Herz des Computers: (Folge 1)

Innenleben

Das Betriebssystem ist zum Betrieb des Computers unentbehrlich. Aber was ist das eigentlich, ein Betriebssystem? Was macht es, woraus besteht es und wozu wird es gebraucht? Fragen, die viele Einsteiger bewegen und hier eine Antwort finden.

von Mark Deskowski
und David Göhler

Das Betriebssystem ist neben der Hardware das Herzstück des Computers. Man rechnet es zur Software, also zu dem Teil des Computers, den man nicht anfassen kann (Hardware ist all das, was man anfassen kann). Es sorgt dafür, daß Ihr Computer kurz nach dem Einschalten betriebsbereit vor Ihnen steht, daß Mausbewegungen sichtbar werden, Disketten gelesen werden können, sich Fenster öffnen und Tastatureingaben auch etwas bewirken.

Als Anwender, der zum ersten Mal seinen Computer einschaltet, sehen Sie vom Betriebssystem einen Bildschirm mit Disketten-Symbolen und einem Mauszeiger, der durch Bewegen der Maus auf dem Bildschirm hin- und hergeschoben werden kann.

»Sehen« trifft's nicht ganz, Sie sehen nur die Ergebnisse der Betriebssystemarbeit, das OS (Abk. von »Operating System«, dem englischen Begriff für Betriebssystem) selbst kann man natürlich nicht sehen. Das ist ja Software (und die kann man weder anfassen noch sehen).

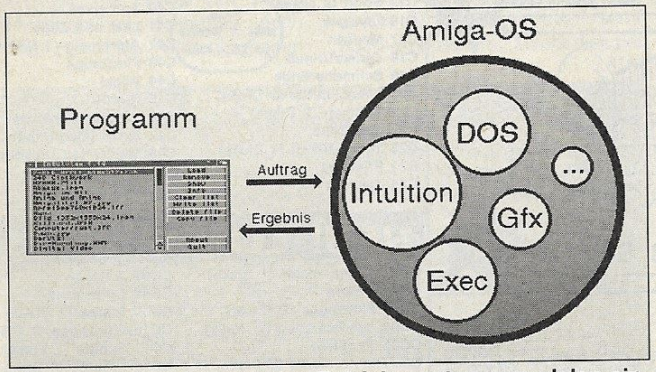
Vereinfacht betrachtet, ist das Betriebssystem ein großes Programm mit vielen Teilen, von denen sich jedes um eine bestimmte Aufgabe kümmert. Die Aufgaben sind recht verschieden. Einige Teile widmen sich Maus, Tastatur und Joystick, andere verwalten Disketten und Festplatten, wieder andere haben Aufgaben, die man kaum mitbekommt: sie verwalten den eingebauten Hauptspeicher und die Teile, an denen ein Drucker oder andere Geräte angeschlossen sind.

Was macht nun so ein Teil wirklich? Nehmen wir einen einfachen Fall an: Sie drücken auf die Taste, die mit dem Buchstaben »z« beschriftet ist. Damit wird zwischen zwei Drähten unter der Tastenkappe eine Verbindung hergestellt. Ein kleiner Computer in der Tastatur (da ist wirklich einer!) merkt das, da er ständig alle Leitungen kontrolliert, und schickt die Nummer der gedrückten Taste an den »großen« Computer.

Dieser kriegt ein spezielles Signal, wenn eine Tastennummer an ihn übermittelt wurde, die ungefähr »Hey, der Benutzer hat eine Taste gedrückt, kümmere dich mal drum« bedeutet. Der Schwerstarbeiter im Computer, der Prozessor (auch CPU genannt), wird aus seiner normalen Arbeit gerissen, kümmert sich um die Tastennummer (sprich, er tut sie zu den anderen Tastennummern) und guckt nach, welches Programm jetzt eigentlich dran ist, Tastendrücke zu verarbeiten.

