidirektionalen Signalfluß (wie in Abbildung 3.9 gezeigt).

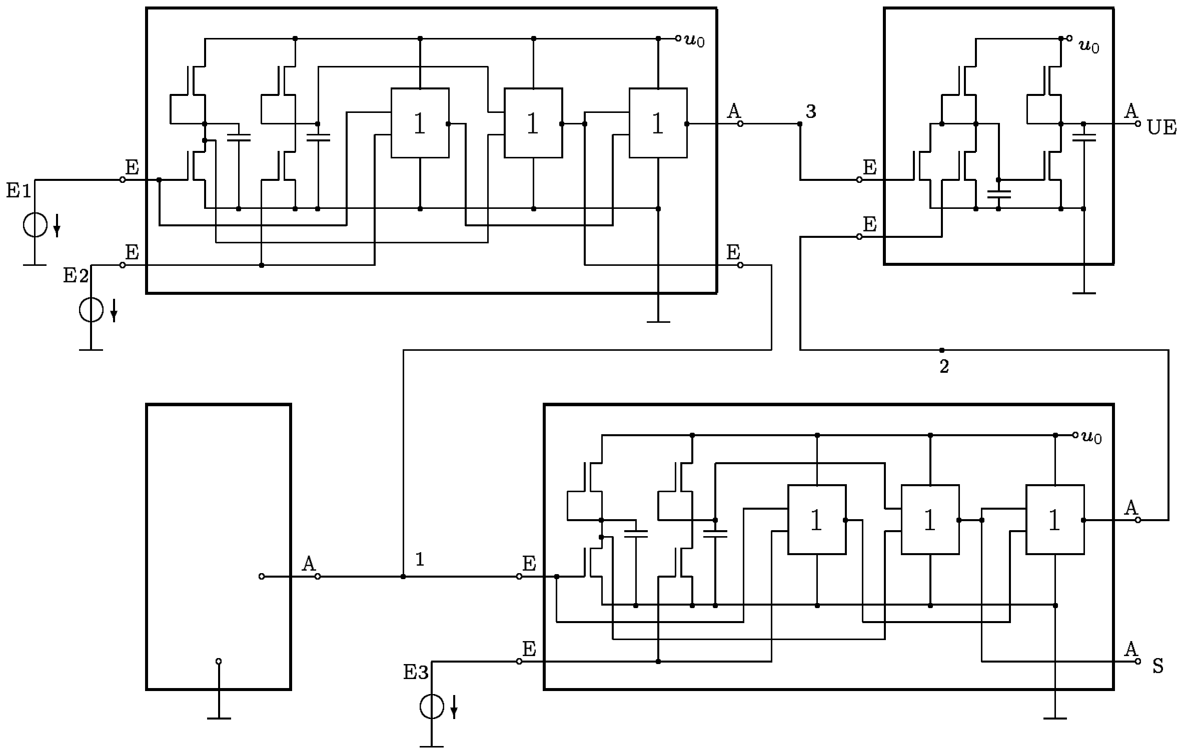


Abbildung 3.9: Überführung des Signalflusses

Eine Approximation der Klemmenspannungen und –ströme an den Blocknetzwerkklemmen führt auf die in Abbildung 3.10 dargestellte Entkopplung. Zur Zusammenfassung der Gleichungssystemstrukturen der Blocknetzwerke wird (wie in Abschnitt 3.2.2. beschrieben) auch die Struktur des Hauptnetzwerkes (vgl. Abbildung 3.11) mit entsprechenden Netzwerkstrukturen ergänzt.

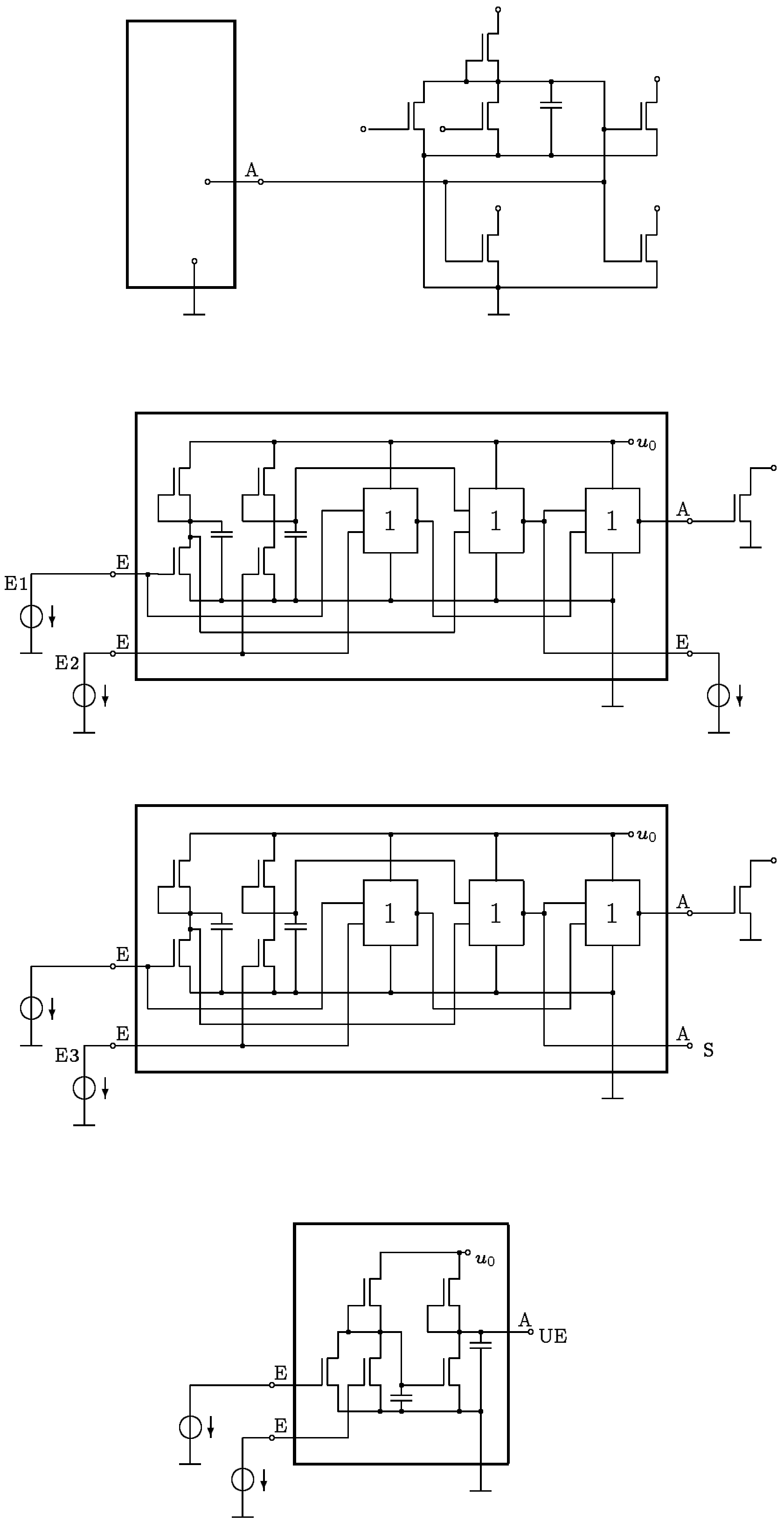


Abbildung 3.10.: Entkopplung der Blocknetzwerkstruktur

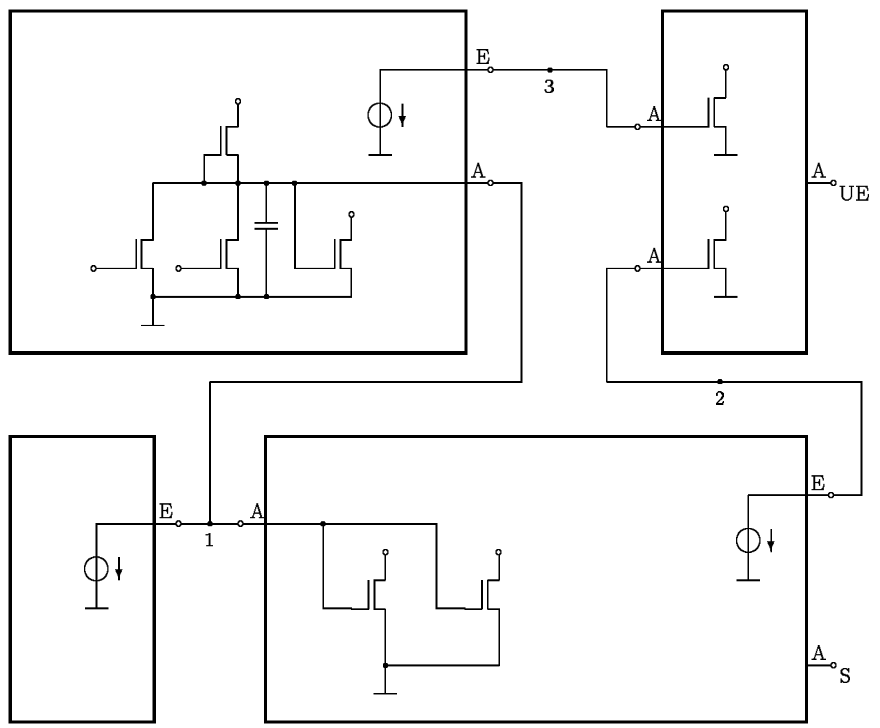


Abbildung 3.11: Ergänzte Hauptnetzwerkstruktur mit dualer Markierung der Blocknetzwerkklemmen

3.2. Netzwerkgleichungssystem.

In diesem Abschnitt wird die Hierarchie des Gleichungssystems des Netzwerkanalyseprogramms MAGNUS beschrieben. Dazu stelle man sich eine Netzwerkstruktur in 3 Ebenen vor, die auf eine topologische hierarchische Beschreibung zurückzuführen ist (siehe Abbildung 3.1). Ein Hauptnetzwerk mit s Netzwerken in der Blockebene ($s > 0$), wobei jedem Blocknetzwerk r_i Netzwerke der Basisebene ($r_i \geq 0,\ i = 1, \ldots s$) zugeordnet werden können, wird diesen Anforderungen gerecht.
Das Gleichungssystem des Netzwerkanalyseprogramms MAGNUS wird für jede Hierarchiestufe (Haupt–, Block–, Basisebene) nach der Methode der erweiterten Knotenspannungsanalyse aufgestellt. Die Variablen der Gleichungssysteme der Block– und Basisebene können grundsätzlich in innere und äußere Variable aufgeteilt werden. Das Gleichungssystem des Hauptnetzwerkes hat nur innere Variable.

3.2.1. *Basisebene.*

Innere Variable der Gleichungssysteme der Basisnetzwerke sind alle Knotenspannungen u_{ij}^k und solche Zweigströme i_{ij}^z von Netzwerkelementen, für deren Darstellung der u–i–Relation keine Admittanzdarstellung angegeben werden kann. Die inneren Variablen werden im Vektor x_{ij} zusammengefaßt:

$$x_{ij} := \left(u_{ij}^k, i_{ij}^z\right)^T \quad \text{mit} \quad j = 1, \ldots, r_i;\ i = 1, \ldots, s.$$

Äußere Variable sind alle Paare von Klemmenspannungen u_{ij} und Klemmenströmen i_{ij} der Basisnetzwerke, die im Vektor

$$y_{ij} := (u_{ij}, i_{ij})^T \quad \text{mit} \quad j = 1, \ldots, r_i;\ i = 1, \ldots, s.$$

zusammengefaßt sind. Als Klemmenspannungen werden dabei die Knotenspannungen der Randknoten des Basisnetzwerkes bezeichnet. Ein Klemmenstrom bezeichnet die Summe der Zweigströme eines Basisnetzwerkes bezüglich eines Randknotens. Demnach werden mit Hilfe der Klemmenströme Hilfsvariable definiert, die für das Zusammenwirken der Teilgleichungssysteme benötigt werden. Damit läßt sich das Gleichungssystem für Basisnetzwerke in der Form

$$0 = F_{ij}\left(x_{ij}, \dot{x}_{ij}, y_{ij}, \dot{y}_{ij}, t\right) \tag{3.1}$$

$$F_{ij} : \mathbb{R}^{p_{ij}} \times \mathbb{R}^{p_{ij}} \times \mathbb{R}^{2q_{ij}} \times \mathbb{R}^{2q_{ij}} \times \mathbb{R} \to \mathbb{R}^{p_{ij}+q_{ij}}$$

aufschreiben. Das Gleichungssystem (3.1) ist etwas allgemeiner als das im Programmsystem MAGNUS tatsächlich verwendete System:

$$0 = \tilde{F}_{ij}\left(x_{ij}, \dot{x}_{ij}, u_{ij}, \dot{u}_{ij}, i_{ij}, t\right)$$

$$\tilde{F}_{ij} : \mathbb{R}^{p_{ij}} \times \mathbb{R}^{p_{ij}} \times \mathbb{R}^{q_{ij}} \times \mathbb{R}^{q_{ij}} \times \mathbb{R}^{q_{ij}} \times \mathbb{R} \to \mathbb{R}^{p_{ij}+q_{ij}},$$

da zeitliche Ableitungen der Klemmenströme als Netzwerkvariable im Programmsystem MAGNUS nicht vorkommen. Gründe dafür sind Einschränkungen an die u–i–Relationen von Netzwerkelementen, die mit Randknoten korrespondieren und die fehlende Beschreibungsmöglichkeit von Klemmenströmen innerhalb der Netzwerkeingabe. Aus Gründen der einfacheren Darstellung werden wir im weiteren grundsätzlich Gleichungsdarstellungen der Form (3.1) betrachten.

3.2.2. *Blockebene.*

Blocknetzwerke können Basisnetzwerke enthalten. Deshalb sind zu den inneren Variablen der Blocknetzwerke, neben den Knotenspannungen u_i^k an inneren Knoten und zusätzlichen Zweigströmen i_i^z, auch die Paare der Klemmenspannungen und –ströme der angeschlossenen Basisnetzwerke y_{ij} zu rechnen. Die inneren Variablen werden im Vektor x_i zusammengefaßt:

$$x_i := \left(u_i^k, i_i^z, y_{i1}, \ldots, y_{ir_i}\right)^T \quad \text{mit} \quad j = 1, \ldots, r_i;\ i = 1, \ldots, s.$$

Äußere Variable sind alle Paare von Klemmenspannungen u_i und Klemmenströme i_i der Blocknetzwerke, die im Vektor

$$y_i := (u_i, i_i)^T \quad \text{mit} \quad i = 1, \ldots, s$$

zusammengefaßt sind. Damit hat das Gleichungssystem für Blocknetzwerke die Form:

$$0 = G_i\left(x_i, \dot{x}_i, y_{i1}, \dot{y}_{i1}, \ldots, y_{ir_i}, \dot{y}_{ir_i}, y_i, \dot{y}_i, t\right), \tag{3.2}$$

$$\begin{aligned} G_i : \mathbb{R}^{p_i} \times \mathbb{R}^{p_i} \times \mathbb{R}^{2q_{i1}} \times \mathbb{R}^{2q_{i1}} \times \cdots \times \\ \times \mathbb{R}^{2q_{ir_i}} \times \mathbb{R}^{2q_{ir_i}} \times \mathbb{R}^{2q_i} \times \mathbb{R}^{2q_i} \times \mathbb{R} \to \mathbb{R}^{p_i+q_i+\sum_{j=1}^{r_i} q_{ij}} \end{aligned}.$$

Im Abschnitt 3.1.2. wurde beschrieben, wie mit Hilfe einer automatischen Partitionierungsmethode der im allgemeinen bidirektionale Signalfluß zwischen den Blocknetzwerken auf einen unidirketionalen Signalfluß zurückgeführt wird. Die aus diesem Prozeß resultierende E–A–Markierung der Klemmen wird beim Aufstellen von zusätzlichen Gleichungen für Netzwerke der Blockebene ausgenutzt, so daß für jedes Blocknetzwerk m_i zusätzliche Gleichungen erzeugt werden (vgl. dazu Abbildung 3.5 bzw. Abbildung 3.10):

(i) Randknoten mit E–Markierung:

$$0 = u - \tilde{U}.$$

Der Wert $\tilde{U}$ wird dabei von der Spannung jenes A–Knotens bestimmt, mit dem dieser E–Knoten korrespondiert.

(ii) Randknoten mit A–Markierung:

$$0 = i - g^* \left(u, \tilde{U}_1, \ldots, \tilde{U}_k.\right)$$

Die Funktion g bezeichnet hier die Summe aller Klemmenströme der korrespondierenden E–Knoten. Mit $\tilde{U}_1, \ldots, \tilde{U}_k$ bezeichnen wir die Spannungen an den inneren Knoten der korrespondierenden Blocknetzwerke, die nötig sind, um Zweigströme angeschlossener Netzwerkelemente und daraus den resultierenden Klemmenstrom zu berechnen. Der Wert der Spannung der korrespondierenden E–Knoten wird durch die Klemmenspannung u des A–Knotens ersetzt.

Diese Gleichungen werden im System

$$0 = \hat{G}_i \left(y_i, \dot{y}_i, t\right) \qquad (3.3)$$
$$\hat{G}_i : \mathbb{R}^{2q_i} \times \mathbb{R}^{2q_i} \times \mathbb{R} \to \mathbb{R}^{q_i}$$

zusammengefaßt. Aus den obengenannten Gründen können auch in diesem System keine Ableitungen der Klemmenströme auftreten.

Werden Blocknetzwerke durch das Gleichungssystem $\begin{bmatrix} G_i \\ \hat{G}_i \end{bmatrix}$ beschrieben, dann ist es möglich, ein Iterationsverfahren vom Gauß–Seidel–Typ bezüglich der Systeme der Blockebene zur Lösung des Netzwerkgleichungssystems zu definieren (vgl. Abschnitt 4.2.3.2.). Variable des Iterationsprozesses sind die Spannungen $\tilde{U}, \tilde{U}_1, \ldots, \tilde{U}_k$ aus den zusätzlichen Gleichungen Z_i der Blocknetzwerke.

3.2.3. *Hauptebene.*

Das Hauptnetzwerk muß mindestens ein Blocknetzwerk enthalten. Unter Beachtung der Aussagen von Abschnitt 3.2. besteht das Gleichungssystem des Hauptnetzwerkes nur aus linearen Gleichungen, die aus den Kirchhoffschen Gesetzen gewonnen werden und die topologische Struktur der Zusammenschaltung der Blocknetzwerke beschreiben. Somit gilt:

$$0 = H \left(y_1, \ldots, y_s, t\right) \qquad (3.4)$$
$$H : \mathbb{R}^{2m_1} \times \cdots \times \mathbb{R}^{2m_s} \times \mathbb{R} \to \mathbb{R}^{\sum_{i=1}^{s} m_i}.$$

Die Variablen des Hauptnetzwerkes werden dabei durch die Paare der Klemmenspannungen u_i und der Klemmenströme i_i der Blocknetzwerke gebildet, die im Vektor $y_i := \left(u_i, i_i\right)^T$ $(i = 1, \ldots, s)$ für jedes Blocknetzwerk zusammengefaßt sind.
Die numerischen Verfahren des Netzwerkanalyseprogramms MAGNUS sind so implementiert, daß automatisch versucht wird, Konvergenzprobleme beim Iterationsprozeß des Gauß–Seidel–Verfahrens dadurch zu beheben, daß die „nichtkonvergenten“ Gleichungssysteme der Blockebene mit Hilfe zusätzlicher Gleichungen des Hauptnetzwerkgleichungssystems und durch Anwendung eines Mehrebenen–Newtonverfahrens behandelt werden.

Im Abschnitt 3.1.2. wurde beschrieben, wie eine Struktur des Hauptnetzwerkes definiert wird sowie Randknoten der Zusammenschaltungsstruktur des Hauptnetzwerkes definiert und mit Markierungen versehen werden. Die aus diesem Prozeß resultierende E–A–Markierung der Klemmen des Hauptnetzwerkes wird beim Aufstellen von zusätzlichen Gleichungen ausgenutzt, indem insgesamt $\sum_{i=1}^{s} m_i$ zusätzliche Gleichungen ergänzt werden (vgl. dazu Abbildung 3.3 bzw. Abbildung 3.11):

(i) Randknoten mit E–Markierung:

$$0 = u - \tilde{U}.$$

Der Wert $\tilde{U}$ wird dabei von der Spannung des A–Knotens jenes Blocknetzwerks bestimmt, mit dem dieser E–Knoten verbunden ist.

(ii) Randknoten mit A–Markierung:

$$0 = i - g^* \left(u, \tilde{U}_1, \ldots, \tilde{U}_k\right).$$

Die Funktion g bezeichnet hier den Klemmenstrom des korrespondierenden Blocknetzwerk E–Knotens. Mit $\tilde{U}_1, \ldots, \tilde{U}_k$ bezeichnen wir die Spannungen an den inneren Knoten des korrespondierenden Blocknetzwerkes, die nötig sind, um Zweigströme angeschlossener Netzwerkelemente und daraus den resultierenden Klemmenstrom zu berechnen. Mit dem Symbol u wird der Wert der Klemmenspannung des A–Knotens bezeichnet.

Damit sind für jedes Blocknetzwerk Gleichungen der Form

$$\begin{gathered} 0 = \hat{H}_i\left(y_i, \dot{y}_i, t\right) \\ \hat{H}_i : \mathbb{R}^{2q_i} \times \mathbb{R}^{2q_i} \times \mathbb{R} \to \mathbb{R}^{q_i} \end{gathered} \tag{3.5}$$

angebbar, die bei der Zusammenfassung von Blocknetzwerken deren Klemmeneigenschaften auf die Zusammenschaltungsstruktur übertragen und auf diese Art und Weise die Fortsetzung der Iterationsverfahren mit einem modifizierten Gleichungssystem gestatten.

