

**SDHC\_1.1**

<b>COLLABORATORS</b>
----------------------

	<i>TITLE :</i> SDHC_1.1		
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>	<i>SIGNATURE</i>
WRITTEN BY		August 22, 2024	

<b>REVISION HISTORY</b>
-------------------------

NUMBER	DATE	DESCRIPTION	NAME

# Contents

<b>1</b>	<b>SDHC_1.1</b>	<b>1</b>
1.1	xpkSDHC V1.1 . . . . .	1
1.2	einleitung . . . . .	1
1.3	benutzung . . . . .	2
1.4	kodierung . . . . .	2
1.5	komprimierung . . . . .	3
1.6	arbeitsmodi . . . . .	3
1.7	effiency . . . . .	3
1.8	vergleich . . . . .	4
1.9	vergleich2 . . . . .	5
1.10	geschichte . . . . .	7
1.11	todo . . . . .	8
1.12	autor . . . . .	8
1.13	disclaimer . . . . .	8

# Chapter 1

## SDHC\_1.1

### 1.1 xpkSDHC V1.1

xpkSDHC.library V1.1

Inhaltsverzeichnis

-----

Einleitung

Benutzung

Effektivität

Vergleich

Geschichte

Zu Tun

Autor

Disclaimer

### 1.2 einleitung

xpkSDHC.library (V1.1)

-----

Diese Library ist eine Standard-Sublibrary für das XPK-System. Sie wurde geschrieben, um insbesondere Samples (fast) aller Arten besser als mit den schon vorhandenen Sublibraries zu komprimieren. Neben der Unterstützung für 8-bit-Samples, wie z.B. im IFF-8SVX-Format, lassen sich auch 16-Bit-Samples, z.B. im AIFF-Format, komprimieren. SDHC ist dabei die Abkürzung für "Sample Delta Huffman Compression".

Das H für Huffman ist ein Relikt aus Version 1.0, inzwischen wird für die Komprimierung statt xpkHFMN die xpkGZIP.library eingesetzt.

Die xpkSDHC.library liefert im allgemeinen gleiche bis bessere Ergebnisse als die xpkSMPL.library, ist allerdings deutlich langsamer beim Komprimieren.

Darüber hinaus kann sie auch die xpkDLTA.library ersetzen, da die Daten auch kodiert, aber unkomprimiert gespeichert werden können.

Weiter

## 1.3 benutzung

Benutzung

-----

Um die Sublibrary zu benutzen, muß sie nur, wie bei XPK üblich, in das Verzeichnis Libs:Compressors kopiert werden. Danach ist sie sofort verfügbar. Für die Benutzung der Sublibrary wird Kickstart V2.0 oder besser vorausgesetzt. Um die komprimierenden Modi von SDHC zu nutzen, muß die xpkGZIP.library korrekt installiert sein. Die z.Z. aktuellste Version im Aminet ist V1.0.

Der grundsätzliche Arbeitsablauf der Library ist wie folgt:

Dateneingabe -> **kodieren** -> **komprimieren** -> Datenausgabe

Dabei läßt sich einstellen, wie die Daten kodiert werden sollen. Weiterhin läßt sich einstellen, ob die Daten komprimiert werden sollen.

Weiter

## 1.4 kodierung

Datenkodierung

Die Sublibrary bietet verschiedene Modi für die verschiedene Sample-Arten.

Dies wären:

Datenart Beispielformat Modus Bezeichnung

8 Bit IFF-8SVX B Byte

16 Bit AIFF W Word

16 Bit, Stereo AIFF SW Stereo Word

In jedem dieser Modi werden die Daten Delta-kodiert. Hier bietet SDHC zwei Stufen an: einfach oder doppelt Delta-kodiert. Dies wird im Mode-Descriptor durch eine "1" oder "2" angegeben. Die doppelte Delta-Kodierung verbessert bei einigen Sampledaten die Komprimierung um bis zu 10%. Um z.B. die xpkDLTA.library zu ersetzen, ist der Modus "B1" auszuwählen.

Weiter

---

## 1.5 komprimierung

### Datenkomprimierung

SDHC verwendet für die Datenkomprimierung die xpkGZIP.library. Diese Sublibrary ist im Aminet zu finden (util/pack/xpkGZIP). Ist GZIP nicht installiert, gibt SDHC die Fehlermeldung "Can't find required XPK library" zurück.

Um Daten zu komprimieren, ist ein Modus auszuwählen, in dessen Mode-Descriptor ein "P" enthalten ist.

