A = 37 + cplbegf*12 B = (A+1/2)*0.09375 kHz C = (A+1/2)*0.08613 kHz

256 sample block

band #	low coeff #	high coeff #	fs = 48 kHz low freq (kHz)	fs = 48 kHz high freq (kHz)	fs = 44.1 kHz low freq (kHz)	fs = 44.1 kHz high freq (kHz)
0	7	12	1.22	2.34	1.12	2.15
1	13	18	2.34	3.47	2.15	3.19
2	19	30	3.47	5.72	3.19	5.25
3	31	A	5.72	B	5.25	C

A = 19 + cplbegf*6 B = (A+1/2)*0.1875 kHz C = (A+1/2)*0.1723 kHz

if coupling is used (cplinu == 1) and 2 ≥ cplbegf > 0

nrematbd = 3

512 sample block

band #	low coeff #	high coeff #	fs = 48 kHz low freq (kHz)	fs = 48 kHz high freq (kHz)	fs = 44.1 kHz low freq (kHz)	fs = 44.1 kHz high freq (kHz)
0	13	24	1.17	2.30	1.08	2.11
1	25	36	2.30	3.42	2.11	3.14
2	37	A	3.42	B	3.14	C

A = 37 + cplbegf*12 B = (A+1/2)*0.09375 kHz C = (A+1/2)*0.08613 kHz

256 sample block

band #	low coeff #	high coeff #	fs = 48 kHz low freq (kHz)	fs = 48 kHz high freq (kHz)	fs = 44.1 kHz low freq (kHz)	fs = 44.1 kHz high freq (kHz)
0	7	12	1.22	2.34	1.12	2.15
1	13	18	2.34	3.47	2.15	3.19
2	19	A	3.47	B	3.19	C

A = 19 + cplbegf*6 B = (A+1/2)*0.1875 kHz C = (A+1/2)*0.1723 kHz

if coupling is used (cplinu == 1) and cplbegf == 0

nrematbd = 2

512 sample block

			fs = 48 kHz	fs = 48 kHz	fs = 44.1 kHz	fs = 44.1 kHz
band #	low coeff #	high coeff #	low freq (kHz)	high freq (kHz)	low freq (kHz)	high freq (kHz)
0	13	24	1.17	2.30	1.08	2.11
1	25	36	2.30	3.42	2.11	3.14

256 sample block

			fs = 48 kHz	fs = 48 kHz	fs = 44.1 kHz	fs = 44.1 kHz
band #	low coeff #	high coeff #	low freq (kHz)	high freq (kHz)	low freq (kHz)	high freq (kHz)
0	7	12	1.22	2.34	1.12	2.15
1	13	18	2.34	3.47	2.15	3.19

Note: At 32 kHz sampling rate the subband frequency ranges are 2/3 the values of those for 48 kHz.

7.3. Encoding Technique

If the 2/0 mode is selected, then rematrixing is employed by the encoder. The squares of the transform coefficients are summed up over the previously defined rematixing frequency bands for the following combinations: L, R, L+R, L-R.

```
if(minimum sum for a rematrixing sub-band j is L or R)
{
    the variable rematflg[j] = 0
    transmitted left = input L
    transmitted right = input R
}
if(minimum sum for a rematrixing sub-band j is L+R or L-R)
{
    the variable rematflg[j] = 1
    transmitted left = input (L+R)
    transmitted right = input (L-R)
}
```

This selection of matrix combination is done on a block by block basis

7.4. Decoding Technique

For each band, a single bit is sent in the data stream, indicating whether or not the two channels have been rematrixed for that band. If the bit is clear, no further operation is required. If the bit is set, the AC-3 decoder performs the following operation to restore the individual channels:

left(band n) = received left(band n) + received right(band n)

right(band n) = received left(band n) - received right(band n)

Note that if coupling is not in use, the two channels may have different bandwidths. As such, rematrixing is only applied up to the lower bandwidth of the two channels. Regardless of the actual bandwidth, all four rematrixing flags are sent in the data stream (assuming the rematrixing strategy bit is a one).

8. DIALOG NORMALIZATION AND DYNAMIC RANGE COMPRESSION

The AC-3 syntax provides elements which allow the encoded bit stream to satisfy listeners in many different situations. The dialnorm element allows for uniform reproduction of spoken dialogue when decoding any AC-3 bit stream. The dynrng element allows the program provider to implement subjectively pleasing dynamic range reduction for most of the intended audience, while allowing individual members of the audience the option to experience more (or all) of the original dynamic range. The compr element allows the program provider (or broadcaster) to implement a large dynamic range reduction (heavy compression) in a way which assures that a monophonic downmix will not exceed a certain peak level. The heavily compressed audio program may be desirable for certain listening situations such as movie delivery to a hotel room, or to an airline seat. The peak level limitation is useful when, for instance, a monophonic downmix will feed an RF modulator and overmodulation must be avoided.

8.1. Dialogue Normalization

8.1.1. Overview

When audio from different sources is reproduced, the apparent loudness often varies from source to source. The different sources of audio might be different program segments during a broadcast (i.e. the movie vs. a commercial message); different broadcast channels; or different media (laserdisc vs. VHS tape). The AC-3 coding technology solves this problem by explicitly coding an indication of loudness into the AC-3 bit stream.

The subjective level of normal spoken dialogue is used as a reference. The 5-bit dialogue normalization word which is contained in BSI, dialnorm, is an indication of the subjective loudness of normal spoken dialogue compared to digital 100%. The 5-bit value is interpreted as an unsigned integer (most significant bit transmitted first) with a range of possible values from 0 to 31. The unsigned integer indicates the headroom in dB above the subjective dialogue level. This value can also be interpreted as an indication of how many dB the subjective dialogue level is below digital 100%.