Ist das ermittelt, muß noch festgestellt werden, welcher Buchstabe zu der Taste mit der übermittelten Nummer gehört und – um das Ganze abzukürzen – irgendwann erscheint der Buchstabe »z« auf dem Bildschirm. Erstaunlich, wie schnell das noch geht.

Wozu eigentlich dieser Aufwand? Berechtigte Frage; sobald die Tastennummer an den Computer übermittelt wurde, könnte man sie ja solange liegen lassen, bis das Programm, was gerade läuft, danach fragt und die Nummer einliest und verarbeitet. Dann müßte jedes Programm, das Tastatureingaben verarbeiten will, die Sache selbst in die Hand nehmen. Jedes Programm müßte die Tastennummer interpretieren und einem Buchstaben zuordnen. Jedes Programm müßte oft genug die Nummer abfragen, um keine Taste zu verpassen; müßte eine Warteschlange verwalten, um bei vielen Tastendrücken erst einmal alle aufzusammeln und später zu verarbeiten.



Modular: Intern besteht das Betriebssystem aus vielen einzelnen Teilen, die Programmen zur Verfügung stehen

Das Betriebssystem ist König und Sklave zugleich

Alle diese Programme müßten das gleiche tun. Da macht es doch Sinn, die Teile, die man in jedes Programm integrieren müßte, herauszunehmen und allen Programmen zur Verfügung zu stellen.

Das gleiche gilt natürlich auch für die Teile der Programme, die man sehen kann: Fenster, Screens, Buchstaben, Linien, Menüs und Bildchen (meistens als Icons bezeichnet), auf die man klicken kann. Für Programmierer ist es viel einfacher, dem Betriebssystem zu sagen: »Du, mach mal ein Fenster an dieser oder jener Position auf«, als jede Linie, jeden Punkt und Buchstaben des Fensters selbst mit einem eigenen Befehl auf den Bildschirm zu pinseln.

Das Betriebssystem besteht also hauptsächlich aus einer Reihe von Funktionen, die jedes Programm nutzen kann.

Das ist aber nicht der einzige Vorteil: Nehmen wir (wieder) mal an, es hat jemand ein Programm geschrieben, das einen eigenen Bildschirm und darauf einige Fenster öffnet. Dieses Programm soll auf allen Amigas funktionieren, egal wie alt oder neu sie sind und ob sie eine Grafikkarte eingebaut haben oder nicht.

Einige Amigas können nun maximal 16 Farben auf einmal auf einem Screen darstellen, andere (neuere wie ein Amiga 1200) 256; mit Grafikkarte sind viel größere Bildschirmauflösungen möglich als ohne Erweiterung. Wer das Betriebssystem dann nicht nutzt, muß für alle verschiedenen Rechner und Grafikkarten jeweils eigene Routinen schreiben. Gibt es in der Zukunft dann neue Amigas, muß er sein Programm erweitern, um es auch auf diesen funktionsfähig zu halten.

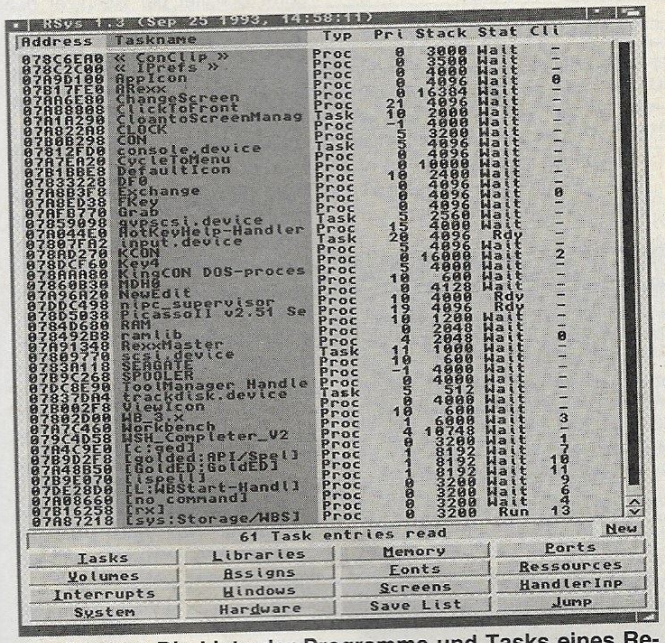
Viel einfacher und sicherer ist es, *nur* Betriebssystemfunktionen zu nutzen, da das System *weiß*, ob es auf einem alten oder neuen Amiga arbeitet. Gibt es neuere Maschinen, wird von Commodore das OS automatisch angepaßt. Das Programm kann man dann völlig unverändert lassen, es wird auch auf den neueren Computern funktionieren.