Weiter

## 1.6 arbeitsmodi

### Übersicht der Arbeitsmodi

Die Tabelle gibt Modusnummer und -descriptor an, wie sie auch von xQuery ermittelt werden.

Modusnummer Modedescriptor ausführlich

0 .. 7	B1 Byte, einfach Delta
8 .. 15	B2 Byte, doppelt Delta
16 .. 23	W1 Word, einfach Delta
24 .. 31	W2 Word, doppelt Delta
32 .. 39	SW1 Stereo Word, einfach Delta
40 .. 47	SW2 Stereo Word, doppelt Delta
48 .. 55	P B1 Packed, Byte, einfach Delta
56 .. 63	P B2 Packed, Byte, doppelt Delta
64 .. 71	P W1 Packed, Word, einfach Delta
72 .. 79	P W2 Packed, Word, doppelt Delta
80 .. 87	PSW1 Packed, Stereo Word, einfach Delta
88 .. 100	PSW2 Packed, Stereo Word, doppelt Delta

Weiter

## 1.7 effiency

### Effektivität

-----

Die Effektivität einer Datenkomprimierung anzugeben, ist etwas kompliziert. Eine Datei ist nämlich dann ideal komprimiert, wenn sie nur noch durch ein Bit (!) repräsentiert wird. Dies ist dann der Fall, wenn die Daten mit den Referenzdaten der Komprimierung übereinstimmen. Da mit SDHC jedoch beliebige Sampledaten komprimiert werden sollen, sind die Komprimierungsraten naturgemäß schlechter als im Idealfall.

Die Deltakodierung ist ein relativ einfacher Rechenvorgang, der sehr schnell abgearbeitet wird. Für die Komprimierung ist die xpkGZIP.library zuständig. Daher sind die Geschwindigkeit und Komprimierungsraten von dieser Sublibrary abhängig. Die von xQuery ausgegebenen Werte sind nicht überzubewerten. Erstens habe ich keinen A3000/25 MHz und zweitens besitze ich nicht das Programm "AmigaVision". Die jetzigen Werte habe ich ermittelt, indem ich xBench mit dem Programm "PPaint V7.1" gefüttert habe. Falls mir jemand die korrekten Werte auf dem Referenzsystem des XPK-Systems bestimmen könnte, wäre ich ihm sehr dankbar.

Weiter

## 1.8 vergleich

Fakten, Fakten, Fakten

-----

Erst einmal die vollständige Ausgabe von xQuery:

Packer : SDHC

Name : Sample Delta Huffman Compressor

Descr. : Coder for audio data

DefMode: 50 DefChunk: 128 Kb MaxChunk: 1023 Kb

Pack Unpack Pack Unpack

Name Mode Ratio Speed Speed Mem Mem Description

SDHC 0..7 0.0% 1150K/s 1300K/s 0K 0K B1

SDHC 8..15 0.0% 700K/s 800K/s 0K 0K B2

SDHC 16..23 0.0% 1700K/s 1800K/s 0K 0K W1

SDHC 24..31 0.0% 1050K/s 1200K/s 0K 0K W2

SDHC 32..39 0.0% 1700K/s 1800K/s 0K 0K SW1

SDHC 40..47 0.0% 1050K/s 1200K/s 0K 0K SW2

SDHC 48..55 37.0% 15K/s 200K/s 256K 128K P B1

SDHC 56..63 28.9% 15K/s 200K/s 256K 128K P B2

SDHC 64..71 29.7% 15K/s 200K/s 256K 128K P W1

SDHC 72..79 20.4% 15K/s 200K/s 256K 128K P W2

SDHC 80..87 27.8% 15K/s 200K/s 256K 128K PSW1

SDHC 88..100 16.1% 15K/s 200K/s 256K 128K PSW2

Die Angaben für den Speicherbedarf sind als Richtwerte zu sehen, die für den DefChunk-Größe gelten. Der genaue Bedarf hängt von der Chunkgröße ab. Beim Komprimieren wird 2\*Chunkgröße, beim Dekomprimieren 1\*Chunkgröße benötigt.

Weiter

## 1.9 vergleich2

Vergleich mit anderen Sublibrarys

-----  
Für den Vergleich wurden verschiedene Samples verwendet und alle Daten mit xBench bestimmt. Der für den Test verwendete Rechner ist ein A500, ausgestattet mit 68030+68882/50 MHz, 1+16 MB RAM, Festplatte, CD-ROM und ZIP-Laufwerk. Alle Daten befanden sich in der RAMDisk.

