default

COLLABORATORS			
	TITLE : default		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		August 22, 2024	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

default

1.1 BootMain

```
**********************************
*                                *
*   BootMonitor-Dokumentation    *
*                                *
**********************************
```


1.2 copyright

Der Bootmonitor ist Freeware, daß heißt, er darf frei kopiert werden jedoch ist es nicht gestatted den Bootmonitor zu kommerziellen Zwecken, in irgendeiner Art und Weise verändert oder unvollständig weitergegeben werden. Es dürfen sich jedoch zu diesem Paket jederzeit neue Icone zu den originalen Iconen mitgegeben werden.
Die originalen Iconnamen dürfen dann jedoch nicht mit .image sonder müssen mit .original enden.

Die Version 1.3 des BootMonitor’s ist jedoch GIFTWARE !!

Die Version 1.4 des BootMonitor‘s ist Shareware und Update-Fähig.

Die Version 1.5 des BootMonitor's ist Shareware (siehe auch Zukunft) !!

1.3 paket

Zu diesem Paket zählt:

```
  Archive.image         : internes Icon
  BootMon               : Der BootMonitor selbst
  BootMon.info          : Das Icon für den BootMonitor
  BootMon.dok           : Die Dokumentation des BootMons
  BootMon.guide         : Die Guide-Datei des BootMons
  Clear.image           : internes Icon
  Device.image          : internes Icon
  Edit.image            : internes Icon
  Execute.image         : internes Icon
  Exit.image            : internes Icon
  Info.image            : internes Icon
  Read.image            : internes Icon
  Reset.image           : internes Icon
  Show.image            : internes Icon
  UnArchive.image       : internes Icon
  Undo.image            : internes Icon
  Vektor.image          : internes Icon
  Write.image           : internes Icon
```

1.4 ownicon

Es ist natürlich erlaubt eigene Icons zu malen und sie einem Paket beizugeben.
Die Vorhandenen Icone dürfen dabei jedoch nicht gelöscht werden, alternativ dazu darf man sie mit einem neuen Suffix umbenennen,

z.B.: Archive.image --> Archive.original !!

Die eigenen Icone müssen dann den alten Namen ersetzen, damit der BootMonitor diese Icone auch findet, denn die Namen der Icone sind im BootMonitor fix vorgegeben, andernfalls findet er sie nicht und werden in einem solchen Fall auch nicht angezeigt.
Außerdem müssen sich die Icon-Dateien auf dem Starthauptverzeichnis befinden, sonst werden sie ebenfalls nicht gefunden, außer der BootMonitor wird vom CLI aus gestartet.

>>>>> Achtung <<<<<, Icone die nicht gefunden werden, belegen ebenfalls Speicher, und zwar genausoviel als ob sie vorhanden wären.

Zur Sicherheit gibt der BootMonitor unter jedem Icon noch einen IconText aus, der ebenfalls fest vorgegeben ist und ist mehr oder weniger dafür da, falls sich Icone nicht einlesen lassen, daß der User noch immer den Überblick über die Gadgets hat, über die der BootMonitor hauptsächlich gesteuert wird.

Das Anlegen eigener Icone funktioniert so:

Man nimmt ein Zeichenprogramm, egal welches, und malt sich seine Bildchen.
Dabei ist zu beachten, daß das Icon folgende Größen nicht überschreitet:

```
  Breite = 60  Pixel im HighRes-Format
  Höhe   = 18  Zeilen im HighRes-Format
  Planes = 3   (entspricht 4 Farben)
```

ansonsten werden die Icone nur teilweise oder sogar falsch angezeigt.

Ist dies abgeschlossen, nimmt man einen IFF-Konverter, welcher auch die
Grafiken im rohen Bitmap-Format abspeichern kann.
Dafüer eignet sich z.b. der Image-Maker.
Es darf dabei keine ColorMap (Farbtabelle) mit abgespeichert werden und bei
einer Frage von eventuellen Offsets ist diese mit 0 zu beantworten.
Solche Graphiken sind dann etwa 300 Byte lang.

Sollte ein Icon zu lange ausfallen, z.b.: 20 Zeilen Höhe, dann nimmt der
BootMonitor diese Grafik ebenfalls an, schneidet jedoch diese Grafik dann
ab der 19.en Zeile ab.

Nun benennt man das Original-Icon mit einem neuen Suffix.
Vom Cli aus funktioniert das in etwa so:

```
  rename df0:Archive.image to df0:Archive.original
```

Nun steht dem Abspeichern des eigenen Icons nichts mehr im Wege, man muß
nur noch die eigene Rohgrafik mit dem Namen wie im Beispiel mit
Archive.image abspeichern, den BootMonitor starten und schon hat man ein
neues Archive-Gadget.

1.5 allgemein

Zum BootMonitor selbst:

Der BootMonitor wurde 100% in Assembler geschrieben und sollte keine Bugs
mehr beinhalten (aber was weiß man schon, wo der Fehlerteufel zuschlägt).
Der Author übernimmt mit diesem Programm keine Haftung an eventuell zer-
störten Bootblocken oder Datenträgern.
In der Version 1.0 liest der BootMonitor nur Bootblocke von der Diskette,
auf welche mit dem Trackdisk-Device zugegriffen werden kann.
Er unterstützt hier die Gerätenummern 0 bis 3 dies entspricht den Lauf-
werken DF0: bis DF3:.
Besitzer der Kickstart 1.3 oder niedriger müssen sich nach einer geeigneten
ASL-Library umsehen, welche auch für ihre Kickstart lauffähig ist.
Besitzer der Kickstart 2.0 oder höher sollten mit einer ASL-Library
normalerweise keine Probleme mit dem BootMonitor haben.
Die Steuerung kann wahlweise über die Gadgets oder über Tasten erfolgen,
außer er befindet sich im Editiermodus, wo nur Tasten angenommen werden.