Amiga-Betriebssystemversionen

Version	Bemerkung
OS 1.0	Die erste Version von 1984. Nur in den USA ausgeliefert.
OS 1.1	Die erste in Deutschland erhältliche Amiga-OS-Version nur für den Amiga 1000 (reine Disketten-Version).
OS 1.2	Version für Amiga 500 und 2000. Stark fehlerbereinigt und im Aussehen leicht verändert.
OS 1.3	Autoboot eingeführt. Ab 1.3 ist das Starten (Booten) von der Festplatte (oder vom Netz) möglich.
OS 2.0	Mit Amiga 3000 eingeführt. Erstmalig ist das OS 512 KByte groß; die Oberfläche hat 3-D-Aussehen (auch OS/2.0-Look genannt); viele neue Funktionen und Libraries; viele von Grund auf neu programmierte Routinen; viele Fehler behoben. ARexx gehört ab jetzt zum Lieferumfang.
OS 2.1	Reines Softwareupdate, Einführung der Locale-Library (Sprache der Workbench und vieler Programme einstellbar), DOSDrivers, besseres Monitor-Konzept.
OS 3.0	Einführung von DataTypes, AmigaGuide und Multiview. Unterstützung des AA-Chipsatzes von Amiga 1200 und 4000. Viele Optimierungen bei der Grafikausgabe.
OS 3.1	Fehlerbereinigung von OS 3.0, neue Libraries, zusätzliches CD-ROM-Dateisystem. Erstmalig im CD^{32} eingesetzt.

des Amiga

Einige Spieleprogrammierer haben das nie verstanden, oder sich nie dazu aufgerafft, das Betriebssystem und die Konventionen zu beachten. Sie haben alles selbst geschrieben, ohne darauf zu sehen, ob das nicht schon im Betriebssystem vorhanden ist. Änderten sich dann die Hardware, läuft das Spiel nicht mehr.



Der Bär los: Die Liste der Programme und Tasks eines Redaktions-Amiga. Die Task-Namen sind grau unterlegt.

Das OS ist also ein Mittler, der Programme unabhängig von der Hardware macht. Das hat viele Vorteile: Der Benutzer kann sich sicher sein, daß alte Programme auch dann laufen, wenn man sie auf einem neuen Computer benutzt. Ein neueres Betriebssystem ist immer »abwärtskompatibel«: Alle Funktionen des Vorgängers finden sich auch in der aktuellen Version. Programme, die sich also auf Betriebssystemfunktionen einer älteren Version stützen, werden sie auch in der neueren Version finden.

Ein Programm nennt sich »aufwärtskompatibel«, wenn es nicht nur mit der OS-Version zusammen funktioniert, für die es geschrieben wurde, sondern auch noch mit allen neueren, die eventuell noch kommen. Commodore hat für die Benutzung der OS-Funktionen sogar Richtlinien herausgegeben, an die sich jeder Programmierer halten sollte.

Benutzen Programmierer in allen möglichen Fällen das Amiga-OS, hat das außerdem den Vorteil, daß alle Programme gleich aussehen, die Menüs immer gleich funktionieren, Mausklicks immer die gleichen Aktionen nach sich ziehen. Als Einsteiger weiß man das zu schätzen, wenn nicht jedes Programm anders funktioniert.