The dialnorm value is not directly used by the AC-3 decoder. Rather, the value is used by the section of the sound reproduction system responsible for setting the reproduction volume, e.g. the system level control. The system level control is generally set based on listener input as to the desired sound pressure level (SPL). The listener adjusts a volume control which generally directly adjusts the reproduction system gain. With AC-3 and the dialnorm value, the reproduction system gain becomes a function of both the listeners desired reproduction sound pressure level for dialogue, and the dialnorm value which indicates the level of dialogue in the audio signal. The listener is thus able to reliably set the volume level of dialogue, and the subjective level of dialogue will remain uniform no matter which AC-3 program is decoded.

Example

The listener adjusts the volume control to 72 dB. (With AC-3 dialogue normalization, it is possible to calibrate a system volume control directly in sound pressure level, and the indication will be accurate for any AC-3 encoded audio source). A high quality entertainment program is being received, and the AC-3 bit stream indicates that dialogue level is 20 dB below 100% digital level. The reproduction system automatically sets the reproduction system gain so that full scale digital signals reproduce at a sound pressure level of 92 dB. The spoken dialogue (down 20 dB) will thus reproduce at 72 dB SPL.

The broadcast program cuts to a commercial message, which has dialogue level at -3 dB with respect to 100% digital level. The system level gain automatically drops, so that digital 100% is now reproduced at 75 dB SPL. The dialogue of the commercial (down 3 dB) reproduces at a 72 dB SPL, as desired.

In order for the dialogue normalization system to work, the dialnorm value must be communicated from the AC-3 decoder to the system level control so that dialnorm can interact with the listener adjusted volume control. If the volume control function for a system is performed as a digital multiply inside the AC-3 decoder, then the listener selected volume setting must be communicated into the AC-3 decoder. The listener selected volume setting and the dialnorm value must be brought together an combined in order to adjust the final reproduction system gain.

Adjustment of the system volume control is not an AC-3 function. The AC-3 bit stream simply conveys useful information which allows the system volume control to be implemented in a way which automatically removes undesirable level variations between program sources. It is mandatory that the dialnorm value and the user selected volume setting both be used to set the reproduction system gain.

8.2. Dynamic Range Control

8.2.1. Overview

A consistent problem in the delivery of audio programming is that different members of the audience wish to enjoy different amounts of dynamic range. Original high quality programming (such as feature films) are typically mixed with quite a wide dynamic range. Using dialogue as a reference, loud sounds like explosions are often 20 dB or more louder, and faint sounds like leaves rustling may be 50 dB quieter. In many listening situations it is objectionable to allow the sound to become very loud, and thus the loudest sounds must be compressed downwards in level. Similarly, in many listening situations the very quiet sounds would be inaudible, and must be brought upwards in level to be heard. Since most of the audience will benefit from a limited program dynamic range, soundtracks which have been mixed with a wide dynamic range are generally compressed: the dynamic range is reduced by bringing down the level of the loud sounds and bringing up the level of the quiet sounds. While this satisfies the needs of much of the audience, it removes the ability of some in the audience to experience the original sound program in its intended form. The AC-3 audio coding technology solves this conflict by allowing dynamic range control values to be placed into the AC-3 bit stream.

The dynamic range control values, dynrng, indicate a gain change to be applied in the decoder in order to implement dynamic range compression. Each dynrng value can indicate a gain change of ± 24 dB. The sequence of dynrng values are a compression control signal. An AC-3 encoder (or a bit stream processor) will generate the sequence of dynrng values. Each value is used by the AC-3 decoder to alter the gain of one or more audio blocks. The dynrng values typically indicate gain reduction during the loudest signal passages, and gain increases during the quiet passages. For the listener, it is desirable to bring the loudest sounds down in level towards dialogue level, and the quiet sounds up in level, again towards dialogue level. Sounds which are at the same loudness as the normal spoken dialogue will typically not have their gain changed.

The compression is actually applied to the audio in the AC-3 decoder. The encoded audio has full dynamic range. It is permissible for the AC-3 decoder to (optionally, under listener control) ignore the dynrng values in the bit stream. This will result in the full dynamic range of the audio being reproduced. It is also permissible (again under listener control) for the decoder to use some fraction of the dynrng control value, and to use a different fraction of positive or negative values. The AC-3 decoder can thus reproduce either fully compressed audio (as intended by the compression control circuit in the AC-3 encoder); full dynamic range audio; or audio with partially compressed dynamic range, with different amounts of compression for high level signals and low level signals.

Example

A feature film soundtrack is encoded into AC-3. The original program mix has dialogue level at -25 dB. Explosions reach full scale level of 0 dB. Some quiet sounds which are intended to be heard by all listeners are 50 dB below dialogue level (or -75 dB). A compression control signal (sequence of dynrnge values) is generated by the AC-3 encoder. During those portions of the audio program where the audio level is higher than dialogue level the dynrnge values indicate negative gain, or gain reduction. For full scale 0 dB signals (the loudest explosions), gain reduction of -15 dB is encoded into dynrng. For very quiet signals, a gain increase of 20 dB is encoded into dynrng.

A listener wishes to reproduce this soundtrack quietly so as not to disturb anyone, but wishes to hear all of the intended program content. The AC-3 decoder is allowed to reproduce the default, which is full compression. The listener adjusts dialogue level to 60 dB SPL. The explosions will only go as loud as 70 dB (they are 25 dB louder than dialogue but get a -15 dB gain applied), and the quiet sounds will reproduce at 30 dB SPL (20 dB of gain is applied to their original level of 50 dB below dialogue level). The reproduced dynamic range will be 70 - 30 = 40 dB.

