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

ISO/IEC JTC1/SC29/WG11 N2005
MPEG98
February. 1998/San Jose

Source: **Audio Subgroup**
Title: **Revised Report on Complexity of MPEG-2 AAC Tools**
Status: **Approved**
Author: **S. R. Quackenbush and Y. Toguri**

This document is based on document N1712 [1] , which presented the complexity of the tools in MPEG-2 AAC Main and Low Complexity Profiles. We add to this an analysis of the complexity of AAC Scaleable Sampling Rate (SSR) Profile, so that this document presents a unified report on the complexity of all tools in MPEG-2 AAC profiles.

We desire to quantify the complexity of the tools in the MPEG-2 Advanced Audio Coding (AAC) decoder. They are:

- huffman decoding
- inverse quantization and scaling
- M/S dematrixing
- intensity stereo
- coupling channel
- backward adaptive prediction
- temporal noise shaping (TNS)
- inverse modified discrete cosine transform (IMDCT)
- gain control and hybrid filter bank (inverse polyphase quadrature filter (IPQF)+IMDCT)

Unless otherwise indicated, complexity is specified in terms of

- machine instructions required to realize the tool’s computations, as run on a typical (but unspecified) programmable digital signal processor
- read/write storage locations
- read-only storage locations

We assume that:

- the target machine uses only IEEE floating point arithmetic, so that all floating point data require four bytes of storage. All storage is specified in terms of 32-bit words.
- the coder block size is 1024 input samples, equivalent to 1024 spectral coefficients per channel.
- an audio signal is sampled at 48 kHz, 16-bits per sample
- the compressed bit rate is 64000 bits per second per audio channel

Furthermore, we only indicate storage that is required by a tool and cannot be shared or re-used by other tools. Specifically, we do not count temporary, stack-based scratch storage (“automatic” variables), as such storage is implicitly shared across tools.

Unless explicitly indicated, all complexity figures are for one audio channel.

Overview

One should consider two important categories of AAC decoder implementations: software decoders running on general-purpose processors, and hardware decoders running on single-chip ASICs. For these two categories the data presented in this document, augmented by demonstrated real-time software decoder implementations, can be summarized in the following table:

Decoder	Complexity
2-channel Main profile software decoder	40 % of 133 MHz Pentium
2-channel Low Complexity profile software decoder	25 % of 133 MHz Pentium
5-channel Main profile hardware decoder	90 sq. mm die, 0.5 micron CMOS
5-channel Low Complexity profile hardware decoder	60 sq. mm die, 0.5 micron CMOS
2-channel Scaleable Sampling Rate profile software decoder	not estimated
5-channel Scaleable Sampling Rate profile hardware decoder	53 sq. mm die, 0.5 micron CMOS (3 band decoder)

In this Table, the 3-band SSR decoder complexity is shown. The 4-band SSR decoder has almost the same complexity as Low Complexity Profile.

Specification of AAC Tool Complexity

Input/Output Buffers

Because of the encoder bit reservoir structure, a real-time decoder receiving a bitstream over a constant-rate channel must, to accomodate worst case buffering conditions, collect a number of input bits equal to the nominal rate per block plus the size of the encoder bit buffer before it can start decoding. This constraint specifies the minimum input buffer size. On output, we assume that the IMDCT result is copied to a 16-bit PCM output buffer in a conventional double-buffered manner.

Table 1 Input/Output Buffer Storage Requirements

	Bits	Words
Input buffer	6144	192
Output buffer (two 16-bit values per word)		512
Totals		704

Huffman Decode

In order to decoding a Huffman codeword the decoder must traverse a Huffman code tree from “root node” to “terminal node” (or leaf). The route taken depends on the Huffman codeword that is being decoded: if the next bit to be processed in the codeword is a “zero” then the “left” branch is taken relative to the current node; otherwise the “right” branch is taken. The decoder must be at the root note when it begins processing a new Huffman codeword, and should be at a terminal node when the entire codeword has been processed. The code fragment that does this processing is

```
v = *p;
while (v & Tnleaf) {
        if (cword & 1)
                p++;
        else
                p += v & (Tnleaf-1);
        v = *p;
        cword >>= 1;
}
```

where to start p points to the root node, cword contains the Huffman codeword to process (lsb first) and Tnleaf is a mask equal to 0x8000 that signals a terminal node. Based on this code it requires approximately 10 instructions per bit for the Huffman decoding. Table 2 shows the instruction complexity for both peak bits per block (3.5 times average) and average bits per block. The summary statistics use the complexity for average bits per block because, in the case of a software-only decoder, there are software speed-ups that can be used to reduce that complexity to 2 instructions per bit (using additional tables) and in the case of an ASIC decoder, the huffman decoding is highly amenable to hardware acceleration.

