JADOS INTERN

ALLGEMEINES

Im vorliegenden Abschnitt werden tiefere Einblicke in das Betriebssystem geboten. Insbesondere werden die Speicherverwaltung und die Dateiverwaltung behandelt.

SPEICHERVERWALTUNG

JADOS verwaltet den ersten zusammenhängenden Speicherbereich des NKC. Das Speichermodell richtet sich danach, ob das Grundprogramm ab Adresse 0 beginnt oder nicht.

Wenn das Grundprogramm nicht ab Adresse 0 beginnt, dann existiert noch ein zweiter lokaler Speicherbereich hinter dem Grundprogramm, der aber ausschließlich vom Grundprogramm verwaltet wird. Auch ist es möglich, daß das Grundprogramm von einem Massenspeicher in den ersten Speicherbereich geladen wird und nach einem Reset seinen Betrieb aufnimmt. JADOS verwaltet dann nur den Bereich vor dem Grundprogramm.

Es gibt keine Möglichkeit, JADOS in Eproms betriebsfähig zu machen. Zwar ist das Betriebssystem vollständig unabhängig von seiner Ablaufadresse (relokativ). Es richtet aber relativ zu seiner Anfangsadresse mit genau definierten Offsets den Datenspeicher und den Stack ein.

Der von JADOS verwaltete Speicher besteht aus bis zu sieben Bereichen, die teils festgelegt, teils variabel sind.

Kernpunkt ist der Programmteil von JADOS. Er wird vom Urlader in Abhängigkeit vom Speicherausbau, der Ramdisk und der Lage des Grundprogramms ausgerechnet. JADOS selbst prüft nach dem Start, ob es an dieser Adresse auch tatsächlich beginnt. Die Größe des Programms beträgt zur Zeit 28 KByte.

An das Programm schließt der Datenteil an. Auch er ist festgelegt. Zur Zeit beträgt seine Größe 8 KByte. Der Beginn des Speichers liegt 28 KByte hinter dem Beginn des Programms.

Auf den Datenspeicher folgt die Ramdisk, falls sie eingerichtet wurde. Die Ramdisk kann nur im Urlader eingerichtet werden. Der Urlader prüft ob die Ramdisk schon vorhanden ist und läßt sie dann unverändert. Nur wenn sie nicht vorhanden ist, dann wird dem Benutzer die Gelegenheit gegeben, sie einzurichten. Die Ramdisk beansprucht den gesamten Speicher zwischen dem Datenteil des JADOS und dem Ende des ersten Speicherbereichs bzw. dem Beginn des Grundprogramms, falls sich dieses im Ram befindet.

Vor dem Programmteil von JADOS liegt der Systemstack, der mit 2 KByte genügend Platz bietet. Der Systemstack wird ausschließlich intern benutzt.

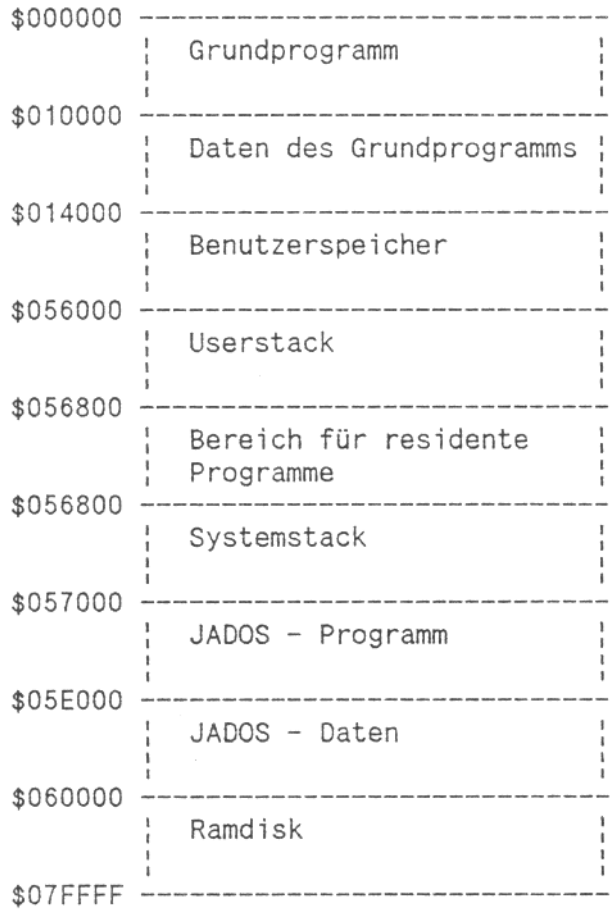
Vor dem Systemstack liegt der Bereich für die residenten Programme, die während der Laufzeit angelegt werden dürfen. Sind keine Programme resident, dann beansprucht der Bereich keinen Platz. Pro residentem Programm wird die Dateigröße des Programms plus 1 KByte angesetzt. Um diesen Betrag vergrößert sich dann der Bereich der residenten Programme.

