Laufwerke

CP/M verwaltet bis zu 16 Laufwerke, die mit den Buchstaben A…P angesprochen werden. Die Laufwerksbuchstaben müssen nicht in alphabetischer Reihenfolge vorliegen; sie können beliebig vergeben werden. So gibt es häufiger das Laufwerk M: für eine RAM-Floppy.

Ein Laufwerk A: sollte aber immer vorhanden sein. Das originale Skeleton CBIOS stellt bei Warmstart Laufwerk 0 (A:) aktiv.

Sektoren

Aus CP/M-Sicht besteht ein Laufwerk zunächst nur aus **logischen Sektoren pro Laufwerk** (sogenannte Records, 128 Byte groß). CP/M zählt diese absoluten logischen Sektoren ab 0 .. Laufwerksgröße=Blockanzahl*Blockgröße/128.

Rec 0 | Rec 1 | ... | Rec nnn

Zur vereinfachten Umsetzung auf Diskettenlaufwerke und deren physikalischen Aufbau in Seiten, Spuren, und physikalische Sektoren rechnet das BDOS den absoluten logischen Sektor in **Spuren und relative logischen Sektoren** um.

In jeder Spur werden die einzelnen logischen Sektoren wieder ab 0 gezählt: Die Diskette wird unterteilt in Spuren 0..t-1 und jeweils einzelne logischen Sektoren (Records) 0..SPT-1.

OFF Anzahl der reservierten Spuren am Anfang der Diskette (engl. track OFFset)
SPT Anzahl der logischen Sektoren pro Spur (engl. Sectors Per Track)

Track OFF	Rec 0 | Rec 1 | ... | Rec SPT-1
Track OFF+1	Rec 0 | Rec 1 | ... | Rec SPT-1
	...
Track t	Rec 0 | Rec 1 | ... | Rec SPT-1

Das BDOS berechnet aus Block- und Recordnummer des FCB zuerst eine absolute logische Sektornummer der Diskette. Mit der Anzahl von logischen Sektoren pro Spur (SPT, siehe DPB) wird daraus die Spur- und logische Recordnummer innerhalb einer Spur bestimmt.

Aus **physikalischer Sicht** besteht eine Diskette aus Seiten, Spuren, physikalischen Sektoren.

Bei 8-Zoll-Disketten ist das Format fest vorgegeben: die physikalischen Sektoren sind 128 Byte groß, die Sektornummern laufen von 1..26.

Auf Mini- und Microdisketten sind die physikalischen Sektoren größer als 128 Byte. Häufig sind sie 512 Byte oder 1KByte groß. Physikalischen Sektoren zählen meist von 1..n. Die Sektornummern können beim Formatieren frei vergeben werden, sie wird auf der Diskette am Anfang eines jeden Sektors

gespeichert.

OFF Anzahl der reservierten Spuren am Anfang der Diskette (engl. track OFFset)
SPT Anzahl der logischen Sektoren pro Spur (engl. Sectors Per Track)

Track OFF	phys. Sektor 1 Rec 0 | Rec 1 | ... | Rec r	... ...	phys. Sektor n Rec m | Rec m+1 | ... | Rec SPT-1
Track OFF+1	phys. Sektor 1 Rec 0 | Rec 1 | ... | Rec r	... ...	phys. Sektor n Rec m | Rec m+1 | ... | Rec SPT-1
	...		
Track t	phys. Sektor 1 Rec 0 | Rec 1 | ... | Rec r	... ...	phys. Sektor n Rec m | Rec m+1 | ... | Rec SPT-1

Sektorversatz beim Formatieren: Die Sektoren können gleich mit Versatz auf der Diskette angelegt werden, so dass durch die Pause, die beim Verarbeiten der gelesenen Daten besteht und in dieser Zeit sich die Diskette auch weitergedreht hat, dann die hochgezählte nächste Sektornummer unter dem Lesekopf des Laufwerks angekommen ist und sofort gelesen werden kann. Im BIOS muss kein Sektorversatz beachtet werden.

Da dieser Sektorversatz rein durch das Laufwerk abgebildet wird, ändert sich auch am obigen Bild nichts, auch wenn auf der Diskette physisch die Sektoren in abweichender Reihenfolge vorliegen!

