4. Grundlagen der Bewertung

Um leistungsfähige und technisch- ökonomisch sinnvolle Schaltungsanordnungen zu schaffen, sind verschiedene Lösungsansätze zu erarbeiten und zu bewerten. Teillösungen, die sich als zweckmäßig erwiesen haben, sind zu Systemlösungen zusammenzufassen. Dafür sind Bewertungsmaßstäbe erforderlich: Schaltungsanordnungen sind nach ihrer Leistungsfähigkeit zu bewerten, Algorithmen nach der Eignung zur Vergegenständlichung und Aufwendungen nach ihrer Nützlichkeit. Weiterhin sind die verschiedenen Lösungsansätze untereinander zu vergleichen.

4.1. Bewertung der Schaltungsanordnungen

Zunächst wird das absolute Leistungsvermögen von Hardware untersucht; die Aufwendungen bleiben also außer Betracht. Allgemein übliche Bewertungsgrundlagen sind:[1]

1. Ausführungszeiten bestimmter Algorithmen (Anwendungsprogramme)
2. Ausführungszeiten "repräsentativer" Algorithmen ("benchmark"- Programme)
3. Ausführungszeiten vergegenständlichter Algorithmen (bei üblichen Rechnern in herkömmlicher Redeweise: die Ausführungszeiten der Maschinenbefehle).

Es ist klar, daß brauchbare Angaben nur selten analytisch oder durch einmalige Probeläufe zu erhalten sind, sondern daß Mittelwerte bzw. Erwartungswerte gebildet werden müssen.

Die Messung der Ausführungszeiten von Anwendungsprogrammen liefert einem Nutzer, der nur an bestimmten Algorithmen interessiert ist, gut auswertbare Vergleichsdaten über die Eignung verschiedener Maschinen. Sie hat aber folgende Nachteile:

- Anwendbarkeit nur auf fertige Hardware- Software- Komplexe (Maschinen und Programme müssen vorhanden sein); kaum geeignet für die Bewertung von Lösungsansätzen neuer Schaltungsstrukturen (Simulation ist aufwendig und langsam)

- sehr beschränkte Aussagekraft hinsichtlich des allgemeinen Leistungsvermögens

- vergleichsweise hohes Bewertungsrisiko: geringe Änderungen in den Algorithmen, Programmen bzw. Compilern können sich erfahrungsgemäß in manchen Maschinen deutlich auf die Laufzeiten auswirken, in anderen nicht.

Derartige Messungen testen nicht nur die Hardware, sondern auch den Compiler und die Kunstfertigkeit des Programmierers.

---

1 Derzeitig gibt es noch kein gesichertes Fachwissen darüber, wie ausgewogene ("well-balanced") Hardware- Software- Systeme zu entwerfen sind, die hohe Leistungen für normale Nutzer liefern. Man kann sich für die Leistungsbewertung nur auf empirische Resultate verlassen. (Nach: /213/.)

So ist in /264/ dargestellt, daß auf der CRAY-1 die rechts angegebene Schleife um 25% schneller läuft als die linke[1]:

```
  DO 1Ø   I = 1,N                    DO 1Ø   I = 1,N
      Y(I) = A*X(I)+Y(I)                 Y(I) = A*X(I)+(Y(I))
 1Ø CONTINUE                        1Ø CONTINUE
```

Praktische Erfahrungen zeigen, daß allein ein Wechsel des Compilers das Laufzeitverhalten der Programme beträchtlich verändern kann.

Repräsentative Algorithmen, die eindeutig dokumentiert sind (z. B. in einer verbreiteten Programmiersprache), sind für das überschlägige Vergleichen zweckmäßiger; sie gestatten es, die Eignung von Maschinen für bestimmte Klassen von Algorithmen zu beurteilen ("benchmark"- Tests). Mit einigen Tests kann man das Leistungsvermögen bezüglich bestimmter Operationen recht genau erfassen.[2] So kann man beim bekannten LINPACK- Benchmark (Lösung linearer Gleichungssysteme) die Gleitkommaoperationen (Addition und Multiplikation) auszählen: ein System aus n Gleichungen erfordert

$$\frac{2}{3}n^3 + 2n^2 + O(n)$$

solche Operationen.
Die gemessenen Werte weichen erheblich von den Angaben der Maximalleistung ("peak performance") ab (Tafel 4).[3]

Hingegen lassen sich die Ausführungszeiten der vergegenständlichten Algorithmen ("Maschinenbefehle") an sich recht einfach aus einem hinreichend detaillierten Schaltungsentwurf bestimmen (durch Auszählen der Taktzyklen und einige statistische Annahmen, z. B. hinsichtlich der Vermittlungszeiten von Speicherzugriffen und der Trefferraten bei Cache- Speichern). Besonders beliebt ist in der Praxis die Annahme der jeweils günstigsten Verhältnisse: so kommen die meisten der üblichen MIPS- bzw. MFLOP- Angaben zustande.
Ein Blick in Tafel 4 zeigt die Fragwürdigkeit solcher Angaben. Etwas bessere Vergleichswerte liefern die bekannten statistischen Befehlsverteilungen (Mix- Werte, z. B. Gibson- Mix, GPO-Mix usw.). Sie können zu Vergleichszwecken recht einfach berechnet werden; man sollte aber nicht errechnete Zahlen mit gemessenen vergleichen, und die betreffenden Maschinenarchitekturen sollten einigermaßen vergleichbar sein.[4]
Des weiteren hat sich gezeigt, daß eine auf gute Mix- Leistung hin entworfene Maschine in der Anwendungsleistung nicht immer im erwarteten Maße überlegen ist.[5]

