

Hart, schnell und sicher

Hard-Disk-Controller mit ST-506-Schnittstelle für ECB-Rechner

Teil 3

Andreas Zippel

Das Opus neigt sich seinem vorläufigen Ende zu. Nach allgemeiner Theorie im ersten, Hardware-Praxis und Testprogrammen im zweiten Akt geht es im vorläufigen Schlußakt um die konkrete Einbindung der Treiber-Software in das Betriebssystem CP/M 3.0, mit ein paar Seitenblicken auf CP/M 2.2. Vorhang auf.

Außer der reinen Treiber-Software gibt es aber noch ein anderes kleines Problem zu lösen, das nicht nur für den Hard-Disk-Liebhaber von Interesse ist: die von Digital Research veröffentlichten Makros zur Erstellung von Disk-Parameter-Blöcken funktionieren nicht unter allen Bedingungen, so daß ein paar Korrekturen angebracht sind.

Teilen und herrschen

Wie schon im ersten Artikel angedeutet, ist die Software eigentlich sehr einfach, einfacher zumindest als etwa Treiber-Software zur Diskettenansteuerung. Aber man kann sehr viel Zeit mit falschen Strategien vertun, was wenig Freude an der Winchester aufkommen lassen würde – von der verplemperten Zeit, das Rad wieder und wieder zu erfinden, ganz zu schweigen.

Zunächst etwas Seltsames. Bisher wurde die ganze Zeit von der Riesenmenge Disk-Speicher-

platz geschwärmt, und jetzt wird der Ratschlag folgen, kleine Häppchen daraus zu machen. Warum? Ganz einfach: CP/M kennt keine hierarchischen Inhaltsverzeichnisse, bei MS-DOS etwa auch als Subdirectories bekannt. Damit fehlt CP/M ein Mechanismus, der eine erhebliche Erleichterung für den Anwender darstellt, die Übersicht über seine möglicherweise recht umfangreichen Datenbestände zu behalten.

Bei einer 25-MByte-Platte zum Beispiel sollte man rund 0,6 Prozent für das Directory vorsehen – das sind immerhin fast 5000 Einträge. Da werden selbst gut sortierte Inhaltsverzeichnisse zu Labyrinthen. Auch genügt ein defekter Sektor im Directory – und es heißt Abschied nehmen von den Dateien, deren Einträge in nachfolgenden Sektoren stehen.

Es existiert jedoch eine Möglichkeit, wenigstens etwas Ordnung zu schaffen und doch noch eine gewisse Hierarchie in Form einer Baumstruktur einzuführen. Dazu unterteilt man den Speicherplatz der Winchester in sogenannte Partitionen, was letztlich nichts anderes als 'Teile' bedeutet. Die Hard-Disk wird dabei aus Sicht des Betriebssystems in mehrere logische Laufwerke unterteilt.

Jedes logische Laufwerk hat sein eigenes Directory. Bei der Implementation wird das durch trickreichen Umgang mit der Angabe für die Anzahl der Systemspuren (System-Offset) bewerkstelligt. So besitzt zum Beispiel das erste logische Laufwerk den Offset 1, das zweite den Offset 1667 und das dritte den Offset 3780. Damit liegt auch der Anfang der zugehörigen Directories fest.

Durch diese Methode verfügt man über drei 'Subdirectories' auf derselben Platte, und mit den verbleibenden sieben bis acht MByte pro Inhaltsverzeichnis kann man der Datenflut schon Herr werden. Als weitere Hilfe zur Untergliederung gibt es ja die beim Floppy-Betrieb unter CP/M kaum beachteten USER-Bereiche.

Drei Stufen eines baumstrukturierten Inhaltsverzeichnisses hat man auf diese Art geschaffen: Die 'Wurzel' ist der Benutzer selbst. Er sollte sich merken, welche Partition was enthält. Die nächste Stufe ist dann die Partition. Und als dritte Ebene

kann man die USER-Nummer von CP/M heranziehen. Für UNIX-Kenner liest sich das so:

```
/ <partition> / <user#> / <file>
```

Schnell? Schnell!

Wie immer wieder angerissen und im ersten Teil der Artikelreihe sehr deutlich dargelegt, muß man seine Optimierungsversuche in Richtung auf höchste Transfer-Geschwindigkeit mit Hinblick auf die jeweilige Anwendung und System-Konfiguration vornehmen. Eine Beispiel-Konfiguration:

- 8 KByte pro Spur
- 16 Sektoren pro Spur
- Sektorlänge 512 Byte
- 6-MHz-DMA mit 7 Zyklen pro Transfer

Der Host-Rechner kann darauf basierend den Sektor-Puffer in rund 600 µs füllen beziehungsweise leeren. Da während dieser Transfer-Zeit der Controller vom Sektor-Puffer abgehängt ist, kann er nicht bereits den nächsten Sektor von der Platte holen oder dorthin zurückschreiben.

Während dieser Zeit laufen knapp 400 Bytes unter dem Kopf der Winchester davon (die Datenrate bei Hard-Disks beträgt üblicherweise 5 MBit/s beziehungsweise 625 KByte/s). Damit läßt sich also nur jeweils jeder zweite Sektor in einer Plattenumdrehung einlesen (beziehungsweise schreiben). Oder anders ausgedrückt: Hier ist eine Platten-Formatierung mit einem Interleave-Faktor von zwei optimal.

Das bedeutet, die 8 KByte einer Spur können in rund 34 ms eingelesen werden, was einer Transfer-Rate von rund 235 KByte/s entspricht. Allerdings immer unter der Voraussetzung, daß das Betriebssystem in der verbleibenden Zeit, in der die restlichen 100 Bytes bis zum nächsten Sektoranfang durchlaufen, alle organisatorischen Probleme gelöst hat, also Sektoranforderungen und ähnliches.

