Eine Publikation der System 99 User-Graupe

A publication of the System 99 User Group

[text of front cover]

Disk Info

Eine Publikation der System 99 User-Graupe

A publication of the System 99 User Group

Table of Contents

1. Introduction

This section of The Cyc contains the original German text of *Disk-Info* by Christopher Winter, and Harald Glaab. The original German typographical layout has been modified so that an English translation of the German text may be provided. The method chosen was to use "parallel paragraphs," i.e. each paragraph of German is matched with the corresponding English translation, irrespective of the length of either paragraph.

The editors of The Cyc are *not* fluent in German. Much of the translation was done by AltaVista (http://world.altavista.com/tr). Machine translators are notorious for not understanding context or colloquialism. On many occasions *Langenscheidt's New Concise German Dictionary*, and *The Oxford Duden German Dictionary* were consulted in an attempt to improve the translation. We also found two other significant area for translation errors — abbreviations and slang.

Where possible we have made translation corrections based on our limited German. But, in the case of any confusion, the original German text should be consulted. Our hope is that one or more language purists will be moved to provide a higher-quality translation.

We would like to thank Christopher Winter and Harald Glaab for allowing us to reproduce their excellent publication *Disk Info* — which represents many, long hours of hard work and exceptional knowledge of the TI-99/4A and its associated disk systems. In particular, Harald Glaab exhibited the patience of Job in dealing with our numerous emails requesting further clarifications.

Dan Eicher
Mike Wright

1.1. German abbreviations

bzw.	beziehungsweise	respectively (oder vielmehr: that is, or to be precise; oder: or)
d.h.	das Heißt	that is (i.e.)
evtl.	eventuell	perhaps, possibly
ff	Folgende Seiten	following pages
ggf.	gegebenenfalls	if necessary
i.d.R.	in der Regel	usually
i.O.	in Ordnung	in order (all right)
o.ä.	oder ähnlich	or the like
s.o.	siehe oben	see above
s.u.	siehe unten	see below
z.B.	zum Beispiel	for example (e.g.)

1.2. Notes on layout

This edition follows the CaDD standard for all TI documentation — but adapted for parallel columns.

Authors Winter and Glaab intended for an independent chapter/page numbering scheme. References to these do not apply to this edition. Pages are numbered sequentially, and cross-references generated accordingly. However, the original chapter numbers have been retained. (Chapter 1 was the Table of Contents in the original edition. In this edition Chapter 1 is this introduction.)

The table of contents (TOC) is auto-generated. In the TOC, German chapters and sections are numbered, and are followed by the English translation.

2. Urheberrechtsvermerke

Für *Disk-Info* gelten Regeln, wie für jedes andere Dokument. Das bedeutet den Verzicht des jeweiligen Besitzers auf die Anfertigung jeder Art von Kopien und/oder die Umsetzung des gedruckten Textes in maschinenlesbare Form bzw. deren Vorbereitung. Selbstverständlich gilt dies für Ausschnitte wie für die Gesamtheit des Werkes.

Auch wird jegliche Haftung bei der Anwendung der hier aufgezeigten Methoden und Zusammenhänge ausgeschlossen, ebenso die Gewähr für die unbeschränkte Richtigkeit und Vollständigkeit der gemachten Aussagen sowie deren Nutzbarkeit für bestimmte Zwecke.

Im Rahmen des Urheberrechtes sind Übersetzungen genauso zulässig wie Ausarbeitungen, die auf den in *Disk-Info* zusammengetragenen Daten beruhen jedoch nur mit unmißverständlicher Quellenangabe.

Die Wiedergabe von Auszügen aus *Disk-Info* im Rahmen von Clubzeitungen wird ausdrücklich erlaubt, sofern diese Clubzeitungen ausschließlich in gedruckter Form vorliegen — ggf. sind die entsprechenden Abschnitte den Disketten in Kopie der *Disk-Info*-Seiten beizulegen.

Sollten Sie eine offizielle Version von *Disk-Info* erwerben wollen oder Vorschläge zur Verbesserung von Inhalt und Struktur haben, so wenden Sie sich bitte an die folgende Adresse:

Christopher Winter
xxxxxxxxxxxxxxxx
xxxxx xxxxxxxxxx

Harald Glaab
xxxxxxxxxxx
xxxxx xxxxxxxxxxx

Copyright notice

To *Disk Info*, rules apply, like to every other document. That means the renouncement (prohibition) of the respective owners for making reproductions of the text as printed copies and/or the conversion to machine-readable form. Of course this applies to extracts as to the whole of the work.

Also any adhesion is excluded with the application of the methods and connections pointed out here, likewise the guarantee for the unrestricted correctness and completeness of the made statements as well as their serviceability for certain purposes.

In the context of copyright translations are just as permissible as elaboration, on the data gathered in *Disk Info* are based however only with unmistakable indication of source.

The rendition of excerpts from *Disk Info* in the context of club newspapers is expressly permitted, if these club newspapers are presented exclusively in printed form — if necessary the appropriate sections are to be attached to the disks in copy of the *Disk Info* pages.

If you wish to acquire an official version of *Disk Info* or you have suggestions on the contents, structure or you wish to recommend improvement, then please write to one of the following address:

Christopher Winter
xxxxxxxxxxxxxxxx
xxxxx xxxxxxxxxx

Harald Glaab
xxxxxxxxxxx
xxxxx xxxxxxxxxxx

Bitte fügen Sie bei Anfragen, zu denen Sie eine Antwort haben möchten, einen adressierten und frankierten Rückumschlag bei. Bitte vermeiden Sie Geldsendungen (statt Rückumschlag) und telefonische Kontaktaufnahme.

Please attach to any inquiries you which to be answered a stamped self addressed envelope. Please avoid sending cash instead of stamps and please, no phone calls.

2.1. Vorwort zur überarbeiteten dritten Auflage

Diese dritte Auflage entstand zu einer Zeit, als der TI auf dem besten Wege war, in die Bedeutungslosigkeit abzusacken. Möglich wurde diese überarbeitete Fassung nur durch den Einsatz eines hochleistungsfähigen PC sowie eines ebenfalls leistungsfähigen Textsystems. Hätte die inzwischen auf einen Umfang von 2 DSDD-Disketten angewachsene Datensammlung, als die sich *Disk-Info* darstellt, auf einem TI bearbeitet werden müssen, wäre sie nie zustande gekommen.

Das Kapitelsystem wurde beibehalten, weshalb es keine kapitelübergreifend fortlaufenden Seitennummern gibt. Ergebnis des PC-Einsatzes ist das umfangreiche Stichwortverzeichnis, welches das alte Sachwortverzeichnis mehr als ersetzt.

Das alles mag den einen oder anderen TI-Freak wundern oder gar schmerzen, doch es kann wohl kaum angehen, den TI in der heutigen Zeit als Maß aller Dinge anzusehen. Indes war und ist die in *Disk-Info* zusammengetragene Informationsmenge, ob sie nun zu 100% korrekt ist oder nicht, einfach zu wertvoll und arbeitsintensiv, um sie solch profanen Weltanschauungen zu opfern.

Bei dieser *Disk-Info* wurde erneut die Gratwanderung versucht, trockenen Stoff mit Beispielen aus der Praxis und launischen Bemerkungen etwas aufzulockern.

Wem dieses Konzept oder *Disk-Info* im allgemeinen gefällt, der ist aufgefordert (aber nicht verpflichtet), dem Autor einen kurzen Brief mit seiner Kritik zukommen zu lassen. Es sollte bedacht werden, dass Kritik sowohl positive als auch negative Meinungsäußerungen umfaßt!

Preface to the revised third edition

This third edition was developed at a time, when the TI at best appears to be headed towards the meaningless. This revised version became possible only by the employment of a high speed PC as well as a likewise efficient word processing system. During the time between the release of the second and third editions, the total volume of material had increased to fill two DSDD disks. If I had to create *Disk Info* on a TI at this point, this version would never have come in to being.

The chapter system was maintained, why there are no chapter-spreading sequential page numbers. Result of the PC employment is the extensive index, which replaces the old technical index to more than.

All this may surprise or hurt or other TI enthusiast, but it can probably hardly concern to regard the TI in the today's time as measure of all things. Meanwhile was and is the information capacity gathered in *Disk Info*, whether them are correct now to 100% or not, simply too valuable and labor intensively, in order to sacrifice them such everyday world views.

With this version of *Disk Info* the balancing act was tried again to up-loosen dry material with practical examples and capricious remarks somewhat.

Whom this concept or *Disk Info* pleases generally, that is requested (however not obligated), to give the author a short letter with its criticism. It should be considered that criticism covers both positive and negative expressions of opinion!

Auch oder gerade in dieser Fassung von *Disk-Info* gibt es noch etliche Themen, die nicht fertiggestellt oder so erschöpfend wie möglich beschrieben werden konnten. An dieser Stelle sei hierfür um Verständnis gebeten und die vage Hoffnung auf ein Update geweckt.

Die Vorworte zur zweiten und ersten Auflage wurden aus der Einleitung gestrichen — ihr Inhalt war nicht mehr gültig bzw. in seinem Neuigkeitswert inzwischen stellenweise überholt. Es sollte immer bedacht werden, dass *Disk-Info* inzwischen schon über 4 Jahre alt ist.

Obertshausen, im August 1990 Christopher Winter

2.2. Vorwort zur vierten Auflage für The Cyc CD von Mike Wright

Nachdem jetzt mittlerweile noch einmal fast 10 Jahr ins Land gegangen sind, fanden wir es an der Zeit, jetzt endlich allen verbliebenen TI-Benutzern in aller Welt die Informationen der *Disk-Info* zukommen zu lassen.

Daher wurde — mit ausdrücklicher Erlaubnis des Autors — eine nochmalige, mehrwöchige Überarbeitung der Textdateien vorgenommen, zumal es damals mit einem heute nicht mehr verfügbaren Textsystem erstellt wurde.

Daher bitte ich zu entschuldigen wenn eine Tabelle oder Graphik nicht so dargestellt wird, wie es vielleicht in der gedruckten Version zu sehen war.

Das bisher verwendete Kapitelsystem und die dazugehörige Seitennummerierung konnte nicht mehr wieder hergestellt werden, so dass entgegen der Vorstellungen des Autors wieder auf eine fortlaufende Nummerierung zurückgegriffen werden mußte. Dies auch im Hinblick darauf, dass es mit großer Wahrscheinlichkeit keine weiteren Updates der

Also or straight in this version of *Disk Info* gives it still some topics, which could not be finished or be described as exhaustively as possible. Here is for this for understanding asked and vague hope for an update waked.

The prefaces to the second and first edition were deleted from the introduction — their contents were not valid respectively in its piece of news value in the meantime outdated in parts any longer. It should be always considered that *Disk Info* is in the meantime already over 4 years old.

Overview, August 1990, Christopher Winter.

Preface to the fourth edition for The Cyc CD of Mike Wright

After nearly ten years, we would like to make available an English translation to all the remaining TI users in the world, the information contained in *Disk Info*.

Therefore became — with express permission of the author — a repeated, revision of several weeks of the text files made, particularly since it was not provided at that time with one today any longer available word processing system.

Therefore please excuse if a table or a graphics is not presented in the same way as it was in the printed version.

The chapter and pagination system used could no longer by retained or repaired, so that against the conceptions of the author, it is again back to sequential numbering. Please keep in mind that it is with great probability that no further additions or updates to *Disk Info* will be made.

Disk-Info mehr geben wird.

Aus Gründen der Aufwandsminimierung wurde auf das Stichwortverzeichnis ganz und gar verzichtet.

Aschaffenburg, im Oktober 2000 — Harald Glaab

2.3. Hinweise zu verwendeten Begriffen und Methoden

In dieser Diskinfo werden einige neue Begriffe eingeführt und einige vielleicht bekannte in einem anderen Sinn verwendet. Der notwendigen einleitenden Klärung sollen die folgenden Absätze dienen

2.3.1. Der Begriff der DSR

Der zentrale Angriffspunkt für Mißverständnisse ist das Kürzel DSR, was ursprünglich für Device-Service-Routine(s) steht. In der Regel bezeichnet dies die Steuerungsroutinen, welche zur 'Bedienung' eines Peripheriegerätes nötig sind. Aufgrund der Komplexität der Funktionen bei der Bedienung von Diskettenstationen wird DSR hier als Synonym für die *Gesamtheit* aller Programme einer Diskettensteuerungssoftware benutzt, so dass sich der weniger versierte Anwender unter DSR am Besten den Speicherchip der Controllerkarte vorstellt.

2.3.2. Level-0, 1, 2-Routinen

Die Einteilung der DSR in Routinen unterschiedlicher Komplexitäts- und Funktionalitätsstufen verlangte, hierfür ein Bezeichnung zu definieren. Die Programmteile der DSR, welche hart an der Hardware arbeiten und demzufolge Funktionen geringer Komplexität erfüllen, werden Level-0, jene, die auf File-Ebene ganze Informationsgefüge bearbeiten, werden Level-2-Routinen genannt. Die dazu benutzten Dienstprogramme der DSR,

For these reasons and reasons of expediency this edition does not have an index as the printed version did.

Aschaffenburg, October 2000 — Harald Glaab

Notes regarding terms and methods used

Some terms introduced here may be familiar, others may be new or new in this context. The following paragraphs serve as a necessary introduction and clarification.

The term DSR

The term DSR is a key term used throughout this document. The term DSR is an acronym for Device Service Routine. Usually this designates the control routines, which are necessary for the operation of a peripheral device. Due to the complexity of the functions performed during the operation of the floppy subsystem, DSR is used here as synonym for the *whole* of all the programs used for the disk controller, so that the less experienced user under DSR at the best one imagines the memory chip of the controller card.

Level 0, 1, 2 routines

Level-0, 1, 2 are the organization of the DSR into routines of different complexity and functionality stages required to define for this designation. The program sections of the DSR, which work hard on the hardware and therefore fulfil functions of small complexity, become Level 0. Those, which work on whole information structures on file level, are called Level 2 routines. The utility programs of the DSR, which take over the role of mediator to the

welche die Vermittlerrolle zur physikalischen Datenstruktur übernehmen, werden folglich Level-1-Routinen genannt. Die Routinen der Stufen (Levels) 1 und 2 sind extern aufrufbar, die Level-0-Routinen müssen ggf. anhand der in den Kapiteln 18ff beschriebenen ROM-Listings nachprogrammiert werden (von direkten Einsprüngen ist dringend abzuraten!).

2.3.3. Bit-Byte-Word-Notation / Bit-Nummerierung

Bei der Nummerierung von Bits, bekanntlich der Basiseinheit, in welcher ein Computer 'denkt', hat Texas Instruments selbst für ziemliche Verwirrung gesorgt.

Üblich ist die Nummerierung von Bits eines Bytes, Words oder was auch immer so geregelt, dass die Bit-Nummer die Wertigkeit im dualen Zahlensystem darstellt. So ist Bit 0 das Bit mit der niedrigsten Wertigkeit 1, Bit 7 eines Bytes etwa dasjenige mit der Wertigkeit 128.

Nicht so bei TI (oder nicht immer, wie es der Kenner verschiedener Veröffentlichungen aus dem Hause Texas Instruments kennt). Hier nummeriert man nach dem Verfahren Links-Nach-Rechts, womit also das höchstwertige Bit ganz und gar nicht standesgemäß die Nullnummer ist. Dass dieses Verfahren so unsinnig nicht ist, wird weiter unten vielleicht deutlicher.

Das Bit mit der höchsten Wertigkeit wird MSB (Most Significant Bit), das mit der niedrigsten LSB (Least Significant Bit) genannt.

Wenn es um Bytes und Words (Worte) geht, so nimmt das Chaos leicht zu. Da die TMS9900 ein 16bit-Mikroprozessor ist, ist sie in der Lage, 2 Bytes auf einmal als ein Word (Wort) anzusprechen. Beim TI wird durch eine

physical data structure, used for it, therefore are called Level 1 routines. The routines of the stages (levels) 1 and 2 are externally callable, which must Level-0 Routine if necessary on the basis the ROM listings described in Chapter 18 are after-programmed (direct re-entry points is to be advised against urgently!).

Bit-Byte-Word notation / Bit numbering

As is well known, bits and bytes are the fundamental unit in which a computer "thinks." Because of Texas Instruments conventions, this has caused considerable confusion.

By standard convention the numbering of the bits of a byte has the bit number representing the priority in the binary number system. So bit 0 is the bit with the lowest priority and represents a 1. Bit 7 of a byte is the bit with the highest priority and represents the numerical value of 128.

Not so with TI (or not always, as the connoisseur of different publications from Texas Instruments knows). Here one numbers the bits from left to right. Thus the most significant bit is not according to rank the zero-number. The fact that this convention is not so unreasonable will become clear as we go on.

The bit with the highest priority is called MSB (Most Significant Bit), that bit with the lowest priority is called the LSB (Least Significant Bit).

Since the TMS9900 is a 16-bit microprocessor, it is able to address 2 bytes at one time as a word. The TI console sees all memory outside the console in 8-bit widths, but the CPU sees things differently. It can address from a word two

entsprechende Hardware in der Konsole zwar ein äußerlicher 8-Bit-ter daraus, doch die CPU sieht das anders. Sie kann aus einem Word heraus noch ein einzelnes Byte adressieren, das Low- oder das High-Byte. Das konfuse daran ist, dass das High-Byte zwar das Byte eines Words ist, welches die höhere Wertigkeit hat (analog zu den Bits), das High-Byte jedoch an der niedrigeren Speicheradresse im 8bit-Adreßbereich des TI liegt — das Pseudo-Adressbit A15 ist beim High-Byte logisch 0. In diesem Fall wäre also die umgekehrte TI-Zählweise für einzelne Bits taktisch klüger.

2.3.4. Bit-Byte-Word-Struktur

Im folgenden wird die Bit-Byte-Word-Struktur graphisch dargestellt.

individual bytes, the low or high byte. The confusion occurs because the high byte is the byte of a word, which has the higher priority (similar to the bits), which is to high byte however because of the lower storage address in the 8-bit address range of the TI — the pseudo address bit A15 is with the high byte logic 0. In this case, TI's reverse counting method for individual bits would be tactically more intelligent.

Bit-Byte-Word Structure

In the following the bit-byte-word structure is shown graphically.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSB	High byte						LSB	MSB	Low byte						LSB
MSB	Word														LSB

2.3.5. Unsaubere Programmierung

Verschiedentlich ist in dieser *Disk-Info* von 'Unsauberer Programmierung' die Rede. Dies mag auf den Uneingeweihten wie Arroganz wirken, was es aber keinesfalls sein soll. Unsaubere Programmierung findet man überall da, wo sich Programmierer von Anwendersoftware nicht an die Richtlinien des Herstellers der benutzten Hard- und / oder Software halten. Im Grundsatz fallen alle Versuche, das Betriebssystem zu hintergehen in diese Kategorie. Speziell beim TI ist dieser Begriff so zu interpretieren, dass sich diverse Hersteller von DSR-Software nicht an die TI-Konventionen bzw. -Vorschriften halten, und so Inkompatibilitäten mit anderer Hard-Software provozieren. Die bekanntesten Vertreter sind fast alle Karten der Firma Myarc sowie die früheren CorComp-Diskcontroller mit eigenem

Careless Programming

Careless programming is a term used in *Disk Info* to describe programmers that do not follow standards. This may appear to non-technical users as arrogance which it should not be interpreted as under any circumstance. In principle this includes all techniques that attempt to deceive the operating system. Particularly with the TI this term can be applied to various manufacturers of DSR software, which do not adhere to TI's standards and conventions. Standards should always be followed or you can expect incompatibilities with other hardware. The most well-known representatives are almost all cards of the company Myarc as well as the earlier CorComp disk controllers with own title page, which made life difficult for those owning these types of disk controllers. Since ramdisks are relatively new

Titelbild, welche so manchem aufgerüsteten TI das 'Leben' schwer machten. Seit kurzer Zeit gehören einige Ramdisks dazu, deren DSR-Schmiede (es ist manchmal fast anmaßend, von Programmierern zu reden...) es als innovativ ansehen, festgelegte DSR-Funktionen nach eigenem Gutdünken zu erweitern, ohne einen Standard zu beachten.

2.3.6. Der PAB

Wenn es um Disketten und Diskettensteuer-software geht, dann führt am PAB kein Weg vorbei. Es handelt sich hier um eine wohldefinierte Gruppierung von Daten, welche über das VDP-RAM an eine DSR übergeben wird. Die DSR protokolliert hier ihre Aktionen und erlaubt so den Datenaustausch mit jedem beliebigen Anwenderprogramm.

2.3.7. Seitennummern

Da *Disk-Info* als offenes Dokument verstanden wird, sollten ggf. notwendige Updates möglichst einfach durchgeführt werden können. Da zudem die Auffindbarkeit einzelner Themen oder Stichworte mittels einer Kapitelnummerierung wesentlich erleichtert wird, besitzt jedes Kapitel eine eigene Nummer und eine Zählweise der Seiten ab 1 bei jedem Kapitelwechsel. Inhalts- und Stichwortverzeichnis entsprechen diesem Muster; Einträge im Inhaltsverzeichnis werden jedoch nicht mit der Abschnitts- und Hierarchiestufe versehen, um bei einem Update keine Verwirrung zu verursachen.

2.3.8. Physikalischer Aufbau

Der Aufbau einer regulären Ausgabe von *Disk-Info* erlaubt es, illegale Kopien leicht zu identifizieren. Die Bindung erfolgt mittels Kunststoff-Bindeleisten mit Rechtecklochung. Als Schutzklappen werden transparente Kunststoffolien verwendet. Zwischen zwei

and have only been around for a short time to some ramdisks, whose DSR framework (sometimes it is almost arrogant to consider from programmers to talk ...) regard it as innovative to extend determined DSR function after own discretion without a standard.

The PAB

If it concerns diskettes and diskette control, then no way leads past the PAB. It concerns here a well-defined grouping of data, which is transferred over VDP RAM to a DSR. The DSR logs here its internal messages and permits the data exchange with any user program

Page numbers

Since *Disk Info* is understood as open document, should updates become necessary, it should be easy to replace the necessary pages. Since besides the discovery of individual topics or glossary words is substantially facilitated by means of chapter numbering, each chapter possesses its own number and a counting method of the pages starting from 1 with each section change. Contents and glossary correspond to this sample; Entries in the table of contents will not provide however with the paragraph and hierarchic level, in order to cause with an update no confusion.

Physical structure

The structure of a regular output of *Disk Info* makes it possible to identify illegal copies easily. The linkage takes place by means of plastic binding borders with rectangle punching. As protection flaps transparent plastic foils are used. Between two chapters the chapter name is

Kapiteln befindet sich jeweils ein mit dem Kapitelnamen bedrucktes, blau eingefärbtes Blatt Papier, Inhalts- sowie Stichwort-verzeichnis sind auf rosafarbenem Papier gedruckt. Jedes Kapitel besitzt eine gerade Anzahl von Seiten.

printed, blue dyed page a paper, a contents as well as a glossary is printed on pink colored paper in each case. Each chapter possesses an even number on the part of.

3. Einleitung

Die TI-Konsole und ein Kassettenrecorder — mit dieser Geräteanordnung haben wohl die meisten Besitzer eines TI-99/4A begonnen. Die Einschränkungen aber, die beim Betrieb eines Kassettenrecorders am TI bestehen, wie limitierte Programmlänge (ca. 13 KByte), kaum Maschinenspracheprogramme verfügbar und besonders die lange Lade- und Speicherzeit werden wohl sehr bald den Wunsch nach einem zeitgemäßeren Datenspeicher geweckt haben.

In diesem Punkt wird wieder einmal die Marktstrategie von Texas Instruments deutlich, mit welcher der 99er unters Volk gebracht werden sollte: erst mit der Konsole relativ preiswert ins Computern einsteigen, nur um dann den Gegenwert eines gebrauchten Mittelklassewagens investieren zu müssen, bis sich die Anlage auch wirklich Computer nennen darf.

In der Tat ist der TI-99/4A erst mit Diskettenlaufwerk und Speichererweiterung konkurrenzfähig, was sicher für den marktwirtschaftlichen Schiffbruch von TI mit verantwortlich war.

Die Leistungsfähigkeit von Floppy-Disk Laufwerken und dem Speichermedium 'Flexible Disk' liegt wohl in der Hauptsache in der recht hohen Übertragungsgeschwindigkeit begründet, daneben aber auch in der Möglichkeit, per Software direkt die Daten auszusuchen, mit denen zum beliebigen Zeitpunkt gearbeitet werden soll. Bezüglich gerade der Transfergeschwindigkeit kann der TI auch 1986, 3 Jahre nach Produktionseinstellung, immer noch im Homecomputerbereich als Vorbild angesehen werden. Dies liegt auch daran, daß TI sich hier auf kein nebulöses, hausgemachtes System verlassen hat, sondern konsequent nach Industriestandard vorging.

Als Besitzer einer TI-99/4A-Anlage ist man

Introduction

Probably most TI owners started out with a console and tape recorder. This posed several restrictions: The size of program could only be approximately 13 Kbytes. Machine language programs are almost non-existent. Long loading and savings time. This combination of limitations caused many users to quickly desire a faster method of program recording and retrieval.

At this point the marketing strategy of Texas Instruments becomes clear. TI's goal was to hook 99er's with a relatively inexpensive console as an entry level investment. This is the equivalent of buying a used car in the middle class range.

Indeed the TI-99/4A is only competitive with floppy disk drive and memory expansion, which was with responsible surely for the free market shipwreck of TI.

The efficiency of floppy diskette drives and the storage medium flexible disk is in the main quite quick, in addition the ability to quickly jump directly to information you need on a floppy disk makes floppy disks quite attractive, compared to slower and less flexible based tape systems. Even three years after TI halted production of the Home Computer (1986) the TI Home Computer can be considered a model of speed. This is because TI didn't rely on some poorly designed home made system, but consistently applied industry standards.

As an owner of a TI-99/4A system one is thus in

somit in der glücklichen Lage, prinzipiell beliebige auf dem Markt erhältliche Diskettenlaufwerke an die diversen Diskcontroller anschließen zu können.

Die Leistungsfähigkeit der Massenspeichertechnik bewirkt jedoch auch, daß der Überblick über Funktionen und Funktionsabläufe schon bei Disketten zunehmend eine Sache für Fachleute wird. Gerade im Bereich der Homecomputer ist es aber üblich, das Innenleben seines Rechners verstehen zu wollen. Soweit es das Feld der Diskettenoperationen betrifft, soll diese *Disk-Info* die vorhandenen Lücken in der Information über den TI-99/4A in kompakter und umfassender Form schließen, ohne dabei andauernd auf andere Publikationen zu verweisen, die dann (Murphy sei Dank) gerade nicht verfügbar sind.

Zum Schluß sei eines noch angemerkt: Die dem durchschnittlichen Anwender normalerweise unzugängliche Möglichkeit, einzelne Sektoren auf der Diskette zu lesen, zu interpretieren und zu manipulieren, schaltet bisher wirkungsvolle Kopierschutztechniken aus. Der Leser und Anwender dieser *Disk-Info*-Blätter ist angehalten, sich selbst Gedanken darüber zu machen, was sinnloses Raubkopierertum gerade dem empfindlichen Softwaremarkt des TI zufügt. Geringe Absatzzahlen haben schon mehr als einem Programmierer die Lust genommen!

the lucky position to be able to attach any available floppy disk drives on the market to the various disk controllers.

The efficiency of the mass storage technique causes however also that the overview of functions and sequences of functions becomes already with diskettes increasing a thing for specialists. Even one within the area of the home computers is however usual it to want to understand the interior life of its computer. As far as it concerns the field of the diskette operations, this *Disk Info* is to close the available gaps in the information about the TI-99/4A in compact and global form, without referring thereby continuously on other publications, which are not available then (Murphy is thanks) even.

In the end one is still marked: Those the average user normally inaccessible possibility of reading individual sectors on the diskette of interpreting and of manipulating, switches effective copy protection techniques off. The reader and user of these *Disk Info* pages are stopped to think over it what causes senseless elite programmers even to the sensitive software market of the TI. Low sales figures surely dampened the enthusiasm of more than one programmer!

3.1. Grundlegende Mechanismen und Konzepte des TI-DSR-Systems

DSRs beziehen sich immer auf externe Erweiterungen (Peripherien) des TI-Systems, die zur Funktion eigene Programme benötigen, welche der TI-Monitor im ROM und GROM der Konsole nicht liefert.

Jede Karte in der (Peripheral-Expansion-Box, Peripherie-Erweiterungs-Box) (PEB), die irgendwelche Verwaltungsarbeiten aktiv übernimmt (RS232, Diskcontroller, Video-Karte, Ramdisk, etc.) benötigt eine DSR.

Dabei stellt jede DSR ein sog. GERÄT zur Verfügung, das einen einzig artigen Namen haben muß, durch den es identifiziert wird. Beim Zugriff auf ein bestimmtes Gerät wie einen Drucker oder eine Datei auf dem Gerät DISKETTENSTATION muß also der Name dieses Gerätes bekannt sein und angegeben werden.

Diese Daten werden zusammen mit der gewünschten Operation (Lesen, Schreiben, Öffnen, Schließen, etc.) und einigen anderen Daten codiert in einem sog. PAB hinterlegt (Peripheral Access Block, Peripherie-Zugriffs-Block). Dieser PAB wird meist im CPU-RAM aufgebaut und kurz vor Aufruf der Gerätesuchroutine ins VDP-RAM kopiert.

3.1.1. Die DSRLNK-Routine

Die Adresse dieses PAB, der immer im VDP-RAM liegt, wird an eine Routine mit Namen DSRLNK übergeben (Device-Service-Routine-Link), welche das Gerät selbständig sucht und die entsprechende Funktion veranlaßt.

Diese Routine ruft die entsprechenden Programme der passenden DSR auf, wobei sie die passenden Adressen aus dem nach bestimmten Regeln aufgebauten sog. DSR-

Fundamental mechanisms and concepts of the TI DSR system

DSRs always refer to external devices (peripherals) of the TI system, which contain their own programs, and which the TI monitor in the console ROMs and GROMs does not provide support for.

Each card in the Peripheral Expansion Box (PEB), which can take control of the system (RS232, disk controller, video card, ramdisk, etc.) requires a DSR.

Each DSR places so-called DEVICE for the order, which must have a only one well-behaved name, by which it is identified. With the access to a certain device like a printer or a file on the device FLOPPY STATION must thus the name of this device admits to be and be indicated.

These data become together with the desired operation (reading, writing, opening, closing, etc.) and some other data coded in one so-called PAB deposits (Peripheral Access Block, peripheral device access block). This PAB is usually structured in the CPU RAM and copied briefly before call of the device search routine in VDP RAM.

The DSRLNK routine

The address of this PAB, which is always stored in VDP RAM, is transferred to a routine called DSRLNK (Device Service Routine Link), which looks up the device independently and which appropriate function arranges.

This routine calls the appropriate programs of the suitable DSR, whereby it so-called the suitable addresses from according to certain rules the structured. DSR Header (see there)

Header (siehe dort) ermittelt, und kehrt danach zum rufenden Programm zurück, dem es eine Rückmeldung liefert, wie die Funktion ausgeführt wurde. Sie meldet auch, wenn der angegebene Name zu keiner im System vorhandenen Gerätesteuersoftware (DSR) paßt.

Der weitere Datenaustausch findet ausschließlich anhand des PAB statt, der weiter unten beschrieben wird.

Alle DSRs belegen den Speicherbereich >4000 - >5FFF bzw. können maximal diesen Bereich nutzen. Größere DSRs arbeiten mit Paging, wobei mehrere Seiten eines oder mehrerer ROMs in den zugewiesenen Adreßbereich eingeblendet werden.

Die Auswahl der jeweiligen DSR erfolgt mittels sog. CRU-Seiten. Der CRU-Bereich ist ein spezieller I/O-Bereich der CPU TMS9900, auf den mit reservierten Befehlen zugegriffen wird. Vereinfacht kann man beim TI-99/4A sagen, daß der Adreßbus 3 weitere Bits erhält. Ursprünglich waren diese Bits übrigens statisch gepuffert in der Box vorgesehen.

Jede CRU-Seite einer PE-Box-Karte umfaßt 128 Bits, wobei das erste (Bit 0) zum ein- und ausschalten der jeweiligen DSR benutzt wird.

DSRLNK schaltet also nacheinander jede Karte ein, prüft auf den Gerätenamen und schaltet bei negativem Ergebnis die aktuelle Karte aus und ggf. die nächste an. Wurde der gesamte CRU-Bereich (>1000 bis >1F00 in Schritten zu >100) abgesucht, so liegt ein I/O-Error-0 vor; das Gerät existiert im System nicht. Bei der Zählweise ist zu beachten, daß CRU-Bits bzw. deren Adressen erst durch Ignorieren von A15 entstehen, über das die 16-bit-CPU TMS9900 ja eigentlich nicht verfügt, und A15 eigentlich A0 ist, Texas-Instruments jedoch früher immer falsch 'rum gezählt hat (entgegen der Wertigkeit eines Bits).

determined, and returns to it to the calling program, to which it supplies with an acknowledgment, as the function was executed. It also reports if the requested name cannot be found in any DSR in the system.

Further data exchange exclusively takes place on the basis the PAB, which is further below described

All DSRs occupy memory locations >4000-->5FFF respectively, and can use all of this area. Larger DSRs operate with paging, whereby several pages or ROMs are paged into the assigned address range.

The selection of the respective DSR is effected by means of so-called CRU pages. The CRU area is a special I/O area of the TMS9900 CPU, which with reserved instruction one accesses. Simply, one can say with the TI-99/4A that the address bus receives 3 further bits. Originally these bits were by the way statically buffered in the box intended.

Each CRU page of a PEB card is allocated 128 bits. The first bit (0) switches the respective DSR on and off.

DSRLNK switches thus successively each card on, checks on the device names and switches with negative result the current card off and if necessary the next on. If the entire CRU area (>1000 to >1F00 in steps of >100) was searched, then I/O Error 0 is returned; the device does not exist in the system. The counting method it is to be noted that CRU bits respectively develop their addresses only by ignoring A15, which the 16-bit TMS9900 CPU actually does not have, and A15 is actually A0, Texas Instruments however in former times always falsely 'rum (= herum = round and round) counted (against the priority of a bit).

3.1.2. Der PAB

Der PAB ist, wie bereits oben angeführt, das Bindeglied zwischen dem Daten anfordernden oder absendenden Programm und dem zugehörigen Gerät, welches die Daten verarbeitet.

Der PAB enthält den sog. I/O-Code, das File-Attribut, die Pufferadresse, die Anzahl zu lesender oder zu schreibender Bytes, den Geräte- und ggf. den Dateinamen sowie dessen Länge. Zusätzlich sind einige Bits für die Rückgabe von Statusinformationen durch die DSR selbst reserviert, sowie ein Byte, das es einer DSR erlaubt, ggf. selbsttätig Meldungen auf den Schirm zu bringen, die auch sichtbar sind (Screen-Offset, wird jedoch nur von GPLLNK benutzt).

Werden Geräte- und Dateinamen verwendet, müssen diese mit einem Punkt getrennt werden. Nachfolgende Trennsymbole für Optionen des jeweiligen Gerätes werden von der DSR vorgegeben, müssen also keine Punkte mehr sein, obwohl dies sicher die sauberste Lösung wäre.

Der PAB muß im VDP-RAM liegen, sobald DSRLNK aufgerufen wird. Um der DSRLNK-Routine die Suche des Gerätes zu ermöglichen, zeigt das Wort an Adresse >8356 auf die Position des Bytes mit der Namenlänge des Geräte- und ggf. Dateinamens. Dieser Pointer *muß* gesetzt werden.

Da der PAB den Informationsfluß zwischen Anwenderprogramm und DSR steuert (ein Zugriff direkt auf das VDP-RAM anhand der in Kapitel 8 beschriebenen Adressen ist unbedingt zu vermeiden), muß er für die gesamte Zeit, in der das betreffende File offen ist, im VDP-RAM vorliegen. Er darf nur in den Parametern für die zu verschiebende Byteanzahl, die Satznummer, die Pufferadresse sowie den I/O-Modus und das

The PAB

The PAB is, as mentioned above, the link between that data requesting or dispatching program and the appropriate device, which process the data.

The PAB contains so-called I/O-Code, the file attribute, the buffer address, the number more reading or writing bytes, device and if necessary the file names as well as its length. Additionally some bits for the return of status information are reserved by the DSR, as well as a byte, which permits it to a DSR, usually automatically messages on the screen to bring, are also visible (screen offset, however only used by GPLLNK).

If device and file names are used, these must be separated by a period. Following one Data delimiters for options of the respective device are given of the DSR, must be thus no more points, although this would be safe the cleanest solution.

The PAB must be situated in the VDP RAM, as soon as DSRLNK is called. In order to enable to the DSRLNK routine the search of the device, the word shows at address >8356 to the position of the byte with the name length of the device and if necessary file name. This pointer *must* be set.

Since the PAB controls the information flow between user program and DSR (an access to VDP RAM on the basis the addresses described in Chapter 8 is direct to absolutely avoid), must do it for the entire time, in which the file concerned is open, are present in the VDP RAM. It may be modified only in the parameters for the number of bytes, the record number, which can be shifted, the buffer address as well as the

Fehlerbyte modifiziert werden; Gerätename und Satzlänge sind tabu!

3.1.2.1. Beispiel für einen PAB

Byte 0-1	I/O Op-code	Flag / Status
Byte 2-3	Adresse Datenpuffer im VDP	
Byte 4-5	Satzlänge	Zeichenanzahl
Byte 6-7	Satznummer (0-32767) — Länge (Memory Image)	
Byte 8-9	Screen Offset	Namenlänge
Byte 10+	Dateiname	

I/O-Opcode

Dieses Byte gibt die Operation an, die durchzuführen ist:

I/O mode and the error byte; Device name and record length are taboo!

Example of a PAB

Byte 0-1	I/O op-code	Flag / Status
Byte 2-3	Data Buffer Address (in VDP)	
Byte 4-5	Logical Record Length	Character Count
Byte 6-7	Record number (0-32767) — Length (Memory Image)	
Byte 8-9	Screen Offset	Name length
Byte 10+	File descriptor	

I/O op-code

This byte indicates the operation, which is to be executed:

I/O Modus *I/O mode*	*Funktion* *Function*	*Beschreibung*	*Description*
0	Open	Datei öffnen wie im Flag-Byte angegeben	Open file as indicated in the flag byte
1	Close	Datei schließen	Close file
2	Read	Datensatz lesen	Read data record
3	Write	Datensatz schreiben	Write data record
4	Restore/Rewind	Dateizeiger auf Dateianfang setzen	Set file pointer to start of file
5	Load	Laden eines Memory-Image-Files	Load memory-image file
6	Save	Speichern eines Memory-Image-Files	Save memory-image file
7	Delete	Löschen einer Datei(!)	Delete file (!)

I/O Modus *I/O mode*	*Funktion* *Function*	*Beschreibung*	*Description*
8	Scratch record	Löschen eines Datensatzes (bisher in keiner DSR unterstützt!)	Delete record (not supported by any DSR at present)
9	Status	Ermitteln von Informationen zur Struktur	Determine information from structure

Flag / Status

Hier ist eine Bitmaske abgelegt anhand der die DSR erkennt, um welche Art von File es sich grundsätzlich handelt. Die drei höchstwertigen Bits sind zudem für die Übermittlung von sog. File-Error-Codes reserviert. Der Aufbau der Bitmaske ist wie folgt:

Flag Status

Here is a filter stored on the basis that the DSR detects, which type of file it concerns basically. The three most significant bits are besides for the transmittal of so-called File Error Codes reserved. The structure of the filter is as follows:

Bit	*Beschreibung*
37 44 1	Fehlercode 101 Ungültiger Gerätename 001 Gerät Schreibgeschützt 110 Ungültiger Datentyp, Satzlänge, 010 Dateityp 111 Von DSR nicht unterstützter Modus 020 Gerät kann keine Daten mehr 0 aufnehmen Versuch über das Ende hinaus zu lesen Gerätefehler Dateifehler, Programm als Datei gelesen
4	Satzlänge 1 fest variabel
3	Datentyp 1 Display Internal
37 28 7	Operation 110 Update 11 Output Input Append
0	Dateityp 1 Sequent. Relativ

Bit	*Description*
37 37 7	Error code 101 Invalid device name 001 Device write protected 110 Invalid data type, record length, type 010 of file 111 Mode 020 Device not supported by DSR no 0 more data take up Attempt beyond the end to read Device errors Data errors, program than file can read
3	Record type 1 Fixed Variable
4	Data type 1 Display Internal
37 38 1	Mode of operation 110 Update 11 Output Input Append
7	File type 1 Sequential Relative

Für weitere Informationen lesen Sie bitte unter Ein-/Ausgabeoperationen weiter unten in diesem Kapitel. Dieses Byte wird zudem in leicht modifizierter Form in jedem sog. File-Directory-Sektor verwaltet, was im Kapitel 7 weiter ausgeführt wird.

For further information you read please under in- or output operations further down in this chapters. This byte becomes besides in easily modified form in everyone so-called File directory sector administers, which is continued to execute in Chapter 7.

Datenpufferadresse

Dieses Wort gibt an, ab welcher Adresse im VDP-RAM die zu schreibenden Daten abgelegt sind bzw. wo im VDP-RAM die aus der Datei zu lesenden Bytes an das Anwenderprogramm übergeben werden sollen.

Satzlänge

Anzahl Bytes in einem Datensatz fester Länge bzw. maximale Länge von Datensätzen variabler Länge.

Zeichenanzahl

Anzahl der Bytes eines Eintrages, z.B. aktuelle Satzlänge (Stringlänge o.ä.) beim Schreiben eines Satzes variabler oder fester Länge. Beim Lesen die Anzahl tatsächlich gelesener Zeichen im Datensatz (variable und feste Satzlänge).

Satznummer

Logische Nummer des zu lesenden/schreibenden Datensatzes einer Direktzugriffsdatei. Beim Laden eines Memory-Image-Files die Puffergröße und damit die maximale Filelänge; beim Schreiben die Anzahl zu schreibender Bytes eines Memory-Image-Files.

Screen-Offset

Optionaler Zeichenoffset, falls die DSR selbst Meldungen ausgeben muß. Derzeit wird dies nur von der Cassetten-DSR benutzt (und wer, der Diskinfo liest, speichert schon auf Cassetten …).

Namenlänge

Gesamtlänge des folgenden Geräte- und ggf. Filenamens (inkl. aller Sonderzeichen wie Punkte etc.).

Scratchpad memory address

This word indicates, starting from which address in the VDP RAM the data which can be written are stored respectively where in the VDP RAM from the file to reading bytes to the user program to be transferred are.

Record length

Number of bytes in a data record of fixed length respectively maximum length of data records of variable length.

Number of characters

Number of bytes of an entry, e.g. current record length (string length, or the like) during the writing of a set of variable or fixed length. During the reading the number of actually read characters in the data record (variable and fixed record length).

Record number

Logical number of the data record of a direct access file which can be read/written. While the loading memory image files the buffer size and with it the max. file length; during the writing the number writing bytes memory image files.

Screen offset

Optional character offset, if the DSR must output messages. At present this is used only of the cassette DSR (and which, which reads disk info., stores already on cassettes …).

Name length

Total length of the following device and if necessary File name (inclusive all special characters such as periods etc.).

Gerätename

Der erste Teil des Geräte- und Filenamen- bzw. Optionsstrings bezeichnet die DSR, die für diesen Vorgang zuständig ist. Dabei wird der Punkt ('.') als Trennsymbol bzw. Ende des Gerätenamens verwendet. Anhand dieses Namens sucht das DSRLNK die passende Karte in der Peripheriebox. Der ggf. vorhandene Rest dieses Strings wird von der DSR selbst ausgewertet. Der Gerätename ist von TI auf eine maximale Länge von 7 Zeichen begrenzt worden. Ist nur der Gerätename vorhanden, kann auf den Punkt an dessen Ende verzichtet werden.

Filename / Option

Nach dem Gerätenamen folgt bei einer Disk-DSR i.d.R. [in der Regel] ein Filename (Ausnahme Diskkatalog). Bei Schnittstellenkarten wie z.B. der RS232-Karte kann hiermit der Kanal konfiguriert werden.

3.1.2.2. Beispiel für einen Basic-PAB

Der Basic-PAB verfügt über 2 weitere Wörter (1 Wort + 2 Byte), welche primär als Verwaltungshilfen für Basic zu verstehen sind. Alle weiteren Einträge sind identisch zu den oben beschriebenen.

Bit 0-1	Link zum nächsten PAB	
Bit 2-3	Kanal- / Filenummer	Datenpuffer-offset
Bit 4-5	I/O - Opcode	Flag / Status

PAB-Link

Basic benötigt diesen Zeiger zur Verwaltung mehrerer offener Files. Alle PABs sind hiermit in einer einfach verketteten Liste angeordnet, beginnend mit dem zuerst geöffneten File. Dies

Device name

The first part of the device and file name respectively define option strings for the DSR, which is responsible for this process. The period becomes ('.') as data delimiter respectively end of the device name uses. On the basis this name looks the DSRLNK the suitable card up in the peripheral device box. The usually available remainder of this string is analyzed of the DSR themselves. The device name was limited by TI to a max. length of 7 characters. Only if the device name is available, can be done without the point at its end.

Filename/option

After the device name a file name with a disk DSR usually (exception disk catalog) follows. With interface cards e.g. the RS232 card can hereby the channel are configured.

Example of a Basic PAB

The Basic PAB has 2 further words (1 word + 2 byte), which are to be understood primarily as administrative assistance for Basic. All further entries are identical to described the above.

Bit 0-1	Link to the next PAB	
Bit 2-3	Channel/ file number	Scratchpad memory offset
Bit 4-5	I/O op-code	Flag/status

PAB link

Basic needs this pointer for the administration of several open files. All PABs is hereby in a simply chained list arranged, beginning with the file opened first. This simplified among other things.

vereinfacht u.a. die sog. Garbage-Collection, mit der unbenutzter Speicherplatz (z.B. nach dem Schließen einer Datei) wieder zugänglich gemacht wird. Der PAB-Link ist ein Zeiger auf die nächste PAB-Link-Adresse des nachfolgend geöffneten Files.

Kanal-/Filenummer

Dies ist die Filenummer, unter der Basic alle Zugriffe zwischen dem Öffnen und Schließen der Datei verwaltet.

Datenpufferoffset

Insbesondere bei sog. Schwebenden Ausgabebedingungen wird ein Datenpuffer erst dann auf Diskette geschrieben, wenn er voll ist bzw. überläuft. Dies gilt analog beim Lesen, wenn ein Datenpuffer schrittweise ausgelesen wird. Näheres zur Umsetzung von blockstrukturierten Geräten finden Sie in Kapitel 4.

3.1.3. Ein-/Ausgabeoperationen

Beim TI-99/4A wird grundsätzlich zwischen zwei verschiedenen Dateitypen unterschieden:

- en sequentiellen Dateien mit dem Sonderfall P-File,
- und den Dateien mit direktem bzw. wahlfreiem Zugriff, auch als Relative Dateien bezeichnet.

Die sequentiellen Dateien werden von fast jedem Ein-/Ausgabegerät unterstützt (Drucker, serielle Kommunikation etc.), die Dateien für wahlfreien Zugriff (sog. Random-Files) sind dagegen eine Spezialität des Diskettenbetriebssystems.

Folgende Schlüsselworte werden im Filesystem benutzt, um Daten- und Dateitypen zu beschreiben:

- Datentypen: DISPLAY oder INTERNAL
- Satzlänge: VARIABLE oder FIXED

so-called Garbage Collection, with which unused storage space (e.g. after closing a file) is made again accessible. That PAB link is a pointer to the next PAB link address in the following opened files.

Channel/file number

This is the file number, under which Basic administers all accesses between opening and closing the file.

Scratchpad memory offset

With so-called in particular floating conditions of issue a scratchpad memory is only then written on diskette if it is full or overflows. This applies similarly during the reading, if a scratchpad memory is gradually selected. You find details for the conversion of block-structured devices in Chapter 4.

Input/output operations

With the TI-99/4A one differentiates basically between two different file types:

- the sequential files with the special case p-file,
- and the files with direct or. optional access, also as contiguous files defines.

The sequential files are supported by almost each in- or output device (printer, serial communication etc.), the files for optional access (so-called Random files) are against it a specialty of the diskette operating system.

The following keywords are used in the file system, in order to describe data and file types:

- Data types: DISPLAY or INTERNAL
- Record length: VARIABLE or FIXED

- Operation: I N P U T , O U T P U T , APPEND oder UPDATE

3.1.3.1. Datentypen

Das TI-File-System unterscheidet nicht zwischen reinen ASCII- und sog. Binärdaten (bezeichnet mit den Datentypen DISPLAY und INTERNAL), da es für die Speicherung auf einem Datenträger unerheblich ist, ob jedes Byte nun den gesamten Wertebereich von 0 - 255 umfaßt oder dieser Wertebereich nur teilweise genutzt wird. Überall da, wo ein solcher Unterschied auftritt, ist es eine Spezialität z.B. des Basic-Interpreters o.ä.

Zusätzlich steht noch eine Sonderdateiform zur Verfügung, das sog. Memory-Image-Format, auch als P-File-Format bezeichnet. Hierbei wird ein zusammenhängender, meist unstrukturierter Block von Daten als Speicherabzug an einem Stück auf Diskette geschrieben bzw. von ihr gelesen. Der wesentliche Vorteil dieses Formats ist die hohe Lese- bzw. Schreibgeschwindigkeit

3.1.3.2. Satzlängen

Bei der Eröffnung einer Datei im Ausgabemodus muß die Satzlänge vorgegeben werden, bei der Eröffnung einer Datei im Eingabemodus ist dies abhängig von der Diskcontroller-DSR — nicht nötig, da eine Länge von Null (0) vorgegeben werden kann und dann die DSR selbst die Länge in den PAB einsetzt.

Sequentielle Dateien können aus Datensätzen variabler oder fester Länge bestehen. Auf Datensätze einer sequentiellen Datei kann immer nur in aufsteigender Reihenfolge zugegriffen werden, da das File-System nach außen keinen Byte-Zeiger auf die Datei zur Verfügung stellt (der jedoch von der DSR verwaltet wird). Ein Zugriff auf irgendeinen vorhergehenden Datensatz ist somit nur möglich, wenn vom ersten Datensatz an

- Operation: I N P U T , O U T P U T , APPEND or UPDATE

Data types

The TI file system does not differentiate between pure ASCII and so-called Binary-coded data (DISPLAY names and INTERNAL) the data types, since it is insignificant for storage on a data carrier whether each byte now the entire scope of 0-255 is covered or used this scope only partly. Anywhere, where such a difference occurs, it is a speciality e.g. the Basic Interpreter.

Additionally still another special file type is available, the so-called memory image format, also as PROGRAM file format marks. Here a connected, usually unstructured block of data is written as memory dump at a piece on diskette or of it read. The substantial advantage of this format is the high reading or writing rate.

Record lengths

With the initialization of a file in the output mode the record length must be given, with the initialization of a file in the input mode is this dependent on the disk controller DSR — not necessary, since a length can be given of zero (0) and then the DSR the length into the PAB begins.

Sequential files can consist of data records of variable or fixed length. Data records of a sequential file can be accessed in each case in ascending order, since the file system makes outward no byte pointer available on the file (however of the DSR one administers). An access to any preceding data record is thus possible only if from the first data record either one reads or one writes, until the requiring is achieved.

entweder gelesen oder geschrieben wird, bis der Gewünschte erreicht ist.

Voraussetzung für den Einsatz von Dateien mit wahlfreiem Zugriff ist die eindeutige Lokalisierbarkeit eines jeden Datensatzes. Hierzu muß jeder Datensatz die gleiche Länge haben. Somit kann das Betriebssystem feststellen, in welchem Sektor und an welcher Stelle im Sektor ein bestimmter Datensatz, dessen Nummer angegeben wird, zu finden ist.

Abschließend bleibt zu bemerken, daß eine Datei explizit als aus Datensätzen gleicher Länge aufgebaut deklariert werden muß. Es reicht nicht, implizit eine Datei mit Datensätzen fester Länge dergestalt aufzubauen, daß sie im VARIABLE-Format aufgebaut wird und ausschließlich z.B. mit Leerzeichen aufgefüllte Datensätze einer einheitlichen Länge geschrieben werden. Auf mögliche Tricks, wie das Eröffnen einer solchen Datei als Direktzugriffsdatei mit einer Länge +1 soll hier nicht weiter eingegangen werden.

3.1.3.3. Operationen

Input- und Output-Modus dürfen als die elementaren Ein-/Ausgabeoperationen angesehen werden. Eine Datei, welche im Input-Modus geöffnet wird, kann nur gelesen werden, jeder Schreibversuch bewirkt eine Fehlermeldung des DOS.

Eine im Output-Modus geöffnete Datei wird, sofern Sie existierte, gelöscht und neu angelegt.

Der Append-Modus stellt eine Sonderform des Schreibens in eine sequentielle Datei dar. Dabei werden Datensätze ausschließlich am aktuellen Ende der Datei angehängt.

In diesem 3 Modi existiert außerhalb der DSR kein Datensatzzeiger. Diesen verwaltet die DSR intern, womit z.B. beim sequentiellen Zugriff

A prerequisite for the use of files with optional access is the unique ability to locate the data record. For this each data record must have the same length. Thus the operating system can determine, in which sector and in which place in the sector a certain is to be found data record, whose number is indicated.

It remains locking noticing that a file must be defined explicitly as from data records of same length structured. It is not enough to structure implicitly a file with data records of fixed length such that it is structured in the variable format and exclusively e.g. with blanks filled up data records of a uniform length are written. On possible cheat, as opening such a file is not as direct file with a length +1 here is continued to enter.

Operations

Input and output mode may be regarded as the elementary in- or output operations. One file, which is opened in the input mode, can be only read, each write attempt causes an error message of the DOS.

A file opened in the output mode is deleted, if it existed, and the it is recreated.

The Append mode represents a special form of writing into a sequential file. Data records are attached exclusively at the current end of the file.

In these 3 modes no data record pointer exists outside of the DSR. The DSR administers these internally, with which e.g. with the sequential

immer auf die aktuelle Dateiposition für die folgende Lese- oder Schreiboperation gezeigt wird.

Der Update-Modus ist nur für Dateien mit wahlfreiem Zugriff bzw. solche mit festen Satzlängen zulässig. Zum Zugriff auf einen bestimmten Datensatz muß dessen logische Satznummer angegeben werden. Beim Lesen aus oder Schreiben in eine UPDATE-Datei inkrementiert die DSR den Datensatzzeiger selbsttätig. Die Satzanzahl kann 1 bis 32768 betragen, wobei der erste Datensatz die Nummer 0 bekommt. Daher reicht der Wertebereich des Satzzeigers von 0 bis 32767.

access always to the current file position for the following reading or write operation one points.

The update mode is only for files with optional access or such with fixed record lengths admissible. To the access to a certain data record its logical record number must be indicated. With reading or writing into an update file the DSR the data record pointer increments reading automatically. The record number can amount to 1 to 32768, whereby the first data record gets the number 0. Therefore the scope of the record pointers from 0 to 32767 is enough.

3.1.4. Aufbau des DSR-Headers

Jede DSR, ob sie nun per CRU in den Bereich >4000->5FFF eingeblendet wird oder im Modul-Port-ROM oder gar GROM vorliegt, muß einem einheitlichen Aufbauschema folgen, wenn sie korrekt eingebunden werden will. Damit soll sichergestellt werden, daß die Idee der Vereinheitlichung auch realisiert wird.

Der sog. Implementationsteil eines DSR-ROMs beginnt an der Adresse >4000 mit den 16 Bytes des DSR-Headers:

Structure of the DSR header

Each DSR, whether it now by CRU into the area >4000->5FFF is inserted or in module port ROM or GROM is present, must a uniform schematic scheme follow, if it wants to be merged correctly. Thus it is to be guaranteed that the idea of the standardization is also implemented.

The so-called Implementation section of a DSR ROM begins at address >4000 with the 16 bytes of the DSR Headers:

>4000	BYTE	>AA	Identifikation eines DSR-ROMs	Identification of a DSR ROM
>4001	BYTE	>01	Versionsnummer, beliebig	Version number, at will
>4002	DATA	>0000	Reserviert	Reserved
>4004	DATA	PWRLNK	Power-Up-Link, Initialisierung der Karten Hard- und Software nach d. Einschalten bzw. Reset	Power-up link, initialization of the cards hard and software after d. switching on, or RESET
>4006	DATA	>0000	Reserviert	Reserved
>4008	DATA	DSRLNK	Startadresse der DSRLNK-Tafel	Start address of DSRLNK Table
>400A	DATA	SBRLNK	Startadresse der SBRLNK-Tafel	Start address of SBRLNK Table
>400C	DATA	INTLNK	Startadresse der INTLNK-Tafel	Start Address of INTLNK Table
>400E	DATA	>0000	Reserviert für zukünftige Erweiterungen	Reserved for future extensions

Lediglich diese 16 Bytes sind in ihrer Adreßzuordnung fest vorgeschrieben - auf alle damit verknüpfen Listen und Tabellen wird über Zeiger zugegriffen, deren einzige Beschränkung darin liegt, daß sie im Adreßbereich der DSR auf der ersten ROM-Seite referenziert werden müssen.

Die Listen für PWRLNK, DSRLNK, SBRLNK und INTLNK sind alle gleich aufgebaut, wobei das Strukturmuster jeweils nur leicht

Only these 16 bytes are fixed prescribed in their address assignment — on all thereby lists and tables link over pointers are accessed, whose only limitation is situated in the fact that they must be referenced within the address range of the DSR on the first ROM page.

The lists for PWRLNK, DSRLNK, SBRLNK and INTLNK are directly structured all, whereby the structure sample is in each case only easily

modifiziert wird. Sogar die Aufruf-Sequenzen für die entsprechenden Funktionen ähneln sich weitestgehend. Dabei werden Einsprünge über diese Zeiger immer nur in bestimmten Situationen vorgenommen.

So wird beim Reset des Systems (Einschalten oder Hard- bzw. Software-Reset) die PWRLNK-Tabelle eines jeden DSR-ROMs abgesucht und eine entsprechende Routine abgearbeitet.

Wird ein externer Interrupt ausgelöst, so sucht der Interrupt-Handler der Konsole alle DSR-ROMs auf einen gültigen Wert ab und springt in die Interrupt-Routine der Karte.

Ein Zeiger mit dem Wert >0000 an einer dieser Adressen zeigt dem System an, daß entsprechende Funktionen von der Karte bzw. deren DSR nicht unterstützt werden.

Das Byte >AA an der Adresse >4000 stellt den sogenannten Validation-Code dar, anhand dessen ein DSR-ROM als gültig bzw. vorhanden erkannt wird. Es ist durchaus denkbar, daß eine beliebige Erweiterung den Adressbereich >4000->5FFF selbst nicht als DSR nutzt und den Validation-Code anders belegt (jeder andere Wert als >AA). Dabei ist jedoch ein prinzipieller Fehler vieler DSRLNK-Routinen zu beachten:

Wird auf einer aktiven CRU-Seite kein gültiger Validation-Code erkannt, so wird das Seiten-Bit (CRU-Bit 0) NICHT wieder mit SBZ >0 deaktiviert! Eine danach eingeschaltete DSR einer anderen Karte würde somit mit dieser ungültigen DSR parallel aktiviert, was in der Regel einen Kurzschluß auf dem PEB-Bus auslöst.

Das Byte an der Adresse >4001 ist frei nutzbar und stellt meist die Revisionsnummer der jeweiligen Firmware dar.

Die Bytes an den Adressen >4002, >4006 und >400E sind für spätere Erweiterungen

modified. Even the calling sequences for the appropriate functions resemble each other as far as possible. Re-entry points are made over these pointers in each case in certain situations.

Thus becomes with the RESET of the system (power-on or hard or. Software RESET) the PWRLNK table of each DSR ROM searched and an appropriate routine processed.

If an external INTERRUPT is released, then the INTERRUPT Handler of the console looks up all DSR ROM to a valid value off and branches into the INTERRUPT routine of the card.

A pointer with the value of >0000 at one these addresses displays to the system that appropriate functions of the card or. their DSR not to be supported.

The byte >AA at the address >4000 represents the so-called Validation code, on the basis it's a DSRROM as valid or. available one detects. It is quite conceivable the fact that any extension the address range >4000->5FFF not when DSR uses and differently occupies the Validation code (every other value than >AA). However an error in principle of many DSRLNK routines is to be considered:

On an active CRU page if no valid Validation code is detected, then the side bit (CRU bit 0) is deactivated NOT again with SBZ >0! A DSR of another card switched on thereafter was thus activated with these invalid DSR parallel, which usually releases a short-circuit on the PEB bus.

The byte at the address >4001 is freely usable and represents usually the revision number of the respective firmware.

The bytes at the addresses >4002, >4006, and >400E are for later extensions reserved. As is to

reserviert. Wie am Beispiel etwa des MSX-Video-Chips zu sehen ist, sollte man reservierte Bits und Bytes immer unangetastet lassen, um späteren Konflikten aus dem Weg zu gehen.

3.1.4.1. PWRLNK

Die mit PWRLNK adressierte Liste leitet den sog. Power-Up-Link ein, welcher zur Initialisierung von Soft- und Hardware der jeweiligen Karte benutzt werden sollte. Die Struktur ist wie folgt:

be seen by the example for instance the MSX video chip, one should leave reserved bits and bytes always untouched, in order to go to later conflicts out of the way.

PWRLNK

The list addressed with PWRLNK leads so-called Power-up link, which should be used for the initialization of soft- and hardware of the respective card. The structure is as follows:

Deutsch

```
PWRLNK DATA >0000      Zeiger auf Folgeeintrag (keiner)
       DATA PWRPRG     Zeiger auf Start der Routine
       BYTE >00        Namenlänge (Kein Name, also 0)
       EVEN            wir haben eine 16bit-CPU
       *
PWRPRG ...             hier die Init-Sequenz
       *
       RT              Rücksprung in den Monitor
```

English

```
PWRLNK DATA >0000      pointers on subsequent entry (none)
       DATA PWRPRG     pointer on start of the routine
       BYTE >00        name length (no name, thus 0)
       EVEN            we have a 16bit-CPU
       *
PWRPRG ...             here the init sequence
       *
       RT              return branch into the monitor
```

Die Power-Up-Routine kann die beim Aufruf aktuellen CPU-Register R0 bis R10 benutzen; R12 bis R15 sind tabu bzw. müssen vor dem Rücksprung wiederhergestellt werden. Daß R11 gesichert wird ist klar, oder?

Das VDP-RAM kann von der Adresse >0000 bis zu dem VDPHI-Zeiger an @PAD + >70 bzw. WP - > E0 + >70 (normal >8370) belegt werden (was z.B. der Diskcontroller mit Vorliebe tut).

The power-up routine can use with the call current CPU registers R0 to R10; R12 to R15 are taboo or. must be re-created before the return branch. That R11 becomes secured is clearly, or?

VDP RAM can do up to the VDP rear pointer at @PAD + >70 of the address >0000 or. WP -> E0 + >70 (normally >8370) to be occupied (which does e.g. to the disk controller with preference).

Im PAD-RAM können alle Adressen mit Ausnahme der Offsets >55, >6D und >C0 - >DF (normal >8355, >836D und >83C0 bis >83DF) frei belegt werden.

3.1.5. INTLNK

Der Interrupt-Link ähnelt dem Power-Up-Link im Listenaufbau. Ein gravierender Unterschied ist jedoch der, daß der die INTLNK-Tabelle bearbeitende Monitor aufgrund der neutralen Einbindung jeder DSR nicht erkennen kann, welche Karte den Interrupt auslöste (auf die vom PC bekannten Probleme einer sauberen Interrupt-Verwaltung soll hier nicht eingegangen werden).

Daher muß die Interrupt-DSR selbst überprüfen, ob der Interrupt von der eigenen Hardware ausgelöst wurde. Entsprechend muß in den Monitor entweder mit RT (Interrupt stammt von anderem Gerät) oder mit INCT R11, RT (Interrupt kam von der eigenen Karte) zurückgesprungen werden. Ein ggf. selbst ausgelöster Interrupt muß natürlich von der DSR zurückgesetzt werden.

In the PAD RAM all addresses with exception of the offsets can be occupied >55, >6D and >C0-> DF (normally >8355, >836D and >83C0 to >83DF) freely.

INTLNK

The Interrupt Link resembles power-up link in the listing format. A serious difference is however that the INTLNK table processing monitor due to the neutral integration of each DSR not detect cannot, which card the interrupt released (on of the PC problems of a clean interrupt administration admitted are not not to be received here).

Therefore the Interrupt DSR must check whether the interrupt was released by the own hardware. Accordingly must into the monitor either with blank (interrupt comes from other device) or with INCT R11, blank (interrupt came from the own card) to be branched back. If necessary interrupt even released must be naturally reset of the DSR.

Deutsch

```
INTLNK DATA >0000     Zeiger auf Folgeeintrag (keiner)
       DATA INTPRG    Zeiger auf Start der Routine
       BYTE >00       Namenlänge (Kein Name, also 0)
       EVEN           s.o.
       *
INTPRG ...            hier die Interrupt-Sequenz. Sie muß prüfen, ob der
                      Interrupt von dieser Karte stammt, sonst NIXINT!
       *
INTEND INCT R11       Weitersuchen verhindern
NIXINT RT             Rücksprung in den Monitor
```

English

```
INTLNK DATA >0000     pointers on subsequent entry (none)
       DATA INTPRG    pointer on start of the routine
       BYTE >00       name length (no name, thus 0)
       EVEN           s.o.
```

```
       *
INTPRG ...             here the INTERRUPT sequence. It must check, whether
                       the INTERRUPT comes from this card, otherwise NIXINT!
       *
INTEND INCT R11        far searches prevent
NIXINT RT              return branch into the monitor
```

Eine Interrupt-Routine darf alle Register bis auf R0, R9 und R12 bis R15 des beim Aufruf aktuellen Workspace benutzen. Belegbare Adressen im PAD sind die Offsets >4A bis >6D (normal >834A bis >836D)

An interrupt routine may all registers up to R0, R9 and R12 to R15 of the Workspace current with the call use. Provable addresses in the PAD are the off sets >4A to >6D (normal >834A to >836D)

3.1.5.1. DSRLNK

DSRLNK

Die Device-Service-Routine-Link-Table (schrecklicher Anglizismus ...) stellt gewissermaßen das Herz des I/O-Systems des TI-99/4A dar. Hier finden sich die meisten und bedeutendsten Programmsequenzen. Im Gegensatz zu INTLNK und PWRLNK muß bei DSRLNK (und auch bei SBRLNK) bereits vom Monitor entschieden werden, welches Programm der DSR das Richtige ist. Dementsprechend ist die DSRLNK-Tafel aufgebaut:

The Device Service Routine link table (terrible Anglicism ...) represent to a certain extent the heart of the I/O Systems of the TI-99/4A. Here are most and most important program sequences. Contrary to INTLNK and PWRLNK must be already decided with DSRLNK (and also with SBRLNK) by the monitor, which program of the DSR is the correct. Accordingly the DSRLNK board is structured:

Deutsch

```
DSRLNK DATA  DSRL02     Sog. Link-Zeiger auf weitere Geräte
       DATA  DSR001     Einsprungadresse für dieses Gerät
       BYTE  4          Länge des Gerätenamens (ohne Punkt!)
       TEXT  'DSK1'     Gerätename
       EVEN             Word-Boundary
DSRL02 DATA  DSRL03     Link-Zeiger
       DATA  DSR002     Einsprungadresse
       BYTE  4          Namenlänge
       TEXT  'DSK2'     Name
       EVEN
DSRL03 DATA  >0000      Kein weiteres Gerät
       DATA  DSR003     Einsprungadresse
       BYTE  3          Namenlänge
       TEXT  'DSK'      Name
       EVEN
DSR001 ...              Programm für Gerät DSK1
       B     @DSREND
DSR002 ...              Programm für Gerät DSK2
```

```
       B     @DSREND
DSR003 ...               Programm für Gerät DSK
       B     @DSREND
*
DSREND INCT  R11
NXTDSR RT
```

English

```
DSRLNK DATA DSRL02       So-called Link pointer on further devices
       DATA DSR001       re-entry point address for this device
       BYTE 4            length of the device name (without period!)
       TEXT 'DSK1'       device name
       EVEN              word boundary
DSRL02 DATA DSRL03       link pointer
       DATA DSR002       re-entry point address
       BYTE 4            name length
       TEXT 'DSK2'       name
       EVEN
DSRL03 DATA >0000        no further device
       DATA DSR003       re-entry point address
       BYTE 3            name length
       TEXT 'DSK'        name
       EVEN
DSR001 ...               Program for device DSK1
       B    @DSREND
DSR002 ...               Program for device DSK2
       B    @DSREND
DSR003 ...               Program for device DSK
       B    @DSREND
*
DSREND INCT R11
NXTDSR RT
```

Erstmals wird dabei in der Funktionshierarchie ein Weiterleiten einer Geräteanfrage erlaubt, wie später bei der Muster-DSRLNK-Routine zu sehen sein wird. Dabei kann nach erfolgter (oder auch nicht erfolgter) Bearbeitung einer I/O-Anforderung das DSRLNK mit gleichem Namen zur weiteren Suche oberhalb der aktuellen CRU-Seite aufgefordert werden. Hierzu zählt die DSRLNK-Routine einen Index mit, welcher die Anzahl erfolgter Bearbeitungen zählt. Dieser liegt in R1 vor, weshalb R1 des beim Aufruf aktuellen Workspace innerhalb der DSR nicht geändert werden darf bzw. vor deren Ende restauriert werden muß, wenn über NXTDSR in die DSRLNK-Routine zurückgesprungen wird. An diesem Index kann eine DSR erkennen, ob sie die erste oder folgende in der Liste ist.

A blank is permitted to a device inquiry for the first time thereby in the function hierarchy passing on, how will be to see later with the sample DSRLNK routine. The DSRLNK with same name can be requested to the further search above the current CRU page after effected (or also not effected) handling of a I/O request. For this the DSRLNK routine takes in account an index, which counts the number of effected processing. This is present in g 1, why g 1 of the Workspace current with the call may not be modified within the DSR or. before their end to be restored must, if over NXTDSR into the DSRLNK routine one branches back. By this index a DSR can detect whether it is first or following in the list.

Diese Funktion wird z.B. von den Original TI-RS232-Karten benutzt, deren DSR identisch ist, jedoch auf verschiedene CRU-Seiten gelegt werden kann. Hier prüft die jeweilige DSR, ob sie als RS232/1,2 oder RS232/3,4 bzw. analog als PIO/1 oder PIO/2 reagieren soll.

This function will be put e.g. by the original TI RS232 card used, whose DSR is identical, however on different CRU pages can. Here the respective DSR checks whether her as RS232/1,2 or RS232/3,4 or similarly as PIO/1 or PIO/2 to react is.

3.1.5.2. SBRLNK

SBRLNK

Die Tafel der Zeiger auf Subroutine-Links ist gegenüber der DSRLNK-Tafel nur in Bezug auf den Geräte- bzw. Funktionsnamen modifiziert. Der Gerätename ist in der Regel nur 1 Zeichen lang und beginnt beim ASCII-Code >10. Der Rücksprung wird identisch zur DSRLNK-Routine gehandhabt, wobei auch das Weiterleiten ohne INCT R11 möglich ist. Bitte ziehen Sie das Beispiel zur DSRLNK-Struktur heran.

The table of the pointers on subroutine links is opposite the DSRLNK table only regarding devices or Function names modifies. The device name is usually only 1 character long and begins with the ASCII code >10. The return branch is handled identical to the DSRLNK routine, whereby also passing on without INCT R11 is possible. Please you consult the example for the DSRLNK structure.

3.1.5.3. Beispiel für eine DSRLNK-Routine

Example of a DSRLNK routine

Ein DSRLNK-Routine erfüllt viele Funktionen:

A DSRLNK routine fulfills many functions:

- Sie sucht den gesamten CRU-Adressraum ab >1000 nach vorhandenen DSR-ROMs ab.
- You looks up the entire CRU address area off after >1000 available DSR ROM off.
- Sie sucht im DSR-ROM nach dem passenden
- You look up in DSR ROM for the suitable

Gerätenamen.

- Sie ermittelt die passende Einsprungadresse der zum Gerät passenden DSR-Sequenz.
- Sie bereitet den Gerätenamen für die DSR auf.
- Sie sorgt für eine Verkettung ähnlicher oder gleicher DSRs.

Bei der Besprechung der DSRLNK-Routine sind zwei Seiten zu beachten:

die eigentliche DSRLNK-Seite und die DSR-Seite. Für dieses Zusammenspiel sind die Pointer an den Offsets >56 und >54 (normal >8356 und >8354) im PAD-RAM von zentraler Bedeutung.

- Der Zeiger an >8356 (W) zeigt beim Aufruf des DSRLNK auf das Namenlängebyte im PAB (Abweichung GPLLNK; siehe dort). Beim Einsprung in die DSR zeigt er auf das erste Byte hinter dem Gerätenamen (Beispiel: bei DSK1.MUSTER auf den Punkt hinter DSK1).
- Der Zeiger an >8354 (W) ist beim Einsprung in das DSRLNK undefiniert; beim Einsprung in die DSR enthält er die gesamte Länge des Namens abzüglich des Gerätenamens.

3.1.6. SBRLNK und GPLLNK

SBRLNK (Subroutine-Link) ist eine recht geringfügige Abwandlung des DSRLNK. Hier wird der Offset-Pointer in den DSR-Header auf eine andere Tabelle umgeleitet (Offset >A statt >8), womit auch etwas andere PAB-Strukturen Anwendung finden.

GPLLNK ist die Variante des DSRLNK für GROMs und ROMs im Bereich >6000 - >7FFF. Die Header der betreffenden Speicher sind dabei exakt so wie die der DSR im Bereich >4000 -

device name.

- The suitable re-entry point address DSR sequence of the fitting the device determines you.
- You edit the device names for the DSR.
- You provides for a concatenation of similar or same DSRs.

During the discussion of the DSRLNK routine two pages are to be considered:

The actual DSRLNK page and the DSR page. For this interaction the pointers at the off sets are >56 and >54 (normally >8356 and >8354) in the PAD RAM of central importance.

- the pointer >8356 (W) shows on with the call of the DSRLNK on the name length byte in the PAB (deviation GPLLNK; see there). With the re-entry point into the DSR it points to the first byte behind the device name (example: with DSK1.MUSTER on the point behind DSK1).
- The pointer >8354 (W) is undefined with the re-entry point into the DSRLNK on; with the re-entry point into the DSR it contains the entire length the name less the device name.

SBRLNK and GPLLNK

SBRLNK (Subroutine Link) is a quite slight modification of the DSRLNK. Here the offset pointer is rerouted into the DSR header on another table (offset >A instead of >8), with which also something other PAB structures application find.

GPLLNK is the version of the DSRLNK for GROMs and ROM in the area >6000 ->7FFF. The headers of the memory concerned are structured thereby accurately as those the DSR

>5FFF aufgebaut. GPLLNK sucht i.d.R. jedoch zusätzlich noch den normalen DSR-Bereich ab. Dabei bedient es sich des normalen TI-Monitors.

Der Aufruf erfolgt dann mit der Sequenz

```
CALL  >10
DATA  >0008
```

wobei zuvor ein PAB wie bei DSRLNK aufgebaut werden muß. Der wesentliche Unterschied besteht in der Tatsache, daß der Zeiger am PAD-Offset >56 (normal >8356) nicht wie bei dem DSRLNK auf das Namenlänge-Byte des PABs, sondern direkt auf das erste Zeichen des Namens zeigt.

3.1.7. Abweichung des Myarc-DOS im 9640

Der Myarc-9640-Geneve, zu Anfang als beinahe 100% kompatibel bezeichnet, inzwischen als potentiell inkompatibel zum TI-99/4A erkannt und mit einer eigenen Softwarelinie versehen, benutzt aus verschiedenen Gründen das TI-DSR-System nicht

Wie aus der DDCC-1-DSR (dem Myarc Diskcontroller) zu ersehen, hat sich Myarc von je her nie an die TI-Vorgaben gehalten, was eine DSR darf und was nicht.

Bei der Konzeption des Geneve war es zum einen diese Grundhaltung, zum anderen innovativer Geist und wohl auch Angst vor den eigenen Holzhammer-DSRs, welche das alte DSR-Konzept als unbrauchbar erscheinen ließen. Man kann mit der notwendigen Sachkenntnis aus dem DDCC-1 Listing sehr leicht sehen, daß schon der erste Disketten-Schreibzugriff den Monitor und alle XOPs bzw. Interrupt-Vektoren des MDOS gelyncht hätte, würde dieser Controller mit der normalen DSR laufen.

Da man bei Myarc zudem keine Hardware-synchronisation mittels DRQ und INTREQ

in the area >4000->5FFF. GPLLNK looks up usually however additionally still the normal DSR area off. It avails itself of the normal TI Monitors.

The call with the sequence

```
CALL  >10
DATA  >0008
```

whereby before a PAB as with DSRLNK to be effected then structured must. The substantial difference exists in the fact that the pointer at the PAD offset shows >56 (normally >8356) not as with the DSRLNK to the name length byte of the PABs, but directly to the first character of the name.

Deviation of the Myarc DOS in 9640

The Myarc-9640-Geneve, at the beginning as almost 100% compatibly defined, in the meantime as potentially incompatible to the TI-99/4A detected and with its own software line provided, does not use the TI DSR system from different reasons.

As from the DDCC-1 DSR (the Myarc Disk Controller) to seen, Myarc from ever ago adhered never to the TI defaults, which may do a DSR and which not.

With the conception of the Geneve it was on the one hand this basic attitude, on the other hand innovative spirit and probably fear of the own mallet DSRs, which let the old DSR concept appear as useless. One can see listings very easy with necessary expertise from the DDCC-1 that already the first diskette write access the monitor and all XOPs or. Would have lynched interrupt vectors of the MDOS, this controller with the normal DSR would run.

Since one with Myarc besides no hardware synchronization by means of DRQ and INTREQ

beherrschte, wurde der Bussteuerung des Geneve dieses Feature nicht mitgegeben (vergessen?). Damit war es also unmöglich, den eigenen Controller einzubinden (DSR zu gefährlich) und alle anderen Controller konnten keinen Datentransfer synchronisieren, weil die TMS9995 einfach über den DRQ 'drüberlief'. Soweit zum Lapsus der Herren um Mr.Phillips. Die innovative Idee war, anhand eines simulierten DSR-ROMs (der DSR- Bereich >4000 - >5FFF ist beim Geneve üblicherweise nicht transparent) alle Geräteanfragen umzuleiten und von MDOS-eigenen Routinen abarbeiten zu lassen. Damit konnte man einerseits Kollisionen verhindern, da alle Software aus einem Guß war (was immer das bei Myarc auch implizieren mag ...), andererseits war ein Austausch evtl. fehlerhafter Software (man wußte schon, wozu das gut war ...) über eine MDOS-Revision bis hin zu dem DSRs möglich.

Der prinzipielle Nachteil ist aber, daß sich neue Karten nun nicht mehr selbst im System 'anmelden', indem sie einen neuen Namen einführen, sondern nun jedesmal beim Erscheinen einer neuen Karte das MDOS erweitert werden muß. Da dies i.d.R. nur durch 'Reverse Engineering' geschieht, sind die Wartezeiten vorprogrammiert.

Zum Zeitpunkt des Abschluß' dieses Kapitels war jedoch zu hören, daß Myarc das alte DSR-Konzept wieder entdeckt hätte ...

Aufgrund der abweichenden Busstruktur des 9640 ist es zudem nicht generell möglich, doch die eingebauten DSRs anzusprechen, da man damit rechnen muß, daß diese auf einen TI-Bus hin ausgelegt sind (Low-High-Byte-Sequenz ist vertauscht, CRU-Adressraum benötigt weniger Bits, Wait-States erlaubt, Data-Setup-Time ist länger etc.).

controlled, this feature was not given to the bus control of the Geneve (forgotten?). Thus it was impossible, and their own controller could not synchronize to merge (DSR too dangerously) and all other controllers data transfer, because the TMS9995 simply "skips over" [pass over, ignore] the DRQ. So far to the slip of the gentlemen around Mr. Phillips. The innovative idea was, on the basis a simulated DSR ROM (the DSR area >4000 - >5FFF is usually not to be rerouted with the Geneve transparency) all device inquiries and be let from MDOS own routines process. Thus one could prevent on the one hand collisions, since all software from a casting was (which may always also imply with Myarc ...), on the other hand one was an exchange perhaps incorrect software (one already knew, to which that was good ...) over a MDOS revision up to the DSRs possible.

The disadvantage in principle is however that new cards now any longer do not "announce" themselves in the system, by introducing a new name, but now with the appearance of a new card the MDOS must be extended each time. Since this is done usually only via "reverse engineering," the waiting periods are pre-programmed.

At the point in time termination of this chapter was to be heard however that Myarc would again have discovered the old DSR concept ...

Due to the deviating bus structure for 9640 it is to respond besides not generally possible, but the inserted DSRs, since one must count on the fact that these are designed for a TI bus (Low High byte sequence is exchanged, CRU address area needs fewer bits, wait states permitted, data set up time is longer etc.).

4. Arten des Zugriffs auf die Diskettendaten

Am Beispiel der Dateiverwaltung (z.B. in Basic) soll gezeigt werden, wie die logische Einheit 'FILE' aufgebaut ist und wie die Daten abgelegt bzw. abgerufen werden.

Das DOS des TI bringt von Hause aus sehr leistungsfähige Routinen zur Dateiverwaltung mit, die sonst nur im PC-Bereich, keinesfalls aber für einen Home-Computer selbstverständlich sind.

So bleibt dem Anwender als einzige Planung, die Struktur seiner Datei festzulegen und die Satzlänge zu spezifizieren. Für einfache Anwendungen reichen die Standard-Werte DISPLAY-Datenformat, VARIABLE Satzlänge von maximal 80 Bytes bereits aus.

Eine Datei kann, wenn sie existiert, ohne genaue Kenntnis von Datenart und Satzlänge im Eingabemodus eröffnet werden, und es können Daten bis zum logischen Ende der Datei gelesen werden.

Bei Eröffnung im Ausgabemodus können Daten in beliebiger Reihenfolge und fast beliebiger Anzahl geschrieben werden, wobei nur die Speicherkapazität der Diskette beschränkend wirkt.

Dateien, welche eine feste Länge der Einträge besitzen, können im sogenannten UPDATE-Modus eröffnet werden, bei dem mit einfachsten Kommandos beliebige Einträge der Datei gelesen und/oder geschrieben werden können.

Dateien jeder beliebigen Blocklänge können im APPEND-Modus eröffnet werden, womit dann eine Veränderung bereits vorhandener Daten nicht möglich ist. Am derzeitigen logischen Ende der Datei können jedoch neue Datensätze angefügt werden.

Kinds of access to the diskette data

By the example of the file management (e.g. in Basic) is to be demonstrated, how the logical unit "file" is developed and how the data are stored and/or recalled.

The DOS of the TI bring along very efficient routines for file management, which are not natural for a home computer otherwise only in the PC area, under any circumstances however from house from.

Thus remains for the user as only planning to determine the structure of its file and specify the record length. For simple applications the default values DISPLAY data format, VARIABLE record length of maximum 80 bytes are already sufficient.

A file can be opened, if it exists, without exact knowledge by data type and record length in the input mode, and data up to the logical end of the file can be read.

With initialization in the output mode data in any order and almost any number can be written, whereby only the storage capacity of the diskette works limiting

Files, which possess a fixed length of the entries, can be opened in the so-called update mode, with which with simplest commands any entries of the file can be read and/or written.

Files of any block length can be opened in the APPEND mode, with which then a modification of already available data is not possible. At the present logical end of the file however new data records can be added.

Zusätzlich lassen sich Dateien per Kennzeichnung vor dem Überschreiben schützen, was einen besonderen Schutz der enthaltenen Daten bewirkt.

Files can additional be protected by indication against the overwriting, which causes a special protection of the contained data.

Das Arbeiten mit Dateien (Files) ist also die für den Benutzer einfachste Art, das Medium Diskette als Datenspeicher zu verwenden. Er muß nur darauf achten, daß der Platz auf der Diskette ausreicht — an welchen Orten auf der Diskettenoberfläche sich die Daten befinden ist für ihn dann ohne Belang; er sieht die Datei als zusammenhängende logische Einheit.

Operating with files (files) is thus the type simplest for the user to use the medium diskette as memories. It must only make certain that the place on the diskette is sufficient — at which places on the disk surface the data are are it then without importance; he sees the file as connected logical unit.

Daß aber einige Probleme bei dem Aufbau eines Files entstehen können, soll ein Beispiel zeigen:

The fact that however some problems can with the structure files develop is to show an example:

Angenommen, die von Ihnen verwendete Diskette ist bis auf wenige Sektoren belegt. Um Platz für die Erweiterung einer Datei zu bekommen, kopieren Sie ein auf der Diskette befindliches Programm auf eine andere Diskette. Nun können Sie die freigewordenen Sektoren benutzen, um Ihre Datei zu erweitern. Es ist nun aber im Allgemeinen so, daß sich dieses nun gelöschte Programm nicht unbedingt direkt hinter dem zu erweiternden File befand — es sind also an verschiedenen Orten auf der Diskette Sektoren frei geworden. Für Sie als Anwender des logischen Verbundes 'FILE' war das aber bisher kein Problem und soll es auch nicht werden.