The listening situation changes, and the listener now wishes to raise the reproduction level of dialogue to 70 dB SPL, but still wishes to limit how loud the program plays. Quiet sounds may be allowed to play as quietly as before. The listener instructs the AC-3 decoder to continue using the dynrng values which indicate gain reduction, but to attenuate the values which indicate gain increases by a factor of _. The explosions will still reproduce 10 dB above dialogue level, or 80 dB SPL. The quiet sounds are now increased in level by 20 dB / 2 = 10 dB. They will now reproduce at 40 dB below dialogue level, or -30 dB SPL. The reproduced dynamic range is now 80 - 30 = 50 dB.

Another listener wishes the full original dynamic range of the audio. This listener adjusts the reproduced dialogue level to 85 dB SPL, and instructs the AC-3 decoder to ignore the dynamic range control signal. For this listener the quiet sounds reproduce at 35 dB SPL, and the explosions hit 110 dB SPL. The reproduced dynamic range is 110 - 35 = 75 dB. This reproduction is exactly as intended by the original program producer.

In order for this dynamic range control method to be effective, it must be used by program providers. Since all broadcasters wish to supply programming in the form that is most usable by their audience, nearly all broadcasters will apply dynamic range compression to any audio program which has a wide dynamic range. This compression is not reversible unless it is implemented by the technique imbedded in AC-3. If broadcasters make use of the imbedded AC-3 dynamic range control system, then listeners can have some control over their reproduced dynamic range. Broadcasters will only use the AC-3 compression system if they are confident that the compression the introduce will be reproduced by default. Therefore, the AC-3 decoder must, by default, implement the compression characteristic indicated by the dynrng values in the data stream. AC-3 decoders may optionally allow listener control over the use of the dynrng values, so that the listener may select full or partial dynamic range reproduction.

8.2.2. Detailed Implementation

The dynrng field in the AC-3 data stream is 8-bits in length. The dynrng value may be present in any audio block. When the value is not present, the value from the previous block is used, except for block 0. In the case of block 0, if a new value of dynrng is not present, then a value of 0000 0000 should be used. The most significant bit of dynrnge is transmitted first. The first three bits indicate gain changes in 6.02 dB increments which can be implemented with an arithmietic shift operation. The following five bits indicate linear gain changes, and require a 6-bit multiply. We will represent the 3 and 5 bit fields of dynrng as following:

$$X_0\, X_1\, X_2\, .\, Y_3\, Y_4\, Y_5\, Y_6\, Y_7$$

The meaning of the X values is most simply described by considering X to represent a 3-bit signed integer with values from -4 to 3. The gain indicated by X is then (X+1) * 6.02 dB. The following table shows this in detail.

X_0	X_1	X_2	Integer Value	Gain Indicated	Arithmetic Shifts
0	1	1	3	+24.08 dB	4 left
0	1	0	2	+18.06 dB	3 left
0	0	1	1	+12.04 dB	2 left
0	0	0	0	+6.02 dB	1 left
1	1	1	-1	0 dB	none
1	1	0	-2	-6.02 dB	1 right
1	0	1	-3	-12.04 dB	2 right
1	0	0	-4	-18.06 dB	3 right

The value of Y is a linear representation of a gain change of up to 6 dB. Y is considered to be an unsigned fractional integer, with a leading value of 1, or: $0.1Y_3Y_4Y_5Y_6Y_7$ (base 2). Y can represent values between 0.111111_2 (or 63/64) and 0.100000_2 (or 1/2). Y can represent gain changes from -6.02 dB to -0.14 dB.

The combination of X and Y values allows dynrng to indicate gain changes from 24.08 - 0.14 = +23.94 dB, to -18.06 - 6 = -24.06 dB. The bit code of 0000 0000 indicates 0 dB (unity) gain.

Partial Compression

The dynrng value may be operated on in order to make it represent a gain change which is a fraction of the original value. In order to alter the amount of compression which will be applied, consider the dynrng to represent a signed fractional number, or:

$$X_0\, .\, X_1\, X_2\, Y_3\, Y_4\, Y_5\, Y_6\, Y_7$$

where X_0 is the sign bit and $X_1\, X_2\, Y_3\, Y_4\, Y_5\, Y_6\, Y_7$ are a 7 bit fraction. This 8 bit signed fractional number may be multiplied by a fraction indicating the fraction of the original compression to apply. If this value is multiplied by 1/2, then the compression range of ± 24 dB will be reduced to ± 12 dB. After the multiplicative scaling, the 8-bit result is once again considered to be of the original form $X_0\, X_1\, X_2\, .\, Y_3\, Y_4\, Y_5\, Y_6\, Y_7$ and used normally.

8.3. Compression

8.3.1. Overview

Some products which decode the AC-3 bit stream will need to deliver the resulting audio via a link with very restricted dynamic range. One example is the case of a television signal decoder which must modulate the received picture and sound onto an RF channel in order to deliver a signal usable by a low cost television receiver. In this situation, it is necessary to restrict the maximum peak output level to a known value with respect to dialogue level in order to prevent overmodulation. Most of the time, the dynamic range control signal, dynrng, will produce adequate gain reduction so that the absolute peak level will be constrained. However, since the dynamic range control system is intended to implement a subjectively pleasing reduction in the range of perceived loudness, there is no assurance that it will control instantaneous signal peaks adequately to prevent overmodulation.

In order to allow the decoded AC-3 signal to be constrained in peak level, a second control signal, compr, may be present in the AC-3 data stream. This control signal should be present in all bit streams which are intended to be receivable by, for instance, a television set top decoder. The compr control signal is similar to the dynrng control signal in that it is used by the decoder to alter the reproduced audio level. The compr control signal has twice the control range as dynrng (±48dB compared to ±24 dB) with 1/2 the resolution (0.5 dB vs. 0.25 dB). Also, since the compr control signal lives in BSI, it only has a time resolution of an AC-3 frame instead of a block.

