Heidelberger Taschenbücher

Sammlung Informatik

Giloi

Rechnerarchitektur

Springer-Verlag
Berlin Heidelberg New York

W. K. Giloi

Rechnerarchitektur

Mit 133 Abbildungen

Springer-Verlag
Berlin · Heidelberg · New York 1981

Wolfgang K. Giloi

Professor of Computer Science, University of Minnesota
und
Professor für Technische Informatik,
Technische Universität Berlin

CIP-Kurztitelaufnahme der Deutschen Bibliothek.
Giloi, Wolfgang:
Rechnerarchitektur/W. K. Giloi. —
Berlin, Heidelberg, New York: Springer, 1981. —
(Heidelberger Taschenbücher; Bd. 208: Sammlung Informatik)

ISBN 3-540-10352-X Springer-Verlag Berlin Heidelberg New York
ISBN 0-387-10352-X Springer-Verlag New York Heidelberg Berlin

Bindearbeiten: Konrad Triltsch, Würzburg
2145/3020—543210

Heidelberger Taschenbücher Band 208

Sammlung Informatik

Herausgegeben von F. L. Bauer, G. Goos und M. Paul

Dieses Buch
ist dem großen Rechnerpionier
Konrad Zuse gewidmet

Vorwort

> The types of architectures are established not by architects but by society, according to the needs of the different institutions. Society sets the goals and assigns to the architect the job of finding the means of achieving them.
>
> Encyclopaedia Britannica, Macropaedia 1, 1089

In der bisherigen Entwicklung elektronischer Rechenanlagen konnte etwa alle 5—6 Jahre eine Leistungssteigerung um eine Größenordnung festgestellt werden. Dieser Fortschritt war bisher jedoch fast ausschließlich ein Fortschritt der Hardware-Technologie. Die Entwicklung der Halbleiter-Technologie in Richtung auf hohe Integrationsdichte, hohe Arbeitsgeschwindigkeit und geringe Kosten verlief in den letzten Jahren besonders stürmisch. An die Stelle der Hardware als dominierendem Kostenfaktor einer Rechenanlage ist die Software getreten.

Die geringen Kosten hochintegrierter, standardisierter Bausteine erlauben heute auch bei kleinen bis mittleren Rechnern einen Hardware-Aufwand, den man vor wenigen Jahren nur bei einem Großrechner als angemessen betrachtet hätte. Selbst Mikroprozessoren haben inzwischen eine Wortlänge von 16 oder 32 bit und können mit einem Arbeitsspeicher von mehreren Megabyte ausgestattet werden; die Größe des Arbeitsspeichers ist damit kein Kriterium zur Unterscheidung von „kleinen" und „großen" Rechnern. Der große, kostengünstige Speicher ist aber nur ein erster Schritt zur Nutzung der durch die Großintegration eröffneten Möglichkeiten, der als nächster Schritt eine weitere Verbesserung des Preis/Leistungs-Verhältnisses folgen muß. Da die Technologie großintegrierter Rechnerkomponenten verhältnismäßig „langsam" ist, kann eine Leistungssteigerung nicht mehr, wie in der Vergangenheit, durch einfache Erhöhung der Arbeitsgeschwindigkeit von Rechenwerk und Speicher erzielt werden, sondern nur durch entsprechende Vervielfachung dieser zwar billigen, aber auch verhältnismäßig langsamen Hardware-Betriebsmittel eines Rechners. Damit kommt man erstmalig auch der Verwirklichung von Zielvorstellungen nahe, die man seit etwa zwei Jahrzehnten bereits als äußerst wünschenswert erkannt hat, deren Realisierung bisher aber aus technischen oder wirtschaftlichen Gründen nicht oder nur in unbefriedigendem Maße möglich war: der modularen Erweiterbarkeit und der Ausfalltoleranz einer Anlage.