Pulse lossless coding follows the Huffman decode of the quantized spectral coefficients. It has a very simple reconstruction algorithm as follows:

```
k = start;
for (i=0; i<=number_pulse; i++) {
        k += pulse_offset[i];
        if (quant_coef[k] > 0) {
                quant_coef[k] += pulse_amp[i];
        }
        else {
                quant_coef[k] += pulse_amp[i];
        }
}
```

The bitstream syntax permits “number_pulse” to be no greater than 4 and the loop requires no more than 10 instructions per iteration, so the instruction complexity for pulse lossless coding is no more than 40 instructions per block, as indicated in Table 2. Based on figures for peak compression (50 bits per block or 4%) and average compression (0.25 percent), a value of one tenth the peak complexity is used to approximate the average complexity.

Table 2 Huffman Decoding Instruction Complexity

Channel rate (bps)		64000				Instruct.
Sample rate		48000				
Block length		1024				
				peak		average
Bits per block				4778.7		1365.3
Instructions per bit				10		10
Pulse lossless coding				40		4
Totals				47827		13657

The Huffman codewords can represent signed or unsigned values.
Table 3 shows the storage complexity for the Huffman codebooks in which spectrum tables 1, 2, 5 and 6 are signed.

Huffman decoding requires the storage of the tree and the value corresponding to the codeword. Interior notes must store an offset to the child nodes. The size of this offset does not have to be any larger than the total number of nodes in the table. In
Table **3** the offset is 8 or 16 bits. Furthermore, the offset to the left child can be implicit (it can always follow the parent) so only one offset must be stored. At the terminal notes instead of storing an offset, the decoded value is stored, in compressed form if necessary.

Table 3 Huffman Decoding Read-Only Storage

Huffman Table			Leaves	Nodes	Wds/Nd	Words
Scale factor			121	242	0.25	61
Spectrum	LAV	Tuple				
1	1	4	81	162		41
2	1	4	81	162		41
3	2	4	81	162		41
4	2	4	81	162		41
5	4	2	81	162		41
6	4	2	81	162		41
7	7	2	64	128		32
8	7	2	64	128		32
9	12	2	169	338	0.5	169
10	12	2	169	338		169
11	16	2	289	578		289
Totals				2724		995

Inverse Quantization and Scaling

Each coefficient must be inverse quantized by a 4/3 power nonlinearity and then scaled by the quantizer stepsize. Since the range of values represented by the decoded Huffman values is limited by the codebook itself (except for the escape codebook), the inverse quantization can be done by table lookup. The stepsize, or scale factor, is itself logarithmicaly encoded and is similarly limited in dynamic range, so that it can be decoded by a table lookup as well. We assume that only 854 spectral coefficients (20 kHz bandwidth) must be inverse quantized and scaled by a scale factor. This is summarized in Table 4.

Table 4 Inverse Quantization and Scale Factor Complexity

Block len	1024					
			Read-Only Storage			Instructions
Inverse quantation			128			854
Stepsize scaling			128			854
Totals			256			1708

M/S Synthesis

This is a very simple tool that couples two channels into a stereo pair. For each sample in each channel of the stereo pair the samples may already be the left and right signals, in which case no computation is necessary, or the pair must be de-matrixed via one add and one subtract per pair of samples. Since the computation is done in-place, there is no additional storage requirements. It is assumed that only a 20 kHz bandwidth needs the M/S computation. This is summarized in Table 5.