1.6 gadget

Zu den Gadgets:

- The End
- Read BB
- Write BB
- Archive BB
- UnArch. BB
- Laufwerk
- Execute
- Edit BB
- Undo BB
- Show BB
- Clear
- Info
- Vektor
- Reset

1.7 gend

HotKey 'X'

beendet ohne Rückfrage den Bootmonitor.
Vorsicht eventuell editierte Bootblocke, welche noch in der ursprünglichen Form im Undo-Buffer standen und modifiziert zurückgeschrieben wurden sind hiermit unwiederruflich verloren, deshalb einen kleinen Tip:
Bevor Bootblöcke editiert werden, immer eine Kopie mit 'Arch. BB' anlegen.
Der BootMonitor läßt sich ebenfalls mit dem CloseGadget des Windows beenden.

1.8 read

HotKey 'R'

Steht für Read BootBlock und
liest einen Bootblock von der Diskette in den Speicher. Hierbei bleibt der Undo-Buffer unberührt und es können eventuell überschriebene Bootblöcke, welche schon im Undo-Buffer stehen wieder restauriert werden.
Bei nichterfolg gibt der BootMonitor eine Fehlermeldung aus.

1.9 write

HotKey 'W'

Steht für Write BootBlock und
schreibt einen Bootblock auf die Diskette zurück, jedoch nicht den Undo-Buffer. Vorsicht, hier ein eventuell vorhandener Bootblock auf der Diskette gnadenlos überschrieben.
Auch hier gibt der BootMonitor bei nichterfolg eine Fehlermeldung aus.

Eine Besonderheit von 'Write BB' ist, daß diese Funktion die richtige
Checksumme des Bootblockes, bevor der Bootblock noch geschrieben wird,
berechnet wird und dann erst der Schreibvorgang eingeleitet wird.
Hilfreich, wenn man Selbstgeschrieben Bootblöcke hat, jedoch nicht weiß,
wie man jene auf den Bootblock bringen (installieren) soll.
Eine Struktur des Bootblockes wird noch beschrieben.

1.10 archive

HotKey 'A'

Steht für Archive BootBlock und
gibt einen Filerequester aus und fragt nach dem genauen Pfadnamen und dem
Filenamen, unter welchem der BootBlock, welcher sich im Speicher befindet,
als Datei abgelegt werden soll.
Diese Files haben kein bestimmtes Format und beinhalten nur die Daten,
welche der Bootblock trägt.
Solche Files sind immer 1024 Byte oder exakt 1 KByte groß.

1.11 unarchive

HotKey 'U'

Steht für UnArchive BootBlock
gibt ebenfalls einen Filerequester aus. Funktioniert ansonsten genauso wie
das Archivieren nur in der Umgekehrten Reihenfolge, also wird hier kein Block
geschrieben sonder eine Datei eingelesen (Der Undo-Buffer bleibt bei dieser
Aktion unberührt.)

1.12 laufwerk

HotKey 'L'

versucht das Laufwerk, z.b. von DF0: nach DF1: wechseln. Dabei erhöht jeder
Aufruf von Laufwerk die Gerätenumer um 1. Ist das Laufwerk DF3: erreicht,
fängt diese Routine wieder bei (DF)0 an.
Bei Problemen werden hier ebenfalls Fehlermeldungen ausgegeben.
Diese Routine eröffnet KEINEN Schreib- oder Lesezugriff auf irgendwelche
Daten oder Blöcke des Laufwerks.

1.13 execute

HotKey 'T'

Steht nicht für das exekutieren des Bootblockes, sondern führt einen
Bootblock, falls ein gültiger Bootblock im Speicher steht, aus.
Der Unterschied zum normalen Booten einer Diskette besteht hierbei, daß

man

1. Den Bootblock direkt von der Workbench aus starten kann.

2. Die Checksumme des Bootblockes ignoriert wird und es somit auch möglich
   ist, eventuelle Bootblöcke, welche eine zerstörte oder fehlerhafte
   Checksumme beinhalten, ebenfalls wieder gestartet werden können.

Dabei werden die Daten für den Bootblock, soweit mir bekannt ist, die Exec-
base im Register A6 und der IORequest im Register a1 abgelegt.
Sollten jemandem bekannt sein, das noch zusätzliche Register beim Booten von
einer Diskette vorhanden sind, so soll er mich einfach kontaktieren.
Es läßt sich kein ungültiger oder nicht vorhandener Bootblock starten, denn
es wird bevor der Bootblock gestartet wird noch geprüft, ob das erste Code-
Langwort überhaupt Daten enthält und dann erst losgelegt.
Dies ist ebenfalls eine kleine Sicherheitsvorkehrung des BootMonitors.
Übrigens der HotKey 'T' leitet sich hier von Trace (Verfolgen,Spur) ab.