Diese Entwicklung macht eine Abkehr von der jahrzehntelang dominierenden Architektur der von Neumann-Maschine und eine Suche nach neuen, angemesseneren Architekturformen unumgänglich. Aber auch um die neueren Erkenntnisse der Softwaretechnik hinsichtlich Modularität,

Zuverlässigkeit und Wartbarkeit großer Softwaresysteme, hinsichtlich Datenintegrität, Anpassungsfähigkeit der Betriebssysteme, Realisierung der Konzepte abstrakter Datentypen und Datenmodelle, Realisierung transaktionsorientierter Sprachen und anderem mehr zu nutzen und eine solche Software besser hardwaremäßig zu unterstützen (Schließung der „semantischen Lücke" zwischen Software und Hardware), sind neue Wege in der Rechnerarchitektur zu beschreiten.

Dieses Buch handelt von der Rechner*architektur*. Betrachtungen über Rechner*organisation* — das ist der Aufbau und die Arbeitsweise der in der Architektur verwandten „Hardware-Betriebsmittel" — werden in dem Maße angestellt, wie diese organisatorischen Maßnahmen bestimmte Prinzipien der Rechnerarchitektur unterstützen (z. B. Speichersegmentierung, Fließbandverarbeitung, u. ä.) und zum Verständnis gewisser Eigenheiten der Strukturen von Rechnerarchitekturen und deren Leistungsfähigkeit betrachtet werden müssen. Dadurch werden gewisse Exkursionen von der Architekturebene in die Ebene der Organisation der Komponenten einer Architektur unvermeidbar sein. Als ergänzende Schrift, die vorwiegend die Organisationsebene behandelt (mit gewissen Exkursionen in die Architekturebene) sei das vorzügliche Buch von Jessen [JES 75] empfohlen.

Aber auch dort, wo von uns organisatorische Fragen behandelt werden, wird der Leser keine Diagramme von Schaltnetzen und Schaltwerken vorfinden. Für den Rechnerarchitekten umfaßt der Begriff der Hardware komplexe Komponenten mit einem bestimmten funktionellen Verhalten, die er als Betriebsmittel beim architektonischen Entwurf eines Rechners einplanen kann, und die in steigendem Maße als fertige, großintegrierte Bausteine erhältlich sein werden. Schaltungen gehören damit in das Arbeitsgebiet des Entwerfers solcher Bausteine.

In der Vergangenheit wurde die Suche nach innovativen Rechnerarchitekturen zu sehr oft mit reinen Strukturüberlegungen gleichgesetzt, das heißt mit der Suche nach Verbindungsstrukturen von Prozessoren, Speichern und anderen Betriebsmitteln, die besondere Vorteile erhoffen ließen. Erst in den letzten Jahren hat man erkannt, daß solche reinen Strukturüberlegungen nicht weit führen; daß es vielmehr darauf ankommt, sich bereits vor Eintritt in Strukturüberlegungen Klarheit über das verschafft zu haben, was wir unter dem Begriff „Operationsprinzip" subsummieren, nämlich die Informationsstrukturen und Kontrollstrukturen der gesuchten Rechenarchitektur.

Wir sehen damit den Rechnerentwurf der Zukunft vorwiegend als einen *top-down*-Entwurfsprozeß: Ausgehend von den Zielen, die mit einer Rechnerarchitektur erreicht werden sollen, legt man zunächst ein Operationsprinzip fest, welches am besten die gestellten Ziele zu verwirklichen verspricht, und das gewählte Operationsprinzip impliziert dann in gewissem Grade die Systemstruktur. In jedem dieser Entwurfsschritte wird aber ein gewisser Freiraum für Entwurfsentscheidungen bestehen bleiben.

Zu der am Anfang stehenden Festlegung der Entwurfsziele *(requirement engineering)* gehören auch bereits detaillierte Überlegungen darüber, wie und mit welchen Sprachen das System später programmiert werden soll (gerade gegen dieses Postulat ist in der Vergangenheit bei den Bemühungen um innovative Rechnerarchitekturen oft — und meist mit sehr nachteiligen Folgen — verstoßen worden). Dabei werden innovative Operationsprinzipien zwangsläufig zu neuartigen Sprachkonstruktionen führen, wie dies bei den Feldrechnern oder den Datenfluß-Architekturen geschehen ist und für die verteilten Polyprozessor-Systeme zu erwarten steht. So wie in der Vergangenheit die von Neumann-Architektur die Entwicklung der „von Neumann-Sprachen" nach sich zog, so wird vielfach auch weiterhin die Sprachentwicklung der Architekturentwicklung nachfolgen. Innovative Architekturentwürfe, die in Zukunft in stärkerem Maße neben anderen auch das Entwurfsziel der Verringerung der semantischen Lücke zwischen Hardware und Anwendungssoftware verfolgen werden, werden aber auch mehr als bisher eine umgekehrte Entwicklung zeitigen (die bisher nur bei den „Keller-Maschinen" zu verzeichnen war), nämlich, daß die Architektur der Sprache folgt. Dies soll erklären, warum die Betrachtung von Datentypen und Datenstrukturen bis hin zu den Strukturen höherer Programmiersprachen einen so breiten Raum in einem Buch über Rechnerarchitektur einnimmt, wie dies bei dem vorliegenden Text der Fall ist.

