

1. Einführung

1.1 Definitionen und Begriffe

Der **Mikroprozessor**¹ (μ P, *Micro Processing Unit* – MPU) ist die Zentraleinheit (*Central Processing Unit* – CPU) eines Digitalrechners, die auf einem einzigen Halbleiterplättchen (*Chip*) untergebracht ist. Er umfasst die beiden Komponenten Steuerwerk und Operationswerk (bzw. Rechenwerk), die untereinander Informationen über Steuer- und Meldesignale austauschen (s. Abb. 1.1). Während das Steuerwerk für den zeitgerechten Ablauf der Befehlsbearbeitung eines Programms zuständig ist, führt das Operationswerk (auch Rechenwerk genannt) die eigentlichen arithmetisch/logischen Operationen aus.

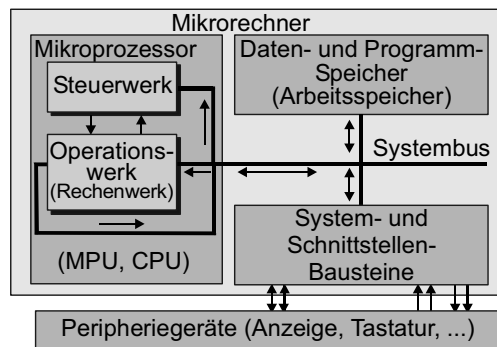


Abb. 1.1: Aufbau eines Mikrorechners

Ein **Mikrorechner** (μ R, Mikrocomputer) enthält neben dem Mikroprozessor als Zentraleinheit zusätzlich noch einen Daten- und Programmspeicher, die wir zusammenfassend als Arbeitsspeicher bezeichnen. Die Verbindung des Speichers mit dem Mikroprozessor geschieht über den (bidirektionalen) Systembus². Er leitet die Befehle aus dem Speicher zum Steuerwerk des Prozessors, die Operanden vom Speicher zum Operationswerk und die Ergebnisse zurück zum Speicher. Daten und Programme können gemischt in einem einzigen Speicher untergebracht oder aber in getrennten Daten- bzw. Programm-Speichern abgelegt sein. Zum Mikrorechner gehören weiterhin Schnittstellen-Bausteine für den Anschluss diverser Peripheriegeräte sowie weitere Systembausteine³ (Zähler, Echtzeituhr, Digital/Analog- bzw. Analog/Digital-Wandler usw., vgl. Kapitel 9).

Unter einem **Mikrorechner-System** (μ RS, Mikrocomputer-System – MCS) verstehen wir einen Mikrorechner mit allen angeschlossenen Peripheriegeräten, der sog. **Peripherie** des Systems. Dazu gehören z.B. Festplatten, Tastatur und Bildschirm, Drucker, aber auch einfache Sensoren, Aktoren, Anzeigen usw.

¹ Die Vorsilbe „Mikro-“ bezieht sich nur auf die Größe des Prozessors, nicht jedoch auf seine Leistungsfähigkeit oder die Art der Realisierung des Steuerwerks. Die Namensgebung geschah zu einer Zeit, als der Prozessor eines Digitalrechners noch ein schrankgroßes Gehäuse füllte

² Unter einem Bus versteht man eine Sammlung von mehreren Leitungen, auf denen gleichartige Informationen parallel und in dieselbe Richtung übertragen werden und an denen alle Komponenten derart angeschlossen sind, dass zu jedem Zeitpunkt höchstens eine senden kann, aber alle anderen empfangen können

³ Wir sprechen in diesem Buch vereinfachend auch dann von „Bausteinen“, wenn es sich dabei um Komponenten handelt, die auf demselben Chip untergebracht sind

Nach den Grundzügen ihrer Architektur unterscheiden wir bei den Prozessoren zwischen den folgenden Typen:

- **CISC-Prozessoren** (*Complex Instruction Set Computer*) verfügen über einen (relativ) umfangreichen Befehlssatz. Sie sind meist mikroprogrammiert, d.h., sie besitzen ein Mikroprogramm-Steuernetzwerk (MPStW), dessen Mikroprogramme vom Hersteller in einem Festwertspeicher (*Control Memory/Store*) untergebracht wurden und vom Benutzer nicht geändert werden können. Ihre Architektur folgt (meist) dem von-Neumann-Prinzip, d.h., Befehle und Daten liegen im selben Arbeitsspeicher gemischt vor und werden über ein einziges Bussystem¹ transportiert (vgl. Abb. 1.1). Diese Prozessoren werden heute hauptsächlich in Steuerungssystemen verwendet. In früheren Jahren dominierten die CISC-Prozessoren der Intel-80x86-Familie (bis 80486) den Markt der Personal Computer (PC).
- **RISC-Prozessoren** (*Reduced Instruction Set Computer*) sind nicht mikroprogrammierte Mikroprozessoren. Sie besitzen stattdessen ein fest verdrahtetes Steuerwerk, bei dem die Befehle durch direkte (kombinatorische) Schaltlogik realisiert werden. Die Bezeichnung RISC stammt von dem (ursprünglich) relativ kleinen Befehlssatz dieser Prozessoren. RISC-Prozessoren besitzen üblicherweise eine sog. Harvard-Architektur, bei der Befehle und Daten in getrennten Speichern untergebracht und gleichzeitig über getrennte Bussysteme transportiert werden können (s. Abb. 1.2). Sie werden hauptsächlich in den Mikrorechnern der höheren Leistungsklasse für anspruchsvolle Steuerungsaufgaben eingesetzt.