Als Konkurrenten wählte ich aus:

Typ Zweck

RAKE.100 allgemein

SMPL 8-Bit-Samples

SQSH 8-Bit-Samples

GZIP.70 allgemein (Stufe 70 wird auch in SDHC verwendet)

Test 1

einfaches Testfile, Inhalt ist nicht wichtig

Type Num Version P CSize CTime CSpd USize UTime USpd Rate

NONE: 100 1.2 568256 0.12 4733900 568068 0.12 4733900 0.0

SDHC: 7 1.1 568196 0.24 2366950 568068 0.21 2705085 0.0

SDHC: 15 1.1 568196 0.41 1385531 568068 0.36 1577966 0.0

SDHC: 23 1.1 568196 0.17 3341576 568068 0.15 3787120 0.0

SDHC: 31 1.1 568196 0.27 2103955 568068 0.23 2469860 0.0

SDHC: 39 1.1 568196 0.17 3341576 568068 0.15 3787120 0.0

SDHC: 47 1.1 568196 0.27 2103955 568068 0.23 2469860 0.0

Dieser Test soll die Geschwindigkeit der Delta-Routinen zeigen. Immerhin erreichen sie bis zu 3/4 der Geschwindigkeit der Kopierroutine von NONE.

Test 2

AIFF-File: 617454 Bytes, 16 bit per sample, 2 tracks, 22050 Hz

(Ausschnitt aus Oasis: Wonderwall, "I don't believe...")

Type Num Version P CSize CTime CSpd USize UTime USpd Rate

RAKE: 100 1.7 617632 3.07 201125 617454 0.18 3430300 0.0

SMPL: 100 1.0 617640 2.00 308727 617454 0.14 4410385 0.0

SQSH: 100 1.10 617652 9.29 66464 617454 0.13 4749646 0.0

GZIP: 70 1.0 587720 23.06 26775 617454 2.51 245997 4.9

SDHC: 55 1.1 618112 20.82 29656 617454 0.66 935536 0.0

SDHC: 63 1.1 618112 20.96 29458 617454 0.81 762288 0.0

SDHC: 71 1.1 573956 24.42 25284 617454 2.70 228686 7.1

SDHC: 79 1.1 595048 24.58 25120 617454 2.78 222105 3.7

SDHC: 87 1.1 555084 25.11 24589 617454 2.71 227842 10.2



SDHC: 100 1.1 550268 25.37 24337 617454 2.80 220519 10.9

Wie man gut erkennt, greift hier der StereoWord-Modus von SDHC besonders gut, die doppelte Delta-Kodierung bringt noch mal knapp 1%. Der einzige Komprimierer im Vergleich, der außer SDHC die Daten noch komprimiert, ist GZIP selbst.

Test 3

IFF-8SVX-File: 220548 Bytes, 8 bit per sample, 1 track, 22050 Hz

(Ausschnitt aus Rammstein: Engel, 10 s vom Anfang)

Type Num Version P CSize CTime CSpd USize UTime USpd Rate

RAKE: 100 1.7 75700 2.10 105022 220548 0.14 1575342 65.7

SMPL: 100 1.0 106868 0.53 416128 220548 0.26 848261 51.6

SQSH: 100 1.10 76928 4.90 45009 220548 0.24 918950 65.2

GZIP: 70 1.0 64612 5.99 36819 220548 0.42 525114 70.8

SDHC: 55 1.1 67088 7.64 28867 220548 0.49 450097 69.6

SDHC: 63 1.1 71032 10.30 21412 220548 0.57 386926 67.8

SDHC: 71 1.1 85540 6.05 36454 220548 0.55 400996 61.3

SDHC: 79 1.1 99456 6.26 35231 220548 0.66 334163 55.0

SDHC: 87 1.1 104224 5.79 38091 220548 0.64 344606 52.8

SDHC: 100 1.1 141760 6.59 33467 220548 0.82 268960 35.8

Gegenüber den direkten Konkurrenten SMPL und SQSH zeigen sich hier Vorteile bei der Komprimierung, allerdings bei geringerer Geschwindigkeit. GZIP selbst ist noch etwas besser, was an der geringen Aussteuerung des Samples liegen kann.