Assumed, the diskette used by you is occupied up to few sectors. In order to receive place for the extension of a file, you copy a program present at the diskette on another diskette. Now you can use the freed sectors, in order to extend your file. It is now however generally like that that this program deleted now was not necessarily directly behind the file which can be extended — thus at different places on the diskette sectors became free. For you as users of the logical group "FILE" however so far no problem was and is it also to become.

Es ist aber die Aufgabe des Diskettenbetriebssystems festzustellen, ob der zusammenhängende freie Raum auf der Diskette ausreicht, oder ob das File in mehrere, physikalisch getrennte Teile aufgespalten werden muß.

However the function of the diskette operating system is to be determined whether the connected free space on the diskette is sufficient, or whether the file must be split up into several, physically separated sections.

Es kann also passieren, daß ein 20 Sektoren langes File die Sektoren 10 bis 14, 100 bis 110 und 88 bis 91 belegt, obwohl es als logische Einheit ein zusammenhängendes Gebilde darstellt.

It can occur thus that a long file the sectors 10 to 14, 100 to 110 and 88 to 91 occupies 20 sectors, although it represents a connected thing as logical unit.

File-Operationen sind gleichermaßen in TMS9900 Assembler möglich, besonders einfach durch die DSRLNK-Routine. Das File ist also der einfachste und zugleich eleganteste Weg der Datenverwaltung.

Die Programmteile des DOS (der Diskcontroller DSR), welche z.B. die File-Verwaltung auf Basic-Ebene steuern, werden Level-2-Routinen genannt.

Die in der Hierarchie folgende Zugriffsart auf die Diskettendaten ist der Sektorzugriff mit den Level-1-Routinen (siehe hierzu der Abschnitt '5.1 Einteilung der Diskette in Sektoren')

Hier werden den Zellen, aus denen die Diskette besteht, fortlaufende Nummern von 0 bis 359 (Hex 0 bis 167) bei einfacher Dichte und jeweils bis zum doppelten bei doppelter Dichte und doppelseitigen Disketten zugeordnet. Die maximale Anzahl ist also bei 40 Spur Laufwerken 1440 Sektoren, entsprechend einer Numerierung von 0 bis 1439.

Aber auch hier ist es für den Anwender ohne Belang, auf welcher Spur und welcher Seite der Diskette sich der Sektor befindet, nur seine 'Hausnummer' muß angegeben werden.

Der Zugriff auf bestimmte Sektoren ist nur in Maschinensprache möglich.

Die letzte Zugriffsart stellt der Spurzugriff über Level-0-Routinen dar. Für die Anwendung dieser Funktion muß die Hardware des Diskettencontrollers vollständig bekannt sein. Damit ist ein Spurzugriff im Gegensatz zum Sektor- und Filezugriff von der Hardware abhängig, so daß solche Programme nur auf bestimmten Systemen arbeiten oder aber angepaßt werden müssen.

Die Möglichkeiten und Probleme des Spurzugriffs werden in den Kapiteln 13, 14 und 15, neben anderen Aspekten der Level-0-

File operations are equally in TMS9900 assembler possible, particularly simply by the DSRLNK routine. The file is thus the simplest and at the same time most elegant way of the data management.

The program sections of the DOS (the disk controller DSR), which e.g. the file administration on basic level control, Level-2 Routine are called.

The access method on the diskette data, following in the hierarchy, is that sector access with the Level-1 routines (see section "5.1 Organization of the diskette into sectors").

Here sequential numbers are assigned to the cells, of which the diskette consists, in each case from 0 to 359 (Hex 0 to 167) with simple density and up to the double with double density and double-sided diskettes. The maximum number is thus with 40 track drives 1440 sectors, according to a numbering from 0 to 1439.

In addition, it is here for the user without importance, on which track and which side of the diskette the sector is, only its "house number" must is indicated.

That access to certain sectors is possible only in machine language.

The last access method represents track access over Level-0-Routine to that. For the application of this function the hardware of the diskette controller must completely admits to be. Thus a track access depends in contrast to the sector and file access on the hardware, so that such programs must operate only on certain systems or be adapted however.

The possibilities and problems of the track access are discussed in Chapters 13, 14 and 15, apart from other aspects of the Level-0 Routine.

Routinen, besprochen. Die universellen Zugriffsmöglichkeiten auf Level-0 des DOS, also da, wo eigentlich kein DOS mehr verfügbar ist, werden mit einem im Vergleich zu Level-2 und Level-1 sehr hohen Programmieraufwand erkauft.

The universal access options on Level-0 of the DOS, thus, where no DOS is more available actually, are bought with a programming effort very high compared with Level-2 and Level-1.

5. Rolle der DSR-Software des Controllers

Das Kürzel DSR bedeutet 'Device Service Routine', was etwa mit 'Gerätesteuerungs-routine' übersetzt werden kann. Diese, bisher nur vom TI bekannte Art des Einbindens vollkommen unterschiedlicher und neuer Systemerweiterungen in das Betriebssystem des Rechners, macht seine besonders intelligente Konzeption deutlich. Durch eine genormte 'Softwareschnittstelle' ist es möglich, alle Erweiterungen identisch zu behandeln, indem lediglich deren Name geändert wird.

Eine Spezifizierung des Begriffs 'DSR' stellt das Kürzel 'DOS' dar, was 'Diskette Operating System' heißt und direkt mit 'Disketten Operations System' übersetzt werden kann. Diese besondere DSR steuert alle notwendigen Abläufe, die zum Betrieb einer Diskettenstation gehören.

Dies sind z.B.
- Formatieren neuer Disketten,
- Verwalten von Daten in Files, etc.

Eine genaue Kenntnis aller Befehle, aus denen sich diese Software zusammensetzt, ist nicht notwendig, da es Unterschiede zwischen den Controllern gibt, die man bei ausschließlicher Beachtung der notwendigen Pointer ignorieren kann.

Die DSR macht also die eigene Software unabhängig von dem Typ der vorhandenen Hardware. Es ist damit problemlos möglich, bestimmte Sektoren einer Diskette mit dem gleichen Programm auf einem TI, CorComp, Myarc oder Atronic Diskcontroller zu lesen oder zu schreiben.

Role of the DSR software of the controller

The acronym DSR means Device Service Routine, which can be translated for instance with "device control routine." This, so far only of the TI well-known type of merging perfectly different and new system expansions into the operating system of the computer, makes its particularly intelligent conception clear. By a standardized "software interface" it is possible to treat all extensions identically as their name is only modified.

A specification of the term "DSR" represents the contraction "DOS," which "diskette operating system" is called and directly with "diskettes operation system" can be translated. This special DSR controls all necessary operational sequence, which belong to the operation of a floppy station.

These are e.g.
- formatting new diskettes,
- administration of data in files, etc..

An exact knowledge of all instruction, of which this software consists, is not necessary, since it differences between controller gives, which one can ignore with exclusive attention of the necessary pointers.

The DSR makes thus the own software independent of the type of the available hardware. It is problem-free possible for certain sectors of a diskette with the same program on a TI, CorComp thereby, to read or write Myarc or Atronic disk controllers.

5.1. Einteilung der Diskette in Sektoren

Die Daten, welche auf der Diskette abgelegt werden, befinden sich dort nicht 'am Stück', sondern werden in kleine Segmente zu je 256 Byte aufgespalten. Diese Einteilung orientiert sich an der 'Parzellierung' der Diskette nach dem Formatieren. Hier werden in jeder Spur Marken auf die Diskette gesetzt ähnlich denen, die man zur synchronen Steuerung eines Diaprojektors über ein Tonband benutzt. Erst danach ist es möglich, gezielt auf bestimmte Bereiche der Diskette zuzugreifen, da erst jetzt eine Art 'Wegenetz' existiert.

Die Länge von 256 Bytes pro Sektor stellt einen Kompromiß dar zwischen maximaler Ausnutzung des Speicherraums auf Diskette und minimaler Ladezeit. Das ist deshalb so, weil ein einziger Sektor nur Daten eines einzigen Files enthalten kann und keine Mischbelegung zulässig ist. Es ist daher in der Regel so, daß ein Sektor z.B. eines DIS/VAR 80 Files an seinem Ende maximal 81 Zeichen unbenutzt lassen muß, weil vielleicht der letzte Eintrag gerade 80 Zeichen lang war. Dieser Eintrag würde dann in den nächsten freien Sektor geschrieben, der dann bis auf 82 Zeichen (incl. Längenbyte!) leer wäre. Bei einer Sektorlänge von z.B. 1024 Byte wären dann 942 Byte unbenutzt! Ein Sektor mit 1024 Bytes Länge wird aber schneller geladen, als vier mit je 256 Bytes.

Wie diese Zahlen zustande kommen, wird in den folgenden Kapiteln besprochen.

Dieses Wissen um die Belegung der Sektoren ermöglicht es, bei speicherintensiven Dateien den verfügbaren Platz auf der Diskette optimal zu nutzen. Zwar wird bei der Beschreibung der Datensektoren noch darauf eingegangen, hier aber ein erster Hinweis. Bei Datensätzen variabler Länge wird ein Byte angefügt, das die

Organization of the diskette into sectors

The data, which are stored on the diskette, are not "at the piece," but into small segments to ever 256 byte are split up there. This organization orients itself at the "Parzellierung" the diskette after formatting. Here in each track labels set on the diskette similarly those, which one uses for the clocked control of a still-picture projector over a tape. Only after it is possible accessing directly at certain areas of the diskette since a type "wegenetz" exists only now.

The length of 256 bytes per sector represents a compromise between max. utilization of the memory space on diskette and minimum loading time. That is like that, because only one sector can contain only data only one files and no mixing allocation is admissible. It therefore is usually like that that a sector e.g. a DIS/VAR must leave 80 files at its end of max. 81 characters unused, because the last entry was long perhaps even 80 characters. This entry was then written into the next free sector, then up to 82 characters (inclusive. Length byte!) would be empty. With a sector length of e.g.. 1024 byte then 942 byte would be unused! A sector with 1024 bytes length is however faster loaded, than four with 256 bytes each.

As these numbers come is discussed, in the following chapters.

This knowledge around the allocation of the sectors makes it possible to use with memory-intensive files the available place on the disk optimally. During the description of the data sectors with it, here however a first reference is still dealt. With data records of variable length a byte is added, which indicates

Länge des Datensatzes jeweils neu angibt. Man sollte die Datensatzlänge dann so wählen, daß eine bestimmte Anzahl von Datensätzen inclusive des Längenbytes 256 ergibt. Bei FIXED-Dateien fehlt dieses Längenbyte, die Berechnung einer optimalen Satzlänge ist analog zu VAR-Dateien. Überhaupt ist die optimale Relation zwischen Platzökonomie und Ladezeit nur bei FIXED-Dateien gegeben. Besonders deutlich wird dies beim Suchen eines Datensatzes. Dazu später mehr.

Die Abfolge der Sektoren einer Spur ist nicht numerisch aufsteigend, näheres hierzu in den Abschnitten 'Formatierung von Disketten' und 'Aufzeichnungsverfahren'.

Neben der 'Feinstruktur' der Sektoren ist die Diskette in Spuren aufgeteilt. Diese sind konzentrische Ringe, deren Umfang nach der Diskettenmitte hin abnimmt. Die Datenspuren hängen also nicht in der Art einer Spirale zusammen, sondern sind voneinander getrennt. Über die Spur, in der ein Sektor zu finden ist, braucht sich jedoch kein Programmierer den Kopf zu zerbrechen, solange er sich an die fortlaufende Sektor-Numerierung hält.

5.2. Aus der Sektorstruktur resultierende Kompatibilität

In allen für den TI verfügbaren Diskcontroller-schaltungen, ob als Karte für die P-Box oder eigenständig, werden Floppy-Disk Controller/Formatter ICs nach dem Muster der Firma Western Digital eingesetzt, Typen-bezeichnung 1771 oder 1773. Diese integrierten Schaltungen bzw. die formatkompatiblen NEC 765 werden auch in vielen anderen Rechnern, so z.B. frühe IBM-Modelle, eingesetzt. Da die IC's bestimmte Sektoren selbständig suchen und an die CPU übergeben, muß das verwendete Diskettenformat bestimmte Bedingungen erfüllen. Das bedeutet praktisch, daß bei

the length of the data record again in each case. One should select the data record length in such a way then that a certain number of data records inclusive of the length byte 256 results in. With FIXED files this length byte, the computation of an optimal record length is missing is similar to VAR files. At all the optimal relation is given between place economics and load time only with FIXED files. This becomes particularly clear with looking for a data record. In addition later more.

The sequence of the sectors of a track is not numerically ascending, details for this in the paragraph "formatting of diskettes" and "recording methods."

Beside the "fine structure" of the sectors the diskette is divided in tracks. These are concentric rings, whose scope decreases toward the diskette center. The data tracks are connected thus not in the type of a spiral, but are from each other separated. Over the track, in which a sector is to be found, however no programmer the heading needs to break itself, as long as he adheres to the sequential sector numbering.

From the sector structure resulting compatibility

In all disk controller circuits whether as card for the PEB or independently, available for the TI, become Floppy Diskette Controller/Formatter IC's after the sample of the company Western Digital assigned, type designation 1771 or 1773. These integrated circuits, or the format-compatible NEC 765, are used also in many other computers, e.g. early IBM models. Since the ICs looks up and to the CPU transfers certain sectors independently, the used diskette format must fulfill certain conditions. That means it practically that during agreement of the sector length all diskettes all that computer

Übereinstimmung der Sektorlänge alle Disketten all jener Rechner gelesen werden können, die einen solchen Chip als Controller-IC besitzen. Zwar wird der Sektorinhalt von jedem Rechner anders interpretiert, mittels eines Konversionsprogramms ist es aber denkbar, z.B. Textdateien oder Speicherauszüge mit Besitzern anderer Systeme zu tauschen, was in manchen Fällen sicherlich interessant sein kann.

Die Controller von Atronic oder CorComp verwenden bei doppelter Dichte mit 18 Sektoren pro Spur ein Format, das den Mindestanforderungen des IBM System 34 Formats (MFM) entspricht. Hier stimmt auch die Sektorlänge von 256 Bytes überein. Das Original TI-Format bei Single Density ist jedoch in keiner Weise mit dem IBM 3740 Format (FM) kompatibel.

Leider erweist sich die Praxis als etwas schwieriger als das nun vielleicht aussehen mag. Die PC's von IBM mit den 360K-Laufwerken verwenden nämlich eine Abwandlung des MFM-Systems 34 mit 9 Sektoren pro Spur und 512 Bytes pro Sektor. Zusätzlich wechseln die logischen Spurnummern immer die Seiten, also gerade Nummern auf der Ober- und ungerade Nummern auf der Unterseite. Hier ist die DOS-Routine 'Sektor lesen' nicht anwendbar, sie muß neu geschrieben werden.

can be read, which possesses such a chip as controller IC. Sector contents of each computer are differently interpreted, by means of a conversion programs are it however conceivably, e.g. text files or dumps with owners of other systems to be exchanged, which in some cases can be surely interesting.

The controllers of Atronic or CorComp use a format, which corresponds to the minimum requirements of IBM system 34 of format (MFM) with double density with 18 sectors per track. Here also the sector length of 256 bytes corresponds. The original TI format with single Density is however in no way with IBM 3740 format (FM) compatible.

Unfortunately practice proves as somewhat more difficult than now perhaps looking likes. The PCs of IBM with the 360K drives use a modification of the MFM system 34 with 9 sectors per track and 512 bytes per sector. Additionally the logical track numbers change always the sides, thus even numbers on upper and odd numbers on the lower surface. Here the DOS routine "sector is reads" not applicably, it must again be written.

6. Formatierung von Disketten

Der Prozeß der Formatierung muß von jeder neuen Diskette durchlaufen werden, bevor sie als Speichermedium benutzt werden kann. Auf einer neuen Diskette befinden sich im Allgemeinen keine sinnvollen Informationen, außer den Bitfolgen, die bei der Einzelprüfung aufgespielt und gelesen wurden. Aufgabe des Formatierens ist es nun, Marken auf der Diskette aufzubringen, anhand derer sich die einzelnen Informationsblöcke, die Sektoren, wiederfinden und eindeutig erkennen lassen. Dieser Vorgang wird spurweise vorgenommen, wobei in jeder Spur die Sektoren mit Zahlen von 0 bis 8 bei Single Density und von 0 bis 17 bei Double Density (0 bis 15 bei Myarc DD) bezeichnet werden.

Neben den bekannten je 256 Bytes eines Sektors, die das 'Datenfeld' darstellen, befinden sich noch folgende Daten auf der Diskette, die nur für den Floppy-Controller wichtig sind:

Am Anfang einer Spur befinden sich Synchronisierzeichen, ähnlich dem hohen Ton zu Anfang eines Cassettenfiles, anschließend folgen die Sektoren, die wie folgt aufgebaut sind:

- Startbyte für den Prüfsummengenerator, Kennung des Adressfeldes.
- Adreßfeld, bestehend aus Spur-, Seiten- und Sektornummer.
- Sektorlängenkennung,
- Prüfsumme des Adressfeldes mit anschließender Datenpause.
- Startbyte für den Prüfsummengenerator, Kennung des Datenfeldes.
- 256 Datenbytes, beim Formatieren zu >E5 gesetzt.

Formatting of diskettes

The process of formatting must be passed through by each new diskette, before it can be used as storage medium. On a new diskette generally no meaningful information, except the bit sequences, is which were up-played and read with the individual examination. Function of formatting is it now to apply labels on the diskette on the basis those itself the individual information blocks, which show sectors to regain, and unique. This process is made spurweise, whereby in each track the sectors with numbers from 0 to 8 with single Density and from 0 to 17 with doubles Density (0 to 15 with Myarc of DD) are defined.

Beside the well-known 256 bytes each of a sector, which represent the "data field," are the still following data on the diskette, which are important for the floppy controller only:

At the start of a track synchronous idle characters, similarly the high tone at the beginning cassette files, are afterwards follow the sectors, which are structured as follows:

- Start byte for the check total generator, identifier of the address field.
- Address field, consisting of track, side and sector number.
- Sector length identifier,.
- Checksum of the address field with following data break.
- Start byte for the check total generator, identifier of the data field.
- 256 data bytes, when formatting too >E5 set.

- Prüfsumme der Daten mit anschließender Pause zum nächsten Sektor.
- nächste Sektoren wie zuvor,
- Ende der Spur mit Taktsignalen

Wie zu sehen ist, geht ein respektabler Anteil des prinzipiell verfügbaren Speicherplatzes auf der Diskette für Kennung und Prüfung den eigentlichen Daten verloren. Daraus erklärt sich der Unterschied zwischen der Disketten-kapazität formatiert und unformatiert. Eine genauere Beschreibung der Spurstruktur erfolgt in Kapitel 13.

Die Reihenfolge der Sektoren in einer Spur ist nicht auf- oder absteigend, die Ordnung unterliegt bestimmten Zwängen. Hierzu ein Gedankenbeispiel:

Es soll ein Programm geladen werden, welches aus mehreren Sektoren innerhalb einer Spur besteht. Nach Lesen des ersten Sektors, z.B. Nummer 3, muß dessen Inhalt abgespeichert und dann der nächste gelesen werden. Würde nun der Sektor 4 unmittelbar auf Sektor 3 folgen, dann hätte sich während der Zeit, in der Sektor 3 verschoben wurde, die Diskette natürlich weitergedreht und der Kopf befände sich beim nächsten Zugriff bereits über Sektor 5. Dann müßte nahezu eine ganze Umdrehung der Diskette abgewartet werden, um wieder auf Sektor 4 zu treffen. Dieses 'Spiel' würde sich bei jedem neuen Sektor wiederholen und die Zeit zum Laden dieses Programmes unnötig verlängern.

Hier bedient man sich nun eines Verfahrens, das mit 'Sector Interlace' bezeichnet wird. Die Abfolge der Sektoren ist nun nicht mehr in numerischer Reihenfolge, sondern verschachtelt. Am Beispiel einer Single Density Diskette sei dies kurz dargestellt:

Die Reihenfolge der Sektoren bei SD ist im

- Checksum of the data with following break for the next sector.
- Next sectors like before.
- End of the track with clock pulses.

How is to be seen, a respectable proportion available of the always storage space goes on the diskette for identifier and lost check the actual data. From this the difference between the diskette capacity explains itself formatted and unformated. A specification of the track structure takes place in Chapter 13.

The sequence of the sectors in a track is not ascending or descending, the order is subject to certain obligations. For this a thought example:

A program is to be loaded, which consists of several sectors within a track. After reading of the first sector, e.g. number 3, its contents must be stored and read then the next. Now if the sector 4 after sector 3 would follow directly, then the sector would have itself 3 shifted, the diskette was naturally kept turning during the time, in and the heading would be with the next access already over sector 5. Then a whole revolution of the diskette would have to be almost waited for, in order to meet again on sector 4. This "play'" would repeat itself with each new sector and the time unnecessarily to the loading of this program would extend.

Here one avails oneself now a procedure, which is marked with "second gate Interlace." The sequence of the sectors is now no more in numeric order, but nested. By the example of a single Density diskette this is briefly represented:

The sequence of the sectors with SD is generally

Allgemeinen

0, 7, 5, 3, 1, 8, 6, 4, 2

d.h. auf Sektor 0 folgt Sektor 7 etc. Verfolgt man die Sektoren mit fortlaufenden Nummern, so ist erkennbar, daß zwischen zwei logisch aufeinanderfolgenden Sektoren 3 Stück mit abweichender Numerierung liegen. Die Anzahl der 'eingeschobenen' Sektoren wird mit 'Interleaving Factor' bezeichnet, der in diesem Fall 3 beträgt. Beim fortlaufenden Lesen innerhalb einer Spur kann nun die Behandlung der Daten des vorangegangenen Sektors während des 'Vorbeilaufens' von 3 Sektoren erfolgen, zum Lesen der gesamten Spur werden dann 4 Umdrehungen der Diskette benötigt. Bei Double Density wird ein Interleaving Faktor von 4 verwendet, da hier die Wartezeit aufgrund der höheren Datenrate sonst zu kurz wäre.

In gleichem Maß erhöht sich jedoch die Anzahl der Umdrehungen der Diskette die nötig sind, alle Sektoren in numerischer Reihenfolge (das ist der wahrscheinlichste Fall) zu lesen. Diese Anzahl berechnet sich aus Interleave-Faktor + 1

6.1. Wie werden nun Disketten formatiert?

Man verwendet dazu eine Level-1-Routine.

Im VDP-RAM wird ein PAB mit den Daten >0111 aufgebaut. Hierbei handelt es sich um eine Subroutine mit Namen '>11' und der Namenlänge >01. Dementsprechend muß der Namelength Pointer an @>8356 auf das Byte >01 im VDP-RAM zeigen. An der Adresse >834C (Byte) steht die Laufwerknummer, in >834D (Byte) steht die Anzahl der Spuren, die formatiert werden sollen. Einige Diskcontroller (Myarc) erlauben hier nur den Wert >28 (dez. 40), andere formatieren auch 1 bis 80 Spuren. Nach dem Formatieren steht hier die Anzahl der Sektoren pro Spur, mit der formatiert wurde. An

0, 7, 5, 3, 1, 8, 6, 4, 2

i.e. on sector 0 follows sector 7 etc. If one pursues the sectors with sequential numbers, then it is recognizable that between two logically following each other sectors are appropriate for 3 pieces with deviating numbering. The number of "inserted" sectors is marked with "Interleaving Factor," which amounts to in this case 3. With sequential reading within a track now the handling of the data of the preceding sector can take place during "running by" from 3 sectors, become reading the entire track then 4 revolutions of the diskette necessarily. With doubles Density is used a Interleaving factor of 4, since the waiting period would be otherwise too short due to the higher data rate here.

In same measure however the number of revolutions of the diskette those increases is necessary to read all sectors in numeric order (that is the most probable case). This number calculates itself from Interleave factor + 1.

How are diskettes formatted?

One uses for it a Level-1-Routine.

In the VDP RAM a PAB with the data is structured >0111. Here it concerns a subroutine with names ">11" and the name length >01. Accordingly the name length pointer at @>8356 must show to the byte >01 in the VDP RAM. At the address >834C (byte) is the drive number, in >834D (byte) is the number of tracks, which are to be formatted. Some disk controllers (Myarc) permit here only the value to >28 (dec. 40), others format also 1 to 80 tracks. After formatting is here the number of sectors per track, with which one formatted. >834E is on the address of a VDP buffer area, which must be

>834E steht die Adresse eines VDP-Pufferbereichs, der mindestens 6,5 KByte fassen können muß (etwa die Hälfte bei einfacher Dichte). An der Adresse >8350 steht im High-Byte die Dichte, mit der formatiert werden soll (>02 - doppelte Dichte, >01 - einfache Dichte), im Low-Byte steht die Anzahl der Seiten, die formatiert werden sollen (>01 — einseitig, >02 — doppelseitig.)

Nachdem alle Vorbereitungen getroffen wurden, wird der Befehl mit einem Unterprogrammaufruf der Art

```
BLWP @DSRLNK
DATA >A
```

ausgeführt. Da das DATA von der DSRLNK-Routine als Pointer-Offset in das DSR-ROM verwendet wird, werden nun keine DSR's sondern Unterprogramme angesprungen, ähnlich z.B. CALL FILES, was die selbe Tafel verwendet.

Nach einer solchen Formatierung kann die Diskette aber noch nicht verwendet werden (außer für Sektorkopierprogramme), es müssen erst die Sektoren 0 und 1 mit für das DOS wichtigen Daten über die Diskette beschrieben werden. Verwandte Level-1-Routinen werden in Kapitel 10 vorgestellt.

able to seize at least 6.5 KByte (about half with simple density). At the address >8350 the density is in the High byte, with which to be formatted is (>02 — double density, >01 — singe density), in the Low byte is the number of sides, which are to be formatted (>01 — on one side, >02 — on both sides.)

After all preparations were made, the instruction with a subroutine reference of the type

```
BLWP @DSRLNK
DATA >A
```

is executed. Since that is used DATA of the DSRLNK routine as pointer offset into DSR ROM, now no DSRs is branched to to separate subroutines, similarly e.g. CALL FILES, which uses the same table.

After such formatting the diskette cannot be used however yet (except for sector copying programs), it must only the sectors 0 and 1 also for the DOS important data over the diskette be described Related Level-1 Routine is introduced in Chapter 10.

7. Zugriff auf die Sektoren

Der Zugriff auf die einzelnen Sektoren einer Diskette ist in Maschinensprache mittels einer Level-1-Routine recht einfach möglich.

An der Adresse >834C befindet sich im High-Byte die Nummer des gewünschten Laufwerks, im Low-Byte befindet sich der sogenannte I/O-Opcode (>00 — Sektor schreiben, >01 oder generell <> 0 — Sektor lesen).

An der Adresse >834E steht die Adresse des 256 Byte-Puffers im VDP-RAM. An der Adresse >8350 steht die Nummer des gewünschten Sektors. Im High-Byte dieses Wortes wird nach Rückkehr der Fehlercode übergeben (>00 — kein Fehler).

Ein PAB wird mit >0110 aufgebaut, >8356 zeigt auf das Byte >01.

Die Ausführung erfolgt wie beim Formatieren mit

```
BLWP @DSRLNK
DATA >A
```

Nach dem Lesen befinden sich die Daten im VDP-Puffer und zwar selbst dann, falls beim Lesen ein Prüfsummenfehler auftrat. Es finden sich nur dann vollkommen sinnlose Daten, wenn der passende Sektor überhaupt nicht gefunden wurde.

Beim Schreiben von Sektoren muß beachtet werden, daß nur ein Schreibschutz an der Diskettenhülle Fehler unterbinden kann; ein etwaiger Fileschutz, der per Software das Überschreiben verhindern soll, ist bei Sektor-Operationen ohne Effekt!

Access to the sectors

That access to the individual sectors of a diskette is quite simply possible in machine language by means of a Level-1 Routine.

At the address >834C is in the High byte the number of the desired drive, in the Low byte is the so-called I/O Opcode (>00 — sector write, >01 or generally <> 0 — sector read).

At the address >834E is the address 256 of the byte buffer in the VDP RAM. At the address >8350 is the number of the desired sector. In the High byte of this word will after return the error code transferred (>00 — no error).

A PAB is structured with >0110, >8356 shows to the byte >01.

The execution effected as when formatting with

```
BLWP @DSRLNK
DATA >A
```

After reading the data are in the VDP buffer even then, if during the reading a check total error occurred. Are perfectly senseless data only if the suitable sector were not at all found.

When writing sectors it must be noted that only a write protection at the diskette jacket can prevent errors; a any file protection, which is to prevent the overwriting by software, is with sector operations without effect!

7.1. Art und Inhalt der Sektoren

Wer gelernt hat, direkt auf Sektoren zuzugreifen, wird sich für dieses Themengebiet sicher besonders interessieren.

Prinzipiell gibt es beim TI-99/4A vier verschiedene Sektorstrukturen:

1. Sektor 0, auch Root Sector genannt,
2. Sektor 1, die Directory-Sektor Liste,
3. die Directory Sektoren, einer für jedes File,
4. die Datensektoren, welche die reinen Daten der Files beinhalten.

Zu 1.: Sektor 0 beinhaltet alle wichtigen Informationen über die Diskette. Dies sind: Diskname, Anzahl der vorhandenen Sektoren, Anzahl der Spuren, Anzahl der Seiten, Schreibdichte, Anzahl der Sektoren pro Spur, Kopierschutz sowie die sogenannte 'Sector-Bitmap', in der belegte Sektoren markiert sind, damit diese nicht aus Versehen überschrieben werden. Vorgesehen war ein Datumsfeld, das beim Formatieren geschrieben werden sollte; in Ermangelung einer Hardwareuhr sind diese Bytes üblicherweise zu null gesetzt.

7.1.1. Aufbau von Sektor 0

Anhand eines Ausdrucks mit einem Disk-Editor soll der Inhalt von Sektor Null analysiert werden.

Type and contents of the sectors

Who learned to access directly sectors in this topic area will be surely particularly interested.

Always there are four different sector structures with the TI-99/4A:

1. Sector 0, also root second gate mentioned,
2. Sector 1, the directory sector list,
3. The directory sectors, one for each file,
4. The data sectors, which contain the pure data of the files.

To 1.: Sector 0 contains all important information about the diskette. These are: Disk name, number of available sectors, number of tracks, number of pages, density of writing, number of sectors per track, copy protection as well as the so-called ' second gate bit-map ', in which occupied sectors are marked, so that these are not overwritten by mistake. A date field, which should be written when formatting, was intended; in the absence of a hardware clock these bytes are usually set to zero.

Structure of sector 0

On the basis a printout with a disk editor is to be analyzed contents of sector zero.

```
Adr.  0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F  ASCII
-----------------------------------------------------------------
0000  47 52 41 50 48 49 4B 5F 5F 32 01 68 09 44 53 4B GRAPHIK__2*h*DSK
0010  20 28 01 01 00 00 00 00 00 00 00 00 00 00 00 00 (**************
0020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ****************
0030  00 00 00 00 00 00 00 00 FF FF 47 FF FF FF FF FF **********F*****
0040  FF FF FF FF FF FF FF FF FF A5 FF FF FF FF FF FF ****************
0050  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ****************
0060  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ****************
0070  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ****************
0080  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ****************
0090  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ****************
00A0  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ****************
00B0  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ****************
00C0  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ****************
00D0  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ****************
00E0  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ****************
00F0  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ****************
```

- Bytes 0 bis 9: Diskettenname, bei weniger als 10 Zeichen mit Blanks aufgefüllt.
- Bytes A und B: Gesamtzahl der Sektoren, hex 168, dez 360. Hier können, abhängig von der Formatierung, andere Werte stehen.
- Byte C: Anzahl der Sektoren pro Spur, hier 9, auch möglich sind 12 und 10 bei doppelter Dichte.
- Bytes D bis F: Formatierungsindex. Steht hier nicht 'DSK' dann hält ein Disk-Manager die Diskette für nicht formatiert.
- Byte >10: Protection Flag. Steht hier ein Blank, so ist die Diskette zum Kopieren freigegeben, steht hier ein 'P', ist ein normales Kopieren unmöglich.
- Byte >11: Anzahl der Spuren pro Seite, hier hex 28, dez 40.
- Byte >12: Seitenanzahl, >01 — einseitig, >02 — doppelseitig.
- Byte >13: Schreibdichte, >01 — einfache Dichte, >02 — doppelte Dichte.

- Byte 0 to 9: Diskette name, with fewer than 10 characters with blanks filled up
- Bytes A and B: Total number of the sectors, hex 168, decimal 360. Here, dependent on formatting, other values can be.
- Byte C: Number of sectors per track, here 9, also possible are 12 and 10 with double density.
- Byte D to F: Formatting index. Is not here "DSK" then considers a disk manager the diskette not formatted.
- Byte >10: Protection flag. Here if a blank is, then the diskette is approved for copying, is here a "P", is impossible normal copying.
- Byte >11: Number of tracks per page, here hex 28, decimal 40.
- Byte >12: Side's number, >01 — on one side, >02 — on both sides.
- Byte >13: Density of writing, >01 — single density, >02 — double density.

- Byte >14: Sektoren Pro Cluster (-1), >00 — direkte Adressierung, >01 — 2 Sektoren pro Cluster
- Bytes >15 - >37: Nicht benutzt. Diese Bytes waren für Erweiterungen vorgesehen und sind zu null gesetzt.
- Bytes >38 - >FF: Sektor Bitmap. Jedes gesetzte Bit gibt einen belegten Sektor an. Diese Karte ist 200 Bytes lang und kann daher die Belegung von bis zu 1600 (dez) Sektoren verwalten, entsprechend 400 KByte Daten. Jedes Byte beschreibt 8 Sektoren wie folgt:

```
      Bit  7 6 5 4 3 2 1 0
Sektor X + 7 6 5 4 3 2 1 0
```

Die Einträge in der Sektor-Bitmap sind nur bis zu einer bestimmten Position gültig. Dies liegt einfach daran, daß bei 360 freien Sektoren eben nur 45 Bytes notwendig sind, um alle Sektoren zu beschreiben. Bei 1440 Sektoren sind dann nur 180 der 200 Bytes notwendig. Die Sektor-Bitmap enthält somit immer Bytes, die keinen vorhandenen Sektor beschreiben, also diejenigen Bytes, die auf das letzte gültige Byte folgen. Um Fehlfunktionen des DOS zu vermeiden, müssen jedoch die nicht benutzten Bytes >FF enthalten. Ist dies nicht der Fall, so kann es passieren, daß Sektoren außerhalb der Diskette gesucht werden, was logischerweise eine Fehlermeldung erzeugt.

7.1.2. Arbeiten mit der Sektor-Bitmap

Beim Arbeiten mit der Sektor-Bitmap sind zwei Fragen zu beantworten:

- Erstens: Ist ein Sektor, dessen Nummer bekannt ist, belegt oder nicht, und
- Zweitens: Welche Sektoren beschreibt ein bestimmtes Byte in der Sektor-Bitmap?

- Byte >14: Sectors per cluster (-1), >00 — direct addressing, >01 — 2 sectors per cluster
- Byte >15 - >37: Does not use. These bytes were intended for extensions and are set to zero.
- Byte >38 - >FF: Sector bitmap. Each set bit indicates an occupied sector. This block is 200 bytes long and can therefore the allocation of up to 1600 (decimal) sectors administer, according to 400 KByte data. Each byte describes 8 sectors as follows:

```
      Bit  7 6 5 4 3 2 1 0
Sector X + 7 6 5 4 3 2 1 0
```

The entries in the sector bit-map are valid only up to a certain position. This is simply because of the fact that with 360 free sectors evenly only 45 bytes are necessary, in order to describe all sectors. With 1440 sectors then only 180 of the 200 bytes is necessary. The sector bitmap contains thus always bytes, which do not describe an available sector, thus those bytes, which follow after the last valid byte. Over malfunctioning of the DOS to avoid, must contain however the not used bytes of >FF. If this is not the case, then it can occur that sectors are looked up outside of the diskette, which logical-proves an error message produced.

Operating with the sector bitmap

When operating with the sector bit-map two questions are to be answered:

- First of all: Is if a sector, whose number admits is, occupies or not, and
- Secondly: Which sectors does a certain byte in the sector bitmap describe?

Beispiel zur ersten Frage: Es soll geprüft werden, ob Sektor 138 (dez) belegt ist. Jedes Byte der Sektor-Bitmap beschreibt, wie bereits gesagt, 8 Sektoren. Um das entsprechende Byte zu finden, muß die Nummer des Sektors durch 8 geteilt werden. 138 geteilt durch 8 ergibt 17 Rest 2. Der ganzzahlige Wert, 17, gibt an, in welchem Byte der Sektor-Bitmap sich das dem Sektor zugeordnete Bit befindet, der Rest gibt die Nummer des Bits in diesem Byte an. Die Dezimalzahl 17 ist gleich der Hex-Zahl 11. Da der Anfang der Sektor-Bitmap bei Byte >38 beginnt, muß >11 zu >38 addiert werden, um unser gesuchtes Byte zu finden. Es steht also an der Position >38 + >11 = >49. Byte >49 enthält >A5. Der Wert >A5 muß nun in eine Dualzahl umgewandelt werden. Man erhält

```
>A5 -> 1 0 1 0 0 1 0 1
```

Nun wird Bit 2 gesucht und festgestellt, daß dieses Bit gesetzt ist — der Sektor 138 ist also belegt!

Beispiel zur zweiten Frage: Was bedeutet das Byte an Position >3A?

Byte >3A enthält hex 47. Das Byte >3A ist 2 Byte vom Anfang der Sektor-Bitmap entfernt, das bedeutet, daß die Nummer des niedrigsten erfaßten Sektors 2 mal 8 ist, also dez 16 oder hex 10. Wandelt man nun den Inhalt des Bytes an >3A in die duale Schreibweise um, so erhält man

```
hex 47 -> 0 1 0 0 0 1 1 1 dual
```

Setzt man obiges Schema an, dann folgt:

```
                  0   1   0   0   0   1   1   1
    Sektoren frei x       x   x   x               Free sectors
  Sektoren belegt     x                 x   x   x  Occupied sectors
   Sektor Nr. hex  17  16  15  14  13  12  11  10  Sector No. Hex
```

Example for the first question: It is to be checked whether sector 138 (decimal) is occupied. Each byte the sector bitmap describes, like already said, 8 sectors. In order to find the appropriate byte, the number of the sector must be divided by 8. 138 divided by 8 results in 17 remainder of 2. The integral value, 17, indicates, in which byte the sector bitmap is that the sector assigned bits, the remainder indicates the number of the bit in this byte. The decimal number 17 is equal the hex number 11. Since the start the sector bitmap begins with byte >38, must >11 too >38 be added, in order to find our looked up byte. It is thus at the position to >38 + >11 = >49. Byte >49 contains >A5. The value >A5 must be converted now into a dual number. One receives

```
>A5 -> 1 0 1 0 0 1 0 1
```

Now is looked up and stated bit 2 that this bit is set — the sector 138 is thus occupied!

Example for the second question: What means the byte at position >3Á?

Byte >3Á contains hex 47. The byte >3Á is 2 bytes far away from the start the sector bitmap, that means that the number of the lowest entered sector is 2 times 8, thus decimal 16 or hex 10. If one converts now contents of the byte on >3Á into the binary way of writing, then one receives

```
hex 47 -> 0 1 0 0 0 1 1 1 binary
```

If one sets above pattern, then follows:

Wichtig ist noch zu wissen, daß beim Lesen von Files der Inhalt von Sektor 0 unwichtig ist, die Daten werden nur beim Schreiben verwendet!

7.1.3. Aufbau von Sektor 1

In Sektor 1 befinden sich nur Zahlen (12 Bit Worte). Jede Zahl gibt die Nummer eines Sektors an, in dem sich File-Daten befinden. Damit ist Sektor 1 das eigentliche Directory, eine Liste, in der die Adressen sämtlicher gültiger File-Directory Sektoren zu finden sind. Die Reihenfolge der Zahlen scheint nicht geordnet zu sein, sie ist es aber trotzdem. Es darf als bekannt gelten, daß die Filenamen beim Aufbau eines Directories in ihrer alphabetischen Reihenfolge auftreten. Dies wird dadurch erreicht, daß dieser Directory Sektor auf die verschiedenen File-Directory Sektoren derart verweist, daß deren Namen in alphabetischer Reihenfolge stehen. Es spielt also keine Rolle, in welcher Reihenfolge die einzelnen Files die Diskette bevölkern, lediglich Sektor 1 wird jedesmal neu sortiert.

Hier nun ein Beispielausdruck eines solchen Sektor 1.

Importantly it is to be still known that when reading files contents of sector 0 are insignificant, the data only during the writing is used!

Structure of sector 1

In sector 1 are only numbers (12 bits of words). Each number indicates the number of a sector, in which file data are. Thus sector 1 is the actual directory, a list, in which the addresses of all valid file directories sectors are to be found. The sequence of the numbers does not seem to be arranged, it is it however nevertheless. It may apply to be familiar that the file names occur with the structure of a directories in their alphabetical order. This is achieved by the fact that these directories sector refers sectors in such a manner to the different file directory that their name is in alphabetical order. It plays thus no role, in which order the individual files populate the diskette, sector 1 each time again is only sorted.

Here now an example printout such sector of a 1.

```
Adr.  0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F  ASCII
-------------------------------------------------------------------
0000  00 07 00 08 00 02 00 09 00 06 00 0A 00 03 00 0B ****************
0010  00 04 00 0C 00 05 00 00 00 00 00 00 00 00 00 00 ****************
0020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ****************
0030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ****************
0040  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ****************
0050  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ****************
0060  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ****************
0070  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ****************
0080  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ****************
0090  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ****************
00A0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ****************
00B0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ****************
00C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ****************
00D0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ****************
00E0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ****************
00F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ****************
```

Wort 1: 0007 File-Directory Sektor 'ASSM1'

Wort 2: 0008 File-Directory Sektor 'ASSM2'

Wort 3: 0002 File-Directory Sektor 'BFIX'

Wort 4: 0009 File-Directory Sektor 'BSCSUP'

Wort 5: 0006 File-Directory Sektor 'BUGOUTO'

Wort 6: 000A File-Directory Sektor 'DEBUG'

Wort 7: 0003 File-Directory Sektor 'DSKNAV'

Wort 8: 000B File-Directory Sektor 'EDIT1'

Wort 9: 0004 File-Directory Sektor 'EDITOR'

Wort A: 000C File-Directory Sektor 'SAVE'

Wort B: 0005 File-Directory Sektor 'SBUG'

Wort C: 0000 Ende der Directory Einträge.

Wird das erste Wort zu null gesetzt und alle weiteren nach hinten verschoben, so wird kein Directory erstellt (dies betrifft die gängigen Diskmanager wie das TI Modul, DM1000, Atronic und Basic Katalogprogramme), es werden aber alle Files noch geladen. Werden zwei Worte vertauscht, so kann das File mit im Alphabet vorangestelltem Filenamen nicht mehr gelesen werden! Prinzipiell ist es sogar fast vollkommen unerheblich, wie viele Nullwörter zwischen zwei Sektornummern stehen, solange die alphabetische Reihenfolge eingehalten wird und der Suchalgorithmus nicht gestört wird.

Ein Blick auf den beim Suchen eines Files verwendeten Algorithmus soll diese Besonderheit aufzeigen.

Gesucht wird ein File immer dann, wenn sein Name und die Diskettenstation bekannt ist, wo das File zu finden ist. Der betreffende Sektor 1 wird gelesen und sodann von der MITTE dieses Sektors aus, also dem 64. Wort, eine Sektornummer ermittelt. Wird nun ein Nullwort

Word 1: 0007 file directory sector "ASSM1"

Word 2: 0008 file directory sector "ASSM2'"

Word 3: 0002 file directory sector "BFIX"

Word 4: 0009 file directory sector "BSCSUP"

Word 5: 0006 file directory sector "BUGOUTO"

Word 6: 000A file directory sector "DEBUG"

Word 7: 0003 file directory sector "DSKNAV"

Word 8: 000B file directory sector "EDIT1"

Word 9: 0004 file directory sector "EDITOR"

Word A: 000C file directory sector "SAVE"

Word B: 0005 file directory sector "SBUG"

Word C: 0000 end of the directories of entries.

If the first word is set to zero and is to the rear shifted all further, then no directory created (this concerns the usual disk managers like DM 2, DM1000, Atronic and basic of catalog programs), it is still loaded however all files. If two words are exchanged, then the file can not be read also in the alphabet placed in front file name any longer! Always it is even almost perfectly insignificant, how many zero-words between two sector numbers are, as long as the alphabetical sequence is observed and the search algorithm is not disturbed.

A view of with looking files a used algorithm up is to point this special feature out.

A file is looked up whenever its name and the floppy station admit are, where the file is to be found. The sector concerned 1 is read and then by the CENTER of this sector out, thus that 64. Word, a sector number determines. Now if a zero word is read, then the incrementation is bisected

gelesen, so wird die Schrittweite halbiert und 32 Byte tiefer gesucht, bei erneuter Null ggf. 16 Byte tiefer usw. Wird eine Sektornummer gefunden, so wird der entsprechende FDS gelesen und der Filename mit dem angegebenen verglichen. Je nachdem, wie der Namensvergleich endet, kann aus den letzten beiden Bytes der Filenamen ermittelt werden, in welcher Richtung im Alphabet und damit im Sektor 1 weitergesucht werden muß; dabei wird permanent die Schrittweite halbiert. Das Suchen endet, wenn entweder der richtige FDS gefunden wurde, oder wenn die Schrittweite der Interpolation Null ist (File nicht gefunden).

Es handelt sich also um eine klassische Interpolation, oder aber um das Absuchen eines ausgeglichenen binären Baumes, wobei jede Sektornummer einen Knoten darstellt, von dem aus in der alphabetisch richtigen Richtung weiter verzweigt wird. Ein Nullknoten (-sektor) gibt dabei die Richtung 'nach unten' bzw. 'nach links' an.

Mit dieser Information läßt sich errechnen, daß ein beliebiger Filename, also der FDS eines beliebigen Files nach maximal 7 Versuchen gefunden wird, da der Logarithmus von 128 zur Basis 2, dem binären Zahlensystem, 7 ist.

Der weit verbreitete Disk Manager 1000 von Bruce Caron verwendet nicht die binäre Suche. Daher benötigt dieses Programm sehr viel mehr Zugriffe, um ein File zu finden bzw. umzukopieren. Bei einer großen Anzahl Files auf der Zieldiskette kann daher das DM 2 Modul oder jedes andere Programm, das auf Controllerroutinen zugreift, Geschwindigkeitsvorteile gegenüber DM1000 verbuchen.

7.1.4. Aufbau der File-Directory Sektoren (FDS)

Im Gegensatz zu den Sektoren 0 und 1, deren Bedeutung schon aus der Nummer zu erkennen ist, ist die Bedeutung der folgenden Sektoren 2

and looked up 32 byte more deeply, with renewed zero usually 16 byte of deep etc.. If a sector number is found, then the appropriate FDS is read and the file name with the indicated compared. Depending on how the name comparison ends, can be determined from the last two bytes of the file names, in which direction in the alphabet and thus in the sector 1 must be further looked up; permanently the incrementation is halved. The lookup ends if either the correct FDS is found, or if the incrementation of the interpolation is zero (file not found).

One concerns thus a classical interpolation, or however around searching a balanced binary tree, whereby each sector number represents a node, from which in the alphabetically correct direction continues to branch out. A zero-node (-sector) gives thereby the direction "downward" or "after left" on.

With this information that any file name is thus found, the FDS any files after max. 7 attempts, there the logarithm of 128 to the base 2, which is binary number system, 7, can be calculated.

The wide-spread Disk Manger 1000 of Bruce Caron the binary search does not use. Therefore this program needs more access, over a file to find very much or over too copy. At a large number of therefore the DM 2 module or every other program, which accesses controller routines, can book files on the target diskette rate advantages opposite DM1000.

Structure of the file directories sectors (FDS)

Contrary to the sectors 0 and 1, whose meaning is to be detected from the number, is always not determined the meaning of the following sectors

bis zum Ende der Diskette prinzipiell nicht festgelegt. Zwar werden die Sektoren 2 bis 21 (hex) bevorzugt mit File-Directory Sektoren belegt, je nach Belegung der Diskette können aber hier 'unten' auch Datensektoren zu finden sein, wie sich auch an höheren Sektoradressen File-Directory Einträge finden können. Zwar läßt es sich nicht für jeden Fall so einfach sagen, doch sind auf Disketten mit vielen kleinen Files auch mal FDS über >21 zu finden, große Dateien können das TI-DOS schon mal zwingen, Datensektoren in den FDS-Bereich zu verlegen. Wie Sektor 0 den Zustand der gesamten Diskette wiederspiegelt, so besitzt jedes File, ob nun Datei oder Programm, einen eigenen Sektor, der den Zustand des Files beschreibt.

2 up to the end of the diskette. The sectors 2 to 21 (hex) preferentially with file directory sectors are occupied, depending upon allocation of the diskette can be to find "down" also data sectors however here, how even at higher sector addresses entries file directory can be. It cannot be said for each case so simply, but is on diskettes with many small files also times FDS over >21 to be found, large files can the TI-DOS already times force to shift data sectors into the FDS area. As sector 0 again-reflects the status of the entire diskette, then each file possesses whether now file or program, an own sector, which describes the status files.

Dieser jeweilige File-Directory Sektor gibt an, um welchen Typ von File es sich handelt, wie viele und welche Sektoren es belegt usw. Damit sind diese Sektoren wohl die interessantesten Objekte. Bei der Bestimmung der Anzahl der von einem File belegten Sektoren wird dieser Sektor immer dazugezählt.

These respective file directory sector indicates, which type of file it concerns, like many and which sectors it occupies etc.. Thus these sectors are probably the most interesting objects. With the determination of the number of sectors occupied by a file this sector is always to it counted.

Nun wieder ein Beispielausdruck eines solchen Sektors.

Now again an example printout of such a sector.

```
Adr.  0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F  ASCII
-----------------------------------------------------------------
0000  41 44 52 45 53 53 39 39 20 20 00 00 8A 01 00 3D ADRESS99  *****=
0010  EA FE 3D 00 00 00 00 00 00 00 00 00 0B C3 03 00 jß=**********C**
0020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ****************
0030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ****************
0040  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ****************
0050  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ****************
0060  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ****************
0070  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ****************
0080  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ****************
0090  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ****************
00A0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ****************
00B0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ****************
00C0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ****************
00D0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ****************
00E0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ****************
00F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ****************
```

Die Bytes und ihre Bedeutung:

Bytes >0 - >9

Filename. Die Maximallänge ist wie bekannt 10 Bytes. Fehlende Bytes werden mit Blanks (>20) aufgefüllt. Es sind alle ASCII-Codes zulässig!

Bytes >A, >B

Reserviert, zu >0000 gesetzt.

Byte >C

File Status, gibt den Filetyp und einen etwaigen Schreibschutz an. Die Bedeutung der einzelnen Bits ist wie folgt:

Bit	*0*	*1*
7	FIXED — Feste Satzlänge	VARIABLE — variable Satzlänge
6	nicht benutzt	
5	nicht benutzt	
4	nicht benutzt	
3	Kein Schreibschutz	Schreibschutz gesetzt
2	nicht benutzt	
1	DISPLAY	INTERNAL
0	Daten-File	Program-File

Byte >D

Anzahl (VAR: Minimalzahl) der Datensätze (Records) pro Sektor. Im Gegensatz zu anderslautenden Veröffentlichungen wird dieses Byte sowohl bei VAR wie auch bei FIXED Dateien verwendet. Bei PROGRAM-Files ist dieses Byte immer >00.

The bytes and their meaning:

Bytes >0 - >9

File name. The maximum length is as admits 10 bytes. Missing bytes are filled up with blanks (>20). There is admissible all ASCII codes!

Bytes >A, >B

Reserved, set to >0000.

Byte >C

File status, indicates the type of file and any write protection. The meaning of the individual bits is as follows:

Bit	*0*	*1*
7	FIXED — fixed record length	VARIABLE — variable record length
6	not used	
5	not used	
4	not used	
3	no write protection	write protection set
2	not used	
1	DISPLAY	INTERNAL
0	Data file	Program file

Byte >D

Number (for VAR minimum number) of data records per sector. Contrary to some publications, this byte is used both with VAR as well as with FIXED files. With PROGRAM files this byte is always >00.

Bytes >E, >F

Gesamtzahl der vom File belegten Datensektoren. Hierzu muß aber noch der File-Directory Sektor addiert werden!

Byte >10

EOF-Offset im letzten Sektor. Dieses Byte wird nur benutzt bei PROGRAM-Files und Dateien mit variabler Satzlänge. Es gibt die Anzahl der Bytes im letzten belegten Sektor an und berechnet sich bei P-Files aus Länge MOD 256 oder einfach durch Ausmaskieren der beiden niederwertigsten Hex-Ziffern. Alle satzorientierten Dateien werden mit einem EOF-Symbol abgeschlossen, und zwar mit dem Byte >FF, das sich bei VAR-Dateien am Ende jedes Sektors befindet. PROGRAM-Files werden entgegen anderslautenden Angaben nicht mit >AA abgeschlossen, ein bestimmtes Kennungsbyte existiert nicht, da ja bereits dieser Zeiger auf das letzte Byte diese Information liefert. Damit ist eine Möglichkeit gegeben, die Länge von PROGRAM-Files auf das Byte genau zu bestimmen.

Byte >11

Satzlänge, wie sie bei der File-Eröffnung spezifiziert wurde. Bei P-Files zu Null gesetzt.

Bytes >12, >13

VAR-Dateien: Anzahl der mit Daten belegten Sektoren.

FIXED-Dateien: Anzahl der Datensätze (Records) im File.

Beide Bytes sind vertauscht, Low-Byte zuerst, daher müssen sie ge'swap't werden.

Bytes >14 bis >1B

Reserviert. Diese 8 Bytes sind zu null gesetzt.

Bytes >E, >F

Total number of data sectors occupied by the file. To this, however, the file directory sector must still be added!

Byte >10

EOF offset in the last sector. This byte only used with PROGRAM files and files with variable record length. It indicates the number of bytes in the last occupied sector and calculates themselves with p-files from length MOD 256 or simply by masking the two adjacent hex digits. All record-oriented files are locked with an EOF symbol, with the byte >FF, which for VAR files is at the end of each sector. PROGRAM files are not locked against different phrased specification with >AA, a certain identifier byte existed not, since already this pointer supplies this information on the last byte. Thus a possibility is given to exactly determine the length of PROGRAM files on the byte.

Byte >11

Record length, as it was specified during the file initialization. With p-files it is set to zero.

Bytes >12, >13

VAR files: Number of sectors containing data.

FIXED data files: Number of data records in the file.

Both bytes are exchanged, Low byte first, therefore they must be swapped.

Bytes >14 to >1B

Reserved. These 8 bytes are set to zero.

Vermutlich werden beim CC-40 und dem TI-99/5 die Bytes >14 und >15 für die Dateilänge in Bytes für Basic-Programme verwendet.

Bytes >1C und folgende

Dieser Bereich wird mit File-Link Map bezeichnet, gelegentlich auch mit Cluster-Block. Hier wird in Blöcken zu je 3 Byte aufgezeichnet, aus welchen örtlich getrennten Bereichen das File besteht. Hierzu wird in jedem 3 Byte-Block angegeben, wo ein Teilblock des Files beginnt und bis zu welchem Sektor-Offset der Teilblock reicht. Dabei hat der erste Datensektor logischerweise den Sektor-Offset 0. Die Reihenfolge der Nibbles ist vertauscht. Folgendes Schema ist dabei anzuwenden:

Vorliegende Reihenfolge:

```
Ss2 Ss1 So1 Ss3 So3 So2
```

wobei
Ss= Startsektor und
So= Sektor-Offset bedeuten.

Diese Reihenfolge muß in Ss3 Ss2 Ss1 und So3 So2 So1 umgestellt werden.

Wie man sieht, ergeben sich 12 Bit Worte. Mit dem Beispiel des o.ä. Sektors ergibt sich:

Vorher 0B C3 03, nachher >30B und >03C.

Da die nachfolgenden Bytes nur Nullen sind, ist zu erkennen, daß das betrachtete File aus nur einem Cluster (Block) besteht, der im Sektor >30B beginnt und an den sich >3C Sektoren anschließen (der Sektor mit dem Offset 0, also die Nummer >30B zählt natürlich mit). Der Sektor-Offset ist bei verteilten Files fortlaufend, so daß immer der letzte Offset-Wert von dem folgenden abgezogen werden muß, um die Anzahl der Sektoren im Cluster zu erhalten. Diese File-Link Map kann fast bis zum Ende des

Probably with the CC-40 and the TI-99/5 the bytes >14 and >15 are used for the file length in bytes for Basic programs.

Byte >1C and the following

This area marked with file link map, occasionally also with cluster block. Here in blocks to ever 3 byte one records, of which locally separated areas the file consists. For this in each 3 byte block is indicated, where a section block files begins and up to which sector offset the section block reaches. Has the first data sector logical-proves the sector offset 0. The sequence of the Nibbles is exchanged. The following pattern is to be applied with it:

The available sequence:

```
Ss2 Ss1 So1 Ss3 So3 So2
```

whereby
Ss = start sector and
So = sector offset mean in such a way.

This sequence must be changed over in Ss3 Ss2 Ss1 and So3 So2 So1.

How one sees, 12 bits of words result. With the example or the like Sector arises:

Beforehand 0B C3 03, afterwards >30B and >03C.

Since the following bytes are only zeros, that the regarded file consists of only one cluster (block), that is to be detected in the sector >30B begins and to the >3C sectors follows (the sector with the offset 0, thus the number >30B counts naturally with). The sector offset is to be received with distributed files sequentially, so that the last offset-value must be always taken off from the following, in order the number of sectors in the cluster. This file left map can grow almost up to the end of the file directory of

File-Directory Sektors wachsen! Fast deshalb, weil ein >000000-Tripel als Endesymbol nötig ist.

7.1.4.1. Ausnutzung der Bytes am Ende eines FDS

Es wird nur äußerst selten passieren, daß die Block-Link Map bis ans Ende eines File-Directory-Sektors reicht. Daher ist es denkbar, daß man z.B. die letzten 80 Bytes eines FDS benutzt, um dort zusätzliche Daten zum File unterzubringen. Das ist jedoch nicht ganz ohne Risiko, denn wenn die Block-Link Map dann irgendwann in diesen Bereich vorstößt, was bei jeder normalen Schreiboperation in das betreffende File passieren kann, dann werden nicht dazugehörige Datensektoren überschrieben.

Der dabei greifende Mechanismus ist bei Kenntnis der FDS-Struktur recht einfach zu verstehen:

Das beschriebene Szenario soll eine FIXED-Datei zugrunde legen.

Das I/O-System versucht bei einer APPEND- oder RANDOM-WRITE-Operation den Platz für einen neuen Eintrag zu ermitteln. Dabei wird anhand der Anzahl der Datensätze und deren Länge der derzeitige EOF-Offset berechnet. Dann wird in der Block-Link-Map der EOF-Offset ermittelt, der bis dahin nicht existierte. Üblicherweise trifft die DSR nun auf einen >000000-Eintrag und fordert einen neuen Cluster an, der korrekt über Sektor 0 (Bitmap) zugewiesen wird, sofern Platz dafür ist. Im Falle des falschen Clusterblocks wird, basierend auf einem total falschen Startsektor, ein File-Offset als zugewiesen angesehen, der es de facto nicht ist. Die DSR läßt sich in diesem Fall den Sektor nicht (mehr) zuweisen und schreibt folglich auf einen nicht zugewiesenen Sektor, der nach Murphy ein immens wichtiges File ruiniert.

sector! Almost, because a >000000 Tripel is necessary as end symbol.

Utilization of the bytes at the end of a FDS

It will occur only extremely rarely that those is enough block left map to to the end of a file directory sector. Therefore it is conceivable that one uses e.g. the last 80 bytes of a FDS, in order to accommodate there additional data for the file. That is however not whole without risk, because if those block link map into this area advanced then sometime, which can with each normal write operation into the file concerned occur, then not pertinent data sectors are overwritten.

The mechanism accessing thereby is quite simple with knowledge of the FDS structure to understand:

The described scenario is to put a FIXED data file as the base.

The I/O system tries to determine the place for a new entry with a Append or an Random WRITE operation. On the basis the number of the data records and their length the present EOF offset is calculated. Then in the block link map the EOF offset is determined, which did not exist up to then. Usually the DSR meets now on one >000000-Eintrag and requests a new cluster, which is assigned correctly over sector 0 (bitmap), if place for it is. In the case of the false cluster block, based on a totally false start sector, a file offset is regarded as assigned, which is it in fact not. The DSR can be assigned in this case the sector not (more) therefore and writes on an not assigned sector, which ruins after Murphy an immensely important file. If one has luck, the sector offset is situated outside of the diskette, which releases an error.

Wenn man Glück hat, liegt der Sektor-Offset außerhalb der Diskette, was einen Fehler auslöst.

Auf Disketten, auf denen aber nichts mehr geändert wird, also z.B. Back-Up's oder sowieso ganz volle Floppies, ist von Byte >22 bis zum Ende aller Platz für File-Informationen nutzbar. Ein 3-Byte Block mit Nullen muß aber mindestens als Trennung vorhanden sein, um das Ende der Block-Link Map zu indizieren!

7.1.5. Aufbau der Datensektoren

Inhalt und Aufbau der Datensektoren sind für die DSR-Software von geringer Bedeutung. Besondere Informationen enthalten nur Datei-Sektoren, da bei ihnen der Anfang und das Ende eines Datensatzes (Eintrags bzw. Records) bekannt sein müssen. Auch ist die optimale Ausnutzung des Speicherraums von der Datenstruktur abhängig.

Bei PROGRAM-Files fällt jegliche Strukturierung weg. Hier wird jeder Sektor soweit möglich vollkommen mit Daten gefüllt, lediglich im letzten Sektor können an dessen Ende einige Bytes übrig bleiben. Es wird aber auch dann ein neuer Sektor belegt, wenn nur ein Byte nicht mehr in den vorangegangenen Sektor passte. Bei der Strukturierung von P-Files in eigenen Assemblerprogrammen sollte daher die File-Länge immer ein Vielfaches von 256 (>100) betragen.

Bei satzorientierten Dateien variabler Länge steht vor jedem Datensatz ein Byte, welches die Länge des folgenden Eintrages bezeichnet. Bei Dateien vom Typ FIXED, also solchen mit fester Satzlänge, fehlt dieses Byte natürlich, da alle Einträge gleich lang sind und damit deren Position im Sektor einfach berechnet werden kann. Wichtig ist auch die EOS-Kennung (End-Of-Sector), die ebenfalls bei FIXED-Dateien nicht existiert. Sie steht ggf. am Ende eines

On diskettes, on which however nothing more, thus e.g. back-ups or anyway completely full Floppies is modified, up to the end all place for file information is usable by byte >22. A 3-byte block with zeros must however at least be available as separation, in order to indicate the end that block link map!

Structure of the data sectors

Contents and structure of the data sectors are for the DSR software of small importance. Special information contains only file sectors, there with them the at the beginning and the end of a data record (entry or. Records) admits to be must. Also the optimal utilization of the memory space depends on the data structure.

With PROGRAM files any structuring is omitted. Here each sector is filled possible perfectly so far with data, only in the last sector can remain remaining at its end some bytes. In addition, a new sector is occupied if only one byte no more did not fit into the preceding sector. With the structuring of p-files in own assembler programs therefore the file length should amount to always a multiple of 256 (>100).

With record-oriented files of VARIABLE length a byte before each data record indicates the length of the following entry. With files of type FIXED, this byte is unnecessary since all entries are the same length and their position in the sector can be simply calculated. Also important is the EOS identifier (End of Sector), which likewise does not exist with FIXED length files. It is usually at the end of an occupied sector.

belegten Sektors.

Da ein einzelner Datensatz nicht aufgebrochen werden darf, wird er, wenn im vorhandenen Sektor auch nur um 1 Byte zu wenig Platz ist, komplett in den nächsten Sektor geschrieben. Die optimale Raumausnutzung liegt dann vor, wenn die Länge der Datensätze (ggf. inclusive Längenbyte) ein ganzzahliger Bruchteil von 256 ist. So wird z.B. bei einer FIXED 64 Datei jeder Sektor optimal genutzt, bei VAR-Dateien ist ein Optimum wegen der variablen Satzlänge sehr schwer zu erreichen.

Since an individual data record may not be split, even if space in the available sector is exceeded by only 1 byte, it is written completely into the next sector. The optimal utilization of space is present if the length of the data records (if necessary inclusive length byte) are an integral fraction of 256. Thus e.g. with a FIXED 64 file each sector is used optimally, while with VARIABLE files it is very difficult to achieve an optimum because each record varies in length.

Aus dieser Verwaltung der Daten resultiert die Einschränkung, daß VAR-Dateien Satzlängen von maximal 254 Byte haben dürfen, FIXED dagegen 255 — es muß eben immer noch Platz für das Längen- und/oder EOS-Byte bleiben.

From this administration of the data the restriction that VAR files may have record lengths of max. 254 byte, results to FIXED against it 255 — evenly still place for the lengthening and/or EOS byte must remain.

Es besteht nun die Möglichkeit, bei einer VAR-Datei ein beliebiges Längenbyte durch ein >FF zu ersetzen. Der 'Erfolg' liegt auf der Hand: alle nachfolgenden Datensätze werden als nicht vorhanden angesehen (>FF setzt EOF-Flag), werden aber beim Kopieren mittels eines Diskmanagers, der sektororientiert vorgeht, mit kopiert. Damit lassen sich besonders TI-Writer Files schützen um z.B. persönliche Einträge unsichtbar zu machen. Man sollte sich aber zwecks Wiederherstellung der Daten die alten Längenbytes merken!

It exists now the possibility of replacing with a VAR file any length byte by an >FF. "success" is obvious: all following data records are not copied as available regarded (>FF sets EOF flag), however when copying by means of a disk manager, who proceeds sector-oriented, with. Thus TI-Writer files can be particularly protected around e.g. personal entries to make invisible. One should note however for recreation of the data the old length bytes!

Eleganter ist aber die Methode, einen ganzen Sektor verschwinden zu lassen, in dem er in der File-Link Map gewissermaßen 'weggeclustert' wird.

More elegant, however, is the method to let a whole sector in which a "cut away cluster" is made to disappear from the file link map to a certain extent.

Hierzu ein Beispiel:

For this an example:

Es soll der vorletzte Sektor aus unserem vorhergehenden Beispiel versteckt werden. Dazu muß der erste Block des Files verkürzt und ein zweiter erzeugt werden, der einen Sektor später beginnt. Aus den in die richtige Reihenfolge

The last but one sector from our preceding example is to be hidden. In addition must the first block files be shortened and second be produced, which begins a sector later. From the cluster data brought into the correct order

gebrachten Clusterdaten

>30B, >03C, >000, >000 muß werden

>30B, >03A, >347, >03B, Sektor >346 fehlt nun.

Nachdem diese Bytes in der richtigen Weise verdreht und geschrieben wurden, muß nun noch die Gesamtanzahl korrigiert werden, und der Sektor ist verschwunden. Wird das File gelöscht, so bleibt der so isolierte Sektor bestehen, da er von keiner Link-Map überdeckt wird! Er wird allerdings auch nicht mehr kopiert.

Außer z.B. mit dem Diskmanager-1000 (Option Sweep Disk) kann dieser Sektor nicht mehr gelöscht werden, es sei denn durch Formatieren.

Wie zu sehen ist, ist es nicht möglich die Struktur eines Datensektors anhand eines Beispiels zu verdeutlichen. Daher wird auf den gewohnten Sektor-Dump hier verzichtet. Der Leser ist aufgefordert, selbst seine Analysen vorzunehmen.

7.2. Zugriff auf einzelne Datensätze

Auf der höchsten Programmierebene des TI werden einzelne Datensätze ohne Beachtung ihres Inhalts, unabhängig von ihrer physikalischen Position auf der Diskette und ungeachtet einer eventuellen Fragmentierung der Datei gelesen. Beim Lesen von VAR-Dateien wird ein prinzipiell nicht vorhersagbares Längenbyte übergeben, die Optionen bei FIXED-Dateien erlauben neben dem einfachen APPEND auch ein Update bereits gespeicherter Datensätze.

Welche 'Kopfstände' die DSR der obersten Ebene (Level 2) machen muß, um an die Daten zu kommen, sei im Folgenden kurz umrissen. P-Files sind, da nicht satzorientiert, nicht angeführt.

>30B, >03C, >000, >000 must become

>30B, >03Á, >347, >03B, sector >346 is missing now.

After these bytes were rotated and written in the correct way, now still the total number must be corrected, and the sector disappeared. If the file is deleted, then the in such a way isolated sector remains existing, since it is covered by no link map! It is however also not copied more.

Except e.g. with the Disk Manager 1000 (option Sweep Disk) this sector cannot be deleted any longer, it is by formatting.

As is to be seen, it is to be clarified not possible the structure of a data sector on the basis an example. Therefore without the used sector dump one does here. The reader is requested to make even its analyses.

Access to individual data records

On the highest programming level of the TI individual data records without attention of their contents are read, independently of their physical position on the diskette and regardless of a possible fragmenting of the file. When reading VAR files not an always predictable length byte is transferred, the options with fixed data files permits beside the simple APPEND also an update of data records already stored.

Which must make "heading statuses" the DSR of the highest level (level 2), in order to come to the data, is briefly outlined in the following one. P-files are aforementioned, there not record-oriented.

7.2.1. Zugriff auf bestimmte Datensätze bei FIXED-Dateien

Bei FIXED-Dateien wird in der Regel ein bestimmter Datensatz über eine Satznummer angesprochen, ob nun zum Lesen oder Schreiben. Wichtig ist der Hinweis, daß auch beim sequentiellen Lesen einer FIXED-Datei diese Satznummer benötigt wird, die jedoch ggf. vom Interpreter/Compiler selbständig verwaltet wird.

Da jeder Datensatz die gleiche Länge aufweist, kann die DSR anhand der Sektorlänge, der Satzlänge und der Satznummer den Offset im File ermitteln, d.h. den Abstand in Sektoren und Bytes (im Zielsektor) vom Beginn der Datei. Diese Zahl wird gespeichert und es wird dann anhand der Block-Link-Map so lange gesucht, bis der Startsektor und der Offset in diesem Sektorblock bekannt sind. Sodann wird über die Satzlänge direkt der Sektor ausgelesen (die oben berechneten Bytes geben den Offset im Sektor an). Ein Überschreiten des EOF ist nicht möglich, da dieser im FDS vermerkt ist und leicht abgefragt werden kann.

Das Schreiben läuft ähnlich ab, jedoch gibt es hier einen Sonderfall, der besondere Beachtung verdient: Wird ein Datensatz angesprochen, dessen Satznummer über der des derzeitig letzten Eintrages in der Datei liegt, so werden alle Sätze bis zum gewünschten Satz hin erzeugt, ggf. also auch solche mit ungültigem Eintrag dazwischen, denn das Füllen der Datensätze ist Aufgabe des Anwenderprogramms, nicht der DSR. Wertet die DSR dabei ausschließlich die Block-Link-Map aus, so wirken sich Zusatzinformationen im FDS möglicherweise fatal aus (s.o.)

Access to certain data records with fixed data files

With fixed datafiles is usually addressed a certain data record via a record number whether now to reading or writing. Important the note that also during the sequential reading of a fixed data file this record number is needed, is however usually by the interpreter/compiler independently one administers.

Since each data record indicates the same length, the DSR can determine the offset in the file on the basis the sector length, the record length and the record number, ie. the distance in sectors and bytes (in the target sector) of the beginning of the file. This number is stored and it is then looked up on the basis the block link map so for a long time, until the start sector and the offset in this sector block admit are. Then over the record length directly the sector is picked out (the bytes calculated above to give the offset in the sector on). A exceeding of the EOF is not possible, since this is noted in the FDS and can be easily queried.

Writing runs similarly, however there is a special case, which earns here special attention: If a data record is addressed, whose record number is situated over that of the presently last entry in the file, then all records up to the desired record are produced, usually thus also such with invalid entry between them, because filling the data records is function of the user program, not the DSR. If the DSR analyses thereby excluding the block link map, then additional information in the FDS affects itself possibly fatal (see above).

7.2.2. Zugriff auf bestimmte Datensätze bei VAR-Dateien

Der Zugriff auf Datensätze einer Datei mit variabler Satzlänge gestaltet sich etwas schwieriger. Hier ist es nicht möglich, allein aus Satzlänge, Sektorlänge und Satznummer die Position eines Eintrages zu ermitteln, da ja jeder Eintrag Längen von 1 Byte bis zur spezifizierten maximalen Satzlänge aufweisen kann. Einzig das Anfügen neuer Datensätze ans Ende der Datei ist möglich, da ja der EOF-Offset im FDS vermerkt wird. Aus diesem und der ebenfalls im FDS vermerkten Sektoranzahl kann die Position eines neuen Eintrags einfach errechnet werden.

Der Zugriff auf eine bestimmte Eintragnummer ist nur durch sequentielles Lesen aller davor befindlicher Datensätze möglich. Zwar ermittelt die DSR intern, wieviele Sätze bei der aktuellen Länge in einen Sektor passen, doch ist das ziemlich witzlos, da es bei VAR-Dateien allenfalls statistischen Wert hat.

Access to certain data records with VAR files

That access to data records of a file with variable record length becomes somewhat more difficult. Here it is not possible, however from record length to determine sector length and record number the position of an entry since each entry can indicate lengths from 1 byte to the specified max. record length only. Adding new data records to the end of the file is possible, since the EOF offset is noted in the FDS. By this and the number of sectors likewise noted in the FDS the position of a new entry can be simply calculated.

That access to a certain entry number is possible only by sequential reading of all data records present before it. The DSR determines internally, how many records fit with the current length into a sector, but is that rather jokeless, since it has statistical value with VAR files if necessary.

8. Datenverwaltung im VDP-RAM

Es dürfte mittlerweile bekannt sein, daß das RAM des Video-Prozessors TMS 9929A/9918/9935 ... in der Konsole der einzige größere und zusammenhängende Speicherbereich des TI in der Grundversion ist.

Da es bei der Konzeption des TI-99/4A gefordert war, alle Erweiterungen wie Schnittstelle und Diskettenstation unabhängig vom Ausbauzustand der Anlage benutzen zu können, kommt eben diesem Speicherraum, der nur durch eine Pipeline von 2 Adressen zu erreichen ist, große Bedeutung zu. Erfahrene Assemblerprogrammierer werden jetzt vielleicht einwenden, daß es doch 4 Adressen gibt, über die man den Video Prozessor ansprechen kann. Das ist zwar nicht falsch, durch die Benutzung von 4 Adressen werden aber lediglich bestimmte Registeranwahlleitungen des VDP gesetzt, so daß die Register logisch getrennt angesprochen werden können.

In diesem Kapitel soll gezeigt werden, welche für die Diskettenverwaltung relevanten Informationen sich im VDP-RAM befinden, was sie bedeuten und was mit ihnen angestellt wird.

Die Belegung des VDP-RAMs wird mittels einiger Pointer im Scratch Pad RAM der Konsole verwaltet, das bekannterweise der einzige CPU-adressierbare 16 Bit Bereich des TI 99/4A in der Grundversion ist. Strenggenommen ist dabei nur der Pointer >8370 festgelegt — alle anderen werden speziell von der Diskcontroller-DSR beim Aufruf oder beim File-Handling verwendet.

Diese Pointer sind:

- >8356 Arbeitszeiger auf die momentan zu bearbeitenden Daten. Dieser Zeiger ist als POINT in Sachen DSRLNK bekannt, wird aber so stark frequentiert und geändert, daß

Data administration in VDP RAM

It might meanwhile admits to be that that RAM of the video processor TMS 9929A/9918/9935 ... in the console the only larger and connected storage area of the TI in the basic version is.

Since it was required with the conception of the TI-99/4A to be able to use all extensions such as interface and floppy station independently of the development status of the system evenly great importance is attached to this memory space, which can be attained only by a pipeline from 2 addresses to. Experienced assembler programmers will now perhaps object that there are nevertheless 4 addresses, over whom one the video processor respond can. That is not false, by the use of 4 addresses however only certain register selection lines VDP is set, so that the registers can be addressed logically separately.

In this chapter is to be demonstrated, which the diskette administration relevant information in the VDP RAM to be, what it to mean and which is employed with them.

The allocation of the VDP RAM is administered by means of some pointers in the Scratch PAD RAM of the console, which well-known-proves only CPU addressable 16 bits the area of the TI 99/4A in the basic version is. Strict-taken thereby only the pointer is >8370 determined — all different are used particularly of the disk controller DSR with the call or with the file handling.

These pointers are:

- >8356 work pointers on those at the moment to processing data. This pointer is well-known as POINT in things DSRLNK, is however so strongly frequented and modified

diese eine Bedeutung fast nebensächlich ist.

- >8358 Zeiger auf den Volume ID Block. Dieser Pointer zeigt auf den Beginn des Speicherbereiches, in dem sich eine Kopie von Sektor 0 der zuletzt angesprochenen Diskette befindet.

- >8366 VDP-Stackpointer. Aus den eingangs gezeigten Gründen ist es für die DSR Software, welche es auch sei, problematisch, mit den effektiv nur 12 verfügbaren Registern auszukommen. Daher werden Rückkehradressen oder globale Pointer zwischenzeitlich im VDP-Stack Bereich abgelegt. Dieser Pointer wird mit jedem neuen Eintrag nach unten, also zu niedrigeren Adressen hin, verschoben, und zeigt immer auf eine freie Speicherstelle.

- >8370 Zeiger auf die höchste verfügbare VDP-Adresse bzw. den VDP-Area-Link. Nach einem CALL FILES(3), das ist der Zustand nach Anschalten des Rechners und Ausführung der Power-Up-Routine des Diskcontrollers, enthält dieser Pointer üblicherweise den Wert >37D7, also ein Byte unterhalb der Disk-Puffer Zone. Ist ein Modul gesteckt, welches über eine eigene Power-Up-Routine verfügt, so zeigt >8370 auf das Byte im VDP-RAM unterhalb dieses Bereiches. Befindet sich eine DSR mit Power-Up-Sequenz auf einer CRU-Seite vor dem Diskcontroller, ist der gesamte Disk-Pufferbereich um die hier reservierte Byteanzahl nach unten verschoben.

Wie die folgende Speicherbelegung zu zeigen scheint, tummeln sich im oberen VDP-RAM Bereich ausschließlich Disketteninformationen. Das ist aber nur verständlich, wenn man sich die vielfältigen Funktionen vor Augen führt, die ein Diskettenmanagement haben muß. Näheres hierzu enthalten die ROM-Listings zu den Diskcontrollern. Die Adreßangaben gelten nur,

that this is almost unimportant a meaning.

- >8358 pointers on the volume ID block. This pointer points to the beginning of the storage area, in which a copy of sector 0 of the diskette addressed last is.

- >8366 VDP stack pointer. For the reasons initially shown it is problematic for the DSR software, whatever it is, to get along with that effectively only 12 available registers. Therefore return addresses or global pointers are stored in the meantime in the VDP stack area. This pointer is shifted with each new entry downward, thus to lower addresses, and always points to a free storage point.

- >8370 pointers to the highest available VDP address or. VDP AREA left. After a CALL FILES(3), that is the status after turning the computer on and execution of the power-up routine of the disk controller, contains this pointer usually the value >37D7, thus a byte below the disk buffers zone If a module is put, which has its own power UP routine, then shows >8370 to the byte in the VDP RAM below this area. If a DSR with power UP sequence on a CRU page before the disk controller is, the entire disk buffer area is downward shifted the number of bytes reserved here.

Like the following storage allocation to show, tummeln in the upper VDP RAM area seems itself excluding diskette information. That is however understandable only if one leads oneself the various functions before eyes, which a diskette management must have. Details for this the ROM listings contain to disk controllers. The address specifications apply only, if no coming

sofern kein Gerät auf CRU >1000 VDP-RAM für sich allokiert!

8.1. Funktion des VDP-Area-Links

Wie eingangs dieses Kapitels angedeutet, steht das VDP-RAM prinzipiell allen anderen DSRs ebenfalls zur Verfügung. Um diese Mehrfachnutzung ohne Kollisionen oder Überschneidungen der einzelnen Bereiche zu gewährleisten, hat sich TI etwas trickreiches einfallen lassen:

Beim Power-Up (dem Einschalten des Systems), wird vom Konsolen-Betriebssystem die (festgelegte) höchste freie Adresse im VDP-RAM an die CPU-Adresse >8370 geschrieben. Diese Adresse ist bekannterweise >3FFF, was aus dem Design des VDP-Chips bzw. dessen Adressierungslogik herrührt. Jede DSR bzw. jedes Modul, welche(s) über eine Power-Up-Routine verfügt, kann durch Vermindern des Inhaltes von >8370 und Pflegen einer einfach verketteten Liste (s.u.) VDP-Speicher 'abzweigen'. Die jeweilige Routine muß dazu lediglich den aktuellen Inhalt von >8370 als VDP-RAM-Obergrenze akzeptieren, diesen Wert in die Kette im VDP-RAM eintragen, den Inhalt von >8370 um den Betrag reduzieren, der an Speicher benötigt wird und einen Vermerk in die Liste aufnehmen, anhand dessen die DSR später den selbst reservierten Bereich erkennt. Bei DSRs auf P-Box-Karten ist dies sinnvollerweise die CRU-Seite, bei Modulen kann dies der GROM-Zähler o.ä. sein (TE II vermerkt z.B. eine >40).

Sobald nun eine DSR auf das VDP-RAM zugreifen will, sucht sie den ersten Eintrag der Liste (auf den >8370 zeigt), prüft das ID-Byte und sucht ggf. an der Adresse weiter, welche in der Liste vorhanden ist (VDP-Area-Link).

on CRU allocated >1000 VDP RAM for itself!

Function of VDP area links

As initially this chapter suggested, is that to VDP RAM always all other DSRs likewise at the disposal. In order to ensure, TI could this multiple use without collisions or overlaps of the individual areas be broken in somewhat sophisticated:

With the power-up (the system power-on), by the console operating system (fixed) the highest free address in VDP RAM is written to CPU address >8370. This address is the familiar >3FFF, which is due from the design of the VDP chip and/or its addressing logic. Each DSR and/or each module, some (s) had a power UP routine, can by decreasing contents of >8370 and care of a simply concatenated list (see below) VDP memory "branches". The respective routine must accept in addition only current contents of >8370 as VDP RAM upper limit, register this value into the chain in the VDP RAM, reduce contents of >8370 by the amount, which is needed at memory and a note to the list to take up, on the basis whose the DSR recognizes reserved range later. With DSRs in PEB cards this is naturally the CRU side, with modules can this the GROM counter or the like exist (TE II note e.g. one >40).

As soon as now a DSR wants to access VDP RAM, it looks the first entry up of the list (on the >8370 shows), checks the ID byte and looks up usually at the address further, which is available in the list (VDP area link).

8.2. Memory-Map des oberen Teils des VDP-RAMs

VDP-Area-Link, 4 Bytes

>37D8	Ident-Code eines allokierten Blocks	>AA
>37D9	Zeiger auf Vorgängerliste (>8370 alt)	>3FFF
>37DB	CRU-Seite als Bereichskennung	>11
>37DC	Maximalzahl gleichzeitig offener Files	>03

Filedaten des zuerst geöffneten Files, 518 Bytes

Zeigerblock zum File, 6 Bytes

>37DD	Zeiger auf den aktuellen File Record (Offset)
>37DF	Sektornummer des File-Directory Sektors
>37E1	Logischer File Record Offset bei VAR-Dateien
>37E2	Laufwerknummer, in dem sich die Diskette mit dem File befindet

Beginn des Puffers für den File-Directory Sektor, 256 Bytes

>37E3	Filename — 10 Bytes
>37ED	Reserviert >0000
>37EF	File Status
>37F0	Maximalzahl der Einträge (Records) pro Sektor
>37F1	Gesamtzahl bisher belegter Sektoren, incl. File-Dir Sektor!
>37F3	EOF-Offset (Eintrag Nr.) im letzten belegten Sektor
>37F4	Satzlänge der Einträge (Record Length)
>37F5	Record Anzahl bei FIXED-Dateien, Anzahl der Sektoren bei bei VARIABLE-Dateien. Die Bytes müssen vertauscht werden (SWAP)!
>37F7	Reserviert >0000,>0000,>0000,>0000
>37FF	File Link Daten. Geben an, ob das File aufgebrochen wurde und wenn ja, wo die Teile beginnen und wie lang sie sind. Näheres siehe Beschreibung der File-Directory Sektoren.

Memory map of the upper section of the VDP RAM

VDP Area Link, 4 Bytes

>37D8	Ident code of a allokierten block	>AA
>37D9	Pointer on predecessor list (>8370 old)	>3FFF
>37DB	CRU page as area identifier	>11
>37DC	Maximum number of at the same time open files	>03

File data of first opened file, 518 bytes.

Pointer block to the file, 6 bytes

>37DD	Pointer on the current file record (offset)
>37DF	Sector number of the file directory of sector
>37E1	Logical file record offset with VAR files
>37E2	Drive number, in that the diskette with the file is

Beginning of the buffer for the file directory sector, 256 bytes

>37E3	File name — 10 bytes
>37ED	Reserved >0000
>37EF	File status
>37F0	Maximum number of entries (records) per sector
>37F1	Total number of occupied sectors, including file directory sector!
>37F3	EOF offset (entry no.) in the last occupied sector
>37F4	Record length of the entries (record length)
>37F5	Record number with fixed data files, number of sectors with variable files. The bytes must be exchanged (SWAP)!
>37F7	Reserved >0000, >0000, >0000, >0000
>37FF	File link data. It indicates whether the file was broken open and if, where the parts begin and are as long them. Details see description of the file directories sectors.