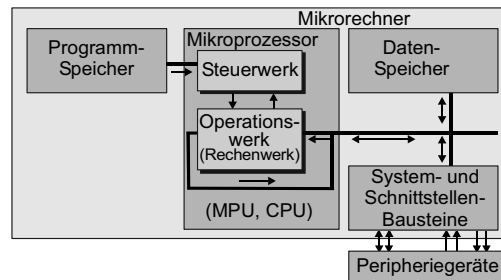


Abb. 1.2: Aufbau eines Mikrorechners mit Harvard-Architektur

- **Hybride CISC/RISC-Prozessoren**, stellen eine Mischform mit CISC- und RISC-Eigenschaften dar. Die Mehrzahl der heute im universellen Rechnerbereich eingesetzten Hochleistungs-Mikroprozessoren gehört dieser Klasse an. Insbesondere in den Personal Computern sind sie millionenfach vertreten (als Intel Pentium oder dazu „kompatible“ Prozessoren).

Nach der Art ihres Einsatzes unterscheiden wir die folgenden **µP-Klassen**:

- **Universelle Mikroprozessoren** (*General Purpose Processors*) sind Prozessoren ohne spezielle Schnittstellen oder Komponenten, wie sie z.B. in PCs oder Laptops eingesetzt werden. Mit geeigneten Programmen und externen Erweiterungsbausteinen sind sie für alle allgemeinen Anwendungen einsetzbar. Universelle Mikroprozessoren können sowohl vom Typ CISC oder RISC oder eine Mischform aus beiden sein.

¹ Da zu jedem Taktzyklus nur jeweils ein Befehl oder ein Datum transportiert werden kann, bildet dieses Bussystem den sog. von-Neumann-Flaschenhals (*Bottleneck*)

- **Anwendungsorientierte Mikroprozessoren** sind Mikroprozessoren für spezielle Anwendungen. Dazu gehören:
 - die **Mikrocontroller** (μC , *Microcontroller Unit* – MCU), die über besondere Schnittstellen, integrierte Speicher und festgelegte Programme zur Steuerung bestimmter Vorgänge verfügen. Sie stellen – nach unserer Klassifikation – vollständige Mikrocomputer auf einem einzigen Chip dar (*Computer on a Chip*, *Single-Chip Computer*). Im Unterschied zu den Prozessrechnern der Vergangenheit, die – oft schrankgroß – „von außen“ ein System steuerten, können Mikrocontroller wegen ihrer geringen Größe direkt in dieses System integriert werden (s. Abschnitt 1.2). Sie werden nach dem CISC- oder RISC-Prinzip gebaut.
 - die **Digitalen Signalprozessoren** (*Digital Signal Processor* – DSP), die besonders zur digitalen Verarbeitung analoger Signale mit speziellen Befehlen und Hardwarekomponenten konzipiert sind. Ihr Rechenwerk ist auf die schnelle Reihenberechnung (*Multiply-Accumulate* – MAC) ausgelegt, die in vielen Algorithmen der digitalen Signalverarbeitung eine zentrale Rolle spielt.
 - Immer häufiger verfügen Mikrocontroller aber auch über erweiterte Rechenwerke und Speicher-/Bus-Architekturen, die eine effiziente Verarbeitung analoger Signale ermöglichen. Andererseits besitzen DSPs häufig aber auch eine große Anzahl von integrierten Schnittstellen zur Kommunikation und Steuerung. Sie werden dann oft als **Digitale Signalcontroller** (DSC) bezeichnet.