Vor den residenten Programmen liegt der Userstack. Alle Programme und auch die internen Kommandos benutzen als Stack den Userstack. Die Lage des Userstacks verschiebt sich während der Laufzeit. Wurde ein Programm resident gemacht, dann verschiebt sich der Userstack um den Platz für das residente Programm. Wird ein residentes Programm wieder entfernt, dann verschiebt sich der Userstack wieder um den gewonnenen Platz nach hinten. Der Userstackpointer kennzeichnet das Ende des Benutzerspeichers. Diese Grenze wird von allen Funktionen, die in den Speicher schreiben, abgefragt, damit die dahinterliegenden Bereiche nicht zerstört werden.

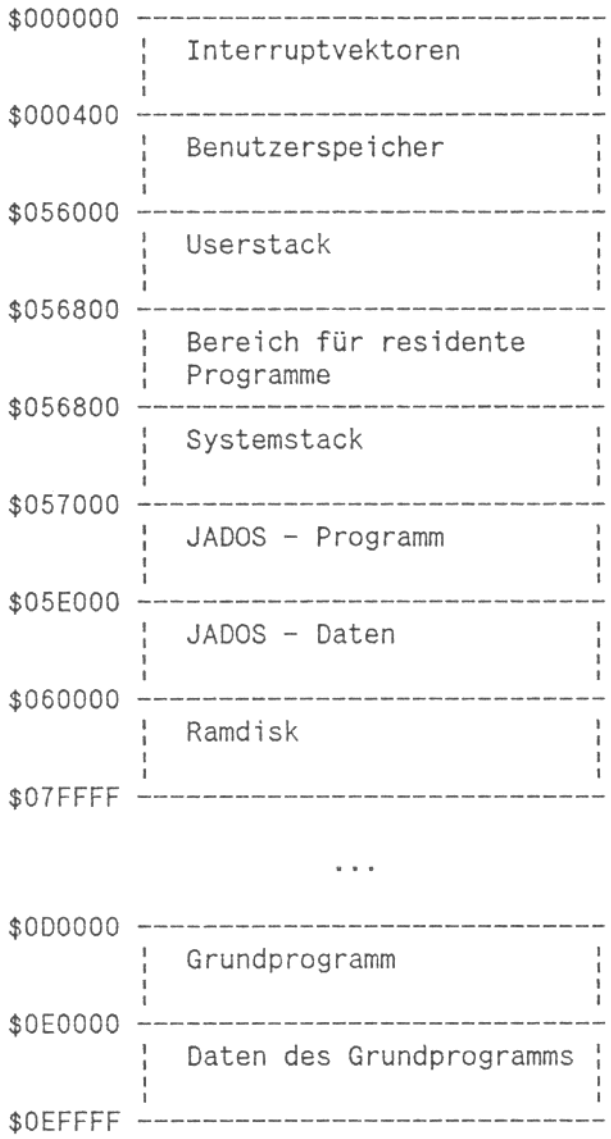
Der Benutzerspeicher steht dem Anwender für seine Programme zur Verfügung. Der Beginn des Benutzerspeichers ist gleichzeitig die Ablaufadresse für die Programme des Typs 68K. Programme des Typs COM dagegen werden von JADOS soweit wie möglich nach hinten geladen. Dazu verschiebt JADOS temporär, also nur während der Laufzeit des COM-Programms, den Userstack um den Platzbedarf des Programms nach vorne. Nach Beendigung des Programms wird der Stack wieder nach hinten verschoben. Auch die Kommandodateien des Typs BAT werden so behandelt.

