

S P E E D Y 1 0 5 0

E i n A n w e n d e r H a n d b u c h
u n d R O M L i s t i n g

(c)1986 Compy-Shop
Version 1.0 vom 16.10.86

Autoren:
Peter Bee, Erwin Reuß und Reinhard Wilde

Copyright Notiz:

ATARI, ATARI 1050, ATARI 800XL, ATARI 130XE und sind eingetragene Warenzeichen der Firma ATARI CORP. DEUTSCHLAND.

MAC/65, ACTION!, BASIC XL, BASIC XE und DOS XL sind eingetragene Warenzeichen der Firma OPTIMIZED SYSTEM SOFTWARE INC., SAN JOSE, CA, USA.

"Die Informationen im vorliegendem Handbuch werden ohne Rücksicht auf einen eventuellen Patentschutz veröffentlicht. Warennamen werden ohne Gewährleistung der freien Verwendbarkeit benutzt. Bei der Zusammenstellung von Texten und Abbildungen wurde mit größter Sorgfalt vorgegangen. Trotzdem können Fehler nicht vollständig ausgeschlossen werden. Die Autoren und Herausgeber dieses Handbuches können für fehlerhafte Angaben und deren Folgen weder eine juristische Verantwortung noch irgendeine Haftung übernehmen. Für Verbesserungsvorschläge und Hinweise auf Fehler sind die Autoren dankbar.

Alle Rechte vorbehalten, auch die der fotomechanischen Wiedergabe und der Speicherung in elektronischen Medien sowie der Übersetzung in fremde Sprachen. Die gewerbliche Nutzung der in diesem Handbuch gezeigten Modelle, Programme und Arbeiten ist nur mit der ausdrücklichen Genehmigung der Autoren erlaubt.

SPEEDY 1050 ANWENDER HANDBUCH
(c) 1986 Compy-Shop Ohg, 4330 Mülheim Ruhr
Alle Rechte vorbehalten

Druck: DER DRUCKER, BOCHUM
Printed in Germany

INHALTSVERZEICHNIS:

Vorwort	1
Der Aufbau der SPEEDY 1050 Platine	3
Die Grundversion	3
Die Erweiterte Version	3
Die Funktionsweise der SPEEDY 1050	4
Die Datenübertragung zum Computer	6
Programmieren der SPEEDY 1050	7
Speicheraufteilung SPEEDY 1050	8
Erklärung zur Speicherbelegung	9
Die Einsprungsadressen	11

LABEL	SEITE
RESET	12
RESET2	12
BEREIT	13
MOTON	13
TSTMON	14
MOTOFF	14
SDELAY	14
SDRDDP	15
XWAIT	15
X2WAIT	15
TRACKO	15
TRADJA	16
TRADJ	16
TRVR	16
CONRES	17
CONRE2	17
WREADY	17
RD128	17
RD256	17
RDBTS	18
RDSFOL	18
RDTRA	19
RDTRAV	19
TSTWR	20
TSTDAT	21
SD128B	21
SD256B	21
SDBTS	21

LABEL	SEITE
SEND41	22
SEND43	22
SEND45	22
SEND4E	22
RDSECT	23
RDSEC1	23
WRSECT	24
WRSEC1	24
TSTWRP	24
VERSEC	25
VERSE1	25
STELL	25
QUITT	26
RDHEAD	26
RDHD1	26
CALCTS	27
SETBUF	27
SETBUF2	27
SEXBUF	28
SETRWL	28
COPSLT	28
BELL1	29
CLRDSP	29
TRAANZ	29
DEZOUT	30
HEXOUT	30
DENDSP	30
SETTIM	31
CLRTRA	31
CLRDSK	31
RAMTST	32
ROMTST	33
SPEEDT	33

DIE BEFEHLE DER SPEEDY 1050

34

BEFEHL	SEITE
\$52	35
\$50	35
\$57	35
\$53	36
\$21	37
\$22	37
\$20	38
\$3F	38
\$4E	39
\$4F	39
\$51	40
\$44	40
\$4B	41
\$4C	41
\$4D	42
\$41	42
\$68	43
\$69	43
\$60	44
\$62	44

ANHANG A - DEMONSTRATIONS-PROGRAMME

Demonstration 1 für Kommando \$52
Demonstration 2 für Kommando \$52
Demonstration 1 für Kommando \$50
Demonstration 2 für Kommando \$50
Demonstration für Kommando \$3F
Demonstration für Kommando \$44
Demonstration für Kommando \$4B
Demonstration für Kommando \$4C
Demonstration für Kommando \$4D
Demonstration 1 für Kommando \$41
Demonstration 2 für Kommando \$41
Demonstration für Kommando \$60
Demonstration für Kommando \$62

Lesen der SIO Routine vom Laufwerk

ANHANG B - ERWEITERUNG DER SPEEDY N AUF SPEEDY T
EINBAUANLEITUNG
BESTÜCKUNGSPLAN

ANHANG C - DAS ROM-LISTING DER SPEEDY

VORWORT

Wenn Sie dieses Handbuch gekauft haben, sind Sie wahrscheinlich schon Besitzer einer SPEEDY 1050, und kennen die Vorteile, die Ihnen dieser Hardwarezusatz für Ihr Laufwerk bietet, und wollen jetzt etwas mehr über die Möglichkeit der Programmierung der SPEEDY 1050 erfahren.

Genau das wollen wir Ihnen mit diesem ANWENDER HANDBUCH ermöglichen: Die Programmierung Ihrer SPEEDY 1050.

Und genau hierfür haben wir dieses Handbuch geschrieben. Sie sollen die Möglichkeit bekommen, die speziellen Fähigkeiten der SPEEDY 1050 für Ihre eigenen Programme zu nutzen. So können Sie zum Beispiel sehr leicht die SUPER SPEED Routine der SPEEDY 1050 in Ihre Programme einbauen, oder die Fähigkeit der SPEEDY, eine Diskette zu formatieren, ohne daß Sie dazu den Computer oder ein DOS benutzen müssen. Oder Sie können sich selber ein Kopierprogramm schreiben, mit dem Sie Ihre Originaldisketten kopieren können.

Sie werden dazu in diesem ANWENDER HANDBUCH die genaue Dokumentation des ROM-Listings, sowie eine ausführliche Dokumentation der Einsprungsadressen und natürlich Demoprogramme finden. Wir glauben, das wir dieses ANWENDER HANDBUCH leicht verständlich geschrieben haben.

Wir wenden uns mit diesem ANWENDER HANDBUCH an die Programmierer, die wissen, wie Sie einen Floppy-Controller vom Typ 2797 oder 2793 programmieren müssen. Es würde zu weit führen, die Programmierung dieses Floppy-Controllers in diesem Handbuch zu erklären. Wir möchten in diesem Zusammenhang auch auf die ausgezeichnete Dokumentation des Herstellers dieses Bausteins hinweisen, die Sie in jedem guten Zubehörhandel bekommen können.

Hier also noch einmal: Dieses ANWENDER HANDBUCH ist nicht gedacht für den Anfänger in der Maschinensprach Programmierung! Es ist speziell geschrieben worden für den fortgeschrittenen Programmierer.

Wichtig für einen Programmierer ist auch noch der folgende Hinweis: Sie brauchen an uns keine weitere Lizenzgebühr zu zahlen, wenn Sie Software für die SPEEDY 1050 schreiben, oder Routinen aus der SPEEDY 1050 innerhalb Ihrer Software für die SPEEDY 1050 benutzen. Sie können also Ihre Programme Kommerziell vermarkten. Wir möchten Sie nur bitten, uns eine Kopie Ihres Programmes zu kommen zu lassen.

Und nun viel Spaß bei der Programmierung Ihrer SPEEDY 1050.

Ihr COMPY-SHOP TEAM

DER AUFBAU DER SPEEDY 1050 PLATINE

Es gibt zwei verschiedene Versionen der SPEEDY 1050. Die Erste ist die Grundaussführung. Die zweite Version der SPEEDY 1050 ist eine erweiterte Ausführung mit Trackanzeige und einem akustischen Fehlermelder, dem Summer. Technisch, und auf die Laufwerksleistungen bezogen, sind beide Versionen identisch.

DER AUFBAU DER GRUNDVERSION

Die Grundversion besteht aus der Platine, einem 8K RAM-IC, einem 8K EProm mit dem Betriebssystem, dem Mikroprozessor 65C02 und diversen Kodier-IC's.

DER AUFBAU DER ERWEITERTEN VERSION

Zusätzlich zu den Bauteilen der Grundversion kommen bei der erweiterten Version noch die Bauteile für den Summer und die Trackanzeige hinzu.

Die Grundversion lässt sich leicht durch einen entsprechenden Bausatz mit Trackanzeige und Summer nachrüsten.

DIE FUNKTIONSWEISE DER SPEEDY 1050

Ein normales ATARI 1050 Laufwerk besitzt einen RAM Buffer von 256 Bytes Größe. Diesen RAM-Bereich müssen sich Datenspeicher und Mikroprozessor teilen. Da für einen Sektor ja bereits 128 Bytes gebraucht werden, hat der Mikroprozessor nicht mehr viel Platz zum arbeiten. Aus diesem Grund kann bei einem ATARI 1050 Laufwerk pro Umdrehung der Diskette jeweils nur ein Sektor eingelesen und zum Computer gesendet werden. In einem Track liegen, bei SINGLE DENSITY, 18 Sektoren. Die Diskette rotiert mit ca. 5 Umdrehungen pro Sekunde (288 Umdrehungen/Minute).

Das ergibt, für einen kompletten Track, eine Ladezeit von ca. 3,6 Sekunden. Das ist die Zeit, die das ATARI 1050 Laufwerk benötigt, um einen kompletten Track einzulesen.

Die SPEEDY 1050 besitzt einen 8K-Byte großen RAM Speicher. Dieser arbeitet als DATENSPOLER. Pro Umdrehung der Diskette kann nun ein kompletter Track in den RAM-Buffer eingelesen werden. Das ergibt eine Ladezeit für einen Track von 0,2 Sekunden. Die Geschwindigkeit ist also um den Faktor 18 erhöht worden. Diese Geschwindigkeit ist die normale Arbeitsgeschwindigkeit eines SPEEDY 1050 Laufwerkes.

Durch die Zwischenspeicherung der Daten kann nun auch die Übertragungsgeschwindigkeit zum Computer erhöht werden. Durch diesen grossen RAM Buffer können nun auch mehr als 128 Bytes in einem Sektor geschrieben werden. Es wird also echte DOUBLE DENSITY, 256 Bytes per Sektor, möglich. Auch bei dieser Speicherdichte, von nun 176K Byte auf einer Diskettenseite, arbeitet das SPEEDY 1050 Laufwerk mit der hohen Geschwindigkeit.

Auf der SPEEDY 1050 Platine befindet sich neben dem RAM aber auch noch ein neuer Mikroprozessor. Dieser Mikroprozessor ist der 65C02. Gegenüber dem 6507 einer normalen ATARI 1050 bietet der 65C02 zwei grosse Vorteile.

Erstens kann der 65C02 bis zu 64K adressieren, der 6507 nur 8K-Bytes, und zweitens besitzt der 65C02 einen erweiterten Befehlssatz mit zusätzlichen, sehr nützlichen Befehlen. Aufgrund dieser zusätzlichen Befehle konnte das Betriebssystem der SPEEDY 1050 kurz gehalten werden, und die Geschwindigkeit der Programmausführung wird durch die geschickte Ausnutzung des erweiterten Befehlssatzes gesteigert.

DIE DATENÜBERTRAGUNG ZUM COMPUTER

Bei einem normalen ATARI 1050 Laufwerk wird, wie wir schon erwähnt haben, pro Diskettenumdrehung ein einzelner Sektor eingelesen und dann sofort an den Computer weitergegeben.

Diese Methode ist sehr Zeitraubend. Bei einem SPEEDY 1050 Laufwerk wird, bei einer Diskettenumdrehung, ein kompletter Track in den RAM Buffer eingelesen. Dadurch hat der Computer jederzeit Zugriff auf alle Sektoren, die sich in diesem Track befinden. Durch diese Zwischenspeicherung der Daten ist, im Normalmodus, eine Übertragung der Daten ohne Pause (bedingt durch die Ladezeit zwischen den Sektoren) möglich.

Das bedeutet, daß im Normalmodus die Lesegeschwindigkeit im Laufwerk mit maximaler Geschwindigkeit läuft, die Datenübertragung zum Computer aber mit normaler Geschwindigkeit geschieht. Aber aufgrund der wegfallenden Pausen verkürzt sich die Ladezeit um ca. 50%.

Beim aktivieren der speziellen SPEEDY 1050 Geschwindigkeit, der SUPER SPEED, wird die Datenübertragung vom Laufwerk zum Computer auf das Maximum gesetzt. Bei einem normalen ATARI 1050 Laufwerk geschieht die Datenübertragung zum Laufwerk mit 19.200 Baud. Beim aktivieren der SUPER SPEED erhöht sich diese Baudrate auf das 4 fache.

PROGRAMMIEREN DER SPEEDY 1050

Aufgrund der besonderen Fähigkeiten der SPEEDY 1050 haben Sie nun auch die Möglichkeit das Laufwerk individuell zu programmieren.

Die Möglichkeiten, die sich Ihnen damit eröffnen, sind fast unerschöpflich. So können Sie zum Beispiel Diskettenformate nach Ihrem eigenen Bedarf erstellen, einen eigenen Kopierschutz erzeugen oder einen fremden kopieren.

Auch die SUPER SPEED können Sie sehr leicht für Ihre eigenen Programme nutzen, wie Sie anhand eines Demoprogrammes sehen werden.

SPEICHERAUFTeilUNG SPEEDY 1050

Nachfolgend finden Sie ein Blockschaltbild mit der genauen Speicherbelegung der SPEEDY 1050.

\$FFFF	ROM - 8 K B Y T E
\$E000	
\$DFFF	U N B E N U T Z T
\$A000	
\$9FFF	RAM - 8 K B Y T E
\$8000	
\$7FFF	U N B E N U T Z T
\$0404	
\$0403	C O N T R O L L E R 2 7 9 3 / 9 7
\$0400	
\$03FF	U N B E N U T Z T
\$0300	
\$02FF	P O R T (I / O + T I M E R)
\$0280	
\$027F	U N B E N U T Z T
\$0200	
\$01FF	S T A C K
\$0100	
\$00FF	Z E R O P A G E
\$0000	

ERKLÄRUNG ZUR SPEICHERBELEGUNG

\$E000 - \$FFFF - BETRIEBSSYSTEM

Hier liegt das Betriebssystem Ihrer SPEEDY 1050. Änderungen können Sie hier nicht vornehmen. Ein genaues Listing dieses Betriebssystems finden Sie im Anhang dieses Handbuches.

\$8000 - \$9FFF - ARBEITSSPEICHER

Der 8K Ram Block ist in 5 Bereiche unterteilt:

\$9F80 - \$9FFF - Hier liegen Einsprung und Rücksprungvektoren für die Bereitschaftsroutine des Betriebssystems. Außerdem können Sie hier Erweiterungen der RESET Routine vornehmen.

\$9F00 - \$9F7F - Die normale und erweiterte Kommandotabelle und die entsprechenden Einsprünge sind hier zu finden. Über das Kommando \$41 können Sie diese Tabelle beliebig verändern. Diesen Befehl (und andere) haben wir Ihnen bereits im SPEEDY 1050 Handbuch erklärt, Sie werden Sie aber auch noch einmal, etwas später, in diesem ANWENDER HANDBUCH finden.

\$9E00 - \$9EFF - Der EXTENDED BUFFER dient zur Zwischenspeicherung von Sektordaten bei FAST WRITE oder beim SLOW MODE diverser Laufwerksfunktionen.

\$8C00 - \$9DFF - In diesem Bereich liegt der Trackbuffer. Hier werden bei FAST WRITE oder FAST READ erst alle Sektordaten eines Tracks zwischengespeichert. Schalten Sie die SPEEDY 1050 zum Beispiel mit Hilfe der Menu-Diskette (Menupunkt SLOW MODE CONTROL) für READ SECTOR und WRITE SECTOR in den SLOW MODUS, wird dieser Speicherbereich nicht mehr vom Betriebssystem benutzt. So können Sie auch hier eigene Routinen ablegen.

\$8000 - \$8BFF - Freier Speicherbereich, der dem Programmierer ständig zur Verfügung steht, also wo Sie Ihre eigenen Programme ablegen können!

\$0400 - \$0403 - Hier liegen die Register des Disk-Controllers
2793/97:

\$0400 : lesen=Statustregister, Schreiben=Command-
register
\$0401 : Lesen + Schreiben=Trackregister
\$0402 : Lesen + Schreiben=Sektorregister
\$0403 : Lesen + Schreiben=Datenregister

\$0280 - \$02FF - Hier befinden sich die Register des Port IC's
6532 (RIOT)

Die gebräuchlichsten Register:

\$0280 : Port A Datenregister
\$0281 : Port A Richtungsregister
\$0282 : Port B Datenregister
\$0283 : Port B Richtungsregister
\$0296 : Timer lesen/schreiben, Timer IRQ
abschalten
\$029F : Timer mit Teilverhältnis 1:1K lesen/
schreiben, Timer IRQ einschalten

\$0000 - \$00FF - Die Zeropage und die Page 1 überlagern sich.
Das heißt, Speicherstelle \$0000 entspricht der
Speicherstelle \$0100, Speicherstelle \$0001 ent-
spricht der Speicherstelle \$0101, usw. In der
Zeropage stehen dem Benutzer die Speicherstellen
\$0090 bis \$00CF zur freien Verfügung.

DIE EINSPRUNGADRESSEN

Nachdem wir Ihnen nun die Speicherbelegung der SPEEDY 1050 erklärt haben, erfahren Sie jetzt etwas über die Einsprungadressen. Wichtig ist dabei eines zu wissen: Wir haben am Ende des ROM-Bereiches ab \$FF00 eine Jumptable eingerichtet. Wenn Sie nun eine Funktion innerhalb der SPEEDY ausführen wollen, brauchen Sie nur eine, oder mehrere Adressen dieser Jumptable anzuspringen.

Welche Versionen es in Zukunft auch von der SPEEDY 1050 Software geben wird, diese Jumptable wird immer an der gleichen Stelle und in der gleichen Reihenfolge erhalten bleiben. Somit ist gewährleistet, das alle Software, die jetzt für die SPEEDY 1050 geschrieben wird, auch auf den zukünftigen Versionen laufen wird. (Wenn die internen Routinen über diese Tabelle angesprungen werden!)

Hier die Beschreibung der einzelnen Jump Vektoren:

RESET - \$FF00 - DRIVE KALTSTART

Port A und Port B (6532) werden initialisiert, der komplette RAM Bereich gelöscht und der Disk-Controller getestet. Sollte ein Fehler beim Testen auftreten, erfolgt ein Sprung nach "SYSERO", wo 2 mal "BELL" ausgegeben wird, und anschliessend folgt ein Sprung nach "RESET2".

RESET2 - \$FF03 - DRIVE WARMSTART

Die System-Variablen werden neu gesetzt, das Laufwerk in den "SINGLE-DENSITY-MODUS" gebracht, der SchreibLesekopf auf Track 0 justiert und jede Controller Tätigkeit gestopt. Zum zurücksetzen des Laufwerkes sollte diese Routine angesprungen werden, da bei "\$FF00 - RESET" alle im RAM befindlichen Daten und Programme gelöscht werden.

BEREIT - \$FF06 - BEREITSCHAFTSRoutine

Zuerst wird der "SLOW" Schalter abgefragt und das Laufwerk gegebenenfalls in den "SLOW" Modus geschaltet. Sollten sich noch zu schreibende Sektoren im RAM befinden, wird direkt nach "TSTCO" verzweigt, damit, falls die Laufwerksklappe geöffnet wird, der Motor nicht ausgeschaltet wird. Wurde nicht verzweigt, wird die Laufwerksklappe mit der letzten Stellung verglichen. Sollte die Klappe geöffnet worden sein, so werden der Antriebsmotor und der Steppermotor ausgeschaltet und der Controller-Status nach "CONST" kopiert. Ist die Klappe geschlossen worden, wird die DENSITY neu festgestellt, das Laufwerk entsprechend eingestellt und die Sektor-Folge neu gelesen. In der "TESTCO" Routine wird die "COMMAND" Leitung vom Computer abgefragt. Sollte diese "gesetzt" werden, erfolgt ein Sprung nach "RDINF", wo das Kommando vom Computer eingelesen wird. Ist die "COMMAND"-Leitung nicht gesetzt wird der Motor-Timer heraufgezählt. Sollte dieser den Wert \$9800 (16 Bit) erreichen, wird "TSTWR" aufgerufen um alle Sektoren zu schreiben, die sich noch im RAM befinden.

MOTON - \$FF09 - MOTOR ZWINGEND EINSCHALTEN

Der Antriebsmotor wird ohne Rücksicht auf die Stellung der Laufwerksklappe eingeschaltet.

TSTMON - \$FFOC - MOTOR EINSCHALTEN, WENN DIE LAUFWERKSKLAPPE
GESCHLOSSEN IST

Erst wird die Laufwerksklappe abgefragt. Ist diese geschlossen, wird der Antriebsmotor eingeschaltet und eine Zeitverzögerung durchgeführt, um den Motor auf Touren kommen zu lassen.

MOTOFF - \$FFOF - MOTOR AUSSCHALTEN

Der Antriebsmotor wird ausgeschaltet und das entsprechende Bit in "DRSTAT" gesetzt.

SDELAY - \$FF12 - MOTOR TIMER EINSTELLEN

Diese Routine wird nach einer Kommandoausführung abgearbeitet. Zuerst wird die Zeit gestellt, wie lange der Motor nach einer Kommandoausführung noch eingeschaltet bleiben soll, dann der Kommando-Buffer gelöscht. Danach erfolgt ein Rücksprung in die Motor-Timer-Routine innerhalb der Bereitschafts-Routine.

SDRDDP - \$FF15 - DRIVE DENSITY EINSTELLEN UND ANZEIGEN

Zuerst wird nach "DENDSP" gesprungen um die aktuelle DENSITY auf dem Display anzuzeigen. Danach wird, je nach Wert in "FORKEN" der Drive-Status, die Anzahl der Sektoren pro Track und die Anzahl der Bytes per Sektor gesetzt.

XWAIT - \$FF18 - WARTESCHLEIFE KURZ

Der Wert im X-Register gibt die Anzahl der Schleifendurchläufe an. Ein Schleifendurchlauf entspricht ca. 100 Taktzyklen (0.1 msec/100µsec).

X2WAIT - \$FF1B - LANGE WARTESCHLEIFE

Der Wert im X-Register gibt die Anzahl der Schleifendurchläufe an. Ein Schleifendurchlauf entspricht ca. 100.000 Taktzyklen (0.1 sec/100msec).

TRACKO - \$FF1E - TRACK 0 POSITIONIEREN

Der Disk-Controller wird gestoppt und der Schreib/Lesekopf so lange zurückgezogen, bis der Track-0-Sensor anspricht. Danach wird eine Zeitverzögerung zum ausschwingen der Kopfmehchanik durch geführt.

TRADJA - \$FF21 - TRACK ANZEIGEN UND SCHREIB/LESEKOPF POSITIONIEREN

Zuerst wird die Tracknummer angezeigt und der Controller gestoppt. Ist die Klappe geschlossen wird der Motor eingeschaltet und die Anzahl der Doppel-Steps errechnet, die durchgeführt werden müssen, um die gewünschte Kopfposition zu erreichen. Sollte der Trackwechsel über 40 Tracks gehen, wird 2*"BELL" ausgegeben und der Kopf auf Track 0 positioniert.

Nach erfolgreicher Schreib/Lesekopf Positionierung wird die Tracknummer in das Track-Register des Controllers kopiert und der Schreib/Lesekopf Mechanik Zeit zum ausschlagen gegeben.

TRADJ - \$FF24 - SCHREIB-LESE-KOPF POSITIONIEREN

Entspricht der \$FF21 Routine, mit dem Unterschied, das die Tracknummer nur dann angezeigt wird, wenn ein Trackwechsel statt gefunden hat.

TRVR - \$FF27 - 1 STEP VORWÄRTS ODER RÜCKWÄRTS AUSFÜHREN

Im Y-Register ist die Kennung für Step vorwärts oder rückwärts (plus oder minus). Die Bitposition des Stepermotors wird entsprechend herauf- oder herabgezählt und das neue Bitmuster in Port B des Portbausteins geschrieben.

Anmerkung:

Für 1 Trackwechsel müssen immer 2 Steps erfolgen. Das heißt aber nicht, daß ohne bedenken 80 Tracks geschrieben oder gelesen werden können. Die beiden Steps sind verschieden lang!

CONRES - \$FF2A - DISK CONTROLLER STOPPEN

Dem Disk-Controller wird der Befehl gegeben, alle laufenden Aktionen zu stoppen. In "WREADY" wird gewartet, bis das der Controller den Befehl als "ausgeführt" meldet.

CONRE2 - \$FF2D - \$FF2A WIRD ZWEIMAL AUSGEFÜHRT

Beim Einsprung in diese Routine wird die Routine bei \$FF2A - DISK CONTROLLER STOPPEN zweimal ausgeführt.

WREADY - \$FF30 - AUF CONTROLER "IN USE" FLAG=0 WARTEN

Bei dieser Routine wird darauf gewartet, daß der Controller meldet, daß der letzte Befehl abgeschlossen wurde.

RD128 - \$FF33 - 128 BYTES VOM COMPUTER NACH "EXBUF" HOLEN

Der Buffer wird auf "EXBUF" (Extended Buffer) und die IO-Länge auf 128 Bytes gesetzt.

RD256 - \$FF36 - 256 BYTES VOM COMPUTER NACH "EXBUF" HOLEN

Wie bei der vorhergehenden Routine wird der Buffer auf "EXBUF", aber die IO-Länge auf 256 Bytes gesetzt.

RDBTS - \$FF39 - ANZAHL BYTES IN DEN BUFFER HOLEN (X/Y REGISTER)

Im Accu stehen die Anzahl der Bytes, im X- und Y-Register die Low- und High-Adresse des Buffers, in die die Daten vom Computer abgelegt werden sollen. Der Timer wird gesetzt, um zu verhindern, das der Prozessor hängen bleiben kann. Es wird jeweils ein Byte über einen indirekten Jump-Vector vom Computer geholt (ausser bei HIGH SPEED) und in dem Buffer abgelegt. Anschließend wird die Checksumme heraufgezählt und geprüft, ob alle Bytes und Datenblocks geholt wurden. Die Checksumme wird verglichen und die IO-Länge neu gesetzt.

RDSFOL - \$FF3C - NACH VERZÖGERUNG SEKTORFOLGE VOM AKTUELLEM TRACK LESEN

Die Verzögerungsschleife am Anfang der Routine verhindert, daß versucht wird, die HEADER schon zu lesen, wenn die Laufwerks-Klappe noch nicht vollständig geschlossen ist. Danach wird die Zeit gestellt, die der Controller zur Verfügung hat, um alle HEADER zu lesen. Ist ein HEADER eingelesen, wird die Track- und Sektor- Nummer auf Gültigkeit geprüft. Befindet sich die Sektornummer des gelesenen HEADERS bereits in der Sektorliste (doppelter Sektor), so wird die Sektorfolge nicht mehr weiter gelesen und das Laufwerk in den Slow-Modus geschaltet. Stimmt die Anzahl der gelesenen Sektor-HEADER nicht mit der vorgegebenen (18 oder 26 Sektoren) überein, wird das Laufwerk ebenfalls in den Slow-Modus geschaltet.

RDTRA - \$FF42 - AKTUELLEN TRACK INS RAM EINLESEN

Die Track-Slow-Kennung wird gesetzt. Über RDHDSP wird der Kopf positioniert und der nächste HEADER eingelesen. Anschließend wird die Position der Sektor-Nummer des HEADERS in der Sektorliste gesucht. Ab der nächsten Position werden die Sektoren dann in der richtigen Reihenfolge eingelesen. Die Statuswerte der Sektoren werden in die Statusliste eingetragen. Statuswerte, ausser \$08 (CRC-ERROR) und \$20 (AM-ERROR) unterbrechen dabei das Einlesen der Sektoren. Nur wenn alle Sektoren des Tracks gelesen wurden, wird die Kennung für Track-Slow zurückgesetzt.