Test 4

AIFF-File, 441054 Bytes, 16 Bit per Sample, 2 tracks, 22050 Hz

(5 Sekunden Sprache)

Type Num Version P CSize CTime CSpd USize UTime USpd Rate

RAKE: 100 1.7 394016 2.32 190109 441054 0.23 1917626 10.7

SMPL: 100 1.0 415012 1.41 312804 441054 0.16 2756587 6.0

SQSH: 100 1.10 404960 10.72 41143 441054 0.40 1102635 8.2

GZIP: 70 1.0 354152 16.65 26489 441054 1.68 262532 19.8

SDHC: 55 1.1 405380 15.37 28695 441054 0.93 474251 8.1

SDHC: 63 1.1 406992 14.39 30650 441054 0.75 588072 7.8

SDHC: 71 1.1 346484 16.88 26128 441054 1.78 247783 21.5

SDHC: 79 1.1 360988 16.80 26253 441054 1.86 237125 18.2

SDHC: 87 1.1 341648 17.03 25898 441054 1.78 247783 22.6

SDHC: 100 1.1 345724 17.04 25883 441054 1.83 241013 21.7

Auch hier bieten sich, entsprechend den vorliegenden Daten, die Word-Modi von SDHC an. Der geringe Unterschied zwischen Mono- und Stereo-Kodierung liegt wohl in der hohen Korrelation der beiden Tonspuren.

Test 5

---

PGM-Bild, 262159 Byte, 512\*512 Pixel, 8 bit Graustufe

Type Num Version P CSize CTime CSpd USize UTime USpd Rate

RAKE: 100 1.7 193184 2.94 89169 262159 0.25 1048636 26.4

SMPL: 100 1.0 133288 0.68 385527 262159 0.33 794421 49.2

SQSH: 100 1.10 141976 4.25 61684 262159 0.30 873863 45.9

GZIP: 70 1.0 176984 10.36 25304 262159 0.92 284955 32.5

SDHC: 55 1.1 146056 13.03 20119 262159 0.92 284955 44.3

SDHC: 63 1.1 158176 10.93 23985 262159 1.05 249675 39.7

SDHC: 71 1.1 159940 11.09 23639 262159 0.96 273082 39.0

SDHC: 79 1.1 179012 9.84 26642 262159 1.11 236179 31.8

SDHC: 87 1.1 172732 10.48 25015 262159 1.02 257018 34.2

SDHC: 100 1.1 194736 9.85 26615 262159 1.18 222168 25.8

In verschiedenen Seminaren an der Uni hatte ich PGM-Bilder zu bearbeiten. Bei der Archivierung hatte ich bisher SMPL benutzt. Leider kommt SDHC nicht an das Ergebnis von SMPL heran. Die Delta-Kodierung der Daten verbessert aber die Komprimierung durch GZIP um immerhin 12%. Auch DMCB liefert hier gute Ergebnisse - allerdings bei ca. 2 KByte pro Sekunde.

Bemerkungen

Leider erhielt ich zur Version 1.0 von SDHC Fehlermeldungen, die von zerstörten Daten berichteten. Leider konnte ich die Ursache dafür nicht entdecken. Nach dem Wechsel von HFMN zu GZIP als Komprimierer erhielt ich keine Fehlermeldungen mehr. Leider arbeitet GZIP deutlich langsamer als HFMN, bringt aber meistens bessere Ergebnisse. Alte mit HFMN gepackte Daten lassen sich trotzdem dekomprimieren, da dies über die xpkmaster.library abgewickelt wird.

Weiter

## 1.10 geschichte

Geschichte

-----

Version Datum

1.0 08.03.98 erste Version

1.1 30.03.98 fehlendes Bit in PackerInfo gesetzt

Deltaroutinen überarbeitet

HFMN durch GZIP ersetzt aufgrund nicht nachvollziehbarer

Fehlermeldungen

Weiter

## 1.11 todo

Noch zu tun:

-----

- korrekte Werte für xQuery ermitteln (lassen)
- Vorschläge: ???

Ob hier erwähnte Punkte realisiert werden, hängt stark von meinen zeitlichen Möglichkeiten und meiner Motivation ab.

Weiter

## 1.12 autor

Über mich

-----

Ich freue mich über jede Anmerkung und jeden Vorschlag. Bugmeldungen sind zwar kein Grund zur Freude, werden aber trotzdem bearbeitet.

Gerd Wieczorek

SMail: Berliner Str.1

14959 Trebbin

BRD

EMail: gwieifjc@sp.zrz.tu-berlin.de

(1x wöchentlich kontrolliert)

Weiter

## 1.13 disclaimer

Disclaimer

-----

Dieses Programm ist Freeware. Über Geschenke (fast) jeder Art freut sich der **Autor** trotzdem.

Es gelten die allgemein üblichen Hinweise:

Die Benutzung der xpkSDHC.library erfolgt auf eigene Gefahr. Für Schäden, die sich aus der Benutzung von SDHC ergeben, bin ich nicht haftbar zu machen.

Wenn jemand SDHC in einer anderen Art und Weise als per Aminet (-CD) verbreiten will, muß er mich vorher per e-mail kontaktieren.

Zurück

---