Im folgenden werden die einzelnen Speicherteile anhand von Beispielen in einem Diagramm dargestellt, um das oben erklärte zu veranschaulichen.

Beispiel 1: Grundprogramm 6.2 ab Adresse 0; 448 KByte Ram



Beispiel 2: Grundprogramm 6.2 im Eprom ab Adresse $0D0000; 512 KByte Ram



Beispiel 3: Grundprogramm 6.2 im Eprom ab Adresse $0D0000; 512 KByte Ram ge{ndertes Grundprogramm im Ram

HINWEISE

Im Zusammenhang mit der Entwicklung von Programmen werden noch einige Hinweise betreffs der Speicherverwaltung gegeben.

Das Ende des Benutzerspeichers wird durch den Stackpointer markiert. Die Funktionen zum Laden von Dateien oder Teilen einer Datei prüfen, ob sie auch nicht das Ende des Benutzerspeichers überschreiben. Es wird dann gegebenenfalls eine Meldung angezeigt, wie z.B.

Speicher voll

oder ein Returncode für Programme generiert, der dies anzeigt. Manchmal kann die Prüfung aber lästig sein, z.B. wenn ein COM-Programm Platz für Daten gelassen hat und dorthin Teile einer Datei laden will. Im Normalfall geht das nicht. Es stehen aber Traps zur Verfügung, die die Prüfung abschalten bzw. wieder aktivieren.

Da ein COM-Programm noch hinter dem Userstack liegt, darf es Platz für Daten nicht einfach hinter dem Programmcode anlegen, da ansonsten Teile des Betriebssystem zerstört werden können. Es gibt deshalb nur zwei Wege, um Datenspeicher anzulegen. Der erste Weg besteht darin, hinter dem Programmcode mit den Assembleranweisungen "ds.b" Platz zu reservieren. Dies vergrößert aber den Platzbedarf der COM-Datei auf dem Massenspeicher und erhöht die Zeitdauer des Ladens. Der zweite Weg besteht darin, nach dem Programmstart Platz auf dem Userstack anzulegen. Vor dem Programmende muß der Platz aber wieder freigegeben werden.

Beim Entfernen residenter Programme werden die verbleibenden residenten Programme so zusammengeschoben, daß sie nur einen zusammenhängenden Bereich beanspruchen. Es enstehen dabei keine ungenutzten Lücken.

Beim Assemblieren legt das Grundprogramm eine Symboltabelle an. Diese Tabelle wächst nach hinten. Falls das Grundprogramm auf Adresse 0 liegt, kann es zu einer Kollision zwischen der Symboltabelle und dem Benutzerspeicher kommen. In diesem Fall dürfen die Programme nur soviele Symbole enthalten, daß die Symboltabelle nicht in den Benutzerspeicher hineinragen kann. Es wird dann empfohlen, das Grundprogramm auf eine andere Adresse zu legen, lokalen Speicher von 64 KByte vorzusehen und eventuell eine Bootkarte einzusetzen.

DATEIVERWALTUNG

Die Massenspeicher

JADOS unterstützt drei Arten von Massenspeichern, nämlich bis zu vier Diskettenlaufwerke, eine SCSI-Festplatte und eine Ramdisk. Alle Massenspeicher werden gleich verwaltet. Lediglich die kapazitätsbestimmenden Parameter unterscheiden sich. Im folgenden werden die Parameter vorgestellt.

Die Ramdisk

Die Ramdisk wird im Urlader eingerichtet. Die Größe richtet sich nach dem Speicherausbau und liegt zwischen minimal 20 KByte und maximal 1280 KByte. Jeder Sektor der Ramdisk umfaßt 1024 Bytes. Die Parameter sind:

```
Zahl der Oberflächen.....: 1
Zahl der Zylinder........: abhängig von der Größe
Zahl der Sektoren je Spur.....: 5
Dichtecode...............: 0
Nummer der Verwaltungsspur....: 0
Freie Kapazität..........: abhängig von der Größe
Index des letzten FAT-Eintrags: abhängig von der Größe
Zahl der freien Spuren.....: anhängig von der Größe
Index des ersten FAT-Eintrags.: 4
```