Table 5 M/S Synthesis Complexity

Block length		1024				
Instructions per block per stereo pair						854
Storage per block						0

Intensity Stereo

In this tool a region of coefficients for a stereo pair is identical except for a “position” scaling of the coefficients of the second channel in the pair. Even though intensity stereo saves bits, the encoder will allocate those bits elsewhere (which is the point of intensity stereo compression) such that the huffman decoding comlexity is unchanged. Similarly, even though the right channel of intensity stereo coded regions do not have scale factors, they do have intensity stereo position factors that require the same decoding complexity. Left-channel intensity stereo regions must have inverse quantization and scaling applied. Right-channel intensity stereo regions use the left-channel inverse quantized and scaled coefficients, which must be re-scaled by the intensity position factors. Hence the net complexity of intensity stereo is a *savings* of one inverse quantization per intensity stereo coded coefficient. Intensity stereo does not use any additional read-only or read-write storage. This complexity estimate is summarized in Table 6.

Table 6 Intensity Stero Complexity

Complexity:					per blk:	
			per IS coefficient		min	max
Instruction complexity per stereo pair				-1	0	-854
Read-only memory complexity				0	0	0
Read-write memory complexity				0	0	0

Coupling Channel

The coupling channel is at its core a single channel element. Since bits allocated for the coupling channel are removed from other channels, there is no increase in Huffman decoding complexity. The coupling channel’s intrinsic scaling is approptiate for the first target channel of the set of coupled channels, while the other coupled channels scale factors must be transmitted and decoded. The final stage in the coupling decoding is to add the coupled channel to the target channel in the frequency domain (dependently switched coupling channel) or in the time domain (independently switched coupling channel).

Table 7 shows two cases for typical coupling channel compexity: one dependent coupling channel with three target channels (1 dcc, 3 tc) such as would be used in the Low Complexity profile, and one independent coupling channel and three target channels (1 icc, 3 tc) such as could be used in the Main profile.

Table 7 Coupling Channel Complexity

Max coupling bandwidth				20000		
Max coupling coef.				854		
Max number of coupling channels (cc)				2		
Max number of coupled channels (tc)				5		
				1-dcc, 3-tc		1-icc, 3-tc
Instructions						
	huffman decode			0		0
	inv. quant. and scale for first tc			1708		1708
	scale for subsequent tc			1708		2
	prediction			0		44352
	TNS			8130		13630
	IMDT			0		19968
	coupling mix			2562		3072
	Total			11546		79660
Read-write storage, words				854		1536
Read-only storage, words				0		0

Prediction

The backward-adaptive predictors must run at every block in the decoding process for every coefficient that will ever use prediction. In this analysis we that only the first 672 coefficients will use prediction and that all prediction and coefficient adaptation calculations are done in IEEE floating point arithmetic (although the calculations can be done on a fixed point platform as well). To reduce memory requirements, variables are truncated to 16 bits prior to storage.

Table 8 shows the instruction complexity of the prediction tool, with instruction counts specified for each step in the prediction computation. Table 9 shows the read-write storage required by the prediction tool.

Table 8 Prediction Instruction Complexity

Number of coef. using prediction:			672			
Bandwidth			15750			
Predictor Order			2			
Calculation			Instructions per predictor			Instructions per block
retrieval and inv. quant.			12			
error summation			4			
LMS prediction coef adaption			18			
reflection summation			2			
new prediction coefs (2 div)			8			
quant. for error control			6			
prediction			2			
misc			2			
quant. and storage			12			
Totals			66			44352

Table 9 Prediction Read-Write Storage Complexity

Number of coef. using prediction				672		
Predictor Order				2		
Function				Words per Predictor		Words per Channel
state variables (delay elements)				1		
correlation coefs				1		
variance estimates				1		
Totals				3		2016

TNS

Temporal noise shaping (TNS) has a variable load, depending on the order of its filters and the number of spectral coefficients that are filtered. Table 10 shows the “worst-case” complexity permitted by TNS. Table 11 shows that TNS requires negligible storage.

