INTERNATIONAL ORGANISATION FOR STANDARDISATION
ORGANISATION INTERNATIONALE DE NORMALISATION
ISO/IEC JTC1/SC29/WG11
CODING OF MOVING PICTURES AND AUDIO

ISO/IEC JTC1/SC29/WG11
MPEG2005/N7687
October 2005, Nice, France

Source: **Audio Subgroup**
Title: **Verification Report on MPEG-4 SLS**
Status: **Approved**

1 Introduction

MPEG-4 SLS provides a scalable lossless extension of AAC. The lossless enhancement is done in a fine-grain scalable way, allowing for signal representations from the quality of the AAC core to numerically lossless, including near-lossless signal representation. It also works as a standalone lossless coder when the AAC core is not present.

In this document the performance of this new codec is reported in terms of coding efficiency, complexity, and audio quality at near-lossless operation.

2 SLS Architectures

The SLS algorithm [1][3] is a scalable transform-based coder, providing a gradual refinement of the description of the transform coefficients, starting with perceptually weighted reconstruction levels provided by the AAC core bitstream, up to the resolution of the original input signal. In order to achieve lossless reconstruction, the SLS decoder uses an inverse Integer MDCT (IntMDCT) transform, and deterministic versions of several tools used for AAC-core bitstream decoding (see Fig.1). SLS does also provide a non-core mode operation where the AAC-core bitstream is not present in the lossless bit-stream. The AAC encoder and decoder are not required for this non-core mode operation. (Fig.2)

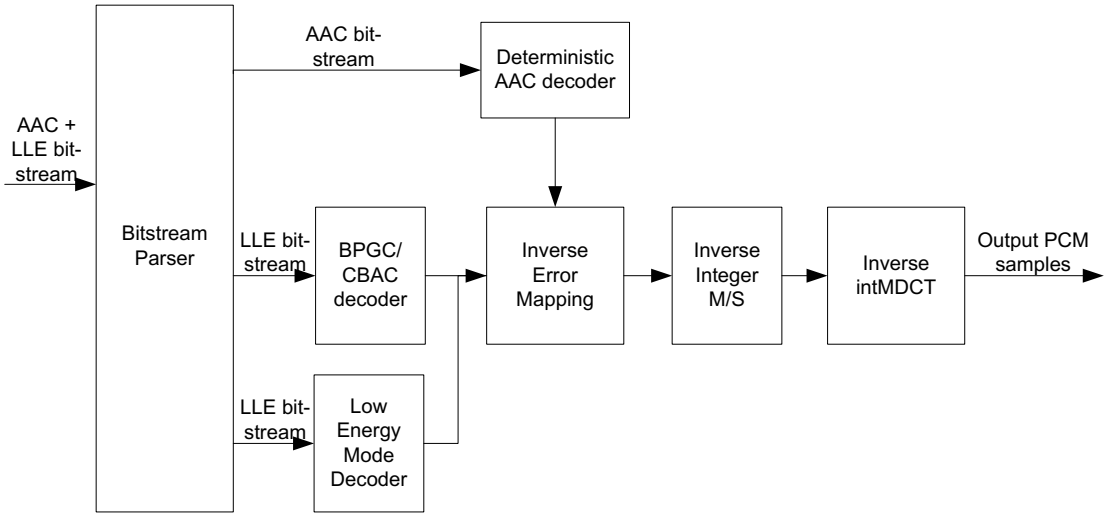


Figure 1. SLS decoder block diagram.

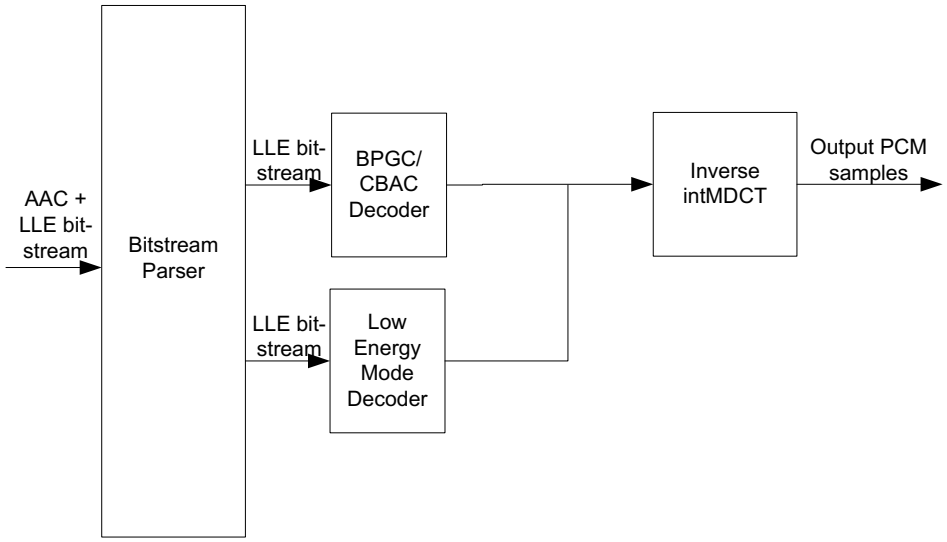


Figure 2. SLS non-core decoder block diagram

3 Performance and Complexity of SLS

3.1 Criteria and Test Methodologies

We evaluate the compression ratio performance for SLS by using the MPEG-4 lossless audio coding testing sets donated by Matsushita Corporation [2], and 20 commercial music CDs. Details of these testing items can be found at Appendix A. In our result, all compression results are given in terms of either Compression Ratio

$$Compression\ Ratio = \frac{OriginalSize}{CompressedSize},$$

or Compression Size

$$Compression\ Size = 100 \times \frac{CompressedSize}{OriginalSize}\%,$$

wherever they fit into.

The *computational complexity* for SLS is evaluated by counting the total numbers of standard instructions (multiplications, additions, bit-shifts, comparisons, memory transfers, etc) required for performing the decoding process on a generic 32-bit fixed-point CPU.

For the purpose of comparison we compute a weighted average number (with weights: 14.0 – multiplications, 56.0 – divisions, 4.0 – shifts, and 0.5 – for additions, comparisons, and memory-access operations) corresponding to the estimated latencies of instructions in Intel Pentium processors [4]. Additional complexity data is provided in Appendix C.

The complexity of deterministic algorithms is evaluated exactly, while the complexity of arithmetic coding engines is estimated as the upper bound for the average complexity, assuming it achieves compression ratio of 2:1. Estimation of memory requirement of the SLS implementation in terms of *ROM* usage is also given.

The functionalities for SLS are evaluated in terms of lossless coding capability, scalability, backward compatibility, and random access.

3.2 Coding efficiency

In our test, coding efficiency performance of SLS is evaluated for both AAC core based and non-core mode with the following settings

1. SLS with AAC core – the bit-rate of the AAC core bit-stream is 64 kbps/channel and the oversampling factors (osf) are 1, 2 and 4 respectively for testing sequences with 48kHz, 96kHz, and 192 kHz sampling rate. Hence the AAC core always works at 48kHz sampling rate in this evaluation.

2. SLS non-core mode (osf=4).

The results for the averaged coding efficiency performance for the MPEG-4 lossless audio coding testing sequences are listed in Table 1, and further visualized in Figure 3. Performance for each individual testing sequence can be found in Appendix B.

Table 1. Coding efficiency of SLS for audio signals with different sampling rate and wordlength

	SLS + AAC @ 64kbps/channel (AAC at 48kHz sampling rate)		SLS non-core (osf=4)	
	Compression Ratio	Compression size	Compression Ratio	Compression Size
48kHz16Bit	2.09	47.83%	2.20	45.44%
48kHz24Bit	1.55	64.65%	1.58	63.12%
96kHz24Bit	2.09	47.77%	2.13	46.87%
192kHz24Bit	2.60	38.44%	2.63	38.08%

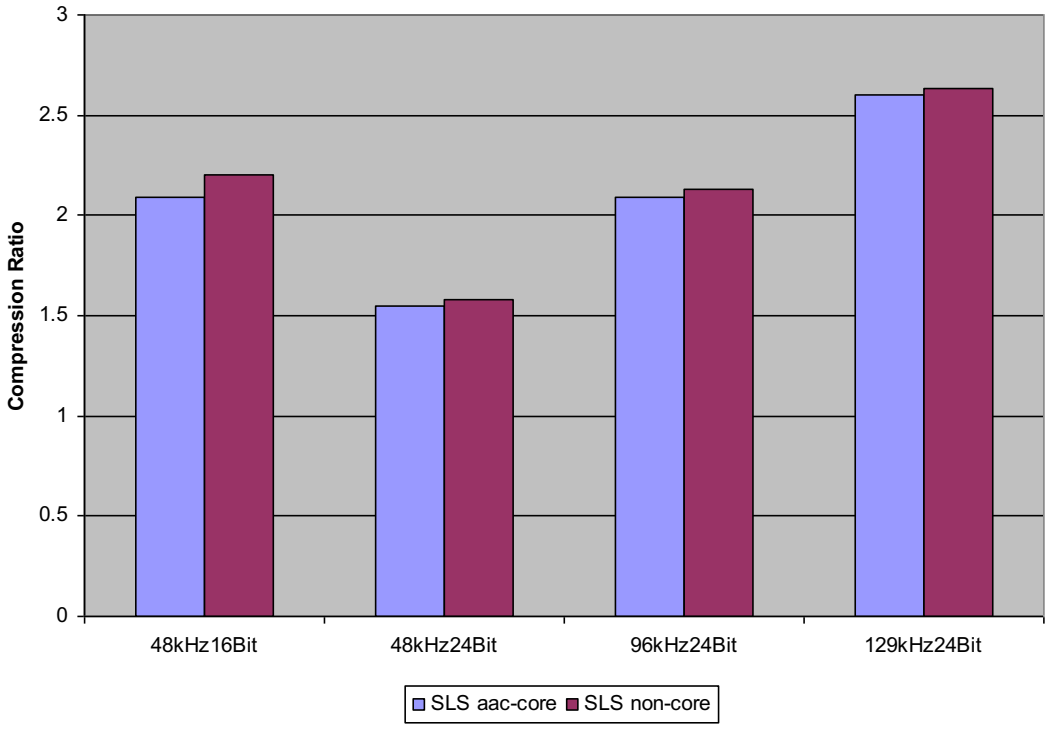


Figure 3. Compression ratio performance of MPEG-4 SLS

Coding efficiency results for a set of 20 commercial CDs can be found in Appendix B.

3.3 Complexity

In the following table, the complexity in terms of combined Pentium latency per decoded audio sample of the SLS decoder is listed. To facilitate DSP and other potential embedded implementation of SLS only the estimated maximum value of complexity is reported here. Detailed analysis of the complexity of SLS can be found in Appendix C.

Table 2. Complexity of SLS decoder in terms of combined Pentium latency (cycle/sample)

Transform length	SLS + AAC @ 64kbps/channel	SLS non-core
4096/512 (osf=4)	729.655	677.575
2048/256 (osf=2)	724.415	672.335
1024/128 (osf=1)	642.29	590.21

To facilitate any potential comparative study between SLS and other lossless audio coding schemes, in the following table, the compression size performances of SLS from Section 3.2.1 and their corresponding complexity is explicitly listed in the following table.

Table 3. Complexity and Compression of SLS decoder

	SLS + AAC @ 64kbps/channel		SLS non-core	
	Compression size	Combined Pentium latency (cycle/sample)	Compression size	Combined Pentium latency (cycle/sample)
48kHz16Bit	47.83%	642.29	45.44%	677.575
48kHz24Bit	64.65%	642.29	63.12%	677.575
96kHz24Bit	47.77%	724.415	46.87%	677.575
192kHz24Bit	38.44%	729.655	38.08%	677.575

3.4 ROM usage requirements

Memory requirements of an SLS decoder are summarized in the following table. Detailed analysis can be found at Appendix C

Table 4. ROM requirement for SLS decoder

	SLS no core	SLS + AAC core
ROM	4 K bytes	45 K bytes

4 Functionalities

SLS provides fine-grain scalable extension of MPEG-4 AAC. The feature sets of SLS are presented in the following table.