Products which require peak audio level to be constrained should use compr instead of dynrng when compr is present in BSI. Since most of the time the use of dynrng will prevent large peak levels, the AC-3 encoder may only need to insert compr occasionally, during those instants when the use of dynrng would lead to excessive peak level.

In some applications of AC-3, some receivers may wish to reproduce a very restricted dynamic range. In this case, the compr control signal may be present at all times. Then, the use of compr instead of dynrng will allow the reproduction of audio with very limited dynamic range. This might be useful, for instance, in the case of audio delivery to a hotel room or an airplane seat.

8.3.2. Detailed Implementation

The compr field in the AC-3 data stream is 8-bits in length. The most significant bit is transmitted first. The first four bits indicate gain changes in 6.02 dB increments which can be implemented with an arithmietic shift operation. The following four bits indicate linear gain changes, and require a 5-bit multiply. We will represent the two 4-bit fields of dynrng as following:

$$X_0\, X_1\, X_2\, X_3\, .\, Y_4\, Y_5\, Y_6\, Y_7$$

The meaning of the X values is most simply described by considering X to represent a 4-bit signed integer with values from -8 to +7. The gain indicated by X is then (X+1) * 6.02 dB. The following table shows this in detail.

X_0	X_1	X_2	X_3	Integer Value	Gain Indicated	Arithmetic Shifts
0	1	1	1	7	+48.16 dB	8 left
0	1	1	0	6	+42.14 dB	7 left
0	1	0	1	5	+36.12 dB	6 left
0	1	0	0	4	+30.10 dB	5 left
0	0	1	1	3	+24.08 dB	4 left

0	0	1	0	2	+18.06 dB	3 left
0	0	0	1	1	+12.04 dB	2 left
0	0	0	0	0	+6.02 dB	1 left
1	1	1	1	-1	0 dB	none
1	1	1	0	-2	-6.02 dB	1 right
1	1	0	1	-3	-12.04 dB	2 right
1	1	0	0	-4	-18.06 dB	3 right
1	0	1	1	-5	-24.08 dB	4 right
1	0	1	0	-6	-30.10 dB	5 right
1	0	0	1	-7	-36.12 dB	6 right
1	0	0	0	-8	-42.14 dB	7 right

The value of Y is a linear representation of a gain change of up to -6 dB. Y is considered to be an unsigned fractional integer, with a leading value of 1, or: $0.1Y_4Y_5Y_6Y_7$ (base 2). Y can represent values between 0.11111_2 (or 31/32) and 0.10000_2 (or 1/2). Y can represent gain changes from -6.02 dB to -0.28 dB.

The combination of X and Y values allows dynrng to indicate gain changes from 48.16 - 0.28 = +47.88 dB, to -42.14 - 6 = -48.14 dB.

9. DOWNMIXING

9.1. Downmixing Equations

Let L, C, R, Ls, Rs refer to the 5 discrete channels which are to be mixed down to 2 channels. In the case of a single surround channel (n/1 modes), S refers to the single surround channel. Two types of downmix should be provided: downmix to an LtRt matrix surround encoded stereo pair; and downmix to a conventional stereo signal, LoRo. The conventional stereo downmixed signal (LoRo) may be further mixed to mono, M, by a simple summation of the 2 channels. The LtRt downmix should not be combined to mono because the surround information would be lost. Downmix coefficients should be accurate to ± 0.25 dB.

Prior to the scaling needed to prevent overflow, the general 3/2 downmix equations for an LoRo stereo signal are:

Lo = 1.0 L + clev C + slev Ls
Ro = 1.0 R + clev C + slev Rs

If LoRo are subsequently combined for monophonic reproduction, the effective mono downmix equation becomes:

M = 1.0 L + 2.0 clev C + 1.0 R + slev Ls + slev Rs

If only a single surround channel, S, is present (3/1 mode) the downmix equations are:

Lo = 1.0 L + clev C + 0.7 slev S
Ro = 1.0 R + clev C + 0.7 slev S
M = 1.0 L + 2.0 clev C + 1.0 R + 1.4 slev S

The values of clev and slev are indicated by the cmixlev and surmixlev bit fields in the BSI data, as shown in the following tables:

cmixlev	clev
0 0	0.707 (-3 dB)
0 1	0.596 (-4.5 dB)
1 0	0.500 (-6.0 dB)
1 1	0.596 (-4.5 dB) (Reserved)

surmixlev	slev	0.7 slev
0 0	0.707 (-3 dB)	0.500 (-6 dB)
0 1	0.500 (-6 dB)	0.354 (-9 dB)
1 0	0.000	0.000
1 1	0.500 (-6 dB) (Reserved)	0.354 (-9 dB)

If the cmixlev or surmixlev bit fields indicate the reserved state (value of 1 1), the decoder should use the intermediate coefficient values indicated by the bit field value of 0 1. If the Center channel is missing (2/1 or 2/2 mode), the same equations may be used without the C term. If the surround channels are missing, the same equations may be used without the Ls,Rs, or S terms.

Prior to the scaling needed to prevent overflow, the 3/2 downmix equations for an LtRt stereo signal are:

Lt = 1.0 L + 0.707 C - 0.707 Ls - 0.707 Rs
Rt = 1.0 R + 0.707 C + 0.707 Ls + 0.707 Rs

If only a single surround channel, S, is present (3/1 mode) these equations become:

Lt = 1.0 L + 0.707 C - 0.707 S
Rt = 1.0 R + 0.707 C + 0.707 S

If the center channel is missing (2/2 or 2/1 mode) the C term is dropped.