---

1 Gilt für Übersetzung mit Fortran- Compiler Level 1.Ø9.
2 Auch dieses Verfahren ist ohne fertige Maschine (mit Betriebssystem und Compiler) kaum anwendbar.
3 Die Darstellung (einschließlich Tafel 4) stammt aus /161/; s. weiterhin /159/, /16Ø/.
4 Man vergleiche also nur CISC- Maschinen, RISC- Maschinen, Vektorprozessoren usw. jeweils untereinander.
5 Z. B. EC 1Ø55 im Vergleich mit EC 1Ø4Ø.

lfd. Nr.	Maschine	Zyklus (ns)	Prozessoren
1	Culler PSC	200	1
2	Multiflow TRACE 7/200	130	1
3	Convex C-1	100	1
4	SCS-40	45	1
5	FPS 264	38	1
6	Alliant FX/8	170	8
7	Amdahl 500	7,5	1
8	CRAY-1	12,5	1
9	CRAY X-MP-1	9,5	1
10	IBM 3090/VF-200	18,5	2
11	Amdahl 1100	7,5	1
12	NEC SX-1E	7	1
13	CDC CYBER 205	20	1
14	CRAY X-MP-2	9,5	2
15	IBM 3090/VF-400	18,5	4
16	Amdahl 1200	7,5	1
17	NEC SX-1	7	1
18	CRAY X-MP-4	9,5	4
19	Hitachi S-810/20	14	1
20	NEC SX-2	6	1
21	CRAY-2	4,1	4

lfd. Nr.	Leistung (MFLOP) maximal	LINPACK	Effizienz
1	5	2	0,4
2	15	6	0,4
3	20	3	0,15
4	44	8	0,18
5	54	5,6	0,1
6	94	7,6	0,08
7	133	14	0,11
8	160	12	0,075
9	210	24	0,11
10	216	12*	0,11 (0,056)
11	267	16	0,06
12	325	35	0,11
13	400	17	0,043
14	420	24*	0,11 (0,057)
15	432	12	0,11 (0,028)
16	533	18	0,034
17	650	39	0,06
18	840	24*	0,11 (0,029)
19	840	17	0,02
20	1300	46	0,035
21	2000	5*	0,03 (0,0075)

* Angabe für einen Prozessor

Tafel 4 Das Leistungsvermögen von Hochleistungsrechnern; mit dem LINPACK- "benchmark" gemessen

Hier sollen nicht nur bekannte bzw. von vornherein leicht vergleichbare Prinzipien untersucht werden, und es ist wünschenswert, Schaltungslösungen in einem frühen Bearbeitungsstand[1] überschlagsmäßig beurteilen zu können. Dafür wird ein Verfahren vorgeschlagen, das auf folgenden Überlegungen beruht:

Für einen einzelnen Algorithmus interessiert an sich nur die reine Ausführungszeit; die Zeit, die die Maschine braucht, um die gewünschten Resultate zu liefern.
Weiterhin kann man sich vorstellen, für den besagten Algorithmus eine Sondermaschine zu bauen. Daß programmgesteuerte Universalmaschinen verwendet werden, hat unter diesem Gesichtspunkt nur den Grund, daß es nicht durchführbar ist, für jeden Algorithmus eine Sondermaschine zu bauen. Die programmgesteuerte sequentielle Ausführung eines Algorithmus ist also nur ein Notbehelf, eben wegen der technisch- ökonomischen Gegebenheiten. Man kann deshalb die Abarbeitung von Befehlen eher als Störfaktor auffassen und nur die Argumente und Resultate der Algorithmen betrachten. Diese Werte sind binär codiert, und sie werden in aufeinanderfolgenden Taktzyklen von Speichermitteln zu Speichermitteln über Verbindungsleitungen und kombinatorische Netzwerke bewegt.
Für eine Sondermaschine ist die Frage nach "MIPS" an sich gegenstandslos. Solche Angaben sind nur sinnvoll, wenn es darum geht, den Geschwindigkeitszuwachs einer Sondermaschine im Vergleich zur Nutzung einer Universalmaschine für den selben Zweck zu beurteilen. Beispiel: Der leistungsentscheidende Ablauf eines Algorithmus erfordere 5Ø Befehle einer geläufigen Rechnerarchitektur. Eine Sondermaschine leiste dasselbe in 1 µs. Es müßte also ein Universalrechner mit 5Ø MIPS beschafft werden, um das Leistungsvermögen der Sondermaschine zu erreichen[2] (an diese Überlegung wird sich üblicherweise eine Kostenabschätzung anschließen). Solche Betrachtungen wurden z. B. in /243/ angestellt, um die Sinnfälligkeit konkret entworfener Sondermaschinen im Vergleich zu Universalrechnern beurteilen zu können. Hier stellt sich die Aufgabe gerade anders herum: das Leistungsvermögen der Sondermaschine ist bekannt, und es ist die Universalmaschine zu bewerten. Die Sondermaschine verkörpert die leistungsfähigste technisch beherrschbare Vergegenständlichung des Algorithmus. Sie erzeugt die Resultate in einer minimalen Anzahl von Maschinenzyklen.

