

SDHC_1.1

COLLABORATORS

	TITLE : SDHC_1.1		
ACTION	NAME	DATE	SIGNATURE
WRITTEN BY		August 22, 2024	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1	SDHC_1.1	1
1.1	xpkSDHC V1.1	1
1.2	introduction	1
1.3	usage	2
1.4	coding	2
1.5	compression	2
1.6	modes	3
1.7	effiency	3
1.8	comparision	4
1.9	comparision2	4
1.10	history	7
1.11	todo	7
1.12	author	8
1.13	disclaimer	8

Chapter 1

SDHC_1.1

1.1 xpkSDHC V1.1

xpkSDHC.library V1.1

Contents

Introduction

Usage

Efficiency

Comparision

History

ToDo

Author

Disclaimer

1.2 introduction

xpkSDHC.library (V1.1)

This library is a standard sublibrary for the XPK-system. It was programmed to compress especially samples of nearly all kinds better than already available sublibraries. Besides the compression of 8-bit-samples like IFF-8SVX you can also compress 16-bit-Samples, e.g. in AIFF-Format. SDHC is an abbreviation of "Sample Delta Huffman Compression".

The H for Huffman is somewhat obsolete, for compression GZIP is used instead of HFMN.

The xpkSDHC.library normally achieves equal or even better results than xpkSMPL.library, but with major decrease in speed. You also could replace xpkDLTA.library, because it's possible to save the coded, but uncompressed data.

In the following text most times I used only the 4-letter-name (SDHC) of a sublibrary instead of the whole name like xpkSDHC.library.

[Next](#)

1.3 usage

Usage

To use SDHC you must copy the sublibrary to the Libs:Compressors directory.

After doing this you can use it immediately.

To run SDHC Kickstart V2.0 or better is required. To use the compressing modes of SDHC you also have to install the xpkGZIP.library. The newest version of xpkGZIP.library available in Aminet is V1.0.

The normal working process of SDHC is:

Data input -> **coding** -> **compressing** -> Data output

You can use different kinds of coding. You can also switch the compression off.

[Next](#)

1.4 coding

Data coding

SDHC provides different modes for different kinds of samples.

These are:

type of data example mode name

8 bit IFF-8SVX B byte

16 bit AIFF W word

16 bit, stereo AIFF SW stereo word

Each of these three modes the data will be delta-coded. Here you can choose between two levels: simple or double delta-coding. In the mode-descriptor this is visualized by "1" or "2". Sometimes the double delta-coding improves the compression up to 10%. To replace the xpkDLTA.library you have to choose the mode "B1", which means 8-bit data with simple delta-coding.

[Next](#)

1.5 compression

Data Compression

SDHC uses the xpkGZIP.library for compression. This sublibrary can be found in Aminet (util/pack/xpkGZIP). If the sublibrary isn't installed, SDHC answers with the XPK-errormessage "Can't find required XPK library".

To compress data you have to choose a mode, which has a "P" in it's descriptor.

[Next](#)

1.6 modes

Modes of Operation

The table gives you the number und description of all modes, like xQuery shows you.

number	description in words
0 .. 7	B1 byte, simple delta
8 .. 15	B2 byte, double delta
16 .. 23	W1 word, simple delta
24 .. 31	W2 word, double delta
32 .. 39	SW1 stereo word, simple delta
40 .. 47	SW2 stereo word, double delta
48 .. 55	P B1 packed, byte, simple delta
56 .. 63	P B2 packed, byte, double delta
64 .. 71	P W1 packed, word, simple delta
72 .. 79	P W2 packed, word, double delta
80 .. 87	PSW1 packed, stereo word, simple delta
88 .. 100	PSW2 packed, stereo word, double delta

[Next](#)

1.7 efficiency

Effiency

To define the efficiency of a data compression is rather complicated. A file is compressed ideal, if it is represented by only 1 bit (!). That is the case, if the data are equal to the reference-data of the compression-scheme. Because SDHC should compress any sample-data, the compressionrate can't be compared with the ideal case.

The delta coding is a relatively simple calculation, which is done very fast.

The xpkGZIP.library is responsible for compression. Because of that speed and compression-rate depends on this sublibrary.

The values xQuery gives you aren't of much use. I don't own neither a A3000/25 MHz nor the program "AmigaVision". The present values are found out by using xBench with the program-file "PPaint V7.1". If there is anyone, who could make the standard test of the XPK-system and send me the results, I would be glad.