>38E3 Beginn des Datenpuffers — 256 Bytes

Filedaten des an zweiter Stelle geöffneten Files, 518 Bytes

>39E3 Beginn des Zeigerblocks — Bytes

>39E9 Beginn des File-Dir Puffers — 256 Bytes

>3AE9 Beginn des Datenpuffers — 256 Bytes

Filedaten des an dritter Stelle geöffneten Files, 518 Bytes

>3BE9 Beginn des Zeigerblocks — 6 Bytes

>3BEF Beginn des File-Dir Puffers — 256 Bytes

>3CEF Beginn des Datenpuffers — 256 Bytes

VDP-Stack Bereich, 252 Bytes

>3DEF Unteres Ende

>3EEA Oberes Ende

Laufwerk Informationsblock, 4 Bytes

>3EEB Nummer des zuletzt angesprochenen Laufwerks

>3EEC Aktuelle Spur, über der der Kopf von LW 1 steht

>3EED Aktuelle Spur, über der der Kopf von LW 2 steht

>3EEE Aktuelle Spur, über der der Kopf von LW 3 steht

Unbenutzter Bereich — 5 Bytes

>3EEF

>3EF3

Beginn des Sektor 0 Puffers — 257 Bytes

>3EF4 Laufwerknummer, von der Sektor 0 stammt — 1 Byte

>3EF5 Sektor 0 Puffer der Diskette, auf die zuletzt GESCHRIEBEN wurde - 256 Bytes

Puffer für den Vergleich von Filenamen - 11 Bytes

>3FF5 Laufwerknummer - 1 Byte

>3FF6 Filename - 10 Bytes

>38E3 Beginning of the scratchpad memory — 256 bytes

File data in second place opened files, 518 bytes

>39E3 Beginning of the pointer block — bytes

>39E9 Beginning file you buffer — 256 bytes

>3AE9 Beginning of the scratchpad memory — 256 bytes

File data in third place opened files, 518 bytes

>3BE9 Beginning of the pointer block — 6 bytes

>3BEF Beginning file directory buffer — 256 bytes

>3CEF Beginning of scratchpad memory — 256 bytes

VDP stack area, 252 bytes

>3DEF Lower end

>3EEA Upper end

Drive information block, 4 bytes

>3EEB Number of the drive addressed last

>3EEC Current track, over that the head of DSK1 is

>3EED Current track, over that the head of DSK2 is

>3EEE Current track, over that the head of DSK3 is

Unused area — 5 bytes

>3EEF

>3EF3

Beginning sector 0 of buffer — 257 bytes

>3EF4 Drive number, of the sector 0 — 1 byte comes

>3EF5 Sector of 0 buffer of the Diskette, that was WRITTEN last — 256 bytes

Buffers for the comparison of file name — 11 bytes

>3FF5 Drive number — 1 byte

>3FF6 Filename — 10 bytes

8.3. Detaillierte Beschreibung der Daten

8.3.1. Bereich >37D8 - >37DC

Dieses Datenfeld kennzeichnet für die Diskcontroller-Software den Beginn des mit CALL FILES( ) freigegebenen Bereiches. Diese 5 Bytes *müssen* am Anfang des Disk-Pufferbereiches stehen, sonst droht bei bestimmten Controllern ein Absturz. Es reicht also nicht, ein CALL FILES durch direktes Beeinflussen des Pointers an >8370 zu simulieren, auch der Header des Pufferbereiches muß stimmen! Aus der Speicherbelegung ist direkt zu sehen, was bei einem CALL FILES passiert: Der Header wird um 518 Bytes (oder ein mehrfaches davon) nach unten oder oben versetzt und ein Disk Puffer Bereich entfernt oder hinzugefügt.

8.3.2. Bereich >37DD - >37E2

Der Record Zeiger (es gibt eigentlich 2) zeigt, wie aus dem Basic wohl bekannt, auf den aktuellen Datensatz. Besondere Bedeutung kommt aber der Sektornummer des File-Dir Sektors zu. Wird ein File mit der 'DELETE' Option geschlossen, so wird mithilfe dieser Zahl im Sektor 1 der betroffenen Diskette der Sektor gesucht und aus der Link-Map entfernt. Ist bei der Laufwerknummer das höchstwertige Bit gesetzt, so wurde der Datensektor beim letzten Zugriff aktualisiert. Nach einem RESTORE ist der File-Record Zeiger >FFFF und der File-Record Offset >00, ebenso direkt nach der Eröffnung im Output- oder Append-Mode.

8.3.3. Bereich >37E3 - >39E2

Puffer für den File-Directory Sektor und den aktuellen Datensektor. Diese Daten wurden bereits im Kapitel 5 abgehandelt. Es gibt jedoch Besonderheiten, welche die Kennung freier bzw. belegter File-Control Blöcke betreffen. Ist das

Detailed description of the data

Area >37D8 - >37DC

This data field indicates the beginning of the area released with CALL FILES() for the disk controller. These 5 bytes *must* be at the start of the disk buffer area, otherwise a crash threatens with certain controllers. It is not thus enough to simulate, a CALL FILES by direct influencing of the pointer on >8370, also the header of the buffer area must be correct! From the storage allocation is to be seen direct, what with a CALL FILES occurs: The header is shifted downward or above by 518 bytes (or a repeated of it) and disk is removed or added a buffer area.

Area >37DD - >37E2

Of the records pointers (there are actually 2) points, as from the basic probably admits, to the current data record. Special meaning is attached however to the sector number file you sector. If a file with the "DELETE" option is closed, then assistance of this number in the sector 1 of the diskette concerned the sector is looked up and removed from the link map. If the most significant bit is set with the drive number, then the data sector was updated with the last access. After a RESTORE the file record pointer is just as direct >FFFF and the file record offset >00, after the initialization in the output or Append mode

Area >37E3 - >39E2

Buffer for the file directory sector and the current data sector. These data were already handled in Chapter 5. There are however special features, which the identifier more freely or. occupied file control concern blocks. If the first

erste Byte des Filenamenbereiches gleich >00, so ist dieser Puffer nicht belegt und kann neu verwendet werden. Ist das erste Byte ein Blank (Leerzeichen >20), dann ist dieser Puffer für die Daten beim Aufbau eines Directory reserviert. Ist das höchstwertige Bit im 1. Byte des Filenamens gesetzt, so ergaben sich Änderungen im FDS. Dieser muß dann beim Abschluß einer File-Operation (CLOSE o.ä.) neu geschrieben werden. Da dies im Allgemeinen vom DOS erledigt wird, ist ein gesetztes High-Bit nur bei Fehlfunktionen des DOS zu erkennen.

8.3.4. Bereich >39E3 - >3DEE

Daten der weiteren Files, Bedeutung wie 1. File.

8.3.5. Bereich >3DEF - >3EEA

Hier werden zwischendurch Rückkehradressen und globale Registerinhalte gesichert. Bei Verwendung dieses Stacks in eigenen Programmen ist darauf zu achten, daß er so verlassen wird, wie er angetroffen wurde, der Pointer auf >8366 stimmt und insbesondere der zulässige Bereich nicht überschritten wird, wodurch die File Daten zerstört würden. Fassungsvermögen 126 Worte.

8.3.6. Bereich >3EEB - >3EEE

Dieser Bereich wird nur von solchen Diskcontrollern benutzt, die kein eigenes System-RAM besitzen, so z.B. der TI-Controller. Andere Controller aktualisieren lediglich die Laufwerknummer. Für die Steuerung der Kopfposition der Diskettenlaufwerke ist es nötig zu wissen, wo sich der Schreib/Lesekopf im Moment befindet. Diese Bytes dienen als Puffer dafür.

byte of the file name area is equal to >00, then this buffer is not occupied and cannot again be used. If the first byte is a blank (blank >20), then this buffer for the data is reserved with the structure of a directory. The most significant bit is in the 1. Set, then modifications in the FDS resulted byte of the file name. This must then at the time of the termination of a file operation (CLOSE or the like) again to be written. Since this is completed generally by the DOS, a set High bit is to be detected only with malfunctioning of the DOS.

Area >39E3 - >3DEE

Data of the further files, meaning like 1. File.

Area >3DEF - >3EEA

Become occasionally return addresses and global register contents here secured. In the case of use of this stack in own programs it is to be made certain that he will leave in such a way, how he was found, which tunes pointers on >8366 and in particular the admissible area is not exceeded, whereby the file data were destroyed. Capacity of 126 words.

Area >3EEB - >3EEE

This area is only used by disk controllers which do not possess their own system RAM, e.g. the TI controller. Other controllers update only the drive number. For the controlling of the heading position of the floppy disk drives it is to be known necessarily, where the write/read head is for the moment. These bytes serve as buffers for it.

8.3.7. Bereich >3EEF - >3EF3

Eine Verwendung dieser 5 Bytes ist momentan nicht bekannt, der Bereich ist aber wohl zu klein, um für eigene Anwendungen interessant zu sein.

8.3.8. Bereich >3EF4 - >3FF4

Hier befindet sich immer eine Kopie des Sektor 0 derjenigen Diskette, auf die zuletzt geschrieben wurde. Alle Änderungen der Sektor Bitmap werden in diesem Bereich vorgenommen, anschließend wird der Sektor zurück-geschrieben. Die Laufwerknummer gibt wiederum im höchstwertigen Bit ein evtl. durchgeführtes Update an.

8.3.9. Bereich >3FF5 - >3FFF

File Name Compare Buffer (FNCB). Hier werden bei Bedarf die vom Programm geforderten Filenamen mit den auf der Diskette angetroffenen verglichen. Wird ein Filename also irgendwann spezifiziert, dann wird er hier hineinkopiert, die Laufwerknummer vorangestellt und auf der Diskette nach diesem Namen gesucht. Die Funktionsabläufe im Einzelnen sind sehr komplex, hier muß wieder auf detaillierte ROM-Listings verwiesen werden.

8.4. Noch einige Hinweise und Tips

Soll ein File gelöscht werden, so sucht die Diskcontroller DSR nach dem Namen zuerst bei den offenen Files. Soll keines von denen gelöscht werden, so wird nach einem nicht geöffneten File gesucht, um in diesem Bereich die Daten zu bearbeiten. Existiert kein freier Platz im VDP-RAM, so kann kein File gelöscht werden! In diesem Fall muß ein File geschlossen werden.

Beim Löschen eines Files wird dann KEINE Fehlermeldung gegeben, wenn das File nicht existierte. Das ist eine Nachlässigkeit der Softwareentwickler von TI gewesen, die auch bei

Area >3EEF - >3EF3

A use of these 5 bytes is not at the moment well-known, the area is however probably too small, in order to be interesting for own applications.

Area >3EF4 - >3FF4

Always is here a copy sector 0 of that diskette, on which one wrote last. All modifications the sector bitmap are made within this area, written back afterwards the sector. The drive number gives again in the most significant bit perhaps executed update on.

Area >3FF5 - >3FFF

File name Compare Buffer (FNCB). Here if necessary the file names required by the program are compared with on the diskette the found. If a file name is specified thus sometime, then it is in-copied here, the drive number and looked up on the diskette is placed in front for this name. The sequences of functions in detail are very complex, here must again to detailed ROM listings be referred.

Still some notes and tips

If a file is to be deleted, then the disk controllers look DSR up for the name first with the open files. None of those is to be deleted, then for an not opened file one looks up, in order to process within this area the data. If no free space in the VDP RAM exists, then no file can be deleted! In this case a file must be closed.

With the deletion files NO error message is given if the file did not exist. That was a carelessness of the software developers of TI, which is to be found also at all foreign manufacturers

allen Fremdherstellern zu finden ist.

Beim Eröffnen einer Datei wird deren FDS gelesen, sofern er existierte, oder ein leerer, neuer FDS auf der Diskette erzeugt. In jedem Fall aber befindet sich die einmal ermittelte Laufwerknummer vor dem Beginn der Filenamenzone. Diese Nummer bleibt bestehen, solange das File offen ist und wird nicht mehr verifiziert. Gleiches gilt für den Volume-ID Block, dessen Laufwerknummer bis zum Schreiben des ID-Blockes oder dem Lesen eines neuen gültig bleibt. Wird nun nach dem Öffnen einer Datei im Output- oder Append-Mode die Diskette gewechselt, ein Datensatz geschrieben und die Datei geschlossen, so bekommt die eingelegte Diskette den Sektor 0 der ursprünglich eingelegten Diskette 'verpaßt'. Die Folgen sind weitreichend.

When opening a file their FDS is read, if he existed, or an empty, new FDS on the diskette produces. In each case however is the once determined drive number before the beginning of the file name zone. This number remains existing, as long as the file is open and no more not verified. Same applies to the volume ID block, whose drive number up to the writing of the ID block or reading a new remains valid. Now to opening a file in the output or Append mode if the diskette is changed, if a data record is written and if the file is closed, then the inserted diskette gets the sector 0 of the originally inserted diskette "missed." The consequences are extensive.

Bestand das File auf beiden Disketten in allen Sektoren identisch, dann ist das File auf der eingelegten Diskette noch in Ordnung. War jedoch auf der zweiten Diskette ein Kopierschutz über 'P' in Sektor 0 und keiner auf der ersten Diskette, so ist die zweite Diskette nun nicht mehr geschützt.

If the file insisted on both diskettes in all sectors identically, then the file on the inserted diskette still is correct. However if a copy protection over ' P ' was in sector 0 and none on the first diskette on the second diskette, then the second diskette is now no longer protected.

Es klingt unglaublich, aber so kann ein Kopierschutz der Diskette auch in Basic entfernt werden!

It sounds unbelievable, but so a copy protection of the diskette can be removed also in Basic!

Voraussetzung ist aber, daß auf der geschützten Diskette noch mindestens 2 Sektoren frei sind, in die das neu eröffnete File geschrieben werden kann. Der Filename darf nicht bereits bestehen, da sonst vorhandene FDS verfälscht werden.

A prerequisite is however that on the protected diskette still at least 2 sectors are free, into which the again opened file be written can. The file name may not already exist, since otherwise available FDS is falsified.

Hier ein kurze Anleitung für DSK1:

Here a short guidance for DSK1:

Auf beiden Disketten wird ein File, z.B. DSK1.PLACEBO im DIS/VAR 80 Format oder irgendeinem anderen eröffnet. Dieser Filename darf auf der zweiten, zu manipulierenden Diskette noch nicht existieren. Das File wird

On both diskettes a file, e.g. DSK1.PLACEBO in the DIS/VAR 80 format or any different one is opened. This file name may exist on the second, diskette which can be manipulated yet. The file is again closed right after the initialization. Now

gleich nach der Eröffnung wieder geschlossen. Nun wird die erste, ungeschützte Diskette eingelegt, das File im Append-Mode wieder eröffnet und die geschützte Diskette eingelegt. Nun wird ein Datensatz geschrieben und das File geschlossen. Nun ist die eingelegte Diskette nicht mehr geschützt

first is inserted, unprotected diskette, the file in the Append mode is again opened and the protected diskette is inserted. Now a data record is written and the file is closed. Now the inserted diskette is no longer protected.

Es gibt jedoch mögliche Fehlerquellen. Neben der Laufwerknummer ist auch die Sektornummer des FDS eine Konstante, der FDS wird also an die Stelle auf der zweiten Diskette geschrieben, an der er sich auf der ersten befand. Es ist aber nicht gesagt, daß dies in beiden Fällen der gleiche Sektor ist. In diesem Fall wäre ein File der zweiten Diskette zerstört.

There are however possible sources of error. Beside the drive number also the sector number of the FDS is a constant, which FDS thus written the place on the second diskette, at which it was on first. It is however not said that this is in both cases the same sector. In this case a file of the second diskette would be destroyed.

Hier hilft der umgekehrte Weg. Das Pseudo-File wird auf der geschützten Diskette eröffnet und auf der ungeschützten geschlossen. Dadurch stimmen die Sektornummern überein. Jetzt kann das o.ä. Verfahren ohne Gefahr angewandt werden.

Here the reverse way helps. Dummy file is opened on the protected diskette and closed on the unprotected. Thus the sector numbers correspond. Now can the or the like procedure without danger be applied.

Maschinenprogramme für Extended-Basic (DIS/FIX 80) können von geschützten Disketten mittels des Editors zum E/A-Modul kopiert werden, oder ganz einfach mit einem Basic-Programm. Das File wird mit dem Editor geladen und danach als DIS/FIX 80 File wieder abgespeichert. Da Maschinenprogramme für XB keine Symbole außerhalb des normalen ASCII-Zeichensatzes enthalten dürfen, werden diese Control-Characters auch nicht gelöscht. Ein Basic-Programm zum Kopieren eröffnet beide Dateien normal, liest einen Eintrag aus der geschützten Datei und schreibt ihn auf eine ungeschützte - ganz einfach!

Coded programs for Extended Basic (DIS/FIX 80) can be copied by protected diskettes by means of the editors of the E/A module, or completely simply with a Basic program. The file is loaded with the editor and stored to it as DIS/FIX 80 file again. Since coded programs for XB may not contain symbols outside of the normal ASCII character set, this control character are not also deleted. A Basic program for copying opens both files normally, reads an entry from the protected file and writes it on an unprotected — very simple!

9. Abriss der Funktionen eines Disketten-Managers

In diesem Kapitel soll weniger eine Anleitung zur Programmierung eines Disk-Managers gegeben werden, das würde den Rahmen in jedem Falle sprengen. Vielmehr sollen die grundlegenden Probleme und logischen Abläufe verdeutlicht werden, die ein solches Programm beinhaltet.

9.1. Initialisieren von Disketten

Bereits in den Einleitungskapiteln wurde eine Anleitung zu einem Programm gegeben, welches Disketten formatiert. Nach dem Abschluß dieser Routine sind aber die Sektoren 0 und 1 noch nicht definiert, da vorerst alle Sektoren mit dem Wert >E5 gefüllt sind. Nach diesem mehr physikalischen Vorgang des Formatierens muß die Diskette auch softwareseitig nutzbar gemacht, also initialisiert werden.

Nach dem Formatieren muß also geprüft werden, ob auch wirklich in allen Sektoren der Anfangswert >E5 steht. Ist dies bei einem Sektor nicht der Fall oder treten beim Lesen eines bestimmten Sektors Fehler anderer Art auf, dann muß dieser Sektor, um Datenverlust beim späteren Gebrauch zu vermeiden, in der Sektor-Bitmap als belegt gemeldet werden. Es muß also nach dem Formatieren ein 'Bad Block Scan' durchgeführt werden, während dem im RAM ein Bild des späteren Sektor 0 aufgebaut wird. Die Bits der Sektor-Bitmap, die bei der gewählten Formatierung nicht belegbar sind (die Bitmap ist ja immer etwas zu lang), müssen in jedem Fall als belegt gemeldet werden, sonst wird beim Schreiben versucht, über den Diskettenrand hinaus zu arbeiten. Natürlich müssen auch die Sektoren 0 und 1 belegt gemeldet werden.

Neben dem Isolieren defekter Sektoren (Sektor

Outline of the functions of a diskette manager

In this chapter is to be given fewer a guidance for the programming of a disk manager, which would blow up the framework in each case. Rather the fundamental problems and logical operational sequence are to be clarified, the one such program contained.

Initialization of diskettes

Already in the introduction chapters a guidance to a program one gave, which formats diskettes. After the termination of this routine however the sectors 0 and 1 are not yet defined, since all sectors are filled with the value >E5 for the time being. After this more physical process of formatting the diskette must be made usable also in terms of software therefore initialized.

After formatting it must be checked thus whether also really in all sectors the initial value is to >E5. If this is not with a sector the case or occur during the reading of a certain sector error of other type, then this sector must, in order to avoid overrun to the later use, in the sector bitmap than being occupied announced. Thus a "bad block scan" must be executed after formatting, while in the RAM a picture later sector 0 is structured. The bits the sector bit-map, which are not provable during selected formatting (the bitmap is always somewhat too long), must in each case as occupied are announced, otherwise during the writing one tries to operate beyond the diskette edge. Naturally also the sectors 0 and 1 must be announced occupied.

Apart from isolating defective sectors (sector 0

0 und 1 müssen o.K. sein!) müssen die Diskettenparameter geschrieben werden (Siehe Kapitel 7). Besonders wichtig ist hier die Textsequenz 'DSK '(Leerzeichen beachten!), die als Kennung einer formatierten Diskette benötigt wird. Ist der Sektor 0 so normiert, dann kann Sektor 1 mit Nullen gefüllt werden.

Erst jetzt wird die Diskette vom DOS akzeptiert.

Will man nun eine Diskette komplett löschen, dann war das bisher nur durch das erneute Initialisieren möglich. Es reicht aber auch, die Sektoren 0 und 1 auf ihren 'Urzustand' zurückzusetzen. Dies birgt jedoch die Gefahr, daß evtl. defekte Sektoren nun wieder zugänglich sind. Ein Sektor-Scan mit Lesen und Schreiben würde den Zeitvorteil zunichte machen.

Maßgeblich für die Ladezeit eines Files ist das sogenannte 'Sektor Interlace'. Je nachdem wie man es wählt, lassen sich die Ladezeiten bei allen Filearten, deren Sektoren nacheinander (am Stück) gelesen werden bis etwa um den Faktor 2 vergrößern oder verringern. Diese Wahl hat man auf normalem Wege nur bei dem Diskmanager zum CorComp 9900 Controller. Beim einfachen Formatieren über die Subroutine >11 wird ein festgelegtes Interlace verwendet. Schaut man sich nach dem Formatieren den Pufferbereich im VDP-RAM an, sieht man direkt, welche Daten die zuletzt geschriebene Spur enthält. Bei Kenntnis des Spuraufbaus kann daraus auf das Interlace geschlossen werden. Näheres findet sich in dem entsprechenden ROM-Listing.

Das Standard-Interlace ist im Allgemeinen ein recht guter Kompromiß für alle Filearten und Loader, kann aber im Einzelfall verbesserungsfähig sein. Es ist jedoch nur bei genauer Kenntnis der Diskcontroller-Hardware möglich, hier einzugreifen.

and 1 must be OK!) must the diskette parameters be written (see Chapter 7). Particularly importantly here the text sequence is "DSK" (blanks consider!), as identifier of a formatted diskette one needs. If the sector 0 is in such a way standardized, then sector 1 can be filled with zeros.

The diskette is accepted only now by the DOS.

If one wants to delete now a diskette completely, then was so far only by renewed initializing possible. In addition, it is enough to reset the sectors 0 and 1 to its "original state." This saves however the danger that perhaps defective sectors now are again accessible. A sector scan with reading and writing would make the time advantage destroyed.

Considerably for the loading time files is so-called "sector interlace." Ever after like one selects it, the loading times leave themselves with all types of file, whose sectors are read successively (at the piece) to approximately around the factor 2 to increase or reduce. This selection has one on normal way only with the disk manager to the CorComp 9900 controllers. When simply formatting over the subroutine >11 a determined Interlace is used. If one looks at oneself after formatting the buffer area in the VDP RAM, one sees direct, the track written last contains which data. With knowledge of the structure of track can be closed from it on the Interlace. Details are in the appropriate ROM listing.

The standard interlace is generally a quite good compromise for all types of file and Loader, can however be in individual cases capable of improvement. It is possible however only with exact knowledge of the disk controller hardware to intervene here.

9.2. File-Operationen

File-Operationen beziehen sich auf die logische Struktur 'File', wie bereits in den einführenden Kapiteln angedeutet. Aufgrund der komplexen Struktur, welche insbesondere lange und fragmentierte Files aufweisen können, wird die Behandlung sinnvollerweise nicht auf Sektorebene durchgeführt (obwohl dies natürlich genau das ist, was zum täglichen Brot einer DSR gehört).

9.2.1. Filenamen ändern

Zuerst wird der neue Filename in den entsprechenden File-Directory Sektor (FDS) geschrieben. Danach muß in Sektor 1 die Nummer des FDS so verschoben werden, daß alle Filenamen wieder in alphabetisch aufsteigender Form vorliegen. Beim Filenamen sind prinzipiell alle ASCII-Symbole (7 Bit) erlaubt, es muß aber Verträglichkeit mit der verwendeten Software (Basic o.ä.) bestehen. Weitere Änderungen sind nicht erforderlich.

Einfacher ist hier die Anwendung der Level-1-Routine >13. Die Zeiger auf den alten und den neuen Namen werden übergeben und die Routine aufgerufen. Die Umbenennung des FDS und die Sortierung von Sektor 1 werden von der Routine übernommen. Näheres enthält ein Beispielprogramm.

9.2.2. Files löschen

Ein File wird gelöscht, indem die Nummer seines FDS aus Sektor 1 entfernt wird und alle nachfolgenden Einträge um 2 Byte nach unten verschoben werden. Auch muß anhand der Cluster-Daten im jeweiligen FDS die Sektor-Bitmap in Sektor 0 von einigen gesetzten Bits befreit werden. Unbedingt notwendig ist, daß das zu löschende File vorher geschlossen wurde, da sonst die Cluster-Daten nicht unbedingt stimmen. Das ist alles, was beim Löschen eines

File operations

File operations refer to the logical structure "file," as already suggested in the introductory chapters. Due to the complex structure, which is enough in particular and fragmented files indicate can, the handling is not executed for good reason on sector level (although this is naturally exactly that, which belongs to daily bread of a DSR).

File names modify

First the new file name into the appropriate file directory sector (FDS) are written. Afterwards the number of the FDS must be shifted in such a way in sector 1 that all file names are present again in alphabetically ascending form With the file name always all ASCII symbols (7 bits) are permitted, it must however compatibility with the used software (Basic perhaps. Further modifications are not necessary

Simple is here the application of the Level-1 Routine >13. The pointers on the old and the new names are transferred and the routine is called. Renaming the FDS and the assortment of sector 1 are taken over by the routine. Details contain a sample program.

Files delete

A file are deleted, as the number of its FDS is removed from sector 1 and all following entries 2 byte is downward shifted. Also the sector bit-map in sector 0 must be released from some set bits on the basis the cluster data in the respective FDS. It is absolutely necessary that the file which can be deleted was beforehand closed, since otherwise the cluster data are not correct necessarily. That is everything that occurs with the deletion files. All data sectors

Files geschieht. Alle Datensektoren und der FDS existieren immer noch unverändert. Dadurch ist es möglich, ein gelöschtes File komplett wieder herzustellen, wenn nur sein Name bekannt ist und inzwischen nicht auf die Diskette geschrieben wurde!

and the FDS exist still unchanged. Thus it is to be repaired possible, a deleted file completely, if only its name admits is and on the diskette were in the meantime not written!

Zum Löschen eines Files steht die Level-2-Option 'DELETE' zur Verfügung.

For the deletion files the Level-2 Option "DELETE" is for order.

9.2.3. Files kopieren

Files copy

Hier gibt es prinzipiell zwei mögliche Verfahren. Das einfachste (programmiertechnisch gesehen) ist die Eröffnung des Files im INPUT- Modus, die Eröffnung eines Files im OUTPUT-Modus auf der Zieldiskette und der satzweise Transfer der Daten. Gegen dieses Verfahren ist bis auf die niedrige Geschwindigkeit nichts einzuwenden. Es scheitert aber spätestens bei überlangen PROGRAM-Files. Diese können nur sektororientiert kopiert werden. Wenn das aber bei PROGRAM-Files nötig wird, dann kann auch der ganze Rest der Kopiererei sektororientiert geschehen, da dann kein Unterschied mehr zwischen den Files existiert. Zuerst wird der FDS kopiert, und zwar in einen Puffer im RAM. Nun werden die zum File gehörigen Sektoren anhand der Cluster-Tafel bis zum Ende (des Files oder des Datenpuffers) gelesen und auf der Zieldiskette geschrieben (davor einen Sektor für den FDS reservieren!). Während des Transfers wird ein geänderter FDS erzeugt, der eine eventuell notwendige Aufspaltung des Files auf der Zieldiskette protokolliert. Nach dem Transfer wird dieser FDS geschrieben und die alphabetisch korrekte Eintragung in Sektor 1 vorgenommen. Die Sektor-Bitmap wird aktualisiert und der Kopiervorgang ist beendet.

Here give it two always to possible procedures. The simplest (seen program technical) is record by record the initialization files in the input of the mode, the initialization files in the output mode on the target diskette and that transfer of the data. To this procedure nothing is to be objected up to the low rate. It fails however at the latest with excessive PROGRAM files. These can be copied only sector-oriented. If becomes however necessary with PROGRAM files, then also the whole remainder of the Kopiererei can occur sector-oriented, since then no more difference exists between the files. First the FDS is copied, into a buffer in the RAM. Now the sectors belonging to to the file are read on the basis the cluster board up to the end (files or the scratchpad memory) and written on the target diskette (before it a sector for the FDS to reserve!). During the transfer a modified FDS is produced, which logs a possibly necessary fragmentation files on the target diskette. After the transfer this FDS is written and the alphabetically correct entry in sector 1 is made. The sector bitmap is updated and the copying process is terminated.

Auch hier hilft eine Level-1-Routine, das komplette Erstellen eines solchen Programms zu umgehen. Eigentlich sind es zwei Routinen, die vefügbar sind. Die erste liest entweder Teile des

A Level-1-Routine helps also here to go around complete creating of such a program. Actually there is two routines, which are vefuegbar. First reads either sections of the FDS or a certain

FDS oder eine bestimmte Anzahl von Datensektoren. Die zweite schreibt entweder den FDS und reserviert gleichzeitig die notwendigen Sektoren, oder schreibt eine bestimmte Anzahl von Datensektoren in das Copy-File. Die Namen der Routinen sind >14 und >15. Auch hierzu sind Beispielprogramme vorhanden.

9.2.4. Gelöschte Files wiederherstellen

Zu einem ordnungsgemäß wiederhergestellten File gehören: richtiger Eintrag der FDS-Nummer in Sektor 1 und angepasste Sektor-Bitmap mit gesetzten Bits für jeden von dem File nun wieder okkupierten Sektor (FDS nicht vergessen!). Zuerst muß der FDS gesucht werden, wozu der ehemalige Filename benötigt wird. Dieser wird auf der Diskette gesucht und es wird geprüft, ob es sich bei dem evtl. gefundenen Filenamen auch um einen FDS handelt (am Besten, Null-Bytes prüfen — Murphy ist überall!!). Beim anschließenden 'Reparieren' der Sektor-Bitmap muß geprüft werden, ob als belegt zu meldende Sektoren noch frei sind, andernfalls wurde das File bereits überschrieben. Wurde der FDS überschrieben, so ist nichts mehr zu retten (im Allgemeinen!), fehlende Datensektoren sind manchmal zu verkraften.

9.2.5. Fileschutz ändern

Hier steht wiederum eine Level-1-Routine zur Verfügung, näheres im Kapitel mit den Beispielprogrammen. Es gibt aber auch den 'direkten' Weg. Zuerst den FDS suchen, dann in den Puffer holen und das File-Status Byte nach Bedarf beeinflussen. Anschließend wird der FDS an seine alte Stelle zurückgeschrieben.

9.2.6. Diskettenkatalog

Der Aufbau eines Diskettenkataloges ist auf 2 Arten möglich. Zum einen kann die Standard Directory Datei 'DSKX.' im INPUT-Mode

number of data sectors. Second writes either the FDS and reserves at the same time the necessary sectors, or writes a certain number of data sectors into the copy file. The names of the routines are >14 and >15. Also for this sample programs are available.

Deleted files re-create

To correctly re-created a file belong: correct entry of the FDS number in sector 1 and adapted sector bit-map with set bits for everyone of the file now again okkupierten sector (FDS do not forget!) First the FDS must be looked up, to which the former file name is needed. This is looked up on the diskette and it is checked whether it itself with evtl. found file name also around a FDS concerns (at the best one, zero bytes check — Murphy is everywhere!!). During following "repairing" the sector bitmap must be checked, whether as sectors occupied which can be announced are still free, otherwise one the file was already overwritten. If the FDS was overwritten, then is nothing more to save (generally!), missing data sectors are to be sometimes borne.

File protection modify

Here is again available a Level-1 routine, details in the chapter with the sample programs. In addition, there is the "direct" way. First look up, then into the buffer get the FDS and influence file status the byte as required. Subsequently, the FDS is written back to its old place.

Diskette catalog

The structure of a diskette catalog is possible in 2 types. On the one hand the standard directory file "DSKX" can. In the input mode to be opened

eröffnet werden (das X ist selbstverständlich durch die entsprechende Laufwerkskennung zu ersetzen), deren Einträge satzweise gelesen und angezeigt werden. Diese Methode erinnert aber stark an Basic, und ist daher wohl dem versierten Assembler-Programmierer suspekt. Eine größere Einschränkung bedeutet aber wohl die eigentlich vollkommen unverständliche Eigenart dieser Routine des DOS, daß bei jedem Eintrag, der gelesen werden soll, vor dem eigentlichen FDS immer erst Sektor 1 gelesen wird. Das macht diese Art des Inhaltsverzeichnisses sehr langsam. Aber es geht selbstverständlich auch anders.

(the X is naturally by the appropriate drive identifier to replace), their entries to be read record by record and displayed. This method reminded however strongly of basic, and is suspect therefore probably the experienced assembler programmer. A larger restriction however probably means the actually perfectly incomprehensible characteristic of this routine of the DOS that with each entry, which is to be read before the actual FDS always only sector 1 reading becomes. That makes this type of the table of contents very slow. But it goes naturally also differently.

9.2.6.1. Diskettendaten aus Sektor 0

Sektor 0 wird gelesen, der Name angezeigt. Sodann wird die Gesamtzahl der Sektoren ermittelt und anhand der Sektor-Bitmap so lange dekrementiert (jedes nicht gesetzte Bit reduziert die Anzahl der belegten Sektoren) bis das Ende des Sektor 0 erreicht ist. Nun existieren zwei Zähler, einer mit der Anzahl der belegten Sektoren und derjenige mit der Gesamtzahl aller Sektoren. Diese werden in Dezimalzahlen umgewandelt und angezeigt.

Der Vorteil dieses Verfahrens liegt darin, daß das Ende der Bitmap nicht bekannt sein muß, da nur nicht belegte Sektoren zählen, das Bitmap-Ende aber entsprechend der Vorschrift mit >FF gefüllt ist.

Diskette data from sector 0

Sector 0 is read, the name is displayed. Then the total number of the sectors and on the basis the sector bitmap is determined so for a long time decremented (each not set bit reduces the number of occupied sectors) to the end sector 0 achieved is. Now two counters, of one exist with the number of the occupied sectors and that with the total number of all sectors. These are converted and displayed into decimal numbers.

The advantage of this procedure is in the fact that the end does not admit to the bit-map be must, since not occupied sectors count the bit-map end however according to the regulation with >FF is filled.

9.2.7. Filedaten anzeigen

File data display

Hier wird eine Kopie von Sektor 1 im RAM angelegt, mit dessen Hilfe nacheinander alle eingetragenen Sektoren (FDS) gelesen werden. Deren Filenamen werden angezeigt, die Anzahl der belegten Sektoren um eins inkrementiert und angezeigt und der Filestatus umgewandelt und ebenfalls angezeigt. Bei einer FDS-Nummer von Null endet die Ausgabe.

Here a copy by sector to 1 in the RAM are created, with whose assistance successively all entered sectors (FDS) are read. Their file name is displayed, the number of occupied sectors around one is incremented and displayed and converted the file status and likewise displayed. With a FDS number of zero the output ends.

9.3. Disketten kopieren

Für das Kopieren von Disketten bieten sich 3 Möglichkeiten an:

1. Das komplette Kopieren aller Files,
2. Das Kopieren aller belegten Sektoren,
3. Das Kopieren aller Sektoren der Diskette.

Das erste Verfahren ist dann sinnvoll, wenn alle Files auf eine neue Diskette gebracht werden sollen, wo dann jedes File aus nur einem zusammenhängenden Block besteht. Leider besticht dieses Verfahren nicht durch eine hohe Geschwindigkeit, es werden aber keine bereits belegten Sektoren überschrieben, wie dies bei den beiden anderen Verfahren der Fall sein kann.

Das zweite Verfahren kann bei nur teilweise belegten Disketten enorm zeitsparend sein, vorausgesetzt die Sektor-Bitmap wurde nicht manipuliert. Bei diesem Verfahren wird nämlich die Sektor-Bitmap in einen Puffer im RAM verlegt und anhand der Einträge geprüft, welche Sektoren kopiert werden müssen.

Es ist übrigens nicht, wie einige nun vielleicht denken werden, notwendig, die Nummern der gelesenen Sektoren zu protokollieren um diese später wieder an den richtigen Platz zu bringen. Diese Information steckt ja in der Sektor-Bitmap. Es reicht also, die Sektoren anhand der Bitmap zu lesen, bis der Puffer voll oder die Bitmap zu Ende ist. Wird beim Schreiben gleichermaßen verfahren, dann ist die richtige Reihenfolge automatisch sichergestellt. Da aber, wie bereits angesprochen, der Zustand der Sektor-Bitmap beim Lesen von Files irrelevant ist, läßt sich ein solches, die Bitmap benutzendes Programm aber risikolos austricksen.

Dieses Austricksen ist bei einem Kopierprogramm des dritten Typs nicht möglich. Hier

Diskettes copy

For copying diskettes offer themselves 3 possibilities:

1. Complete copying of all files,
2. Copying all occupied sectors,
3. Copying all sectors of the diskette.

The first procedure is meaningful if all files are to be brought on a new diskette, where then each file consists of only a connected block. Unfortunately this procedure does not captivate it by a high rate, however no sectors already occupied is overwritten, as this can be with the two other procedures the case.

The second procedure can be enormously time-saving with only partly occupied diskettes, presupposed the sector bit-map was not manipulated. With this procedure the sector bit-map is shifted into a buffer in the RAM and checked on the basis the entries, which sectors to be copied to have.

It is not by the way, as some will now perhaps think, necessarily, to log the numbers of the read sectors around this back to the correct place to later bring. This information is in the sector bitmap. It is enough thus to read the sectors on the basis the bitmap until the buffer is fully or the bitmap to end. During the writing will equally proceed, then the correct sequence is automatically guaranteed. Since however, as already addressed, the status is irrelevant the sector bitmap when reading files, such, which out-cheat bitmap using program however without risk, leaves itself.

This Austricksen is not possible with a copying program of the third type. Here the diskette

wird die Diskette Sektor für Sektor kopiert, ohne auf deren Belegung zu achten. Dazu muß aber die Anzahl der Sektoren auf der Diskette bekannt sein. Diese läßt sich aus Sektor 0 lesen oder aber vom Bediener abfragen, was sicherer ist. Gültige Angaben sind hier:

- 360 single sided, single density
- 720 double sided, single density
- 640 single sided, double density (Myarc)
- 720 single sided, double density
- 1280 double sided, double density (Myarc)
- 1440 double sided, double density

Alle Zahlen dezimal!

9.4. Diskettentests

Diskettentests dienen im Allgemeinen dazu, defekte Sektoren aufzuspüren, sei es beim erstmaligen Einsatz der Diskette, also nach dem Formatieren oder beim Auftreten von Lesefehlern im Betrieb. Bei Fehlern die während des Einsatzes beschriebener Disketten auftreten, kommen nur Tests in Frage, welche die Daten unverändert lassen. Zum eingehenden Prüfen neuer Disketten können Schreib/Lesetests angewandt werden.

9.4.1. Logische Tests

Hier wird fileorientiert vorgegangen. Der FDS eines Files wird gelesen, anschließend wird anhand dessen Daten geprüft, ob sich jeder vom File belegte Sektor lesen lässt. Hierbei ist zu beachten, daß ein Lesefehler bei einem Sektorzugriff nicht zwangsläufig bedeutet, daß die Daten verloren sind. Eine Fehlermeldung wird nämlich auch dann gegeben, wenn das Adressfeld in Ordnung ist und lediglich die CRC-Prüfung der Daten negativ war. Die Fehlercodes >22, >23 bedeuten, daß Daten vom Controller

sector for sector is copied, without paying attention to their allocation. In addition however the number must of the sectors on the diskette admits to be. This can be read from sector 0 or tested however by the user, which is safer. Valid specification is here:

- 360 single sided, single density
- 720 double sided, single density
- 640 single sided, double density (Myarc)
- 720 single sided, double density
- 1280 double sided, double density (Myarc)
- 1440 double sided, double density

All numbers decimal!

Diskette tests

Diskette tests serve generally to seek out defective sectors is it with the first application of the diskette, thus after formatting or when occurring read errors in the operation. With errors during the application of described diskettes, are applicable only tests occur, which unchanged the data leave. For detailed checking of new diskettes write/reading tests can be applied.

Logical tests

Fle-oriented one proceeds here. The FDS files is read, checked afterwards on the basis its data whether everyone occupied itself by the file sector read lets. Here it is to be noted that a read error does not mean with a sector access inevitably that the data are lost. An error message is given also if the address field is correct and only the CRC check of the data were negative. The error codes >22, >23 mean that data were transferred by the controller, which can be written back the address field thus correct

übergeben wurden, die zurückgeschrieben werden können, das Adressfeld also in Ordnung ist. Ein Fehlercode >11 deutet auf ein defektes Adressfeld hin, weshalb der Sektor nicht gefunden wurde und auch nicht zurückgeschrieben werden kann. Tritt dies bei einem FDS auf, dann ist das File in den meisten Fällen verloren. Ist irgendein Datensektor defekt, der nicht zurückgeschrieben werden kann, dann kann dieser im FDS 'ausgeclustert' werden, wodurch er keine Lesefehler mehr produziert. Ein fehlender Datensektor ist leichter zu verkraften als ein total unbrauchbares File.

are. An error code >11 points on a defective address field, why the sector was not found and also cannot be written back. If this occurs with a FDS, then the file is in most cases lost. If any data sector is defective, which cannot be written back, then this can become "out of cluster" in the FDS, whereby he produces no more read errors. A missing data sector is easier to bear than a totally useless file.

9.4.2. Tests ohne Datenverlust

Tests without overrun

Tests dieser Art sind immer dann angebracht, wenn der fehlerhafte Datenträger bereits wichtige Daten enthält, die nicht gelöscht werden können. Dabei kann u.a. eine ggf. vorhandene Laufwerksabhängigkeit eines Fehlers erkannt werden, was die Wiederherstellung der Daten erleichtern kann.

Tests of this type are appropriate whenever the incorrect data carrier contains already important data, which cannot be deleted. Among other things one can if necessary available drive dependency of an error to be detected, what can facilitate the re-creation of the data.

Oftmals ist es so, daß ein Fehler aufgrund einer mangelhaften Zentrierung der Diskette im beschreibenden oder lesenden Laufwerk auftritt. Dann sinkt der Lesepegel zu stark ab, was einen Datenverlust bewirkt. Ein erneutes Einlegen ins gleiche oder ein anderes Laufwerk kann hier schon hilfreich sein.

Often it can be that an error occurs due to an unsatisfactory centering of the diskette in the while writing or reading the drive. Then the reading level drops too strongly, which causes an overrun. Renewed inserting into same or another drive can be here quite helpful.

9.4.2.1. Physikalische Nur-Lese-Tests

Physical only reading tests

Hierbei werden alle Sektoren der Diskette gelesen. Tritt ein Lesefehler auf, dann sollte versucht werden, den Sektorinhalt, auch wenn er falsch ist, wieder zurückzuschreiben. Bei einem defekten Adressfeld ist das nicht mehr möglich. Ist das Adressfeld aber in Ordnung, dann werden durch das Rückschreiben die Prüfsummenbytes wieder aktualisiert, so daß der nächste Zugriff auf diesen Sektor keinen Lesefehler mehr produziert. Soll überhaupt nicht geschrieben werden, so kann dieser Test

Here are read all sectors of the diskette. If a read error occurs, then should be tried to write back sector contents, even if it is false, again. With a defective address field that is no longer possible. If the address field is however correct, then by rewriting the check total bytes are again updated, so that the next access to this sector produces no more read error. Is not to be written at all, then this test can only display, which was not in order.

nur anzeigen, wo was nicht i.O. ist.

9.4.2.2. Schreib-Lese-Tests

Jeder Sektor wird gelesen und wieder an die selbe Stelle geschrieben. Tritt beim Lesen oder Schreiben ein Fehler auf, dann wird der Vorgang für diesen Sektor wiederholt bzw. nach n Versuchen ein Fehler gemeldet.

9.4.3. Physikalische Mediumsprüfung

Bei dieser Prüfung können Disketten auf Bits und Bytes geprüft werden. Zuerst wird formatiert, dann wird jeder Sektor auf >E5 geprüft. Anschließend wird jeder Sektor gezielt geschrieben und gelesen. Besonders interessant ist eine Prüfung auf Seitenübersprechen. Hier wird bei doppelseitigen Disketten auf einen bestimmten Sektor einer Seite ein Muster aus lauter >00 geschrieben; auf den direkt gegenüberliegenden werden >A5 geschrieben (mehrfach schreiben!). Nun wird mehrere Male der Sektor mit dem Nullmuster gelesen, ob kein Bit 'umgefallen' ist. Wenn doch, dann kopieren Daten auf die andere Seite über und die Diskette sollte nur einseitig benutzt werden.

Welche Sektornummer der gegenüberliegende Sektor hat, kann einfach aus der Formatierungsart bei Kenntnis des Sektor-Interlace bestimmt werden. Im Zweifelsfall werden alle Sektoren einer Spur mit dem gleichen Muster beschrieben und alle geprüft.

Auf die gleiche Weise kann ein Spur-übersprechen erfasst werden. Hier werden die Nullen auf den Sektor X geschrieben, die >A5 auf den Sektor X plus Y, wobei Y die Anzahl der Sektoren pro Spur ist. Dieser Test ist besonders bei 80 Spur Laufwerken interessant.

Write reading test

Each sector is read and written again the same place. If an error occurs during reading or writing, then the process for this sector is repeated or. after n attempts an error announced.

Physical medium check

With this check diskettes can be checked for bits and bytes. First one formats, then each sector is checked for >E5. Subsequently, each sector is directly written and read. Particularly a check on side cross modulation is interesting. Here with double-sided diskettes on a certain sector of a page a sample is written out louder >00; on the directly facing >A5 are written (to write several times!). Now several marks the sector with the zero-sample is read whether no bit is "fallen down." If nevertheless, then data copy on the other page over and the diskette should is only on one side used

Which sector number the opposite sector has, can be determined simply from the type of formatting with knowledge of the sector Interlace. In the case of doubt all sectors of a track with the same sample are described and checked all.

In the same way a track cross-talk can be entered. Here the zeros are written on the sector X, the >A5 on the sector X plus Y, whereby Y is the number of sectors per track. This test is particularly interesting with 80-track drives.

9.5. Wiederherstellung defekter Adreßfelder

Die Adreßfelder der Sektoren einer Spur werden beim Formatieren geschrieben und dann immer nur gelesen. Ist also so ein Adressfeld defekt, kann es nur durch formatieren der betreffenden Spur wiederhergestellt werden. Zuerst werden alle lesbaren Sektoren im RAM zwischengespeichert. Dann wird die Spur neu formatiert und die Sektoren werden, bis auf die ehemals defekten, wieder geschrieben. Soweit die 'brutale' Methode.

Besser ist es, die gesamte Spur zu lesen, die Sektordaten zwischenzuspeichern, die Spur in ein formatierungsfähiges Format zu bringen (Datenfeld mit >E5 füllen, CRC-Start/Stop-Bytes setzen etc.) das betreffende Adressfeld zu reparieren und die Spur nach dem Rückschreiben wieder mit den Sektordaten zu füllen.

Wie zu sehen ist, ist das nur für den weit fortgeschrittenen Anwender möglich, der die Hardware genau kennt. Dann ist aber schon einiges nötig, um irreparable Disketten zu produzieren (Magnetanschläge o.ä.).

9.6. Erzeugen defekter Sektoren

Dies ist eine relativ einfache Methode, eine Diskette mit einem 'Fingerabdruck' zu versehen. Man erzeugt beim Formatieren (natürlich mittels einer eigenen Formatierungssoftware) defekte Sektoren, macht im Programm Sektorzugriffe auf diese Sektoren (mit den Standard Level-1-Routinen) und prüft das Fehlerbyte. Danach kann eine etwaige Raubkopie erkannt werden.

Wie kann man das nun anstellen?

Re-establishment of defective address fields

The address fields of the sectors of a track when formatting and read then in each case. Thus such if an address field is defective, it can format only through the track concerned to be re-created. First all readable sectors in the RAM become buffered. Then the track will become again formatted and the sectors, up to the formerly defective, again written. So far the "brutal" method.

It is better to read the entire track to buffer the sector data the track into a formatable format to bring (fill data field with >E5, set CRC Start/Stop bytes etc.) to repair the address field concerned and to fill the track after rewriting again with the sector data.

As is to be seen, is possible only for the user progressed far, who knows the hardware exactly. Then however already some is necessary, in order to produce irreparable diskettes (magnet impacts or the like).

Producing defective sectors

This is a relatively simple method to provide a diskette with a "finger mark." One produced when formatting (natural by means of an own formatting software) defective sectors, makes sector accesses to these sectors (with the standard Level-1 routine) in the program and checks the error byte. Afterwards a any pirate copy can be detected.

How can one employ now?

9.6.1. Erzeugen eines LOST-DATA Zustandes

Hier wird ein Datenfeld mit der falschen Längenkennung erzeugt. Das kann z.B. ein mit 512 Byte angegebenes Feld sein, in das aber nur 256 Byte geschrieben werden. Die Daten werden richtig geschrieben und gelesen, es wird aber immer ein Fehler gemeldet.

9.6.2. Erzeugen nicht normierter Spuren

Das ist recht einfach. Es werden die in den Datenblättern zum Controller-IC angegebenen Mindestparameter unterschritten, so beispielsweise der ID-GAP oder es werden alle Adreßfelder mit einer falschen Spurnummer versehen — die Möglichkeiten sind hier so vielfältig, daß der Platz nicht reicht. Das Ergebnis ist aber immer, daß die ganze Spur unlesbar wird.

Dies sind nur Beispiele, dem versierten Programmierer werden noch mehr bösartige Ideen kommen, einem Kopieren vorzubeugen. Dieses Kapitel befindet sich daher auch nur der Vollständigkeit wegen unter dem Punkt 'Diskmanager-Funktionen'.

Producing LOST DATA of a status.

Here a data field with the false length identifier is produced. That can be e.g. a field indicated as 512 byte, into which however only 256 byte is written. The data are written and however always announced correctly read, it an error.

Producing non-standard tracks

That is quite simple. The minimum parameters indicated in the data sheets for the controller IC will become fallen below, so for example ID-gap or it all address fields with a false track number to provide — the possibilities are here so various that the place is not enough. The result however always is that the whole track becomes illegible.

These are only examples, which become experienced programmer still more malicious ideas to come to prevent copying. This chapter is therefore also only the completeness because of under the point "disk manager functions."

10. Nutzung der Dienstprogramme in der Controller-DSR

Für den 'normalen' Betrieb benötigt die Disk-Controller-DSR diverse Routinen zur Hardware-Ansteuerung, Sektormanipulation und Bearbeitung von Daten auf File-Ebene. Daraus ergibt sich eine funktionelle Hierarchie, welche innerhalb von *Disk-Info* mit folgenden Bezeichnungen belegt wird:

- Level-0-Routinen — DSR-interne Ansteuerung des Floppy-Disk-Controller-Formatter-Chips (FDC)
- Level-1-Routinen — Extern verfügbare, jedoch auch intern genutzte Routinen zur Bearbeitung einzelner Sektoren
- Level-2-Routinen — Externe Routinen zur Behandlung kompletter Files

Die Level-0-Routinen sind extern nicht verfügbar, zumindest nicht über normierte Funktionsaufrufe wie DSR- oder SBRLNK. Diese Routinen arbeiten 'hart an der Hardware' und erfordern entsprechende Sachkenntnis bei der Anwendung. Da sie innerhalb der DSR als Teile der Level-1-Routinen genutzt werden, sind sie i.d.R. nur durch Nachbildung innerhalb eigener Programme nutzbar. Beispiele zur Programmierung finden sich in den Kapiteln 16 und 17 sowie in den ROM-Listings.

Die Level-1-Routinen werden über den standardisierten Aufruf DSRLNK mittels des Offset >A aktiviert und bearbeiten hauptsächlich Daten auf Sektor-Ebene (Ausnahme CALL FILES, >16). Selbst die Routinen >12 und >13, die auf den ersten Blick wie File-Operationen aussehen mögen, bearbeiten nur bestimmte Bytes des jeweiligen FDS (>13 noch dazu Sektor 1). Das

Use of the utility programs in the controller DSR

For the "normal" operation needs the disk controller DSR various routines for the hardware control, sector manipulation and handling of data on file level. From it results a functional hierarchy, which is occupied within *Disk Info* with the following designations:

- Level-0 routine — DSR internal control of the Floppy Disk Controller Formatter chip (FDC)
- Level-1 routine — external available, however also internally used routines for the handling of individual sectors
- Level-2 routine — external routines for the handling of complete files

Level-0-Routine are externally not available, at least not over standardized function calls such as DSR or SBRLNK. These routines operate "hard on the hardware" and require appropriate expertise with application. Since they are used within the DSR as sections of the Level-1 Routine, they are usable usually only by reproduction within own programs. Examples for programming are in chapters 16 and 17 as well as in the ROM listings.

The Level-1-Routine is activated over the standardized call DSRLNK by means of the offset >A and to process mainly data on sector level (exception CALL FILES, >16). Even the routines >12 and >13, which may look at first sight such as file operations, process only certain bytes of the respective FDS (>13 still in addition sector 1). The programming INTERFACE is limited to a rudimentary PAB (2 bytes) as well as

Programmierinterface beschränkt sich auf einen rudimentären PAB (2 Bytes) sowie einige reservierte Adressen im PAD- und VDP-RAM.

Die Level-2-Routinen schließlich stellen die höchste Ebene der Funktionalität aller von der Disk-DSR zur Verfügung gestellter Programme dar. Entsprechend werden diese Routinen von der DSR selbst nicht genutzt, können also nur aus einer Anwendersoftware heraus mittels eines recht komfortablen Programmierinterface' genutzt werden. Die entsprechenden Beispielprogramme des Editor/Assembler-Handbuches sollen hier nicht wiederholt werden - vielmehr werden die Funktionen beschrieben, welche das E/A-Handbuch verschweigt.

Anhand des oben Gesagten kann vereinfachend festgehalten werden:

Die Level-0-Routinen dienen der Hardware-steuerung, die Level-1-Routinen arbeiten sektororientiert und die Level-2-Routinen schließlich bearbeiten ganze Files bzw. Ausschnitte daraus.

10.1. Beispielprogramme zu den Level-1 Routinen

10.1.1. Vorbemerkungen

Obwohl die verwendeten Parameter für alle Diskcontroller identisch sind, so existieren doch einige geringfügige Unterschiede zwischen den Controllern verschiedener Hersteller. Sollte also ein Beispielprogramm Ärger machen, dann sollte zuerst geprüft werden, ob die hier gemachten Voraussetzungen beim verwendeten Controller zutreffen.

Der Pointer an >8356, der sog. Namelength-Pointer zum DSRLNK, wird bei den Level-1-Routinen genauso verwendet wie beim normalen DSRLNK - er zeigt immer auf die Länge des Routinennamens.

some reserved addresses in the PAD and VDP RAM.

The Level-2 routine finally represents the highest level of functionality all of programs provided of the disk DSR. Accordingly these routines are not used of the DSR themselves, can be used thus only from a user software by means of a quite comfortable programming interface. The appropriate sample programs of the Editor/Assembler system reference manual are not to be repeated here — rather the functions are described, which conceals the E/A manual.

On the basis above the saying can be held simplifying:

The Level-0 Routine serves the hardware control, the Level-1-Routine operates sector-oriented and the Level-2-Routine finally processes whole files or sections from it.

Sample programs to the Level-1 routines

Preface

Although the used parameters for all disk controllers are identical, then exist nevertheless some slight differences between controllers different manufacturers. Thus if a sample program should make annoyance, then it should be first checked whether the prerequisites made here apply with the used controller.

The pointer on >8356, which so-called Name length pointer to the DSRLNK, is exactly the same used with the Level-1 Routine as with the normal DSRLNK — it always points to the length of the routine name.

Die vorgestellten Programmbeispiele sind aber, wie der Name sagt, nur Beispiele und keine fertigen, jedem Anwendungsfall gewachsenen Universalroutinen. Sie sollen zeigen, wie man mit den DOS-Routinen umzugehen hat und die Grundlage weiterer, eigener Experimente sein. Um hier Anfangsfrustrationen vorzubeugen: alle Beispielprogramme sind geprüft und bei korrektem Abtippen lauffähig!

10.1.2. Subroutine >10, Sektor lesen/schreiben

Diese Routine wurde zwar bereits weiter oben angesprochen, soll aber zusammenfassend hier nochmals erwähnt werden. Die Routine benötigt einen exakt 256 Bytes großen Puffer irgendwo im VDP und verlangt folgende Pointer:

>834C (B) Laufwerknummer, >01 bis >04

>834D (B) I/O-Code, >00 — schreiben, sonst — lesen

>834E (W) Adresse das VDP-Datenpuffers, 256 Bytes

>8350 (W) Sektornummer

In >8350 wird das Fehlerbyte übergeben, siehe die entsprechende Liste in Kapitel 11.

Hier nun das Programm, das mit dem Start über READ den Sektor 150 in den Puffer im VDP-RAM ab >1100 liest und über WRITE die Daten des Puffers in den Sektor 150 schreibt:

10.1.2.1. Beispielprogramm SBR >10

The presented program examples are however, as the name says, only examples and no finished, each application grown universal routines. They are to show, like one with the DOS routines going around have and the basis of further, own experiments be. In order to prevent starting frustrations here: all sample programs are checked and when correct typing executable!

Subroutine >10, sector read/write

This routine one addressed already further above, is however to be in summary here again mentioned. The routine needs one accurately 256 bytes large buffer somewhere in VDP and requires the following pointers:

>834C (B) drive number, >01 to >04

>834D (B) I/O-Code, >00 — write, otherwise — read

>834E (W) address VDP scratchpad memory, 256 bytes

>8350 (W) sector number

In >8350 the error byte are transferred, see the appropriate list in Chapter 11.

Here now the program, which reads the sector 150 off with the start over READ into the buffer in the VDP RAM >1100 and writes over WRITE the data of the buffer into the sector 150:

Sample program SBR >10

```
* (c) 1986 Ch. Winter
*
       DEF   READ,WRITE
       REF   DSRLNK,VMBW
*
PABAD  EQU   >1000     Adresse des PABs im VDP-RAM
BUFFER EQU   >1100     Adresse des Datenpuffers
*
LW     BYTE  >01       Laufwerk 1
```

```
       EVEN
*
NAME   DATA  >0110     Subroutinenname
SEKTOR DATA  150       Sektornummer, beliebig
MYWS   BSS   32        Eigener Workspace
*
READ   LWPI  MYWS      Eigene Register laden
       SETO  R0        R0 = >FFFF, lesen indizieren
RW     MOVB  R0,@>834D           I/O-Code
       MOVB  @LW,@>834C          Laufwerknummer
       MOV   @SEKTOR,@>8350      Sektornummer
       LI    R0,BUFFER
       MOV   R0,@>834E      Adresse des Puffers angeben
       LI    R0,PABAD
       MOV   R0,@>8356      Zeiger auf das Namelength-Byte
       LI    R1,NAME        Position des SBR-Namens
       LI    R2,2           2 Byte verschieben
       BLWP  @VMBW          PAB ins VDP-RAM schaffen
       BLWP  @DSRLNK        Routine aufrufen
       DATA  >A             Routinenindex
       MOVB  @>8350,@>8350  Fehlerbyte prüfen
       JEQ   OK             kein Fehler
ERROR  ****  ****          hier Fehlerbehandlung einbauen
OK     ****  ****          hier Programm weiterführen
*
WRITE  LWPI  MYWS
       CLR   R0       Schreiben indizieren
       JMP   RW       alle weiteren Aktionen sind gleich!
*
       END            Ende des Beispielprogramms*
```

Die Sektor-I/O-Routinen von CorComp und Myarc weisen Besonderheiten auf: CorComp erlaubt es, den Datenpuffer nicht im VDP- sondern alternativ im CPU-RAM zu halten, was in einigen Fällen das Speicher-Herumgeschiebe von und nach VDP-RAM entbehrlich macht. Aktiviert wird diese Funktion durch Setzen des Bit 5 im System-RAM an Adresse >400B (mit >20 verODERn) sowohl für Lese- wie auch Schreib-Operationen. Zusätzlich kann die Verify-Funktion beim Schreiben von Sektoren deaktiviert werden, was jedoch außer der Gefahr des Daten-Harakiri keinen weiteren Sinn ergibt — Tempo gegen Datensicherheit aufzuwiegen (0,2 Sekunden Gewinn pro Sektor) ist generell nicht akzeptabel.

The sector I/O routines of CorComp and Myarc wise special features up: CorComp permits it to keep the scratchpad memory not in the VDP but alternative in the CPU RAM which makes the memory Herumgeschiebe of and after VDP RAM dispensable in some cases. This function is activated by setting the bit 5 in the system RAM at address >400B (with >20 verODERn) both for reading as well as write operations. Additionally the verify function can be deactivated when writing sectors, what does not result in a broader sense however except the danger of the data harakiri — speed against data security to counterbalance (0.2 seconds gain per sector) is generally not acceptable.

Auch hier erlaubt sich Myarc wieder eine Extrawurst, die sich keiner so richtig erklären kann: Beim 'Proofreading', dem Lesen des Sektors unmittelbar nachdem er geschrieben wurde, wird nicht etwa Byte für Byte mit dem Pufferinhalt verglichen, sondern der Sektorinhalt ins ROM-0 befördert. Wieso man diesen destruktivsten (und dümmsten) aller Wege beschritt, statt in den genauso schnellen Registerbereich im PAD zu schreiben, ist unklar. Klar ist, daß man sich so unter anderem das inkompatible DSR-Konzept des Geneve einhandelte (alle Design-Fehler des Myarc-Controllers bedingen die Design-Fehler des Geneve) und daß jeder, der seine Konsolen-ROMs durch RAMs ersetzt hat, von diesem Diskcontroller seine RAMs gelöscht bekommt.

Myarc takes the liberty also here again an extra sausage, which cannot explain itself so correctly: With the "Proofreading," which reading the sector directly after it, not byte for byte was written with the buffer contents compared, but sector contents in the ROM-0 carried. Why one took this most destructive (and stupidest) all paths, instead of writing into that just as fast register area in the PAD, is unclear. It is clear that one in-concerned oneself so among other things the incompatible DSR concept of the Geneve (all design errors of the Myarc controller cause the design errors of the Geneve) and that everyone, which replaced its console ROMs by RAMs gets RAM deleted by this disk controller.

10.1.3. Subroutine >11, Diskette formatieren

Subroutine >11, diskette format

Auch diese Routine wurde bereits beschrieben. Die Pointer sind wie folgt zu benutzen:

Also this routine already one described. The pointers are to be used as follows:

>834C (B) Laufwerknummer

>834C (B) drive number

>834D (B) Spuranzahl (Hex)

>834D (B) number of tracks (hex)

>834E (W) Datenpuffer für den Spuraufbau, bei DD werden etwa 6,5 KByte benötigt

>834E (W) scratchpad memory for the track structure, with DD about 6.5K bytes are needed

>8350 (B) Dichte, >01 = Single, >02 = Double Density

>8350 (B) density, >01= single, >02 = double density

>8351 (B) Seiten, >01 = Single, >02 = Double Sided

>8351 (B) sides, >01 = single, >02 = double sided

Nach dem Formatieren werden folgende Werte übergeben:

After formatting the following values are returned:

>834A (W) Gesamtzahl aller Sektoren

>834A (W) total number of all sectors

>834D (B) Anzahl der Sektoren pro Spur

>834D (B) number of sectors per track

>8350 (B) Fehlercode

>8350 (B) error code

Nun das Beispielprogramm. Es formatiert die Diskette in Laufwerk 1 im Format Single Sided,

Now the sample program. It formats the diskette in drive 1 in the format single-sided,

Single Density:

10.1.3.1. Beispielprogramm SBR >11

single-density:

Sample program SBR >11

```
* (c) 1986 Ch. Winter
*
       DEF  INIT
       REF  DSRLNK,VMBW
*
PABAD  EQU  >1000       Adresse des PABs im VDP-RAM
BUFFER DATA >1100       Adresse des Datenpuffers
*
LW     BYTE >01         Laufwerk 1
TRACKS BYTE 40          40 Spuren
*
NAME   DATA >0111       Subroutinenname
FCODE  DATA >0101       Formatierungscode, SS, SD
MYWS   BSS  32          Eigener Workspace
*
INIT   LWPI MYWS
       MOVB @LW,@>834C        Laufwerk
       MOVB @TRACKS,@>834D    Spuranzahl
       MOV  @BUFFER,@>834E    Spurpuffer
       MOV  @FCODE,@>8350     Formatierungscode
       LI   R0,PABAD
       MOV  R0,@>8356         Zeiger auf das Namelength-Byte
       LI   R1,NAME           Position des SBR-Namens
       LI   R2,2              2 Byte verschieben
       BLWP @VMBW             PAB ins VDP-RAM schaffen
       BLWP @DSRLNK           Routine aufrufen
       DATA >A                Routinenindex
       MOVB @>8350,@>8350     Fehlerbyte prüfen
       JEQ  OK                kein Fehler
ERROR  **** ****       hier Fehlerbehandlung einbauen
OK     **** ****       hier Programm weiterführen
*
       END             Ende des Beispielprogramms
*
```

Auch bezüglich der Formatierroutine gibt es Besonderheiten bei den CorComp- und Myarc-Produkten: So kann bei CorComp die Sektor-Interleave-Liste durch Setzen eines Steuerbits im System-RAM von der Default-Tabelle im ROM auf eine anwenderdefinierte Tabelle im PAD-RAM umgeleitet werden. Da jedoch die Standard-Interleave-Werte einen Kompromiß zwischen optimaler Datenrate und breiter Austauschbarkeit der Disketten darstellen (langsame Systeme lesen z.B. DD-Disks mit IL=3 nur mühsam), sollte diese Option auf wenige Spezialanwendungen beschränkt bleiben.

Myarc benötigt einen kleineren VDP-Puffer, da der Pre-Index-GAP (der sog. Spur-Trailer, also der Bereich nach dem letzten Sektor) 'von Hand' geschrieben wird. Auch hier bleibt die Frage nach dem Warum offen.

10.1.4. Subroutine >12, Fileschutz ändern

Diese Routine ermöglicht es, den Schreibschutz irgendeines Files zu setzen oder zu löschen. Es muß lediglich der Filename und das Laufwerk angegeben werden. Folgende Pointer werden verwendet:

>834C (B) Laufwerknummer

>834D (B) Schutzcode, >00 = nicht schützen, >FF = schützen

>834E (W) Zeiger auf den Filenamen im VDP-RAM

Der Filename muß 10 Byte lang sein, ggf. muß ein kürzerer Name mit Leerzeichen aufgefüllt werden. Die Fehlercodes in >8350 entsprechen nun, da eine Fileoperation vorliegt, den Codes in der Tabelle der File-Errors! Das Programm setzt beim Einsprung über PROT den Fileschutz von 'FILENAME' und entfernt ihm beim Einsprung über UNPROT.

Nun das Beispielprogramm:

Also concerning the formatting routine there are special features with the CorComp and Myarc products: So the sector Interleave list can be rerouted by setting a control bit in the system RAM by the default table in the ROM on a user-defined table in the PAD RAM with CorComp. Since however the standard Interleave values represent a compromise between optimal data rate and broad exchangeability of the diskettes (slow systems read e.g. DD disks with IL=3 only laboriously), should this option to few special applications remain limited.

Myarc needs a smaller VDP buffer, there the pre-index gap (the so-called Track Trailer is thus written, the area after the last sector) "by hand." Remains also here the question about the why openly.

Subroutine >12, File protection modify

This routine enables it to set or reset the write protection any files. The file name and the drive must be only indicated. The following pointers are used:

834C (B) drive number

>834D (B) protection code, >00 = do not protect, >FF = protect

>834E (W) pointer to the file name in VDP RAM

The file name must be 10 bytes long, if necessary blanks are used to fill a shorter name. Those Error codes in >8350 correspond now, since a file operation is present, to the codes in the table of the file errors! The program sets the file protection of "FILE NAME" with the re-entry point over PROT and removes for it with the re-entry point over UNPROT.

Now the sample program:

10.1.4.1. Beispielprogramm SBR >12 *Sample program SBR >12*

```
* (c) 1986 Ch. Winter
*
       DEF PROT,UNPROT
       REF DSRLNK,VMBW
*
PABAD  EQU >1000 Adresse des PABs im VDP-RAM
*
LW     BYTE >01 Laufwerk 1
       EVEN
*
NAME   TEXT 'FILENAME'     Filename (beliebig), 10 Bytes lang!
       DATA >0112          Subroutinenname
MYWS   BSS 32              Eigener Workspace
*
PROT   LWPI MYWS
       SETO R0             Fileschutz setzen
       JMP  PU             weitermachen
UNPROT LWPI MYWS
       CLR  R0             Fileschutz entfernen
PU     MOVB @LW,@>834C     Laufwerk
       MOVB R0,@>834D      Schutzcode angeben
       LI   R0,PABAD
       MOV  R0,@>834E      Zeiger auf Filenamen
       LI   R1,NAME        Position des Namens
       LI   R2,12          12 Bytes verschieben
       BLWP @VMBW          PAB ins VDP-RAM schaffen
       LI   R0,PABAD+10    Zeiger auf SBR-Namelength
       MOV  R0,@>8356      Namelength Pointer
       BLWP @DSRLNK        Routine aufrufen
       DATA >A             Routinenindex
       MOVB @>8350,@>8350  Fehlerbyte prüfen
       JEQ  OK             kein Fehler
ERROR **** ****            hier Fehlerbehandlung einbauen
OK  **** ****              hier Programm weiterführen
*
```

10.1.5. Subroutine >13, Filenamen ändern

Subroutine >13, File name modify

Mit dieser Routine kann der Name eines bestehenden Files geändert werden. Der alte und der neue Name müssen im VDP-RAM übergeben werden. Beide müssen jeweils 10 Byte lang sein, ggf. mit Leerzeichen auffüllen!

With this routine can the name be modified The old and the new name must be transferred in the VDP RAM. Both must in each case be 10 bytes long. If necessary, fill with blanks!

Die Pointer:

The pointers:

>834C (B) Laufwerknummer

>834E (W) Zeiger auf den neuen Filenamen

>8350 (W) Zeiger auf den alten Filenamen

Die Übergabe des File-Error Bytes geschieht wie gehabt in >8350 (B). Es kann nur ein bestehendes File umbenannt werden, das nicht schreibgeschützt sein darf!

>834C (B) drive number

>834E (W) pointer to the new file name

>8350 (W) pointer to the old file name

The transfer file error of the byte occurs as had in >8350 (B). Only an existing file can be renamed, which may not be write protected!

10.1.5.1. Beispielprogramm SBR >13

Sample program SBR >13

```
*  (c)1986 Ch. Winter
*

       DEF CHGNAM
       REF DSRLNK,VMBW
*
PABAD  EQU >1000          Adresse des PABs im VDP-RAM
*
LW     BYTE >01           Laufwerk 1
       EVEN
*
NAME   TEXT 'NAME-ALT'    Filename (alt), 10 Bytes lang!
       TEXT 'NAME-NEU'    Filename (neu), 10 Bytes lang
       DATA >0113         Subroutinenname
MYWS   BSS  32            Eigener Workspace
*
CHGNAM LWPI MYWS
       MOVB @LW,@>834C    Laufwerk
       LI   R0,PABAD
       LI   R1,NAME       Position des Namens
       LI   R2,22         22 Bytes verschieben
       BLWP @VMBW         PAB ins VDP-RAM schaffen
       MOV  R0,@>8350     Zeiger auf den alten Namen
       LI   R0,PABAD+10   Zeiger auf den neuen Namen
       MOV  R0,@>834E
       LI   R0,PABAD+20   Zeiger auf Namenlänge
       MOV  R0,@>8356
       BLWP @DSRLNK       Routine aufrufen
       DATA >A            Routinenindex
       MOVB @>8350,@>8350 Fehlerbyte prüfen
       JEQ  OK
ERROR **** ****           hier Fehlerbehandlung einbauen
OK  **** ****             hier Programm weiterführen
*
       END                Ende des Beispielprogramms
```

10.1.6. Subroutine >14, File-Copy Input

Hierbei handelt es sich um eine funktionell sehr umfangreiche Routine, zu der die Routine >15 gehört. Beide Routinen ermöglichen es, ein komplettes File, wie viele Sektoren es auch umfassen mag, stückweise zu kopieren. Dementsprechend sind die über Pointer zwischen den beiden Routinen übergebenen Parameter von sehr komplexer Struktur.

Die Pointer:

>834C (B) Laufwerknummer

>834D (B) Sektoranzahl, Filestatus wenn >00!

>834E (W) Zeiger auf den Filenamen (10-stelliger String)

>8350 (B) Offset des Parameterblocks im PAD. Dieses Byte ist das niederwertige Byte eines Zeigers in das PAD-RAM, wo die File-Parameter stehen. Das höchstwertige Byte ist demnach >83.

Aufbau des Parameterblocks im PAD-RAM:

Subroutine >14, File Copy input

Here it concerns a functionally very extensive routine, to which the routine >15 belongs to. Both routines enable to also cover it, a complete file, as many sectors it may copy by the piece. Over pointers between both routines are corresponding the transferred parameter of very complex structure.

The pointers:

>834C (B) drive number

>834D (B) number of sectors, file status if >00!

>834E (W) pointers on the file name (string with 10 digits)

>8350 (B) offset of the parameter block in the PAD. This byte is the low order byte of a pointer in PAD RAM, where the file parameters are. The most significant byte is therefore >83.

Structure of the parameter block in PAD RAM:

Byte	*Input*	*Output*	*Input*	*Output*
0.1	Datenpufferadresse		Scratchpad memory address	
2.3	Sektoroffset bei dem begonnen werden soll, 0 entspricht dabei dem Anfang des Files.	Anzahl vom File belegter Sektoren	Sector offset with to be begun is, 0 corresponds thereby to the at the beginning files.	Number of the file of occupied sectors
4		File-Status aus dem FDS		File status from the FDS
5		Datensätze pro Sektor		Data records per sector
6		EOF-Offset im letzten Sektor		EOF offset in the last sector

Byte	*Input*	*Output*	*Input*	*Output*
7		Satzlänge		Record length
8.9		Anzahl der Einträge/Records (geswapt!)		Number of entry/records (geswapt!)

Der Output in den Bytes 2 bis 9 geschieht nur, wenn die Anzahl der zu lesenden/kopierenden Sektoren >00 war. Damit kann das steuernde Programm den Status eines Files lesen, bevor mit dem Kopieren begonnen wird.

Nach Ausführung der Routine befindet sich in >834C (W) die Anzahl der tatsächlich gelesenen Sektoren, falls mehr als vorhanden spezifiziert wurden, ansonsten steht hier der in >834D angegebene Wert.

Das Kopieren eines Files geschieht dann in einer Art Schneeballverfahren, indem die Routinen >14 und >15 abwechselnd aufgerufen werden, bis keine Sektoren mehr zu kopieren sind. Es ist einfach möglich, immer die Anzahl von Sektoren, die kopiert werden sollen, anzugeben, die der VDP-Puffer maximal fassen kann. Ohne Rücksicht auf die tatsächliche Filelänge wird nun so lange kopiert, bis die Rückgabe in >834C (W) null ist. Natürlich muß der Startsektor eines Durchganges immer mitlaufen, also entsprechend der Puffergröße inkrementiert werden! Die Routinen prüfen selbst, wann das Ende des Files erreicht ist.

Die gelesenen Daten werden ab der im ersten Wort des Parameterblocks angegebenen VDP-Adresse in 256 Byte Blöcken, einer pro Sektor, aufsteigend abgelegt.

Die DOS-Routine >14 kann, wie o.ä., nicht getrennt von der Routine >15 besprochen werden. Vor dem zusammenfassenden Beispielprogramm soll daher auf SBR >15 eingegangen werden.

The output in the bytes 2 to 9 occurs only, if the number of sectors which can be read/copied were >00. Thus the controlling program can read the status files, before with copying one begins.

After execution of the routine the number of actually read sectors is in >834C (W), if more were than available specified, is otherwise stands here the value indicated in 834C (W).

Copying files occurs then in a type snow ball procedure, as the routines are alternating called >14 and >15, until no more sectors are to be copied. It is simply possible, always the number of sectors, which to be copied to be supposed to indicate which the VDP buffer can seize max. Without consideration for the actual file length now so for a long time one copies, until the return is in >834C (W) zero. Naturally the start sector of a passage must always run along, increment thus according to the buffer size! The routines check, when the end files is achieved.

The read data are ascending stored starting from the VDP address indicated in the first word of the parameter block into 256 byte blocks, one per sector.

The DOS routine >14 can, as or the like separately from the routine >15 is discussed. Before the recapitulatory sample program is to be dealt therefore with SBR >15.

10.1.7. Subroutine >15, File-Copy Output

Wegen der engen Beziehung zu SBR >14 sind die Pointer und deren Verwendung identisch, jedoch mit dem Unterschied, daß der File-Status ein neues File erzeugt, zumindest dessen Directory-Eintrag. Hierfür werden die Daten des PAD-Parameter Blocks verwendet.

Ebenso werden bei der Eröffnung des Zielfiles alle für die Länge des Files notwendigen Sektoren in der Sektor-Bitmap reserviert. Führt man nun nur einen Status-Transfer von der Quelle zum Ziel aus, so existiert ein ordnungsgemäßes File, dessen Datensektoren aber nicht die richtigen Daten enthalten.

Die Anzahl zu kopierender Sektoren wird nicht geprüft, hier sollte also der Rückgabewert der Routine >14 nicht verändert werden.

Sollte es passieren, daß das Zielfile den Namen eines bestehenden Files auf der Diskette hat, so wird die Routine mit einem File-Error abgeschlossen, der in >8350 zurückgegeben wird. Die Namen des Quellfiles und des Zielfiles (Source und Copy) müssen nicht übereinstimmen!

Bei jedem Durchlauf der Sequenz SBR >14 - SBR >15 wird immer wieder der FDS des entsprechenden Files gelesen. Das ist zwar lästig und wenig sinnvoll, aber nicht zu ändern, da Firmware. Es erfordert aber, daß die Filenamen immer übergeben werden und während des Kopierens die dem Quell- und dem Zielfile zugeordneten Namen immer gleich bleiben, bis das gesamte File kopiert ist!

Zum Beispielprogramm: Es ist diesmal etwas kompakter programmiert, um ein paar kleine Tricks zur Reduzierung des belegten Speicherplatzes zu zeigen. Das Programm kopiert in nur einem Durchlauf und maximal 32 Sektoren. Sollen längere File kopiert werden, so

Subroutine >15, File Copy output

Ways of the close relationship with SBR >14 are the pointers and their use identically, however with the difference that the file status a new file produces, at least its directory entry. For this the data PAD parameters of the block are used.

Likewise with the initialization target files all sectors in the sector bitmap, necessary for the length files, are reserved. If one executes now only a status transfer from the source to the target, then exists a file correct, whose data sectors do not contain however the correct data.

Copying sectors will not checked the number, here should thus the return value of the routine >14 not be changed.

If it should occur that the target file has the name existing files on the diskette, then the routine with a file error is locked, which is returned in >8350. The names source files and target files (Source and Copy) do not have to correspond!

With each run of the sequence SBR >14 - SBR >15 is read again and again the FDS appropriate files. That is annoying and a little meaningful to modify but not there of firmware. It requires however that the file names are always transferred and during copying those to the pouring and the target file assigned name to remain always alike, until the entire file is copied!

To the sample program: It is programmed this time somewhat more compactly, in order a few small cheats for the reduction of the occupied storage space to show. The program copies in only one run and max. 32 sectors. Is to be copied longer file, then its number of sectors is to be

ist zu Anfang nach dem Status des Quellfiles dessen Sektoranzahl zu lesen und in jedem Durchlauf zu dekrementieren, wobei natürlich auch der Startsektor des jeweiligen Kopiervorgangs erhöht werden muß. Ende ist dann, wenn die Sektoranzahl 0 ist.

Der Kopiervorgang beginnt beim Sektoroffset 0 und endet mit dem letzten vom File belegten Sektor. Die Rückgabe der Anzahl der File-Sektoren (Status) wie im vorigen Absatz beschrieben, schließt den FDS nicht mit ein! Dieser wird bei der Übergabe der File-Parameter durch den Status-Aufruf erzeugt. Im Beispielprogramm wird dieser Wert direkt übergeben bzw. nicht geändert zwischen dem Aufruf der beiden Routinen hintereinander.

Es wurde auf jegliche Fehlerbehandlung verzichtet, diese muß bei Bedarf nach jedem Routinenaufruf durchgeführt werden. Da es sich hierbei um eine File-Operation handelt, werden die entsprechenden File-Error Codes in >8350 geliefert.

Es können nun während des Kopiervorgangs an zwei verschiedenen Stellen Fehler auftreten, die unterschiedlich behandelt werden müssen. Zuerst kann es sein, daß bei der Erzeugung des FDS des Zielfiles nicht alle notwendigen Sektoren reserviert werden können, daß also die Diskette bereits zu voll ist. In diesem Fall wird in >8350 der entsprechende Fehlercode >04 geliefert und es werden alle bisher für dieses File reservierten Sektoren gelöscht- die Zieldiskette bleibt unangetastet. Tritt beim Schreiben eines Datensektors ein Fehler auf, so bleibt der FDS unberührt. In diesem Fall muß das File anschließend gelöscht werden, um den Platz auf der Diskette frei zu machen.

read and be decremented in each run at the beginning after the status source files, whereby naturally also the start sector of the respective copying process must be increased. End is if the number of sectors is 0.

The copying process begins with the sector offset 0 and ends with the last sector occupied by the file. Those Return of the number of file sectors (status) as in the previous paragraph described, includes the FDS not also! This is produced with the transfer of the file parameters by the status call. In the sample program this value will transfer directly or. not modified between the call of the two routines consecutively.

One did without any error handling, these must be executed if necessary after each routine call. Since it concerns here a file operation, the appropriate file error codes are supplied to >8350.

Now errors can occur during the copying process in two different places, which must be differently treated. First it can be that with the production of the FDS target files cannot be reserved all necessary sectors that thus the diskette is already too full. In this case supplied to >8350 the appropriate error code will become >04 and it all sectors deletion the target diskette reserved so far for this file remains untouched. If an error occurs during the writing of a data sector, then the FDS remains unaffected. In this case the file must be deleted afterwards, in order to make the place on the diskette free.

10.1.7.1. Beispielprogramm Subroutinen >14 und >15

Sample program subroutines >14 and >15.

```
*
* (c) 1986 Ch. Winter

       DEF START
       REF DSRLNK,VMBW

PABAD  EQU >F00           PAB-Pufferadresse

LWA    DATA >0100         LW A Status
LWB    DATA >0200         LW B Status
LWA2   DATA >0120         LW A kopieren
PUFFER DATA >1000         Datenpuffer

FILE   TEXT 'FILENAME'    Filename
SUB    DATA >0114         SBR-Name, eigenes Label für Änderungen
MYWS   BSS 32             Workspace

START  LWPI MYWS
       MOV @LWA,@>834C    LW und Status
       BL @SBR2           Filestatus lesen
       INC @SUB           auf >0115
       MOV @LWB,@>834C    LW und Status
       BL @SBR2           Filestatus erzeugen
       DEC @SUB           auf >0114
       MOV @LWA2,@>834C
       BL @SBR1
       INC @SUB           auf >0115
       MOVB @LWB,@>834C   Sektoranzahl stehenlassen wenn unter 256!!
       BL @SBR1

       BLWP @0            Programmende

SBR1   CLR @>8302         Startsektor=0, nur ein Lauf!!
SBR2   LI R0,PABAD
       LI R1,FILE         Filename
       LI R2,12
       BLWP @VMBW
       CLR @>8350          Parameterblock ab >8300
       MOV @PUFFER,@>8300  Datenpufferadresse
       MOV R0,@>834E       Filenamenzeiger
       LI R0,PABAD+10
       MOV R0,@>8356       POINT
       BLWP @DSRLNK
       DATA >A
       RT
```

```
END
```

Zum Beispielprogramm:

Die Daten LWA, LWB und LWA2 geben direkt die Laufwerknummer und den Operationsmodus an. Im ersten Aufruf von SBR2 wird also der Status des Files 'FILENAME' auf der Diskette in Laufwerk 1 ermittelt.

Dieses File muß existieren, andernfalls tritt eine Fehlerbedingung auf.

Direkt danach wird nach Änderung des Subroutinennamens die Routine >15 im Status-Mode aufgerufen, die mit den Daten im PAD-Block einen neuen FDS (File-Directory-Sektor) erzeugt und die Sektor-Bitmap aktualisiert.

Sodann wird wieder der Name repariert und SBR1 aufgerufen, die im Gegensatz zu SBR2 den Startsektor zu null setzt. Dabei ist LWA2 im Low-Byte die Maximalzahl zu lesender Sektoren enthalten.

Der tatsächliche Wert wird in >834C (W) zurückgegeben. Da dieser nur im Low-Byte stehen kann (kleiner gleich 32), kann die Ziellaufwerknummer ins High-Byte. Da der VDP-Puffer maximal etwa 15 kByte fassen kann, wird dieser Wert im Low-Byte auch nie größer als 60 werden!