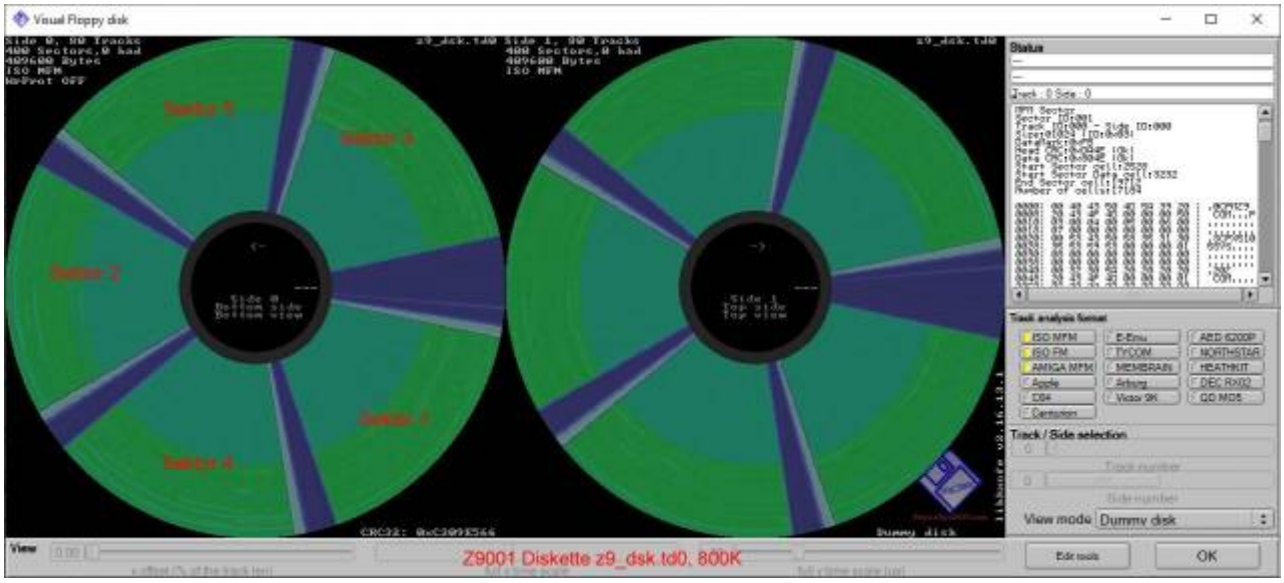
Beispiel:

Tool: HxCFloppyEmulator_soft.zip von https://hxc2001.com/download/floppy_drive_emulator/

Man startet HxCFloppyEmulator.exe und lädt das Disketten-Image (hier im Beispiel ein Teledisk-Format). Danach klickt man auf Track Analyzer. Eventuell muss die Ansicht angepasst werden: View-Mode: dummy disk. Mit der Maus kann dann die Diskette inspiziert werden.

Diskette: Z9001-Systemdiskette, 2 Seiten, 80 Spuren, 5 Sektoren, Sektorgröße 1024 Byte, 800K, keine Systemspuren.
Im Bild ist Spur 0, Seite 0, Sektor 1 ausgewählt. Er enthält den Beginn des Directories. Man erkennt die Verteilung der Sektoren auf der Diskette und ihre pysikalische Anordnung 1,4,2,5,3.

https://www.sax.de/~zander/z9001/tip/tipd.html, „Ein weiteres Diskettenabbild (TD0)“ z9_dsk.td0



Interleave/Skew

Bei 8-Zoll-Disketten ist dagegen ein Software-Sektorversatz üblich:

(orig. J. Plate, von mir korrigiert)

Beim Lesen von Diskette oder beim Schreiben auf Diskette tritt ein weiteres Problem auf. Angenommen, Sie wollen ein Programm mit 2 KByte Länge abspeichern, dann belegt das Programm 16 Sektoren. Das Betriebssystem ist aber nicht in der Lage, die 16 Sektoren in einem Zuge hintereinander zu lesen oder zu schreiben. Es ist ja allerhand zu berechnen und zu prüfen und bis z.B. Sektor 1 gelesen und geprüft ist, hat sich die Diskette weitergedreht und es befindet sich schon Sektor 3 oder Sektor 4 unter dem Lesekopf. Es muss also für jeden eine ganze Umdrehung der Platte abgewartet werden, was die Zugriffszeit drastisch anhebt.

Um diesen Nachteil zu vermeiden, kann ein Trick angewendet werden: Die Sektoren werden nicht mehr hintereinander beschrieben oder gelesen, sondern es werden andere dazwischengeschoben.