1.2 Einsatzgebiete anwendungsorientierter Mikroprozessoren (Theo Ungerer)

Das wesentliche Einsatzgebiet anwendungsorientierter Mikroprozessoren, insbesondere der Mikrocontroller, sind die sog. **eingebetteten Systeme** (*Embedded Systems*). Hierunter versteht man Datenverarbeitungssysteme, die in ein technisches Umfeld integriert sind. Dort stellen sie ihre Datenverarbeitungsleistung zur Steuerung und Überwachung dieses Umfeldes zur Verfügung. Ein Beispiel für ein eingebettetes System ist etwa die mit einem Mikrocontroller oder Mikroprozessor realisierte Steuerung einer Kaffeemaschine. Hier dient die Datenverarbeitungsleistung dazu, die umgebenden Komponenten wie Wasserbehälter, Heizelemente und Ventile zu koordinieren, um einen guten Kaffee zu bereiten. Der PC auf dem Schreibtisch zu Hause ist hingegen zunächst kein eingebettetes System. Er wirkt dort als reines Rechnersystem zur Datenverarbeitung für den Menschen. Ein PC kann jedoch ebenfalls zu einem eingebetteten System werden, sobald er z.B. in einer Fabrik zur Steuerung einer Automatisierungsanlage eingesetzt wird.

Eingebettete Systeme sind weit zahlreicher zu finden als reine Rechnersysteme. Ihr Einsatzgebiet reicht von einfachen Steuerungsaufgaben für Haushaltsgeräte, Unterhaltungselektronik oder Kommunikationstechnik über Anwendungen in der Medizin und in Kraftfahrzeugen bis hin zur Koordination komplexer Automatisierungssysteme in Fabriken. In unserem täglichen Leben sind wir zunehmend von solchen Systemen umgeben. Die folgende Abb. 1.3 zeigt über der Zeitachse eine Auswahl von Geräten, durch die man von morgens 06:00 Uhr bis abends 22:00 Uhr mit eingebetteten Systemen in Kontakt kommt.

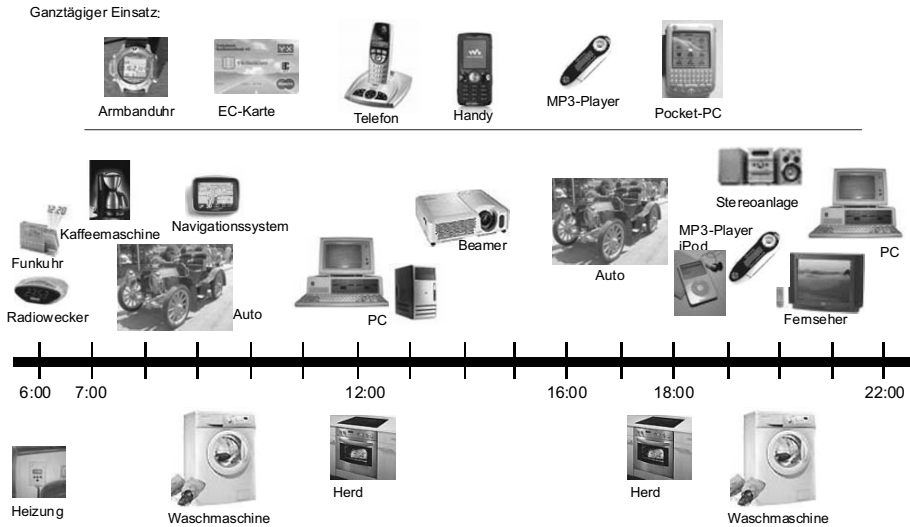


Abb. 1.3: Eingebettete Systeme in Geräten des täglichen Lebens

Als einfaches Beispiel aus dieser Geräte-Palette wollen wir in Abb. 1.4 den Aufbau eines sog. *MP3-Players* (MP3-Abspielgerät¹) darstellen. Die zentrale Komponente des Gerätes ist ein großer, nicht flüchtiger Speicherbaustein (Flash-EEPROM), d.h., er verliert auch nach dem Ausschalten der Betriebsspannung seinen Inhalt nicht.

Die Umwandlung der gespeicherten digitalen MP3-Dateien wird vom integrierten Digitalen Signalprozessor (DSP) und dem Digital/Analog-Wandler (DAC) vorgenommen. Der nachgeschaltete Verstärker (*Amplifier* – AMP) sorgt für die vom Kopfhörer benötigte elektrische Spannung und Leistung. Gesteuert werden der Speicherbaustein, der DSP und die anderen Peripheriekomponenten wie Tasten und Flüssigkristall-Anzeige (*Liquid Crystal Display* – LCD) vom Mikrocontroller (μC), der über eine USB-Schnittstelle (*Universal Serial Bus*) und den Anschluss-Stecker mit einem PC verbunden werden kann. Über den USB-Anschluss und eine integrierte Ladeschaltung kann bei manchen Geräten auch der Akku geladen werden.