Das Beispielprogramm kopiert also das File 'FILENAME' von Laufwerk 1 auf Laufwerk 2 mit demselben Filenamen. Die festgelegte Maximallänge von 32 Datensektoren sei hier nochmals erwähnt.

Damit ist die Beschreibung der File-Routinen beendet.

10.1.8. Subroutine >16, CALL FILES

Auch aus einem Assemblerprogramm heraus ist es möglich, dieses Kommando auszuführen. Je

The sample program:

The data LWA, LWB and LWA2 indicate directly the drive number and the operation mode. In the first call of SBR2 the status files "FILE NAME" on the diskette in drive of the 1 is thus determined

This file must exist, otherwise one an error condition occurs.

After it after modification of the subroutine name the routine is called directly >15 in the Status mode, which with the data in the PAD block a new FDS (file directory sector) produced and which updates sector bitmap.

Then again the name is repaired and called SBR1, which sets the start sector contrary to SBR2 to zero. LWA2 is contained in the Low byte the maximum number of reading sectors.

The actual value is returned in >834C (W). Since this can be located only in the Low byte (smaller directly 32), can the target drive number in the High byte. Since the VDP buffer max. about 15 kByte seize can, this value in the Low byte will also never become larger than 60!

The sample program copies thus the file "FILE NAME" from drive 1 on drive 2 with the same file name. The determined maximum length of 32 data sectors is here again mentioned.

Thus the description of the file routines is terminated.

Subroutine >16, CALL FILES

Also from an assembler program is possible it to execute this command. Depending upon disk

nach Diskcontroller sind hier sogar bis zu 16 gleichzeitig offene Files erlaubt (Atronic V1.0 und 1.1), ansonsten gilt das im Basic festgelegte Limit von maximal 9 gleichzeitig offenen Files.

Der Aufruf dieses Programms ist so trivial, daß hier kein Beispielprogramm aufgeführt wird. Es muß jedoch sichergestellt sein, daß zum Zeitpunkt, da diese Funktion aufgerufen wird, keine Datei mehr offen ist, da es sonst passieren kann, daß der entsprechende Puffer gelöscht wird (nur bei einer Reduzierung der Pufferzone).

Die Anzahl der Files (nicht 0) wird in >834C (B) übergeben, der Aufruf erfolgt mittels des Standard SBR-PABs wie er in den vorangegangenen Beispielen verwendet wurde. Die Zahl im FAC muß in hexadezimaler Notation angegeben werden, nicht etwa in ASCII. Wurde eine falsche Zahl angegeben, so erfolgt die Fehlermeldung in >8350 (>FFFF bei Fehler, >0000 wenn o.K.).

10.1.9. Weitergehende Funktionen

10.1.9.1. Diskettenkatalog im Standardverfahren

An einem solchen Programm wird wohl niemand verzweifeln, es sei denn er (sie) wüßte nicht, welche Daten ein Record der Datei 'DSKX.' beinhaltet. Da die Möglichkeiten, einen Diskettenkatalog darzustellen extrem vielfältig sind, soll hier nur das Prinzip gezeigt werden.

Zuerst wird die Datei im Format 'INPUT, INTERNAL, FIXED 38' normal eröffnet, wozu ein Standard-PAB verwendet wird. Zulässige Namen sind hier, je nach Controller 'DSK1.' bis 'DSK4.'.

Sodann werden die Datensätze in den Pufferbereich gelesen. Die gelieferten Daten gliedern sich wie folgt:

1. Byte: Namenlänge.

controllers are here even up to 16 at the same time open files permitted (Atronic v1.0 and 1,1), otherwise applies the limit of max. 9 at the same time open files, determined in the Basic.

The call of this program is so trivial that no sample program is specified here. It must be however guaranteed that at the point in time, since this function is called no file is more open, since it can occur otherwise that the appropriate buffer is deleted (only with a reduction of the buffer zone).

The number of files (not 0) was transferred into >834C (B), the call effected by means of standard the SBR PABs as was used it in the preceding examples. The number in the FAC must be indicated in hexadecimal notation, not in ASCII. If a false number was indicated, then the error message takes place in >8350 (>FFFF in the case of error, >0000 if OK).

Large functions

Diskette catalog in the standard technique

At such a program nobody will probably despair, it is not it (it) would know not, which data a record of the file "DSKX." contained. Since the possibilities are of representing a diskette catalog extremely various, only the principle is to be demonstrated here

First the file in the format "INPUT, INTERNAL, FIXED 38" is normally opened, to which a standard PAB is used. Admissible names are here, depending upon controller "DSK1." to "DSK4.".

Then the data records are read into the buffer area. The supplied data were arranged as follows:

1. Byte: Name length.

2. Byte und folgende: Disk- bzw. Filename.

dann >08 Anzahl Bytes der folgenden Zahl.

dann 8 Byte im RADIX 100 Format.
>08 nächste Zahl.
8 Byte der nächsten Zahl.
>08 letzte Zahl.
8 Bytes der letzten Zahl.

Wie man sieht, ist der Output keineswegs immer FIXED 38, sondern variiert in der Länge je nach File- bzw. Diskettenname.

Im ersten Record werden die Diskettendaten übergeben. Der String zu Anfang enthält also den Diskettennamen. Die erste Zahl enthält eine Null im RADIX 100 Format, die zweite die Gesamtzahl aller Sektoren der Diskette und die letzte Zahl die Anzahl freier Sektoren.

In jedem weiteren Record werden Filedaten übergeben. Zuerst der Filename, dann der Filetyp, die Anzahl vom File belegter Sektoren inclusive seines FDS und zum Schluß die Datensatzlänge.

Es stellt sich nun als einziges Problem die Umwandlung einer RADIX 100 Zahl in eine Dezimalzahl. Vereinfacht wird dies dadurch, daß nur ganze Zahlen zwischen -9999 und 9999 übergeben werden.

Eine Zahl im Radix-100 Format besteht im TI aus 8 Bytes, wobei das Längenbyte nicht gezählt wird. Das erste Byte stellt den Exponenten dar, also die Potenz, in die 100 erhoben wird. Jedes folgende Byte gibt 2 Stellen einer Dezimalzahl an. Das bedeutet, daß der Wert eines jeden Bytes nur von >0 bis >63 betragen kann, entsprechend 0 bis 99 im Dezimalsystem. Die Bytes müssen also in Dezimalzahlen umgewandelt werden. Das erste, dem Exponenten folgende Byte gibt den Wert der Vorkommastellen an, alle 6 folgenden die

2. Byte and the following: Disk or File name.

then >08 number of bytes of the following number.

then 8 byte in the radix 100 format.
>08 next number.
8 byte of the next number.
>08 last number.
8 bytes of the last number.

As one sees, the output is not by any means always, but varied FIXED 38 in the length depending upon file or diskette name.

In the first record the diskette data will transfer. The string at the beginning contains thus the diskette name. The first number contains a zero in the radix 100 format, second the total number of all sectors of the diskette and the last number the number of free sectors.

In each further record file data will transfer. First the file name, then the type of file, the number of the file of occupied sectors inclusive its FDS and in the end the data record length.

Now only problem the transformation of a radix 100 number places itself into a decimal number as. Simplified becomes this by the fact that only whole numbers are transferred between -9999 and 9999.

A number in the radix 100 format consists in the TI of 8 bytes, whereby the length byte is not counted. That first byte represents the exponents, thus the power, into which 100 one raises. Each following byte indicates 2 places of a decimal number. That means it that the value of each byte can amount to only from >0 to >63, according to 0 to 99 in the decimal system. The bytes must be converted thus into decimal numbers. First, the exponent the following byte indicates the value of the integer digits, all 6 the following post-decimal positions.

Nachkommastellen.

Besondere Aufmerksamkeit verdient der Exponent. Dieser nimmt jedoch in diesem speziellen Fall nur den Wert 0 oder 1 an. Zur eindeutigen Erkennung negativer Mantissen wurde jedoch die Mitte des Zahlenkreises für den Exponenten auf >40 gelegt. Das bedeutet, daß ein Exponent von tatsächlich 0 als >40 erscheint, einer von 1 als >41 übergeben wird.

Negative Mantissen werden daran erkannt, daß das erste Wort der Radix 100 Zahl negativ ist (High-Bit gesetzt, siehe weiter unten). Das bedeutet aber, daß der Zahlenbereich für den Exponenten von >00 bis >7F festgelegt ist. Dementsprechend sind Zahlen nur im Bereich ±100 E 63 darstellbar.

Vom Exponenten muß also >40 oder, um direkt einen ASCII-Wert zu erhalten, >10 abgezogen werden. Eine Anzeige des Exponenten ist aber nicht immer die sinnvollste, zumal im Falle der Datei 'DSKX.' die einzige aus dem Exponenten ableitbare Information die ist, ob die Zahl 4- oder nur 2-stellig ist.

Ist der Exponent Null (>40), so folgt dem Exponentenbyte nur ein einziges weiteres Byte, welches Zehner und Einer darstellt. Ein Exponent von Eins zeigt an, daß im folgenden Byte die Tausender und Hunderter stehen, auf das die Zehner und Einer folgen.

Liegen negative Zahlen vor, wie es beim Filetyp passieren kann, so ist das erste Wort der Radix 100 Zahl, also Exponent und Vorkommastellen, negativ (erstes Bit gesetzt). Dieses Wort liegt dann in der Zweierkomplement-Form vor und muß mit NEG in eine positive Zahl umgewandelt werden. Nun stimmt der Exponent wieder (0 oder 1) und vor der Anzeige der Zahl muß lediglich ein '-' ausgegeben werden.

Nach dieser Theorie nun einige Beispiele:

Special attention earns the exponent. This takes however in this special case only the value 0 or 1. For the unique recognition of negative mantissas however the center of the number set for the exponent was put on >40. That means that an exponent of actually 0 as >40 appears, one of 1 as >41 is transferred.

Negative mantissas are detected by the fact that the first word of the radix is negative 100 number (High bits set, see further below). That means it however that the number range is determined for the exponent from >00 to >7F. Accordingly numbers are representable only in the area ±100 E 63.

Of the exponent must thus >40 or, in order to be received, >10 taken off directly a ASCII value. A display of the exponent is however not always the most meaningful, particularly in the case of the file "DSKX." the only information those derivable from the exponent is whether the number is only with 4 or 2 digits.

Is the exponent zero (>40), then follows the exponent byte a only one further byte, which decimal and one represent. An exponent of unity displays that in the following byte thousands and hundreds are located, on which the decimals and one follows.

Couches negative numbers forwards, as it can occur with the type of file, then the first word of the radix 100 number, thus exponent and integer digits, is negative (first bit set). This word is present then in the complement on two form and must be converted with NEG into a positive number. Now the exponent is correct again (0 or 1) and before the display of the number must be output only "-".

According to this theory now some examples:

Freie Sektoren: 1438 (dezimal)

Radix 100 Zahl:
>08 - Längenbyte
>41 - Exponent 1, also Mantisse mal 100 hoch 1
>0E - dezimal gleich 14
>26 - dezimal gleich 38
5 mal >00 - Leerpositionen

Filetyp: -5, geschütztes PROGRAM-File

Radix 100 Zahl:
>08 Längenbyte
>BFFB 1. Wort negativ, Negation liefert
>40 Exponent 0
>05 dezimal gleich 5
6 mal >00 Leerpositionen

Weitere Beispiele finden sich im Manual zum Editor/Assembler Modul.

Für die Zahlenanzeige ist also eine Routine notwendig, welche Hexzahlen in vierstellige Dezimalzahlen umwandelt, am besten gleich als ASCII-String. Hier sei abschließend auf die XMLLNK/CFI-Routine verwiesen.

Free sectors: 1438 (decimal)

Radix 100 number:
>08 — Length byte
>41 — exponent 1, thus mantissa times 100 to the power of 1
>0E — decimal equivalent 14
>26 — decimal equivalent 38
5 times >00 — empty positions

Type of file: -5, protected PROGRAM file

Radix 100 number:
>08 length byte
>BFFB 1. Word negatively, negation supplies >40 exponent 0
>05 decimal directly 5
6 times >00 empty positions

Further examples can be found in the Editor/Assembler manual.

For the number display thus a routine is necessarily, which hex digits converts into four-digit decimal numbers, best alike as ASCII string. Here is finally referred to the XMLLNK/CFI routine.

11. Liste der Fehlercodes bei Diskettenoperationen

Die Übergabe der Fehlercodes erfolgt nach Ausführung einer DOS-Routine entweder im Status-Byte des PAB's oder in >8350 (B) wenn eine Routine aufgerufen wurde, die keinen regulären PAB benötigt.

Fehlercodes im PAB sind immer File-Error Codes, Fehlerbytes in >8350 (B) können File-Error Codes oder Hardware-Error Codes sein.

11.1. Liste der Fehlercodes bei Sektor-I/O oder Formatieren

- >00 kein Fehler
- >06 Hardwarefehler, Laufwerk oder Controller nicht i.O.
- >07 Softwarefehler, Laufwerk- oder Sektornummer falsch
- >11 Spur nicht gefunden oder alle Schreib/Leseversuche sind fehlgeschlagen
- >21 Lesefehler, Record-Not-Found, RNF
- >22 Lesefehler, Prüfsummenfehler im Sektor, CRC
- >23 Lesefehler, Lost-Data Zustand, sollte nicht vorkommen, da dies signalisiert, daß die CPU mit dem FDC nicht Schritt halten konnte.
- >28 Verify-Fehler, ein Sektor konnte mehrfach nach dem Lesen nicht verifiziert werden.
- >31 Schreibfehler, Record-Not-Found, RNF
- >33 Schreibfehler, Lost-Data, siehe Code >23
- >34 Diskette ist schreibgeschützt

List of error codes with diskette operations

The transfer of the error codes effected after execution of a DOS routine either in the status byte of the PAB's or in >8350 (B) if a routine called was needed, no regular PAB.

Error codes in the PAB are always file error code, error byte in >8350 (B) can file error code or hardware error code be.

List of the error codes with sector I/O or formatting

- >00 no error
- >06 hardware errors, drive or controller not in order.
- >07 software faults, drive or sector number falsely
- >11 track not found or all write/read attempts is not missed
- >21 read errors, record NOT Found, RNF
- >22 read errors, check total errors in the sector, (carriage return character)
- >23 read errors, lost data status, should not occur, since this signals that the CPU could not keep up with the FDC.
- >28 Verify errors, a sector could not be verified several times after reading.
- >31 write errors, record not found, RNF
- >33 write errors, draw DATA, see code >23
- >34 diskette is write protected

Die Fehler >11 bis >33 werden erst ausgegeben, nachdem mehrere Wiederholungen fehlschlugen. Das DOS läßt im Allgemeinen bis zu 10 Versuche (Retries) zu, bevor mit einer dieser Fehlermeldungen abgebrochen wird. Beim Fehler >22 befinden sich jedoch alle Daten des betreffenden Sektors im spezifizierten Puffer. War also lediglich die Prüfsumme 'umgefallen', so kann der Sektor durch einfaches Rückschreiben wieder repariert werden!

11.2. Liste der File-Error Codes

- >00 Kein Fehler oder falscher Gerätename. Hier muß zuvor das Equal-Bit direkt nach der Rückkehr von DSRLNK geprüft werden, ob der Fehlercode gültig ist!
- >01 Schreibgeschützt
- >02 Fehlerhafte Parameter bei der Eröffnung eines Files
- >03 Illegale Operation
- >04 Kein Platz für einen File-Puffer im VDP-RAM gefunden oder Diskette ist voll
- >05 Versuch, über EOF hinaus zu lesen
- >06 Allgemeiner Hardware-Fehler
- >07 File-Error, alle Fehler, die die vorangegangenen Codes nicht überdecken

The errors >11 to >33 are only output after several repetitions failed. That DOS permits generally up to 10 attempts (retries), before with one of these error messages one aborts. In the case of the error >22 is however all data of the sector concerned in the specified buffer. If the check total "had thus only fallen down," then the sector can be repaired by simply rewriting again!

List of file error codes

- >00 no error or false device name. Here the Equal bit must be checked before directly after the return by DSRLNK whether the error code is valid!
- >01 write protected
- >02 incorrect parameters with the initialization files
- >03 illegal operation
- >04 No space for a file buffer in VDP RAM found or diskette is full
- >05 attempt to read beyond EOF
- >06 general hardware error
- >07 file error, all errors, which do not cover the preceding codes.

12. Hardwareinformationen

12.1. Pinbelegung des Platinensteckers am Laufwerk

Die Laufwerke werden über einen 34-poligen Platinenstecker und ein 34-poliges Flachbandkabel mit dem Controller verbunden, das maximal 3 Meter lang sein darf. Die Kontakte mit den ungeraden Nummern befinden sich auf der Unterseite der Platine und führen alle Massepotential. Somit ist durch die Verwendung von Flachkabel und der entsprechenden Stecker sichergestellt, daß zwischen zwei Signalleitungen immer Masse liegt.

Die Kontakte mit geraden Nummer liegen folgerichtig an der Oberseite und führen Signalspannungen.

Die Belegung der Kontakte ist genormt, wenn auch die Stecker z.B. bei 3,5 Zoll Laufwerken anders aussehen! Man spricht hier von einem sog. Shugart-Bus, was die Kompatibilität durch Anlehnung an einen gemeinsamen Standard andeuten soll.

Im Folgenden nun die Bezeichnung der einzelnen Kontakte (Pins) und deren Funktion:

Hardware information

Pin allocation of the circuit board plug at the drive

By a 34-position circuit board plugs and a 34-conductor flat cable are connected to the drives with the controller, which may be max. 3 meters long. The contacts with the odd numbers are on the lower surface of the circuit board and are all connected to ground potential. Thus it is guaranteed by the use of a flat cable and the appropriate plugs that between two signal lines ground is always situated.

The contacts with even numbers are logically on the top side and carry signal voltages.

The allocation of the contacts is standardized, even if the plugs look e.g. with 3.5 inch drives different! One speaks here of one so-called Shugart bus, which is the compatibility by support to a common standard suggest.

In the following one now the designation of the individual contacts (pin) and their function:

Pin	*Out/In*	*Beschreibung der Funktion*	*Description of function*
2	OI	Optional, frei verfügbar	Optional, freely available
4	I	IN USE	In use
6	I	DS3, Anwahl für Laufwerk 4	DS3, selection for drive 4
8	O	INDEX, Ausgang des Index-Sensors, 5 Hz nominal!	INDEX, output of the index sensor, 5 Hz nominally!
10	I	DS0, Anwahl für Laufwerk 1	DS9, selection for drive 1
12	I	DS1, Anwahl für Laufwerk 2	DS1, selection for drive 2

Pin	*Out/In*	*Beschreibung der Funktion*	*Description of function*
14	I	DS2, Anwahl für Laufwerk 3	D23, selection for drive 3
16	I	MOTOR ON, direktes Anschalten des Motors, alle LW!	MOTOR ON, direct turning of the motor, all drives on!
18	I	DIR, Richtung, in welcher der Kopf bewegt werden soll	DIR, direction, in which the head is to be moved
20	I	STEP, Schrittimpulse an den Stepper Motor	STEP, clocks to stepper motor
22	I	WRITE DATA, serieller Datenstrom vom Controller	WRITE DATA, serial data stream of the controller
24	I	WRITE GATE, umschalten auf Schreibbetrieb	WRITE GATE, switches to write operation
26	O	TRK 00, aktiv wenn der Kopf über Spur 0 steht	TRK 00, active if head is over track 0
28	O	WRTPT, Write Protect, Schreibschutz aktiviert	WRTPT, Write protect, write protection activates
30	O	READ DATA, serieller Datenstrom zum Controller	READ DATA, serial data stream to the controller
32	I	SIDE SELECT, Anwahl der Ober- bzw. Unterseite	SIDE SELECT, selection of the upper or lower surface
34	O	READY, Laufwerk bereit, Drehzahl o.K.	READY, drive ready, number of revolutions OK.

12.2. Wellenwiderstand und Anpassung

Alle Laufwerke sind parallel über ein 34-poliges Flachbandkabel miteinander verbunden. Da die höchste zu übertragende Frequenz auf dem Kabel etwa bei 250 kHz liegt, muß der Wellenwiderstand an den Kabelenden mit dem des Kabels übereinstimmen. Der Wellenwiderstand von Flachkabeln kann in der Praxis mit etwa 100 Ohm angesetzt werden.

Characteristic impedance and adjustment

All drives are parallel connected by a 34-conductor flat cable together. Since the highest frequency which can be transferred is on the cable approximately about 250 kHz, the characteristic impedance at the cable ends must correspond with that of the cable. The characteristic impedance of flat cables can be set in practice with approximately 100 ohms.

Die verwendeten Schaltglieder in TTL-Technik, für Ausgänge vorzugsweise 7438 und für Eingänge 74LS04 oder 74LS14, weisen jedoch typische Eingangswiderstände von ca. 1 kOhm auf.

Um nun bei der hohen Datenfrequenz ein Signalüber- oder allgemein -nachschwingen zu vermeiden, muß das Ende des Anschlußkabels, also am letzten Laufwerk, mit sogenannten 'Terminator'-Widerständen gegen +5V abgeschlossen werden. Diese sollten als DIP-Widerstände etwa 150 Ohm aufweisen. Es ist nur ein 'Terminator-Chip' notwendig, mehrere davon belasten die Treiber über Gebühr!

Besonders wichtig sind die korrekten Werte der Abschlußwiderstände bei modernen Laufwerken, deren Eingangsstufen mit CMOS-ICs bestückt sind, um deren hohe Eingangsimpedanzen zu kompensieren.

12.3. Signaltiming

Dieser besonders kritische Punkt ist in den jeweiligen Datenblättern zum Laufwerk beschrieben. Allgemein ist nur folgendes zu sagen, sofern das richtige Timing nicht einem Floppy-Disk Controller-Formatter überlassen wird:

- Vor einem Lesezugriff muß der Kopf über der richtigen Spur stehen, die richtige Seite der Diskette und das richtige Laufwerk angewählt und der Motor auf Nenndrehzahl sein. Nach dem Steppen muß dem Kopf eine Beruhigungszeit von typisch 30 msec gewährt werden, um Trägheitseffekte abklingen zu lassen.
- Beim Schreiben muß zudem vor Beginn des Datentransfers der Schreibschutz geprüft und das Write-Gate aktiviert werden.
- Vor einem Step-Impuls muß die

The used switching elements in TTL technique, for outputs preferably 7438 and for inputs 74LS04 or 74LS14, point however typical input impedances from approx. 1 kOhm up.

In order now with the high data frequency a signal over or generally after-swing to avoid, must the end of the lead, thus at the last drive, with so-called "Terminator" resistances approximately +5V to be locked. These should indicate as DIP resistances about 150 ohms. Is only a "terminator chip" necessary, several of it load the drivers over fee!

The correct values of the terminal resistances are particularly important with modern drives, whose input stages are equipped with CMOS ICs, in order to compensate their high feed impedances.

Signal timing

This particularly critical point is described in the respective data sheets to the drive. Generally the only following is to be said, if the correct timing will not leave Floppy Diskette Controller Formatter:

- Before a read access the heading must be over the correct track, be the correct side of the diskette and the correct drive selected and the engine on rated speed. After stepping, a damping time must be granted of typically 30 msec, in order to let inertia effects fade away.
- During the writing must besides before beginning of the data transfers the write protection are checked and the Write gate activated.
- Before a step impulse the direction line must

Richtungsleitung definiert und stabil sein, die Pulsdauer und Frequenz der Step-Impulse richtet sich nach dem verwendeten Laufwerk, wobei die Frequenz vom Anwender (Programmierer) festgelegt wird.

Näheres finden Sie in den Kapiteln 16 und 17.

12.4. Signalbeschreibung

Die Signalbeschreibung findet aus der Sicht des Laufwerkes statt. Alle Leitungen sind Low-Aktiv, was bedeutet, daß ihr logischer Zustand 'WAHR' ist wenn ein TTL-Low Signal anliegt.

Die Signale im Einzelnen:

12.4.1. Pins 6,10,12,14, DS0 bis DS3 — DRIVE SELECT

Mit diesen Leitungen wird eines der 4 möglichen Laufwerke direkt aktiviert. Es darf immer nur eine dieser Leitungen aktiv, also Low sein! Je nach Codierung des Laufwerks startet unmittelbar nach dieser Anwahl der Motor und wird der Kopf geladen, sofern ein Head-Load vorgesehen ist. Ggf. muß der Motor über MOTOR ON vor dem Zugriff gestartet werden, um die eine Sekunde für das Hochlaufen auf die Solldrehzahl zu gewährleisten. Hätte man hier eine BCD-Codierte Anwahl vorgesehen, so könnten 15 Laufwerke angeschlossen werden.

12.4.2. Pin 8 INDEX — INDEX PULSE

Zur Regelung der Drehzahl und der Erkennung des Startbereiches einer Spur befindet sich das Indexloch in der Diskette. Damit wird der Strahlengang einer Lichtschranke periodisch geschlossen. Aufgrund der Umdrehungsgeschwindigkeit bei Minidisketten liegt an diesem Kontakt ein TTL-Signal mit einer Frequenz von 5 Hz an.

be defined and stable, the pulse duration and frequency of the step impulses depends on the used drive, whereby the frequency is determined by the user (programmer).

You find details in Chapters 16 and 17.

Description of signal

The description of signal takes place from the view of the drive. All lines are Low active, which means that its logical status "TRUE" is if a TTL Low signal fits.

The signals in detail:

Pin 6, 10, 12, 14, DS0 to DS3 — DRIVE SELECT

Wwith these lines is activated directly one of the 4 possible drives. It may actively, thus be in each case one of these lines Low! Depending upon coding of the drive the engine starts and the heading is loaded immediately after this selection, if a Head load is intended. If necessary the motor over MOTOR ON before the access must be started, in order to ensure the one second for starting on the debit number of revolutions. If one would have designated BCD coding a selection here, then 15 drives could be attached.

Pin 8 INDEX — INDEX PULSE

For the regulation of the number of revolutions and the recognition of the start area of a track is the index hole in the diskette. Thus the path of rays of a light barrier is periodically closed. Due to the rotational speed with mini-diskettes a TTL signal with a frequency of 5 Hz rests against this contact.

12.4.3. Pin 16 — MOTOR ON/MOTOR START

Diese Leitung ermöglicht den Start der Antriebsmotoren aller Laufwerke. Da jedesmal auf das Erreichen der Solldrehzahl gewartet werden muß, kann mit dieser Leitung bereits in Erwartung eines Diskettenzugriffs die Motordrehzahl stabilisiert werden.

Pin 16 — MOTOR ON/MOTOR START

This line enables the start of the drive motors of all drives. Since for achieving the debit number of revolutions must be waited each time, the engine speed can be already stabilized with this line in expectation of a diskette access.

12.4.4. Pin 18 — DIR/STEP-DIRECTION

Ist diese Leitung aktiv, also Low, so bewegt sich der Kopf nach einem Step-Impuls nach außen zum Rand der Diskette. Andernfalls wird auf die Diskettenmitte hin gefahren. Dieses Signal muß eine bestimmte Zeit vor einem Step-Impuls stabil sein.

Pin 18 — DIR/STEP-DIRECTION

This line is, actively thus Low, then the heading induces itself after a step impulse outward to the edge of the diskette. Otherwise on the diskette center one drives. This signal must be stable a certain time before a step impulse.

12.4.5. Pin 20 — STEP/STEP-PULSE

Pro Impuls, dessen Dauer und Frequenz laufwerkspezifisch sind, bewegt sich der Schreib-Lesekopf um den radialen Abstand einer Spur weiter. Richtung durch DIR.

Pin 20 — STEP/STEP-PULSE

The read-write head moves on pro impulse, whose duration and frequency are drive specific, around the radial distance of a track. Direction through DIR.

12.4.6. Pin 22 — WRITE DATA

Diese Leitung führt den seriellen Datenstrom vom Controller zum Laufwerk. Die Frequenz hängt vom Aufzeichnungsformat ab. Im Laufwerk selbst werden diese bereits codierten Rechtecksignale auf der Diskettenoberfläche aufmagnetisiert.

Pin 22 — WRITE DATA

This line leads the serial data stream of the controller to the drive. The frequency depends on the recording format. In the drive these square wave signals on the disk surface, already coded, are up-magnetized.

12.4.7. Pin 24 — WRITE GATE

Hiermit wird von Lesen (Inaktiv, High) auf Schreiben (Aktiv, Low) umgeschaltet. Nach Änderung dieses Pegels muß der Elektronik des Laufwerkes Zeit gegeben werden, alle Einschwingvorgänge zu beenden. Ist die Diskette schreibgeschützt, so ist bei einigen Modellen (nicht allen!) die Schreibelektronik abgeschaltet. Vorher muß also WRTPT geprüft werden.

Pin 24 — WRITE GATE

Hereby is switched by reading (inactive, High) to writing (asset, Low). After modification of this level time must be given to electronics of the drive to terminate all transients on engagement. Is the diskette write protected, then is with some models (not all!) write electronics switched off. Beforehand thus WRTPT must be checked.

12.4.8. Pin 26 — TRK 00/TRACK ZERO

Pin 26 — TRK 00/TRACK ZERO

Ist diese Leitung aktiv, so steht der Kopf über Spur Null, also ganz am äußeren Rand der Diskette. Außer dieser Rückmeldung ist zu keiner Zeit vom Laufwerk zu erfahren, wo der Kopf steht!

Die erste Aufgabe beim erstmaligen Ansprechen eines Laufwerkes ist demnach: DIR auf 'nach außen' setzen und so lange Step-Impulse geben, bis das TRK 00 Signal aktiv ist (hierzu gibt es den FDC-Befehl RESTORE, siehe dort). Passiert nach mehr als 80 (40) Impulsen nichts, dann ist das Laufwerk defekt.

Wurde TRK 00 aktiv, so muß der Rechner nach jedem Step-Impuls ein eigenes Register mitlaufen lassen, um immer zu wissen, wo der Kopf gerade steht. Ein FD-Controller kann dies mit einem 'Verify on track' anhand der Formatierung kontrollieren und bei Bedarf selbsttätig ein Spurregister auf den aktuellen Stand bringen.

12.4.9. Pin 28 — WRTPT/WRITE PROTECT

Hier wird angezeigt, ob eine Diskette schreibgeschützt ist oder nicht (aktiv oder inaktiv). Bei einer schreibgeschützten Diskette, also einer Diskette, bei der die entsprechende Kerbe zugeklebt wurde, ist kein Schreiben möglich, da die Schreibelektronik (des FDC und evtl. des Laufwerks, s.o.) blockiert ist.

12.4.10. Pin 30 — READ DATA

Analog zu Pin 22 liefert hier das Laufwerk die aus den Flußwechseln 'demodulierten' Daten als TTL-Signale zum Controller.

12.4.11. Pin 32 — SIDE SELECT

Wenn aktiv, dann wird mit dieser Leitung die Unterseite der Diskette angewählt. Bei einseitigen Laufwerken ist dieser Pin N.C., ein Pegelwechsel an dieser Leitung bewirkt bei rein einseitigen Laufwerken nichts, bei solchen mit

Is active this line, then the heading is thus completely over track zero, at the outside edge of the diskette. Except this acknowledgment is at no time of the drive to be experienced, where the heading is!

The first function when first responding a drive therefore is: On "outward" set to DIR and so long step impulses give, until the TRK is active 00 signal (for this there is the FDC instruction RESTORE, sees there). Occurred after more than 80 (40) impulses nothing, then the drive is defective.

If TRK 00 became active, then the computer must let its own register run along after each step impulse, in order to always know, where the heading even is. An FD controller can control this with a "Verify on Track" on the basis formatting and bring if necessary automatically a track register on the current status.

Pin 28 — WRTPT/WRITE PROTECT

Is displayed here whether a diskette is write protected or not (actively or inactively). With a write protected diskette, thus a diskette, with which the appropriate notch was sealed, no writing is possible, there write electronics (the FDC and perhaps the drive, see above) is blocked.

Pin 30 — READ DATA

Similar to pin 22 supplies here the drive from the flux changes "demodulated" data as TTL signals to the controller.

Pin 32 — SIDE SELECT

If active, then is selected with this line the lower surface of the diskette. With one-sided drives this pin N.C., a level change at this line is not caused with purely one-sided drives anything, with such with missing second heading (optional) can it be

fehlendem zweiten Kopf (optional) kann es sein, daß bei falscher Seitenwahl keine Daten übertragen werden können.

12.4.12. Pin 34 — READY/DRIVE READY

Hiermit wird angegeben, ob das Laufwerk bereit ist. Diese Funktion ist optional und nicht bei jedem Laufwerk bzw. Controller anzutreffen. Wenn doch, dann wird im Allgemeinen nur angezeigt, daß die Drehzahl erreicht wurde.

12.4.13. Pins 2,4 — SPARE-PINs

Ihre Funktionen sind ebenfalls optional und nicht immer verfügbar. Meist werden diese Signale nur bei Laufwerken für PCs eingesetzt, z.B. für ein Media-Change (Diskettenwechsel), was bei modernen DSRs für den TI auch recht nützlich wäre.

Ein Vielzahl der mit der korrekten Ansteuerung von Laufwerken verbundenen Aufgaben wird vom Floppy-Disk-Controller-Chip übernommen, so daß der Programmierer nicht direkt auf der Hardwareebene arbeiten muß.

12.5. Prozessor-Interface

Das verbindende Element zwischen CPU und Floppy-Disk-Laufwerk stellt ein hochintegrierter Baustein mit Namen Floppy-Disk-Controller/Formatter dar, auch in Kurzform 'FDC' genannt. Wie er sich softwareseitig der CPU präsentiert und im Detail angesprochen wird, erfahren Sie in den Kapiteln 14 bis 16. Die Hardware stellt dieses Kapitel vor.

12.5.1. Vom FDC übernommene Aufgaben

Nicht alle Leitungen, von denen bisher die Rede war, sind der CPU direkt zugänglich; einige davon sind nur über den FDC beeinfluß- bzw. lesbar. Direkt damit hängt es zusammen, wenn der FDC der CPU einige Aufgaben abnimmt bzw. es der CPU unmöglich ist, manche

that with false side selection no data will transfer can.

Pin 34 — READY/DRIVE READY

Hereby is indicated whether the drive is ready. This function is optional and not with each drive or to find controller. If nevertheless, then generally only displayed that the number of revolutions was achieved.

Pins 2,4 — SPARE-PINs

Your functions are likewise optional and not always available. Usually these signals only with drives for PCS begun, e.g. for a Media CHANGE (diskette change), which with modern DSRs for the TI would be also quite useful.

Multiplicity of the functions connected with the correct control of drives is taken over by the Floppy Diskette Controller chip, so that the programmer does not have to operate directly on the hardware level.

Processor interface

The connecting item between CPU and floppy diskette drive represents a highly integrated building block with the name Floppy Disk Controller/Formatter, also in short form "FDC" mentioned. How it presents themselves to the CPU in terms of software and is addressed in the detail, experience in Chapters 14 to 16. The hardware presents this section.

From the FDC taken over functions

All lines, from which so far the speech was, are not directly accessible to the CPU; some of it is influencing or only over the FDC. readably. Thereby it is connected directly, if the FDC of the CPU removes some functions or. it the CPU is impossible to execute some control functions

Steuerfunktionen ohne Mithilfe des FDC auszuführen.

12.5.1.1. Kopfpositionierung

Alle notwendigen Bewegungen des Schreib/Lesekopfes werden vom FDC ausgelöst. Dabei sorgt der FDC für die Einhaltung der Wartezeiten zwischen Schalten des DIR-Pegels und Ausgabe der Step-Impulse. Die Leitungen DIR und STEP sind von der CPU über den FDC nur indirekt beeinflußbar.

12.5.1.2. Datentransfer

Für die CPU erfolgt der Datentransfer vom und zum Laufwerk parallel, jeweils ein Byte (8 Bit) auf einmal. Der FDC liefert und empfängt die Laufwerk-Daten jedoch seriell, noch dazu codiert. Das von der CPU gelieferte Byte wird in ein, der CPU unzugängliches, Schieberegister übernommen (DATA-SHIFT-REGISTER), aus dem die Daten seriell herausgeschoben und zum Laufwerk übertragen werden. Beim Lesen von Diskette werden die seriellen Daten decodiert und im gleichen Schieberegister gesammelt. Ist ein Byte komplett, so wird es in das der CPU zugängliche Datenregister kopiert.

Synchronisation des Datentransfers

Der CPU muß beim Datentransfer, ganz gleich in welche Richtung, mitgeteilt werden, wann das Datenregister neu geladen bzw. wann es gelesen werden kann. Hierzu verfügt der FDC über 3 prinzipiell verschiedeneMöglichkeiten.

1. Er kann der CPU mittels eines Bits im Statusregister anzeigen, wann das Datenregister bereit für einen neuen Zugriff ist (DRQ-Bit). Dies wird im POLLING-MODE ausgewertet. Eine Variante testet per CRU laufend den Pegel am DRQ-Pin (Myarc).

2. Er legt den logischen Zustand dieses Bits an

without assistance of the FDC.

Head positioning

All necessary movements of the write/read head are released by the FDC. The FDC provides for the adherence to the waiting periods between switching the you level and output of the step impulses. Those Lines DIR and STEP are only indirectly influenceable by the CPU over the FDC.

Data transfer

For the CPU the data transfer takes place from and to the drive parallel, in each case a byte (8 bits) at one time. The FDC supplies and receives the drive data however serially, still in addition coded. The byte supplied by the CPU is transferred to, the CPU inaccessible, shift register (DATA SHIFT registers), from which the data will transfer serially shifted out and to the drive. When reading diskette the serial data are decoded and collected in the same shift register. If a byte is complete, then it is copied into that the CPU accessible data pointers.

Synchronization of the data transfers

The CPU must with the data transfer, completely directly in which direction, is indicated, when the data pointer again loaded or. when it can be read. For this the FDC has 3 always different possibilities.

1. It can display to the CPU by means of a bit in the status register, when the data pointer is ready for new access (DRQ bits). This is analyzed in the Polling mode. A version tests the level at the DRQ pin (Myarc) by CRU constantly.

2. It puts the logical status of this bit to a link

einen Anschluß gleichen Namens (DRQ-Pin) und stoppt den Prozessor nach einen FDC-Registerzugriff solange, bis ein Byte gewandelt wurde. Dies ist der WAIT-STATE-MODE.

3. Er gibt über den DRQ-Pin und entsprechende Hardware die Anforderung eines direkten Speicherzugriffs aus. Dies ist der DMA-MODE.

Es sollte angemerkt werden, daß es natürlich nicht reicht, nur DRQ zu testen, wie auch immer das geschehen mag. Natürlich muß auch INTREQ einbezogen werden, da schließlich auch Schreib-/Lesefehler auftreten können, die nicht über DRQ, wohl aber über INTREQ gemeldet werden!

Der Polling-Mode ist nur bei solchen Prozessoren möglich, die innerhalb der sehr kurzen Wandlungszeit eines Bytes mehrere Maschinenbefehle abarbeiten können. Dies ist beim TI aufgrund verschiedener geschwindigkeitsmindernder Hardware-Schaltungen nicht möglich (Myarc probiert es mit mittlerem Erfolg in der CRU-Polling-Variante doch, siehe ROM-Listing im Kapitel ROM-Listing Myarc DDCC-1ab Seite 248).

Beim Status-Register-Polling wirkt sich eine Eigenart der 1773/72/71 aus, die nicht leicht zu beherrschen ist.

Nachdem ein Befehl in das Kommandoregister geschrieben wurde, muß abhängig von der eingeschalteten Dichte eine unterschiedliche Zeit verstreichen, innerhalb der kein Statusregisterzugriff erfolgen darf.

Wird diese Regel mißachtet, so werden einige Bits im Statusregister 'wackelig', so daß es z.B. Checksum- oder RNF-Fehler gibt, die an sich keine sind, aber wer kann das im konkreten Fall entscheiden.

of same name (DRQ pin) and stops after processor a FDC Register access until a byte was changed. This is the WAIT state mode.

3. It outputs the request of a direct memory access over the DRQ pin and appropriate hardware. This is the DMA (direct memory access) mode.

It should be marked the fact that it is not enough naturally to test only DRQ however may occur. Naturally also INTREQ does not have to be included, since finally also read/write errors can occur, those over DRQ, probably however over INTREQ to be announced!

The Polling mode is possible only with such processors, which can process several machine instructions within the very short transformation time of a byte. This is not possible with the TI due to different rate-reducing hardware circuits (Myarc tries it with middle success in the CRU Polling version nevertheless, sees ROM listings in the chapter ROM listing Myarc DDCC-1 on page 248).

With the status register Polling a characteristic of the 1773/72/71 affects itself, which is to be controlled not easily.

After an instruction was written into the command register, dependent a different time must elapse on the switched on density, within which no status register access may take place.

If this rule is ignored, then some bits in the status register become "wobbly," so that there is e.g. checksum or RNF errors, which are actually none, but who can decide in the concrete case.

Bei den Controllern von Atronic, CorComp und TI wird das Wait-State-Verfahren angewendet. Dabei ist es nach Freigabe der entsprechenden Logik möglich, daß der FDC die CPU dann in Wartezyklen zwingt, wenn diese auf das noch nicht bereite Datenregister zugreift. Eine Synchronisation wird dann derart erreicht, daß von Seiten des steuernden Programms gewährleistet ist, daß die CPU nach Abholung bzw. Anlieferung des vorangegangenen Bytes immer wieder vor Ablauf einer Byte-Wandlungszeit auf das Datenregister zugreift. Die Wait-State-Logik wird immer unmittelbar vor Beginn eines Datentransfers aktiviert und sofort nach Bearbeitung des letzten Bytes wieder abgeschaltet.

Das Polling-Verfahren wird trotz aller Risiken beim Myarc-Controller angewandt. Hier werden einige Parameter zur Befehlsart übergeben, die Bearbeitung gestartet und mittels Abfrage von CRU-Bit 0 auf das Ende der Operation gewartet, wobei in der Zwischenzeit die Daten in der angegebenen Richtung übergeben werden. Durch die CRU-Abfrage wird eine höhere Abfragefrequenz erreicht, als dies bei einer Bitmaskenabfrage möglich wäre. Nachteilig ist trotzdem die hohe Anzahl asynchroner Prozessorzyklen im Gegensatz zu einer Synchron-Methode über Hardware-Wait-States!

Datentransferrichtung

Wie bekannt sein wird, findet der Datentransfer zwischen CPU, FDC und Laufwerk generell in 2 verschiedenen Richtungen statt. Es werden entweder Daten geschrieben oder Daten gelesen. Dabei müssen die Laufwerke verschieden angesteuert werden.

Beim Lesen von Daten muß der Schreibverstärker der Laufwerkelektronik ausgeschaltet werden und es müssen die Impulse der Leitung READ-DATA gesammelt, decodiert und in Parallel-Form gewandelt werden. All dies

With controllers from Atronic, CorComp and TI the WAIT State procedure one applies. It is possible after release of the appropriate logic that the FDC forces the CPU in wait states if this does not access that yet ready data pointers. Synchronisation is then achieved in such a manner that it is ensured on the part of the controlling program that the CPU after collection or Delivery of the preceding byte before flow of a byte transformation time on the data pointer accesses again and again. The WAIT State logic is activated always directly before beginning of a data transfers and switched off immediately after handling of the last byte again.

The Polling procedure is applied despite all risks with the Myarc controller. Here some parameters are transferred to the instruction type, the handling and waited by means of query of CRU bit 0 is started for the end of the operation, whereby in the meantime the data are transferred in the indicated direction. A higher interrogation frequency is achieved by the CRU query, than this would be possible with a filter query. Unfavorable nevertheless the high number of asynchronous processor cycles is contrary to a synchronous method over hardware WAIT States!

Data transfer direction

As admits will be, takes place the data transfer between CPU, FDC and drive generally in 2 different directions. Either data are written or read data. The drives must be headed for differently.

When reading data the write amplifier of drive electronics must to be switched off and it have the impulses of the line READ DATA to be collected, decoded and in parallel form changed. All this completes the FDC.

erledigt der FDC.

Prinzipiell genauso ist der Vorgang beim Schreiben auf Diskette. Hier muß die Schreibelektronik aktiviert werden (WRITE GATE) und der Schreibschutz geprüft werden. Bevor ein Schreibbefehl ausgeführt wird, wird der Pegel der Leitung WRTPT vom FDC geprüft. Ist der Schreibschutz aktiviert, dann wird jeder Schreibbefehl abgebrochen. Wird WRTPT während eines laufenden Schreibbefehls aktiv, so wird nicht abgebrochen, sondern der Sektor/die Spur fertig geschrieben.

12.5.1.3. Statusmeldungen des Laufwerks

Folgende Signalleitungen des Laufwerks kann die CPU über das Status-Register des FDC abfragen:

READY, TRK 00, WRTPT, MOTOR ON, HEAD LOADED, INDEX.

Dabei ist zu beachten, daß die Signale bereits mit positiver Logik vorliegen, also vom Laufwerk kommend bereits invertiert wurden. Nicht alle FDC-Typen lassen die Abfrage aller aufgeführten Signale zu. Dies wird bei der Besprechung der Status-Register-Zustände in Kapitel 16 genau beschrieben.

12.5.2. Von der CPU auszuführende Arbeiten

Der FDC arbeitet immer nur auf dem Laufwerk, der Spur und der Seite der Diskette, wie es die CPU vorgab. Das bedeutet, daß die CPU neben den eigentlichen Aufgaben beim Datentransfer und der Erteilung von Befehlen an den FDC noch dafür sorgen muß, daß immer das korrekte Laufwerk zugeschaltet ist, der Schreib/Lesekopf korrekt positioniert ist, der passende Kopf (bei zweiseitigen Laufwerken) geschaltet ist und die korrekte Speicherdichte gewählt wurde. Erst nach Abschluß dieser Einstellungen kann mit einem Befehl an den FDC mit einem Datentransfer begonnen werden.

Exactly the same in principle the procedure is during the writing on disk. Here write electronics must be activated (WRITE GATE) and the writing protection to be examined. Before a write instruction is implemented, the level of the line WRTPT of the FDC is examined. If the writing protection is activated, then each write instruction is broken off. If WRTPT becomes active during a current write instruction, then, but is not finished written sector/the track is not broken off.

Status messages of the drive assembly

The CPU can test the following signal lines of the drive over the status register of the FDC:

READY, TRK 00, WRTPT, MOTOR ON, HEAD LOADED, INDEX.

It is to be considered that the signals are already present with positive logic, thus by the drive coming was already inverted. Not all FDC types permit the query of all specified signals. This is described exactly with the discussion of the status register statuses in Chapter 16.

Of the CPU work which can be required

The FDC operates in each case on the drive, the track and the side of the diskette, as it gave the CPU. That means it that the CPU apart from the actual functions with the data transfer and the distribution of instruction must still provide to the FDC for it that the correct drive is connected always, the write/read head is correctly positioned, which is switched suitable heading (with bilateral drives) and which was selected correct memory density. Only after termination of these adjustments can be begun with an instruction to the FDC with a data transfer.

Die CPU kann dabei direkt die folgenden Leitungen beeinflussen:

DRIVE SELECT 0 bis 3, MOTOR ON, SIDE SELECT, HEAD LOAD (IN USE)

Zur programmtechnischen Manipulation dieser Leitungen sei hier auf Kapitel 16ff verwiesen.

12.6. Die verschiedenen FDC-Typen in TI-Diskcontrollern

Dieser Abschnitt soll einige Informationen zu den verschiedenen FDCs geben und Unterschiede in deren Interfacing zu CPU und Laufwerk zeigen.

In allen TI-Diskcontrollern, ob nun dem Original von TI, den CorComp-Typen, Atronic-Controllern (Box oder Stand-Alone), CPS99 oder Myarc, werden FDCs der Firma Western Digital eingesetzt. Folgende Typen kommen zum Einsatz:

The CPU can influence thereby directly the following lines:

DRIVE SELECT 0 to 3, MOTOR ON, SIDE SELECT, HEAD LOAD (IN USE)

For the manipulation by programming of these lines is here referred to Chapter 16.

The different FDC types of TI disk controllers

This paragraph is to give some information to the different FDCs and differences in their interfacing to CPU and drive to point.

In all TI disk controllers whether now the original of TI, the CorComp types, Atronic controller (box or status alone), CPS99 or Myarc, it becomes FDCs of the company Western digital assigned. The following types are used:

Typ-Bezeichnung	*Einsatz in*	*Besonderheiten*	*Special features*
WD 1771-01	TI-Controller	nur einfache Dichte, eigenes Schrittmotor-Interface, eigene Head-Load Steuerung, eingebauter Daten-Separator, Schreibstromreduktionssignal, 2 MHz Takt, invertierter Datenbus	Only single density, own stepping motor interface, own head load control, inserted data separator, write current reduction signal, 2 MHz clock, inverted data bus
WD 2793-02	CorComp (alt)	SD und DD möglich, eigene Head-Load Steuerung, PLL als Daten-Separator, Schreibstromreduktionssignal, 1 oder 2 MHz Takt, extern einstellbare Schreib-Vorkompensation	SD and DD possible, own head load control, PLL as data separator, write current reduction signal, 1 or 2 MHz clock, externally adjustable write before compensation

Typ-Bezeichnung	*Einsatz in*	*Besonderheiten*	*Special features*
WD 1772	Myarc	SD und DD möglich, Gehäuse nur 28 polig, digitaler Daten-Separator, Schreib-Vorkompensation per Software schaltbar, Hohe Step-Raten, 8 MHz Takt	SD and DD possible, housing only 28 polig, digital data separator, write before compensation by software adjustably, high step rates, 8 MHz clock
WD 1773	CorComp (neu)	SD und DD möglich, Gehäuse Atronic (alle nur 28 polig, digitaler Typen) Daten-Separator, Schreib-Vorkompensation per Hardware schaltbar, softwarekompatibel zu 2793, 8 MHz Takt	SD and DD possible, housing Atronic (everything only 28 polig, digitaler types) data separator, write before compensation by hardware adjustably, software-compatibly to 2793, 8 MHz clock

Alle Unterschiede zwischen den Controller-Chips betreffen nur die Laufwerkseite. Auf der CPU-Seite sind die Unterschiede gering. Diese Leitungen auf CPU-Seite sind:

All differences between the controller chips concern only the drive page. On the CPU page the differences are small. These lines on CPU page are:

/CS	Chip-Select Decodiert die 5 Register des Chips im Adreßraum der CPU	Chip Select Decodes the 5 registers of the chip in the address area of the CPU R/W
R/W	Read/Write Schreiben bzw. Lesen der Register	Read/Write Writing or reading of registers
A0	Address Bit 0 Adressleitung für Registerauswahl	Address bit 0 Address line for register selection
A1	Address Bit 1 dto.	Address Bit 1 Same as Address Bit 0
DAL	0 - 7 Data Lines Datenleitungen für Datenaustausch mit CPU	0-7 Data Lines Data lines for data exchange with CPU
/MR	Master Reset Hardware-Reset, softwareseitig über einen Interrupt realisierbar	Master Reset Hardware reset, in terms of software over an interrupt realizable
/DDEN	Double-Density Speicherdichte umschalten Enable	Double Density Memory density switch enable

DRQ	Data Request Zeigt an, daß auf das Datenregister zugegriffen werden muß/kann	Data Request Display the fact that the data pointer can be accessed
INTRQ	Interrupt Request Gibt nach Befehlsausführung diese an und zeigt an, daß Daten im Statusregister verfügbar sind	Interrupt Request Indicates after execution this and displays that data are available in the status register/incoming goods
/WE	nur 1771 und 2793 Indiziert einen Schreibzugriff der CPU. Kann direkt an R/W ange schlossen werden	Only 1771 and 2793 Indicates a write access of the CPU. Knows directly at R/W ange closed to become
/RE	nur 1771 und 2793 Indiziert einen Lesezugriff der CPU. Kann über einen Inverter an R/W angeschlossen werden	Only 1771 and 2793 Indicates a read access of the CPU can over an inverter to R/W be attached

Die FDCs der Typen 1772 und 1773 stellen die modernsten Typen dar, die im TI eingesetzt werden. Sie haben nicht mehr das 40-polige Gehäuse und benötigen nicht mehr so viele externe Bauteile. Besonders kritisch ist der WD 2793-02, der die Einstellung von Schreib-Vorkompensation (Write Precompensation) und Leseimpulsbreite über externe Potentiometer erfordert. Diese Serie wurde von CorComp kurz nach Erscheinen der ersten Exemplare, die fast alle nicht korrekt arbeiteten, eingestellt.

Erkennbar sind diese Karten am beigen Gehäuse und der alten CorComp-Adresse in Anaheim. Leider hat aber auch der 1773 ein Problem: Vermutlich aufgrund eines Maskenfehlers, liest er bei FM-codierten Signalen die Adreßfelder gerne mit einem CRC-Error, obwohl es keine gibt. Diese Eigenart sollten alle Programme beachten, die mit einem 1773 Adreßfelder in FM lesen müssen!

The FDCs of the types can be attached 1772 and 1773 to place the most modern types over an inverter to R/W, which are used in the TI. It do not have no more the 40-polige housing and to need no longer so many external components. Particularly the WD 2793-02, which requires the adjustment of write before compensation (Write Pre-compensation, is critical) and read pulse width over external potentiometers. This series was stopped briefly by CorComp after appearance of the first copies, which did not operate almost all correctly.

These cards at the beige housing and the old CorComp address are recognizable in Anaheim. In addition, unfortunately the 1773 a problem has: Probably due to a mask error, he reads the address fields with FM-coded signals gladly with an CRC error, although there are none. All programs should consider this characteristic, which must read with 1773 address fields in FM!

13. Aufzeichnungsverfahren

Recording methods

Wollte man sich grundsätzlich zu Datenaufzeichnungsverfahren auslassen, so könnte hier eine neue Ausarbeitung begonnen werden.

If one wanted to omit oneself basically to data recording procedures, then a new elaboration could be begun here.

Gemeinsam ist allen Verfahren jedoch, daß mit ihrer Hilfe die binärenInformationen des Rechners derart umgewandelt werden, daß sie den derzeit verwendeten analogen Speichermedien optimal angepasst sind. Ziele sind dabei hohe Datensicherheit, hohe Transfergeschwindigkeiten und Portabilität, was im konkreten Fall mit Austauschbarkeit übersetzt werden kann.

Common it is all procedures however that with their assistance the binary information of the computer is converted in such a manner that they are optimally adapted to the similar storage media used at present. Targets are thereby high data security, high data transfer rates and portability, which can be translated in the concrete case with exchangeability.

Bei der Datenaufzeichnung dominieren 2 Verfahren. Systeme mit Trägerschwingung (Cassettenrecorder, auch Standard-DFÜ) und getaktete Systeme. Beide arbeiten bitseriell.

During the data recording 2 procedures dominate. Systems with frequency wave (cassette recorder, also standard data-communications) and clocked systems. Both use serial bits.

Die geträgerten Systeme basieren auf einer Grundfrequenz, dem sog. Carrier, und werten Frequenzabweichungen in die eine oder andere Richtung als die Übertragung einer Null oder Eins. Dabei wird im Allgemeinen nach jeder Datenzelle (üblicherweise 1 Byte), synchronisiert. Schlagworte hierzu sind: FSK (Frequency-Shift-Keying, Frequenzumtastung), Manchester-Verfahren und Turbo- bzw Hypertape.

The carrier systems are based on a basic frequency, which so-called carrier, and frequency departures in or other direction than the transmission of a zero or a unity. Generally after each data cell (usually 1 byte), one synchronizes. Key words for this are: FSK (frequency shift keying, frequency shift keying), Manchester procedure and turbo or Hypertape.

Als Nachteile sind zu nennen: die geringe Transfergeschwindigkeit der klassischen Systeme (Einige Derivate werten keine Momentanfrequenzen mehr aus und zählen Perioden unterschiedlicher Dauer, was eine starke Geschwindigkeitserhöhung bewirkt), sowie die starke Sensibilität gegenüber Bandgeschwindigkeitsschwankungen bzw. Frequenzverwerfungen. Vorteile der Standardmethode mit analoger PLL ist die fast vollständige Unempfindlichkeit gegenüber Pegelschwankungen.

As disadvantages are to be called: the small data transfer rate of the classical systems (some derivatives analysz no more inside frequencies out and count periods different duration, what causes a strong speed increase), as well as strong sensitivity in relation to tape rate fluctuations or. frequency rejection. Advantages of the standard method with similar PLL is the almost complete insensitivity in relation to fluctuations in level.

Diese Systeme erreichen ihre physikalischen

These systems achieve their physical boundaries

Grenzen jedoch sehr schnell, da es eine Relation zwischen Trägerfrequenz und maximaler Modulationsfrequenz gibt. Bei den klassischen Verfahren mit Frequenzmodulation ist die maximale Datenrate etwa ein Zehntel, bei den Pulsdauerverfahren etwa ein Drittel der Carrierfrequenz. Eine Erhöhung der maximalen Datenrate durch Erhöhen der Trägerfrequenz scheitert an dem begrenzten Frequenzgang billiger Recorder und den mit zunehmender Annäherung an die Grenzfrequenz steigenden Phasenfehlern.

Die getakteten Systeme trifft man bei Floppy-Disk Systemen an. Hierbei werden die Datenbits des Rechners mit einem festgelegten Takt als Magnetisierungsimpulse zusammen mit den Taktsignalen auf dem Medium abgelegt. Es wird also keine eigentliche Trägerschwingung benutzt, der die Daten aufmoduliert werden. Dadurch wird die Transfergeschwindigkeit bis auf den physikalischen Grenzwert erhöht. Zudem wird quasi bei jedem gelesenen Bit eine Information an die Taktrückgewinnung gegeben, wodurch die Datensicherheit steigt.

13.1. Randbedingungen

Unterteilt man die Oberfläche einer Diskette in Sektoren, um die Daten in kleinen Portionen besser bearbeiten zu können, so stellt sich sofort die Frage "Wie findet der Rechner die Sektoren wieder?". Gleich zu Anfang dieses Kapitels soll festgestellt werden, daß diese Aufgabe nicht eigentlich dem Rechner angelastet wird, sondern von dem 'Floppy-Disk-Controller-Formatter Chip'-übernommen wird (FDC). Diesem, mit beschränkter Intelligenz ausgestatteten Chip teilt der Rechner mit, welchen Sektor auf welcher Spur er gerne lesen würde. Der FDC sucht nun innerhalb der Spur, über der der Kopf der Diskettenstation positioniert wurde, nach diesem Sektor.

Wie bereits im Kapitel 4 beschrieben, befinden

however very fast, since there is a relation between carrier frequency and max. modulating frequency. With the classical procedures with frequency modulation the max. data rate is for instance a tenth, with the pulse continuous procedures for instance a third of the carrier frequency. An increase of the max. data rate by increasing the carrier frequency fails because of the limited frequency response of cheap recorders and the phase errors rising with increasing approximation to the critical frequency.

One finds the clocked systems with floppy disk systems. Here the data bits of the computer with a fixed clock are put down as magnetization impulses as well as the clock pulses on the medium. No actual frequency wave is thus used, which the data are up-modulated. Thus the data transfer rate up to the physical limit value is increased. Besides quasi with each read bit information is given to the clock recuperation, whereby data security rises.

Boundary conditions

If one divides the surface of a diskette into sectors, in order to be able to process the data in small portions better, then immediately the question arises "as regains the computer the sectors?" Equally at the beginning of this chapter it was stated that this function is not actually charged to the computer, but is taken over by the "Floppy Disk Controller Formatter chip" (FDC). The computer indicates to this chip equipped with limited intelligence, which sector on which track it would read gladly. The FDC looks now within the track, over which the heading of the floppy station was positioned, for this sector.

How already described in Chapter 4, is for this

sich hierzu einige zusätzliche Informationen auf der Diskette, die sog. Adreßfelder. All das wurde also bereits besprochen.

13.1.1. Was kann aber alles passieren, wenn ein Sektor gesucht wird?

In dem Moment, wenn der Schreib/Lese-Kopf sich auf die Diskettenoberfläche senkt, kann ja nicht angenommen werden, daß sich die Diskette an einem ganz bestimmten Ort befindet, ja im Allgemeinen wird der Kopf sicher mitten in einem Datensektor landen. Es müssen sich also zwischen den einzelnen Datensektoren Lücken befinden, anhand derer erkannt wird (vom FDC), wo ein Sektor beginnt bzw. endet. Diese Lücken werden dann auch dazu verwendet, den Taktgenerator der Leseelektronik auf den aufgezeichneten Schreibtakt zu synchronisieren. Zudem werden besondere Bytes auf die Diskette geschrieben, bei denen 2 Taktimpulse in der Mitte des Bytes fehlen. Durch diese 'Law-Violation' (Regelverletzung) erkennt der FDC ein ID- oder Adreßfeld. Damit synchronisiert sich der FDC auf den Start der jeweiligen Datenbytes ein.

Dieses relativ komplizierte Verfahren entstand erst einige Zeit nach der umfassenden Einführung von Disketten (damals aufgrund technologischer Probleme in der Hauptsache 8 Zoll), weshalb man sich anders behalf. Um den Start eines Sektors zu finden, wurden so viele Indexlöcher in die Diskette gestanzt, wie diese Sektoren in einer Spur besaß. Um nun den Sektor am Anfang einer Spur zu kennzeichnen wurde ein zusätzliches Index-Loch gestanzt, welches nicht in die Reihenfolge passte. Nun wurde ein Hardwarezähler von dem durch die Indexlöcher erzeugten Takt permanent weitergeschaltet und bei Erkennen der Taktverletzung durch das Zusatzloch auf Null gesetzt. Da der Rechner den Zähler auslesen konnte (resp. der FDC), war jederzeit klar, wo sich die Diskette befand. Damit waren zwar die

some additional information on the diskette, which so-called address fields. All that was thus already discussed.

What can occur however everything, if a sector is looked up?

In the moment, if the write/read head lowers itself on the disk surface, that the diskette at a completely determined place is, generally the heading cannot be assumed in a data sector will surely in the middle land. Thus gaps must be, on the basis those are detected (of the FDC) between the individual data sectors, where a sector begins or. ends. These gaps are used then also to synchronize the clock generator of read electronics on the noted writing act. Besides special bytes are written on the diskette, with which 2 clock pulses in the center of the byte are missing. By this "Law Violation" (violation of rules) the FDC detects an ID or an address field. With it the FDC synchronizes on the start of the respective data bytes.

This relatively complicated procedure developed only some time after the comprehensive introduction of disks (at that time due to technological problems in the main thing 8 inch), why one managed differently. In order to find the start of a sector, so many index holes were punched into the disk, as possessed these sectors in a track. Over now the sector in the beginning of a track to mark an additional index hole was punched, which did not fit into the order. Now a hardware counter was permanently advanced by the clock produced by the index holes and set when recognizing the clock injury by the auxiliary hole to zero. Since the computer could select the counter (respectable the FDC), was at any time clearly, where the disk was. Thus the sector numbers were fixed, which data density could do by nearly

Sektorzahlen festgelegt, die Datendichte konnte aber so um fast ein Drittel erhöht werden. Dieses Verfahren nennt man 'Hard-Sektorierung'; man findet es im Allgemeinen nur bei 8 Zoll Disketten.

a third is however in such a way increased. One calls this procedure "hard sectoring"; one finds it generally only with 8-inch disks.

Doch nun wieder zu den 'Soft-Sektorierten' Disketten.

But now again to the "soft-sectored ones" disks.

Würde man auf die Lücken verzichten, so entstünden neben einer fehlenden Synchronisation auch Probleme beim Schreiben von Sektoren. Da die Umdrehungsgeschwindigkeit des Antriebsmotors nicht immer konstant ist, würde z.B. ein Sektor, der in einer Phase geringer Geschwindigkeit auf kleinem Raum geschrieben wurde, beim Rückschreiben unter höherer Umdrehungsgeschwindigkeit die nachfolgenden Daten zerstören. Die Lücken dienen also auch dem Auffangen von Drehzahlschwankungen.

If one would do without the gaps, then also problems would develop when writing sectors apart from a missing synchronisation. Since the rotational speed of the driving motor is not always constant, e.g. a sector, which was written in a phase of low speed on small space, would destroy the following data when rewriting under higher rotational speed. The gaps serve thus also catching speed fluctuations.

Wurde nun mittels der Vorgabe vom Rechner der passende Sektor gefunden, so muß dieser immer noch gelesen bzw. geschrieben werden. Auch dabei können Fehler auftreten, entweder durch einen kurzzeitigen Einbruch der Umdrehungszahl oder durch einen Drop-Out, der z.B. durch ein Staubkorn verursacht wurde. Es ist also nötig, auf irgendeine Art zu prüfen, ob die Daten richtig gelesen wurden. Dies geschieht, wie bereits zuvor angesprochen, mithilfe der CRC-Bytes, die jedes Adressfeld und jeden Datensektor kontrollieren. Schreib- und Lesefehler müssen vom DOS einkalkuliert werden, üblich sind 5 bis 10 Versuche die fehlschlagen müssen, ehe endgültig eine Fehlermeldung gegeben wird.

Now by means of the specification by the computer if the suitable sector was found, then this must still read or. are written. Also with it error can occur, either by a brief failure of the number of revolutions or by a drop out, which was caused e.g. by a dust grain. It is thus necessary to check in some way whether the data were read correctly This occurs, as already before addressed, assistance of the CRC bytes, which control each address field and each data sector. Writing and read errors must to be taken into account by the DOS, usually are 5 to 10 attempts those to fail have, before an error message is finally given.

Um einen Sektor zu finden, kann es dann notwendig sein, eine gesamte Spur abzusuchen, wenn nämlich mitten in dem gesuchten Sektor der Kopf aufgesetzt hat - der Erwartungswert liegt jedoch bei einer halben Umdrehung, die beim Zugriff auf einen einzelnen Sektor

In order to find a sector, it can be necessary then to search an entire track if in the looked up sector the heading put in the middle — the expectancy value is however with a half revolution, which must be waited for with the access to an individual sector

abgewartet werden muß.

13.2. Realisierung

Im Zusammenhang mit der Aufzeichnung der Daten auf Floppy-Disks fällt des öfteren das Wort 'Frequenzmodulation', oft auch als FM abgekürzt. Bei der moderneren Aufzeichnung mit doppelter Dichte taucht dann MFM auf, was soviel wie 'Modifizierte Frequenzmodulation' bedeutet. Gemeinsam ist beiden Verfahren, daß sie nichts mit der tatsächlichen Frequenzmodulation zu tun haben, sondern lediglich der Effekt unterschiedlicher Modulationsfrequenzen bei 0 und 1 diese Namensgebung be wirkte. Wieso es nun dazu kam, soll im folgenden Abschnitt erklärt werden.

Betrachtet man die Übertragungs-geschwindigkeit von 250000 bzw. 500000 Bits pro Sekunde, so wird klar, daß eine klassische FM eine Trägerfrequenz von mindestens 2,5 MHz benötigt, wenn der Modulationsindex unter 10 bleiben soll. Eine so hohe Mittenfrequenz verteuert jedoch die ganze Schreib-Leseelektronik, ganz abgesehen von den extremen Anforderungen an das Disketten-material.

Bei der Aufzeichnung auf der Floppy-Disk werden lediglich rein digital bestimmte Bereiche durchmagnetisiert. Es existieren also nur Flußwechsel mit positiver oder negativer Flanke, wobei diese Flußwechsel so gesteuert werden, daß im Ausgangssignal kein Gleichspannungs-anteil auftritt, der bei dieser Modulationsart ohnehin nicht übertragen werden kann.

Daneben ist es notwendig, neben den reinen Daten auch den bei der Aufzeichnung verwendeten Datentakt mit abzuspeichern. Es kann nämlich nicht davon ausgegangen werden, daß der Takt, mit dem geschrieben wurde, auch beim Lesen vorliegt, ganz besonders nicht bei Fremddisketten. Wollte man davon ausgehen, so

Implementation

In connection with the recording of the data on floppy diskettes the more frequent word "frequency modulation" falls, often also than FM abbreviated. During the more modern recording with double density then MFM emerges, which means as much as "modified frequency modulation." Both procedures are common that they do not have to do anything with the actual frequency modulation, but only the effect of different modulating frequencies with 0 and 1 this naming worked. Why it came now to it, was to be explained in the following paragraph.

Regards one the transmission speed of 250,000 or. 500,000 bits per second, then it becomes clear that a classical FM needs a carrier frequency of at least 2.5 MHz, if the modulation index under 10 is to remain. A so high center frequency raises the price of however whole write reading electronics, completely apart from the extreme request to the diskette material.

During the recording on the floppy diskette only purely digitally determined areas are through-magnetized. Thus only river change with positive or negative flank exists, whereby these river changes are controlled in such a way that in the output signal no DC voltage proportion occurs, which will not transfer with this type of modulation anyway can.

It is necessary to store beside the pure data also during the recording used the data clock with. It can not assumed that that the clock, with which was written is present also during the reading, completely particularly not with foreign diskettes. If one wanted to assume that, then all floppy diskette drive would have to be equipped

müssten wegen der hohen Datenrate alle Floppy-Disk-Drive Antriebsmotoren mit einer quarzstabilisierten Regelung ausgestattet sein! Diese nebenbei auch teure Lösung kann durch die Aufzeichnung des Taktes umgangen werden.

Dabei synchronisiert die PLL des FDC auf die aufgezeichnete Datenrate und stellt somit sicher, daß alle Bytes richtig 'rüberkommen'.

Zudem muß noch unterschieden werden zwischen den Daten, die vom FDC kommen und den Daten, die die Elektronik des Laufwerkes dann tatsächlich auf die Diskette bringt.

Aus den Daten, die der Rechner parallel an den FDC liefert, wird ein serieller Datenstrom, der aus Taktsignalen und dazwischengesteckten Datenimpulsen besteht. Bei FM liegt am Ausgang des FDC ein 250 kHz Signal (Frequenz im Mittel!) an, das den Takt repräsentiert. Soll nun eine 1 geschrieben werden, so liegt zwischen zwei Taktimpulsen noch ein Impuls, bei einer 0 jedoch keiner. Die Elektronik im Laufwerk produziert nun bei jedem Impuls, ob Takt oder Daten, einen Flußwechsel in der Magnetschicht der Diskette, wobei die Richtung alternierend ist (s.o.). Beim Lesen von Diskette wird dann wieder bei jedem Flußwechsel ein Impuls an den FDC gesendet, der den Takt abtrennt und die Daten aus den Lücken holt.

Nun wird auch klar, warum man dieses Verfahren FM nennt. Bei einer dauernden Aufzeichnung von lauter Nullbits wird ein Signal mit der Taktfrequenz auf die Diskette gebracht, bei einer 1-Folge liegt die doppelte Frequenz vor.

Damit ist aber auch der Nachteil dieses Systems deutlich: es wird viel zu viel Energie auf die Speicherung des Taktes verwendet. Auch stellt das Frequenzverhältnis 1:2 recht hohe Anforderungen an die Taktrückgewinnung, wodurch die maximale Bitrate reduziert wird.

with a quartz-stabilized regulation driving motors because of the high data rate! This besides also expensive solution can be gone around by the recording of the clock.

The PLL of the FDC synchronizes on the recorded data rate and guarantees thus that all bytes "come across" correctly.

Besides between the data, those must be still differentiated from the FDC be come and the data, which electronics of the drive actually brings then on the diskette

From the data, which the computer supplies parallel to the FDC, becomes a serial data stream, which consists of clock pulses and in between-put data impulses. With FM is because of the output of the FDC 250 kHz a signal (frequency on the average!) on, represents the clock. Now if a 1 is to be written, then still another impulse is, with 0 however none between two clock pulses. Electronics in the drive produces now with each impulse whether clock or data, a river change in the magnetic stratum of the diskette, where with the direction is alternating (see above). When reading diskette again with each river change an impulse is then transmitted to the FDC, which separates the clock and which gets data from the gaps.

Now it becomes also clear why one calls this procedure of FM. During a continuing recording of loud zero-bits a signal is brought with the clock frequency on the diskette, with a 1 following is situated the double frequency forwards.

In addition, thus the disadvantage of this system is clear: much too much energy is used on the storage of the clock. Also the frequency relation 1:2 makes quite high demands against the clock recuperation, whereby the max. bit rate is reduced.

Abhilfe schafft hier das MFM-Verfahren. Hierbei lag die Überlegung zugrunde, daß es unnötig sei, bei einer 1-Folge auch noch den Takt zu senden, wo ja ohnehin die Taktfrequenz auch dann auftaucht, wenn man ihn weglässt. Probleme geben nur Nullfolgen, bei denen dann nach einer gewissen Anzahl, die der PLL bzw. dem Datenseparator maximal zuzumuten ist, von der Datenübermittlung auf die Taktfrequenz umgeschaltet wird.

Remedy creates here the MFM procedure. Here the consideration was the basis that it was unnecessary to transmit with a 1-Folge also still the clock where the clock frequency emerges anyway also if one omits it. Problems give only zero-sequences, with those then after a certain amount of, those the PLL or. the data separator max. to attempting too much is, by the data communication to the clock frequency one switches.

Nun wird also nicht bei jedem Takt- und Datensignal ein Impuls an das Laufwerk geschickt, sondern immer nur dann, wenn ein Datenbit logisch 1 war. Erscheint eine zu lange Folge von Datenbits logisch 0, dann wird, wenn sich die Daten nicht ändern, nun auf die Ausgabe des Taktes übergegangen, bis wieder eine Datenänderung eintritt.

Now thus not with each clock and data signal an impulse is sent to the drive, but only if a data bit were logic 1. If a too long consequence of data bits appears logic 0, then, if the data do not change, now to the output of the clock one changes over, until again a data change occurs.

Diese gedächtnisbehafteten Codes, deren Entstehung fast ausschließlich mit sehr viel, mit Formeln gefülltem Papier verbunden ist, sind im Detail nur etwas für Signaltheoretiker. Hier soll es reichen, wenn im Endeffekt bei diesem Verfahren (MFM) bei einer doppelt so hohen Datenrate die Flußwechselfrequenz genau die gleiche ist wie bei FM. Ohne eine höhere Anforderung an das Diskettenmaterial kann also die doppelte Datenmenge darauf gespeichert werden.

These memory-afflicted codes, whose emergence is connected to filled paper almost exclusively with very much, with formulas, are in the detail only something for signal theoreticians. Here it is to be enough, if in the final result is the same exactly with this procedure (MFM) with a twice as high data rate the flux change frequency as with FM. Without a higher request of the diskette material thus the double quantity of data can be stored on it.

Aufgrund des fehlenden Taktes existiert jedoch nun kein 'Grundton´ mehr, auf den der Schreib/Leseverstärker einschwingen könnte. Die Anforderungen besonders an die Phasentreue sind bei MFM, also doppelter Dichte, demnach höher. Und so kommt es, daß manche Laufwerke, die für einfache Dichte konstruiert sind, bei doppelter Dichte noch mitmachen, andere eben nicht. Dies ist hauptsächlich Material- und Abgleichtoleranzen zuzuschreiben.

Due to the missing clock however now no more "basic tone" exists, on which the write/reading amplifier could in-swing. The request of the phase loyalty are particularly higher with MFM, thus double density, therefore. And in such a way it that some drives, which are designed for simple density still go through in double density, does not come others evenly. This is to be charged mainly material and alignment tolerances.

Abwärtskompatibel sind jedoch alle Laufwerke.

Downward compatible are however all drives. A

Ein für doppelte Dichte konstruiertes LW kann auch Disketten in einfacher Dichte verarbeiten, sprich lesen und schreiben.

13.3. FM und MFM-Codierung

Bei beiden Verfahren existieren sogenannte 'Bit-Zellen', in denen jeweils Datenbits übertragen werden. Die Bit-Zellen existieren jedoch nicht als magnetische Informationen auf der Diskette, sondern sind über Zeitbeziehungen definiert. Zwischen zwei Bit-Zellen treten nur Taktimpulse auf, in den Zellen können Datenbits auftreten, je nachdem, ob ein 1-Bit oder ein 0-Bit übertragen wurde.

Die einzelnen Datenbits werden aus dem im Datenregister abgelegten Wert, der in das Data-Shift-Register kopiert wird, durch Links-Schieben erzeugt. Das herausgeschobene Bit wird nach der Codierung an das Laufwerk gesendet. Beim Lesen wie beim Schreiben wird also mit dem höchtwertigen Bit begonnen.

13.3.1. FM — Single Density

Die Länge einer Bit-Zelle beträgt bei FM 4 Mikrosekunden. Daraus ergibt sich eine Wandlungszeit für 8 Bit von 64 Mikrosekunden. Dies ist die Zeit, welche der CPU zum Verarbeiten eines vom FDC gelieferten bzw. eines an den FDC zu liefernden Bytes bleibt. Diese Zeit kann im schlimmsten Fall (Taktfrequenz am oberen Limit, Drehzahlabweichung Schreiben zu Lesen maximal) auf 55 bzw. 47 Mikrosekunden beim Lesen bzw. Schreiben reduziert werden. Mit der nominalen Bit-Zelle ergibt sich eine Transfergeschwindigkeit der Daten von 250000 Bits pro Sekunde.

Am Anfang einer jeden Bit-Zelle wird ein Takt-Impuls (Clock) gesendet, innerhalb der Bit-Zelle, also zwischen 2 Clock-Impulsen, wird je nach Datenbit ein Datenimpuls (Bit = 1) oder keiner

drive designed for double density can process also diskettes in single density, speaks reads and to write.

FM and MFM coding

With both procedures exist so-called "bit cells," in which data bits will transfer in each case. Those bit cells exist however not as magnetic information on the diskette, but are defined over time relations. Between two bit cells only clock pulse occurs, into which cells can occur to data bits, according to whether a 1-Bit or a 0-Bit became to transfer.

The individual data bits are produced from the value, which is copied into the Data Shift Register, stored in the data pointer, by left shifting. The shifted out bit is transmitted to the coding to the drive. During the reading as during the writing thus with the highest valued bit one begins.

FM — Single Density

The length of a bit cell amounts to with FM 4 microseconds. From it a transformation time for 8 bits of 64 microseconds results. This is the time, which the CPU for processing of the FDC a supplied or. one to the FDC to supplying bytes remains. This time can in the worst case (clock frequency at the upper limit, number of revolutions deviation writing to reading max.) on 55 or 47 microseconds during the reading or writing to be reduced. With the nominal bit cell a data transfer rate of the data of 250,000 bits per second results.

At the start of each bit cell a clock pulse (Clock) is transmitted, inserted within the bit cell, thus between 2 Clock impulses, depending upon data bit a data impulse (bit = 1) or none (bit = 0).

(Bit = 0) eingefügt.

13.3.2. MFM — Double Density

Die Länge einer Bit-Zelle beträgt hier 2 Mikrosekunden. Das entspricht einer Byte-Wandlungszeit von nominal 32 µs, die im schlimmsten Fall 27,5 bzw. 23,5 µs beim Lesen bzw. Schreiben betragen kann. Die Transfergeschwindigkeit liegt bei 500000 Bits pro Sekunde.

In MFM werden nur dann Datenimpulse in der Mitte einer Bit-Zelle geschrieben, wenn das entsprechende Datenbit logisch 1 war. Tritt ein einziges Null-Bit auf, so wird kein Datenimpuls gegeben. Folgen jedoch 2 oder mehr Null-Bits aufeinander, so wird nach dem ersten Null-Bit am Anfang einer Null-Daten Bit-Zelle ein Taktimpuls gesendet. Es wird also bei einer längeren Folge von Null-Bits von der Datenübertragung auf die Taktübertragung umgeschaltet. So ist gewährleistet, daß immer ein Referenzsignal vorhanden ist, mit dessen Hilfe sich der FDC auf die momentane Datenrate (bedingt durch Drehzahlschwankungen und unterschiedliche Taktfrequenzen bei Fremddisketten) einstellen kann.

13.3.3. Beispiel

Es soll das Byte >9B in FM und MFM übertragen werden. Dabei ergeben sich folgende Diagramme der zum Laufwerk gesendeten Impulse:

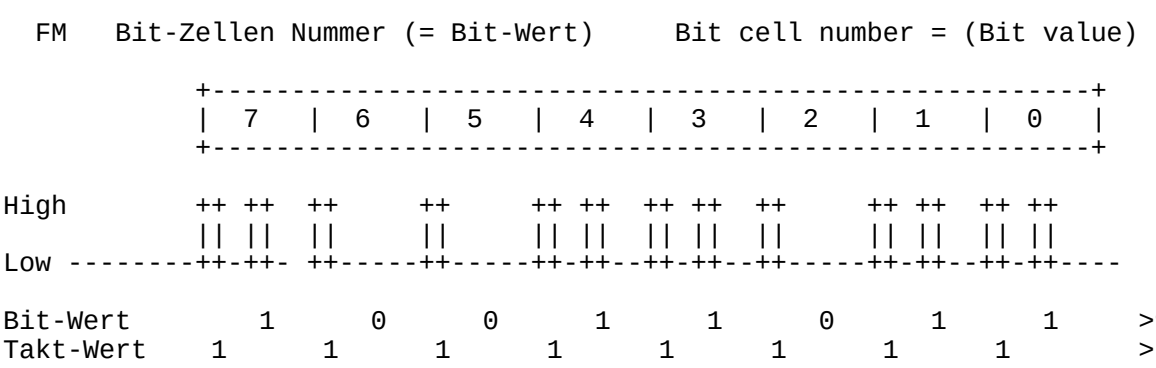
MFM — Double Density

The length of a bit cell amounts to here 2 microseconds. That corresponds byte transformation time from nominally 32 µs, those in the worst case 27.5 or. 23,5 µs during the reading or. Writing amounted to can. Those Data transfer rate is with 500,000 bits per second.

Only then data impulses in the center of a bit cell are written to page 65 in MFM, if the appropriate data bit were logic 1. If only one zero-bit occurs, then no data impulse is given. However if 2 or more zero-bits sequence, then to the first zero-bit at the start zero-data of a bit cell a clock pulse is transmitted. It is thus switched with a longer consequence by zero-bits by the data communication to the clock transfer. So is ensured that a reference signal is available always, with whose assistance the FDC to the data rate momentary (causes by speed fluctuations and different clock frequencies with foreign diskettes) adjust can.

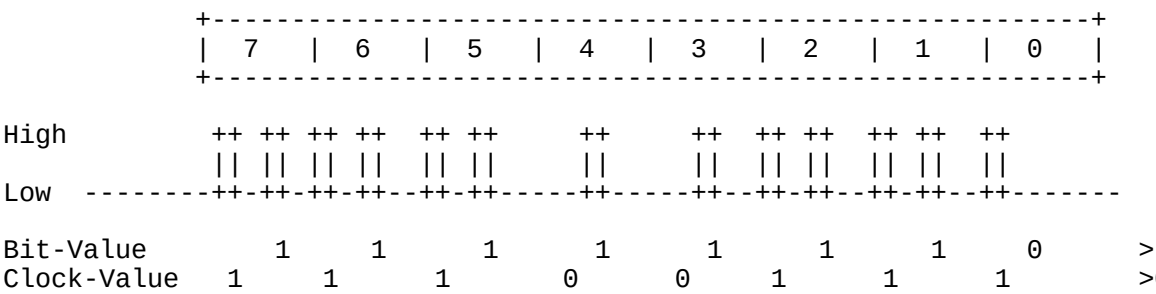
Example

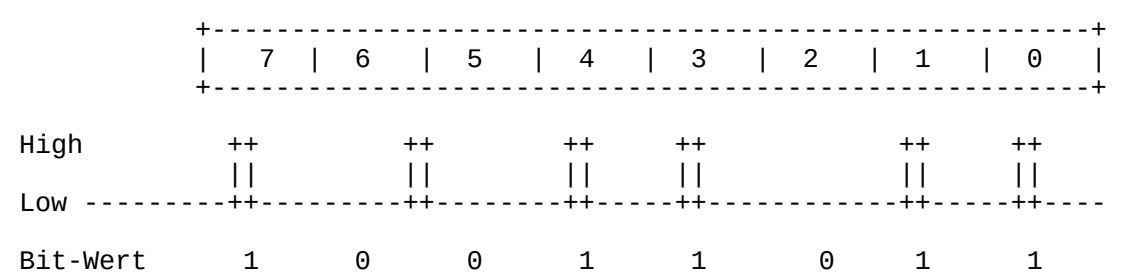
The byte >9B is to be transferred into FM and MFM. The following diagrams of the impulses transmitted to the drive result



Ein modifiziertes Signal wie z.B. die AF-Kennung besitzt die folgende Impulsstruktur:

A modified signal e.g. the AF identifier possesses the following impulse structure:





Wie zuvor beschrieben, sind Impulse am Anfang einer Bit-Zelle immer Taktsignale, solche inmitten einer Bit-Zelle immer Datenbits.

As before described, are impulses at the start of a bit cell always clock pulses, such in the midst of a bit cell always data bits.

Im Vergleich FM zu MFM muß beachtet werden, daß die Bit-Zellen bei MFM nur noch halb so groß (halb so lang) sind wie bei FM. Daraus

In the comparison of FM to MFM it must be noted that the bit cells are only half as large with MFM (half so long) as with FM. From it it

ergibt sich, daß die maximale Impulsfrequenz dann auftritt, wenn eine kontinuierliche Folge gleicher Datenbits gesendet wird. Diese Frequenz entspricht der Frequenz, die eine dauernde 1-Folge in FM erzeugt, mit dem wesentlichen Unterschied, daß bei FM nur jeder zweite Impuls Daten repräsentiert, bei MFM aber jeder Impuls ein Datenbit darstellt. Dies zeigt direkt, daß das Double-Density-Verfahren (MFM) die doppelte Datenübertragungs-geschwindigkeit und die doppelte Speicher-kapazität erzielt, ohne eine Erhöhung der maximal auftretenden Impulsfrequenz zu erfordern.