The actual coefficients used must be scaled downwards so that arithmetic overflow does not occur if all channels contributing to a downmix signal happen to be at full scale. For each audio coding mode, a different number of channels contribute to the downmix, and a different scaling could be used to prevent overflow. For simplicity, the scaling for the worst case may be used in all cases. This minimizes the number of coefficients required. The worst case scaling occurs when cmixlev and smixlev are both 0.707. In the case of the LoRo downmix, the sum of the unscaled coefficients is 1 + 0.707 + 0.5 = 2.207, so all coefficients must be multiplied by 1 / 2.207 = 0.453 (downwards scaling by 6.88 dB). In the case of the LtRt downmix, the sum of the unscaled coefficients is 1 + 0.707 + 0.707 + 0.707 = 3.121, so all coefficients must be multiplied by 1 / 3.121, or 0.3204 (downwards scaling by 9.89 dB). The scaled coefficients will typically be converted to binary values with limited wordlength. The 6-bit coefficients shown below have sufficient accuracy.

In order to implement the LoRo 2-channel downmix, scaled (by 0.453) coefficient values are needed which correspond to the values of 1.0, 0.707, 0.596, 0.500, 0.354.

Table of 6-bit scaled coefficient values for LoRo downmix.

Unscaled Coefficient	Scaled Coefficient	Gain	Relative Gain	Coefficient Error
1.0	28/64	-7.2 dB	0.0 dB	—
0.707	20/64	-10.10 dB	-2.9 dB	+0.10 dB
0.596	17/64	-11.5 dB	-4.3 dB	+0.2 dB
0.500	14/64	-13.2 dB	-6.0 dB	0.0 dB
0.354	10/64	-16.1 dB	-8.9 dB	+0.1 dB

In order to implement the LtRt 2-ch downmix, scaled (by 0.3204) coefficient values are needed which correspond to the values of 1.0 and 0.707.

Table of 6-bit scaled coefficient values for LtRt downmix.

Unscaled Coefficient	Scaled Coefficient	Gain	Relative Gain	Coefficient Error
1.0	20/64	-10.1 dB	0.0 dB	—
0.707	14/64	-13.20 dB	-3.1 dB	-0.10 dB

Downmix to a single monophonic channel is presumed to be done in the analog domain external to the AC-3 decoder. If it is necessary to implement a mixdown to mono digitally, a further scaling of 1/2 will have to be applied to the LoRo downmix coefficients to prevent overload of the mono sum of Lo+Ro.

9.2. Implementation Strategy

In an ASIC design, channel downmixing must be accomplished in two phases. First, for uncoupled bands, the input channels that contribute to the target output channel must be appropriately scaled and stored into the inverse transform RAM. For the coupled bands, the coupling coefficients themselves may be downmixed prior to uncoupling, thus reducing the required number of scaling operations. This sequence of operations is as follows:

1) Clear the inverse transform RAM.

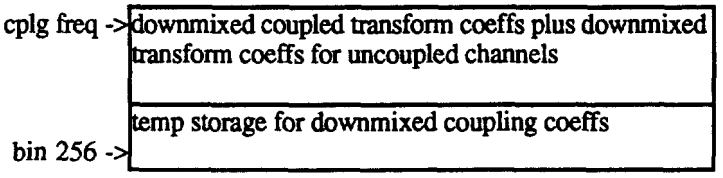
2) Scale the coupling coefficients for each coupled channel by the appropriate downmix coefficient and add to the accumulated downmixed coupling coefficients, stored at the top of the inverse transform RAM. (Note that some channels may not be coupled, and these are not considered in this step).

3) Scale the coupled MDCT/MDST transform coefficients by the downmixed coupling coefficients and store the result in the appropriate bins of the inverse transform RAM.

4) For each channel, downmix the uncoupled MDCT/MDST transform coefficients by the appropriate downmix coefficient and add to the appropriate bins of the inverse transform RAM.

The inverse transform RAM layout and usage is as follows:

bin 0 -> downmixed uncoupled transform coeffs



Note that this strategy requires that the individual channel transform coefficients be routed to the inverse transform RAM at most twice each - once for left and once for right downmixing. Of course, if it is known a priori that a given channel's downmix coefficient is 0.0, that channel does not need to be unpacked and scaled twice. However, certain channels (like the center channel, or the surrounds in Surround downmix mode) may need to be unpacked twice.

10. TRANSFORM EQUATIONS AND BLOCK SWITCHING

10.1. Overview

The choice of analysis block length is fundamental to any transform-based audio coding system. A long transform length is most suitable for input signals whose spectrum remains stationary, or varies only slowly, with time. A long transform length provides greater resolution of frequency, and hence improved coding performance for such signals. On the other hand, a shorter transform length, possessing greater time resolution, is more desirable for signals which change rapidly in time. Therefore, the time vs. frequency resolution tradeoff should be considered when selecting a transform block length.

The traditional approach to solving this dilemma is to select a single transform length which provides the best tradeoff of coding quality for both stationary and dynamic signals. AC-3 employs a more optimal approach, which is to adapt the frequency/time resolution of the transform depending upon spectral and temporal characteristics of the signal being processed. This approach is very similar to behavior known to occur in human hearing. In transform coding, the adaptation occurs by switching the block length in a signal dependent manner.