Table 5. Functionalities of SLS

	SLS AAC core	SLS non-core
Numerically lossless coding	Yes	Yes

Scalability	Yes	Yes
Backward compatibility	Yes	No
Support for multi-channel audio	Yes	Yes
Random Access	Minimum possible granularity for RA: 48 kHz: 21.3 ms 96 kHz: 10.6 ms 192kHz: 5.3 ms RA granularity used in above evaluation: 48 kHz: 21.3 ms 96 kHz: 21.3 ms 192kHz: 21.3 ms	Minimum possible granularity for RA: 48 kHz: 21.3 ms 96 kHz: 10.6 ms 192kHz: 5.3 ms RA granularity used in above evaluation: 48 kHz: 85.3 ms 96 kHz: 42.7 ms 192kHz: 21.3 ms

5 Intermediate quality

MPEG-4 Audio provides numerous tools for coding audio signals from very low bit rates (e.g. 2 to 8 kbps) up to bit rates allowing transparent quality at about 128 to 256 kbps stereo. The high-quality range is covered by codecs like AAC, AAC Scalable and BSAC, providing additional functionalities for large-step and fine-grain scalability.

On top of these codecs, the MPEG-4 SLS codec provides a fine-grain scalable extension to lossless compression. For lossless compression, typically a bit rate of about 768 kbps stereo is necessary for 48 kHz / 16 bit signals (clearly, the required bit rate varies for different input signals). For higher resolutions (sampling rates and word length) this bit rate increases up to about 4000 kbps stereo for 192 kHz / 24 bit signals. Consequently, there is a very large gap between the bit rates required for perceptually transparent and for lossless coding. This is illustrated in Figure 4.

The MPEG-4 SLS codec fills this gap by allowing any intermediate bit rate in a fine-grain scalable way. Nevertheless, an evaluation method is necessary to show the merits of the MPEG-4 SLS codec in this intermediate, “near-lossless” bit rate range.

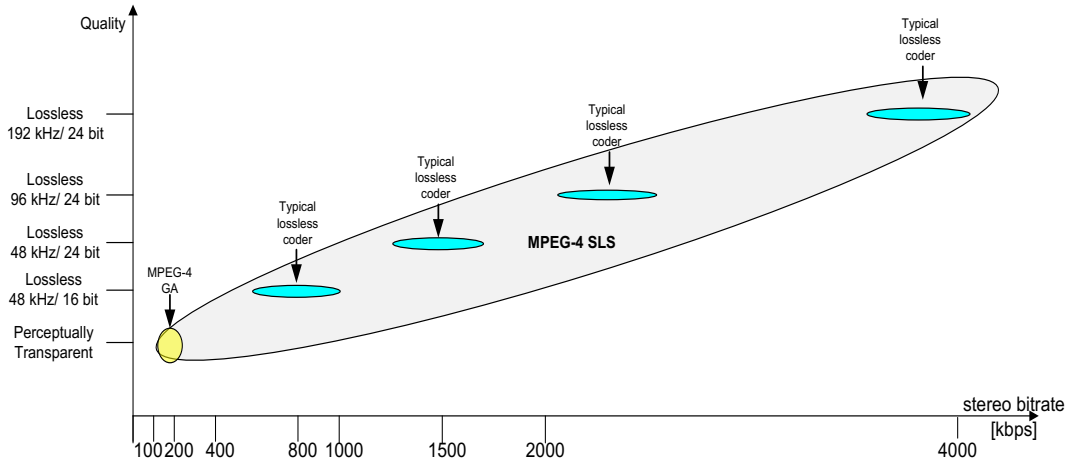


Figure 4. Bit rate range covered by MPEG-4 SLS

5.1 Evaluation of Near-Lossless Coding according to ITU-R BS.1548-1

While up to now MPEG has not evaluated perceptual codecs at bit rates beyond perceptual transparency, other standardization bodies have spent considerable effort to define evaluation criteria for near-lossless coding.

The ITU-R recommendation BS.1548-1 [8] defines requirements for audio coding systems for digital broadcasting. According to this recommendation, audio codecs for contribution and distribution should fulfill the following requirements:

"The quality of sound reproduced after a reference contribution/distribution cascade (five contribution codecs and three distribution codecs working consecutively) should be subjectively indistinguishable from the source for most types of audio programme material. Using the triple stimuli double blind with hidden reference test, described in Recommendation ITU-R BS.1116 – Methods for the subjective assessment of small impairments in audio systems including multichannel sound systems – this requires mean scores generally higher than 4.5 in the impairment 5-grade scale, for listeners at the reference listening position. The worst rated item should not be graded lower than 4 (Recommendation ITU-R BS.775)."

Furthermore it is noted that: *"The contribution/distribution cascade, when placed in tandem with the emission codec, should not cause a significant reduction in quality compared to the basic audio quality of the emission codec."*

5.2 Objective Measurement of Audio Quality based on ITU-R BS.1387-1 (PEAQ) [9]

The PEAQ measurement provides methods for objective measurements of perceived audio quality. It focuses on applications, which are normally assessed in the subjective domain by applying an ITU-R BS.1116 test [8].

The PEAQ system provides the following important output values:

- **Objective Difference Grade (ODG)**
  The ODG values are designed to mimic the BS.1116 listening test ratings obtained from typical test listeners by means of an objective measurement procedure. The ODG corresponds to the subjective difference grade. The grading scale ranges from 0 to -4, having the following meaning:
  0 “imperceptible”
  -1 “perceptible but not annoying”
  -2 “slightly annoying”
  -3 “annoying”
  -4 “very annoying”

- **Noise-To-Mask Ratio (NMR)**
  The NMR estimates the ratio between the actual distortion (“Noise”) and the maximum inaudible distortion, i.e. the masking threshold (“Mask”). NMR values smaller than 0dB indicate the headroom between the noise and the threshold of audibility whereas values larger than 0dB indicate audible distortions.

5.3 Evaluation of MPEG-4 SLS under Tandem Coding

Several tandem coding cascades of AAC and SLS were performed and the audio quality was estimated using the PEAQ implementation PQEvalAudio from the AFsp library, which is an implementation of the basic version of PEAQ [9]. The MPEG perceptual coding test set (48 kHz, 16 bit) was used as input. In each coding cycle an additional delay of 31 samples was introduced to avoid a frame alignment in the tandem coding process.

The following tandeming configurations were measured:

- **AAC tandeming**
  N x AAC@128kbps (N=1,…,16)
- **AAC+SLS tandeming**
  N x (AAC+SLS)@{512,768kbps}
  (AAC@128kbps+SLS enhancement@{384,640kbps})
  (N=1,…,16)
- **AAC+SLS tandeming, followed by AAC**
  (N-1) x (AAC+SLS)@512kbps
  (AAC@128kbps+SLS enhancement@384kbps),
  followed by 1 x AAC@128kbps

(N=1,…,16)

- **SLS non-core tandeming**
  N x SLS non-core @{512,768kbps} (N=1,…,16)

All the SLS bitrates correspond to maximum rates obtained by assigning a fixed number of bits to each frame. In this evaluation, no bit reservoir is used to allow for higher peak bitrates. Consequently, the resulting average bitrate per item can be much lower than the allowed maximum bitrate.

The different tandem coding configurations are motivated by the following application scenarios:

1. Application scenario for SLS tandeming: The broadcast chain

In a broadcast environment, MPEG-4 SLS could be used in all stages comprising archiving, contribution/distribution and emission. This is illustrated in Figure 5. For archiving the codec can operate in a lossless way, for contribution/distribution a constant, high bit rate (e.g. 512 kbps) can be used, and finally the AAC core can be used for emission.

In this broadcast chain, one main feature of the MPEG-4 SLS architecture can be used: In every stage where no post-processing but only a lower bit rate is required, the bit stream is just truncated, and no re-encoding is therefore required.

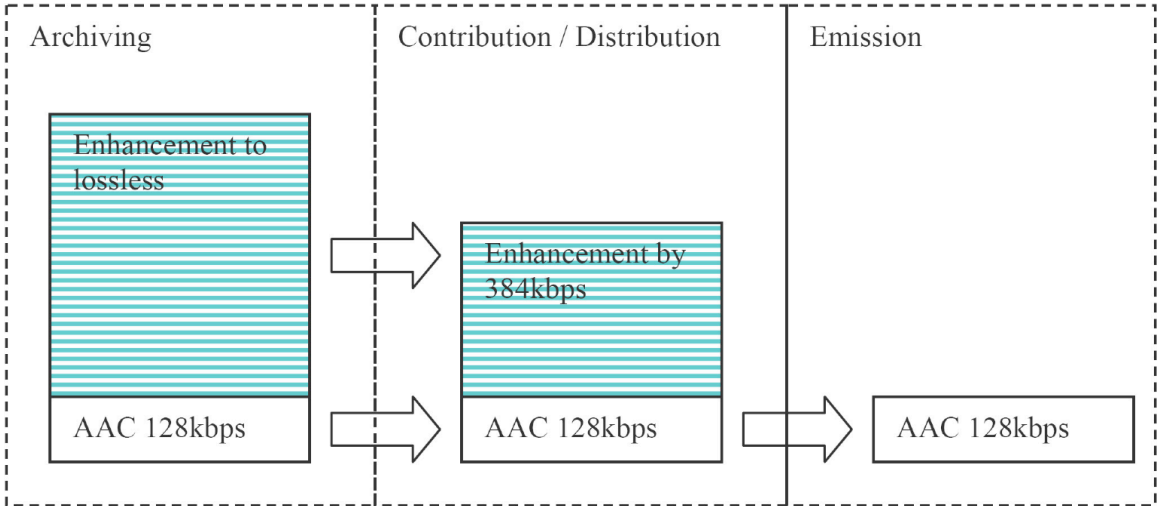


Figure 5. MPEG-4 SLS in the broadcast chain

2. Application scenario for SLS tandeming: Guaranteed compression of 2:1

In situations where a constant rate transmission is required, SLS can provide a high near-lossless to lossless audio quality at e.g. 768 kbps stereo for 48 kHz / 16 bit, corresponding to a guaranteed compression of 2:1.

This is a unique feature of SLS. While pure lossless codecs can provide an average compression of 2:1 for certain test material, this compression cannot be guaranteed, and thus a transmission at a reduced constant bitrate is not possible. This can easily be proven by assuming the contrary and concluding that not all possible audio signals can be represented with the reduced bitrate.

Table 1 lists the average bitrates required for lossless coding using AAC@128kbps + SLS for the MPEG perceptual coding testset (48 kHz, 16 bit). It can be observed that at least one item requires an average bitrate that is higher than 768kbps.

Table 6. Average bitrates for AAC@128kbps + SLS for lossless coding

item	avg. bitrate [kbps]
es01	699
es02	492
es03	514
sc01	496
sc02	700
sc03	867
Sm01	698
Sm02	442
Sm03	707
si01	665
si02	690
si03	642
All	644

As the bitstream truncation used in this evaluation assigns a constant number of bits per frame, i.e. it does not use a bit reservoir to allow for local fluctuations, it turns out that in fact in *most of the items one or more frames would have to be truncated* in order to assure a maximum bitrate of 768kbps in each frame. This illustrates the usefulness of a *near-lossless operation* mode, provided that this mode still provides excellent subjective audio quality. This is assessed in the subsequent evaluations.