Um ein Maß für die Verarbeitungsleistung zu gewinnen, ist es mithin ausreichend, zu zählen, wieviele Bits an Nutz- Information (Argumente und Resultate der jeweils betrachteten Algorithmen) in einer bestimmten Zeiteinheit verarbeitet bzw. erzeugt werden. Bezeichnungsvorschlag: "Effektive Bits je Sekunde" (EB/s; mit Faktoren $1Ø^6$ bzw. $1Ø^9$ dann MEB/s bzw. GEB/s).

---

1 Z. B. nach Ausarbeitung der Funktionsprinzipien und des Blockschaltbildes bzw. der Register- Transfer- Struktur.
2 Die Aussage "die Sondermaschine leistet x MIPS" ist falsch! Zutreffend ist vielmehr: "Die Sondermaschine ersetzt für den betreffenden Algorithmus einen Universalrechner von x MIPS".

Um dieses Leistungsmaß (im folgenden mit PM bezeichnet) zu bestimmen, wird in einem Intervall t_x gezählt, wieviele Zugriffe zur Nutz- Information (die durch $\mathcal{V}$ in (3.2) beschrieben wird) dabei ausgeführt werden (die Anzahl sei r).[1] Die Anzahl der Nutzbits in Zugriff i ($1 \leqslant i \leqslant r$) sei CARDBi. Dann gilt:

$$PM = \frac{1}{t_x} \sum_{i=1}^{r} CARDBi. \qquad (4.1)$$

Da die Ausführungszeiten der meisten Algorithmen datenabhängig sind, müssen in der Praxis Mittelwerte bzw. Erwartungswerte gebildet werden. Die Angaben müssen stets auf den jeweiligen Algorithmus bzw. Komplex von Algorithmen bezogen werden.
Für überschlägige Abschätzungen und Vergleiche eignet sich beispielsweise der Vorrat an elementaren Operatoren (numerische und nichtnumerische), der in eingeführten Programmiersprachen (z. B. Fortran, C, Ada) definiert ist.[2] Oft lassen sich die Operatoren direkt auf Maschinenbefehle abbilden; gelegentlich erfordert ein Operator eine Folge von Maschinenbefehlen (z. B.: Multiplikation in RISC- Architekturen). Es wird von Verzweigungen usw. abgesehen und eine lückenlose Folge von Operatoren angenommen, die gespeicherte Argumente verarbeiten und die Resultate wieder in einem allgemein zugänglichen Speicher ablegen.[3] Dem eigentlichen Vergleich wird dann eine sinnfällige Folge ("Mix") solcher Operatoren zugrunde gelegt, wobei die Ausführungszeit (t_x) und die unbedingt notwendigen Argument- und Resultattransporte (CARDBi) bestimmt werden.[4] Bei Hochleistungssystemen reicht es bisweilen aus, den Idealfall anzunehmen, daß alle Datenpfade und Verarbeitungswerke voll ausgelastet sind: es wird dann die maximale Datenrate betrachtet.[5]

4.2. Bewertung der Algorithmen

Um Algorithmen zur Vergegenständlichung auszuwählen, ist deren Eignung zu bewerten. Dabei geht es nicht um die Nützlichkeit oder universelle Anwendbarkeit - eine solche Vorauswahl wird als gegeben angenommen -, sondern um die grundsätzliche Möglichkeit, leistungsmäßig überlegene Schaltmittel entwerfen zu können. Die zeiteffektivste Form der Vergegenständlichung ist dann gegeben, wenn die Argument- und Resultatbits nur jeweils einmal transportiert werden müssen.[6]

---

1 Für künftige Maschinen sollte auch an technische Vorkehrungen zur Leistungsmessung gedacht werden.
2 Vgl. etwa Tafel 8 (S. 72).
3 Die Speicher müssen für andere Prozessoren bzw. für die Ein- und Ausgabe zugänglich sein. Register-Register-Verknüpfungen verlängern t_x, zählen aber nicht für CARDBi.
4 Datentransporte aus Emulationsgründen (z. B. zu Hilfsbereichen im RAM) verlängern t_x, zählen aber nicht für CARDBi.
5 Zugriffe zu Befehlen, Deskriptortabellen usw. verlängern manchmal (wenn nicht parallelisierbar) t_x, zählen aber nicht für CARDBi.
6 Dieser Ansatz wurde in /243/ erstmals beschrieben.

Um diesen Sachverhalt zu bewerten, wird der Begriff der Implementierungseffizienz[1] e_i wie folgt eingeführt:

$$e_i = \frac{\sum_{i=1}^{n} CARDB(A_i) + \sum_{j=1}^{m} CARDB(R_j)}{z \cdot (ARG\_LINES + RES\_LINES)} \qquad (4.2)$$

Bedeutung der Symbole:

- CARDB(A_i), CARDB(R_j): Anzahl der Bits, mit denen die Argumente A_i ($1 \leqslant i \leqslant n$) bzw. Resultate R_j ($1 \leqslant j \leqslant m$) jeweils codiert sind
- ARG_LINES, RES_LINES: Anzahl der genutzten Leitungen für die Zuführung der Argumente bzw. den Abtransport der Resultate
- z: Anzahl der Maschinenzyklen, die für die Bildung aller Resultate benötigt werden ($z \geqslant 1$).