Wie bei jedem neuen Gebiet, so ist auch die bisherige Entwicklung in der Rechnerarchitektur zunächst durch das Vorherrschen von *ad hoc*-Lösungen gekennzeichnet. Systematisierungen folgen in der Regel dem Prozeß der Entwicklung von *ad hoc*-Lösungen erst dann, wenn ein gewisser Erfahrungsschatz erarbeitet und ein gewisser Reifegrad erreicht ist. Der vorliegende Text ist der Versuch einer solchen Systematisierung.

Der Text ist entstanden als das Ergebnis mehrjähriger Vorlesungs- und Forschungstätigkeit des Autors an der University of Minnesota und der Technischen Universität Berlin. Er hat zunächst den Charakter eines Lehrbuchs; darüber hinaus ist er aber auch als allgemeine Monographie angelegt. Insbesondere glauben wir, daß dieses Buch auch dem in der Praxis stehenden Ingenieur, Informatiker oder Mathematiker, der mit der Entwicklung, der Planung oder dem Einsatz von Rechenanlagen befaßt ist, helfen kann, die Leistungsfähigkeit und Grenzen von innovativen Rechnersystemen beurteilen und Entwicklungstendenzen erkennen zu lernen.

Der Verfasser ist Professor Dr. G. Goos und Dr. G. Hommel für wertvolle Kritik und Hinweise zum Dank verpflichtet. Seinen Mitarbeitern, P. Behr, Dr. H. K. Berg und R. Güth verdankt er anregende Diskussionen, und sein besonderer Dank für viele ausführliche Gespräche über zukünftige Rechnerstrukturen richtet sich an C. A. (Sandy) Wilson.

Berlin, im Frühjahr 1980 W. K. Giloi

W. Giloi, H. Liebig

Logischer Entwurf digitaler Systeme

Hochschultext
2., überarbeitete Auflage. 1980. 183 Abbildungen.
XII, 328 Seiten
DM 48,–
ISBN 3-540-10091-1

Inhaltsübersicht: Aussagenkalkül und Boolesche Algebra.– Boolesche Algorithmen. – Schaltnetze. – Schaltketten. – Asynchron-Schaltwerke. – Synchron-Schaltwerke und Mikroprozessoren. – Literaturverzeichnis.– Sachverzeichnis.

Aus den Besprechungen: „Das ausgezeichnete Buch von W. Giloi und H. Liebig wird voraussichtlich längere Zeit als Standardwerk für die Informatikausbildung auf dem Gebiet logischer Entwurf digitaler Systeme zu empfehlen sein. Es ist beachtenswert mit welcher Konsequenz die Kapitel einheitlich und folgerichtig aufgebaut sind. An der Darstellung ist zu erkennen, daß das Buch von erfahrenen Hochschullehrern geschrieben wurde. Das Bemühen um Anschaulichkeit und Verständlichkeit sowie die zahlreichen Beispiele, die der Praxis entnommen sind, unterstreichen das."
Zeitschrift für angewandte Mathematik und Mechanik

H. Liebig

Logischer Entwurf digitaler Systeme

Beispiele und Übungen
Hochschultext
1975. 92 Abbildungen. VIII, 175 Seiten
DM 28,–
ISBN 3-540-06912-7

