Kassettenformate

Das Kassetten-Magnetband-Interface des Z1013 ist ein Diphase-Verfahren. Es kam speziell auf dem Z1013 zum Einsatz und ist auf anderen Computern nicht verbreitet.

Aufgezeichnet wird in Blöcken zu je 32 Byte. Jeder Block besteht aus einem Kopf: einem einzelnen Word 0000h, sowie den Daten; gespeichert als 10h Words. Anschließend folgt eine Prüfsumme (wieder ein Word) über den Datenblock. Tatsächlich werden also pro Block 36 Byte aufgezeichnet.

Programm:

<ditaa noedgesep> +——-+——-+——-+——-+ +——-+

Vorton	Block	Block	Block	...	Block

+——-+——-+——-+——-+ +——-+ </ditaa>

Physisches Aufzeichnungsformat

Die Aufzeichnung erfolgt in Blöcken zu je 32 Datenbytes. Jeder Block hat folgenden Aufbau:

<ditaa noedgesep> +——-+————-+————-+————-+————+

Vorton	Trennschwingung	Blocknummer	Datenbereich	Pruefsumme

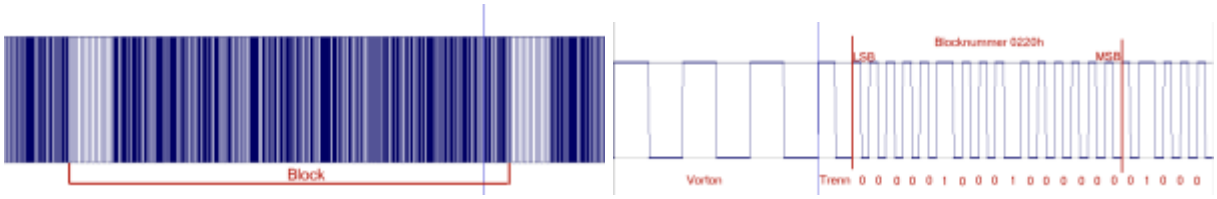
+——-+————-+————-+————-+————+ </ditaa>

1. Vorton: 14 Halbschwingung a 640 Hz, beim ersten Block 2000 Schwingungen
2. Trennschwingung: 1 Vollschwingung a 1280 Hz
3. Blocknummer: 16 Bit, Bedeutung siehe „logisches Aufzeichnungsformat“
4. Datenbereich: 32 Bytes in Form von 16 Datenwörtern
5. Prüfsumme: 16-Bit-Addition über die Blocknummer und die 16 Datenwörter

Die Datenwörter sind Little-Endian-kodiert, d.h. niederwertiges Byte zuerst. Es wird jeweils das Bit 0 zuerst gespeichert.

Bit-Codierung: 0-Bit: 1 Vollschwingung mit 2560 Hz (2 Phasenwechsel nach jeweils 0,39 ms) 1-Bit: 1 Halbschwingung mit 1280 Hz (1 Phasenwechsel nach 0,78 ms)

Mit Blocknummer und Prüfsumme zusammen besteht ein Block damit aus 36 Byte (= 18 Word).



Es werden immer vollständige Blöcke aufgezeichnet, auch wenn die angegeben Endadresse in der

Mitte eines Blocks liegt. Zwischen zwei Blöcken, d.h. zwischen dem letzten Phasenwechsel der Prüfsumme und dem ersten Phasenwechsel des Vortons, gibt es eine etwa 2,5 ms lange Pause.

Damit passen ca. **400 KByte** auf eine Kassettenseite (30 min)!

Logisches Aufzeichnungsformat

Original Z1013

Die Aufzeichnung enthält ausschließlich den zu sichernden Speicherbereich ohne jegliche Verwaltungsinformationen.

Die Blocknummer wird außer zur Berechnung der Prüfsumme inhaltlich nicht verwendet und kann deshalb jeden beliebigen Wert enthalten. Die originalen Monitorprogramme schreiben als Blocknummer immer 0000h.