Die von der Größe abhängigen Parameter werden wie folgt berechnet. Von der Größe der Ramdisk in Bytes werden 1600 Bytes subtrahiert und dann durch 1024 dividiert. Das Ergebnis ist gleich der Zahl der Sektoren insgesamt. Diese Zahl wird durch die Zahl der Sektoren je Spur dividiert und ergibt die Zahl der Zylinder. Diese Zahl mit 4 multipliziert ergibt den Index des letzten FAT-Eintrags. Die Zahl der Zylinder minus 1 ergibt die Zahl der freien Spuren. Diese Zahl multipliziert mit der Zahl der Sektoren je Spur ergibt die freie Kapazität.

Beispiel für eine Ramdisk von 128 KByte.

Die Zahl der Bytes beträgt 131072. Davon 1600 abgezogen und dann durch 1024 dividiert ergibt 126 Sektoren. Bei 5 Sektoren je Spur ergeben sich 25 Zylinder und 24 freie Spuren. Der Index des letzten FAT-Eintrags beträgt 100. Die freie Kapazität liegt bei 120 KByte.

Auf der ersten Spur der Ramdisk werden im ersten Sektor die Spurtabelle (FAT) und auf den folgenden vier Sektoren das Inhaltsverzeichnis verwaltet. Es sind 128 Dateien auf der Ramdisk möglich.

Die Diskettenlaufwerke

JADOS unterstützt bis zu vier Diskettenlaufwerke im Standardformat:

- doppelseitig
- doppelte Dichte
- Miniformat 5,25" oder 3,5"
- 80 Zylinder
- 5 Sektoren je Spur
- 1024 Bytes je Sektor

Die Parameter sind:

Zahl der Oberflächen.....:	2
Zahl der Zylinder.....:	80
Zahl der Sektoren je Spur.....:	5
Dichtecode.....:	$20
Nummer der Verwaltungsspur....:	3
Freie Kapazität.....:	780
Index des letzten FAT-Eintrags:	640
Zahl der freien Spuren.....:	156
Index des ersten FAT-Eintrags.:	16

Es ist möglich, diese Parameter im Patchfeld von JADOS zu ändern. Hierbei ist aber größtmögliche Sorgfalt zu beachten. In jedem Fall muß die Zahl der Bytes je Sektor 1024 betragen. Im Interesse einer uneingeschränkten Kompatibilität beim Datenträgeraustausch sollte man jedoch auf Änderungen möglichst verzichten.

Auf den ersten drei Spuren einer Diskette ist Platz für den Urlader vorgesehen. Auf der vierten Spur werden im ersten Sektor die Spurtabelle (FAT) und auf den folgenden vier Sektoren das Inhaltsverzeichnis verwaltet. Es sind 128 Dateien je Diskette möglich.

Die Festplatte

JADOS unterstützt eine SCSI-Festplatte auf Adresse 1. Diese Platten werden nur über logische Blöcke angesprochen. Die Umrechnung in Oberfläche, Zylinder und Sektor wird vom eingebauten SCSI-Controller durchgeführt. JADOS verlangt, daß die Platte im Format 1024 Bytes je Sektor formatiert ist. Ein anderes Format wird nicht akzeptiert. Da Festplatten im Verhältnis zu Diskettenlaufwerken eine sehr hohe Kapazität aufweisen, unterteilt JADOS die Gesamtkapazität der Platte in Partitionen. Jede Partition hat dabei folgende Parameter:

```
Zahl der Oberflächen.....:  1
Zahl der Zylinder........:  256
Zahl der Sektoren je Spur.....:  10
Dichtecode...............:  0
Nummer der Verwaltungsspur....:  0
Freie Kapazität..........:  2550
Index des letzten FAT-Eintrags:  1024
Zahl der freien Spuren.....:  255
Index des ersten FAT-Eintrags.:  4
```