Table 10 TNS Instruction Complexity

Maximum filter order			12	20
Maximum coefs to filter			672	672
				Instructions
Filter coef inv quant			66	190
Filtering			8064	13440
Totals			8130	13630

Table 11 TNS Storage Requirements

						Words
Read-write storage						0
Read-only storage						
	Filter coef inv quant tables					24

IMDCT

It is assumed that the IMDCT calculation is done in floating point, although fixed point realizations are feasible. The only requirement is that any roundoff noise due to computational error (such as finite word length errors) be less than 1/2 lsb after the transform result is rounded to 16-bit PCM. Fixed point realizations using 24 bit words are certainly adequate, and word lengths as low as 20 or 21 bits may be sufficient. One compromise to this requirement is made in this analysis, which is that the windows used in the overlap-add portion of the transform are stored as 16-bit. This is reasonable since the window and overlap-add is the final computation prior to rounding to 16-bit PCM and therefore computational errors do not accumulate.

Table 12 shows the IMDCT complexity in multiply/add operations per block (1024 samples). Table 13 and Table 14 show the IMDCT complexity in terms of words of read/write and read-only storage. Note that the coefficient storage listed in Table 13 is actually the decoder’s “working storage” and is used by all the tools in the decoder.

Table 12 IMDCT Arithmetic Complexity

M =	1024					Instructions
first modulation			2*M			2048
complex FFT of size			M2 = M/2		512	
	number of bfy		(M2/2)		256	
	operations per bfy		6		6	
	number of stages		log2(M2)		9	
	total = 6*log2(M2)(M2/2)				13824	13824
second modulation			2*M			2048
window and ovlp add			2*M			2048
Total						19968

Table 13 IMDCT Read/Write Storage Requirements

Block len	1024					Words
coefficient storage						1024
state variable storage						512
Totals						1536

Table 14 IMDCT Read-Only Storage Requirements

Block length				128	1024	
				Words	Words	Words
First modulation sin/cos table				64	512	
FFT twiddle table				12	18	
Second modulation sin/cos table					512	
Windows are 16-bit values						
	Sin window table			64		
	Alternate window table			64		
	Dolby window table				512	
	Alternate window table				512	
Total				204	2066	2270

Gain Control and Hybrid Filter Bank (IPQF + IMDCT)

Table 15 shows the Gain-control tool instruction complexity. The Hybrid Filterbank which consists of IPQF and IMDCT is also included in this table. The block size of the IMDCT for the Gain Control tool is 256 in the case of LONG WINDOW and 32 in the case of SHORT_WINDOW. Note that the SSR profile has a scaleable complexity due to the division of the 1024 spectral coefficients into four bands. Therefore an N-band (from 1-band though 4-band) decoder can be implemented. For example, the 3-band SSR decoder needs only three IMDCT operations per frame.
The m*aximum* instructions per channel for the Gain-control tool is shown in Table 15. It is assumed that the instruction count associated with a single IPQF does not depend on the number of implemented bands and that the maximum instructions per single IPQF band of the Gain Window Reconstruction is 896 instructions per band for the case of EIGHT_SHORT_WINDOW.

Table 15: Gain Control Tool Instruction Complexity

M =	256				Instructions			
					1 band	2 band	3 band	4 band
first modulation			2*M	512				
complex FFT of size			M2 = M/2	128				
	number of bfy		(M2/2)	64				
	operations per bfy		6	6				
	number of stages		log2(M2)	7				
	total = 6*log2(M2)(M2/2)			2688				
second modulation			2*M	512				
window and ovlp add			2*M	512				
IMDCT Total				4224	4224	8448	12672	16896
PQF synthesis (96-tap, 4-band split)					28672	28672	28672	28672
Gain Compensation					0	512	1024	1536
Gain Window Reconstruction					0	896	1792	2688
Gain control & Filerbank totals					32896	38528	44160	49792

Summary of Tool Complexity

The following tables summarize the complexity of each tool based on number of instructions, amount of read-write storage and amount of read-only storage for Main profile Low Complexity profile and Scaleable Sampling Rate profile. Storage for the program itself has not been counted. The tables first list complexity on a per-channel basis and then factor this up to get the complexity for a 5-channel coder. Resources scale linearly with some exceptions: M/S joint stereo, intensity stereo and stereo prediction are stereo pair operations and there are only two stereo pairs in a 5-channel system; and obviously read-only memory is a shared resource so that its complexity is the same for 1- and 5-channel coders.

The most revealing data in the tables is the last column, which lists the complexity of a tool's requirements (instructions, read-write storage or read-only storage) as a percentage of the total amount of that resource used in the entire 5-channel coder.