Das Buch umfaßt 72 Übungsaufgaben und vollständig durchgerechnete Lösungen mit dem Charakter von Entwurfsbeispielen für die systematische Entwicklung von Schaltnetzen und Schaltwerken. Die in dem zugehörigen Hauptwerk "Logischer Entwurf digitaler Systeme" von GILIO/LIEBIG behandelten mathematischen Methoden werden voll angewendet und zum Teil ergänzt. Der aktuellen technischen Praxis wird durch die Auswahl der Aufgaben und der ergänzenden Themen Rechnung getragen.
Studenten wie Ingenieuren wird die gegenüberstellende Darstellung von Formeln und Schaltbildern willkommen sein. Das Buch ist für beide Gruppen gleichermaßen geeignet, für Studenten zum Üben und Entwerfen und für Ingenieure zum Nachschlagen und Entwickeln.

Springer-Verlag
Berlin
Heidelberg
New York

H. Liebig

Rechner-organisation

Hardware und Software digitaler Rechner

Hochschultext
Unter Mitarbeit von T. Flik, K. Horn
1976. 102 Abbildungen. X, 282 Seiten
DM 58,–
ISBN 3-540-07596-8

Inhaltsübersicht: Klassische Rechnerorganisation. – Rechnerhardware. – Rechnersoftware. – Assemblerprogrammierung und Prozessororganisation. – Speicherorganisation. – Ein- und Ausgabeorganisation.

Aus den Besprechungen:

Nach einer Einführung in die klassische Rechnerorganisation, die Hard- und Software von Rechenanlagen werden die Organisationen der Hauptelemente digitaler Rechner – Prozessor, Speicher und Ein-/Ausgabegeräte in verschiedenen Variationen – besprochen... Die prinzipiell möglichen Realisierungen der Einzelorganisationen werden sehr ausführlich behandelt und verglichen, eine große Anzahl detaillierter Beispiele erleichtert das Verständnis...
Gerade durch die große Anzahl praktischer Beispiele und deren Ausführung in einer durch Hardwareentscheidungen ausgezeichnet motivierten Assemblersprache eignet sich das Buch vorzüglich als Nachschlagewerk für Praktiker, aber auch als Einführung in die Probleme der Rechnerorganisation im Rahmen von Hoch- und Fachhochschulen."

VDI-Zeitschrift

Springer-Verlag
Berlin
Heidelberg
New York

Inhaltsverzeichnis

1 Einleitung

1.1 Was ist Rechnerarchitektur?

Rechnerarchitektur ist eine Übersetzung des englischen Begriffs *computer architecture*. *Architecture* bedeutet im Englischen *Baukunst* oder *Baustil*; aber auch das einzelne Produkt der Baukunst wird als *architecture* bezeichnet. Die letztgenannte Bedeutung hat das deutsche Wort Architektur nicht. Wenn wir von Rechnerarchitektur sprechen, meinen wir jedoch den vollen Bedeutungsinhalt von *computer architecture*.

Rechnerarchitektur wie Architektur im allgemeinen dient einem Zweck. Wir zitieren hier die Encyclopaedia Britannica („Art of Architecture", Macropaedia 1,1089): "The types of architecture are established not by architects but by society, according to the needs of the different institutions. Society sets the goals and assigns to the architect the job of finding the means of achieving them."

Entsprechend ihrem Zweck unterscheidet man in der Baukunst Typen von Architekturen wie die Architektur des Wohnungsbaus, des Fabrikbaus, des Sakralbaus, des Verwaltungsbaus, etc. Spezifischer noch kann man von Sportstätten-Architektur, Flughafen-Architektur, Krankenhaus-Architektur, usw. reden.

Die Grundaufgabe einer Rechenanlage ist die Sammlung, Speicherung, Verarbeitung und Darstellung von Information. Diese Grundaufgabe kann auf den verschiedensten Anwendungsgebieten zu erfüllen sein. Eine Rechenanlage, die auf jedem beliebigen Gebiet einsetzbar sein soll, soll Universalrechner genannt werden. Daneben gibt es Anlagen, die für eine spezielle Aufgabe konzipiert wurden und die deshalb *Spezialrechner* heißen. Dementsprechend ist zwischen Universalrechner-Architekturen und Spezialrechner-Architekturen zu unterscheiden.