5.3.1 AAC-only Tandeming

As a baseline for the evaluation of combined AAC/SLS performance,

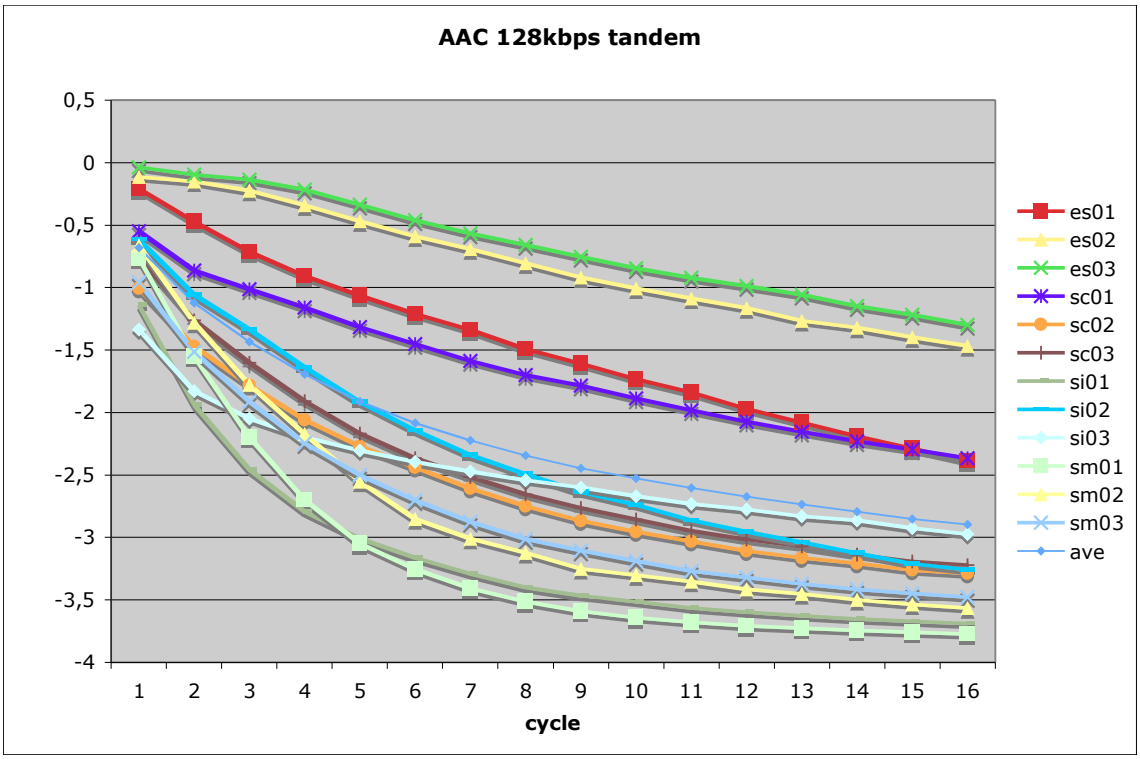


Figure 6 shows the tandem coding results for AAC @ 128 kbps. As expected, it can be observed that the audio quality significantly degrades with increased number of tandem cycles, tandem coding not being a recommended practice for such coders.

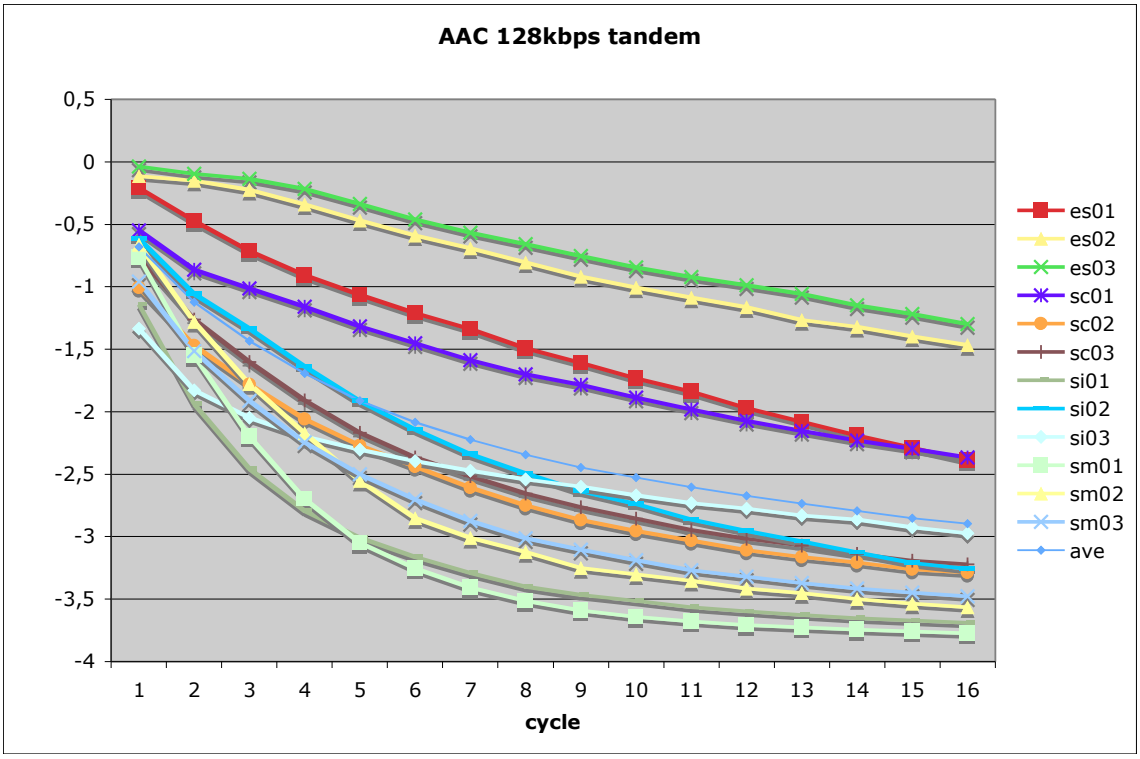


Figure 6. Tandem coding results for MPEG-4 AAC (AOT 2) @ 128 kbps

5.3.2 AAC+SLS Tandeming

Application scenario: Studio/production quality representation of audio that is resistant to repeated encoding/decoding (tandem coding). Alternatively: guaranteed 3:1 near lossless compression.

Figure 7 shows the tandem coding results for AAC+SLS @ 512 kbps (AAC @ 128 kbps + SLS enhancement @ 384 kbps). It can be observed that the audio quality remains at a very high level, even after 16 tandem cycles. This illustrates the robustness of such a representation towards tandeming.

Note: Positive ODG values can also occur in the PEAQ measurement in case of very high audio quality (no error). For example, in case of identical reference and test file the ODG value is about 0.2 in the applied PEAQ implementation.

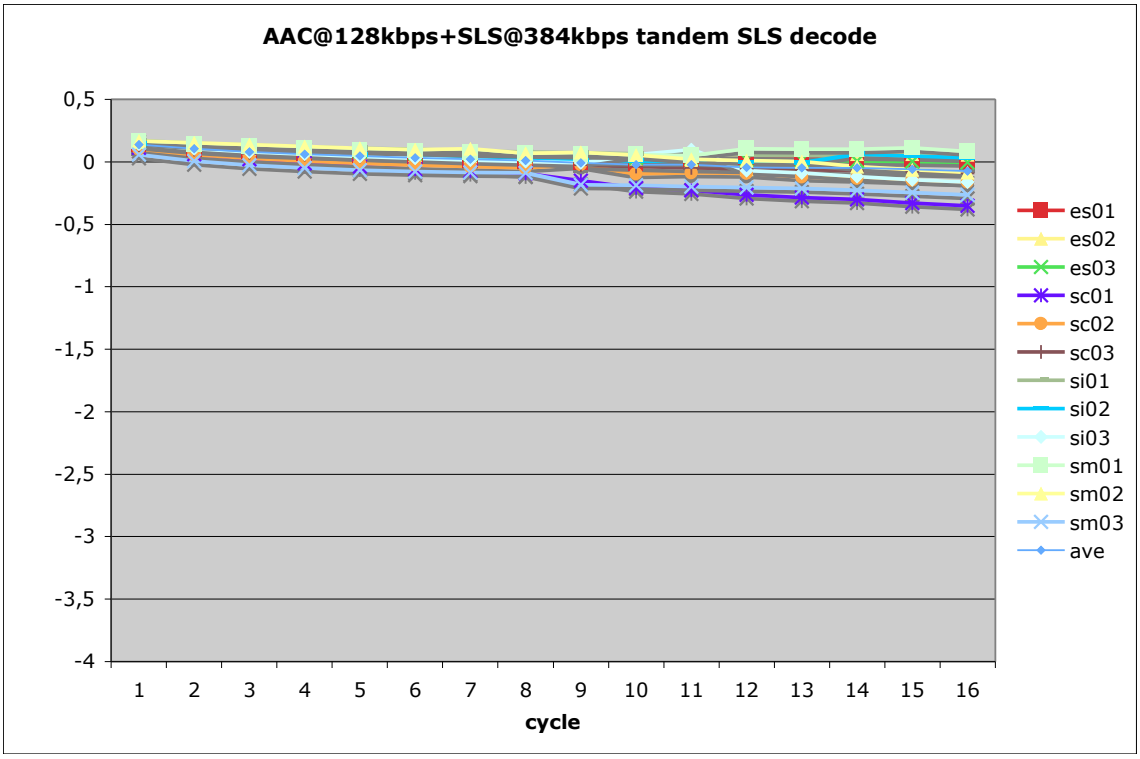


Figure 7. Tandem coding results for AAC+SLS @ 512 kbps (AAC @ 128 kbps + SLS enhancement @ 384 kbps)

Figure 8 shows the tandem coding results for AAC+SLS @ 768 kbps (AAC @ 128 kbps + SLS enhancement @ 640 kbps). This corresponds to a guaranteed compression of 2:1. At this high bitrate the audio quality expressed by the ODG values stays at the highest level over all tandem cycles.

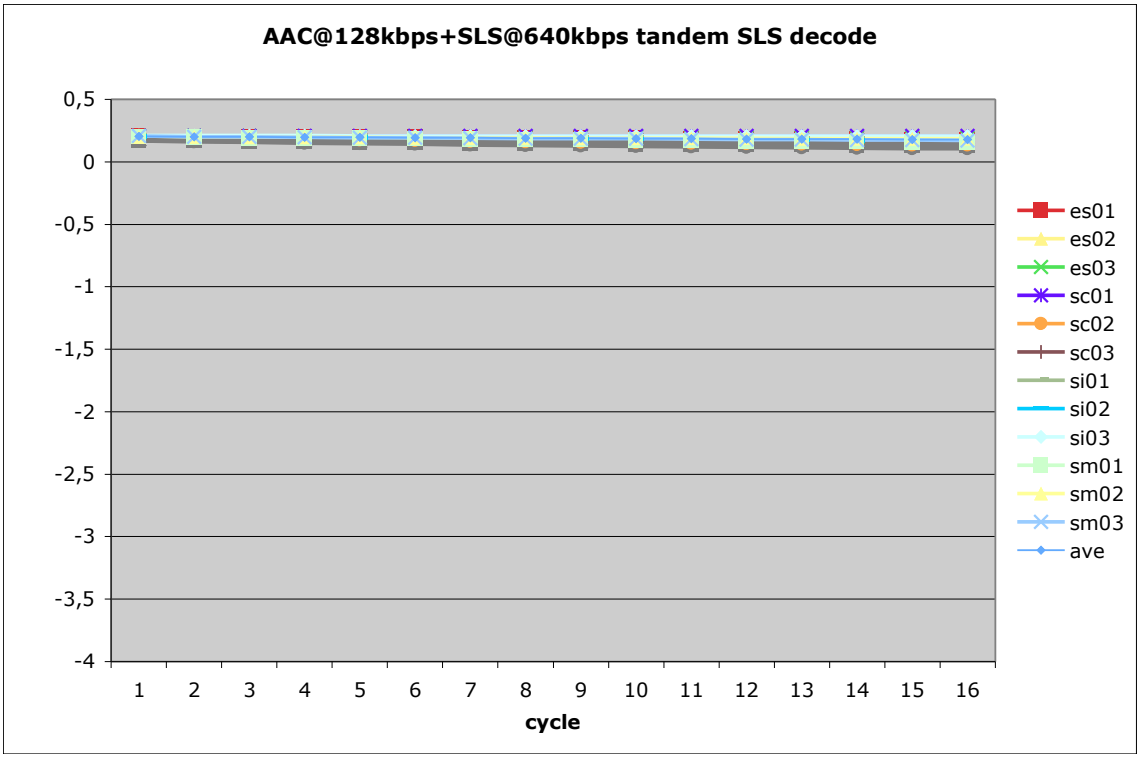


Figure 8. Tandem coding results for AAC+SLS @ 768 kbps (AAC @ 128 kbps + SLS enhancement @ 640 kbps)

5.3.3 AAC+SLS Tandeming, followed by AAC

Application scenario: Repeated encoding/decoding (tandem coding) during production / contribution process. Final distribution using the embedded AAC part.