10.2. Technique

In the AC-3 transform block switching procedure, a block length of either 512 or 256 samples (time resolution of 10.7 or 5.3 ms for sampling frequency of 48 kHz) can be employed. In the encoder, the shorter blocks are constructed by taking the usual 512 sample, windowed audio segment and splitting it into two segments containing 256 samples each. The first half of an MDCT block is transformed separately but identically to the second half of that block. Each half of the block produces 128 unique non zero transform coefficients representing frequencies from 0 to fs/2, for a total of 256. This is identical to the number of coefficients produced by a single 512 sample block, but with two times improved temporal resolution. A similar, mirror image procedure is applied in the decoder during signal reconstruction.

Transitions from one block length to another are seamless, with aliasing cancellation at the boundaries and no increase in required data rate. Audio signals coded using the 256 sample block produce exactly one half of the

number of compressed bits as a 512 sample block, simplifying the task of constructing, editing, and deformatting the compressed bitstream.

Prior to transforming the audio signal from time to frequency domain, the encoder performs an analysis of the spectral and/or temporal nature of the input signal and selects the appropriate block length. This analysis occurs in the encoder only, and therefore can be upgraded and improved without altering the existing base of decoders. A one bit code per transform block (called blksw) is embedded in the bitstream which conveys length information: (blksw = 0 or 1 for 512 or 256 samples, respectively). The decoder uses this information to deformat the bitstream, reconstruct the mantissa data, and apply the appropriate inverse transform equations.

10.3. Decoder Implementation

TDAC transform block switching is accomplished in AC-3 by making an adjustment to the conventional forward and inverse transformation equations for the 256 length transform. The same window and FFT sine/cosine tables used for 512 sample blocks can be reused for inverse transforming the 256 sample blocks; however, the pre and post-FFT complex multiplication step requires an additional 128 table values for the block-switched transform.

Since the input and output arrays for blksw = 1 are exactly one half of the length of those for blksw = 0, the size of the inverse transform RAM and associated buffers is the same with block switching as without.

The adjustments required for inverse transforming the 256 sample blocks are:

1. The input array contains 128 instead of 256 coefficients.
2. The IFFT pre and post-twiddle use a different cosine table, requiring an additional 128 table values (64 cosine, 64 sine).
3. The complex IFFT employs 64 points instead of 128. The same FFT cosine table can be used with subsampling to retrieve only the even numbered entries.
4. The input pointers to the IFFT post-windowing operation are initialized to different start addresses, and operate modulo 128 instead of modulo 256.

10.4. Transformation Equations

512-Sample IMDCT Transform

The following procedure describes the technique used for computing the IMDCT for a single N-length real data block using a single N/4 point complex IFFT with simple pre and post-twiddle operations. These are the inverse transform equations used when the blksw flag is set to zero (indicating absence of a transient and 512 sample transforms).

1) Define the MDCT transform coefficients = X(k), k=0,1,...N/2-1.

2) (Pre-IFFT complex multiply step)

Compute N/4-point complex multiplication product Z(k), k=0,1,...N/4-1 as:

$$Z(k) = [X(N/2-2k-1) + j * X(2k)] * [xcos1(k) + j * xsin1(k)]$$

$$= [X(N/2-2k-1) * xcos1(k) - X(2k) * xsin1(k)] + j * [X(2k) * xcos1(k) + X(N/2-2k-1) * xsin1(k)]$$

xcos1(k) = -cos(2pi*(8k+1)/(8N)), xsin1(k) = -sin(2pi*(8k+1)/(8N))

3) (Complex IFFT step)

Compute N/4-point complex IFFT of Z(k) to generate complex-valued sequence z(n).

4) (Post-IFFT complex multiply step)

Compute N/4-point complex multiplication product y(n), n=0,1,...N/4-1 as:

y(n) = z(n) * [xcos1(n) + j * xsin1(n)]

= [zr(n) * xcos1(n) - zi(n) * xsin1(n)] + j * [zi(n) * xcos1(n) + zr(n) * xsin1(n)]

zr(n) = real[z(n)], zi(n) = imag[z(n)]

xcos1(n) and xsin1(n) as defined in step 2 above

5) (Windowing and de-interleaving step)

Compute windowed time-domain samples x(n) as:

x(2n) = -yi(N/8+n) * w(2n)

x(2n+1) = yr(N/8-n-1) * w(2n+1)

x(N/4+2n) = -yr(n) * w(N/4+2n)

x(N/4+2n+1) = yi(N/4-n-1) * w(N/4+2n+1)

x(N/2+2n) = -yr(N/8+n) * w(N/2-2n-1)

x(N/2+2n+1) = yi(N/8-n-1) * w(N/2-2n-2)

x(3N/4+2n) = yi(n) * w(N/4-2n-1)

x(3N/4+2n+1) = -yr(N/4-n-1) * w(N/4-2n-2)

for n=0,1,...N/8-1

yr(n) = real[y(n)], yi(n) = imag[y(n)]

w(n) is the transform window sequence (see Table 10.1 below)

<u>256-Sample IMDCT Transforms</u>

The following equations should be used for computing the inverse transforms in the case of blksw = 1 (indicating the presence of a transient and two 256 sample transforms).

1) Define the MDCT transform coefficients = X(k), k=0,1,...N/2.

Let X1(k) = X(2k)

Let X2(k) = X(2k+1)

for k=0,1,..N/4-1

2) (Pre-IFFT complex multiply step)

Compute N/8-point complex multiplication products Z1(k) and Z2(k), k=0,1,...N/8-1 as:

Z1(k) = [X1(N/4-2k-1) + j * X1(2k)] * [xcos2(k) + j * xsin2(k)]

= [X1(N/4-2k-1) * xcos2(k) - X1(2k) * xsin2(k)] + j * [X1(2k) * xcos2(k) + X1(N/4-2k-1) * xsin2(k)]

Z2(k) = [X2(N/4-2k-1) + j * X2(2k)] * [xcos2(k) + j * xsin2(k)]

= [X2(N/4-2k-1) * xcos2(k) - X2(2k) * xsin2(k)] + j * [X2(2k) * xcos2(k) + X2(N/4-2k-1) * xsin2(k)]

xcos2(k) = -cos(2pi*(8k+1)/(4N)), xsin2(k) = -sin(2pi*(8k+1)/(4N))

3) (Complex IFFT step)

Compute N/8-point complex IFFTs of Z1(k) and Z2(k) to generate complex-valued sequences z1(n) and z2(n).