Die Aufgaben sind nun klar. Wahrscheinlich haben Sie aber noch keine genaue Vorstellung davon, was das OS ist. Vorhin hieß es, es sei ein großes Programm, dann wieder eine Sammlung von Funktionen; was denn nun?

Alles zusammen. Vielleicht wird es klarer, wenn man versucht nachzuvollziehen, was beim Anschalten des Rechners passiert: Kaum hat der erste Strom den Prozessor ereilt, sucht der nach Befehlen, die er ausführen kann. Die findet er immer in einem ROM (Read only memory), das schon das Betriebssystem im ganzen enthält, oder nur soviel beinhaltet, daß das Betriebssystem von Diskette oder Festplatte geladen werden kann.

Danach wird der Kern des Amiga-OS aktiviert. Im Prinzip ist das ein Super-Programm, das die Oberaufsicht führt, andere Programme lädt, ausführt, beendet, Funktionen bereitstellt und Anfragen so gut es geht, erledigt. So eine Anfrage könnte sein: »Du, OS, ich will Fenster öffnen, wo finde ich denn die Funktionen dazu?«, worauf es einen Hinweis kriegt, wo sie liegen.

Die Funktionen sind nach Aufgabengebieten zusammengefaßt und heißen »Libraries« (zu deutsch: Bibliotheken). Einige sind fest eingebaut, selten gebrauchte findet man als Dateien im Verzeichnis »Libs:«. Kommt eine Anfrage wie gerade geschildert, und stellt das OS fest, daß die Funktionen noch gar nicht existieren, sucht es in »Libs:« danach und lädt sie (wenn sie dort zu finden waren, sonst gibt es ein »ham wa nich« an das Programm zurück, das seinerseits dann auf dem Schlauch steht und meist den Benutzer darauf hinweist, daß eine Library »xyz« nicht zu finden war).

Einige Libraries sind vielen Benutzern von den Namen her schon geläufig: »Intuition« hält alles für Fenster, Menüs und Bildschirme bereit, »Gadtools« hantiert mit »Gadgets«, den Schaltern, auf die man klicken kann und damit Aktionen auslöst, und »DOS« (das **D**isk **O**perating **S**ystem) beinhaltet Funktionen für Dateien, Verzeichnisse und Festplatten.

Ähnliche Libraries haben auch andere Betriebssysteme, wie »Windows« und »MS-DOS«. Eine ist jedoch etwas Besonderes: »Exec«. Hier steckt der Kern des »Multitasking«.

Ein Multitasking-Betriebssystem ist in der Lage, mehrere (multi) Programme (auch Tasks genannt) gleichzeitig auszuführen. Fast jedenfalls. So richtig gleichzeitig kann es nicht gehen, weil im Gehäuse nur ein Prozessor steckt. Es kann immer nur ein Programm zur Zeit laufen. Jedoch läßt es sich einrichten, daß in sehr kurzen Abständen (etwa alle 20 Millisekunden) das »Superprogramm« des Betriebssystems aufwacht, nachguckt, welches Programm jetzt an der Reihe ist, und die CPU darauf losläßt. Nach weiteren 20 Millisekunden wird wieder nachgeschaut, die Kontrolle ans nächste weitergegeben usw. Eine Sekunde hat 50 x 20 Millisekunden. Selbst wenn zehn Programme gleichzeitig laufen, kommt jedes fünfmal in einer Sekunde an die Reihe.

Der Unterschied zwischen Windows und dem Amiga-OS ist, daß bei Windows das aktive Programm selbst entscheiden kann, ob es die Kontrolle ans Betriebssystem zurückgeben will, und beim Amiga-OS das Programm – ob es ihm gerade paßt oder nicht – einfach abgemeldet wird und erst später wieder drankommt.