3.2.4. *Beispiel.*

Für das im Abschnitt 3.1.4. eingeführte Volladder–Netzwerk wird jetzt ein hierarchisches Gleichungssystem angegeben. Alle u–i–Relationen der Transistor– und Kapazitätsmodelle sind in Admittanzform darstellbar, so daß für diese Modelle keine Zweigströme als zusätzliche Variable auftreten. Zur Vereinfachung der Schreibweise der u–i–Relation eines Transistormodells Q_j wird der folgende Vektor der Klemmenspannungen

$$u_{Qj} := \left(u_{Qj}^G, u_{Qj}^D, u_{Qj}^S, u_{Qj}^B\right)$$

eingeführt und damit die nichtlinearen Funktionen der Klemmenströme des Transistormodells

$$\begin{aligned} i_{Qj}^G &:= g_G\left(u_{Qj}, \dot{u}_{Qj}\right) \\ i_{Qj}^D &:= g_D\left(u_{Qj}, \dot{u}_{Qj}\right) \\ i_{Qj}^S &:= g_S\left(u_{Qj}, \dot{u}_{Qj}\right) \end{aligned}$$

definiert.

Für die zum Bezugsknoten gerichteten Kapazitäten C wird die Darstellung

$$i_C = C\dot{u}$$

benutzt, wobei $\dot{u}$ die zeitliche Ableitung der Knotenspannung bezeichnet. Für ideale Spannungsquellen existiert keine Admittanzdarstellung der u–i–Relation. Folglich gilt:

$$u_E = U^0.$$

Das Potential des Bezugsknotens sei $u_M := 0V$.

Zur Definition der Klemmenspannungen und –ströme des Basisnetzwerkes NOR benutzen wir die Schreibweise $u_{i,j}$ für die Spannung an der j-ten Klemme des i-ten Basisnetzwerkes bzw. $i_{i,j}$ für den zugeordneten Klemmenstrom.

Aus Abbildung 3.7 kann man das Gleichungsystem des **Basisnetzwerkes NOR** ablesen. Die Klemmenspannungen der Transistormodelle werden in den Vektoren

$$\begin{aligned}
u_{Q1} &:= (u_1, U_0, u_1, u_M) \\
u_{Q2} &:= (u_2, u_3, u_M, u_M) \\
u_{Q3} &:= (u_3, u_3, u_M, u_M)
\end{aligned}$$

zusammengefaßt. Demnach gilt:

$$\begin{aligned}
0 &= g_G\left(u_{Q2}, \dot{u}_{Q2}\right) - i_1 \\
0 &= g_G\left(u_{Q3}, \dot{u}_{Q3}\right) - i_2 \\
0 &= g_G\left(u_{Q1}, \dot{u}_{Q1}\right) + g_S\left(u_{Q1}, \dot{u}_{Q1}\right) + g_D\left(u_{Q2}, \dot{u}_{Q2}\right) + g_D\left(u_{Q3}, \dot{u}_{Q3}\right) - C\dot{u}_3 - i_3.
\end{aligned}$$

Das **Blocknetzwerk BHA1** (vgl. Abbildung 3.10) wird durch 3 Aufrufe des Basisnetzwerkes NOR vervollständigt. Als Klemmenvariable des Netzwerkes BHA1 werden nur die Spannungen der Klemmen UE und S benötigt. Die Spannungen an den Klemmen E1 und E2 sind durch die angegebenen gleichnamigen Quellen vorgegeben. Die Klemmenspannungen der Transistormodelle werden in den Vektoren

$$\begin{aligned}
u_{Q1} &:= (u_{2,2}, U_0, u_{2,2}, u_M) \\
u_{Q2} &:= (u_{2,1}, U_0, u_{2,1}, u_M) \\
u_{Q3} &:= (u_{E1}, u_{2,2}, u_M, u_M) \\
u_{Q4} &:= (u_{E2}, u_{2,1}, u_M, u_M)
\end{aligned}$$

zusammengefaßt. Das Gleichungssystem hat somit die Form:

$$\begin{aligned}
0 &= u_{1,3} - u_{3,2} \\
0 &= u_{3,1} - u_{2,3} \\
0 &= u_{2,3} - u_S \\
0 &= u_{3,3} - u_{UE} \\
0 &= i_{1,3} + i_{3,2} \\
0 &= i_{2,3} + i_{3,1} - i_s \\
0 &= i_{3,3} - i_{UE} \\
0 &= g_G\left(u_{Q1}, \dot{u}_{Q1}\right) + g_S\left(u_{Q1}, \dot{u}_{Q1}\right) + g_D\left(u_{Q3}, \dot{u}_{Q3}\right) - C\dot{u}_{2,2} + i_{2,2} \\
0 &= g_G\left(u_{Q2}, \dot{u}_{Q2}\right) + g_S\left(u_{Q2}, \dot{u}_{Q2}\right) + g_D\left(u_{Q4}, \dot{u}_{Q4}\right) - C\dot{u}_{2,1} + i_{2,1}.
\end{aligned}$$

Aus Abbildung 3.10 liest man die **zusätzlichen** Gleichungen für den Entkopplungsansatz ab:

$$0 = i_{UE} + i_{Q3}^{G(BOR)}$$
$$0 = u_S - u^{(BZUS)}.$$

Für das Gleichungssystem des **Blocknetzwerkes BHA2** (vgl. Abbildung 3.10) werden die Variablen der Klemmen E1, UE und S benötigt. Die Spannung an der Klemme E2 ist durch die angegebene Quelle E3 vorgegeben. Die Klemmenspannungen der Transistormodelle werden in den Vektoren

$$\begin{aligned}
u_{Q1} &:= (u_{2,2}, U_0, u_{2,2}, u_M) \\
u_{Q2} &:= (u_{2,1}, U_0, u_{2,1}, u_M) \\
u_{Q3} &:= (u_{E1}, u_{2,2}, u_M, u_M) \\
u_{Q4} &:= (u_{E2}, u_{2,1}, u_M, u_M)
\end{aligned}$$

zusammengefaßt. Das Gleichungssystem hat die folgende Form:

$$\begin{aligned}
0 &= u_{1,3} - u_{3,2} \\
0 &= u_{3,1} - u_{2,3} \\
0 &= u_{1,1} - u_{E1} \\
0 &= u_{2,3} - u_S \\
0 &= u_{3,3} - u_{UE} \\
0 &= i_{1,3} + i_{3,2} \\
0 &= i_{3,3} - i_{UE} \\
0 &= i_{2,3} + i_{3,1} - i_s \\
0 &= g_G(u_{Q3}, \dot{u}_{Q3}) + i_{1,1} - i_{E1} \\
0 &= g_G(u_{Q1}, \dot{u}_{Q1}) + g_S(u_{Q1}, \dot{u}_{Q1}) + g_D(u_{Q3}, \dot{u}_{Q3}) - C\dot{u}_{2,2} + i_{2,2} \\
0 &= g_G(u_{Q2}, \dot{u}_{Q2}) + g_S(u_{Q2}, \dot{u}_{Q2}) + g_D(u_{Q4}, \dot{u}_{Q4}) - C\dot{u}_{2,1} + i_{2,1}
\end{aligned}$$

Aus Abbildung 3.10 liest man die **zusätzlichen** Gleichungen

$$\begin{aligned}
0 &= u_{E1} - u^{(BZUS)} \\
0 &= i_{UE} + i_{Q4}^{G(BOR)} \\
0 &= i_S + 0
\end{aligned}$$

des Entkopplungsansatzes ab.

Für das Gleichungssystem des **Blocknetzwerkes BOR** (vgl. Abbildung 3.10) werden die Variablen der Klemmen E1, E2 und UE sowie die innere Knotenspannung u benötigt. Die Klemmenspannungen der Transistormodelle werden in den Vektoren

$$\begin{aligned}
u_{Q1} &:= (u, U_0, u, u_M) \\
u_{Q2} &:= (u_{UE}, U_0, u_{UE}, u_M) \\
u_{Q3} &:= (u_{E1}, u, u_M, u_M) \\
u_{Q4} &:= (u_{E2}, u, u_M, u_M) \\
u_{Q5} &:= (u, u_{UE}, u_M, u_M)
\end{aligned}$$

zusammengefaßt.

Es gilt:

$$
\begin{aligned}
0 &= i_G\left(u_{Q3}, \dot{u}_{Q3}\right) - i_{E1} \\
0 &= i_G\left(u_{Q4}, \dot{u}_{Q4}\right) - i_{E2} \\
0 &= i_G\left(u_{Q2}, \dot{u}_{Q2}\right) + i_S\left(u_{Q2}, \dot{u}_{Q2}\right) + i_D\left(u_{Q5}, \dot{u}_{Q5}\right) - C\dot{u}_{UE} - i_{UE} \\
0 &= i_G\left(u_{Q1}, \dot{u}_{Q1}\right) + i_S\left(u_{Q1}, \dot{u}_{Q1}\right) + i_D\left(u_{Q3}, \dot{u}_{Q3}\right) + i_D\left(u_{Q4}, \dot{u}_{Q4}\right) + i_G\left(u_{Q5}, \dot{u}_{Q5}\right) - C\dot{u}.
\end{aligned}
$$

Aus Abbildung 3.10 liest man die **zusätzlichen** Gleichungen

$$
\begin{aligned}
0 &= u_{E1} - u_{UE}^{(BHA1)} \\
0 &= u_{E2} - u_{UE}^{(BHA2)} \\
0 &= i_{UE} - 0
\end{aligned}
$$

für den Entkopplungsansatz ab.
Dem **zusätzlichen Blocknetzwerk BZUS** wird für jede Klemme eine ideale Stromquelle mit dem Wert 0 zugewiesen (vgl. Abbildung 3.10). Klemmenspannung und Klemmenstrom werden mit u bzw. i bezeichnet. Demnach gilt:

$$0 = i + 0.$$

Für den Entkopplungsansatz gilt als zusätzliche Gleichung:

$$0 = i + i_G^{Q3(BHA2)} + i_{1,1}^{(BHA2)} + i_{3,1}^{(BHA1)} + i_{2,3}^{(BHA1)}.$$

Wie in Abbildung 3.10 dargestellt, werden die Klemmenströme der hier genannten Basisnetzwerke durch die Klemmenströme der entsprechenden Transistormodelle ersetzt.
Das Gleichungssystem des **Hauptnetzwerkes** kann in Form des Kirchhoffschen Stromgesetzes und des Kirchhoffschen Spannungsgesetzes aus Abbildung 3.9 abgelesen werden:

$$
\begin{aligned}
0 &= i_{UE}^{(BHA1)} + i^{(BZUS)} + i_{E1}^{(BHA2)} \\
0 &= i_{E2}^{(BOR)} + i_{UE}^{(BHA2)} \\
0 &= i_S^{(BHA1)} + i_{E1}^{(BOR)} \\
0 &= i_{UE}^{(BOR)} \\
0 &= i_S^{(BHA2)} \\
0 &= u^{(BZUS)} - u_{UE}^{(BHA1)} \\
0 &= u^{(BZUS)} - u_{E1}^{(BHA2)} \\
0 &= u_S^{(BHA1)} - u_{E1}^{(BOR)} \\
0 &= u_{UE}^{(BHA2)} - u_{E2}^{(BOR)}.
\end{aligned}
$$

Für das Zusammenfassen von Blocknetzwerken werden die zusätzlichen Gleichungen

$$
\begin{aligned}
0 &= u^{(BZUS)} - u \\
0 &= u_S^{(BHA1)} - u_{2,3}^{(BHA1)} \\
0 &= i_{UE}^{(BHA1)} - i_{3,1}^{(BHA1)} - i_{2,3}^{(BHA1)} \\
0 &= i_{E1}^{(BHA2)} - i_G^{Q3(BHA2)} + i_{1,1}^{(BHA2)} \\
0 &= u_S^{(BHA2)} - u_{2,3}^{(BHA2)} \\
0 &= u_{UE}^{(BHA2)} - u_{3,3}^{(BHA2)} \\
0 &= i_{E1}^{(BOR)} - i_{Q3}^{G(BOR)} \\
0 &= i_{E2}^{(BOR)} - i_{Q4}^{G(BOR)} \\
0 &= u_{UE}^{(BOR)} - u_{Q5}^{D(BOR)}
\end{aligned}
$$

für das Hauptnetzwerk notiert.

Auch in diesen Gleichungen werden (wie in Abbildung 3.10 dargestellt) die genannten Klemmenströme von Basisnetzwerken durch die zugeordneten Klemmenströme der Transistormodelle aus dem Netzwerk NOR ersetzt.

4. Netzwerkanalyse mit MAGNUS

4.1. Analyseverfahren.

Das Netzwerkanalyseprogramm MAGNUS dient zur **statischen und dynamischen Analyse** sehr großer hierarchisch strukturierter elektrischer Netzwerke.

Zu den **statischen Analysen** zählen die Berechnung des Gleichstromarbeitspunktes resistiver und dynamischer Netzwerke sowie die Kennlinienberechnung resistiver Netzwerke. Der Arbeitspunkt eines elektrischen Netzwerkes ist durch konstante Funktionen der Netzwerkvariablen gekennzeichnet, dessen Berechnung im Programmsystem MAGNUS mit Hilfe verschiedener Einbettungsverfahren erfolgt. Ein Verfahren regelt die Werte der Spannungs– und Stromquellen des Netzwerkes von Null auf ihre Nennwerte, die den Arbeitspunkt bestimmen. Ein anderes Verfahren arbeitet mit einer Homotopie bezüglich vorgegebener Anfangswerte von charakteristischen Netzwerkvariablen. Wenn ein Netzwerk mehrere Arbeitspunkte besitzt, kann nur derjenige gefunden werden, der durch die Anfangswerte festgelegt ist.
Bei dynamischen Netzwerken wird der Arbeitspunkt als Grenzwert eines Einschwingvorganges mit dem o.g. Hochregeln der Quellen berechnet. Ist die stationäre Lösung konstant, so ist ein Arbeitspunkt des Netzwerkes gefunden. Andere Möglichkeiten sind periodisches oder instabiles Lösungsverhalten. Arbeitspunkte dynamischer Netzwerke benötigt man im allgemeinen als Anfangswerte für nachfolgende dynamische Analysen.
Die Kennlinienberechnung resistiver Netzwerke wird auf eine dynamische Analyse mit zeitabhängigen Netzwerkparametern zurückgeführt.

Die **Analyse des dynamischen Verhaltens** elektrischer Netzwerke im Zeitbereich erfolgt im Programmsystem MAGNUS durch die Berechnung von Einschwingvorgängen bezüglich zeitabhängiger Netzwerkelemente (im allgemeinen Spannungs– und Stromquellen). Aufgrund der implementierten numerischen Verfahren ist die Berechnung periodischer Lösungen oder die Analyse von selbstschwingenden Netzwerken (Oszillatoren) nur in wenigen Ausnahmefällen möglich.

In Abschnitt 2.2. wurde die Erzeugung des Netzwerkgleichungssystems nach der Methode der erweiterten Knotenspannungsanalyse und die Interpretation des Systems als **Anfangswertaufgabe** beschrieben. Bei der Auswahl der numerischen Verfahren ist die Problematik des Index von Algebro–Differentialgleichungssystemen [Gea88], [Rei92] unbedingt zu berücksichtigen. Diese Untersuchungen sind gegenwärtig noch nicht abgeschlossen. Es sind jedoch Aussagen möglich, nachdem ladungs– und flußorientierte Modelle elektrischer Netzwerke, die in Gleichungssystemen benutzt werden, in die außerdem noch Spanungen und Ströme als Variable eingehen, zu Algebro–Differentialgleichungen mit einem Index größer 1 führen können. Für solche Gleichungssysteme sind die üblicherweise angewandten impliziten Diskretisierungsformeln nicht geeignet und führen in der Regel zu keinen oder unbrauchbaren Ergebnissen.
Im Netzwerkanalyseprogramm MAGNUS wurde bei der Modellierung der Kapazitäten und Induktivitäten auf eine ladungs– bzw. flußorientierte Formulierung der u–i–Relationen verzichtet.

Als Grundprinzip der Analyse großer strukturierter elektrischer Netzwerke mit 3 Hierarchiestufen (vgl. 3.1.), wie es im Netzwerkanalyseprogramm MAGNUS verwircklicht wurde, kann folgender Ablauf angesehen werden:

(i) Berechnung des Klemmenverhaltens aller Basisnetzwerke eines Blocknetzwerkes;

(ii) Analyse der Zusammenschaltung der Netzwerke der Basisebene mit dem zugeordneten Blocknetzwerk, wobei jedes Basisnetzwerk in Form eines Makromodells ersetzt wird und Berechnung des daraus resultierenden Klemmenverhaltens des Blocknetzwerkes;

(iii) Analyse des Gleichungssystems der Hauptebene, d.h. der Zusammenschaltung der Blocknetzwerke, wobei jedes Netzwerk der Blockebene wiederum durch ein Makromodell ersetzt wird;

(iv) Berechnung der Spannungen und Ströme innerhalb der Blocknetzwerke sowie innerhalb der zugeordneten Basisnetzwerke.

Diese Analysestrategie ist im Programmsystem MAGNUS mit Hilfe von Iterationsverfahren auf der Ebene der Differentialgleichungen und auf der Ebene der nichtlinearen Gleichungen realisiert worden.
Grundsätzlich werden dabei die Differentialgleichungen mit dem impliziten Eulerverfahren diskretisiert. Die dadurch entstehenden im allgemeinen nichtlinearen Gleichungssysteme werden üblicherweise mit einem Newton-artigen Verfahren gelöst. Zur Berechnung der Iterierten beim Newtonverfahren sind lineare Gleichungssysteme zu lösen. Dafür werden direkte Verfahren auf der Basis der „Technik schwachbesetzter Matrizen“ eingesetzt [HBG71].

In vielen praktischen Anwendungsfällen ist es möglich, eine sehr einfache Modellierung elektronischer Bauelemente zu benutzen. Besonders für die Klasse der digitalen MOS–Schaltungen hat dieser Sachverhalt zu Überlegungen geführt, wie angepaßte Simulationsverfahren in Netzwerkanalyseprogrammen eingesetzt werden können [HPS87], [CGK75], [Sch87].
Es werden einfache Verfahren zur Diskretisierung der Differentailgleichungen (implizite Euler– oder Trapezformel) und Relaxationsverfahren zur Lösung der dadurch entstehenden nichtlinearen Gleichungen eingesetzt. Die spezielle Modellierung der Bauelemente gestattet es, das ursprünglich nur auf die Klemmen der Blocknetzwerke angewandte Markierungsverfahren bezüglich des Signalflußes nun auch auf die Bauelementemodelle auszudehnen. Eine entsprechende Notation erfolgt innerhalb der Modellbeschreibung [BBG89a]. Ein auf dieser Grundlage arbeitendes Partitionierungsverfahren ermöglicht die Entkopplung der Gleichungen der Blocknetzwerke in kleinere Untersysteme und Einzelgleichungen. Dabei werden Basisnetzwerkgleichungssysteme nicht zerlegt und jeweils einem dieser Untersysteme zugeordnet.
Nach der Diskretisierung der zeitlichen Ableitungen werden diese Entkopplungsansätze zur Lösung der nichtlinearen Gleichungen auf der Blockebene mit Iterationsverfahren vom Jacobi– bzw. Gauß–Seidel–Typ bearbeitet. Dieser Sonderfall der Simulation wird als **Timing–Mode** [BBG89a] bezeichnet.

Im Netzwerkanalyseprogramm MAGNUS wurden rechenzeitaufwendige numerische Verfahren so implementiert, daß die Möglichkeit einer vektoriellen Abarbeitung dieser Algorithmen durch den Einsatz entsprechender Hardware besteht. Alle weiteren wesentlichen Punkte der Analysestrategie des Simulators (insbesondere die Strukturierung des Netzwerkes in drei Ebenen, sowie die Möglichkeit eines automatischen Wechsels der numerischen Verfahren nach Konvergenzkriterien) wurden dabei unverändert übernommen.

Die Wahl der Steuerparameter der eingesetzten numerischen Verfahren wurde der Problemstellung angepaßt [Uhl90a] und konnte vielfach aus schaltungstechnischen Überlegungen gewonnen werden.

4.2. Numerische Verfahren.

4.2.1. *Lineare algebraische Gleichungssysteme.*

4.2.1.1. Direkte Verfahren.

Eine wesentliche Aufgabenstellung bei der Analyse elektrischer Netzwerke ist die Lösung linearer algebraischer Gleichungssysteme. Aufgrund der Zusammenschaltung und der u–i–Relation der Netzwerkelemente sind die Koeffizientenmatrizen dieser Gleichungssyteme schwach besetzt, d.h. der überwiegende Teil der Koeffizienten besitzt den Wert Null. Die Besetzung der Netzwerkmatrizen mit von Null verschiedenen Elementen, im folgenden mit NNE (Nicht–Null–Elemente) bezeichnet, ist beliebig.
Es sei vorausgesetzt, daß diese Matrizen quadratisch und regulär sind. Während einer Netzwerkanalyse ändert sich i.a. die Besetzungsstruktur der Matrizen sehr selten.