1.14 edit

HotKey 'E'

Beginnt mit dem Edieren des Bootblockes, dabei wird ein Cursor ab dem
ersten Byte des Bootblockes gesetzt. Dieses Gadget versetzt den Boot-
Monitor in den EditierModus, wo er nur noch begrenzt steuerbar ist.
Der Cursor wird mit folgenden Tasten bewegt:

```
  <CTRL A>    --- Bewegt den Cursor nach Links
  <CTRL S>    --- Bewegt den Cursor nach Rechts
  <CTRL W>    --- Bewegt den Cursor nach Hinauf
  <CTRL Y>    --- Bewegt den Cursor nach Hinunter
  <CTRL Z>    --- Bewegt den Cursor nach Hinunter
  <ENTER>     --- Beendet den Editiermodus
```

Das CTRL steht hier für die Controll-Taste und die muß gleichzeitig mit
der Steuertaste gedrückt sein, um den Cursor zu positionieren.
Jede andere Taste, insofern sie von VANILLAKEY erfaßt wird, modifiziert
den Bootblock, dazu zählen mindestens die ASCII-Tasten.
Ein Versuch den Hauptfenster des BootMonitors zu schließen bewirkt hier
keinen Ausstieg aus dem Programm sondern beendet hier nur den Editiermodus.
Besonderheiten des Editiermoduses sind zum einen, daß bei Aufruf des Editier-
moduses der aktuelle Bootblock sofort in den Undo-Buffer gerettet wird.
Ab hier wird der Undo-Buffer als besetzt (nicht frei) angezeigt.
Siehe dazu das INFO-Gadget.
Zum anderen wird hier ein zweites kleineres Fenster geöffnet, welches einem
sofort anzeigt, daß sich der BootMonitor im Editiermodus befindet und einige
hilfreiche Features (Merkmale) aufzeigt.

Jeder Editiervorgang, welche keine Cursorsteuerung oder aussteigen aus dem
Editormodus anstreben, werden ab sofort in weiß dargestellt.
Ein Überfahren solcher Änderungen mit dem Cursor bewirkt, das diese Werte
wieder in schwarz dargestellt werden.
Es existiert im Editiermodus keine Del- oder Backspacetaste, denn dafür
läßt sich der Cursor bei Schreibfehler mit den Cursorsteuertasten des
Editors wieder zurücksetzen und der Editiervorgang kann wiederholt werden.

Nachteil des Editors ist, das hier die Tasten Ctrl_A, Ctrl_S, Ctrl_W, Ctrl_Y,
Ctrl_Z und die Enter-Taste belegt sind und keine Zeichenänderung bewirken.
Der Grund für diese Belegung war dieser, das auch Amiga-User, welche einen
Amiga OHNE Zehnerblock besitzen, wie den A600, ebenfalls in den Genuß
des Editierens kommen.

1.15 calcwin

Dieses Fenster nimmt KEINE Eingaben an, und dient nur der Information.

Dabei bedeutet:

Bytewert:	Stellt die Überschrift dar.
Hex:	Hier wird das Byte, wo sich der Cursor befindet im Hexadezimalen Zahlenformat ausgegeben.
Dez:	Hier wird das Byte, wo sich der Cursor befindet im Dezimalen Zahlenformat ausgegeben.
Position:	Stellt die zweite Überschrift dar.
Hex:	Hier wird die aktuelle Position im Bootblock, ab der der Cursor steht im Hexadezimalen Zahlenformat angezeigt.
Dez:	Hier wird die aktuelle Position im Bootblock, ab der der Cursor steht im Dezimalen Zahlenformat angezeigt.

Zum Editieren bitte immer das Hauptfenster des BootMonitors aktivieren, was
aber nicht bedeutet, daß man das Calculator-Fenster nicht in den Hintergrund
legen kann oder darf.

1.16 undo

HotKey ‘N‘
Kommt vom N im Wort U(n)do, da die U-Taste bereits belegt ist (UnArchive).
Mit Undo-BootBlock wird die letzte und nur die letzte Editieraktion
wieder Rückgängig gemacht. Im Klartext der Editierte Bootblock geht hier
wieder verloren und wird durch den im Buffer stehenden Bootblock voll-
ständig Überschrieben.

1.17 show

HotKey ‘S‘

Kommt vom S im Wort (S)how.
Mit dieser Funktion kann der aktuelle BootBlock, und nicht der Undo-Buffer
wieder neu angezeigt werden, da Fehlermeldungen und andere Aktionen das
selbe Ausgabefenster benutzen.

1.18 clear

HotKey 'C'

Kommt vom C im Wort (C)lear.
Mit dieser Funktion wird der aktuelle Bootblock, welcher im Speicher steht wieder gelöscht. Der gelöschte Bootblock wird sofort wieder angezeigt. Der Undo-Buffer bleibt bei dieser Aktion wieder Unberührt.
Aber Achtung, ein zurückschreiben eines solchen BootBlockes kann das AmigaDOS dazu veranlassen, die Diskette als NichtDosDiskette zu betrachten. Abhilfe hierbei wäre das installieren eines AmigaDOS-Bootblockes, egal ob installiert oder nicht installiert oder mit dem Editor wieder die DOS-Kennung in den Bootblock zu schreiben und auf die Diskette zurückzuschreiben.
Dazu geht man wie folgt vor:

Man geht mit dem BootMonitor in den Editier-Modus und schreibt ab dort ab der Positon 0 in GROßBUCHSTABEN wieder das Wort 'DOS' hinein und drückt die Enter-Taste.