RDTRAV - \$FF45 - WIE \$FF42, JEDOCH MIT VERIFY UND EINEM RETRY

Hier wird zuerst die Track- und Sektornummern errechnet und anschließend \$FF42 (RDTRA) aufgerufen. Dann wird die Statusliste auf \$08 (CRC-ERROR) und \$20 (AM-ERROR) getestet. Ist einer der beiden Statuswerte eingetragen, wird der entsprechende Sektor nochmals gelesen.

TSTWR - \$FF48 - NOCH ZU SCHREIBENDE SEKTOREN AUS DEM RAM AUF
DIE DISKETTE SCHREIBEN

In "WRKEN" steht die Anzahl der zu schreibenden Sektoren. Ist der Wert=0 wird die Routine sofort verlassen. Sonst wird der Motor eingeschaltet, die Tracknummer für die zu schreibenden Sektoren aus "LWRTRA" nach "TRACK" kopiert und der Schreib/Lesekopf positioniert. Anschließend wird die nächste HEADER eingelesen und die Sektornummer in der Sektorliste gesucht. Ist der Sektor nicht eingetragen (zum Beispiel bei geschützten Disketten, die mit "FAST-WRITE" beschrieben wurden), wird 1 Bell ausgegeben. Ansonsten wird ab der gefundenen Sektorposition die Statusliste auf negative Werte (Kennung für zu schreibende Sektoren) überprüft. Wird ein solcher Wert gefunden, wird der entsprechende Sektor geschrieben und in der Statusliste als "geschrieben" (\$40) eingetragen. "WRKEN" wird um eins herunter gezählt und die Position in der Sektorliste heraufgezählt.

Sollte beim schreiben eines Sektors ein Fehler auftreten, in dem der "WRITE PROTECT-SCHALTER" umgeschaltet oder die Laufwerksklappe geöffnet wurde, so wird die Routine "WRERR" aufgerufen, in der dem Anwender auf optischem und akustischem Wege ca. 5 Sekunden Zeit gegeben wird, um die Laufwerksklappe wieder zu schliessen. Alle "WRITE-PROTECTIERTEN" Sektoren werden in der Statusliste als geschrieben (\$40) eingetragen.

TSTDAT - \$FF4B - \$FF48 AUFRUFEN UND STATUSLISTE MIT \$40 FÜLLEN

Diese Routine muß abgearbeitet werden, wenn das Laufwerk in den "SLOW-MODE" geschaltet wurde, damit alle Sektoren als geschrieben bzw. als nicht gelesen markiert werden.

SD128B - \$FF4E - 128 BYTES VOM EXBUF ZUM COMPUTER SENDEN

Die Übertragungslänge wird auf 128 Bytes und der Buffer auf "EXBUF" gesetzt.

SD256B - \$FF51 - 256 BYTES VOM EXBUF ZUM COMPUTER SENDEN

Die Übertragungslänge wird auf 256 Bytes und der Buffer auf "EXBUF" gesetzt.

SDBTS - \$FF54 - ACCU=ANZAHL DER BYTES AUS BUFFER
(X/Y-REGISTER) SENDEN

Es wird jeweils ein Byte aus dem Buffer geladen, die Checksumme heraufgezählt und über die Jump-Tabelle der Senderoutinen gesprungen, um das gelesene Byte zum Computer zu senden. Dieser Vorgang wird in einer Schleife solange wiederholt, bis der gesamte Buffer gesendet wurde. Anschließend wird die Checksumme gesendet.

SEND41 - \$FF57 - QUITTUNG \$41 ZUM COMPUTER SENDEN

Der Accu wird mit dem Wert \$41 (A) geladen und nach einer Verzögerung (damit die Quittung an den Computer nicht zu schnell kommt) über die Jump-Tabelle der Senderoutinen gesprungen, um die Quittung in der richtigen Übertragungsgeschwindigkeit zu senden.

SEND43 - \$FF5A - QUITTUNG \$43 ZUM COMPUTER SENDEN

Das gleiche wie bei "SEND41"-\$FF57, jedoch mit dem Quittungsbyte \$43 (C).

SEND45 - \$FF5D - QUITTUNG \$45 ZUM COMPUTER SENDEN

Das gleiche wie bei "SEND41"-\$FF57, jedoch mit dem Quittungsbyte \$45 (E).

SEND4E - \$FF60 - QUITTUNG \$4E ZUM COMPUTER SENDEN

Das gleiche wie bei "SEND41"-\$FF57, jedoch mit dem Quittungsbyte \$4E (N).

RDSECT - \$FF63 - AKTUELLEN SEKTOR VON DER DISKETTE IN DEN
VORBEZEICHNETEN RAM BEREICH EINLESEN

Sektornummer in das Sektor-Register kopieren.
"READ-SEKTOR"- Befehl an den Controller geben.
"TIME-OUT"-Zeit setzen. Nun werden die Daten
Byte für Byte vom Controller übernommen und in
den bezeichneten Buffer (indirekt "IND") abge-
legt. Sind alle Daten gelesen wird auf den Con-
troller gewartet, bis dieser seine Arbeit einge-
stellt hat und der Status des gelesenen Sektors
in das Status-Register des Controllers übernom-
men. Sollte ein "TIME-OUT" aufgetreten sein,
wird noch ein Leseversuch gestartet.

RDSEC1 - \$FF66 - BEZEICHNETEN SEKTOR IN BEZEICHNETEN RAM
EINLESEN

Die selbe Routine wie RDSECT-\$FF63. Nur muß die
Sektornummer bereits in das Sektor-Register des
Controllers geschrieben sein.

WRSECT - \$FF69 - AKTUELLEN SEKTOR VON VORBEZEICHNETER
RAMADRESSE AUF DIE DISKETTE SCHREIBEN

Sektornummer in das Sektor-Register kopieren.
"WRITESECTOR"- Befehl an den Controller geben.
"TIME-OUT"-Zeit setzen. Nun werden die Daten
Byte für Byte an den Controller übergeben.

Sind alle Daten übergeben, wird auf den Control-
ler gewartet, bis dieser seine Arbeit eingestel-
lt hat, und der Write-Status vom Controller über-
nommen wird. Sollte ein "TIME-OUT" aufgetreten
sein, wird überprüft, ob der Controller noch ar-
beitet. Wenn ja, wird noch ein Schreibversuch ge-
startet.

WRSEC1 - \$FF6C - BEZEICHNETEN SEKTOR VON BEZEICHNETER
RAMADRESSE AUF DISKETTE SCHREIBEN

Die selbe Routine wie WRSECT- \$FF69. Nur muß die
Sektornummer bereits in das Sektor- Register des
Controllers geschrieben sein.

TSTWRP - \$FF6F - "WRITE PROTECT" UND LAUFWERKSKLAPPE TESTEN

Es wird "CONRES" aufgerufen, wo der Disk-Control-
ler seine augenblickliche Arbeit unterbricht und
der Controller Status gelesen wird. Anschließend
werden Bit 6 (WRITE PROTECT) und Bit 7 (LAUF-
WERKS-KLAPPE) ausmaskiert. Ist eines der beiden
Bits gesetzt, können keine Daten geschrieben
werden!

VERSEC - \$FF72 - AKTUELLEN SEKTOR MIT ANGEgebenEN RAM
VERGLEICHEN

Sektornummer in das Sektor-Register kopieren.
"READ-SEKTOR"-Befehl an den Controller geben.
"TIME-OUT"- Zeit setzen.

Nun werden die Daten Byte für Byte vom Controller übernommen und mit der bezeichneten Adresse (indirekt "IND") verglichen. Ist ein Wert ungleich, wird das Lesen unterbrochen, der Controller gestoppt, die Kennung für "DATEN UNGLEICH" (ACCU <>0) gesetzt und CARRY für "KEIN LESE FEHLER AUFGETRETEN" gelöscht.

Stimmen alle gelesenen Daten mit dem angegebenen Buffer überein, wird die Kennung für "DATEN GLEICH" (ACCU=0) und "KEIN LESEFEHLER" (CARRY=0) gesetzt. Tritt während des Vergleiches ein "TIME OUT" auf, wird geprüft, ob der Controller noch arbeitet. Wenn ja, wird der Vergleich der Daten fortgesetzt. Ansonsten wird "CARRY" gesetzt (KENNUNG FÜR LESEFEHLER).

VERSE1 - \$FF75 - BEZEICHNETEN SEKTOR MIT ANGEgebenEM RAM
VERGLEICHEN

Die selbe Routine wie VERSEC - \$FF72. Nur muß die Sektornummer in das Sektor-Register des Controllers geschrieben sein.

STELL - \$FF78 - COM-STATUS AUF ERROR UND 2 RETRY'S SETZEN

"RETRY" wird auf zwei Versuche und "COMST" vorsorglich auf "COMMAND-ERROR" gesetzt.

QUITT - \$FF7B - QUITTUNG "C" ODER "E" ZUM COMPUTER SENDEN

Den Controller Status übernehmen. Wenn "CONST" auf "COMMAND ERROR" steht, wird die Kennung für "FEHLER BEI LETZTER LAUFWERKS-OPERATION" in "DRSTAT" gesetzt. Wenn Bit 7 und Bit 0 in "DSPCTR" gesetzt sind, wird der Controller Status auf dem Display angezeigt und ein "BELL" ausgegeben, und die Quittung \$45 ("E") zum Computer gesendet. Ist "COMMAND STATUS" o.k. wird die Kennung für "LAUFWERKS-OPERATION-IN-ORDNUNG" gesetzt und die Quittung \$43 ("C") zum Computer gesendet.

RDHEAD - \$FF7E - DIE NÄCHSTEN HEADER DATEN LESEN

"TIME OUT"-Zeit setzen. "READ HEADER"-Befehl an den Controller geben. Nun werden die 6 Bytes des nächsten auffindbaren "HEADER'S" in einer Schleife eingelesen und ab der Adresse \$7A abgelegt. In "WREADY" wird darauf gewartet, daß der Controller seine Arbeit einstellt. CARRY als Lesefehler-Flag wird zurückgesetzt.

RDHD1 - \$FF81 - WIE RDHEAD, ABER TIMER NICHT SETZEN

Die gleiche wie \$FF7E. Nur muß "TIME OUT" bereits gesetzt sein.

CALCTS - \$FF87 - TRACK- UND SEKTORNUMMER ERRECHNEN

Sektornummer LOW und HIGH werden zum "IND"-Pointer kopiert (für RAM/ROM Adressen) und auf Nummer=0 oder Nummer größer als \$7FFF geprüft. Ist das der Fall, wird in "DUMKEN" noch der SLOW-Status getestet. Andernfalls wird die Sektornummer in IND/IND+1 solange um die Anzahl der Sektoren pro Track herabgezählt, bis diese die Nummer Null unterschreitet. Als Ergebnis hat man die gewünschte Track- und Sektornummer. Die Tracknummer wird noch mit dem Wert 40 verglichen. Das CARRY-Flag wird durch den Vergleich entsprechend gesetzt. Nach der Rückkehr aus dieser Routine stehen die Prozessor Status-Flags wie folgt:

C=1 SEKTORNUMMER UNZULÄSSIG
N=1 RAM/ROM ADRESSE
Z=1 ZERO-PAGE ADRESSE 0

SETBUF - \$FF8A - BUFFER NACH AKTUELLEM SEKTOR SETZEN

Je nach Sektorlänge wird die Sektornummer durch zwei geteilt und zur Anfangsadresse des Datenbuffers hinzu addiert. Die Bufferadresse befindet sich dann in IND/ IND+1.

SETBUF2 - \$FF8D - BUFFER NACH SEKTORNUMMER IM ACCU SETZEN

Entspricht der Routine "SETBUF" - \$FF8A, jedoch muß die Sektornummer (1...26) bereits im ACCU stehen.

SEXBUF - \$FF90 - ADRESSE DES EXTENDED BUFFERS SETZEN

Die Adresse von "EXBUF" wird nach IND/IND+1 geladen.

SETRWL - \$FF93 - ANZAHL DER BYTES FÜR ZU ÜBERTRAGENDEN
DATENBLOCK SETZEN

Bei Sektornummern von 1 bis 3 wird die Übertragungslänge auf 128 Bytes, ansonsten auf den Wert von "SECLEN" gesetzt. Anzahl der Datenblocks wird auf 1 gesetzt.

COPSLT - \$FF96 - SEKTORLISTE FÜR AKTUELLE DENSITY IN ZERO PAGE
KOPIEREN

In "SDRDDP" werden die Werte für die aktuelle DENSITY richtig gesetzt und die DENSITY auf dem Display angezeigt. Anschließend wird je nach Wert in "FORKEN" (DENSITY-Kennung) die entsprechende Sektorliste für SINGLE- oder DOUBLE-DENSITY nach "SECLST" (\$20) kopiert.

BELL1 - \$FF99 - 1 MAL BELL ÜBER DEN SUMMER AUSGEBEN

Der Summer wird mittels einer Verzögerungsschleife mit einer bestimmten Frequenz angesteuert.

CLRDSP - \$FF9C - DISPLAY ABSCHALTEN

In die Display Speicherstellen \$4000, \$4001 und \$4002 werden Nullen geschrieben. Die Speicherstellen sollten nur im Schreibzugriff (z.B. STA \$4000) angesprochen werden, da sonst auf dem Display unkontrollierbare Zeichen erscheinen.

TRAANZ - \$FF9F - AKTUELLE TRACKNUMMER ANZEIGEN

Der ACCU wird mit dem Wert von "TRACK" geladen, und es wird, je nach Wert in "DSPCTR" zur Dezimal- oder Hexadezimal Ausgaberroutine gesprungen.

DEZOUT - \$FFA2 - WERT IM ACCU WIRD IN DEZIMALER FORM ANGEZEIGT

Der Wert im ACCU wird in einer Schleife um 10 heruntergezählt, bis er den Wert 0 unterschreitet. Der Schleifenzähler entspricht dann dem Wert für das 10'ner Stellen- Display, und die Restsumme dem Wert für das 1'ner Stellen- Display. Die Werte für die richtige Segmentsteuerung wird einer Konstantentabelle (SEGTBL) entnommen.

HEXOUT - \$FFA8 - WERT IM ACCU WIRD IN HEXADEZIMALER FORM ANGEZEIGT

Zuerst werden die unteren vier Bits des Wertes im ACCU ausmaskiert, die dem Wert für das rechte Display entsprechen, dann die oberen vier Bits.

DENDSP - \$FFA8 - AKTUELLE DENSITY AUF DEM DISPLAY ANZEIGEN

Je nach Wert in "FORKEN" (DENSITY Kennung) werden die entsprechenden Segmente des Display's angesteuert.

SETTIM - \$FFAB - TIMER MIT DEM WERT IM ACCU SETZEN

Timer Interrupt Flag wird gelöscht und der Timer mit dem Wert im ACCU gestartet.

CLRTRA - \$FFAE - EINEN TRACK REFORMATIEREN

In "FSTART" wird das "WRITE-TRACK"-Kommando gestartet und der Timer gesetzt. Nun wird der Track mit dem Wert \$AA beschrieben, bis der Timer abgelaufen ist. Es wird der Track "gelöscht", auf dem sich der Schreib/Lesekopf befindet.

CLRDSK - \$FFB1 - GANZE DISKETTE REFORMATIEREN

Hier werden alle Track's nacheinander, beginnend bei Track 39 (39...0) gelöscht. Der Schreib/Lesekopf wird jeweils positioniert und "CLRTRA" aufgerufen.

RAMTST - \$FFB4 - TEST DES LAUFWERK INTERNEN RAM'S

Im ersten Teil wird die Zero Page getestet. Der Wert der Speicherstelle, die getestet wird, wird jeweils um EXBUF+1 zwischengespeichert. Zuerst wird die Zero Page mit dem Wert \$55 getestet, daß heißt, der Wert \$55 wird in die Speicherstelle geschrieben und wieder gelesen. Ist der Wert gleich geblieben, ist die Speicherstelle in Ordnung. Anschließend wird die Zero Page noch einmal mit dem Wert \$AA getestet.

Ist während des Testes kein Fehler festgestellt worden, wird der Speicherbereich von \$8000 bis RAMTOP (\$A000) auf die gleiche Art getestet, wie die Zero Page. Tritt bei einer Speicherstelle ein Fehler auf, wird die Adresse jener Speicherstelle in \$90/\$91 abgelegt und der Test abgebrochen.

Ist kein RAM-Fehler festgestellt worden, steht in \$90/ \$91 die höchste RAM-Adresse. Nach Abschluß der RAM-Testroutine wird die Adresse, die in \$90/ \$91 steht, zum Computer gesendet.

ROMTST - \$FFB7 - ROM TEST

Vorsorglich wird Command-Status auf ERROR gesetzt. In IND/IND+1 wird die Adresse \$E000 gesetzt. Anschliessend wird für eine Page die Checksumme errechnet und mit dem Originalwerten in einer Tabelle verglichen. Ist die Checksumme gleich, wird die High-ROM-Adresse in \$91 um eins heraufgezählt und die nächste ROM-Page getestet. Insgesamt werden so 32 ROM-Pages (\$E000 bis \$FFFF) getestet. Stimmen alle Checksummen mit denen der "ROMCHK" Tabelle überein, wird der Command-Status zurück gesetzt und die Quittung ("C") zum Computer gesendet. Ist ein Fehler gefunden worden, wird Command-Status nicht zurückgesetzt und die Quittung ("E") zum Computer gesendet.

SPEEDT - \$FFBA - MOTOR SPEED TEST

Vorsorglich Command-ERROR setzen und Schreib/Lesekopf auf Track 0 positionieren. In "FDSEC1" wird Sektor 1 zweimal direkt nacheinander gelesen, und die Zeit über Taktzyklen gemessen. Dann wird in einer Schleife die gemessenen Zeit von der Konstanten \$COE1E4 solange abgezählt, bis der Wert 0 unterschritten wird. Die folgende Nachkomma Stellenrundung wird mit dem Rest der vorhergehenden Rechnung vorgenommen, in dem die gemessene Zeit durch zwei geteilt und von dem Rest der vorhergehenden Rechnung abgezogen wird. Ist das CARRY-Flag dann gesetzt, wird die Nachkommastelle um eins erhöht. Der aus zwei Daten bestehende Speed-Wert wird zum Computer gesendet. Der Hexadezimale Wert \$2875 bedeutet dabei 287,5 UPM.

DIE BEFEHLE DER SPEEDY 1050

So, nachdem Sie nun mehr über die Einsprung-
adressen wissen, und bevor wir zum ROM-Listing
kommen, hier nun noch einmal alle Befehle der
SPEEDY 1050 und Ihre Anwendung.

KOMMANDO ist der Wert, der sich vor Aufruf der
SIO - Routine (\$E459) in der Speicherstelle
\$0302 befindet.

AUX1 und AUX2 entsprechen den Werten, die sich
in den Speicherstellen \$030A und \$030B befinden.
Bei einigen Befehlen werden AUX1 und AUX2 nicht
benutzt und dürfen beliebige Werte annehmen.

Die Befehle sind im übrigen nicht nach den
Hexadezimalnummern geordnet, sondern nach Ihrer
Funktion!

KOMMANDO: \$52
FUNKTION: Sektor lesen
AUX1: Sektornummer oder ROM-Adresse LOW BYTE
AUX2: Sektornummer oder ROM-Adresse HIGH BYTE
BESCHREIBUNG: Es werden je nach DENSITY 128 oder 256 Bytes
gesendet. Sektoren 1-3 sind immer 128 Bytes
lang.

KOMMANDO: \$50
FUNKTION: Sektor schreiben ohne Verify
AUX1: Sektornummer oder ROM-Adresse LOW BYTE
AUX2: Sektornummer oder ROM-Adresse HIGH BYTE
BESCHREIBUNG: Das Laufwerk erwartet je nach DENSITY 128 oder
256 Bytes. Sektoren 1-3 sind immer 128 Bytes
lang.

KOMMANDO: \$57
FUNKTION: Sektor schreiben mit Verify
AUX1: Sektornummer oder ROM-Adresse LOW BYTE
AUX2: Sektornummer oder ROM-Adresse HIGH BYTE
BESCHREIBUNG: Das Laufwerk erwartet je nach DENSITY 128 oder
256 Bytes. Die Sektoren 1-3 sind immer 128
Bytes lang.

KOMMANDO: \$53
FUNKTION: Laufwerkstatus
AUX1: nicht Benutzt
AUX2: nicht Benutzt
BESCHREIBUNG: Das Laufwerk sendet 4 Bytes, die den Status der letzten Diskettenoperation beinhalten.

Byte 1: Drive Status

Bit 0 - COMMAND FRAME ERROR
Bit 1 - CHECKSUM ERROR
Bit 2 - OPERATION ERROR
Bit 3 - WRITE PROTECT
Bit 4 - MOTOR ON
Bit 5 - DOUBLE DENSITY
Bit 6 - unbenutzt
Bit 7 - DUAL DENSITY

Byte 2: Controller Status

Bit 0 - BUSY
Bit 1 - DRQ
Bit 2 - LOST DATA
Bit 3 - CRC ERROR
Bit 4 - RECORD NOT FOUND
Bit 5 - RECORD TYPE
Bit 6 - WRITE PROTECT
Bit 7 - NOT READY

Byte 3: Time-Out Wert für Format Disk (\$E0)

Byte 4: unbenutzt (immer 0)

KOMMANDO: \$21
FUNKTION: Formatiere Diskette (SINGLE/DOUBLE DENSITY)
AUX1: nicht benutzt
AUX2: nicht benutzt
BESCHREIBUNG: Dieses Kommando wird benutzt um Disketten in SINGLE- oder DOUBLE - DENSITY (720 Sektoren) zu formatieren. Das DENSITY-Format wird durch einen vorherigen \$4F - Befehl (Laufwerkskonfiguration) eingestellt. Wird das Laufwerk nach dem Einschalten nicht konfiguriert, wird automatisch in SINGLE-DENSITY formatiert. Das Laufwerk sendet nach dem formatieren je nach DENSITY 128 oder 256 Bytes an den Computer. Die ersten zwei Bytes müssen immer \$FF sein.

KOMMANDO: \$22
FUNKTION: Formatiere Diskette (MEDIUM DENSITY)
AUX1: nicht benutzt
AUX2: nicht benutzt
BESCHREIBUNG: Dieses Kommando wird benutzt um Disketten in 1050 DUAL DENSITY (MEDIUM = 1040 Sektoren) zu formatieren. Es werden immer 128 Bytes zum Computer gesendet. Die ersten beiden Bytes müssen immer \$FF sein.

KOMMANDO: \$20
FUNKTION: Automatisches Formatieren
AUX1: Konfigurationsbyte
AUX2: nicht benutzt
BESCHREIBUNG: Dem Laufwerk wird nur der Befehl zum Form-
 tieren gegeben. Es wird sofort ein 'Complete'
 zurückgesendet. Mit diesem Befehl können alle
 drei Formate, abhängig vom Konfigurationsbyte
 (00=SINGLE, \$20= DOUBLE, \$80=MEDIUM) generiert
 werden. Ein Write-Protect wird sofort zurück-
 gemeldet.

Fehler beim formatieren können dem Computer nicht gemeldet werden, da keine Daten nach Befehlsausführung zurückgesendet werden. Der Formatierungsvorgang und eventuelle Formatierungsfehler können auf dem Display verfolgt werden. Abhängig vom Drive/Display - Status Befehl wird nach dem formatieren automatisch die VTOC (Dos 2.5 kompatibel) und 3 Bootsektoren geschrieben.

KOMMANDO: \$3F
FUNKTION: SIO - Geschwindigkeitsbyte ermitteln
AUX1: nicht benutzt
AUX2: nicht benutzt
BESCHREIBUNG: Es wird ein Byte zum Computer gesendet, das
 die HIGH SPEED Übertragungsgeschwindigkeit be-
 inhaltet. Dieses Byte wird für die HIGH SPEED
 SIO - Routine benötigt und beträgt bei der
 SPEEDY 1050 normalerweise \$09.

KOMMANDO: \$4E
FUNKTION: Laufwerkskonfiguration auslesen
AUX1: nicht benutzt
AUX2: nicht benutzt
BESCHREIBUNG: Es werden die 12 Bytes der Konfigurations-
tabelle zum Computer gesendet. Die Bedeutung
der einzelnen Bytes sind:

Byte 1 - Anzahl der Tracks (immer 40)
Byte 2 - Step Rate (immer 1)
Byte 3 - Sektoren pro Track HIGH (immer 0)
Byte 4 - Sektoren pro Track LOW (18 oder 26)
Byte 5 - Anzahl der Köpfe (immer 0)
Byte 6 - Aufzeichnungformat (0=FM/4=MFM)
Byte 7 - Bytes pro Sektor HIGH (1=256/0=128)
Byte 8 - Bytes pro Sektor LOW (0=256/128=128)
Byte 9 - Laufwerk aktiv (immer 255)
Byte 10 - unbenutzt (immer 0)
Byte 11 - unbenutzt (immer 0)
Byte 12 - unbenutzt (immer 0)

KOMMANDO: \$4F
FUNKTION: Laufwerk konfigurieren
AUX1: nicht benutzt
AUX2: nicht benutzt
BESCHREIBUNG: Dieser Befehl wird benutzt um das Laufwerk für
den nächsten Formatierungsbefehl einzustellen.
Das Laufwerk erwartet 12 Bytes, die genau der
Reihenfolge des vorherigen Befehls (\$4E) ent-
sprechen müssen.

KOMMANDO: \$51
FUNKTION: Schreibvorgang beenden
AUX1: nicht benutzt
AUX2: nicht benutzt
BESCHREIBUNG: Nach jedem Schreibbefehl wartet das Laufwerk ca. 2 Sekunden bis die Daten aus dem Trackbuffer auf die Diskette geschrieben werden. Dieses wird durch den Befehl \$51 beschleunigt. Alle Daten im Trackbuffer werden unverzüglich auf die Diskette geschrieben und abhängig vom Drive/Display Befehl (\$44) wird der Motor nach erfolgtem Schreibvorgang sofort gestoppt.

KOMMANDO: \$44
FUNKTION: Drive/Display Einstellung
AUX1: Konfigurationsbyte
AUX2: nicht benutzt
BESCHREIBUNG: Der Wert in AUX1 setzt das Drive/Display Byte im Laufwerk. Dieses Byte kann über keinen Befehl direkt ausgelesen werden, so daß immer alle Bits richtig gesetzt werden müssen.

Die einzelnen Bits beinhalten die folgenden Funktionen:

- Bit 0 - BELL bei ERROR zulassen
- Bit 1 - unbenutzt
- Bit 2 - unbenutzt
- Bit 3 - bei Kommando \$51 Motor nicht aus
schalten
- Bit 4 - bei Kommando \$20 VTOC+Boot Sektoren
nicht schreiben
- Bit 5 - Formatieren ohne Verify
- Bit 6 - Trackanzeige in Hexadezimal
- Bit 7 - ERROR - Anzeige zulassen

KOMMANDO: \$4B
FUNKTION: Slow/Fast Konfiguration
AUX1: Konfigurationsbyte
AUX2: nicht benutzt
BESCHREIBUNG: Mit dem Wert in AUX1 wird das Drive-Slow-Status Byte des Laufwerkes beeinflusst. Dieses Byte kann über keinen Befehl direkt ausgelesen werden, so daß alle Bits richtig gesetzt werden müssen.

Die einzelnen Bits haben folgende Funktionen:

- Bit 0 - Read Sektor slow
- Bit 1 - Write Sektor slow
- Bit 2 - Kommando \$57 Verify ausschalten
- Bit 3 - Laufwerk vollständig im 'Slow-Mode'
- Bit 4 - unbenutzt
- Bit 5 - unbenutzt
- Bit 6 - 1 Track slow (nach Trackwechsel 0)
- Bit 7 - 1 Diskette slow (nach Diskettenwechsel 0)

KOMMANDO: \$4C
FUNKTION: Direkter Sprungbefehl ohne Rückmeldung
AUX1: Sprungadresse LOW BYTE
AUX2: Sprungadresse HIGH BYTE
BESCHREIBUNG: Der Prozessor im Laufwerk wird durch diesen Befehl veranlaßt, direkt zur Speicherstelle zu springen, die sich in AUX1 und AUX2 befindet. Das Laufwerk gibt keine Rückmeldung an den Computer zurück, so daß eine Rückmeldung von dem Programm aus gegeben werden muß, zu dem der Prozessor gesprungen ist.