Dann ist folgende Aufgabe zu lösen:

$$\tilde{A}x = b \text{ mit } \tilde{A} \in \mathbb{R}^{n\times n} \text{ und } x, b \in \mathbb{R}^n. \tag{4.1}$$

Bei der Anwendung des im Abschnitt 4.2.2.2. beschriebenen mehrstufigen Newtonverfahrens zeichnet sich die folgende Aufgabenstellung ab:
Es sind viele lineare Systeme mit unsymmetrischen Matrizen und gleicher Besetztheit mit Nichtnull– und Nullelementen zu lösen. Die Aufgabe wird mit dem Gaußschen Eliminationsverfahren behandelt und auf die genannte Problemstellung angepaßt. Die Gleichung (4.1) wird mit der Transformation

$$P\tilde{A}Q = L \cdot U \tag{4.2}$$

$$Ly = Pb \tag{4.3}$$

$$UQ^{-1}x = y, \tag{4.4}$$

gelöst. Dabei sind P und Q Permutationsmatrizen, L ist eine untere und U ist eine obere Dreiecksmatrizen. Alle Hauptdiagonalelemente von L sind Eins. Die Gleichungen (4.2), (4.3) und (4.4) beschreiben die charakteristischen Schritte bei der Lösung linearer Gleichungssyteme: Zerlegung oder Faktorisierung der Matrix $\tilde{A}$, Vorwärts– bzw. Rückwärtsrechnung. Bei der Bestimmung der Permutationsmatrizen P und Q, d.h. der Festlegung einer Pivotstrategie sind verschiedene Kriterien zu erfüllen. Die Strategie muß so gewählt werden, daß das Eliminationsverfahren numerisch stabil und die Anzahl der während der Elimination entstehenden Nichtnullelemente, d.h. das Fill–in möglichst klein sind. Das Fill–in hat direkten Einfluß auf die Rechenzeit für die Faktorisierung, sowie die Vorwärts– und Rückwärtsrechnung. Es sei erwähnt, daß sich die beiden Forderungen widersprechen können. Es sei

$$\tilde{A} = (a_{i,j}).$$

Zur Wahl als Pivotelement werden nur diejenigen Elemente zugelassen, die die sogenannte β–Bedingung erfüllen. Mit $\mathbb{I} = \{1, 2, \ldots, n\}$ sei

$$\hat{a}_j = \max_{i\in\mathbb{I}} |a_{i,j}|, \quad \forall j \in \mathbb{I}.$$

Ein Matrixelement $a_{i,j} \neq 0$ erfüllt die β–Bedingung, falls zu einem vorgegebenen β mit $(0 \leq \beta \leq 1)$ die Beziehung

$$\hat{a}_j \cdot \beta \leq |a_{i,j}| \quad i,j \in \mathbb{I}$$

gilt. Entsprechend der Definition erfüllt in jeder Matrixspalte wenigstens ein Element die β–Bedingung.
Es bezeichne r_i die Anzahl der NNE in der i–ten Zeile und c_j in der j–ten Spalte. Nach Markowitz [Mar57] sind für jedes $a_{i,j} \neq 0$ Kosten nach

$$(r_i - 1)\ (c_j - 1)$$

definiert.
Pivot wird das Element, das die geringsten Kosten hat und die β–Bedingung erfüllt. Falls mehrere Elemente die Bedingung erfüllen, wird das betragsmäßig größte Element als Pivot genommen. Der numerische Aufwand für die Pivotstrategie ist

$$O(n \cdot m).$$

Die bisherigen Ausführungen sind im folgenden **Algorithmus (1)** zusammengefaßt, wobei der Einfachheit halber nur der 1. Schritt betrachtet wird.

Algorithmus (1)

Step 1: Gegeben sei die quadratische Matrix $\tilde{A} = (a_{i,j})$ der Dimension n;
Wähle ein β mit $0 \leq \beta \leq 1$; $k := 0$;

Step 2: (Suche Pivotkandidaten a_k^{Piv} mit der β–Bedingung)
do while $(j \leq n)$
  do while $(i \leq n)$
    $\hat{a}_j = max\ |a_{i,j}|$
  end do;
  do while $(i \leq n)$
    Test $(\hat{a}_j \cdot \beta \leq |a_{i,j}|\ ?)$
    ja: begin $k := k + 1;$
      $a_k^{Piv} := a_{i,j};$
      $i_k := i;$
      $j_k := j;$
    end;
  end do;
end do;

Step 3: (Wähle den kostengünstigsten Pivotkandidaten aus a_k^{Piv})
$\hat{k}^{Piv} := n * n;$
do while $(i \leq k)$
  r_i: Anzahl der NNE der i_i–ten Spalte;
  c_j: Anzahl der NNE der j_i–ten Spalte;
  $k_i^{Piv} := (r_i - 1)(c_j - 1);$
  $\hat{k}^{Piv} := min\ (\ k_i^{Piv},\ \hat{k}^{Piv});$
end do;

$\hat{a}^{Piv} := 0;$

do while $(i \leq k)$

Test $(k_i^{Piv} = \hat{k}^{Piv})$?

ja: Test $(\hat{a}^{Piv} \leq |a_i^{Piv}|$?)

ja: begin $\hat{a}^{Piv} := |a_i^{Piv}|;$

$l := i;$

end;

end do;

Step 4: $a(i_l, j_j)$ ist Pivotelement. Die Elemente i_l und j_l sind Einträge für die Permutationsvektoren P und Q (siehe (4.2)–(4.4)).

Wie bereits oben dargelegt, sind bei der numerischen Lösung von strukturierten Systemen von Algebro–Differentialgleichungen viele lineare Gleichungssysteme mit der gleichen Besetztheit mit Nichtnullelementen zu lösen. Hierzu werden entweder ausführbare Maschinenprogramme generiert [Gru82], deren Ausführung die Lösung der Gleichungssysteme mit dem speziellen Gaußschen Eliminationsverfahren ist oder es wird ein sogenannter Pseudo–Code erzeugt, dessen Interpretation die Lösung des Systems gibt. Die erste Methode ist sehr schnell und vollständig vom Maschinencode des benutzten Computers abhängig. Das zweite Verfahren kann unabhängig von einem Computer formuliert werden. Der Vorteil der Anwendung des Pseudo–Codes gegenüber der des generierten Maschinenprogramms besteht darin, daß er für seine Speicherung weniger Platz benötigt. Die Rechenzeit für die Interpretation des Pseudo–Codes ist allerdings größer als die für die Ausführung des generierten Maschinenprogramms.
Da mit den Nullelementen der Matrizen von (4.1) keine Operationen auszuführen sind, werden nur die Nichtnullelemente (NNE) gespeichert. Es wird deshalb ein Schema benutzt, in dem die Matrixelemente zeilenweise gespeichert werden. Die Nichtnullelemente von $\tilde{A}$ und deren Spaltenindizes werden in den Vektoren A und JA, die Indizes der Zeilenanfänge von $\tilde{A}$ in A im Vektor IA gespeichert. Die quadratische Matrix $\tilde{A}$ der Dimension n habe m Einträge für Nichtnullelemente. Dann gilt

$$A \in \mathbb{R}^m, \; JA \in \mathbb{N}^m \text{ und } IA \in \mathbb{N}^{n+1},$$

wobei der Wert des letzten Elementes von IA die um Eins erhöhte Anzahl der Nichtnullelemente ist.

4.2.1.2. Vektorisierung der Verfahren.

Unter Nutzung des in Abschnitt 4.2.1.1. erwähnten Pseudo–Codes können Verfahren entwickelt werden, die die Möglichkeiten der Vektorrechner ausnutzen. Die Idee hierbei ist, bei den Anweisungen für den Pseudo–Code jene zu erkennen, die voneinander unabhängig sind. Diese werden extrahiert und falls möglich, zu entsprechenden Vektorbefehlen zusammenfaßt.

Eine Erkennung der entsprechenden Unabhängigkeiten ist mit dem Algorithmus von Yamamoto und Takahashi [YT85] möglich. Hierzu wird der Matrix

$$LU = P\tilde{A}Q$$

aus (4.2) die Matrix

$$M = (m_{i,j}), \qquad m_{i,j} \in \mathbb{N} \cup \{0\}$$

zugordnet und $m_{i,j}$ als das Niveau der Unabhängigkeit bezeichnet. Falls einem Matrixelement das Niveau Null zugeordnet wurde, sind bei der Faktorisierung keine Operationen erforderlich. Alle Matrixelemente mit gleichem Niveau können unabhängig voneinander berechnet werden. Bei der Faktorisierung sind zuerst alle Matrixelemente mit dem Niveau 1, dann mit Niveau 2 usw. zu bestimmen. Nachfolgend ist der Algorithmus für die Berechnung der Elemente $m_{i,j}$ der Matrix M aus [YT85] angegeben:

```
M = 0
for I = 1, N - 1 do
    for all {J : A(J,I) ≠ 0 & J > I} do
        M(J,I) = 1 + max(M(J,I), M(I,I))
        for all {K : A(I,K) ≠ 0 & K > I} do
            M(J,K) = 1 + max(M(J,K), M(J,I), M(I,K))
        end
    end
end
```

Für die Nutzung von Vektorcomputern müssen auf den einzelnen Niveaus Vektorbefehle erkannt werden. Es haben sich die folgenden bewährt:

Nr. Vektoroperation	Operation
1	Skalarprodukt
2	$A(K) = 1/A(K)$
3	$A(K) = A(K) * A(L)$
4	$A(K) = (A(I) * A(J) + A(L) * A(M))A(M)$

Die Schwierigkeit besteht hier darin, daß die Feldelemente indirekt adressiert werden. Im allgemeinen stehen für die einzelnen Vektorcomputer entsprechende Befehle zur Verfügung. Unsere Erfahrungen besagen jedoch, daß diese Befehle in der Regel langsamer sind, als die entsprechenden Vektorbefehle für direkten Speicherzugriff.

4.2.1.3. Beispiel.

Die Besetzungsstruktur und die Werte der Matrix

$$\tilde{A} = \begin{bmatrix} 9 & & & 2 & 1 \\ 1 & & 3 & & 5 \\ & 2 & & 4 & \\ 1 & & 7 & & 8 \\ & 5 & & 7 & 9 \end{bmatrix}. \tag{4.5}$$

werden nach den Ausführungen im Abschnitt 4.2.1.1. in den Vektoren

$$\begin{aligned} A &= [9,2,1,\ 1,3,5,\ 2,4,\ 1,7,8,\ 5,7,9]^T \\ JA &= [1,4,5,\ 1,3,5,\ 2,4,\ 1,3,5,\ 2,4,5]^T \\ IA &= [1,4,7,9,12,15]^T \end{aligned}$$

gespeichert.
Das Vorgehen nach Algorithmus (1) soll nun an der gewählten Matrix $\tilde{A}$ (4.5) erläutert

werden. Nach der Bestimmung der Permutationsmatrizen P und Q ist

$$P\tilde{A}Q = \begin{bmatrix} 2 & 4 & & & \\ 5 & 7 & & & 9 \\ & 2 & 9 & & 1 \\ & & 1 & 7 & 8 \\ & & 1 & 3 & 5 \end{bmatrix}. \tag{4.6}$$

Die Matrix $P\tilde{A}Q$ ist im Vektor A wie folgt gespeichert, wobei i das i–te Element in A bezeichnet.

$$\begin{bmatrix} 7 & 8 & & & \\ 12 & 13 & & & 14 \\ & 2 & 1 & & 3 \\ & & 9 & 10 & 1 \\ & & 4 & 5 & 6 \end{bmatrix}$$

Die Faktorisierung von $\tilde{A}$ erfordert die folgenden Operationen:

```
A(12) = A(12)/A(7)
A(13) = A(13) - A(12) * A(8)
A(2)  = A(2)/A(13)
A(3)  = A(3) - A(2) * A(14)
A(9)  = A(9)/A(1)
A(11) = A(11) - A(9) * A(3)
A(4)  = A(4)/A(1)
A(5)  = A(5)/A(10)
A(6)  = A(6) - A(4) * A(3) - A(5) * A(11)
```

Für die Arbeit mit dem Pseudo–Code werden von den obigen Ergibtanweisungen nur ein Kennzeichen für die Art der Operation (es sind 6 ausreichend), die Indizes und die Länge des Skalarproduktes gespeichert.

Bei der numerischen Lösung eines Gleichungssystems mit 2904 Gleichungen und 58142 Nichtnullelementen ergaben sich die folgenden Werte (MicroVAX 3600, doppelte Genauigkeit, FORTRAN–Programme; $\beta = 10^{-6}$).

Fill–in	10 232	
Operationen Faktorisierung	271 133	
Pseudo–Code Faktorisierung	2 259 272	Byte
Pivotstrategie	725,77	Sek.
Generierung Pseudo–Code	35,74	Sek.
Faktorisierung	4,17	Sek.
VR–Rechnung	0,48 Sek.	
geschätzte Kondition	$1,14 \cdot 10^{12}$	

Zum Abschluß soll für den Fall der Dreieckszerlegung der Matrix (4.5) die Art und Weise der Anordnung der Operationen in den einzelnen Niveaus für eine Vektorisierung erläutert werden. Für unser Beispiel folgt

$$M = \begin{bmatrix} 0 & 0 & & & \\ 1 & 2 & & & 0 \\ & 3 & 0 & & 4 \\ & & 1 & 0 & 5 \\ & & 1 & 1 & 6 \end{bmatrix} .$$

Die Faktorisierung wird nun in der folgenden Reihenfolge der Ergibtanweisungen vorgenommen:

Niveau			
1	A(12)	=	A(12)/A(7)
	A(9)	=	A(9)/A(1)
	A(4)	=	A(4)/A(1)
	A(5)	=	A(5)/A(10)
2	A(13)	=	A(13) − A(12) ∗ A(8)
3	A(2)	=	A(2)/A(13)
4	A(3)	=	A(3) − A(2) ∗ A(14)
5	A(11)	=	A(11) − A(5) ∗ A(3)
5	A(6)	=	A(6) − A(4) ∗ A(3) − A(5) ∗ A(11)

Für das oben zitierte Gleichungssystem mit 2904 Gleichungen wurden 171 Niveaus gefunden und folgendes festgestellt:

Niveau	Anzahl Anweisungen
0	20 565
1	1 952
2	18 399
3	1 362
4	4 238
5	5 124
⋮	⋮

Die größten Vektorlängen bei den Vektoroperationen waren:

Vektoroperationen	Vektorlängen
2	1 952
3	18 399
3	357
3	639

Bei dem Gleichungssystem waren 91,2 % der Operationen vektorisierbar, wobei nur solche mit einer Vektorlänge größer als 3 gezählt wurden.

4.2.2. *Nichtlineare Gleichungssysteme.*

4.2.2.1. Newtonverfahren.

Zur Erläuterung des im Programmsystem MAGNUS implementierten Newtonschen Algorithmus soll das Gleichungssystem

$$G(z) = 0 \tag{4.7}$$
$$G : \mathbb{R}^n \to \mathbb{R}^n,\ G \in C^1$$

betrachtet werden. Das klassische Newtonverfahren basiert auf dem Abbruch einer Taylorreihenentwicklung der Funktion G nach der Ableitung 1. Ordnung

$$G(z + \Delta z) = G(z) + \frac{dG(z)}{dz}\Delta z + O(\Delta z^2)$$

und führt zu folgender Verfahrensweise:

Algorithmus (2)

Step 1: Wähle z^0; $i := 0$;

Step 2: $i := i + 1$;

$$\frac{dG(z^i)}{dz}\Delta z^i = G(z^i);$$

$$z^{i+1} = z^i - \Delta z^i;$$

Step 3: **Test** (Konvergenz ?) **ja:** **goto** *Step 4*;

nein: **goto** *Step 2*;

Step 4: z^{i+1} ist eine Lösung des Problems (4.7).

Das in Algorithmus (2) beschriebene klassische Vorgehen beim Newtonverfahren wird im Programmsystem MAGNUS an verschiedenen Punkten modifiziert.
Prinzipiell ist in jedem Iterationsschritt bei *Step 2* ein lineares Gleichungssystem zu lösen. Das hierbei verwendete Verfahren, welches besonders die schwache Besetzungsstruktur der Koeffizientenmatrix ausnutzt, ist in Abschnitt 4.2.1. beschrieben. Die Koeffizientenmatrix wird erst dann neu berechnet, wenn die Ungleichung zum Test der Konvergenzrate gemäß

$$0.2 * \|\Delta z^{i-1}\| \geq \|\Delta z^i\|$$

nicht erfüllt ist.

Im zweiten Teil von *Step 2* wird die neue Iterierte berechnet. Bedingt durch eine ungenaue Formulierung der Modellgleichungen oder durch das numerische Verhalten des Verfahrens ist es durchaus möglich, daß physikalisch nicht erklärbare Werte der Iterierten berechnet werden können. Dies soll durch die Einführung des Dämpfungsfaktors α, $0 < \alpha \leq 1$ verhindert werden. Die neue Iterierte wird demzufolge nach

$$z^{i+1} = z^i - \alpha * \Delta z^i$$

bestimmt, wobei das Dämpfungsmaß α nach der Vorschrift

$$\alpha = \min_{i=1,\ldots,n} \alpha_i$$

und

$$\alpha_i = \frac{|z^i| + 1}{|\Delta z^i| + 1} \tag{4.8}$$

berechnet wird. Gilt für ein α_i, $\alpha_i < 0.8$, dann wird zusätzlich eine neue Auswertung der Koeffizientenmatrix gefordert. Diese Modifikation hat sich in der Praxis als konvergenzbeschleunigend erwiesen.

Eine weitere Maßnahme zur Begrenzung des Wertes der Iterierten und zur Vermeidung unnötiger Iterationsschritte liegt darin, die Anzahl der Iterationsschritte durch Abtesten von

$$\|G(z^0)\| \leq \delta_1, \; 0 < \delta_1 \ll 1 \tag{4.9}$$

insoweit zu beschränken, daß bei Nichterfüllung der Ungleichung eine neue Startiterierte zu erzeugen ist und die Iteration abgebrochen wird. Eine neue Startiterierte ist auch in dem Fall zu erzeugen, wenn nach einer vorgegebenen Zahl von Iterationsschritten (meist 3 ... 5) keine Konvergenz des Verfahrens erreicht wurde.

Die Akzeptanz der Lösung wird in *Step 3* mit Hilfe von Konvergenztests ermittelt. Gilt für jede Iterierte

$$|\Delta z^i| \leq \varepsilon + \delta_3 * \min\left(|z^i|, \; |z^{i+1}|\right) \tag{4.10}$$

und

$$\|G(z^i)\| \leq \max\left(\left(\|z^i\| + 1\right) * \delta_3, \delta_2\right) \tag{4.11}$$

mit

$$\begin{aligned} 0 &\leq \varepsilon \ll 1, \\ 0 &\leq \delta_2 \ll 1, \\ 0 &\leq \delta_3 \ll 1, \end{aligned}$$

dann ist die Iterierte z^{i+1} Lösung von $G(z) = 0$.

Die Parameter $\varepsilon, \delta_1, \delta_2$ und δ_3 wurden aus Sicht der Aufgabenstellung unter Beachtung physikalischer Größen gewählt. So bezeichnet ε den absoluten Fehler für Spannungs– bzw. Stromwerte. Im Programmsystem MAGNUS werden verschiedene Größen für die absolute Fehlerschranke für Spannungs– bzw. Stromwerte benutzt. Während der Parameter δ_1 ein Maß für den absoluten Fehler der Stromsumme in einem Netzwerkknoten mit geschätzten Werten der Lösungsfunktionen in einem neuen Zeitschritt beschreibt, garantiert der Parameter δ_2 die Einhaltung des Kirchhoffschen Stromgesetzes. Der Wert des Parameters δ_1 wird weiterhin durch den Wert einer Konditionszahlschätzung der Jacobimatrix beeinflußt.
Der Parameter δ_3 bezeichnet den relativen Fehler der Spannungs– und Stromwerte. Er dient der Relativierung des Parameters δ_2 für den Fall sehr großer Stromwerte.