Danach schreibt man sich den BootBlock mit 'Write BB' wieder auf die Disk zurück

Ein anderer Weg wäre nach dem Schreibzugriff SOFORT die Funktion 'Undo BB' aufrufen und den BootBlock nochmals schreiben.

Der nächste Weg ginge so:

Man ruft die Funktion 'UnArch BB' auf, liest sich einen Bootblock von der Diskette oder sonst von wo ein und schreibt diesen auf die Diskette.

1.19 info

HotKey 'I'

Steht für (I)nformation übers Laufwerk.
Diese Funktion dient als Hilfsfunktion, wenn man mit öfterem Wechseln des Laufwerkes nicht mehr genau weiß, welches Laufwerk nun den wirklich angesprochen wird.
Dabei wird der Device-Name ausgegeben, normalerweiße das 'trackdisk.device'.
Die Gerätenummer, welches Device nun angesprochen wird, z.b.:

Gerätenummer 1 steht hier für df1:
Gerätenummer 0 steht hier für df0:

Existiert hier das Laufwerk oder liegt hier überhaupt eine AmigaDOS-Diskette im Laufwerk (wird angezeigt mit gültig od. nicht gültig).

Ebenfalls wird hier der Status des Undo-Buffers angezeigt (ein leer sagt einem, das keine Undo-Funktion aufgerufen werden kann und dient dazu mehr oder weniger als Schutzfunktion vor versehntlichem Überschreiben eines im Speicher stehenden Bootblocks.)
Achtung: Schon ein einziger Aufruf des Editors füllt den Undo-Buffer und dies bleibt er auch bis zum Ausstieg aus dem BootMonitors und Neustart.

1.20 vektor

HotKey 'V'

Steht für (V)ektorenanzeige.
Diese Funktion dient ebenfalls als kleine Schutzfunktion wie eventuell gestartete BootViren.Die meisten Viren verbiegen im Amiga bei ihrem Start (wie z.b. mit 'Execute' durchaus möglich) die sogenannten Captures.
Weist einer dieser Vektoren einen anderen Wert als 0 auf so wird der Wert hier im Hexadezimalem Zahlensystem und in weiß dargestellt. Ist dies der Fall, so heißt es, alle Disketten-, oder auch andere Datenträger können ebenfalls davon betroffen sein,zugriffe mit höchster Vorsicht zu behandeln.
Der BootMonitor ist nicht in der Lage solche Vektoren zu setzen, geschweige erst sie zu löschen.
Einzig allein in der Lage dazu wären hier mit dem BootMonitor gestartete unbekannte BootBlöcke, welhalb der Autor dieses Programmes auch keine Garantie und Haftung für etwaige Schäden an Disketten oder anderen Datenträgern übernimmt, ebensowenig eine Garantie.
Einziger Schutz hier wäre, einen sogenannten Viruskiller, wie z.b. den VirusZ von Georg Hörmann oder ähnliches.

1.21 reset

HotKey 'O'

Steht für (O)ld oder Neu.
Diese Funktion setzt weder den BootMonitor noch den Amiga zurück, sondern einzig allein das aktuell angesprochene Device oder Gerät.
Es wird hier lediglich das Device-Commando CMD_RESET ausgeführt und dient allein dazu, Laufwerke oder ähnliches wieder in den Einschaltzustand zu versetzen (warum auch immer, ich weiß es nicht warum).

1.22 fprint

HotKey 'F'

Steht für (F)ile-Print.
Diese Funktion gibt einen ASCII-Dump des Bootblockes in ein File aus.
Es erscheint hier jedoch ein Asl-Requester, in welchem der komplette Pfad + Dateinamen einzugeben ist, wohin danach der Dump geschrieben wird.
Ansonsten ist diese Funktion gleich wie Print.

1.23 fdump

HotKey 'H'

Steht für (H)ex-Dump in eine Datei
Es erscheint hier jedoch ein Asl-Requester, in welchem der komplette Pfad + Dateinamen einzugeben ist, wohin danach der Dump geschrieben wird.

Diese Funktion gibt einen genauen Hex-Dump des Bootblockes in ein File aus.

Ansonsten ist diese Funktion gleich wie Dump.

1.24 print

HotKey 'P'
Steht für (P)rint ASCII auf Drucker.
Diese Funktion gibt einen ASCII-Dump des Bootblockes auf den Drucker aus.
Im Unterschied zu FPrint erscheint hier kein Asl-Requester,
sondern es wird hier der ASCII-Dump direkt über das Gerät PRT: auf
den Drucker gelenkt.

Der ASCII-Dump besteht wie folgt aus

OFFSET : ------- 65 ASCII-Zeichen -------

0000 : DOS.I^ü®ð-..©r^{3}¾..^{3}.5...dos.library....

Dabei besteht der ASCII-Dump aus nur druckbaren Zeichen, weiters werden Bytes,
welche kleiner als $20 sind herausgefiltert und durch '.' ersetzt.
Dies hat den Grund, da unter diesem Wert unter anderem Steuerzeichen für
den Drucker liegen und auch dann als solche interpretiert, was zur folge
hat, das dabei das Druckbild sich fatal verändern kann.