Die ermittelte Datenrate von 235 KByte/s ist damit rund 7,5mal so groß wie bei 5,25-Zoll-Floppies mit doppelter Schreibdichte (250 KBit/s beziehungsweise 31,25 KByte/s). Um möglichst effizient zu arbeiten und das Betriebssystem etwas zu entlasten, empfiehlt es sich, Platz für einen Spurpuffer

Drucker vom Spezialdistributor für jede Anwendung, in jeder Klasse



- Riteman** Matrixdrucker F+, C+, II, 15
Olympia Schreibmaschinen mit Interface, z. B. Carrera Typenraddrucker
Selkoshia Matrixdrucker für alle Rechner, z. B. SP 1000, MP 1300 AI

Wiesemann-Interface und Kabel

Wir sind autorisierter Händler und liefern nur Originalprodukte! Auch das Zubehör, wie Farbbänder usw. erhalten Sie ab Lager! Lieferung bundesweit an Endkunden, Firmen und Wiederverkäufer



Telemanstraße 18
 7250 Leonberg
 ☎ (0 71 52) 7 10 74



SYSTECH GmbH

Cross-Assembler für MS-DOS/PC-DOS: MCS-48 und MCS-51-Serie

SASS48 und SASS51

- 2-stufig schachtelbare INCLUDE-Ebenen
- schnelle Assemblierung im RAM, wenn Sourcecode kleiner als 64 K
- Codeausgabe in Binär- oder im HEX-Format

Einführungspreis bis 31. 10. 86

SASS48 DM 298,—
 SASS51 DM 398,—

passend dazu für 8035, 8039, 8048, 8049, 8748 und 8749

TRAC 48: Software-Simulator

- Schrittweise Befehlsausführung (TRACE)
- bis zu 10 Breakpoints mit Durchlaufzähler
- Disassemblierung
- Zyklenzählung

TRAC 48 DM 398,—

Alle 3 Programme sind unter MS-DOS/PC-DOS ab Lager lieferbar. CCP/M-86 und ATARI 520-Versionen sind in Vorbereitung.

SYSTECH GmbH, Birkenweg 42, 3300 Braunschweig, Tel.: 05 31/36 12 34

HEISE

Gustav Wostrack

Turbo Tools und Utilities unter MS-DOS

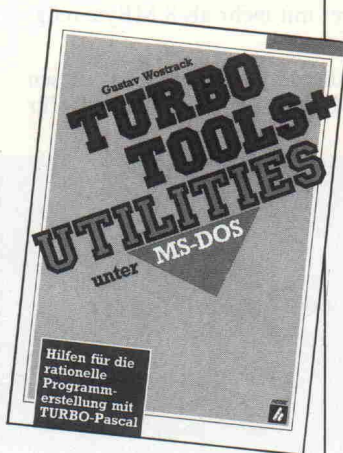
Hilfen für die rationelle
Programmerstellung
mit TURBO-Pascal

Broschur

Format 16,4 x 22,9 cm

erscheint September 1986

ISBN 3-88 229-142-7



Dieses Buch beinhaltet viele Tools und Utilities, die das Programmieren mit TURBO-Pascal einfach und rationell gestalten:

- Einen Debugger zum Austesten der Programme.
 - Ein Formatierprogramm für Quelltexte.
 - Ein Programm zur Unterstützung der Druckersteuerung, Registererstellung etc.
 - Und weitere Programme mit kommentiertem Quelltext.
- Dieses Buch ist für den Einsteiger in die Systemprogrammierung wie auch für den fortgeschrittenen Anwender geeignet.

Lieferbar über Ihren Computer-, Elektronik- und Buchhändler oder den Verlag.

Verlag H. HEISE Postf. 61 04 07 · 3000 Hannover 61

c't 1986, Heft 10

Exklusiv bei:



Der C-Werkzeugkasten für C-Programmierer in deutscher Version

- eine beständige Benutzerschnittstelle
- Masken-orientierte Dateneingabe und Editiermöglichkeiten
- einfache und schnelle Maskenerzeugung
- einen generierenden Bildschirmreiber
- Stringarithmetik
- Listengenerator
- Hilfsprogramme, Dateieditor, Datenbank-editor

High C der leistungsfähigste C-Compiler

Csharp Echtzeit Werkzeuge in deutscher Version mit Graphik

C-tree B+ Isam Dateiverwaltung in deutscher Version, mit fester und variabler Satz- und Schlüssellänge

VEDIT-Plus der stärkste Texteditor mit vielen Funktionen, in deutscher Version

★ Heiße Preise ★ Heiße Preise ★ Heiße Preise ★ Heiße Preise ★

Drucker

Citizen 120 D, 120 Z/sec, NLQ, 4 K RAM	nur 579,— DM
Citizen MSP-10, 160 Z/sec, NLQ	nur 798,— DM
Star NL-10, komplett mit Interface	nur 789,— DM
Okidata-Drucker ML 182, 192, 292, 293, 294	Lieferbar

Akustik-Koppler

Dataphon s21/23, 300 bzw. 1200/75 Baud/BTX	nur 338,— DM
ASA 2480 für Apple 2+/e kpl. mit Software ohne FTZ-Nr.	nur 169,— DM

Disketten

5 1/4" No Name SD/DD	ab 100 St. je	0,99 DM
5 1/4" No Name DS/DD	ab 100 St. je	1,69 DM
3 1/2" No Name MF 1DD	ab 100 St. je	4,49 DM
3 1/2" No Name MF 2DD	ab 100 St. je	5,90 DM
3 1/2" SKC-Skytec MF 1DD	ab 50 St. je	5,90 DM
5 1/4" SKC-Skytec 2D 96 tpi, 2HD	ab 50 St. je	6,98 DM

Festplatte für IBM-komp. Rechner

Seagate ST 225 incl. Omti Controller u. Kabelsatz	nur 1248,— DM
---	---------------

Wippermann Computer

Großhandel ★ Export ★ Einzelhandel

Füllekgund 18, 4799 Borchon-Dörehagen, ☎ 052 93/12 41

Händleranfragen erwünscht!

COMPUTER GbR

Uwe Walter und Carsten Frank

0531/18281

3300 BRAUNSCHWEIG
Kupfertwete 9



TELEX
952 637 fwgbr d

- Direktimporteure
- eigener Werkstattservice
- 6 Monate GARANTIE

ACHTUNG!

HÄNDLER-
SUPERPREISE

ARCA PC/XT

IBM-PC/XT-Kompatibel



Mehr als
10 000fach
bewährt

ARCA AT

IBM-AT-Kompatibel



In Einzelteilen oder Komplett

640K Motherboard mit 8088 CPU (4.77 MHz) Color-Grafik-Karte, Hercules-Karte, Multi-I/O-Karte, Laufwerke, Controller, Festplatten, Printer-Karte, RS-232-Karte, Multifunktionskarte und, und...

ab 1199,— DM

1024KB Mainboard mit 80286 CPU (6/8 MHz) EGA-Karte, HDD/FDD-Controller, Laufwerk, Harddisk, Multifunktionskarte, Seriell/Parallel-Karte, Speichererweiterungskarten und, und...

ab 3499,— DM

Fordern Sie unsere KOMPLETTE KOSTENLOSE LISTE an oder Sie setzen sich mit uns telefonisch in Verbindung und lassen sich kostenlos und unverbindlich beraten. Wir stellen Ihnen gern Ihr individuelles System zu optimalen Preisen zusammen.

Apple-Kompatibles



Komplettsysteme
Interfacekarten
Laufwerke

ab 899,— DM

Weiter im Programm:

Commodore, Apple, Star, NEC, Teac, Okidata, Brother, Panasonic, Zenith und, und...