KOMMANDO: \$4D
FUNKTION: Direkter Sprungbefehl mit Rückmeldung
AUX1: Sprungadresse LOW BYTE
AUX2: Sprungadresse HIGH BYTE
BESCHREIBUNG: Dieser Befehl gleicht dem Vorhergehenden bis auf den kleinen Unterschied, daß das Laufwerk vor ausführen des Programms eine Rückmeldung an den Computer gibt.

KOMMANDO: \$41
FUNKTION: Kommandotabelle verlängern oder verkürzen
AUX1: nicht benutzt
AUX2: nicht benutzt
BESCHREIBUNG: Das Laufwerk erwartet 3 Bytes vom Computer. Das 1. Byte ist das neue Kommando. Das 2. und 3. Byte ist die Startadresse des über das neue Kommando erreichten Programms in LOW/HIGH-Byte Format. Falls sich der neue Befehl schon in der Kommandotabelle befindet, wird dieser mit der neuen Startadresse versehen. Ist die Startadresse \$0000 wird der Befehl aus der Kommandotabelle gelöscht.

KOMMANDO: \$68
FUNKTION: Länge der SIO - Routine ermitteln
AUX1: nicht benutzt
AUX2: nicht benutzt
BESCHREIBUNG: Mit diesem Befehl wird die Länge der SIO -
Routine ermittelt, die mit dem Befehl \$69 aus
dem Laufwerk in den Computer geladen wird. Das
Laufwerk sendet 2 Bytes, die die Länge (LOW/
HIGH) beinhalten.

KOMMANDO: \$69
FUNKTION: SIO - Routine zum Computer senden
AUX1: Relokator - Adresse LOW BYTE
AUX2: Relokator - Adresse HIGH BYTE
BESCHREIBUNG: Dieser Befehl sendet die HIGH SPEED SIO -
Routine mit der vom Befehl \$68 ermittelten
Länge zum Computer. Diese Routine wird bereits
im Laufwerk zur Startadresse hin Relokiert,
die sich in AUX1 und AUX2 befindet.

KOMMANDO: \$60
FUNKTION: Track schreiben
AUX1: Track Anfangssektor oder Anfangsadresse LOW
BYTE
AUX2: Track Anfangssektor oder Anfangsadresse HIGH
BYTE
BESCHREIBUNG: Die kompletten Daten für einen Track werden
mit diesem Befehl auf die Diskette oder in den
Trackbuffer geschrieben. Die Anzahl der zu
übertragenden Bytes errechnet sich aus der An-
zahl der Sektoren mal der Bytes pro Sektor.
Wegen des sehr schwierigen Timings funktio-
niert dieser Befehl nur in normaler Übertra-
gungsgeschwindigkeit.

KOMMANDO: \$62
FUNKTION: Track lesen
AUX1: Track Anfangssektor LOW BYTE
AUX2: Track Anfangssektor HIGH BYTE
BESCHREIBUNG: Lesen eines kompletten Tracks mit einem Befehl
von der Diskette oder aus dem Trackbuffer. Die
Anzahl der zu erwartenden Bytes errechnet sich
aus der Anzahl der Sektoren mal der Bytes pro
Sektor.

ANHANG A


```

0100 ;*****
0110 ;* Demonstration 1 fuer Kommando $52 *
0120 ;*****
0130 ;
0140 DATBUF = $5000
0150 SECNUM = 1
0160 ;
0170     .OPT NO LIST
0180     .OPT OBJ
0190     *= $4000
0200 ;
0210     LDA #$31      ; Bus ID
0220     STA $0300
0230     LDA #1       ; Laufwerks Nummer = 1
0240     STA $0301
0250     LDA #$52     ; Kommando $52
0260     STA $0302
0270     LDA #$40     ; Status fuer Daten lesen
0280     STA $0303
0290     LDA # <DATBUF ; Adresse fuer Datenbuffer Low
0300     STA $0304
0310     LDA # >DATBUF ; Adresse fuer Datenbuffer High
0320     STA $0305
0330     LDA #7       ; Wert fuer Timeout = 7 Sekunden
0340     STA $0306
0350     LDA #$80     ; 128 Bytes (in SD+MD) schreiben
0360     STA $0308
0370     LDA #0
0380     STA $0309
0390     LDA # <SECNUM ; Sector Nummer Low Byte
0400     STA $030A
0410     LDA # >SECNUM ; Sector Nummer High Byte
0420     STA $030B
0430     JSR $E459    ; Einsprung der SIO-Routine im OS
0440     BMI ERROR
0450     CLC
0460     RTS
0470 ERROR SEC
0480     RTS

```

```

0100 ;*****
0110 ;* Demonstration 2 fuer Kommando $52 *
0120 ;*****
0130 ;
0140 PRGBUF = $8000
0150 DATBUF = $5000
0160 ;
0170     .OPT NO LIST
0180     .OPT OBJ
0190     *= $4000
0200 ;
0210     LDA #$31      ; Bus ID
0220     STA $0300
0230     LDA #1       ; Laufwerks Nummer = 1
0240     STA $0301
0250     LDA #$52     ; Kommando $52
0260     STA $0302
0270     LDA #$40     ; Status fuer Daten lesen
0280     STA $0303
0290     LDA # <DATBUF ; Adresse fuer Datenbuffer Low
0300     STA $0304
0310     LDA # >DATBUF ; Adresse fuer Datenbuffer High
0320     STA $0305
0330     LDA #7       ; Wert fuer Timeout = 7 Sekunden
0340     STA $0306
0350     LDA #$80     ; 128 Bytes (in SD+MD) schreiben
0360     STA $0308
0370     LDA #0
0380     STA $0309
0390     LDA # <PRGBUF ; Adresse fuer Programmbuffer Low
0400     STA $030A
0410     LDA # >PRGBUF ; Adresse fuer Programmbuffer High
0420     STA $030B
0430     JSR $E459    ; Einsprung der SIO-Routine im OS
0440     BMI ERROR
0450     CLC
0460     RTS
0470 ERROR SEC
0480     RTS

```

```

0100 ;*****
0110 ;* Demonstration 1 fuer Kommando $50 *
0120 ;*****
0130 ;
0140 DATBUF = $5000
0150 SECNUM = 1
0160 ;
0170 .OPT NO LIST
0180 .OPT OBJ
0190 *= $4000
0200 ;
0210 LDA #$31 ; Bus ID
0220 STA $0300
0230 LDA #1 ; Laufwerks Nummer = 1
0240 STA $0301
0250 LDA #$50 ; Kommando $50
0260 STA $0302
0270 LDA #$80 ; Status fuer Daten schreiben
0280 STA $0303
0290 LDA # <DATBUF ; Adresse fuer Datenbuffer Low
0300 STA $0304
0310 LDA # >DATBUF ; Adresse fuer Datenbuffer High
0320 STA $0305
0330 LDA #7 ; Wert fuer Timeout = 7 Sekunden
0340 STA $0306
0350 LDA #$80 ; 128 Bytes (in SD+MD) schreiben
0360 STA $0308
0370 LDA #0
0380 STA $0309
0390 LDA # <SECNUM ; Sector Nummer Low Byte
0400 STA $030A
0410 LDA # >SECNUM ; Sector Nummer High Byte
0420 STA $030B
0430 JSR $E459 ; Einsprung der SIO-Routine im OS
0440 BMI ERROR
0450 CLC
0460 RTS
0470 ERROR SEC
0480 RTS

```

```

0100 ;*****
0110 ;* Demonstration 2 fuer Kommando $50 *
0120 ;*****
0130 ;
0140 PRGBUF = $8000
0150 ;
0160 .OPT NO LIST
0170 .OPT OBJ
0180 *= $4000
0190 ;
0200 LDA #$31 ; Bus ID
0210 STA $0300
0220 LDA #1 ; Laufwerks Nummer = 1
0230 STA $0301
0240 LDA #$50 ; Kommando $50 Sector ohne Verify schreiben
0250 STA $0302
0260 LDA #$80 ; Status fuer Daten schreiben
0270 STA $0303
0280 LDA # <DATBUF ; Adresse fuer Datenbuffer Low
0290 STA $0304
0300 LDA # >DATBUF ; Adresse fuer Datenbuffer High
0310 STA $0305
0320 LDA #7 ; Wert fuer Timeout = 7 Sekunden
0330 STA $0306
0340 LDA #$80 ; 128 Bytes (in SD+MD) schreiben
0350 STA $0308
0360 LDA #0
0370 STA $0309
0380 LDA # <PRGBUF ; Adresse fuer Programmbuffer Low
0390 STA $030A
0400 LDA # >PRGBUF ; Adresse fuer Programmbuffer High
0410 STA $030B
0420 JSR $E459 ; Einsprung der SIO-Routine im OS
0430 BMI ERROR
0440 CLC
0450 RTS
0460 ERROR SEC
0470 RTS
0480 ;
0490 DATBUF
0500 LDA #$FA
0510 JMP $FFA5 ;HEXOUT

```



```

0100 ;*****
0110 ;*      Demonstration fuer Kommando $3F      *
0120 ;*  Uebertragungsrate fuer High-Speed ermitteln *
0130 ;*****
0140 ;
0150 DATBUF = $5000
0160 ;
0170     .OPT NO LIST
0180     .OPT OBJ
0190     *= $4000
0200 ;
0210     LDA #$31      ; Bus ID
0220     STA $0300
0230     LDA #1        ; Laufwerks Nummer = 1
0240     STA $0301
0250     LDA #$3F      ; Kommando $3F
0260     STA $0302
0270     LDA #$40      ; Status fuer Daten lesen
0280     STA $0303
0290     LDA # <DATBUF ; Adresse fuer Datenbuffer Low
0300     STA $0304
0310     LDA # >DATBUF ; Adresse fuer Datenbuffer High
0320     STA $0305
0330     LDA #7        ; Wert fuer Timeout = 7 Sekunden
0340     STA $0306
0350     LDA #1        ; 1 Byte lesen
0360     STA $0308
0370     LDA #0
0380     STA $0309
0390     JSR $E459     ; Einsprung der SIO-Routine im OS
0400     BMI ERROR
0410     CLC
0420     RTS
0430 ERROR SEC
0440     RTS

```

```

0100 ;*****
0110 ;* Demonstration fuer Kommando $44 *
0120 ;*****
0130 ;
0140 .OPT NO LIST
0150 .OPT OBJ
0160 *= $4000
0170 ;
0180 LDA #$31 ; Bus ID
0190 STA $0300
0200 LDA #1 ; Laufwerks Nummer = 1
0210 STA $0301
0220 LDA #$44 ; Kommando $44
0230 STA $0302
0240 LDA #0 ; Status fuer keine Daten senden oder empfangen
0250 STA $0303
0260 LDA #7 ; Wert fuer Timeout = 7 Sekunden
0270 STA $0306
0280 LDA #$20 ; Bit 4 fuer Format ohne Verify
0290 STA $030A
0300 JSR $E459 ; Einsprung der SIO-Routine im OS
0310 BMI ERROR
0320 CLC
0330 RTS
0340 ERROR SEC
0350 RTS

```

```

0100 ;*****
0110 ;* Demonstration fuer Kommando $4B *
0120 ;*****
0130 ;
0140 .OPT NO LIST
0150 .OPT OBJ
0160 *= $4000
0170 ;
0180 LDA #$31 ; Bus ID
0190 STA $0300
0200 LDA #1 ; Laufwerks Nummer = 1
0210 STA $0301
0220 LDA #$4B ; Kommando $4B
0230 STA $0302
0240 LDA #0 ; Status fuer keine Daten senden oder empfangen
0250 STA $0303
0260 LDA #7 ; Wert fuer Timeout = 7 Sekunden
0270 STA $0306
0280 LDA #3 ; Bit 0+1 fuer Sector Read + Write Slow
0290 STA $030A
0300 JSR $E459 ; Einsprung der SIO-Routine im OS
0310 BMI ERROR
0320 CLC
0330 RTS
0340 ERROR SEC
0350 RTS

```

```

0100 ;*****
0110 ;* Demonstration fuer Kommando $4C *
0120 ;* Einsprungbefehl. "C" - Complete *
0130 ;* muss selbst gesendet werden ! *
0140 ;*****
0150 ;
0160 GOADR = $FF5A ; Einsprungsadresse fuer "C" - Complete senden
0170 ;
0180 .OPT NO LIST
0190 .OPT OBJ
0200 *= $4000
0210 ;
0220 LDA #$31 ; Bus ID
0230 STA $0300
0240 LDA #1 ; Laufwerks Nummer = 1
0250 STA $0301
0260 LDA #$4C ; Kommando $4C
0270 STA $0302
0280 LDA #0 ; Status fuer keine Daten senden oder empfangen
0290 STA $0303
0300 LDA #7 ; Wert fuer Timeout = 7 Sekunden
0310 STA $0306
0320 LDA # <GOADR ; Einsprungsadresse Low Byte
0330 STA $030A
0340 LDA # >GOADR ; Einsprungsadresse High Byte
0350 STA $030B
0360 JSR $E459 ; Einsprung der SIO-Routine im OS
0370 BMI ERROR
0380 CLC
0390 RTS
0400 ERROR SEC
0410 RTS

```

```

0100 ;*****
0110 ;*   Demonstration fuer Kommando $4D   *
0120 ;* Einsprungbefehl. "C" - Complete wird *
0130 ;* vom Laufwerk sofort zurueckgesendet *
0140 ;*****
0150 ;
0160 GOADR = $FF03   ; Einsprungsadresse fuer "Drive Reset"
0170 ;
0180     .OPT NO LIST
0190     .OPT OBJ
0200     *= $4000
0210 ;
0220     LDA #$31     ; Bus ID
0230     STA $0300
0240     LDA #1      ; Laufwerks Nummer = 1
0250     STA $0301
0260     LDA #$4D    ; Kommando $4D
0270     STA $0302
0280     LDA #0      ; Status fuer keine Daten senden oder empfangen
0290     STA $0303
0300     LDA #7      ; Wert fuer Timeout = 7 Sekunden
0310     STA $0306
0320     LDA # <GOADR ; Einsprungsadresse Low Byte
0330     STA $030A
0340     LDA # >GOADR ; Einsprungsadresse High Byte
0350     STA $030B
0360     JSR $E459   ; Einsprung der SIO-Routine im OS
0370     BMI ERROR
0380     CLC
0390     RTS
0400 ERROR SEC
0410     RTS

```

```

0100 ;*****
0110 ;* Demonstration 1 fuer Kommando $41 *
0120 ;* Kommando $54 Installieren *
0130 ;*****
0140 ;
0150 .OPT NO LIST
0160 .OPT OBJ
0170 *= $4000
0180 ;
0190 LDA #$31 ; Bus ID
0200 STA $0300
0210 LDA #1 ; Laufwerks Nummer = 1
0220 STA $0301
0230 LDA #$41 ; Kommando $41
0240 STA $0302
0250 LDA #$80 ; Status fuer Daten schreiben
0260 STA $0303
0270 LDA # <COMBUF ; Adresse fuer Kommandobuffer Low
0280 STA $0304
0290 LDA # >COMBUF ; Adresse fuer Kommandobuffer High
0300 STA $0305
0310 LDA #7 ; Wert fuer Timeout = 7 Sekunden
0320 STA $0306
0330 LDA #3 ; 3 Bytes schreiben
0340 STA $0308
0350 LDA #0
0360 STA $0309
0370 JSR $E459 ; Einsprung der SIO-Routine im OS
0380 BMI ERROR
0390 CLC
0400 RTS
0410 ERROR SEC
0420 RTS
0430 COMBUF .BYTE $54 ; Kommando $54
0440 .WORD $8000 ; Einsprung $8000

```

```

0100 ;*****
0110 ;*   Demonstration 2 fuer Kommando $41   *
0120 ;*           Kommando $3F Loeschen     *
0130 ;*****
0140 ;
0150     .OPT NO LIST
0160     .OPT OBJ
0170     *= $4000
0180 ;
0190     LDA #$31       ; Bus ID
0200     STA $0300
0210     LDA #1         ; Laufwerks Nummer = 1
0220     STA $0301
0230     LDA #$41       ; Kommando $41
0240     STA $0302
0250     LDA #$80       ; Status fuer Daten schreiben
0260     STA $0303
0270     LDA # <COMBUF ; Adresse fuer Kommandobuffer Low
0280     STA $0304
0290     LDA # >COMBUF ; Adresse fuer Kommandobuffer High
0300     STA $0305
0310     LDA #7         ; Wert fuer Timeout = 7 Sekunden
0320     STA $0306
0330     LDA #3         ; 3 Bytes schreiben
0340     STA $0308
0350     LDA #0
0360     STA $0309
0370     JSR $E459      ; Einsprung der SIO-Routine im OS
0380     BMI ERROR
0390     CLC
0400     RTS
0410 ERROR SEC
0420     RTS
0430 COMBUF .BYTE $3F ; Kommando $3F
0440     .WORD $00      ; Kennung fuer Kommando loeschen

```

```

0100 ;*****
0110 ;* Demonstration fuer Kommando $60 *
0120 ;*   'Write Track' - Befehl   *
0130 ;*****
0140 ;
0150 TRKDAT = $5000 ; Adresse der kompletten Trackdaten
0160 TRKLEN = $0900 ; $900 SD, $D00 MD, $1200 DD
0170 SECTOR = 1 ; Anfangssector eines Tracks
0180 ;
0190 .OPT NO LIST
0200 .OPT OBJ
0210 *= $4000
0220 ;
0230 LDA #$31 ; Bus ID
0240 STA $0300
0250 LDA #1 ; Laufwerks Nummer = 1
0260 STA $0301
0270 LDA #$60 ; Kommando $60
0280 STA $0302
0290 LDA #$80 ; Status fuer Daten senden
0300 STA $0303
0310 LDA # <TRKDAT ; Trackdaten Low Byte
0320 STA $0304
0330 LDA # >TRKDAT ; Trackdaten High Byte
0340 STA $0305
0350 LDA #7 ; Wert fuer Timeout = 8 Sekunden
0360 STA $0306
0370 LDA # <TRKLEN ; Tracklaenge Low Byte
0380 STA $0308
0390 LDA # >TRKLEN ; Tracklaenge High Byte
0400 STA $0309
0410 LDA # <SECTOR
0420 STA $030A
0430 LDA # >SECTOR
0440 STA $030B
0450 JSR $E459 ; Einsprung der SIO-Routine im OS
0460 BMI ERROR
0470 CLC
0480 RTS
0490 ERROR SEC
0500 RTS

```



```

0100 ;*****
0110 ;* Demonstration fuer Kommando $62 *
0120 ;*      'Read Track' - Befehl      *
0130 ;*****
0140 ;
0150 TRKDAT = $5000 ; Adresse der kompletten Trackdaten
0160 TRKLEN = $0900 ; $900 SD, $D00 MD, $1200 DD
0170 SECTOR = 1 ; Anfangssector eines Tracks
0180 ;
0190 .OPT NO LIST
0200 .OPT OBJ
0210 *= $4000
0220 ;
0230 LDA #$31 ; Bus ID
0240 STA $0300
0250 LDA #1 ; Laufwerks Nummer = 1
0260 STA $0301
0270 LDA #$62 ; Kommando $62
0280 STA $0302
0290 LDA #$40 ; Status fuer Daten lesen
0300 STA $0303
0310 LDA # <TRKDAT ; Trackdaten Low Byte
0320 STA $0304
0330 LDA # >TRKDAT ; Trackdaten High Byte
0340 STA $0305
0350 LDA #7 ; Wert fuer Timeout = 8 Sekunden
0360 STA $0306
0370 LDA # <TRKLEN ; Tracklaenge Low Byte
0380 STA $0308
0390 LDA # >TRKLEN ; Tracklaenge High Byte
0400 STA $0309
0410 LDA # <SECTOR
0420 STA $030A
0430 LDA # >SECTOR
0440 STA $030B
0450 JSR $E459 ; Einsprung der SIO-Routine im OS
0460 BMI ERROR
0470 CLC
0480 RTS
0490 ERROR SEC
0500 RTS

```

```

0100 ;*****
0110 ;* Lesen der SIO-Routine vom Laufwerk *
0120 ;*****
0130 ;
0140 ADR =    $5000    ; Adresse fuer die SIO-Routine
0150 ;
0160     .OPT NO LIST
0170     .OPT OBJ
0180     *=    $4000
0190 ;
0200     LDA    $$31    ; Bus ID
0210     STA    $0300
0220     LDA    #1      ; Laufwerks Nummer = 1
0230     STA    $0301
0240     LDA    $$68    ; Kommando $68
0250     STA    $0302
0260     LDA    $$40    ; Status fuer Daten lesen
0270     STA    $0303
0280     LDA    #8
0290     STA    $0304    ; Adresse fuer Laengenbyte Low
0300     STA    $0306    ; Wert fuer Timeout = 8 Sekunden
0310     LDA    #3
0320     STA    $0305    ; Adresse fuer Laengenbyte High
0330     LDA    #2
0340     STA    $0308    ; 2 Bytes lesen
0350     LDA    #0
0360     STA    $0309
0370     JSR    $E459    ; Einsprung der SIO-Routine im OS
0380     BMI    ERROR
0390     INC    $0302    ; Kommando $69
0400     LDA    # <ADR
0410     STA    $0304    ; Target Adresse der SIO-Routine Low
0420     STA    $030A    ; Original Adresse der SIO-Routine Low
0430     LDA    # >ADR
0440     STA    $0305    ; Target Adresse der SIO-Routine High
0450     STA    $030B    ; Original Adresse der SIO-Routine High
0460     LDA    $$40
0470     STA    $0303    ; Status fuer Daten lesen
0480     JSR    $E459    ; Einsprung der SIO-Routine im OS
0490     BMI    ERROR
0500     CLC
0510     RTS
0520 ERROR SEC
0530     RTS

```

ANHANG B

NACHRÜSTEN DER SPEEDY N

Die Nachrüstung der Trackanzeige und des Summers erfordert eine gute Lötpraxis und eine gute Erfahrung im Umgang mit Mikroelektronischen Schaltungen. Diese Erweiterung sollten also nur Diejenigen vornehmen, die schon des öfteren Bauteile auf Platinen gelötet haben. Wir übernehmen keinerlei Garantie für eine durch fehlerhaften Einbau zerstörte SPEEDY- Platine oder Diskettenlaufwerks.

Besorgen Sie sich bitte folgende Bauelemente:

2 x 74 LS 74 mit passenden Sockeln
2 x 74 LS 273 mit passenden Sockeln
3 x 7-Segmentanzeige HA1077 oder D100PK oder
FND 357 mit gem. Kathode
D 1 N 4148
T BC 517
C 0,1 μ F
R1 4,7 k Ohm
R2 47 Ohm
R3 270 Ohm
LS Miniaturlautsprecher 8 Ohm
R Widerstandsarray 8 x 270 Ohm
oder 16 Einzelwiderstände 270 Ohm
Pfeifenfeldstiftleisten 2 x 10 polig mit
passendem Stecker.
ca. 10cm Flachbandkabel 20 polig
Lochrasterplatine

Bei Beschaffungsschwierigkeiten helfen wir natürlich gerne weiter.

Alle Bauteile werden laut Bestückungsplan auf die SPEEDY- Grundplatine gelötet. Achten Sie hierbei auf die Richtung der IC's, der Diode und des Transistors. Die Widerstandsarrays können durch Einzelwiderstände ersetzt werden.

ACHTUNG!!! Die Platine ist doppelseitig Beschichtet und Durchkontaktiert. Ein einmal eingelötetes Bauteil läßt sich nur sehr schwer wieder auslöten.

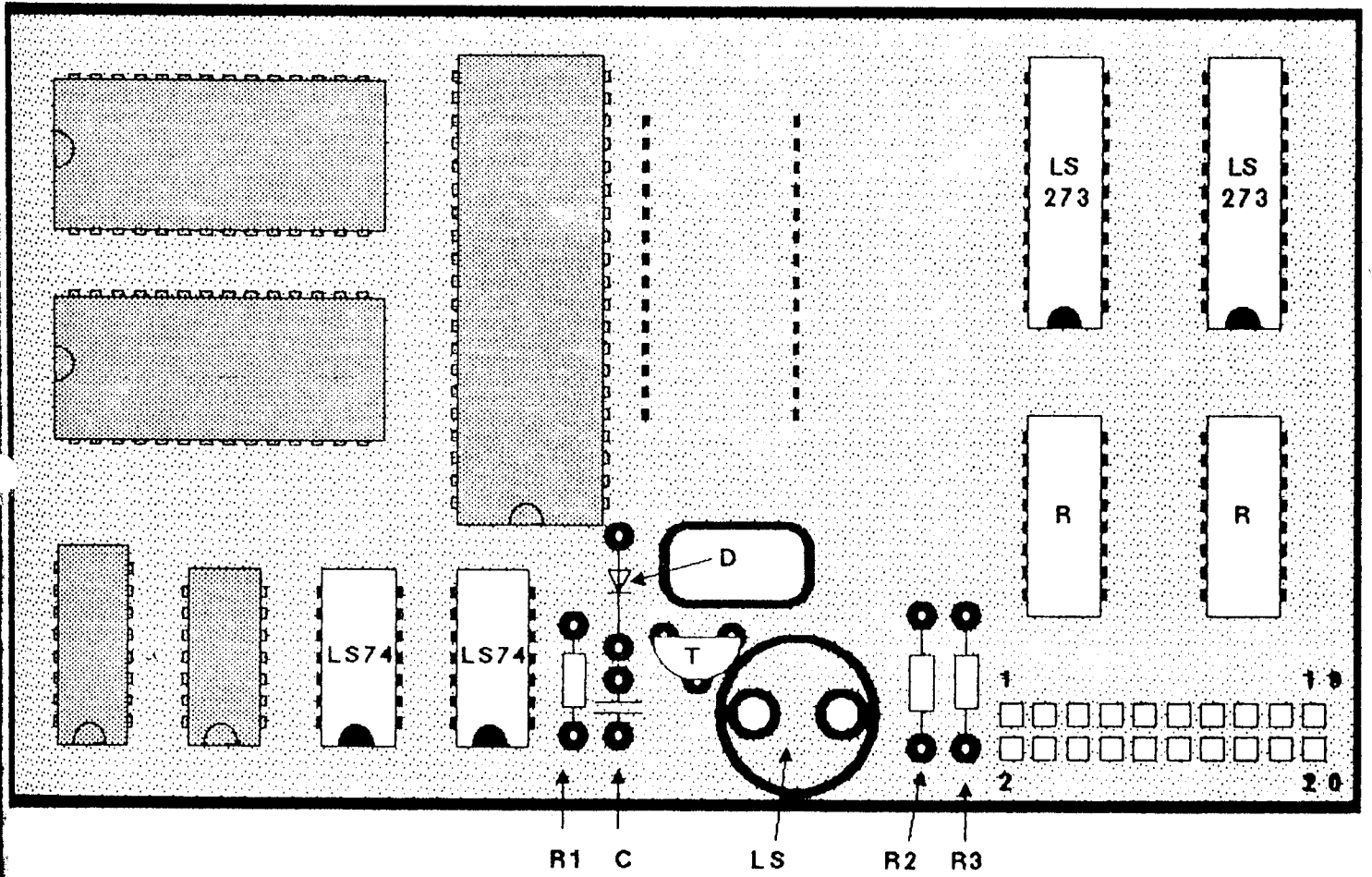
Die Pinbelegung der 7-Segmentanzeigen erhalten Sie aus den Datenblättern der Hersteller oder durch Testen mit einem Ohmmeter. Löten Sie die drei Anzeigen nebeneinander auf ein kleines Stück Lochrasterplatine.

Das eine Ende des Flachbandkabel wird an den 20-poligen Stecker angepreßt und das lose Ende wird abisoliert und mit den Siebensegment - Anzeigen verbunden.