Figure 9 shows the results for the same tandem cycles as in Figure 7, but with the last cycle only decoding the AAC core. Thus, a sequence of several stages of AAC+SLS @ 512 kbps is followed by one AAC @ 128 kbps coding stage. It can be observed that the audio quality stays at approximate the same level as for a single stage of AAC @ 128 kbps (cycle 1). Thus, the cascade of AAC+SLS tandem coding stages does not significantly degrade the audio quality received after the final AAC coding stage.

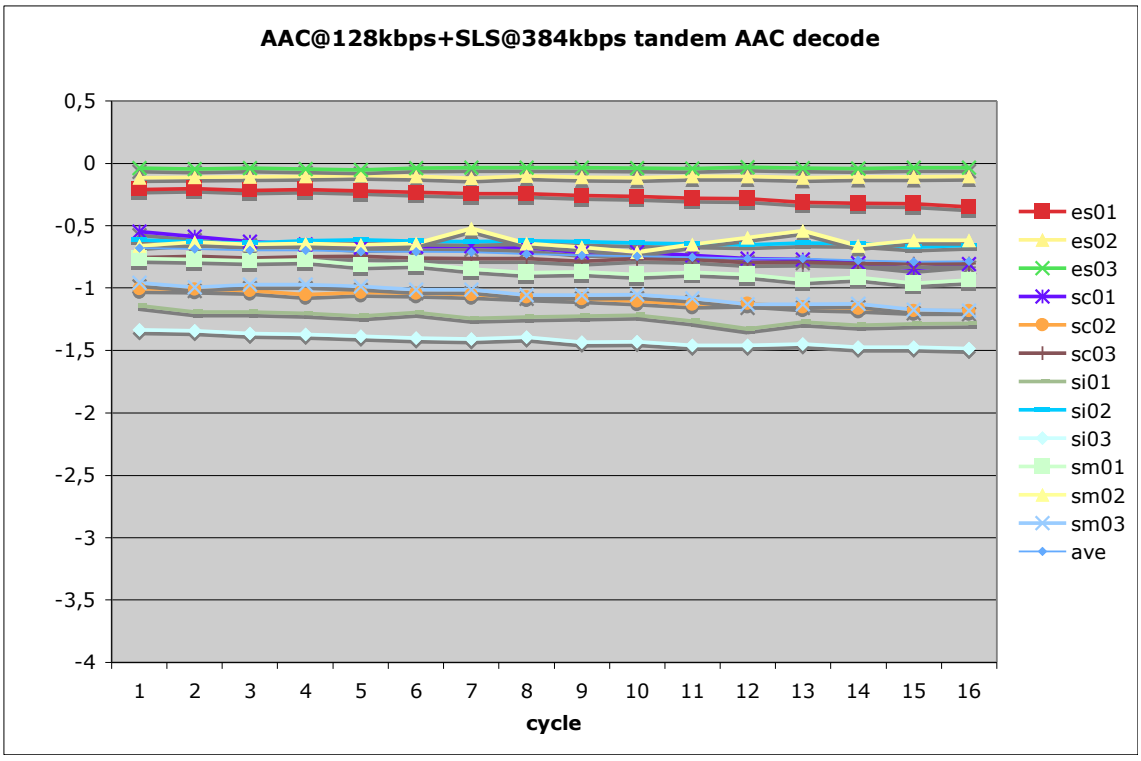


Figure 9. Tandem coding results for AAC+SLS @ 512 kbps (AAC @ 128 kbps + SLS enhancement @ 384 kbps), in the last cycle only the AAC core is decoded

5.3.4 SLS non-core Tandeming

Application scenario: Stand-alone lossless audio coding with a representation that may be easily truncated for transmission across constant-rate channels (e.g. guaranteed 3:1 or 2:1 compression).

In case of SLS operating in the non-core mode, the SLS bitstream can also be truncated, but the question is whether this truncation maintains a reasonable audio quality.

As the SLS enhancement does not rely on a separate perceptual model, an audio quality equivalent to an optimized perceptual audio codec cannot be expected for low bitrates. Nevertheless, the SLS coder in a non-core-based mode provides a consistent increase of SNR in each scale factor band. As a consequence, for high bitrates a high subjective audio quality is obtained.

Figure 10 shows the tandeming results for SLS non-core operating at 512kbps. It can be observed that for this high bitrate the obtained audio quality is equally high as for the AAC-based mode.

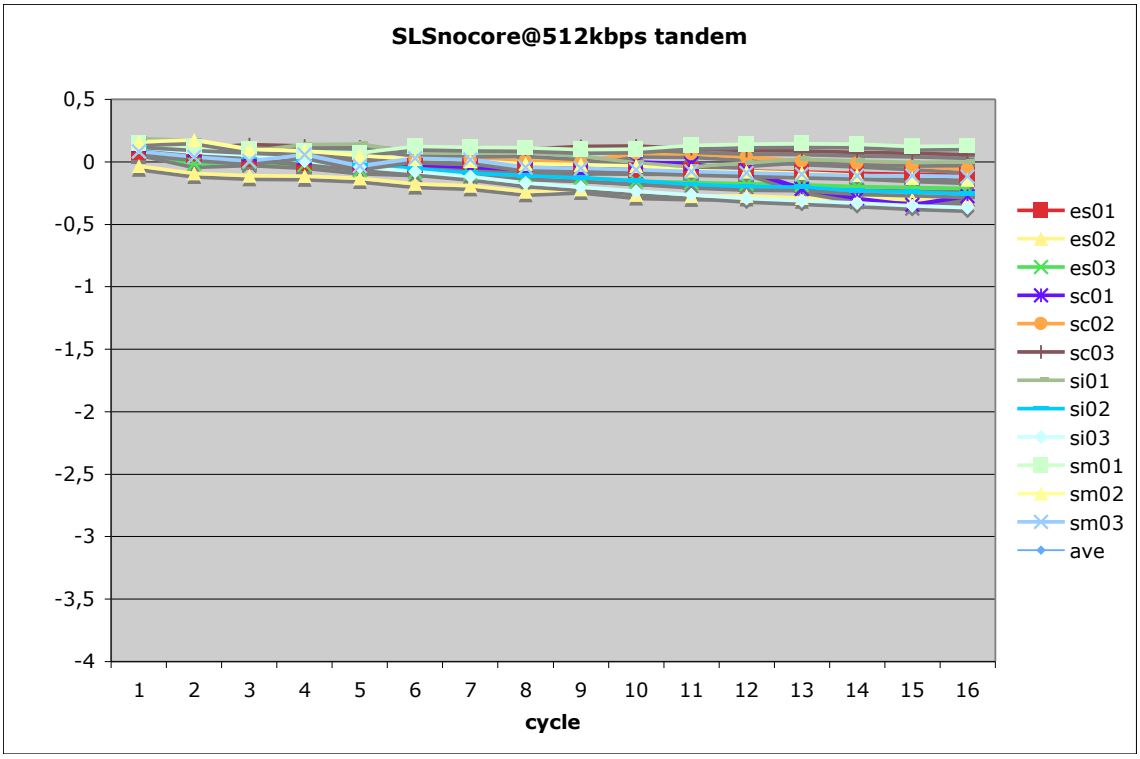


Figure 10. Tandem coding results for MPEG-4 SLS non-core @ 512 kbps

Figure 11 shows the tandeming results for SLS non-core operating at 768kbps. This corresponds to a guaranteed compression of 2:1. At this high bitrate the audio quality expressed by the ODG values stays at the highest level over all tandem cycles.

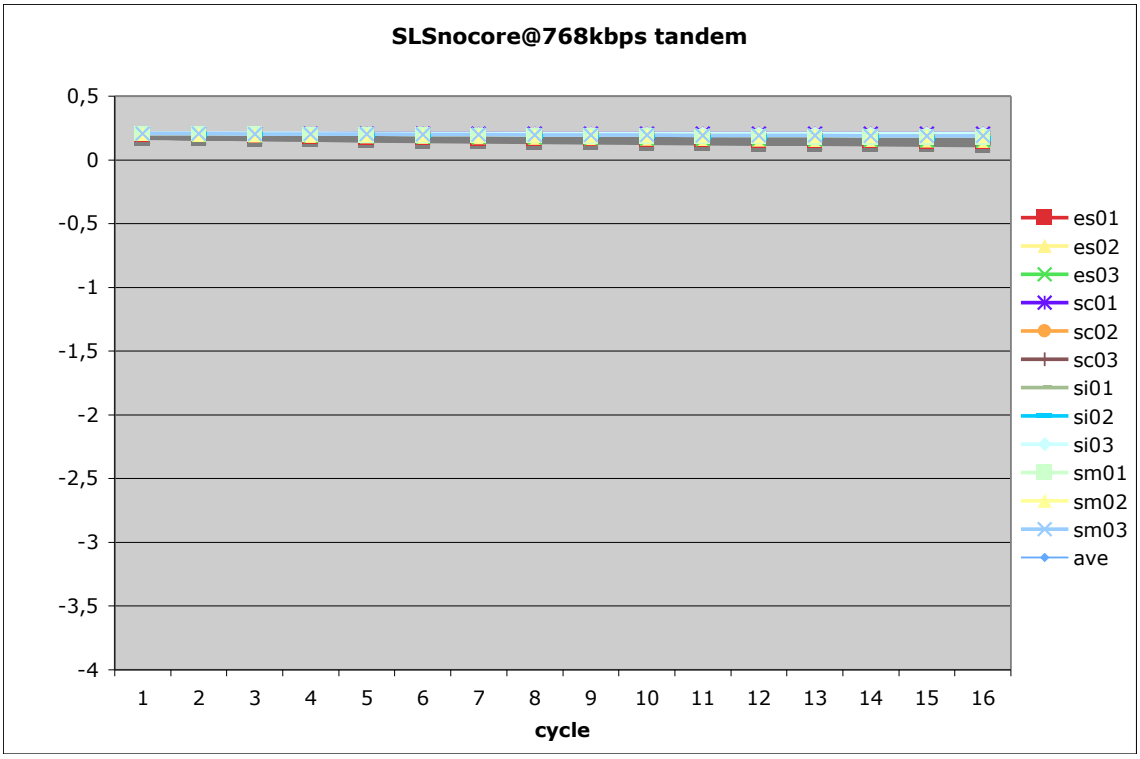
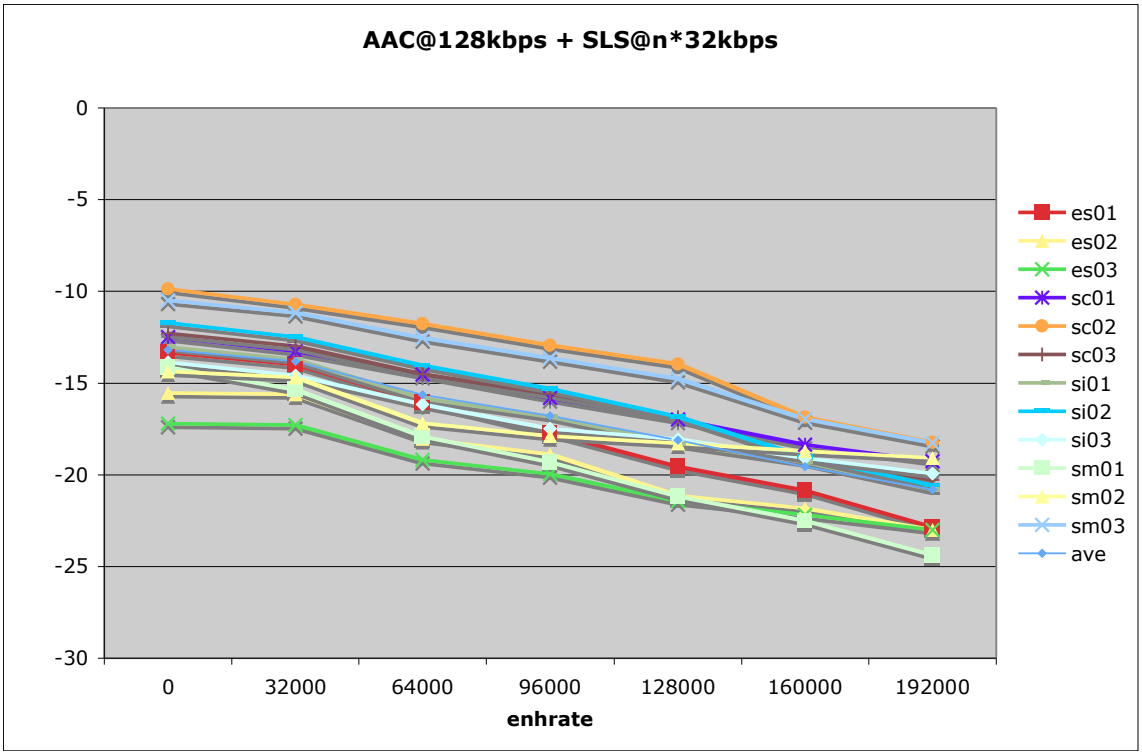