Monitore, Drucker, Laufwerke, Disketten und, und...

Alles zu Superpreisen

KOMPLETTE LISTE anfordern!!!


```
; Korrigierte Macro Definition für DPBs
; bei Laufwerkskapazitäten viel größer als 300 KByte
```

```
; dpb physical$sector$size, - disk parameter block
; physical$sectors$per$track,
; number$tracks,
; block$size,
; number$dir$entries,
; track$offset,
; checksum$vec$size (optional)
; ex16 (optional)
```

```
dpb macro ?psize, ?pspt, ?trks, ?bls, ?ndirs, ?off, ?ncks, ex16, ?nocheck
local ?spt, ?bsh, ?blm, ?exm, ?dsm, ?drm, ?al0, ?all, ?cks, ?psh, ?psm
local ?n

;; physical sector mask and physical sector shift
?psm set 0
?n set ?psize/128
?psm set ?n-1

rept 8
?n set ?n/2
if ?n = 0
exitm
endif
?psm set ?psm + 1
endm

?spt set ?pspt*(?psize/128)
?bsh set 3
?n set ?bls/1024

rept 8
?n set ?n/2
if ?n = 0
exitm
endif
?bsh set ?bsh + 1
endm

?blm set ?bls/128-1
if ?psize > ?bls
?n set ?psize/?bls
?dsm set (((?trks - ?off) * ?pspt) * ?n) - 1
endif
if ?psize = ?bls
?n set ?bls / ?psize
?dsm set (((?trks - ?off) * ?pspt) / ?n) - 1
endif
if ?size set (?trks-?off)*?spt
if ?dsm set ?size/(?bls/128)-1

?exm set ?bls/1024
if ?dsm > 255
if ?bls = 1024
exitm
endif
?exm set ?exm/2
endif

?exm set ?exm-1
if not nul ex16
?exm set ex16
endif

?all set 0
?n set (?ndirs + (?bls/32)-1)/(?bls/32)
if ?n = 0
endif
if ?n > 16
endif

rept ?n
?all set ((?all shr 1) or 8000h)
endm

?al0 set high ?all
?all set low ?all
?drm set ?ndirs-1
if not nul ?ncks
?cks set ?ncks
else
?cks set ?ndirs/4
endif
if not nul ?nocheck
?cks set ?cks or 8000h
endif

dw ?spt ; 128 byte records per track
db ?bsh, ?blm ; block shift and mask
db ?exm ; extent mask
dw ?dsm ; maximum block number
dw ?drm ; maximum directory entry number
db ?al0, ?all ; alloc vector for directory
dw ?cks ; checksum size
dw ?off ; offset for system tracks
db ?psh, ?psm ; physical sector size shift
and mask

endm
```

Bild 1.

Bild 1. Der Original-Makro, den Digital Research (DRI) in seinen Applikationen zur Berechnung von DPBs vorschlägt, funktioniert nicht sicher bei hohen Plattenkapazitäten. Deshalb hier eine korrigierte Version, die Änderungen gegenüber dem Original sind fett gedruckt.

im Arbeitsspeicher zu spendieren (CP/M 3.0 unterstützt das).

Achtung, Fehler!

Die meisten CP/M-Anpasser kennen und schätzen die von Digital Research angegebenen Makros zum Generieren der verschiedenen Betriebssystemtabellen, so vermutlich auch den Makro 'DPB'. Leider wird man beim Generieren eines Systemes mit den Dimensionen einer Winchester gelegentlich verladen, weil der Makro unter gewissen Umständen versagt. In Bild 1 finden Sie daher eine korrigierte Version, mit deren Hilfe auch das korrekte Generieren von Disk-Parametern für Drives mit mehr als 8 MByte möglich ist.

Allerdings sollte man diesen Makro nicht unbedingt für

Disk-Formate mit weniger als 300 KByte einsetzen. Da versagt wiederum die für große Kapazitäten korrigierte Version manchmal. Das liegt nicht etwa an schlecht programmierten Makros, sondern an der verfluchten 16-Bit-Arithmetik des RMAC und anderer Assembler. Aber vielleicht findet ja einer unserer Leser den ultimativen Supermakro ...

Chamäleon

CP/M bietet einen unschätzbaren Vorzug: Man kann es allen möglichen Anforderungen und Wünschen anpassen. Um das aber korrekt über die Bühne zu kriegen, sollte man sich vorab erst einmal in den 'System Alteration Guide' von Digital Research einlesen. Denn all das darzulegen, was bei einer Implementation berücksichtigt werden muß, würde den Rahmen dieses Artikels ganz gewaltig sprengen. Aber es bleiben natürlich ein paar Tips, die man unter Umständen nicht im Handbuch findet. Zunächst für CP/M 3.0.

CP/M Plus

Benutzer dieses Betriebssystems kennen vermutlich bereits dessen modulare Struktur, die von DRI vorgeschlagen wurde. Hier werden dem DPH (Disk Parameter Header) des 'alten' CP/M 2.2 einfach ein paar Bytes

DW	HD\$WRITE	; Schreiben auf Harddisk
DW	HD\$READ	; Lesen von Harddisk
DW	HD\$LOGIN	; Ein"loggen" der Harddisk
DW	HD\$INIT0	; Initialisierung (erfolgt im BOOT)
DB	0	; fuer die gesamte HD
DB	0	; relative LW-Nummer
DB	0	; Disk-Modus
HD\$K0: DW	0	; SKEW-Table (0 fuer NoSkew)
DB	0, 0, 0, 0, 0, 0, 0, 0	; BDOS SCRATCH AREA
DB	0	; MEDIA FLAG (0, falls kein
		; Plattenwechsel moeglich)
DW	DPB0	; DISK PARAMETER BLOCK
DW	0	; CHECKSUM VECTOR (0, falls keine
		; Checksum-Ueberpruefung)
DW	0FFFEH	; ALLOCAtion VECTOR ALLOCATED
		; (wird von GENCPM versorgt)
DW	0FFFEH, 0FFFEH, 0FFFEH	; DIR\$CB, DTAB\$CB, HASH ALLOC'D
		; (ebenfalls von GENCPM gesetzt)
DB	0	; HASH BANK
DW	0	; CHECKSUM VECTOR

Bild 2.

Bild 2. Die Disk-Parameter-Header (DPH) sind bei CP/M 3.0 gegenüber CP/M 2.2 zu sogenannten XDPHs (eXtended DPH) erweitert worden.

vorangestellt. Über den Laufwerksbuchstaben ('A:' bis 'P:') findet das BIOS aus der Drivetable – meist mit dem Label DRV'TBL gekennzeichnet – den aktuellen DPH. Bild 2 zeigt nochmals die einzelnen Teile des XDPH, also des erweiterten (eXtended) DPH aus CP/M 2.2.