4.2.2.2. Mehrebenen–Newtonverfahren.

Das Gleichungssystem eines hierarchisch strukturierten Netzwerkes läßt sich nach Abschnitt 3.2. in der Form

$$\begin{aligned} 0 &= F_{ij}\left(x_{ij}, \dot{x}_{ij}, y_{ij}, \dot{y}_{ij}, t\right) \\ 0 &= G_i\left(x_i, \dot{x}_i, y_{i1}, \dot{y}_{i1}, \ldots, y_{ir_i}, \dot{y}_{ir_i}, y_i, \dot{y}_i, t\right) \\ 0 &= H\left(y_1, \ldots, y_s, t\right), \end{aligned} \tag{4.12}$$

beschreiben, wobei $i = 1, \ldots, s;\ j = 1, \ldots, r_i$ sowie $s, r_1, \ldots, r_s \in \mathbb{N}$ gilt.
Die Variablen werden mit Hilfe eines impliziten linearen k–Schrittverfahrens, zum Beispiel nach [Gea71] in der Form

$$z^n = \sum_{i=1}^{k} \alpha_i z^{n-1} + h\beta_0 \dot{z}^n \tag{4.13}$$

an diskreten Stellen t_n nährungsweise approximiert, wobei z^n eine Approximation für $z(t_n)$ bezeichnet. Durch Umstellen der Integrationsformel (4.13) nach $\dot{z}$ und Einsetzen in (4.12) erhält man ein hierarchisch strukturiertes System nichtlinearer Gleichungen

$$0 = f_{ij}\left(x_{ij}, y_{ij}\right) \tag{4.14}$$

$$0 = g_i\left(x_i, y_{i1}, \ldots, y_{ir_i}, y_i\right) \tag{4.15}$$

$$0 = h\left(y_1, \ldots, y_s\right), \tag{4.16}$$

wobei

$$f_{ij} : \mathbb{R}^{p_{ij}} \times \mathbb{R}^{2q_{ij}} \to \mathbb{R}^{p_{ij}+q_{ij}}$$

$$g_i : \mathbb{R}^{p_i} \times \mathbb{R}^{2q_{i1}} \times \cdots \times \mathbb{R}^{2q_{ir_i}} \times \mathbb{R}^{2q_i} \to \mathbb{R}^{p_i+q_i+\sum_{j=1}^{r_i} q_{ij}}$$

$$h : \mathbb{R}^{2q_1} \times \cdots \times \mathbb{R}^{2q_s} \to \mathbb{R}^{\sum_{i=1}^{s} q_i}$$

gilt. Die Linearisierung der Gleichungen (4.14), (4.15) und (4.16) führt auf:

$$0 = f_{ij}(\cdot) + \partial_1 f_{ij}(\cdot)\Delta x_{ij} + \partial_2 f_{ij}(\cdot)\Delta y_{ij} \tag{4.17}$$

$$0 = g_i(\cdot) + \partial_1 g_i(\cdot)\Delta x_i + \sum_{j=1}^{r_i} \partial_{1+j} g_i(\cdot)\Delta y_{ij} + \partial_{2+r_i} g_i(\cdot)\Delta y_i \tag{4.18}$$

$$0 = h(\cdot) + \sum_{i=1}^{s} \partial_i h(\cdot)\Delta y_i. \tag{4.19}$$

Das weitere Vorgehen basiert auf einem in der Literatur [EP85] angegebenen Algorithmus, der die Regularität des Gesamtsystems (4.17), (4.18) und (4.19) voraussetzt und auf der Grundlage überbestimmter Teilsysteme eine Zerlegung der Systeme der Basis– und der Blockebene bestimmt. Durch diese Zerlegungen werden stets reguläre Teilsysteme definiert, die es gestatten, die inneren Variablen der Systeme der Basis– und der Blockebene zu eliminieren. Damit erhält man ein Restsystem, welches nur noch von den jeweiligen Klemmenvariablen abhängig ist. Diese Zerlegung erfolgt mit dem Gaußschen Eliminationsverfahren bezüglich der Matrix $\partial_1 f_{ij}$ zunächst auf der Basisebene. Die Teilmatrix $\partial_1 f_{ij,1}$ definiert eine Aufspaltung der Basisnetzwerkgleichungen, so daß aus (4.17)

$$0 = \begin{bmatrix} f_{ij,1}(\cdot) \\ f_{ij,2}(\cdot) \end{bmatrix} + \begin{bmatrix} \partial_1 f_{ij,1}(\cdot) \\ \partial_1 f_{ij,2}(\cdot) \end{bmatrix} \Delta x_{ij} + \begin{bmatrix} \partial_2 f_{ij,1}(\cdot) \\ \partial_2 f_{ij,2}(\cdot) \end{bmatrix} \Delta y_{ij} \tag{4.20}$$

folgt. Die Teilsysteme $f_{ij,1}$ in (4.20) lassen sich gemäß

$$\Delta x_{ij} = -\partial_1 f_{ij,1}(\cdot)^{-1} f_{ij,1}(\cdot) - \partial_1 f_{ij,1}(\cdot)^{-1} \partial_2 f_{ij,1}(\cdot)\Delta y_{ij} \tag{4.21}$$

oder abgekürzt mit

$$\Delta \bar{x}_{ij} = -\partial_1 f_{ij,1}(\cdot)^{-1} f_{ij,1}(\cdot) \tag{4.22}$$

und

$$b_{ij} = -\partial_1 f_{ij,1}(\cdot)^{-1} \partial_2 f_{ij,1}(\cdot) \tag{4.23}$$

in der Form

$$\Delta x_{ij} = \Delta \bar{x}_{ij} + b_{ij} \Delta y_{ij} \tag{4.24}$$

darstellen. Die Elimination von Δx_{ij} in den Teilsystemen $f_{ij,2}$ von (4.20) führt auf

$$0 = f_{ij,2}(\cdot) + \partial_1 f_{ij,2}(\cdot)\,(\Delta \bar{x}_{ij} + b_{ij} \Delta y_{ij}) + \partial_2 f_{ij,2}(\cdot) \Delta y_{ij}. \tag{4.25}$$

Nach Einführung der Abkürzungen

$$\begin{aligned} a_{ij} &= f_{ij,2}(\cdot) + \partial_1 f_{ij,2}(\cdot) \Delta \bar{x}_{ij} \\ c_{ij} &= \partial_1 f_{ij,2}(\cdot) b_{ij} + \partial_2 f_{ij,2}(\cdot) \end{aligned}$$

läßt sich (4.25) als

$$0 = a_{ij} + c_{ij} \Delta y_{ij} \tag{4.26}$$

schreiben. Damit sind für jedes Basissystem die chrakteristischen Größen $\Delta \bar{x}_{ij}, b_{ij}, c_{ij}$ und a_{ij} zweckmäßig durch

$$\begin{aligned} \begin{bmatrix} \partial_1 f_{ij,1}(\cdot) & 0 \\ \partial_1 f_{ij,2}(\cdot) & I \end{bmatrix} \begin{bmatrix} \Delta \bar{x}_{ij} \\ a_{ij} \end{bmatrix} &= -\begin{bmatrix} f_{ij,1}(\cdot) \\ f_{ij,2}(\cdot) \end{bmatrix} \\ \begin{bmatrix} \partial_1 f_{ij,1}(\cdot) & 0 \\ \partial_1 f_{ij,2}(\cdot) & I \end{bmatrix} \begin{bmatrix} b_{ij} \\ c_{ij} \end{bmatrix} &= -\begin{bmatrix} \partial_2 f_{ij,1}(\cdot) \\ \partial_2 f_{ij,2}(\cdot) \end{bmatrix} \end{aligned} \tag{4.27}$$

zu bestimmen. Für jedes Gleichungssystem der Basisebene wird ein System der Form (4.26) dem korrespondierenden Gleichungssystem der Blockebene (4.18) zugeordnet, so daß man das resultierende System durch

$$\begin{aligned} 0 =& a_{i1} + c_{i1} \Delta y_{i1} \\ & \vdots \\ 0 =& a_{ir_i} + c_{ir_i} \Delta y_{ir_i} \\ 0 =& g_i(\cdot) + \partial_1 g_i(\cdot) \Delta x_i + \sum_{j=1}^{r_i} \partial_{1+j} g_i(\cdot) \Delta y_{ij} + \partial_{2+r_i} g_i(\cdot) \Delta y_i \end{aligned}$$

oder

$$\begin{bmatrix} 0 & c_{i1} & & & 0 \\ \vdots & & \ddots & & \vdots \\ 0 & & & c_{ir_i} & 0 \\ \partial_1 g_i(\cdot) & \partial_2 g_i(\cdot) & \dots & \partial_{1+r_i} g_i(\cdot) & \partial_{2+r_i} g_i(\cdot) \end{bmatrix} \begin{bmatrix} \Delta x_i \\ \Delta y_{i1} \\ \vdots \\ \Delta y_{ir_i} \\ \Delta y_i \end{bmatrix} = -\begin{bmatrix} a_{i1} \\ \vdots \\ a_{ir_i} \\ g_i(\cdot) \end{bmatrix}$$

bzw. mit den Abkürzungen

$$\partial_1\gamma_i := \begin{bmatrix} 0 & c_{i1} & & \\ \vdots & & \ddots & \\ 0 & & & c_{ir_i} \\ \partial_1 g_i(\cdot) & \partial_2 g_i(\cdot) & \dots & \partial_{1+r_i} g_i(\cdot) \end{bmatrix},$$

$$\partial_2\gamma_i := \begin{bmatrix} 0 \\ \vdots \\ 0 \\ \partial_{2+r_i} g_i(\cdot) \end{bmatrix}, \quad \Delta\xi_i := \begin{bmatrix} \Delta x_i \\ \Delta y_{i1} \\ \vdots \\ \Delta y_{ir_i} \end{bmatrix} \text{ und } \bar{\gamma}_i(\cdot) := \begin{bmatrix} a_{i1} \\ \vdots \\ a_{ir_i} \\ g_i(\cdot) \end{bmatrix}$$

durch

$$[\partial_1\gamma_i(\cdot) \quad \partial_2\gamma_i(\cdot)] \begin{bmatrix} \Delta\xi_i \\ \Delta y_i \end{bmatrix} = \bar{\gamma}_i(\cdot) \tag{4.28}$$

darstellen kann. Das System (4.28) wird analog zum Vorgehen bei den Systemen der Basisebene wiederum durch den in [EP85] beschriebene Algorithmus in 2 Teilsysteme zerlegt. Dabei ist das erste Teilsystem regulär bezüglich der inneren Variablen ξ_i. Man wird nach wenigen Rechenschritten auf eine zu (4.27) analoge Matrixdarstellung der Form

$$\begin{bmatrix} \partial_1\gamma_{i,1}(\cdot) & 0 \\ \partial_1\gamma_{i,2}(\cdot) & I \end{bmatrix} \begin{bmatrix} \Delta\bar{\xi}_i \\ a_i \end{bmatrix} = - \begin{bmatrix} \bar{\gamma}_{i,1}(\cdot) \\ \bar{\gamma}_{i,2}(\cdot) \end{bmatrix} \tag{4.29a}$$

$$\begin{bmatrix} \partial_1\gamma_{i,1}(\cdot) & 0 \\ \partial_1\gamma_{i,2}(\cdot) & I \end{bmatrix} \begin{bmatrix} b_i \\ c_i \end{bmatrix} = - \begin{bmatrix} \partial_2\gamma_{i,1}(\cdot) \\ \partial_2\gamma_{i,2}(\cdot) \end{bmatrix} \tag{4.29b}$$

geführt, wobei mit

$$\Delta\xi_i = \Delta\bar{\xi}_i + b_i\Delta y_i \tag{4.30}$$

eine Vorschrift zur Berechnung der Werte der inneren Variablen gegeben ist und das System

$$0 = a_i + c_i\Delta y_i \tag{4.31}$$

eines jeden Blocknetzwerkes dem Hauptsystem zugeordnet wird. Mit der soeben beschriebenen Transformation kann man das Hauptsystem zu einem regulärem Gleichungssystem bezüglich der Klemmenvariablen der Blocknetzwerke gemäß

$$\begin{aligned} 0 =& a_1 + c_1\Delta y_1 \\ & \vdots \\ 0 =& a_s + c_s\Delta y_s \\ 0 =& h(\cdot) + \sum_{i=1}^{s} \partial_i h(\cdot)\Delta y_i \end{aligned} \tag{4.32}$$

oder in der Matrixscheibweise gemäß

$$\begin{bmatrix} 0 & c_1 & & \\ \vdots & & \ddots & \\ 0 & & & c_s \\ \partial_1 h(\cdot) & \partial_2 h(\cdot) & \dots & \partial_{1+s} h(\cdot) \end{bmatrix} \begin{bmatrix} \Delta y_1 \\ \vdots \\ \Delta y_s \end{bmatrix} = - \begin{bmatrix} a_1 \\ \vdots \\ a_s \\ h(\cdot) \end{bmatrix} \tag{4.33}$$

ergänzen.

Zusammenfassend ist folgender Algorithmus für ein Mehrebenen–Newtonverfahren notierbar:

Algorithmus (3):

Step 1: Wähle $x_{ij}^0, \xi_i^0, y_i^0 (j = 1, \dots, r_i, i = 1, \dots, s) : k := 0;$

Step 2.1: (Auswertung der Teilsysteme) $k := k + 1;$

do while $(i \leq s)$

do while $(j \leq r_i)$

(Basisebene):

$$\begin{bmatrix} \partial_1 f_{ij,1}(\cdot) & 0 \\ \partial_1 f_{ij,2}(\cdot) & I \end{bmatrix} \begin{bmatrix} \Delta \bar{x}_{ij} \\ a_{ij} \end{bmatrix} = - \begin{bmatrix} f_{ij,1}(\cdot) \\ f_{ij,2}(\cdot) \end{bmatrix}$$

$$\begin{bmatrix} \partial_1 f_{ij,1}(\cdot) & 0 \\ \partial_1 f_{ij,2}(\cdot) & I \end{bmatrix} \begin{bmatrix} b_{ij} \\ c_{ij} \end{bmatrix} = - \begin{bmatrix} \partial_2 f_{ij,1}(\cdot) \\ \partial_2 f_{ij,2}(\cdot) \end{bmatrix}$$

end do;

(Blockebene):

$$\begin{bmatrix} \partial_1 \gamma_{i,1}(\cdot) & 0 \\ \partial_1 \gamma_{i,2}(\cdot) & I \end{bmatrix} \begin{bmatrix} \Delta \bar{\xi}_i \\ a_i \end{bmatrix} = - \begin{bmatrix} \gamma_{i,1}(\cdot) \\ \gamma_{i,2}(\cdot) \end{bmatrix}$$

$$\begin{bmatrix} \partial_1 \gamma_{i,1}(\cdot) & 0 \\ \partial_1 \gamma_{i,2}(\cdot) & I \end{bmatrix} \begin{bmatrix} b_i \\ c_i \end{bmatrix} = - \begin{bmatrix} \partial_2 \gamma_{i,1}(\cdot) \\ \partial_2 \gamma_{i,2}(\cdot) \end{bmatrix}$$

end do;

(Hauptebene):

$$\begin{bmatrix} 0 & c_1 & & \\ \vdots & & \ddots & \\ 0 & & & c_s \\ \partial_1 h(\cdot) & \partial_2 h(\cdot) & \dots & \partial_{1+s} h(\cdot) \end{bmatrix} \begin{bmatrix} \Delta y_1 \\ \vdots \\ \Delta y_s \end{bmatrix} = - \begin{bmatrix} a_1 \\ \vdots \\ a_s \\ h(\cdot) \end{bmatrix}$$

Step 2.2: (Berechnung der Iterierten)

do while $(i \leq s)$

$y_i^{k+1} := y_i^k - \Delta y_i^k;$

$\xi_i^{k+1} := \xi_i^k - \Delta \bar{\xi}_i^k - b_i \Delta y_i^k$

$$\left(\text{Beachte: } \Delta\xi_i := \begin{bmatrix} \Delta x_i \\ \Delta y_{i1} \\ \vdots \\ \Delta y_{ir_i} \end{bmatrix}\right)$$

do while $(j \leq r_i)$

$$x_{ij}^{k+1} := x_{ij}^k - \Delta\bar{x}_{ij}^k - b_{ij}\Delta y_{ij}^k$$

end do;

end do;

Step 3: (Konvergenztests)

do while $(i \leq s)$

do while $(j \leq r_i)$

Test (Basisnetzwerk konvergent?)
nein: goto *Step 2.1*;

end do;

Test (Blocknetzwerk konvergent?)
nein: goto *Step 2.1*;

end do;

Step 4: $x_{ij}^{k+1}, \xi_i^{k+1}, y_i^{k+1} (j = 1, \ldots, r_i,\ i = 1, \ldots, s)$
ist eine Lösung des Problems (4.14 – 4.16).

Das in Algorithmus (3) beschriebene Vorgehen wird im Netzwerkanalyseprogramm MAGNUS an verschiedenen Punkten modifiziert.

Prinzipiell sind in jedem Iterationsschritt bei *Step 2.1* lineare Gleichungssysteme, unter Umständen großer Dimension zu lösen. Hierbei werden die in Abschnitt 4.2.1. beschriebenen Verfahren zur Lösung linearer Gleichungssysteme eingesetzt. Wie in 4.2.2.1. erwähnt, wird im Programmsystem MAGNUS die Koeffizientenmatrix erst neu berechnet, wenn eine Konvergenzrate von 20 % von Iterations– zu Iterationsschritt nicht erreicht wird. Dies hat zur Folge, daß im günstigsten Fall auf jeder Ebene in jedem Iterationsschritt nur **ein** lineares Gleichungssystem zu lösen ist.

Zur Minimierung des Rechenaufwandes werden vorhandene Nullelemente in den Matrizen c_{ij} und c_i bei der Speicherung aussortiert. Diese reduzierten Besetzungsstrukturen waren bei vielen aus der Praxis stammenden Rechenbeispielen sehr stabil, so daß die rechenzeitintensiven Verfahren zur Bestimmung der optimalen Eliminationsreihenfolge (Pivotisierung) nur sehr selten angewandt werden mußten.

In *Step 2.2* werden die neuen Iterierten berechnet. Im Netzwerkanalyseprogramm MAGNUS werden dafür in einem Vorschritt die Vektoren

$$\Delta\xi_i^k := \Delta\bar{\xi}_i^k - b_i\Delta y_i^k$$
$$\Delta x_{ji}^k := \Delta\bar{x}_{ij}^k - b_{ij}\Delta y_{ij}^k$$

berechnet, sowie der für alle Systeme einheitliche Dämpfungsfaktor α nach (4.8) bestimmt und zur Konvergenzbeschleunigung benutzt. Die Konvergenztests zum Finden der Lösungen werden analog zum Algorithmus (1) (vgl. Abschnitt 4.2.2.1.) für jedes Teilsystem durchgeführt. Weitere Variationen des Algorithmus (3) insbesondere Konvergenzbeweise sind in der Literatur [EP85], [HS84], [ELD82], [GBG85] und [RSVH79] zu finden.

4.2.2.3. Gauß–Seidel–Verfahren.

Eine wesentliche Voraussetzung für die Anwendung des hier beschriebenen Verfahrens ist neben den in Abschnitt 4.1. beschriebenen Modellierungsanforderungen die Entkopplung des hierarchischen dreistufigen Netzwerkgleichungssystems auf der Hauptebene und die Anwendung eines modifizierten Waveform–Iterationsverfahrens (vgl. Abschnitt 4.2.3.2.) auf die verbleibenden Gleichungssysteme der Basis– und Blockebene, d.h. auf das $\forall t \in [t_a, t_e]$ durch

$$0 = F_{ij}\left(x_{ij}, \dot{x}_{ij}, y_{ij}, \dot{y}_{ij}, t\right) \tag{4.34a}$$

$$0 = \begin{cases} G_i\left(x_i, \dot{x}_i, y_{i1}, \dot{y}_{i1}, \ldots, y_{ir_i}, \dot{y}_{ir_i}, y_i, \dot{y}_i, t\right) \\ \hat{G}_i\left(y_i, \dot{y}_i, t\right) \end{cases} \tag{4.34b}$$

beschriebene Gleichungssystem. Dabei wird mit $i = 1, \ldots, s$ der Zähler der Systeme der Blockebene und mit $j = 1, \ldots, r_i$ der Zähler für die einem System der Blockebene zugeordneten Systeme der Basisebene bezeichnet.

Das funktionale Iterationsverfahren wird auf das System (4.34a), (4.34b) bezüglich der Teilintervalle $T_1, T_2, \ldots$ mit $\bigcup\limits_{i=1}^{n} T_i = [t_a, t_e]$ angewandt. Innerhalb eines Intervalls T_i werde das System an den Punkten t_k $(k = 1, \ldots)$, $t_k \in T_i$ mit Hilfe der impliziten Eulerformel diskretisiert, wodurch man

$$0 = f_{ij}\left(x_{ij}, y_{ij}\right) \tag{4.35a}$$

$$0 = \varphi_i\left(x_i, y_{i1}, \ldots, y_{ir_i}, y_i\right) := \begin{cases} g_i\left(x_i, y_{i1}, \ldots, y_{ir_i}, y_i\right) \\ \hat{g}_i(y_i) \end{cases} \tag{4.35b}$$

mit

$$f_{ij} : \mathbb{R}^{p_{ij}} \times \mathbb{R}^{2q_{ij}} \longrightarrow \mathbb{R}^{p_{ij}+q_{ij}}$$

$$\varphi_i : \mathbb{R}^{p_i+2q_i} \times \mathbb{R}^{2q_{i1}} \times \cdots \times \mathbb{R}^{2q_{ir_i}} \longrightarrow \mathbb{R}^{p_i+2q_i+\sum\limits_{j=1}^{r_i} q_{ij}}$$

erhält.

Das System (4.35a), (4.35b) wird durch eine Partitionierungsmethode [BBG89a] so in Einzelgleichungen und kleine Teilsysteme zerlegt, daß jedes System der korrespondierenden Basisebene genau einem dieser entstandenen Teilsysteme zugeordnet werden kann. Unter der Voraussetzung, daß ein Gleichungssystem der Blockebene genau in l_i Teilsysteme zerlegt wurde, erhält man damit die Zuordnung

$$\Xi_{Glg} : \begin{bmatrix} w_1(\cdot) \\ w_2(\cdot) \\ \vdots \\ w_{l_i}(\cdot) \end{bmatrix} \mapsto \begin{bmatrix} \varphi_{i,1}(\cdot) \\ \varphi_{i,2}(\cdot) \\ \vdots \\ \varphi_{i,n}(\cdot) \\ f_{i1} \\ \vdots \\ f_{ir_i} \end{bmatrix}, \qquad \text{mit} \quad n = p_i + 2q_i + 2\sum_{j=1}^{r_i} q_{ij}$$

der Gleichungen zu den Teilsystemen, die Zuordnung

$$\Xi_{Sys}: \; w_l \in \{w_1, \ldots, w_{l_i}\} \mapsto \begin{cases} \emptyset & \text{Teilsystem ohne Basissystem} \\ f_{ij}, \ldots & \text{Teilsystem enthält Basissysteme} \end{cases}$$

der Basissysteme zu den Teilsystemen w_l und mit

$$\Xi_{Var}: \; (z_1, z_2, \ldots, z_{l_i})^t \mapsto (x_{i1}, \ldots, x_{ir_i}, x_{i,1}, \ldots, x_{i,p_i}, y_{i1}, \ldots, y_{ir_i}, y_{i,1}, \ldots, y_{i,q_i},)^t$$

eine eineindeutige Zuordnung der Variablen dieser Blocksysteme zu neuen Variablen z_i, wobei die z_i Einzelgrößen oder Vektoren sein können. Allgemein dargestellt, führt dieser Prozeß auf ein durch

$$W(Z) = \begin{bmatrix} w_1(z_1, z_2, & \ldots, & z_n) \\ w_2(z_1, z_2, & \ldots, & z_n) \\ & \vdots & \\ w_n(z_1, z_2, & \ldots, & z_n) \end{bmatrix} = 0, \tag{4.36}$$

mit $w_k : \mathbb{R}^{l_1} \times \mathbb{R}^{l_2} \times \cdots \times \mathbb{R}^{l_n} \to \mathbb{R}^{l_k}, (k = 1, \ldots, n)$ dargestelltes Gleichungsystem.