Wird kein Resultat gespeichert (wenn beispielsweise der Algorithmus lediglich eine Bedingung prüft), so sind die Resultat-Terme gemäß der jeweiligen binären Codierung (z. B.: einzelne Bedingungsleitung, 2-bit- Bedingungscode o. ä.) anzusetzen.
Ist eine technisch gegebene Leitungszahl (z. B. aus ⌐¬ ablesbar) größer als die betreffende Anzahl an Bits, so gilt letztere.
Die Implementierungseffizienz ist eine dimensionslose Zahl im Intervall $\emptyset < e_i \leqslant 1$, und es ist ersichtlich, daß im günstigsten Fall $e_i = 1$ ist. Um das zu erreichen, müssen die beiden folgenden Sätze erfüllt sein:

1. $e_i = 1$ kann grundsätzlich nur dann erreicht werden, wenn für jeden Abschnitt des Resultats gilt, daß dieser durch Zuordnung (d. h. gemäß dem Schema von Bild 4) aus den jeweils aktuellen Abschnitten der Argumente erzeugt werden kann, wobei in diese Zuordnung höchstens noch Zustands- Information einbezogen ist, die eindeutig durch die zuvor verarbeiteten Abschnitte bestimmt ist.

2. Um $e_i = 1$ praktisch verwirklichen zu können, müssen die Abläufe des Algorithmus so zerlegbar sein, daß jeder Abschnitt der Resultate sowie erforderlichenfalls die Zustands- Information im Sinne von Satz 1 ausschließlich durch kombinatorische Verknüpfungen gemeinsam selektierbarer Abschnitte der Argumente sowie der besagten Zustands- Information des jeweils vorausgegangenen Verarbeitungszyklus gebildet wird.

Satz 1 muß stets erfüllt sein, da nur so gewährleistet ist, daß zur Gewinnung eines Resultatabschnittes keine zusätzlichen Zustandswechsel erforderlich sind. Man braucht hingegen solche Zustandswechsel, wenn auch nur ein Resultatabschnitt von Argumentabschnitten abhängt, die nicht im jeweils aktuellen Zyklus zugänglich sind oder in voraufgegangenen Zyklen bereits verarbeitet wurden: diese Argumentabschnitte sind nur mit

---

1 Die Bezeichnung aus /243/ wird zunächst beibehalten; sie wäre ggf. künftig durch Vergegenständlichungseffizienz ("efficiency of incarnation") zu ersetzen.

zusätzlichen Zugriffen erreichbar. Für $e_i = 1$ ist es aber nicht notwendig, daß ein Resultatabschnitt nur von den aktuellen Argument- Abschnitten abhängt: es können auch Abhängigkeiten von zuvor verarbeiteten Abschnitten bestehen; sofern diese über die Zustands- Information vermittelt werden können. Bild 12 zeigt, wie dafür das Schema von Bild 4 bzw. 5 zu erweitern ist
Die praktische Konsequenz von Satz 2 besteht darin, daß $e_i = 1$ nur von bestimmten Verarbeitungsbreiten an realisierbar ist, d. h. nur im Rahmen technischer Auslegungen, die gewährleisten, daß die jeweils erforderlichen Argument- und Zustandsbits parallel in den Speichermitteln selektiert und den Verknüpfungsnetzwerken zugeführt werden können.
Grundsätzlich ist die Implementierungseffizienz kleiner als 1, wenn:

1. die abschnittsweise Zuordnung technisch nicht zu verwirklichen ist (Aufwand, Kompliziertheit), so daß bestimmte Folgen von Maschinenzuständen notwendig sind, um einen Resultatabschnitt zu erzeugen

2. Resultatbits von Argumentbits abhängen, die sich in verschiedenen Abschnitten befinden können, so daß Zugriffe zu mehreren Argumentabschnitten nötig sind, um diese Resultatbits zu bestimmen

3. bestimmte Argumentbits veranlassen, daß Teile bereits erzeugter Resultatabschnitte geändert werden müssen, so daß erneute Zugriffe zu diesen Abschnitten auszuführen sind.

Ein Sachverhalt nach 1. ist nicht immer ein unüberwindliches Hindernis: es ist letztlich eine Ermessensfrage, was man als "zu aufwendig" oder "zu kompliziert" ansieht.
Hingegen bezeichnen die Sachverhalte 2. oder 3. objektive Grenzen: solche Algorithmen können nie mit $e_i = 1$ vergegenständlicht werden. Ein plausibles Beispiel dafür ist das Umordnen eines Vektors gemäß den Angaben einer Indexliste: jede Argumentposition kann grundsätzlich in jede Resultatposition transportiert werden, so daß es notwendig sein kann, bereits abgespeicherte Resultatabschnitte nochmals aufzurufen, um neue Werte einzufügen.

Alle diese Überlegungen gehen von der grundsätzlichen Wünschbarkeit der direkten Zuordnung aus. Durch abschnittsweise Zuordnung soll dieser Ansatz technisch verwirklicht werden, und es ist klar, daß damit Schaltmittel in bestmöglicher Weise ausgenutzt werden können, nämlich durch lückenlose Folgen von Nutzoperationen. Läßt sich $e_i = 1$ von einer gewissen Mindest-Verarbeitungsbreite an verwirklichen, so heißt das, daß jede so ausgelegte Schaltung in jedem ihrer internen Zyklen einen Anteil zum Endresultat beiträgt, der ihrer Verarbeitungsbreite entspricht; man erhält also das Maximum an Verarbeitungsleistung, das mit den vorgesehenen Aufwendungen überhaupt zu verwirklichen möglich ist.