Das Windows- wie auch das Mac-Multitasking nennt sich »kooperatives« Multitasking, da die beteiligten Programme kooperativ sein müssen, damit die anderen auch mal drankommen. Beim »preemptive« (zuvorkommenden) Multitasking hat das einzelne Programm keinen Einfluß darauf, wann es unterbrochen wird; das OS kommt ihm zuvor, verliert also nie die Kontrolle über die Programme (soweit die Theorie, in der Praxis gibt es Ausnahmen).

Aus einem Nicht-Multitasking-System (wie MS-DOS) ein kooperatives zu machen, ist recht einfach. Die Programme laufen nur nebeneinander her und wenn es gerade paßt, kommt ein anderes dran. Daß Windows und Mac-OS bis jetzt keine Preemptive-Multitasking-Systeme sind, zeigt, wie schwierig es ist, das nachträglich einzubauen.

Streit im Computer

Wie im wirklichen Leben, wo man sich gern und oft um das streitet, was selten ist, geht es auch im Computer zu: Was tun, wenn mehrere Programme zur gleichen Zeit auf die Festplatte

Kursübersicht

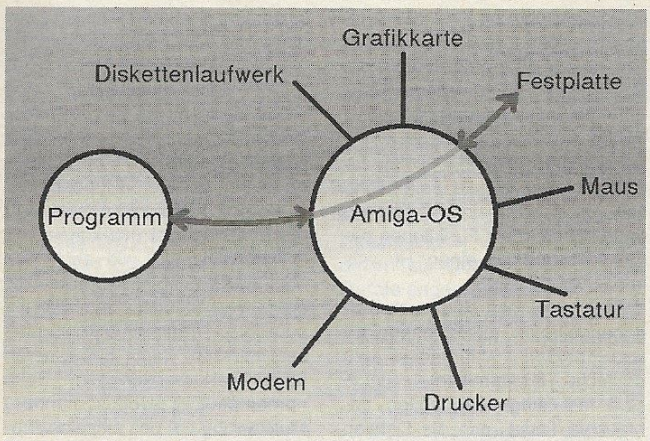
Als Anfänger hat man es nicht leicht. Programmnamen, Fachbegriffe und englische Bezeichnungen machen einem das Leben schwer. Hier hilft die Einsteiger-Serie mit Grundlagen in einfachen Worten.

Folge 1: Betriebssystem: Was ist das? Wie funktioniert ein Betriebssystem, was für Aufgaben hat es, Vorteile und Nachteile, was heißt Multitasking?

Folge 2: Die Komponenten eines Computers. Woraus besteht der Amiga, wie funktionieren die Komponenten und was heißt eigentlich RAM, ROM, Harddisk, CPU, CD-ROM, AA-Chip ...?

Folge 3: Die CPU: das unbekannte Wesen. Was macht der Prozessor im Rechner, wie funktioniert der genau, was sind die Unterschiede und was ist ein Assembler?

Folge 4: Das große Einmaleins der Dateien. Wie sind Disketten und Festplatten aufgebaut, wozu muß man Disketten formatieren und warum sind PC-Disketten anders?



Kommunikativ: Das Amiga-OS dient als Mittler zwischen Programmen und der Hardware im Computer

zugreifen wollen? Das kommt öfter vor, als man denkt. Kennen Sie den Moment, wenn der Requester »Datenträger xxx ist voll« erscheint?

In solchen Momenten kann der Amiga-Benutzer ein weiteres Programm starten, den Inhalt des Datenträgers untersuchen, nicht mehr Benötigtes löschen und dann auf »Weiter« klicken. Bei der parallelen Schnittstelle, an der man einen Drucker anschließen kann, geht das nicht. Das gäbe auch ein schönes Kuddelmuddel, wenn zwei Programme gleichzeitig drucken würden.

Es gibt also Dinge, die selten oder nur einmal vorhanden sind. Der Zugriff darauf muß geregelt werden, weil sonst mehrere Programme quasi gleichzeitig zugreifen und damit Chaos verursachen können. Für fast alle limitierten Dinge gibt es »Devices« (logische Geräte), die fast wie die Libraries funktionieren, aber einen entscheidenden Vorteil haben.