Ausgehend vom als bekannt vorausgesetzten Standard-DPH

werden die vorangestellten Extension-Bytes 'rückwärts' besprochen. Als erstes ist dann die relative Laufwerksnummer zu erwähnen. Mit ihr ist es möglich, von einem Treiber (so etwas wie dem hier vorgestellten Programm) mehrere Laufwerke bedienen zu lassen. Mit unserer Hard-Disk-Controller-Karte ist jedoch nur ein Laufwerk ansprechbar, wodurch dieser Parameter hier seine Bedeutung einbüßt.

Der nächste Wert, das sogenannte 'Mode Byte', ist für unseren Treiber ebenfalls nicht von Belang, jedoch ist es immer sicherer, auch unbenutzte Bytes definiert auf 00h zu setzen – man weiß ja nie ...

Alles Routinen

Es folgt die Adresse (Wort) der Init-Routine. Nur ein einziger XDPH zeigt auf eine echte Routine, die anderen führen auf eine einfache Return-Anweisung. (Wenn man den Empfehlungen folgt.) Die Init-Routine stellt nur fest, ob die Winchester anspricht, also im System ist, und merkt sich diese Tatsache.

Das nächste Wort enthält die Adresse einer Login-Routine. Diese wird bei der ersten Selektion eines Laufwerkes nach einem Ctrl-C angesprungen. Sie holt nur den Merker der Init-Routine und meldet die Disk als 'invalid' (ungültig), wenn von der Init-Routine ein Fehler vermerkt wurde.

Es verbleiben die Vektoren für die Read- und Write-Routinen. Diese besetzen den Vektor für Fehlermeldungen vor. Anschließend setzen sie mit Hilfe der Routine UPTASK das Task File des WD1010 (siehe hierzu die vorausgegangenen Teile des Artikels).

Beide Routinen starten dann ihre eigentlichen Aktivitäten, indem sie das entsprechende Kommando an den WD1010 schicken. Die Routine READ wartet ab, bis der Controller fertig ist, und transferiert die Daten mit der Routine HDTRANS1 zum Rechner.

Die Write-Routine wartet auf die Anforderung des Controllers (DRQ-Bit im Status-Register). Sobald diese Anforderung erfolgt, startet WRITE den Transfer der Daten zum Controller mit Hilfe von HDTRANS2. Danach wartet sie, bis der Controller fertig ist.

Dieses Warten in der Routine BUSY ist sehr wichtig, da der Controller-Chip während seiner Plattenzugriffe vom Rechner-Bus abgehängt ist und dadurch kein Zugriff auf sein internes Status-Register möglich ist. Während dieser Aktivitäten des Controllers stellt ein per Zusatz-Hardware simuliertes Busy-Bit die einzige definierte Verbindung zum Host-Rechner dar (siehe Hardware-Beschreibung im zweiten Teil des Artikels).

Die eingebaute Schleife für Wiederholungen im Fehlerfall ist eigentlich unnötig (der Controller besorgt schon einiges davon selbst), dennoch erhöht es ein wenig die Datensicherheit. Wenn Fehler auftreten, münden beide Routinen in eine gemeinsame Fehlerbehandlung. Ansonsten geht es in eine kleine Schlußroutine, die die Winchester deselektiert (bei UPTASK wurde sie ja selektiert). Die Fehlerroutine meldet den jeweiligen Fehler auf dem Bildschirm und haucht ihr Leben dann in der gleichen Schlußroutine aus.

Noch ein Wort zum Banking, einer Methode von CP/M 3.0 zur Speicherplatzverwaltung, die CP/M 2.2 nicht unterstützt. Der gesamte Treiber mit Ausnahme der Transferoutine HDTRANS1 und HDTRANS2 kann im 'gebankten' Teil liegen. Bitte beachten Sie aber, daß die Reihenfolge zwischen Laden der DMA-Adresse und dem Bankumschalten nicht vertauscht wird. Manche Systeme verstecken nämlich die DMA-Adresse in einer Bank (und nicht im Common-Bereich). Und mit einer 'weggeschwitten' DMA-Adresse bekommen sie dann meist einen fürchterlichen Anfall von Selbstzerstörung.

Zur Assembler-Terminologie ist vielleicht auch noch ein Hinweis angebracht. Routinen, die mit einem Fragezeichen versehen sind, gelten global und müssen vom Rest des Systems bereitgestellt werden. Im 'System Guide' von DRI ist deren Funktion genau beschrieben, wobei einige Routinen allerdings nur in den Beispieldrivers zu finden sind.

Hard-Disk unter CP/M 2.2?

Im Prinzip ja, konkret ausprobiert haben wir es nicht. Für kreative Tüftler ein paar Denkanstöße, wie eine Änderung

des 3.0-Treibers den 2.2-Betrieb ermöglichen sollte.

Zunächst entfallen die Zusätze zu den DPHs. Statt dessen sind einfach die Einsprünge zu den Routinen, die vorher in den DPHs standen, als globale Namen zu vereinbaren. Ein Verteiler vor den schon vorhandenen Floppy-Routinen kann dann dafür sorgen, daß die Unterprogramme für die Hard-Disk-Steuerung angesprungen werden.

Für den bei CP/M 2.2 benötigten Blocking-Deblocking-Algorithmus kann man die zusätzlichen Bytes am Ende der DPBs nutzen. Diese Bytes sollten dann die Sektorlänge in einer Form enthalten, die es einfach macht, die einzelnen logischen Sektoren aus dem physikalischen Sektor herauszuarbeiten.

Abzurufen ist davon, die Sektoren der Einfachheit halber zu 128 Byte Länge zu wählen, da damit Geschwindigkeit und Speicherplatz verlorengehen. Aber wer die Schwierigkeiten im Umgang mit dem Blocking-Deblocking scheut, braucht nicht zu verzweifeln: Der Treiber erlaubt auch Sektorgrößen von 128 Bytes.

Mit diesem vorerst letzten Beitrag ist der Winchester-Controller aber noch nicht aus dem Blickfeld verschwunden. Die nächste Monitorversion für den c't86 wird als eine ihrer Neuerungen die ganze Treiber-Software für die Controller-Karte bereits enthalten. Die neue Monitorversion ist jedoch in ihrer Gesamtheit für eine Freigabe noch nicht ausreichend erprobt, so daß c't86-Besitzer noch ein paar Hefte Geduld aufbringen müssen.