Main Profile

Tables 16 through 19 summarize the complexity of AAC Main profile.

Table 16 Summary of Instruction Complexity

	1-Chan	5-Chan	
	Instr.	Instr.	percent
Huffman, pulse decode	13657	68285	13.3
Inv. quant. and scale	1708	8540	1.7
M/S synthesis		1708	0.3
Prediction	44352	221760	43.2
Coupling channel (1 icc)		79661	15.5
TNS (average)	6815	34075	6.6
IMDCT	19968	99840	19.4
Totals	86500	513869	100.0

Table 17 Summary of Read-Write Storage

	1-Chan	5-Chan	
	Words	Words	percent
Input buffer	192	960	4.5
Output	512	2560	12.0
Working buffer	1024	5120	24.1
Prediction state vars.	2016	10080	47.4
Coupling (1 dcc, 1 icc)		2390	11.2
IMDCT state vars	512	2560	12.0
Totals	4256	21280	100.0

Table 18 Summary of Read-Only Storage

	1-Chan	5-Chan	
	Words	Words	percent
Huffman decode		995	28.1
Inv. quant. and scale		256	7.2
Prediction		0	0.0
TNS		24	0.7
IMDCT		2270	64.0
Totals		3545	100.0

Table 19 lists the estimated area which each tool's resources would consume if the AAC decoder were fabricated as a single-chip device using a 0.5 micron CMOS. The ALU used in this analysis is a MIPS R3000 RISC core with 1K instruction cache, 4K data cache and a fast 32 by 32 (64-bit result) integer multiplier. Each read-write memory cell (bit) is assumed to take six transistors while each read-only memory cell is assumed to take one transistor, so that the area of read-only cells are one sixth the area of read-write cells. Judging from a photo of the R3000 die, the 20 Kbytes of cache memory is 1/3 of the total die area. Therefore, the size of 1 K byte of read-write memory was assumed to be 1/60 of the total die area.

Table 19 Estimated Chip Area Required for Each Tool

		Words	(mm)^2	% of die
Area per 1 Kbyte read-write memory			0.67	
ALU core (less cache memories)			26.67	29.68
Read-Only Memory		Words		
	Huffman tables	995	0.43	0.48
	Inv quant and scaling tables	256	0.11	0.12
	TNS tables	24	0.01	0.01
	IMDCT tables	2270	0.99	1.10
Read-Write Memory				
	Input buffer	960	2.50	2.78
	Output buffer	2560	6.67	7.42
	Working buffer	5120	13.33	14.84
	Prediction state variables	10080	26.25	29.22
	Coupling channel (1 dcc, 1 icc)	2390	6.22	6.93
	IMDCT state variables	2560	6.67	7.42
Totals			89.85	100.00

Low Complexity Profile

Tables 20 through 22 summarize the complexity of the AAC Low Complexity profile. The Low Complexity profile has the following features relative to the Main profile:

- no prediction
- TNS limited to 12 coefficients, but still over an 18 kHz bandwidth

Table 20 Summary of Instruction Complexity, Low Complexity Profile

	1-Chan	5-Chan	
	Instr.	Instr.	percent
Huffman, pulse decode	13657	68285	32.5
Inv. quant. and scale	1708	8540	4.1
M/S synthesis		1708	0.8
Coupling channel (1 dcc)		11546	5.5
TNS (average)	4065	20325	9.7
IMDCT	19968	99840	47.5
Totals	39398	210244	100.0

Table 21 Summary of Read-Write Storage, Low Complexity Profile

			1-Chan		5-Chan	
			Words		Words	percent
Input buffer			192		960	8.6
Output			512		2560	22.9
Working buffer			1024		5120	45.7
Coupling channel (1 dcc)					854	7.6
IMDCT state vars			512		2560	22.9
Totals			2240		11200	100.0