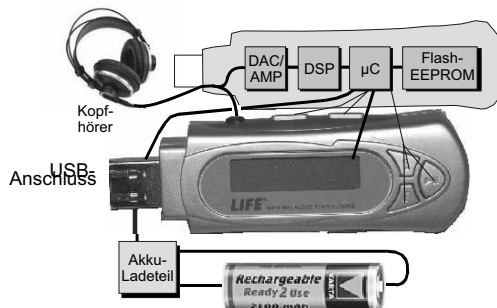


Abb. 1.4: Aufbau eines MP3-Players

¹ MP3 steht für MPEG-1 Audio Layer 3, MPEG für *Moving Picture Experts Group*

Gegenüber reinen Rechnersystemen werden an eingebettete Systeme einige zusätzliche Anforderungen gestellt:

- **Schnittstellenanforderungen:** Umfang und Vielfalt von Ein-/Ausgabeschnittstellen ist bei eingebetteten Systemen üblicherweise höher als bei reinen Rechnersystemen. Dies ergibt sich aus der Aufgabe, eine Umgebung zu steuern und zu überwachen. Hierzu müssen die verschiedensten Sensoren und Aktuatoren (Aktoren) bedient werden, welche über unterschiedliche Schnittstellentypen mit dem eingebetteten System verbunden sind.
- **Mechanische Anforderungen:** An eingebettete Systeme werden oft erhöhte mechanische Anforderungen gestellt. Für den Einsatz in Fabrikhallen, Fahrzeugen oder anderen rauen Umgebungen muss das System robust sein und zahlreichen mechanischen Belastungen standhalten. Aus diesem Grund werden z.B. gerne spezielle, mechanisch stabile Industrie-PCs eingesetzt, wenn – wie oben erwähnt – ein PC als eingebettetes System dient. Normale PCs würden den mechanischen Anforderungen einer Fabrikhalle oder eines Fahrzeugs nicht lange standhalten. Weiterhin werden der zur Verfügung stehende Raum und die geometrische Form für ein eingebettetes System in vielen Fällen vom Umfeld diktiert, wenn dieses System in das Gehäuse eines kleinen Gerätes wie z.B. eines Telefons untergebracht werden muss.
- **Elektrische Anforderungen:** Die elektrischen Anforderungen an eingebettete Systeme beziehen sich meist auf den Energieverbrauch und die Versorgungsspannung. Diese können aus verschiedenen Gründen begrenzt sein. Wird das System in eine vorhandene Umgebung integriert, so muss es aus der dortigen Energieversorgung mitgespeist werden. Versorgungsspannung und maximaler Strom sind somit vorgegebene Größen. In mobilen Systemen werden diese Größen von den Eigenschaften einer Batterie oder eines Akkumulators diktiert. Um eine möglichst lange Betriebszeit zu erzielen, sollte hier der Energieverbrauch so niedrig wie möglich sein. Ein niedriger Energiebedarf ist auch in Systemen wichtig, bei denen die Abwärme gering gehalten werden muss. Dies kann erforderlich sein, wenn spezielle isolierende Gehäuse (wasser- oder gasdicht, explosionsgeschützt usw.) den Abtransport der Abwärme erschweren oder die Umgebungstemperatur sehr hoch ist.
- **Zuverlässigkeitsanforderungen:** Bestimmte Einsatzgebiete stellen hohe Anforderungen an die Zuverlässigkeit eines eingebetteten Systems. Fällt z.B. die Bremsanlage eines Kraftfahrzeugs oder die Steuerung eines Kernreaktors aus, können die Folgen katastrophal sein. In diesen Bereichen muss durch spezielle Maßnahmen sichergestellt werden, dass das eingebettete System so zuverlässig wie möglich arbeitet und für das verbleibende Restrisiko des Ausfalls ein sicherer Notbetrieb möglich ist.
- **Zeitanforderungen:** Eingebettete Systeme müssen oft in der Lage sein, bestimmte Tätigkeiten innerhalb einer vorgegebenen Zeit auszuführen. Solche Systeme nennt man **Echtzeitsysteme**. Reagiert z.B. ein automatisch gesteuertes Fahrzeug nicht rechtzeitig auf ein Hindernis, so ist eine Kollision unvermeidlich. Erkennt es eine Abzweigung zu spät, wird es einen falschen Weg einschlagen.
- Eine weitere Anforderung an Echtzeitsysteme ist die längerfristige **Verfügbarkeit**. Dies bedeutet, dass ein solches System seine Leistung über einen langen Zeitraum hinweg unterbrechungsfrei erbringen muss. Betriebspausen, z.B. zur Reorganisation interner Datenstrukturen, sind nicht zulässig.