Die im Abschnitt 4.1. beschriebenen elektrischen Eigenschaften des zugeordneten Netzwerkes garantieren eine blockorientierte Hauptdiagonaldominanz des Systems (4.36). Diese Eigenschaft sichert die Konvergenz des durch

$$\begin{aligned} 0 =& w_1\left(z_1^{k+1}, z_2^k, \ldots, z_n^k\right) \\ 0 =& w_2\left(z_1^{k+1}, z_2^{k+1}, \ldots, z_n^k\right) \\ \vdots& \\ 0 =& w_n\left(z_1^{k+1}, z_2^{k+1}, \ldots, z_n^{k+1}\right) \end{aligned} \tag{4.37}$$

beschriebenen Gauß–Seidel–Ansatzes [OR70]. Zur Lösung von (4.37) sind folgende 2 Varianten geeignet, die mit Hilfe der Abkürzungen

$$Z^{k,i} := \left(z_1^{k+1}, \ldots, z_{i-1}^{k+1}, z_i^k, z_{i+1}^k, \ldots, z_n^k\right) \tag{4.38}$$

und

$$z_i^{k+1} = z_i^k + \Delta z_i^{k+1}$$

durch

$$\begin{bmatrix} \frac{\partial w_1\left(Z^{k,1}\right)}{\partial z_1} & \ldots & & 0 \\ & \frac{\partial w_2\left(Z^{k,2}\right)}{\partial z_2} & & \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \ldots & & \frac{\partial w_n\left(Z^{k,n}\right)}{\partial z_n} \end{bmatrix} \begin{bmatrix} \Delta z_1^k \\ \Delta z_2^k \\ \vdots \\ \Delta z_n^k \end{bmatrix} = - \begin{bmatrix} w_1\left(Z^{k,1}\right) \\ w_2\left(Z^{k,2}\right) \\ \vdots \\ w_n\left(Z^{k,n}\right) \end{bmatrix} \tag{4.39}$$

und

$$\begin{bmatrix} \frac{\partial w_1\left(Z^{k,1}\right)}{\partial z_1} & \ldots & & 0 \\ \frac{\partial w_2\left(Z^{k,1}\right)}{\partial z_1} & \frac{\partial w_2\left(Z^{k,1}\right)}{\partial z_2} & & \\ & \vdots & \ddots & \vdots \\ \frac{\partial w_n\left(Z^{k,1}\right)}{\partial z_1} & \frac{\partial w_n\left(Z^{k,1}\right)}{\partial z_2} & \ldots & \frac{\partial w_n\left(Z^{k,1}\right)}{\partial z_n} \end{bmatrix} \begin{bmatrix} \Delta z_1^k \\ \Delta z_2^k \\ \vdots \\ \Delta z_n^k \end{bmatrix} = - \begin{bmatrix} w_1\left(Z^{k,1}\right) \\ w_2\left(Z^{k,1}\right) \\ \vdots \\ w_n\left(Z^{k,1}\right) \end{bmatrix} \tag{4.40}$$

notiert werden können.

Die erste Variante (4.39) beschreibt eine Linearisierung des Systems (4.36) auf der Basis der Teilsysteme w_i unter Berücksichtigung bereits bestimmter Komponenten des Lösungsvektors $Z^{k,i}$ (4.38).

Im Gegensatz dazu beschreibt die zweite Variante (4.40) eine Linearisierung des Gesamtsystems bezüglich der Variablen z_i^{k+1}, wobei die Jacobimatrix ∂W und die Funktion W mit dem Vektor $Z^{k,1} := \left(z_1^k, \dots, z_n^k\right)$ ausgewertet werden.

Die Systeme (4.39) und (4.40) repräsentieren dabei die beim Newtonverfahren zu lösenden linearen Gleichungssysteme. Die Lösung von (4.39) wurde in [BBG89a] anhand von 2 Möglichkeiten diskutiert.
Die zweite Vorgehensweise (vgl. (4.40)) ist im Programmsystem MAGNUS implementiert worden. Da Jacobimatrix ∂W und Funktion W stets an der gleichen Stelle auszuwerten sind, kann bei diesem Verfahren sowohl für die Berechnung der Werte von ∂W, W und der Lösungen des linearen Systems von (4.40) eine vektorielle Abarbeitung ausgenutzt werden.

Wie bereits erwähnt, können Blocknetzwerke, deren Modellierung eine Lösung der nichtlinearen Gleichungen nach (4.36) erlaubt, auch Basisnetzwerke enthalten. Demzufolge können die Koeffizienten in der Jacobimatrix von (4.40) die Matrixstruktur eines zweistufigen Newtonverfahrens besitzen. Die Auswertung und algorithmische Einbindung der Lösung der Basissysteme unterscheidet sich dabei nicht von der unter 4.2.2.2. beschriebenen Verfahrensweise zwischen Block– und Basisebene.

Im folgenden Algorithmus wurde die Einbindung des hier abgeleiteten Gauß–Seidel–Verfahrens (4.40) als Sonderfall (TIMING–System) beschrieben, wobei von der Lösung des zweistufigen Systems (4.35a), (4.35b) ausgegangen wurde.

Algorithmus (4):

Step 1: Wähle $x_{ij}^0, y_{ij}^0, x_i^0, y_i^0 (j = 1, \dots, r_i, i = 1, \dots, s); k := 0;$

Step 2.1: (Auswertung der Teilsysteme) $k := k + 1;$

do while $(i \leq s)$

(Blockebene):

Test (Ist i ein TIMING–Blocknetzwerk?)

nein: **do while** $(j \leq r_i)$

(Basisebene):

$$\begin{bmatrix} \partial_1 f_{ij,1}(\cdot) & 0 \\ \partial_1 f_{ij,2}(\cdot) & I \end{bmatrix} \begin{bmatrix} \Delta \bar{x}_{ij} \\ a_{ij} \end{bmatrix} = - \begin{bmatrix} f_{ij,1}(\cdot) \\ f_{ij,2}(\cdot) \end{bmatrix}$$

$$\begin{bmatrix} \partial_1 f_{ij,1}(\cdot) & 0 \\ \partial_1 f_{ij,2}(\cdot) & I \end{bmatrix} \begin{bmatrix} b_{ij} \\ c_{ij} \end{bmatrix} = - \begin{bmatrix} \partial_2 f_{ij,1}(\cdot) \\ \partial_2 f_{ij,2}(\cdot) \end{bmatrix}$$

end do;

$$\begin{bmatrix} 0 & c_{i1} & \dots & & 0 \\ \vdots & & \ddots & & \vdots \\ 0 & & & c_{ir_i} & 0 \\ \partial_1\varphi_i(\cdot) & \partial_2\varphi(\cdot) & \dots & \partial_{1+r_i}\varphi(\cdot) & \partial_{2+r_i}\varphi(\cdot) \end{bmatrix} \begin{bmatrix} \Delta x_i \\ \Delta y_{i1} \\ \vdots \\ \Delta y_{ir_i} \\ \Delta y_i \end{bmatrix} = \begin{bmatrix} a_{i1} \\ \vdots \\ a_{ir_i} \\ \varphi_i(\cdot) \end{bmatrix}$$

$$\begin{bmatrix} \Delta x_i^k \\ \Delta y_{i1}^k \\ \vdots \\ \Delta y_{ir_i}^k \\ \Delta y_i^k \end{bmatrix} := \begin{bmatrix} \Delta x_i \\ \Delta y_{i1} \\ \vdots \\ \Delta y_{ir_i} \\ \Delta y_i \end{bmatrix}$$

ja: (Der Vektor ANZAHL_BAN ordnet jedem Teilsystem die Anzahl der entsprechenden Basisnetzwerke zu. Der Vektor INDEX_BAN ordnet jedem Teilsystem die entsprechenden Basisnetzwerke zu.)

do while $(l \leq l_i)$

Test: (Sind dem Teilsystem l Basisnetzwerke zugeordnet?)

ja: do while $(jj \leq$ ANZAHL_BAN (l));

$j :=$ INDEX_BAN (jj);

(Basisebene):

$$\begin{bmatrix} \partial_1 f_{ij,1}(\cdot) & 0 \\ \partial_1 f_{ij,2}(\cdot) & I \end{bmatrix} \begin{bmatrix} \Delta\bar{x}_{ij} \\ a_{ij} \end{bmatrix} = - \begin{bmatrix} f_{ij,1}(\cdot) \\ f_{ij,2}(\cdot) \end{bmatrix}$$

$$\begin{bmatrix} \partial_1 f_{ij,1}(\cdot) & 0 \\ \partial_1 f_{ij,2}(\cdot) & I \end{bmatrix} \begin{bmatrix} b_{ij} \\ c_{ij} \end{bmatrix} = - \begin{bmatrix} \partial_2 f_{ij,1}(\cdot) \\ \partial_2 f_{ij,2}(\cdot) \end{bmatrix}$$

end do;

end do;

$$\frac{\partial w_l\left(Z^{k,1}\right)}{\partial z_l}\Delta z_l^k = w_l\left(Z^{k,1}\right) - \sum_{m=1}^{l-1} \frac{\partial w_m\left(Z^{k,1}\right)}{\partial z_m}$$

end do;

Step 2.2: (Berechnung der Iterierten)

do while $(i \leq s)$

$x_i^{k+1} := x_i^k - \Delta x_i^k$;

$y_i^{k+1} := y_i^k - \Delta y_i^k$;

do while $(j \leq r_i)$

$x_{ij}^{k+1} := x_{ij}^k - \Delta\bar{x}_{ij}^k - b_{ij}\Delta y_{ij}^k$;

$y_{ij}^{k+1} := y_{ij}^k - \Delta y_{ij}^k$;

end do;

end do;

Step 3: (Konvergenztests)

do while $(i \leq s)$

  do while $(j \leq r_i)$

    Test: (Basisnetzwerk konvergent?)
    nein: goto *Step 2.1*;

  end do;

  Test: (Blocknetzwerk konvergent?)
  nein: goto *Step 2.1*;

end do;

Step 4: $x_{ij}^{k+1}, y_{ij}^{k+1}, x_i^{k+1}, y_i^{k+1} (j = 1, \ldots, r_i,\ i = 1, \ldots, s)$
ist eine Lösung des Problems (4.35a), (4.35b).

Im *Step 2.1* des hier beschriebenen Algorithmus wird zusätzlich zur Lösungsmethode nach (4.40) die Variante eines zweistufigen Newtonverfahrens für den Fall vorgesehen, daß keine Partitionierung des zu lösenden Blocknetzwerkgleichungssystems vorliegt oder das Verfahren zu schlecht konvergiert. Dafür werden im Netzwerkanalyseprogramm MAGNUS zusätzliche Strukturinformationen des nicht partitionierten Gleichungssystems bereitgehalten. Die in *Step 3* benutzten Konvergenztests sind mit den Formeln aus Abschnitt 4.2.2.1. für das einfache Newtonverfahren identisch.

4.2.3. *Implizite gewöhnliche Differentialgleichungen.*

4.2.3.1. Prädiktor–Korrektor–Verfahren.

Netzwerkgleichungssysteme lassen sich für viele Netzwerkklassen in Form von Systemen von Algebro–Differentialgleichungen 1. Ordnung darstellen (vgl. Abschnitt 2.2.):

$$\begin{aligned} &0 = f(x, \dot{x}, t), \quad x(t_0) = x^0,\ t \in [t_0, t_e] \subset \mathbb{R} \\ &f : \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^n \\ &x : [t_0, t_e] \to \mathbb{R}^n \\ &\dot{x} : [t_0, t - e] \to \mathbb{R}^n. \end{aligned} \tag{4.41}$$

Bei der von uns betrachteten Aufgabenklasse, der Wahl der Netzwerkvariablen und der Art des Netzwerkgleichungssystems [Rei92] sind im wesentlichen DAE–Systeme mit dem Index 0 oder dem Indes 1 [Gea88] zu lösen. Für solche Gleichungssysteme haben sich Diskretisierungsverfahren auf der Grundlage impliziter Mehrschrittformeln bewährt. Die Algorithmen vieler bekannter Netzwerkanalyseprogramme arbeiten auf der Basis der nach GEAR [Gea71] benannten Diskretisierungsformeln. Danach werden Nährungen für die Lösungsfunktionen x im Integrationsintervall $[t_0, t_e]$ an diskreten Stellen t_n mit Hilfe von linearen k–Schrittverfahren der Form

$$x^n = \sum_{i=1}^{k} \alpha_i x^{n-1} + h\beta_0 \dot{x}^n \tag{4.42}$$

berechnet.Der Vektor x^n bezeichnet die Näherung für den exakten Wert $x(t_n)$. Als Sonderfall ist in (4.42) die implizite Eulerformel

$$x^n = x^{n-1} + h\dot{x}^n \tag{4.43}$$

enthalten.

Aufgrund eines sehr hohen Bedarfs an Speicherkapazität für Vektoren und Matrizen von Variablen, Koeffizienten von Jacobimatrizen und Strukturinformationen bei der Lösung von hierarchischen Gleichungssystemen mit mehreren 10000 Gleichungen, wurde im Netzwerkanalyseprogramm MAGNUS auf eine zusätzliche Ordnungssteuerung verzichtet.

Zur numerischen Lösung des Systems (4.41) wird ein Prädiktor–Korrektor–Verfahren angewendet. Als Korrektor dient die implizite Eulerformel (4.43). Als Prädiktorformel wird die explizite Eulerformel

$$x^n = x^{n-1} + h\dot{x}^{n-1} \tag{4.44}$$

verwendet. Bei Prädiktor–Korrektor–Verfahren müssen beide Formeln von gleicher Ordnung sein, um damit den lokalen Diskretisierungsfehler ε der Korrektorformel

$$x(t_n) = x^K(t_n) + \varepsilon(t_n) \tag{4.45}$$

abzuschätzen. Aus (4.43) und (4.44), d.h. dem Ansatz

$$\begin{aligned} x(t_n) &= x^P(t_n) - 0.5h^2\ddot{x}(t_n) \\ x(t_n) &= x^K(t_n) + 0.5h^2\ddot{x}(t_n) \end{aligned}$$

erhält man für die 2. Ableitung $\ddot{x}$ in Näherung

$$\ddot{x}(t_n) \cong \frac{x^P(t_n) - x^K(t_n)}{h^2},$$

womit sich der Betrag des lokalen Diskretisierungsfehlers nach

$$|\varepsilon(t_n)| \cong \frac{|x^P(t_n) - x^K(t_n)|}{2} \tag{4.46}$$

bestimmen läßt. Für einen vorgegebenen lokalen Diskretisierungsfehler ϵ_0 läßt sich demzufolge die maximal zulässige Schrittweite des folgenden Integrationsschrittes über die Beziehung

$$h_{\text{neu}} \leq h\sqrt{\frac{2 * \epsilon_0}{|x^P - x^K|}}$$

abschätzen.

Der Wert eines Prädiktors nach dem expliziten Eulerverfahren kann als Startwert für das Newtonverfahren dienen, welches für die Lösung der diskretisierten nichtlinearen Gleichungen angewendet wird. Im praktischen Einsatz hat es sich jedoch als sehr vorteilhaft erwiesen, den Startwert für das Newtonverfahren durch ein Interpolationspolynom zu bestimmen, bei dessen Auswertung auch die Werte der Lösungen und ihrer zeitlichen Ableitungen an weiter zurückliegenden Diskretisierungspunkten berücksichtigt werden.

Zur Bestimmung der Akzeptanz eines numerischen Lösungspunktes wird die Beziehung (4.46) nur für solche Variablen abgetestet, für die auch tatsächlich Werte zeitlicher Ableitungen für die Auswertung des Systems (4.41) benötigt werden. Rein algebraische Größen bleiben deshalb unberücksichtigt.

In Algorithmus (5) wird die Lösungsstrategie für das DAE–System (4.41) zusammengefaßt.

Algorithmus (5):

Step 1: Gegeben sei $x^0, [t_o, t_e]$, h_{Start}, n (Anzahl der Variablen), ε (Schranke für den lokalen Diskretisierungsfehler);
$k := 0;\ h_{\text{neu}} := h_{\text{Start}};\ t_a := t_0;$

Step 2: (Prädiktorberechnung)
$k := k+1;\ t := \min(t_e, t_a + h_{\text{neu}});\ h_{\text{neu}} := t - t_a;$
do while $(i \leq n)$
$$x_i^P(t) := x_i(t_a) + h * \dot{x}_i(t_a);$$
$$\dot{x}_i(t) := \dot{x}_i(t_a);$$
end do;

Step 3: (Lösung des diskretisierten Systems)
Löse mit dem Newtonverfahren nach Algorithmus (2) das System
$$0 = f(x(t), \frac{x(t) - x(t_a)}{h}, t);$$
Test: (Newtonverfahren konvergent ?)
ja: **goto** *Step 4;*
nein: $h_{\text{neu}} := \frac{h}{4}$; **goto** *Step 2;*

Step 4: (Test des lokalen Diskretisierungsfehlers)
$\varepsilon_{\max} = 0;$
do while $(i \leq n)$
$$\textbf{Test}\ \left(\varepsilon \leq \frac{|x_i^P(t) - x(t)|}{2}?\right)$$
nein: $h_{\text{neu}} := \frac{h}{2}$; **goto** *Step 2;*
ja: $\varepsilon_{\max} := \max\left(|x_i^P(t) - x(t)|,\ \varepsilon_{\max}\right)$
end do :

Step 5: (Schrittweitenbestimmung)
$$\text{faktor} := 0.9 * \sqrt{\frac{2 * \varepsilon}{\varepsilon_{\max}}};$$
$h_{\text{neu}} := h * \max\ (\text{faktor},\ 2);$

Step 6: **Test:** $(t = t_e?)$
nein: **goto** *Step 2;*
ja: Das Problem (4.41) ist im Intervall $[t_0, t_e]$ gelöst.

Bei einer Implementation des Algorithmus (5) wird in vielen Fällen der 1. Schritt in einem gesonderten Programmabschnitt ausgeführt. Das Problem im Startschritt besteht in einigen Fällen darin, geeignete zeitliche Ableitungen für die gegebenen Anfangswerte zum gegebenen Startzeitpunkt $t := t_0 + h_{\text{Start}}$ zu bestimmen. Diese Aufgabenstellung wird meist mit Einbettungsverfahren [Sch79] gelöst. Für den Startwert des Newtonverfahrens wird, wie oben bereits vermerkt, ein Schätzwert über eine Interpolationsformel berechnet. Die Unterschiede in der Verkleinerung der Schrittweite in *Step 3* und *Step 4* sind durch praktische Erfahrungen geprägt. Das gleiche trifft auf die Wahl der Faktoren 0.9 bzw. 2 in den Formeln des *Step 5* zur Berechnung bzw. Begrenzung der neuen geschätzten Schrittweite zu.

Bei dynamischen Netzwerkanalysen ist das Übertragungsverhalten des Netzwerkes auf bestimmte gegebene Erregungsfunktionen $u,\ u : [t_0, t_e] \to \mathbb{R}$ gesucht. Deshalb wurde im Programmsystem MAGNUS vor jeder Prädiktorberechnung eine Überprüfung der neuen Integrationsschrittweite h_{neu} an den gegebenen Erregungsfunktionen durchgeführt. Dabei wird mit Hilfe der Formel (4.44) ein Prädiktor

$$u^P\left(t + h_{\text{neu}}\right) := u(t_a) + h * \dot{u}(t_a)$$

berechnet, bei dem die Werte der rechten Seite exakt sind. Der Korrektorwert $u^k\left(t + h_{\text{neu}}\right)$ wird durch die gegebene Funktion gemäß

$$u^K\left(t + h_{\text{neu}}\right) := u\left(t + h_{\text{neu}}\right)$$

festgelegt.
Mit einem sich anschließenden Test gemäß *Step 4* können die Schrittweite h_{neu} korrigiert und Unstetigkeiten in den Erregungsfunktionen erkannt werden. Diese Strategie ermöglicht auch bei sehr steilen Flanken eine optimale Wahl der Schrittweite der Erregungsfunktionen.

4.2.3.2. Gauß–Seidel–Iterationsverfahren.

Das Programmsystem MAGNUS wurde für die Analyse von Netzwerken mit mehreren tausend Bauelementen entwickelt. Es ist ein hierarchisch strukturiertes Gleichungssystem zu lösen, welches Gleichungssysteme der Basisebene

$$0 = F_{ij}\left(x_{ij}, \dot{x}_{ij}, y_{ij}, \dot{y}_{ij}, t\right) \quad (i = 1, \ldots, s\ ,\ j = 1, \ldots, r_i), \tag{4.47}$$

der Blockebene

$$0 = G_i\left(x_i, \dot{x}_i, y_{i1}, \dot{y}_{i1}, \ldots, y_{ir_i}, \dot{y}_{ir_i}, y_i, \dot{y}_i, t\right), \tag{4.48a}$$

$$0 = \hat{G}_i\left(y_i, \dot{y}_i, t\right) \quad (i = 1, \ldots, s), \tag{4.48b}$$

und der Hauptebene

$$0 = H\left(y_i, \ldots, y_s, t\right), \tag{4.49a}$$

$$0 = \hat{H}_i\left(y_i, \dot{y}_i, t\right) \tag{4.49b}$$

enthält. Dabei wird mit s die Anzahl der Blocknetzwerke und mit r_i die Anzahl der einem Blocknetzwerk zugeordneten Basisnetzwerke bezeichnet. Die Dimension der Gleichungen eines Teilsystems kann mehrere tausend Variable umfassen.

Eine Möglichkeit der Lösung des Systems (4.47), (4.48a), (4.49a) ist die Diskretisierung der Lösungsfunktionen mit einer linearen k–Schrittformel (vgl. Abschnitt 4.2.3.1.) und die Lösung des so diskretisierten Systems mit einem Mehrebenen–Newtonverfahren (vgl. Abschnitt 4.2.2.2.).
Aufgrund eines sehr hohen Bedarfs an Speicherkapazität für Vektoren von Variablen, Koeffizienten von Jacobimatrizen und Strukturinformationen bei der Lösung von sehr großen Gleichungssystemen wurde die implizite Eulerformel als Diskretisierungsverfahren für das Netzwerkanalyseprogramm MAGNUS ausgewählt und auf eine zusätzliche Ordnungssteuerung (siehe [Gea71]) verzichtet.