Vorsicht jedoch, wenn kein Drucker eingeschalten oder angeschlossen ist !!
In diesem Fall ist der BootMonitor in der Lage das File zu öffnen und die
Daten dorthin zu schicken, er erkennt jedoch nicht, ob der Drucker die
Daten erhält.
Das File wird erst bei Beendigung des Dumpes geschlossen und dies könnte
unter umständen sehr lange dauern, bis der BootMonitor wieder ansprechbar
ist.

1.25 dump

HotKey 'D'
Steht für (D)rucke Dump.
Dies stellt sicher auch eine sehr interessante Variante des Druckens von
Bootblöcken dar.
Dieser Dump sieht nämlich so aus:

OFFSET : 20 Bytes als Hex-Zahlen 20 Zeichen als ASCII-Dump

0000: 44 4F 53 00 34 20 43 21 00 00 03 70 00 00 00 00 00 00 00 00 DOS.4C!...p ↩
    ....

Auf dem Drucker stellt dies dann natürlich eine komplette Zeile dar,
vorrausgesetzt, der Drucker ist in der Lage 90 Zeichen in einer Zeile
darzustellen.

Vorsicht jedoch, wenn kein Drucker eingeschalten oder angeschlossen ist !!
In diesem Fall ist der BootMonitor in der Lage das File zu öffnen und die

Daten dorthin zu schicken, er erkennt jedoch nicht, ob der Drucker die
Daten erhält.
Das File wird erst bei Beendigung des Dumpes geschlossen und dies könnte
unter umständen sehr lange dauern, bis der BootMonitor wieder ansprechbar
ist.

1.26 device

Hotkey 'M'

Lässt das wechseln des Devices zu.
Es wird hier automatisch in dem Filerequester das Verzeichnis DEVS: ausge-
geben. Somit lässt sich leicht ein neues und eventuell Trackdiskfremdes
Device auswählen um Bootblöcke von z. b.: MS-Dos-Disketten lesen zu können.
Jedoch es werden hier, egal wie lang die ersten zwei Blöcke auch sein mögen,
nur die ersten 512 Bytes für den ersten BootSektor und die zweiten 512 Bytes
für den zweiten BootSektor gelesen.
Das heißt, wenn der Rigid-Diskblock einer Festplatte im ersten Block 1024
umfasst und ein zweiter BootSektor nicht vorhanden ist, dann wird hier nur
die ersten 512 Bytes des RDB's gelesen und die zweiten 512 Bytes des zweiten
Datenblockes gelesen.
Daraus setzt sich dann auch der dargestellte Bootblock zusammen.
Achtung, in diesem Filerequester werden keine im ROM befindlichen Devices
angezeigt, lassen sich jedoch einfach durch eingeben des Namens inkl. der
Endung .device, z. b.: scsi.device, eingeben und ansprechen.
Ein zurücksetzen auf die trackdisk.device funktioniert genauso wie bei dem
oben genannten Beispiel.
Der Undo-Buffer bleibt bei dieser Aktion unberührt, was zur folge hat,
dass sich auch Bootblöcke von anderen Speichermedien einfach auf andere
Speichermedien speichern lässt.

Dieser Menüpunkt benutzt ebenfalls die ASL-Library und wird bei nichtvor-
handensein derselben automatisch gesperrt.
Es wird jedoch in einem solchem fall eine kleine Info-Meldung ausgebeben.

1.27 bootblock

Der BootBlock des AmigaDOS:

Er ist wie folgt aufgebaut:

Die ersten vier Bytes sind mit der sogenannten Dos-Kennung ausgestattet.
Wobei die ersten drei Bytes immer die ASCII-Folge 'DOS' enthalten.
Das vierte Byte gibt Aufschluß über eventuell benutzte FileSysteme.
z.b.: 0 = OldFileSystem (OS1.3)
      1 = FastFileSystem (OS2.0)

Danach folgt ein Langwort, welches die Checksumme des Bootblockes enthält.
Ist diese Checksumme nicht korrekt, so kann und wird der Bootblock vom
Amiga nicht gestartet.

Das nächste Langwort zeigt auf den Rootblock des Datenträgers.

Man sollte es möglichst nicht ändern und enthält meistens den Wert Dez: 880.
Obwohl ich sagen muß, daß ich auch schon Bootblöcke gesehen habe, insbesonder
bei sogenannten Trackloadern, das dieser Block auch schon wo anders hin-
gezeigt hat. Warum das so ist, kann ich leider auch nicht sagen.

Die nächsten 1012 Bytes stehen nun zur ziemlich freien verfügung und ent-
halten den eigentlichen BootCode, der auch ausgeführt wird.
Der BootCode muß immer verschiebbar, das heißt in Assembler, PC-Relativ,
geschrieben sein. Erfahrene Assemblerprogrammierer wissen sicher von
was ich hier schreibe.

Mit einem Assembler, wie zum Beispiel dem MaxonAss von Maxon kann man sich
somit leicht seine eigenen Bootblöcke schreiben.

```
  dc.b  'DOS',0     ; Die Doskennung
  dc.l  0     ; Die Checksumme
  dc.l  880     ; Zeiger auf den RootBlock
```

Ab hier kann nun der eigene BootCode geschrieben werden und muß auch mit
einem simplen

```
    RTS
```

enden, ansonsten endet der Bootblock nie !!!