Figure 11. Tandem coding results for MPEG-4 SLS non-core @ 768 kbps

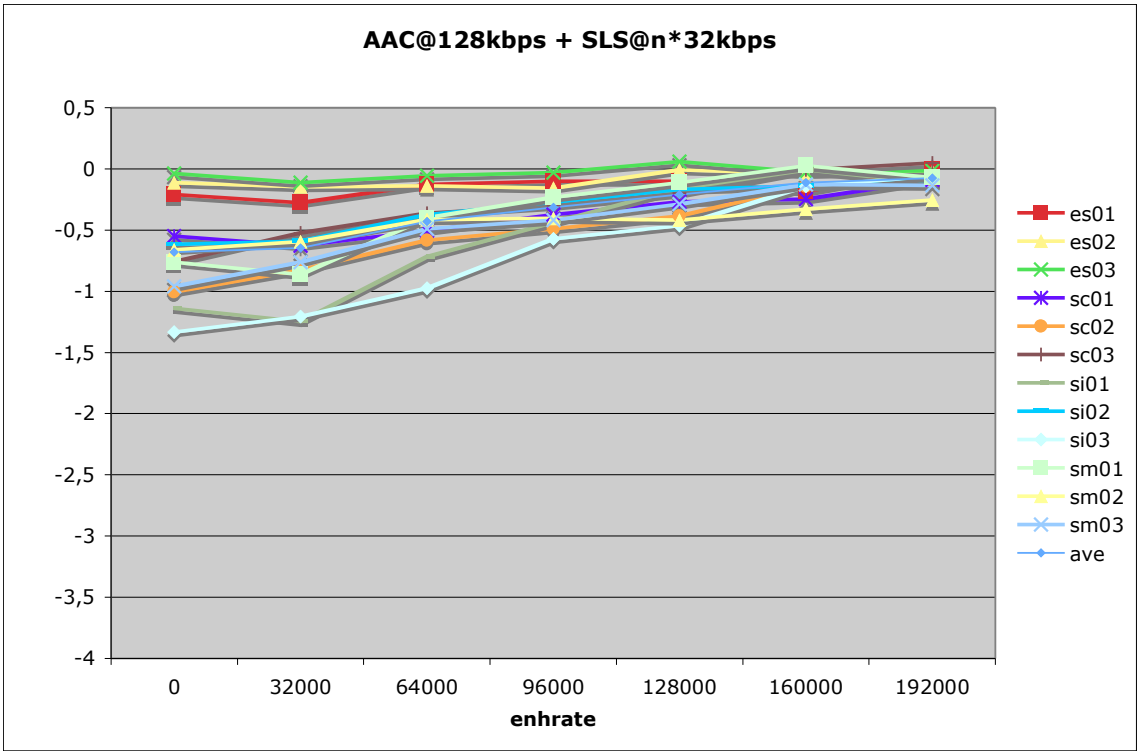
5.3.5 Results for MPEG-4 SLS in case of Small Bitrate Increments

The behavior of SLS in case of small bitrate increments is measured by evaluating AAC@128kbps + SLS enhancement@{32,64,96,128,160,192kbps}

NMR and ODG values are shown in the following figures. It can be observed that the average audio quality increases with increased bitrate. The NMR values indicate that each step of increased bitrates adds additional headroom towards audibility and thus enhances post-processing robustness.



a) NMR (Noise to Mask Ratio) results



b) ODG (Objective Difference Grade) results

Figure 12. Performance for AAC@128kbps + SLS enhancement @ {32, 64, 96, 128, 160, 192kbps}

6 Summary

MPEG-4 SLS provides a scalable lossless extension of AAC. It allows for an efficient lossless compression with low complexity. It also works as a standalone lossless coder when the AAC core is not present.

The lossless enhancement provides fine-grain scalability, and covers a very wide range of bitrates between the AAC core bitrate and that of lossless operation, resulting in additional operation points. This additional functionality introduces only very marginal bitrate overhead compared to lossless-only operation.

Performance measurements indicate that MPEG-4 SLS provides high robustness to tandem coding at near-lossless operation points. In addition, measurements for small bitrate increments show that the average audio quality increases with increased bitrate. MPEG-4 SLS can provide constant bit-rate compression (e.g., guaranteed 2:1 compression), while maintaining very high audio quality, both in AAC based mode and in non-core mode.

References

[1] R. Yu, R. Geiger, S. Rahardja, J. Herre, X. Lin, and H. Huang "MPEG-4 Scalable t o Lossless Audio Coding," 117th Convention of Audio Engineering Society (AES), October 28-31, 2004, San Francisco, CA, USA.

[2] ISO/IEC JTC 1/SC 29/WG 11, N5208, Final Call for Proposals on MPEG-4 Lossles s Audio Coding, October 2002, Shanghai, China

[3] ISO/IEC JTC 1/SC 29/WG 11, N7366, Text of ISO/IEC 14496-3:2001/FDAM5, Sc alable Lossless Coding (SLS), July 2005, Poznan, Poland.

[4] IA-32 Intel Architecture Optimization Reference Manual, ON: 248966-009, Intel Corp.

[5] A. Moffat, R. Neal, and I. Witten, Arithmetic Coding Revisited, ACM Transactions on Information Systems, Vol. 16, No. 3, July 1998.

[6] A. Moffat and A. Turpin, On the Implementation of Minimum-Redundancy Prefix Codes, IEEE Trans. Communications, Vol. 45, No. 10, 1997.

[7] Recommendation ITU-R BS.1548-1, "User requirements for audio coding systems for digital broadcasting", International Telecommunications Union, Geneva, Switzerland, 2001-2002.

[8] Recommendation ITU-R BS.1116-1, "Methods for the Subjective Assessment of Small Impairments in Audio Systems including Multichannel Sound Systems", International Telecommunications Union, Geneva, Switzerland, 1994.

[9] Recommendation ITU-R BS.1387-1, "Method for Objective Measurements of Perceived Audio Quality", International Telecommunications Union, Geneva, Switzerland, 1998.

Appedix A. Testing Sequences

A.1. MPEG-4 Lossless Audio Coding Testing Sequences

Group	Spec.	File name	Source
a)	192kHz/24bit /stereo	Avemaria.wav	Avemaria / C. Gounod
		Etude.wav	Etude / F.Chopin
b)	96kHz/24bit/ stereo	Flute.wav	Concerto for Two Flutes and Orchestra RV.533 Op.42 No.2 in C major / Vivaldi
		Clarinet.wav	Concerto for Clarinet and Orchestra in A major K.622 / Mozart
		Violin.wav	Concerto for Violin and String Orchestra No.1, BWV1041 / Bach
		Haffner.wav	Symphony No.35 in D major "Haffner", K.385 / Mozart
c)	192/24/stereo	Cymbal192.wav	MEI original recording
	96/24/stereo	Cymbal96.wav	
d)	192kHz/24bit /stereo	Broadway	
		Dcymbals	
		Mfv	
e)	96kHz/24bit/ stereo	blackandtan	
		Broadway	
		Cherokee	
		Dcymbals	
		Fouronsix	
		Mfv	
		Unfo	
		Waltz	

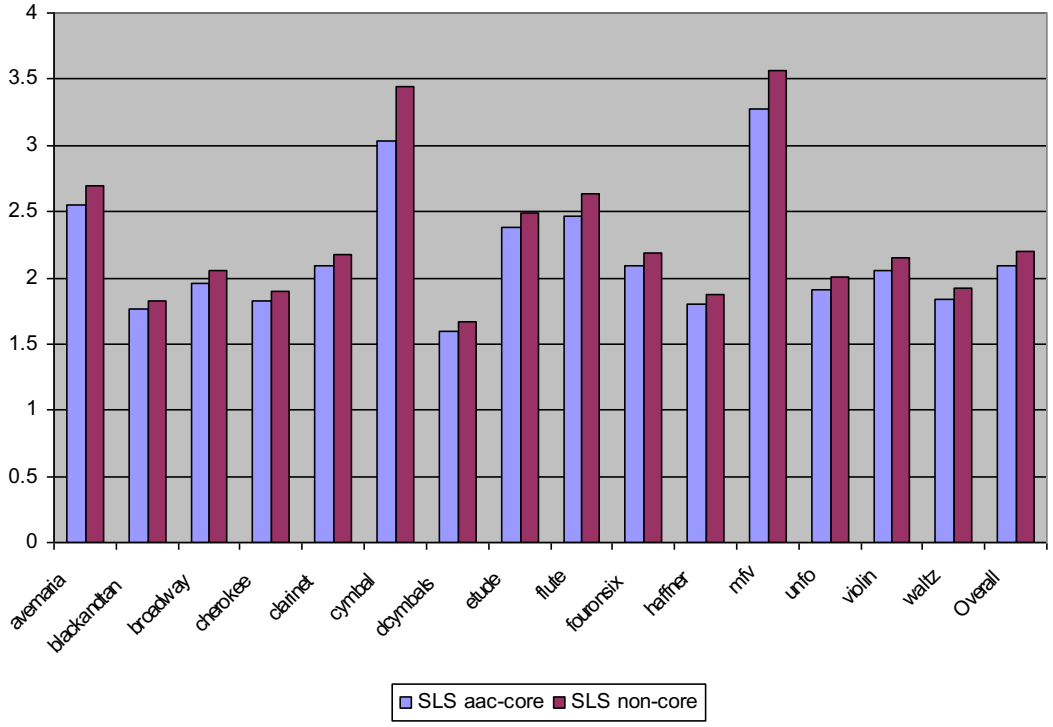
Note for groups a), b) and c): Music source: recording of New York Symphonic Ensemble

Note: Sequences of lower sampling rate (96 kHz, 48 kHz) and lower wordlength (16 bits/sample) are made available by downsampling and truncating (with proper dither) sequences of higher sampling rates and higher length.