Beispiel:
Alte Anordnung:
12 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
Neue Anordnung:
1 10 6 15 2 11 7 16 3 12 8 17 4 13 9 18 5 14

Nun hat sich der Abstand zwischen den Sektoren erhöht, und damit kann innerhalb einer Umdrehung mehr als nur ein Sektor gelesen werden, wenn die Rechenzeit kleiner als der Zeitabstand zwischen den jetzt logisch aufeinanderfolgenden Sektoren ist. Es gibt nun mehrere Möglichkeiten, eine solche Anordnung zu erreichen. Zum einen könnte mit der verwendeten Floppy durch spezielle Formatierung eine solche Numerierung erreicht werden. In CP/M aber gibt es im BIOS eine Übersetzungstabelle, die jedem ankommenden logischen Sektor des CP/M-Systems einen physikalischen Sektor zuweist, wie ihn das Floppy-System versteht. Eine solche Tabelle könnte wie folgt aussehen:

1 5 9 13 17 3 7 11 15 2 6 10 14 18 4 8 12 16

Nehmen wir einmal an, wir wollten zunächst Sektor 0 schreiben oder lesen (logischer Sektor). Dann

nehmen wir den ersten Eintrag unserer Tabelle und erhalten den Sektor 1 als physikalischen Sektor. Nun wollen wir den nächsten logischen Sektor schreiben oder lesen, also Sektor 1. In der Tabelle steht an der zweiten Stelle der Wert 5. Wir greifen damit also in Wirklichkeit auf den Sektor 5 (selbe Spur) auf der Diskette zu. Zwischen Sektor 1 und Sektor 5 liegt der Abstand 3 Sektoren, es bleibt also Rechenzeit zwischen den Sektoren 0 und 1 (logische Sektoren), da wir in Wirklichkeit die Sektoren 1 und 5 verwendet haben. Mit den anderen Sektoren verhält sich dies ganz analog. Diese Tabelle wird auch Sektorsprungtabelle genannt. Der Abstand der Sektoren ist der sogenannte Interleaving-Faktor.

In unserem Beispiel war es der Wert 4. Aus dem Wert 4 läßt sich die Tabelle eindeutig aufbauen. Dazu wird bei Sektor 1 begonnen. Dann wird 4 addiert, und es ergibt sich als zweiter Eingang der Wert 5; dann nochmals, und es ergibt sich 9, dann 13, dann 17 - und was nun? 21 gibt es nicht, also minus 18 rechnen, damit ergibt sich Jetzt geht es weiter mit 7, 11, 15 und dann 19, 19 - 18 ergibt 1, aber den Sektor 1 gab es schon in der Tabelle. Nun wird nach dem nächsten nicht in der Tabelle schon vorhandenen Wert gesucht und es ergibt sich der Sektor 1. Dann wird wieder fortgefahren, bis schließlich alle Sektor-Zuordnungen ermittelt sind. Zur Konstruktion dieser Tabelle (wie auch der anderen Tabellen) gibt es zu CP/M die Macrobibliothek DISKDEF.LIB, siehe dazu diskdeflib.

Der BIOS-Funktion SECTRAN werden logische Recordnummer (innerhalb einer Spur) und die Adresse der Sektorversatztabelle übergeben (s.u.). Die BIOS-Funktion muss daraus den physikalischen Sektor berechnen:

aus A Skeletal CBIOS:

```
  249                   SECTRAN:
  250                       ;TRANSLATE THE SECTOR GIVEN BY BC USING THE
  251                       ;TRANSLATE TABLE GIVEN BY DE
  252   4BA7 EB             XCHG            ;HL=.TRANS
  253   4BA8 09             DAD    B        ;HL=.TRANS (SECTOR)
  254   4BA9 6E             MOV    L, M        ;L=TRANS (SECTOR)
  255   4BAA 2600           MVI    H, 0        ;HL=TRANS (SECTOR)
  256   4BAC C9             RET            ;WITH VALUE IN HL
```

Die Macrobibliothek DISKDEF.LIB erzeugt bei weniger als 256 Sektoren/Track eine Byte-Liste, bei mehr als 256 Sektoren würde eine Liste mit 16-Bit-Werten erzeugt. Die Bios-Funktion SECTRAN muss dann auch darauf ausgelegt sein.

Fazit Beim Software-Sektorversatz würfelt das BIOS die Verwendung der physikalischen Sektoren durcheinander. Diese liegen physikalisch in Reihenfolge 1,2,3,... auf Diskette vor, aber werden inhaltlich nicht mehr in dieser Reihenfolge genutzt. Man muss die Translate-Tabelle kennen, um die Daten wieder lesen zu können. Im Gegensatz dazu übernimmt beim Hardware-/Formatier-Sektorversatz der Floppycontroller das Lesen in logischer Reihenfolge.