Bei Libraries ruft man eine Funktion auf und muß warten, bis das OS diese ausgeführt hat. Erst dann geht es im Programm weiter. Stellen Sie sich vor, die Funktion kann nicht ausgeführt werden, das Programm muß warten und kann nicht reagieren. Der Benutzer klickt schon wie wild im Fenster rum, weil er meint, das Programm habe sich aufgehängt, dabei wartet es nur darauf, daß das Betriebssystem endlich mit der Ausführung der aufgerufenen Funktion fertig wird.

Da wäre es doch besser, man könnte einen Auftrag erteilen, wie »lies mal die Daten ein und sag' Bescheid, wenn Du fertig bist, ich mache derweil was anderes«. Genau das macht ein Device aus: daß man nicht auf die Erledigung warten muß. Ein Device kann auch mehrere physikalische Einheiten wie Festplatten verwalten. Im Auftrag wird deshalb festgehalten, für welche Einheit (Festplatte) ein Befehl gedacht ist.

Außerdem läßt sich gleich ein ganzer Schwung von Aufträgen erteilen, die dann nach und nach abgearbeitet werden. Wie bei den Libraries sind wichtige Devices ständig vorhanden, andere werden erst bei Bedarf geladen.

Der Vorteil liegt aber in der Unabhängigkeit. Deshalb gibt es für die serielle und parallele Schnittstelle, die Diskettenlaufwerke, den Drucker und die Festplatten Devices und keine Libraries, da Aktionen mit der genannten Hardware auf sich warten lassen können und ein Programm davon unabhängig sein sollte.

Neben- und Miteinander

Das Pfiffige am Multitasking ist aber nicht nur das Nebeneinander, sondern auch das Miteinander. Programme können auch Mitteilungen und Aufträge an andere Programme verschicken oder auf Nachrichten warten.

Eine Sprache, die ganz auf diesem Prinzip beruht, ist »Rexx«, die in der Amiga-Version »ARexx« heißt. Sie wartet darauf, eine Liste von Befehlen zu erhalten um sie anschließend auszuführen. Das ist das eine. Andererseits kann ARexx aber auch anderen Programmen Befehle erteilen und diese damit (fern)steuern. So kann ein Programm ein anderes steuern.

Eine ganz andere Sache, die ein Betriebssystem beherrschen muß, ist das Hantieren mit Dateien, Verzeichnissen und Datenträgern. Schaltet der Benutzer den Rechner aus, sind alle Daten im RAM (**R**andom **A**ccess **M**emory) futsch. Wer also diese dauerhaft sichern will, muß sie auf Disketten oder Festplatten speichern.

Für die Wartung der vielen Daten hat der Anwender mehrere Möglichkeiten: Beispielsweise kann er mit der Maus, durch Klicken und Ziehen von Symbolen und Bildchen Dateien erzeugen, verschieben, kopieren und löschen.

Der Benutzer löscht hierbei nicht selbst, er erteilt dem Betriebssystem nur den Auftrag, das zu tun. Solch ein Auftrag wird weggeschickt, bearbeitet und mit der Meldung »OK« oder »Fehlgeschlagen« zurückgereicht. Im Falle der Mausbedienung gibt es das Programm »Workbench«, das die Benutzer-Aktionen in Aufträge umwandelt und wegschickt.

Statt per Mausbewegungen und Klicks, Bildchen und Fenster, arbeitet man in der »Shell« mit der Tastatur. Aufträge werden als Wortfolgen formuliert und mit der Return-Taste »abgeschickt«. Die Shell wandelt wiederum diese Befehle in Aufträge für das Betriebssystem um und liefert die Ergebnisse in Form von Textausgaben zurück. Gerade bei komplizierten und oft zu wiederholenden Aktionen bietet die Shell bessere Möglichkeiten, schnell zum Ziel zu gelangen.