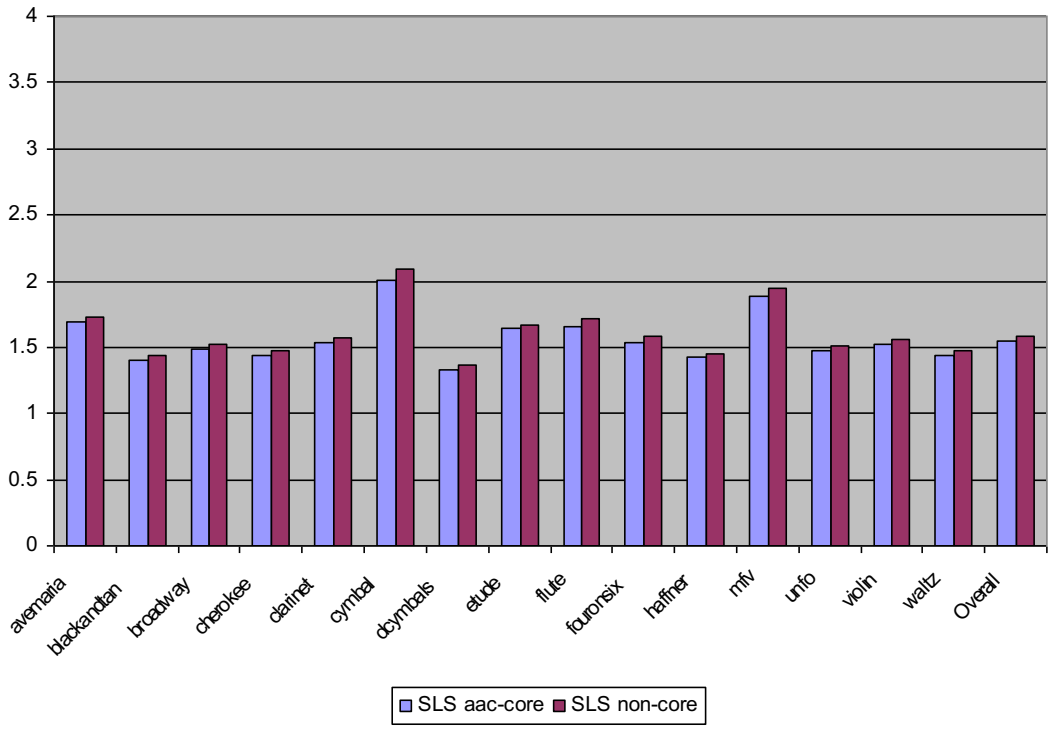
A.2 Name list for 20 commercial CDs

CD	Title	Catalog	Total Size in Bytes
CD1	ACDC – Highway to Hell	Sony 80206	441487304
CD2	Avril Lavigne - Let Go	Arista 14740	515596604
CD3	Backstreet Boys - Greatest Hits Chapter One	Jive 41779	682791008
CD4	Brian Setzer – The Dirty Boogie	Interscope 90183	526025372
CD5	Cowboy Junkies – Trinity Session	RCA-8568	560540928
CD6	Grieg – Peer Gynt – von Karajan	DG 439010	658555780
CD7	Jannifer Warnes - Famous Blue Raincoat	BMG 258418	439612716
CD8	Marlboro Music Festival – 50th Anniversary Album Bridge\DISC A	Bridge 9108	827634048
CD9	Marlboro Music Festival – 50th Anniversary Album Bridge\DISC B	Bridge 9108	727909776
CD10	Nirvana - Nirvana	Interscope 493523	584190420
CD11	Philip Jones – 40 Famous Marches\CD1	Decca 416241	828904480
CD12	Philip Jones – 40 Famous Marches\CD2	Decca 416241	815498080
CD13	Pink Floyd – Dark Side of the Moon	Capitol 46001	454195116
CD14	Rebecca Pidgeon – The Raven	Chesky 115	488040572
CD15	Ricky Martin	Sony 69891	624809416
CD16	Shubert Piano Trio in E-flat	Sony 48088	615459820
CD17	Spaniels – The Very Best 0f	Collectables 7243	677821628
CD18	Steeleye Span – Below the Salt	Shanachie 79039	420714396
CD19	Suzanne Vega – Solitude Standing	A&M 5136	469631380
CD20	Westminster Concert Bell Choir – Christmas Bells	Gothic Records 49055	589623672

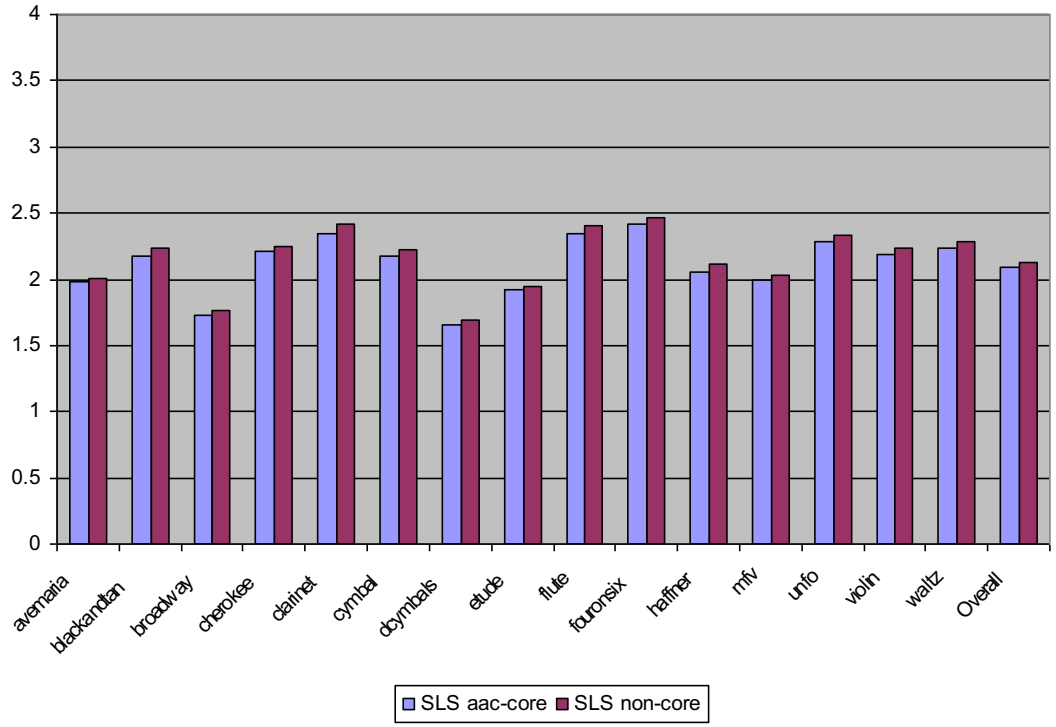
Appendix B. Compression Ratio of SLS for MPEG-4 Lossless Audio Coding Testing Sequences



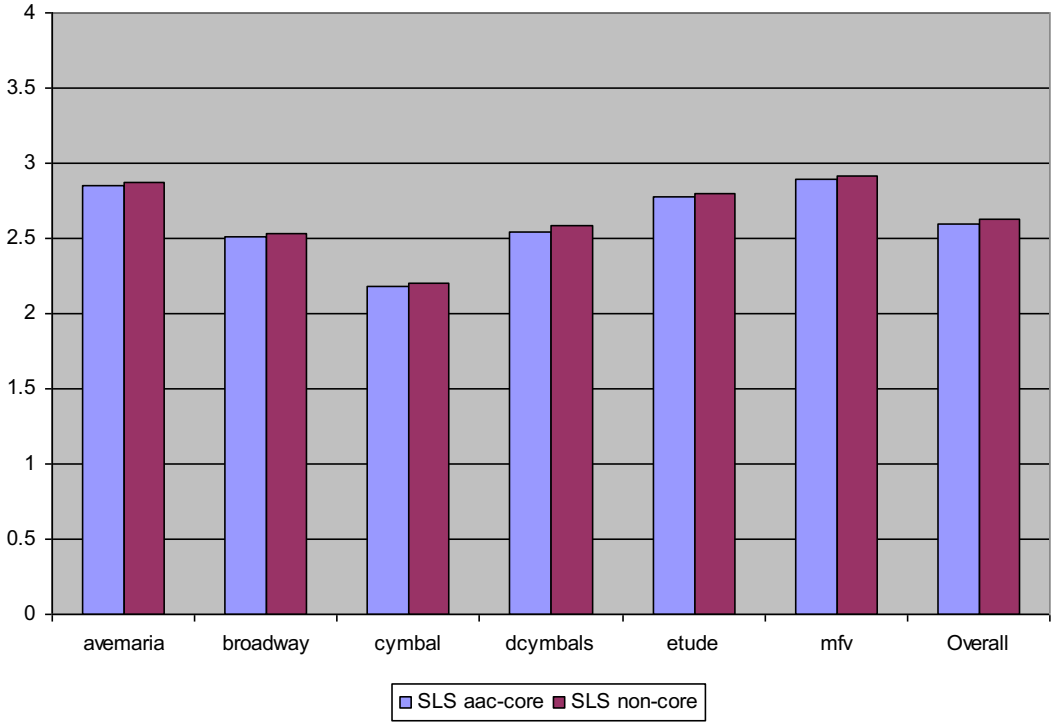
a) 48 kHz/16 bit



b) 48 kHz/24 bit



c) 96 kHz/24 bit



d) 192 kHz/24 bit

Figure B.1. Compression ratio performance of SLS

The averaged coding efficiency performance of SLS for the 20 commercial CDs listed in Appendix A are given the following figure.

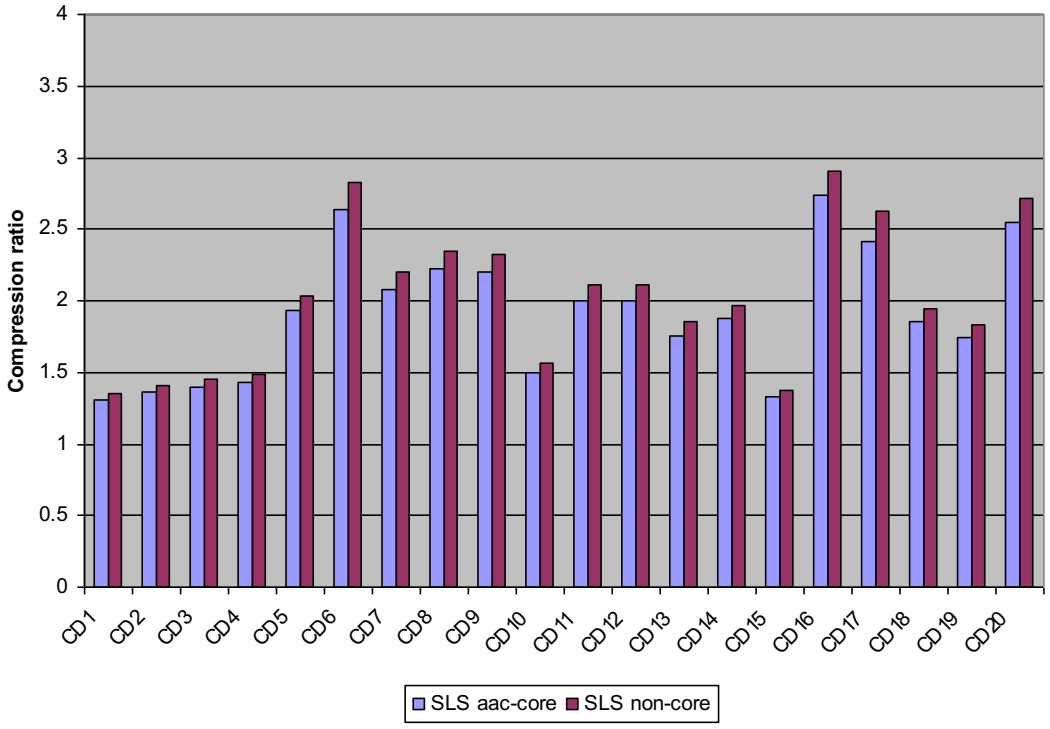


Figure 13. Compression ratio performance of SLS for 20 commercial music CDs

Appendix C. Analysis of Algorithm Complexities of SLS decoder

The main components contributing to SLS computational complexity are IntMDCT filterbank and bit-plane arithmetic encoder.