Die ersten 100 Blöcke der Platte sind für den Urlader und zukünftige Erweiterungen reserviert. Es folgen dann je Partition 2560 Blöcke. Die Zahl der Partitionen hängt nur von der Gesamtkapazität der Festplatte ab und wird von JADOS automatisch eingerichtet. Es sind aber nicht mehr als 26 Partitionen möglich. Typische Festplatten und die Zahl der Partitionen sind:

```
Seagate ST125/ST225:  20 MByte netto      =  8 Partitionen (A..H)
Seagate ST138/ST238:  30 MByte netto      = 12 Partitionen (A..L)
Seagate ST151/ST251:  40 MByte netto      = 16 Partitionen (A..P)
Seagate ST178/ST278:  65 MByte netto      = 26 Partitionen (A..Z)
```

Je Partition wird auf dem ersten Sektor der ersten Spur die Spurtabelle (FAT) und den folgenden neun Sektoren das Inhaltsverzeichnis verwaltet. Es sind daher 288 Dateien je Partition möglich.

Das Inhaltsverzeichnis

Auf jedem Massenspeicher wird ein Verzeichnis der Dateien geführt, das sogenannte Inhaltsverzeichnis oder auf englisch directory. Je Datei wird ein Eintrag verwaltet. Das Format ist:

Bytenummer	Inhalt	Bedeutung
00..01	Kennung	$E5E5 = freier Eintrag $FFFF = gelöschter Eintrag 0 = gültiger Eintrag
02..09	Dateiname	Name der Datei
10..11	reserviert	mit 0 belegt
12..14	Dateityp	Typ der Datei
15	reserviert	mit 0 belegt
16..17	Startspur	Nummer der ersten Spur der Datei. Die folgenden Spuren werden aus der Spurtabelle entnommen
18..19	Endesektor	Nummer des letzten gültigen Sektors auf der Spur, die zu der Datei gehört
20..21	Endebyte	Nummer des letzten gültigen Bytes des Endesektors. Ist immer gleich 0, da nicht geführt
22..25	Datum	Datum des letzten schreibenden Zugriffs auf die Datei. Das erste Byte enthält eine Null, die folgenden Bytes enthalten Jahr, Monat und Tag in BCD codiert
26..27	Dateilänge	Anzahl der Sektoren, die zu der Datei gehören
28	Dateimodus	$E5 = zum schreiben und lesen freigegeben $E4 = nur zum lesen freigegeben
29..31	reserviert	

Die Spurtabelle

Bei sehr einfachen Betriebssystemen werden Dateien fortlaufend Spur für Spur abgespeichert. Dies ermöglicht zwar ein schnelles Laden und Speichern der Dateien, hat aber einen wesentlichen Nachteil. Wenn eine schon vorhandene Datei vergrößert werden soll, so paßt die Ergänzung häufig nicht mehr auf den der Datei zugewiesenen Bereich. Für die gesamte Datei (einschließlich Erweiterung) muß in diesem Fall ein Bereich gesucht werden, der ausreichend groß ist. Meistens liegt ein solcher Bereich hinter der letzten Datei. Der der ursprünglichen Datei (vor der Erweiterung) zugewiesene Bereich wird in der Regel gelöscht und steht einer neuen Datei zur Verfügung. Da Dateien nun aber immer nur fortlaufend, d.h. an einem Stück gespeichert werden können, darf diese neue Datei auf keinen Fall größer als der frei gewordene Bereich sein. Da es in der Praxis aber nicht immer möglich ist, eine Datei "herbeizuzaubern", die genau in die frei gewordene Lücke paßt oder nur wenig kleiner ist, kann es häufig vorkommen, daß ein gelöschter Bereich von einer sehr viel kleineren Datei belegt wird oder überhaupt nicht mehr belegt wird, da man zufällig nur noch größere Dateien zu speichern hat. Dabei geht dann sehr viel Speicherkapazität verloren, was eine recht unökonomische Speicherplatznutzung darstellt. In ungünstigen Fällen (nach vielen Löschvorgängen) paßt keine Datei mehr auf den Massenspeicher, obgleich vielleicht nur wenige Prozent der Kapazität augenutzt sind.