Ein auf diese Weise geschriebenes Assembler-Programm kann dann assembliert
werden und als (wichtig) absoluter Code abgespeichert werden.
Dabei ist es egal ob das Programm weniger als 1024 Byte belegt, wichtig
ist, daß es nicht mehr hat.
Nun kann man sich dieses Programm mit dem BootMonitor mit der Funktion
'UnArch BB' einlesen, der BootMonitor füllt dabei die übrigen Bytes, welche
noch auf die 1024 fehlen automatisch mit 0en auf.
Zum Testen kann man nun das Programm 'Execute'n und das Ergebnis anschauen,
wenn alles Ok ist, steht einem das Zurückschreiben auf eine Disk oder
ähnlichem nichts mehr im Wege.
Beim nächsten Bootvorgang, mit dieser präparierten Diskette, wird nun genau
dieses (eigene) Programm gestartet und hat somit einen eigenen Bootblock
geschrieben.

Viel Spaß mit dem BootMonitor wünscht dir dann noch der Autor.

1.28 adresse


```
    A.C.M. the Assembler-Magician
    Postfach 68 (PostBox 68)
    9508  Villach
    Austria
```

Internet: andreas.kobuaritsch@telecom.at

Der Author, der hier diese Adresse hinterlässt, ist keine Klagemauer und
übernimmt wie schon oft genug geschrieben in dieser Dokumentation
keine Haftung an etwaigen Schäden von Geräten.
Er hat jedoch bei Hilfestellungen immer einen 'offenen Kugelschreiber' be-

reit.

1.29 englisch

Sollte jemand Lust und Laune haben, diese Anleitung in Englisch zu übersetzen, dann kann er sich ja an die Arbeit machen und mir dieses Script als kleine Anerkennung zusenden (GIFTWARE).
If there is someone, who want translate this Documentation in Englisch, then you can send me your translated Script as a little Gift (GIFTWARE).

1.30 history

Geschichte:

Etwas Geschichte muß hier auch schon mal sein, überhaupt, wo der Bootmonitor nun schon die Version V1.3 angelangt ist.

Die Idee dazu lieferte mir ein ebenfalls von mir geschriebenes Programm, welches den Namen BootExecutor hatte.
Dabei war es nur so, das jenes Programm den aktuellen BootBlock der Diskette startete und nach Beendigung wieder zurückkehrte.
Eines Tages jedoch, wollte ich mehr aus den Bootblöcken als nur starten machen und ich überlegte mir, was alles noch hilfreich wäre.
Primär jedoch ging es mir um das Editieren der Bootblöcke, um kleinere Änderungen bei selbstgeschriebenen Bootblöcken leichter vornehmen zu können.
Dabei entstand nun der BootMonitor.

Nun zu den Versionen des BootMonitors:

V1.0 ist die bereits beschriebene.

V1.1 keine Veröffentliche Beta-Version. Es wurde hier nur ein kleiner aber folgenschwerer Bug behoben.

V1.2 keine Veröffentliche Beta-Version. Es wurde hier sogar schon ein kleiner Virenschutz eingearbeitet.

V1.3 wäre dann wieder zu anbieten.

V1.4 ist ab 26.04.1996 verfügbar.

V1.5 ist ab 19.05.1996 verfügbar.

Für die Zukunft des BootMonitors.

1.31 version13

Zu Version 1.3:

Es wurde hier ein Bug behoben, welcher fatale Auswirkungen beim Drücken

einer anderen als beschriebenen HotKey-Taste hatte.
Der BootMonitor empfahl sich danach mit besten Grüßen und wir sahen uns dann in Guretanien (Guru) wieder.
Der Grund dafür war nur eine falsche Reihenfolge der Abfrage zwischen existenten Hotkeys und der Sprungtabelle.
Dieser Bug wurde schon ab V1.1 behoben.
Ab Version V1.2 wurde wie oben schon geschrieben einige Änderungen und Sicherheit(sabfrag)en implementiert.
Ab der Version V1.3 wurde der BootMonitor noch weiter verfeinert und er bekam nun ein Menu eingepflanzt.
Außerdem gab es bei den vorherigen Versionen noch Probleme mit den Fonts ab Kickstart 2.0 (Vielleicht sogar schon 1.3, wurde jedoch nie getestet.).
Diese Version behielt seine HotKeys genauso wie schon ab 1.0, jedoch sind sie in die Menus eingebunden worden und somit nicht mehr über den "normalen" Tasten sondern nun in Zusammenhang mit der !!rechten!! AMIGA-Taste erreichbar.
Einzige Umstellung zu V1.0 ist hier, daß man nun halt noch die Amiga-Taste mit drücken muß ;-).
Außerdem wurde der Editor etwas geändert.
Es erscheint dazu noch ein Menu namens "CURSOR", welche ausschließlich den Editor steuern und auch nur im Editor Anwendung finden.
Aus diesem Grund wurden für den Editor nun auch die HotKeys

```
            CTRL-W
      CTRL-A        CTRL-S
            CTRL-Y
            CTRL-Z
```

herausgenommen.
Stattdessen finden nun die Tasten:

```
           RAMIGA-8
   RAMIGA-4        RAMIGA-6
           RAMIGA-2
```