C.1. IntMDCT filterbank

The numbers of 32-bit integer operations required by the inversed IntMDCT per sample in *stereo mode* are presented in the following tables.

Table C.1. Average numbers of INT32 operations per sample in IntMDCT algorithm.

N	Muls	Adds/ Subs	Ors	Shifts	Negs	Movs	Combined All = 1 cycle	Combined Pentium
4096 or 512	12	38.63	31.35	44.80	28.97	9.54	165.29	401.44
2048 or 256	12.75	36.39	28.82	45.29	26.90	9.29	159.44	410.36
1024 or 128	10.5	34.16	26.25	39.79	24.80	8.99	144.49	353.26

Table C.2. Maximum numbers of INT32 operations per sample in IntMDCT algorithm.

N	Muls	Adds/ Subs	Ors	Shifts	Negs	Movs	Combined All = 1 cycle	Combined Pentium
4096 or 512	12	38.63	34.50	46.10	29.93	9.54	170.7	408.7
2048 or 256	12.75	36.39	31.50	47.04	27.92	9.29	164.89	419.21
1024 or 128	10.5	34.16	28.5	42.00	26.52	8.99	150.67	364.09

These estimates were obtained by enabling instruction counting in the reference IntMDCT implementation and running SLS encoder and decoder over the set of MPEg lossless test files.

C.2. Unpacking DCT tables

Table C.3. shows the maximum additional computation required for unpacking the compact tables representation.

Table C.3. Maximum numbers of INT32 operations for unpacking of DCT tables.

N	Muls	Adds/ Subs	Shifts
4096 or 512	0	29.75	15.5
2048 or 256	0	26.25	12
1024 or 128	0	16.25	6.5

This operation could be partially or fully avoided by spending more memory on pre-computing some of the most frequently used table values.

C.3. Integer M/S stereo decoding

The inverse Stereo IntMDCT expects an M/S spectrum by default. Hence M/S has to be applied to the scale factor bands where the M/S flag is switched off. The M/S coding is performed in the IntMDCT domain by a lossless $\pi/4$ Givens rotation with the lifting scheme as follows:

Step 1:

$$S = S - NINT\left(c_1 \cdot M\right);$$

Step 2:

$$M = M - NINT\left(c_2 \cdot S\right);$$

Step 3:

$$R = M;$$
$$L = S - NINT\left(c_1 \cdot R\right);$$

where M,S,R,L denotes the spectral data of Mid, Side, Left, and Right channels and

$$c_1 = (\cos\frac{\pi}{4} - 1) \Big/ \sin\frac{\pi}{4}, \text{ and } \; c_2 = \sin\frac{\pi}{4}.$$

These operations are performed in fixed point using integer multiplies and post-shifts, requiring the total of

1.5 muls + 1.5 adds + 1.5 shifts

32-bit operations per sample per channel.

Table C.4. Numbers of INT32 operations required by AAC-SLS M/S stereo decoder

Muls	Adds	Shifts	Combined All = 1 cycle	Combined Pentium
1.5	1.5	1.5	4.5	27.75

C.4 Complexity of AAC-SLS inverse error mapping.

SLS decoder uses the following deterministic implementation of procedure for calculation of quantization threshold:

thr(x) = thr_mantissa[*scale_res*][|*x*|]<<(thr_exp[*scale_res*][|*x*|]+*scale_int*)

where

scale_int = *(scale_factor(sfb)+core_scale_factor[window_type][word_length])* >> 2;

And

scale_res = *scale* – (*scale_int* << 2).

It is clear, that this procedure requires

3 adds + 3 shifts + 6 movs

for calculating thr(x), plus at most

1 cmp + 1 neg

for finding absolute value of x.

The total numbers of instructions and cycles are summarized in the following table:

Table C.5. Numbers of INT32 operations required by AAC-SLS inverse error mapper.

Adds	Shifts	Cmps	Negs	Movs	Combined All = 1 cycle	Combined Pentium
3	3	1	1	6	14	17.5

C.5. Bit-plane arithmetic decoder

The pseudo-code of a binary arithmetic decoder required by the SLS specification [3] is shown below:

```
/* update interval: */
range = (long)(high - low + 1);                    // 2 adds
cum = (long)(value - low + 1) << PRE_SHT;          // 2 adds + 1 shift

/* decode symbol:*/
if (cum < range * freq) {                          // 1 mul
   sym = 1;
   ac->high = low + (range*freq >> PRE_SHT)-1;     // 2 adds + 1 shift
} else {
   sym = 0;
   ac->low = low + (range*freq >> PRE_SHT);// 1 add + 1 shift
}

/* interval renormalization: */
for (;;) {                              // per each decoded bit:
          if (high<HALF_VALUE) {
             /* do nothing */
   } else if (low>=HALF_VALUE) {
             value -= HALF_VALUE;                          // 3 adds
      low -= HALF_VALUE;
             high -= HALF_VALUE;
          } else if (ac->low>=QTR_VALUE && ac->high<TRDQTR_VALUE){
      value -= QTR_VALUE;                           // 3 adds
      low -= QTR_VALUE;
      high -= QTR_VALUE;
   } else
      break;
   low = ac->low << 1;                       // 3 shifts + 1 add
   high = ac->high << 1;
   value = (ac->value << 1) + input_bit(); // 1 bit-read
       }
```

Based on this pseudo-code, the complexity of this algorithm can be upper bounded by:

N * (T * (1 mul + 6 adds + 2 shifts) + R * (4 adds + 3 shifts + 1 bit-read))

and lower-bounded by

N * (T * (1 mul + 5 adds + 1 shift) + R * (1 add + 3 shifts + 1 bit-read))

where N is the transform size, T is the average maximum number of bitplanes encoded by arithmetic encoder, and R is the average output bitrate

Assuming that only 6 bitplanes are encoded using arithmetic code ([5]), and that the effective rate produced by the arithmetic encoder is ½ of the input rate, then the upper bound for the complexity becomes:

N * (6 * (1 mul + 6 adds + 2 shifts) + 3 * (4 adds + 3 shifts) + R bit-reads).

= N * (6 muls + 48 adds + 21 shifts + R bit-reads).

The following table lists these numbers along with the total number of all 32-bit operations and the estimated number of cycles on Intel Pentium platform.

Table C.6. Numbers of INT32 operations per sample required by SLS bit-sliced arithmetic decoder.

Muls	Adds	Shifts	Combined All = 1 cycle	Combined Pentium
6	48	21	75	192

For simplicity, we have removed bit-reads from the above estimates.

C.6. Complexity of AAC Huffman decoding

To assess the number of operations required for decoding of Huffman-encoded spectral lines in AAC bitstream, we will assume that AAC decoder utilizes Moffat-Turpin algorithm [6]. The complexity of decoding of a single symbol using this algorithm can be expressed as:

1 add + 1 shift + 1 move + R * (1 cmp + 1 add + 1 bit-read),

where R is the length of a codeword, and the first 3 instructions are needed to calculate the index of a decoded codeword in a canonically-ordered set, and then map it the original symbol [6].

Assuming now, that the rate of AAC bitstream is restricted to 64Kbps /channel, and that in the worst case a single codeword can represent a single spectral line, we obtain the following complexity estimates:

Table C.7. Numbers of INT32 operations required by AAC Huffman decoder

Sampling Rate	Adds	Shifts	Cmps	Movs	Combined All = 1 cycle	Combined Pentium
48kHz	2.33	1	2.33	1	6.66	20.33
96kHz	1.66	1	1.66	1	5.33	19.66

C.7. Estimated total complexity of an SLS decoder.

The estimated maximal complexity (per sample per channel) of stereo-mode IntMDCT + bit-sliced arithmetic decoder (modules required for decoding of SLS bitstream without AAC core layer) is shown in the following tables:

Table C.8. Maximum numbers of INT32 operations in SLS's IntMDCT + Arithmetic decoder (DCT tables pre-unpacked).

N	Muls	Adds/ Subs	Ors	Shifts	Negs	Movs	Combined All = 1 cycle	Combined Pentium
4096 or 512	18	86.63	34.50	67.10	29.93	9.54	245.7	600.7
2048 or 256	18.75	84.39	31.50	68.04	27.92	9.29	239.89	611.21
1024 or 128	16.5	82.16	28.5	63.00	26.52	8.99	225.67	556.09

Table C.9. Maximum numbers of INT32 operations in SLS's IntMDCT + Arithmetic decoder (DCT tables are unpacked in place).

N	Muls	Adds/ Subs	Ors	Shifts	Negs	Movs	Combined All = 1 cycle	Combined Pentium
4096 or 512	18	116.38	34.50	82.6	29.93	9.54	290.05	677.575
2048 or 256	18.75	110.64	31.50	80.04	27.92	9.29	278.14	672.335
1024 or 128	16.5	98.41	28.5	69.5	26.52	8.99	248.42	590.21

The estimated maximal complexity (per sample per channel) of decoding SLS bitstream, containing the 64Kbps/channel AAC core stream at 48kHz (assuming that it requires Huffman decoding + inverse error mapping + M/S stereo decoding) is shown in the following table:

Table C.10. Estimated maximum numbers of INT32 operations in SLS decoder (AAC core present) (DCT tables pre-unpacked).

N	Muls	Adds/ Subs	Ors	Shifts	Negs	Movs	Combined All = 1 cycle	Combined Pentium
4096 or 512	19.5	93.46	34.5	72.6	34.26	16.54	270.86	652.78
2048 or 256	20.25	91.22	31.5	73.54	32.25	16.29	265.05	663.29
1024 or 128	18	88.99	28.5	68.5	30.85	15.99	250.83	608.165

Table C.11. Estimated maximum numbers of INT32 operations in SLS decoder (AAC core present) (DCT tables are unpacked in place).

N	Muls	Adds/ Subs	Ors	Shifts	Negs	Movs	Combined All = 1 cycle	Combined Pentium
4096 or 512	19.5	123.21	34.5	88.01	34.26	16.54	316.1	729.655
2048 or 256	20.25	117.47	31.5	85.54	32.25	16.29	303.3	724.415
1024 or 128	18	105.24	28.5	75	30.85	15.99	273.58	642.29

C.8. SLS memory usage

The main tables used by SLS decoder are shown below:

sineDataCompact 512 INT32 entries = 2 Kbytes
sineDataCompact_cs 512 INT32 entries = 2 Kbytes
KBDWindow 4K INT32 entries = 16 Kbytes
KBDWindow_cs 4K INT32 entries = 16 Kbytes .

invQuantCompact 1025 INT32 entries = 4Kbytes
thrCompact 192 INT32 entries = 0.7Kbytes

AAC Huffman codebook: 995 32-bit words = 4 Kbytes (see N2957)

Totals assuming dynamic unpacking of DCT tables:
Total for lossless-only mode = 2K+2K = 4K bytes (only sineData tables are used)
Total SLS decoder = 4K+32K+5K + 4K = 45K bytes.