Der Computer-Architekt hat seine Aufgabe aber noch nicht damit gelöst, daß er ein architektonisches Konzept vorlegt, welches den vorgegebenen Zweck erfüllt. Es wird vielmehr von ihm erwartet, daß dieses Konzept gewisse Forderungen möglichst optimal erfüllt. Solche Forderungen können unter verschiedenen Aspekten gestellt werden, nämlich

- dem Leistungs-Aspekt,
- dem Ausfalltoleranz-Aspekt,
- dem Erweiterbarkeits-Aspekt,
- dem pragmatischen Aspekt.

Leistungs-Aspekt. Bei manchen Spezialrechnern für Echtzeit-Aufgaben kann mitunter die Forderung nach maximaler absoluter Leistung in Vordergrund stehen, hinter den Kostenerwägungen zurücktreten müssen. Im allgemeinen wird man aber nicht die absolute Leistungs- sondern die Kosteneffektivität, d. h. die Leistung pro Kosteneinheit, zu maximieren trachten (wie immer diese meßbar sein mag). Es ist dabei zu beachten, daß zwei verschiedene Systeme nur dann in ihrer Leistung vergleichbar sind, wenn auch ihre Anwendungsbreite etwa gleich ist. Bei Allzweckrechnern kann es daher nicht darum gehen, die Leistung für eine bestimmte Aufgabe zu maximieren, sondern die mittlere Leistung, betrachtet über alle Anwendungsgebiete.

Ausfalltoleranz-Aspekt. Das Optimierungskriterium ist hier die Gewährleistung einer gewissen Mindest-Verfügbarkeit des Systems. Ein System heißt dann ausfalltolerant, wenn bei Ausfällen einzelner Komponenten des Systems noch ein betriebsfähiger Kern übrigbleibt, so daß eine bestimmte Mindestleistung zu allen Zeiten zur Verfügung steht.

Erweiterbarkeits-Aspekt. Bei diesem Aspekt handelt es sich darum, ein Rechnersystem so zu entwerfen, daß seine Erweiterbarkeit durch verschiedene Ausbaustufen ermöglicht wird.

Pragmatik-Aspekt. Unter diesem Stichwort fassen wir Architekturen zusammen, bei denen eine höhere Programmiersprache unmittelbar durch die Hardware des System interpretiert wird. Das bedeutet, daß die höhere Programmiersprache bereits die Maschinensprache des Systems ist, und daß zwischen der Darstellung eines Programms in der höheren Programmiersprache und der das Programm ausführenden Hardware keine weitere Zwischendarstellung existiert. Das Ziel ist hier, daß der Anwendungsprogrammierer, der Systemprogrammierer und möglichst auch der Wartungstechniker alle dieselbe Sprache sprechen [CHU 75]. Da dies die Beziehung zwischen einem Rechnersystem und den Personen, die mit seiner Programmierung und Wartung betraut sind, betrifft, sprechen wir von dem Pragmatik-Aspekt (wobei wir uns den Begriff *Pragmatik* aus der Semiotik ausleihen). Zum Pragmatik-Aspekt gehört auch die Forderung nach einer möglichst guten *Benutzbarkeit* und *Wartbarkeit* der Anlage.

Es muß betont werden, daß die genannten Aspekte sich keineswegs gegenseitig ausschließen. Gewisse architektonische Konzepte mögen dazu dienen, mehr als einem der genannten Aspekte Genüge zu tun. So mag man bestimmte Betriebsmittel zu einer Rechnerstruktur verbinden, die zunächst ausfalltolerant ist, daneben aber auch eine möglichst hohe Rechenleistung erbringt. Es muß ferner kaum besonders betont werden, daß der Leistungs-Aspekt und der Ausfalltoleranz-Aspekt mit Abstand die wichtigsten Gesichtspunkte beim Entwurf von Rechnerarchitekturen sind. Es ist für die Zukunft zu erwarten, daß als weiterer wichtiger Aspekt innovativer Rechnerarchitekturen der Aspekt der Systemsoftware-Vereinfachung ins Spiel kommt.

Dem Gebäude-Architekten stehen bestimmte Materialien wie Holz, Stein, Beton, Stahl, Glas zur Verfügung. Aus diesen Materialien bildet er zu-