PAGE 70

TITLE 'WD1010 WINCHESTER CONTROLLER DRIVER [00.00.85] fuer PROF-00'

```
; CP/M-80 VERSION 3 -- MODULAR BIOS
;
; Version 0.03 (C) by A.Zippel 29.4.86
; Deutsche Kommentierung (P) by A.Zinser 12.6.86
;
; Alle verwendeten Parameter sind fuer eine Winchester
; MicroScience HI-725, 4 Koepfe, ausgelegt
```

```
; Laufwerk-Tabellen (fuer BIOS-1/0 & DRVTL.ASM)
```

```
PUBLIC HDSK0,HDSK1,HDSK2
```

```
; verwendete/benoeigte Parameter aus dem BIOS/BIOS
```

```
EXTRN @0DRV,@drv ; absolute LW-Nummer
EXTRN @DMA ; DMA-Adresse
EXTRN @TRK,@SECT ; Sektor-Adresse
EXTRN @DBNK,@CBNK ; Daten- und Common-Bank-Nummern
```

```
; SYSTEM CONTROL BLOCK VARIABLES
```

```
EXTRN @ERRMODE ; BIOS Fehlermeldungs-Modus
```

```
; Utility-Routinen aus dem Standard-BIOS
```

```
EXTRN ?WBOOT ; Warm-Boot-Adresse
EXTRN ?PMMSG ; Print Message
EXTRN ?PDERR ; Print BIOS DISK ERROR HEADER
EXTRN ?BANK ; BANK SELECT-Routine
```

```
; CP/M 3 DISK DEFINITION MACROS
```

```
MACLIB CP3IN ; modifiziertes Makro fuer DPB
```

```
; Z80 MACRO LIBRARY INSTRUCTION DEFINITIONS
```

```
MACLIB Z80
```

```
; Port-Adressen der Hard-Disk-Karte/Controller
```

```
; In der c't-86-Version wird Bit 0 durch Bit 3 ersetzt
```

```
ct86 equ 7
```

```
ect86 equ 0
```

```
; Dadurch gibt es bei den betroffenen Adressleitungen einen
```

```
; Offset von 7.
```

```
jumper equ ct86
```

```
hdcbase equ 000h ; basis adresse des controllers
hdcdat equ hdcbase ; datenregister
hdcerr equ hdcbase+1+jumper ; (r) error register
hdcwpc equ hdcbase+1+jumper ; (w) write-precompensation start
hdcscn equ hdcbase+2 ; sector-zaehler fuer format
hdcsek equ hdcbase+3+jumper ; sector register
hdcctl equ hdcbase+4 ; low byte des cylinders
hdcchy equ hdcbase+5+jumper ; high byte des cylinders
```



```

hdcsdh equ   hdcbase+6       ; sdh-register
hdccmd equ   hdcbase+7+ jumper ; (w) command register
hdcsa equ    hdcbase+7+ jumper ; (r) status register

```

```

; Allgemeine EQUals

```

```

CR EQU 0dh
LF EQU 0ah
BELL EQU 07h

```

```

; Controller-Kommandos

```

```

c$rest EQU 10h      ; Restore Drive (Recalibrieren)
c$read EQU 20h      ; Lesekommando (mit Retry)
c$write EQU 30h     ; Schreibkommando (mit Retry)

```

```

;*****

```

```

; physikalische Definitionen der Harddisk. Diese Parameter muessen an das ***
; gewaehlte Format und Laufwerk angepasst werden. ***

```

```

MPVALUE EQU 32      ; Begin der Precompensation (s.Text) ***
SECSIZ EQU 1024     ; Sektorgroesse [Bytes] ***
secmask EQU 01000000 ; Codierung der Sektorgroesse (s.Text) ***
SECTPRK EQU 9       ; Sektoren pro Spur und Kopf ***
heads EQU 4         ; Anzahl Koepfe (muss fuer diese
                    ; Implementation unbedingt ein *
                    ; Vielfaches von 2 sein) ***
cylIND EQU 612      ; Anzahl Zylinder *

```

```

;*****

```

```

; STATUSBIT MASKEN (siehe auch c't 8/86)

```

```

DRQFLG EQU 00h      ; Data Request
BUSYFLG EQU 00h     ; Controller Busy
ERRFLG EQU 01h      ; Error
RDYFLG EQU 40h      ; DRIVE READY

```

```

; Select-Deselect-Maskierung (l=select)

```

```

selmask equ 00h      ; LW Selektieren
desmask equ 00h      ; und deselektieren

```

```

PAGE

```

```

; EXTENDED DISK PARAMETER HEADERS (XPDHS)
; fuer jedes logische Laufwerk getrennt,
; (Partition = log. LW)

```

```

dseg

```

```

DW HD$WRITE      ; Schreiben auf Harddisk
DW HD$READ       ; Lesen von Harddisk
DW HD$LOGIN      ; Ein"loggen" der Harddisk
DW HD$INIT0      ; Initialisierung (erfolgt im BOOT) fuer die gesamte HD
DB 0             ; relative LW-Nummer
DB 0             ; Disk-Modus
HDSK0: DW 0      ; SKEW-Table (0 fuer Noskew)
DB 0,0,0,0,0,0,0,0 ; BDOS SCRATCH AREA
DB 0             ; MEDIA FLAG (0, falls kein Plattenwechsel moeglich)
DW DPB0         ; DISK PARAMETER BLOCK
DW 0            ; CHECKSUM VECTOR (0, falls keine Checksumueberpruefung)
DW 0FFFFH       ; ALLOCATION VECTOR ALLOCATED (wird von GENCPM versorgt)
DW 0FFFFH,0FFFFH,0FFFFH ; DIRBCB, DTABCB, HASH ALLOC'D (ebenfalls von GENCPM gesetzt)
DB 0            ; HASH BANK
DW 0            ; CHECKSUM VECTOR

```

```

DW HD$WRITE      ; Schreiben auf Harddisk
DW HD$READ       ; Lesen von Harddisk
DW HD$LOGIN      ; Ein"loggen" der Harddisk
DW HD$INIT1      ; Initialisierung erfolgte bereits mit INIT$0
DB 0             ; relative LW-Nummer
DB 0             ; Disk-Modus
HDSK1: DW 0      ; SKEW-Table (0 fuer Noskew)
DB 0,0,0,0,0,0,0,0 ; BDOS SCRATCH AREA
DB 0             ; MEDIA FLAG (0, falls kein Plattenwechsel moeglich)
DW DPB1         ; DISK PARAMETER BLOCK
DW 0            ; CHECKSUM VECTOR (0, falls keine Checksumueberpruefung)
DW 0FFFFH       ; ALLOCATION VECTOR ALLOCATED (wird von GENCPM versorgt)
DW 0FFFFH,0FFFFH,0FFFFH ; DIRBCB, DTABCB, HASH ALLOC'D (ebenfalls von GENCPM gesetzt)
DB 0            ; HASH BANK
DW 0            ; CHECKSUM VECTOR