Nachfolgende Liste soll Ihnen bei der Anschlußbelegung behilflich sein:

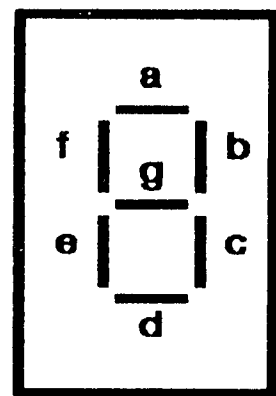
Anschlußpunkt	Anzeige	Segment	Bemerkung
1 2	links	d	Single Density
2 1	links	g	Medium Density
3 4	links	a	Double Density
4 3			frei
5 6	mitte	g	
6 5	alle		gem. Masse
7 8	mitte	e	
8 7	mitte	f	
9 10	mitte	c	
10 9	mitte	d	
11 12	mitte	a	
12 11	mitte	b	
13 14	rechts	g	
14 13	alle		gem. Masse
15 16	rechts	e	
16 15	rechts	f	
17 18	rechts	c	
18 17	rechts	d	
19 20	rechts	a	
20 19	rechts	b	

Nach erfolgter Aufrüstung Ihre SPEEDY 1050 können Sie die Funktionstüchtigkeit mit Hilfe der SPEEDY-Systemdiskette (Display-Test) überprüfen.



NACHRÜSTUNG DER TRACKANZEIGE

7 - SEGMENT ANZEIGE
BEZEICHNUNG DER SEGMENTE



ANHANG C

*= \$E000

;

VERSION = \$10

;

Vom System verwendete Zero-Page Adressen:

MERK1	= \$00	
MERK2	= \$01	
MERK3	= \$02	
DLYT1	= \$03	Timer LO fuer Motor-Timer Routine
DLYT2	= \$04	Timer HI fuer Motor-Timer Routine
LDSW	= \$05	Letzte 'Dumm'-Schalter Position
WRKEN	= \$06	Anzahl der zu schreibenden Sektoren im Ram
EXSECT	= \$07	Sector # der Daten in Extended Buffer
DUMKEN	= \$08	Drive 'Dumm'-Status
FORKEN	= \$09	Aktuelles Density 0=DD, 41=MD, 82=SD
FORKEN2	= \$0A	Density fuer Format, wird von COMAF gesetzt
LWRTRA	= \$0B	Track # der zu schreibenden Sektoren im RAM
LTRACK	= \$0C	Track # des zuletzt gelesenen Sectors
TRACK	= \$0D	Aktuelle Track #
SECTOR	= \$0E	Aktuelle Sector #
CONST	= \$0F	Controller Status
DRSTAT	= \$10	Drive Status
COMST	= \$11	Command Status
RETRY	= \$12	Anzahl der Retry's fuer Read/Write (normal 2)
RWLEN	= \$13	I/O Laenge
SECLN	= \$14	Anzahl der Bytes pro Sector
USKEN	= \$15	Kennung fuer Uebertragungsgeschwindigkeit
DLYTIM	= \$16	Zeit, wie lange der Motor nach einem Befehl noch laeuft
STPTIM	= \$17	Verzoegerung fuer Steppermotor
COMPOS	= \$18	Position des letzten Befehls in Command-Tabelle
IND	= \$19	Indirekt-Vektor fuer Daten-Buffer
CHKSUM	= \$1B	Checksumme fuer Datenuebertragung
RDDATK	= \$1C	Kennung, ob Daten vom Computer geholt werden muessen
KLAPPE	= \$1D	Letzte Klappen-Position
SECANZ	= \$1F	Sector-Anzahl pro Track, die vorhanden sein muessen
SECANZ1	= \$1E	Sector-Anzahl pro Track, die vorhanden sind
SECLST	= \$20	Sektorenliste
STALST	= \$40	Sektoren Statusliste
STPPOS	= \$60	Bit-Position fuer Steppermotor
DSPCTR	= \$61	Display/Drive-Controllbyte
BLOCKS	= \$62	Anzahl der Datenblocks fuer Datenuebertragung
IOIND	= \$63	

Die Zero-Page Adressen \$90-\$CF sind unbenutzt

DATBUF	= \$8C00	Datenbuffer fuer Sektoren
EXBUF	= \$9E00	Extended-Buffer
CMTBL	= \$9F00	Command-Tabelle

Fuer eigene Programme steht der Speicherbereich \$8000-\$8BFF zur Verfuegung

```

E000 D8      RESET    CLD                * Kaltstart *
E001 A2FF                    LDX #$FF
E003 9A                    TXS                Stackpointer neu setzen
E004 A938                    LDA #$38
E006 8D8102                   STA $0281                PADIR
E009 A938                    LDA #$38                8=Motor aus
E00B 8D8002                   STA $0280                PADAT
E00E A93D                    LDA #$3D
E010 8D8302                   STA $0283                PBDIR
E013 8D8202                   STA $0282                PBDAT
E016 2078E3                   JSR CONRE2
E019 A980                    LDA #$80
E01B 851A                    STA IND+1                Adresse $8000 setzen
E01D A900                    LDA #0
E01F 8519                    STA IND
E021 A220                    LDX #$20
E023 A8                      TAY
E024 9119      DRAML    STA (IND),Y        Speicherbereich $8000-$9FFF loeschen
E026 C8                      INY
E027 D0FB                    BNE DRAML
E029 E61A                    INC IND+1
E02B CA                      DEX
E02C D0F6                    BNE DRAML
;
E02E A960                    LDA #$60                =RTS: Erweiterung der Reset-Routine vorgesehen
E030 8D849F                   STA CMTBL+$84
E033 A955                    LDA #$55
E035 8D0104                   STA $0401
E038 8D0204                   STA $0402
E03B A21E                    LDX #$1E
E03D CA                      RZS2
E03E D0FD                    BNE RZS2
E040 AD0104                   LDA $0401
E043 4D0204                   EOR $0402                Disk-Controller auf Funktionstuechtigkeit pruefen
E046 D01B                    BNE SYSERO
E048 A948                    LDA #$48
E04A 8D0004                   STA $0400
E04D A228                    LDX #$28
E04F 20F2E2                   JSR XWAIT
E052 AD0004                   LDA $0400
E055 4A                      LSR A
E056 900B                    BCC SYSERO
E058 A228                    LDX #$28
E05A 20F2E2                   JSR XWAIT
E05D AD0004                   LDA $0400
E060 4A                      LSR A
E061 9010                    BCC RESET2
;
E063 2068E0      SYSERO    JSR SYSERR        * System-Error Routine *
E066 800B                    BRA RESET2            * 2xBell und Reset *
;
E068 20B2EF      SYSERR    JSR BELL1        System-Error Routine

```

```

E06B A280          LDX #$80          gibt 2 x Bell aus
E06D 20F2E2        JSR XWAIT
E070 4CB2EF        JMP BELL1
;
E073 A200  RESET2  LDX #0          * Reset-Einsprung *
E075 7400  DELL    STZ 0,X
E077 EB          INX          Zeropage loeschen
E078 D0FB        BNE DELL
;
E07A E61D          INC KLAPPE        =1 Initialisieren der System-Variablen
E07C E605          INC LDSW          =1
E07E E662          INC BLOCKS        =1
E080 A940          LDA #$40
E082 8516          STA DLYT1M        Zeit fuer Motor-Timer Routine
E084 A92C          LDA #$2C
E086 8517          STA STPTIM        Step-Zeit fuer Steppermotor testen
E088 A982          LDA #$82          Single Density - Status testen
E08A 8509          STA FORKEN
E08C 850A          STA FORKEN2
E08E 20A9E1        JSR SDRDDP        System auf Single Density setzen und anzeigen
E091 20CCE2        JSR TROJUS        Kopf auf Track 0 positionieren
E094 20F6EF        JSR TRAAZ        und Track 0 anzeigen
E097 207BE3        JSR CONRE2        2 x Controller Reset ausfuehren
;
E09A A23C          LDX #COMEND-COMTBL
E09C BDA9E5  COMMOV LDA COMTBL,X      Kommando-Tabelle vom Rom
E09F 9D009F        STA CMTBL,X      ins Ram kopieren
EOA2 CA          DEX
EOA3 10F7          BPL COMMOV
;
EOA5 A900          LDA # <TSTCO2
EOA7 8D809F        STA CMTBL+$80    Vektor fuer Ruecksprung in die Motor-Timer-Routine setzen
EOAA A91E          LDA # >TSTCO2    * Wenn ueber diesen Vektor in die Motor-Timer-Routine *
EOAC 8D819F        STA CMTBL+$81    * gesprungen wird, muss zuvor in DLYT1 und DLYT2 die *
EOAF A913          LDA # <MOTIM     * Zeit eingetragen werden, wie lange der Motor noch laufen *
EOB1 8D829F        STA CMTBL+$82    * soll, wenn kein Kommando mehr kommt *
EOB4 A9E1          LDA # >MOTIM
EOB6 8D839F        STA CMTBL+$83
EOB9 20849F        JSR CMTBL+$84
;
EOBC D8  BEREIT  CLD          * Bereitschaftsroutine *
EOBD A2FF        LDX #$FF        * Testet ob Diskette gewechselt wird *
EOBF 9A          TXS
EOC0 2079E5        JSR TSTD5W        'Dumm'-Schalter pruefen
EOC3 A506          LDA WRKEN        noch Daten zu schreiben ?
EOC5 D036          BNE TSTCO
EOC7 AD0004        LDA $0400
EOCA 2980          AND #$80          Klappe pruefen
EOCC C51D          CMP KLAPPE
EOCE F02D          BEQ TSTCO
EOD0 851D          STA KLAPPE        Klappen-Status setzen
EOD2 AA          TAX

```

```

E0D3 1005      BPL KLZU
E0D5 202DE1    JSR MOTOFF      bei Klappe auf Motor ausschalten
E0D8 801E      BRA CTSTAT
E0DA A9FF      KLZU      LDA ##FF
E0DC 850B      STA LWRTRA      Kennung fuer keine zu schreibenden Daten im RAM
E0DE A200      LDX #0
E0E0 9EB08C    KZDL      STZ DATBUF+$80,X      2.Haelfte der Sector-Buffer Sector 1-3 fuer
E0E3 9EB08D      STZ DATBUF+$0180,X      Double Density loeschen
E0E6 9EB08E      STZ DATBUF+$0280,X
E0E9 E8        INX
E0EA 10F4      BPL KZDL
E0EC 2056E1    JSR TSTDEN      Density feststellen
E0EF 3003      BMI TSTKLX
E0F1 20CDE1    TKLOK      JSR RDSFOL      Sectorfolge lesen
E0F4 A9FF      TSTKLX     LDA ##FF
E0F6 8504      STA DLYT2      Delay Timer High-Byte
E0F8 AD0004    CTSTAT     LDA $0400
E0FB 850F      STA CONST      Controller-Status uebernehmen
;
E0FD 6C809F    TSTCO      JMP (CMTBL+$80)      ($9F80) Vector fuer Kommando-Erkennungs-Routine
E100 A902      TSTCO2     LDA #2              Kommando-Erkennungs-Routine
E102 2C8202    BIT $0282
E105 D00C      BNE MOTIM      Computer aus
E107 100A      BPL MOTIM      Command Bit
;
E109 A515      LDA USKEN      Uebertragungsgeschwindigkeit feststellen
E10B 2901      AND #1
E10D 0A        ASL A
E10E 8563      STA IOIND      Kennung fuer Uebertragungsgeschwindigkeit setzen
E110 4C90E3    JMP RDINF      Befehl vom Computer empfangen
;
E113 E603      MOTIM      INC DLYT1        Motor-Timer-Routine
E115 D0A5      BNE BEREIT
E117 E604      INC DLYT2
E119 F00A      BEQ MOTTOF
E11B A504      LDA DLYT2      bei Fast-Write nach ca. 2 Sekunden die Daten
E11D C998      CMP #$98      aus dem RAM-Speicher auf die Diskette schreiben
E11F D09B      BNE BEREIT
E121 A506      LDA WRKEN
E123 F097      BEQ BEREIT
E125 2064EB    MOTTOF     JSR TSTWR      noch Daten zu schreiben ?
E128 202DE1    JSR MOTOFF     Motor ausschalten
E12B 80BF      BRA BEREIT
;
E12D A908      MOTOFF     LDA #8            Motor ausschalten
E12F 0C8002    TSB $0280     'Motor on' Bit zuruecksetzen
E132 A93C      LDA #$3C
E134 0C8202    TSB $0282     Die 4 Bits des Steppermotors zuruecksetzen
E137 A910      LDA #$10      Drive-Status (Motor aus) setzen
E139 1410      TRB DRSTAT
E13B 60        RTS
;

```

E13C 2C0004	TSTMON	BIT \$0400	Klappe auf ?
E13F 3014		BMI TMOEX	ja
E141 48	MOTON	PHA	sonst Motor einschalten
E142 A908		LDA #8	'Motor on' Bit setzen
E144 1C8002		TRB \$0280	
E147 F009		BEQ TMOX	
E149 A910		LDA #\$10	Motor On Status setzen
E14B 0410		TSB DRSTAT	
E14D A205		LDX #5	
E14F 20FBE2		JSR X2WAIT	Verzoegerungsschleife, dem Motor Zeit geben,
E152 68	TMOX	PLA	auf Touren zu kommen
E153 A200		LDX #0	OK-Status
E155 60	TMOEX	RTS	
;			
E156 203CE1	TSTDEN	JSR TSTMON	* Density von Diskette feststellen *
E159 A92C		LDA #\$2C	
E15B 8517		STA STPTIM	Zeit fuer Stepwechsel kurz setzen
E15D 20D1E2		JSR TRACK0	Kopf auf Track 0 positionieren
E160 640D		STZ TRACK	
E162 A000	TSTD0	LDY #0	OK-Kennung
E164 2C0004	TSTD1	BIT \$0400	
E167 3035		BMI TDERR	
E169 5A		PHY	Status retten
E16A 202FE3		JSR TRADJA	Kopf positionieren
E16D 7A		PLY	
E16E A920		LDA #\$20	
E170 1C8002		TRB \$0280	Set MFM
E173 2030EB	TSDER	JSR RDHDV	Sector lesen
E176 B00E		BCS SETFM	Lesefehler
E178 A941		LDA #\$41	Medium Density - Status
E17A A67D		LDX #7D	Sector Laenge
E17C 30F5		BMI TSDER	Daten ungueltig
E17E F002		BEQ SMFMF	
E180 A900		LDA #0	Double Density - Status
E182 8509	SMFMF	STA FORKEN	
E184 801D		BRA TSTDEX	
E186 A920	SETFM	LDA #\$20	
E188 0C8002		TSB \$0280	Set FM
E18B A982		LDA #\$82	
E18D 8509		STA FORKEN	
E18F 2030EB	RDHSD	JSR RDHDV	Sector Header lesen (SD)
E192 900F		BCC TSTDEX	OK ?
E194 A50D		LDA TRACK	
E196 C903		CMP #3	Density bis Track 3 suchen
E198 F004		BEQ TDERR	
E19A E60D		INC TRACK	Track # 1
E19C 80C6		BRA TSTD1	
E19E 202DE2	TDERR	JSR RSFE80	Drive 'Dumm'-Status setzen
E1A1 A080		LDY #\$80	Error-Kennung
E1A3 5A	TSTDEX	PHY	
E1A4 20A9E1		JSR SDRDDP	Drive Density und Read/Write-Laenge einstellen
E1A7 68		PLA	

```

E1A8 60          RTS
;
E1A9 201FF0  SDRDDP  JSR DENDSP          Density auf Display anzeigen
E1AC A920    SETDRD  LDA #$20            DD
E1AE 1C8002          TRB $0280          MFM
E1B1 A212          LDX #18            18 Sektoren/Track
E1B3 A409          LDY FORKEN
E1B5 F00F          BEQ SDRD
E1B7 3006          BMI SSD            SD
E1B9 A980          LDA #$80            MD
E1BB A21A          LDX #26            26 Sektoren/Track
E1BD 8005          BRA SDDL80
E1BF 0C8002  SSD     TSB $0280          Set FM
E1C2 A900          LDA #0
E1C4 A080    SDDL80  LDY #$80            128 Bytes/Sector
E1C6 8510    SDRD     STA DRSTAT        Drive-Status setzen
E1C8 861F          STX SECANZ        Sektoren/Track setzen
E1CA 8414          STY SECLEN        Bytes/Sector setzen
E1CC 60          RTS
;
Sectorfolge auf aktuellem Track lesen
;
E1CD A202    RDSFOL  LDX #2
E1CF 20FBE2          JSR X2WAIT          warten bis Klappe vollstaendig geschlossen ist
E1D2 203DEB  RDSFD1  JSR RDHEAD
E1D5 B056          BCS RSFE80
E1D7 A51F          LDA SECANZ          Sektoren/Track
E1D9 1A          INA
E1DA 8500          STA MERK1
E1DC 641E          STZ SECANZ1
E1DE A9CF          LDA #$CF
E1E0 8D9F02          STA $029F          Timer setzen
E1E3 2042EB  RDSFCL  JSR RDHD1
E1E6 B027          BCS RDSFT
E1E8 A57A          LDA $7A
E1EA 30F7          BMI RDSFCL
E1EC C50D          CMP TRACK          Track Nummer ueberpruefen
E1EE D03D          BNE RSFE80
E1F0 A57C          LDA $7C          SECTOR Nummer
E1F2 F0EF          BEQ RDSFCL          ungueltig
E1F4 30ED          BMI RDSFCL          ungueltig
E1F6 C500          CMP MERK1          )SECANZ ?
E1F8 B033          BCS RSFE80
E1FA A61E          LDX SECANZ1
E1FC 9520          STA SECLST,X          Sector in Sectorliste eintragen
E1FE F00B          BEQ RSFI
E200 A200          LDX #0
E202 D520    RSFCL  CMP SECLST,X          Sector schon in Sector-Liste ?
E204 F027          BEQ RSFE80          ja - Duumm schalten
E206 EB          INX
E207 E41E          CPX SECANZ1
E209 90F7          BCC RSFCL

```

```

E20B E61E  RSFI      INC  SECANZ1
E20D D0D4                BNE  RDSFL
E20F A51E  RDSFT    LDA  SECANZ1      gefundene Sectoranzahl=vorgegebene Sectoranzahl ?
E211 C51F                CMP  SECANZ
E213 D018                BNE  RSFE80      nein - 'Dumm' schalten
E215 A508  RDSFOK   LDA  DUMKEN      Status 'Sector-Folge OK'
E217 297F                AND  #$7F
E219 B508  SETDK    STA  DUMKEN      Drive dennoch Dumm schalten ?
E21B 2938                AND  #$38
E21D D015                BNE  DSPD2
E21F A9FF  SNEWTR   LDA  #$FF
E221 850C                STA  LTRACK
E223 6407                STZ  EXSECT
E225 6406                STZ  WRKEN
E227 A92C                LDA  #$2C      Steppmotor Verzögerungswert kurz stellen
E229 8517                STA  STPTIM
E22B 18                CLC
E22C 60                RTS
E22D A980  RSFE80   LDA  #$80      Status 'Sector-Folge ERROR'
E22F 0408                TSB  DUMKEN
E231 20B2EF DSPDUM   JSR  BELL1
E234 A96D  DSPD2   LDA  #$6D      'SL' - Anzeige auf Display
E236 8D0140                STA  DISP10
E239 A938                LDA  #$38
E23B 8D0040                STA  DISP1
E23E A964                LDA  #$64      Steppmotor Verzögerung normal
E240 8517                STA  STPTIM
E242 38                SEC
E243 60                RTS
;
E244 20E4E9 RDTRAV   JSR  CALCTS      * Aktuellen Track ins RAM einlesen - mit Verify *
E247 F024                BEQ  RDTRVE      Sector # = 0
E249 B022                BCS  RDTRVE      Sector # grösser als zulaessig
E24B 206FE2                JSR  RDTRA      alle Sektoren vom Track lesen
E24E B01D                BCS  RDTRVE
E250 A51F  RDTRV2   LDA  SECANZ
E252 8500                STA  MERK1
E254 E8                RDTRVL   INX
E255 E41F                CPX  SECANZ
E257 9002                BCC  RDTRT
E259 A200                LDX  #0
E25B B420  RDTRT    LDY  SECLST,X
E25D B94000                LDA  STALST,Y
E260 F005                BEQ  RDTRD      Status OK
E262 20B6E2                JSR  RDSS2      Sector noch einmal lesen
E265 D006                BNE  RDTRVE
E267 C600  RDTRD   DEC  MERK1
E269 D0E9                BNE  RDTRVL
E26B 18                CLC      'Carry' = 0 OK-Status
E26C 60                RTS
E26D 38  RDTRVE   SEC      'Carry' = 1 Error-Status
E26E 60                RTS

```

```

;
E26F A940  RDTRA  LDA #$40          * Aktuellen Track ins RAM einlesen - ohne Verify *
E271 0408          TSB DUMKEN
E273 201DEB       JSR RDHDSP       Sector-Track-Position feststellen
E276 B02F          BCS RDTRAX
E278 A67A          LDX $7A
E27A E40D          CPX TRACK       Track # in Ordnung ?
E27C D029          BNE RDTRAX
E27E A200          LDX #0
E280 D520  RDTRSL  CMP SECLST,X       Sector in Liste suchen
E282 F007          BEQ RDTRA2
E284 E8            INX
E285 E41F          CPX SECANZ
E287 90F7          BCC RDTRSL
E289 801C          BRA RDTRAX

;
E28B A51F  RDTRA2  LDA SECANZ       Anzahl Sektoren/Track
E28D 8500          STA MERK1
E28F 201FE2       JSR SNEWTR       Kennung fuer neuen Track setzen
E292 20AFE2  RDTRIL  JSR RDSSPE
E295 2916          AND #$16         CRC/AM-ERR zulassen
E297 D00E          BNE RDTRAX
E299 C600  RDTRA3  DEC MERK1
E29B D0F5          BNE RDTRIL

;
E29D A940          LDA #$40          Track-Read Error-Status ruecksetzen
E29F 1408          TRB DUMKEN
E2A1 A50D          LDA TRACK
E2A3 850C          STA LTRACK       Track # fuer zuletzt gelesenen Track merken
E2A5 18            CLC
E2A6 60            RTS
E2A7 207BE3  RDTRAX  JSR CONRES       Controller zuruecksetzen
E2AA 2C0004       BIT $0400       Klappe auf ?
E2AD 38            SEC
E2AE 60            RTS

;
E2AF E8  RDSSPE  INX          * Unterprogramm zum lesen der Sektoren nach *
E2B0 E41F          CPX SECANZ       * Sector-Liste *
E2B2 9002          BCC RDSS2
E2B4 A200          LDX #0
E2B6 B520  RDSS2  LDA SECLST,X
E2B8 8D0204       STA $0402       Sector #
E2BB 201CEA       JSR SETBUF2     Buffer fuer Sector setzen
E2BE 207BE3       JSR CONRES
E2C1 20D4EA       JSR RDSEC1     Sector lesen
E2C4 B420          LDY SECLST,X   Sector-Status in Statusliste eintragen
E2C6 293F          AND #$3F
E2C8 994000       STA STALST,Y
E2CB 60            RTS

;
E2CC A904  TROJUS  LDA #4          * Track 0 Justierung *
E2CE 2045E3       JSR TRADJ1     4 Tracks vorwaerts

```



```

E2D1 207BE3 TRACK0 JSR CONRE2
E2D4 A0FF LDY ##FF Step-Rueckwaerts-Kennung
E2D6 AD0004 SENSOR LDA $0400
E2D9 2904 AND #4 Track-0 Sensor pruefen
E2DB F005 BEQ B5TST
E2DD 200FE3 JSR TRVR 1 Track zurueck
E2E0 80F4 BRA SENSOR
E2E2 A560 B5TST LDA STPPOS Track 0 nach Stepperposition fein justieren
E2E4 C903 CMP #3
E2E6 F005 BEQ SETTRO
E2E8 200FE3 JSR TRVR
E2EB D0F5 BNE B5TST
E2ED 9C0104 SETTRO STZ $0401 Track-Register des Controllers = 0
E2FO A616 LDX DLYTIM
;
E2F2 A912 XWAIT LDA #18 * Verzoegerungsschleife *
E2F4 3A XWA1 DEB * Wert der Verzoegerung im X-Register *
E2F5 D0FD BNE XWA1 * 1 X-Wert ca. 100 Taktzyklen *
E2F7 CA DEX
E2F8 D0FB BNE XWAIT
E2FA 60 RTS
;
E2FB 8604 X2WAIT STX DLYT2 * Schleife fuer lange Verzoegerungen *
E2FD A904 X2WA1 LDA #4 * 1 X-Wert ca. 100000 Taktzyklen *
E2FF 8503 STA DLYT1
E301 A2FA X2WA2 LDX ##FA
E303 20F2E2 JSR XWAIT
E306 C603 DEC DLYT1
E308 D0F7 BNE X2WA2
E30A C604 DEC DLYT2
E30C D0EF BNE X2WA1
E30E 60 RTS
;
E30F DA TRVR PHX * Trackwechsel Routine *
E310 A660 LDX STPPOS
E312 E8 INX * Y=00-7F 1 Step vorwaerts *
E313 98 TYA * Y=80-FF 1 Step rueckwaerts *
E314 3002 BMI RWARTS
E316 CA DEX
E317 CA DEX
E318 8A RWARTS TXA
E319 2903 AND #3
E31B 8560 STA STPPOS Bitposition des Steppermotors merken
E31D AA TAX
E31E BD2BE3 LDA SMDAT,X Bitmuster setzen
E321 8D8202 STA $0282
E324 A617 LDX STPTIM Wert fuer Step-Verzoegerung
E326 20F2E2 JSR XWAIT
E329 FA PLX
E32A 60 RTN1 RTS
E32B 39352D1D SMDAT .BYTE $39,$35,$2D,$1D
;