anwendung.
Besitzer eines Amigas, welcher einen Zehnerblock beinhaltet sollten sich den mal mit diesen Tasten genauer unter die Lupe nehmen und sie werden feststellen, daß auch hier die Editierung etwas komfortabler geworden ist Amiga 600er Besitzer: Sorry ;-), jedoch auch solche Amigaianer kommen in den Genuß des BootMonitors, den es existieren ja ebenfalls noch die Zahlenleiste, über welche der Editor genauso ansteuern läßt, und wenn alle Stricke reißen, ist ja noch immer das Menu da.
Aber auch Besitzer einer Kickstart 1.3 können den BootMonitor, wenn auch nur eingeschränkt, es sei denn, sie haben eine geeignete ASL-Library zur verfügung, nutzen.
Die ASL-Library wird nur zur Ausgabe von FileRequestern verwendet.
Sollte diese Library nicht verfügbar sein, so sperrt der BootMonitor automatisch die

```
  Gadgets: Archive BB
                UnArchive BB
```

und die

```
    Menus: Load BB
                Write BB
```

ansonsten kann er normal weiter benutzt werden.
Außerdem besitzt der BootMonitor nun kleine Schutzfunktionen, welche zur Sicherheit dienen.
Die wären:

1.
Der BootMonitor prüft vor dem eigentlichem Start noch die Vektoren wie den CoolCapture, ColdCapture usw. und kommt mit einer entsprechenden Meldung, daß irgendetwas nicht stimmen könnte.
Danach fragt er Dich, ob Du die Vektoren zurücksetzen willst oder nicht, worauf er ebenfalls sich zurückmeldet und startet.

2.
Nach dem Ausführen eines BootBlockes werden ebenfalls die Vektoren getestet wie beim Start.

3.
Nach der Ausgabe der Vektoren wird genauso, falls nicht alle Vektoren auf 0 (NULL) sind diese Abfrage gestartet.

4.
Beim Beenden des BootMonitors werden als erstes die Vektoren geprüft, falls hier wieder alle Vektoren nicht auf 0 (NULL) sind auch eine Meldung ausgegeben und es wird vor dem Ausstieg aus dem BootMonitor nochmals rückgefragt ob man auch wirklich aussteigen will.

5.
Bei einer (krankhaften) Veränderung der Systemvektoren wird zusätzlich noch ein Beep (DisplayBeep) ausgegeben.

6.
Die einzelnen Images liegen nun in einem Unterverzeichnis names "BImages/".

7.
Das Paket sieht nun so aus:

```
  BootMon                  : Der BootMonitor selbst
  BootMon.info             : Das Icon für den BootMonitor
  BootMon.dok              : Die Dokumentation des BootMons
  BootMon.dok.info         : Das Icon für die Dokumentation
  BImages                  : Das Unterdirektory für die Images
   BImages/Archive.image   : internes Icon
   BImages/Clear.image     : internes Icon
   BImages/Device.image    : internes Icon
   BImages/Edit.image      : internes Icon
   BImages/Execute.image   : internes Icon
   BImages/Exit.image      : internes Icon
   BImages/Info.image      : internes Icon
   BImages/Read.image      : internes Icon
   BImages/Reset.image     : internes Icon
   BImages/Show.image      : internes Icon
   BImages/UnArchive.image : internes Icon
   BImages/Undo.image      : internes Icon
   BImages/Vektor.image    : internes Icon
   BImages/Write.image     : internes Icon
```

8.

Das Directory BImages sollte nun ein (Unter)Directory des BootMon's sein.

Es wurde nun auch ab dieser Version die Möglichkeit gegeben, daß eventuell (versehentlich) gestartete Viren nicht ihr Unwesen im Amiga fortsetzen können. Der Grund, warum der BootMonitor V1.3 dennoch vor dem Zurücksetzen der Vektoren rückfragt ist, daß es nicht unbedingt ein Virus sein muß, welcher die Systemvektoren verbiegt.

Der BootMonitor V1.3 ist nun auch nicht mehr FREEWARE sonder GIFTWARE, daß heißt, daß der Autor dieses Programmes bei Gefallen eine kleine Aufmerksamkeit, Brief, Reaktion oder vielleicht sogar finanzielle Anerkennung erwartet.
Der BootMonitor V1.3 soll ebenfalls frei kopiert, weitergegeben werden. Sollte er auch seinen Platz in PD-Serien finden, so darf dafür nicht mehr als der Diskettenpreis + Kopierkosten verlangt werden (wären etwa 35 ÖS od. 5 DM).
Ebenfalls darf der BootMonitor V1.3 nicht in kommerziellen Wegen vertrieben werden.
Als Anregung aller User des BootMonitors, welche den Autor mit etwas beglücken wollen, können ein ausreichend frankiertes Retourkuvert und eine 3,5" Diskette mitschicken, worauf sie dann noch ein kleines Programm aus der Werkstatt des A.C.M. the Assembler-Magician zurückgesandt bekommen.
Bitte die volle Retouradresse angeben, Adressen die Unvollständig sind oder welche Postfächer oder dergleichen enthalten werden als Spende zu Gunsten der Wohltätigkeit angesehen.
Meine Adresse ist bereits oben beschrieben.
Bei Post vom Ausland darf auch in Englisch geschrieben werden (ansonsten könnte es Kommunikationsschwierigkeiten geben :-)), Danke im Voraus.
If you will send me Mail from Countries, who have another Language as German, you can also write in ENGLISH (because it will possible, that i have my difficultes to understand you :-), thank you.