4) (Post-IFFT complex multiply step)

Compute N/8-point complex multiplication products y1(n) and y2(n), n=0,1,...N/8-1 as:

y1(n) = z1(n) * [xcos2(n) + j * xsin2(n)]

= [zr1(n) * xcos2(n) - zi1(n) * xsin2(n)] + j * [zi1(n) * xcos2(n) + zr1(n) * xsin2(n)]

y2(n) = z2(n) * [xcos2(n) + j * xsin2(n)]

= [zr2(n) * xcos2(n) - zi2(n) * xsin2(n)] + j * [zi2(n) * xcos2(n) + zr2(n) * xsin2(n)]

zr1(n) = real[z1(n)], zi1(n) = imag[z1(n)]

zr2(n) = real[z2(n)], zi2(n) = imag[z2(n)]

xcos2(n) and xsin2(n) as defined in step 2 above

5) (Windowing and de-interleaving step)

Compute windowed time-domain samples x(n) as:

x(2n) = -yi1(n) * w(2n)

x(2n+1) = yr1(N/8-n-1) * w(2n+1)

x(N/4+2n) = -yr1(n) * w(N/4+2n)

x(N/4+2n+1) = yi1(N/8-n-1) * w(N/4+2n+1)

x(N/2+2n) = -yr2(n) * w(N/2-2n-1)

x(N/2+2n+1) = yi2(N/8-n-1) * w(N/2-2n-2)

x(3N/4+2n) = yi2(n) * w(N/4-2n-1)

x(3N/4+2n+1) = -yr2(N/8-n-1) * w(N/4-2n-2)

for n=0,1,...N/8-1

yr1(n) = real[y1(n)], yi1(n) = imag[y1(n)]

yr2(n) = real[y2(n)], yi2(n) = imag[y2(n)]

w(n) is the transform window sequence (see Table 10.1 below)

10.5. PCM Regeneration

Once the transform coefficients have been inverse transformed, the output PCM data values are regenerated by overlap-adding successive blocks of x(n) data values, using a 256-point overlap offset, and doubling the resultant sum.

Table 10.1: Transform window sequence (w[address]) The address is (10 × A) + B

	B=0	B=1	B=2	B=3	B=4	B=5	B=6	B=7	B=8	B=9
A=0	0.00014	0.00024	0.00037	0.00051	0.00067	0.00086	0.00107	0.00130	0.00157	0.00187
A=1	0.00220	0.00256	0.00297	0.00341	0.00390	0.00443	0.00501	0.00564	0.00632	0.00706
A=2	0.00785	0.00871	0.00962	0.01061	0.01166	0.01279	0.01399	0.01526	0.01662	0.01806
A=3	0.01959	0.02121	0.02292	0.02472	0.02662	0.02863	0.03073	0.03294	0.03527	0.03770
A=4	0.04025	0.04292	0.04571	0.04862	0.05165	0.05481	0.05810	0.06153	0.06508	0.06878
A=5	0.07261	0.07658	0.08069	0.08495	0.08935	0.09389	0.09859	0.10343	0.10842	0.11356
A=6	0.11885	0.12429	0.12988	0.13563	0.14152	0.14757	0.15376	0.16011	0.16661	0.17325
A=7	0.18005	0.18699	0.19407	0.20130	0.20867	0.21618	0.22382	0.23161	0.23952	0.24757
A=8	0.25574	0.26404	0.27246	0.28100	0.28965	0.29841	0.30729	0.31626	0.32533	0.33450
A=9	0.34376	0.35311	0.36253	0.37204	0.38161	0.39126	0.40096	0.41072	0.42054	0.43040
A=10	0.44030	0.45023	0.46020	0.47019	0.48020	0.49022	0.50025	0.51028	0.52031	0.53033
A=11	0.54033	0.55031	0.56026	0.57019	0.58007	0.58991	0.59970	0.60944	0.61912	0.62873
A=12	0.63827	0.64774	0.65713	0.66643	0.67564	0.68476	0.69377	0.70269	0.71150	0.72019
A=13	0.72877	0.73723	0.74557	0.75378	0.76186	0.76981	0.77762	0.78530	0.79283	0.80022
A=14	0.80747	0.81457	0.82151	0.82831	0.83496	0.84145	0.84779	0.85398	0.86001	0.86588
A=15	0.87160	0.87716	0.88257	0.88782	0.89291	0.89785	0.90264	0.90728	0.91176	0.91610
A=16	0.92028	0.92432	0.92822	0.93197	0.93558	0.93906	0.94240	0.94560	0.94867	0.95162
A=17	0.95444	0.95713	0.95971	0.96217	0.96451	0.96674	0.96887	0.97089	0.97281	0.97463
A=18	0.97635	0.97799	0.97953	0.98099	0.98236	0.98366	0.98488	0.98602	0.98710	0.98811
A=19	0.98905	0.98994	0.99076	0.99153	0.99225	0.99291	0.99353	0.99411	0.99464	0.99513
A=20	0.99558	0.99600	0.99639	0.99674	0.99706	0.99736	0.99763	0.99788	0.99811	0.99831
A=21	0.99850	0.99867	0.99882	0.99895	0.99908	0.99919	0.99929	0.99938	0.99946	0.99953
A=22	0.99959	0.99965	0.99969	0.99974	0.99978	0.99981	0.99984	0.99986	0.99988	0.99990
A=23	0.99992	0.99993	0.99994	0.99995	0.99996	0.99997	0.99998	0.99998	0.99998	0.99999
A=24	0.99999	0.99999	0.99999	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000
A=25	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000				