```

```

DW HD$WRITE      ; Schreiben auf Harddisk
DW HD$READ       ; Lesen von Harddisk
DW HD$LOGIN      ; Ein"loggen" der Harddisk
DW HD$INIT1      ; Initialisierung erfolgte bereits mit INIT$0
DB 0             ; relative LW-Nummer
DB 0             ; Disk-Modus
HDSK2: DW 0      ; SKEW-Table (0 fuer Noskew)
DB 0,0,0,0,0,0,0,0 ; BDOS SCRATCH AREA
DB 0             ; MEDIA FLAG (0, falls kein Plattenwechsel moeglich)
DW DPB2         ; DISK PARAMETER BLOCK
DW 0            ; CHECKSUM VECTOR (0, falls keine Checksumueberpruefung)
DW 0FFFFH       ; ALLOCATION VECTOR ALLOCATED (wird von GENCPM versorgt)
DW 0FFFFH,0FFFFH,0FFFFH ; DIRBCB, DTABCB, HASH ALLOC'D (ebenfalls von GENCPM gesetzt)
DB 0            ; HASH BANK
DW 0            ; CHECKSUM VECTOR

```

```

CSEG

```

```

; Die Disk-Parameter-Blocke muessen entweder mit dem
; modifizierten DPB-Makro DPM3N.LIB oder ebenfalls per
; Hand erstellt und an das jeweilige LW/Format
; angepasst werden.

```

```

DPB0 ;DPB 1024,9,777,4096,2048,1,0
DW 72      ; 128 BYTE RECORDS PER TRACK
DB 5,1FH   ; BLOCK SHIFT AND MASK
DB 1       ; EXTENT MASK
DW 1745    ; MAXIMUM BLOCK NUMBER
DW 7FFH    ; MAXIMUM DIRECTORY ENTRY NUMBER
DW 0FFFFH  ; ALLOC VECTOR FOR DIRECTORY
DW 0000H   ; CHECKSUM SIZE
DW 1       ; OFFSET FOR SYSTEM TRACKS
DB 3,7     ; PHYSICAL SECTOR SIZE SHIFT

```

```

DPB1 ;DPB 1024,9,1665,4096,2048,778,0
DW 72      ; 128 BYTE RECORDS PER TRACK
DB 5,1FH   ; BLOCK SHIFT AND MASK
DB 1       ; EXTENT MASK
DW 1995    ; MAXIMUM BLOCK NUMBER
DW 7FFH    ; MAXIMUM DIRECTORY ENTRY NUMBER
DW 0FFFFH  ; ALLOC VECTOR FOR DIRECTORY
DW 0000H   ; CHECKSUM SIZE
DW 778     ; OFFSET FOR SYSTEM TRACKS
DB 3,7     ; PHYSICAL SECTOR SIZE SHIFT

```

```

DPB2 ;DPB 1024,9,2451,4096,2048,1666,0
DW 72      ; 128 BYTE RECORDS PER TRACK
DB 5,1FH   ; BLOCK SHIFT AND MASK
DB 1       ; EXTENT MASK
DW 1765    ; MAXIMUM BLOCK NUMBER
DW 7FFH    ; MAXIMUM DIRECTORY ENTRY NUMBER
DW 0FFFFH  ; ALLOC VECTOR FOR DIRECTORY
DW 0000H   ; CHECKSUM SIZE
DW 1666    ; OFFSET FOR SYSTEM TRACKS
DB 3,7     ; PHYSICAL SECTOR SIZE AND SHIFT

```

```

PAGE

```

```

DSEG

```

```

; Hard-Disk-I/O-Routinen

```

```

;

```

```

; Initialisierung, wird im BOOT vorgenommen
; Bei der Initialisierung wird ein RESTORE-DRIVE-Kommando abgesetzt
; und in Abhaengigkeit der Controller-(Fehler-)Meldungen
; ein Fehler-Bit gesetzt.

```

```

HD$INIT0:

```

```

CALL RESTORE      ; RESTORE absetzen
LXI H,RESTERROR   ; Fehlermeldungstext
MVI A,0FFH        ; und Fehler setzen
JRC HD$INIT2      ; FALLS CARRY gesetzt!
MVI A,0           ; ansonsten fehlerflag zuruecksetzen

```

```

HD$INIT2:

```

```

STA HD$BLOCKED    ;
CC ?MSG           ; Nur Meldung, falls Fehler
MVI A,0C3H
STA HD$INIT0      ; Weitere Initialisierungen unterbinden

```

```

HD$INIT1:

```

```

JP HD$EXIT        ; Deselect Drive

; LOGIN. Diese Routine wird zur Bestimmung des
; Diskettenformates verwendet. Ausserdem werden mit ihrer
; Hilfe die entsprechenden Parameter im XDPH erstellt. Da
; auf der Harddisk kein Formatwechsel erfolgen kann und der
; XDPH schon fertig vorliegt, erfolgt nur die Pruefung, ob
; auf die Disk zugegriffen werden kann (das Fehlerflag wird
; bereits von INIT gesetzt). Im Fehlerfall wird das Carry
; gesetzt und der aufrufenden BIOS-Routine damit der Fehler
; signalisiert.

```

```

HD$LOGIN:

```

```

LDA HD$BLOCKED
ANA A
RZ                ; OK, DISK AVAILABLE
STC
RET               ; DISK NOT AVAILABLE

```

```

PAGE

```

```

; Lese-/Schreib-Routinen

```

```

;

```

```

; Die folgenden beiden Routinen werden mit folgenden
; Parametern aufgerufen:

```

```

; @dwa (16) DWA-Adresse
; @dbnk (8) Banknummer des Datenbereiches
; @cbnk (8) Banknummer des Common-Bereiches
; @trk (16) Spurnummer
; @sect (16) Sektornummer

```

```

; Lesen von der Hard-Disk

```

```

HD$READ:
MVI A,2           ; Anzahl Wiederholungen im Fehlerfall
HD$READ1:

```

```

PUSH PSM
LXI H, READMSG ; Speichere Information ueber die Zugriffsart fuer
SHLD OPERATIONNAME ; eine detaillierte Fehlermeldung ab
CALL UPTASK ; Setze TASK-Register
MVI A, c$read
OUT HDCCMD ; Beginne Leseoperation
CALL BUSY ; Warte, bis Lesen zu Ende
JrC HD$ERROR ; Falls C-Fleg gesetzt, trat Fehler auf
POP PSM
CALL WAITDRQ ; Ansonsten warte auf DATA-REQUEST
CALL HDTRANS1 ; Uebertrage die Daten (Sektorpuffer => @DMA)
JMP HD$EXIT