1.32 version14

Zu Version 1.4 des Bootmonitors:

in dieser Version des Bootmonitors wurden 4 Druckroutinen implementiert und die Menü's, welche die Asl-Library benötigen, geben ab jetzt bei Aufruf ohne derselben Library eine kleine Nachricht aus, das dieser Menüpunkt ohne Asl-Library nicht verfügbar ist.

Von der Beschreibung her sind der Rest der Menüpunkte identisch mit den vorhergehenden Versionen des BootMonitors und bedarf hier keiner weiter Erklärung, ansonsten ist hier zu Version 1.3 lesen.

Die Menüpunkte, welche hier neu dazugekommen sind wären folgende:

 Print Datei
 Dump Datei
 Print BB
 Dump BB

Zu den Copyrightbestimmungen ist der BootMonitor V1.4 Shareware !!
Es liegt eine eingeschränkte Version vor, welche die neuen Menüpunkte maximal einmal erfolgreich ausführen läßt.

Bei einem Versuch, einen solchen Menüpunkt ein zweites mal zu starten,
wird mit einer Shareware-Meldung quittiert.
Um an die Vollversion zu gelangen muß man an folgende Adresse richten.
Entweder schickt man eine Registrierungsmeldung + der geforderten Share-
waregebühr und erhält danach eine Diskette mit der Vollversion zurück oder
man setzt sich erstmals mit dem Author des Programmes selbst in Verbindung und
spricht/schreibt usw. sich über die Entrichtung einer Gebühr aus.
Sollte jemand vorhaben, den Bootmonitor kommerziell zu vertreiben ist hier
ebenfalls vorher eine Absprache mit dem Author notwendig.
Die Sharewareversion des Bootmonitors darf/soll/muß frei kopiert werden und
darf auch ohne Absprache des Authors in PD-Serien weitergegeben werden.
In PD-Serien gelten bei der Sharewareversion die gleichen Bestimmungen wie
in der vorhergehenden Version 1.3.

1.33 version15

Zu Version 1.5 des BootMonitors:

hier wurde der BootMonitor um die Fähigkeit erweitert, Bootblöcke auch von
anderen als dem 'trackdisk.device' zu lesen.
Zu Finden ist dies in dem Menupunkt Device.
Achtung, der BootMonitor wurde jedoch nicht um die Fähigkeit erweitert, auch
höheren Geräteadressen wie z. b.: 4 zuzugreifen.

Zu den Copyrightbestimmungen gilt hier das gleich gesagte wie in der
Version 1.4 des Bootmonitors, es wurde jedoch für User,
welche bereits in der Version 1.4 des BootMonitors registriert worden sind
eine Update-Möglichkeit eingerichtet.
Somit kostet der BootMonitor für Registrierte Personen der Version 1.4 nur
noch die Hälfte, also 50,-- ÖS oder 7,50 DM oder 5 U$.
Für nichtregistrierte Personen kostet die Vollversion des BootMonitors genau
wie in der Version 1.4 100,-- ÖS oder 10,-- DM oder 10 U$.

Die Shareware-Version darf/muß/soll ebenfalls wie in Version 1.4 in PD-Serien,
Mailboxen, CD's usw. weitergegeben werden.
Genauso darf die Shareware-Version des BootMonitors 1.5 frei kopiert werden.
Untersagt ist jedoch das erscheinen des Bootmonitors ohne Absprache des
Author's auf kommerziellem Wege oder Disketten.
Ändern des BootMonitors in irgendeinerweise ist ebenfalls untersagt und wird
strafrechtlich Verfolgt.
Erlaubt ist es aber die Guides oder Dok's des Bootmonitors unverändert in
anderen Sprachen als Deutsch zu übersetzen und dem Paket beizugeben.
Bei Einsendung eines übersetztem Skriptes winken dem Author's eine update-
fähige Version des Bootmonitors V1.5.
Ebenfalls dem Copyright unterliegen die internen Icone des Bootmonitors,
hier jedoch werden eigene Werke bei Einsendung an den Authors nicht be-
rücksichtigt.

1.34 future

Für die Zukunft des BootMonitor's hat der Author des BootMonitors einige
Verbesserungen und Feature's vor.

Zum ersten soll das Korsett der 4 Unit‘s (Geräteadressen) gesprengt werden.

Außerdem ist der Editor noch um einiges Verbesserungsbedürftig.

Das Einbinden eines Disassemblers (Author sucht noch einige Tip‘s dafür).

Das Einbinden eines Disassemblers auf der Copper-Basis, um eventuelle Copperlistings aus dem BootBlockes zu ersehen und verstehen.

Das Konvertieren von absolutem Programm-Code in Bootblöcke um ausführbare Programme in Bootblöcke unterzubringen.

Das Konvertieren von Bootblöcken in Ausführbarem Programm-Code.

Der BootMonitor soll ebenfalls Locale-Fähig werden, was jedoch zur Folge hat, daß ich mich ab dieser Version von den Usern der Kickstart‘s 1.3 und darunter verabschieden muß (Sorry, but the Show must go on).

Es sollen auch die Möglichkeit gegeben werden, Bytes einzeln über Eingabe von Zahlenwerten geändert werden können, was darauf auch einige bessere Möglichkeiten zum Editieren von Bootblöcken zur Folge hat.

Eine Passwort-Abfrage für User soll implementiert werden um vor unerlaubten Zugriffen auf eigene Speichermedien zu unterbinden.

Sämtliche Menüpunkte sollen auch über Gadget‘s wieder erreichbar werden.