11. ERROR DETECTION

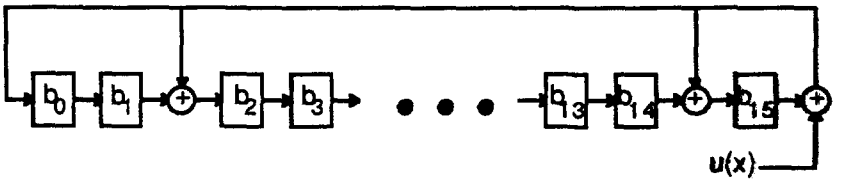
11.1. CRC

Each AC-3 frame contains two 16 bit CRC words. crc1 is the second 16 bit word of the frame, immediately following the sync word. crc2 is the last 16-bit word of the frame, immediatelty preceeding the sync word of the next frame. crc1 applies to the first 5/8 of the frame, not including the sync word (see Table 11.1 for definition of 5/8 point for each frame size). crc2 provides coverage for the last 3/8 of the frame, or for the entire frame, not including the sync word. Decoding of either CRC word is optional, although highly recommended.

The following generator polynomial is used to generate the two 16 bit CRC words for an AC-3 frame:

$$x^{16}+x^{15}+x^{2}+1$$

The CRC may be implemented by one of several standard techniques. A convenient hardware implementation is a linear feedback shift register (LFSR). An example of an LFSR circuit for the above generator polynomial is the following:



Checking for valid CRC with the above circuit consists of shifting the AC-3 data bits serially into the circuit in the order in which they appear in the data stream. The sync word is not covered by either CRC word so it should not be included in the CRC calculation. crc1 is considered valid if the above register contains all zeros after the 16 bit data word defined as the 5/8 point in the frame is reached. crc2 is considered valid if the register contains zero after the entire data block, including both CRC words, has been shifted through the circuit. Some decoders may choose to only check crc2 and not check for a valid crc1 at the 5/8 point in the frame.

Note that crc1 is determined by encoders so that a valid CRC will appear at the appropriate 5/8 word in the frame. It is *not* the value generated by calculating the CRC of the first 5/8 of the frame using the above generator polynomial. Therefore decoders should not attempt to save crc1, calculate the CRC for the first 5/8 of the frame, and then compare the two.

Block size restrictions within each frame guarantee that blocks 0 and 1 are completely covered by crc1. Therefore, decoders may immediately begin processing block 0 when the 5/8 point in the data frame is reached. This may allow a reduced input buffers in some applications. Decoders that are able to store an entire frame may choose to process only crc2. These decoders would not begin processing block 0 of a frame until the entire frame is received.

Decoder strategy when invalid CRC is detected is user definable. Possible responses include muting, block repeats, or frame repeats.

Table 11.1: 5/8 Frame Location

frmsizecod	nominal bitrate	fs = 32 kHz 5/8 location	fs = 44.1 kHz 5/8 location	fs = 48 kHz 5/8 location
000000 (0)	32 kb/s	60	42	40
000001 (0)	32 kb/s	60	43	40
000010 (1)	40 kb/s	75	53	50
000011 (1)	40 kb/s	75	55	50
000100 (2)	48 kb/s	90	65	60
000101 (2)	48 kb/s	90	65	60
000110 (3)	56 kb/s	105	76	70
000111 (3)	56 kb/s	105	76	70
001000 (4)	64 kb/s	120	86	80
001001 (4)	64 kb/s	120	87	80
001010 (5)	80 kb/s	150	108	100
001011 (5)	80 kb/s	150	108	100
001100 (6)	96 kb/s	180	130	120
001101 (6)	96 kb/s	180	130	120
001110 (7)	112 kb/s	210	152	140
001111 (7)	112 kb/s	210	152	140
010000 (8)	128 kb/s	240	173	160
010001 (8)	128 kb/s	240	173	160
010010 (9)	160 kb/s	300	217	200
010011 (9)	160 kb/s	300	218	200
010100 (10)	192 kb/s	360	260	240
010101 (10)	192 kb/s	360	261	240
010110 (11)	224 kb/s	420	303	280
010111 (11)	224 kb/s	420	305	280
011000 (12)	256 kb/s	480	347	320
011001 (12)	256 kb/s	480	348	320
011010 (13)	320 kb/s	600	435	400
011011 (13)	320 kb/s	600	435	400
011100 (14)	384 kb/s	720	521	480
011101 (14)	384 kb/s	720	522	480
011110 (15)	448 kb/s	840	608	560
011111 (15)	448 kb/s	840	610	560
100000 (16)	512 kb/s	960	696	640
100001 (16)	512 kb/s	960	696	640
100010 (17)	576 kb/s	1080	782	720
100011 (17)	576 kb/s	1080	783	720
100100 (18)	640 kb/s	1200	870	800
100101 (18)	640 kb/s	1200	871	800

12. BIT STREAM CONSTRAINTS

The following constraints are placed upon the encoded bit stream by the AC-3 encoder. These constraints allow AC-3 decoders to be manufactured with slightly smaller memory buffers.

Block 0 will never exceed 1/2 of the frame.

Block 0 and block 1 will never exceed 5/8 of the frame.

The sum of block 5 mantissa data and auxiliary data will never exceed 3/8 of the frame.

Block 0 always contains all necessary information to begin correctly decoding the bit stream.