13.4. Spurstruktur

Nach all diesen theoretischen Überlegungen soll nun anhand einer eingehenden Besprechung der Spurstruktur die Umsetzung in die Praxis erläutert werden.

Anhand einer Art Basic-Programm soll der verwendete Algorithmus erklärt werden:

```
100   REM  Spurstruktur  Einfache
      Dichte
110   Schreibe 22 mal >FF
      REM Index-Gap, GAP 1 (CorComp:
      12 mal >FF)
120   Schreibe 6 mal >00
      REM Address-Gap
130   Schreibe 1 mal >FE
      REM ID-Address-Mark (IDAM)
131   Schreibe 1 mal Spurnummer
132   Schreibe 1 mal Seitennummer
133   Schreibe 1 mal Sektornummer
134   Schreibe 1 mal >01
      REM Sektorlänge 256 Bytes
135   Schreibe 1 mal >F7
      REM Erzeugt 2 CRC-Bytes
140   Schreibe 11 mal >FF
      REM ID-Gap, GAP 2
141   Schreibe 6 mal >00
150   Schreibe 1 mal >FB
      REM DATA-Address Mark(DAM)
151   Schreibe 256 mal >E5
```

results that the max. pulse frequency occurs if a continuous consequence of same data bits is transmitted. This frequency corresponds to the frequency, which produces a continuing 1-Folge in FM, with which substantial difference that with FM only each second impulse represents data, with MFM however each impulse a data bit represents. This shows directly that the double Density procedure (MFM) obtains the double channel transfer rate and the double storage capacity, without requiring an increase of the max. occurring pulse frequency

Track structure

After all these theoretical considerations now the conversion to the practice is to be described on the basis a detailed discussion of the track structure.

On the basis a pseudo Basic program is to be used to explain the algorithm:

```
100   REM  track  structure  single
      density
110   write 22 times >FF
      REM index Gap, GAP 1 (CorComp:
      12 times >FF)
120   write 6 times >00
      REM ADDRESS Gap
130   writes 1 time >FE
      REM ID ADDRESS Mark (IDAM)
131   write 1 time track number
132   write 1 time side number
133   write 1 time sector number
134   write 1 time >01
      REM sector length 256 bytes
135   writes 1 time >F7
      REM produces 2 CRC bytes
140   writes 11 time >FF
      REM ID Gap, GAP 2
141   write 6 times >00
150   write 1 time >FB
      REM Data Address mark (DAM)
151   write 256 times >E5
```

```
      REM Leerdaten
152   Schreibe 1 mal >F7
      REM Erzeugt 2 CRC-Bytes
153   Schreibe 45 mal >FF
      REM Data-Gap, GAP 3
160   IF NOT(Alle Sektoren fertig)
      THEN GOTO 120
170   Schreibe so oft >FF bis Ende
      REM GAP 4
180   END
```

Der Vorgang des Schreibens einer Spur beginnt und endet mit dem vom Laufwerk gelieferten Index-Puls. Durch die unvermeidbaren Drehzahlschwankungen des Laufwerkmotors ist es daher um das Index-Loch (Bezugspunkt beim Formatieren) herum Essig mit einer guten Synchronisation auf den Datentakt. Daher ist der Post-Index-Gap (Zeile 110) unbedingt notwendig, um Zeit zur Synchronisation auf den ersten Sektor der Spur zu haben.

Die Anzahl in Zeilen 140 und 141 (Gap 2) muß exakt eingehalten werden, alle anderen Gaps können länger, sollten jedoch nicht kürzer werden, will man nicht Synchronisations-aussetzer mit den entsprechenden Wartezeiten haben. Gap 4 muß so groß sein, daß dem FDC mehr Bytes angeboten werden, als er bis zum nächsten Index-Puls schreiben kann. Durch die Änderung der Umdrehungsgeschwindigkeit (Regelschwankungen etc.) variiert die effektive Spurlänge um einige Bytes, so daß hier Reserven für die niedrigste zulässige Drehzahl vorgesehen werden müssen.

Insbesondere Gap 2 und 3 sind von großer Bedeutung beim Schreiben von Sektoren, da der Beginn eines neu geschriebenen Sektors aus einer festgelegten Zeitverzögerung nach dem Ende des korrespondierenden Adreßfelds bestimmt wird. Näheres in Kapitel 16. Da auch während des Schreibens die Drehzahl schwanken kann, ist die Position des Sektor — Endes nicht bzw. nicht exakt vorhersagbar. Es muß daher eine 'Auslaufzone' hinter dem Sektor

```
      REM empty data
152   writes 1 time >F7
      REM produces 2 CRC bytes
153   writes 45 times >FF
      REM DATA Gap, GAP 3,
160   IF NOT(All sectors finished)
      THEN GOTO 120
170   write so often >FF to end
      REM GAP 4
180 END
```

The process of the writing of a track begins and ends with the index pulse supplied by the drive. By the unavoidable speed fluctuations of the drive engine it is therefore around the index hole (point of reference when formatting) around vinegar with a good synchronisation on the data clock. Therefore the post office index Gap (line 110) is absolutely necessary, in order to have time for the synchronization on the first sector of the track.

The number in lines 140 and 141 (Gap 2) must be kept accurately, all other Gaps can be longer, should not however become shorter, does not want one not synchronisation misfires with the appropriate waiting periods have. Gap 4 must be so large that are offered to the FDC of more bytes, than he can write up to the next index pulse. By the modification of the rotational speed (rule fluctuations etc.) varies the effective track length around some bytes, so that reserves for the lowest admissible number of revolutions must be designated here.

In particular Gap 2 and 3 are from great importance when writing sectors, since the beginning of a again written sector from a determined time delay is determined according to the end of the corresponding address field. Details in Chapter 16. Since also during writing the number of revolutions can vary, the position sector — of the end is not or. not accurately predictably. It must be available therefore a "discharge zone" behind the sector, which can be

bereitstehen, die belegt werden kann, ohne andere Sektoren bzw. deren Adreßfelder zu zerstören. Diese Auslaufzone stellt Gap 3 (Zeile 153) dar, dessen Länge mindestens 10 Bytes >FF und 4 Bytes >00 betragen muß.

Das Spurformat bei doppelter Dichte ist fast identisch, abgesehen von einigen Besonderheiten aufgrund der geforderten schnellen Synchronisation.

```
100   REM  Spurstruktur  Doppelte
      Dichte
110   Schreibe 32 mal >4E
      REM Index-Gap, GAP 1 (Myarc: 50
      mal >4E)
120   Schreibe 12 mal >00
      REM Address-Gap
130   Schreibe 3 mal >F5
      REM CRC-Start
131   Schreibe 1 mal >FE
      REM IDAM
132   Schreibe 1 mal Spurnummer
133   Schreibe 1 mal Seitennummer
134   Schreibe 1 mal Sektornummer
135   Schreibe 1 mal >01
      REM Sektorlänge 256 Bytes
136   Schreibe 1 mal >F7
      REM Erzeugt 2 CRC-Bytes
140   Schreibe 22 mal >4E
      REM ID-Gap, GAP 2
141   Schreibe 12 mal >00
150   Schreibe 3 mal >F5
      REM CRC-Start
151   Schreibe 1 mal >FB
      REM DAM
152   Schreibe 256 mal >E5
      REM Leerdaten
153   Schreibe 1 mal >F7
      REM Erzeugt 2 CRC-Bytes
154   Schreibe 28 mal >4E
      REM Data-Gap, GAP 3
160   IF  NOT(Alle  Sektoren  fertig)
      THEN GOTO 120
170   Schreibe  so  oft  >4E  bis  Ende
      REM GAP 4
180   END
```

Sowohl in einfacher wie auch in doppelter Dichte

occupied, without other sectors or to destroy their address fields. This discharge zone represents Gap 3 (line 153), whose length must amount to at least 10 bytes >FF and 4 bytes >00.

The track format with double density is almost identical, apart from some special features due to the required fast synchronization.

```
100   REM  track  structure  double
      density.
110   write 32 times >4E
      REM index Gap, GAP 1 (Myarc: 50
      times >4E)
120   write 12 times >00
      REM ADDRESS Gap
130   write 3 time >F5
      REM CRC start
131   write 1 time >FE
      REM IDAM
132   write 1 time track number
133   write 1 time side number
134   write 1 time sector number
135   writes 1 time >01
      REM sector length 256 bytes
136   write 1 time >F7
      REM produces 2 CRC bytes
140   writes 22 times >4E
      REM ID Gap, GAP 2,
141   write 12 times >00,
150   write 3 time >F5
      REM CRC start
151   writes1 time >FB
      REM DAM
152   write 256 times >E5
      REM empty data
153   write 1 time >F7
      REM produces 2 CRC bytes
154   write 28 times >4E
      REM DATA Gap, GAP 3,
160   IF  NOT(All  sectors  finished)
      THEN GOTO 120
170   writes so often >4E to end
      REM GAP 4,
180   END
```

Both in single as well as in double density bytes

werden Bytes benötigt, welche die Kennungen für Adreßfeld und Sektorstart aus dem allgemeinen Datenstrom hervorheben. Bei einfacher Dichte sind dies die Bytes >FE und >FB (die anderen werden in Kapitel 14 behandelt). Beide Bytes werden als >FE bzw. >FB geschrieben, jedoch fehlen ihnen an bestimmten Stellen Taktimpulse. Durch vorangegangene Synchronisationsdaten ist gewährleistet, daß der FDC bereits bytesynchron ist und diese Codierungsfehler korrekt erkennt. Da es ziemlich unwahrscheinlich ist, daß genau diese Fehler zufällig auftreten, werden diese Kennungsbytes sehr zuverlässig erkannt.

Bei doppelter Dichte ist die Sachlage etwas anders. Hier dienen nicht die ID-Bytes >FB und >FE selbst als Kennung, sondern die ihnen vorangehenden Bytes >F5, die nicht als >F5 sondern als >A1 mit einer fehlenden Taktflanke zwischen Bit 4 und 5 geschrieben werden. Auch hier wird ein Codierfehler simuliert, der erst dann korrekt erkannt wird, wenn der FDC bereits bytesynchron war.

are needed which emphasize the identifiers for address field and sector start from the general data stream. With single density these are the bytes >FE and >FB (the others in Chapter 14 are treated). Both bytes become as >FE or. >FB written, however in certain places clock pulses are missing to them. By preceding synchronisation data it is ensured that the FDC is already byte synchronous and detects these coding errors correctly. Since it is rather improbable the fact that exactly these errors occur coincidentally is detected very reliably these identifier bytes.

With double density is the state of affairs somewhat different. Here serve not the ID bytes >FB and >FE as identifier, but them preceding bytes >F5, which not when >F5 but when >A1 with a missing clock flank are written between bits 4 and 5. A coding error is simulated also here, which is only then correctly detected if the FDC were already byte synchronous.

14. Disketten

Das eigentliche Objekt unserer Betrachtungen tritt erst jetzt in Erscheinung — die Floppy-Disk oder genauer der Datenträger. Doch ist es wie bei jedem Datenträgersystem mit dieser Zuverlässigkeit, daß sich das Hauptaugenmerk auf die Daten richtet und von der 100% korrekten Funktion von Trägermedium und Laufwerk ausgegangen wird.

Doch das ist nicht zwangsweise so. Man kann durchaus durch Wahl von Diskettenmaterial und Laufwerk mitentscheiden, wie hoch die Fehlerquote in der Programmsammlung, bewirkt durch Kopierfehler oder ähnliches wird.

Als Fachausdrücke werden für die, die Diskette umgebende feste Hülle das Wort 'Jacket' benutzt, der eigentliche, beschichtete Datenträger wird mit 'Cookie' bezeichnet.

Bei der Wahl der geeigneten Disketten für das eigene System muß beachtet werden, daß beim TI prinzipiell 2 verschiedene Laufwerktypen zum Einsatz kommen: solche mit 40 Spuren und solche mit 80 Spuren. Hierbei ist die Seiten- bzw. Kopfanzahl ohne Belang bzw. muß im konstruktiven Zusammenhang beachtet werden. Exotische Laufwerke wie Apple-Restposten mit 35 Spuren sind nicht von Bedeutung, jedoch können sie zu den 40-Spur Laufwerken gezählt werden, da hier im Allgemeinen nur ein nachträglich montiertes Anschlagblech nicht mehr als 35 Spuren zulässt. Diese Laufwerke sind jedoch halbspurfähig, weshalb diese bei Anschlagausbau und entsprechender Modifikation der Stepper-Elektronik auch auf 80 Spuren 'aufgebohrt' werden können.

Daß und warum jedoch 80 Spur Laufwerke weniger für den Diskettentausch geeignet sind, soll weiter unten erklärt werden.

Um die folgenden Passagen wirklich zu

Diskettes

The actual object of our views goes only now into action — the floppy diskette or more exactly the data carrier. But it is as with each data carrier system with this reliability that the special attention is directed toward the data and is proceeded with the 100% correct function from carrier medium and drive.

But that is not obligatorily like that. One can quite along decide by selection by diskette material and drive, as highly the error rate in the program collection, caused by copying errors or the like becomes.

As technical terms become for those, which uses diskette surrounding fixed covering the word "Jacket," which actual, coated data carrier marked with "Cookie."

With the selection of the suitable diskettes for the own system it must be noted that with the TI 2 always different drive types are used: such with 40 tracks and such with 80 tracks. Here is the page or. number of headings without importance or. must be considered in the constructional connection. Exotic drives such as Apple residual items with 35 tracks are not of importance, however they can be ranked among the 40-track drives, since here generally only a subsequently installed impact sheet metal does not permit any more than 35 tracks. These drives are however half-trackable, why these with impact development and appropriate modification of the stepping electronics also on 80 tracks to become "drilled out" to be able.

That and why however 80 track of drives is suitable for the diskette exchange fewer, is to be further down explained.

In order to really understand the following

verstehen, muß an dieser Stelle noch einiges nachgetragen werden, was thematisch zwar zu den Aufzeichnungsverfahren gehört, jedoch auch bei der Qualitätsbetrachtung wichtig ist, nämlich das Zusammenspiel zwischen Diskettenoberfläche und Schreib-/Lesekopf.

Bei einseitigen Diskettenlaufwerken befindet sich der Kopf dort, wo nach Einlegen der Diskette in das Laufwerk die Unterseite des Cookies liegt. Ihm gegenüber liegt, in Ermangelung eines zweiten Kopfes, ein sog. Andruckfilz der dafür sorgt, daß das Diskettenmaterial immer so fest am Kopf anliegt, daß eine einwandfreie magnetische Kopplung zwischen Cookie und Kopf gegeben ist. Dabei ist der Kopf feststehend montiert und der Filz an einem beweglichen Arm befestigt, der mittels des sog. Head-Load-Magneten von dem Cookie abgehoben bzw. auf dieses abgesenkt werden kann. Bei zweiseitigen Diskettenlaufwerken befindet sich an der Stelle des Andruckfilzes ein zweiter Schreib-/Lesekopf, der wie der Andruckfilz beweglich an einem kurzen Arm montiert ist.

Beim Öffnen der Verriegelung eines Floppy-Disk-Laufwerkes wird der Andruckfilz bzw. der zweite Kopf angehoben, um beim Herausziehen der Disketten diese nicht zu beschädigen bzw. nicht selbst beschädigt zu werden. Dies kann im Betrieb mittels der Head-Load-Magnete auch dann geschehen, wenn das Laufwerk längere Zeit nicht angesprochen wurde. Doch dazu mehr im folgenden Abschnitt.

14.1. Allgemeine Qualitätsmerkmale

Nach dem zuvor gesagten kann man also Diskettenlaufwerke in verschiedene Kategorien einteilen: solche mit nur einem Schreib-/Lesekopf, solche mit zweien, Laufwerke mit 40 Spuren und solche mit 80 Spuren und Laufwerke mit Head-Load-Magnet und solche ohne. Doch bei allen ist eines gleich: sind die

passages, still some must be entered afterwards, which belongs thematically to the recording methods, however also during the view of quality importantly is i.e., interaction between disk surface and write/ read head here.

With one-sided floppy disk drives is the heading, where after inserting into the drive the lower surface of the Cookies is appropriate for the diskette. Opposite it is situated, in the absence of a second heading, sucked. Pressure felt for it ensures that the diskette material rests always so firmly against the heading that a perfect magnetic coupling between Cookie and heading is given. The heading is being certain installed and felt to a mobile lever fastened, which sucked by means of. Head load magnet taken off from the Cookie or. on this to be lowered can. With bilateral floppy disk drives a second write/ read head in the place of pressure felt, which is installed at a short lever like pressure felt mobile, is.

When opening the interlock of a floppy diskette drive pressure felt becomes or. the second heading raised not to damage over when pulling the diskettes these out or. not to be damaged. This can occur in the operation by means of the Head load magnets also if the drive longer time were not addressed. But in addition more in the following paragraph.

General quality criteria

After said the before one can divide thus floppy disk drives into different categories: such with only one write/ read head, such with two, drives with 40 tracks and such with 80 tracks and drives with Head load magnet and such without. But with all one is alike: if the drives are addressed, then Cookie have and heading always

Laufwerke angesprochen, so haben Cookie und Kopf immer einen sehr engen Kontakt, da die Köpfe nicht über der Cookie-Oberfläche fliegen, wie bei Harddisks oder großen Plattenlaufwerken, sondern richtig fest angedrückt werden. Dies stellt hohe Anforderungen an die Oberflächengüte von Disketten, die visuell nur unzureichend bewertet werden kann.

a very close contact, since the headings over the Cookie surface do not fly, as with hard disks or large disk drives, but to be pressed in slightly correctly firmly. This makes high demands against the surface quality of diskettes, which can be evaluated visually only insufficiently

Aufgrund des so verursachten Abriebs geben Diskettenhersteller an, wie oft auf eine Spur zugegriffen werden kann bis deren Magnetschicht so beschädigt ist, daß die Funktion gefährdet wird. Praktisch sinkt dabei der Ausgangspegel der Magnetschicht (Remanenz oder MOL, Maximum Output Level) unter die geforderten 70% des Aufsprechpegels — die Daten drohen im Rauschen zu verschwinden.

Due to the in such a way caused abrasion diskette manufacturers indicate, how often a track it can be accessed to their magnetic stratum is so damaged that the function is endangered. Practically thereby the output level of the magnetic stratum (remanence or MOL, maximum output level) sinks under the required 70% of the output levels — the data threaten to disappear in the noise.

Aus diesem Grund werden Disketten, nachdem sie mit der magnetisierbaren Schicht versehen wurden, in der späteren Aufzeichnungszone geschliffen, was die Rauhtiefe reduziert und die Oberflächenhärte erhöht. Hier hat man bereits das erste Qualitätsmerkmal: eine homogene, leicht glänzende Oberfläche.

From this reason diskettes, after they became to provide with the magnetizable layer, in the later recording zone become polished, which the roughness depth reduced and which increases surface hardness. Here one has already the first quality criterion: a homogeneous, easily shining surface.

Disketten, deren Cookie eine matte oder inhomogene Oberfläche hat, sollten nicht zur laufenden Benutzung oder besser gar nicht verwendet werden. Natürlich ist dies nur ein grober Anhaltspunkt und nur geeignet, eine erste Entscheidung zu erlauben. Doch es nutzt sich ja nicht nur die Diskette durch Reibung ab, auch der Kopf bekommt seinen Teil ab.

Diskettes, whose Cookie has a matte or non-homogenous surface, should not not be used for the current use or better at all. This only a rough reference point is and naturally only suitable to permit a first decision. But does not only wear itself the diskette out by friction, also the heading gets its section off.

Neben der reinen Oberflächengüte ist der Rundlauf, speziell der Seitenschlag einer Diskette wichtiges Qualitätsmerkmal. Besonders bei zweiseitigen Laufwerken fällt ein verzogenes Cookie durch Schleifgeräusche auf, die nicht toleriert werden sollten. Dies merkt man leider aber erst dann, wenn die Diskette im Laufwerk

Apart from the pure surface quality is special the cyclic testing, the lateral run out of a diskette important quality criterion. Particularly with bilateral drives a spoiled Cookie is noticeable by sharpening noises, which should not be tolerated. One notices this unfortunately however only then if the diskette is situated and

liegt und rotiert. Diese Diskette sollte verschenkt werden oder zur Tauschdiskette degradiert werden. Tritt dies bei Disketten einer Firma öfters auf, so sollte man deren Disketten nicht mehr kaufen.

Ein Qualitätsmerkmal, das insbesondere für den Einsatz auf 80-Spur-Laufwerken maßgeblich ist, stellt die maximale Speicherdichte einer Diskette dar, angegeben in 'tracks per inch' oder als übliches Kürzel 'tpi'.

Hierzu ein kurzer Blick auf den Schreib-/Lesekopf eines Disklaufwerkes. Dieser besteht genau betrachtet aus 3 Systemen: dem eigentlichen Schreib-/Lesesystem und zwei jeweils rechts und links dahinter angeordneten Löschköpfen, den sogenannten 'Tunnel Erase Heads'. Beim Schreiben von Daten wird nun das Nutzsignal über den Schreib-Lesekopf, der nun Schreibkopf ist, als Magnetisierung (Flußwechsel) in die Diskettenoberfläche eingebracht. Sodann läuft die magnetisierte Zone an den Löschköpfen vorbei, welche die Ränder der Datenspur weglöschen.

Damit wird die Datenspur von den Nachbarspuren getrennt und so ein Spur-Übersprechen verhindert. Man verschenkt also Platz auf der Diskette, um die Datensicherheit zu erhöhen. Erinnern wir uns: Bei der Formatierung haben wir ähnliches kennengelernt.

Eine kleine Graphik soll dies erläutern:

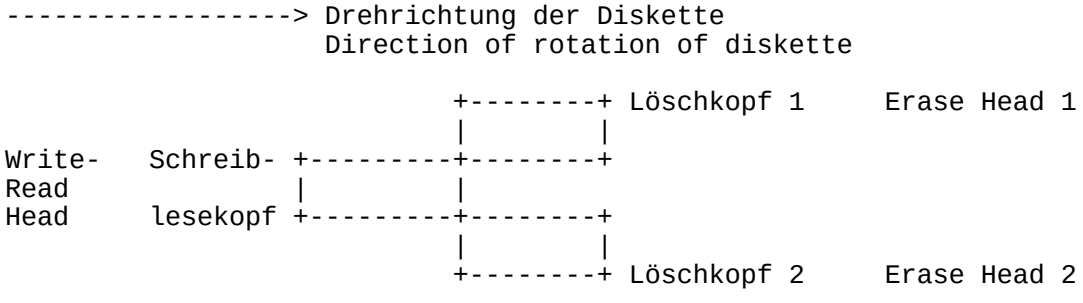
rotates in the drive. This diskette should be given away or degradiert to the exchange diskette. Footstep this with diskettes of a company repeatedly up, then one should buy their diskettes any longer

A quality criterion, which is relevant for the application on 80-track drives in particular, represents the max. memory density of a diskette, indicated in "tracks per inch" or as usual contraction "tpi."

For this a short view of the write/ read head of a disk drive. This consists exactly regarded of 3 systems: the actual write/ reading system and two on the right of and left in each case behind it arranged erase heads, the so-called ' tunnel Erase Heads '. When writing data now the information signal is brought over the read-write head, which is now recording head, as magnetization (flux change) into the disk surface. Then the magnetized zone runs by the erase heads, which away delete the edges of the data track.

Thus the data track is separated from the neighbor tracks and so a track cross-talk is prevented. One given away thus place on the diskette, in order to increase data security. We remember: During formatting we became acquainted with something similar.

A small graph is to describe this:



Die Zahl vor dem Kürzel 'tpi' gibt (eher unzureichend) an, wie fein strukturiert die magnetisierbare Oberfläche der Diskette ist und wie schmal demzufolge eine Datenspur minimal werden kann, ohne daß die Daten 'in der Materialkörnung verloren gehen' (strenggenommen ist die Oberfläche kristallin, aber das ist hier von geringer Bedeutung).

The number before the contraction "tpi" indicates (rather insufficiently), as fine the magnetizable surface of the diskette structures is and as narrowly therefore a data track can become minimum, without the data are lost "in the material granulation" (strictly the surface is crystalline, but that is here of small importance).

Dabei ergibt sich aufgrund der Abmessungen eine Spurdichte von 1,9 Spuren pro Millimeter bei 48 tpi und eine Dichte von 3,6 Spuren pro Millimeter bei 96 tpi. Dieser Wert für die radiale 'Auflösung' der Diskettenoberfläche ist also abhängig von der Laufwerksmechanik. Für die axiale Auflösung spielt dies keine Rolle, da hier die Anzahl der Flußwechsel pro Zeit- bzw. aufgrund der festen Rotationsgeschwindigkeit (einer Spur) pro Radiant wesentlich ist.

A track density results 1.9 tracks per millimeter in the case of 48 tpi and a density of 3.6 tracks per millimeter in the case of 96 tpi due to the dimensions. This value for the radial "dissolution" of the disk surface depends thus on the drive mechanics. For the axial dissolution this no role plays there here, the number of flux changes per time or. due to the fixed rotation rate (a track) per radian is substantial.

Natürlich beeinflussen sich beide Parameter, da es recht aufwendig wäre, ein anisotropes Material zu verwenden, nur um unterschiedlichen Datendichten gerecht zu werden.

Naturally both parameters, since it would be quite complex to use an anisotropic material influence each other only in order different data densities to become fair.

Wie bei den Formatierungsarten indirekt gezeigt, unterscheidet sich die Anzahl der Flußwechsel innerhalb einer Spur nicht wesentlich, wenn statt FM nun MFM benutzt wird - die FM-Flußwechsel übersteigen im Mittel eher die der MFM. Dies ist unabhängig davon, ob mit 40 oder 80-Spur-Laufwerken gearbeitet wird.

As shown is the case for the types of formatting indirectly, the number of river changes differs within a track not substantially, if instead of FM now MFM is used - the FM river changes exceed on the average rather those the MFM. This is independent of whether with 40 or 80-track drives one operates.

Da jedoch die Spurbreite bei 80 Spur Laufwerken nur halb so groß ist wie bei 40 Spur Laufwerken, erfordern 80-Spur-Laufwerke solche Disketten mit 96 tpi, 40 Spur Systeme nur solche mit 48 tpi. Das ist (hinkend) zu vergleichen mit der Qualität von Billigst-Eisenoxid Compact Cassetten und z.B. Reineisencassetten oder Video-Cassetten.

Since however the track width is only half as large with 80 track drives as with 40 track drives, 80-track drives require such diskettes with 96 tpi, 40 track of systems only such with 48 tpi. That is (limping) to compare with the quality of approving of ferric oxide Compact cassettes and e.g. pure iron cassettes or video cassettes.

Durch die geringere Spurbreite bei 80 Spur

By the smaller track width with 80 track systems

Systemen muß u.a. der Aufsprechstrom reduziert werden, um ein Seiten- bzw. Spurübersprechen zu verhindern. Durch die engere Fokussierung des S-/L-Kopfes wird zudem eine tiefere Magnetisierung erreicht. Je nach Laufwerk kann dies dazu führen, daß auch 48 tpi Disketten auf 80 Spur Laufwerken verwendet werden können, sowohl bei einfacher wie doppelter Dichte. Diese Sicherheitsreserven weisen aber im Allgemeinen nur Markendisketten auf.

must be reduced among other things the output current, over a side or. Track-cross talk to prevent. Besides a deeper magnetization is achieved by the closer focusing of the S-/L-head. Depending upon drive this can lead to it that also 48 tpi diskettes can be used on 80 track drives, both with simpler and double density. This security reserves indicate however generally only label diskettes.

Ein wesentlich den Preis einer Diskette bestimmender Faktor ist die Angabe EINSEITIG oder ZWEISEITIG (doppelte Speicherdichte kann immer als gegeben angenommen werden). Disketten, die ausdrücklich nur für einfache Dichte gedacht sind, sollte man lassen, wo sie sind — dies sind nur bessere Schleifscheiben.

The price of a diskette determining factor the instruction single-sided or double-sided (double memory density can be always assumed as given) is substantial. One should leave diskettes, which are meant for single density expressly only, where they are — these are only better grinding wheels.

Ein- oder Zweiseitig gibt bei Markenware nur an, ob die Disketten auf einer oder allen beiden Seiten geprüft wurden. Dies ist ähnlich der Fertigung von Transistoren und ihrer Einstufung in Stromverstärkungsgruppen: je nachdem, ob die Verstärkung reicht oder nicht, wird der Stempel entsprechend aufgedrückt. Genauso bei Disketten. Bestehen Sie die Prüfung auf beiden Seiten, werden sie als zweiseitige Typen (Kennung 2D oder DS/DD) verkauft. Fallen sie bei der Prüfung der zweiten Seite aus, bzw. werden sie hier erst gar nicht getestet (spart immerhin Zeit!), so werden sie als einseitige Disketten angeboten (1D oder SS/DD). Dabei ist jedoch verschiedenes zu beachten.

Single- or double-sided only indicates for branded articles whether the diskettes were checked for or all two sides. This is similar to the manufacturing of transistors and its classification in groups of current reinforcement: according to whether goes the reinforcement or not, the stamp accordingly pushed open. Exactly the same with diskettes. If you insist the check on both sides, they are sold as double-sidedl types (identifier 2D or DSDD). Fail they with the check of the second sidee, or. they are not tested here only at all (time nevertheless saves!), thus they are offered as one-sided diskettes (1D or SSDD). However different is to be considered.

So hängt es vom Prüfverfahren des jeweiligen Herstellers ab, ob die ausgesonderten Disketten (einseitig bzw. solche, die als weiße Ware auf Umwegen wieder in den Markt kommen) tatsächlich fehlerhaft arbeiten oder ob man den Vermerk '100% geprüft' besser gleich vergißt. Manche Hersteller prüfen die ganze Oberfläche jeder Seite, erkennen also Fehler zwischen den

Thus it depends on the testing method of the respective manufacturer whether the selected diskettes (one-sided or. such, which come as white commodity on detours again into the market) actually incorrectly operate or whether one forgets the note "100% checked" better directly. Some manufacturers check the whole surface of each side, detect thus errors between

Spuren. Das soll sicherstellen, daß auch fehljustierte Laufwerke mit diesen Disketten arbeiten. Solche Disketten arbeiten dann in der Regel auch bei 80 Spur Systemen, wenn sie nicht gleich als 96tpi-Typen verkauft werden.

Andere Prüfverfahren setzen (fast) normale Laufwerke ein, wo geschrieben, zurückgelesen und hinterher gelöscht wird. Diese billigen und schnellen Verfahren sind naturgemäß weniger zuverlässig. Die Laufwerke werden übrigens so ausgestattet, daß sie nicht nur die Daten, sondern auch deren Pegel ermitteln können. Gelöscht wird übrigens (wenn nach ECMA/DIN verfahren wird) mit einem statischen Magnetfeld.

Generell kann man aber davon ausgehen, daß einseitige und doppelseitige Disketten eines Typs auf ein und derselben Fertigungsstraße hergestellt werden und man durch eigenen Disktest (DM-2 Modul oder Formatieren mit Verify) den Test der zweiten Seite nachholt und feststellt, ob es klappt oder nicht. Der vielfach benutzte Allgemeinplatz 'Es mag ja ein paar Mal gehen, aber man darf sich über plötzlichen Datenverlust nicht wundern.' ist daher fern ab der Realität bei der Herstellung von Floppy-Disks.

14.2. Technische Qualitätsnormen

Mit der großtechnischen Herstellung von Disketten und den daraus resultierenden Anforderungen an die Einhaltung bestimmter Normen bezüglich Oberflächengüte, Maßhaltigkeit, Parameterbeständigkeit etc. ergibt sich zwangsläufig die Notwendigkeit, bestimmte Normen festzulegen. Ziel dieser Normen ist dabei nicht primär, die Vergleichbarkeit der Produkte unterschiedlicher Hersteller zu sichern, sondern die Gewährleistung der Austauschbarkeit des Produktes durch einen Mindeststandard.

the tracks. That is to guarantee that drives with these diskettes, also false adjusted, operate. Such diskettes operate then usually also at 80 track systems, if they are sold not directly as 96 tpi-types.

Other testing methods use (almost) normal drives, where one writes, one reads backwards and one deletes afterwards. These cheap and fast procedures are naturally fewer reliable. The drives are equipped by the way in such a way that them not only the data, but also their level to determine to be able. One deletes by the way (if in accordance with ECMA/DIN one proceeds) with a static magnetic field.

Generally one can assume however single-sided and double-sided diskettes of a type are manufactured on the same production line and one retrieves and tightens the test of the second side by own disk test (DM 2 module or formatting with verify), whether it folds or not. The often used general saying "it may go a few marks, but one may not be surprised at sudden overrun." is far therefore starting from the reality with the production of floppy diskettes.

Technical quality standards

With the industrial production of diskettes and the request resulting from it to the adherence to certain standards concerning surface quality, accuracy to size, parameter stability etc.. inevitably if the necessity results, determined standards determine. Target of these standards is not primary to protect the comparability of the products of different manufacturers but the guarantee of the exchangeability of the product by a minimum standard.

Bei Disketten betrifft dies die rein mechanischen Eigenschaften (Maße, Materialien, Abrieb etc.) sowie die magnetischen Eigenschaften des Datenträgers (Koerzitivität, Remanenz-flußdichte, Temperaturabhängigkeit der gespeicherten Daten etc.). Daneben werden sog. physikalische Eigenschaften vorgegeben (wer auch immer diese Unterscheidung vorgab war sich wohl nicht im Klaren, daß auch die anderen beiden Eigenschaften zur Physik gehören …).

Diese Standards setzen die ECMA (European Computer Manufacturers Association) für Europa sowie international die ISO (International Organisation for Standardization). Auch das DIN (Deutsches Institut für Normung) hat, wie könnte es anders sein, hier seine Normen festgelegt. Die PTB (Physikalisch Technische Bundesanstalt) liefert Meßdisketten bzw. Meßparameter für Prüfanlagen.

Festgelegt werden z.B. die Normen für 5¼"-Disketten mit 40 Spuren in der Vorschrift ECMA-70. Die DIN-Norm (erschlägt alle Diskettenstandards) hat die Nummer DIN 66247. Und da es keinen Sinn ergibt, europäische Standards als Insellösung zu etablieren, entsprechen diese Normen in allen wesentlichen Punkten den entsprechenden ISO-Vorgaben (ISO 6596, 7487, 8630 und 8378).

Für den interessierten Leser hier ein Überblick über die genormten Merkmale, die eine Diskette aufweisen muß. Ein Auflistung der exakten Zahlenwerte und Prüfschemata würde den Rahmen für dieses Thema sprengen.

Die Lagerung von Disketten sollte möglichst bei einer Raumtemperatur von 23°C (maximal 26°C) und einer Luftfeuchtigkeit von ca. 50% (maximal 80%) geschehen. Dabei darf ein magnetisches Feld von maximal 4 kA/m wirken, andernfalls die Daten (erlaubterweise) in Mitleidenschaft gezogen werden. Dabei sollte beachtet werden,

With diskettes this concerns the purely mechanical characteristics (mass, materials, abrasion etc..) as well as the magnetic characteristics of the data carriers (coercivity, remanence flux density, temperature dependence of the stored data etc.). Besides become so-coalled physical characteristics given (whoever did not give this distinction realized itself probably that also the other two characteristics belong to physics …).

These standards set the ECMA (European Computer Manufacturers Association) for Europe as well as internationally the ISO (International Standards Organization). Also DIN (German Institute for Standardization) determined, as could it differently be, here its standards. The PTB (Physics Technical Federal Institute) supplies measuring diskettes or. Measuring parameter for testing equipment.

E.g. the standards for 5¼"-Diskette with 40 tracks in the regulation ECMA-70 are determined. The DIN standard (overrides all diskette standardsl) has the number DIN 66247. And there it a sense does not result in to establish European standards as isolated solution corresponds these standards in all substantial points to the appropriate ISO specifications (ISO 6596, 7487, 8630 and 8378).

For the interested reader here an overview of the standardized features, which a diskette must indicate. Listing of the accurate values of the number and test patterns would blow up the framework for this topic.

The storage of diskettes should if possible with an ambient temperature of 23°C (max. 26°C) and an air humidity of approx.. 50% (max. 80%) occur. A magnetic field of max. 4 may kA/m be worked, otherwise the data (permit-proves) in be pulled. It should be noted that the normal write current corresponds to a field strength of

daß der normale Schreibstrom einer Feldstärke von etwa 24 kA/m entspricht, weshalb es bei der Lagerung auch etwas weniger sein darf (Disketten weg vom Monitor!).

Der Schreibstrom von 24 kA/m bei üblichen DD-Disketten ist übrigens der Grund, wieso die an sich höherwertigeren High-Density Disketten eines PC der AT-Klasse auf den gängigen Laufwerken am TI nicht zufriedenstellend arbeiten. Diese Disketten benötigen einen Schreibstrom von etwa 50 kA/m, weshalb sie nach Formatierung mit 24 kA/m einen völlig unzureichenden Wiedergabepegel (MOL) aufweisen.

Neben so selbstverständlichen Anforderungen wie der Einhaltung aller Maße (Jacket-Höhe, -Breite, -Dicke, Cookie-Durchmesser, Zentrierungslochdurchmesser, innere und äußere Begrenzung der Datenzone), werden genaue Vorgaben zur erlaubten Krümmung/Wölbung des Jackets (diese beeinflussen die Jacket-Dicke) und zu den statischen und dynamischen Eigenschaften der magnetisierbaren Schicht auf beiden Seiten des Cookies gemacht. Die prinzipielle Überlegenheit des Kunststoffgehäuses von 3½"-Disketten wird hierbei übrigens nicht immer ausgespielt!

Genormt sind die Bereiche, in denen sich die zur Beschleunigung und konstanter Rotation des Cookies benötigten Drehmomente bewegen, der Durchmesser bzw. die Lichtdurchlässigkeit des Index-Loches zusammen mit der Undurchlässigkeit des Cookie-Materials für das Licht der zur Index-Erkennung benutzten Infrarot-LEDs (Intensität am Indexloch dient als Bezugswert), die Pegelbereiche, innerhalb derer Signaleinbrüche bzw. Reste an sich gelöschter Signale erlaubt sind, der Wiedergabepegel einer einmalig formatierten Spur nach 3 Millionen Umdrehungen bei geladenem Kopf sowie nach 24-stündiger Lagerung unter den oben genannten Grenzwerten für Temperatur und Luftfeuchte.

approximately 24 kA/m, why it may be also somewhat fewer with the storage (diskettes away of the monitor!).

The write current of 24 kA/m with usual DD diskettes is by the way the reason, why the High Density actually with higher order diskettes PC of the AT class on the usual drives on the TI satisfyingly does not operate. These diskettes need a write current of approximately 50 kA/m, why they indicate a completely insufficient playback level (MOL) after formatting with 24 kA/m.

Apart from as natural request as the adherence to all mass (Jacket height, width, thickness, Cookie diameter, centring hole diameter, internal and outside delimitation of the data zone), exact specifications are made the permitted curvature/curvature of the Jackets (this to influence the Jacket thickness) and the static and dynamic characteristics of the magnetizable layer on both pages of the Cookies. The superiority in principle of the plastic housing of 3½" diskette is not by the way always out-played here!

The areas, within which the torques needed for acceleration and constant rotation of the Cookies move, are standardized the diameter or. the light permeability of the index hole as well as the impermeability of the Cookie material for the light of the infrared LED's used for the index recognition (intensity at the index hole serves as reference value), the ranges of level, within those signal failures or. Remainders deleted signals are permitted to a uniquely formatted track actually, the playback level after 3 million revolutions with loaded heading as well as after 24-hour storage under the limit values for temperature and humidity, specified above.

15. Kopierschutztechniken und was man dagegen tun kann

Dieses, zugegeben etwas heikle Thema ist fast so alt wie die Datenspeicherung auf der Diskette.

Kopierschutztechniken wurden entwickelt, um das unberechtigte Kopieren von Programmen oder Daten zu unterbinden und so eine Kalkulation über Aufwand und Gewinn aus einem Programm oder anderen auf Datenträger gespeichertem Material erst zu ermöglichen. Der Kopierschutz ist also aus Sicht dessen, der ihn installiert, eine vernünftige, ja notwendige Sache.

Anders aus der Sicht des Anwenders dieser Daten oder Programme. Er ist in einem solchen Fall auf die Originaldiskette angewiesen, die, wie im letzten Kapitel besprochen, einem prinzipiellen Verschleiß unterliegt. Er muß daher ständig mit dem Ausfall dieser Diskette rechnen, insbesondere bei häufigem Gebrauch des Programms. Bedenkt man dabei noch, daß Rechner und Programme Betriebsmittel sind, deren Ausfall hohe Kosten verursachen kann, so sieht man damit die andere Seite eines Kopierschutzes, wenn keine Kopie verfügbar ist, die sofort einsetzbar ist. Daher bieten verschiedene Hersteller Sicherheitskopien ihrer Programme an, die beim Erwerb des eigentlichen Programms günstig mit verkauft werden.

Für einen TI-Besitzer sieht die Sache etwas anders aus: hier muß ein defektes Originalprogramm auf langen Wegen wieder beschafft werden, was, Betriebsmittel oder nicht, immer lästig ist. Kurzum: der Kopierschutz muß weg oder ein entsprechendes Programm muß her!

Wie ein solches Programm arbeiten kann, sei im folgenden beschrieben.

Dabei wird bei den elementaren Tricks begonnen und bei der eingehenden

Copy protection techniques and what one can do against it

This, somewhat delicate topic is admitted almost as old as the data storage on the diskette.

Copy protection techniques were developed, in order to prevent the unauthorized copying from programs or data to and to only enable so a calculation over expenditure and gain from a program or other material stored on data carrier. The copy protection is thus from view its, which installs it, a reasonable, necessary thing.

Differently from the view of the user of these data or programs. It is dependent in such a case on the original diskette, which, as discussed in the last chapter, is subject to a wear in principle. He must count therefore constantly on the failure of this diskette, in particular with frequent use of the program. If one considers with the fact still that computers and programs are resources, whose failure can cause high costs, then one sees thereby the other page of a copy protection, if no copy is available, which is applicable immediately. Therefore different manufacturers offer backup copies of their programs, which are sold favorably with the acquisition of the actual program with.

For a TI owner the thing looks somewhat differently: here a defective original program on long ways must be procured again, which, or not, is always annoying of resources. In short: the copy protection must away or an appropriate program must ago!

As such a program can operate, is described in the following.

Begun stopped with the elementary cheat and with the detailed manipulation of track contents

Manipulation von Spurinhalten und Adreßfeldern aufgehört. Voraussetzung für das Verständnis der folgenden Seiten ist jedoch die eingehende Kenntnis von Spurstruktur und Dateninterpretation. Dazu wird das Programm COPY-C des Verfassers als Experimentalwerkzeug empfohlen.

Es werden aber nur Themen behandelt, bei denen auf irgendeine Weise das Kopieren von Disketten unterbunden wird. Listschutz und andere Trivialitäten sind nicht Gegenstand der Untersuchungen, ebenso wie die Frage, wie ein Schutz aus dem Programm selbst entfernt werden kann, nicht gestellt oder beantwortet wird.

Alle Betrachtungen gelten, sofern nicht anders vermerkt, im Prinzip für alle auf dem TI verfügbaren Diskcontroller, auch Myarc und Percom. Ebenso ist die Speicherdichte prinzipiell ohne Belang, von der größeren Speicherkapazität einmal abgesehen.

Dabei können die Schutzmethoden in drei Kategorien eingeteilt werden:

- Methoden, bei denen der Schutz mittels des normalen DOS der Diskcontrollerkarte erzeugt und nachher geprüft wird (Nutzung der Level-1-Routinen).
- Methoden, bei denen der Schutz mittels eines speziellen Programms erzeugt, später jedoch mit dem Standard-DOS geprüft wird.
- Verfahren, die bei Erzeugung und Test auf eigene Level-0 oder Level-1-Routinen angewiesen sind.

Zusätzlich muß noch beachtet werden, ob geschützte Zonen der Diskette nur eine Kennung auf Raubkopien sind (prinzipiell also Flags, z.B. fast alle Quality-99 Software) oder ob sich in diesen Zonen programmrelevante Daten befinden (etwa TP'99 oder die Millers Graphics Tools). Eine abweichende Methode ist die, daß der Loader das Hauptprogramm patcht und

and address fields. A prerequisite for the understanding of the following pages is however the detailed knowledge of track structure and data interpretation. In addition the program COPY-C of the author is recommended as experimental tool.

However only topics are treated, with which somehow copying diskettes is prevented. Cunning protection and other trivia are not the subject of the investigations, just as the question, how a protection can be removed from the program, is not asked or is not answered.

All views apply, if differently does not note, in principle to all disk controllers, also Myarc and Percom available on the TI. Likewise memory density is always without importance, refrained from the larger storage capacity once.

The protection methods can be divided in three categories:

- Methods, with which the protection is produced by means of the normal DOS of the disk controller card and checked afterwards (using the Level-1 Routine).
- Methods, with which the protection produces by means of a special program, later however with the standard DOS one checks.
- Procedures, which with production and test are dependent on own Level-0 or Level-1-Routine.

Additionally it must be still considered whether protected zones of the diskette are only one identifier on pirate copies (thus always flag, e.g. almost all Quality-99 software) or whether in these zones program-relevant data (for instance TP'99 or the Millers Graphics Tools) are. A deviating method is those the fact that the Loader the main program and this Patches in

diese Patches im Programm gelegentlich abgefragt oder als feste Pointer verwendet werden (z.B. der GPL-Assembler, bei dem der Aufwand für den verschlüsselten Loader in keinem Verhältnis zur lachhaften Abfrage der o.ä. Patches steht).

Im Vorgriff auf die folgenden Passagen soll hier kurz beschrieben werden, was unter einem sog. Standard-DOS zu verstehen ist, welche Beschränkungen diesem auferlegt sind und worin ein sog. eigenes DOS von den Standard-Spezifikationen abweicht.

Wie aus den vorangegangenen Kapiteln bekannt ist, lassen sich die für den TI verwendbaren Diskcontroller auf zwei verschiedenen Softwareebenen ansprechen.

Auf dem höchsten Level geschieht dies mit dem Standard-PAB und dem DSRLNK mit DATA 8. Hierbei sind sowohl einfache als auch komplexe File-Operationen möglich, wobei die Diskette nur in logische Verbunde mit der Bezeichnung 'File' aufgeteilt wird.

Das nächste Niveau sind die Level-1-Routinen mit einem reduzierten VDP-PAB und dem DSRLNK mit DATA 10. Hierbei tritt die Sektorstruktur der Diskette in Erscheinung, besonders bei den Routinen >10, >11 und in gewisser Weise auch bei >14, >15.

Die physikalische Einheit 'Sektor' ist ein beim TI-DOS immer 256 Byte langer Block von Daten, unabhängig von der gewählten Speicherdichte und seiner Position auf der Diskette. Ein solcher Sektor ist durch seine laufende Nummer immer exakt definiert d.h. er kann allein durch seine Nummer gefunden werden. Dabei werden die Sektoren innerhalb einer Spur immer von 0 bis 8 bei einfacher Dichte und von 0 bis 17 bei doppelter Dichte durchnumeriert. Die Entscheidung, ob der korrekte Sektor in der korrekten Spur gefunden wurde, ermöglichen Spur- und Seitennummer im Adressfeld des betreffenden Sektors.

the program is patched is occasionally queried or used as fixed pointer (e.g. the GPL assembler, with that the expenditure for the encoded Loader in no relation to the laughful query or the like Patches is).

In the anticipation on the following passages is to be described here briefly, what under one sucked. Standard DOS to understand is, which limitations are imposed upon to this and where sucked. own DOS of the standard specifications deviates.

As admits from the preceding chapters is, the disk controllers on two different software levels, usable for the TI, can be responded.

On the highest level this occurs with the standard PAB and the DSRLNK with DATA 8. Here both simple and complex file operations are possible, whereby the diskette is divided only into logical networks with the designation "file."

The next level are the Level-1-Routine with a reduced VDP PAB and the DSRLNK with DATA 10. Here the sector structure of the diskette steps >10, >11 into appearance, particularly with the routines and in certain way also with >14, >15.

The physical unit "sector" is a block of data, long with the TI-DOS always 256 byte, independent of selected memory density and its position on the disk. Such a sector is always accurately defined by its serial-number i.e. it can alone by its number be found. The sectors are always numbered consecutively within a track from 0 to 8 with single density and from 0 to 17 with double density. Track and side number in the address field of the sector concerned make the decision possible, whether the correct sector in the correct track was found.

Um nun auf einen Sektor zugreifen zu können, wird dem Standard-DOS die Sektornummer übergeben. Nachdem die Speicherdichte ermittelt wurde, wird diese Sektornummer durch die Anzahl der Sektoren pro Spur dividiert, wodurch sich im ganzzahligen Anteil die Spurnummer und im Rest die relative Sektornummer in dieser Spur ergibt. Überschreitet die Spurnummer dabei die festgelegte maximale Anzahl von Spuren, so wird diese Zahl am Zahlenkreis gespiegelt und ergibt so die Spurnummer auf der zweiten Diskettenseite.

In order to be able, the standard DOS the sector number hand over to access now a sector. After memory density was determined, this sector number is divided by the number of sectors per track, whereby in the integral portion the track number results and in the remainder the relative sector number in this track. If the track number exceeds thereby the fixed maximum number of tracks, then this number at the number circle is reflected and results in so the track number on the second diskette side.

Anschließend wird diese Spur mit dem Kopf des Laufwerkes angefahren. Nachdem diese Spur (theoretisch) erreicht ist, wird anhand eines beliebigen Adressfeldes geprüft, ob dies die korrekte Spur ist, andernfalls wird die Spur erneut angefahren. Dies ist möglich, da alle Adreßfelder einer Standard-Spur gleiche Spurnummern haben. Danach wird dem Floppy-Disk-Steuerchip der Befehl gegeben, einen Sektor zu lesen bzw. zu schreiben, dessen Adressfeldparameter wie Seite, Spurnummer und Sektornummer zuvor berechnet und an den Steuerchip übergeben wurden. Beim Lesen von Sektoren wird nach Erkennen des Adressfeldes auf die Sektorstartmarkierung gewartet, beim Schreiben wird mit einer Zeitverzögerung gearbeitet.

Subsequently, this track with the head of the drive assembly is started. After this track (theoretical) is reached, on the basis any address field is examined, whether this is the correct track, otherwise one the track is again started. This is possible, since all address fields have a standard track and track number. Afterwards the instruction is given to the floppy disk controlling chip to read and/or write a sector, its address field parameters such as side, track number and sector number was computed before and to the controlling chip handed over. When reading sectors after recognizing the address field for the sector starting marking one waits, during the writing with a time delay operates.

Man erkennt aus all dem, daß beim Standard-DOS viel gerechnet und wenig gedacht wird - schließlich wurde für die korrekte Spurstruktur bereits beim Formatieren gesorgt.

One detects from all that the fact that with the standard DOS much is counted and few are thought was already provided — finally for the correct track structure when formatting.

Es ist aber auch zu sehen, daß ein eigenes DOS sich an keine dieser Konventionen zu halten braucht, wenn es erst einmal aus normalen Sektoren geladen wurde. So ist es z.B. denkbar, einen Sektor mit einer zu hohen Sektornummer zu versehen, der von dem Standard-DOS nicht erkennbar ist. Dieses Thema wird jedoch weiter unten, quasi 'vor Ort' genauer behandelt.

In addition, it is to be seen that its own DOS needs to adhere to none of these conventions, if it were loaded only from normal sectors. Like that it is e.g. conceivable to provide a sector with a too high sector number which is not recognizable by the standard DOS. This topic is treated more exactly however further down, quasi "locally."

Ein eigenes DOS, das ist im einfachsten Fall ein abgeschriebenes und gekonnt manipuliertes

Its own DOS, that is in the simplest case a copied and skillfully manipulated piece from a ROM

Stück aus einem ROM-Listing eines Diskcontrollers.

15.1. Schutzmethoden mit dem Standard-DOS

Diese Schutzmethoden zeichnen sich dadurch aus, daß sie auch ohne größeren Aufwand realisierbar sind. In der Regel ist jedoch auch die damit verbundene Schutzwirkung ohne größeren Aufwand zu eliminieren …

15.1.1. Flaggesteuerter Kopierschutz

Hier sei an erster Stelle das Protection-Byte in Sektor 0 einer TI-Diskette erwähnt, siehe auch Kapitel 7. Sofern das Programm nicht selbst dieses Byte prüft (z.B. ID-DATA oder -KONTO), kann es durch ein Leerzeichen ersetzt werden (mit jedem beliebigen Sektoreditor), woraufhin diese Diskette auch mit dem DM-2 Modul kopierbar ist. Ansonsten lassen sich solche Disketten mit jedem Sektorkopierer kopieren — der Schutz wird in diesem Fall mit kopiert.

Alle flaggesteuerten Schutzmethoden haben einen Nachteil: sie greifen nur, wenn das Kopierprogramm bzw. die File-Verwaltung dieses Flag auch prüft. So verhält es sich auch mit dem Schreibschutz eines Files. Es kann nur auf höchster Ebene (Filezugriff) geschützt werden; vor einem Sektorzugriff ist es nicht schreibgeschützt. Eine derart manipulierte Diskette ist jedoch als einfache Sektorkopie (Programm unter Verwendung z.B. der DOS-Routine >10) lauffähig.

15.1.2. Sektormanipulation

Sektoren, die über das normale DOS les- bzw. schreibbar sein sollen, müssen die normalen Sektorkennungen besitzen. Damit sind sie normal mit einem Sektorkopierer kopierbar. Der einzige Weg, eine Kennung in einen Standard-Sektor zu bringen, die nicht über Standard-Sektorkopie kopiert werden kann, ist, diesen Sektor zu zerstören. Das kann so geschehen, daß

listing of a disk controller.

Protection methods with the standard DOS

These protection methods are characterised by the fact that they are realizable without larger expenditure even. Usually is however also the associated protective effect without larger expenditure to eliminate …

Flag-controlled copy protection

Is mentioned in the first place the protection byte in sector 0 of a TI diskette here, see also Chapter 7. If the program does not check this byte (e.g. ID DATA or account), can it by a blank is replaced (with any sector editor), whereupon this diskette is copyable also with the DM 2 module. Otherwise such diskettes with each sector copier can be copied — in which case the protection is copied as well.

All flag-signaled protection methods have a disadvantage: they access only, if the copying program or. the file administration this flag also checks. Thus it behaves also with the write protection files. It can be protected only on highest level (file access); before a sector access it is not write protected. An in such a manner manipulated diskette is however executable as simple sector copy (program using e.g. the DOS routine >10).

Sector manipulation

Sectors, those over the normal DOS read or to be writable, must the normal sector identifiers possess. Thus they are normally copyable with a sector copier. The only way to bring an identifier into a standard sector which cannot be copied over standard sector copy, is, to destroy this sector. That can occur in such a way that a program always writes and for the "Program

ein Programm mittels DOS-Routine >10 immer Daten auf ein und denselben Sektor schreibt und der 'Programmschützer' mitten drin das Laufwerk öffnet. Der Sektor ist dann mit ziemlicher Sicherheit defekt und jeder Standard-Zugriff auf diesen Sektor erzeugt einen Lesefehler. Dies kann abgefragt und im geschützten Programm ausgewertet werden (Beispiel COPY-A von R. Frommer). Kopieren lässt sich ein solcher Sektor auch mit einem Standard-Sektorkopierer, bei dem nach dem normalen Lauf, der ja alle Sektoren korrekt schreibt, ein eigenes Programm mittels Routine >10 diese Sektoren mit ihren gelesenen Daten immer wieder schreibt und der 'Programmwilderer' auch im richtigen Moment die Klappe öffnet.

protector" in the middle in it the drive opens data by means of DOS routine >10 on the same sector. The sector is then defective with considerable security and each standard access to this sector produces a read error. This can be queried and analysed in the protected program (example COPY-A von R. Frommer). To copy such a sector leaves itself also with a standard sector copier, with which after the normal run, which writes all sectors correctly, writes its own program again and again by means of routine >10 to these sectors with their read data and the "program programmwilderer" opens the flap also in the correct moment.

So lassen sich nur Disketten kopieren, bei denen der Inhalt des manipulierten Sektors nicht ausgewertet wird. Ist dies jedoch der Fall, so müssen andere Verfahren angewandt werden.

Thus only diskettes can be copied, with which contents of the manipulated sector are not analysed. If this is however the case, then other procedures must be applied.

Ein Sektor kann auch als defekt ausgewiesen werden, wenn er gar nicht existiert. So sind z.B. MACROWORD plus und EASYDATA 99 so geschützt, indem die Diskette nicht komplett formatiert wurde und so einige Spuren keine Sektoren enthalten. Der Zugriff auf sie erzeugt nur Fehlermeldungen. Derartige Disketten lassen sich jedoch mit partiell arbeitenden Sektorkopierern wie Nibbler auf eine fabrikneue Diskette kopieren, sofern kein Myarc-Diskcontroller verwendet wird, der ja bekanntlich nur 40 Spuren formatieren läßt (da COPY-C auf dieser Karte nicht arbeitet, ist sowieso alles zu spät …).

A sector can be proven also as defective, if it does not exist at all. Like that e.g. MACROWORD plus and EASYDATA 99 is so protected, as the diskette was not completely formatted and so some tracks no sectors contains. That access to it produces only error messages. Such diskettes can be copied however with partially operating sector copiers such as Nibbler on a diskette fresh from the factory, if no Myarc Disk controller is used, which lets there only 40 tracks format as well known (COPY-C on this card does not operate, is anyway everything too late …).

15.1.3. Doppelte Sektoren

Double sectors

Diese Sektoren können nur über eine eigene Formatierroutine erzeugt werden. Bekannter Vertreter dieser Gattung ist der SPAD-XIII Flugsimulator, der diese Besonderheit als Flag auswertet, ob die Diskette kopiert wurde oder eine Originaldiskette vorliegt.

These sectors can be produced only over their own formatting routine. Well-known representative of this kind is SPAD XIII the flight simulator, which analyses this special feature as flag whether the diskette was copied or an original diskette is present.

Beim Formatieren werden zweimal die

When formatting twice the identical address

identischen Adreßfelder zusammen mit den korrekten Sektorinhalten geschrieben. Versieht man diese Sektoren mit verschiedenen Daten, so sieht es aus, als wäre der Sektor defekt und lieferte laufend verschiedene Daten. Ein Schreiben dieses Sektors über das Standard-DOS ist ein nicht kalkulierbares Glücksspiel, da auch der Verify des DOS scheinbar schiefgeht.

Das Programm liest nun diesen Sektor zweimal kurz nacheinander und wertet die unterschiedlichen Daten aus. Da bei diesem Verfahren ein Sektor nun den Platz von zweien einnimmt, muß ein anderer Sektor dafür entfallen. Damit ist dann wieder ein Flag verfügbar, nämlich das Fehlen eines anderen Sektors.

15.1.4. Sektoren mit definierten Fehlern

Hierzu muß einleitend bemerkt werden, daß Sektoren mit Datenfehlern (CRC-Fehler) vom Standard-DOS gelesen werden können. Da ein solcher Fehlertest erst NACH dem Lesen aller 256 Bytes vorgenommen wird, befinden sich die Daten bereits in dem spezifizierten Puffer. Die Information des Fehlerbytes ist dann gewissermaßen eine Zugabe. Schrittfehler oder nicht erfolgte Sektoridentifikation übergeben generell keine Daten in den Puffer bzw. liefern nur ein festes Bitmuster, im Allgemeinen >00.

Spuren, bei denen ein bestimmter Sektor eine falsche Spurkennung aufweist, werden zwar korrekt angefahren, jedoch wird dieser Sektor nicht erkannt und ein Record-Not-Found-Fehler (RNF-Error) gemeldet. Haben alle Adreßfelder einer Spur ein falsches Spurbyte, so wird ein SuchFehler (Seek-Error) ausgegeben. Hierbei wurde zwar die Spur möglicherweise korrekt angefahren, doch simulierten die falschen Spurnummern einen Fehlzugriff. Mithilfe der Fehlercodes (siehe Kapitel 11) kann nach Zugriff auf diese Sektoren über das Standard-DOS die Fehlerart ermittelt werden. Kopierprogramme müssen diese Eigenschaften einer manipulierten Spur erkennen und korrekt behandeln.

fields as well as correct sector contents are written. If one provides these sectors with different data, then it looks, as if the sector would be defective and supplied constantly different data. A writing of this sector over the standard DOS is a not calculable gambling, since also the verify of the DOS goes wrong apparently.

The program reads now this sector twice briefly successively and analyses the different data. Since with this procedure a sector takes now the place of two, another sector for it must be omitted. Thus then again a flag is available, i.e. the absence of another sector.

Sectors with defined errors

For this it must be noticed introductory that sectors with data errors (CRC errors) can be read by the standard DOS. Since such a error test is only made AFTER reading all 256 bytes, the data already are in the specified buffer. The information of the error byte is then to a certain extent an addition. Step errors or effected sector identification generally transfer no data into the buffer or. supply only a fixed bit pattern, generally >00.

Tracks, with which a certain sector indicates a false track identification, are correctly started, however this sector is not detected and a record NOT Found error (RNF error) is announced. Credit all address fields of a track a false track byte, then a search error (Seek error) is output. Here the track was started possibly correctly, but simulated the false track numbers false access. Assistance of the error codes (see Chapter 11) can be determined after access to these sectors over the standard DOS the error. Copying programs must detect and correctly treat these characteristics of a manipulated track.

15.2. Spurstrukturen 'Hausmacher-Art'

Damit begeben wir uns auf die Spielwiese dessen, was uns ein eigenes, im Idealfall vollkommen flexibles DOS ermöglicht: Das freie Spielen mit Adreßfeldern. Dies ist bereits die Stufe der Diskettenmanipulation, auf der die meistens Kopierschutzverfahren angesiedelt sind. Das Grundprinzip ist immer folgendes:

Eine bestimmte Anzahl von Spuren und Sektoren genügen den Anforderungen des normalen DOS, da aus ihnen ja schließlich der Loader geholt werden muß, der einen Zugriff mittels eines eigenen DOS' erst erlaubt.

Anschließend wird das eigentliche Programm mittels der nun geladenen Software installiert. Beispiele sind XB-Detective und alle 3 Millers Graphics Utilitites. Turbo-Pasc'99 und TI-Artist weisen jeweils nur einige wenige (programmrelevante) Sektoren mit verfälschten Kennungsbytes auf und besitzen kein ausgesprochen eigenes DOS.

Auf dem Niveau der Spurstruktur, bei der allein aus den Adreßfeldern die korrekte Reproduktion der Spur erfolgen kann, arbeiten alle Kopierprogramme, die im folgenden als 'Adressfeld-Scanner' bezeichnet werden sollen.

15.2.1. Das Adressfeldproblem

Für einen einfachen Adressfeld-Scanner wie Backup 1.1 oder Trackhack ist es in gewissem Sinn 'lebenswichtig', alle Adreßfelder einer Spur ab dem Indexloch in der korrekten Reihenfolge zu lesen. Da bei dem Floppy-Disk-Controller-Chip (kurz FDC genannt) jedoch während des Lesens von Adreßfeldern keine Möglichkeit gegeben ist, den Index-Puls der Floppy-Station zu testen, kann zwar der Anfang einer Spur mittels eines Pseudo-Seek-Befehls erkannt werden, jedoch nicht deren Ende. Adressfeld-Scanner müssen also immer davon ausgehen, daß sich in einer Spur nur eine bestimmte maximale Anzahl von Adreßfeldern befinden kann, ein paar mehr lesen und dann nach einer

Track structures of "home-made" type

Thereby we go on the play meadow its that enables us its own DOS perfectly flexible in the ideal case: Free playing with address fields. This is already the level of the diskette manipulation, on which those are settled mostly copy protection procedures. The basic principle is the always following:

A certain number of tracks and sectors meet the request of the normal DOS, since from them the Loader must be finally gotten, that access by means of its own DOS' only permitted.

Subsequently, the actual program is installed by means of the software loaded now. Examples are XB Detective and all 3 Millers Graphics Utilitites. Turbo-Pasc'99 and TI-Artist indicate only in each case some few (program-relevant) sectors with falsified identifier bytes and possess none expressed own DOS.

On the level of the track structure, with which alone from the address fields the correct reproduction of the track can come, operate all copying programs, which are to be called in the following "address field scanners."

The address field problem

For a simple address field scanner such as Backup 1.1 or Trackhack is "vital" it in certain sense to read all address fields of a track starting from the index hole in the correct order. However there with the Floppy Diskette Controller chip (FDC) during reading address fields no possibility is given to test the index pulse of the floppy station can the at the beginning of a track by means of a pseudo seek instruction be detected, however not their end. Address field scanners must thus always assume that that in a track only a certain max. number of address fields can be, few a more to read and then for a repetition of the first address field look. From this these programs determine the

Wiederholung des ersten Adressfeldes suchen. Daraus bestimmen diese Programme die Anzahl der Sektoren pro Spur.

Durch das Lesen aller Adreßfelder werden jedoch die Bedingungen der vorangegangenen Abschnitte erfüllt. Doppelte Sektoren oder nicht existente Sektoren können jedoch sowohl die Berechnung der Sektoranzahl als auch den (notwendigen) Fluß beim Lesen der Sektoren stören. Zudem erheben einfache Adressfeldscanner wie die oben genannten Programme die gelesenen Adreßfelder zum Maß aller Dinge und versagen spätestens dann, wenn Disketten in einfacher Dichte auf Systemen mit Controllern für beide Dichten kopiert werden sollen (z.B. Trackhack).

15.2.1.1. Adreßfelder mit CRC-Fehlern

Genau wie ein Sektor wird auch ein Adressfeld über eine Prüfsumme gegen Fehler abgesichert. Besonders dann, wenn ein Double-Density-FDC eine Single-Density-Diskette liest, werden Sequenzen als Adreßfelder erkannt, die keine sind. In der Regel weisen diese Adreßfelder (ab jetzt AF genannt) dann CRC-Fehler auf. Ein solches AF ist auf unserem momentanen Niveau der Spuranalyse ohne Bedeutung, da ein zu einem Adressfeld mit CRC-Fehler gehörender Sektor in keinem Fall über einen Sektor-Lesebefehl erreichbar ist. Wurde ein solches AF gelesen, so ist es aus der Liste zu streichen.

15.2.1.2. Auswertung der Adreßfelder

Wurden alle AF gelesen, alle fehlerhaften entfernt und die Anzahl der Sektoren in der momentan zu bearbeitenden Spur ermittelt, so kann mit der Reproduktion der Spur, d.h. der Formatierung der Kopie begonnen werden. Dies jedoch nur, wenn keine weitere Prüfung auf Spurlängenüberschreitung, nicht existente Sektoren oder ähnliches vorgenommen werden soll. Für viele Kopierschutzmethoden beim TI ist dies auch nicht notwendig, soll aber wenigstens kurz angesprochen werden.

number of sectors per track.

However the conditions of the preceding paragraph are fulfilled by reading all address fields. Double sectors or non-existent sectors can disturb however both the calculation of the number of sectors and (necessary) the flux during the reading of the sectors. Besides simple address field scanners raise like the programs specified above the read address fields for the measure of all things and fail at the latest if diskettes in single density on systems with controllers for both densities to be copied to be supposed (e.g. Trackhack).

Address fields with CRC errors

Exactly like a sector, an address field array is also secured by a checksum against errors. Particularly then, if a double-density FDC reads a single Density diskette, sequences are detected as address fields, which are not. Usually these address fields (from now on simply called AF) indicate then CRC errors. Such AF is on our level momentary of the track analysis without meaning, since a sector belonging to an address field with CRC error is attainable over a sector read instruction in no case. If such AF was read, then it is to be deleted from the list.

Analysis of the address fields

All AF was read, all incorrect and the number of sectors in that was removed was determined track at the moment which can be processed, then can be begun with the reproduction of the track, i.e. formatting the copy. This however only, if no further check is to be made on track length exceeding, existente sectors or the like. For many copy protection methods with the TI this is also not necessary, is to be addressed however at least briefly.

15.2.1.3. Singuläre Adreßfelder

Singuläre AF sind solche AF, zu denen kein Sektor gehört. Diese AF täuschen gewissermaßen die Existenz eines Sektors vor. Da sich direkt an ein solches AF ein weiteres AF anschließen kann ist es möglich, in einer Spur ca. 50 AF unterzubringen, wovon keines einen Sektor 'im Schlepptau' hat (Beispiel GPL-Assembler).

Ein Kopierprogramm, welches auf eine Diskette trifft, die innerhalb einer Spur sowohl mit doppelten Sektoren als auch mit nicht existenten Sektoren geschützt ist, wird in der Regel nicht in der Lage sein, die doppelten Sektoren in der korrekten Sequenz zu lesen.

Es ist daher unbedingt nötig, nach dem Lesen aller AF zuerst die von ihnen angekündigten Sektoren zu lesen und den Ausgang des Leseversuches zu protokollieren. Ergibt sich dabei zu einem AF ein RNF-Fehler, dann handelt es sich um ein singuläres AF. Es muß zwar bei der Formatierung, nicht aber während des Lesens der originalen Sektoren (von der Originaldiskette) beachtet werden.

Ein gutes Kopierprogramm wird daher nach der Ermittlung der Adressfeldstruktur zuerst alle so angemeldeten Sektoren 'ins Leere' lesen und die dabei gewonnenen Statusmeldungen weiter auswerten.

Als weiterer Vorteil dieser Leseversuche ist anzusehen, daß die Berechnung der Summe aller Bytes der Sektoren dieser Spur nun auf einem festeren Fundament steht. Näheres bringt der folgende Abschnitt.

15.2.1.4. Berechnung der Spurlänge aus den AF

Das Lesen eines AF liefert 6 Bytes an Informationen, entsprechend den 6 Bytes, aus denen beim Formatieren das AF aufgebaut wird. In der korrekten Reihenfolge sind dies: Spurnummer, Seitennummer, Sektornummer,

Singular address fields

Singular AF are such AF, to which no sector belongs. This AF pretends to a certain extent the existence of a sector. There directly such AF a further AF to follow can do is it possible, in a track approx. to accommodate 50 AF, about which no "in the dragging rope" has a sector (example GPL-Assembler).

A copying program, which does not meet a diskette, which is protected with non-existent sectors within a track both with double sectors and, usually able will be to read the double sectors in the correct sequence.

It is absolutely necessary to read after reading all AF first the sectors announced by them and to log the output of the read attempt. In the case of it if a RNF Error results to a AF, then it concerns a singular AF. It must be considered with formatting, not however during reading the original sectors (the original diskette).

A good copying program will read all in such a way announced sectors therefore after the determination of the address field structure first "in emptiness" and will continue to analyse the status messages won thereby.

As the further advantage of these read attempts it is to be regarded that the calculation of the total of all bytes of the sectors of this track is now on a fixed foundation. Details bring the following paragraph.

Calculation of the track length from the AF

Reading a AF supplies 6 bytes at information, according to which 6 bytes, of which when formatting the AF is composed. In the correct order is this: Track number, side number, sector number, sector length identifier as well as 2 CRC

Sektorlängenkennung sowie 2 CRC-Prüfsummenbytes. Die CRC-Bytes können vernachlässigt werden, es sei denn, man will die Funktion des CRC-Generators im FDC überprüfen (...). Wichtigste Variable für ein Kopierprogramm ist jedoch die Sektorlänge. Allein aus ihr berechnet der FDC beim Lesen und Schreiben die Anzahl Bytes, die der folgende Datensektor enthält. Die steuernde CPU hat keine Möglichkeit, mehr Bytes zu lesen, als der FDC anhand des Längenbytes erkennt. Näheres zu den vom FDC gelieferten Informationen im betreffenden Kapitel 16, zur Spurstruktur sind bei den Aufzeichnungsverfahren (Kapitel 13) mehr Informationen verfügbar.

Durch Berechnung der Zahl der Bytes eines Sektors über das Längenbyte und Summation aller so gewonnenen Sektorlängen der momentan bearbeiteten Spur lässt sich die Anzahl der Datenbytes in einer Spur berechnen. Diese kann, bedingt durch notwendige Rücksichtnahme auf Drehzahlschwankungen des Laufwerks etc., einen Maximalwert nicht überschreiten, da in diesem Fall der letzte Sektor einer Spur den ersten überschreiben würde.

Ein Richtwert kann aus der Maximalzahl von 10 Standard-Sektoren pro Spur bei einfacher und der doppelten Anzahl bei doppelter Dichte berechnet werden. Pro Sektor sind das 256 Bytes, entsprechend einer maximalen Datenbyteanzahl von 2560 (>A00) bzw. 5120 (>1400).

Ein singuläres AF geht in diese Berechnung naturgemäß nicht ein. Existieren jedoch nichtsinguläre AF, mit denen sich eine zu große Spurlänge ergeben würde, so ist zumindest 1 Sektor nicht komplett enthalten. Dieser Sektor kann aus bestimmten Gründen nur Daten unterhalb des Wertes >F5 enthalten, sein Inhalt muß bereits beim Formatieren festgelegt worden sein! Ein solcher Sektor darf nicht mittels eines Sektorschreibbefehls geschrieben werden, da dann der o.ä. Fall eintritt, daß er den ersten Sektor am Spuranfang mit Sicherheit

checksum bytes. The CRC bytes can be neglected, it is, one wants the function of the CRC generator in the FDC to check (...). Most important variable for a copying program is however the sector length. However from it the FDC calculates the number of bytes, which the following data sector contains during reading and writing. The controlling CPU does not have the possibility to read more bytes than the FDC detects on the basis the length byte. Details to the information supplied by the FDC in Chapter 16 concerned, to the track structure are available with the recording methods (Chapter 13) more information.

By calculation of the number of the bytes of a sector over the length byte and summation of all in such a way won sector lengths of the track at the moment processed the number of data bytes in a track can be calculated. This can not exceed, due to necessary consideration for speed fluctuations of the drive etc, a maximum value, since in this case the last sector of a track would overwrite first.

An approximate value can be calculated from the maximum number by 10 standard sectors per track with simpler and the double number with double density. Per sector that is 256 bytes, according to a max. number of data bytes of 2560 (>A00) or. 5120 (>1400).

A singular AF does not enter this calculation naturally. However if not-singular AF exists, with which a too large track length would result, then at least 1 sector is not completely contained. This sector can contain for certain reasons only data below the value of >F5, its contents must when formatting have been already determined! Such a sector may not be written by means of a sector write instruction, there then or the like case arises that it overwrites the first sector at the beginning of track with security.

überschreibt.

Beim TI wird das bei Advanced Diagnostics und DISkASSEMBLER gemacht.

15.2.1.5. Gar keine Adreßfelder auffindbar

Vor dem Beginn einer Spuranalyse muß bekannt sein, ob und in welcher Dichte eine Spur formatiert ist. Dies kann nur ausprobiert bzw. vom Bediener abgefragt werden. Es gibt zwei mögliche Verfahren zum Ausprobieren. Das schnellste ist, einen Spurinhalt in einer beliebigen Dichte zu lesen und nach Adressfeld- oder Sektormarken zu suchen, also bei SD nach den Bytes >FB, >FE oder >CB. Wird keines dieser Bytes innerhalb 60 Bytes nach Spuranfang gefunden, so handelt es sich bei der gewählten Dichte um die falsche. So schnell dieses Verfahren ist, so unzuverlässig ist es.

Sicherer ist es, in der willkürlich gewählten Dichte ein Adressfeld zu lesen. Wurde eines ohne CRC-Fehler erkannt, so wurde die richtige Dichte gewählt. Im anderen Fall muß auf eine andere Speicherdichte umgeschaltet werden, woraufhin der Versuch wiederholt wird. Verwendet man einen Speicher für die zuletzt erkannte Dichte, so ist es möglich, die Dichteerkennung zu beschleunigen.

Wurde jedoch auch hier kein korrektes AF entdeckt, so ist diese Spur im allgemeinen als nicht formatiert zu betrachten. Das bedeutet aber nur, daß sie keine AF oder Sektoren im klassischen Sinn enthält. Vielmehr ist anzunehmen, daß Daten spurorientiert abgelegt wurden. Dieses Thema wird in Abschnitt 'Spurkopie' besprochen.

Kann mit hinreichender Sicherheit angenommen werden, daß die Spur leer ist, so wird die Kopie mit >00 formatiert und ist fortan in dieser Spur unlesbar.

With the TI with Advanced Diagnostics and DISkASSEMBLER is made.

No address fields discovered

Before the beginning of a track analysis it must admits to be whether and in which density a track is formatted. This can only tried out or. by the user to be queried. There are two possible procedures for trying out. The fastest is to read and look for address field or sector labels a track contents up in any density, thus with SD after the bytes >FB, >FE or >CB. None of these bytes is found within 60 bytes to beginning of track, then it concerns with the selected density the false. So this procedure is fast, so unreliably is it.

It is safer to read in the arbitrarily selected density an address field. If one was detected without CRC errors, then the correct density was selected. In the other case must be switched to another memory density, whereupon the attempt is repeated. If one uses a memory for the density detected last, then it is possible to accelerate the density recognition.

However even if no correct AF was discovered here, then this track is to be regarded generally as not formatted. That means it however only that she does not contain AF or sectors in the classical sense. Rather it is to be assumed that data were track-oriented stored. This topic is discussed in paragraph "track copy."

It can be assumed with sufficient security that the track is empty, then the copy with >00 is formatted and is from now on in this track unreadable.

15.2.2. Reproduktion der Spur anhand der Adreßfelder

Wurde mithilfe der AF-Analyse eine Spurstruktur analysiert, so kann mit dem Kopieren begonnen werden. Zuerst muß dabei die Spur der Zieldiskette (der Kopie) formatiert werden, bevor mit dem Transfer der Sektoren begonnen wird.

15.2.2.1. Formatieren der Spur anhand der Adreßfelder

Beim Aufbau der Spur in einem Puffer (vor dem Schreiben auf Diskette) müssen die vom 'blinden Lesen' ermittelten Fehlercodes der einzelnen Sektoren beachtet werden.

Begonnen wird mit dem normalen Index-Gap, der das erneute Synchronisieren nach dem durch das Index-Loch verursachten Aussetzer des Datenstroms gewährleistet.

Nach dem Adress-Gap wird das erste AF wie gelesen wieder einformatiert. Es wird mit einem CRC-Auslöser abgeschlossen. Hatte der korrespondierende Sektor die Fehlermeldung Record-Not-Found (RNF) verursacht, so wird der Data-Gap geschrieben und mit dem nächsten AF fortgefahren.

War der Sektor in Ordnung, so wird aus dem Längenbyte die effektive Sektorlänge mit >E5 rohformatiert, gefolgt vom CRC-Auslöser. Der Rest wiederholt sich nach gleichem Schema.

Bei überlangen Spuren kann es sein, daß beim Aufbau des letzten Sektors das Spurende erreicht wird. Die Überwachung der Spurlänge beim Formatieren muß also immer gewährleistet sein!

War ein überzähliger Sektor gemeldet, der eine Überschreitung der maximalen Datenbyteanzahl hervorrief, empfiehlt es sich, diesen Sektor zu lesen und ihn beim Spuraufbau mit diesen Daten in die Rohformatierung einzubeziehen. Das setzt

Copying the track on the basis of the address fields

If a track structure is deduced with the help of an address field analysis, then copying can begin. The track on the target diskette (copy diskette) must be formatted before any sectors are transferred.

Formatting the track on the basis of the address fields

With the structure of the track in a buffer (before writing on diskette) must of "blind reading" determined error codes of the individual sectors be considered.

One begins with the normal index gap, which ensures renewed synchronizing after the misfire of the data stream caused by the index hole.

After the address gap the first AF is read again in-formatted like. It is locked with a CRC trip. If the corresponding sector had caused the error message Record Not Found (RNF), then the Data Gap is written and with the next AF continues.

If the sector was correct, then from the length byte the effective sector length with >E5 is raw formatted, followed of the CRC trip. The remainder repeats itself according to same pattern.

With excessive tracks it can be that with the structure of the last sector the end of track is achieved. Those Monitoring of the track length when formatting must be always ensured thus!

Was a surplus sector announced, which caused an exceeding of the max. number of data bytes, is recommended it to read this sector and to include it with the structure of track with these data into the raw formatting. That presupposes

voraus, daß die in ihm enthaltenen Daten programmrelevant sind, was jedoch beim TI bisher nicht aufgetaucht ist.

Anschließend kann der Pufferinhalt auf die Kopiediskette geschrieben werden.

15.2.2.2. Kopieren der Datensektoren

Auch hier spielen die Adreßfelder eine Rolle. Sie legen die Reihenfolge fest, in der die Sektoren gelesen werden müssen (kritisch ist dies allerdings nur bei doppelten Sektoren), wieviel Bytes pro Sektor gelesen bzw. geschrieben werden müssen und welche Sonderbehandlung ggf. notwendig wird. Der letzte Punkt 'Sonderbehandlung' wird erst möglich, wenn neben den Adreßfeldern auch die FDC-Fehlercodes beim 'ins Leere lesen' der korrespondierenden Sektoren protokolliert wurden.

So muß ein Sektor mit einem RNF-Fehler beim Lesen wie beim Schreiben übergangen werden, da ein Leseversuch durch einen Timeout des FDC die Kontinuität des Lesens stören und ein Schreibversuch nachfolgende Sektoren zerstören würde.

Ein Sektor, der einen CRC-Fehler aufweist, muß so geschrieben werden, daß entweder keine oder eine falsche Prüfsumme erzeugt wird. Durch einfaches Aufhören mit dem Datentransfer von Seiten der CPU kann dies jedoch nicht erreicht werden, da der FDC aus Sicherheitsgründen in einem solchen Fall den Sektor mit Nullen auffüllt und die Prüfsumme dann halt unter Beachtung dieser Nullen trotzdem korrekt schreibt.

Das Ergebnis: Daten falsch aber Sektor in Ordnung.

Entweder wird beim Formatieren das CRC-Auslösebyte gleich falsch gesetzt und der Sektor beim Schreiben wie beim Lesen übergangen, oder aber der FDC wird wenige Bytes vor Abschluß des Sektors mit einem sofort

that the data contained in it are program relevant, which did not emerge however with the TI so far.

Subsequently, the buffer contents can be written on the copy diskette.

Copying the data sectors

Address fields play a role here too. They determine the order, in which the sectors must be read (this is critical however only with double sectors), how much bytes per sector read or. to be written must and which special treatment if necessary is required. The last point "special treatment" becomes only possible even if beside the address fields the FDC error codes read with "in emptiness" the corresponding sectors were logged.

So a sector with an RNF error must be ignored during the reading as during the writing, since a read attempt would destroy the continuity of reading by a Timeout of the FDC to disturb and a write attempt following sectors.

A sector, which indicates an CRC error, must be written in such a way that either no or a false check total is produced. This cannot be achieved by simply stopping with the data transfer on the part of the CPU however, since the FDC for safety reasons in such a case fills up the sector with zeros and then stop writes the check total considering these zeros nevertheless correctly.

The result: Data falsely however sector in order.

Either when formatting equal CRC release byte and the sector during the writing as during the reading is falsely set is ignored, or however the FDC is prevented few bytes before termination of the sector with an interrupt which can be

auszuführenden Interrupt am Fertigstellen des Sektors gehindert.

Es sei nochmals gesondert darauf hingewiesen, daß der kontinuierliche Fluß des Ablaufs beim Lesen der Sektoren der Originaldiskette ein primärer Faktor für die Zuverlässigkeit eines Kopierprogramms ist. Doch auch beim Schreiben ist es wichtig, keine unnötigen Pausen zu verursachen. Dies ist insbesondere bei mehrfach vorhandenen Sektoren mit identischen Adreßfeldern wichtig.

15.2.2.3. Prüfung der Datensektoren

Ein Verify ist auch bei einem eigenen DOS kein Luxus, zumal er sehr einfach zu realisieren ist (auch wenn das bei der Programmierung des Moduls TI-IBM Connection wohl anders gesehen wurde).

Nachdem alle AF der Originaldiskette analysiert und die entsprechenden Sektoren zum Kopieren in einen Puffer geholt wurden, geschieht dies identisch auf der Kopie. Dabei kann auf die Adressfeldanalyse verzichtet werden, da deren Struktur durch die Formatierung hinreichend sicher reproduziert wurde. Wurden alle kopierten Sektoren gelesen, folgt der Vergleich der Fehlerprotokolle.

Stimmen diese überein, kann die Anzahl der Datenbytes der Spur berechnet werden (oder der vorher gesicherte Wert verwendet werden) und diese Anzahl Bytes zwischen beiden Pufferzonen verglichen werden. Bei einem Fehler können die Sektorinhalte aus dem Puffer der Originaldisk erneut geschrieben werden.

15.2.3. Spurkopie

Es kann nun aber vorkommen, daß eine Spur absolut keine erkennbare Struktur aufweist, jedoch nachweislich genutzte Daten enthält (z.B. DISkASSEMBLER).

In einem solchen Fall gibt es das Problem der Synchronisation.

executed immediately from finishing the sector.

It is again separately pointed out that the continuous river of the flow is during the reading of the sectors of the original diskette a primary factor for the reliability of a copying program. But also during the writing it is important to cause no unnecessary tracing. This is in particular important with several times available sectors with identical address fields.