Table 22 Estimated Chip Area Required for Each Tool, Low Complexity Profile

					(mm)^2	% of die
Area per 1 Kbyte read-write memory					0.67	
ALU core (less cache memories)					26.67	44.75
Read-Only Memory				Words		
	Huffman tables			995	0.43	0.72
	Inv quant and scaling tables			256	0.11	0.19
	TNS tables			24	0.01	0.02
	IMDCT tables			2270	0.99	1.65
Read-Write Memory						
	Input buffer			960	2.50	4.19
	Output buffer			2560	6.67	11.19
	Working buffer			5120	13.33	22.37
	Coupling channel (1 dcc)			854	2.22	3.73
	IMDCT state variables			2560	6.67	11.19
Totals					59.60	100.00

Scaleable Sampling Rate Profile

Tables 23 through 26 summarize the complexity of the AAC Scaleable Sampling Rate profile. The Scaleable Sampling Rate profile has the following features relative to the Main profile:

- no prediction
- no coupling channel
- gain control
- Hybrid Filter Bank (IPQF + divided IMDCT)
- TNS is limited to 12 coefficients , and is limited to 6 kHz bandwidth

Table 23: Summary of Instruction Complexity, SSR Profile

			1-Chan	5-Chan		MIPS/ch
			Instr.	instr.		
Huffman, pulse decode			13657	68285		0.64
Inv. quant. and scale			2048	10240	(4 band)	0.10
			1536	7680	(3 band)	0.07
			1024	5120	(2 band)	0.05
			512	2560	(1 band)	0.02
M/S synthesis						
Coupling channel (1 dcc)						
TNS (average)			2946	14730		
IMDCT			16896	84480	(4 band)	0.79
			12672	63360	(3 band)	0.59
			8448	42240	(2 band)	0.40
			4224	21120	(1 band)	0.20
Gain control and PQF			32896	164480	(4 band)	1.54
			31488	157440	(3 band)	1.48
			30080	150400	(2 band)	1.41
			28672	143360	(1 band)	1.34
total			68443	342215	(4 band)	3.21
			62299	311495	(3 band)	2.92
			56155	280775	(2 band)	2.63
			50011	250055	(1 band)	2.34

Table 24: Summary of Read-Write Storage, SSR Profile

			1-Chan		5-Chan	
			Words		Words	
Input buffer			192		960	
Output			512		2560	
Working buffer			1024		5120	(4 band)
			768		3840	(3 band)
			512		2560	(2 band)
			256		1280	(1 band)
Coupling channel (1 dcc)						
IMDCT state vars			512		2560	(4 band)
			384		1920	(3 band)
			256		1280	(2 band)
			128		640	(1 band)
Totals			2240		11200	(4 band)
			1856		9280	(3 band)
			1472		7360	(2 band)
			1088		5440	(1 band)

Table 25: Summary of Read-Only Storage, SSR Profile

			1-Chan		5-Chan	
			Words		Words	percent
Huffman decode					995	52.2
Inv. quant. and scale					256	13.4
Prediction					0	0.0
TNS					24	1.3
IMDCT					582	30.6
Gain control					48	2.5
Totals					1905	100.0

Table 26: Estimated Chip Area Required for Each Tool, SSR Profile

					(mm)^2	% of die
Area per 1 Kbyte read-write memory					0.67	(4band)
ALU core (less cache memories)					26.67	46.06
Read-Only Memory				Words		
	Huffman tables			995	0.43	0.75
	Inv quant and scaling tables			256	0.11	0.19
	TNS tables			24	0.01	0.02
	IMDCT tables			582	0.25	0.44
	Gain control			48	0.02	0.04
Read-Write Memory						
	Input buffer			960	2.50	4.32
	Output buffer			2560	6.67	11.51
	Working buffer		(4 band)	5120	13.33	23.03
			(3 band)	3840	10.00	
			(2 band)	2560	6.67	
			(1 band)	1280	3.33	
	Coupling channel					
	IMDCT state variables		(4 band)	2560	6.67	11.51
			(3 band)	1920	5.00	
			(2 band)	1280	3.33	
			(1 band)	640	1.67	
	Gain control		(4 band)	475	1.24	2.14
			(3 band)	470	1.22	
			(2 band)	465	1.21	
			(1 band)	461	1.20	
Totals			(4 band)		57.90	100.00
			(3 band)		52.88	
			(2 band)		47.87	
			(1 band)		42.86	

References

[1] Report on Complexity of MPEG-2 AAC Tools, N1712, April 1997, Bristol.