Der Echtzeit-Aspekt bedarf einiger zusätzlicher Erläuterungen. Allgemein betrachtet, unterscheidet sich ein Echtzeitsystem von einem Nicht-Echtzeitsystem dadurch, dass zur logischen Korrektheit die zeitliche Korrektheit hinzukommt. Das Ergebnis eines Nicht-Echtzeitsystems ist korrekt, wenn es den logischen Anforderungen genügt, d.h., z.B. ein richtiges Resultat einer Berechnung liefert. Das Ergebnis eines Echtzeitsystems ist nur dann korrekt, wenn es logisch korrekt ist und zusätzlich zur rechten Zeit zur Verfügung steht. Anhand der Zeitbedingungen können drei **Klassen von Echtzeitsystemen** unterschieden werden:

- **Harte Echtzeitsysteme** sind dadurch gekennzeichnet, dass die Zeitbedingungen unter allen Umständen eingehalten werden müssen. Man spricht auch von harten Zeitschranken. Das Überschreiten einer Zeitschranke hat katastrophale Folgen und kann nicht toleriert werden. Ein Beispiel für ein hartes Echtzeitsystem ist die bereits oben angesprochene Kollisionserkennung bei einem automatisch gesteuerten Fahrzeug. Eine zu späte Reaktion auf ein Hindernis führt zu einem Unfall.
- **Feste Echtzeitsysteme** definieren sich durch Zeitschranken, sog. feste Zeitschranken, deren Überschreitung das Ergebnis einer Berechnung zwar wertlos macht, ohne dass die Folgen unmittelbar katastrophal sind. Ein Ergebnis, das nach der Zeitschranke geliefert wird, kann verworfen werden, und es wird auf das nächste Ergebnis gewartet. Ein Beispiel ist die Positionserkennung eines automatischen Fahrzeugs. Eine zu spät gelieferte Position ist wertlos, da sich das Fahrzeug mittlerweile weiterbewegt hat. Dies kann zu einer falschen Fahrstrecke führen, die jedoch ggf. später korrigiert werden kann. Allgemein sind feste Echtzeitsysteme oft durch Ergebnisse oder Werte mit Verfallsdatum gekennzeichnet.
- **Weiche Echtzeitsysteme** besitzen Zeitschranken, die eher eine Richtlinie als eine harte Grenze darstellen. Man spricht deshalb auch von weichen Zeitschranken. Ein Überschreiten um einen gewissen Wert kann toleriert werden. Ein Beispiel ist die Beobachtung eines Temperatursensors in regelmäßigen Abständen für eine Temperaturanzeige. Wird die Anzeige einmal etwas später aktualisiert, so ist dies häufig nicht weiter schlimm.

Die zeitliche Vorhersagbarkeit des Verhaltens spielt für ein Echtzeitsystem die dominierende Rolle. Eine hohe Verarbeitungsgeschwindigkeit ist eine „nette Beigabe“, jedoch bedeutungslos, wenn sie nicht genau bestimmbar und vorhersagbar ist. Benötigt beispielsweise eine Berechnung in den meisten Fällen nur eine Zehntelsekunde, jedoch in wenigen Ausnahmefällen eine ganze Sekunde, so ist für ein Echtzeitsystem nur der größere Wert ausschlaggebend. Wichtig ist immer der schlimmste Fall der Ausführungszeit (*Worst Case Execution Time* – WCET). Aus diesem Grund sind z.B. moderne Mikroprozessoren für Echtzeitsysteme nicht unproblematisch, da ihr Zeitverhalten durch ihre leistungssteigernden Techniken (auf die hier nicht näher eingegangen werden kann) schwer vorhersagbar ist. Einfache Mikrocontroller ohne diese Techniken besitzen zwar eine weitaus geringere Verarbeitungsleistung, ihr Zeitverhalten ist jedoch genauer zu bestimmen.

Der Begriff ***Ubiquitous Computing*** wurde bereits Anfang der 90er Jahre des letzten Jahrhunderts von der Firma Xerox geprägt und bezeichnet eine Zukunftsvision: Mit Mikroelektronik angereicherte Gegenstände sollen so alltäglich werden, dass die enthaltenen Rechner als solche nicht mehr wahrgenommen werden. Die Übersetzung von „*ubiquitous*“ ist „allgegenwärtig“ oder „ubiquitär“, synonym dazu wird oft der Begriff „*pervasive*“ im Sinne von „durchdringend“ benutzt.

Ubiquitäre Systeme sind eine Erweiterung der eingebetteten Systeme. Als ubiquitäre (allgegenwärtige) Systeme bezeichnet man eingebettete Rechnersysteme, die selbstständig auf ihre Umwelt reagieren. Bei einem ubiquitären System kommt zusätzlich zu einem eingebetteten System noch Umgebungswissen hinzu, das es diesem System erlaubt, sich in hohem Maße auf den Menschen einzustellen. Die Benutzer sollen nicht in eine virtuelle Welt gezogen werden, sondern die gewohnte Umgebung soll mit Computerleistung angereichert werden, so dass neue Dienste entstehen, die den Menschen entlasten und ihn von Routineaufgaben befreien.