[Next](#)

1.8 comparision

Facts

At first the complete output of 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

The values for required memory are given for working on DefChunk-sized blocks.

The exact memory requirement depends on the chunksize. When compressing, SDHC needs 2*chunksize and on decompression 1*chunksize.

Next

1.9 comparision2

Comparision with other Sublibraries

For comparision different data/samples are used. All tests are done with

xBench. The computer used is an Amiga 500, equipped with 68030+68882/50 MHz, 1+16 MB RAM, harddisk, CD-ROM and ZIP-drive. All data are copied to RAM-disk.

As competitors I choosed:

type purpose

RAKE.100 all kind of data

SMPL 8-Bit-Samples

SQSH 8-Bit-Samples

GZIP:70 all kind of data (mode 70 is used in SDHC too)

Test 1

Simple testfile, content doesn't matter

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	

This test should show the speed of the delta-coding routines. Speed varies from a third up to three quarter of speed of simple copying, like NONE does.

Test 2

AIFF-file: 617454 bytes, 16 bit per sample, 2 tracks, 22050 Hz

(part of 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	

As you could see, the "PSW"-modes of SDHC are the best in the test. The double delta-coding gives an advantage of nearly 1%. The only compressor other than SDHC, which gives a compression at all, is GZIP.

Test 3

IFF-8SVX-file: 220548 bytes, 8 bit per sample, 1 track, 22050 Hz

(part of Rammstein: Engel, 10s from beginning)

Type	Num	Version	P	CSize	CTime	CSpd	USize	UTime	USpd	Rate
RAKE:	100	1.7	75700	2.09	105525	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	4.88	45194	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

Against the direct competitors, SMPL and SQSH, SDHC shows a better compression, but is also slower. GZIP without Delta-coding is even better, perhaps because this sample is very quiet.

Test 4

AIFF-file, 441054 bytes, 16 bit per sample, 2 tracks, 22050 Hz
 (5 seconds of speech)

	Type	Num	Version	P	CSize	CTime	CSpd	USize	UTime	USpd	Rate
RAKE:	100	1.7	394016	2.27	194296	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	10.74	41066	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		

This test shows the importance to choose the right mode for the data. According to the data you have to use the word-modes of SDHC. The minimal difference between mono- and stereo-word-coding is probably due to a high correlation of both sound channels.

Test 5

PGM-picture, 262159 bytes, 512*512 pixel, 8 bit grayscale

	Type	Num	Version	P	CSize	CTime	CSpd	USize	UTime	USpd	Rate
RAKE:	100	1.7	193184	2.97	88269	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

At the university I had to work with some PGM-pictures. When archiving them I used normally SMPL, sometimes also SQSH. Unfortunately SDHC doesn't reach the results of SMPL or SQSH. But the Delta-coding improves the compression of GZIP by 12%. Another good compressor for this type of data is DMCB - but with only 2 KByte per second.

Note

For V1.0 of SDHC I got some error-messages, but I was unable to find the reason for this. After changing from HFMN to GZIP as compressor, I never got any error-messages. Unfortunately GZIP works much more slowly than HFMN, but on the other side most times it gives you a better compression. Old files packed with SDHC/HFMN are also unpacked with this version, because decompression is done by xpkmaster.library.

[Next](#)

1.10 history

History

version date

1.0 08.03.98 first version

1.1 30.03.98 set a missing bit in PackerInfo-structure

reworked Delta-coding-routines

replaced HFMN by GZIP because of some strange

error-messages

[Next](#)

1.11 todo

ToDo-List

- (let) determine correct values for xQuery

- other suggestions: ???

- only English AmigaGuide: May someone tell me the worst grammar mistakes?

(of course with the right phrase, I wish to learn English! :-)

It depends on my motivation and spare time, if and when I will program these points.

[Next](#)

1.12 author

About me

If you have any questions, remarks, suggestions or even bug reports please let me know.

Gerd Wieczorek

SMail: Berliner Str.1

14959 Trebbin

BRD

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

(checked only once per week, so don't await immediate response)

[Next](#)

1.13 disclaimer

Disclaimer

This program is freeware, but gifts of (nearly) all kinds are welcome by the [author](#).

The entire risk as to the use of the program and the information presented is assumed by the user. In no event will I be liable for any damage resulting from any claim arising of the use of the program or the information presented herein, even if I have been advised of the possibilities of such damages.

If anyone wants to include SDHC into another archive, he should send me an e-mail before.

[Back to Contents](#)