Block:

<ditaa noedgesep> +—–+——-+—-+——-+—-+

0000	word0	...	word15	cks

+—–+——-+—-+——-+—-+ </ditaa>

Headersave

Beim Headersave-Format wird ein **zusätzlicher** Kopfblock mit Verwaltungsinformationen (Anfangs-, End- und Startadresse, Dateityp, Headersave-Kennung, Dateiname) vorangestellt. Dieser Kopfblock und der erste nachfolgende Datenblock haben einen langen Vorton, die anderen Datenblöcke einen kurzen Vorton.

<ditaa noedgesep> +—-+—-+—-+—-+—-+—-+—-+—-+—-+—-+—-+—-+—-+—-+—-+—-+

00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
aadr	eadr	sadr	frei/Autor/CRC	Typ	D3	D3	D3								

+—-+—-+—-+—-+—-+—-+—-+—-+—-+—-+—-+—-+—-+—-+—-+—-+—-+

10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E	1F
Programmname, mit Leerzeichen aufgefuellt															

+———————————————————————-+ </ditaa>

Aufbau des Kopfblocks:

Byte	**Bedeutung**
0, 1	Anfangsadresse der Datei
2, 3	Endadresse

Byte	Bedeutung
4, 5	Startadresse bei Fall eines ausführbaren Programms (Dateityp: „C“)
6-11	frei (wird manchmal benutzt für CRC oder Autor)
12	Dateityp (z.B. „C“: Ausführbares Maschinencodeprogramm)
13-15	Headersave-Kennung (3x D3h)
16-31	Dateiname, mit Leerzeichen aufgefüllt

Die Blocknummer hat bei Headersave eine inhaltliche Bedeutung, anhand derer man die Blockreihenfolge überprüfen kann. Als Blocknummer wird die jeweilige Blockanfangsadresse verwendet:

Block	Blocknummer
Kopfblock	00E0h (Anfangsadresse des Kopfpuffers)
1. Datenblock	Dateianfangsadresse
2. Datenblock	Dateianfangsadresse + 20h
3. Datenblock	Dateianfangsadresse + 40h
…	u.s.w.

Die Blocknummer 0FFFFh wird als Endeblock erkannt.

Aufgrund der Kompatibilität zum orignalen Format kann eine Headersave-Aufzeichnung mit den originalen Monitorprogrammen geladen werden, wenn das Laden erst **nach** dem Kopfblock gestartet wird.

<ditaa noedgesep> +——-+——-+——-+——-+——-+——-+ +——-+

vorton	Header	vorton	Block	Block	Block	…	Block

+——-+——-+——-+——-+——-+——-+ +——-+ </ditaa>

Tiny-Basic

Auch beim **TINY-BASIC** (originales CSAVE-Kommando) wird ein zusätzlicher Header geschrieben. Dieser hat einen einfacheren Aufbau als der von Headersave. Das Tiny-BASIC-Verfahren stand Pate für das von R. Brosig entwickelte Headersave.

Tiny-Basic-Kopf

<ditaa noedgesep> +—-+—-+—-+—-+—-+—-+—-+—-+—-+—-+—-+—-+—-+—-+—-+—-+

00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
aadr	eadr	alles 00													

+—-+—-+—-+—-+—-+—-+—-+—-+—-+—-+—-+—-+—-+—-+—-+—-+

10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E	1F
Programmname, mit Leerzeichen aufgefuellt															

+————————————————————————-+ </ditaa>

Weitere

Neben dem Hausformat gibt es auch 10K-BASIC-Programme im Kassettenaufzeichnungsformat des Z9001 sowie Basicode-Programme im BASICODE3-Kassettenaufzeichnungsformat.

Formate der Emulatoren

*.**z13** Originalformat des Z1013
also Bin-Datei (ohne Kopf) oder Tiny-Basic (Tiny-Basic-Programme haben Kopf ähnlich wie Headersave: nur aadr (1000), eadr und filename)
kein Headersave-Kopfblock

*.**z80** Header-Save-Programme
32 Byte Vorblock (aadr, eadr, sadr, 6 byte frei, typ, 3x 0d3h, 16 Zeichen Filename, mit Leerzeichen aufgefüllt. Die 6 freien Byte enthalten den (Programm-)Autor, (z.B. Brosig), Müll, eine CRC-Summe oder andere Identifikatoren, sind aber nicht notwendig)

Sowohl bei *.z13 als auch bei *.z80 werden die Kopf-Daten der Blöcke nicht gespeichert. Die *.z13-Dateien sind damit reine Speicherdumps (außer Tiny-Basic, auch hier gibt es einen Header), bei *.z80 kommt noch der Headersave-Kopf davor.