```

E32F 20F6EF	TRADJA	JSR TRAAZ	Track-Justierung mit Trackanzeige
E332 207BE3	TRADJ	JSR CONRES	-- ohne Trackanzeige
E335 203CE1		JSR TSTMON	Motor einschalten wenn Klappe geschlossen ist
E338 301C		BMI JPE80	Klappe war auf
E33A A50D		LDA TRACK	neue Track #
E33C 38	TRADJO	SEC	
E33D ED0104		SBC \$0401	Track-Register Controller
E340 F0EB		BEQ RTN1	neuer Track=alter Track
E342 20F6EF		JSR TRAAZ	bei Trackwechsel - Trackanzeige
E345 A8	TRADJ1	TAY	
E346 1003		BPL TRADJ2	
E348 49FF		EOR #\$FF	
E34A 1A		INA	
E34B 0A	TRADJ2	ASL A	Steps x2
E34C C950		CMP #80	
E34E 9009		BCC TRADJ3	mehr als 40 Tracks nicht zulassen
E350 2068E0		JSR SYSERR	2 x Bell ausgeben
E353 20D1E2		JSR TRACK0	Track 0 Justage
E356 4C50E5	JPE80	JMP STELL2	=ERR80
E359 AA	TRADJ3	TAX	
E35A 200FE3	TRADJL	JSR TRVR	1 Step ausfuehren
E35D CA		DEX	
E35E D0FA		BNE TRADJL	
E360 A60D	SETTRN	LDX TRACK	Track # in Trackregister Controller kopieren
E362 8E0104		STX \$0401	
E365 A910		LDA #\$10	
E367 E014		CPX #20	
E369 9005		BCC TRKL20	Track groesser 20
E36B 0C8002		TSB \$0280	Controller umschalten
E36E 8003		BRA TRADJX	
E370 1C8002	TRKL20	TRB \$0280	
E373 A228	TRADJX	LDX #\$28	Kurze Verzoeegerung
E375 4CF2E2		JMP XWAIT	
;			
E378 207BE3	CONRE2	JSR CONRES	
E37B A9D0	CONRES	LDA #\$D0	Controller Reset
E37D 8D0004		STA \$0400	
E380 A907		LDA #7	
E382 3A	CONRL	DEA	
E383 D0FD		BNE CONRL	
;			
E385 A901	WREADY	LDA #1	Warten auf Controller
E387 2D0004	WRDYL	AND \$0400	
E38A D0FB		BNE WRDYL	
E38C AD0004		LDA \$0400	Controller-Status
E38F 60		RTS	
;			
E390 6411	RDINF	STZ COMST	
E392 A904		LDA #4	4 Bytes nach Buffer \$80
E394 A280		LDX #\$80	
E396 A000		LDY #0	
E398 20BCE4		JSR RDBTS	Kommando vom Computer empfangen

```

E39B AD8202  CWAIT  LDA $02B2      Warten bis Command - Leitung zurueckgesetzt ist
E39E 30FB      BMI  CWAIT
E3A0 A204      LDX  #4
E3A2 20F2E2   JSR  XWAIT
;
E3A5 2411      BIT  COMST      Fehler bei Datenuebertragung ?
E3A7 702F      BVS  DELINF
E3A9 20F7E3   JSR  TSTCOM     Kommando auf Gueltigkeit ueberpruefen
E3AC 2411      BIT  COMST
E3AE 3028      BMI  DELINF
E3B0 7031      BVS  ERR4E1
E3B2 20FBE4   JSR  SEND41
E3B5 241C      BIT  RDDATK     Flag fuer Datenblock lesen
E3B7 100A      BPL  EXECCO
E3B9 20C2E4   JSR  RDBYTS
E3BC 2411      BIT  COMST
E3BE 7027      BVS  ERR4E2
E3C0 20FBE4   JSR  SEND41     'A' senden
E3C3 20A5E5   EXECCO JSR  EXEC       Kommando ausfuehren
E3C6 207BE3   JSR  CONRES
E3C9 A903      LDA  #3
E3CB 1410      TRB  DRSTAT     BIT 0+1=0
E3CD A93C      LDA  #$3C
E3CF 0C8202   TSB  $02B2     Steppmotor aus
;
E3D2 A516      SDELAY LDA  DLYTIM     'Motor aus' Zeit setzen
E3D4 8504      STA  DLYT2
E3D6 6403      STZ  DLYT1
;
E3D8 6480      DELINF STZ  $80       Kommandobuffer loeschen
E3DA 6481      STZ  $81
E3DC 6482      STZ  $82
E3DE 6483      STZ  $83
E3E0 6C829F   JMP  (CMTBL+$82) Ruecksprungvector in Motor-Timer Routine
;
E3E3 A901      ERR4E1 LDA  #1
E3E5 8002      BRA  ERR4E
E3E7 A902      ERR4E2 LDA  #2
E3E9 0410      ERR4E TSB  DRSTAT
E3EB 2007E5   JSR  SEND4E     'N' senden
E3EE 80E8      BRA  DELINF
;
E3F0 33323431 DRKEN  .BYTE '3','2','4','1'  Laufwerksnummer Tabelle
;
E3F4 4C50E5   ERR80  JMP  STELL2
;
E3F7 AD8002   TSTCOM LDA  $02B0     * Kommando vom Computer auf Gueltigkeit pruefen *
E3FA 2903      AND  #3
E3FC AA       TAX
E3FD BDF0E3   LDA  DRKEN, X   Drive # OK ?
E400 C5B0      CMP  $80
E402 D0F0      BNE  ERR80

```

```

E404 A200          LDX #0
E406 BD009F VERCOM LDA CMTBL,X
E409 F075          BEQ ERR40      Kommando in COM-Tabelle suchen
E40B 0A           ASL A          Ende COM-Tabelle
E40C 661C          ROR RDDATK
E40E 4A           LSR A
E40F C5B1          CMP #81
E411 F011          BEQ COMFND      Kommando gefunden ?
E413 E8           INX
E414 E8           INX
E415 E8           INX
E416 3068          BMI ERR40
E418 A508          LDA DUMKEN
E41A 2938          AND #38         Kommando aus erweiterter Kommandotabelle zulassen ?
E41C F0E8          BEQ VERCOM
E41E E016          CPX #16
E420 90E4          BCC VERCOM
E422 805C          BRA ERR40
;
E424 B618 COMFND  STX COMPOS
E426 E009          CPX #9          Kommando Position (3
E428 B04A          BCS TSTCOX
E42A 20E4E9        JSR CALCTS      Track+Sector errechnen
E42D B051          BCS ERR40      Sector unzulessig
E42F 3043          BMI TSTCOX     RAM oder ROM-Adresse
E431 D006          BNE COMF2
E433 A51C          LDA RDDATK
E435 3049          BMI ERR40
E437 803B          BRA TSTCOX
E439 A508 COMF2  LDA DUMKEN
E43B 29B8          AND #B8
E43D D022          BNE WSEBUF
E43F 201AEA        JSR SETBUF
E442 A618 NOSEC   LDX COMPOS      RAM-Buffer nach Sector # setzen
E444 F00B          BEQ WRSTD      Kommando = Read/Write Sector ?
E446 E003          CPX #3
E448 F017          BEQ WSEBUF     Write Sector verify ?
E44A A508          LDA DUMKEN
E44C 4A           LSR A          Read Sector 'Dumm' geschaltet ?
E44D 9025          BCC TSTCOX
E44F B010          BCS WSEBUF
E451 A508 WRSTD  LDA DUMKEN
E453 29FA          AND #FA        Write Sector 'Dumm' geschaltet ?
E455 D00A          BNE WSEBUF
E457 A60E          LDX SECTOR
E459 A50D          LDA TRACK
E45B C50B          CMP LWRTRA     neuer Track=letzter Track ?
E45D F007          BEQ TSTRS
E45F 8607          STX EXSECT    Sector # fuer Extended-Buffer merken
E461 20DBE9 WSEBUF JSR SEXBUF
E464 800E          BRA TSTCOX
E466 A506 TSTRS  LDA WRKEN

```

Extended Buffer als Sector-Buffer

schon ein Sector auf diesem Track geschrieben (ins RAM) ?

```

E468 F004          BEQ TSTRS2
E46A B540          LDA STALST,X          Sector schon einmal geschrieben ?
E46C 3006          BMI TSTCOX
E46E A980    TSTRS2  LDA ##80          Write Status setzen
E470 9540          STA STALST,X
E472 E606          INC WRKEN          zu schreibende Sektoren+1
E474 A900    TSTCOX  LDA #0
E476 241C          BIT RDDATK
E478 1001          BPL RTN2
E47A 1A           INA
E47B 0411    RTN2   TSB COMST
E47D 60          RTS
;
E47E E615    ER40UK  INC USKEN          Umschalten zwischen normal oder High-Speed
E480 AD9602  ERR40   LDA $0296
E483 A940          LDA ##40
E485 0411          TSB COMST
E487 60          RTS
;
Test, ob Klappe geschlossen und Write-Protect
;
E488 207BE3  TSTWRP  JSR CONRES
E48B 29C0          AND ##C0
E48D 60          RTS
;
E48E A607    TSTMEB  LDX EXSECT          Zu schreibender Sector im Extended-Buffer ?
E490 F020          BEQ TMEBX
E492 A980          LDA ##80
E494 9540          STA STALST,X          Write-Status setzen
E496 20F6EF       JSR TRAAZ          Trackanzeige
E499 201AEA       JSR SETBUF          RAM Buffer setzen
E49C A000          LDY #0
E49E B9009E  MEBL   LDA EXBUF,Y          Sektordaten in Sectorbuffer kopieren
E4A1 9119          STA (IND),Y
E4A3 C8           INY
E4A4 C413          CPY RWLEN
E4A6 D0F6          BNE MEBL
E4A8 6407          STZ EXSECT
E4AA A901          LDA #1
E4AC 8506          STA WRKEN          Anzahl der zu schreibenden Sektoren = 1
E4AE A50D          LDA TRACK          Track # fuer zu schreibenden Sector merken
E4B0 850B          STA LWRTRA
E4B2 60          TMEBX  RTS
;
;
E4B3 A980    RD128B  LDA ##80          128 Bytes
E4B5 2C          .BYTE $2C          =BIT ABS. (Dummy)
E4B6 A900    RD256B  LDA #0          256 Bytes empfangen
E4B8 A200          LDX # (EXBUF
E4BA A09E          LDY # )EXBUF
E4BC 8513    RDBTS  STA RWLEN
E4BE B619          STX IND

```

```

E4C0 841A          STY IND+1
;
E4C2 641B  RDBYTS  STZ CHKSUM          Checksumme loeschen
E4C4 A990  RD1BLK  LDA #$90             maximale Zeit fuer Datenuebertragung festlegen
E4C6 8D9F02     STA $029F
E4C9 A000          LDY #0
E4CB 20F0E4  RDBL    JSR RDBYTE          1 Byte vom Computer empfangen
E4CE 9119          STA (IND),Y
E4D0 18          CLC
E4D1 651B          ADC CHKSUM
E4D3 6900          ADC #0
E4D5 851B          STA CHKSUM
E4D7 C8          INY
E4D8 C413          CPY RWLEN          letztes Byte ?
E4DA D0EF          BNE RDBL
E4DC 2041EA     JSR ADDBUF          mehrere Datenblocks lesen (z.B. COM 60) ?
E4DF D0E3          BNE RD1BLK
E4E1 20F0E4     JSR RDBYTE
E4E4 AC9602  RDEXIT  LDY $0296
E4E7 451B          EOR CHKSUM          Checksumme OK ?
E4E9 D093          BNE ER40UK
E4EB 6411          STZ COMST          COM-Status 'Datenuebertragung Ok'
E4ED 4C2CEA     JMP SETRWL
;
E4F0 A663  RDBYTE  LDX IOIND          1 Byte lesen
E4F2 7CF5E4     JMP (RDIND,X)
E4F5 12FE  RDIND   .WORD NORDB,USRDB,RDPID  Tabelle I/O - Routinen
                                   (RDPID fuer spaetere Erweiterung vorgesehen)
;

E4F7 3BFE
E4F9 00FD
;
E4FB A941  SEND41  LDA #$41          'A' Status-Rueckmeldung an den Computer
E4FD D00A          BNE SENDW
E4FF A943  SEND43  LDA #$43          'C'
E501 D006          BNE SENDW
E503 A945  SEND45  LDA #$45          'E'
E505 D002          BNE SENDW
E507 A94E  SEND4E  LDA #$4E          'N'
E509 8500  SENDW   STA MERK1
E50B A202          LDX #2
E50D 20F2E2     JSR XWAIT
E510 802F          BRA SDBYT2
;
E512 A980  SD128B  LDA #$80          128 Bytes
E514 2C          .BYTE $2C          =BIT ABS.
E515 A900  SD256B  LDA #0           256 Bytes senden
E517 A200          LDX # (EXBUF
E519 A09E          LDY # )EXBUF
E51B 8513  SDBTS   STA RWLEN
E51D 8619          STX IND

```

```

E51F 841A          STY IND+1
;
Datenblocks zum Computer senden
;
E521 641B  SDBYTS  STZ  CHKSUM          Checksum loeschen
E523 A000  SD1BLK  LDY  #0
E525 B119  SDBL    LDA  (IND),Y
E527 8500          STA  MERK1
E529 18          CLC
E52A 651B          ADC  CHKSUM
E52C 6900          ADC  #0
E52E 851B          STA  CHKSUM
E530 2041E5       JSR  SDBYT2
E533 C8          INY
E534 C413          CPY  RWLEN
E536 D0ED          BNE  SDBL
E538 2041EA       JSR  ADDBUF
E53B D0E6          BNE  SD1BLK
E53D A51B          LDA  CHKSUM          Cheksumme senden
;
E53F 8500  SDBYTE  STA  MERK1
E541 A663  SDBYT2  LDX  IOIND          1 Byte senden
E543 7C46E5       JMP  (SDIND,X)
E546 76FE  SDIND   .WORD NOSDB,USSDB,SDPIO  Tabelle der I/O - Routinen
;                               (SDPIO = fuer spaetere Erweiterung vorgesehen)
E548 A5FE
E54A 27FD
;
E54C A902  STELL   LDA  #2          2 Versuche setzen
E54E 8512          STA  RETRY
E850 A980  STELL2  LDA  #80          Command-Error setzen
E552 0411          TSB  COMST
E554 60          RTS
;
E555 AD0004  QUITT   LDA  $0400          * Quittungsbyte an Computer senden *
E558 850F  QUITT2  STA  CONST
E55A 2411          BIT  COMST
E55C 3007          BMI  SERR45
E55E A944          LDA  #44
E560 1410          TRB  DRSTAT
E562 4CFFE4       JMP  SEND43          'C' Senden
E565 20DBEF  SERR45  JSR  ERRDSP
E568 20ADEF       JSR  BELL
E56B A904          LDA  #4
E56D 0410          TSB  DRSTAT
E56F 4C03E5       JMP  SEND45          'E' Senden
;
E572 8D9602  SETTIM  STA  $0296          Set Timer Routine
E575 8D9F02          STA  $029F
E578 60          RTS
;
E579 AD8002  TSTDSW  LDA  $0280          'Dumm' Schalter abfragen

```

```

E57C 2904      AND #4
E57E C505      CMP LDSW
E580 F022      BEQ TSTD SX           gleiche Stellung wie vorher
E582 8505      STA LDSW
E584 A8        TAY
E585 D00A      BNE NODSW
E587 A908      LDA #8               Dumm - Modus setzen
E589 0408      TSB DUMKEN
E58B 2034E2    JSR DSPD2           'SL' anzeigen
E58E 4C04E6    JMP TSTDAT          Testen ob noch Sektoren zu schreiben sind
E591 A508      NODSW LDA DUMKEN        Dumm - Modus zuruecksetzen
E593 29F7      AND #$F7
E595 8508      STA DUMKEN
E597 29B0      AND #$80
E599 D006      BNE NOFAST
E59B 201FE2    JSR SNEWTR          Kennung fuer 'kein Sector im RAM' setzen
E59E 4CF6EF    JMP TRANZ          Track # neu anzeigen
ESA1 4C34E2    NOFAST JMP DSPD2
ESA4 60        TSTD SX RTS
;
ESA5 A618      EXEC   LDX COMPOS   Kommando ausfuehren
ESA7 7C019F    JMP (CMTBL+1,X)
;

```

Normale Kommando - Tabelle:

```

E5AA D0      COMTBL .BYTE $D0      Write Sector
E5AB 00EC    .WORD COM50
E5AD D7      .BYTE $D7      Write Sector+Verify
E5AE 00EC    .WORD COM50
E5B0 52      .BYTE $52      Read Sector
E5B1 56EA    .WORD COM52
E5B3 53      .BYTE $53      Drive Status
E5B4 80EF    .WORD COM53
E5B6 21      .BYTE $21      Format Single/Double
E5B7 E7EC    .WORD COM21
E5B9 22      .BYTE $22      Format Medium
E5BA E3EC    .WORD COM22
E5BC 4E      .BYTE $4E      Read Drive - Options
E5BD 28EF    .WORD COM4E
E5BF 4F      .BYTE $4F      Write Drive - Options
ESC0 5AEF    .WORD COM4F
;

```

Erweiterte Kommando-Tabelle:

```

;
ESC2 3F      .BYTE $3F      Read High-Speed-Wert
ESC3 00FE    .WORD COM3F
ESC5 44      .BYTE $44      Display/Bell/Drive Control
ESC6 E7E5    .WORD COM44
ESC8 4C      .BYTE $4C      Jump Adresse
ESC9 14E6    .WORD COM4C
ESCB 4D      .BYTE $4D      Jump/Quitt
ESCC 11E6    .WORD COM4D

```



```

E5CE 51          .BYTE $51          Write all + Stop Motor
E5CF 17E6       .WORD COM51
E5D1 4B         .BYTE $4B          Set/Reset 'Dumm'
E5D2 EEE5       .WORD COM4B
E5D4 60         .BYTE $60          Write Track (normal Speed)
E5D5 4AE6       .WORD COM60
E5D7 62         .BYTE $62          Read Track
E5D8 26E6       .WORD COM62
E5DA 68         .BYTE $68          SIO-Laenge senden
E5DB 9FE6       .WORD COM68
E5DD 69         .BYTE $69          SIO Routine senden
E5DE ADE6       .WORD COM69
E5E0 41         .BYTE $41          Kommando einfuegen/loeschen
E5E1 16E7       .WORD COM41
E5E3 20         .BYTE $20          Spezial Format
E5E4 7EE7       .WORD COM20
E5E6 00        COMEND .BYTE 0
;
Bedeutung der Bits bei COM 44

$80 : Error Anzeige zulassen
$40 : Trackanzeige in Hexa-Dezimal
$20 : Format ohne Verify
$10 : bei COM 20 Sektoren 1,2,3,360,1024 nicht schreiben
$08 : bei COM 51 Motor anlassen
$01 : Bell bei Error zulassen

E5E7 A582      COM44   LDA $82          * Display/Drive Kontrolle neu setzen *
E5E9 8561      STA DSPCTR
E5EB 4CFFE4    JMP SEND43
;
E5EE 2004E6   COM4B   JSR TSTDAT      * Slow/Fast - Mode Kontrolle *
E5F1 A582      LDA $82
E5F3 8508      STA DUMKEN
E5F5 29B8      AND #$88
E5F7 F005      BEQ C4BHI
E5F9 2031E2    JSR DSPDUM      'SL' anzeigen
E5FC 8003      BRA COM4BX
E5FE 2068ED   C4BHI   JSR COPSLT
E601 4CFFE4   COM4BX  JMP SEND43      'C' Senden
;
E604 2064EB   TSTDAT  JSR TSTWR      Noch zu schreibende Sektoren schreiben
E607 A61F      LDX SECANZ
E609 A940      LDA #$40
E60B 9540     TSTDLS  STA STALST,X   Status 'kein Sector im RAM' setzen
E60D CA        DEX
E60E D0FB      BNE TSTDLS
E610 60        RTS
;
E611 2055E5   COM4D   JSR QUITT
E614 6C8200   COM4C   JMP ($82)      Sprung ueber 'Jump Adresse'
;

```

E617 2064EB	COM51	JSR TSTWR	noch zu schreibende Sektoren schreiben
E61A A561		LDA DSPCTR	
E61C 2908		AND #8	Motor ausschalten ?
E61E D003		BNE C51Q	
E620 202DE1		JSR MTOFF	
E623 4CFFE4	C51Q	JMP SEND43	'C' senden
;			
E626 A583	COM62	LDA #83	RAM - oder ROM - Adresse ?
E628 300A		BMI C62X	
E62A 2050E5		JSR STELL2	Error - Status setzen
E62D 2044E2		JSR RDTRAV	Read Track mit Verify
E630 B002		BCS C62X	
E632 6411	C62DK	STZ COMST	OK Status senden
E634 2055E5	C62X	JSR QUITT	Quittung senden
E637 A51F		LDA SECANZ	
E639 8562		STA BLOCKS	Anzahl der Datenblocks setzen
E63B A514		LDA SECLN	
E63D A682		LDX #82	
E63F A483		LDY #83	RAM - oder ROM - Adresse ?
E641 3004		BMI C62SD	
E643 A200		LDX # (DATBUF	
E645 A08C		LDY #)DATBUF	
E647 4C1BE5	C62SD	JMP SDBTS	alle Datenblocks senden
;			
E64A 2050E5	COM60	JSR STELL2	Error Status setzen
E64D 20E4E9		JSR CALCTS	
E650 F047		BEQ C60X	Sector 0 nicht zulassen
E652 B045		BCS C60X	Track)39
E654 300D		BMI C60RD	Daten ins RAM
E656 A50E		LDA SECTOR	
E658 3A		DEA	
E659 D03E		BNE C60X	
E65B A900		LDA # (DATBUF	
E65D 8519		STA IND	
E65F A98C		LDA #)DATBUF	
E661 851A		STA IND+1	
E663 A51F	C60RD	LDA SECANZ	
E665 A614		LDX SECLN	
E667 F001		BEQ C60RD2	
E669 4A		LSR A	
E66A 8562	C60RD2	STA BLOCKS	Anzahl der Datenblocks setzen
E66C 6413		STZ RWLEN	
E66E 20C2E4		JSR RDBYTS	alle Datenblocks lesen
E671 2411		BIT COMST	Fehler in Datenuebertragung ?
E673 7027		BVS C60E4E	
E675 20FBE4		JSR SEND41	'A' Senden
E678 A583		LDA #83	
E67A 301B		BMI C600K	
E67C A50D		LDA TRACK	Track # uebernehmen
E67E 850B		STA LWRTRA	
E680 208BE4		JSR TSTWRP	Write Protect oder Klappe auf ?
E683 D014		BNE C60X	

```

E685 A61F          LDX SECANZ
E687 B606          STX WRKEN
E689 A980          LDA #$80          Write Status fuer alle Sektoren setzen
E68B 9540   C60L   STA STALST,X
E68D CA           DEX
E68E D0FB          BNE C60L
E690 2064EB        JSR TSTWR          den ganzen Track schreiben
E693 A501          LDA MERK2
E695 D002          BNE C60X
E697 6411   C600K  STZ COMST          OK Status setzen
E699 4C55E5   C60X  JMP QUITT          Quittung senden
E69C 4C07E5   C60E4E JMP SEND4E          'N' Senden
;
E69F 20FFE4   CDM68 JSR SEND43          * Diese Routine gibt die Laenge der SIO-Routine *
E6A2 A902          LDA #2          * an den Computer zurueck *
E6A4 A2AB          LDX # (SIOLEN
E6A6 A0E6          LDY # )SIOLEN
E6A8 4C1BE5        JMP SDBTS
E6AB 1602   SIOLEN .WORD SIOEND-SIO
;
E6AD A953   CDM69  LDA # (SIO          * Routine zum Senden der kompletten SIO-Routine an den *
E6AF 8519          STA IND          * Computer *
E6B1 38           SEC
E6B2 E582          SBC #82
E6B4 8582          STA #82
E6B6 A9F0          LDA # )SIO
E6B8 851A          STA IND+1
E6BA E583          SBC #83
E6BC 8583          STA #83
E6BE 20FFE4        JSR SEND43
E6C1 A000          LDY #0
E6C3 841B          STY CHKSUM
E6C5 B969F2   C69L  LDA ABSTBL,Y          eine zu relocierende Adresse ?
E6C8 C519          CMP IND
E6CA D01E          BNE C69SB
E6CC B96AF2        LDA ABSTBL+1,Y
E6CF C51A          CMP IND+1
E6D1 D017          BNE C69SB
E6D3 2001E7        JSR C69LDB
E6D6 38           SEC          absolute Adresse relocieren
E6D7 E582          SBC #82
E6D9 08           PHP
E6DA 200AE7        JSR C69SDB
E6DD 2001E7        JSR C69LDB
E6E0 28           PLP
E6E1 E583          SBC #83
E6E3 200AE7        JSR C69SDB
E6E6 C8           INY
E6E7 C8           INY
E6E8 D006          BNE C69TE
E6EA 2001E7   C69SB JSR C69LDB
E6ED 200AE7        JSR C69SDB

```

```

E6F0 A519 C69TE LDA IND
E6F2 C969 CMP # (SIOEND Ende der SIO-Routine
E6F4 D0CF BNE C69L
E6F6 A51A LDA IND+1
E6F8 C9F2 CMP # )SIOEND
E6FA D0C9 BNE C69L
E6FC A51B LDA CHKSUM
E6FE 4C3FE5 JMP SDBYTE
E701 B219 C69LDB LDA (IND) naechstes Byte der SIO-Routine empfangen
E703 E619 INC IND
E705 D002 BNE C69LDX
E707 E61A INC IND+1
E709 60 C69LDX RTS
E70A 8500 C69SDB STA MERK1 1 Byte zum Computer senden
E70C 18 CLC
E70D 651B ADC CHKSUM
E70F 6900 ADC #0
E711 851B STA CHKSUM
E713 4C41E5 JMP SDBYT2
;
E716 A903 COM41 LDA #3 3 Bytes in den Extended-Buffer holen
E718 A200 LDX # (EXBUF
E71A A09E LDY # )EXBUF
E71C 20BCE4 JSR RDBTS
E71F 2411 BIT COMST Fehler bei Datenuebertragung ?
E721 7058 BVS C41E4E
E723 20FBE4 JSR SEND41 'A' senden
E726 2050E5 JSR STELL2
E729 AD009E LDA EXBUF
E72C 297F AND #$7F
E72E 8500 STA MERK1 Kommando ohne Bit 7 merken
E730 A200 LDX #0
E732 BD009F C41SL LDA CMTBL,X Testen ob das Kommando schon in der Kommando-Tabelle ist
E735 08 PHP
E736 F026 BEQ C41AC2 Tabelleneende!
E738 297F AND #$7F
E73A C500 CMP MERK1
E73C F00A BEQ C41AC1
E73E 28 PLP
E73F E8 INX
E740 E8 INX
E741 E8 INX
E742 E07E CPX #126
E744 90EC BCC C41SL
E746 8030 BRA C41X Command-Tabelle voll
E748 AD019E C41AC1 LDA EXBUF+1 Command-Adresse = 0000 ?
E74B 0D029E ORA EXBUF+2
E74E D00E BNE C41AC2
E750 BD039F C41ML LDA CMTBL+3,X Kommando loeschen und Tabelle kuerzen
E753 9D009F STA CMTBL,X
E756 E8 INX
E757 E07B CPX #123

```

```

E759 90F5          BCC C41ML
E75B 28           PLP
E75C 8018         BRA C410K
E75E AD009E C41AC2 LDA EXBUF          Kommando anhaengen
E761 9D009F       STA CMTBL,X
E764 AD019E       LDA EXBUF+1
E767 9D019F       STA CMTBL+1,X
E76A AD029E       LDA EXBUF+2
E76D 9D029F       STA CMTBL+2,X
E770 28           PLP
E771 D003         BNE C410K          war eine '0' am Tabellenende vorhanden ?
E773 9E039F       STZ CMTBL+3,X      wieder ein '0' anhaengen
E776 6411 C410K   STZ COMST
E778 4C55E5 C41X   JMP QUITT          Quittung senden
E77B 4C07E5 C41E4E JMP SEND4E
;
Spezial Formatierungsroutine
'Complete' wird sofort zurueckgegeben
Das Laufwerk schreibt bei Bedarf selbststaendig
die Sektoren 1,2,3,360 und 1024 (mit COM44, Bit 4 einzustellen)
;
E77E 2050E5 COM20 JSR STELL2
E781 208BE4      JSR TSTWRP          Klappe + Write-Protect testen
E784 D002        BNE C20E1
E786 6411        STZ COMST
E788 2055E5 C20E1 JSR QUITT          Quittung vor dem Formatieren zuruecksenden
E78B A511        LDA COMST
E78D D00C        BNE RTN4
E78F 240A        BIT FORKEN2
E791 7009        BVS GC22
E793 20E7EC      JSR COM21          Format SD oder DD
E796 8007        BRA TFORER
E798 4CB2EF C20ERR JMP BELL1
E79B 60          RTN4
E79C 20E3EC GC22 JSR COM22          Format MD
9F A511 TFORER  LDA COMST
E7A1 D0F5        BNE C20ERR
E7A3 A561        LDA DSPCTR
E7A5 2910        AND #$10          Display/Drive Control - Bit 4 gesetzt ?
E7A7 D076        BNE WRVTX
E7A9 A200        LDX #0            ja, Sektoren nicht schreiben
E7AB 8D6BE8 C20ML  LDA BOOTID,X      Daten fuer Sector 1 in den Extended-Buffer kopieren
E7AE 9D009E       STA EXBUF,X
E7B1 9E809E       STZ EXBUF+$80,X  fuer DD 2.Sectorhaelfte loeschen
E7B4 EB          INX
E7B5 10F4        BPL C20ML
E7B7 20DBE9      JSR SEXBUF        Extended Buffer setzen
E7BA A901        LDA #1
E7BC 206BEC      JSR WRSECN        Sector 1 schreiben
E7BF A200        LDX #0
E7C1 BDEBEB C20M2L LDA BOOTID+$80,X  Daten fuer Sector 2 in den Extended-Buffer kopieren
E7C4 9D009E       STA EXBUF,X

```

```

E7C7 E8          INX
E7C8 10F7       BPL C20M2L
E7CA A902       LDA #2
E7CC 206BEC     JSR WRSECN          Sector 2 schreiben
E7CF A200       LDX #0
E7D1 BD6BE9 C20M3L LDA BOOTID+$0100,X  Daten fuer Sector 3 in den Extended-Buffer kopieren
E7D4 9D009E     STA EXBUF,X
E7D7 E8          INX
E7D8 E070       CPX # (BIDEND-BOOTID-$80  Bufferende loeschen
E7DA D0F5       BNE C20M3L
E7DC 9E009E C20DL  STZ EXBUF,X
E7DF E8          INX
E7E0 D0FA       BNE C20DL
E7E2 A903       LDA #3
E7E4 206BEC     JSR WRSECN          Sector 3 schreiben
E7E7 A20B       LDX #11
E7E9 BD41E8 C20SVT LDA VTTBL1,X
E7EC BC4DE8     LDY VTTBL2,X
E7EF 9219 C20VTL STA (IND)          VTDC erzeugen
E7F1 E619       INC IND
E7F3 88         DEY
E7F4 D0F9       BNE C20VTL
E7F6 CA         DEX
E7F7 10F0       BPL C20SVT
E7F9 A968       LDA #$68
E7FB 8582       STA $82
E7FD A901       LDA #1
E7FF 8583       STA $83
E801 20E4E9     JSR CALCTS          Track errechnen
E804 202FE3     JSR TRADJA          Kopf positionieren + Track # anzeigen
E807 240A       BIT FORKEN2
E809 500A       BVC WRVTOC
E80B A9F2       LDA #$F2
E80D 8D019E     STA EXBUF+1
E810 A903       LDA #3
E812 8D029E     STA EXBUF+2
E815 20DBE9 WRVTOC JSR SEXBUF
E818 2069EC     JSR WRSECT          Sector 360 (VTDC) schreiben
E81B 240A       BIT FORKEN2          Medium Density ?
E81D 7001       BVS C20MD           ja.
E81F 60         WRVTX  RTS
E820 A927 C20MD  LDA #39
E822 850D       STA TRACK
E824 202FE3     JSR TRADJA          Track 39 positionieren
E827 A20B       LDX #8
E829 BC62E8 MDVTL  LDY MDVT2,X
E82C BD59E8     LDA MDVT1,X
E82F 9219 MDVTL2 STA (IND)
E831 E619       INC IND
E833 88         DEY
E834 D0F9       BNE MDVTL2
E836 CA         DEX

```

```

EB37 10F0          BPL MDVTL
EB39 20DBE9       JSR SEXBUF
EB3C A90A         LDA #10
EB3E 4C6BEC       JMP WRSECN          Sector 1024 schreiben (nur bei MD)
;
; Daten fuer VTDC - Single Density
;
EB41 00FF7F00 VTTBL1 .BYTE 0,$FF,$7F,0,$FF,15,0,2,$C3,2,$C3,2
EB45 F0F0002
EB49 C302C302

EB4D 9C2B0101 VTTBL2 .BYTE $9C,43,1,1,44,1,5,1,1,1,1,1
EB51 2C010501
EB55 01010101
;
; Daten fuer VTDC - Medium Density
;
EB5D 7FFF7F00
EB61 FF
;
EB62 04010125 MDVT2 .BYTE 4,1,1,37,1,43,1,1,39
EB66 012B0101
EB6A 27
;
; Es folgt das Bootprogramm fuer die Sektoren 1-3
;
; C = $D6
; MASK = $E1
;
EB6B 00030007 BOOTID .BYTE 0,3,0,7,$77,$E4
EB6F 77E4
;
EB71 A9D6          LDA # (IDTEXT-BOOTID+$0700)
EB73 8500          STA $00
EB75 A907          LDA # )IDTEXT-BOOTID+$0700
7 8501            STA $01
EB79 A950          LDA #$50          Screen-Buffer $5000
EB7B 85D5          STA $D5
EB7D A200          LDX #0
EB7F 86D4          STX $D4
EB81 A000          LDY #0
EB83 A100 L2       LDA ($00,X)          Text in Screen-Buffer kopieren
EB85 300B          BMI OV          naechste Zeile
EB87 91D4          STA ($D4),Y
EB89 C8 LLP        INY
EB8A E600          INC $00
EB8C D0F5          BNE L2
EB8E E601          INC $01
EB90 D0F1          BNE L2
;
EB92 C9FF OV      CMP #$FF          Text-Ende ?