JADOS verwaltet die Massenspeicherkapazität wesentlich effizienter. Jede Datei belegt eine oder mehrere komplette Spuren. Umfangreichere Dateien müssen dabei nicht unbedingt auf aufeinanderfolgenden Spuren untergebracht sein. Soll z.B. eine Datei vergrößert werden, so sucht JADOS automatisch nach freien Bereichen (z.B. einen durch Löschen einer Datei freigeworden Bereich) und speichert die Dateiergänzung darauf ab. Auf diese Weise können alle auf dem Massenspeicher vorhandenen Spuren auch wirklich ausgenutzt werden. Im Inhaltsverzeichnis ist eingetragen, auf welcher Spur eine Datei beginnt. Da größere Dateien an verschiedenen Stellen auf dem Massenspeicher gespeichert sein können, wird in der sogenannten Spurtabelle festgehalten, wo es weitergeht. Und zwar werden hier für jede Spur einer Datei jeweils die Nachfolgerspur (also in Richtung auf das Dateiende) und die Vorgängerspur (also in Richtung auf den Dateianfang) eingetragen. Die Spurtabelle ist folgendermaßen aufgebaut:

Bytenummer	Inhalt	Bedeutung
00..01	Vorgängerspur	$E5E5 = freie Spur
		$FFFF = kein Vorgänger
		0..255 = Nummer der Vorgängerspur
02..03	Nachfolgerspur	$E5E5 = freie Spur
		$FFFF = kein Nachfolger
		0..255 = Nummer der Nachfolgerspur

Der Vorteil dieses Verfahrens liegt darin, daß vorhandene Dateien problemlos wachsen können und daß die durch Löschen einer Datei frei gewordenen Spuren wieder beliebig zur Verfügung stehen. Ein kleiner Nachteil besteht darin, daß das Laden und Speichern etwas länger dauern kann.

Der Dateisteuerblock

Jede Datei wird von einem Dateisteuerblock, im folgenden mit FCB (engl. file control block) abgekürzt, verwaltet. Es ist dies ein Datensatz, der beim Öffnen oder Anlegen einer Datei gebildet wird, und beim Schließen der Datei wieder unwirksam wird. Der Datensatz ist folgendermaßen aufgebaut:

Bytenummer	Inhalt	Bedeutung
00..01	Laufwerk	Nummer des Laufwerks, auf dem sich die Datei befindet. Es gilt: 0 = Ramdisk 01..04 = Diskette 1 bis 4 05..30 = Festplattenpartition A bis Z
02..09	Dateiname	Name der Datei
10..11	reserviert	mit 0 belegt
12..14	Dateityp	Typ der Datei
15	reserviert	mit 0 belegt
16..17	Startspur	Nummer der ersten Spur der Datei. Die folgenden Spuren werden aus der Spurtabelle entnommen
18..19	Endesektor	Nummer des letzten gültigen Sektors auf der Spur, die zu der Datei gehört
20..21	Endebyte	Nummer des letzten gültigen Bytes des Endesektors. Ist immer gleich 0, da nicht geführt
22..25	Datum	Datum des letzten schreibenden Zugriffs auf die Datei. Das erste Byte enthält eine Null, die folgenden Bytes enthalten Jahr, Monat und Tag in BCD codiert
26..27	Dateilänge	Anzahl der Sektoren, die zu der Datei gehören
28	Dateimodus	$E5 = zum schreiben und lesen freigegeben $E4 = nur zum lesen freigegeben
29..31	reserviert	
32..33	Dir-Sektor	Sektor des Inhaltsverzeichnisses
34..35	Dir-Byte	Startbyte im Sektor des Inhaltsverzeichnisses
36..37	Dateistatus	Statusinformation 0 = geschlossen 1 = geöffnet, kein Directory-Update 2 = geöffnet, mit Directory-Update
38..39	Akt-Spur	Nummer der aktuellen Spur

40..41	Akt-Sektor	Nummer des aktuellen Dateisektors der aktuellen Spur
42..43	Last-Spur	Nummer der letzten Dateispur
44..47	Adresse	Aktuelle Speichertransferadresse

INHALTSVERZEICHNIS

Allgemeines 2

Speicherverwaltung 2

Dateiverwaltung 7

Die Massenspeicher 7

Die Ramdisk 7
Die Diskettenlaufwerke 8
Die Festplatte 9

Das Inhaltsverzeichnis 10

Die Spurtabelle 11

Der Dateisteuerblock 12