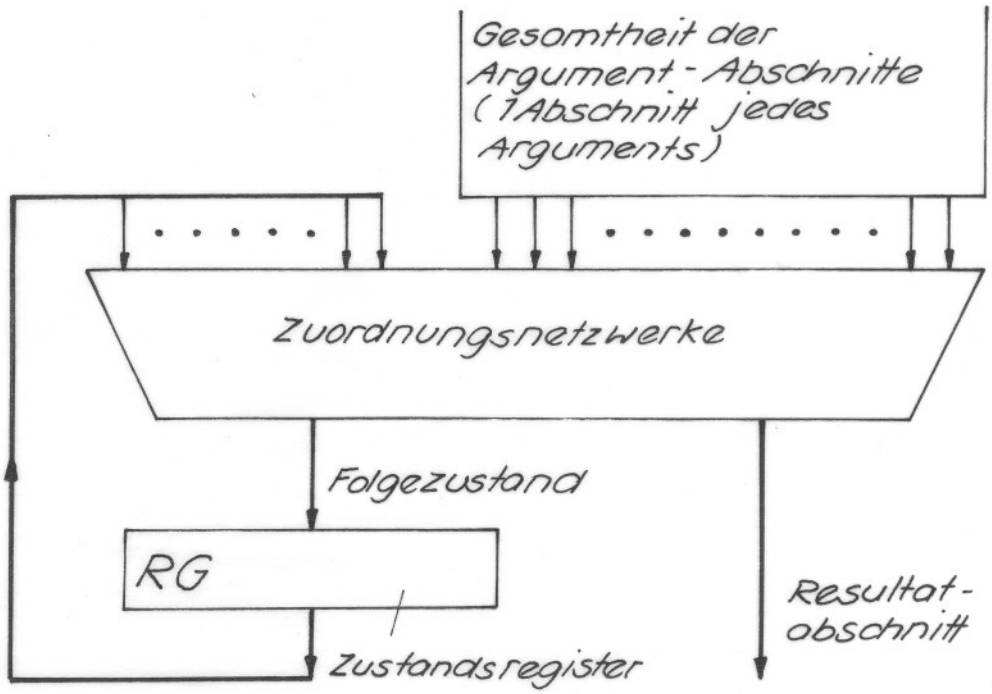
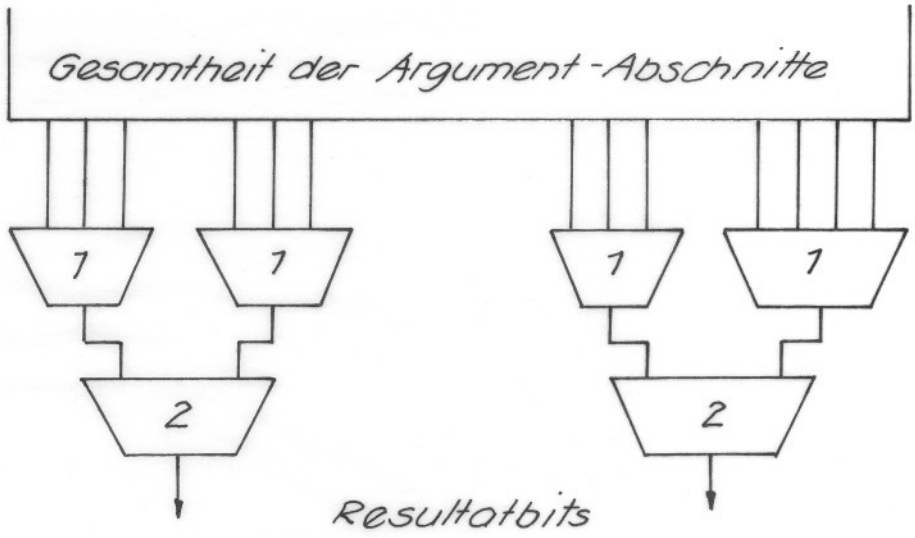


Bild 12 Prinzip der abschnittsweisen Zuordnung unter Einbeziehung von Zustands-Information



1: Netzwerke zur Verknüpfung unabhängiger Gruppen von Argumentbits
2: Netzwerke zur kombinatorischen Zusammenfassung

Bild 13 Illustration der gruppenweisen Zerlegbarkeit

Die Frage, ob für einen gegebenen Algorithmus eine technisch-ökonomisch sinnvolle Vergegenständlichung gelingt, ist letzten Endes nur durch zielgerichtete erfinderische Bemühungen entscheidbar, also durch Versuche, entsprechende Schaltungsanordnungen auszuarbeiten. Im besonderen sollte man sich nicht darauf beschränken, einen prozedural beschriebenen Algorithmus lediglich schaltungstechnisch umzusetzen, sondern man sollte gleichsam vorurteilsfrei nach Lösungen suchen, die die gewünschten Wirkungen hervorbringen.[1] Ein solcher Weg kann recht langwierig sein; deshalb seien einige Kriterien angeführt, die oft ein überschlagsmäßiges Urteil ermöglichen:

1. Beherrschbare Leitungszahlen liegen in der Größenordnung von 32 bis etwa 512. Dem Schema von Bild 4 entsprechen direkt arithmetisch- logische Einheiten, die aus 2 Argumenten zu 32 oder 64 bit ein Resultat gleicher Länge erzeugen und zusätzlich einige Bedingungsleitungen erregen (z. B. "Übertrag", "Resultat = Ø"). Die Grenzen sind im wesentlichen durch die beschränkten Kontaktzahlen an Schaltkreisen und durch Störeinflüsse gegeben (kapazitive und induktive Kopplung; impulsförmige Beeinflussung der Speisespannung, wenn viele Ausgangstreiber gleichzeitig schalten).[2]

2. Ob sich ein Zuordner technisch verwirklichen läßt, hängt von der Anzahl der Verknüpfungen, den technologischen Voraussetzungen und der Kompliziertheit der Booleschen Gleichungen ab, die die Verknüpfungen beschreiben.

3. Beliebig komplizierte Verknüpfungen können mit ROM- oder RAM- Zuordnern vergegenständlicht werden; die Variablenzahlen sind allerdings beschränkt:

- 8 bit/Verknüpfung: unproblematisch
- 12 bit/Verknüpfung: noch beherrschbar
- 16 bit/Verknüpfung: noch möglich.

Die absolute Obergrenze (wenn Kosten keine Rolle mehr spielen) liegt vielleicht bei 2Ø...24 bit/Verknüpfung.[3]

4. Die Chancen der Realisierbarkeit steigen an, wenn es gelingt, die notwendigen Verknüpfungen in Gruppen zu zerlegen, die jeweils für sich realisierbar sind und die entweder voneinander unabhängig sind oder mit einfachen Mitteln verknüpft werden können.[4] Das ist in Bild 13 veranschaulicht.

---

1 Das wurde in /241/ bzw. /243/ gezeigt. Man vergleiche die prozedurale Beschreibung des Algorithmus (/281/, /297/), deren technische Umsetzung in /123/ und die Lösung nach /241/, die durch abschnittsweise Parallelarbeit mit $e_i = 1$ deutlich (wenigstens 8-16 mal) schneller ist.

2 512-bit- Datenpfade sind schon ausgeführt worden (/151/).

3 1...16 Mbit je Resultat- Bitposition sind technisch durchaus noch beherrschbar (z. B. mit 1 bzw. 4Mbit DRAM).

4 Für Weiteres s. /243 /. Beispiel: die Schaltung nach /123/, die aus einem beliebigen Binärvektor einen Vektor erzeugt, der nur die erste Eins enthält (Indexvektor).

Die Verhältnisse lassen sich genauer untersuchen, wenn man für den betreffenden Algorithmus eine Abhängigkeitsmatrix |D| aufstellt. Das ist eine binäre Matrix, deren Zeilen den Bitpositionen aller Argumente entsprechen und deren Spalten den Bitpositionen aller Resultate. Hängt ein Resultatbit i von einem Argumentbit j ab (andersherum: übt das Argumentbit j Einfluß auf das Resultabit i aus), so steht in der Position ij der Matrix |D| eine Eins, sonst eine Null. (|D| repräsentiert nicht eine aktuelle Abhängigkeit, sondern alle grundsäzlich möglichen Abhängigkeiten bei beliebigen Werten.) Beispielsweise sieht die Matrix einer bitweise unabhängigen Verknüpfung von zwei 3-bit- Argumenten zu einem 3-bit- Resultat so aus:

$$|D| = \begin{array}{c|ccc} & r_1 & r_2 & r_3 \\ a_{11} & 1 & & \\ a_{12} & & 1 & \\ a_{13} & & & 1 \\ a_{21} & 1 & & \\ a_{22} & & 1 & \\ a_{23} & & & 1 \end{array}$$

Die Spaltensummen von |D| geben für jedes Resultatbit an, von wievielen Argumentbits es abhängt. Damit läßt sich abschätzen, ob direkte Zuordnungen realisierbar sind. Die Zeilensummen von |D| geben für jedes Argumentbit an, auf wieviele Resultatbits es Einfluß hat. Damit läßt sich abschätzen, welche Aufwendungen für die Informationswege bzw. für das Selektieren der Argumentbits zu veranschlagen sind. Bildet man die elementweise Konjunktion zwischen 2 Spalten und erhält dabei keine einzige Eins, so können die beiden Resultatbits unabhängig voneinander gebildet werden: folglich lohnt es sich, die Schaltung so auszulegen, daß die benötigten Argumentbits parallel für die Verknüpfungen zugänglich sind bzw. abschnittsweise parallel selektiert werden können.