```

```

HD$ERROR:
CALL CLEARDRQ ; Loesche anliegende DATA-REQUESTS
POP PSM
DCR A ; noch ein Versuch ?
JrNZ HD$READ1 ; JMP, falls ja
Jr wr$error1 ; sonst Fehler behandeln

; Schreibe auf Harddisk (Ablauf aquivalent zu LESEN)

```

```

HD$WRITE:
MVI A, 2
HD$WRITE1:
PUSH PSM
LXI H, WRITEMSG
SHLD OPERATIONNAME
CALL UPTASK
MVI A, c$write
OUT HDCCMD
CALL WAITDRQ ; Uebertrage Daten in den Sektorpuffer
CALL HDTRANS2
CALL BUSY
JrC WR$ERROR
POP PSM
JMP HD$EXIT
WR$ERROR:
POP PSM
DCR A
JrNZ HD$WRITE1
JMP HD$ERROR

```

; Setzen des TASK-Register
 ; Dem BIOS/BDOS praesentiert sich die Harddisk als
 ; "Disklaufwerk" mit einem Kopf und einen entsprechend
 ; grossen Anzahl Spuren (s. auch Text). Daher muss
 ; in der folgenden Routine die physikalische Sektor-Kopf-Spur-
 ; information aus der logischen Sektor-Spur-Information
 ; gewonnen werden. (Dabei kann fuer alle Partitionen die
 ; gleiche Routine verwendet werden, da sie fortlaufende
 ; Spurnummern verwenden. Die erste Spur der zweiten
 ; Partition ist die auf die letzte Spur der ersten Partition
 ; folgende. Dies wurde durch einen entsprechenden
 ; SYSTEM-TRACK-OFFSET erreicht (siehe DPB)).
 ; Man kann alerdings die Selektion der verschiedenen
 ; Partitionen auch ueber @drv, relative drive number, die im
 ; dph angegeben ist, vornehmen. In diesem Fall besteht auch
 ; eine einfache moeglichkeit, mit z.B. 6-Koepfern zu arbeiten

```

UPTASK:
CALL CLEARDRQ ; RESTLICHE DRQ'S ABHOLEN (WENN DA)

; ----- SEKTOR
LDA @SECT ; HOLE SEKTORNUMMER
OUT HD$SEK ; SETZE REGISTER

; ----- SDH-REGISTER
;
; Die unteren Bits der logischen Spurnummer repraesentieren
; die physikalische Kopfnummer (2 Koepfe - 1 Bit, 4 Koepfe - 2 Bit etc.)
; Die Kopfadresse wird zusammen mit den S = Sektor-Size- und D = Drive-
; select-Informationen auf das SDH-Register geschrieben
LDA @TRK ; Die unteren 1/2/3 Bit der Spur
ANI heads-1 ; repraesentieren die Kopfnummer.
ORI secmask or selmask ; SEKTORLAENGE (maskiert) und Select
OUT HD$SDH ; dazu und REGISTER SETZEN

; ----- Zylinder
;
; Da die unteren Bytes der log.Spur den Kopf darstellen,
; muessen sie aus der log.Spur-Information entfernt werden,
; um die phys.Spur zu erhalten. Dieser 10-Bit-Wert wird in die
; entsprechenden HD-Register geschrieben
LDA @TRK ; LOW BYTE DER SPUR
MOV B, A
LDA @TRK+1 ; HIGH BYTE DER SPUR
SRLR A ; Kopfinformation herausrotieren
RARR B ; H1(@TRK)/2
IF heads>2
SRLR A
RARR B ; H1(@TRK)/4
IF heads>4
SRLR A
RARR B ; H1(@TRK)/8
ENDIF
ENDIF

```

```

OUT HDCCYH ; SETZEN DES HIGH BYTES
MOV A, B
OUT HDCCYL ; SETZEN DES LOW BYTES
RET

```

; Loeschen anliegender DATA-Requests

```

CLEARDRQ:
IN HD$STA
ANI DRQFLG
RZ
IN HD$DAT ; DUMMY READ
Jr CLEARDRQ

```

; Warte, bis DRQ aktiv wird

```

WAITDRQ:
IN HD$STA
ANI DRQFLG
RZ
Jr WAITDRQ

```

; Warte, bis Kommando fertig abgearbeitet ist

```

BUSY:
IN HD$STA
ANA A
JM BUSY ; NEGATIV -> NOCH BUSY
RAR ; ERROR BIT IN CARRY
RET
; CONTROLLER- UND LAUFWERKS-INITIALISIERUNG.
; Es wird gepueft, ob der Controller "frei" ist (kein busy); ist
; dieses der Fall, wird das RESTORE-Kommando abgesetzt und
; auf korrekte Abarbeitung (kein Busy, kein Error) gewartet;
; geschieht dies nicht innerhalb einer angemessenen Frist,
; wird das Laufwerk als fehlerhaft zurueckgemeldet. Bei
; langsam "hochfahrenden" Laufwerken ist ggf. der Zaehler
; innerhalb der WAIT-Routine zu erhoehen.

```

```

RESTORE:
mvi a, selmask ; Selektiere LW
OUT HD$SDH
lxi h, -1 ; timeout-zaehler auf 0
restore0:
in hdcsta ; Laufwerk busy ?
ana a ; negativ -> noch busy
jm restore4 ; ueberspringe restore-anforderung, da
CALL CLEARDRQ ; Controller nicht bereit ist
OUT HDCCMD ; RESTORE COMMAND
restore2:
in hdcsta
ana a
jm restore4 ; laufwerk noch busy
rar ; falls nicht busy und kein fehler:
JrNC RESTORE1 ; OK, DISK MELDET SICH FERTIG
restore4:
CALL WAIT
LXI D, 1
DAD D ; TIMEOUT TESTEN
JrNC restore3 ; NOCH NICHT, WEITER VERSUCHEN
RET ; FEHLER, (DISK NICHT BEREIT), CY=1
RESTORE1:
MVI A, MPVALUE ; START TRACK FOR PRECOMPENSATION
OUT HD$MPC
xra a
ret ; CY=0 (kein Fehler)

```

; Warteschleife (fuer RESTORE)

```

WAIT:
PUSH PSM
PUSH H
LXI H, 8000H
wait1:
dcx h
mov a, h
ora l
JrNZ WAIT1
POP H
POP PSM
RET

```

cseg

; Daten-Uebertragungs-Routinen

; Einlesen der Daten in den DMA-Puffer
 ; WICHTIG: Die sector-groesse muss fuer diese routine
 ; groesser 128 Byte/Sektor sein, allerdings ist dieses
 ; format aus anderen Gruenden auch nicht zu empfehlen (s.Text)

```

HDTRANS1:
LHLD @DMA ; AUF DMA PUFFER ZEIGEN
DI ; Interrupts verbieten
LDA @DBANK
CALL ?BANK ; BANK FUEER DATEN SELECTIEREN
lxi b, hdcdat ; b = 0, c = Datenregister
rept secsiz/256
inir
enda
Jr hdtrans3 ; Ende der Datenuebertragung