```

```

E894 F00F      BEQ PICOK
E896 297F      AND #7F      X-Position naechste Zeile uebernehmen
E898 A8        TAY
E899 A5D4      LDA $D4
E89B 692B      ADC #40      Screen-Buffer fuer 1 Zeile heraufzaehlen
E89D 85D4      STA $D4
E89F 90E8      BCC LLP
E8A1 E6D5      INC $D5
E8A3 D0E4      BNE LLP

;
E8A5 A954      PICOK LDA # <DLIST-BOOTID+$0700  Display-List fuer Screen-Buffer setzen
E8A7 8D3002    STA $0230
E8AA A908      LDA # >DLIST-BOOTID+$0700
E8AC 8D3102    STA $0231
E8AF A900      LDA #0
E8B1 8DC802    STA $02C8
E8B4 8DC602    STA $02C6
E8B7 8D0ED4    STA $D40E      Antic-Zugriff ausschalten
E8BA 85D6      STA COLDR
E8BC 85E1      STA MASK
E8BE A920      LDA #$20
E8C0 8DF402    STA $02F4      Zeichensatz-Basisadresse auf $2000
E8C3 A9B7      LDA # <DLI-BOOTID+$0700
E8C5 8D0002    STA $0200
E8C8 A907      LDA # >DLI-BOOTID+$0700
E8CA 8D0102    STA $0201
E8CD A9C0      DS          LDA #$C0
E8CF 8D0ED4    STA $D40E      Antic-Zugriff und DLI zulassen
E8D2 AC0AD2    LP          LDY $D20A
E8D5 B900E0    LDA $E000,Y    Zeichensatz per Bit-Mapping kopieren
E8D8 20BE07    JSR CODE-BOOTID+$0700
E8DB 990020    STA $2000,Y
E8DE AC0AD2    LDY $D20A
E8E1 B900E1    LDA $E100,Y
E8E4 20BE07    JSR CODE-BOOTID+$0700
E8E7 990021    STA $2100,Y
E8EA AC0AD2    LDY $D20A
E8ED B900E3    LDA $E300,Y
E8F0 20BE07    JSR CODE-BOOTID+$0700
E8F3 990023    STA $2300,Y
E8F6 4C6707    JMP LP-BOOTID+$0700
E8F9 48        CODE    PHA
E8FA E6D6      INC C
E8FC A5D6      LDA C
E8FE 2C6307    BIT DS-BOOTID+$0701
E901 100C      BPL A1
E903 500A      BVC A1
E905 A900      LDA #0
E907 85D6      STA C
E909 E6E1      INC MASK      Bit-Maske + 1
E90B D002      BNE A1
E90D C6E1      DEC MASK

```



```

E90F ASE1  A1      LDA MASK
E911 C9FF                CMP ##FF           alle Bits gesetzt
E913 D002                BNE D0
E915 68                PLA
E916 60                RTS
E917 AD0AD2  D0      LDA $D20A
E91A 25E1                AND MASK
E91C 85E0                STA $E0
E91E 68                PLA
E91F 25E0                AND $E0           Bits vom Zeichensatz ausblenden
E921 60                RTS
E922 48      DLI      PHA
E923 98                TYA
E924 48                PHA
E925 A00F                LDY #15
E927 98      DLIL     TYA
E928 0920                ORA ##20
      BD0AD4                STA $D40A
E92D BD16D0                STA $D016           setzen der Farbhelligkeit
E930 49AF                EOR ##AF
E932 BD17D0                STA $D017           Setzen der Helligkeit fuer Schrift
E935 88                DEY
E936 10EF                BPL DLIL
E938 A90C                LDA #$0C
E93A BD17D0                STA $D017
E93D 68                PLA
E93E A8                TAY
E93F 68                PLA
E940 40                RTI

```

```

;
Texte der Bootsectoren
;

```

```

E941 2469736B IDTEXT  .SBYTE "Disk formatiert mit:"
E945 00666F72
E949 6D617469
E94D 65727400
      6D69741A
E955 81                .BYTE $81
E956 33302525        .SBYTE "SPEEDY "
E95A 243900
E95D 51505550        .BYTE 81,80,85,80,0
E961 00
E962 36                .SBYTE "V"
E963 11                .BYTE VERSION/16^16
E964 0E                .SBYTE "."
E965 10                .BYTE VERSION&15^16
E966 83                .BYTE $83
E967 08630900        .SBYTE "(c) Bibosoft -- COMPY SHOP 1986"
E96B 2269626F
E96F 736F6674
E973 000D0D00
E977 232F2D30

```

```

E97B 39003328
E97F 2F300011
E983 191816
E986 80          .BYTE $80
E987 37656974   .SBYTE "Weitere Informationen bei:"
E98B 65726500
E98F 296E666F
E993 726D6174
E997 696F6E65
E99B 6E006265
E99F 691A
E9A1 90          .BYTE $90
E9A2 232F2D30   .SBYTE "COMPY SHOP"
E9A6 39003328
E9AA 2F30
E9AC 90          .BYTE $90
E9AD 34656C0E   .SBYTE "Tel.: 0208/497169"
E9B1 1A001012
E9B5 10180F14
E9B9 19171116
E9BD 19
E9BE FF          .BYTE $FF
;
Display - List
;
E9BF 70707070 DLIST .BYTE $70,$70,$70,$70,$70,$70,$70,$42,0,$50
E9C3 70707042
E9C7 0050
E9C9 7070F007   .BYTE $70,$70,$F0,7,6,$70,$70,2,$70,$70,2,$70,$70,2,2,$41
E9CD 06707002
E9D1 70700270
E9D5 70020241
E9D9 5408        .WORD DLIST-BOOTID+$0700
;
E9DB          BIDEND
;
;
E9DB A900    SEXBUF  LDA # <EXBUF          IND-Vector auf den Extended-Buffer setzen
E9DD 8519          STA IND
E9DF A99E          LDA # >EXBUF
E9E1 851A          STA IND+1
E9E3 60           RTS
;
E9E4 A582    CALCTS  LDA $82          Routine zum errechnen der Track- und Sectornummer
E9E6 8519          STA IND
E9E8 A583          LDA $83
E9EA 851A          STA IND+1
E9EC 3004          BMI CTSD          Sector # )$7FFF -> Buffer Adresse ROM/RAM
E9EE 0582          ORA $82          Sector 0 ?
E9F0 D00B          BNE CALC2         nein
E9F2 A508    CTSD   LDA DUMKEN
E9F4 2938          AND #$38

```

```

E9F6 C901          CMP #1          Bei 'Dumm' Carry setzen
E9F8 A582          LDA #82
E9FA 0583          ORA #83
E9FC 60            RTS
E9FD A0FF  CALC2   LDY #FF
E9FF C8  CALC1    INY
EA00 A519          LDA IND
EA02 850E          STA SECTOR
EA04 38            SEC
EA05 E51F          SBC SECANZ
EA07 8519          STA IND
EA09 B004          BCS ADDTRA
EA0B C61A          DEC IND+1
EA0D 3004          BMI CALCT
EA0F 051A  ADDTRA  ORA IND+1
EA11 D0EC          BNE CALCL
EA13 C02B  CALCT   CPY #40          'Carry' - Flag fuer Sector # - Error setzen
EA15 840D          STY TRACK
EA17 A50E          LDA SECTOR
EA19 60  CALCX    RTS
;
EA1A A50E  SETBUF  LDA SECTOR          * Diese Routine errechnet den RAM-Buffer *
EA1C 6419  SETBUF2 STZ IND          * aus der Sectornummer *
EA1E 3A            DEB
EA1F 18            CLC
EA20 2414          BIT SECLN
EA22 1003          BPL ADDHIB
EA24 4A            LSR A
EA25 6619          ROR IND
EA27 698C  ADDHIB  ADC # >DATBUF
EA29 851A          STA IND+1
EA2B 60  SETBX    RTS
;
EA2C A414  SETRWL  LDY SECLN          * Routine zum richtigen setzen der Read/Write - Laenge *
EA2E A583          LDA #83
EA30 D008          BNE SRWL
EA32 A582          LDA #82
EA34 C904          CMP #4          Sector # ( 4 ?
EA36 B002          BCS SRWL
EA38 A080          LDY #80          128 Bytes
EA3A 8413  SRWL    STY RWLEN
EA3C A901          LDA #1
EA3E 8562          STA BLOCKS          Anzahl der Datenblocks = 1
EA40 60            RTS
;
EA41 A562  ADDBUF  LDA BLOCKS          * Buffer erhoehen falls meherere Daten-Buffer uebertragen *
EA43 3A            DEB          * werden sollen *
EA44 F00F          BEQ ADDB4
EA46 A513          LDA RWLEN
EA48 F007          BEQ ADDB2
EA4A 18            CLC
EA4B 6519          ADC IND

```

```

EA4D 8519          STA IND
EA4F 9002          BCC ADDB3
EA51 E61A  ADDB2   INC  IND+1
EA53 C662  ADDB3   DEC  BLOCKS
EA55 60    ADDB4   RTS
;
EA56 A5B3  COM52   LDA  $83          * READ-SECTOR Routine *
EA58 306D          BMI  C520K      verzweigen bei Read RAM oder ROM
EA5A 0582          ORA  $82
EA5C F069          BEQ  C520K      verzweigen bei Read Zero-Page
;
EA5E 204CE5       JSR  STELL      Retry's und ERROR-Status setzen
EA61 207BE3       JSR  CONRES
EA64 3063          BMI  C52X
EA66 A508          LDA  DUMKEN      Wenn 'Dumm'-geschaltet, den Sector normal lesen
EA68 29B9          AND  $$B9
EA6A D044          BNE  C52TRA
;
EA6C A50D          LDA  TRACK      richtiger Track schon im RAM ?
EA6E C50C          CMP  LTRACK
EA70 F006          BEQ  TRACK
EA72 2064EB  C52TRR JSR  TSTWR      muss ein neuer Track komplett eingelesen werden, RAM vorher
EA75 206FE2       JSR  RDTRA      auf noch zu schreibende Sektoren pruefen
EA7B 2408  TRACK   BIT  DUMKEN
EA7A 7031          BVS  C52DUM
EA7C A60E          LDX  SECTOR
EA7E B540          LDA  STALST,X   Status des zu lesenden Sectors OK ?
EA80 C940          CMP  $$40
EA82 B0EE          BCS  C52TRR      Sector befindet sich noch nicht im RAM
EA84 291F          AND  $$1F
EA86 D025          BNE  C52DUM
EA88 201AEA       JSR  SETBUF      Buffer fuer entsprechenden Sector setzen
EA8B B540          LDA  STALST,X
EA8D F038          BEQ  C520K
EA8F 2058E5       JSR  QUITT2     Quittung und Daten senden
EA92 4C21E5       JMP  SDBYTS
;
EA95 C612  C52RTY   DEC  RETRY      Retry bei Read-Sector
EA97 3030          BMI  C52X
EA99 292F          AND  $$2F
EA9B D005          BNE  FLIP
EA9D 20D1E2       JSR  TRACK0     bei defektem Datenfeld Track 0
EAA0 800E          BRA  C52TRA
EAA2 A0FF  FLIP    LDY  $$FF      1 Track Rueckwaerts
EAA4 200FE3       JSR  TRVR
EAA7 C8           INY           1 Track vorwaerts
EAA8 200FE3       JSR  TRVR
EAB0 8003          BRA  C52TRA
;
EABD 201AEA  C52DUM JSR  SETBUF
EAB0 2032E3  C52TRA JSR  TRADJ      * Normale Read-Sector Routine *
EAB3 D014          BNE  C52X

```

```

EAB5 A60E          LDX SECTOR
EAB7 20CFEA       JSR RDSECT
EABA A8           TAY
EABB A508         LDA DUMKEN          wenn nicht 'Dumm' geschaltet
EABD 29B9         AND #$B9           Status in Statusliste uebernehmen
EABF D003         BNE C52NSS
EAC1 98           TYA
EAC2 9540         STA STALST,X
EAC4 98           C52NSS TYA
EAC5 D0CE         BNE C52RTY
;
EAC7 6411         C520K  STZ COMST
EAC9 2055E5       C52X   JSR QUITT          Quittung und Daten senden
EACC 4C21E5       JMP SDBYTS
;
EACF A50E         RDSECT LDA SECTOR          * Einzelnen Sector in vorbezeichneten RAM einlesen *
EAD1 8D0204       STA $0402
   D4 A988         RDSEC1 LDA ##88           Read Sector fuer Controller
EAD6 8D0004       STA $0400
EAD9 A000         LDY #0
EADB A9E6         RDSST  LDA #$E6
EADD 8D9F02       STA $029F          Timer setzen
EAE0 2C8002       RDSL  BIT $0280
EAE3 5029         BVC RDSTO          Time out
EAE5 10F9         BPL RDSL
EAE7 AD0304       LDA $0403          1 Byte vom Controller uebernehmen
EAEA 49FF         EOR ##FF
EAE8 9119         STA (IND),Y       und in den Sectorbuffer schreiben
EAE9 AD9602       LDA $0296
EAF1 C8           INY
EAF2 2C8002       RDSL2 BIT $0280
EAF5 5017         BVC RDSTO
EAF7 10F9         BPL RDSL2
EAF9 AD0304       LDA $0403
EAFB 49FF         EOR ##FF
EAFE 9119         STA (IND),Y
   00 C8           INY
EB01 C414         CPY SECLN
EB03 D0ED         BNE RDSL2
EB05 2085E3       JSR WREADY        Warten auf Controller 'Ready'
EB08 AD0004       RDSRS LDA $0400
EB0B 293E         AND #$3E          Status pruefen
EB0D 60           RTS
EB0E AD0004       RDSTO LDA $0400
EB11 4A           LSR A             'In Use' Flag noch gesetzt ?
EB12 B0C7         BCS RDSST
EB14 AC9602       LDY $0296
EB17 4A           LSR A             'Lost Data' Flag gesetzt ?
EB18 4A           LSR A
EB19 B0B9         BCS RDSEC1
EB1B 80EB        BRA RDSRS
;

```

```

EB1D 2032E3 RDHDSF JSR TRADJ      * Diese Routine fuehrt 3 Header-Read Operationen aus *
EB20 D00D      BNE RTN3      * und gibt bei erfolgreichem Leseversuch die Sector # *
EB22 A003      RDHDS1 LDY #3      * im Accu zurueck *
EB24 2030EB RDHSL JSR RDHDV
EB27 A57C      LDA #7C
EB29 9004      BCC RTN3      Leseversuch erfolgreich: 'Carry' = 0
EB2B 88        DEY
EB2C D0F6      BNE RDHSL
EB2E 38        SEC
EB2F 60      RTN3      RTS
;
EB30 A920      RDHDV      LDA ##20      Zeit fuer den Controller setzen einen Header zu finden
EB32 8D9F02    STA $029F
EB35 2042EB RDHDVL JSR RDHD1
EB38 B002      BCS RDHVX      wird kein Header gefunden, so ist das 'Carry'-Flag gesetzt
EB3A D0F9      BNE RDHDVL
EB3C 60      RDHVX      RTS
;
EB3D A9D8      RDHEAD     LDA ##D8      * Hier liegt die gleiche Funktion wie bei RDHDV vor, nur wird *
EB3F 8D9F02    STA $029F      * dem Controller etwas ueber eine komplette Umdrehung der *
EB42 A27A      RDHD1      LDX ##7A      * Diskette Zeit gegeben. *
EB44 A9CB      LDA ##CB
EB46 8D0004    STA $0400      Der Header besteht aus 6 Bytes, die ab Adresse $7A abgelegt
EB49 2C8002 RDHDL BIT $0280      werden:
EB4C 5011      BVC SINFTD
EB4E 10F9      BPL RDHDL      1. Byte ($7A) Track Nr.
EB50 AD0304    LDA $0403      2. Byte ($7B) Side Nr.
EB53 9500      STA $00,X      3. Byte ($7C) Sector Nr.
EB55 E8        INX      4. Byte ($7D) Sector laenge
EB56 10F1      BPL RDHDL      5.+6. Byte ($7E,$7F) 2 CRC Bytes
EB58 2085E3    JSR WREADY
EB5B 290C      AND ##0C
EB5D 18        CLC
EB5E 60      RTS
EB5F 207BE3 SINFTD JSR CONRES
EB62 38        SEC
EB63 60      RTS
;
Die Sektoren, die als zu schreibend in der Statusliste gekennzeichnet sind
auf die Diskette schreiben
;
EB64 A506      TSTWR      LDA WRKEN      Anzahl der zu schreibenden Sektoren
EB66 F06A      BEQ TWREX
EB68 2041E1    JSR MOTON      Motor zwingend einschalten
EB6B A50D      LDA TRACK      aktuellen Track merken
EB6D 48        PHA
EB6E A50B      LDA LWRTRA
EB70 850D      STA TRACK
EB72 20F6EF    JSR TRAANZ      Track Nr. anzeigen
EB75 203CE3    JSR TRADJO      Kopf positionieren
EB78 2022EB    JSR RDHDS1      1. Header suchen
EB7B FA        PLX

```

```

EB7C 860D      STX TRACK          Track # zurueckholen
EB7E 6401      STZ MERK2
EB80 B020      BCS TWRERR
EB82 A200      LDX #0
EB84 D520     TWIL      CMP SECLST,X      Sector in Sectorliste suchen
EB86 F03B      BEQ NOWRS
EB88 E8        INX
EB89 E41F      CPX SECANZ
EB8B 90F7      BCC TWIL
EB8D 20B2EF   TWERR2     JSR BELL1         Sector nicht in Sectorliste gefunden, 1 Bell ausgeben
EB90 8031      BRA NOWRS
;
EB92 B420     TSTWL      LDY SECLST,X
EB94 B94000    LDA STALST,Y     ein zu schreibender Sector ?
EB97 102A     BPL NOWRS      nein
EB99 2088E4   JSR TSTWRP     Write Protect + Klappe testen
EB9C F00B     BEQ TSTW1
EB9E A501     LDA MERK2
EBA0 D018     BNE TSTW2
EBA2 20D3EB   TWERR2     JSR WRERR        5 Sekunden auf dem Display herunterzaehlen und Bell ausgeben
EBA5 8501     STA MERK2
EBA7 801A     BRA NOWRS
EBA9 98       TSTW1      TYA
EBAE BD0204   STA $0402       Sector # ins Sectorregister des Contollers schreiben
EBAD 201CEA   JSR SETBUF2
EBB0 206EEC   JSR WRSEC1      Sector schreiben
EBB3 F005     BEQ TSTW2       Status OK
EBB5 2088E4   JSR TSTWRP     Write Protect + Klappe testen
EBB8 D009     BNE NOWRS
EBBA B420     TSTW2      LDY SECLST,X     'Sector written'-Status setzen
EBBC A940     LDA #$40
EBBE 994000   STA STALST,Y
EBC1 C606     DEC WRKEN
;
EBC3 E8       NOWRS     INX              naechste Position in Sectorliste setzen
EBC4 E41F     CPX SECANZ
EBC6 9002     BCC TSTWE
EBC8 A200     LDX #0
EBCA A506     TSTWE      LDA WRKEN
EBCD D0C4     BNE TSTWL
EBCF A9FF     LDA #$FF
EBD0 850B     STA LWRTRA     'Track written'-Status setzen
EBD2 60       TWREX      RTS
;

```

Diese Routine wird von der Write-Track Routine aufgerufen wenn die Diskette vor einer abgeschlossenen Write-Sequenz aus dem Laufwerk genommen oder Write-Protected wird. 5 Sekunden werden auf dem Display herabgezaehlt und nach jeder Sekunde wird ein Warnton abgegeben

```

;
EBD3 DA       WRERR     PHX
EBD4 5A       PHY

```

```

EBD5 A005          LDY #5          5 Sekunden setzen
EBD7 98           WRERRL      TYA
EBD8 20D1EF       JSR ANZEIGE      Sekunden zur Anzeige bringen
EBDB 20B2EF       JSR BELL1        1 Bell ausgeben
EBDE A203         LDX #3          grosse Verzoeigerungsschleife (ca. 1 Sekunde) setzen
EBE0 A9FF         WEWL1          LDA #$FF
EBE2 8D9F02       STA $029F
EBE5 208BE4       WEWL2          JSR TSTWRP      noch Write-Protected oder Klappe auf ?
EBE8 F00B         BEQ WRERX
EBEA 2C8002       BIT $0280
EBED 70F6         BVS WEWL2
EBEF CA          DEX
EBF0 D0EE        BNE WEWL1
EBF2 88          DEY
EBF3 10E2        BPL WRERRL
EBF5 20F6EF       WRERX          JSR TRAANZ      aktuelle Track # wieder anzeigen
EBF8 AE9602       LDX $0296      Timer IRQ zuruecksetzen
EBFB 7A          PLY            Y-Register zurueckholen
EBFC FA          PLX            X-Register zurueckholen
EBFD 4C8BE4       JMP TSTWRP
;
EC00 204CE5       COM50         JSR STELL      * Write Sector Routine *
EC03 A583         LDA $83
EC05 305D         BMI C500K2     Sector # ) 7FFF -> RAM - oder ROM - Adresse
EC07 A581         LDA $81
EC09 C957         CMP #$57       Write mit Verify ?
EC0B F022         BEQ C50DUM
EC0D 208BE4       JSR TSTWRP     Write Protect + Klappe testen
EC10 D054         BNE C50X
EC12 203CE1       JSR TSTMON     Motor einschalten
EC15 A508         LDA DUMKEN
EC17 29FA         AND #$FA       'Dumm' geschaltet ?
EC19 D014         BNE C50DUM
;
EC1B A50D         LDA TRACK      einen kompletten Track im RAM ?
EC1D C50B         CMP LWRTRA
EC1F D006         BNE C50B
EC21 A506         LDA WRKEN
EC23 C51F         CMP SECANZ
EC25 D03D         BNE C500K2
EC27 2064EB       C50B          JSR TSTWR      alle zu schreibenden Sektoren schreiben
EC2A 208EE4       C50C          JSR TSTMER     falls vorhanden, Sectordaten aus Extended Buffer
                                                in den Sectorbuffer kopieren
EC2D 8035         BRA C500K2
;
Normale Write-Sector Routine
;
EC2F 208BE4       C50DUM       JSR TSTWRP     Disk 'Write Protect' oder Klappe auf ?
EC32 D032         BNE C50X      js
EC34 A50D         LDA TRACK
EC36 C50C         CMP LTRACK
EC38 D004         BNE C50TRA

```



```

EC3A A60E          LDX SECTOR          zu schreibenden Sector als im RAM stehende Daten markieren
EC3C 7440          STZ STALST,X       braucht bei 'Read Sector' nicht mehr gelesen zu werden
EC3E 2032E3 C50TRA JSR TRADJ           Kopf positionieren
EC41 D023          BNE C50X
EC43 2069EC       JSR WRSECT         Sector vom angegebenen Buffer schreiben
EC46 F009          BEQ C500K
EC48 C612          DEC RETRY
EC4A F01A          BEQ C50X
EC4C 20D1E2       JSR TRACK0        Track 0 positionieren
EC4F 80DE          BRA C50DUM        Retry ausfuehren
;
EC51 A581 C500K LDA $81
EC53 C957          CMP #$57          Write mit Verify
EC55 D00D          BNE C500K2
EC57 A508          LDA DUMKEN
EC59 2904          AND #4            Verify bei COM57 ausfuehren ?
EC5B D007          BNE C500K2
EC5D 20A4EC       JSR VERSEC        Verify Sector
EC60 B004          BCS C50X
EC62 D002          BNE C50X
EC64 6411 C500K2 STZ COMST         OK-Status setzen
EC66 4C55E5 C50X  JMP QUITT        Quittung senden
;
EC69 A50E WRSECT LDA SECTOR          * Einzelnen Sector aus vorbezeichneter RAM-Adresse *
EC6B 8D0204 WRSECN STA $0402        * auf die Diskette schreiben *
EC6E A000 WRSEC1 LDY #0
EC70 A9A8          LDA #$A8          Write Sector Befehl an Controller
EC72 8D0004       STA $0400
EC75 A9E6 WRSST  LDA #$E6
EC77 8D9F02       STA $029F        Timer setzen
EC7A B119 WRSL   LDA (IND),Y    1 Byte invertiert
EC7C 49FF          EOR #$FF
EC7E 2C8002 WRSDR BIT $0280
EC81 5013          BVC WRSTO        Time out
EC83 10F9          BPL WRSDR
EC85 8D0304       STA $0403        Byte an den Controller uebergeben
EC87 AD9602       LDA $0296
EC8B CB          INY
EC8C C414          CPY SECLN        alle Bytes geschrieben ?
EC8E D0EA          BNE WRSL
EC90 2085E3       JSR WREADY       auf Controller 'Ready' warten
EC93 290C          AND #$0C
EC95 60           RTS
EC96 AD0004 WRSTO LDA $0400
EC99 4A           LSR A            'In Use' Bit gesetzt ?
EC9A B0D9          BCS WRSST
EC9C 4A           LSR A            'Data Request' gesetzt ?
EC9D 4A           LSR A
EC9E B0CE          BCS WRSEC1
ECA0 AD0004       LDA $0400
ECA3 60           RTS
;