Eine wesentliche Eigenschaft der zu simulierenden Schaltungen besteht im unterschiedlichen dynamischen Verhalten der Lösungskomponenten der einzelnen Blocknetzwerkgleichungssysteme. Ein Gauß–Seidel–ähnliches Iterationsverfahren für Differentialgleichungssysteme auf einem zeitlich diskreten Raster nutzt diese Eigenschaft wesentlich aus. Dabei werden die Lösungen der einzelnen Blocknetzwerkgleichungssysteme innerhalb eines für alle Systeme gleichen Intervalls an unterschiedlichen Diskretisierungspunkten approximiert. Die Reihenfolge der Abarbeitung der Systeme wird mit Hilfe der Richtung des Signalflusses festgelegt. Nach Abschnitt 3.2. werden für die Gleichungssysteme der Block– und Hauptebene zusätzliche Gleichungen (4.48b), (4.49b) erzeugt, die die Wechselwirkungen mit anderen Blocknetzwerken beschreiben. Diese Gleichungsanteile werden für ein Iterationsverfahren vom Gauß–Seidel–Typ für gewöhnliche Differentialgleichungssysteme ausgenutzt.

Zur formelmäßigen Beschreibung dieser Vorgehensweise fassen wir zunächst die Gleichungen eines Blocknetzwerkes und der zugeordneten Basisnetzwerke mit dem Ausdruck

$$\Phi_i := \begin{cases} F_{i1}\left(x_{i1}, \dot{x}_{i1}, y_{i1}, \dot{y}_{i1}, t\right) \\ \vdots \\ F_{ir_i}\left(x_{ir_i}, \dot{x}_{ir_i}, y_{ir_i}, \dot{y}_{ir_i}, t\right) \\ G_i\left(x_i, \dot{x}_i, y_{i1}, \dot{y}_{i1}, \dots, y_{ir_i}, \dot{y}_{ir_i}, y_i, \dot{y}_i, t\right) \\ \hat{G}_i\left(y_i, \dot{y}_i, t\right) \end{cases} \tag{4.50}$$

zusammen, für den

$$\begin{aligned} &\Phi_i(v_i, \dot{v}_i, w_1, \dot{w}_1, \dots, w_i, \dot{w}_i, \dots, w_s, \dot{w}_s, t) = 0 \\ &\Phi_i : \mathbb{R}^{n_{i1}} \times \mathbb{R}^{n_{i1}} \times \mathbb{R}^{2n_{12}+\cdots+2n_{i2}+\cdots+2n_{s2}} \to \mathbb{R}^{n_{i1}+n_{i2}} \end{aligned} \tag{4.51}$$

gilt.
Die neu eingeführten Variablen v_i und w_i

$$(v_i, w_i) := (x_{i1}, \dots x_{ir_i}, y_{i1}, \dots, y_{ir_i}, x_i, y_i) \tag{4.52}$$

bezeichnen die Variablen des Systems (4.51). Die Variablen

$$W := (w_1, \dots w_s)$$

bezeichnen wir in ihrer Gesamtheit als Relaxationsvariable.

Zur näheren Erläuterung betrachte man die in Abschnitt 3.1.2. gezeigte Abbildung 3.5. Es wird die Ergänzung des Blocknetzwerkes V mit einer Netzwerkstruktur demonstriert, die aus den Blocknetzwerken I, II und III stammt und zur Entkopplung des Netzwerkes V von den anderen Blocknetzwerken dient. Das Gleichungssystem dieser Netzwerkstruktur kann durch ein System der Form (4.48b), d.h.

$$0 = \hat{G}_5(y_5, \dot{y}_5, t)$$

angegeben werden. Die Zusammenschaltung dieser Netzwerkstruktur mit dem Blocknetzwerk V kann durch das zugeordnete System

$$\Phi_5(v_5, \dot{v}_5, w_1, \dot{w}_2, \dots, w_5, \dot{w}_5, t) = 0$$

beschrieben werden. Der Vektor der Variablen w_5 besteht hier aus der der Spannung $u_{V,1}$ zugeordneten Variablen u_{51}, d.h.

$$w_5 := (u_{51}).$$

Elemente des Vektors w_1, w_2 und w_3 sind unter anderem die Variablen u_{11}, u_{12}, u_{21} bzw. u_{31}, die den Spannungen $u_{I,i}, u_{I,ii}, u_{II,i}$ und $u_{III,i}$ zugeordnet sind, d.h.

$$\begin{aligned} w_1 &:= (u_{11}, u_{12}, \dots), \\ w_2 &:= (u_{21}, \dots) \end{aligned}$$

und

$$w_3 := (u_{31}, \dots).$$

Demnach können grundsätzlich alle Gleichungssysteme der Blockebene und der ihnen jeweils zugeordneten Basisebene durch ein System gemäß

$$\begin{aligned}
0 &= \Phi_1\left(v_1, \dot{v}_1, w_1, \dot{w}_1, \ldots, w_i, \dot{w}_i, \ldots, w_s, \dot{w}_s, t\right)\\
0 &= \Phi_2\left(v_2, \dot{v}_2, w_1, \dot{w}_1, \ldots, w_i, \dot{w}_i, \ldots, w_s, \dot{w}_s, t\right)\\
&\vdots\\
0 &= \Phi_s\left(v_s, \dot{v}_s, w_1, \dot{w}_1, \ldots, w_i, \dot{w}_i, \ldots, w_s, \dot{w}_s, t\right)
\end{aligned} \tag{4.53}$$

dargestellt werden. Entsprechend einer durch die Indizierung der Systeme Φ festgelegten Abarbeitungsreihenfolge ergibt sich folgender Ansatz für ein Gauß–Seidel–Verfahren:

$$\begin{aligned}
0 &= \Phi_1\left(v_1^{k+1}, \dot{v}_1^{k+1}, w_1^{k+1}, \dot{w}_1^{k+1}, w_2^{k}, \dot{w}_2^{k}, \ldots, w_s^{k}, \dot{w}_s^{k}, t\right)\\
0 &= \Phi_2\left(v_2^{k+1}, \dot{v}_2^{k+1}, w_1^{k+1}, \dot{w}_1^{k+1}, w_2^{k+1}, \dot{w}_2^{k+1}, \ldots, w_s^{k}, \dot{w}_s^{k}, t\right)\\
&\vdots\\
0 &= \Phi_s\left(v_s^{k+1}, \dot{v}_s^{k+1}, w_1^{k+1}, \dot{w}_1^{k+1}, w_2^{k+1}, \dot{w}_2^{k+1}, \ldots, w_s^{k+1}, \dot{w}_s^{k+1}, t\right).
\end{aligned} \tag{4.54}$$

Mathematisch stellt das im nachfolgenden Algorithmus (6) angegebene Iterationsverfahren auf der Ebene der Differentialgleichungen eine Modifikation der in den Programmen RELAX [LRSV82] bzw. ELDO [HPS87] implementierten Algorithmen dar.

Die Besonderheit unserer Vorgehensweise liegt

(i) in der Art und Weise der Aufstellung des hierarchischen Netzwerkgleichungssystems und in der Erzeugung der Blocknetzwerkgleichungsanteile für das Gauß–Seidel–Iterationsverfahren;

(ii) der Berücksichtigung der zusätzlichen Ebene der Basisnetzwerke;

(iii) in der Kombination der Vorteile der Iterationsverfahren auf dem Differentialgleichungsniveau mit denen der Mehrebenenverfahren vom Newtontyp auf einem einheitlichen Zeitintervall zum Zusammenfassen hierarchisch strukturierter Blocknetzwerke;

(iv) in der Implementation automatischer Partitionierungstechniken zur Erzeugung regulärer linearer Teilsysteme bei regulärem Gesamtsystem nach [EP85].

Der hier genannte Punkt (iii) ist für den Nutzer des Programmsystems MAGNUS von besonderer Bedeutung. Bei Nichtkonvergenz von hierarchischen Blocknetzwerkgleichungssystemen werden diese mit Hilfe der Gleichungen des Hauptnetzwerkes (4.49a), (4.49b) zusammengefaßt. Die Möglichkeit dafür liegt in der Struktur der Jacobimatrix des Hauptsystems bei einem 3–Ebenen–Newtonverfahren. Die Koeffizientenmatrizen der Systeme (4.49b) sind Approximationen der Jacobimatrizen jener reduzierter Gleichungssysteme, die beim 3–Ebenen–Newtonverfahren (vgl. Abschnitt 4.2.2.2., Formel (4.31)) vom Blocknetzwerk dem Hauptnetzwerk zugeordnet werden. Die numerische Realisierung der Zusammenfassung nichtkonvergenter Gleichungssysteme der Blockebene erfolgt demnach durch ein 3–Ebenen–Netwtonverfahren unter Berücksichtigung der Gleichungsanteile (4.47), (4.48a), (4.49a) und (4.49b). Dabei werden die Gleichungsanteile (4.49b) für nichtkonvergente Systeme der Blockebene durch die beim 3–Ebenen–Newtonverfahren benutzten reduzierten Systeme der Blockebene überschrieben. Die verbliebenen Systeme (4.49b) sind aufgrund ihrer Definition ein Ersatz für die Gleichungen (4.48b) der nichtkonvergenten Blocknetzwerke

und ermöglichen so eine Fortsetzung des Gauß–Seidel–Iterationsprozesses bezüglich der neu zusammengefaßten Blocknetzwerkstruktur und den restlichen nicht zusammengefaßten Blocknetzwerken.
Werden nach dieser Methode die Systeme der Blocknetzwerke m und n zusammengefaßt, wobei das System m vor dem System n in der Abarbeitungsreihenfolge liegt, ist folgendes zu beachten:

(i) Die Anzahl der Gleichungssysteme, die mit dem hier beschriebene Iterationsverfahren behandelt werden, reduziert sich um die um Eins verminderte Anzahl der zusammenzufaßenden Blocknetzwerkgleichungssyteme (in unserem Fall um genau ein System).

(ii) Die Zusammenfaßung der Systeme Φ_m und Φ_n durch das System $\tilde{\Phi}_m$ wird ausführlich durch

$$\tilde{\Phi}_m := \begin{cases} F_{m1}\left(x_{m1}, \dot{x}_{m1}, y_{m1}, \dot{y}_{m1}, t\right) \\ \vdots \\ F_{mr_m}\left(x_{mr_m}, \dot{x}_{mr_m}, y_{mr_m}, \dot{y}_{mr_m}, t\right) \\ G_m\left(x_m, \dot{x}_m, y_{m1}, \dot{y}_{m1}, \ldots, y_{mr_m}, \dot{y}_{mr_m}, y_m, \dot{y}_m, t\right) \\ F_{n1}\left(x_{n1}, \dot{x}_{n1}, y_{n1}, \dot{y}_{n1}, t\right) \\ \vdots \\ F_{nr_n}\left(x_{nr_n}, \dot{x}_{nr_n}, y_{nr_n}, \dot{y}_{nr_n}, t\right) \\ G_n\left(x_n, \dot{x}_n, y_{n1}, \dot{y}_{n1}, \ldots, y_{nr_n}, \dot{y}_{nr_n}, y_n, \dot{y}_n, t\right) \\ H\left(y_1, \cdots, y_m, \cdots, y_n, \cdots, y_s, t\right) \\ \hat{H}_1\left(y_1, \dot{y}_1, t\right) \\ \vdots \\ \hat{H}_{m-1}\left(y_{m-1}, \dot{y}_{m-1}, t\right) \\ \hat{H}_{m+1}\left(y_{m+1}, \dot{y}_{m+1}, t\right) \\ \vdots \\ \hat{H}_{n-1}\left(y_{n-1}, \dot{y}_{n-1}, t\right) \\ \hat{H}_{n+1}\left(y_{n+1}, \dot{y}_{n+1}, t\right) \\ \vdots \\ \hat{H}_s\left(y_s, \dot{y}_s, t\right) \end{cases}$$

beschrieben. Abkürzend nach (4.50) – (4.52) schreibt man

$$0 = \hat{\Phi}_m(v_m, \dot{v}_m, v_n, \dot{v}_n, w_1, \dot{w}_1, \ldots, w_i, \dot{w}_i, \ldots, w_s, \dot{w}_s, t) \tag{4.55}$$

(iii) Das Gleichungssystem $\tilde{\Phi}_m$ wird an der Stelle des Systems Φ_m in den Abarbeitungsprozeß integriert.

Falls während einer Netzwerkanalyse mit dem Programmsystem MAGNUS Blocknetzwerkstrukturen zusammengefaßt werden, ist dies fast immer ein Hinweis darauf, daß der Signalfluß zwischen diesen Blocknetzwerken fehlerhaft definiert wurde. Im ungünstigsten Fall werden alle Systeme der Blockebene zusammengefaßt. Dabei benutzen die statischen Analyseverfahren $T und $M bzw. die dynamischen Analyseverfahren T und M den gleichen numerischen Algorithmus (3–Ebenen–Newtonverfahren) zur Lösung der diskretisierten Gleichungen. Der Vorteil dieser Verfahrensweise kommt vor allem bei sehr rechenzeitintensiven numerischen Berechnungen zum Tragen, da Zwischenergebnisse nicht verloren gehen und automatisch eine den numerischen Schwierigkeiten angepaßte Wahl des Lösungsverfahrens erfolgen kann.

Der Algorithmus (6) beschreibt dieses Vorgehen im Überblick.

Algorithmus (6):

Step 1: Gegeben sei $v_1^0, \cdots, v_s^0,\ w_1^0, \cdots, w_s^0,\ T := [t_o, t_e],\ h_{\text{Start}}$ (Startschrittweite); ε (Schranke für den lokalen Diskretisierungsfehler); $k_{\max}$ (maximale Iterationsanzahl); $h_{\text{neu}} := h_{\text{Start}}$; $t_a := t_0$;

Step 2: (Prädiktorberechnung für Relaxationsvariable)
$k := 0$; $t := t_a + h_{\text{neu}}$;

do while $(i \leq s)$

$$w_i^k(t) := w_i(t_a) + h * \dot{w}_i(t_a);$$

$$\dot{w}_i^k(t) := \dot{w}_i(t_a);$$

end do;

Step 3: (Neuer Iterationsschritt)
$k := k + 1$;

Step 4: (Auswahl eines Blocknetzwerkgleichungssystems)
do while $(i \leq s)$

Löse nach Algorithmus (5) mit den Anfangswerten $v_i^{k,0} = v_i(t_a)$ und $w_i^{k,0} = w_i(t_a)$ im Intervall T das System :
$0 = \Phi_i\ (v_i^k, \dot{v}_i^k,\ w_1^k, \dot{w}_1^k, \ldots,\ w_i^k, \dot{w}_i^k, \ldots,\ w_s^{k-1}, \dot{w}_s^{k-1},\ t)$;
Konvergenztest des Gleichungssytems Φ_i an Hand der Variablen w_i^k;

end do;

Step 5: **Test** (Konvergenz für alle Gleichungssyteme Φ_i ?)
ja: goto *Step 2*;
nein: goto *Step 6*;

Step 6: **Test** ($k \geq k_{\max}$?)
ja: Blocknetzwerke mit nichtkonvergenten Gleichungssystemen werden zusammengefaßt;
nein: goto *Step 3*;

Step 7: **Test:** ($t = t_e$?)
nein: goto *Step 2;*
ja: Das Problem (4.54) ist im Intervall $[t_0, t_e]$ gelöst.

Das hier beschriebene Verfahren beruht auf der Entkopplung der Gleichungssysteme der Blockebene. Diese Entkopplung der Systeme der Blockebene ist nur für den Fall anwendbar, daß das Gleichungssystem der Hauptebene ein lineares algebraisches System ist. Für den allgemeinen Fall eines nichtlinearen Hauptsystems sind entsprechende numerische Partitionierungsverfahren in [Uhl91] angegeben.

5.1. BSPREP.

Das Netzwerk BSPREP stellt das Modell einer Volladderschaltung dar und wurde bereits in den Abschnitten 3.1.4. und 3.2.4. zur Demonstration der Eingabemöglichkeiten des Netzwerkanalyseprogrammes MAGNUS bzw. zur Illustration des Gleichungssystems verwendet. Die in diesem Abschnitt angegebenen Rechenzeiten beziehen sich auf einen Rechner des Typs VAXstation 4000–60.
Der vollständige Quelltext der MSPICE–Beschreibung ist aus Abbildung 3.8 ersichtlich. Wesentlich für die Auswertung der Analyseergebnisse ist der in Abbildung 5.1 gezeigte MAGNUS–Protokollausschnitt zur Aufstellung der Netzwerkgleichungen.

MAGNUS Version V1.0–2
Analyse von : BSPREP

Generierung der Gleichungen

Basisnetzwerk: NOR/E				
1 Gleichung	3	Matrixelemente	8	X-Elemente
	2	Koppelvariablen	0	LOAD-Wert(e)
Basisnetzwerk: NOR_0001/E				
3 Gleichungen	11	Matrixelemente	10	X-Elemente
	6	Koppelvariablen	0	LOAD-Wert(e)
Basisnetzwerk: NOR_0002/E				
2 Gleichungen	7	Matrixelemente	9	X-Elemente
	4	Koppelvariablen	0	LOAD-Wert(e)
Blocknetzwerk: *******/X				
2 Gleichungen	4	Matrixelemente	11	X-Elemente
	2	Koppelvariablen	6	LOAD-Wert(e)
Blocknetzwerk: BHA1/E				
18 Gleichungen	61	Matrixelemente	26	X-Elemente
	18	Koppelvariablen	2	LOAD-Wert(e)
Blocknetzwerk: BHA2/E				
22 Gleichungen	76	Matrixelemente	29	X-Elemente
	22	Koppelvariablen	2	LOAD-Wert(e)
Blocknetzwerk: BOR/E				
7 Gleichungen	18	Matrixelemente	13	X-Elemente
	6	Koppelvariablen	2	LOAD-Wert(e)
Hauptnetzwerk: BSPREP/E				
18 Gleichungen	64	Matrixelemente	36	X-Elemente
	18	Koppelvariablen	12	LOAD-Wert(e)

Gleichungssystem mit 4 Block- und 6 Basis-Teilsystemen
insgesamt:

82 Gleichungen	277	Matrixelemente	172	X-Elemente
	96	Koppelvariablen	24	LOAD-Wert(e)

Abbildung 5.1: MAGNUS–Protokoll der Gleichungssystemgenerierung

Für Basisnetzwerke wird grundsätzlich nur eine Gleichungsdarstellung erzeugt. Im Programmsystem MAGNUS werden Knotenspannungen nicht als Variable geführt, wenn zum Bezugsknoten gerichtete eingeprägte Spannungsquellen an den Knoten führen. Deshalb werden in unserem Beispiel drei Gleichungssystemdarstellungen für die verschiedenen Beschaltungsvarianten des Basisnetzwerkes NOR erzeugt.

Zwischen den Blocknetzwerken BHA1, BHA2 und BOR wird der Signalfluß berücksichtigt. Aufgrund der vorhandenen U–Markierungen wird das zusätzliche Blocknetzwerk *******/X (vgl. Abschnitt 3.1.4.) erzeugt.

Mit Hilfe der Steueranweisungen (vgl. Abbildung 3.8) werden drei Analysen des Netzwerkes BSPREP definiert. Die erste Analyse .OP $T definiert einen Einschwingvorgang zur Bestimmung eines Arbeitspunktes, in dem die Quellen des Netzwerkes auf ihre Nominalwerte geregelt werden und die Simulation so lange fortgesetzt wird, bis alle zeitlichen Ableitungen der Netzwerkvariablen einer vorgegebenen Schranke genügen. Durch den Befehl .OP T schließt sich eine dynamische Analyse an, die im Intervall [0ns, 200ns] die Reaktion des Netzwerkes auf die vorgegebenen Eingangssignale berechnet. Mit den Parametern $T bzw. T wird dabei das in Abschnitt 4.2.3.2. beschriebenen Gauß–Seidel–Iterationsverfahren zur Lösung der jeweiligen Aufgabenstellung ausgewählt.

DC - ANALYSE : EINSCHWINGVORGANG

STATISTIK

ANZAHL DER INTEGRATIONSSCHRITTE: 67
ANZAHL DER ABLEHNUNGEN: 0

NETZWERKNAME	DIM	ABL	NEW	FKT	JAC	PIV	FAK	LIN
*******/X	2	1	107	107	15	1	14	106
NOR/E	2	1	120	120	37	1	36	155
NOR_0001/E	6	1	120	120	38	1	37	230
NOR_0001/E	6	1	119	119	37	0	36	230
BHA1/E	18	1	119	119	20	1	37	119
NOR_0002/E	4	4	163	163	84	1	80	319
NOR_0001/E	6	4	163	163	84	0	79	399
NOR_0001/E	6	4	159	159	80	0	79	399
BHA2/E	22	4	159	159	51	1	80	159
BOR/E	7	4	157	157	34	1	30	153
BSPREP/E	18	0	0	0	0	0	0	0

Analysezeit : 5.95 sec.

Abbildung 5.2: Ergebnistafel der statischen Analyse .OP $T

Die Befehle .STORE und .LOAD dienen zum Retten bzw. Laden der Werte der Netzwerkvariablen im Arbeitspunkt. Dadurch ist es möglich, die dynamische Analyse des Netzwerkes im Intervall [0ns, 200ns] mit einem anderen numerischen Verfahren zu wiederholen. Mit der Anweisung .OP M wird ein Mehrebenen–Newtonverfahren ausgewählt, welches in jedem Zeitschritt ein hierarchisches in drei Ebenen strukturiertes und diskretisiertes DAE–System löst. Die numerischen Ergebnisse beider dynamischer Analysen sind im Rahmen der vorgegebenen Genauigkeit identisch.
Zur Beurteilung dieses Sachverhaltes enthalten die im folgenden dargestellten Ausschnitte aus den numerichen Protokollen Angaben über die Dimension der Gleichungssyteme (DIM), die Anzahl der lokalen Zeitschrittzurückweisungen (ABL), die Anzahl der Schritte beim Newtonverfahren (NEW), die Anzahl der Auswertungen der Netzwerkgleichungen (FKT), die Anzahl der Auswertung der Jacobimatrizen (JAC), die Anzahl der notwendigen Pivotstrategien (PIV), die Anzahl der Faktorisierungen der Koeffizientenmatrizen (FAK) und die Anzahl der gelösten linearen Gleichungssysteme (LIN) für jedes Gleichungssystem.
Die Protokolle (Abbildung 5.2 – 5.4) weisen jedoch für den Simulationsmodus .OP M deutlich mehr gelöste lineare Gleichungssysteme auf der Blockebene aus, als beim Simulationsmodus .OP T bzw. .OP $T. Die Ursache dafür liegt in der zusätzlichen Lösung von linearen Systemen bei der Berechnung von Darstellungen des Klemmenverhaltens der Blocknetzwerke nach der Formel (4.29b).