Betrachten wir das Beispiel eines Fahrkartenautomaten: Während bis vor einigen Jahren rein mechanische Geräte nur Münzen annehmen konnten, diese gewogen, geprüft und die Summe mechanisch berechnet haben, so ist der Stand der Technik durch eingebettete Rechnersysteme charakterisiert. Heutige Fahrkartenautomaten lassen eine Vielzahl von Einstellungen zu und arbeiten mit recht guter computergesteuerter Geldscheinprüfung. Leider muss der häufig überforderte Benutzer die Anleitung studieren und aus einer Vielzahl möglicher Eingabemöglichkeiten auswählen. In der Vision des *Ubiquitous Computing* würde beim Herantreten an den Fahrkartenautomaten der in der Tasche getragene „Persönliche Digitale Assistent“ (PDA) oder das Mobiltelefon über eine drahtlose Netzverbindung mit dem Fahrkartenautomaten Funkkontakt aufnehmen und diesem unter Zuhilfenahme des im PDA oder Handy gespeicherten Terminkalenders mitteilen, wohin die Reise voraussichtlich gehen soll. Der Fahrkartenautomat würde dann sofort unter Nennung des voraussichtlichen Fahrtziels eine Verbindung und eine Fahrkarte mit Preis vorschlagen, und der Benutzer könnte wählen, ob per Bargeldeinwurf gezahlt oder der Fahrpreis von der Geldkarte oder dem Bankkonto abgebucht werden soll.

In diesem Sinne wird der Rechner in einer dienenden und nicht in einer beherrschenden Rolle gesehen. Man soll nicht mehr gezwungen sein, sich mit der Bedienung des Geräts, sei es ein Fahrkartenautomat oder ein heutiger PC, auseinanderzusetzen zu müssen. Stattdessen soll sich das Gerät auf den Menschen einstellen, d.h., mit ihm in möglichst natürlicher Weise kommunizieren, Routinetätigkeiten automatisiert durchführen und ihm lästige Tätigkeiten, soweit machbar, abnehmen. Daraus ergeben sich grundlegende Änderungen in der Beziehung zwischen Mensch und Maschine. *Ubiquitous Computing* wird deshalb als die zukünftige dritte Phase der Computernutzung gesehen:

- Phase I war die Zeit der Großrechner, in der wegen seines hohen Preises ein Rechner von vielen Menschen benutzt wurde.
- Phase II, die heute vorliegt, ist durch die Mikrorechner geprägt. Diese sind preiswert genug, dass sich sehr viele Menschen einen eigenen Rechner leisten können – zumindest in den gut entwickelten Industriestaaten. Auch das Betriebssystem eines Mikrorechners ist üblicherweise für einen einzelnen Benutzer konfiguriert. Technische Geräte arbeiten heute meist schon mit integrierten Rechnern. Diese können jedoch nicht miteinander kommunizieren und sind nicht dafür ausgelegt, mittels Sensoren Umgebungswissen zu sammeln und für ihre Aktionen zu nutzen.
- Phase III der Computernutzung wird durch eine weitere Miniaturisierung und einen weiteren Preisverfall mikroelektronischer Bausteine ermöglicht. Diese Phase der ubiquitären Systeme soll es ermöglichen, den Menschen mit einer Vielzahl nicht sichtbarer, in Alltagsgegenstände eingebetteter Computer zu umgeben, die drahtlos miteinander kommunizieren und sich auf den Menschen einstellen.

Fünf Merkmale sind zur Kennzeichnung **ubiquitärer Systeme** hervorzuheben:

- Sie sind eine Erweiterung der „eingebetteten Systeme“, also von technischen Systemen, in die Computer eingebettet sind.
- Sie integrieren eingebettete Computer überall und in hoher Zahl (Allgegenwart).
- Sie nutzen drahtlose Vernetzung; dazu zählen die Technologien der Mobiltelefone, Funk-LAN (*Local Area Network*), Bluetooth und Infrarot.
- Sie verwenden Umgebungswissen, das es ihnen erlaubt, sich in hohem Maße auf den Menschen einzustellen.
- Sie binden neue Geräte ein, wie z.B. mobile „PCs“ (PDA, *Handheld*), Mobiltelefone und am Körper getragene („*wearable*“) Rechner.

Technisch gesehen sind für ein ubiquitäres System viele kleine, oftmals tragbare, in Geräten oder sogar am und im menschlichen Körper versteckte Mikroprozessoren und Mikrocontroller notwendig, die über Sensoren mit der Umwelt verbunden sind und bisweilen auch über Aktuatoren (Aktoren) aktiv in diese eingreifen. Verbunden sind diese Rechner untereinander und mit dem Internet über drahtgebundene oder drahtlose Netzwerke, die oftmals im Falle von tragbaren Rechnern spontan Netzwerke bilden. Die Rechnerinfrastruktur besteht aus einer Vielzahl unterschiedlicher Hardware und Software: kleine tragbare Endgeräte, leistungsfähige Server im Hintergrund und eine Kommunikationsinfrastruktur, die überall und zu jeder Zeit eine Verbindung mit dem „Netz“ erlaubt. Ein wesentliches Problem für die Endgeräte stellt die elektrische Leistungsaufnahme und damit die eingeschränkte Nutzdauer dar. Ein weiteres Problem ist die Bereitstellung geeigneter Benutzerschnittstellen: Die Benutzer erwarten auf ihre Bedürfnisse zugeschnittene und einfach zu bedienende, aber leistungsfähige Dienste.