Checking the data sectors

A verify is not also with its own DOS luxury, particularly since it is to be implemented very simply (even if with the programming of the module TI-IBM Connection were probably differently seen).

After all AF of the original diskette was analyzed and the appropriate sectors were gotten for copying into a buffer, this occurs identically on the copy. Can be done without the address field analysis, since their structure was reproduced sufficiently surely by formatting. Became all copied sectors read, the comparison of the error logs follows.

When these correspond, the number of data bytes of the track can be calculated (or the beforehand secured value to be used) and this number of bytes between both buffer zones to be compared. In the case of an error sector contents from the buffer of the original disk can again be written.

Track copy

Can now however occur that a track indicates absolutely no recognizable structure, however as can be prove used data contains (e.g. DISkASSEMBLER).

In such a case there is the problem of the synchronisation.

Noch mehr als bei einem Sektorzugriff muß beim puren Lesen einer Spur damit gerechnet werden, daß entweder von vornherein unsauber synchronisiert wird, oder aber der FDC dann, wenn keine Bytes mit MissingClock auftauchen, nach einigen Bytes 'out of sync' ist.

Eine derartige Spur muß also mehrfach gelesen werden, noch besser wäre es, etwa eine Quersumme über den Spurinhalt zu bilden und so oft wiederholt zu lesen, bis die Änderungen dieser Quersumme einen Minimalwert unterschreiten.

Da ein Header einer derart geschützten Spur so aufgebaut sein muß, daß auch das Originalprogramm die Daten lesen kann, ist es sodann durchaus legitim, das Spurbild mit einem eigenen Header (Sync-Bytes, dann z.B. ein DAM oder IDAM, siehe Abschnitt 13, Spurstruktur) zu versehen und auf die Zieldiskette zu schreiben.

Aber Achtung: Hierbei darf man niemals übersehen, daß der FDC beim Spur schreiben, also Formatieren, NICHT DATENTRANSPARENT ist, und einige Bytes nicht so geschrieben werden, wie sie von der CPU kommen (hier liefert das Kapitel 16 weitere Details). Beim Diskassembler etwa befinden sich nur Bytes mit ASCII-Codes unter >7F in der betreffenden Spur.

15.3. Ausblick

Der Wettlauf zwischen Kopierschützern und denen, die den Schutz kopieren oder entfernen, würde Herrn Darwin eine Menge Freude bereiten. So analysieren die Kopierschützer die Kopierprogramme, decken Schwachstellen im Algorithmus auf und passen den Schutz eines neuen Programmes an. Kopierprogramme sind daher verderbliche Ware und erzwingen gewissermaßen ein laufendes Update. Sinnvoller ist es, dem Programm Features mitzugeben, mit deren Hilfe der Anwender selbst Disketten 'von Hand' kopieren kann.

Still more than with a sector access must be counted during the pure reading of a track on the fact that either from the beginning carelessly one synchronizes, or however the FDC then, if no bytes with Missing Clock emerge, after some bytes "out of sync" is.

Such a track must be read thus several times, it would be still better to form and so often repeated read about a checksum over track contents, until the modifications of this checksum fall below a minimum value.

A header of an in such a manner protected track there to be so structured must provide and on the target diskette write that also the original program can read the data, is then quite legitimate it, the track picture with its own header (to Sync bytes, then e.g. a DAM or a IDAM, see paragraph 13, track structure) too.

But note: Here one may never ignore that the FDC with track write, thus a formatting, NOT DATA TRANSPARENCY is, and some bytes to be written not in such a way, like her from the CPU does not come (here Chapter 16 supplies further details). With the disk assembler about only bytes with ASCII codes under >7F are in the track concerned.

View

The race between copy protections and those who copy or remove the protection, became Mr. Darwin a quantity of joy prepares. Thus the copy protector analyzes the copying programs, uncovers weak points in the algorithm and adapts the protection of a new program. Copying programs therefore perishable commodity is and forces to a certain extent a current update. It is more meaningful to give to the program feature with whose assistance of the users themselves can copy diskettes "by hand."

16. Der Floppy-Disk-Controller-Formatter-Chip (FDC)

Der Chip um den sich so viele Geheimnisse ranken soll nun Gegenstand einer eingehenden Betrachtung sein. In diesem Kapitel werden Hinweise zu Funktion und Ansteuerung der verschiedenen auf dem TI eingesetzten FDC-Chips gegeben und im nächsten Kapitel mit Beispielprogrammen illustriert.

16.1. Anatomie einer Diskcontroller-Karte

Eine Diskcontrollerkarte besteht im wesentlichen aus den folgenden Komponenten:

- Dem DSR-ROM mit den Level 2, 1 und 0 Routinen.
- Dem CRU-Interface zur Steuerung der Laufwerke, der Wait-State-Logik, dem DSR-Banking und des FDC.
- Dem FDC selbst.

sowie jeder Menge Leim (das sind die TTLs, die das Ganze zusammen halten).

Das DSR-ROM beinhaltet alle Programme und Programmteile die notwendig sind, ein standardisiertes DOS zu realisieren. Auf dem niedrigsten Level, Level 0, stehen Operationen wie 'Spur anfahren', 'Spur schreiben (formatieren)','Sektor schreiben/lesen' und 'Kopf auf Spur Null fahren' zur Verfügung. Auf Level 1 können z.B. ganze Disketten formatiert, Files kopiert und Pufferzonen reserviert werden. Auf Level 2 findet die ganze Datenverwaltung im logischen Verbund (File) statt.

Das CRU-Interface erlaubt die Anwahl bestimmter Laufwerke, das Umschalten der Schreib-Leseköpfe, die Wahl der Speicherdichte sowie den Start der Laufwerkmotoren und die Freigabe von Head-Load und CPU-Wait-States durch den FDC.

The Floppy Disk Controller Formatter chip (FDC)

The chip around so many secrets to climb should be now the subject of a detailed view. In this chapter notes to function and control of the different FDC chips used on the TI are given and illustrated in the next section with sample programs.

Anatomy of a disk controller card

A disk controller card essentially consists of the following components:

- DSR ROM with the level 2, 1 and 0 routines.
- The CRU interface for the controlling of the drives, the Wait State logic, the DSR Banking and the FDC.
- The FDC itself.

as well as each quantity glue (that are the TTLs, which holds the whole together).

The DSR ROM contains all programs and program sections those is necessary to implement a standardized DOS. On the lowest level, level 0, operations are like "track start," to "track write (format)," "sector write/read" and "heading on track zero drive" at the disposal. On level 1 e.g. whole diskettes can be copied be formatted, files and reserved buffer zones. On level 2 the whole data adminstration in the logical network (file) takes place.

The CRU interface permits the selection of certain drives, a switching of the read-write heads, the selection as well as the start of the drive engines and the release of Head load and CPUWait States to memory density by the FDC.

Der FDC selbst stellt das Bindeglied zwischen Diskstation und Rechner dar, wozu er über eine gewisse 'Eigenintelligenz' verfügt - mit all den Vorbehalten, die dieses Wort in Bezug auf Computer fordert.

Der FDC ist in der Lage, einfache Kommandos wie 'Spur suchen', 'Sektor suchen und lesen/schreiben' usw. auszuführen. Diese oft zeitkritischen Arbeiten werden der CPU abgenommen, die nach Erteilung eines Befehls/Kommandos an den FDC nur noch für die Anlieferung bzw. Abholung von Daten sorgen muß oder im einfachsten Fall nur abwartet, bis der FDC den Befehl abgearbeitet hat und dann die korrekte Ausführung prüft.

Im TI kommen FDC-Chips von Western Digital zum Einsatz, vorwiegend die Typen 1771 (Original Controller), 1772 (Myarc DDCC-1) und 1773 (CorComp Rev.1 und Atronic). Alle FDC besitzen 5 Register, die im DSR-Adreßbereich von >5FF0 bis >5FFE liegen (Myarc: >5F01 - >5F07). Durch die Aufspaltung auf 8 Adressen (nur gerade Byte-Adressen werden verwendet), wird die Software transparenter, da jede Speicheradresse nur für eine Funktion nutzbar ist. Bei Myarc findet keine Aufspaltung statt - alle Ports sind bidirektional, daher nur 4 Bytes, jedoch an ungeraden Adressen. Details zum Myarc-Controller finden sich im ROM-Listing.

16.1.1. Bedeutung der FDC-Register

Mit Einschalten der Floppy-Disk-Steuerkarte auf der ihr zugewiesenen CRU-Seite (allgemein >1100) mit dem SBO 0 Befehl, werden 8 Adressen am oberen Rand des DSR-ROMs zugänglich, die wie folgt belegt sind:

- >5FF0 Status-Register des FDC lesen. Dieses Register kann nur gelesen werden und liefert nach bzw. während einer Operation des FDC Informationen über den Verlauf der Befehlsausführung.
- >5FF2 Spur-Register lesen. Hier findet sich im Normalfall die Nummer der Spur, über

The FDC represents the link between disk station and computer, to which it has a certain "self-intelligence" — with all the reservations, which this word requires regarding computers.

The FDC is able, simple commands like "track looks up," "sector looks up and read/writes," etc.. to execute. These often time-critical work is removed from the CPU, those after distribution of an instruction/a command to the FDC only for the delivery or. Collection of data to ensure must or in the simplest case is only waiting, until the FDC processed the instruction and then the correct execution checks.

In the TI FDC chips of Western digital are predominantly used, the types 1771 (original controller), 1772 (Myarc DDCC-1) and 1773 (CorComp Rev.1 and Atronic). All FDC possesses 5 registers, which are situated in the DSR address range from >5FF0 to >5FFE (Myarc: >5F01->5F07). By the fragmentation in 8 addresses (only even byte displacements used), the software becomes more transparent, since each storage address is usable for a function only. At Myarc no fragmentation takes place - all port are bi-directional, therefore only 4 bytes, however at odd addresses. Details to the Myarc controller are in the ROM listing.

Meaning of the FDC registers

With switching on of the floppy disk control card it assigned the CRU side (generally >1100) with the SBO 0 instruction, 8 addresses at the top margin of DSR ROM to become accessible, which are occupied as follows:

- >5FF0 status registers of the FDC read. This register can be only read and delivers subsequently or. during an operation of the FDC information about the process of the execution.
- >5FF2 track registers read. Here the number of the track is under normal

der sich der Schreib/Lesekopf des gewählten Disklaufwerkes gerade befindet.

- >5FF4 Sektor-Register lesen. Dieses Register enthält die Nummer des Sektors der gelesen bzw. geschrieben wurde oder werden soll.

- >5FF6 Daten-Register lesen. Hier werden während eines Spur-, Adreßfeld- oder Sektorlesebefehls die von der Diskette seriell gelesenen Daten in 8 Bit Parallel-Form an den Rechner übergeben. Liest der Rechner dieses Register bevor ein komplettes Byte von Disk gewandelt wurde, so wird er bis zum Ende der Seriell-Parallel-Wandlung angehalten.

- >5FF8 Kommandoregister schreiben. Ein in dieses Register geschriebenes Byte veranlaßt den FDC zur sofortigen Ausführung des Kommandos, welches das betreffende Byte vorgab. Der Inhalt dieses Registers kann nicht rückgelesen werden!

- >5FFA Spur-Register schreiben. Bei einem Wechsel des Laufwerks muß in dieses Register die Nummer der Spur geschrieben werden, über welcher der S/L-Kopf beim letzten Zugriff auf dieses Laufwerk stehen gelassen wurde. Ein in dieses Register geschriebenes Byte kann an >5FF2 rückgelesen werden.

- >5FFC Sektor-Register schreiben. In dieses Register wird die Nummer des Sektors geschrieben, der als nächster gelesen bzw. geschrieben werden soll. Rücklesbar über >5FF4.

- >5FFE Daten-Register schreiben. Beim Schreiben auf Diskette übergibt die CPU über dieses Register die Daten. Verfrühte Anlieferung wird wie beim Lesen aus >5FF6 mit Wartezyklen der CPU synchronisiert.

conditions, over which the write/read head of the selected disk drive even is.

- >5FF4 sector registers read. This register contains the number the sector read or. written became or will is.

- >5FF6 data pointers read. Here become during track-, address field-, or sector read instructions the data serially read by the diskette in 8 bits to parallel form to the computer transfer. If the computer reads this register before a complete byte by disk was changed, then it is stopped up to the end of the serial parallel transformation.

- >5FF8 command registers write. A byte written into this register arranges the FDC to the immediate execution of the command, which gave the byte concerned. Contents of this register cannot be read back!

- >5FFA track registers write. With a change of the drive the number of the track must be written into this register, over which the R/W head with the last access on this drive are one left. A byte written into this register can be read back at >5FF2.

- >5FFC sector registers write. Into this register the number of the sector is written, as next is read or. to be written is. read back at >5FF4.

- >5FFE data pointers write. During the writing on diskette the CPU transfers the data via this register. Premature delivery is synchronized as during the reading out >5FF6 with wait states of the CPU.

16.1.2. CRU-Interface

Auf der CRU-Seite des Controllers sind bis zu 20 CRU-Bits mit verschiedenen Funktionen belegt. 8 davon sind bei allen in dieser *Disk-Info* behandelten Typen verfügbar, 8 Eingabebits gibt es nur beim TI-Controller und 8 spezielle Eingabebits werden vom CorComp-Controller benutzt. Welche wie belegt sind zeigt die folgende Tabelle (Ausgabebits 8 - 11 nicht bei TI-Controller!).

CRU interface

The CRU page of the controller can contain up to 20 CRU bits with different functions. 8 of the bits apply to all controller types covered in *Disk Info*. 8 input bits gives it only with the TI controller and 8 special input bits are only used by the CorComp controller. The bit assignments are shown in the following table (output bits 8-11 do not apply to the TI controller!).

Relative Bit-Nr.	*'E'in-'A'usgang*	*Beschreibung*		*Relative Bit No.*	*In/ Out*	*Description*
0	A	Schaltet Karte an/aus		0	O	Switch card on/off
1	A	Negative Flanke triggert Laufwerkmotoren Sequenz SBZ 1, SBO 1		1	O	Negative flank triggers drive motors. Sequence: SBZ 1, SBO 1
2	A	Wartezyklen erlauben		2	O	Wait states permit
3	A	Head-Load aktivieren		3	O	Head load activate
4	A	Laufwerk 1 anschalten		4	O	Drive 1 turn on
5	A	Laufwerk 2 anschalten		5	O	Drive 2 turn on
6	A	Laufwerk 3 anschalten		6	O	Drive 3 turn on
7	A	Seiten-/Kopfwahl SBZ 7 Unterseite (normal) SBO 7 Oberseite		7	O	Side/Head select SBZ 7 lower side (normal) SBO 7 upper side
8	A	Laufwerk 4 anschalten		8	O	Drive 4 turn on
9	A	nicht benutzt		9	O	not used
10	A	Dichte umschalten. SBO >A — Single Density SBZ >A — Double Density		10	O	Density switch SBO >A — single density SBZ >A — double density
11	A	Zweite Bank des DSR-ROMs schalten		11	O	Second bank of DSR ROM switch

Folgende Eingabebits stehen nur beim TI-Controller zur Verfügung:

The following input bits are available only with the TI controller:

Bit	*Selbst TI*		*Bit*	*TI only*
0	Head-Load Ausgang des FDC (hier 1771)		0	Head load output of the FDC (here 1771)
1	Sensor, daß LW 1 vorhanden ist. Vor Zugriff auf ein Laufwerk kann dessen Existenz so getestet werden Pegel 1 — nicht angeschlossen Pegel 0 — angeschlossen		1	Sensor that drive 1 is available. Before access to a drive its existence can be tested in such a way: Level 1 — not attached Level 0 — attached
2	Sensor für LW 2		2	Sensor for drive 2
3	Sensor für LW 3		3	Sensor for drive 3
4	Motor-An Sensor. Pegel 1 — aus, 0 — an		4	Motor on sensor. Level 1 — off, 0 — on
5	nicht benutzt, an 5 Volt gelegt		5	not used, at 5 Volt left
6	nicht benutzt, an 5 Volt gelegt		6	not used, at 5 Volt left
7	nicht benutzt, an Masse gelegt		7	not used, at ground left

Mit den folgenden Eingabebits testet der CorComp-Controller die Stellung der 8 DIP-Schalter auf der Karte und ermittelt so die jeweils gewünschte Step-Zeit.

With the following input bits the CorComp controller tests the position of the 8 DIP switches on the card and determines so the step time required in each case.

Bit	*Selbst CorComp*		*Bit*	*CorComp only*
>12	LW 1 Low-Order Step-Bit		>12	Drive 1 low-order step bit
>16	LW 1 High-Order Step-Bit		>16	Drive 1 high-order step bit
>15	LW 2 Low-Order Step-Bit		>15	Drive 2 low-order step bit
>0F	LW 2 High-Order Step-Bit		>0F	Drive 2 high-order step bit
>14	LW 3 Low-Order Step-Bit		>14	Drive 3 low-order step bit
>0E	LW 3 High-Order Step-Bit		>0E	Drive 3 high-order step bit
>13	LW 4 Low-Order Step-Bit		>13	Drive 4 low-order step bit
>11	LW 4 High-Order Step-Bit		>11	Drive 4 high-order step bit

Die etwas chaotische Reihenfolge ergibt sich aus einem möglichst einfachen Layout für den DIP-Schalter an den TMS9901 des CorComp-Controllers. Da die Eingabebits ohnehin aus einer Tabelle heraus abgearbeitet werden, spielt deren tatsächliche Nummer eine eher untergeordnete Rolle. Die Bedeutung der Step-Bits wird bei den Befehlen des FDC erklärt.

The somewhat chaotic sequence results from as simple a layout for the DIP switch as possible to the TMS9901 of the CorComp Controller. Since the input bits are processed anyway from a table, their actual number plays a role rather subordinated. The meaning of the step bits is explained with the instruction of the FDC.

16.2. Befehlsvorrat des FDC

Instruction set of the FDC

Die verschiedenen Befehle, die der FDC versteht, können aufgrund der von ihnen ausgelösten Aktionen in 4 verschiedene Gruppen eingeteilt werden, die mit römischen Ziffern gekennzeichnet werden. Die Befehlstypgruppen sind wie folgt benannt:

The different instructions which the FDC understands can be divided due to the internal messages released by them in 4 different groups, which are marked by Roman digits. The groups of address formats are designated as follows:

Typ I	Steuerung der Kopfposition		Type I	Control of the head position
Typ III	Sektoroperationen		Type II	Sector operations
Typ III	Spuroperationen		Type III	Track operations
Typ IV	Interruptsteuerung		Type IV	Interrupt control

Alle Befehle bestehen — wie der Maschinencode der CPU — aus einem Grundbefehlswort, in dem durch einzelne Bits eine Befehlsspezifikation erfolgt.

All instruction consist — like the machine code of the CPU — of a basic instruction word, in which via individual bits an instruction specification takes place.

Die vollständig definierten Befehle werden über das Kommandoregister direkt an den FDC übergeben, der direkt im Anschluß an diese Übergabe mit der Abarbeitung des Befehls beginnt. Dabei muß sichergestellt sein, daß ein eventuell vorher gegebener Befehl bereits abgearbeitet ist (Test über BUSY im Statusregister, siehe dort). Eine Ausnahme bilden die Typ IV-Kommandos. Nur sie werden in das Kommandoregister übernommen, wenn das BUSY-Bit gesetzt ist. Sie stoppen den aktuellen Befehl und lösen nach dem Eintreten bestimmter Bedingungen einen Interrupt aus, der jedoch im TI nicht direkt ausgewertet wird. Alle anderen Befehlstypen werden während der

The completely defined instruction is transferred via the command register directly to the FDC, which begins directly following this transfer with the processing of the instruction. It must be guaranteed that possibly beforehand a given instruction is already processed (test over BUSY in the status register, see there). The Type IV commands form an exception. Only they are transferred to the command register, if the BUSY bit is set. They stop the current instruction and release after occurring certain conditions an interrupt, which is not analyzed directly however in the TI. All other address formats are not transferred during the handling of an instruction to the command register.

Bearbeitung eines Befehls nicht in das Kommandoregister übernommen.

Leider ist der FDC bei Befehlen, die aufgrund eines 'drive not ready' nicht ausgeführt werden können, etwas nachtragend. Wird in einem solchen Fall nicht über einen Interrupt der FDC rückgesetzt, so 'merkt' sich der Chip den Befehl, und führt in bei nächster Gelegenheit aus. So kann es fatale Folgen haben, wenn eine Formatierung versucht wird, die fehlschlägt und gleich danach mit einem Sektor-Lesebefehl versucht wird, nachzusehen, was denn los war ...

Für die Abarbeitung der Befehle im FDC wird ein controllereigener Takt verwendet. Da das Timing aus diesem Takt abgeleitet wird und bei den Double-Density-Systemen zusätzlich ein über CRU-Bit 10 schaltbarer Teiler vorhanden ist, müssen folgende Mindestwartezeiten zwischen zwei Zugriffen auf Register des FDC beachtet werden (entnommen aus den Datenblättern):

- Kommandoregister schreiben, dann Statusregister lesen: 32 µs
- Register schreiben, dann selbes Register lesen: 16 µs

Die Zeiten verdoppeln sich, wenn ein 1771-Chip angesprochen wird bzw. wenn Single Density aktiviert wurde.

Da die TMS9900 CPU keinen direkten Bit-Test wie z.B. die 6502-CPU auf Speicherinhalte anwenden kann, wird der Sonderfall des direkten Tests von Status-Bit 0 (BUSY-Bit), bei dem nur 24 µs Wartezeit anfallen, hier nur der Vollständigkeit halber erwähnt.

16.2.1. Das Statusregister

Von diesem Register wird auf den folgenden Seiten sehr viel die Redesein. Daher sollen die Bedeutungen der 8 Bits bei den verschiedenen Befehlen vorab erklärt werden, auch wenn die Befehle, die zu diesen Bedeutungen führen, erst

Unfortunately the FDC is entering afterwards with instruction, which cannot be executed due to a "drive not ready," something. One does not reset in such a case over an interrupt of the FDC, thus the chip "notices" the instruction, and executes themselves in at next opportunity. So it can have fatal consequences, if formatting is tried, which fails and is tried equal thereafter with a sector read instruction to check what was the matter ...

For the processing of the instruction in the FDC a controller-owned clock is used. Since the timing is derived from this clock and with the double density systems additionally a divisor adjustable over CRU bit 10 available is, the following minimum waiting periods between two accesses to registers of the FDC must be considered (taken out of the data sheets):

- Command register write, then status register read: 32 µs
- Register write, then same register read: 16 µs

The times doubles themselves, if a 1771 Chip is addressed or if single density was activated.

Since the TMS9900 CPU does not ask direct e.g. the 6502 CPU to memory contents apply can, the special case of the direct test of status bit 0 (BUSY bit), with which only 24 µs waiting period result, here only for the sake of the completeness mentioned.

The status register

Of this register becomes on the following pages very much is spoken of. Therefore the meanings of the 8 bits are to be explained with the different instruction first, even if the instruction, which lead to these meanings, are discussed only

später besprochen werden. Je nach Art des abgearbeiteten Befehls variiert die Bedeutung einiger Bits.

later. Depending upon type of the processed instruction the meaning of some bits varies.

Bit	*Kurz-Name*	*Funktion*	*Function*
7	NOT READY	Laufwerk ist nicht bereit (z.B. Drehzahl nicht in Ordnung)	Drive is not ready (e.g. number of revolutions not in order)
6	WRITE PROTECT	Diskette ist schreibgeschützt. Kann zum gefahrlosen Testen des Schreibschutzes benutzt werden.	Diskette is write protected. Can be used for safe testing of the write protection.
5	HEAD LOADED	Kopf des gewählten Laufwerkes liegt auf Disk	Head of the selected drive on disk
4	SEEK ERROR	Dieses Bit ist nur gültig, wenn das v-Bit im entsprechenden Befehl gesetzt war. Ist dieses Bit log. 1, so wurde die angefahrene Spur nicht verifiziert. Das kann durch einen Fehlschritt des Laufwerkes oder aber durch fehlerhafte Adreßfelddaten bewirkt werden.	This bit is valid only if the v-bit was set in the appropriate instruction. This bit is log. 1, were not verified in such a way the started track. That can be caused by a false step of the drive or however by incorrect address field data.
3	CRC-ERROR	Auch dieses Bit ist nach Typ I Kommandos nur dann gültig, wenn das v-Bit aktiv war. Ist dieses Status-Bit log.1, so wurde während der Verifikation ein defektes AF gelesen.	Also this bit is valid according to type I command only if the v-bit were active. If this status bit is log. 1, then during the verification a defective AF was read.
2	TRACK ZERO	Durchgeschaltete Leitung vom Laufwerk, jedoch im Pegel invertiert.	Connected through line of the drive, however in the level inverts.
1	INDEX-PULSE	Durchgeschaltete Leitung vom Laufwerk, invertiert.	Index pulses connected through line of the drive, inverts.
0	BUSY	FDC ist noch beschäftigt, nicht stören!	FDC is still busy, do not disturb!

16.2.2. Typ II und III Kommandos *Type II and III commands*

Bit			
7	NOT READY	Laufwerk ist nicht bereit	Drive is not ready
6	RECORD-TYPE	1771: siehe Befehl 'Sektor lesen'	1771: see instruction "sector read"
	WRITE PROTECT	Alle: Disk ist schreibgeschützt	All: disk is write protected
5	RECORD-TYPE	1771: siehe Befehl 'Sektor lesen'	1771: see instruction "sector read"
	WRITE FAULT	Alle: Schreibfehler	All: write error
4	RECORD NOT FOUND	Kein AF bzw. gewünschter Sektor nicht gefunden	No address field or desired sector not found

Bit			
3	CRC-ERROR	Daten fehlerhaft (Bit 4=0) oder CRC-Adreßfehler (Bit 4=1)	Data error (bit 4=0), or CRC address error (bit 4=1)
2	LOST DATA	DRQ wurde nicht korrekt bedient, d.h. zu spät Daten an Datenregisters geliefert bzw. geholt	DRQ correctly one did not serve, i.e. too late of at data registers supplied or got
1	DATA REQUEST, DRQ	Datenregister ist voll (Lesebefehl) bzw. geleert (Schreibbefehl)	Data register is full (read command) or empty (write command)
0	BUSY	FDC ist noch beschäftigt	FDC is still busy

16.2.3. Typ I Kommandos des FDC

Wie bereits einleitend bemerkt, finden sich in dieser Befehlsgruppe all jene Kommandos, welche die Position des Schreib-Lesekopfes des angewählten Laufwerkes beeinflussen.

Type I commands of the FDC

As already mentioned in the introduction, all the commands in this group of instructions influence the position of the read-write head of the selected drive.

	7	*6*	*5*	*4*	*3*	*2*	*1*	*0*		
Restore	0	0	0	0	h	V	s1	s2	Kopf über Spur Null fahren	Head over track 0
Seek Track	0	0	0	1	h	V	s1	s2	Kopf über Spur Null fahren	Head over track 0
Step	0	0	1	u	h	V	s1	s2	1 Schritt in letzte Richtung	1 step in last direction
Step In	0	1	0	u	h	V	s1	s2	1 Schritt nach innen	1 step in
Step Out	0	1	1	u	h	V	s1	s2	1 Schritt nach außen	1 step out

Die Options-Bits sind wie folgt definiert (aktiv wenn log. 1):

h:

1771/73: Head-Load-Timing. Aktiviert Head-Load.

1772: Spin-Up-Delay. Bewirkt eine Verzögerung von 6 Umdrehungen der Diskette um die Nenndrehzahl zu erreichen.

v: Verify-On-Track

Ist dieses Bit gesetzt, dann liest der FDC nach Erreichen der gewünschten Spur ein beliebiges

The option bits are defined as follows (actively if log. 1):

h:

1771/73: Head load timing. Activated Head load.

1772: Spin up delay. Causes a delay of 6 revolutions of the diskette around the rated speed to achieve.

v: Verify on Track

If this bit is set, then the FDC reads after achieving the desired track any AF, whose track

AF, dessen Spurkennung er mit dem Inhalt des Track-Registers vergleicht. Der 1771 legt zuvor jedoch noch eine 10 ms lange Pause ein, sofern weniger als 2 Umdrehungen zuvor nicht ein Typ II Kommando kam, dessen e-Bit Null war. Stimmt entweder die Spurnummer oder die CRC-Prüfsumme eines AF mit korrekter Spurnummer nicht, so wird der Befehl automatisch für die Dauer von 5 (2 bei 1771) Umdrehungen der Diskette wiederholt. Wird in dieser Zeit keine passende Spurnummer gefunden, so wird das SEEK-ERROR-Bit im Status-Register gesetzt. Wurde eine korrekte Spurnummer in einem fehlerhaften AF entdeckt, wird das CRC-ERROR-Bit gesetzt.

identifier it compares with contents of the track register. The 1771 however before still another long break inserts 10 ms, if fewer than 2 revolutions a type II command did not come before, whose public exhibition for multimedia was zero. Either the track number or the CRC check total of a AF with correct track number is not correct, then the instruction is repeated automatically for the duration by 5 (2 at 1771) revolutions of the diskette. In this time if no suitable track number is found, then that is set seek error bit in the status register. If a correct track number in an incorrect address field was discovered, the CRC error bit is set.

u: Update-Flag

Wird dieses Bit gesetzt, dann wird der Inhalt des Track-Registers mit jedem Schritt des Kopfes nach innen oder außen inkrementiert bzw. dekrementiert.

u: Update flag

If this bit is set, then contents of the track register with each step of the heading become inward or outside increments or decremented.

s1, s2: Step-Timing-Bits

Diese Bits legen fest, welche Folgefrequenz die Step-Impulse haben, mit denen der Kopf im momentan bearbeiteten Befehl bewegt wird. s1 ist dabei das 'High-Order-Bit' und s2 entsprechend das 'Low-Order-Bit'. Es ergeben sich folgende Step-Zeiten:

s1, s2: Step timing bits

These bits determine, which recurrence rate the step impulses have, with which the heading in the instruction at the moment processed are moved. s1 is thereby the "high order bit" and according to s2 the "low order bit." The following step times result:

s1	*s2*	*Resultierende Wiederholrate*		*s1*	*s2*	*Resulting repeat rate*
0	0	6 Millisekunden		0	0	6 ms
0	1	12 Millisekunden		0	1	12 ms
1	0	20 Millisekunden		1	0	20 ms
1	1	30 Millisekunden		1	1	30 ms

Beim CorComp-Controller alter Bauart (beiges Gehäuse) ergeben sich durch die 2 MHz (statt 1 MHz) Taktfrequenz die doppelten Wiederholraten d.h. die Zeiten sind halbiert. Die Raten für 1772 sind 2, 3, 5 und 6 ms.

With the CorComp controller of old design (beige housing) arise as a result of 2 MHz (instead of 1 MHz) clock frequency the double repeating rates i.e. the times are bisected. The rates for 1772 are 2, 3, 5 and 6 ms.

Die Breite der Step-Impulse ist bei DD (Double-Density) 4 µs und bei SD (Single-Density) 8 µs.

The width of the step impulses is with DD (double Density) 4 µs and with SD (single Density) 8 µs.

16.2.3.1. RESTORE — Kopf über Spur Null bringen

RESTORE — Bring head over track 0

Dies ist der einzige Befehl, mit dem es möglich ist den Aufenthaltsort des Schreib-Lesekopfes über Hardware zu erfahren. Dieser Befehl muß immer dann ausgeführt werden, wenn zum ersten Mal auf dieses Laufwerk zugegriffen wird.

This is possibly the only instruction, with it is the place of residence of the read-write head over hardware to be experienced. This instruction must be executed whenever for the first time this drive one accesses.

Nach Erhalt dieses Befehls prüft der FDC die Leitung TRK 00 des angesprochenen Laufwerks. Ist dieser bereits aktiv, so wird das Track-Register mit >00 geladen und der Befehl beendet.

After receipt of this instruction the FDC checks the line TRK 00 of the addressed drive. If this is already active, then the track register is loaded with >00 and the instruction terminates.

Ist dieses Signal nicht aktiv, so wird die Richtungsleitung DIR auf 'nach außen', also log. 0 gesetzt. Anschließend werden Schritt-Impulse mit der gewählten Wiederholrate so lange über die Leitung STEP ans Laufwerk gegeben, bis die Leitung TRK 00 aktiv wird.

If this signal is not active, then the direction line DIR becomes on "outward," thus log. 0 set. Subsequently, clocks with the selected repeating rate are given so for a long time over the line STEP to the drive, until the line becomes active TRK 00.

Wird nach 255 Impulsen kein TRK 00 gemeldet, dann wird der Befehl abgebrochen und das SEEK-ERROR-Bit im Status Register gesetzt. Dies geschieht nur, wenn das v-Bit im RESTORE-Befehl gesetzt war.

After 255 impulses if no TRK 00 is announced, then the instruction is aborted and seek error bit in the status is set the register. This occurs only, if the v-bit were set in the RESTORE command instruction.

Wird die Spur Null (TRK 00) korrekt erreicht und ist das v-Bit im Befehl gesetzt, so wird ein Verify wie oben besprochen durchgeführt. Das SEEK-ERROR-Bit wird entsprechend gesetzt oder gelöscht. Das Track-Register wird jedoch in jedem Fall auf >00 gesetzt, wenn TRK 00 aktiv wurde.

If the track zero (TRK 00) is achieved correctly and if the v-bit is set in the instruction, then a verify is executed as discussed above. Seek error bit is set accordingly or reset. The track register is set however in each case to >00, if TRK 00 became active.

16.2.3.2. SEEK — Suche Spur

SEEK — search track

Dieses Kommando bewirkt eine software-abhängige Suche nach einer bestimmten Spur. Dabei wird vom FDC angenommen, daß sich die Nummer der Spur, über welcher sich der Kopf des aktuellen Laufwerks gerade befindet, im Spurregister (Track-Register) befindet. Die

This command causes a software-dependent search for a certain track Assumed of FDC that the number of the track, over which the heading of the current drive even is is in the track register. The number of the track which can be started is expected in the data register.

Nummer der anzufahrenden Spur wird im Datenregister erwartet.

Der FDC berechnet selbständig die Differenz, erkennt die notwendige Richtung (DIR) und gibt die erforderliche Anzahl Schrittimpulse mit der in s1 und s2 spezifizierten Wiederholrate an das aktive Laufwerk. Das Spurregister wird dabei so lange inkrementiert bzw. dekrementiert, bis dessen Inhalt mit dem im Datenregister angegebenen Wert übereinstimmt.

!! ACHTUNG !!

Der Programmierer muß hier darauf achten, daß im Extremfall 256 Step-Impulse (bis zum Overflow) gegeben werden können, wenn die Start- oder Zielspur falsch angegeben wurde. Es muß zur Schonung der Laufwerke sichergestellt sein, daß keinesfalls die maximal zulässige Spurnummer überschritten wird!

Je nach Zustand von h und v-Flag wird eine Pause vor Beginn der Ausführung eingelegt bzw. anhand eines AF das korrekte Erreichen der spezifizierten Spur geprüft und im Status-Register protokolliert.

16.2.3.3. STEP — Schritt in letzte Richtung

Da die DIR-Leitung nach Befehlsende auf ihrem letzten Pegel bleibt, kann mit diesem Befehl in die zuletzt eingeschlagene Richtung weiter 'gefahren' werden. Es wird 1 Step-Impuls ausgelöst, die in s1 und s2 angegebene Zeit gewartet und anschließend der Befehl beendet.

War das v-Bit im Befehl gesetzt, wird nach Ablauf der Wartezeit mittels eines AF die Spurnummer im Spur-Register geprüft und ggf. ein SEEK-ERROR gemeldet.

War das u-Bit gesetzt, so wird der Inhalt des Spur-Registers je nach Schrittrichtung in- oder dekrementiert. Durch Setzen des h-Bits kann die Anlaufpause aktiviert werden.

16.2.3.4. STEP IN — Schritt nach innen

The FDC calculated independently the difference, detects the necessary direction (DIR) and gives the necessary number of clocks with the repeating rate specified in s1 and s2 to the active drive. That track register is so for a long time incremented thereby or decremented, until its contents correspond with the value indicated in the data pointer.

!! WARNING !!

The programmer must make certain here that in extreme cases 256 step impulses (up to the overflow) can be given, if the starting or desired track wrongly were indicated. It must be guaranteed for the indulgence of the drive assemblies that under any circumstances the maximally permissible track number is not exceeded!

Depending upon status of h and v-flag a break is inserted before beginning of the execution or on the basis a AF correct achieving of the specified track checked and in the status register logs.

STEP — step in last direction

The DIR line after instruction end on their last level remains there, can be further "driven" with this instruction in the direction taken last to become. 1 step impulse is released, which in s1 and s2 indicated time waited and afterwards the instruction terminates.

If the v-bit was set in the instruction, at flow of the waiting period by means of an address field the track number in the track register is checked and if necessary seek error announced.

If the u-bit was set, then contents of the track register are decremented depending upon step direction or. The start break can be activated by setting the h-bit.

STEP IN — step inward

Nach Erhalt dieses Befehls schaltet der FDC die Leitung DIR auf log. 1 (innen) und gibt einen Schritt-Impuls an das aktive Laufwerk.

Die Optionsbits h und v werden wie bei den zuvor besprochenen Befehlen behandelt, bei aktivem u-Bit wird der Inhalt des Spur-Registers inkrementiert.

16.2.3.5. STEP OUT — Schritt nach außen

Wie STEP-IN, jedoch wird hier DIR auf log. 0 gelegt und bei aktivem u-Bit der Inhalt des Spur-Registers dekrementiert.

16.2.4. Typ II Kommandos des FDC

In dieser Befehlsgruppe finden wir die Sektorlese- und Sektorschreibbefehle. Die Anzahl der bei einem Sektorzugriff zu bewegenden Bytes wird allein durch das Längenbyte im Adreßfeld des gewählten Sektors bestimmt. Der Programmierer hat keine Möglichkeit, mehr Bytes aus einem Sektor zu lesen, als dies das beim Formatieren festgelegte AF vorsieht.

Im Gegensatz zum FDC des Typs 1795/97 ist es beim 1772 oder 1773 nicht möglich, die Interpretation des AF-Längenbytes umzuschalten, womit dann Sektoren 'übergelesen' werden können. Ein Kopierprogramm für einen 1772 oder 1773 ist beim Auftauchen solcher Sektoren an der Grenze der Hardware angekommen.

Die beim 1771 wählbare Interpretation des Längenbytes als Vielfaches von 16 wird weiter unten erwähnt.

Hier nun die Befehlsstruktur:

After receipt of this instruction switches the FDC the line DIR to log. 1 (inside) and gives a clock to the active drive.

The option bits h and v are treated as with the instruction discussed before, incremented with active u-bit contents of the track register.

STEP OUT — Step outward

As step in, however becomes here DIR on log. 0 put and with active u-bit contents of the track register decrements.

Type II commands of the FDC

We find the sector reading and sector write instructions to Type II command of the FDC in this group of instructions. The number with a sector access to moving bytes alone by the length byte in the address field of the selected sector is determined. The programmer does not have to read a possibility, more byte from a sector, than this designates the AF determined with formatting.

In contrast to the FDC of the type 1795/97 it is not possible at the 1772 or 1773 to switch the interpretation of the AF length byte with which then sectors to become "over-read" to be able. A copying program for 1772 or 1773 arrived when emerging such sectors at the boundary of the hardware.

With 1771 selectable interpretation of the length byte as multiple of 16 further down one mentions.

Here now the instruction format:

	7	6	5	4	3	2	1	0	
Read Sector	1	0	0	m	b	e	a1	9	Sektor lesen
Write Sector	1	0	1	m	b	e	a1	a2	Sektor schreiben

Die Optionen sind:

m: Multiple-Record-Flag

Erlaubt das en-bloc Lesen bzw. Schreiben von mehreren Sektoren mit fortlaufenden Nummern. Dazu wird das Sektor-Register laufend inkrementiert. Wird dabei die größte Sektornumer innerhalb der Spur überschritten, so wird der Befehl nach 5 Umdrehungen der Diskette abgebrochen. In diesem Fall wird das RECORD-NOT-FOUND-Bit im Status-Register gesetzt. Weist ein Sektor einen CRC-Fehler auf, so wird der Befehl abgebrochen und das CRC-ERROR-Bit gesetzt. Der Befehl kann bzw. sollte nach Abarbeitung aller gewünschter Sektoren mit einem Interrupt beendet werden um nicht den Timeout abwarten zu müssen. Ist das m-Bit nicht gesetzt, so wird jeweils nur 1 Sektor bearbeitet.

m: Multiple record flag

Permits en-bloc reading or writing of several sectors with sequential numbers. In addition the sector register is constantly incremented. Thereby if the largest sector number is exceeded within the track, then the instruction is aborted after 5 revolutions of the diskette. In this case the RECORD NOT FOUND bit in the status register is set. If a sector indicates an CRC error, then the instruction is aborted and the CRC Error bit is set. The instruction can or should not have to be waiting after processing of all desired sectors with an interrupt to be terminated around the Timeout. Is the m-bit set, then is not processed only in each case 1 sector.

b 1771: Block-Length. Sektorlängen-interpretation.

Bit gesetzt: IBM-Zuordnung, wie 1772, 1773.

Bit nicht gesetzt: Sonderformat.

b 1771: Block length. Sector length interpretation.

Bit set: IBM allocation, like 1772, 1773.

Bit not set: Special format.

AF-Längenbyte	*Sektorlänge in Bytes*		*AF length byte*	*Sector length in bytes*
>00	4096		>00	4096
>01	16		>01	16
>02	32		>02	32
>03	48		>03	48
…				…
>FF	4080		>FF	4080

Man kann also das Längenbyte des AF als Vielfaches der Byteanzahl 16 ansehen, wobei bei >00 ein Übertrag hinzukommt. Ein 'überlesen' eines derart gekennzeichneten Sektors ist jedoch nur in Ausnahmefällen möglich.

1772: Siehe Typ I Kommandos h-Bit.

1773: Side-Compare. Testet das Spurnummerbyte im AF auf den Wert dieses Bits (a1 muß 1 sein, sonst erfolgt kein Test). Die ser Test wird zum Update des SEEK-ERROR-Bits herangezogen.

e: Enable-Delay. Head-Load Beruhigungszeit.

1771: Head-Load-Magneten aktivieren, 10 ms Pause.

1772, 1773: 30 ms Pause vor Befehlsbeginn.

a1 1772: Write-Precompensation-Disable.

Ist dieses Bit aktiv, so wird auf den inneren Spuren so geschrieben wie auf den äußeren. Wird dieses Bit Null gesetzt, so wird auf den inneren Spuren zur Vermeidung von Bit-Verschiebungen ein Versatz von 125 ns zwischen Daten und Takt eingefügt. Diese Option ist nur in DD verfügbar.

a1 1773: Enable-Side-Compare.

Aktiviert Seitentest mit 'b'.

a2 1772, 1773: Data-Address-Mark.

Normalerweise ist dieses Bit nicht gesetzt, weshalb beim Schreiben eines Sektors zu Beginn der Datenzone das Byte >FB geschrieben wird, was einen Sektor mit gültigen Daten anzeigt. Bei gesetztem Bit wird eine Kennung >F8 geschrieben, was ein sog. 'DELETED-DATA-MARK' ist. Beim Lesen eines solchen Sektors wird der Wert dieses Bits im Status-Bit 'Record-Type' übergeben.

One can regard thus the length byte of the AF as multiple of the number of bytes of 16, whereby with >00 a transfer is added. An "overlooked" in such a manner indicated sector is possible however only in exceptional cases.

1772: See type I command h-bit.

1773: Side Compare. The track number byte in the AF tests to the value of this bit (a1 must be 1, otherwise no test takes place). The ser test is consulted for the update of the seek error bits.

e: Enable Delay. Head load damping time.

1771: Head load magnets activate, 10 ms break.

1772, 1773: 30 ms break before instruction beginning.

a1 1772: Write precompensation disable.

If this bit is active, then on the internal tracks one writes in such a way as on the outside. If this bit zero is set, then on the internal tracks for the avoidance of bit shifts a misalignment is inserted by 125 ns between data and clock. This option is available only in DD.

a1 1773: Enable side compare.

Activate side test with "b."

a2 1772, 1773: Data address mark.

Normally this bit is not set, why during the writing of a sector at the beginning of the data zone the byte is written >FB, to what a sector with valid data displays. With set bit an identifier is written >F8, what so-called "Deleted Data Mark" is. During the reading of such a sector the value of this bit will transfer "record type" in the status bit.

a2 1771

Hier werden a1 und a2 gemeinsam betrachtet:

a1	*a2*	*Sektorkennung*	*Data-Mark Typ*	
1	1	>F8	DELETED-DATA-MARK	
1	0	>F9	User-Defined 1	
0	1	>FA	User-Defined 2	
0	0	>FB	Normales DATA-MARK	

Beim Lesen der Sektoren über einen 1771 werden die Bits a1 und a2 in den Bits 6 und 5 des Status-Registers übergeben.

Die Standard-Sektorlängenzuordnung (1772, 1773 oder 1771 mit b-Bit log.1) ist wie folgt:

AF-Inhalt	*Sektorlänge in Bytes*	
>00	128	
>01	256	
>02	512	
>03	1024	

16.2.4.1. Sektor lesen

Die Funktionen, die der FDC beim Lesen eines Sektors ausführt sind vielfältig. Bevor jedoch der Lesebefehl gegeben wird, muß im Sektorregister die Nummer des Sektors abgelegt werden, der gelesen werden soll. Ebenso muß im Spurregister ein Wert stehen, der im AF des betreffenden Sektors zu erwarten ist. Nach Befehlsbeginn sucht der FDC ein AF, dessen Spur- und Sektornummern mit den Registerinhalten übereinstimmen. Vor dieser Aktion werden ggf. spezifizierte Pausen

a2 1771

Here a1 and a2 are regarded together:

a1	*a2*	*Sector identifier*	*Data mark type*
1	1	>F8	Deleted data mark
1	0	>F9	User defined 1
0	1	>FA	User defined 2
0	0	>FB	Normal data mark

During the reading of the sectors over 1771 are transferred the bits to a1 and a2 in the bits 6 and 5 of the status register.

The standard sector length allocation (1772, 1773 or 1771 with b-bit log. 1) is as follows:

AF contents	*Sector length in bytes*
>00	128
>01	256
>02	512
>03	1024

Sector read

The functions that read the FDC during the reading of a sector execute are various. Before however the read instruction is given, the number of the sector must be stored in the sector register, which is to be read. Likewise a value must be located in the track register, which is to be expected in the AF of the sector concerned. After instruction beginning looks up the FDC an AF, whose track and sector numbers correspond with register contents. Before this internal message become if necessary specified pauses

eingelegt.

Wurde ein passendes AF ohne CRC-Fehler gefunden, so wird auf das DATA-ADDRESS-MARK gewartet, nach dessen Eintreffen mit dem Transfer der Daten zur CPU begonnen wird. Dabei muß die CPU entweder laufend das DRQ-Bit testen, ob ein Byte komplett angeliefert wurde (das Datenregister also mit dem Wert des vollen DATA-SHIFT-Registers geladen wurde), oder die CPU greift zu früh zu und wird per Hardware über den DRQ-Pin des FDC in Wartezyklen gezwungen (so beim TI). Wurden alle spezifizierten Bytes gelesen, so wird die CRC-Prüfsumme vom FDC gelesen und mit der inzwischen berechneten verglichen. Bei Übereinstimmung beider Werte ist der Lesevorgang korrekt abgeschlossen, andernfalls wird das CRC-ERROR-Bit im Status-Register gesetzt, woraufhin die CPU den Leseversuch wiederholen kann.

Ein korrektes AF (CRC O.K., Spur und Sektor O.K.) muß beim 1771 innerhalb 2, bei 1772 und 1773 innerhalb 5 Umdrehungen der Diskette gefunden werden. Zusätzlich muß bei 1772 und 1773 das DATA-ADDRESS-MARK bei SD innerhalb 30 (1771 erlaubt nur 28 Bytes) und bei DD innerhalb 43 Bytes nach der CRC-Prüfsumme des zuletzt gelesenen AF gefunden werden. Wird eine dieser Bedingungen verletzt, so wird der Befehl abgebrochen, wobei im Status-Register das RECORD-NOT-FOUND-Bit gesetzt wird.

16.2.4.2. Sektor schreiben

Dieser Befehl ist das logische Gegenstück zum Sektorlesebefehl. Auch hier muß vor Beginn der Abarbeitung das Sektorregister mit der Sektornummer und das Spurregister mit der korrekten Spurnummer geladen werden. Zusätzlich muß der Typ des DATA-MARKs bestimmt werden.

Nun sucht der FDC das korrekte AF mit intakter Prüfsumme. Wurde dieses gefunden, so wartet der FDC die Zeit von 11 Bytes (SD) bzw. 22

inserted.

If a suitable AF without CRC errors was found, then for Data Address Mark one waits, one begins after its arrival with the transfer of the data to the CPU. The CPU must test either constantly the DRQ bit whether a byte was completely delivered (the data pointer was thus loaded with the value of the full Data Shift Register), or the CPU accesses too early and by hardware over the DRQ pin of the FDC in wait states forced (so with the TI). If all specified bytes were read, then the CRC check total of the FDC is read and with calculated the in the meantime compared. Agreement of both values is correctly final reading, otherwise one the CRC error bit in the status register is set, whereupon the CPU can repeat the read attempt.

A correct AF (CRC OK, track and sector OK) must be found at the 1771 within 2, at 1772 and 1773 within 5 revolutions of the diskette. Additionally Data Address mark must with SD within 30 (1771 permitted only 28 bytes) at 1772 and 1773 and with DD within 43 bytes after the CRC check total of the AF read last to be found. If one of these conditions is hurt, then the instruction is aborted, whereby in the status register the RECORD NOT FOUND bit is set.

Sector write

This instruction is the logical counter-part to the sector read instruction. Also here the sector register with the sector number and the track register must be loaded with the correct track number before beginning of processing. Additionally the type of the Data Marks must be determined.

Now the FDC looks up the correct AF with intact check total. If this was found, then the FDC waits the time of 11 byte (SD) or 22 bytes

Bytes (DD) und aktiviert dann, wenn sich bereits ein Byte im Datenregister befindet, die Schreibelektronik. Vor dem eigentlichen Datentransfer werden 6 Bytes (SD) bzw. 12 Bytes (DD) >00 geschrieben, um GAP 2 zu erhalten. Ist dies geschehen, so muß die CPU zu den richtigen Zeitpunkten die Daten im Datenregister >5FFE abliefern, wobei deren Zahl wie zuvor bekannt sein muß. Nach dem letzten Datenbyte schreibt der FDC noch die inzwischen berechnete CRC-Prüfsumme (2 Bytes).

(DD) and activate then, if already a byte is in the data pointer, write electronics. Before the actual data transfer 6 bytes (SD) become or >00 written 12 bytes (DD), in order to receive Gap 2. If this occurred, then the CPU must deliver the correct data pointer to the data in data register >5FFE, whereby their number admits like before be must. After the FDC writes still the CRC checksum calculated in the meantime (2 bytes) to the last data byte.

Das LOST-DATA-Bit im Status-Register wird gesetzt, wenn die CPU auch nur 1 Mal ein Byte zu spät anlieferte. Befand sich nach Ablauf der Wartezeit noch kein von der CPU geliefertes Byte im Datenregister, so wird nichts geschrieben, der FDC beendet den Befehl bereits jetzt und setzt das LOST-DATA-Bit.

The Lost Data bit in the status register is set, even if the CPU delivered only 1 time a byte too late. Still no of the CPU if supplied byte was in the data pointer at flow of the waiting period, then nothing is written, the FDC terminates the instruction already now and sets the Lost Data bit.

Beendet die CPU die Datenübergabe zu früh oder kommt sie in Verzug, so schreibt der FDC ein 0-Byte und läßt diesen Wert in die CRC-Erzeugung normal eingehen.

If the CPU terminates the data transfer too early or if it comes into delay, then the FDC a 0-Byte writes and lets this value CRC production normally enter.

Wurde versucht, auf eine nicht formatierte Diskette zu schreiben, so wird ein Schreibfehler gemeldet.

If one tries to write on an unformatted diskette then a write error is announced.

16.2.5. Typ III Kommandos

Type III commands

Diese Kommandos kommen dann zur Anwendung, wenn Spuren entweder geschrieben, gelesen oder analysiert werden sollen. Folgende Befehle sind verfügbar:

These commands are used if tracks are to be written, read or analyzed either. The following instructions are available:

	7	6	5	4	3	2	1	0	
Read Address	1	1	0	0	0	e	0	0	Adreßfeld lesen
Read Track	1	1	1	0	0	e	0	s	Spurinhalt lesen
Write Track	1	1	1	1	0	e	0	0	Spur schreiben/formatieren

Das e-Bit im Read-Address-Befehl wird wie gehabt interpretiert, beim Read-Track und Write-Track-Befehl des 1771 ist das e-Bit fest auf 1 zu setzen, 1772/73 werten es auch hier aus.

Das s-Bit ist eine Spezialität des 1771. Es erlaubt während des Lesens einer Spur die laufende Neu-Synchronisation nach jedem DATA-ADDRESS-MARK, das ja, wie bereits zuvor beschrieben, einen gewollten Codierungsfehler als Kennung besitzt. Ist das s-Bit gesetzt, so wird NICHT laufend neu synchronisiert, ansonsten wird synchronisiert. Die Typen 1772/73 kennen dieses Bit nicht, sie synchronisieren laufend. Bei ihnen ist s auf Null zu setzen.

16.2.5.1. Adreßfeld lesen

Nach Auslösung dieses Befehls liest der FDC das erste Adreßfeld, das gerade vorbei kommt. Er übergibt Daten an die CPU ähnlich dem Sektorlesebefehl.

Es werden immer 6 Bytes übergeben: Spurnummer, Seitennummer, Sektornummer, Längenkennung und 2 CRC-Bytes.

Wies das gelesene AF einen CRC-Fehler auf, so wird das CRC-ERROR-Bit im Status-Register gesetzt. Nach Ausführung des Befehls befindet sich die Spurnummer (beim 1771 die Sektornummer) des gerade gelesenen AF im Sektorregister. Damit kann die korrekte Datenübergabe an die CPU kontrolliert werden.

16.2.5.2. Spur lesen

Die Abarbeitung dieses Befehls beginnt mit der nächsterreichbaren positiven Flanke des Index-Pulses und endet mit der folgenden positiven Flanke. Alle in dieser Zeit ankommenden Bytes werden über das Datenregister >5FF6 der CPU zur Verfügung gestellt. Die Anzahl der Bytes variiert mit der Laufwerksdrehzahl. Die CPU sollte daher immer mehr Bytes zu lesen versuchen. Bei Timing-Fehlern wird das LOST-DATA-Bit wie gehabt gesetzt, das CRC-ERROR-

The e-bit in the read address command is interpreted as had, with the Read Track and Write Track instruction of the 1771 is the e-bit firmly to 1 to be set, 1772/73 analyzes it also here.

The s-bit is a speciality of the 1771. It permitted during reading a track the current new synchronisation after each Data Address Mark, that, as described already before, an intended coding error as identifier possesses. If the s-bit is set, then NOT constantly again one synchronizes, otherwise one synchronizes. The types 1772/73 do not know this bit, them synchronize constantly. s is to be set to them to zero.

Address field read

After clearing of this instruction reads the FDC the first address field, which comes past even. It transfers data to the CPU similarly to the sector read instruction.

Always 6 bytes will transfer: Track number, side number, sector number, length identifier and 2 CRC bytes.

If the read AF indicated an CRC error, then the CRC Error bit in the status register is set After execution of the instruction the track number (with 1771 the sector number) is even read AF in the sector register. Thus the correct data transfer can be controlled to the CPU.

Track read

The processing of this instruction begins with the next-richable positive flank of the index pulse and ends with the following positive flank. All in this time arriving bytes are made available over the data pointer >5FF6 of the CPU. The number of bytes varies with the drive number of revolutions. The CPU should try to read therefore ever more byte. In the case of timing errors the draw DATA bit is had set as, the CRC Error bit is however without meaning. If the

Bit ist jedoch ohne Bedeutung. Liest die CPU mehr Daten als vorhanden, so entsprechen die 'übergelesenen' Daten dem zuletzt von Diskette gelesenen Byte.

CPU more data than available reads, then the "over-read" data correspond to the byte read last by diskette.

Bei einfacher Dichte werden Adreßfeld-ID, AF-Inhalt und CRC, DATA-ADDRESS-MARK, Daten und CRC richtig gelesen, die GAPs können in der Länge variieren. Entgegen dieser Aussage des Datenblattes ist es jedoch nicht sicher, das DAM (DATA-ADDRESS-MARK) korrekt zu lesen.

With simple density address array ID, AF contents and (carriage return character), Data Address Mark, are read data and (carriage return character) correctly the GAPs can vary in the length. Against this predicate of the data sheet it is however not safe to read the DAM (Data Address Mark) correctly.

Ebenso ist es bei doppelter Dichte (beim TI-DD-Format) entgegen den Datenblattangaben nur möglich, AF-ID, AF-Inhalt und CRC sowie DAM korrekt zu lesen. Die Daten der Sektoren sind in der Regel ab etwa dem zwanzigsten Byte fehlerhaft.

Likewise it is only possible with double density (with the TI DD format) against the data sheet specification, to read AF-ID, AF contents and CRC as well as DAM correctly. The data of the sectors are usually incorrect starting from for instance the twentieth byte.

16.2.5.3. Spur schreiben/formatieren

Track write/format

Das ist der Befehl, der es unmöglich macht ein ideales Kopierprogramm zu schreiben. Wäre es möglich, die Daten so in die Spur zu schreiben, wie sie der Lesebefehl las, dann wäre es Essig mit Kopierschutz. Doch leider - Absicht oder nicht - schreibt der FDC nicht alle Daten so, wie er sie geliefert bekommt. Und das hat leider gute Gründe.

This is the instruction, it makes ideal a copying program impossible to write. If it would be possible to write the data in such a way into the track how it read the read instruction, then it would be vinegar with copy protection. But unfortunately — intentionally or not — the FDC does not write all data in such a way, as it gets supplied. And that has unfortunately good reasons.

Doch zuerst zur Funktion.

But first to the function.

Wie beim Spur-Lesebefehl beginnt die Abarbeitung mit der nächstfolgenden positiven Flanke des Index-Pulses und endet mit der darauf folgenden positiven Flanke. In der Zwischenzeit übergibt die CPU dem FDC die zuvor bereitgestellten Daten

As is the case for the track read instruction processing begins with the next positive flank of the index pulse and ends with the positive flank following on it. In the meantime the CPU transfers the data supplied before to the FDC.

Die Status-Bits LOST-DATA und WRITE-FAULT sowie WRITE-PROTECT werden wie beim Sektorschreibbefehl verwendet, jedoch ist RECORD-NOT-FOUND sinnvollerweise nicht gültig.

The status bits Lost Data and Write Fault as well as Write Protect are with the sector write instruction used, however is for good reason not valid Record Not Found.

Je nach gewählter Speicherdichte werden einige

Depending upon selected memory density some

Bytes nicht so geschrieben, wie sie ins Datenregister gebracht werden. Wie die Interpretation im Einzelfall aussieht, zeigt die Tabelle:

bytes are not written in such a way, how they are brought in the data pointer. How the interpretation looks in individual cases, is shown in the table:

Deutsch

Byte	*Interpretation Single Density*		*Double Density*
	1771	*1772/1773*	*1772/1773*
>00 - >F4	Byte schreiben, Takt >FF	Byte schreiben, Takt >FF	Byte schreiben
>F5	Byte schreiben, Takt >FF	Nicht erlaubt !	>A1 schreiben, p
>F6	Byte schreiben, Takt >FF	Nicht erlaubt !	>C2 schreiben
>F7	2 CRC-Bytes schreiben	2 CRC-Bytes schreiben	2 CRC-Bytes schreiben
>F8 - >FB	Byte schreiben, Takt >C7 (DAM) p	Byte schreiben, Takt >C7, p	Byte schreiben
>FC	Byte schreiben, Takt >D7 (IAM)	Byte schreiben, Takt >D7	Byte schreiben
>FD	Nicht erlaubt !	Byte schreiben, Takt >FF	Byte schreiben
>FE	Byte schreiben, Takt >C7 (IDAM) p	Byte schreiben, Takt >C7, p	Byte schreiben
>FF	Byte schreiben, Takt >FF	Byte schreiben, Takt >FF	Byte schreiben

English

Byte	*Interpretation Single Density*		*Double Density*
	1771	*1772/1773*	*1772/1773*
>00 - >F4	Byte written, clock >FF	Byte write, clock >FF	Byte written
>F5	Byte written, clock >FF	Not permitted	>A1 written, p
>F6	Byte written, clock >FF	Not permitted	>C2 written
>F7	2 CRC bytes written	2 CRC bytes written	2 CRC bytes written
>F8 - >FB	Byte written, clock >C7 (DAM) p	Byte written, clock >C7 p	Byte written
>FC	Byte written, clock >D7 (IAM)	Byte written, clock >D7	Byte written
>FD	Not permitted	Byte written, clock >FF	Byte written
>FE	Byte written, clock >C7 (IDAM) p	Byte written, clock >C7 p	Byte written
>FF	Byte written, clock >FF	Byte written, clock >FF	Byte written

'p' als Index gibt an, daß der CRC-Generator ab dem folgenden Byte mit der Berechnung der Prüfsumme beginnt. Zu jedem Zeitpunkt kann die seit Übergabe dieses Bytes berechnete Prüfsumme durch Übergabe von >F7 geschrieben werden. Beim Aufbau des Spurbildes (vor dem Schreiben!) muß daher auf korrekten Abstand zwischen Start des CRC-Generators und dessen Auswertung geachtet werden. Nach Auslösen der 2 CRC-Bytes muß der CRC-Generator neu gestartet werden.

'DAM' DATA-ADDRESS-MARK

Dieses Byte ist das erste Byte eines Sektors und kennzeichnet den Beginn einer Datenzone. Es wird beim Lesen eines Sektors nicht mit den Sektordaten übergeben.

"p" as index indicates that the CRC generator begins at the following byte with the calculation of the check total. At each point in time the check total calculated since transfer of this byte can be written by >F7. With the structure of the track picture (before writing!) therefore its analysis must be paid attention to correct distance between start of the CRC generator and. After releasing the 2 CRC bytes the CRC generator must again be started.

Data Address Mark (DAM)

This byte is the first byte of a sector and indicates the beginning of a data zone. It will not transfer during the reading of a sector with the sector data.

'IAM' INDEX-ADDRESS-MARK

Bei den auf TI-Disketten anzutreffenden Spurstrukturen findet dieses Byte keine Anwendung. Es kann jedoch vor dem ersten Sektor einer Spur (genau: vor dessen AF) eingefügt werden, um die Synchronisation beim Spur-Lesen zu verbessern.

'IDAM' ID-ADDRESS-MARK

Ist das Kennbyte eines AF.

16.2.6. Typ IV Kommandos

Genau genommen handelt es sich hier nicht um mehrere Kommandos, sondern um nur eines, das jedoch 4 Optionen besitzt.

Index Address Mark (IAM)

With the track structures which can be found on TI diskettes this byte does not apply. It can however before the first sector of a track (exact: before its AF) to be inserted, in order to improve the synchronization with track reading.

ID Address Mark (IDAM)

The identification byte for an address field.

Type IV commands

Exact taken concerns it here not around several commands, but only one, which possesses however 4 options.

Name	*Bit-Nr. im Befehlswort*								*Funktion*
	7	*6*	*5*	*4*	*3*	*2*	*1*	*0*	
Force Interrupt	1	1	0	1	i1	i2	i3	i4	Interrupt auslösen

Mit diesem Befehl kann ein momentan in Arbeit befindlicher Befehl abgebrochen und der FDC in einen definierten Zustand gebracht werden.

Dieser Befehl sollte angewandt werden, wenn entweder eine Power-Up-Initialisierung durchgeführt wird oder wenn der Controller mit einem falschen Befehl durcheinander gebracht wurde.

Die Spezifikationsbits i1 bis i4 erlauben es, nach Eintreten bestimmter Bedingungen einen Prozessor-Interrupt auszulösen. Von dieser Möglichkeit wird beim TI kein Gebrauch gemacht (auch jeder andere Befehl an den FDC löst nach Beendigung einen Interrupt aus).

Die Spezifikation der Interrupt-Bedingung ist wie folgt möglich:

With this command an instruction momentarily in progress can be broken off and the FDC brought into a defined condition.

This instruction should be applied, if either a power-up initialization is executed or if the controller were brought in disorder with a false instruction.

The specification bits i1 to i4 make it possible to release after occurring certain conditions a processor interrupt. With the TI no use is made by this possibility (also every other instruction to the FDC releases an interrupt after termination).

The specification of the interrupt condition is possible as follows:

Bit Funktion

- i1 alle : Sofortiger Interrupt
- i2 alle : Interrupt nach nächstem Index-Puls
- i3 1771/73: Interrupt nach Betriebs-bereitschaft des Laufwerks (READY wechselt von Low nach High).
  1772 : nicht ausgewertet.
- i4 1771/73: Interrupt mit Ende der Betriebsbereitschaft des LW (READY wechselt von High nach Low). 1772 : nicht ausgewertet.

Ist kein Bit gesetzt, so wird kein Interrupt ausgelöst, der Befehl jedoch ausgeführt.

Damit sind alle Befehle der verschiedenen FDC in Disk-Controllern für den TI-99/4A beschrieben. In der Software zum Original-TI-Controller wird nicht von allen Möglichkeiten Gebrauch gemacht, die der 1771 den anderen voraus hat.

Das nächste Kapitel zeigt anhand von Beispielprogrammen, wie der Programmierer den FDC mit den beschriebenen Befehlen dazu veranlassen kann, bestimmte Dinge zu tun.

Im Anschluß daran wird mit Auszügen aus den FDC-relevanten Passagen verschiedener DSR-ROMs von Diskcontrollern illustriert, wie die Funktionen in vorhandener Software realisiert sind.

Bit Function

- i1 everything: Immediate interrupt
- i2 everything: Interrupt after next index pulse
- i3 1771/73: Interrupt after ready status of the drive (READY changes from Low to High).
  1772: not analysed.
- i4 1771/73: Interrupt with end of the ready status of the drive (READY changes from High to Low).
  1772: not analysed.

If no bit is set, then no interrupt is released, the instruction is however executed.

Thus all instruction of the different FDC are in disk controllers for the TI-99/4A described. In the software to the original TI controller by all possibilities use is not made, which has the 1771 the other one ahead.

The next chapter shows certain things on the basis sample programs, as the programmer can arrange the FDC with the described instruction to it, to do.

Subsequently, illustrated with single dumps from the FDC relevant passages of different DSR ROMs of disk-controller, how the functions are implemented in available software.

17. Ansteuerung des FDC

Dieses Kapitel ist gewissermaßen eine Einstimmung auf Kapitel 18ff, in denen sich ausführlich kommentierte ROM-Listings zu den vier gängigsten Disk-Controllern finden.

17.1. Vorbereitende Arbeiten

Wie bereits im Kapitel 10 anhand des Prozessor-Interface angesprochen, ist der FDC nur bedingt in der Lage, die angeschlossenen Laufwerke zu steuern — er kann immer nur das bedienen, welches ihm die CPU zuteilte. Die CPU muß also dafür sorgen, daß das korrekte Laufwerk angeschaltet wurde. Auch die Seite (der Kopf) des Laufwerkes muß korrekt vorgewählt werden, da die eingesetzten FDC-Typen diese Leitung nicht verwalten (es gibt welche, die das können, z.B. 2795-02 und 2797-02). Zudem muß bei den 1772/73-Typen die Einhaltung der Head-Load Beruhigungszeiten per Software sichergestellt werden und bei allen außer 1771 die notwendige Codierungsart (FM oder MFM, Single oder Double Density) korrekt eingestellt sein.

Erst wenn diese Optionen von Seiten der CPU aktiviert wurden, kann dem FDC der erste Befehl gegeben werden.

17.2. Allgemeiner Ablauf der Ansteuerung

17.2.1. Betriebsbereitschaft des Laufwerks

Bevor ein Befehl an den FDC übergeben wird, muß das angewählte Laufwerk betriebsbereit sein, d.h. die Nenndrehzahl des Antriebs muß erreicht sein. Der 1772 kann diese Aufgabe übernehmen, 1771 und 1773 benötigen eine entsprechende Programmierung. Dem Laufwerksmotor muß ca. 1 Sekunde Zeit gegeben werden, auf die Nenndrehzahl zu kommen.

Daher muß vor Befehlsübergabe das READY-Bit des Statusregisters getestet und ggf. eine

Control of the FDC

This chapter is to a certain extent a joining in for Chapter 18, in which in detail commentated ROM listings to the four most usual disk controllers yourself to find.

Preparatory work

As already in Chapter 10 on the basis the processor interface addressed, the FDC is only conditioned able, to control which attached drives — it can serve in each case that, which it the CPU assigned. The CPU must ensure thus that the correct drive was turned on. Also the page (the heading) of the drive must be preselected correctly, since the assigned FDC types do not administer this line (there are which, which can do that, e.g.. 2795-02 and 2797-02). Besides the observance the head load damping times must be guaranteed by software and be correctly adjusted with all except 1771 the necessary type of coding (FM or MFM, single or double density) with the 1772/73 types.

Only if these options were activated on the part of the CPU, the first instruction can be given to the FDC.

General operational sequence of the control

Ready status of the drive assembly

Before an instruction to the FDC is transferred, must the selected drive be ready for use, i.e. the rated speed of the drive must be achieved. The 1772 this function can take over, 1771 and 1773 needs an appropriate programming. The drive motor must approx. 1 second time to be given to come on the rated speed.

Therefore the ready bit of the status register must tested and if necessary before instruction

Warteschleife ausgeführt werden.

transfer. a wait loop to be executed.

17.2.2. Beispielprogramm

17.2.2.1. Example program

```
DEMO1 LI   R12,>1100   CRU-Basis Diskcontroller
      SBO  0           Karte an und FDC-Register einblenden
      SBO  4           z.B. Laufwerk 1 anschalten
      SBZ  1           Laufwerkmotoren anwerfen bzw. bereits
      SBO  1           laufende 'bei Laune' halten
      MOVB @>5FF0,R0   Statusregister lesen
      JLT  READY       Bei gesetztem höchsten Bit ist das LW
*                      bereits auf Nenndrehzahl
      LI   R0,DELAY    Warteschleife für ca. 1 Sekunde
LOOP1 SWPB R0          Dummy-Befehl
      SWPB R0          macht gar nichts
      DEC  R0          Wert reduzieren
      JNE  LOOP1       bis auf Null
READY MOVB @CMD,@>5FF8 Kommando ins Kommandoregister
      SWPB R0          Wartezeit vor nächstem Zugriff auf FDC
      SWPB R0          dto.
      SBO  3           Head-Load aktivieren
      RT               Ende
```

18. ROM-Listings der gängigen Diskcontrollerkarten

18.1. Vorbemerkungen

Mit den auf den folgenden Seiten abgedruckten kommentierten ROM-Listings wird die Behandlung des Themas 'Floppy-Disk-Ansteuerung' abgeschlossen. Wie bereits zuvor angemerkt, sind die Listings nicht komplett, sondern umfassen nur die Teile, die mit dem FDC direkt oder indirekt zusammenhängen. Der interessierte Leser kann die kompletten Listings beim Autor erhalten.

Zu diesen Listings sei hier nur folgendes als wichtige Anmerkung festgehalten:

Obwohl in Kapitel 3 auf Seite 28 von einer genormten Softwareschnittstelle bezüglich der Disk-DSR gesprochen wurde, existiert diese Normung de facto nicht. TI hat es seinerzeit bei einer eher globalen Definition des nutzbaren PAD-Workspace durch eine DSR bewenden lassen (nachzulesen in den holländischen Schemablaaden) und mit der Erstellung der TI-Diskcontroller DSR quasi einen Standard geschaffen, der später lediglich interpretiert wurde. So ist verständlich, daß in Ermangelung einer schriftlichen Vorschrift für die Nutzung des VDP-RAMs, durch Diskcontroller anderer Hersteller fast zwangsläufig Inkompatibilitäten entstanden. Schließlich war das TI-Urmuster nur für 3 Laufwerke und lediglich auf Single-Density ausgelegt (obwohl man die Diskkapazität recht ordentlich dimensionierte). Doch muß man TI in jedem Fall den Vorwurf machen, das DSR-Konzept für den Diskcontroller nicht zu Ende gedacht zu haben (damals war die CES daran schuld, welche die Soft- und Hardwerker in Druck brachte). So ist es zwar möglich, prinzipiell beliebig viele Diskcontroller im System zu haben (allerdings sollte man dann nie CALL FILES ausführen!), es ist aber in keinem Fall feststellbar (da nicht vorgesehen), herauszubekommen, von wo (CRU-Basis des Controllers) ein Programm geladen

ROM listings of the usual disk controller cards

Preface

With the commentated ROM listings printed on the following pages the handling of the topic "floppy diskette control" are locked. As already before marked, the listings are complete, but do not cover only the sections, which are connected directly or indirectly with the FDC. That the complete listings can receive interested readers with the author.

To this listings here only the following is held as important note

Although in Chapter 3 on page 28 of a standardized software interface concerning the disk DSR one spoke, this standardization does not exist in fact. TI has it at that time with a rather global definition of the usable PAD Workspace by a DSR bewenden to leave (to reread into the Dutch Schemablaaden) and with the creation of the TI Disk controller DSR quasi a standard created, which was only interpreted later. Like that it is understandable that in the absence of a written regulation for the use of the VDP RAM, by disk controllers of other manufacturers almost inevitably incompatibilities developed. Finally the TI sample was only for 3 drives and only for single density appropriate (although one dimensioned the disk capacity quite properly). But one must make the reproach for TI in each case at that time the DSR concept for the disk controller end to have thought (the CES was to it debt, which brought the soft and hard workers in printing). Like that it is possible to have as many as desired always disk controllers in the system (however one should then never execute CALL FILES!), it can be recognized however in no case (there not designated), out-get from where (CRU base of the controller) a program was loaded.

wurde.

18.1.1. Neuigkeitswert der ROM-Listings

Das TI-ROM-Listing wurde im Zusammenhang mit der Entwicklung eines softwareverträglichen 80-Track-DOS Anfang 1987 erstellt. Mit Erscheinen des *Technical Drive* Buch von Monty Schmitt ist eine zweite 'Auslegung' des DOS verfügbar. Wie bei allen ROM-Listings, so ist insbesondere dann, wenn mit Sachkenntnis vorgegangen wurde, das Ergebnis in der Regel gleich bis identisch. Das liegt in der Natur der Sache und ist nicht zu ändern. Gleichwohl kann daraus nicht abgeleitet werden, es sei abgeschrieben worden!

Das Atronic-ROM-Listing entstand um die Jahreswende 1985/86, und diente dem Autor als Vorlage zur Erstellung der anderen Ausführungen. Dieses DOS zeichnet sich durch die klare Programmierung aus, der zwar der letzte Schliff fehlt, die jedoch den Neuling am wenigsten verwirren wird. Nach derzeitigem Wissensstand ist es das einzige ROM-Listing zu diesem Controller, der über lange Jahre seine enorme Zuverlässigkeit und Softwareverträglichkeit bewiesen hat.

Das CorComp-Listing bezieht sich auf die zweite Version der DSR-ROMs von Millers Graphics, also die zweite mit eigenem Titelbild. Durch die Vielzahl der Optionen mag der sofortige Überblick bisweilen auf sich warten lassen, doch wird gezeigt, was man mit relativ wenig Aufwand im Atronic-Controller noch hätte realisieren können.

Das ROM-Listing zum Myarc DDCC-1 (alt) schließlich ist, nach Ansicht des Autors, das Paradebeispiel dafür, wie man es NICHT machen sollte. Speicherintensive Verwaltung fast nutzloser Daten, die Verwendung redundanter Flags sowie die 'Fast-Schon-Endlosschleife' bei einem Lost-Data-Zustand beim Sektor-I/O (mit welcher der Autor manche

Piece of news value of the ROM listings

The TI ROM listing was created in connection with the development of a software-compatible 80-track DOS at the beginning of 1987. With appearance of the *Technical Drive* book of Monty Schmitt a second "interpretation" of the DOS is available. As is the case for all ROM listings, then in particular if with expertise one proceeded, the result is usually alike to identical. That is situated in the nature of the thing and is not to be modified. Nevertheless from it, it cannot be derived was copied!

The Atronic ROM listing developed around the turn of the year 1985/86, and served the author as collecting main for the creation of the variations in type. This DOS is characterised by the clear programming, to which the final touch is missing, which will confuse however the beginner to few. After present knowledge status it is the only ROM listing to this controller, which proved its enormous reliability and software compatibility over long years.

The CorComp listing refers to the second version that of DSR ROM of Millers Graphics, thus second with own frontispiece. Sometimes by the multiplicity of the options the immediate overview may take time, but is shown, what one could still have implemented at relatively few expenditure in the Atronic controller.

Finally the ROM listing to the Myarc DDCC-1 (old) is, in opinion of the author, the prime example for it as one should not make it. Memory-intensive administration of almost useless data, the use of redundant flags as well as the "almost careful lot loop" with a Lost Data status with the sector I/O (with which the author had to make some bad experience) and that the

schlechte Erfahrung machen mußte) und der fast chaotisch zu nennende Umgang mit dem VDP-RAM sollten der Erheiterung, keinesfalls aber als Vorbild dienen.

Nur das TI- und das Atronic-Listing stellen den letzten Stand der Softwareentwicklung des jeweiligen Herstellers dar (in beiden Fällen gestoppt durch Produktionsschluß), zu den beiden anderen Typen gibt es mittlerweile z.T. mehrfach revidierte DOS-Versionen.

Der Leser, der über ein CPS99 oder einen neueren Controller aus USA verfügt, sollte mit den hier verfügbaren Hilfestellungen durch die kommentierten alten Versionen in der Lage sein, selbst ein Listing der neuen Generation zu erstellen. Es muß jedoch angemerkt werden, daß die Unterschiede seltener den FDC-Bereich als Dinge wie Hilfsroutinen u.ä. betreffen.

18.1.2. Hardwarebedingte Unterschiede

Die DD-Systeme (Double Density) steuern ihren jeweiligen FDC im 'Klartext' an, der TI-Controller verwendet einen FDC mit invertiertem Datenbus (siehe auch Kapitel 16). Beim Vergleich der FDC-Befehle im Listing mit denen in Kapitel 14 ist auf diesen Umstand zu achten.

Durch den bei DD auftretenden hohen Datendurchsatz ist es nicht mehr möglich, die Ansteuerung des FDC beim Datentransfer aus dem Wait-State-ROM des Diskcontrollers auszuführen. Die DD-Systeme kopieren daher die entsprechenden Routinen kurz vorher ins PAD-RAM, das sie zuvor im eigenen System-RAM sicherten. Das hierzu benutzte System-RAM dient daneben der Verwaltung der letzten Spur eines jeden Laufwerks, sowie der Speicherung anderer Informationen. Besonders intensiv wird das System-RAM vom Myarc-Controller benutzt, äußerst sparsam geht man bei Atronic damit um.

Alle ROM-Listings der DD-Systeme entstammen einem Speicherbereich, der nicht über das

VDP RAM, which can be called almost handling chaotically, should this amusement under no circumstances however as model serve.

Only the TI and the Atronic listing represent the last status of the software development of the respective manufacturer (in both cases stopped by production conclusion), to the two other types give it to meanwhile partly several times revised DOS versions.

The reader, that has a newer controller from the USA a CPS99 or should be with the here available assistance by the commentated old versions able, even a listing of the new generation create. It must be marked however that the differences more rarely concern the FDC range than things such as auxiliary routines etc.

Hardware-conditioned differences

The DD systems (double density) head for their respective FDC in the "plain text," which uses TI controllers a FDC with inverted data bus (see also Chapter 16). With the comparison of the FDC instruction in the listing with those in Chapter 14 is to be paid attention to this circumstance.

By the high information flow-rate occurring with DD is no longer possible it to execute the control of the FDC with the data transfer from the wait state ROM of the disk controller. The DD systems copy therefore the appropriate routines briefly beforehand in PAD RAM, which they protected before in the own system RAM. System RAM for this used serves beside it the administration of the last track of each drive, as well as the storage of other information. That system RAM of the Myarc controller used, extremely economically deals particularly intensively one with Atronic with it.

All ROM listings of the DD systems come of a storage area, which is not connected over the

normale DSRLNK zugeschaltet wird. Alle 3 verfügen im Gegensatz zum TI-Controller über ein ROM, das in zwei Bänke aufgespalten ist. Die zweite Bank wird über ein CRU-Bit an- und abgeschaltet. Bei Myarc bleibt dabei das RAM an der gleichen Stelle, wohingegen bei Atronic und CorComp das RAM erst in der zweiten Bank verfügbar ist.

18.1.2.1. Identifikation der Controllerkarten

Für bestimmte Anwenderprogramme kann es recht nützlich (bisweilen lebenswichtig, z.B. Kopierschutz) sein, den genauen Typ eines Diskcontrollers zu ermitteln. Die Erkennung des FDC-Chips wird im entsprechenden Kapitel behandelt. Zur Kartenerkennung gibt es folgende Methoden:

Identifikation des DSR-ROMs

Das ist sicher die schlechteste aller Methoden. Hier wird ein indirektes Verfahren angewandt, wobei aus einer Soft- auf eine Hardware geschlossen wird. Die Abfrage eines bestimmten Bytes, egal welches es nun ist, funktioniert nur dann, wenn bei deren Programmierung sicher ist, daß sich ALLE Controller in diesem Byte unterscheiden und alle zukünftigen dies auch tun werden. Somit dürfte klar sein, daß es so nicht geht.

Eine Prüfsumme ist auch nicht besser, da eine leicht modifizierte DSR (Offset des ersten Datensektors, geänderte Retry-Zahl etc.) ja nicht bedeutet, man habe einen anderen FDC im System.

Analyse des CRU-Mappings

Das klappt nur auf Systemen, welche über Input-Bits verfügen. Man kann zwar erkennen, ob das Paging-Bit (>B oder >3) das ROM umblendet, aber auch wieder nur über Bytevergleiche oder Prüfsummen, zudem kann man dann nur TI oder Nicht-TI ermitteln.

Zu den Inputs: Der TI-Controller erkennt damit

normal DSRLNK. All 3 has in contrast to the TI controller a ROM, which is split up into two banks. The second bank is switched on and off over a CRU bit. With Myarc thereby RAM in the same place remains, whereas with Atronic and CorComp RAM is available only in the second bank.

Identification of the controller cards

For certain user programs can be it quite useful (sometimes vital, e.g. copy protection) to determine the exact type of a disk controller. The recognition of the FDC chip is treated in the appropriate chapter. For card recognition there are the following methods:

Identification of DSR ROMs

This is reliably the worst of all methods. Here an indirect procedure is applied, whereby from a soft- or hardware one judges. The query of a certain byte, all the same which it now is, functions then only if it is safe with their programming that ALL controllers will also do all future this with respect to this byte to differentiate and. Thus it might be clear that it does not go in such a way.

A check total is not also better, there an easily modified DSR (offset of the first data sector, modified Retry number of etc.) not meant, one has another FDC in the system.

Analysis of the CRU mappings

That works only on systems, which have input bits. One can detect whether the Paging bit (>B or >3) the ROM around inspection, in addition, again only over byte comparisons or check totals, besides one can determine then only TI or not TI.

To the inputs: The TI controller does not detect

die Präsenz eines jeden Laufwerks, CorComp und Myarc steuern damit (und über DIP-Schalter auf der Karte) die Step-Zeiten der Laufwerke und der Atronic hat keine Inputs. Da aber nicht eindeutig erkannt werden kann, ob z.B. alle DIPs ON sind, also NULL liefern oder ob gar kein Input da ist, fällt auch diese Methode aus.

thereby the operational readiness level of an each drive, CorComp and a Myarc controls thereby (and over DIP switches on the card) the step times of the drives and the Atronic has inputs. Since however unique it cannot be detected whether e.g. all DIPs is ON, thus ZERO to supply or whether no input is there, it precipitates also this method.

Analyse des System RAMs

Analysis of system RAMs

Ist schon besser, bzw. genau das, worauf es hinauslaufen sollte. Gibt es nach Umschalten von CRU-Bit >B immer noch kein RAM, ist's ein TI-Controller, liegt es vor und nachher auf >5000 als 8-Bit-RAM vor, so handelt es sich höchstwahrscheinlich um einen Myarc. Liegt 4-Bit-RAM ab >4000 vor, so ist es wahrscheinlich ein CorComp, sonst ein Atronic.

Is already better, or exactly that, on which it should run out. There is after switching CRU bits >B still no RAM, it is a TI controller present, forwards and afterwards on >5000 as 8-bit RAM, then it concerns most likely a Myarc. If 4-bit RAM is present off >4000, then it is probably a CorComp, otherwise an Atronic.

Doch auch diese Methode scheitert in bestimmten Situationen. Im Kapitel 15 wurde bei der Besprechung der FDC-Programmierung ein Verfahren gezeigt, welches in Zusammenhang mit den hier genannten Hardwareanalysen das fast 100%ige Erkennen des FDC-Typs und damit des ControllerBoards erlauben.

But also this method fails in certain situations. In Chapter 15 with the discussion of FDC programming a procedure was shown, which in connection with the hardware analyses that specified here almost 100% detecting the FDC type and thus the controller board to permit.