BIOS-Funktionen

Die Sektor-Verschränkungs-Tabelle (XLT, s.o.) dient zur Umrechnung von logischen zu physikalischen Sektornummern einer Spur. Die Länge dieser Tabelle entspricht der Anzahl logischer Sektoren die im DPB definiert sind.

Das BDOS berechnet aus Block- und Recordnummer des FCB eine absolute logische Sektornummer

der Diskette.

Mit der Anzahl von logischen Sektoren pro Spur (siehe DPB) kann daraus die Spur- und Sektornummer bestimmt werden.

Das BIOS erhält die Information, wo konkret auf die Datenspeicher zugriffen werden soll, via BIOS-Funktion

- SETDSK (Laufwerk, 0…15),

sowie

- SETTRK (Track, Spur OFF…x, berechnet aus Blocknummer + Systemspuren OFF),
- SECTRAN (Transformation der aktuellen log. Record-Nummer 0…BSM, Sektorversatz, skewing)
- und SETSEC (Sector, transformierte Record-Nummer).

Das BDOS ruft die letzten drei genannten BIOS-Funktionen nur innerhalb der BDOS-Routine SEEK und alle vier immer **vor** Aufruf von READ oder WRITE in dieser Reihenfolge auf. Ein BIOS speichert daher normalerweise die Werte für Laufwerk, Spur, Sektor zwischen und greift bei Lese- und Schreiboperationen darauf zu, sie könnten aber auch gleich direkt an den Laufwerkscontroller übergeben werden.

Pro Format/Laufwerkstyp gibt es dazu üblicherweise im BIOS eine Funktion zur Adressberechnung, die aus logischer Spurnummer und transformierter Record-Nummer die physisch anzusprechenden Werte ermittelt. Bei Disketten sind das Seite, Spur, phys. Sektor, Pos. innerhalb des physikalischen Sektors.

BDOS22

```
SEEK:
        ...
        CALL    SETTRKF    ;track set up
        ...
        LD    HL,(TRANV)   ; XLT aus DBH
        EX    DE,HL    ;BC=sector#, DE=.tran
        CALL    SECTRAN    ;HL = tran(sector)
        LD    C,L
        LD    B,H    ;BC = tran(sector)
        JP    SETSECF    ;sector selected
```

SECTRAN

SECTRAN übernimmt eine Umrechnung der logischen Sektornummer pro Spur in eine transformierte Nummer. Bei 8-Zoll-Floppies mit Software-Sektorversatz erfolgt das über eine Versatz-Tabelle, bei anderen Geräten oder Systemen wird einfach der übergebene Sektor unverändert zurückgegeben oder aber um 1 erhöht.

SECTRAN Logisch-zu-physischer Sektor wurde durchgeführt, um die allgemeine Reaktion von CP/M zu verbessern. Standard-CP/M-Systeme nutzen einen Skew-Faktor 6, wo sechs physische Sektoren zwischen jeder logischen Leseoperation übersprungen werden. Der Skew-Faktor lässt zwischen den Sektoren genügend Zeit für die meisten Programme, um ihre Puffer zu laden, ohne den nächsten Sektor zu vermissen.

in: BC = logischer Sektor 0..
DE = XLT-Translate-Tabelle, 0, wenn es keine Tabelle gibt
out: HL = transformierter Sektor

Das folgende Beispiel stammt aus einer CP/A-Implementation.
Im CP/A gibt es keine Translate-Tabellen, da die Sektornummernverwaltung verallgemeinert im nicht-Standard-DPB enthalten ist (auch fuer physische Sektorlaenge <>128). Es wird im BIOS immer ab Sektor 1 gezählt. Das bringt hier Vorteile bei der weiteren Bearbeitung. Deshalb steht in sectran ein inc hl. Die Zählung ab 1 muss bei den direkten Zugriffen (Adressberechnung, Read, Write) beachtet werden!

```
sectran:    ld    h, b
        ld    l, c
        inc    hl        ;Sektoren zaehlen in CP/A ab 1
        ret
```

Ein SECTRAN-Funktion für Software-Sektor-Versatz bei 8-Zoll-Floppies steht bereits weiter oben.


Da weder Translate-Tabelle noch die physikalische Sektornummer im BDOS gebraucht und daher auch nicht genutzt wird, kann man die Konvertierung Logisch-zu-physischer Sektor auch anders lösen und das Feld XLT des DPHs für eigene Zwecke nutzen.