```

```

ECA4 A50E   VERSEC   LDA SECTOR           * Sector mit Datenbuffer vergleichen *
ECA6 8D0204          STA $0402
ECA9 A988   VERSE1   LDA #$88             Read Sector an Controller
ECAB 8D0004          STA $0400
ECAE A000          LDY #0
ECB0 A9D8   VERSST   LDA #$D8
ECB2 8D9F02          STA $029F           Timer setzen
ECB5 B119   VERSL    LDA (IND),Y
ECB7 49FF          EOR #$FF
ECB9 2C8002  VERSL2   BIT $0280
ECBC 501A          BVC VERSTO         Time out
ECBE 10F9          BPL VERSL2
ECC0 CD0304          CMP $0403
ECC3 D00C          BNE VERSER
ECC5 C8           INY
ECC6 C414          CPY SECLN          alle Daten verglichen ?
ECC8 D0EB          BNE VERSL
ECCA 2085E3        JSR WREADY         auf Controller 'Ready' warten
ECCD A900          LDA #0             alle Daten verglichen und OK
ECCF 18           CLC              kein Fehler aufgetreten
ECD0 60           RTS
ECD1 207BE3  VERSER   JSR CONRES
ECD4 A980          LDA #$80          Daten ungleich
ECD6 18           CLC              kein Lesefehler aufgetreten
ECD7 60           RTS
ECD8 AD0004  VERSTO   LDA $0400
ECDB 4A           LSR A             Kommando noch 'In Use'
ECDC B0D2          ECS VERSST
ECDE 207BE3        JSR CONRES
ECE1 38           SEC              'Carry' = ERROR-Flag
ECE2 60           RTS
;
Formatierungs-Routinen
;
ECE3 A941   COM22   LDA #$41           Medium Density (MD)-Kennung
ECE5 D014          BNE FSDATB
;
ECE7 A582   COM21   LDA #82
ECE9 C911          CMP #$11          Sector Nr. 1041 ?
ECEB D006          BNE C21TSD
ECED A583          LDA #83
ECEF C904          CMP #4
ECF1 F0F0          BEQ COM22
ECF3 A982   C21TSD  LDA #$82           Single Density (SD)-Kennung
ECF5 A60A          LDX FORKEN2
ECF7 D002          BNE FSDATB
;
ECF9 A900   C21DD   LDA #0             Double Density (DD)-Kennung
;
ECFB 8509   FSDATB  STA FORKEN
ECFD 2050E5        JSR STELL2
ED00 2068ED        JSR COPSLT         Neue Sectorliste in die Zeropage kopieren

```

```

ED03 A508      LDA DUMKEN
ED05 293F      AND #3F
ED07 2019E2    JSR SETDK
ED0A A509      LDA FORKEN
ED0C 2901      AND #1
ED0E 4901      EOR #1
ED10 8501      STA MERK2      0/1 = Sectorlaenge-Kennung
ED12 A927      LDA #39        40 Tracks
ED14 850D      STA TRACK
ED16 20EFED    FORMIL JSR FORMTR      1 Track formatieren
ED19 B018      BCS FORMIX
ED1B A561      LDA DSPCTR
ED1D 2920      AND #20        Bit 5 (Display+Drive-Kontrollbyte)
ED1F D00C      BNE NOVER      schaltet Verify beim Formatieren aus
ED21 A61F      LDX SECANZ
ED23 208BE2    JSR RDTRA2      alle Sektoren lesen
ED26 B00B      BCS FORMIX
ED28 2050E2    JSR RDTRV2      falls Lesefehler 1 Retry
ED2B B006      BCS FORMIX
ED2D C60D      NOVER  DEC TRACK
ED2F 10E5      BPL FORMIL
ED31 6411      STZ COMST      gesamte Formatierung OK
;
ED33 A9FF      FORMIX LDA #FF
ED35 A200      LDX #0
ED37 9D009E    FORDL  STA EXBUF, X    Datenbuffer mit Sector OK-Bits fuellen
ED3A E8        INX
ED3B E413      CPX RWLEN
ED3D D0F8      BNE FORDL
ED3F 2411      BIT COMST
ED41 1012      BPL SFSTAT
ED43 9C009E    STZ EXBUF      bei Formaterror OK-Bits loeschen
ED46 9C019E    STZ EXBUF+1
ED49 208BE4    JSR TSTWRP     bei Write Protect nicht 'Dumm' schalten
ED4C D007      BNE SFSTAT
ED4E A980      LDA #80        bei Formaterror 'Dumm' schalten
;50 0408      TSB DUMKEN
ED52 2034E2    JSR DSPD2
ED55 A618      SFSTAT  LDX COMPOS     bei Befehl aus erweiterter Kommando-Tabelle
ED57 E012      CPX #18        keine Daten an den Computer senden
ED59 B026      BCS CSLX
ED5B 2055E5    JSR QUITT
ED5E 20DBE9    JSR SEXBUF     Extended Buffer setzen
ED61 A514      LDA SECLN
ED63 8513      STA RWLEN
ED65 4C21E5    JMP SDBYTS
;
ED68 20A9E1    COPSLT JSR SDRDDP     Drive Density setzen und Anzeigen
ED6B A509      LDA FORKEN
ED6D 2903      AND #3
ED6F A8        TAY
ED70 BE82ED    LDX SLTBEG, Y

```

```

ED73 A000          LDY #0          aktuelle Sector-Liste in die Zeropage kopieren
ED75 B085ED CSLTL  LDA SDSTBL,X
ED78 992000        STA SECLST,Y
ED7B E8            INX
ED7C C8            INY
ED7D C41F          CPY SECANZ
ED7F D0F4          BNE CSLTL
ED81 60           CSLX           RTS
ED82 2C1200 SLTBEG .BYTE DDSTBL-SDSTBL,MDSTBL-SDSTBL,0
;
Single Density Sectorliste
;
ED85 01030507 SDSTBL .BYTE 1,3,5,7,9,11,13,15,17,2,4,6,8,10,12,14,16,18
ED89 090B0D0F
ED8D 11020406
ED91 080A0C0E
ED95 1012
;
Medium Density Sectorliste
;
ED97 01030507 MDSTBL .BYTE 1,3,5,7,9,11,13,15,17,19,21,23,25,2,4,6,8,10,12,14,16,18,20,22,24,26
ED9B 090B0D0F
ED9F 11131517
EDA3 19020406
EDA7 080A0C0E
EDAB 10121416
EDAF 181A
;
Double Density Sectorliste
;
EDB1 060C1205 DDSTBL .BYTE 6,12,18,5,11,17,4,10,16,3,9,15,2,8,14,1,7,13
EDB5 0B11040A
EDB9 1003090F
EDBD 02080E01
EDC1 070D
;
EDC3 38           FORERR        SEC          'Carry'=Format ERROR-Flag
EDC4 60           RTS
;
Write Track Kommando starten
;
EDC5 202FE3 FSTART JSR TRADJA          Kopf positionieren und Track # anzeigen
EDC8 D0F9          BNE FORERR
EDCA 208BE4        JSR TSTWRP          Write Protect und Klappe pruefen
EDCD D0F4          BNE FORERR
EDCF A905          LDA #5
EDD1 8D9F02        STA $029F
EDD4 A9F8          LDA #$F8           Write Track an Controller geben
EDD6 8D0004        STA $0400
EDD9 A500          LDA MERK1
EDDB A202          LDX #2
EDDD 2C8002 FORWA1 BIT $0280

```

```

EDE0 10FB          BPL FORWA1
EDE2 8D0304        STA #0403
EDES CA            DEX
EDE6 D0F5          BNE FORWA1
EDE8 A0D0          LDY ##D0
EDEA 8C9F02        STY #029F          Timer setzen
EDED 18            CLC
EDEE 60            RTS
;
EDEF 2409          FORMTR BIT FORKEN
EDF1 1064          BPL FORMMD          in 'MFM' (MD/DD) formatieren
;
EDF3 A900          FORMSD LDA #0          in 'FM' (Single Density) formatieren
EDF5 8500          STA MERK1
EDF7 20C5ED        JSR FSTART        'Write Track' Kommando starten
EDFA 80C7          BCS FORERR
EDFC A050          LDY #80
EDFE 20F1EE        JSR WRBYTS
EE01 A9FC          LDA ##FC
EE03 20E6EE        JSR WRBYTE        Track AM schreiben
;
EE06 98           FSDL  TYA
EE07 A019          LDY #25
EE09 20F1EE        JSR WRBYTS
;
EE0C A9FE          LDA ##FE
EE0E 20E6EE        JSR WRBYTE        ID AM
EE11 A50D          LDA TRACK
EE13 20E6EE        JSR WRBYTE        Track #
EE16 98            TYA
EE17 20E6EE        JSR WRBYTE        Side #
EE1A B520          LDA SECLST,X
EE1C 20E6EE        JSR WRBYTE        Sector #
EE1F 98            TYA
EE20 20E6EE        JSR WRBYTE        Sectorlaenge + 0
EE23 A9F7          LDA ##F7
    5 20E6EE        JSR WRBYTE        2 CRC-Byte schreiben
EE28 EB            INX
;
EE29 98            TYA
EE2A A011          LDY #17
EE2C 20F1EE        JSR WRBYTS
;
EE2F A9FB          LDA ##FB
EE31 20E6EE        JSR WRBYTE        DATA AM
EE34 A9FF          LDA ##FF
EE36 A080          LDY ##80          128 Bytes
EE38 20F1EE        JSR WRBYTS        Datenfeld schreiben
EE3B A9F7          LDA ##F7
EE3D 20E6EE        JSR WRBYTE        2 CRC-Bytes schreiben
;
EE40 E41F          CPX SECANZ

```

```

EE42 D0C2          BNE FSDL
;
EE44 A901    FSDEX    LDA #1
EE46 2D0004  FSDEL    AND $0400          Warten auf 'In Use'-Flag=0
EE49 F00A          BEQ FORM2X
EE4B 2C8002          BIT $0280
EE4E 10F6          BPL FSDEL
EE50 8C0304          STY $0403
EE53 80F1          BRA FSDEL
EE55 18          FORM2X  CLC          Format Track OK
EE56 60          RTNS     RTS
;
EE57 A94E    FORMMD  LDA #$4E          In 'MFM' (MD/DD) formatieren
EE59 8500          STA MERK1
EE5B 20C5ED          JSR FSTART          'Write Track' Kommando starten
EE5E 80F6          BCS RTNS
EE60 A080          LDY #$80
EE62 20F1EE          JSR WRBYTS
EE65 98          TYA
EE66 A00C          LDY #12
EE68 20F1EE          JSR WRBYTS
EE6B A9F6          LDA #$F6
EE6D A003          LDY #3
EE6F 20F1EE          JSR WRBYTS
EE72 A9FC          LDA #$FC
EE74 20E6EE          JSR WRBYTE          Track AM schreiben
EE77 A94E          LDA #$4E
EE79 A032          LDY #$32
EE7B 20F1EE          JSR WRBYTS
;
EE7E 98          FMDL    TYA
EE7F A00C          LDY #12
EE81 20F1EE          JSR WRBYTS
EE84 A9F5          LDA #$F5
EE86 A003          LDY #3
EE88 20F1EE          JSR WRBYTS
;
EE8B A9FE          LDA #$FE
EE8D 20E6EE          JSR WRBYTE          ID AM
EE90 A50D          LDA TRACK
EE92 20E6EE          JSR WRBYTE          Track #
EE95 98          TYA
EE96 20E6EE          JSR WRBYTE          Side #
EE99 B520          LDA SECLST,X
EE9B 20E6EE          JSR WRBYTE          Sector #
EE9E A501          LDA MERK2
EEA0 20E6EE          JSR WRBYTE          Sectorlaenge (0=$80/1=$100)
EEA3 A9F7          LDA #$F7
EEA5 20E6EE          JSR WRBYTE          2 CRC-Bytes schreiben
EEA8 E8          INX
;
EEA9 98          TYA

```

```

EEAA 20E6EE      JSR WRBYTE
EEAD A94E        LDA #$4E
EEAF A016        LDY #22
EEB1 20F1EE      JSR WRBYTS
EEB4 98          TYA
EEB5 A00C        LDY #12
EEB7 20F1EE      JSR WRBYTS
EEBA A9F5        LDA #$F5
EEBC A003        LDY #3
EEBE 20F1EE      JSR WRBYTS
;
EEC1 A9FB        LDA #$FB
EEC3 20E6EE      JSR WRBYTE          DATA AM
EEC6 A9FF        LDA #$FF
EEC8 A414        LDY SECLN          128 oder 256 Bytes
EECA 20F1EE      JSR WRBYTS          Datenfeld schreiben
EECD A9F7        LDA #$F7
EECF 20E6EE      JSR WRBYTE          2 CRC-Bytes
;
EED2 E41F        CPX SECANZ          letzten Sector formatiert ?
EED4 F009        BEQ FMDEX
;
EED6 A94E        LDA #$4E
EED8 A029        LDY #41
EEDA 20F1EE      JSR WRBYTS
EEDD 809F        BRA FMDL
;
EEDF A04E      FMDEX  LDY #$4E
EEE1 4C44EE      JMP FSDEX          warten auf 'In Use' - Flag = 0
;
EEE4 5017      WRBTL   BVC FORT0
EEE6 2C8002    WRBYTE  BIT $0280          1 Byte an Controller uebergaben
EEE9 10F9      BPL WRBTL
EEEB 8D0304    STA $0403
EEEE 60        RTS
;
EEEF 500C      WRBTSL  BVC FORT0
EEF1 2C8002    WRBYTS  BIT $0280          Y-Register = Anzahl der an den Controller zu
EEF4 10F9      BPL WRBTSL          uebergabenden Bytes
EEF6 8D0304    STA $0403
EEF9 88        DEY
EEFA D0F5      BNE WRBYTS
EEFC 60        RTS
;
EEFD 68        FORTO   PLA          Format Time-Out
EEFE 68        PLA
EEFF 38        SEC
EF00 60        RTS
;
EF01 A927      CLRDSK  LDA #39          * Routine zum 'loeschen' einer kompletten Diskette *
EF03 850D      STA TRACK          * Es wird ein unlesbares Format erzeugt *
EF05 200FEF    CLRDKL  JSR CLRTRA

```

```

EF08 B004          BCS CLRDKX
EFOA C60D          DEC TRACK
EFOC 10F7          BPL CLRDKL
EFOE 60           CLRDKX  RTS
;
EFOF A9AA          CLRTRA  LDA ##AA          * Einen Track 'loeschen' *
EF11 8500          STA MERK1
EF13 20C5ED        JSR FSTART          'Write Track' - Kommando starten
EF16 B0F6          BCS CLRDKX
EF18 AD0004        CLRLOP  LDA $0400
EF1B 4A           LSR A          'In Use' - Flag = 0 ?
EF1C 90F0          BCC CLRDKX
EF1E 2C8002        BIT $0280
EF21 10F5          BPL CLRLOP
EF23 8D0304        STA $0403
EF26 B0F0          BRA CLRLOP
;
EF28 A275          COM4E   LDX #$75          Diese Routine gibt folgende Werte an den Computer zurueck:
EF2A 7400          C4EDL   STZ $00,X          Tracks/Disk
EF2C E8           INX           Sektoren/Track
EF2D 10FB          BPL C4EDL          Side #
EF2F A928          LDA ##28          FM/MFM-Kennung
EF31 8574          STA $74          Bytes/Sector
EF33 E675          INC $75
EF35 A51F          LDA SECANZ
EF37 8577          STA $77
EF39 2409          BIT FORKEN
EF3B 3004          BMI SETSEL
EF3D A904          LDA #4           MFM-Kennung
EF3F 8579          STA $79
EF41 A980          SETSEL  LDA ##80          128 Bytes/Sector
EF43 A609          LDX FORKEN
EF45 D003          BNE LOLEN
EF47 0A           ASL A
EF48 267A          ROL $7A          Sektoren/Track HI
EF4A 857B          LOLEN  STA $7B          Sektoren/Track LO
EF4C C67C          DEC $7C          =FF
EF4E 20FBE4        JSR SEND41
EF51 A90C          LDA #12
EF53 A274          LDX ##74          12 Bytes ab Adresse $74 an den Computer senden
EF55 A000          LDY #0
EF57 4C1BE5        JMP SDBTS
;
EF5A A90C          COM4F   LDA #12          * Diese Routine ist das Gegenstueck von COM4E *
EF5C A274          LDX ##74          * Laufwerk konfigurieren *
EF5E A000          LDY #0
EF60 20BCE4        JSR RDBTS
EF63 20FBE4        JSR SEND41          'A' senden
EF66 A982          LDA ##82          SD-Kennung
EF68 A679          LDX $79
EF6A F00A          BEQ C4FSD
EF6C A941          LDA ##41          MD-Kennung

```



```

EF6E A677          LDX #77
EF70 E01A          CPX #26
EF72 B002          BCS C4FSD
EF74 A900          LDA #0                DD-Kennung
EF76 B509          C4FSD          STA FORKEN
EF78 B50A          STA FORKEN2
EF7A 20A9E1        JSR SDRDDP           Drive Density stellen und anzeigen
EF7D 4CFFE4        JMP SEND43          'C' senden
;
EF80 2078E3        COM53          JSR CONRE2          * 'Status' - Kommando *
EF83 A510          LDA DRSTAT          Diese Routine gibt folgende Werte an den Computer:
EF85 29F7          AND #$F7           Drive Status + Time-Out-Wert
EF87 2C0004        BIT $0400
EF8A 5002          BVC ST7C
EF8C 0908          ORA #8             Write Protect Flag
EF8E B57C          ST7C             STA $7C
EF90 A50F          LDA CONST          Controller - Status invertieren
EF92 49FF          EOR #$FF
EF94 B57D          STA $7D
EF96 A9E0          LDA #$E0
EF98 B57E          STA $7E
EF9A 647F          STZ $7F
EF9C AD0004        LDA $0400
EF9F 850F          STA CONST
EFA1 20FFE4        JSR SEND43          'C' senden
EFA4 A904          LDA #4
EFA6 A27C          LDX #$7C           4 Bytes ab Adresse $7C
EFA8 A000          LDY #0
EFAA 4C1BE5        JMP SDBTS          an den Computer senden
;
;
          DISP1      = $4000
          DISP10     = $4001
          DENSITY    = $4002
          BUZZER     = $4003
;
EFAD A561          BELL          LDA DSPCTR          * Bell-Routine - akustische Unterstuetzung diverser *
EFAF 4A           LSR A          * Laufwerksfunktionen und Meldungen *
EFB0 9014          BCC BELLX
EFB2 5A           BELL1         PHY                Y-Register merken
EFB3 A002          LDY #2
EFB5 A200          BELL2         LDX #0
EFB7 A946          S01          LDA #$23
EFB9 3A           S02          DEA
EFBA D0FD          BNE S02
EFBC BD0340        STA BUZZER          Summer ansprechen
EFBF CA           DEX
EFC0 D0F5          BNE S01
EFC2 88           DEY
EFC3 D0F0          BNE BELL2
EFC5 7A           PLY                Y-Register zurueckholen
EFC6 60           BELLX         RTS

```

```

;
EFC7 9C0040 CLRDSP STZ DISP1      * Display loeschen *
EFC8 9C0140          STZ DISP10
EFC9 9C0240          STZ DENSITY
EFD0 60              RTS
;
EFD1 48      ANZEIGE PHA          * Diese Routine wird zur Hex-Darstellung von Werten *
EFD2 DA          PHX          * im Accu auf dem Display verwendet, es werden keine *
EFD3 5A          PHY          * Register ausser des Status-Registers veraendert *
EFD4 20DFEF     JSR HEXOUT
EFD7 7A          PLY
EFD8 FA          PLX
EFD9 68          PLA
EFDA 60          RTS
;
EFDB 2461     ERRDSP BIT DSFCTR
EFDD 1016     BPL HEXX
EFDf 48      HEXOUT PHA          Hex-Darstellung auf dem Display
EFE0 290F     AND #0F
EFE2 AA          TAX
EFE3 BD2BF0   LDA SEGTL,X
EFE6 8D0040   STA DISP1      rechtes Display
EFE9 68          PLA
EFEA 4A          LSR A
EFEB 4A          LSR A
EFEC 4A          LSR A
EFED 4A          LSR A
EFEE AA          TAX
EFEF BD2BF0   LDA SEGTL,X
EFF2 8D0140   STA DISP10    linkes Display
EFF5 60      HEXX  RTS
;
EFF6 48      TRANZ  PHA          * Anzeige der Track # auf dem Display *
EFF7 A50D     LDA TRACK
EFF9 2461     BIT DSFCTR
EFFB 7005     BVS TRAHX
EFFD 2007F0   JSR DEZOUT     Dezimal-Darstellung
F000 68          PLA
F001 60          RTS
F002 20DFEF   TRAHX JSR HEXOUT  Hex-Darstellung
F005 68          PLA
F006 60          RTS
;
F007 A200     DEZOUT LDX #0      * Wert im Accu in dezimaler Form auf dem Display ausgeben *
F009 38          SEC
F00A A8      DEZ1  TAY
F00B E90A     SBC #10         10'er Stellen abzaehlen
F00D 3003     BMI DEZ2
F00F EB          INX
F010 80F8     BRA DEZ1
F012 BD2BF0   DEZ2  LDA SEGTL,X
F015 8D0140   STA DISP10    linkes Display

```

```

F018 B92BF0      LDA SEGTBL,Y
F01B 8D0040      STA DISP1      rechtes Display
F01E 60          RTS
;
F01F A509  DENDSP LDA FURKEN      * Drive-Density zur Anzeige bringen *
F021 2903      AND #3
F023 AA        TAX
F024 BD3BF0      LDA DENSEG,X    Segmenttabelle fuer Density-Anzeige
F027 8D0240      STA DENSITY
F02A 60          RTS
;
Segment-Code Tabelle
;
F02B 3F065B4F SEGTBL .BYTE $3F,6,$5B,$4F,$66,$6D,$7D,7,$7F,$6F,$77,$7C,$39,$5E,$79,$71
F02F 666D7D07
F033 7F6F777C
F037 395E7971
;
Density-Code Tabelle
;
F03B 040201  DENSEG .BYTE 4,2,1
;
;
F03E A27E  BREAK  LDX #$7C      * Break-Routine *
F040 A050      LDY #$50      * Sollte der Prozessor auf einen BRK-Befehl stossen, *
F042 BE0140    STX DISP10   * wird ein 'br' + 2 x Bell ausgegeben und ein *
F045 8C0040    STY DISP1    * System-Warmstart ausgefuehrt *
F048 2068E0    JSR SYSERR
F04B A210      LDX #$10
F04D 20FB2E    JSR X2WAIT    Warteschleife
F050 4C73E0    JMP RESET2      Warmstart
;
;
High-Speed SIO-Routine, wird in relocierter Form zum Computer geschickt
;
F053      SIO      ; STATUS = $30
;                ; CHKSU2 = $31
;                ; BUF   = $32
;                ; LEN   = $34
;                ; CRETRY = $36
;                ; DRETRY = $37
;                ; STACKP = $3F
;
F053 AD0103    LDA $0301
F056 D009      BNE SIO2
F058 A204      LDX #4      bei Laufwerk Nr.=0
F05A 9D64F2  DLWTBLL STA LWTBL-1,X  Laufwerk-Tabelle loeschen
F05D CA        DEX
F05E DOFA      BNE DLWTBLL
F060 60        RTS
F061 AA  SIO2   TAX
F062 BD64F2  ABS21 LDA LWTBL-1,X  Laufwerk schon auf High-Speed gesetzt ?