4.3. Bewertung der Aufwendungen

Aufwendungen, wie Datenwege, Speicher, Verknüpfungsschaltungen, sind nach ihrer Nützlichkeit zu bewerten. Ein Schaltungsentwurf wird auch danach beurteilt, wie gut die Schaltmittel im praktischen Betrieb ausgelastet werden. Eine bestimmte Aufwendung ist um so nützlicher, je mehr sie zum Leistungsvermögen der gesamten Anordnung beiträgt. Um diese Nützlichkeit beurteilen zu können, wird der Begriff der Mehraufwandseffizienz eingeführt. Dieser geht auf /243/ zurück, wo gezeigt wurde, daß damit plausible Überschlagsbetrachtungen möglich sind, um in der Konzeptionsphase schnell Entscheidungen zu treffen.[1] Zunächst wird für die einzelne Schaltungsanordnung eine Hardware- Effizienz HE wie folgt definiert:

$$HE = \frac{\text{Leistungsangabe}}{\text{Aufwandsangabe}}$$

---

1 Z. B. auf Grundlage von Probeentwürfen bis zur Blockschaltbild- oder Register- Transfer- Ebene.

Es wird angenommen, für eine bestimmte Aufgabenstellung seien verschiedene Schaltungsanordnungen entworfen worden. Diese werden nach steigenden Aufwendungen geordnet, und es werden sinnfällige Übergänge von einfacheren zu aufwendigeren Schaltungen betrachtet. Für jeden Übergang von einer Anordnung 1 zu einer Anordnung 2 wird die Mehraufwandseffizienz ME_{12} folgendermaßen ermittelt:

$$ME_{12} = \frac{HE_2}{HE_1}$$

Es ist durchaus sinnvoll, sich auch die Absolutwerte vor Augen zu führen; dazu wird das Aufwandsverhältnis R_a und das Leistungsverhältnis R_p bestimmt:

$$R_a = \frac{\text{Aufwand 2}}{\text{Aufwand 1}} \qquad R_p = \frac{\text{Leistung 2}}{\text{Leistung 1}}$$

Damit ergibt sich: $ME_{12} = \frac{R_p}{R_a}$

Um diese einfachen Formeln anwenden zu können, ist es erforderlich, eine Vielzahl von Probeentwürfen zu erarbeiten. Die Ergebnisse sind aber keineswegs trivial. So läßt sich das Verfahren gut nutzen, wenn man von der Vorstellung einer Reihe verschiedener, aber untereinander kompatibler Hardware- Modelle für den gleichen Verwendungszweck ausgeht. Es dürfte sich dann oft zeigen, daß es besonders sinnfällige Modelle gibt , aber auch solche, deren Verwirklichung sich nicht lohnt.[1]

Im folgenden wird nur eine einzige Schaltungsanordnung betrachtet, um zu untersuchen, wie gut deren Schaltmittel ausgenutzt sind. Dazu wird die binäre Verbindungsmatrix Γ^b des Strukturgraphen $\sum$ verwendet. In der Anordnung seien n Algorithmen $\mathcal{A}_1 \ldots \mathcal{A}_n$ vergegenständlicht. Für jeden Algorithmus $\mathcal{A}_\nu$, $\nu = 1 \ldots n$, wird eine Bedeckungsmatrix $|C_\nu|$ aufgestellt. Sie hat dieselbe Struktur wie Γ^b . In ihr ist jede Verbindung (Kante des Strukturgraphen) mit einer 1 bezeichnet, die vom betreffenden Algorithmus benutzt wird; es werden also alle Verbindungen markiert, die für die Ausführung des Algorithmus notwendig sind; dadurch sind auch die jeweils miteinander verbundenen Knoten als notwendig gekennzeichnet. Praktisch kann dieses Markieren so vonstatten gehen, daß die Bedeutung der einzelnen Schaltmittel und Verbindungen, d. h. deren Aufgabe bei der Vergegenständlichung der Algorithmen, in der Entwurfsphase formal beschrieben und rechentechnisch erfaßt wird. Es ist also nicht nur anzugeben, wie die Schaltmittel untereinander verbunden sind, sondern auch, wozu sie vorgesehen sind.[2]

---

1 So wurde in /243/ genau ein "Einstiegs"- Modell identifiziert, das bei etwa doppeltem Aufwand gegenüber dem vorgeordneten Modell eine 7-fache Leistung aufweist.

2 Wurde der Entwurf in herkömmlicher Weise erfaßt, sind diese Angaben kaum mehr zu ermitteln. Bemerkung: Ein ausgebauter Kalkül über Bedeckungsmatrizen könnte eine Grundlage dafür bilden, Testbelegungen automatisch zu erzeugen.

In der gesamten Anordnung sollte jede Komponente irgendeinen Zweck erfüllen, d. h. die elementweise disjunktive Verknüpfung aller Bedeckungsmatrizen sollte der binären Verbindungsmatrix Γ^b gleich sein:

$$\bigvee_{\nu=1}^{n} |C_\nu| = \Gamma^b$$

Ist das nicht der Fall, enthält die Anordnung überflüssige Verbindungen[1]; diese sind durch elementweise Antivalenzverknüpfung sofort ersichtlich.