TR - ANALYSE : GAUSS-SEIDEL-NEWTON-MEHREBENENVERFAHREN

STATISTIK

ANZAHL DER INTEGRATIONSSCHRITTE: 153
ANZAHL DER ABLEHNUNGEN: 0

NETZWERKNAME	DIM	ABL	NEW	FKT	JAC	PIV	FAK	LIN
*******/X	2	3	277	277	40	1	38	275
NOR/E	2	1	313	313	123	1	123	436
NOR_0001/E	6	1	313	313	127	1	127	694
NOR_0001/E	6	1	313	313	128	0	127	697
BHA1/E	18	1	313	313	52	1	128	313
NOR_0002/E	4	4	334	334	164	1	164	662
NOR_0001/E	6	4	334	334	164	0	163	826
NOR_0001/E	6	4	334	334	164	0	163	826
BHA2/E	22	4	334	334	80	1	164	334
BOR/E	7	6	375	375	88	1	86	373
BSPREP/E	18	0	0	0	0	0	0	0

Analysezeit : 14.01 sec.

Abbildung 5.3: Ergebnistafel der dynamischen Analyse .OP T

TR - ANALYSE : MEHREBENEN-NEWTONVERFAHREN

STATISTIK

ANZAHL DER INTEGRATIONSSCHRITTE: 139
ANZAHL DER ABLEHNUNGEN: 16

NETZWERKNAME	DIM	ABL	NEW	FKT	JAC	PIV	FAK	LIN
*******/X	2	16	328	328	175	1	172	497
NOR/E	2	16	325	325	188	1	187	511
NOR_0001/E	6	16	325	325	187	1	186	882
NOR_0001/E	6	16	325	325	187	0	185	882
BHA1/E	18	16	324	324	141	1	188	700
NOR_0002/E	4	16	324	324	200	1	199	721
NOR_0001/E	6	16	323	323	199	0	198	920
NOR_0001/E	6	16	323	323	199	0	198	920
BHA2/E	22	16	323	323	136	1	199	920
BOR/E	7	16	323	323	107	1	106	640
BSPREP/E	18	16	322	322	1	1	230	322

Analysezeit : 13.31 sec.

Abbildung 5.4: Ergebnistafel der dynamsichen Analyse .OP M

Bei der Aufteilung der Schaltungshierarchie ist daher unbedingt darauf zu achten, daß die Anzahl der inneren Knoten stets deutlich größer ist als die Anzahl der Klemmen. Auf die Definition von Basisnetzwerken sollte vor allem dann verzichtet werden, wenn die Dimension der Basisnetzwerke klein und die Klemmenanzahl sehr groß ist. Unsere hierarchische Beschreibung mit MSPICE gestattet stets eine Auflösung der Basisnetzwerke in die zugeordneten Blocknetzwerke oder eine Umstrukturierung der Hierarchie, d.h., Basisnetzwerke werden zu Blocknetzwerken. Das Netzwerk in Abschnitt 5.2. soll dieses Vorgehen näher erläutern.

5.2. MEGA.

Das Beispiel MEGA stammt aus der Zusammenarbeit mit einem Industriepartner. Die Simulation dient der Überprüfung der Logikfunktion des Schreib–Lese– bzw. Lese–Schreib–Zyklus eines dynamischen MOS–Speicherschaltkreises. Das Netzwerk besteht aus über 38.000 Zweigen mit ca. 18.000 Aufrufen verschiedener MOS–Transistormodelle.

```
Simulationsbeispiel MEGA (4–Mbit–DRAM), Variante MEGA_0
XL PWANNE A0 A1 A10 ... LIRA1
XR PWANNE NRAS NCAS ... RERA1
XGMATS PWANNE NDLV ... GMATS
      .
      . Referenzspannungen und Spannungsquellen mit Erregungsfunktionen
      .
*Blocknetzwerk GMATS
.SUBCKT GMATS/T
+ E(PWANNE NDLV TWLS ... )
XBLK0 PWANNE A0Y NA0Y ... MBLKS0
XBLK1 PWANNE A0Y NA0Y ... MBLKS
XBLK2 PWANNE A0Y NA0Y ... MBLKS
XBLK3 PWANNE A0Y NA0Y ... MBLKS
XBLK4 PWANNE A0Y NA0Y ... MBLKS
XBLK5 PWANNE A0Y NA0Y ... MBLKS
XBLK6 PWANNE A0Y NA0Y ... MBLKS
XBLK7 PWANNE A0Y NA0Y ... MBLKS
      .
      . Aufrufe weiterer Netzwerkelemente
      .
.ENDS GMATS/T
*Blocknetzwerk RERA1
.SUBCKT RERA1/T
+ E(PWANNE NRAS NCAS ... ) A(TBM4 TM8B ... ) E(SPG NWANNE)
      .
      . Aufrufe weiterer Netzwerkelemente und Teilnetzwerke
      .
.ENDS RERA1/T
*Blocknetzwerk LIRA1
.SUBCKT LIRA1/T
+ E(PWANNE A0 A1 ... ) A(A0X A0Y ... ) E(SPG NWANNE ... )
      .
      . Aufrufe weiterer Netzwerkelemente und Teilnetzwerke
      .
.ENDS LIRA1/T
*Basisnetzwerk MBLKS
.SUBCKT MBLKS/E
+ E(PWANNE A0Y NA0Y ... ) U(DLA0 NLA0 ... ) E(SPG NWANNE ... )
      .
      . Aufrufe weiterer Netzwerkelemente und Teilnetzwerke
      .
.ENDS MBLKS/E
*Teilnetzwerk MBLKS0, wird im Netzwerk GMATS aufgeloest
.SUBCKT MBLKS0/E
      .
      . Aufrufe des Teilnetzwerkes MBLKS0
      .
.ENDS MBLKS0/E
      .
      . Definition weiterer Teilnetzwerke, die in den Block- und
      . Basisnetzwerken aufgeloest werden
      .
*Steueranweisungen
.OPTIONS MEMORY=40000 CPUMAX=1200 CPUZYK=120
.OPTIONS XPMAX=0.1
.OUTLIST V(*.*)
.LEVEL *.*=1
.OP $T 1000 0
.TRAN T 400 0
.END
```

Abbildung 5.5: MSPICE–Eingabetext des Netzwerkes MEGA_0

Die Beispielrechnungen MEGA_0, MEGA_1 und MEGA_2 unterscheiden sich nur in der Art der Strukturierung des Netzwerkes.

Abbildung 5.5 zeigt die von Schaltkreisentwerfern mit Hilfe von MSPICE definierte Struktur des Netzwerkes MEGA_0. Das Netzwerk MBLKS modelliert Teile der Speichermatrix und wird als Unternetzwerk siebenmal im Blocknetzwerk GMATS aufgerufen. Deshalb führten die Entwerfer MBLKS als Basisnetzwerk ein.

MAGNUS	Version V1.0–2			
Analyse von :	MEGA_0			

Generierung der Gleichungen

Basisnetzwerk: MBLKS/E				
772 Gleichungen	5942	Matrixelemente	834	X-Elemente
	106	Koppelvariablen	0	LOAD-Wert(e)
Blocknetzwerk: LIRA1/T				
725 Gleichungen	3516	Matrixelemente	3064	X-Elemente
	94	Koppelvariablen	2319	LOAD-Wert(e)
Blocknetzwerk: RRERA1/T				
708 Gleichungen	3540	Matrixelemente	1361	X-Elemente
	68	Koppelvariablen	636	LOAD-Wert(e)
Blocknetzwerk: GMATS/T				
1569 Gleichungen	46437	Matrixelemente	1632	X-Elemente
	850	Koppelvariablen	54	LOAD-Wert(e)
Hauptnetzwerk: MEGA_0/E				
270 Gleichungen	12840	Matrixelemente	3243	X-Elemente
	270	Koppelvariablen	2946	LOAD-Wert(e)

Gleichungssystem mit 3 Block- und 7 Basis-Teilsystemen
insgesamt:

8676 Gleichungen	107927	Matrixelemente	15138	X-Elemente
	2024	Koppelvariablen	5955	LOAD-Wert(e)

Abbildung 5.6: MAGNUS–Protokoll der Gleichungssystemgenerierung für das Netzwerk MEGA_0

In Abbildung 5.6 sind die Ergebnisse der Generierung des Netzwerkgleichungssystems protokolliert. Aufgrund der Steueranweisungen .OP $T und .OP T wurde nach einer Arbeitspunktbestimmung eine dynamische Netzwerkanalyse durchgeführt. Trotz des Einsatzes des Gauß–Seidel–Relaxationsverfahrens nach Abschnitt 4.2.2.3. ergab sich ein erheblicher Rechenzeitbedarf für die beiden Analysen, insgesamt fast 22 Stunden auf einem Rechner des Typs VAXstation 4000–60 (Abbildung 5.7 – 5.8).

DC - ANALYSE : EINSCHWINGVORGANG

STATISTIK

ANZAHL DER INTEGRATIONSSCHRITTE: 496
ANZAHL DER ABLEHNUNGEN: 1

NETZWERKNAME	DIM	ABL	NEW	FKT	JAC	PIV	FAK	LIN
LIRA1/T	725	6	1265	1265	502	1	501	1264
RERA1/T	708	16	1385	1385	542	1	537	1380
MBLKS/E	825	4	1381	1381	688	1	687	39171
MBLKS/E	825	4	1380	1380	684	0	683	39012
MBLKS/E	825	4	1380	1380	683	0	682	38959
MBLKS/E	825	4	1380	1380	697	0	696	39701
MBLKS/E	825	4	1380	1380	697	0	696	39701
MBLKS/E	825	4	1380	1380	683	0	682	38959
MBLKS/E	825	4	1380	1380	684	0	683	39012
GMATS/T	1569	4	1380	1380	730	9	771	1380
MEGA_0/E	270	0	0	0	0	0	0	0

Analysezeit : 8865.00 sec.

Abbildung 5.7 Protokoll der DC–Analyse des Netzwerkes MEGA_0 auf einer VAXstation 4000–60

TR - ANALYSE : GAUSS-SEIDEL-NEWTON-MEHREBENENVERFAHREN

STATISTIK

ANZAHL DER INTEGRATIONSSCHRITTE: 5801
ANZAHL DER ABLEHNUNGEN: 4

NETZWERKNAME	DIM	ABL	NEW	FKT	JAC	PIV	FAK	LIN
LIRA1/T	725	76	11102	11102	3166	1	3134	11069
RERA1/T	708	224	14603	14603	5809	1	5775	14569
MBLKS/E	825	44	11091	11091	3808	1	3803	223717
MBLKS/E	825	44	11086	11086	3774	0	3773	222180
MBLKS/E	825	44	11086	11086	3778	0	3777	222392
MBLKS/E	825	44	11086	11086	3771	0	3770	222021
MBLKS/E	825	44	11086	11086	3775	0	3774	222233
MBLKS/E	825	44	11086	11086	3777	0	3776	222339
MBLKS/E	825	44	11086	11086	3776	0	3775	222286
GMATS/T	1569	44	11086	11086	4325	141	4591	11072
MEGA_0/E	270	0	0	0	0	0	0	0

Analysezeit : 58115.93 sec.

Abbildung 5.8 Protokoll der TR–Analyse des Netzwerkes MEGA_0 auf einer VAXstation 4000–60

Die Ursache für diese lange Rechenzeit liegt in der Anzahl von 59 Klemmen des Basisnetzwerkes MBLKS, d.h., aufgrund des zwischen der Block– und Basisebene angewandten 2–Ebenen–Newtonverfahrens sind unter Umständen pro Basisnetzwerk in jedem Newtonschritt 118 lineare Gleichungssysteme mit 825 Variablen zu lösen. Im Laufe der dynamischen Analyse ändern sich zusätzlich über 140mal die Besetzungsstruktur der Koeffizientenmatrix für die Darstellung des Klemmenverhaltens der Basisnetzwerke. Dies führt zu den aufgelisteten sehr rechenzeitintensiven 141 Neupivotisierungen des Gleichungssystems des Blocknetzwerkes GMATS.

Mit Hilfe von Umstrukturierungen des Netzwerkes MEGA_0 konnte der Rechenzeitbedarf für die Netzwerkanalysen um ca. 75 % gesenkt werden. Abbildung 5.9 beschreibt die neue Netzwerkstruktur MEGA_1, bei der zusätzlich zu den Blocknetzwerken RERA1 und LIRA1 die aus Abbildung 5.5 bekannte Struktur GMATS–MBLKS in 8 Blocknetzwerke MBLKS0 bis MBLKS7 aufgelöst wurde. Die Markierung der Blocknetzwerkklemmen wurde für alle E–Klemmen aus der Struktur GMATS–MBLKS übernommen. Alle weiteren Klemmen wurden mit der Markierung U versehen. Deshalb wird vom Programmsystem MAGNUS das zusätzliche Blocknetzwerk *******/X erzeugt (vgl. Abbildung 5.10). Die Anzahl der Gleichungen des Netzwerkes *******/X ist gleich der Anzahl der Hauptnetzwerkknoten, die mit U–markierten Blocknetzwerkknoten korrespondieren (vgl. Abschnitt 3.1.2.).

Bei der Analyse des Netzwerkes MEGA_1 auf einer VAXstation 4000–60 treten die Vorteile des Relaxationsverfahrens aus Abschnitt 4.2.2.3. deutlich in Erscheinung. Der Iterationsprozeß zwischen den neuen Blocknetzwerken MBLKS0 – MBLKS7 ersetzt wirkungsvoll das 2–Ebenen–Newtonverfahren zwischen den Netzwerken GMATS und den Aufrufen des Netzwerkes MBLKS.
Bei fast gleicher Zeitschrittanzahl in der dynamischen und statischen Analyse belegen die Simulationsergebnisse (Abbildung 5.11 – 5.12) eindeutig die Vorteile des von uns entwickelten Simulationsverfahrens.

```
Simulationsbeispiel MEGA (4-Mbit-DRAM), Variante MEGA_1
XL PWANNE A0 A1 A10 ... LIRA1
XR PWANNE NRAS NCAS ... RERA1
XBLK0 PWANNE A0Y NA0Y ... MBLKS0
XBLK1 PWANNE A0Y NA0Y ... MBLKS1
XBLK2 PWANNE A0Y NA0Y ... MBLKS2
XBLK3 PWANNE A0Y NA0Y ... MBLKS3
XBLK4 PWANNE A0Y NA0Y ... MBLKS4
XBLK5 PWANNE A0Y NA0Y ... MBLKS5
XBLK6 PWANNE A0Y NA0Y ... MBLKS6
XBLK7 PWANNE A0Y NA0Y ... MBLKS7
    .
    . Referenzspannungen und Spannungsquellen mit Erregungsfunktionen
    .
*Blocknetzwerk MBLKS0
.SUBCKT MBLKS0/E
+ E(PWANNE A0Y NA0Y ... ) U(DLA0 NLA0 ... ) E(SPG NWANNE ... )
    .
    . Aufrufe weiterer Netzwerkelemente und Teilnetzwerke
    .
.ENDS MBLKS0/E
    .
    . Analoge Definitionen der Blocknetzwerke MBLKS1 bis MBLKS7
    .
*Blocknetzwerk RERA1
.SUBCKT RERA1/T
+ E(PWANNE NRAS NCAS ... ) A(TBM4 TM8B ... ) E(SPG NWANNE)
    .
    . Aufrufe weiterer Netzwerkelemente und Teilnetzwerke
    .
.ENDS RERA1/T
*Blocknetzwerk LIRA1
.SUBCKT LIRA1/T
+ E(PWANNE A0 A1 ... ) A(A0X A0Y ... ) E(SPG NWANNE ... )
    .
    . Aufrufe weiterer Netzwerkelemente und Teilnetzwerke
    .
.ENDS LIRA1/T
    .
    . Definition weiterer Teilnetzwerke, die in den Blocknetzwerken
    . aufgelöst werden
    .
*Steueranweisungen
.OPTIONS MEMORY=40000 CPUMAX=1200 CPUZYK=120
.OPTIONS XPMAX=0.1
.OUTLIST V(*.*)
.LEVEL *.*=1
.OP $T 1000 0
.TRAN T 400 0
.END
```

Abbildung 5.9: MSPICE–Eingabetext des Netzwerkes MEGA_1

MAGNUS Version V1.0–2
Analyse von : MEGA_1

Generierung der Gleichungen

Blocknetzwerk: *******/X					
16 Gleichungen	25	Matrixelemente	333	X-Elemente	
	16	Koppelvariablen	314	LOAD-Wert(e)	
Blocknetzwerk: LIRA1/T					
725 Gleichungen	3516	Matrixelemente	3064	X-Elemente	
	94	Koppelvariablen	2319	LOAD-Wert(e)	
Blocknetzwerk: RERA1/T					
708 Gleichungen	3532	Matrixelemente	1145	X-Elemente	
	68	Koppelvariablen	420	LOAD-Wert(e)	
Blocknetzwerk: MBLKS1/T					
825 Gleichungen	5995	Matrixelemente	887	X-Elemente	
	106	Koppelvariablen	53	LOAD-Wert(e)	
Blocknetzwerk: MBLKS2/T					
825 Gleichungen	5995	Matrixelemente	887	X-Elemente	
	106	Koppelvariablen	53	LOAD-Wert(e)	
Blocknetzwerk: MBLKS3/T					
825 Gleichungen	5995	Matrixelemente	887	X-Elemente	
	106	Koppelvariablen	53	LOAD-Wert(e)	
Blocknetzwerk: MBLKS4/T					
825 Gleichungen	5995	Matrixelemente	887	X-Elemente	
	106	Koppelvariablen	53	LOAD-Wert(e)	
Blocknetzwerk: MBLKS5/T					
825 Gleichungen	5995	Matrixelemente	887	X-Elemente	
	106	Koppelvariablen	53	LOAD-Wert(e)	
Blocknetzwerk: MBLKS6/T					
825 Gleichungen	5995	Matrixelemente	887	X-Elemente	
	106	Koppelvariablen	53	LOAD-Wert(e)	
Blocknetzwerk: MBLKS7/T					
825 Gleichungen	5995	Matrixelemente	887	X-Elemente	
	106	Koppelvariablen	53	LOAD-Wert(e)	
Blocknetzwerk: MBLKS0/T					
825 Gleichungen	5995	Matrixelemente	887	X-Elemente	
	106	Koppelvariablen	53	LOAD-Wert(e)	
Hauptnetzwerk: MEGA_1/E					
1026 Gleichungen	53214	Matrixelemente	4062	X-Elemente	
	1026	Koppelvariablen	3009	LOAD-Wert(e)	

Gleichungssystem mit 11 Block- und 0 Basis-Teilsystemen
insgesamt:

9075 Gleichungen	108247	Matrixelemente	15700	X-Elemente
	2052	Koppelvariablen	6486	LOAD-Wert(e)

Abbildung 5.10: MAGNUS–Protokoll der Gleichungssystemgenerierung für das Netzwerk MEGA_1

DC - ANALYSE : EINSCHWINGVORGANG

STATISTIK

ANZAHL DER INTEGRATIONSSCHRITTE: 496
ANZAHL DER ABLEHNUNGEN: 0

NETZWERKNAME	DIM	ABL	NEW	FKT	JAC	PIV	FAK	LIN
*******/X	16	3	650	650	24	1	21	647
LIRA1/T	725	8	1274	1274	508	1	507	1273
RERA1/T	708	12	1357	1357	527	1	523	1353
MBLKS1/T	825	6	1234	1234	413	1	410	1231
MBLKS2/T	825	9	1286	1286	437	1	432	1281
MBLKS3/T	825	7	1290	1290	441	1	438	1287
MBLKS4/T	825	7	1328	1328	452	1	449	1325
MBLKS5/T	825	7	1328	1328	452	1	449	1325
MBLKS6/T	825	7	1290	1290	441	1	438	1287
MBLKS7/T	825	9	1286	1286	437	1	432	1281
MBLKS0/T	825	6	1234	1234	413	1	410	1231
MEGA_1/E	1026	0	0	0	0	0	0	0

Analysezeit : 2390.10 sec.

Abbildung 5.11 Protokoll der DC–Analyse des Netzwerkes MEGA_1 auf einer VAXstation 4000–60

TR - ANALYSE : GAUSS-SEIDEL-NEWTON-MEHREBENENVERFAHREN

STATISTIK

ANZAHL DER INTEGRATIONSSCHRITTE: 5840
ANZAHL DER ABLEHNUNGEN: 3

NETZWERKNAME	DIM	ABL	NEW	FKT	JAC	PIV	FAK	LIN
*******/X	16	16	7167	7167	257	1	241	7151
LIRA1/T	725	79	11318	11318	3204	1	3172	11286
RERA1/T	708	222	14799	14799	6017	2	5983	14764
MBLKS1/T	825	20	8555	8555	1588	1	1576	8543
MBLKS2/T	825	15	7843	7843	1308	1	1300	7835
MBLKS3/T	825	15	7846	7846	1304	1	1296	7838
MBLKS4/T	825	16	7803	7803	1269	1	1260	7794
MBLKS5/T	825	16	7818	7818	1284	1	1275	7809
MBLKS6/T	825	15	7855	7855	1315	1	1307	7847
MBLKS7/T	825	15	7868	7868	1328	1	1320	7860
MBLKS0/T	825	47	10721	10721	2787	1	2759	10693
MEGA_1/E	1026	0	0	0	0	0	0	0

Analysezeit : 14421.18 sec.