Die Einbeziehung von Informationen aus der natürlichen Umgebung der Geräte stellt ein wesentliches Kennzeichen ubiquitärer Systeme dar. Die Berücksichtigung der Umgebung, des „Kontexts“, geschieht über die Erfassung, Interpretation, Speicherung und Verbindung von Sensordaten. Oftmals kommen Systeme zur orts- und richtungsabhängigen Informationsinterpretation auf mobilen Geräten hinzu. Verfügt ein Gerät über die Information, wo es sich gerade befindet, so kann es bestimmte Informationen in Abhängigkeit vom jeweiligen Aufenthaltsort auswählen und anzeigen. Das Gerät passt sich in seinem Verhalten der jeweiligen Umgebung an und wird damit ortssensitiv. Beispiele für ortssensitive Geräte sind die GPS-gesteuerten Kfz-Navigationssysteme (*Global Positioning System*), die je nach Ort und Bewegungsrichtung angepasste Fahrhinweise geben. Die GPS-Infrastruktur stellt an jeder Position die geographischen Koordinaten zur Verfügung, und das Navigationssystem errechnet daraus die jeweilige Fahrweisung.

Eine Mobilfunkortung ermöglicht lokalisierte Informationsdienste, die je nach Mobilfunkregion einem Autofahrer Verkehrsmeldungen nur für die aktuelle Region geben oder nahe gelegene Hotels und Restaurants anzeigen. Viele Arten von Informationen lassen sich in Abhängigkeit von Zeit und Ort gezielt filtern und sehr stark einschränken. Der tragbare Rechner entwickelt so ein regelrecht intelligentes oder kooperatives Verhalten.

1.3 Energiespartechniken

(Theo Ungerer)

Die Reduktion des Energiebedarfs stellt ein weiteres wichtiges Kriterium beim Rechnerentwurf dar – insbesondere von eingebetteten Systemen. Durch den Einsatz in immer kleiner werdenden mobilen Geräten ist die verfügbare Energiemenge durch Batterien oder Akkumulatoren begrenzt. Es gilt, möglichst lange mit der vorhandenen Energie auszukommen. Im Wesentlichen lassen sich drei Ansätze unterscheiden:

- Verringerung der Taktfrequenz,
- Verringerung der Versorgungsspannung,
- Optimierung der Mikroarchitektur.

Die Verringerung der Taktfrequenz ist zunächst ein einfacher und wirkungsvoller Weg, den Energiebedarf zu reduzieren. Bei modernen CMOS-Schaltungen (*Complementary Metal-Oxide Semiconductor*) ist der Energiebedarf E proportional zur Taktfrequenz F , d.h.,

$$E \sim F$$

Eine Halbierung der Taktfrequenz reduziert daher auch den Energiebedarf auf die Hälfte. Leider wird hierdurch die Verarbeitungsgeschwindigkeit ebenfalls halbiert.

Eine zweite Möglichkeit zur Reduktion des Energiebedarfs ist die Reduktion der Versorgungsspannung. Der Energieverbrauch ist proportional zum Quadrat der Spannung U , d.h.,

$$E \sim U^2$$

Eine Reduktion der Spannung auf siebzig Prozent führt somit ebenso (ungefähr) zu einer Halbierung des Energiebedarfs.

Variiert man Versorgungsspannung und Taktfrequenz gleichzeitig, so erhält man:

$$E \sim F \cdot U^2$$

Versorgungsspannung und Taktfrequenz sind hierbei aber keine unabhängigen Größen. Je geringer die Versorgungsspannung, desto geringer auch die maximal mögliche Taktfrequenz. Näherungsweise kann hierfür ein linearer Zusammenhang angenommen werden:

$$F_{\max} \sim U$$

Setzt man diesen Zusammenhang in die obige Gleichung für den Energiebedarf ein, so erhält man die **Kubus-Regel** für eine simultane Änderung der Taktfrequenz und Versorgungsspannung:

$$E \sim F^3 \quad \text{oder} \quad E \sim U^3$$

Eine Reduktion von Taktfrequenz und Versorgungsspannung verringert neben dem Energiebedarf auch die Verarbeitungsleistung. Forschungsansätze gehen deshalb dahin, Taktfrequenz und Versorgungsspannung eines Mikrocontrollers im Hinblick auf die Anwendung zu optimieren. Dazu wird z.B. der Versuch unternommen, in einem Echtzeitsystem Versorgungsspannung und Taktfrequenz eines Prozessors dynamisch an die einzuhaltenen Zeitschranken anzupassen. Besteht für eine Aufgabe sehr viel Zeit, so muss der Prozessor nicht mit voller Geschwindigkeit arbeiten, sondern kann mit reduzierter Taktfrequenz und damit reduziertem Energieverbrauch operieren.