Zum anderen gibt es Hilfsprogramme wie »DirOpus«, die eine einfache und mächtige Oberfläche besitzen, und noch einfacher als die Workbench durch Mausbewegungen und Klicks zu steuern sind.

Man sollte dabei aber nicht vergessen, daß man zwar unterschiedlich agiert, die Aktionen und Aufträge aber fast die gleichen sind. Das ist ein weiterer Vorteil beim Amiga-OS, daß man sowohl mit der Maus als auch mit der Tastatur seinen Computer steuern kann. *dg*

Fachbegriffe

Benutzeroberfläche: Das, was man von einem Programm sieht und mit dem man arbeiten muß, um es zu steuern. Auch Bezeichnung für die Workbench, die ebenfalls eine bedienerfreundliche Benutzeroberfläche hat.

Betriebssystem: Software, die den Betrieb des Computers möglich macht, Funktionen bereitstellt und knappe Güter (wie RAM, Festplatten, Schnittstellen etc.) verwaltet.

Booten: Hochfahren des Rechners (oft auch Neustart genannt). Passiert nach dem Anschalten, durch die Tastenkombination <Ctrl Amiga_links Amiga_rechts> oder Programmabsturz.

Device: Software: Teil des Betriebssystems, das ein bestimmtes Gerät oder eine Schnittstelle steuert und dabei unabhängig arbeiten kann. Hardware: engl. Bezeichnung für ein Gerät, z.B. Hardddisk Device.

DOSDrivers: Software, die Daten auf einem Datenträger als Dateien und Verzeichnisse erscheinen läßt und Operationen darauf erlaubt.

Handler: siehe DOSDrivers

Hardware: Der Teil des Computers, den man sehen und meist auch anfassen kann und der nach dem Ausschalten nicht verschwindet.

KByte: Eine Größenangabe. Ein Byte besteht aus 8 Bits, ein KByte (sprich KByte, **nicht** Kilobyte (!)) sind 1024 Byte. Das sind genau 2^{10} Byte. 1 MByte sind 1024 KByte, also 1024 x 1024 = 1 048 576 Byte.

Kompatibel: ist ein Programm, wenn es zum Betriebssystem paßt, also problemlos damit funktioniert. Auf- und abwärtskompatibel heißt, daß ein Programm auch noch mit der nächsten oder vorherigen Version des OS funktioniert.

Libraries: Funktionsbibliotheken des Betriebssystems, die nach Themen geordnet, für Kompatibilität sorgen, einheitliches Aussehen und gleiche Bedienung gewährleisten sowie oft gebrauchte Funktionen enthalten.

Multitasking: Bezeichnung für die Fähigkeit eines OS, mehrere Programme quasi gleichzeitig auszuführen und den Datenaustausch zwischen Programmen zu ermöglichen.

OS: **O**peration **S**ystem, englisch für Betriebssystem.

RAM: **R**andom **A**ccess **M**emory, Speicher auf den frei zugegriffen werden kann, dessen Inhalte gelöscht, beschrieben und gelesen werden können. Ohne Strom geht der Inhalt jedoch komplett verloren.

ROM: **R**ead **O**nly **M**emory, Speicher der nur gelesen werden kann, dafür aber auch ohne Strom seinen Inhalt behält. Darin finden sich oft Teile des Betriebssystems.

Software: Programme und das Betriebssystem sind Software. Diese kann als Datei auf einer Diskette, CD oder Festplatte liegen, ins RAM kopiert werden oder fest in ein ROM eingebrannt sein.

Style Guide: Von Commodore vorgegebene Programmierrichtlinien, an die sich alle Programmierer halten sollten, um höchste Kompatibilität zu erreichen.

Workbench: englisch für Arbeitstisch. Bezeichnet das Programm, das die Bedienung per Maus und Icons ermöglicht. Das Programm läuft auf dem Workbench-Screen und wird zum Betriebssystem gezählt. Fälschlicherweise wird das Amiga-OS oft auch Workbench genannt, da sie der sichtbare Teil des OS ist.