18.1.3. Unterschiede in der Software

Differences in the software

Die Unterschiede zwischen den 3 ersten Systemen (TI, Atronic, CorComp) sind recht gering. Insbesondere beim Vergleich Atronic-CorComp fallen fast identische Sequenzen auf. Wer über die kompletten ROM-Listings, also auch des Level-2-Bereiches verfügt, wird noch mehr Gemeinsamkeiten finden. Lediglich Myarc ist auch hier die (unrühmliche) Ausnahme.

The differences between the 3 first systems (TI, Atronic, CorComp) are quite small. In particular with the comparison Atronic-CorComp are noticeable almost identical sequences. Who has the complete ROM listings, thus also the Level-2 ranges, still more thing in common will find. Only Myarc is (inglorious) the exception also here.

Dennoch gibt es funktionelle Unterschiede.

There are functional differences nevertheless.

18.1.3.1. Die Formatier-Routine

The formatting routine

TI und Atronic formatieren in einfacher Dichte mit einem sog. 'Running-Interlace' oder auch 'Track-Skew'. Dabei werden nicht alle Spuren identisch formatiert (immer mit dem gleichen

TI and Atronic format in single density with one so-called "Running Interlace" or also "Track Skew." Not all tracks are formatted identically (always beginning with the same sector), but it

Sektor beginnend), sondern es wird ein Versatz der Sektornummern von Spur zu Spur erzeugt, der eine Verzögerung in der Größenordnung der Step- und Settle-Time bewirkt. Dies ermöglicht ein fließendes Lesen und Schreiben auch wenn Schrittimpulse dazwischenliegen.

Myarc und CorComp formatieren starr, jede Spur gleich. Jedoch erlaubt CorComp durch Setzen eines Bits im System-RAM, daß das Sektor-Interlace vom Anwender definiert wird. Damit wird zwar immer noch jede Spur gleich formatiert, jedoch kann das Interlace im Einzelfall die Zugriffszeiten optimieren. Myarc formatiert nur mit 16 Sektoren pro Spur bei doppelter Dichte, was wohl einer zu starken Anlehnung an die Western Digital Datenblätter zuzurechnen ist.

Atronic und CorComp formatieren gewissermaßen blind — der Anwender muß selbst prüfen, ob die Formatierung gelang. TI formatiert bei zweiseitigen Disketten zwar auch blind, prüft jedoch (noch in der Level 1 Routine) bei zweiseitiger Formatierung, ob die zweite Seite korrekt formatiert wurde. Myarc prüft sofort nach der ersten doppelseitig formatierten Spur, ob die zweite Seite korrekt ist und schaltet gegebenenfalls selbst auf einseitige Formatierung um.

Auch die Anzahl zu formatierender Spuren wird unterschiedlich interpretiert. TI und Atronic halten sich genau an die Vorgaben über Seiten- und Spuranzahl. Myarc erlaubt nur exakt 40 Spuren, alle anderen Werte erzeugen eine Fehlermeldung.

Besondere Eigenmächtigkeit in Bezug auf die Vorgaben zeigt der CorComp-Controller. Seine Software prüft, ob mehr als 40 Spuren zu formatieren sind. Wenn nicht, dann werden soviele Spuren und Seiten formatiert, wie es beim Aufruf der Routine spezifiziert wurde. Andernfalls wird die Spuranzahl halbiert und auf zweiseitige Formatierung übergegangen. Damit werden Variationsmöglichkeiten für eigene Software verschenkt.

a disalignment of the sector numbers from track to track are produced, which a delay in the order of magnitude the step and settle time. This enables a flowing reading and writing even if clocks lies in between.

Myarc and CorComp format rigidly, each track directly. However CorComp permits by setting a bit in the system RAM that the sector interlace is defined by the user. Thus still each equal track one formats, however the interlace can optimize the access times in individual cases. Myarc formats only with 16 sectors per track with double density, which is probably to be added to the Western Digital data sheets to a too strong support.

Atronic and CorComp format to a certain extent blindly — the user must check even whether formatting succeeded. TI formatted with bilateral diskettes also blindly, checks however (still in the level 1 routine) when bilateral formatting whether the second side was correctly formatted. Myarc checks immediately after first on both sides formatted track whether the second side is correct and switches if necessary even to one-sided formatting.

Also the number formatting tracks is interpreted different. TI and Atronic adhere exact to the specifications over page and number of tracks. Myarc permitted only accurately 40 tracks, all other values produce an error message.

Special arbitrary action regarding the specifications shows the CorComp controllers. Its software checks whether more than 40 tracks are to be formatted. If not, then as many tracks and sides become formatted, as it was specified with the call of the routine. Otherwise the track number is halved and changed over to bilateral formatting. Thus variation options for own software are given away.

18.1.3.2. Die Verify-Routine

Bei TI, Atronic und CorComp wird eine Byte-Für-Byte Verify-Routine verwendet. Hierbei wird bei einem Sektor-Schreib-Befehl neben der eigentlichen Schreibroutine eine Vergleichsroutine ins PAD-RAM kopiert, die unmittelbar nach dem Schreiben aufgerufen wird. Je nach Ergebnis dieses 'On-Line' Datenvergleichs wird der Schreibversuch wiederholt oder aber die Routine beendet. Dabei wird der Verify-Vorgang wie ein Sektor-Lese-Befehl behandelt.

Ganz anders, und eigentlich recht trickreich, ist dies beim Myarc-DOS realisiert. Nachdem der Sektor geschrieben wurde, wird ein modifizierter Sektor-Lese-Zyklus gestartet, bei dem die Daten von Diskette quasi 'ins Leere', nämlich ins Konsolen-ROM geschrieben werden. Dieser Adreßbereich wird benutzt, da auf ihn ohne Wait-States zugegriffen werden kann, und zudem wegen des ROMs nichts zerstört werden kann. Es wird also nicht explizit verglichen, ob die Daten korrekt von der Diskette kommen. Soweit CRC und Lost-Data Fehlertypen betroffen sind, wird diese Aufgabe bereits vom FDC übernommen. Auch wurden eventuelle Lost-Data-Zustände beim Schreiben ggf. bereits dort gemeldet. Soweit mag also diese Art des Datentests ausreichend erscheinen. Doch eine Fehlerart, die auf den ersten Blick konstruiert aussieht, wird mit diesem abgemagerten Verify nicht erkannt: die Verfälschung von geschriebenen Daten, die keinen CRC-Fehler erzeugen. Dies kann bei der Übergabe von Daten an den FDC passieren, und ist nur mit endlicher Wahrscheinlichkeit auszuschließen. Im Endeffekt also wieder ein Punktgewinn für die Konkurrenz.

18.1.3.3. Die Dichte-Erkennung

Alle DD-Systeme müssen erkennen, in welcher Codierungsart die Daten auf der jeweiligen Diskette vorliegen. Dies ist nur durch Probieren möglich. Das Verfahren ist dabei immer gleich: Es wird mit einer willkürlich festgelegten Dichte

The verify routine

With TI, Atronic and CorComp is used byte for byte a verify routine. Here with a sector write instruction beside the actual write routine a comparison routine is copied in PAD RAM, which is called immediately after writing. Depending upon result this "on-line one" of data comparison is repeated however the write attempt or the routine is terminated. The verify process is treated like a sector read instruction.

Completely differently, and quite actually sophisticated, this is implemented with the Myarc DOS. After the sector was written, a modified sector reading cycle is started, with which the data are written by diskette quasi "in emptiness," i.e. in console ROM. This address range is used, since it without wait states can be accessed, and besides because of ROM nothing can be destroyed. It explicitly it is thus not compared whether the data come correctly from the diskette. As far as CRC and Lost Data types of error are concerned, this function is already taken over by the FDC. Also possible draw DATA statuses became during the writing if necessary already there announced. So far thus this type of the data test may appear sufficient. But an error, which looks designed at first sight, is not detected with this wasted verify: the falsification of written data, which do not produce CRC error. This can occur with the transfer of data to the FDC, and is only with finite probability to be excluded. In the final result thus again a point gain for the competition.

Density identifier

All DD systems must detect, in which type of coding the data on the respective diskette to be present. This is possible only by trying. The procedure is always alike thereby: It is tried with an arbitrarily determined density to access the

versucht, auf die Daten der Diskette zuzugreifen. Verlief der Versuch erfolgreich, so bleibt nichts weiter zu tun.

Wurde jedoch ein Lost-Data, CRC oder RNF-Fehler erkannt, dann wird der Versuch mit einer anderen Dichte wiederholt. Führte dieser neue Versuch zum Erfolg, dann wird diese Dichte für das betreffende Laufwerk notiert und beim nächsten Zugriff auf dieses Laufwerk gleich mit der korrekten Dichte begonnen.

Bei Diskettenwechsel auf eine andere Dichte arbeitet dieser Algorithmus gleichermaßen.

Wann nun auf eine andere Dichte umgeschaltet wird, ist verschieden. Atronic schaltet nach 5 Fehlversuchen auf eine andere Dichte um (danach laufend), und bricht nach insgesamt 10 erfolglosen Versuchen ab. CorComp erlaubt ebenfalls insgesamt 10 Versuche, wechselt jedoch nach jedem Fehlversuch die Dichte.

Myarc schließlich wechselt ebenfalls nach einem Fehlversuch die Dichte, benutzt jedoch eine Adreßfeld-Leseroutine, um die wirkliche Dichte zu ermitteln. Durch dieses direkte Verfahren sah man sich in der Lage, nur 5 Retries zulassen zu müssen.

Gegen das Atronic- und Myarc-Verfahren ist prinzipiell nichts einzuwenden. Lediglich die laufende Umschalterei bei CorComp läßt Probleme erwarten, die eine defekte Diskette auslösen könnte. In praxi arbeiten alle 3 DD-Systeme in diesem Punkt weitgehend korrekt, CorComp weist tatsächlich gelegentlich Fehler der beschriebenen Art auf. Die weiteren Details können dem jeweiligen ROM-Listing entnommen werden.

18.1.3.4. Indizierte Adressierung

Die DOS-Routinen bei TI und Atronic sind mit jedem Wert des Workspace-Pointers lauffähig. Lediglich in R15 muß sich die Adresse VDPWA (>8C02) befinden, und die Parameterzeiger

data of the diskette. If the attempt successfully ran, then remains doing nothing further.

However Lost Data, CRC or RNF error was detected, then the attempt with another density is repeated. If this new attempt led to success, then this density for the drive concerned is noted and begun with the next access to this drive equal with the correct density.

At diskette change to another density this algorithm operates equally.

When now to another density one switches, is different. Atronic switches after 5 unsuccessful attempts to another density over (after it constantly), and aborts after altogether 10 unsuccessful attempts. CorComp permitted likewise altogether 10 attempts, changes however to each unsuccessful attempt the density.

Myarc finally changes likewise to an unsuccessful attempt the density, however an address field read routine uses, in order to determine the real density. By this direct procedure one was in the position to have to permit only 5 retries.

Against the Atronic and Myarc procedure is to be objected nothing always. Only the current changing over with CorComp suggests problems, which a defective diskette could release. In practice all 3 DD systems in this point operate to a large extent correctly, CorComp indicates actually occasionally errors of the described type. Further details can be inferred from the respective ROM listing.

Indicated addressing

The DOS routines with TI and Atronic are executable with each value of the Workspace Pointer. Only in R15 the address VDPWA (>8C02) must be, and the parameter pointers

müssen relativ zum WP gleich liegen. Zwar hat CorComp das auch versucht, doch klappt so etwas nur, wenn es im ganzen Programm konsequent durchgehalten wird. Genau das ist aber bei CorComp nicht der Fall (siehe Adressen >4AF0 und >5682). In diesen Befehlen wird absolut auf Adressen zugegriffen, in denen sich bestimmte Registerinhalte des aktuellen WS befinden.

Bei allen dreien befindet sich in Register 9 der Wert >8300, die Standard-Basis des PAD-RAMs. Bei Myarc wird hierzu Register 4 verwendet, welches >83E0, also den echten Workspace-Pointer enthält.

Dadurch werden die Pointer aber nur mittels negativem Offset erreicht, was recht undurchsichtig wirkt. Allerdings wird von R4 nur auf Level 2 richtiger Gebrauch gemacht, sonst wird absolut adressiert. Hier muß also GPLWS geladen sein.

18.2. Hinweise zur Lesart der ROM-Listings

Da es sich nur um Auszüge handelt, wurden einige Passagen aus Platz-gründen gestrichen. An diesen Stellen findet sich das Symbol

```
*
*  ...
*
```

was in Anlehnung an die Notation des Assemblers eine Unterbrechung im Source-Code angibt. An einer solchen Stelle fehlen Adreßbereiche, die dem Rotstift zum Opfer fielen.

Ansonsten wären die Listings nach korrektem Abtippen und anpassen der Labels direkt assemblierbar, wodurch die Erstellung eines eigenen DOS möglich wird.

must be situated relative to the WP directly. CorComp tried also, but works such a thing only, if one holds out in the whole program consistently. Exactly that is not however with CorComp the case (see to addresses >4AF0 and >5682). In these instruction absolutely in addresses one accesses, in which determined register contents of the current WS are.

With all three the value is >8300, the standard base of the PAD RAM in register 9. With Myarc for this register 4 is used, which >83E0 contains, thus the genuine Workspace Pointer.

Thus the pointers are achieved however only by means of negative offset, what works quite obscurely. However by R4 only on level 2 correct use is made, otherwise absolutely one addresses. Here thus GPLWS must be loaded.

Notes to the interpretation of the ROM listings

It only single dumps concerns there, some passages for space reasons were deleted. In these places is the symbol

```
*
*  ...
*
```

which indicates an interruption following the notation of the assembler in the source code. In such a place are missing address ranges, which fell victim to the red pencil.

Otherwise the listings would be after correct typing and adapt the labels directly assemblable, whereby the creation of its own DOS becomes possible.

18.2.1. Besonderheiten im TI-ROM-Listing

18.2.1.1. SEEK-Befehl und Verifikation der Seite

Da der 1771-FDC keinen Seitenvergleich beim Anfahren einer Spur erlaubt, muß mit anderen Methoden ermittelt werden, ob die korrekte Spur erreicht und der richtige Kopf des Laufwerks geschaltet wurde.

Daher wird nach dem theoretischen Erreichen der Spur, deren Nummer über den sehr wohl möglichen Verify-On-Track geprüft. Wenn diese Nummer korrekt war, wird auf der gewählten Seite ein beliebiges Adreßfeld gelesen und dessen Seitenkennung geprüft. Es handelt sich also um das gleiche Verfahren, das die moderneren FDC (1773) beim Sektorzugriff selbst anwenden.

18.2.1.2. Invertierte FDC-Kommandos

Durch den invertierten Datenbus des 1771 liegen die Kommandos bereits invertiert vor. Dabei ist bei den Typ I Kommandos zu beachten, daß die Step-Raten für jeden Befehl (RESTORE, SEEK, STEP IN) fest im Befehl eingebaut wurden. Soll der TI-Controller also kürzere Step-Zeiten 'verbraten' bekommen, so müssen alle Typ I Kommandos geändert werden!

18.2.1.3. Softwareänderungen

Nur per Software ist der 1771 nicht dazu zu bringen, in doppelter Dichte zu arbeiten. Man kann jedoch durch geringfügige Änderungen eine Möglichkeit schaffen, 80 Spur Laufwerke anzusteuern, die es dann bei 2 mal 80 Spuren in Single Density erlauben, die gleiche Menge an Daten auf einer Diskette zu halten, wie es bei 2 mal 40 Spuren in doppelter Dichte möglich ist, nämlich 360 KByte. Hierzu müssen lediglich die Vergleichszahlen am Anfang der Sektor-I/O-Routine entsprechend erhöht werden. Erfolgt noch eine Anpassung der Step-Raten auf die normalerweise sehr schnellen 80 Spur Laufwerke, so wird der Geschwindigkeits-unterschied zu Double Density auf das

Special features of the TI ROM listing

Seek instruction and verification of the side

There the 1771-FDC a side comparison when starting a track permitted, it does not have to be determined with other methods whether the correct track was achieved and the correct heading of the drive was switched.

Therefore after theoretical achieving of the track, their number over very probably the possible verify on tracks is checked. If this number were correct, on the selected side any address field is read and its side identifier is checked. It concerns thus the same procedure, which the more modern FDC (1773) applies with the sector access.

Inverted FDC commands

By the inverted data bus 1771 are situated the commands already inverted forwards. It is to be noted with the Type I commands that the step rates for each instruction (RESTORE, SEEK, STEP IN) were built firmly in the instruction. If the TI controller thus to shorter step times "blow" to get, then all type I commands must be modified!

Software modifications

Only by software is not to be brought the 1771 to operate in double density. One can create however by slight modifications a possibility of heading for 80 track of drives which permit it then with 2 times 80 tracks in single to density to hold the same quantity of data on a diskette like it with 2 times 40 tracks in double density is possible, i.e. 360 KByte. For this the comparative figures at the start of the sector I/O routine must be only increased accordingly. Taken place still another normally very much if 80 track of drives quick adjustment of the step rates on those, then the rate difference to double density is reduced to the theoretical minimum (factor 1.8 without step times). Such "simply DOS-80" is software neutral

theoretische Minimum (Faktor 1.8 ohne Step-Zeiten) reduziert. Ein solches 'Einfach DOS-80' ist zwar softwareneutral (da man nicht an die 5 undeklarierten Bytes im VDP muß, um die Spuranzahl je Laufwerk zu verwalten), jedoch können zweiseitige 40 Spur Disketten damit nicht mehr bearbeitet werden.

18.2.2. Besonderheiten im Atronic-ROM-Listing

Das Atronic-DOS verwendet 2 getrennte Routinen zur Formatierung in einer der beiden möglichen Speicherdichten. Dabei wird jeweils eine Spur formatiert und aus dem gemeinsamen Segment das Step-In ausgeführt. Dabei ist die Sequenz nach Fertigstellung eines Spurinhaltes zweimal fast identisch vorhanden. Der einzige Unterschied besteht in der Spurlänge und der Dichte. Funktionell wirkt sich das nicht aus, doch ist es ein Schönheitsfehler, den sich ein versierter Programmierer nicht erlauben sollte.

18.2.2.1. Soft- und Hardwareänderungen

Das System-RAM wird im Atronic-Controller recht wenig genutzt. So könnte man Optionen wie RAM-Transfer, Verify abschalten, extern definiertes Interlace (alles wie bei CorComp) vorsehen. Denkbar, und im Grunde eigentlich gar nicht so abwegig, wäre die Option auf 42 Spuren pro Diskettenseite. Damit würden, entsprechende Laufwerke vorausgesetzt, 378 KByte pro DSDD-Disk zur Verfügung stehen, die sogar noch mit dem Standard-DOS verwaltet werden könnten (die Sector-Bitmap erlaubt ja bekannterweise bis 400 KByte pro Disk). Oder aber, man formatiert mit 19 Sektoren pro Spur (statt 18 bei DD) und erreicht so bei 42 Spuren pro Seite 1596 Sektoren pro Disk, also 399 KByte pro Disk. Die Möglichkeiten sind vielfältig.

Auch sind die CRU-Bits 9, >C, >D, >E und >F nicht belegt. Zusammen mit dem System-RAM (als Index-Speicher) könnte man 80 Spur Laufwerke ggf. auf 40 Spuren umschalten und dies beim Sektorzugriff in der Spur-Sektor-

(there one not to the 5 undeclared bytes in VDP must to administer by the number of tracks for each drive), however cannot bilateral 40 track diskettes with it any longer be processed.

Special features in the Atronic ROM listing

Unnecessary in the formatting routine the Atronic DOS uses 2 separate routines for formatting in one of two possible memory densities. A track is formatted in each case and executed from the common segment the step in. The sequence is twice almost identically available after completion of track contents. That only difference exists in the track length and the density. Functionally not, but is it affects itself a blemish, which an experienced programmer should not take the liberty.

Soft- and hardware modifications

The system RAM in the Atronic controller quite little is used. So one could switch options off such as RAM transfer, verify, designate externally defined interlace (everything as with CorComp). Conceivably, and in the reason actually not at all so incorrectly, would be the option on 42 tracks per diskette side. Thus, appropriate drives were presupposed, to 378 KByte per DSDD disk at the disposal to be, which could be administered even still with the standard DOS (the second gate bitmap permitted well-known-proved to 400 KByte per disk). Or however, one formats 399 KByte per disk with 19 sectors per track (instead of 18 with DD) and achieves so with 42 tracks per page 1596 sectors per disk, thus. The possibilities are various.

Also the CRU bits 9, >C, >D, >E and >F are not occupied. Together with the system RAM (as index memory) one could do 80 track of drives if necessary switch to 40 tracks and this with the sector access in the track sector calculation

Berechnung auswerten.

Der mit CRU-Bit >B geschaltete Bereich des 16 KByte ROMs ist zudem in weiten Bereichen nicht programmiert, so daß hier etliche neue Routinen Platz finden könnten.

18.2.3. Besonderheiten im CorComp-ROM-Listing

18.2.3.1. Überflüssiges in der Formatierroutine

Das CorComp-DOS ist streckenweise sogar noch etwas umständlicher als das von Atronic. So wird nicht in einem Zug auf beiden Seiten formatiert, wenn zweiseitig gefordert war, sondern es wird erst die eine Seite formatiert, dann ein RESTORE ausgeführt, und erst dann auf der zweiten Seite gearbeitet. Nicht nur, daß das RESTORE unnötig Zeit kostet, es ist auch sonst nicht zu sehen, wieso dieses Verfahren gewählt wurde. Dazu kommt noch, daß bei CorComp sogar das Step-In nach einer Spur für SD und DD getrennt, also 2 Mal vorhanden ist. Zusammen mit solch vielsagenden Befehlen wie 'LI R0,>FFFF' gibt es noch andere Bereiche, die den Leser möglicherweise dazu veranlassen, sich erschrocken an den Kopf zu greifen.

18.2.3.2. Soft- und Hardwareänderungen

Neben der Beseitigung oben genannter Mißstände ist im Prinzip das Gleiche zu sagen wie bei Atronic. Zudem stehen durch den 9901 als CRU-Interface noch etliche CRU-Leitungen für Ein- und Ausgänge zur Verfügung. Wieso CorComp hier einen so teuren Chip anstatt 3 kleiner TTL-Latches eingesetzt hat, wird wohl von hier aus schwer zu erklären sein. Auf jeden Fall sitzt hier ein CRU-Interface, das in einem Floppy-Disk-Controller nur mit gebremstem Schaum arbeitet. Denkbar wären Drive-Sense-Eingänge wie beim TI-Controller. Damit würde der unvermeidliche I/O-ERROR 06 beim Zugriff auf nicht vorhandene Laufwerke wenigstens schneller kommen. Auch könnten direkt Index-Puls, Schreibschutz und andere Leitungen vom Laufwerk getestet werden, ohne erst analyse.

The area 16 KByte of ROM switched with CRU bits >B is besides in wide areas not programmed, so that some new routines could find place here.

Special features in the CorComp ROM listing

Redundancies in the format routine

The CorComp DOS is by sections even still somewhat pedantic than from Atronic. Thus in a course on both sides one does not format, if were bilaterally required, but only the one side is executed formatted, then a RESTORE, and only then on the second page operates. Not only that the RESTORE costs time unnecessarily not to see it is also otherwise why this procedure was selected. In addition it comes still that with CorComp even the step in is available separately according to a track for SD and DD, thus 2 times. Together with such much-saying instruction like LI R0, >FFFF there are still different areas, which cause the reader possibly to access itself frightened to the heading.

Soft- and hardware modifications

Apart from the removal mentioned above of bad states in principle the same is to be said as with Atronic. Besides are by the 9901 as CRU interfaces still some CRU lines for inputs and outputs at the disposal. Why CorComp used a so expensive chip here instead of 3 small TTL Latches will be difficult probably to explain, from here. In any case one sits here an CRU interface, which operates in a Floppy Diskette Controller only with braked foam. Drive sense inputs would be conceivable as with the TI controller. Thus the inevitable I/O error 06 would come with the access to missing drives at least faster. Also could be tested directly index pulse, write protection and other lines of the drive, without setting about only pedantically the FDC with a not executable instruction.

umständlich den FDC mit einem nicht ausführbaren Befehl zu traktieren.

All das wäre durch ein paar mehr Leitungen auf der Platine fast zum Nulltarif möglich gewesen (der versierte Bastler kann das aber jederzeit mit ein paar Drähten nachholen!).

Irgendwie erinnert das an Texas Instruments, die ihre seinerzeit bahnbrechende 9900-CPU mit lächerlichen 256 Byte High-Speed RAM versahen und alles andere über Wait-States bremsten.

Der mit CRU-Bit >B geschaltete Bereich des 16 KByte ROMs ist, im Gegensatz zum Atronic-ROM, weitgehend mit Basic-Routinen, der Verwaltung des eigenen Titelbildes und der Manager-Laderoutine gefüllt, so daß hier kaum Platz für Eigenentwicklungen bleibt. Jedoch ist das eigene Titelbild verzichtbar, ebenso wie der Titelbild-Manager-Loader, der auch aus dem Basic aufgerufen werden kann. Hier fällt neuer Raum an, der für eigene Ideen nutzbar ist.

18.2.4. Besonderheiten im Myarc-ROM-Listing

Besonders undurchsichtig ist die Sektor-I/O-Routine programmiert. Durch die vielen Flags, die den Ablauf beeinflussen, ist kaum zu erkennen, was passiert. Grundsätzlich wird unterschieden zwischen Sektor lesen und schreiben, zwischen Transfer ins oder aus dem CPU-RAM und Transfer über VDP-RAM. Dabei führt ein Verify-Lauf immer ins ROM.

Hier heißt es im Zweifelsfall mehrfach lesen.

18.2.4.1. Unsauberkeiten in der Programmierung

Ein Lost-Data-Zustand, im Allgemeinen bewirkt durch das zeitkritische Polling-Verfahren, führt beim Myarc-DOS zu einer Wiederholung des Befehls ohne Reduktion der Retries. Zum einen sind ohnehin mit 5 nur sehr wenig

All that would have been possible by a few more lines on the circuit board almost to the zero cost (the experienced amateur handiman can retrieve however at any time also a few wires!).

Reminds somehow of Texas Instruments, which its at that time innovative 9900 CPU provided with ridiculous 256 byte high speed RAM and braked everything else over wait states.

The area 16 KByte of ROM switched with CRU bits >B is to a large extent filled with Basic routines, the administration of the own frontispiece and the manager load routine, in contrast to Atronic ROM, so that hardly place for self-developments remains here. However the own frontispiece is renounceable, just like of the frontispiece manager Loader, which can be called also from the Basic. Here new space results, which is usable for own ideas.

Special features in the Myarc ROM listing

The sector I/O routine is particularly obscurely programmed. By the many flags, which influence the flow, is hardly to be detected, what occurs. Basically distinctive between sector to read and write, between transfer into or from the CPU RAM and transfer over VDP RAM. A verify run always leads into the ROM.

Here it means in the case of doubt several times reads.

Carelessnesses in programming

A Lost Data status, generally effectuation by the time-critical polling procedure, leads with the Myarc DOS to a repetition of the instruction without reduction of the Retries. Sometimes on the one hand are anyway with 5 only very few

Wiederholungschancen gegeben, zum anderen kann sich das unter Umständen zu einer Endlosschleife entwickeln. Im praktischen Betrieb konnte beobachtet werden, daß der Controller gelegentlich Pausen von bis zu 30 Sekunden einlegte, in denen sich augenscheinlich nichts tat. Manchmal konnte diese Bedenkpause durch Wackeln am Jacket der Diskette unterbrochen werden. Es sollte daher bei einer Softwareänderung hier zuerst die Retry-Zahl verdoppelt und ein LostData in den Retry-Countdown aufgenommen werden.

repetition chances given, on the other hand can that to a continuous loop develop. In the practical operation could be observed that the controller inserted occasionally a pause of up to 30 seconds, in which apparently nothing did. Sometimes this considering break could be interrupted by wobbling at the jacket of the diskette. Therefore here first the retry number should be doubled during a software modification and Lost Data in the Retry Countdown be taken up

Durch die willkürliche Verwendung des VDP-RAMs ergeben sich Unverträglichkeiten mit Software, die normgerechte VDP-Verwaltung voraussetzt.

As a result of the arbitrary use of the VDP RAM arise incompatibilities with software, which presupposes standard VDP administration.

Auf Level 2 ist hier nur mit viel Aufwand etwas zu ändern, bei Level 1 jedoch kann zumindest das Chaos im VDP-Stack, wo das PAD-RAM ab >83A2 gesichert wird, durch Beachten des Zeigers an >8366 vermieden werden.

On level 2 here something is to be modified, with level 1 however can at least the chaos in the VDP stack, where that off becomes secured PAD RAM >83Á2, by attention of the pointer to >8366 be avoided on only at much expenditure.

Etwas unglücklich ist die Verwaltung mehrerer aufeinanderfolgender Zugriffe auf verschiedene Laufwerke gelöst. Alle 3 anderen Systeme merken sich die Nummer des zuletzt angesprochenen Laufwerks und können so einem neu zugeschalteten die notwendige Zeit zum Erreichen der Zugriffsbereitschaft lassen. Im Myarc-DOS wird erst bei einem Fehler durch Auslösen eines Interrupts eine kurze Pause eingelegt. Das ist nicht nur umständlich, sondern schlicht unsinnig.

Somewhat unfortunately the administration several is successive accesses to different drives solve. All 3 other systems notes the number of the drive addressed last and can so to a again connected the necessary time for achieving the readiness to access leave. In Myarc DOS is only inserted in the case of an error by releasing an interrupt a short break. That is not only pedantic, but simply unreasonable.

18.2.4.2. Soft- und Hardwareänderungen

Soft- and hardware modifications

Im oberen ROM-Bereich, der mit CRU-Bit 3 geschaltet wird (dient gleichzeitig der Dichte-Umschaltung), ist noch ca. 1 KByte Platz. Auch das System-RAM kann, da als 8 Bit-RAM ausgeführt, extern oder für eigene Optionen wie etwa direkten RAM-Transfer genutzt werden. Das CRU-Interface ist jedoch komplett ausgelastet.

Within the upper ROM area, which is switched with CRU bit 3 (serves at the same time density switching), is still approx. 1 KByte place. Also system RAM can be executed, there as 8-bit RAM, be used externally or for own options as for instance direct RAM transfer. The CRU interface is however completely working at full capacity

19. ROM-Listing TI-Controller ROM listing TI Controller

```
*
* Sektor Lesen/Schreiben
*
40E8  LI   R4,10           Anzahl der Versuche = 10
40EC  MOVB @>4630,@>50(R9) Fehlerbyte löschen
      BL   @>4496          Laufwerk anwählen
      BL   @>45F0          VDP-Zeiger auf Puffer der letzten Spur dieses LW
      CLR  R0
      MOVB @>FBFE(R15),R0  Letzte Spur lesen
4100  CI   R0,>D700        mit >28, dez 40 (invertiert) vergleichen
      JH   >410C           tatsächliche Spur ist kleiner - weiter
      BL   @>4524      Spurnr. ist zu groß - LW initialisieren, RESTORE
      SETO R0          Spurnummer 0 (invertiert!)
410C  MOVB R0,@>5FFA     Spurnummer ins Spurregister des FDC
     MOV  @>4A(R9),R1   Kopie der Sektornummer holen (wurde in >8350 übergeben)
      SBZ  7             Disk-Unterseite anwählen
      CLR  R7          Seitenindex
      CI   R1,720      Sektornummer über dem Maximalwert?
      JHE  >41B6       Nummer ist zu groß
      CI   R1,360      Größer als die höchste Nummer auf Seite 1?
      JL   >413E       Nein
      AI   R1,-719     Ja, Komplement bilden, Sektor ist auf der Unterseite!
      ABS  R1          Betrag
      CLR  R0          Rechenregister
      DIV  @>4632,R0   durch 9 teilen, Spur und relativen Sektor berechnen
      AI   R1,-8       Sektornummer korrigieren
      ABS  R1          Betrag
      SBO  7           Unterseite anwählen
      LI   R7,>0100    Seitenindex
      JMP  >414E       weiter
413E  CI   R1,1        Sektornummer 1?
      JH   >4148       größer
      BL   @>4524      RESTORE, Kopf auf Spur 0
4148  CLR  R0          Rechenregister 'Spur' löschen
      DIV  @>4632,R0   Sektornummer geteilt durch 9
414E  SWPB R0          errechnete Spur des gesuchten Sektors ins High-Byte
      INV  R0          auf FDC-Chip Bus anpassen (invertieren)
      BL   @>4614       VDP-Schreibadresse mit Wert in R2 setzen
      MOVB R0,@-2(R15)  Spurnummer in den Puffer der letzen Spur dieses LW
      MOVB R0,@>5FFE    und ins FDC-Datenregister (WRITE TRACK TO SEEK)
      SWPB R1          Sektornummer (relativ) ins High-Byte
      INV  R1          anpassen
      MOVB R1,@>5FFC   ins Sektorregister
      CB   R0,@>5FF2   stimmt die geforderte Spur mit der aktuellen überein?
      JEQ  >417A       Ja, Kopf nicht bewegen!
      BL   @>45CA      Nein, FDC-Kommandoroutine
      DATA >E100       SEEK TRACK
      BL   @>4482      FDC-Status holen, Befehlsende abwarten
      SLA  R0,13       SEEK ERROR (Spur nicht verifiziert)?
      JOC  >41B0       Ja, Fehler >11
417A  BL   @>45CA      Spur gefunden, neues FDC-Kommando
```

```
      DATA >3F00       READ ADDRESS, beliebiges Adressfeld lesen
      SBO  2           WAIT ENABLE BIT, erlaubt das Anhalten der CPU
4182  MOVB @>5FF6,R0   Spurnummer lesen
      LI   R6,4        4 Bytes blind lesen
      MOVB @>5FF6,R5   Seitennummer lesen
      INV  R5          korrigieren
4190  MOVB @>5FF6,R0   Sektor, Sektorlänge und CRC blind lesen, diese Bytes
      DEC  R6          werden nicht benötigt
      JNE  >4190       weiter
      BL   @>4480      Befehlsende abwarten (WAIT DISABLE)
      SLA  R0,13       RECORD NOT FOUND (kein Adressfeld gefunden)?
      JOC  >41BC       Ja, Fehler >21
      JLT  >41C2     CRC ERROR (Adressfeld defekt)! Fehler >22
      SLA  R0,2      LOST DATA (Lesefehler)?
      JOC  >41C8     Ja, Fehler >23
      CB   R7,R5     stimmt die Seitennummer mit der des AF überein?
      JEQ  >41CE     Ja, OK!
*
* Fehler-Routinen, >4590 erlaubt Wiederholung des Versuchs, >45AC nicht!
*
      BL   @>45AC    Fehler >06: - Seitenanwahl fehlgeschlagen
      DATA >0600
41B0  BL   @>4590    Fehler >11: - Seek Error, Spur nicht verifiziert
      DATA >1100
41B6  BL   @>45AC    Fehler >07: - Sektornummer zu groß
      DATA >0700
41BC  BL   @>4590    Fehler >21: - Record not found, kein Adressfeld gef.
      DATA >2100
41C2  BL   @>4590    Fehler >22: - CRC-Error, Adressfeld defekt
      DATA >2200
41C8  BL   @>4590    Fehler >23: - Lost Data, Lesefehler (schwerwiegend)
      DATA >2300
*
41CE  MOVB R1,@>5FFC     Sektornummer schreiben
      MOV  @>4E(R9),R2   VDP-Pufferadresse holen
      MOVB @>4D(R9),R0   I/O-Code lesen
      JEQ  >425C         >00, also Sektor schreiben
      BL   @>4614        VDP-Schreibadresse setzen
      BL   @>45CA        FDC-Kommando
      DATA >7700         READ SECTOR
      LI   R6,>0100      Anzahl Bytes im Sektor
      SETO R5            Index 'LESEN/VERIFY WAR FEHLERHAFT'
      SBO  2             FDC-Chip erlauben, die CPU anzuhalten
      MOVB @>4D(R9),R0   I/O-Code
      JNE  >4214         nicht Null, also lesen
*
* Sektordaten Verify
*
      CLR  R0            I/O-Code ist 0, nach Schreiben jetzt Verify
41F6  MOVB @>5FF6,R0     1 Byte aus dem FDC-Chip lesen
      AB   @>FBFE(15),R0 zum Komplement im Datenpuffer addieren (muß >FF sein!)
      CI   R0,>FF00      Richtig?
      JNE  >422E         Nein, Fehler!
      SZCB @>5FF6,R0       Nullen im >FF-Byte setzen
```

```
      SB   @>FBFE(R15),R0  Sollwert abziehen (muß >00 ergeben!)
      JNE  >422E           Nicht Null, Fehler!
      DECT R6             OK, alle durch?
      JNE  >41F6          Nein, weiter
      JMP  >422C         alle Bytes korrekt - weiter mit Löschung des Index R5!
*
* Sektorinhalt lesen
*
4214  MOVB @>5FF6,R0    ein Datenbyte von Disk holen
      INV  R0           korrigieren
      MOVB R0,@-2(R15)  ins VDP-RAM
      MOVB @>5FF6,R0    dto. nächstes Byte
      INV  R0
      MOVB R0,@-2(R15)
      DECT R6           alle durch?
      JNE  >4214        nein
*
422C  CLR  R5           Fehlerindex nach lesen
*
* Fehler-Einsprung bei Verify-Fehler
*
422E  BL   @>4480    Befehlsende abwarten
      SLA  R0,13     RECORD NOT FOUND (Sektor nicht korrekt erkannt)?
      JOC  >4244     Ja, Fehler >21
      JLT  >424A     CRC ERROR (Daten teilweise verloren)!
      MOV  R5,R5     Fehler beim Verify?
      JNE  >4256     Ja, Fehler >28
      SLA  R0,2      LOST DATA (Daten verspätet angeliefert/abgeholt)?
      JOC  >4250     Ja, Fehler >23
      B    @>4676    Operation erfolgreich, via VDP-Stack zurückspringen
*
* Fehler-Routinen, Wiederholungen sind erlaubt
*
4244  BL   @>4590    Fehler >21: - Sektor nicht korrekt erkannt
      DATA >2100
424A  BL   @>4590    Fehler >22: - CRC-Fehler im Sektor
      DATA >2200
4250  BL   @>4590    Fehler >23: - Datenverlust (schwerwiegend)
      DATA >2300
4256  BL   @>4590    Fehler >28: - Verify nicht erfolgreich
      DATA >2800
*
* Sektor schreiben
*
425C  BL   @>461E       VDP-Leseadresse mit R2 setzen
      BL   @>45CA       FDC-Kommando
      DATA >5700        WRITE SECTOR
      LI   R6,>0100     256 Byte im Sektor
      SBO  2               WAIT ENABLE
426C  MOVB @>FBFE(R15),R0  Datenbyte aus VDP lesen
      INV  R0              invertieren
      MOVB R0,@>5FFE       in FDC-Datenregister schreiben
      MOVB @>FBFE(R15),R0  nächstes Byte
      INV  R0              anpassen
```

```
      MOVB R0,@>5FFE     ins Datenregister
      DECT R6            alle Bytes?
      JNE  >426C       Nein
      BL   @>4480      Ja, Befehlsende abwarten
      SLA  R0,11       WRITE PROTECT (Disk schreibgeschützt)?
      JOC  >429A       Ja, Abbruch mit Fehler >34
      SLA  R0,2        RECORD NOT FOUND (Sektor nicht gefunden)?
      JOC  >42A0       Ja, Fehler >31
      SLA  R0,2        LOST DATA (Daten verspätet angeliefert)?
      JOC  >42A6       Ja, Fehler >33
      BL   @>461E      VDP-Leseadresse erneut setzen
      JMP  >41E0       weiter mit Verify
*
429A  BL   @>45AC    Fehler >34: - Disk ist schreibgeschützt, kein Retry
      DATA >3400
42A0  BL   @>4590    Fehler >31: - Sektor nicht gefunden
      DATA >3100
42A6  BL   @>4590    Fehler >33: - Datenverlust
      DATA >3300
*
* Diskette formatieren
*
42AC  CLR  @>4A(R9)     Index: Disk ist einseitig
      MOVB @>4C(R9),R8  Laufwerknummer
      SRL  R8,12        höchstes Nibble Null?
      JEQ  >42C4        Ja. OK
      C    R8,@>4630    >0001, also Option 1?
      JEQ  >42C4        Ja
      BL   @>45AC       Falsche Option!
      DATA >0700             Fehlercode >07
42C4  SZCB @>4638,@>4C(R9)   Laufwerknummer normieren
      CB   @>51(R9),@>4657   >02, zweiseitig?
      JNE  >42D6             Nein
      SETO @>4A(R9)          Ja, Disk ist zweiseitig zu formatieren
42D6  MOVB @>4630,@>50(R9)   Fehlerbit löschen
      BL   @>4496            Laufwerk (Nr. in >4C(R9)) zuschalten
      BL   @>4524            RESTORE, Kopf über Spur 0 positionieren
      CLR  R3                Spurnummer >00
42E6  MOV  @>4A(R9),@>4A(R9) Einseitig?
      JEQ  >42F8             Ja, Unterseite auslassen!
      SBO  7             Unterseite der Diskette anwählen
      LI   R7,>0100      Seitennummer >01
      BL   @>43AA        Spur im Puffer aufbauen und schreiben
42F8  SBZ  7             Oberseite anwählen
      CLR  R7            Seitennummer >00
      BL   @>43AA        Spur im Puffer aufbauen und schreiben
      BL   @>45CA        FDC-Kommandoroutine
      DATA >A500         STEP IN, Kopf eine Spurweite nach innen fahren
      BL   @>4482        Befehlsende abwarten
      AI   R3,>0100      Spur +1
      CB   R3,@>4D(R9)       alle geforderten Spuren?
      JNE  >42E6             Nein
      MOV  @>4A(R9),@>4A(R9) Einseitig?
      JEQ  >437A             Ja
```

```
      SBO  7             Nein, auf Unterseite schalten
*
* Prüfen, ob Unterseite korrekt formatiert wurde
*
      LI   R4,10         10 Versuche
      BL   @>4524        RESTORE, Kopf über Spur 0 bringen
      MOV  @>4E(R9),R2   Pufferadresse holen
      BL   @>4614        VDP-Schreibadresse mit R2 setzen
      BL   @>45CA        FDC-Kommandoroutine
      DATA >3F00         READ ADDRESS, Adressfeld lesen
      LI   R6,6          6 Byte lesen
      SBO  2             WAIT ENABLE
433A  MOVB @>5FF6,R0     1 Byte lesen
      INV  R0            invertieren
      MOVB R0,@-2(R15)   ins VDP-RAM
      MOVB @>5FF6,0      dto. nächstes Byte
      INV  R0
      MOVB R0,@-2(R15)
      DECT R6            fertig?
      JNE  >433A         Nein
      BL   @>4480        Befehlsende abwarten
      SLA  R0,13         RECORD NOT FOUND (kein Adressfeld da)?
      JOC  >4398         Ja, Fehler >21
      JLT  >439E         CRC ERROR (Adressfeld defekt)! Fehler >22
      SLA  R0,2          LOST DATA (Datenverlust)?
      JOC  >43A4         Ja, Fehler >23
      MOV  @>4E(R9),R2   Pufferadresse holen
      INC  R2            +1
      BL   @>461E        als Leseadresse setzen
      CLR  R0
      MOVB @>FBFE(R15),R0   Seitennummer lesen
      JEQ  >437A            Null, Oberseite!
      MOVB @>4D(R9),R0   Spuranzahl
      SLA  R0,1          mal 2 (doppelseitig)
      JMP  >4384            weiter
437A  MOVB @>4631,@>51(R9)  >01
      MOVB @>4D(R9),R0      Spuranzahl
4384  SRL  R0,8          ins High-Byte
      MPY  @>4632,R0     mal Sektoren pro Spur (9) ergibt Zahl aller Sektoren
      MOV  R1,@>4A(R9)      übergeben
      MOVB @>4633,@>4D(R9)  9 Sektoren pro Spur
      B    @>4676           Rücksprung via VDP-Stack
4398  BL   @>4590  Fehler >21: - Kein Adressfeld gefunden, Record not found
      DATA >2101   Code mit Index 'Formatieren'
439E  BL   @>4590  Fehler >22: - Adressfeld defekt, CRC-Fehler
      DATA >2201   Code mit Index 'Formatieren'
43A4  BL   @>4590  Fehler >23: - Datenverlust, Lost Data
      DATA >2301   Code mit Index 'Formatieren'
*
* Spurbild im Puffer aufbauen und Spur schreiben
*
43AA  MOV  R11,R8        Rückkehr sichern
      MOV  @>4E(R9),R2   Pufferadresse
      BL   @>4614        als Schreibadresse setzen
```

```
      LI   R6,22          Track-Header 22 Nullen, GAP 1
      CLR  R2             Sektorzähler
      JMP  >43C0
43BC  LI   R6,6             ADDRESS GAP
43C0  MOVB @>4630,@-2(R15)
      DEC  R6
      JNE  >43C0
      MOVB @>4639,@-2(R15) >FE, Adress-Mark-ID
      NOP                  VDP nicht überfahren
      MOVB R3,@-2(R15)   Spurnummer schreiben
      NOP
      MOVB R7,@-2(R15)   Seitennummer
      MOVB R3,R0         Spurnummer kopieren
      SRL  R0,8          ins Low-Byte
      SWPB R7            Seitennr. ins Low-Byte
      MPY  @>4635(R7),R0 je nach Seite multiplizieren
      SWPB R7            Seitennr. korrigieren
      A    R2,R1         Sektornummer addieren
      DIV  @>4632,R0             durch 9 teilen
      MOVB @>464F(R1),@-2(R15)  Sektornummer aus Interlace-Tabelle
      LI   R6,>FFEC              negativer Offset der Tafel
43F8  MOVB @>464E(R6),@-2(R15)  AF fertig und ID-GAP (GAP 2) schreiben
      INC  R6                   alle Bytes des GAPs?
      JNE  >43F8         Nein, GAP komplettieren
      LI   R0,>E5E5      Formatierungsdaten
      BL   @>4474        VDP-Fill
      DATA >0100             256 Bytes
      MOVB @>464E,@-2(R15)   >F7, CRC auslösen
      SETO R0                >FF schreiben, GAP 3
      BL   @>4474        Füllen
      DATA >002D         45 Mal
      INC  R2            Sektorzähler+1
      CI   R2,9          fertig?
      JNE  >43BC         nein
      BL   @>4474        ja, Spur-Postambel, GAP 4
      DATA >00E7         füllen
      LI   R4,3          3 Versuche pro Spur
442C  MOVB @>4E(R9),R2   Pufferadresse holen
      BL   @>461E        als Leseadresse setzen
      BL   @>45CA        FDC-Kommandoroutine
      DATA >0B00         WRITE TRACK
      LI   R6,>0CA3      Anzahl Bytes einer Spur
      SBO  2               WAIT ENABLE
4440  MOVB @>FBFE(R15),R0  aus VDP-RAM lesen
      INV  R0              invertieren
      MOVB R0,@>5FFE       ins Datenregister
      MOVB @>FBFE(R15),R0  nochmal
      INV  R0
      MOVB R0,@>5FFE
      DECT 6             Zaehler minus 2
      JGT  >4440         noch nicht fertig
      BL   @>4480    Befehlsende abwarten
      SLA  R0,11     WRITE PROTECT?
      JNC  >4464     Nein
```

```
      B    @>429A    Ja, Fehlersequenz des Sektor-I/O-Segments verwenden
4464  SLA  R0,4      LOST DATA?
      JNC  >4472     Nein
      DEC  R4        Ja, noch ein Versuch erlaubt?
      JNE  >442C     Ja
      BL   @>45AC    Nein, Fehler >33
      DATA >3300
4472  B    *R8       Eine Spur ist komplett, Ende
*
* Füllroutine
*
4474  MOV  *R11+,R6     Anzahl holen
4476  MOVB R0,@-2(R15)  R0 wiederholt ins VDP-RAM
      DEC  6            fertig?
      JNE  >4476        nein
      RT                Ja, Ende
*
4480  SBZ  2           Wait disable
4482  MOVB @>5FF0,R0   FDC-Statusregister lesen
      INV  R0          invertieren
      JLT  >4490       High-Bit gesetzt, DRIVE NOT READY, Index-Puls n.i.O.
      SRC  R0,9        Befehl noch nicht beendet?
      JOC  >4482       Ja, warten
      RT               Befehl beendet, Ende
*
4490  BL   @>45AC      Fehler >06: - Hardware-Fehler
      DATA >0600
*
* Gefordertes Laufwerk zuschalten
*
4496  MOV  R11,R7
      MOV  @>58(R9),R2  Start des File-Blocks im VDP-RAM
      AI   R2,-10       10 abziehen
      BL   @>461E          R2 als VDP-Leseadresse setzen
      MOVB @>FBFE(R15),R0  Nr. des zuletzt benutzten Laufwerks lesen
      CLR  R5              LW-Index
      CB   R0,@>4C(R9)  gleiches LW wie zuvor?
      JEQ  >44B2        Ja
      SETO R5           Nein, Index setzen
44B2  CLR  R0
      MOVB @>4C(R9),R0  Laufwerknummer
      JEQ  >451E        Null - Fehler!
```

20. ROM-Listing CorComp-Controller

ROM listing CorComp Controller

```
*
* System-RAM 2114  Da es sich um ein 4-Bit-RAM handelt, ist je Byte nur das
*                  High-Nibble dekodiert, das Low-Nibble ist immer 0!
*
4000   BYTE 0   Puffer für zuletzt angesprochenes Laufwerk
*
4001   BYTE 0   PAD-RAM Save- und Verify-Index.
*               >10 Kein Verify
*               >20 Leseroutinen im PAD-RAM
*
4002   DATA 0   Letzte Spur auf Laufwerk 1
4004   DATA 0   Letzte Spur auf Laufwerk 2
4006   DATA 0   Letzte Spur auf Laufwerk 3
4008   DATA 0   Letzte Spur auf Laufwerk 4
*
400A   BYTE 0  Dichtemaske, wird im Low-Nibble HBy R5 verwaltet
*               Jedes gesetzte Bit bedeutet Single Density
*
400B   BYTE 0  I/O-Optionsbits
*               >10   Formatieren mit externem Interlace
*               >20   Transfer FDC <-> RAM
*               >40   Systemaufruf (CALL MANAGER), kein normaler Rücksprung
*
* Sicherungszone für das PAD-RAM
*
400C   DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0
*
*...
*
* VDP-Adresse (R0) setzen
*
4390   ORI  R0,>4000     Schreibadresse
4394   SWPB R0           Leseadresse
       MOVB R0,*R15
       SWPB R0
       MOVB R0,*R15
       ANDI R0,>3FFF     Registerinhalt korrigieren
       RT
*
* R1 (Wort) ins VDP-RAM schreiben
*
43A2   SWPB R1
       MOVB R1,@>8C00
       SWPB R1
       MOVB R1,@>8C00
       RT
*
*...
*
```

```
43CC   TEXT '(C) 1984 BY Millers Graphics'
*
43E8   DATA >0100,>0200,>0400,>0800  Test-Bitmuster für Dichte je Laufwerk
43F0   DATA >1000,>2000,>4000,>8000
43F8   DATA >0001,>0002,>0004,>0008
4400   DATA >0010,>0020,>0040,>0080
4408   DATA >0020,>0800             Bitmuster zur Transferrichtung
*
440C   DATA >0009  Sektoren pro Spur SD
440E   DATA >0012  Sektoren pro Spur DD

*
4410   DATA >0001,>0000
4414   DATA >04FC,>FFFE,>F74E,>01F7,>FFFF,>FFFF
4420   DATA >FFFF,>FFFF,>FFFF,>FF00,>0000,>0000
442C   DATA >00FB
*
* Sektor-Interlace Single Density (IL=3)
*
442E   BYTE >00,>07,>05,>03,>01,>08,>06,>04,>02
       EVEN
*
* ID-GAP Double Density
*
4438   BYTE 0,0,0,0,0,0,0,0,0,0,0
4443   BYTE >F5,>F5,>F5,>FB
*
* Sektor-Interlace Double-Density (IL=4)
*
4447   BYTE >00,>0B,>04,>0F,>08,>01,>0C,>05,>10
       BYTE >09,>02,>0D,>06,>11,>0A,>03,>0E,>07
       EVEN
*
* ROUTINE >10  Sektor lesen/schreiben
*
445A   MOV  R11,@>48(R9) R11 nach >8348
       BL   @>4618       PAD-RAM sichern, I/O-Routinen kopieren
       MOV  @>400A,R5    Dichte/Modus-Index lesen
       ANDI R5,>F0F0     relevante Bits maskieren
       MOV  R5,R0        kopieren
       SRL  R5,4         ins Low-Nibble kopieren
       SWPB R0           Rest ins High-Nibble High-Byte
       SOC  R0,R5        ehemaliges Low-Byte ins High-Nibble
       ANDI R5,>FF00     nur das High-Byte zählt
       BL   @>45F8       Step-Zeit des LW ins High-Nibble Low-Byte einbauen
       LI   R4,>000C     12 Versuche sind erlaubt
*
* Einsprung bei Retry
*
447E   MOVB @>4412,@>50(R9) Fehlerbyte ad hoc löschen
       BL   @>49E2          Laufwerk zuschalten
       CLR  R10             Register für Laufwerknummer löschen
       MOVB @>4C(R9),R10    Laufwerknummer holen
       SRL  R10,7           mal 2 ins Low-Byte
```

```
       COC  @>43E6(R10),R5  Dichtemaske in Bezug auf das LW prüfen
       JEQ  >449A           Bit gesetzt -> Single Density
       SBZ  >A           Bit nicht gesetzt -> DDEN-Leitung aktivieren
       JMP  >449C        weiter
449A   SBO  >A              Single Density
449C   MOV  @>4000(R10),R0  Letzte Spur auf diesem Laufwerk (Kopfstellung)
       ANDI R0,>F0F0        High-Nibbles maskieren
       MOV  R0,R1        zur Rechnung kopieren
       SLA  R1,4         in die Low-Nibbles
       SOC  R1,R0        Bits anpassen
       ANDI R0,>FF00     High-Byte isolieren
       CI   R0,>2700     Spur über 39?
       JLE  >44BA        Nein, Laufwerk wurde bereits benutzt
       BL   @>4A32       FDC-Kommando RESTORE, Kopf über Spur Null fahren
       CLR  R0           Spurnummer = 0
44BA   SBZ  7            Diskettenunterseite anwählen
       MOVB R0,@>5FFA    momentane Spur ins Spurregister des FDC-Chips
       MOV  @>4A(R9),R1  Sektornummer holen
       CLR  R0              Rechenregister löschen (32-Bit Division!)
       COC  @>43E6(R10),R5  Ist dem Laufwerk Doppelte Dichte zugeordnet?
       JEQ  >44D2           Nein

       DIV  @>440E,R0    Ja, durch >0012 teilen
       JMP  >44D6
44D2   DIV  @>440C,R0    SD, durch >0009 teilen
44D6   CI   R0,>0028     Spurnummer über 40?
       JL   >44E8        Nein, der betr. Sektor befindet sich auf der Oberseite
       LI   R3,>004F     Ergänzungswert
       S    R0,R3        Spurnummer davon abziehen
       JLT  >4586        negativ -> Sektornummer war zu groß
       SBO  7            Nummer O.K. -> Disk-Unterseite schalten
       MOV  R3,R0        Spurnummer kopieren
44E8   SWPB R0              ins High-Byte
       MOVB R0,@>4000(R10)  Spurpuffer aktualisieren
       SRC  R0,12           Bitpositionen anpassen
       MOVB R0,@>4001(R10)  und die restlichen Bits ablegen
       SRC  R0,4
       MOVB R0,@>5FFE    Nummer ins Sektorregister (hätte auch schon früher
*                        gemacht werden können!)
       SWPB R1           Sektornummer ins High-Byte
       MOVB R1,@>5FFC    ins Sektorregister
       CB   R0,@>5FF2    ist die Spur bereits aktuell?
       JEQ  >4514        Ja, Kopf bleibt da, wo er ist
       BL   @>4AA6       Nein, FDC-Kommando
       DATA >1C00        SEEK TRACK
       BL   @>49D0       Befehlsende abwarten
       SLA  R0,13        SEEK ERROR Bit herausschieben
       JOC  >4580        Gesetzt -> Seek Error!
4514   MOV  @>4E(R9),R2  Pufferadresse holen
       MOVB @>4D(R9),R0  ebenso den I/O-Code
       JEQ  >458C        0 -> schreiben
       COC  @>43F2,R5    RAM-Transfer-Bit aktiv?
       JEQ  >452C        Ja, VDP-Setup umgehen
       BL   @>4AE0       VDP-Schreibadresse setzen
```

```
       LI   R2,>8C00     Zeiger laden
*
* Verify-Einsprung
*
452C   BL   @>4AB2       FDC-Kommando
       DATA >8C00        READ SECTOR
       LI   R6,>0100     256 Byte sind Standard
       SOCB @>43F6,R5    Fehlerbit a priori setzen
       CLR  R0           überflüssig!
       SBO  >02          WAIT ENABLE
       MOVB @>4D(R9),R0  I/O-Code
       JEQ  >4548        Schreiben, also jetzt Verify
       BL   *R9          Routine im PAD aufrufen
       JMP  >454C        Weiter mit Fehlerprüfung
4548   BL   @>A(R9)      Verify-Routine steht 10 Byte hinter der Leseroutine
454C   SZCB @>43F6,R5    Fehlerbit löschen (wird nur beim normalen Ende gemacht)
4550   BL   @>49CE       Busy-Ende (Befehlsende) abwarten
       SLA  R0,13        Record-not-found-Bit gesetzt?
       JOC  >4568        Ja, Fehler >21
       JLT  >456E        CRC-Fehler -> Fehler >22
       COC  @>43F6,R5    Abbruch-Bit gesetzt?
       JEQ  >457A        Ja, Verify wurde abgebrochen! -> Fehler >28
       SLA  R0,2         Lost Data?
       JOC  >4574        Ja, Fehler >23
4564   B    @>46BE       Befehl zu Ende bringen
*
4568   BL   @>4A60       Fehler >21: Sektor nicht gefunden (RNF)
       DATA >2100
456E   BL   @>4A60       Fehler >22: CRC-Fehler (Prüfsumme)
       DATA >2200
4574   BL   @>4A60       Fehler >23: Lost Data
       DATA >2300

457A   BL   @>4A60       Fehler >28: Verify-Fehler
       DATA >2800
4580   BL   @>4A60       Fehler >11: Spur nicht verifiziert
       DATA >1100
4586   BL   @>4A78       Fehler >07: Programmparameter falsch
       DATA >0700
*
* Ausführung Sektor schreiben
*
458C   COC  @>43F2,R5    Transfer-Bit gesetzt?
       JEQ  >459A        Ja -> RAM-Transfer
       BL   @>4AF0       Nein, VDP-Leseadresse setzen
       LI   R2,>8800     Pointer auf VDPRD
459A   BL   @>4AB2       FDC-Kommando
       DATA >AC00        WRITE SECTOR
       LI   R6,>0100     256 Bytes
       SBO  >02          WAIT ENABLE
       BL   *R9          Routine im PAD-RAM ausführen
       BL   @>49CE       Busy-Ende
       SLA  R0,11        Write Protect?
       JOC  >45D6        Ja, Abbruch mit Fehler >34
```

```
       SLA  R0,2          Record not found?
       JOC  >45DC         Ja, Fehler >31
       SLA  R0,2          Lost Data?
       JOC  >45E2         Ja, Fehler >33
       MOVB @>4001,R2     Turbo-Option
       COC  @>43F0,R2     Letztes Bit im High-Nibble HB gesetzt?
       JEQ  >4564         Ja, Routine beenden, kein Verify!
       MOV  @>4E(R9),R2   Nein, Pufferadresse holen
       COC  @>43F2,R5     RAM-Transfer-Bit
       JEQ  >452C         Ja, Verify von dort
       BL   @>4AF0        Nein, VDP-Zugriff vorbereiten
       LI   R2,>8800      Zeiger
       JMP  >452C         nun Verify aus VDP-RAM
*
45D6   BL   @>4A78        Fehler >34: Disk ist schreibgeschützt!
       DATA >3400
45DC   BL   @>4A60        Fehler >31: Sektor nicht gefunden (RNF)
       DATA >3100
45E2   BL   @>4A60        Fehler >33: Datenverlust (LOST DATA)
       DATA >3300
*
45E8   TB   >12   Step-Einstellung
       TB   >16          LW 1
       TB   >15   Step-Einstellung
       TB   >0F          LW 2
       TB   >14   Step-Einstellung
       TB   >0E          LW 3
       TB   >13   Step-Einstellung
       TB   >11          LW 4
*
* Step-Timing Bits je nach LW in R5 einbauen
*
45F8   CLR  R10           Rechenregister vorbereiten
       MOVB @>4C(R9),R10   Laufwerknummer
       SRL  R10,6         mal 4 ins Low-Byte
       X    @>45E4(R10)   F2-Bit testen
       JNE  >460A         Schalter an, schneller
       ORI  R5,>0040      Schalter offen, langsam
460A   X    @>45E6(R10)   F1-Bit testen
       JNE  >4614         nicht aktiv
       ORI  R5,>0080      doch -> Code komplettieren
4614   SRL  R10,1         Laufwerknummer durch 2 (vergebene Mühe, die rufenden
*                         Routinen nutzen das nicht aus!)
       RT                 Ende
*
* I/O-Routinen kopieren, PAD-RAM dabei sichern
*
4618   MOV  R11,R7          R11 sichern
       SOCB @>43F2,@>4001   Index, welche Routinen kopiert wurden
       LI   R6,5            5 Worte für die Leseroutine
       MOVB @>4D(R9),R0   I/O-Code
       JNE  >4634         nicht 0 -> lesen
       AI   R6,8            Schreiben -> 8 Worte mehr (incl. Verify)
       SZCB @>43F2,@>4001   Schreiben indizieren
```

```
4634   MOVB @>4C(R9),R0    Laufwerknummer
       JEQ  >4694          Null -> Fehler >07
       CB   R0,@>4414    gegen 4 vergleichen
       JH   >4694        zu groß -> auch nichts!
       MOV  R9,R2        R2 = >8300
       LI   R3,>400C     Zeiger auf den betreffenden Bereich des System-RAMs
*
4646   MOV  *R2+,R0      PAD-Inhalt nach R0
       MOV  R0,*R3+      zuerst die High-Nibbles
       SLA  R0,4         Rest verschieben
       MOV  R0,*R3+      und nun die High-Nibbles
       DEC  R6           alles durch?
       JNE  >4646        noch nicht
       MOV  R9,R2        PAD-Start wieder holen
       MOVB @>4D(R9),R0  I/O-Code
       JNE  >4664        Lesen
*
       LI   R3,>46A4     Schreib- und Verifyroutinenadresse
       LI   R6,>000D     13 Worte
       JMP  >466C        weiter
4664   LI   R6,>0005     5 Worte
       LI   R3,>469A     Adresse Leseroutine
466C   MOV  *R3+,*R2+    Routinen verschieben
       DEC  R6           Fertig?
       JNE  >466C        Nein
       MOVB @>400B,R8    Transfer-Index
       COC  @>43F2,R8    CPU-RAM-Transfer?
       JNE  >4692        Nein
       MOVB @>4D(R9),R0  Ja, Lesen oder schreiben?
       JNE  >468E        Lesen
       SOC  @>4408,*R9     Schreiben, im TS-Feld ein *R2+ erzeugen
       SOC  @>440A,@>A(R9) dto. im TD-Feld der Verify-Sequenz
       JMP  >4692
468E   SOC  @>440A,*R9   Lesen, im TD-Feld ein *R2+ erzeugen
4692   B    *R7          Ende
*
4694   BL   @>4A78       Fehler >07: z.B. Laufwerknummer falsch
       DATA >0700
*
469A   MOVB @>5FF6,*R2   Sektorleseroutine
       DEC  R6
       JNE  >469A
       RT
*
46A4   MOVB *R2,@>5FFE   Sektorschreibroutine
       DEC  R6
       JNE  >46A4
       RT
*
46AE   CB   @>5FF6,*R2   Datenvergleichsroutine
       JNE  >46BA
       DEC  R6
       JNE  >46AE
       RT
```

```
46BA   B    @>4550

*
* Ende des Sektor-I/O
*
46BE   LI   R6,>0005
       MOVB @>4001,R0    Welche Routinen wurden kopiert?
       COC  @>43F2,R0    Leseroutinen?
       JEQ  >46D0        Ja
       AI   R6,>0008     Nein, Schreibroutinen -> mehr ist wiederherzustellen!
46D0   ANDI R5,>4F00     Transfer- und Dichtebits in R5 isolieren
       MOVB R5,@>400B    ablegen
       SRC  R5,12
       MOVB R5,@>400A    dto. den Rest
       SRC  R5,4
       MOV  @>58(R9),R2  Adresse des Systembereichs im VDP-RAM
       AI   R2,>FFF6     auf Drive-Info-Block zeigen
       BL   @>4AE0       Schreibadresse
       MOVB @>4000,R0    Nummer des zuletzt benutzten Laufwerks
       SRL  R0,4         anpassen
       MOVB R0,@>8C00    in den Disk-Info-Block
       MOV  R9,R2        PAD-Start
       LI   R3,>400C
46FC   MOV  *R3+,R0      PAD wiederherstellen
       ANDI R0,>F0F0     irrelevante Nibbles löschen
       MOV  *R3+,R1      nächster Anteil
       ANDI R1,>F0F0     maskieren
       SRL  R1,4         anpassen
       SOC  R1,R0        zusammenflicken
       MOV  R0,*R2+      ins PAD-RAM
       DEC  R6           alle Bytes restauriert?
       JNE  >46FC        noch nicht
       COC  @>43F4,R5    welcher Art war der Routinenaufruf?
       JNE  >471E        Standard
       MOV  @>48(R9),R11 Nein, System-Aufruf (CALL MGR etc.)
       RT                Internes Ende
*
471E   B    @>567E       Standard-Exit
*
4722   DATA >F000
*
* Diskette formatieren
*
4724   CLR  R1
       MOVB @>4D(R9),R1  Geforderte Anzahl Spuren
       CI   R1,>2800     40?
       JLE  >473C        weniger
       SRL  R1,1         Mehr -> durch 2 teilen
       MOVB R1,@>4D(R9)      wieder zurück
       MOVB @>43EA,@>51(R9)  >02, egal was war - jetzt wird's Double Sided!
473C   MOVB @>51(R9),R1      Seitenzahl
       JEQ  >4748        Null, dann wird's >01
       CB   R1,@>43EA    Wert gültig?
       JLE  >474E            Ja
```

```
4748   MOVB @>43E8,@>51(R9)  >01 wenn's nicht >02 war
474E   SZCB @>4722,@>4C(R9)  Laufwerk normieren
       MOVB @>4D(R9),R1      Spuranzahl
       MOVB @>43E9,@>4D(R9)  I/O-Code Schreiben
       BL   @>4618           Routinen ins PAD-RAM kopieren
       MOVB R1,@>4D(R9)      R1 ist unverändert, Spurnummer zurück
       MOV  @>400A,R5    Optionen
       ANDI R5,>F0F0     relevante Bits maskieren
       MOV  R5,R0        Kopie für Rechnung
       SRL  R5,4         Low-Nibble bilden
       SWPB R0           High-Nibble hoch
       SOCB R0,R5        Byte erzeugen
       ANDI R5,>FF00     Optionsbits und Dichtemaske im HByte
       BL   @>45F8       Step-Zeit ermitteln
       BL   @>49E2       Laufwerk schalten
       SBZ  7            Default Seite 0
       BL   @>4A32       Restore Drive
       CLR  R10              Spur-Index
       CB   @>50(R9),@>43EA  Dichte >02?
       JNE  >4798            Nein
       SBZ  >A           Ja, doppelte Dichte schalten
       B    @>489C       In DD-Formatierroutine springen
*
* Disk in Single Density formatieren
*
4798   SBO  >A           SD schalten
       CLR  R3           Spurnummer
       MOV  @>4E(R9),R2  Pufferadresse VDP
       BL   @>4AE0       als Schreibadresse setzen
       CLR  R2           Sektornummer
       SETO R0           >FF
       BL   @>49C2       Index-GAP
       DATA >000C        12 Bytes, nicht 22 wie üblich!
47AE   CLR  R0           >00
       BL   @>49C2       Füllen
       DATA >0006          6 Bytes
       MOVB @>4417,@>8C00  >FE, AMID
       NOP
       MOVB R3,@>8C00    Spurnummer
       NOP
       MOVB R10,@>8C00   Seitenindex
       COC  @>43F0,R5    Interlace aus ROM?
       JEQ  >47D6               Nein
*
* Befehlsausführung testen
*
49CE   SBZ  >02          Wait-Disable
49D0   MOVB @>5FF0,R0    Status-Byte lesen
       JLT  >49DC        Not Ready!
       SRC  R0,9         FDC noch BUSY?
       JOC  >49D0        Ja, warten
       RT                Ende
*
49DC   BL   @>4A78       Fehler melden
```

```
       DATA >0600        >06, Hardware-Fehler
*
* Laufwerk zuschalten
*
49E2   MOV  R11,R7
       SZCB @>43F6,R5   Index 'altes Laufwerk'
       MOVB @>4000,R0   letztes Laufwerk
       SRL  R0,4         rechtsbündig im High-Byte
       CB   R0,@>4C(R9)  wie neues Laufwerk?
       JEQ  >49F8        Ja
       SOCB @>43F6,R5    Nein, vermerken
49F8   MOVB @>4C(R9),R0  Laufwerknummer holen
       SLA  R0,4         auf 4-Bit-RAM anpassen
       MOVB R0,@>4000  in Puffer übernehmen
       SRL  R0,12      rechtsbündig im Low-Byte
       LI   R2,>0080   Walking-Bit
       SLA  R2,0       entsprechend schieben
       AI   R12,8      CRU-Basis auf Drive-Select
       LDCR R2,4       LW anschalten
       AI   R12,-8     Basis reparieren
       SBZ  8          LW 4 aus
       CI   R0,4       eben das gefragt?
       JNE  >4A1E      Nein
       SBO  8          Ja, zuschalten
4A1E   COC  @>43F6,R5  anderes Laufwerk?
       JNE  >4A30      Nein, fertig
       LI   R0,>0C80   Ja, Umschaltpause
4A28   SWPB R5         Dummy-Befehl
       SWPB R5
       DEC  R0         Schluß?
       JNE  >4A28      Noch nicht!
4A30   B    *R7        Aber jetzt
*
4A32   MOV  R11,R8     Rückkehr sichern
       BL   @>4AA6     FDC-Kommando
       DATA >0800      RESTORE
4A3A   MOVB @>5FF0,R0  Statusregister lesen
       JLT  >49DC      NOT READY
       SRC  R0,9       FDC noch Busy?
       JOC  >4A3A      Ja, warten
      MOVB @>5FF0,R0  Status nochmal holen (überflüssig wegen Circular Shift!)
       SLA  R0,6       TRK 00 aktiv?
       JOC  >4A50      Ja, Spurnummer auf Null setzen
       B    @>49DC     Fehler >06
*
* Laufwerk konnte justiert werden (Spur Null korrekt erreicht)
*
4A50   CLR  R6           Spurnummer = 0
       MOVB @>4C(R9),R6  Laufwerknummer
       SRL  R6,7               als Wort-Zeiger (mal 2)
       MOV  @>4412,@>4000(R6)  >00 in Spurpuffer
       B    *R8                Ende
*
* Fehlerbehandlung mit Retries
```

```
*
4A60   DEC  R4           Fehlerzähler runter
       JEQ  >4A78        Null -> Vorbei!
       CLR  R1           Rechenregister
       MOVB @>4C(R9),R1  Laufwerknummer
       SRL  R1,7           mal 2 ins Low-Byte
       XOR  @>43E6(R1),R5  betreffende Dichte umschalten
       BL   @>4A32         RESTORE, Kopf über Spur Null
       B    @>447E       weiter
*
* Fehler ohne Retry-Möglichkeit
*
4A78   MOV  *R11+,R0     Fehlercode holen
       MOVB R0,@>50(R9)  an die rufende Software weitergeben
       CI   R0,>0600     Hardware-Fehler?
       JNE  >4AA2        Nein
       BL   @>4AB2       Ja, FDC-Chip rücksetzen
       DATA >D000        FORCE INTERRUPT
4A8A   MOVB @>5FF0,R0    Statusregister lesen
       SRC  R0,9         BUSY-Bit herausschieben
       JOC  >4A8A        Chip ist noch nicht bereit, warten
       CLR  R6           Rechenregister
       MOVB @>4C(R9),R6  Laufwerknummer
       SRL  R6,7         mal 2 als Wort-Zeiger
       LI   R0,>FFFF      Au weia...
       MOV  R0,@>4000(R6) Spurnummer zu >FF machen, erzwingt Restore beim
*                          nächsten Zugriff auf dieses Laufwerk
4AA2   B    @>46BE       PAD-RAM wiederherstellen etc.
*
* FDC-Kommandoroutine
*
4AA6   MOV  R5,R0      Step-Maske und Optionsbits kopieren
       SLA  R0,2       in Position
       ANDI R0,>0300   normieren (Options-Bits stören hier)
       SOC  *R11+,R0   Kommando einbauen
       JMP  >4AB4      weiter
4AB2   MOV  *R11+,R0   Kommando übernehmen
4AB4   MOVB @>5FF0,R6  Status-Register
       SLA  R6,1       Ready ins Carry
       SBZ  1          Motor-Strobe
       SBO  1
       JNC  >4ACC      LW ist bereit
       LI   R6,>80E8   Pausenzähler
4AC4   SWPB R5         Dummy zum Erhalt von R5
       SWPB R5
       DEC  R6         Pause um?
       JNE  >4AC4      Nein
4ACC   MOVB R0,@>5FF8  Op-Code in Kommandoregister
       SBO  3          Head-Load
       SWPB R5         Pause vor nächstem Zugriff
       SWPB R5         (Platzverschwendung!)
       SWPB R5
       SWPB R5
       SWPB R5
```

```
       SWPB R5
       RT              Ende
*
* VDP-Schreibadresse mit Wert in R2 setzen
*
4AE0   MOVB @>E5(R9),*R15  Low-Byte R2
       ORI  R2,>4000       High-Byte anpassen
       MOVB R2,*R15      High-Byte setzen
       ANDI R2,>3FFF     Befehlsbit wieder löschen
       RT
*
* VDP-Leseadresse mit Wert in R2 setzen
*
4AF0   MOVB @>E5(R9),*R15  Low-Byte R2
       ANDI R2,>3FFF       Lesebefehl garantieren
       MOVB R2,*R15      High-Byte nachschieben
       RT                Ende
*
* Fortsetzung des Power-Up-Link
*
4AFC   AI   R12,8
       LDCR @>4412,5     Laufwerke alle aus
       AI   R12,-8
       SBZ  1            Motoren anwerfen
       SBO  1
       MOVB @>4A88,@>5FF8  Interrupt-Code als Software-Reset
       LI   R0,>0F0F       Preset-Werte für's System-RAM
       LI   R6,>4000   Beginn der RAM-Zone
       MOV  R0,*R6+    Letztes Laufwerk 0, Verify an
       SLA  R0,4       ab jetzt >F schreiben
       MOV  R0,*R6+    Laufwerk 1 nicht justiert
       MOV  R0,*R6+    dto. Laufwerk 2
       MOV  R0,*R6+    dto. Laufwerk 3
       MOV  R0,*R6+    dto. Laufwerk 4
       SRL  R0,4       wieder >0 schreiben
       MOV  R0,*R6     Dichte komplett DD, Transfer in VDP, ROM-Interlace
       BL   @>49CE           auf Abarbeitung des Interrupts warten
       MOVB @>4412,@>50(R9)  Fehlerbyte auf >00
       B    @>567E           Ende
*
* ...
*
* Beginn der Operationen zur Rückkehr ins untere ROM
*
567E   MOV  @>8366,R11   VDP-Stackpointer
       MOVB @>83F7,*R15  Low-Byte R11 in VDPWA
       ANDI R11,>3FFF    maskieren
       MOVB R11,*R15     High-Byte in VDPWA
       INCT @>8366       Zeiger korrigieren
       MOVB @>8800,R11   Low-Byte der Rückkehradresse
       SWPB R11
       MOVB @>8800,R11   High-Byte der Rückkehradresse
       B    @>5F78       ROM ausblenden etc.
*
```

```
* ...
*
* Äbergang vom unteren ins obere ROM
*
5F70   SBO  >0B            Oberes ROM anschalten
       MOV  @>4080(R1),R1  Zeiger zur Routine aus Liste
       B    *R1            anspringen
*
* Rücksprung ins untere ROM
*
5F78   SBZ  >0B   Oberes ROM aus
       RT         Ende (R11 aus VDP-Stack)
```