Gebräuchlich sind eigentlich nur *.z80-Dateien.

*.**TAP** Arne Fitzenreiter:

- 16 byte Header mit „KC-TAPE by AF“
- 129 byte Blöcke mit Blocknummer aber ohne Prüfumme
- nur für wenige 10K-BASIC-Programme genutzt; gebräuchlicher war HSAVE (Headersave-Aufzeichnung als *.Z80-Datei)

(vp, jmue)

Einlesen am PC



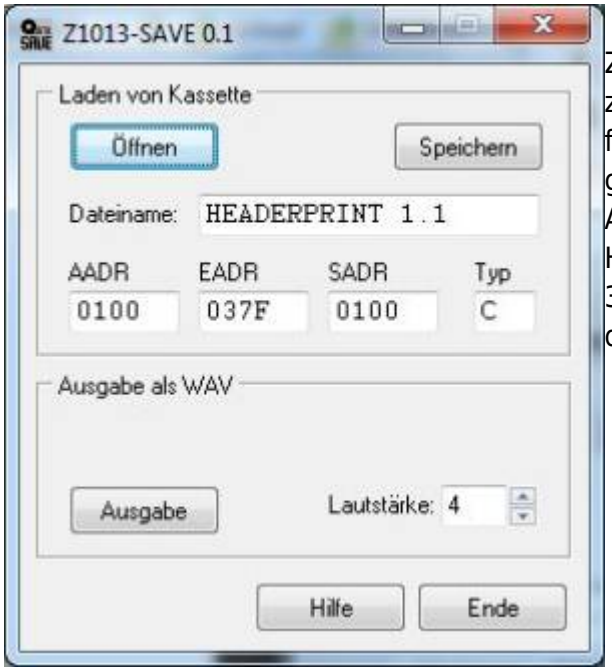
Zum Einlesen der Kassetten am PC gibt es das Programm **KCLOAD** von H. Haftmann (http://www-user.tu-chemnitz.de/~heha/hs_freeware/kcemu/). Das ist ein extrem kleines Windows-16Bit-Programm, das aber auch unter Windows 7 (32 Bit) noch lauffähig ist und zum Einlesen einer Vielzahl von Kassettenformaten genutzt werden kann.

Zum automatisierten Einlesen von 54 übergebenen Z1013-Kassetten habe ich Version 02/02 des Programms ein wenig modifiziert, so dass „automatisch speichern“ auch wirklich funktioniert. Wegen diverser Probleme mit langen Dateinamen werden als Programmnamen dabei einfach hochgezählte Nummern vergeben. Zwei kurze Perl-Programme **rename_z80.pl** und **rename_z13.pl** benennen die eingelesenen Dateien dann in einem Rutsch um und entfernen auch gleich noch Duplikate.

Nun braucht man bloß noch eine Kassette einlegen – nach einer halben Stunde hat man 20..60 Programme eingelesen!

- kcload_auto.zip + Perl-Programme

Ausgeben vom PC



Zum Ausgeben von .Z80-Headersave-Dateien auf den Z1013 müssen die Binärdaten wieder in Audiosignale zurückverwandelt werden. Im obigen KCLOAD funktioniert bei mir unter Windows 7 die Audio-Ausgabe gar nicht mehr; unter Windows 98 funktioniert die Audioausgabe, allerdings gibt KCLOAD keinen Headersave-Vorblock aus. Deshalb gibt es ein kleines 32-Bit-Programm **ZSAVE** von mir. Über Speichern kann das Audiosignal als WAV-Datei gespeichert werden.

- zsave.rar Programm incl. Quellen (Delphi 5)

From:
https://hc-ddr.hucki.net/wiki/ - **Homecomputer DDR**

Permanent link:
https://hc-ddr.hucki.net/wiki/doku.php/z1013/kassettenformate?rev=1444294290

Last update: **2015/10/08 08:51**