```

F065 D034		BNE S103	
F067 FE64F2	ABS22	INC LWTBL-1,X	
F06A A207		LDX #7	
F06C BD0203	S10CL	LDA \$0302,X	S10-Kommando retten
F06F 48		PHA	
F070 BD59F2	ABS23	LDA C3F,X	
F073 9D0203		STA \$0302,X	COM3F - Tabelle kopieren
F076 CA		DEX	
F077 10F3		BPL S10CL	
F079 A931		LDA #\$31	
F07B 8D0003		STA \$0300	
F07E 2059E4		JSR \$E459	Computer S10 - Routine aufrufen
F081 A928		LDA #\$28	fuer ERROR normale Baudrate setzen
F083 AE0103		LDX \$0301	
F086 AC0303		LDY \$0303	
F089 3002		BMI S1021	
F08B A501		LDA \$01	
F08D 9D60F2	S1021	STA SPTBL-1,X	Baud-Rate in Speed-Tabelle eintragen
F090 A000		LDY #0	
F092 68	S1021CL	PLA	
F093 990203		STA \$0302,Y	urspruengliches S10-Kommando zurueckholen
F096 C8		INY	
F097 C008		CPY #8	
F099 90F7		BCC S1021CL	
F09B 78	S103	SEI	
F09C 8A		TXA	
F09D 0930		ORA #\$30	Drive # + Bus ID
F09F 8D3A02		STA \$023A	
F0A2 AD0203		LDA \$0302	
F0A5 8D3B02		STA \$023B	S10 - Kommando
F0A8 AD0A03		LDA \$030A	Sector L0
F0AB 8D3C02		STA \$023C	
F0AE AD0B03		LDA \$030B	Sector H1
F0B1 8D3D02		STA \$023D	
F0B4 8D60F2	ABS31	LDA SPTBL-1,X	
F0B7 8D04D2		STA \$D204	Baud-Rate setzen
F0BA BA		TSX	
F0BB 863F		STX \$3F	Stackpointer retten
F0BD A902		LDA #2	
F0BF 8537		STA \$37	2 Device-Retry's setzen
FOC1 A904	I011	LDA #4	
FOC3 8536		STA \$36	4 Command-Retry's setzen
FOC5 A934	I012	LDA #\$34	
FOC7 8D03D3		STA \$D303	Command-Leitung setzen
FOCA A900		LDA #0	
FOCC 8530		STA \$30	Status = 0
FOCE 853E		STA \$3E	
FODO 8535		STA \$35	
FOD2 8D06D2		STA \$D206	
FOD5 A93A		LDA #\$3A	
FOD7 8532		STA \$32	
FOD9 A902		LDA #2	

F0DB 8533		STA \$33	Buffer \$23A
F0DD 0A		ASL A	Laenge 4 Bytes
F0DE 8534		STA \$34	
F0E0 2027F1	ABS32	JSR SEND1	Buffer+Checksumme senden, auf Quittung warten
F0E3 AD0403		LDA \$0304	
F0E6 8532		STA \$32	Datenbuffer setzen
F0E8 AD0503		LDA \$0305	
F0EB 8533		STA \$33	
F0ED AD0803		LDA \$0308	
F0F0 8534		STA \$34	
F0F2 AD0903		LDA \$0309	Datenlaenge setzen
F0F5 8535		STA \$35	
F0F7 AD0303		LDA \$0303	Daten zum Laufwerk senden ?
F0FA 1003		BPL I02	
F0FC 2027F1	ABS33	JSR SEND1	Datenbuffer+Checksumme senden, auf Quittung warten
F0FF C63E	I02	DEC \$3E	
F101 20BEF1	ABS41	JSR SETT11	Ausfuehrungszeit begrenzen (Time-Out)
F104 2C0303		BIT \$0303	Daten vom Laufwerk uebernehmen ?
F107 5003		BVC I03	
F109 206CF1	ABS42	JSR GETA1	auf 'C' vom Laufwerk warten
F10C A9A0	I03	LDA #\$A0	
F10E 8D07D2		STA \$D207	Soundregister zuruecksetzen
F111 A510		LDA \$10	
F113 8D0ED2		STA \$D20E	Pokey-Maske zuruecksetzen
F116 20E6F1	ABS51	JSR CLRTI	
F119 A530		LDA \$30	
F11B F004		BEQ I04	Status OK ?
F11D C637		DEC \$37	Device-Retry abzaehlen
F11F D0A0		BNE I011	und ausfuehren
F121 A8	I04	TAY	
F122 8C0303		STY \$0303	Status setzen
F125 58		CLI	
F126 60		RTS	
i			
F127 A000	SEND1	LDY #0	
F129 C8	SE1	INY	
F12A D0FD		BNE SE1	
F12C A923		LDA #\$23	
F12E 2043F2	ABS61	JSR POKEY	Pokey auf senden stellen
F131 B132		LDA (\$32),Y	
F133 8531		STA \$31	
F135 8D0DD2		STA \$D20D	Pokey starten
F138 C8		INY	
F139 D011		BNE SE3	
F13B B132	SE2	LDA (\$32),Y	
F13D 2020F2	ABS62	JSR PUTBYTE	Buffer senden
F140 C8		INY	
F141 D009		BNE SE3	
F143 E633		INC \$33	
F145 C635		DEC \$35	
F147 A2E0		LDX #\$E0	
F149 E8	SEWL	INX	

```

F14A D0FD          BNE SEWL
F14C C434  SE3     CPY $34
F14E D0EB          BNE SE2
F150 A535          LDA $35
F152 D0E7          BNE SE2
F154 A531          LDA $31
F156 2020F2 ABS63  JSR PUTBYTE          Checksummen senden
F159 ADOED2 SED1   LDA $D20E
F15C 2908          AND #8
F15E D0F9          BNE SED1
F160 A003          LDY #3
F162 20E8F1 ABS64  JSR STOUTX0          Time-Out setzen
F165 A9C0          LDA #$C0
F167 8DOED2        STA $D20E          IRQ-Status zuruecksetzen
F16A D059          BNE RDQUIT
;
F16C A000  GETA1   LDY #0
F16E 8431          STY $31
F170 20FDF1 GE1    JSR GETBYTE          Datenblock vom Laufwerk holen
F173 9132          STA ($32),Y
F175 203BF2 ABS71  JSR ADDSUM
F178 C8           INY
F179 D004          BNE GE2
F17B E633          INC $33
F17D C635          DEC $35
F17F C434  GE2     CPY $34
F181 D0ED          BNE GE1
F183 A535          LDA $35
F185 D0E9          BNE GE1
F187 20FDF1 ABS72  JSR GETBYTE          Checksumme holen
F18A C531          CMP $31
F18C D00B          BNE ERR8F
F18E 60           RTS
;
F18F A980  IDER80  LDA #$80          Break-Status
F191 8530          STA $30
F193 A63F          LDX $3F
F195 9A           TXS          Stackpointer zuruecksetzen
F196 4C0CF1 EABS0  JMP I03
F199 A98F  ERR8F   LDA #$8F          Checksummen - Error
F19B 2C           .BYTE $2C
F19C A98A  ERR8A   LDA #$8A          Time-Out - Error
F19E 8530  ERROR   STA $30
F1A0 A63F          LDX $3F
F1A2 9A           TXS          Stackpointer zuruecksetzen
F1A3 A53E          LDA $3E
F1A5 3007          BMI ERRA
F1A7 C636          DEC $36
F1A9 F003          BEQ ERRA
F1AB 4CC5F0 EABS1  JMP I012
F1AE A928  ERRA   LDA #$28
F1B0 8D04D2        STA $D204          Baudrate normal setzen

```

```

F1B3 A900          LDA #0
F1B5 AE0103        LDX $0301
F1B8 9D64F2  EABS2  STA LWTBL-1,X      Eintrag in Laufwerktafel loschen
F1BB 4C0CF1  EABS3  JMP IO3
;
F1BE A201  SETT11  LDX #1
F1C0 A05E          LDY #$5E
F1C2 20EAF1  ABS81  JSR STOUT          Time-Out fuer Datenfeld setzen
;
F1C5 A93C  RDQUIT  LDA #$3C
F1C7 8D03D3        STA $D303
F1CA A913          LDA #$13
F1CC 2043F2  RDA1   JSR POKEY          Pokey auf lesen stellen
F1CF 20DFD1  RDA2   JSR GETBYTE       Quittung holen
F1D2 C941          CMP #$41          'A'
F1D4 F010          BEQ CLRTI
F1D6 C943          CMP #$43          'C'
F1D8 F00C          BEQ CLRTI
F1DA C945          CMP #$45          'E'
F1DC F004          BEQ ERR90
F1DE A98B          LDA #$8B          Drive not ready
F1E0 D0BC          BNE ERROR
F1E2 A990  ERR90  LDA #$90
F1E4 8530          STA $30          Status setzen
;
F1E6 A000  CLRTI  LDY #0          Timer loeschen
F1E8 A200  STOUTX0 LDX #0
F1EA ADFBF1  STOUT  LDA ERRABS
F1ED 8D2602        STA $0226          Timer 1 Sprungvector setzen
F1F0 ADFCF1  ST02   LDA ERRABS+1
F1F3 8D2702        STA $0227
F1F6 A901          LDA #1
F1F8 4C5CE4        JMP $E45C
F1FB 9CF1  ERRABS  .WORD ERR8A
;
F1FD A00ED2  GETBYTE LDA $D20E          1 Byte vom Pokey holen
F200 108D  JMPE80  BPL I0ER80
F202 2920          AND #$20          auf 'Shiftregister voll' warten
F204 D0F7          BNE GETBYTE
F206 A9DF          LDA #$DF
F208 8D0ED2        STA $D20E          IRQ - Flag zuruecksetzen
F20B A9E0          LDA #$E0
F20D 8D0ED2        STA $D20E          IRQ - Status neu setzen
F210 A00FD2        LDA $D20F
F213 8D0AD2        STA $D20A
F216 1084          BPL ERR8A
F218 2920          AND #$20
F21A F080          BEQ ERR8A
F21C A00DD2        LDA $D20D
F21F 60           RTS
;
F220 4B  PUTBYTE  PHA          1 Byte an den Pokey uebergeben

```

```

F221 A00ED2 PUTA1   LDA $D20E
F224 2910          AND #$10           Warten auf 'Shift - Register leer'
F226 D0F9         BNE PUTA1
F228 A9EF         LDA #$EF
F22A 8D0ED2       STA $D20E           IRQ - Flag zuruecksetzen
F22D A9D0         LDA #$D0
F22F 8D0ED2       STA $D20E           IRQ - Status neu setzen
F232 68           PLA
F233 8D0DD2       STA $D20D           Byte an Shift - Register uebergeben
F236 AE0ED2       LDX $D20E
F239 10C5         BPL JMPE80
;
F23B 18          ADDSUM   CLC           Checksumme errechnen
F23C 6531         ADC $31
F23E 6900         ADC #0
F240 8531         STA $31
F242 60           RTS
;
F243 8D0FD2 POKEY   STA $D20F           Pokey fuer Datenein- und ausgabe vorbereiten
F246 8D0AD2       STA $D20A
F249 A928         LDA #$28
F24B 8D08D2       STA $D208
F24E A9A8         LDA #$A8
F250 8D07D2       STA $D207           Sound - Register vorbereiten
F253 A9F8         LDA #$F8
F255 8D0ED2       STA $D20E           IRQ - Enable setzen
F258 60           RTS
;
COM3F - Tabelle
;
F259 3F400100 C3F  .BYTE 63,64,1,0,1,0,1,0
F25D 01000100
F261 28282828 SPTBL .BYTE 40,40,40,40           Baudraten - Tabelle fuer Laufwerke 1-4
F265 00000000 LWTBL  .BYTE 0,0,0,0           Pruef - Tabelle fuer Laufwerke 1-4

F269          SIDEND
;
Absolute Adressen Tabelle fuer Relocator
;
F269 5BF0   ABSTBL  .WORD DLWTBLL+1,ABS21+1,ABS22+1,ABS23+1,SID21+1,ABS31+1
F26B 63F0
F26D 68F0
F26F 71F0
F271 8EF0
F273 B5F0
F275 E1F0  .WORD ABS32+1,ABS33+1,ABS41+1,ABS42+1,ABS51+1,ABS61+1,ABS62+1
F277 FDF0
F279 02F1
F27B 0AF1
F27D 17F1
F27F 2FF1
F281 3EF1

```



```

F283 57F1      .WORD ABS63+1,ABS64+1,GE1+1,ABS71+1,ABS72+1,EABS0+1,EABS1+1,EABS2+1
F285 63F1
F287 71F1
F289 76F1
F28B 88F1
F28D 97F1
F28F ACF1
F291 B9F1
F293 BCF1      .WORD EABS3+1,ABS81+1,RDA1+1,RDA2+1,STOUT+1,STO2+1,ERRABS
F295 C3F1
F297 CDF1
F299 DOF1
F29B EBF1
F29D F1F1
F29F FBF1

```

;

Laufwerks - Testroutinen (werden in spaeteren Versionen erweitert)

;

```

F2A1 2050E5  ROMTST  JSR STELL2      * ROM-Testroutine *
F2A4 A200      LDX #0
F2A6 A9E0      LDA #E0
F2A8 851A      STA IND+1      ROM - Adresse $E000
F2AA A000      LDY #0
F2AC 8419      STY IND
F2AE 98        ROMT2   TYA
F2AF 18        CLC
F2B0 7119  ROMTL   ADC (IND),Y      Checksumme fuer 1 ROM-Page errechnen
F2B2 C8        INY
F2B3 D0FB      BNE ROMTL
F2B5 DDE0FE    CMP ROMCHK,X      Checksumme OK
F2B8 D009      BNE ROMTX
F2BA E61A      INC IND+1
F2BC EB        INX
F2BD E020      CPX #E20        32 Pages getestet ?
F2BF 90ED      BCC ROMT2
F2C1 6411      STZ COMST      OK - Status setzen
F2C3 4C55E5  ROMTX   JMP QUITT
;
F2C6 2050E5  RAMTST  JSR STELL2      * RAM-Testroutine *
F2C9 A955      LDA #E55
F2CB 8D009E    STA EXBUF
F2CE A000      LDY #0
F2D0 B90000  RT1L   LDA $00,Y
F2D3 8D019E    STA EXBUF+1
F2D6 AD009E    LDA EXBUF
F2D9 990000    STA $00,Y
F2DC D90000    CMP $00,Y      Zeropage testen
F2DF D03B      BNE ZPERR
F2E1 AD019E    LDA EXBUF+1
F2E4 990000    STA $00,Y
F2E7 C8        INY
F2E8 D0E6      BNE RT1L

```

```

F2EA 0E009E      ASL EXBUF
F2ED 90E1        BCC RT1L
;
F2EF A980        LDA #$80
F2F1 8591        STA $91          RAM - Adresse $8000 setzen
F2F3 A000        LDY #0
F2F5 8490        STY $90
F2F7 A955      RTABSL  LDA #$55
F2F9 8500        STA MERK1
F2FB B190      RT2L   LDA ($90),Y
F2FD 8501        STA MERK2
F2FF A500        LDA MERK1
F301 9190        STA ($90),Y
F303 D190        CMP ($90),Y      RAM von $8000 bis RAMTOP testen
F305 D017        BNE RTERR
F307 A501        LDA MERK2
F309 9190        STA ($90),Y
F30B C8         INY
F30C D0ED        BNE RT2L
F30E 0600        ASL MERK1
F310 90E9        BCC RT2L
F312 E691        INC $91
F314 A591        LDA $91
F316 C9A0        CMP # >CMTBL+$0100  RAMTOP erreicht ?
F318 90DD        BCC RTABSL
F31A B071        BRA SPCAX
;
F31C 6491      ZPERR  STZ $91          fehlerhafte RAM-Adresse zum senden zwischenspeichern
F31E 8490      RTERR  STY $90
F320 B06D      BRA SPTX
;
F322 2050E5    SPEEDT JSR STELL2      * Motor - Speed - Test *
F325 640D      STZ TRACK
F327 202FE3    JSR TRADJA      Track 0 positionieren
F32A 209BF3    JSR FDSEC1      Abstand zwischen SEctor 1 und Sector 1 testen
F32D B060      BCS SPTX
F32F 209BF3    JSR FDSEC1
F332 B05B      BCS SPTX
;
F334 6490      STZ $90
F336 6491      STZ $91          Counter loeschen
F338 A9E4      LDA #$E4
F33A 8592      STA $92
F33C A9E1      LDA #$E1
F33E 8593      STA $93
F340 A9C0      LDA #$C0
F342 8594      STA $94          von Konstante $E4E1C0 die gezaehlten Taktzyklen abzaehlen
F344 A594      SPCAL  LDA $94
F346 38        SEC
F347 E501      SBC MERK2
F349 8594      STA $94
F34B A593      LDA $93

```

F34D E502		SBC MERK3	
F34F 8593		STA \$93	
F351 B008		BCC SPCAD	
F353 A592		LDA \$92	
F355 E900		SBC #0	
F357 8592		STA \$92	
F359 9015		BCC SPCEND	
F35B 2060F3	SPCAD	JSR SPCADD	
F35E 80E4		BRA SPCAL	
;			
F360 FB	SPCADD	SED	Counter im Dezimalmodus heraufzaehlen
F361 A591		LDA \$91	
F363 18		CLC	
F364 6901		ADC #1	
F366 8591		STA \$91	
F368 A590		LDA \$90	
F36A 6900		ADC #0	
F36C 8590		STA \$90	
F36E D8		CLD	
F36F 60		RTS	
;			
F370 A594	SPCEND	LDA \$94	
F372 49FF		EOR #\$FF	
F374 8594		STA \$94	
F376 A593		LDA \$93	
F378 49FF		EOR #\$FF	
F37A 8593		STA \$93	
F37C 4602		LSR MERK3	
F37E A501		LDA MERK2	eine Stelle hinter dem Komma runden
F380 6A		ROR A	
F381 38		SEC	
F382 E594		SBC \$94	
F384 A502		LDA MERK3	
F386 E593		SBC \$93	
F388 9003		BCC SPCAX	
F38A 2060F3		JSR SPCADD	
;			
F38D 6411	SPCAX	STZ COMST	
F38F 2055E5	SPTX	JSR QUITT	Speed-Wert zum Computer senden
F392 A902		LDA #2	
F394 A290		LDX #\$90	
F396 A000		LDY #0	
F398 4C1BE5		JMP SDBTS	
;			
F39B A901	FDSEC1	LDA #1	
F39D 8D0204		STA \$0402	Sector # in Sector-Register des Controllers setzen
F3A0 A988		LDA #\$88	
F3A2 8D0004		STA \$0400	
F3A5 A9D8		LDA #\$D8	
F3A7 8D9F02		STA \$029F	Time-Out setzen
F3AA 6401		STZ MERK2	
F3AC 6402		STZ MERK3	Counter zuruecksetzen

```

F3AE A207          LDX #7
F3B0 CA          FDS1WL  DEX
F3B1 D0FD          BNE FDS1WL
F3B3 2C8002      FDS1L   BIT $0280
F3B6 5016          BVC FDS1T0
F3B8 3010          BMI FDS1DR
F3BA E601          INC MERK2
F3BC D004          BNE FDS1NI
F3BE E602          INC MERK3
F3C0 8002          BRA FDS1TZ
F3C2 48          FDS1NI  PHA
F3C3 68          PLA
F3C4 48          FDS1TZ  PHA
F3C5 68          PLA
F3C6 48          PHA
F3C7 68          PLA
F3C8 80E9          BRA FDS1L
;
F3CA 18          FDS1DR  CLC
F3CB 4C7BE3      JMP CONRES
;
F3CE 38          FDS1T0  SEC
F3CF 4C7BE3      JMP CONRE2
;

```

Zeit nach Maschinenzyklen (1 Mhz) festlegen

Time-Out - Error Kennzeichen

Freier ROM-Speicher fuer zukuenftige Erweiterungen

```

;
FD5C              *= $FE00

FE00 20FBE4      COM3F   JSR SEND41
FE03 A909          LDA #$09
FE05 203FE5      JSR SDBYTE
FE08 A909          LDA #$09
FE0A 203FE5      JSR SDBYTE
FE0D A901          LDA #1
FE0F 8515          STA USKEN
FE11 60           RTS
;
FE12 2C8002      NORDB   BIT $0280
FE15 501F          BVC RDBT0
FE17 2C8202      BIT $0282
FE1A 50F6          BVC NORDB
FE1C A206          LDX #6
FE1E CA          RDBL1   DEX
FE1F D0FD          BNE RDBL1
FE21 A980          LDA #$80
FE23 A207          RDNBIT  LDX #7
FE25 CA          RDBL2   DEX
FE26 D0FD          BNE RDBL2
FE28 2C8202      BIT $0282
FE2B 5003          BVC SETC

```

High-Speed Wert fuer Pokey an den Computer senden

Dateneuebertragung auf High-Speed stellen

* Read-Byte Routine fuer normale Uebertragungsrage *

auf Startbit warten

1 Bit uebernehmen

FE2D 18		CLC	
FE2E 9002		BCC RD8SB	
FE30 38	SETC	SEC	
FE31 EA		NOP	
FE32 6A	RD8SB	ROR A	
FE33 90EE		BCC RDNBIT	
FE35 60		RTS	
;			
FE36 68	RDBTD	PLA	Ruecksprungadresse vom Stack holen
FE37 68		PLA	
FE38 4C7EE4		JMP ER40UK	Kennung fuer Uebertragungsrate umschalten
;			
FE3B 2C8202	USRDB	BIT \$0282	
FE3E 50FB		BVC USRDB	
FE40 A208		LDX #8	
FE42 AD8202	USBITL	LDA \$0282	* 1 Datenblock in High-Speed vom Computer holen *
FE45 0A		ASL A	
FE46 0A		ASL A	
FE47 6600		ROR MERK1	
FE49 CA		DEX	
FE4A 10F6		BPL USBITL	
FE4C 8A	SVSDB	TXA	
FE4D 4500		EOR MERK1	
FE4F 2C8202	USW1	BIT \$0282	1 Bit uebernehmen
FE52 50FB		BVC USW1	
FE54 9119		STA (IND),Y	
FE56 18		CLC	
FE57 651B		ADC CHKSUM	Checksumme heraufzaehlen
FE59 6900		ADC #0	
FE5B 851B		STA CHKSUM	
FE5D A207		LDX #7	Bit - Zaehler fuer 8 Bits
FE5F AD8202	USW2	LDA \$0282	
FE62 0A		ASL A	
FE63 0A		ASL A	
FE64 6600		ROR MERK1	
FE66 CA		DEX	
FE67 10F6		BPL USW2	
FE69 C8		INY	
FE6A C413		CPY RWLEN	Checksumme empfangen ?
FE6C D0DE		BNE SVSDB	
FE6E 68	USRDBX	PLA	
FE6F 68		PLA	
FE70 8A		TXA	
FE71 4500		EOR MERK1	
FE73 4CE4E4		JMP RDEXIT	
;			
FE76 A901	NOSDB	LDA #1	* 1 Byte in Normal Speed an den Computer senden *
FE78 1C8202		TRB \$0282	Startbit setzen
FE7B DA		PHX	
FE7C 5A		PHY	
FE7D A008		LDY #8	8 Bit Zaehler
FE7F A205		LDX #5	

```

FE81 CA      SBWL1  DEX
FE82 DOFD           BNE SBWL1
;
FE84 AD8202  SDBITL  LDA $0282
FE87 4A           LSR A
FE88 4600           LSR MERK1      1 Bit ins PIO-Register shiften
FE8A 2A           ROL A
FE8B 8D8202           STA $0282
FE8E A205           LDX #5
FE90 CA      SBWL2  DEX
FE91 DOFD           BNE SBWL2
FE93 EA           NOP
FE94 EA           NOP
FE95 88           DEY
FE96 DOEC           BNE SDBITL      alle Bits gesendet
;
FE98 0901           ORA #1
FE9A 8D8202           STA $0282      Stopbit setzen
;
FE9D A203           LDX #3
FE9F CA      SBWL3  DEX
FEA0 DOFD           BNE SBWL3
FEA2 7A           PLY
FEA3 FA           PLX
FEA4 60           RTS
;
FEA5 AD8202  USSDB   LDA $0282
FEA8 29FE           AND #$FE      Startbit setzen
FEAA 8D8202           STA $0282
FEAD A208           LDX #8      Bitzaehler
FEAF 4A      USSBL  LSR A
FEB0 6600           ROR MERK1
FEB2 2A           ROL A      1 Bit ins PIO-Register shiften
FEB3 8D8202           STA $0282
FEB6 CA           DEX
FEB7 DOF6           BNE USSBL
FEB9 4A           LSR A
FEBA 38           SEC      Stopbit setzen
FEBB 2A           ROL A
FEBC C500           CMP $00
FEBE 8D8202           STA $0282
FEC1 60           RTS
;
;
FEC2           *= $FEE0
;
Checksummenbytes fuer Romtest
(werden bei'm assemblieren nicht richtig gesetzt)
;
FEE0 01020304 ROMCHK .BYTE 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16
FEE4 05060708
FEEB 090A0B0C

```

```

FEEC 0D0E0F10
FEFO 11121314      .BYTE 17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32
FEF4 15161718
FEF8 191A1B1C
FEFC 1D1E1F20

```

;

Vektortabelle

Von eigenen Programmen sollte nur ueber diese Vektortabelle gesprungen werden,
da die Programme sonst auf den folgenden Speedy-Versionen nicht laufen koennten.

;

* = \$FF00

;

FF00 4C00E0	JMP RESET	Drive Kaltstart
FF03 4C73E0	JMP RESET2	Warmstart
FF06 4CBCE0	JMP BEREIT	Bereitschaftsroutine
FF09 4C41E1	JMP MOTON	Motor zwingend einschalten
FF0C 4C3CE1	JMP TSTMON	Motor einschalten wenn Klappe zu ist
FF0F 4C2DE1	JMP MOTOFF	Motor ausschalten
FF12 4CD2E3	JMP SDELAY	Motor Timer stellen
FF15 4CA9E1	JMP SDRDDP	Drive Density stellen und anzeigen
FF18 4CF2E2	JMP XWAIT	Warteschleife kurz
FF1B 4CFBE2	JMP X2WAIT	Warteschleife lang
FF1E 4CD1E2	JMP TRACK0	Track 0 positionieren
FF21 4C2FE3	JMP TRADJA	Track # anzeigen und Kopf positionieren
FF24 4C32E3	JMP TRADJ	Kopf positionieren, Track # nur anzeigen, wenn Trackwechsel
FF27 4C0FE3	JMP TRVR	1 Step vorwaerts oder rueckwaerts gehen
FF2A 4C7BE3	JMP CONRES	Disk Controller stoppen
FF2D 4C78E3	JMP CONRE2	2 mal CONRES
FF30 4C85E3	JMP WREADY	auf Controller 'In Use'-Flag=0 warten
FF33 4CB3E4	JMP RD12BB	128 Bytes vom Computer nach EXBUF holen
FF36 4CB6E4	JMP RD256B	256 Bytes vom Computer nach EXBUF holen
FF39 4CBCE4	JMP RDBTS	Accu=Anzahl der Bytes nach Buffer (X/Y-Register) holen
FF3C 4CCDE1	JMP RDSFOL	nach Verzoegerung Sectorfolge vom aktuellen Track lesen
FF3F 4CD2E1	JMP RDSFO1	sofort Sectorfolge vom aktuellen Track lesen
FF42 4C6FE2	JMP RDTRA	alle Sektoren des aktuellen Track ins RAM einlesen
FF45 4C44E2	JMP RDTRAV	wie RDTRA aber mit Verify und einem Retry
FF48 4C64EB	JMP TSTWR	noch zu schreibende Sektoren aus RAM auf Diskette schreiben
FF4B 4C04E6	JMP TSTDAT	TSTWR ausfuehren und alle Sektoren als nicht gelesen markieren
FF4E 4C12E5	JMP SD12BB	128 Bytes vom EXBUF zum Computer senden
FF51 4C15E5	JMP SD256B	256 Bytes vom EXBUF zum Computer senden
FF54 4C1BE5	JMP SDBTS	Accu=Anzahl der Bytes aus Buffer (X/Y-Reg.) senden
FF57 4CFBE4	JMP SEND41	'A' zum Computer senden
FF5A 4CFFE4	JMP SEND43	'C' zum Computer senden
FF5D 4C03E5	JMP SEND45	'E' zum Computer senden
FF60 4C07E5	JMP SEND4E	'N' zum Computer senden
FF63 4CCFEA	JMP RDSECT	aktuellen Sector von Diskette in vorbezeichneten RAM einlesen
FF66 4CD4EA	JMP RDSECT1	bezeichneten Sector in bezeichneten RAM einlesen
FF69 4C69EC	JMP WRSECT	aktuellen Sector von vorbezeichneter RAM-Adr. auf Disk schreiben
FF6C 4C6EEC	JMP WRSECT1	bezeichneten Sector von vorbezeichneter RAM-Adr. schreiben
FF6F 4C88E4	JMP TSTWRP	Write Protect und Klappe testen
FF72 4CA4EC	JMP VERSECT	aktuellen Sector mit angegebenem RAM vergleichen
FF75 4CA9EC	JMP VERSECT1	bezeichneten Sector mit angegebenem RAM vergleichen

FF78 4C4CE5	JMP STELL	COM-Status auf 'Error' und 2 Retry's setzen
FF7B 4C55E5	JMP QUITT	Quittung 'C' oder 'E' je nach COM-Status senden
FF7E 4C3DEB	JMP RDHEAD	Die nachsten 'Header'-Daten lesen
FF81 4C42EB	JMP RDHD1	wie RDHEAD aber Timer nicht setzen
FF84 4C1DEB	JMP RDHDSP	Kopf positionieren und nachsten 'Header' lesen
FF87 4CE4E9	JMP CALCTS	Track- und Sectornummer errechnen
FF8A 4C1AEA	JMP SETBUF	Buffer nach aktuellem Sector setzen
FF8D 4C1CEA	JMP SETBUF2	Buffer nach Sectornummer im Accu setzen
FF90 4CDBE9	JMP SEXBUF	Adresse des Extended-Buffers setzen
FF93 4C2CEA	JMP SETRWL	Anzahl der Bytes fuer zu uebertragenden Datenblock setzen
FF96 4C68ED	JMP COPSLT	Sectorliste fuer aktuelles Density in Zeropage kopieren
FF99 4CB2EF	JMP BELL1	1 Bell (Buzzer) ausgeben
FF9C 4CC7EF	JMP CLRDSP	Display abschalten
FF9F 4CF6EF	JMP TRAAZ	aktuelle Track # anzeigen
FFA2 4C07F0	JMP DEZOUT	Wert im Accu in dezimaler Form anzeigen
FFA5 4CDFEF	JMP HEXOUT	Wert im Accu in Hexadezimaler Form anzeigen
FFA8 4C1FF0	JMP DENDSP	aktuelles Density anzeigen
FFAB 4C72E5	JMP SETTIM	Timer mit Wert im Accu setzen
FFAE 4C0FEF	JMP CLRTRA	Einen Track mit unlesbarem Format versehen (reformatieren)
FFB1 4C01EF	JMP CLRDSK	ganze Diskette reformatieren
FFB4 4CC6F2	JMP RAMTST	Einsprung fuer RAM-Test, 2 Bytes werden gesendet
FFB7 4CA1F2	JMP ROMTST	ROM-Test Einsprung, es wird nur quittiert
FFBA 4C22F3	JMP SPEEDT	Einsprung fuer Speed-Test, 2 Bytes werden gesendet

;

;

;

*=\$FFF8

.BYTE VERSION

.WORD RESET

.WORD BREAK

Speedy Versions Nummer

Reset - Vektor fuer den Prozessor

Break - Vektor fuer den Prozessor