21. ROM-Listing Atronic-Controller

ROM listing Atronic Controller

```
4000 bis 400F      DATA 0        16 Byte Nullen um Fehlfunktionen zu vermeiden
                                 falls versehentlich Bit >B an ist
4010 bis 49EF      DATA >FFFF    nicht programmiert

4A00 bis 4AFF      System-RAM des Controllers (2114). Pro Byte ist nur das Low-
                   Nibble verfügbar!

   4A00            Nummer des zuletzt angesprochenen Drives
   4A02            Letzte Spur auf LW 1
   4A04            Letzte Spur auf LW 2
   4A06            Letzte Spur auf LW 3
   4A08            Letzte Spur auf LW 4
   4A0A bis 4A18   keine Verwendung
   4A1A            Density-Bitmap für Sektor-I/O
   4A1B bis 4A1F   keine Verwendung
   4A20 bis 4A53   Puffer für den Inhalt des PAD-RAMs während der Sektor-
                   I/O-Routinen die wegen des Timings den 16 Bit-Bus
                   brauchen!
   4A54 bis 4AFF   keine Verwendung

Vektoren für den Übergang vom unteren in das obere ROM

4B00   DATA >484D  Nicht zugewiesen, der Bereich ist leer.
4B02   DATA >4B16  Fortsetzung des Powerup-Links.
4B04   DATA >4BDE  Sektor Schreib/Lese-Operationen.
4B06   DATA >4F30  Diskette formatieren.
4B08   DATA >51CC  Fortsetzung von CALL MANAGER.

4B0A bis 4B14      DATA 0   Leer

Fortsetzung des Power-Up-Link.

4B16   AI   R12,8          CRU-Basis hochschalten
       LDCR @>4BB6,5       Drive-Select- und Side-Select-Leitungen auf Ground.
       AI   R12,-8         Alte Basis (>1100) wiederherstellen
       SBZ  1
       SBO  1              Drive-Motoren anwerfen (Monoflop triggern).
       MOVB @>4EC6,@>5FF8  FDC-Chip mit >D0 rücksetzen.
                           Force Interrupt, Terminate with no Interrupt.
      MOV  @>58(R9),R2    Aus >8358 die VDP-Adresse des Volume-ID Blocks holen.
       AI   R2,-10         auf die VDP-Startadresse der DISK DRIVE INFO zeigen.
      BL   @>4E7C         VDP-Adresse mit Wert in R2 zum Schreiben vorbereiten.
       LI   R0,7
4B3C   MOVB R0,@-2(R15)    R15 enthält VDPWA da GPLWS, VDPWA-2 = VDPWD !!
       DEC  R0             Disk-Info und 3 folgende Bytes löschen.
       JNE  >4B3C          noch nicht fertig!
      BL   @>4B94           FDC Wait disable, warten bis BUSY-Zustand beendet.
       MOVB @>4BB6,@>50(R9)  Wert in >8350 (HBy) löschen.
```

```
System-RAM 2114 initialisieren.

       CLR  R0           Werte vorbereiten
       SETO R2
       LI   R6,>4A00     Anfangsadresse des 4 Bit RAM
       MOV  R0,*R6+      Letztes Drive = 0
       MOV  R2,*R6+      Letzte Spur auf LW1 = >FF. Das zwingt die Sektor-
       MOV  R2,*R6+      Routine zum Initialisieren der aktuellen Spur.
       MOV  R2,*R6+
       MOV  R2,*R6+      alle 4 Laufwerke.
                        Der VDP-Puffer ab >3EEB ist normalerweise nur bis LW 3
                         ausgelegt, er wird aber nicht benutzt.
       MOV  R0,@>4A1A    Density-Bitmap komplett auf Double Density.

Meldung in das Titelbild setzen.

       LI   R0,>01A6       Bildschirmposition der Meldung.
      BL   @>4BA6         VDP-Adresse mit Wert in R0 zum Schreiben vorbereiten.
       LI   R0,>4B80       Pointer auf den Text
       LI   R2,>14         Länge des Textes
4B74   MOVB *R0+,@-2(R15)  Daten sukzessive in VDPWD schreiben
       DEC  R2             fertig?
       JNE  >4B74          nein - weiter
4B7C   B    @>5F68         ROM ausblenden und Rückkehr aus DSR.

4B80   TEXT '  DISK SYSTEM READY '

4B94   SBZ  2              FDC Wait disable.
4B96   MOVB @>5FF0,R0      FDC Status lesen.
       JLT  >4BA2          High-Bit gesetzt - Drive not ready!
       SRC  R0,9           Low-Bit gesetzt ?
       JOC  >4B96          Ja - noch Busy, weiter warten.
       RT                  Low-Bit auf Null - Ende.
4BA2   B    @>50B8         Fehler - Abbrechen!

VDP-Adresse mit Wert in R0 zum Schreiben setzen.

4BA6   ORI  R0,>4000       Schreib-Operation ankündigen.
       SWPB R0             Low-Byte zuerst.
       MOVB R0,*R15        in VDPWA
       SWPB R0
       MOVB R0,*R15        High-Byte zuletzt.
       RT                  Ende.

4BB4   DATA >0000
4BB6   DATA >0001,>0002,>0004,>0008
4BBE   DATA >0010,>0020,>0040,>0080
4BC6   DATA >0100,>0200,>0400,>0800
4BCE   DATA >1000,>2000,>4000,>8000

4BD6   DATA >0012          Anzahl der Sektoren pro Spur bei Double Density.
4BD8   DATA >0009          Anzahl der Sektoren pro Spur bei Single Density.
4BDA   DATA >0004
4BDC   DATA >0200          Step-Timing Index (0000 bei  6 ms)
```

```
                                          (0100 bei 12 ms)
                          Fabrikeinstellung:(0200 bei 20 ms)
                                          (0300 bei 30 ms)
Sektor Lesen/Schreiben.

4BDE   MOVB @>4A1A,R5     alte Density-Bitmap in R5, R5 dient als Statuspuffer.
       ANDI R5,>0F00       relevante Bits maskieren.
       BL   @>4E0A         Laufwerk prüfen, Sektor I/O-Routine ins PAD setzen.
       LI   R4,10            Anzahl der zulässigen Versuche für Sektor-I/O.
       MOVB @>4BB6,@>50(R9)  >8350 (HBy) löschen (kein Fehler).
       BL   @>4DB0           Laufwerk-Nr. prüfen, Select-Leitung setzen.
       CLR  R7
       MOVB @>4C(R9),R7    LW-Nummer aus >834C holen.
       SRL  R7,7           LW-Nr. mal 2 im Low-Byte.
       MOV  @>4A00(R7),R0  Spur-Nr. des zuletzt benutzten LW holen.
       ANDI R0,>0F0F       Low-Nibs isolieren,
       MOV  R0,R1          kopieren,
       SLA  R1,12          High-Nib vom Low- ins High-Byte,
       AB   R1,R0          Low-Nib aufaddieren,
       ANDI R0,>FF00       und High-Byte isolieren.
       CI   R0,>2700       über die letzte Spur (ggf. 1. Zugriff auf das LW).
       JLE  >4C1E          Nein - weiter.
       BL   @>4D4E         Ja - Kopf über Spur 0 fahren (Restore).
       CLR  R0             Spur-Index zu Null setzen.
4C1E   MOVB R0,@>5FFA      geforderte Spur-Nr. in das FDC-Reg. Daran erkennt
                           der Controller die richtige Position über der Spur!
       CLR  R0
       MOV  @>4A(R9),R1    in >834A wurde die Sektor-Nr. kopiert.
       COC @>4BC4(R7),R5   ist das dem LW entsprechende Bit im Status gesetzt?
       JEQ  >4C36          Ja - dann ist dem LW Single Density zugeordnet!
       SBZ  >A             Nein - Double Density schalten!
       DIV  @>4BD6,R0      Wert in R0 (Sektor-Nr.) durch >12 teilen.
                           Nach der Division steht in R0 die Spur, in R1 der
                           Sektor in dieser Spur.
       JMP  >4C3C          weiter.
4C36   SBO  >A             Single Density schalten.
       DIV  @>4BD8,R0      durch >9 teilen.
4C3C   SBZ  7              Diskettenoberseite anwählen.
       SZC  @>4BD0,R5      Seitenindex im Statusregister löschen.
       CI   R0,40          Wurde eine Spur über 40 berechnet?
       JL   >4C58          Nein
       LI   R2,79          möglicherweise Seite 2.
       S    R0,R2         79 minus Spur-Nr. gibt Komplement, Spur auf Seite 2.
       JLT  >4D18          weniger als Null - Sektor-Nr. war zu hoch.
       SBO  7              Spur-Nr. i.O., Diskettenrückseite anwählen.
       SOC  @>4BD0,R5      Seitenindex setzen.
       MOV  R2,R0          Spur-Nr. in R2.
4C58   SWPB R0             Spur-Nr. ins Low-Byte.
       MOVB R0,@>4A00(R7)  Spur-Puffer aktualisieren.
       SRC  R0,4
       MOVB R0,@>4A01(R7)  Byte komplett ablegen.
       SLA  R0,4           Wort reparieren.
       MOVB R0,@>5FFE      Spur-Nr. für Seek in das Datenregister.
       SWPB R1             Sektor-Nr. ins High-Byte.
```

```
       MOVB R1,@>5FFC      Sektor-Nr. in Sektor-Register für später.
       CB   R0,@>5FF2      Track Register lesen, Spur schon erreicht?
       JEQ  >4C84          Ja - weiter.
       BL   @>4D82         Nein - SEEK, Load Head, Verify on Track.
       DATA >1C00          FDC-Kommando Typ I, mit Stepping Rate verODERn!
       BL   @>4B96        warten bis Busy beendet, Abbruch wenn Drive not ready
       SLA  R0,13          Status prüfen.
       JOC  >4D12          SEEK-Error - Abbruch, Retry möglich!
4C84   MOVB R1,@>5FFC      Sektor-Nr. für Verify in das Sektor Register.
       MOV  @>4E(R9),R2    PAB-Buffer Adresse holen.
       MOVB @>4D(R9),R0    Schreib-Lese Opcode holen.
       JEQ  >4CDA          wenn 00 dann schreiben.
       BL   @>4E7C         Adresse des PAB-Buffers setzen.
       MOV  R15,R2         VDPWA kopieren.
       AI   R2,-2          VDPWD erzeugen.
4C9C   BL   @>4D74        READ SECTOR, Single Record, Compare Side, 30 ms Delay
                           Enable Side Compare.
       DATA >8600          Kommandotyp II
       LI   R6,256         Byteanzahl für Sektor-R/W.
       SOCB @>4BD4,R5      High-Bit setzen.
       SBO  2              Wait enable.
       CLR  R0
       MOVB @>4D(R9),R0    Schreiben oder Lesen?
       JNE  >4CBC          wenn nicht Null, dann lesen.
       CLR  R0
       BL   @>A(R9)        Geschrieben wurde schon, jetzt Verify (Offset A!).
       JMP  >4CBE
4CBC   BL   *R9            entspricht BL  @>8300!
4CBE   SZCB @>4BD4,R5      High-Bit wieder löschen.
4CC2   BL   @>4B94         Wait-Disable etc., Verify-Error-Entry.
       SLA  R0,13          Record not found?
       JOC  >4D24          Ja - Abbruch!
       JLT  >4D2A          CRC-Error - auch Ende!
       COC  @>4BD4,R5      High-Bit gesetzt?
       JEQ  >4D36          Verify-Fehler!
       SLA  R0,2           Lost-Data (Timing-Fehler des Programms)?
       JOC  >4D30          Ja
       B    @>4EDA         R5 zurück nach >4A1A, PAD reparieren etc.  ENDE.

Sektor schreiben.

4CDA   BL   @>4E86       VDP-Adresse mit R2 zum Lesen setzen.
       MOV  R15,R2
       AI   R2,>FBFE     VDPRD herstellen.
       BL   @>4D74      WRITE SECTOR, Single Record, Compare Side, 30 ms Delay,
                          Enable Side Compare.
       DATA >A600        Kommandotyp II.
       LI   R6,256       Byte Count.
       SBO  2            Wait enable.
       BL   *R9          Sektor schreiben.
       BL   @>4B94       Wait disable etc.
       SLA  R0,11        Bit S6 prüfen.
       JOC  >4D3C        Fehler, Write Protect!
       SLA  R0,2         Bit S4 prüfen.
```

```
       JOC  >4D42        Fehler, Record not found!
       SLA  R0,2         Bit S2 prüfen.
       JOC  >4D48        Lost Data, Timing Fehler!
       MOV  @>4E(R9),R2  Buffer-Adresse holen.
       BL   @>4E86       VDP-Adresse mit R2 setzen (Lesen)
       MOV  R15,R2
       AI   R2,>FBFE     VDPRD erzeugen.
       JMP  >4C9C        Sektor lesen zum Verify benutzen.

Fehlerroutinenaufrufe    Aufruf wenn:

4D12   BL   @>4E98       Alle Retries fehlgeschlagen oder SEEK-Error.
       DATA >1100
4D18   BL   @>4E98       Sektor-Nr. zu gross.
       DATA >0700
4D1E   BL   @>4EB6       Drive-Nr. falsch.
       DATA >0700
4D24   BL   @>4E98       READ-Error, Record-Not-Found.
       DATA >2100
4D2A   BL   @>4E98       READ-Error, CRC-Error.
       DATA >2200
4D30   BL   @>4E98       READ-Error, Lost Data.
       DATA >2300
4D36   BL   @>4E98       Verify fehlgeschlagen.
       DATA >2800
4D3C   BL   @>4EB6       WRITE PROTECTED.
       DATA >3400
4D42   BL   @>4E98       WRITE-Error, Record not found.
       DATA >3100
4D48   BL   @>4E98       WRITE-Error, Lost Data.
       DATA >3300

Subroutinen

Kopf über Spur Null positionieren.

4D4E   MOV  R11,R8       R11 sichern.
       BL   @>4D82       RESTORE, Load Head at beginning, no Verify on Track!
       DATA >0800        Kommandotyp I, mit Stepping-Rates verODERn.
       BL   @>4B96       warten auf Busy-Ende.
       BL   @>4D66               Status ein Mal prüfen ob Spur 0 erreicht ist.
       MOV  @>4BB4,@>4A00(R7)    Spur-Puffer loeschen.
       B    *R8                  Ende.

Auf Hardware-Error prüfen, Track 0 nicht erreicht

4D66   MOVB @>5FF0,R0    FDC-Status lesen.
       SLA  R0,6         Bit S2 prüfen, TRK 00.
       JOC  >4D72        OK - weiter.
       B    @>50B8       Error-Exit mit I/O-Error 06! Kopf nicht über Spur 0!
4D72   B    *R11         Ende.

FDC-Kommando Typ II (Read/Write Sector)
```

```
4D74   MOV  *R11+,R0     Data holen.
       COC  @>4BD0,R5    Seite 2?
       JNE  >4D8C        Nein
       SOC  @>4BCC,R0    Ja - Data mit >0800 verodern.
       JMP  >4D8C

FDC-Kommando Typ I (Seek, Restore, Step)

4D82   MOV  *R11+,R0     Data holen.
       SOCB @>4BDC,R0    Je nach Step-Time unterschiedliche Werte in >4BDC!
       JMP  >4D8C

Allgemeine FDC-Kommandoroutine

4D8A   MOV  *R11+,R0     Data holen.
4D8C   MOVB @>5FF0,R6    Status lesen.
       SLA  R6,1         High-Bit gesetzt, Drive not ready?
       SBZ  1
       SBO  1            Motor anwerfen.
       JNC  >4DA4        Bit nicht gesetzt, Drive ready!
       LI   R6,>7530     Schleife.
4D9C   SRC  R5,4         Verzögerung, um den Motor hochlaufen zu
       SRC  R5,4         lassen.
       DEC  R6           Fertig?
       JNE  >4D9C        Nein.
4DA4   MOVB R0,@>5FF8    Data in FDC-Chip.
       SBO  3            Head-Load.
       SRC  R5,8
       SRC  R5,8
       B    *R11

Subroutine Laufwerk zuschalten

4DB0   SZCB @>4BD4,R5    High-Bit loeschen.
       MOVB @>4A00,R0
       ANDI R0,>0F00     Letztes LW, das benutzt wurde.
       CB   R0,@>4C(R9)  ist es das gewünschte LW?
       JEQ  >4DC6        Ja.
       SOCB @>4BD4,R5    Nein - New-Drive Index setzen.
4DC6   CLR  R0
       MOVB @>4C(R9),R0  aktuelle LW-Nr. holen.
       JEQ  >4D1E        Fehler, LW-Nr.=0!
       MOVB R0,@>4A00    aktuelles LW ist jetzt das letzte Drive.
       SRL  R0,8         ins Low-Byte.
       C    R0,@>4BDA    mit 4 vergleichen.
       JH   >4D1E        Fehler, LW-Nr.>4!
       LI   R2,>0080     Bitmuster für CRU
       SLA  R2,0         mit R0 links schieben
       AI   R12,8        CRU-Basis hoch, um die unteren Bits nicht zu ändern.
       LDCR R2,3         Drive-Select Leitung aktivieren.
       AI   R12,-8       Basis restaurieren.
       CI   R0,4         Laufwerk 4?
                         Hier steht in der Version 1.0  CI R2,4 weshalb das LW
4
```

```
                        nicht erkannt wird.
       JNE  >4DF4       nicht LW 4!
       SBO  8           Drive-Select separat für LW 4!
       JMP  >4DF6
4DF4   SBZ  8           LW 4 ggf. abschalten.
4DF6   COC  @>4BD4,R5   High-Bit gesetzt, neues Laufwerk?
       JNE  >4E08       nein - weiter.
       LI   R0,>0BB8    Schleife, Umschaltpause
4E00   SRC  R5,4
       SRC  R5,4
       DEC  R0
       JNE  >4E00
4E08   B    *R11        Ende.

4E0A   CLR  R7
       MOVB @>4C(R9),R0 LW-Nr. holen.
       JNE  >4E16       nicht Null -OK!
4E12   B    @>4D1E      Fehler, falsches LW!
4E16   CB   R0,@>4BDB   mit 4 vergleichen.
       JHE  >4E12       zu hoch, Fehler!
       LI   R6,>A       Anzahl der Bytes der Leseroutine.
       MOVB @>4D(R9),R0 Schreib/Lese-Code.
       JNE  >4E2C       nicht 00, lesen.
       SETO R7          Schreib-Flag.
       LI   R6,>1A      Anzahl der Bytes der Schreibe-Routine(n).
4E2C   LI   R3,>4A20    Adresse des Puffers im 2114.
       MOV  R9,R0       PAD-Basis kopieren.
       MOV  R6,R1       Byteanzahl wird zweimal gebraucht!

4F2A   DATA >484D,>F000,>F700

Diskette formatieren

4F30   CLR  R5
       SOC  @>4BCE,R5          R5= >1000
       MOVB @>4D(R9),R8        Anzahl der Spuren in R8
       MOVB @>4BB4,@>4D(R9)    Löschen
       SZCB @>4F2C,@>4C(R9)    LW-Nr. maskieren.
       BL   @>4E0A             LW-Nr. prüfen und Sektor-I/O Routinen ins PAD.
       MOVB R8,@>4D(R9)        Spuranzahl ins Low-Byte zusammen mit LW-Nr.
       CLR  R7
       MOVB @>4C(R9),R7        LW-Nr. in R7
       SRL  R7,7               mal zwei und ins Low-Byte.
       MOVB @>50(R9),@>52(R9)  Density-Code kopieren.
       CB   @>51(R9),@>4BC8    Seitenzahl mit >02 vergleichen
       JNE  >4F68              einseitig
       SETO @>4A(R9)           FAC als Double Sided Index verwenden
4F68   MOVB @>4BB4,@>50(R9)    ursprünglichen Dichte-Code löschen
       BL   @>4DB0             Drive-Select Leitungen setzen
       BL   @>4D4E             Restore, Kopf über Spur Null.
       CLR  R3                 Spurzähler löschen
4F78   MOV  @>4A(R9),@>4A(R9)  FAC prüfen ob gesetzt.
       JEQ  >4F8A              FAC nicht gesetzt - Single Sided!
       SBO  7                  Side-Select auf 'Opposite'!
```

```
       LI   R10,>0100          Seitenindex für den Spuraufbau
       BL   @>4F4E             Initialisiere eine Spur
4F8A   SBZ  7                  Side-Select 'Upper'
       CLR  R10                Seitenindex auf 'Seite 1'
       BL   @>4F4E             eine Spur initialisieren
       BL   @>4D82             Step in, update track register, load head
       DATA >5800              no verify on destination track.
                               Kommandotyp I, mit stepping rates verODERn.
       BL   @>4B96             auf Status-Bit warten
       AI   R3,>0100           Spurzähler + 1 im High-Byte
       CB   R3,@>4D(R9)        Alle Spuren ?
       JNE  >4F78              Nein - weiter.
       BL   @>4D4E             Restore, Kopf zurück.
       MOVB @>4D(R9),R0        Spuranzahl in R0
       SRL  R0,8               ins Low-Byte
       CB   @>4BC8,@>51(R9)    Seitenzahl mit >02 vergleichen
       JNE  >4FBA              Single Sided!
       SLA  R0,1               Double Sided, hat die doppelte Spuranzahl!
4FBA   CB   @>4BC8,@>52(R9)    Density mit >02 vergleichen
       JNE  >4FCE              Single Density!
       MPY  @>4BD6,R0          Double Density, mit >12 malnehmen!
       MOVB @>4BD7,@>4D(R9)    Anzahl der Sektoren pro Spur
       JMP  >4FD8
4FCE   MPY  @>4BD8,R0          mit >09 malnehmen, Single Density!
       MOVB @>4BD9,@>4D(R9)    Sektoren pro Spur =>09
4FD8   MOV  R1,@>4A(R9)        Gesamtzahl der Sektoren in FAC.
       LI   R6,>1A
       B    @>4EF8             PAD restaurieren und Ende.

Subroutine zum Formatieren einer Spur

4FE4   MOV  R11,R8
       MOV  @>4E(R9),R2        PAB-Buffer Adresse holen
       BL   @>4E7C             Adresse setzen
       CB   @>4BC8,@>52(R9)    Density-Code
       JNE  >4FFA              Single Density!
       B    @>510C             Double Density!

Subroutine zum Formatieren einer Spur Single Density

4FFA   LI   R6,>16           GAP 1, am Anfang der Spur stehen 22 Nullen
       SBO  >A               SD anzeigen
       CLR  R2               Sektor-Zähler
       JMP  >5008            beim 1. Sektor 22 Nullen
5004   LI   R6,6            ADDRESS GAP, alle weiteren Sektoren haben 6 Nullen!
5008   MOVB @>4BB6,@-2(R15)  Nullen schreiben
       DEC  R6               fertig?
       JNE  >5008            nein
       MOVB @>50C9,@-2(R15)  >FE schreiben, AMID
       NOP
       MOVB R3,@-2(R15)      Spur-Nr. schreiben
       NOP
       MOVB R10,@-2(R15)     Seiten-Nr. der Diskette (0 - Lower, 1 - Upper)!
       MOVB R3,R0            Spur-Nr. kopieren
```

```
       SRL  R0,8             ins Low-Byte
       SWPB R10              Seiten-Nr. ins Low-Byte.
       MPY  @>50C5(R10),R0   Seite 0 - Multiplikation mit @>50C4 (=0006)!
                             Seite 1 - Multiplikation mit @>50C6 (=0003)!
       SWPB R10              Seitenindex korrigieren
       A    R2,R1            Sektor-Zähler zu R1 addieren
       DIV  @>4BD8,R0           teilen durch >0009
       MOVB @>50DF(R1),@-2(R15) Sektor-Nr. aus der Interlace-Tabelle.
       LI   R6,-20
5040   MOVB @>50DE(R6),@-2(R15) ID-GAP aufbauen
       INC  R6
       JNE  >5040
       LI   R0,>E5E5           Inhalt des Datenbereichs eines Sektors
       BL   @>50AC
       DATA >0100              Wert in R0 256 mal schreiben.
       MOVB @>50DE,@-2(R15)    >F7, CRC-Auslöser.
       SETO R0
       BL   @>50AC
       DATA >002D              45 mal >FF anhängen
       INC  R2                 nächster Sektor
       CI   R2,9               alle 9 bearbeitet?
       JNE  >5004              nein
       BL   @>50AC
       DATA >00E7              noch 231 mal >FF als GAP 4.
       LI   R4,3               beim Formatieren sind 3 Retries erlaubt.
5074   MOVB @>4E(R9),R2        PAB-Buffer Adresse holen
       BL   @>4E86             Adresse setzen
       BL   @>4D8A             Spur ist bereit, geschrieben zu werden
       DATA >F400              Write track, 30 ms delay, Kommandotyp III.
       MOV  R15,R2             VDPWA kopieren
       AI   R2,>FBFE           VDPRD erzeugen
       LI   R6,>0CA3           Anzahl der Bytes einer Spur in SD!
       SBO  2                  Wait enable
       BL   *R9                Werte in FDC schieben
       BL   @>4B94             Wait disable, warten
       SLA  R0,11              Write protect?
       JNC  >509C              Nein.
       B    @>4D3C             Doch - Abbruch!
509C   SLA  R0,4               Lost data?
       JNC  >50AA              Nein - Schreibversuch war O.K.!
       DEC  R4                 Fehlversuch, Retry möglich?
       JNE  >5074              Ja!
       BL   @>4EB6             Nein - Fehler!
       DATA >3300
50AA   B    *R8                Ende.

Subroutine VDP-Puffer mit gleichen Bytes füllen

50AC   MOV  *R11+,R6           Data holen
50AE   MOVB R0,@-2(R15)        R0 schreiben
       DEC  R6
       JNE  >50AE
       B    *R11              Ende.
```

```
Fehlercodeübergaberoutinen

50B8   BL   @>4EB6
       DATA >0600              Hardware-Error!
50BE   BL   @>4EB6
       DATA >0700              Software-Error!

Data's für den Spuraufbau

50C4   DATA >0006,>0003        seitenabhängige Multiplikatoren.
50C8   BYTE >F0,>FE            AMID

Bytes für ID-GAP

50CA   BYTE >01,>F7,>FF,>FF,>FF,>FF,>FF,>FF
50D2   BYTE >FF,>FF,>FF,>FF,>FF,>00
50D8   BYTE >00,>00,>00,>00,>00,>FB,>F7

Sektor-Interlace Liste Single Density (IL=3)

50DF   BYTE 0,7,5,3,1,8,6,4,2

50E8   DATA 0,0,0,0,0,0         Teil von GAP 2
50F4   BYTE 0
50F5   BYTE >F5,>F5,>F5,>FB     DAM

Sektor-Interlace Liste Double Density (IL=4)

50F9   BYTE 0,11,4,15,8,1,12,5,16,9,2,13,6,17,10,3,14,7

510B   BYTE 0

Subroutine zum Formatieren einer Spur Double Density

510C   CLR  R2                   Sektor-Zähler
       SBZ  >A                   Double Density
       LI   R0,>4E4E             Sync-Muster
       BL   @>50AC               GAP 1
       DATA >0020                32 mal >4E schreiben
511A   LI   R6,-15
511E   MOVB @>50F8(R6),@-2(R15)  ADDRESS-GAP
       INC  R6
       JNE  >511E
       MOVB @>50C9,@-2(R15)      >FE, AMID
       NOP
       MOVB R3,@-2(R15)          Spur-Nr.
       NOP
       MOVB R10,@-2(R15)         Seiten-Index
       MOVB @>50F9(R2),@-2(R15)  Sektor-Nr. aus der Interlace-Tabelle.
       NOP
       MOVB @>4BC6,@-2(R15)      >01, Sektorlänge
       NOP
       MOVB @>4F2E,@-2(R15)      >F7, CRC für AF
       LI   R0,>4E4E             GAP 2
```

```
        BL   @>50AC
        DATA >0016
        LI   R6,-16
515E    MOVB @>50F9(R6),@-2(R15)  GAP 2 und DAM
        INC  R6
        JNE  >515E
        LI   R0,>E5E5             Leerdaten
        BL   @>50AC
        DATA >0100
        MOVB @>4F2E,@-2(R15)      >F7, CRC für Sektor
        LI   R0,>4E4E
        BL   @>50AC               DATA GAP, GAP 3
        DATA >001C                28 Bytes
        INC  R2                   nächster Sektor
        C    R2,@>4BD6            gleich >0012?
        JL   >511A                nein - weiter

GAP 4 schreiben

        BL   @>50AC
        DATA >00BE
        LI   R4,3                 3 Retries
5194    MOVB @>4E(R9),R2          PAB-Buffer
        BL   @>4E86               neu setzen
        BL   @>4D8A               Write track, 30 ms delay.
        DATA >F400                Kommandotyp III
        MOV  R15,R2
        AI   R2,>FBFE
        LI   R6,>190E             Anzahl der Bytes pro Spur bei DD!
        SBO  2                    Wait enable
        BL   *R9                  Datentransfer
        BL   @>4B94               Wait disable, warten
        SLA  R0,11                Write protect?
        JNC  >51BC                Nein.
        B    @>4D3C               Ja - Abbruch!
51BC    SLA  R0,4                 Lost data?
        JNC  >51CA                Nein, OK!
        DEC  R4                   noch Retries?
        JNE  >5194                ja
        BL   @>4EB6               nein!
        DATA >3300
51CA    B    *R8                  Ende.
```

22. ROM-Listing Myarc DDCC-1 ROM listing Myarc DDCC-1

```
*
* Vektortafel
*
4000   DATA 0,0    nicht zugewiesen
*
       DATA >4010  CALL FILES, Direktmodus
       DATA >405A  CALL FILES über SBR >16
       DATA >0000  nicht zugewiesen
       DATA >43E4  Sektor-I/O extern (über DSRLNK)
       DATA >41B0  Formatieren
       DATA >4786  Sektor-I/O intern
*
* CALL FILES im Direktmodus
*
4010   MOV  R11,R10        Rückkehr sicherstellen
       MOV  @>FF4C(R4),R1  >832C liefert VDP-Adresse im Eingabepuffer
       AI   R1,7           plus 7, hinter FILES
       BL   @>40A8     Wort an dieser Adresse nach R0 holen
       CI   R0,>C801   String in Klammern, Länge 1?
       JNE  >404E      Nein, ignorieren
       INCT R1         Ja, nächstes Wort
       BL   @>40A8     lesen
       AI   R0,>CF4A   plus Offset
       CI   R0,>0900   über 9?
       JH   >404E          Ja, nicht erlaubt
       MOVB R0,@>FF6C(R4)  >834C
       BL   @>4062         Pufferzonen anpassen
       MOVB @>FF70(R4),R0  >8350
       JNE  >404E
       A    @>40A6,@>FF4C(R4)      >832C plus 12, Befehlsende
       SZCB @>FF62(R4),@>FF62(R4)  >8342 löschen
*
404E   MOV  R10,R11
4050   LI   R7,>1E03     Op-Code SBZ 3
       LI   R8,>045B     Op-Code RT
       B    R7           ROM ausblenden und Rückkehr
*
* CALL FILES über Level 1 Routine ausführen
*
405A   MOV  R11,R10      Rückkehr sichern
       BL   @>4062       Zuweisung vornehmen
       JMP  >404E        Ende
*
VDP-Pufferzonen anpassen - Diese Routine kümmert sich nicht um andere Systeme,
wel-
che evtl. das VDP-RAM belegen. Damit ist ein Absturz dieser Systeme mit diesem
Con-
troller unvermeidlich, wenn auf diesem Wege mehr VDP-RAM allokiert werden soll!
*
4062   MOVB @>FF6C(R4),R0  >834C, Files-Anzahl
       SRL  R0,8           ins Low-Byte
```

```
       JEQ  >40A2        Null ist unzulässig
       CI   R0,>0009     über Neun?
       JGT  >40A2        auch nicht erlaubt
       MOVB @>508D,R9    momentane Einstellung
       SRL  R9,8           ins Low-Byte
       MOV  @>FF90(R4),R1  Wert aus @>8370 (VDPHI) lesen
407A   C    R0,R9          beide angeglichen?
       JLT  >4088        Nein, alter Wert zu hoch
       JEQ  >4090        beide gleich - Ende
4080   AI   R1,>FDFA     VDPHI reduzieren
       INC  R9           entsprechend mehr Files erlaubt
       JMP  >407A        weiter
4088   AI   R1,>0206     518 Bytes pro File-Daten-Block
       DEC  R9           weniger Files erlaubt
       JMP  >407A        weiter
*
4090   SWPB R9           neue Files-Anzahl
       MOVB R9,@>508D      in Puffer
       MOV  R1,@>FF90(R4)  Neues VDPHI
       CLR  R0             Fehlercode >00
409C   MOVB R0,@>FF70(R4)  in @>8350
       RT                  Ende
*
40A2   SETO R0     Fehlercode >FF
       JMP  >409C  übergeben in @>8350
*
40A6   DATA >000C
*
* Wort aus VDP-RAM lesen
*
40A8   MOVB @3(R4),*R15  Low-Byte R1 setzen
       NOP
       MOVB R1,*R15      High-Byte nachschieben
       NOP
       MOVB @>FBFE(R15),R0   High-Byte lesen
       SWPB R0               umdrehen
       MOVB @>FBFE(R15),R0   Low-Byte lesen
       SWPB R0               Korrektur
       RT                Ende
*
* VDP-Adresse mit Offset setzen
*
40C0   A    *R11+,R1     Offset aus DATA holen
40C2   SWPB R1
       MOVB R1,@>8C02
       SWPB R1
       MOVB R1,@>8C02
       RT
*
40D0   DATA >0009,>0100
40D4   DATA >0005        Retries
40D6   DATA >0400
40D8   DATA >0028        Spuranzahl beim Formatieren
40DA   DATA >467C
```

```
40DC   DATA >FBFE,>D0D4
40E0   DATA >02F7,>0311,>3401
40E6   DATA >12FF,>2000
*
* Adreßfeld lesen, Ermittlung der Formatierung, DRQ wird ignoriert!
*
40EA   DATA >C000        FDC-Kommando READ ADDRESS
40EC   SBZ  3             DD ggf. schalten
       MOVB @>83D4,@>5F01 Befehl an FDC
       LI   R11,>0002     ca. 2 Sekunden
40F8   SETO R10           Preset
40FA   TB   0         Interrupt, Befehl beendet?
       JEQ  >4106     Ja
       DEC  R10       Nein, warten
       JNE  >40FA
       DEC  R11
       JNE  >40F8
4106   SBO  3         SD und ROM an
       RTWP           Ende
*
* Warteschleife auf Befehlsabarbeitung
*
410A   LI   R9,>0003     Standard-Preset
410E   SETO R10          Schleife
4110   TB   0            Interrupt?
       JEQ  >416A        Ja, Befehl beendet
       DEC  R10
       JNE  >4110
       DEC  R9
       JNE  >410E
411C   LI   R6,>0600     Fehlercode >06
       BL   @>46C4         PAD-RAM restaurieren
       C    @>83F6,@>40DA  GPLWS R11 = >467C?
       JNE  >4132
       CI   R2,>0003     Laufwerk 4?
       JNE  >4136        Nein
4132   SBZ  1
       JMP  >413A
4136   BL   @>46C4       PAD-RAM restaurieren
413A   CLR  R9           Kein Fehler für PAB
       MOVB R10,@>8350   auch keiner sonstwie
       JEQ  >4150        Ja
       LI   R9,>C000     >03 in den oberen Bits, Illegale Operation
       CB   @>40E4,R10   >34, Write Protect?
       JNE  >4150        Nein
       LI   R9,>2000       Ja, >01 in den oberen 3 Bits, Schreibschutz
4150   CB   @>506A,@>40E3  >F7? (Interner Op-Code)
       JNE  >415C          Nein, Workspace nicht verändern
       LWPI >83E0        Ja, GPLWS laden
415C   MOV  @>5074,R11   Rückkehradresse holen
       CI   R11,>4832    Interner Sektor-I/O?
       JEQ  >416A        Ja, innerhalb des oberen ROMs zurückkehren
       B    @>4050       Ende
416A   RT
```

```
*
* VDP-Füllroutine
*
416C   MOV  *R11+,R1     Anzahl
416E   MOVB R0,@>8C00    Füllmuster
       DEC  R1           alle?
       JNE  >416E        Nein
       RT                Ende
*
* Sektor (mit DAM) aufbauen
*
4178   MOVB @>40DC,@>8C00   >FB, DAM
       LI   R0,>E5E5        Leerdaten
       LI   R1,>0100     256 Bytes
       JMP  >416E        füllen
*
* Adreßfeld aufbauen
*
4188   MOVB @>40DD,@>8C00   >FE, IDAM
       MOV  @>5046,R5
       MOVB @>508D(R5),@>8C00  Spurnummer
       MOVB R8,@>8C00          Seitenindex
       MOVB @>500F,@>8C00      Sektornummer, R7 LBy
       MOVB @>40E5,@>8C00    >01, Länge
       MOVB @>40E1,@>8C00    >F7, CRC-Auslöser
       RT
*
* Einsprung Formatieren
*
41B0   MOV  R11,@>5074
       LWPI >5000        Workspace im System-RAM
       BL   @>4716       PAD-RAM vorbereiten
       LI   R10,>0700
       CB   @>834D,@>40D9   Spuranzahl = >28 (.40)?
       JNE  >4136           ungleich - Fehler (sehr unflexibel!!)
       BL   @>4698       Laufwerk zuschalten
       CLR  R4                 Zähler für Gesamtsektorsumme
41CE   MOVB @>40D7,@>508E(R2)  >00 in Spurnummernpuffer
       MOVB @>836C,R9          Step-Code?
       JNE  >41E2           Langsam
       MOVB @>40D4,@>5F01  >00, Restore schnell
       JMP  >41E8
41E2   MOVB @>40E0,@>5F01  >02, Restore langsam
*
* Innere Schleife mit Spurinkrement
*
41E8   BL   @>410A       Ende Restore abwarten
       CLR  R8           Seitenindex
41EE   CLR  R7           Sektornummer
       MOV  @>5034,R1    VDP-Adresse
       BL   @>40C0       VDP-Adresse zum
       DATA >4000          Schreiben setzen
       CB   @>5036,@>40E0  >02, Double Density?
       JEQ  >424E          Ja
```

```
*
* Spur SD im Puffer aufbauen
*
       CLR  R0           >00
       BL   @>416C       füllen
       DATA >0016        22 Bytes
420A   BL   @>4188       Adreßfeld aufbauen
       SETO R0           >FF
       BL   @>416C       ID-GAP, GAP 2
       DATA >000B        11 Bytes
       CLR  R0           >00
       BL   @>416C       ID-GAP cont.
       DATA >0006        6 Bytes
       BL   @>4178         Sektor aufbauen
       MOVB @>40E1,@>8C00  >F7, CRC-Auslöser
       SETO R0
       BL   @>416C       DATA-GAP, GAP 3
       DATA >002D        45 Bytes >FF
       CLR  R0           >00
       BL   @>416C       ADDRESS-GAP aufbauen
       DATA >0006        6 Bytes
       AI   R7,>0007     Sektornummer weiter
       CLR  R6           Rechenregister
       DIV  @>40D0,R6    durch 9 teilen
       MOV  R7,R7        Rest?
       JNE  >420A        ja, weiter
       SETO R0           >FF, GAP 4
       AI   R4,>0009     Sektorzähler weiter
       JMP  >42A6        Spur ist fertig zum Schreiben
*
* Spur DD im Puffer aufbauen
*
424E   LI   R0,>4E4E     GAP 1
       BL   @>416C       aufbauen
       DATA >0032        50 Bytes >4E
       CLR  R0           >00
       BL   @>416C       ADDRESS GAP
       DATA >000C        12 Bytes
       LI   R0,>F5F5     AMID-Kennung
       BL   @>416C       aufbauen
       DATA >0003        3 Bytes
       BL   @>4188       AF mit AMID aufbauen
       LI   R0,>4E4E
       BL   @>416C       ID-GAP, GAP 2
       DATA >0016        22 Bytes
       CLR  R0           plus Lücke
       BL   @>416C       >00
       DATA >000C        12 Bytes
       LI   R0,>F5F5     DAM vorbereiten
       BL   @>416C
       DATA >0003        3 mal wiederholen
       BL   @>4178         Sektor aufbauen
       MOVB @>40E1,@>8C00  >F7, CRC-Auslöser
       AI   R7,>0007       Interlace über modulo-Funktion
```

```
       ANDI R7,>000F     Low-Nib stehen lassen
       JNE  >424E        weiter wenn kein Overflow
       LI   R0,>4E4E     GAP 4 Daten
       AI   R4,>0010     Sektorzähler weiter
*
* Pufferinhalt schreiben, Spur formatieren
*
42A6   BL   @>416C       GAP 4 aufbauen
       DATA >02BC          Anzahl Bytes
       MOVB @>508E(R2),R11 Spurnummer holen
       JNE  >42BC          nicht Null
       MOVB @>40DF,@>5F01  >D4, Force Interrupt, unmittelbarer Interrupt
       BL   @>410A         warten
42BC   LI   R9,>4616     Adresse VDP-Transferroutine
       BL   @>4770       verschieben
       LI   R10,>83C4    letztes Wort der FDC-Schreibroutine
       LI   R9,>43D4     Füllroutine
42CC   MOV  *R9+,*R10+   anhängen
       CI   R9,>43E4     alle?
       JNE  >42CC        Nein
       INC  @>83A8       Befehl in Schreibroutine von Sektor- in Spur schreiben
       SBZ  2            Unterseite (Seite 0)
       MOV  R8,R8        welche Seite wird formatiert?
       JEQ  >42E0        unten
       SBO  2            Nein, oben. Anschalten
42E0   MOV  @>5034,R1      VDP-Adresse
       BL   @>40C2         setzen
       CB   @>40E0,@>5036  >02, Double Density?
       JNE  >42FE          Nein
       LI   R10,>17E6    Ja, Spurlänge DD effektiv (GAP 4 durch Füllroutine!)
       MOV  R10,@>83F4   R10 GPLWS
       BLWP @>460E       Double Density Schreibroutine
       JMP  >430A        weiter
42FE   LI   R10,>0BE3    Spurlänge SD ohne GAP 4 (s.o.)
       MOV  R10,@>83F4   R10 GPLWS
       BLWP @>4612       SD-Routineneinsprung
430A   MOVB @>5F01,R10   Statusregister lesen
       ANDI R10,>4000    Write-Protect isolieren
       JEQ  >431A        nicht aktiv
       SRL  R10,4        doch, in >0400 für Fehlercode wandeln
       B    @>4136       Abbruch
*
431A   CB   @>5037,@>40E0  >02, Double Sided?
       JNE  >4372          Nein
       MOV  R8,R8        Ja, schon Seite 2?
       JNE  >432E        Ja, dann weiter
       LI   R8,>0101     Nein, Seitenindex
       B    @>41EE       weitermachen
*
* Auf korrekte Formatierung Seite B prüfen
*
432E   MOV  @>5046,R5      Laufwerknummer (???)
       MOVB @>508E(R2),R0  Spur Null auf diesem LW?
       JNE  >4372          Nein, dann keinen neuen Test machen
```

```
       CB   @>40E0,@>5036  >02, Double Density?
       JEQ  >4346          Ja
       BLWP >477E        Adreßfeld lesen SD
       JMP  >434A
4346   BLWP @>4782       Adreßfeld lesen DD
434A   MOV  @>83F4,R11   R10 GPLWS
       JNE  >4354        R10 ist bis zur GAP 4 Sequenz gekommen!
       B    @>411C       Nein, I/O-Fehler >06 melden
4354   MOVB @>5F01,R5    Statusregister lesen
       SLA  R5,4         Record not found?
       JNC  >4372          Nein, OK
       MOVB @>40E5,@>5037  >01, nur einseitiges Formatieren möglich
       AI   R4,-9          Gesamtsektorzahl korrigieren
       CB   @>40E0,@>5036  Double Density?
       JNE  >4372          Nein
       AI   R4,-7        Ja, nochmal 7 ab gibt 16 weniger
*
4372   CLR  R6             Rechenregister
       MOVB @>508E(R2),R6  Spurprotokoll
       AI   R6,>0100       plus 1
       CB   R6,@>5033    Spuranzahl erreicht?
       JNE  >43B4        Nein
       BL   @>46C4       PAD-RAM restaurieren
       MOV  R4,@>834A    Gesamtsektoranzahl übergeben
       LI   R8,>0900       9 SPS SD
       CB   @>40E0,@>8350  >02, Double Density?
       JNE  >439A          Nein
4396   LI   R8,>1000       Ja, 16 SPS (gleichzeitig SEEK-Code!)
439A   MOVB R8,@>834D    SPS übergeben
       SLA  R2,2           Laufwerk mal 4
       MOV  R4,@>5058(R2)  Gesamtsektorzahl aufheben
       MOV  @>8350,R4      Dichte und (ggf. korrigierte) Seiten
       SWPB R4             umdrehen
       MOV  R4,@>505A(R2)  umgedreht ablegen
       CLR  R10            kein Fehler
       B    @>413A       Ende
*
* Nächste Spur anfahren mit SEEK ohne VOT (Verify-On-Track)
*
43B4   MOVB R6,@>508E(R2)  R6 in Spurprotokoll (wurde bereits inkrementiert)
       MOVB R6,@>5F07      und ins Datenregister
       MOVB @>836C,R6    Step-Code
       JNE  >43CA          langsam
       MOVB @>4398,@>5F01  >10, Seek schnell
       JMP  >43D0
43CA   MOVB @>40E6,@>5F01  >12, Seek langsam
43D0   B    @>41E8         weiter mit warten auf Ausführung
*
* Füllroutine für's Formatieren, bis Interrupt kommt
*
43D4   MOVB @>8800,R10   letzten Wert aus VDP
43D8   MOVB R10,@>5F07   ins Datenregister
       TB   0            INTREQ?
       JNE  >43D8    nein, weiter, gnadenlos
```

```
       SBO  3        SD und ROM an
       RTWP          Schluss
*
* Einsprung Sektor-I/O
*
43E4   MOV  R11,@>5074  Globaler R11-Puffer
       BL   @>4716      PAD-RAM vorbereiten
       MOV  R2,R6       LW-Nummer
       SLA  R6,2        mal 4
       AI   R6,>505A    Zeiger auf Seiten/Dichte-Protokoll
       MOV  *R6,R4      Eintrag lesen
       MOVB @>40D7,R4   >00, Seite egal, Interrupt-Flag dafür einbauen
       MOV  R3,R3       Sektor Null?
       JNE  >442C       Nein
       MOV  R0,R0       Lesen von Disk?
       JEQ  >442C       Nein, Schreiben auf Disk
*
* Retry-Einsprung,Sektor lesen oder Laufwerk-Erstzugriff
*
4402   CLR  R6       Spurnummer = 0
       CLR  R7       spätere Sektornummer
       CLR  R5       wird nicht benutzt!
       SBZ  2            Seite 0 (unten)
       MOVB R12,R4       Interrupt-Flag aktiv
       BL   @>4698       Laufwerk zuschalten
       MOVB @>836C,R9      Step-Flag
       JNE  >441E          Langsam
       MOVB @>40D0,@>5F01  >00, RESTORE, Step 00
       JMP  >4424
441E   MOVB @>40E0,@>5F01  >02, RESTORE, Step 10
4424   MOV  R2,R9          Laufwerknummer kopieren
       AI   R9,>508E     Offset der Spurnummerntafel addieren
       JMP  >449A        weiter mit Übergabe der Nummer
442C   MOV  R2,R6        Laufwerknummer kopieren
       SLA  R6,2         mal 4
       AI   R6,>5058     plus Tafel 'Sektorobergrenze'
       MOV  *R6+,R8      maximale Sektoranzahl+1 holen
       JNE  >4440          nicht null - OK
       MOVB @>40E7,@>836A  >FF, Laufwerk und Parameter nicht initialisiert
       JMP  >4402          wieder von vorne
4440   LI   R10,>0700    Index 'Fehler >07'
       C    R3,R8        geforderter Sektor nicht zu gross?
       JL   >444C        Ja, kleiner - OK
       B    @>4136       Sektornummer zu hoch, I/O-Error 7
444C   BL   @>4698       Laufwerk zuschalten
       LI   R8,>0010       16 SPS
       CB   @>5009,@>40E0  >02, DD?
       JEQ  >4460          Ja
       LI   R8,>0009       Nein, SD, 9 SPS
*
452A   MOV  R0,R0        Schreiben auf Disk?
       JEQ  >4532        Ja
       AI   R1,>C000       Nein, Pufferzeiger korrigieren (für Verify)
4532   MOVB @>500F,@>5F05  Sektornummer ins Sektorregister
```

```
       MOVB R6,@>5F03      Spurnummer ins Spurregister
       MOVB R8,@>5F07      erster Wert im Datenregister
       CB   @>5009,@>40E0  >02, DD?
       JEQ  >454E          Ja
       BLWP @>4612       SD-I/O-Routine
       JMP  >4552
454E   BLWP @>460E       DD-I/O-Routine
4552   MOVB @>5F01,R10   Statusregister lesen
       COC  @>40D6,R10   >0400, LOST DATA?
       JEQ  >44DE        Ja, wiederholen
       ANDI R10,>1800    RNF oder CRC?
       JEQ  >45C6        Nein, keiner davon
*
* Fehler bei Sektor-I/O
*
4562   MOVB @>836A,R8    Fehler - Erstzugriff?
       JNE  >4584          Ja, bereits indiziert
       MOVB @>40E7,@>836A  Nein, >FF - jetzt nachholen
       MOVB @>836D,R9      Retries
       AI   R9,>FF00     minus 1 im High-Byte
       MOVB R9,@>836D    zurück
       JEQ  >45AC        Null - Ende
       MOV  R0,R0        I/O-Typ
       JEQ  >45A0        schreiben
       MOV  R3,R3        Sektor Null?
       JNE  >45A0          Nein
4584   CB   @>5009,@>40E0  >02, DD?
       JEQ  >45BE          Ja, jetzt mit SD probieren
*
* Dichte auf DD umschalten, Retries reduzieren
*
       MOVB @>836D,R9    Retry-Zähler holen
       AI   R9,>FF00     1 abziehen
       MOVB R9,@>836D    Ende bei Null
       JEQ  >45AC
       MOVB @>40E0,@>5009  >02, ab jetzt mit DD probieren
45A0   B    @>4402         Retry-Einsprung
*
45A4   SRL  R10,4         Fehlercode
       AI   R10,>3000     plus Kennung 'Schreiben'
       JMP  >45BA         übergeben
45AC   SRL  R10,4
       MOVB R10,R10       Kein Fehler?
       JNE  >45B6         doch
       AI   R10,>0200     hier muß schon was fehlerhaft sein!
45B6   AI   R10,>2000     Code plus Kennung 'Lesen'
45BA   B    @>4136       Übergeben und Schluss
*
* Dichte auf SD wechseln, Retries  -1 nicht -0 reduzieren
*
45BE   MOVB @>40D7,@>5009  >00, auf Single Density umschalten
       JMP  >45A0          weiter mit Retry
*
* Sektor-I/O war fehlerfrei, Fortsetzung
```

```
*
45C6   MOV  R0,R0        I/O-Typ
       JNE  >45FA        Lesen, fertig
       MOVB @>5F01,R10   Statusregister lesen
       ANDI R10,>4000    Write Protect testen
       JNE  >45A4        Gesetzt - Schreibschutz!
       SETO R0           Nein, umschalten auf Lesen
       SETO @>5048       Index RAM-Transfer (für Dummy-Lesen in ROM!)
       CLR  R1             Pufferadresse ist >0000 (Konsolen-ROM)!
       MOVB @>40E7,@>836B  >FF, Verify-Index
       B    @>444C         weiter mit 'Laufwerk anschalten'
*
45E6   MOVB @>836B,R8  Verify-Durchlauf?
       JNE  >45F4      Ja, fertig
       MOV  R3,R3      Sektor 0?
       JNE  >45F4      Nein
       BL   @>46F0     Laufwerkdefaults bei jedem Zugriff auf Sektor 0 setzen!
*
45F4   CLR  R10        Kein Fehler
       B    @>4136     Fehlerbehandlung für Rückkehr mit benutzen
*
45FA   MOVB @>836A,R8  Parameter neu zu setzen?
       JEQ  >45E6      Nein, es ging mit den alten!
       BL   @>46F0         Laufwerkparameter setzen (Seiten, Dichte, MaxSec)
       MOVB @>40D7,@>836A  >00, keine neue Anpassung
       B    @>442C         nochmal von vorne anfangen
*
460E   DATA >83E0,>83A4  Context-Switch Double Density I/O
4612   DATA >83E0,>83A6  Context-Switch Single Density I/O
*
* Datentransferroutinen, liegen im PAD-RAM ab >83A2
*
4616   DATA >A0F0     FDC-Kommandos Write Sector, Write Track
       SBZ  3              DD und ROM aus (ggf. überspringen)
       MOVB @>83A2,@>5F01  korrekten Code in FDC
4620   TB   1              DRQ?
       JEQ  >462E     Ja
       TB   0         Interrupt?
       JEQ  >4638     Ja
       JMP  >4620     Nein, weiter auf DRQ warten
462A   TB   1         DRQ?
       JNE  >462A          Nein
462E   MOVB @>8800,@>5F07  Ja, bedienen
       DEC  R10            alle Bytes geschrieben?
       JNE  >462A     Nein, nächsten DRQ abwarten
4638   JMP  >4652     Hier wird beim Formatieren die Füllroutine angehängt!
*
463A   DATA >8000     FDC-Kommando Read Sector
*
463C   SBZ  3              ggf. DD schalten
       MOVB @>83A2,@>5F01  FDC-Code übergeben
4644   TB   0              INTREQ
       JEQ  >4654     Ja, vielleicht Schreibschutz
4648   TB   1         DRQ?
```

```
       JNE  >4644          Nein, Interrupt testen
       MOVB @>5F07,@>8C00  DRQ bedienen
4652   JMP  >4648          kein Zähler, Interrupt macht das
4654   SBO  3         SD schalten und ROM an
       RTWP           Ende
*
4658   DATA >A000     FDC-Kommando Write Sector
*
465A   SBZ  3              ggf. DD schalten
       MOVB @>83A2,@>5F01  FDC-Code übergeben
4662   TB   1              DRQ?
       JEQ  >4670     Ja, bedienen
4666   TB   0         INTREQ?
       JEQ  >4678     Ja, Ende
       JMP  >4662     weiter pollen
466C   TB   1         DRQ?
       JNE  >4662         Nein
4670   MOVB *R11+,@>5F07  Daten aus CPU-RAM auf Disk
       DEC  R10           alle?
       JNE  >466C     Nein
4678   JMP  >4694     Schluß
*
467A   DATA >1000
467C   DATA >8000     FDC-Kommando Read Sector
*
467E   SBZ  3              DD und ROM aus
       MOVB @>83A2,@>5F01  FDC-Code
4686   TB   0              INTREQ?
       JEQ  >4694          Ja, Ende
468A   TB   1             DRQ?
       JNE  >4686          Nein
       MOVB @>5F07,*R11+   Ja, DRQ bedienen
       JMP  >468A          und weiter
4694   SBO  3         Ende, ROM an
       RTWP           Schluß
*
* Laufwerk zuschalten
*
4698   SBZ  4        LW 1 aus
       SBZ  5        LW 2 aus
       SBZ  6        LW 3 aus
       SBZ  7        LW 4 aus
       MOV  R12,R10  Basis kopieren
       AI   R12,8    auf Bit 4
       A    R2,R12   Laufwerknummer dazu
       A    R2,R12   nochmal
       SBO  0        dieses Laufwerk anschalten
       MOV  R10,R12  alte CRU-Basis
       AI   R12,14   auf Bit 7
       S    R2,R12   minus Laufwerk
       S    R2,R12   nochmal
       TB   0        Laufwerk mit kurzer Step-Zeit?
       JEQ  >46BE       Ja
       MOVB R12,@>836C  Nein, vermerken
```

```
46BE   MOV  R10,R12     Basis restaurieren
       SBO  1         Motoren an
       RT             Ende
*
* PAD-RAM restaurieren
*
46C4   MOV  R11,R9        R10 darf nicht verändert werden
       MOV  @>5038,R1     Stackpointer
       INC  R1            plus 1 auf Datenzone
       BL   @>40C2        Leseadresse setzen
       LI   R6,>83A2      Zieladresse der gesicherten Daten
46D4   MOVB @>8800,*R6+   aus VDP verschieben
       CI   R6,>8400      Ende erreicht?
       JNE  >46D4         noch nicht
*
       LI   R8,>5030      Pufferzone im System-RAM
       LI   R6,>834A      Zieladresse
46E6   MOV  *R8+,*R6+     verschieben
       CI   R6,>8352      alle?
       JNE  >46E6         Nein
       B    *R9           Ja, Ende
*
* Laufwerk Standard-Werte setzen
*
46F0   LI   R10,>470E     Zeiger DS/DD
       MOV  R2,R9         LW-Nr kopieren
       SLA  R9,2          mal 4
       AI   R9,>5058        plus Offset Parametertafel
       CB   @>40E0,@>5009   DD?
       JEQ  >4708           Ja
       LI   R10,>4712     Nein, SD-Daten
4708   MOV  *R10+,*R9+    Defaults in Protokollzone
       MOV  *R10,*R9
       RT                 Ende
*
470E   DATA >0500,>0202   Defaults DS/DD
4712   DATA >02D0,>0201   Defaults SS/DD
*
* PAD-RAM sichern und Routinen vorbereiten
*
4716   MOV  @>5046,R2     Laufwerknummer
       DEC  R2            korrigieren als Zeiger
       CLR  @>836A          Index 'Laufwerkparameter gültig' & 'Kein Verify'
       MOV  @>40D4,@>836C   Step-Code und Retries
       LI   R6,>834A        Adressfeldroutine kommt hier hin
       LI   R8,>5030      Pufferzone
472E   MOV  *R6+,*R8+     verschieben
       CI   R6,>8352      alles?
       JNE  >472E         noch nicht
       LI   R6,>3EF4      Start Stackbereich
       SB   @>508D,R6     Files-Anzahl als Vielfaches von 256 davon abziehen
       MOV  R6,*R8        nach >5038 als Stackpointer
       MOV  R11,R9        Rückkehr gewährleisten
       MOV  R1,R8         Pufferadresse nicht verändern
```

```
       MOV  R6,R1        VDP-Stackpointer
       INC  R1           Korrektur
       BL   @>40C0       als Schreibadresse sichern
       DATA >4000        Offset bewirkt Indizierung
       LI   R6,>83A2     Hier kommen später die Transferroutinen hin
4752   MOVB *R6+,@>8C00  PAD ins VDP-RAM schaffen
       CI   R6,>8400     Ende erreicht?
       JNE  >4752        Nein
       LI   R1,>40EA     Adressfeldroutine
       LI   R6,>83D4     kommt hier hin
4764   MOV  *R1+,*R6+    verschieben
       CI   R6,>83F4     30 Bytes
       JNE  >4764        noch nicht
       MOV  R8,R1        Pufferzeiger wieder an alten Platz
       B    *R9          Ende
*
* FDC-Routinen kopieren, Routinentyp (R9) extern definiert
*
4770   LI   R10,>83A2    Hier fangen sie alle an
4774   MOV  *R9+,*R10+   verschieben
       CI   R10,>83C6    und hier hören sie alle auf
       JNE  >4774        wenn sie komplett sind
       RT                Schluß
*
477E   DATA >83E0,>83D8  Single Density AF-Routine
4782   DATA >83E0,>83D6  Double Density AF-Routine
*
* Interner Sektor-I/O
*
4786   MOV  @>506A,R0    I/O-Code
       CB   R0,>40D1     >09?
       JEQ  >4838        Ja
       MOV  R11,R9
       CI   R2,>57FE
       JEQ  >487A
       MOV  @4(R3),R5
       SRL  R0,2
       JOC  >47A4
       SRL  R0,1
       JOC  >47EE
47A4   CLR  @4(R3)
       MOV  R5,R6
       SLA  R6,1
       JNC  >47CA
       ANDI R5,>7FFF
       MOV  R5,@>5006    Sektornummer in SYSRAMWS R3
       JEQ  >47CA          Sektor null!
       CLR  @>5048         VDP-Transfer
       MOV  @>5056,@>5002  Pufferadresse in späteres R1
       BL   @>4828         Sektor schreiben
       BL   @>481E         Fehlercode aus GPLWS R10 oder SYSRAMWS R9 holen
47CA   MOV  @2(R3),@>5006  späteres R3, Sektornummer
       MOV  R3,R5          Datenpufferadresse kopiern
       AI   R5,>0012     plus Offset
```

```
       MOV  R5,@>5002    Pufferadresse, Zieladresse der Daten
       SETO @>5048       RAM-Transfer
       BL   @>4828         Sektor-I/O, interner Einsprung
       CB   @>40E8,@>5012  R9 im SYSRAMWS gleich >20?
       JEQ  >47EE          Ja
       BL   @>481E       Fehlercode ermitteln
47EE   MOV  @>506A,R2
       CB   @>40D6,R2
       JEQ  >4806
       CLR  *R3
       MOVB R10,@>8350   Fehlercode übergeben
       AB   R10,@>506B
4802   MOV  R9,R11
       JMP  >487A
*
4806   SLA  R2,12
       JNC  >4814
       CLR  @6(R3)
       MOVB @>40D7,*R3
       JMP  >4802
4814   SLA  R2,4
       JOC  >4802
       CLR  @>5070
       JMP  >4802
*
481E   MOVB R10,R10      R10 = >00?
       JNE  >4826        Nein
       MOVB @>5012,R10   Ja, R9 holen
4826   RT
*
* Subroutine 'Sektor schreiben'
*
4828   LWPI >5000        SYSRAMWS
       CLR  R0           I/O-Modus Schreiben
       BL   @>43E4       normale Routine verwenden
4832   LWPI >83E0        Das auf diese Adresse zeigende R11 indiziert den
       RT                internen Aufruf!!
*
4838   CLR  R1
       MOVB @>1E(R3),R1
       SRC  R1,13
       AB   @3(R4),R1
       MOV  @>24(R3),R0
       SWPB R0
       COC  @>40D6,R1
       JNE  >485E
       C    @6(R3),R0
       JNE  >487E
       CB   *R3,@>22(R3)
       JNE  >487E
       JMP  >4874
485E   MOV  @2(R3),R9
       JNE  >486E
       C    @>5070,@>40D2
```

```
       JL   >487E
       JMP  >4874
486E   C    @>5070,R0
       JL   >487E
4874   AI   R1,>0100
4878   MOV  R10,R11
487A   B    @>4050       Ende
*
487E   MOVB R1,@>5072
       JMP  >487A        Ende
*
4884   TEXT '(C)MYARC1984'
*
4890   Ab hier wiederholt um 1 erhöhte Bytes
       z.B. >90,>91,>92,>93,>94...
*
* ...
*
5000   Start des System-RAMs. Volle 2KByte 8-Bit RAM
*
* ...
*
5030 - 5037      Kopie des PAD-Bereiches >834A - >8351
5038             VDP-'Stackpointer'
*
* ...
*
5046             Laufwerknummer
504A - 5053      Filenamenpuffer
*
* ...
*
5058   Sektormaximum Laufwerk 1
505A   Seiten- und Dichtecode Laufwerk 1
505C   Sec-Max LW 2
505E   SD-Code LW 2
5060   Sec-Max LW 3
5062   SD-Code LW 3
5064   Sec-Max LW 4
5066   SD-Code LW 4
5068   R11-Puffer
*
* 506A bis 508C  PAB-Kopie
*
  506A - 5072    Die ersten 9 Bytes des PAB
  5073           Namenlänge (Gerät + Datei)
*
5074   Wird auch als R11-Puffer benutzt
*
  5074 - 508C    Geräte und Dateinamenpuffer
*
508D   Files-Anzahl (1-9)
508E   Aktuelle Spur Laufwerk 1
508F   Aktuelle Spur Laufwerk 2
```

```
5090   Aktuelle Spur Laufwerk 3
5091   Aktuelle Spur Laufwerk 4
*
5092 - 5191      Puffer für Sektorinhalt
*
* ...
*
53BE - 53C7      Puffer für Disknamen
53C8 - 54C7      Puffer für FDS des Files
54C8 - 57FF      >00
5800 - 5EFF      >FF - Nicht belegt
5F00 - 5FFF      FDC-Bereich, alle ungeraden Adressen belegt, alle 8 Byte
*                wiederholt
```

22.0.1. Zugriffsadressen des FDC im Myarc-Diskcontroller

Access addresses of the FDC in the Myarc Disk Controller

Register	*Funktion*			*Register*	*Function*	
Adresse	*Lesen*	*Schreiben*		*Address*	*Read*	*Write*
5F01	Statusregister	Kommando-register		5F01	Status register	Command register
5F03	Sektorregister	Sektorregister		5F03	Sector register	Sector register
5F05	Spurregister	Spurregister		5F05	Track register	Track register
5F07	Datenregister	Datenregister		5F07	Data register	Data register

5F09 ff Wie an >5F01, siehe Hinweis

5F09 FF like on >5F01 here, see note

23. Ramdisks

Als kleine Zugabe, quasi in letzter Minute, finden sich hier einige Anmerkungen zum Thema Ramdisk.

Mit dem "Einbruch" der Horizon Ramdisk, in den ersten Exemplaren noch mit einer eher vorsintflutlichen DSR ausgestattet, die von sämtlichen Standards unbeeinflußt war, ist dieses Speichermedium vor allem wegen der enormen Geschwindigkeit des Datentransfers heute ein essentieller Faktor bei der 'vernünftigen' Arbeit am TI.

Die Vorgänger waren jedoch schon länger am Markt. Das begann mit der Foundation 128K-Karte, die jedoch nur eine Art Bank-Switching aufwies, mit dem man sich mehr schlecht als recht den Speicher für eigene Assembler-programme erweitern konnte.

Sie diente sowohl der CorComp-Ramdisk als auch der von Mechatronic als Vorbild, die als erste ein umgeschaltetes 32K-RAM zur Simulation einer Floppy-Disk aufwiesen.

Das zentrale Problem des CorComp-Geräts war das aller CorComp-Karten:

eine CorComp-Erweiterung im System war genug, mehrere konnten einige PE-Boxen aufgrund der unzulässigen Busbelastung zu sonderbarem Verhalten veranlassen. Über die DSR sei nur soviel gesagt, daß sie ähnliche Holzhammermethoden wie die von Myarc benutzte (Autor war Galen Reed) und somit im Zusammenspiel mit bestimmten ROS-Versionen anderer Karten regelmäßig für Probleme sorgte.

Die Mechatronic-DSR war mehr ein Notlaufsystem, über das hier nicht weiter ausgeführt werden soll. Der Autor ist (zu seinem Glück) unbekannt.

Ramdisks

As small addition, quasi in last minute, are some notes about ramdisks.

With the "failure" the Horizon Ramdisk, into which first copies still with a rather antiquatedDSR, which was uninfluenced by all standards, this storage medium is equipped particularly because of the enormous rate of the data transfers today an essential factor with the "reasonable" work on the TI.

The predecessors were however already longer at the market. That began itself with the Foundation 128K card, which indicated however only a type bank switching, with which one more badly than quite the memory for own assembler programs extending could.

It served both the CorComp Ramdisk and from Mechatronic and model, a switched 32K-RAM indicated which as first for the simulation of a floppy diskette.

The central problem of the CorComp device was that of all CorComp cards:

A CorComp extension in the system was enough, several could some PE boxes due to the illegal bus load to strange behavior arrange. Over the DSR it is only as much said that it used similar sledgehammer methods as by Myarc (author was Galen Reed) and thus in interaction with certain ROS versions of other cards regularly for problems ensured.

The Mechatronic DSR was more a fail-safe system, over which was not to be executed further here. That author is unknown (to his luck).

23.1. Hardware-Konzepte

Hier existieren im wesentlichen 2 Varianten: CPU-RAM Bank-Switching und DSR-RAM Bank-Switching. Die Horizon-Karte wendet das zweite Verfahren an, alle anderen Karten das erste. Aus verschiedenen Gründen ist das Horizon-Konzept als einziges geeignet, die Daten per Batterie puffern zu lassen. Der CorComp-Versuch mit dem externen Netzteil kann nur als untauglich bezeichnet werden.

Alle Ramdisks simulieren softwareseitig ein Diskettenlaufwerk. Dazu dient ein Betriebssystem mit Namen ROS (Ramdisk Operating-System).

23.1.1. CPU-RAM Bank-Switching

Die schaltungstechnisch einfachste Lösung ist die, den ganzen 32K-Speicher der Speichererweiterung mehrfach vorzusehen und aus einer DSR, die ja im Bereich >4000 nicht umgeschaltet wird, wenn sie erst einmal gewählt wurde, den passenden Bereich der Ramdisk anzuschalten.

In dem Moment stehen in der 32K-Erweiterung nicht mehr die Anwenderprogramme sondern die Sektorinhalte eines bestimmten Bereiches der Ramdisk. Diese werden dann ins VDP-RAM verschoben bzw. mit den dort abgelegten Daten neu gefüllt.

Ein RAM-RAM-Transfer, wie ihn neue ROS-Versionen der Horizon bieten, ist mit diesem Konzept nicht realisierbar. Ferner ist es nicht möglich, bei Anwendung einer 16-bit-32K-Speichererweiterung 16-bit-32K eine Ramdisks dieser Art zu nutzen, da es unmöglich ist, die Standard 16-bit-32K-Schaltungen zu deaktivieren und somit die Bankumschaltung möglich wird. Die Hardware-Details sollen hier nicht näher erläutert werden.

Hardware concepts

Here essentially 2 versions exist: CPU RAM bank switching and DSR RAM bank switching. The Horizon card applies the second procedure, all other cards first. For different reasons the Horizon concept is suitable as only to let the data buffer by battery. The CorComp attempt with the external power pack can be called only unfit.

All ramdisks simulate in terms of software a floppy disk drive. In addition an operating system with names serves ROS (Ramdisk Operating System).

CPU RAM bank switching

The technical circuiting simplest solution is those to designate the whole 32K memory of the memory expansion several times and to turn from a DSR, which is not switched within the area >4000, if it were only selected, the suitable area on of the ramdisk.

In the moment sector contents of a certain area of the ramdisk are into the 32K expansion no longer the user programs separate. These are then shifted in VDP RAM or with the data stored there again filled.

A RAM-RAM transfer, how new ROS versions of the Horizon offer it, is not realizable with this concept. Furthermore it is not possible to use with application of a 16-bit-32K for ramdisks of this type since it is impossible, the standard 16-bit-32K circuits to deactivate and thus bank switching becomes possible.The hardware details are not to be described here more near.

23.1.2. DSR-RAM Bank-Switching

Die verträglichste Lösung für die Implementation eines externen Speichers führt über eine Pipe (einen Zugriffskanal) im DSR-Bereich.

Da dieser nur bei Erkennen der korrekten Gerätenamen aktiviert wird, ist es prinzipiell möglich, soviele Ramdisks in der Box zu haben, wie es freie CRU-Adressen gibt, ohne daß sich diese, von den Laufwerknummern einmal abgesehen, ins Gehege kommen.

23.2. Software-Konzepte

Ramdisks verfügen i.d.R. über jede Menge Speicher; die Horizon sogar über ein DSR-RAM, was neben dem leichten ROS-Austausch auch noch einen eleganten Workspace liefert.

Damit besteht keine Notwendigkeit, das VDP-RAM mitzubenutzen, abgesehen von den normalen PABs und deren Datenpuffern. Im entsprechenden Absatz dieses Kapitels wird kurz auf die Probleme der parallelen Nutzung des VDP-RAMs eingegangen.

23.2.1. Laufwerkumleitung

Viele Programme für den TI scheinen aus einer Zeit zu stammen, als Diskettenlaufwerke mit Gold aufgewogen wurden. Dementsprechend erlauben sie nur Laufwerksnummern von 1 bis 3 oder 4 (das sind schon die etwas progressiveren …).

Wie weiter unten erläutert wird, ist es daher nötig, mittels einer entsprechenden Schaltung die Ramdisk beim Absuchen der CRU-Seiten vor den Diskcontroller zu setzen, de facto also auf CRU-Basis >1000. Hier kann die Ramdisk dann die Laufwerke 1 bis 4 simulieren, wobei allerdings der Zugriff auf den Diskcontroller nur noch mit Umwegen möglich ist.

Hier wurden von TI selbst aber auch von Drittanbietern von Hard- und Software etliche

DSR RAM bank switching

The most compatible solution for the implementation of an external memory leads across a pipe (an access channel) within the DSR area.

Since this is activated only with detecting the correct device names, it is possible always to have as many ramdisks in the box as there are free CRU addresses, without these, from the drive numbers once apart, in the enclosure to come oneself.

Software concepts

Ramdisks have usually each quantity memory; the Horizon even over a DSR RAM, which supplies elegant Workspace apart from the easy ROS exchange also still another.

No necessity exists to share VDP RAM apart from the normal PABs and their scratchpad memories. In the appropriate paragraph of this chapter with the problems of the parallel use of the VDP RAM one deals briefly.

Drive bypass

Many programs for the TI seem to come from a time, when floppy disk drives were also counterbalanced gold. Accordingly they permit only drive numbers from 1 to 3 or 4 (that already are the something more progressive …).

As is further down described, it is necessary to set by means of an appropriate circuit the ramdisk when searching the CRU pages before the disk controllers in fact thus on CRU base >1000. Here the ramdisk can simulate then the drives 1 to 4, whereby however that is possible for access on the disk controllers with detours only.

Here by TI in addition, by third providers of hard and software some errors were committed.

Fehler begangen.

So prüft jede normale DSR z.B. bei einem Interrupt oder etwa die TI RS232 beim normalen DSRLNK ab, ob sie denn gemeint ist. Wenn nicht, dann wird das Flag für die Fortsetzung der Suche gesetzt und die DSR verlassen.

23.2.2. Nicht so beim Diskcontroller

Bei ihm wird die im PAB verfügbare Laufwerksnummer auf die controllerinterne Maximalzahl verglichen und gleich ein Fehler gemeldet. Alle phantasielosen 'Abschreiber' der TI-Disk-DSR haben diesen Fehler übernommen. Das Ergebnis ist, daß Ramdisks über der CRU-Adresse >1100 meist nicht oder unzureichend funktionieren, wenn sie auf Sektorebene angesprochen werden, egal ob sie nun Laufwerksnummern über 4 bzw. 3 aufweisen oder nicht.

23.2.3. Partitionierung

Mit der Verfügbarkeit hoher Speicherkapazitäten zu relativ niedrigen Preisen stieg die Ramdisk-Kapazität stufenweise an, bis theoretische Speichergrößen jenseits des Megabytes realisierbar wurden. Damit stellte sich das Problem, wie man diese Speichermenge mit dem TI-DOS sauber verwalten kann.

Da wie o.ä. die Grenze der Diskettenkapazität 400 KByte beträgt, und man im Sinne der Softwarekompatibilität mit den leider recht vielen unsauber programmierten Tools und Anwenderprogrammen tunlichst nicht über die 360K hinausgehen sollte, muß der Speicherraum einer derart großen Ramdisk in mehrere kleine Laufwerke mit der angeführten Maximalkapazität aufgeteilt werden.

Diesen Vorgang, bei dem bestimmte, zusammenhängende Speicherbereiche einer Ramdisk jeweils einem anderen virtuellen Laufwerk zugeordnet werden, nennt man Partitionierung.

Thus each normal DSR checks the TI RS232 e.g. with an interrupt or for instance with the normal DSRLNK whether it is meant. If not, then the flag for the continuation of the search set and left the DSR.

Not so with the disk controller

With it in the PAB available drive number is compared on the controller-internal maximum number and announced equal an error. All fantasyless "copy outs" the TI disk DSR took over this error. The result is that ramdisks over the CRU address do not function to >1100 insufficiently usually or, if they are responded to sector level all the same whether it now drive numbers over 4 or 3 indicates or not.

Partitoning

With the availability of high storage capacities at relatively low prices the ramdisk capacity rose gradually, until theoretical memory capacities became realizable beyond the Megabyte. Thus the problem placed itself, how one can cleanly administer this memory quantity with the TI DOS.

Since or the like. the boundary of the diskette capacity 400 KByte amounts to, and one in the sense of the software compatibility with that unfortunately quite many carelessly programmed tools and user programs tunlichst not beyond the 360K to go, must the memory space of such a large ramdisk should be divided into several small drives with the aforementioned maximum capacity.

Connected storage areas of a ramdisk another virtual drive assembly to be assigned in each case, one calls this procedure, with which determined, Partitioning.

23.2.4. Namenszugriff

Das TI-DOS erlaubt den Zugriff auf Daten durch Kombination des Dateinamens mit dem Namen der Diskette, auf der sich die betreffenden Dateien befinden. Das DOS sucht dann zuerst die Diskette anhand des Namens und dann die Datei. Intern wird natürlich der bei der Suche ermittelte korrekte Laufwerksbezeichner benutzt, doch merkt das der Anwender meist nicht.

Ramdisk-Betriebssysteme sind nun so aufgebaut, daß sie beim Zugriff über den Disknamen die Anfrage dann weitergeben, wenn sie den Namen nicht in der Ramdisk-Partitionstafel gefunden haben. Alle anderen Diskcontroller laufen bis Laufwerk 3 bzw. 4 und meckern dann (s.o.).

Mit diesem Verfahren kann trotz Umleitung einer Diskettenstation z.B. auf DSK1 für das Originale E/A-Modul, noch auf eine Diskette im Laufwerk 1 zugegriffen werden, wenn nur die Ramdisk 1 anders benannt wird.

Name access

The TI DOS permits the access to data by combination of the file name with the name to the diskette, on which the files concerned are. The DOS looks then first the diskette up on the basis the name and then the file. Naturally with the search determined correct drive designators, but notices that is used internally the user usually not.

Ramdisk operating systems are now in such a way structured that they pass the inquiry on with the access over the disk name if they did not find the name in the ramdisk partition board. All other disk controllers run to drive 3 or 4 and grumble then (see above).

With this procedure can be accessed despite bypass of a floppy station e.g. DSK1 for originals the I/O module, still a diskette in the drive 1, if only the ramdisk 1 is differently designated.

23.3. Konflikte bei der Parallelnutzung des VDP-RAMs

Das VDP-Konzept des VDP-Area-Links soll vermeiden, daß derartiges passiert. Leider ist es nicht zum Allgemeingut geworden, wie dieser VDP-Area-Link dazu benutzt werden kann, jeder DSR ihr eigenes 'Reich' im VDP-RAM zuzuweisen.

Manche Ramdisk DSRs 'pfriemeln' recht mutig im Bereich des Diskcontrollers herum, andere lassen 'die Finger weg'.

Beides ist nicht ganz so, wie es sein könnte.

Im Interesse einer konsistenten und vor allem kompatiblen Verwaltung aus diversen Programmiersprachen, sollte ein offenes File immer den reservierten Bereich der File-Puffer belegen. Da neben dem Namen auch die Laufwerksnummer zum File gehört, die auch

Conflicts during the parallel use of VDP RAM

The VDP concept VDP area links of it is to avoid that such occurs. Unfortunately it did not become the common property/knowledge, as this can be used VDP area link to assign to each DSR their own "realm" in the VDP RAM.

Some ramdisk DSRs use an "awl" quite courageously within the area of the disk controller around, others leave "the fingers away."

Both are not completely like that, as it could be.

In the interest of a consistent and above all compatible administration from various programming languages, an open file should always occupy the reserved area of the file buffers. Since apart from the name also the drive number belongs to the file, which is also part of

Teil des Namensvergleichs ist, gibt es nur dann Probleme, wenn Ramdisk und Diskcontroller auf dem selben physischen Laufwerk eine Datei gleichen Namens offen halten. Die gängigen DSRs für Diskcontroller melden 'File already open', wenn so etwas passiert und greifen munter darauf zu. Um also das VDP-RAM 'mehrbenutzerfähig' zu machen, müssen diese Fehler von der Programmseite restriktiver behandelt werden.

Weiterhin müßte jede DSR so geändert werden, daß sie die Daten eines angeforderten File-Abschnitts immer liest, auch wenn ihr die entsprechenden Pointer sagen, daß der Abschnitt schon im Datenpuffer steht.

Zudem muß eine sog. Updated-Data-Buffer-Area, besonders die von Sektor 0, nach Ende jedes File-I/Os rückgeschrieben werden.

Unter dieser Bedingung können Ramdisk- und Floppy-Disk-DSR das VDP-RAM gemeinsam nutzen.

23.4. Zugriffsgeschwindigkeiten

Eine Ramdisk ist schnell — das schnellste derzeit denkbare Speichermedium, nicht nur für den TI. Der Grund liegt auf der Hand: zum einen ist der TI-99/4A konzeptionell nicht DMA-fähig, zum anderen gehen sämtliche anderen externen Medien wie Floppy- und Hard- bzw. Optical-Disks den Umweg über einen Cache-Speicher, der doch irgendwann mal ins CPU-RAM übertragen werden muß, was eine Ramdisk ja ausschließlich tut.

Wesentlich schlimmer, wenn es darum geht, diese potentielle Geschwindigkeit zu nutzen, macht sich ein langsames File-System bemerkbar.

Beim TI ist dies die satzorientierte Datei. Hier überwiegt der Verwaltungsaufwand für die Pointer, Buffer, Offsets etc., die das flexible Verwalten von Files ermöglichen, so daß eine Ramdisk im Mittel bei Lese- und

the name comparison, there are problems only if ramdisk and disk controllers keep a file of same name open on the same physical drive. The usual DSRs for disk controllers announces to "already open" file, if such a thing occurred and access lively. In order to make thus VDP RAM "multi-userable," these errors must be more restrictively treated by the program page.

Further each DSR would have to be modified in such a way that it always reads the data of a requested file paragraph, even if you say the appropriate pointers that the paragraph is already located in the scratchpad memory.

Besides one must update the so-called Data Buffer Area, especially the one or sector 0, after end of each file I/Os to be particularly rewritten.

Under this condition a ramdisk and floppy disk DSR can use VDP RAM together.

Access rates

A ramdisk is fast — the fastest at present conceivable storage medium, and not only for the TI. That reason is obvious: on the one hand the TI-99/4A is conceptually not capable of DMA, on the other hand goes all other external media such as floppy and hard or optical disks the detour over a cache, which will nevertheless sometime transfer times into CPU RAM must, which exclusively does a ramdisk.

Substantially more badly, if it concerns to use this potential rate a slow file system becomes apparent.

With the TI this is the record-oriented file. Here outweighs the administration effort for the pointers, buffer, offset etc, which enable the flexible administration of files, so that a ramdisk is faster on the average with read and write

Schreibzugriffen (gemischt) nur etwa 30% schneller als eine DSDD-Diskette ist. Mitverantwortlich für das Tempo ist auch der dauernde Umweg über das VDP-RAM, von dem ja nicht nur der PAB sondern auch der Datenpuffer betroffen ist.

Neuere ROS-Versionen erlauben durch Erweiterung des I/O-Codes den direkten RAM-RAM-Transfer, was signifikante Vorteile bringen kann.

Programme jedoch, die üblicherweise nicht zugängliche Daten im VDP-RAM suchen, laufen dann nicht mehr, da der Zeitgewinn bei einem Duplizieren ins VDP-RAM verloren ginge.

accesses (mixed) only about 30% than a DSDD diskette. Jointly responsible for the speed is also the continuing detour over VDP RAM, by which not only the PAB but also the scratchpad memory are affected.

Newer ROS versions permit the direct RAM-RAM transfer, which can bring significant advantages by extension of the I/O Code.

Programs however, which do not look usually accessible data up in the VDP RAM, do not run then any longer, since the time gained would be lost with duplicating into VDP RAM.

24. Harddisks

An Speicherkapazität sind Harddisks den Ramdisks natürlich überlegen, nicht jedoch in Bezug auf die Zugriffsgeschwindigkeit.

Da Harddisks mechanische Systeme sind (zumindest zur Zeit), vergeht eine u.U. signifikante Zeitspanne, bis die Daten verfügbar sind - von Disk-Duplexing, Mirroring und Elevator-Seeking oder SCSI, ESDI und DCPs spricht man beim TI natürlich nicht.

Zudem sind sämtliche I/O-Kanäle des TI an die CPU gebunden, so daß diese nicht entlastet wird (mal abgesehen von den seriellen I/O-Chips) und es nur auf eher niedrige Datenraten bringt.

Hier soll nicht über Sinn oder Unsinn von Festplatten an einem 64K-Rechner wie dem TI oder dem 9640 geredet werden, vielmehr soll gezeigt werden, worin der prinzipielle Unterschied zwischen Disketten (Floppy- bzw. Flexible-Disks) und Harddisks (Fixed Disks) besteht.

24.1. Mechanischer Aufbau

Festplatten bestehen, wie der Name schon sagt, aus fest montierten Platten, die sandwichartig übereinander angeordnet sind.

Die Datenträger bestehen aus einer anti-magnetischen Aluminiumlegierung und sind beidseitig mit einer magnetisierbaren Schicht ähnlich den Floppies beschichtet (natürlich viel feiner strukturiert).

Die höhere Speicherkapazität wird durch mehrere Faktoren erreicht:

- Eine höhere Rotationsgeschwindigkeit der Platten (~3000rpm),
- Geringer Abstände des über die Platten fliegenden Kopfes,
- Mehrere Platten,

Hard disks

For storage capacity are hard disks to ramdisks naturally superior, not however regarding the access rate.

Since hard disks are mechanical systems (at least at present), pass a possibly significant time interval, until the data are available - of disk Duplexing, Mirroring and elevator Seeking or SCSI, ESDI and DCPs one does not speak naturally with the TI

Besides all I/O channels of the TI are bound to the CPU, so that this is not relieved (times apart from the serial I/O chips) and it only on rather low data rates brings.

Here is not being talked about sense or nonsense about fixed disks at a 64K computer like the TI or that 9640, on the contrary it is to be demonstrated where the difference in principle between diskettes (floppy or flexible disks) and hard disks (fixed disks) exists.

Mechanical structure

Fixed disks exist, like the name already says, from firmly installed disks, which are like a sandwich one above the other arranged.

The data carrier consists of a non-magnetic aluminum alloy and is reciprocally coated with a magnetizable layer similar to a floppy (structures naturally much finer).

The higher storage capacity is achieved by several factors:

- A higher rotation rate of the disks (~3000 rpm),
- Small distance between the platter and the flying heads,
- More platters,

- Schmalere Spuren und extrem kleine Köpfe, Kopfspalte,
- Optimierte Codierungstechniken.

Die höhere Rotationsgeschwindigkeit bewirkt eine Frequenzgangverbesserung, was die Flußwechselrate nach oben bringt. Zudem erhöht sich so die Datenübertragungsrate. Hierzu eine kleine Rechnung:

Eine Floppy-Disk, in MFM beschrieben, weist 18 Sektoren in einer Spur auf, von denen jeder 256 Byte enthält. Die Drehzahl beträgt 300 rpm, also 5 Umdrehungen pro Sekunde.

Dadurch werden 4,5 Kilobyte in 0,2 Sekunden, also maximal 22,5 Kilobyte pro Sekunden gelesen bzw. geschrieben, wenn kein Verify stattfindet — mit Verify halbiert sich die Rate beim Schreiben.

Eine MFM-Harddisk verfügt über 17 Sektoren à 512 Byte pro Sektor und rotiert mit 3000 rpm, also 50 Umdrehungen pro Sekunde. Pro Sekunde werden also 25 mal 17 mal 1 Kilobyte übertragen, was eine Rate von 425 Kilobyte pro Sekunde ergibt.

Beide Rechnungen legen einen 1:1 Interleave zugrunde.

Die Kopfsteuerung der Harddisks wurde in den vergangenen Jahren mehrfach überarbeitet. Es finden sich Konzepte wie bei Floppies (Spannbandstepper) oder Voice-Coils und Linearmotoren.

Aufgrund der geringeren Toleranzen ist es jedoch nicht so einfach, eine Spur einfach so wiederzufinden, zudem man immer auch die Wärmeausdehnung aller Komponenten beachten muß.

Hierzu verfügt jede Harddisk normaler Technologie über eine komplett separierte Plattenseite, auf der ausschließlich Spurinformationen für die Positioniereinheit (Servo) aufgezeichnet sind. Neben dem normalen

- Narrower tracks and extremely small heads, head gaps
- Optimized coding techniques.

The higher rotation rate causes a frequency response improvement, which brings the flux change rate upward. Besides so the data transmission rate increases. For this a small calculation:

An MFM floppy diskette has 18 sectors per track, with each sector containing 256 bytes. The diskette rotates at 300 rpm, or 5 revolutions per second.

Thus 4.5 kilobytes can be read in 0.2 seconds, with a maximum write speed of 22.5 kilobytes per second, if no verify takes place. With verify the writing rate is halved.

An MFM hard disk has 512 bytes per sector, 17 sectors and rotates at 3000 rpm, thus 50 revolutions per second. Therefore 25 times per second 17 times 1 kilobyte will be transferred, which results in a rate of 425 kilobytes per second.

Both calculations are based on a 1:1 interleave.

The head control of hard disks has been revised several times in recent years. These embrace concepts used on floppies (stretch band stepper) or voice coils and linear motors.

Due to the smaller tolerances it is however not so simple to regain a track simply in such a way in addition one must consider always also the thermal expansion of all components.

For this each hard disk of normal technology has a completely separated side of the plate, on which excluding track information for the positioning unit (servo) are recorded. Apart from normal stepping a readjusting (fine adjustment)

Steppen kann hier anhand eines Referenzpegels eine Nachjustierung (Feinjustage) erfolgen, falls einige Parameter aufgrund Alterung oder Wärmeeinfluß verschoben sind.

24.2. Ansteuerung

Zusätzlich zu den für Floppies üblichen Signalen können Harddisks einfache Befehle ausführen wie z.B. das Recalibrate anhand der o.ä. Servo-Spuren. Dazu verfügt der, inzwischen veraltete, Shugart-Bus, der sich bei Harddisks als Seagate-Norm ST506 präsentiert, über zusätzliche Leitungen. Zudem hat jede Harddisk ihr eigenes Datenkabel.

Bei der Softwareansteuerung muß beachtet werden, daß neben Spur und Seitennummer nun auch eine Plattennummer benötigt wird.

Praktisch wird mit Zylinderbereichen, Zylindernummern und Kopfnummern sowie Sektornummern gearbeitet.

Zudem ist sich jede Festplatte 'bewußt', daß ihre Fehlerrate aufgrund der höheren Speicherdichte in Bereiche kommen kann, wo es kritisch wird. Daher wird als Analogon zum CRC-Byte bei Floppies ein sog. ECC-Byte (Error-Correction-Code) hinzugefügt, über den die Festplatte selbst fehlerkorrigierend (bis 5 Bit) tätig werden kann. Ist jedoch die ECC-Correction-Span überschritten, tritt der Fehler endgültig auf.

Ob dies ein Fortschritt zum gnadenlosen CRC-Fehler bei Floppies ist, kann nicht so einfach bewertet werden, doch ist mit ECC-Test immerhin eine Früherkennung möglich, wenn man dem Harddiskcontroller das ECC-Byte verbietet.

24.3. Handling

Aufgrund der sehr engen Toleranzen bei Kopf, Platten und Servo reagieren Festplatten sehr empfindlich auf Stöße, Staub und Temperaturschwankungen. In keinem Fall sollte man das Gehäuse einer Festplatte öffnen, da

can take place here on the basis a reference level, if some parameters are shifted due to aging or heat influence

Control

Additionally to the signals usual for floppies, hard disks can execute simple instruction e.g. the recalibrate on the basis or the like servo tracks. In addition that has, in the meantime became outdated, Shugart bus, which presents itself with hard disks as Seagate standard ST506, additional lines. Besides each hard disk has its own data cable.

During the software control must be noted that apart from track and side number now also a disk number is needed.

Practically with cylinder areas, cylinder numbers and head numbers as well as sector numbers one operates.

Besides itself each fixed disk is "conscious" that their error rate can come due to higher memory density into areas, where it becomes critical. Therefore becomes as analogue the CRC byte with Floppies so-called ECC-Byte (error correction code) added, over which the fixed disk can become error-correcting (until 5 bits) active. However if the ECC correction limit is exceeded, the error occurs finally.

Whether this is a progress to the merciless CRC error with floppies, can be not so simply evaluated, but is with ECC test nevertheless an early recognition possible, if one forbids the ECC byte to the hard disk controller.

Handling

Due to the very close tolerance with heading, disks and servo react to fixed disks very sensitively to impacts, dust and variations in temperature. In no case one should open the housing of a fixed disk, since then dust

dann Staub eindringt, der unweigerlich beim nächsten Start Kopf und Platten irreparabel beschädigt. Es soll aber Leute geben, die ihre vermeintlich defekte Festplatte öffneten, ein wenig beim Steppen zuguckten, sie wieder zuschraubten und anschließend steif und fest behaupteten, sie ginge nun gar nicht mehr, weil sie ja vorher schon nicht funktionierte …

Im stehenden Zustand verkraften Harddisks schon mal einen leichten Stoß — im Betrieb hat dies i.d.R. einen Head-Crash zufolge, nach dem man die Kiste am besten wegwirft. Daher gilt: im Betrieb befindliche Geräte, die eine Festplatte beinhalten, in keinem Fall bewegen!

penetrates, which damages heading and disks inevitably with the next start irreparably. There is to be however people, which opened their allegedly defective fixed disk, a little when stepping "looked on", it again screwed on and afterwards rigidly and firmly maintained, she would not go now any longer, because she already did not function beforehand …

In the standing status hard disks bear already times a light impact — in the operation this has usually according to Head Crash, after which one throws the crate away best. Therefore applies: move devices present, which contain a fixed disk, in no case in the operation!

Statement of file origin

This file was created for users of PC99, a TI-99/4A emulator running on an IBM PC.

> Reproduced with permission of the authors Christopher Winter and Harald Glaab. This file may only be distributed with The Cyc CD in Adobe Acrobat pdf format.

While every effort was made to ensure that the text and graphics content of this file are an accurate copy of the original Disk Info publication, CaDD Electronics can assume no responsibility for any errors introduced during scanning, editing, or conversion.

If you find an error, we will attempt to correct it and provide you with an updated file. You can contact us at:

CaDD Electronics
45 Centerville Drive
Salem, NH 03079-2674

Version 20051021