Auch im Bereich der universellen Mikroprozessoren werden ähnliche Ansätze verfolgt. So regelte beispielsweise der Crusoe-Prozessor von Transmeta Taktfrequenz und Versorgungsspannung anhand der durchschnittlichen Prozessorauslastung. Diese wurde durch Messung der Prozessorruhezeiten (*Idle Times*) bei der Prozessausführung ermittelt.

Weitere Ansätze versuchen, die Versorgungsspannung einer CMOS-Schaltung – abhängig von Betriebsparametern, wie etwa der Temperatur – derart zu regeln, dass die Schaltung gerade noch fehlerfrei funktioniert. Ein anderer Forschungszweig befasst sich mit dem Ziel, die Mikroarchitektur eines Mikroprozessors derart zu optimieren, dass der Energiebedarf ohne Einbußen in der Verarbeitungsgeschwindigkeit gesenkt werden kann. Hier ist eine Reihe von Maßnahmen denkbar:

- **Reduktion der Busaktivitäten:** Bei konsequentem Einsatz der von RISC-Prozessoren bekannten *Load-Store-Architektur*, bei der nur durch die Lade- bzw. Speicherbefehle – und nicht z.B. durch einen arithmetischen Befehl – auf den Speicher zugegriffen wird, arbeiten die meisten Prozessor-Operationen auf den internen Registern. Die Busschnittstelle wird während dieser Zeit also nicht benötigt und kann als Energieverbraucher abgeschaltet bleiben. Ein entsprechend mächtiger interner Registersatz hilft ebenfalls, externe Buszugriffe zu verringern. Eine weitere Möglichkeit besteht darin, wo es sinnvoll ist, schmale Datentypen (8 oder 16 Bits anstelle von 32 Bits) zu verwenden. Dabei muss zur Datenübertragung nicht die volle Busbreite aktiviert werden.
- **Statisches Power-Management:** Dem Anwender können Befehle zur Verfügung gestellt werden, die Teile des Mikroprozessors abschalten (z.B. Speicherbereiche, Teile des Rechenwerks) oder den ganzen Mikrorechner in eine Art „Schlafmodus“ versetzen (s. Unterabschnitt 2.3.2). Dadurch kann der Anwender gezielt den Energieverbrauch kontrollieren.
- **Dynamisches Power-Management:** Neben dem statischen Power-Management kann der Prozessor dynamisch für jeden gerade bearbeiteten Befehl die zur Verarbeitung nicht benötigten Komponenten abschalten. Dies kann z.B. in den verschiedenen Stufen der Befehlsbearbeitung (s. Abschnitt 5.2) erfolgen.
- **Erhöhung der Code-Dichte:** Die Code-Dichte kennzeichnet einerseits die Anzahl der Befehle, die für eine Anwendung benötigt werden – was auch als „Mächtigkeit“ eines Befehlsatzes bezeichnet wird. Andererseits bezeichnet sie aber auch die durchschnittliche Anzahl von Bits, die für die Realisierung der Befehle benötigt werden.
Je höher die Code-Dichte, desto weniger bzw. kürzere Befehle sind erforderlich. Eine hohe Code-Dichte spart Energie aus zwei Gründen: Erstens wird weniger Speicher benötigt und zweitens fallen weniger Buszyklen zur Ausführung einer Anwendung an. Von diesem Gesichtspunkt aus stellt eine RISC-Architektur also eher einen Nachteil dar, da durch die einfacheren Instruktionen konstanter Bitbreite die Code-Dichte gegenüber CISC-Architekturen im Allgemeinen geringer ist.

Man sieht, dass durchaus komplexe Zusammenhänge zwischen Architektur und Energiebedarf bestehen. Es ist somit wünschenswert, möglichst frühzeitig bei der Entwicklung eines neuen Mikrorechners Aussagen über den Energiebedarf machen zu können. Neue Forschungsarbeiten zielen deshalb darauf ab, Modelle zur Vorhersage des Energiebedarfs zu entwickeln, die Aussagen in frühen Entwicklungsschritten erlauben. Ziel ist es, zusammen mit den ersten funktionalen Simulationen eines Mikroprozessors auch Daten über den wahrscheinlichen Energiebedarf zu erhalten.

Anwendungsorientierte Mikroprozessoren
Mikrocontroller und Digitale Signalprozessoren

Bähring, H.

2010, XIII, 496 S. 325 Abb., Softcover

ISBN: 978-3-642-12291-0