```



```

; Schreiben der Daten in den Controller-Puffer
HDTRANS2:
  LHALD @DMA ; AUF DMA PUFFER ZEIGEN
  DI
  LDA @BANK
  CALL ?BANK ; BANK FÜR DATEN SELECTIEREN
  lxi b,hdcdat ; b = 0, c = Datenregister
  rept secsiz/256
  outir
  endm

HDtrans3:
  LDA @BANK
  CALL ?BANK ; PROGRAMMBANK SELECTIEREN
  EI
  RET

DSEG
; Fehlerbehandlung
;
; Im Fehlerfall wird, sofern das @ERMODE-Flag entsprechend
; gesetzt ist, eine ausführliche Fehlermeldung in der Form:
; BIOS ERR ON D: T=NN, S=MM (OPERATION) (TYPE), RETRY ?
; ausgegeben. Die Angabe (TYPE) enthaelt alle im Fehler-Byte
; des Controllers markierten Errors in Klartext.

HD#ERROR:
  LDA @ERMODE
  CPI @FFH
  JZ HD#HERROR ; Keine ausführliche Fehlermeldung
  CALL ?PERR ; PRINT MESSAGE HEADER
  LHALD OPERATIONNAME ; wird von READ/WRITE gesetzt
  CALL ?MSG ; LAST FUNCTION
  IN HDCEAR ; Hole Fehler-Byte vom Controller
  LXI H,ERROR$TABLE ; Zeige auf Meldungstabelle

ERRM1:
  MOV E,M
  INX H
  MOV D,M
  INX H ; GET NEXT MESSAGE ADDRESS
  ADD A
  PUSH PSW ; rotiere nach links und sichere
  PUSH H ; die restlichen Bits
  XCHG
  CC ?MSG ; falls Bit gesetzt war, gebe
  XCHG ; Meldung aus
  POP H
  POP PSW
  JNZ ERRM1 ; Falls noch mehr Bits, rotiere...

HD#HERROR:
  CALL HD#EXIT ; Deselect drive
  MVI A,1 ; setze A=1 fuer BIOS/BOOS (Error occurred)
  HD#1#ERROR:
  RET ; RETURN HARD ERROR TO BOOS

HD#EXIT:
  MVI A,desmask
  OUT HD#CSH ; DESELECT DRIVE
  XRA A ; NO ERROR
  RET
  PAGE

CSEG

; Hilfsspeicherstellen, Tabellen & Messages ...
;
HD#BLOCKED DB 0 ; HARDDISK NICHT ANSPRECHBAR WENN () 0

DSEG

READ#MSG DB ' , READ',0
WRITE#MSG DB ' , WRITE',0

OPERATIONNAME DW READ#MSG

; Tabelle der Zeiger auf die detaillierten Fehlermeldungen
; Jede Adresse entspricht einem Fehlerbit des HD100
; Start ist Bit 7.

ERROR$TABLE DW B7#MSG
DW B6#MSG
DW B5#MSG
DW B4#MSG
DW B3#MSG
DW B2#MSG
DW B1#MSG
DW B0#MSG

B7#MSG DB ' , BAD BLOCK',0
B6#MSG DB ' , CRC',0
B5#MSG DB ' , ' ,0
B4#MSG DB ' , REC NOT FOUND',0
B3#MSG DB ' , ' ,0
B2#MSG DB ' , ABORTED',0
B1#MSG DB ' , NO TRACK',0
B0#MSG DB ' , NO DAM',0

RESTEROR DB bell,'** Winchester Drive NOT AVAILABLE **',cr,lf,0

END

```

Das Treiber-Programm für die Hard-Disk-Steuerung zur
Einbindung in CP/M 3.0.

ct

Echtzeit-Multitasking mit 68000-Rechner unter 10 000 DM?

RTOS-UH PEARL

Programmentwicklungssystem mit Echtzeit-Multitasking-
Betriebssystem der Universität Hannover
für die Atari-ST-Serie

Leistungsdaten:

(siehe auch c't-Serie ab Heft 6/86)

Freier Arbeitsspeicher (bei 1 MByte RAM):

ca. 980 KByte

Anzahl quasiparallel laufender Tasks:

praktisch unbegrenzt

Reaktionszeit auf Prozeßinterrupt:

< 220 µs

Maximale Taskwechselfrequenz:

> 2,2 kHz

Compiler-Geschwindigkeit:

ca. 500 Zeilen/Minute

Task-Synchronisierung durch Semaphore



Besonderheiten:

- 2. Nutzer möglich (über Terminal an der RS-232-Schnittstelle)
- Hochauflösende schnelle Farbgrafik wird unterstützt
- Funktionstasten unter RTOS spielend leicht programmierbar
- ST-Userport (c't 3/86) wird unterstützt (Version B)
- RTOS macht RAM-Disk und Druckspooler überflüssig
- Hardware-abhängiger Systemteil voll dokumentiert
- Entwicklung ROM-fähiger Software wird unterstützt

Lieferumfang:

Echtzeitbetriebssystem RTOS-UH (EPROM-resident), PEARL-Compiler, 68000-Assembler, Linker/Lader, Monitor/Debugger mit 68000-Disassembler, Editor, diverse Utility- und Demoprogramme, umfangreiche Dokumentation

Lieferformen:

Version A: Vier EPROMs (27256) zum Betrieb mit der ST-EPROM-Bank (c't 1/86), Utility-Diskette, inkl. Handbuch 218 DM

Version B: Zwei EPROMs (27256) zum Betrieb mit dem ST-Userport (c't 3/86), Diskette mit PEARL-Compiler und Utilities, inkl. Handbuch 198 DM

Steckfertige Zusatzkarten mit RTOS-UH/PEARL sind im Fachhandel erhältlich. Einen Bezugsquellennachweis senden wir Ihnen gern zu.

So können Sie bestellen:

Um unnötige Kosten zu vermeiden, liefern wir nur gegen Vorkasse. Fügen Sie Ihrer Bestellung einen Verrechnungsscheck über die Bestellsomme zuzüglich DM 3,— (für Porto und Verpackung) bei oder überweisen Sie den Betrag auf eines unserer Konten.

Bankverbindungen:

Postgiroamt Hannover, Kt.-Nr. 93 05-308

Kreissparkasse Hannover, Kt.-Nr. 000-019968 (BLZ 250 502 99)

Ihre Bestellung richten Sie bitte an:

Verlag Heinz Heise GmbH
Vertrieb/Software II
Postfach 61 04 07
3000 Hannover 61