Abbildung 5.12 Protokoll der TR–Analyse des Netzwerkes MEGA_1 auf einer VAXstation 4000–60

Um zu demonstrieren, wie sich das Programmsystem MAGNUS bei der Analyse von Netzwerken mit sehr großen Teilnetzwerken verhält, wurde das Beispiel MEGA_2 erzeugt.
Es unterscheidet sich von MEGA_0 dadurch, daß die Basisnetzwerke MBLKS im Blocknetzwerk GMATS aufgelöst wurden. Das Gleichungssystem des Blocknetzwerkes GMATS umfaßt somit 5.860 Gleichungen, das sind mehr als 77 % der Gleichungen des Gesamtsystems (siehe Abbildung 5.13).
Für das in Abschnitt 4.2.2.3. beschriebene Relaxationsverfahren führt die hier beschriebene Auflösung der Struktur Block–/Basisnetzwerk aus MEGA_0, wobei das sehr große Blocknetzwerk GMATS entsteht, in der Regel auf größere Rechenzeiten, als die Umstrukturierung des Netzwerkes MEGA_0 in eine Folge etwa gleich großer Blocknetzwerke gemäß MEGA_1.

MAGNUS	Version V1.0–2			
Analyse von :	MEGA_2			
Generierung der Gleichungen				
Blocknetzwerk: LIRA1/T				
725 Gleichungen	3516	Matrixelemente	3036	X-Elemente
	94	Koppelvariablen	2291	LOAD-Wert(e)
Blocknetzwerk: RERA1/T				
708 Gleichungen	3540	Matrixelemente	1340	X-Elemente
	68	Koppelvariablen	615	LOAD-Wert(e)
Blocknetzwerk: GMATS/T				
5860 Gleichungen	46843	Matrixelemente	5923	X-Elemente
	108	Koppelvariablen	54	LOAD-Wert(e)
Hauptnetzwerk: MEGA_2/E				
270 Gleichungen	12840	Matrixelemente	3215	X-Elemente
	270	Koppelvariablen	2918	LOAD-Wert(e)
Gleichungssystem mit 3 Block- und 0 Basis-Teilsystemen				
insgesamt:				
7563 Gleichungen	66739	Matrixelemente	13514	X-Elemente
	540	Koppelvariablen	5878	LOAD-Wert(e)

Abbildung 5.13: MAGNUS–Protokoll der Gleichungssystemgenerierung für das Netzwerk MEGA_2

Diese Aussage wird durch die Rechenergebnisse auf einer VAXstation 4000–60 bestätigt (siehe Abb. 5.14 – 5.16). Auf diesem Rechner steigt die Analysezeit für das Beispiel MEGA_2 gegenüber der Analysezeit für das Beispiel MEGA_1 um über 60 %. Der wesentlichste Grund hierfür ist die drastische Erhöhung des Aufwandes zur Lösung der linearen Gleichungssysteme. Durch die Größe des dem Blocknetzwerk GMATS zugeordneten Gleichungssystems steigt allein der Aufwand für die Faktorisierung der Matrizen des Beispiels MEGA_2 gegenüber dem Beispiel MEGA_1 um etwa 50%.
Um zu analysieren, in welchen Programmteilen des Programmsystems MAGNUS die Hauptrechenzeit verbraucht wird, wurden für eine Reihe praxisrelevanter Beispiele mit jeweils mehreren Tausend Gleichungen Performance–Analysen durchgeführt. Bei diesen Analysen ergaben sich auf einer VAXstation 4000–60 durchschnittlich Anteile von ca. 40 % für die Auswertung der Gleichungen (Berechnung von Funktionswerten und Jacobimatrixelementen) und von ca. 50 % für die Lösung der linearen Systeme. In einigen Ausnahmefällen (vgl. das hier beschriebene Beispiel MEGA_0) kann sich dieses Verhältnis jedoch erheblich verschieben. Bei ungünstiger Strukturierung des Netzwerkes (Anzahl der Klemmen erreicht die gleiche Größenordnung, wie die Anzahl der inneren Knoten) steigt der Anteil für die Lösung der linearen Systeme bis über 70 %, was einen wesentlichen Anstieg der Gesamtrechenzeit zur Folge hat (siehe Abbildung 5.18).

Für experimentelle Zwecke wurde das Programmsystem MAGNUS unter UNIX auf CRAY–Vektorrechner portiert. Es sollte untersucht werden, inwieweit das Netzwerkanalyseprogramm in seiner gegenwärtigen Implementation für die Anwendung auf Vektorrechnern vorbereitet ist. Dabei haben wir unsere Betrachtungen auf die beiden Programmfunktionen (Auswertung der Gleichungen und Lösung der linearen Systeme) beschränkt, die den Hauptanteil an Rechenzeit verbrauchen.

DC - ANALYSE : EINSCHWINGVORGANG

STATISTIK

ANZAHL DER INTEGRATIONSSCHRITTE: 489
ANZAHL DER ABLEHNUNGEN: 0

NETZWERKNAME	DIM	ABL	NEW	FKT	JAC	PIV	FAK	LIN
LIRA1/T	725	8	1263	1263	500	1	499	1262
RERA1/T	708	12	1356	1356	532	1	529	1353
GMATS/T	5860	5	1376	1376	707	1	705	1374
MEGA_2/E	270	0	0	0	0	0	0	0

Analysezeit : 3713.87 sec.

Abbildung 5.14 Protokoll der DC–Analyse des Netzwerkes MEGA_2 auf einer VAXstation 4000–60

TR - ANALYSE : GAUSS-SEIDEL-NEWTON-MEHREBENENVERFAHREN

STATISTIK

ANZAHL DER INTEGRATIONSSCHRITTE: 5814
ANZAHL DER ABLEHNUNGEN: 5

NETZWERKNAME	DIM	ABL	NEW	FKT	JAC	PIV	FAK	LIN
LIRA1/T	725	81	11154	11154	3180	1	3149	11123
RERA1/T	708	214	14560	14560	5820	2	5748	14523
GMATS/T	5860	38	11232	11232	4064	1	4045	11213
MEGA_2/E	270	0	0	0	0	0	0	0

Analysezeit : 23818.31 sec.

Abbildung 5.15 Protokoll der TR–Analyse des Netzwerkes MEGA_2 auf einer VAXstation 4000–60

Im Programmsystem MAGNUS erfolgt die Berechnung der Funktionswerte teilsystemweise durch interpretative Abarbeitung verschiedener Operationstypen (Basisroutinen), die mit Index– bzw. Adress–Vektorlisten arbeiten. Daher bestand eine einfache Möglichkeit zur Vektorisierung dieser Operationen. Da diese Vektoroperationen teilsystemweise ausgeführt werden, ergeben sich, wie bei der Lösung der linearen Systeme, für größere Teilsysteme in der Regel auch größere Vektorlängen. Durch die Anwendung von Vektorisierungsdirektiven ergab sich für diese Programmfunktion eine durchschnittliche Beschleunigung gegenüber der nicht vektorisierten Variante um den Faktor 3.

Durch Programmmodifikationen und das Einfügen von Vektorisierungsdirektiven in die Subroutinen zur Lösung der linearen Systeme wurde eine Halbierung der Rechenzeit für diese Programme und damit eine Reduktion der Gesamtrechenzeit um bis zu 30 % erreicht.

An der CRAY Y–MP2E/264 beträgt für die Testbeispiele mit mehr als 5000 Gleichungen der Anteil der Programme zur Lösung der linearen Systeme an der Gesamtrechenzeit ca. 50 % und der Anteil der Programme zur Auswertung der Funktion und der Jacobimatrix ca. 20 %.
Dabei wird in den Programmen zur Lösung der linearen Systeme in den Vektorunits der CRAY eine durchschnittliche Performance von mehr als 6 VMflops erreicht. Bei den Programmen zur Auswertung der Gleichungen liegt diese Rate bei knapp 16 VMflops.

Beispiel	Anzahl Blocksyst.	Anzahl Basissyst.	Anzahl Gleichungen	Anzahl Zeitschritte	CPU–Zeit DC–Anal.	CPU–Zeit TR–Anal.
MEGA_0	3	7	8676	5801	8865 sec.	58116 sec.
MEGA_1	10 + 1	–	9075	5840	2390 sec.	14421 sec.
MEGA_2	3	–	7563	5814	3713 sec.	23818 sec.

Abbildung 5.16: Ergebnisse auf einer VAXstation 4000–60

Beispiel	Anzahl Blocksyst.	Anzahl Basissyst.	Anzahl Gleichungen	Anzahl Zeitschritte	CPU–Zeit DC–Anal.	CPU–Zeit TR–Anal.
MEGA_0	3	7	8676	5796	1132 sec.	6634 sec.
MEGA_1	10 + 1	–	9075	5839	286 sec.	1639 sec.
MEGA_2	3	–	7563	5819	197 sec.	1186 sec.

Abbildung 5.17: Ergebnisse auf einer CRAY Y–MP2E/264

Obwohl die Mehrzahl der Basisroutinen für die Berechnung der Funktionswerte und der Jacobimatrixelemente Raten zwischen 150 und 300 VMflops erreicht, wird keine höhere Performance erreicht, da ihr Anteil an den Operationen zur Gleichungsgenerierung weit unter 10 % liegt. Die am häufigsten abzuarbeitenden Operationen sind indizierte Vektor–Additionen und –Multiplikationen, die sich nur mit wesentlich geringerer Effizienz vektorisieren lassen.
Die durchschnittliche Gleitkommaleistung des Programmsystems MAGNUS an einer CRAY Y-MP2E/264 liegt bei ca. 10 Mflops. In der Abbildung 5.18 sind die ermittelten Rechenzeitanteile an einer VAXstation 4000–60 den entsprechenden Anteilen an einer CRAY Y–MP2E/26 gegenübergestellt.

Generell ergibt sich an der CRAY im Vergleich zur VAXstation ein geringerer Anteil für die Funktionsberechnung, ein etwa gleichbleibender Anteil für die Lösung der linearen Systeme und ein höherer Anteil für die verbleibenden Operationen zur Lösung der Differentialgleichung. Dieser Umstand ist wesentlich auf die unterschiedliche Effektivität bei der Vektorisierung dieser Programmteile zurückzuführen. Bei größerer Dimension der Gleichungssyteme steigt wegen der größeren Vektorlänge die Effektivität der vektorisierten Programme sowohl bei der Berechnung des Gleichungssystems, als auch bei der Lösung der linearen Systeme. Somit rechnet das Beispiel MEGA_2 an der CRAY schneller, als das Beispiel MEGA_T1. An der VAXstation ist es dagegen genau umgekehrt. Die durchschnittliche Performance dieser Programmanteile ist in Abbildung 5.19 dargestellt.

Die relativ hohe Performance bei der Lösung der linearen Systeme des Beispiels MEGA_0 ist darauf zurückzuführen, daß in diesem Fall die vergleichsweise höhere Performance der Vektorroutinen der Vorwärts– und Rückwärtsrechnung gegenüber der Performance der Routinen der Faktorisierung dominiert. Die Ursachen hierfür liegen in der beschriebenen Strukturierung des Netzwerkes und dabei insbesondere in der im Vergleich zur Anzahl der inneren Knoten (Variablen) relativ hohen Anzahl an Randknoten (Koppelvariablen) der Basisnetzwerke.
Die höhere Leistung bei der Funktionswertberechnung des Beispiels MEGA_2 ist auf größere Vektorlängen zurückzuführen, die durch die Größe des Teilsystems vom Teilnetzwerk GMATS (5860 Gleichungen) entstehen.

Beispiel	VAXstation 4000–60 Funktions–berechnung	Lösung linearer Systeme	Lösung Differential-gleichung	CRAY Y–MP2E/26 Funktions–berechnung	Lösung linearer Systeme	Lösung Differential-gleichung
MEGA_0	12,2 %	77,0 %	7,4 %	7,6 %	64,7 %	22,7 %
MEGA_1	41,9 %	39,7 %	15,1 %	20,4 %	44,1 %	25,8 %
MEGA_2	34,9 %	51,8 %	11,4 %	15,7 %	57,3 %	17,9 %

Abbildung 5.18: Hauptanteile an verbrauchter CPU–Zeit

Beispiel	Funktionswert–berechnung	Lösung linearer Gleichungssyteme	Lösung von Differentialgleichungen	gesamt
MEGA_0	12.5 Mflops	15.4 Mflops	4.2 Mflops	12.4 Mflops
MEGA_1	12.8 Mflops	9.6 Mflops	7.2 Mflops	8.4 Mflops
MEGA_2	19.0 Mflops	8.1 Mflops	7.5 Mflops	9.2 Mflops

Abbildung 5.19: MAGNUS–Performance auf CRAY Y–MP2E/264

Insgesamt wird von MAGNUS in der gegenwärtigen Form nur eine mittlere Performance auf dem CRAY Vektorrechner erreicht. Wesentlichster Grund dafür sind Einschränkungen, die schon aus der Modellierung resultieren. Zum einen betrifft das die Netzwerkmodelle und zum anderen die Technik der sparsam besetzten Matrizen. So wird weder bei der Lösung der lineren Gleichungen noch bei der Berechnung der Funktionswerte und der Jacobimatrixelemente direkt über kompakt gespeicherte Vektoren operiert. Alle wesentlichen Operatinen erfolgen über Indexvektorlisten.

Literatur

[BBG88] J. Borchardt, I. Bremer, F. Grund, D. Horn, K. R. Schneider, M. Uhle und W. Zielas: *Handbuch des Netzwerkanalyseprogramms MAGNUS.* Karl–Weierstraß–Institut für Mathematik, Berlin, 1988.

[BBG89a] J. Borchardt, I. Bremer, F. Grund, D. Horn und M. Uhle: *Erweiterung des Simulators MAGNUS zum Mixed–Mode–Elektriksimulator. A2–Bericht zum Thema "Netzwerk- und Timinganalyse"*, Karl–Weierstraß–Institut für Mathematik, Berlin, 1989.

[BBG89b] J. Borchardt, I. Bremer, F. Grund, D. Horn und M. Uhle: *MAGNUS – Mehrstufige Analyse großer Netzwerke auf der Basis von Untersystemen.* Proc. 3. Tagung Schaltkreisentwurf Dresden, S. 148–152, Dresden, 1989.

[BBG90] J. Borchardt, I. Bremer, F. Grund, D. Horn und M. Uhle: *MAGNUS – Systemhandbuch.* Karl–Weierstraß–Institut für Mathematik, Berlin, 1990.

[BBG93] J. Borchardt, I. Bremer, F. Grund, D. Horn und M. Uhle: *MAGNUS – Nutzerhandbuch.* Institut für Angewandte Analysis und Stochastik, Berlin, 1993.

[CGK75] B. R. Chawla, H. K. Gummel und P. Kozah: *MOTIS – An MOS timing simulator.* IEEE Trans. CAS–22 (1975), S. 901–909.

[ELD82] W. L. Engl, R. Laur und H. K. Dirks: *MEDUSA – A simulator for modular circuits.* IEEE Trans. CAD–1 (1982) 2, S. 85–93.

[EMR77] H. Elschner, A. Möschwitzer und A. Reibiger: *Rechnergestützte Analyse in der Elektronik.* Verlag Technik, Berlin, 1977.

[EP85] G. Elst und G. Pönisch: *On two–level Newton methods for the analysis of large–scale nonlinear networks.* Proc. ECCTD'85, S. 165–169, Prag, 1985.

[FS88] U. Feldmann und R. Schultz: *TITAN: A universal circuit simulator for latency exploitation.* Proc. ESSCIRC'88, S. 183–185, Manchester, 1988.

[GBG85] F. Grund, J. Borchardt, K. Gärtner, B. Horn, D. Horn und W. Zielas: *Basic principles of the network analysis program ANNET.* Proc. ECCTD'85, S. 113–116, Prag, 1985.

[Gea88] C. W. Gear: *Differential–algebraic equation index transformations.* SIAM J. Sci. Stat. Comput. 9 (1988) 1, S. 39–47.

[Gea71] C. W. Gear: *Simultaneous numerical solution of differential–algebraic equations.* IEEE Trans. CT–18 (1971) 1, S. 89–95.

[Gru82] F. Grund: *Numerische Lösung von Gleichungssystemen mit sparsam besetzten Matrizen.* Tagung "Numerischer Lösung von Differentialgleichungen", Report R–Math–01/83, S. 33-44, Matzlow/Garwitz, 1982.

[Gru93] F. Grund: *Numerical solution of hierarchically structured systems of algebraic–differential equations.* Institut für Angewandte Analysis und Stochastik, Preprint 71, Berlin, 1993.

[HBG71] G. D. Hachtel, R. K. Brayton und F. G. Gustavson: *The sparse tableau approach to network analysis und design.* IEEE Trans. CT–18 (1971) 1, S. 101–113.

[HPS87] B. Hennion, Y. Paradis und P. Senn: *ELDO: A general purpose third generation circuit simulator based on the O.S.R. method.* Proc. ECCTD, S. 113–119, Paris, 1987.

[HRB75] C. Ho, A. E. Ruehli und P. A. Brennan: *The modified nodal approach to network analysis.* IEEE Trans. CAS–22 (1975) 6, S. 504–509.

[HS84] W. Hoyer und J. W. Schmidt: *Newton–type decomposition methods for equation arising in network analysis.* ZAMM 64 (1984) 9, S. 397–405.

[KD69] E. Kuh und C. Desoer: *Basic circuit theory.* McGraw Hill, New York, 1969.

[LRSV82] E. Lelarasmee, A. E. Ruehli und A. L. Sangiovanni-Vincentelli: *The waveform relaxation method for time domain analysis of large scale integrated circuits.* IEEE Trans. CAD–1 (1982) 3, S. 131–145.

[Mar57] H. M. Markowitz: *The elimination form of the inverse und its application to linear programming.* Management Sci. 3 (1957), S. 255–269.

[NSV83] A. R. Newton und A. L. Sangiovanni-Vincentelli: *Relaxation–based electrical simulation.* IEEE Trans. ED–30 (1983) 9, S. 1184–1207.

[OR70] J. M. Ortega und W. C. Rheinboldt: *Iterative solution of nonlinear equations in several variables.* Academic Press, New York, 1970.

[Rei92] S. Reich: *Differential–algebraic equations und applications in circuit theory.* AEÜ 46 (1992), S. 298–304.

[RSt76] A. Reibiger und B. Straube: *Ein Beitrag zum deduktiven Aufbau der Netzwerktheorie, Teil I bis Teil V.* TU Dresden, Preprint 09–08–76 bis 09–12–76, Dresden, 1976.

[RSch76] K. Reinschke und P. Schwarz: *Verfahren zur rechnergestützten Analyse linearer Netzwerke.* Akademie–Verlag, Berlin, 1976.

[RSVH79] N. B. G. Rabbat, A. L. Sangiovanni-Vincentelli und H. Y. Hsieh: *A multilevel Newton algorithm with macromodelling und latency for the analysis of large–scale nonlinear circuits in the time domain.* IEEE Trans. CAS–26 (1979) 9, S. 733–741.

[Sch79] H. Schwetlick: *Numerische Lösung nichtlinearer Gleichungen.* Deutscher Verlag der Wissenschaften, Berlin, 1979.

[Sch87] P. Schwarz: *Timing–Simulation beim Schaltkreisentwurf.* Proc. 32. IWK TH Ilmenau, Reihe B2, S. 245–248, Ilmenau, 1987.

[Uhl87] M. Uhle: *Analyse großer elektrischer Netzwerke, Methoden und Verfahren. Bericht zum Projekt "MAGNUS"*, Karl–Weierstraß–Institut für Mathematik, Berlin, 1987.

[Uhl90a] M. Uhle: *Zur Konzeption moderner Schaltkreissimulatoren.* ITG – Diskussionssitzung "Neue Anwendungen theoretischer Konzepte in der Elektrotechnik", Hannover, 1990.

[Uhl90b] M. Uhle: *Zur Strukturierung großer elektrischer Netzwerke mit Hilfe des Signalflusses.* GME/GI/ITG–Fachtagung "Rechnergestützter Entwurf und Architektur mikroelektronischer Systeme", Dortmund, 1990.

[Uhl91] M. Uhle: *Zur Parallelisierung von Iterationsverfahren für die Schaltkreissimulation.* ITG Diskussionssitzung "Entwicklung von Analogschaltungen", Paderborn, 1991.

[YT85] F. Yamamoto und S. Takahashi: *Vectorized LU decomposition algorithms for large–scale circuit simulation.* IEEE Trans. CAD–4 (1985), S. 232–239.

Recent publications of the
Weierstraß–Institut für Angewandte Analysis und Stochastik

Reports 1992

1. K.–H. Hoffmann and P. Knabner (eds.): Filtration and nonlinear diffusion processes. V. Workshop.
2. G. Simonett: Zentrumsmannigfaltigkeiten für quasilineare parabolische Gleichungen.
3. A. Bovier, F. Koukiou: Mathematical physics of disordered systems. Abstracts of a workshop held at the Centre International de Recontres Mathématiques, Marseille, July 27–31, 1992.
4. B. Fiedler: Global pathfollowing of homoclinic orbits in two–parameter flows.

Reports 1993

5. S. Hengst (ed.): GAMM–Seminar on Multigrid–Methods. Gosen, Germany, September 21 – 25, 1992.
6. H. Gajewski: Analysis und Numerik von Ladungstransport in Halbleitern.
7. B. Sandstede: Verzweigungstheorie homokliner Verdopplungen.
8. R. Funke: Stochastische Modellierung dynamischer Systeme in der Ökologie.