The SLS decoder complexity can be further reduced by pre-unpacking of DCT cosine tables (see C.2), where table sineDataCompact and sineDataCompact_cs can be pre-unpack into the following tables:

sineData	4K INT32 entries	=	16 Kbytes
sineData_cs	4K INT32 entries	=	16Kbytes

This will requires additional 2*16K = 32K bytes of RAM.

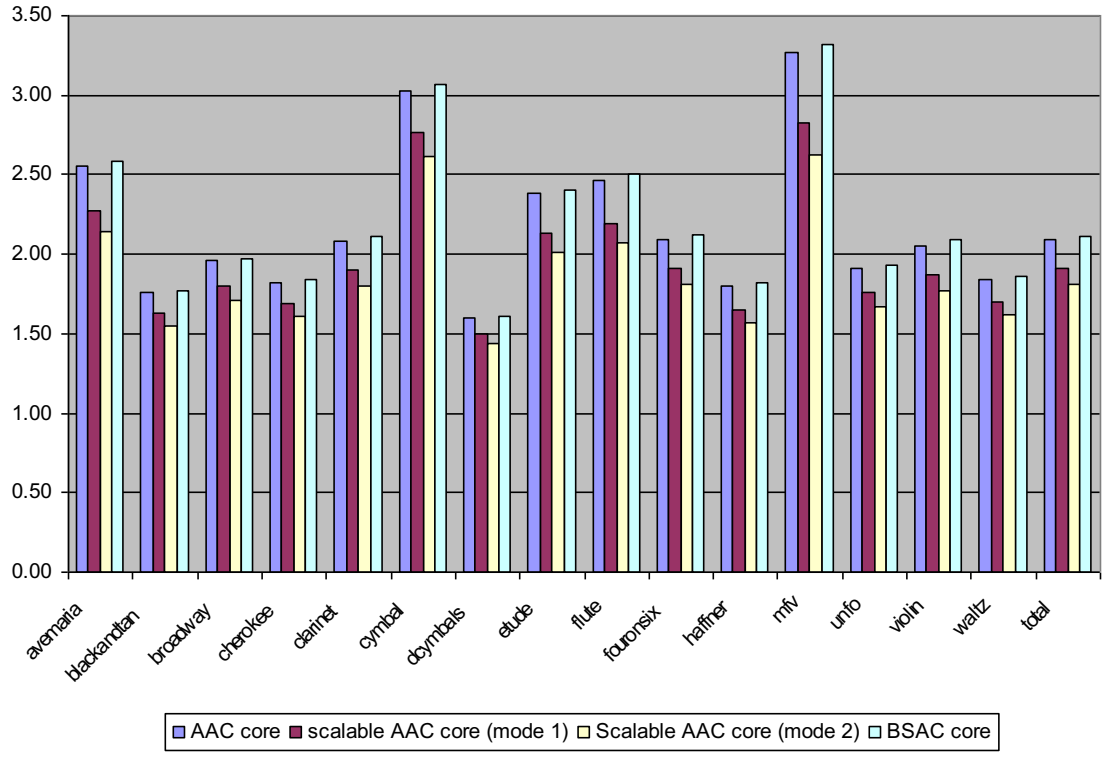
Appendix D. Integration of SLS with MPEG-4 Scalable AAC and BSAC

The coding efficiency performance for SLS integrated with Scalable AAC and BSAC are report here. For SLS+Scalable AAC, we use the following settings for the core Scalable AAC bit-stream:

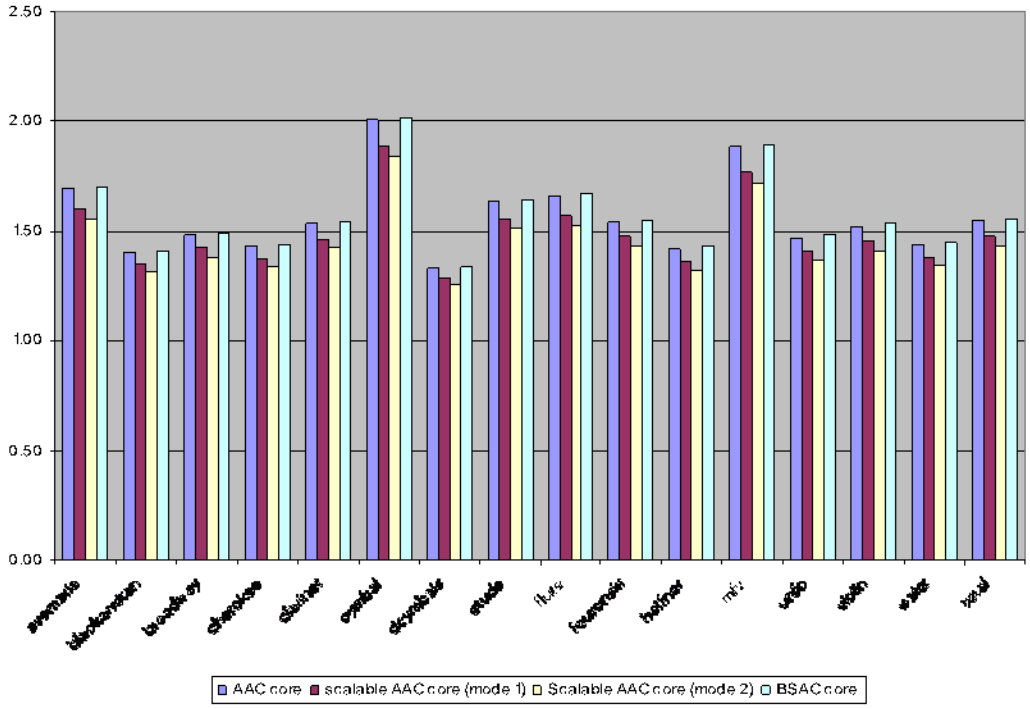
Test 1: Base AAC layer 64kbps mono + 2 extension AAC layers at 32kbps stereo each; total bit-rate is 128kbps.

Test 2: Base AAC layer 32kbps stereo + 3 extension AAC layers at 32kbps stereo each; total bit-rate is 128kbps.

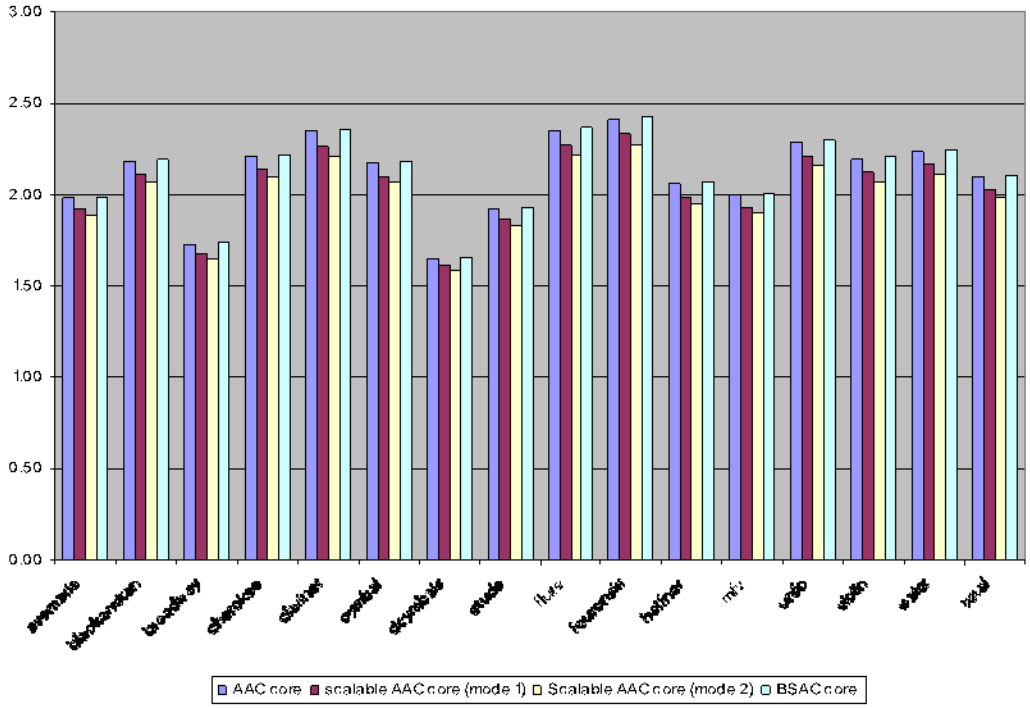
For BSAC, the bit-rate for the core BSAC bit-stream has a bit-rate of 128kbps, two channels. The SBA (Segmental Binary Arithmetic coding) mode of the BSAC is disabled.



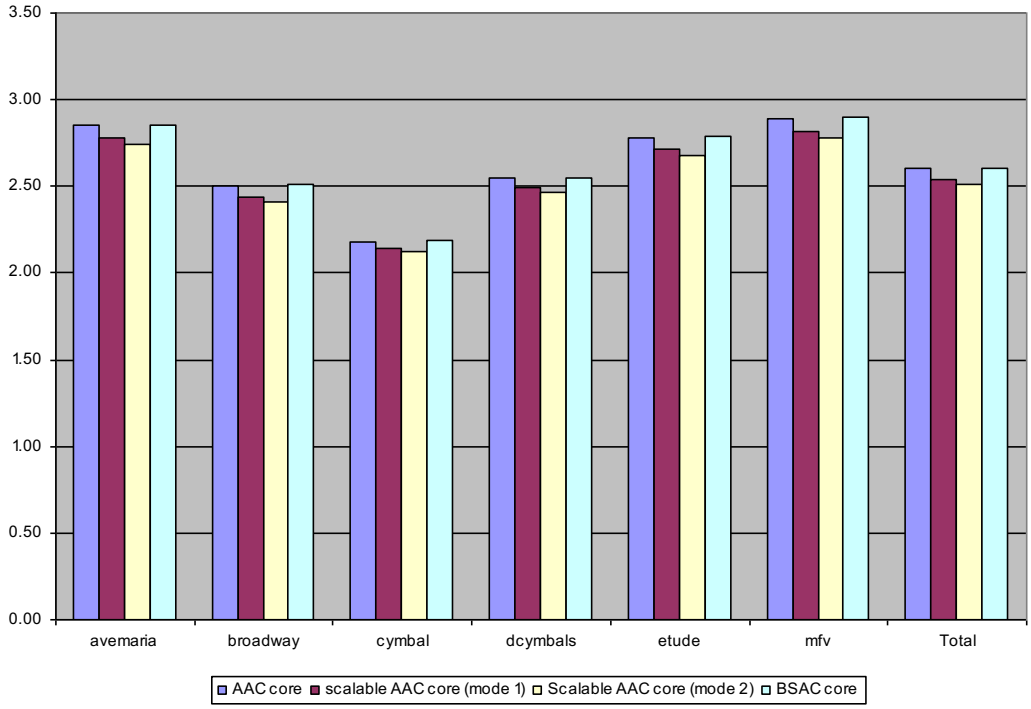
a) 48 kHz/16 bit



b) 48 kHz/24 bit



c) 96 kHz/24 bit



d) 192 kHz/24 bit

Figure D.1. Compression ratio performance for SLS integrated with Scalable AAC and BSAC