Für einen bestimmten Algorithmus $\mathfrak{A}_\nu$ ist die Schaltung dann gut ausgenutzt, wenn $|C_\nu|$ möglichst viele Einsen enthält. Das wird durch den Ausnutzungsgrad Ψ_ν für den jeweiligen Algorithmus $\mathfrak{A}_\nu$ gekennzeichnet:

$$\Psi_\nu = \frac{\sum_{ij} |c_{ij}^\nu|}{\sum_{ij} \Gamma_{ij}^b} \quad ; \quad 0 < \Psi_\nu \leq 1$$

Diese Bewertungsweise allein wird ausgesprochenen Hochleistungsschaltungen nicht gerecht: jede kombinierte Auslegung bzw. Mehrfachnutzung von Schaltmitteln erhöht zwar den Ausnutzungsgrad, ist aber grundsätzlich mit Leistungsbeschränkungen verbunden. Für höchste Leistungsanforderungen sind erfahrungsgemäß die Vergegenständlichungen der einzelnen Algorithmen getrennt voneinander auszuführen[2] und in sich so auszubilden, daß das jeweils höchste Leistungsvermögen erreicht wird. Deshalb wird aus der Häufigkeit der Nutzung der einzelnen Algorithmen eine Ausnutzungsmatrix |UE| über den Erwartungswert bestimmt:

$$|UE| = \sum_{\nu=1}^{n} p_\nu |C_\nu| \quad ; \quad \sum_{\nu=1}^{n} p_\nu = 1$$

Aus |UE| lassen sich wichtige Schlüsse für die weitere Verfeinerung der Schaltungslösung ziehen. Dazu wird der Mittelwert $\overline{UE}$ gebildet, und es wird eine Abweichungsmatrix $|\triangle|$ aufgestellt:

$$\overline{UE} = \frac{\sum_{ij} |UE_{ij}|}{i \cdot j} \quad ; \quad |\triangle| = |UE| - \overline{UE} \quad \text{(elementweise Subtraktion).}$$

Komponenten, die sich durch eine große negative Abweichung hervorheben (bzw. absolut gesehen: die in |UE| Ø nahekommen), sind in ihrer Nützlichkeit grundsätzlich fragwürdig: es lohnt sich zu überprüfen, ob die so gekennzeichneten Schaltungsteile weggelassen werden können, wobei deren Aufgaben durch andere, an sich gegebene Einrichtungen übernommen werden.[3]

---

1 Fehlende Verbindungen sind ebenso erkennbar; dieser (praktisch bedeutsame) Gesichtspunkt wird hier vernachlässigt.

2 Meist werden gewisse Algorithmen zusammengefaßt und jeweils gemeinsame Schaltmittel vorgesehen (z. B. für Gleitkommaoperationen, Adressenrechnung usw.).

3 Z. B. durch mikroprogrammtechnische Emulation.

Komponenten mit außergewöhnlich großer positiver Abweichung (bzw. absolut: in |UE| nahe bei 1) haben eine besondere Bedeutung für das Leistungsvermögen des Systems. Es lohnt sich deshalb zu untersuchen, ob durch Verfeinerungen oder gezielten Einsatz zusätzlicher Mittel die Gesamtleistung weiter verbessert werden kann.

4.4. Bewertung des Wirkungsgrades

Das Ziel ist die universelle Maschine, die vergegenständlichte Algorithmen als Ressourcen bereitstellt, um damit eine Vielfalt von - üblicherweise recht komplexen - Anwendungsalgorithmen implementieren zu können. Mit den bisher vorgeschlagenen Bewertungsgrundlagen läßt sich das absolute Leistungsvermögen angeben, es läßt sich beurteilen, welche Algorithmen sich zur Vergegenständlichung eignen, und es läßt sich bewerten, wie gut die Schaltmittel ausgenutzt sind. Hingegen ist der Anwender ausschließlich daran interessiert, wie schnell seine Aufgaben praktisch bearbeitet werden. Solche Angaben sind durch Schätzungen, Simulation oder Probeläufe zu ermitteln. Damit sind beispielsweise Preis- Leistungs- Vergleiche mit anderen Maschinen möglich.
Wenn man universelle Maschinen mit höchstem Leistungsvermögen schaffen will, braucht man aber den Vergleich mit absoluten Leistungsgrenzen. Anhand solcher Vergleichswerte ist dann die Sinnfälligkeit des Lösungsvorschlages beurteilbar. Im besonderen wird sich herausstellen, ob die vergegenständlichten Algorithmen tatsächlich zweckmäßig ausgewählt wurden. Dazu wird vorgeschlagen:

Es wird für jeden wesentlichen Algorithmenkomplex eine fiktive Sondermaschine ausgearbeitet (so detailliert, daß deren Leistungsvermögen beurteilbar ist). Unterstellt man für die Sondermaschine eine technisch gerade noch beherrschbare Auslegung[1], so wird deren Leistungsvermögen den absoluten Leistungsgrenzen sehr nahe kommen.
Dann läßt sich ein Wirkungsgrad η einführen, der das Verhältnis der Leistungen der zu beurteilenden Universalmaschine und der fiktiven Sondermaschine repräsentiert[2]:

$$\eta = \frac{P_{UNIV}}{P_{SPEZ}}$$

Damit läßt sich ein Ziel für das Entwerfen neuartiger Universalmaschinen angeben: ihr Leistungsvermögen sollte für die anwendungspraktisch wichtigsten Algorithmenkomplexe dem von einschlägigen Sondermaschinen soweit wie möglich entsprechen.

---

1 Es bietet sich an, dafür die Technologien und Aufwendungen der jeweils leistungsfähigsten kommerziell verfügbaren Supercomputer anzusetzen (in /243/ bereits ausgeführt).

2 Je nach Zweckmäßigkeit kann der Wirkungsgrad auf eine bestimmte Technologie oder Größenordnung des Aufwandes bezogen werden; es ist dann für die fiktive Sondermaschine dieselbe Technologie oder ein ähnlicher Kostenrahmen wie bei der zu vergleichenden Universalmaschine anzunehmen.