USING THE DYSON EXPANDERv1.4 (acts as a DolbyA DECODER)

Whenever the term 'expander' is used, it is meant that this expander can do a good job of decoding DolbyA encoded material.

THIS RELEASE IS A MAJOR QUALITY IMPROVEMENT OVER ANY PREVIOUS VERSION. The primary benefit is the fully implemented anti-hash.

NEW FEATURES from 1.3 to 1.4:

Added extreme anti-intermod to the 80-3k channel. Special filtering without losing any response between 80-3k. This extreme anti-intermod code is disjoint from the anti-hash code, but utilizes the anti-hash code when it is enabled. The extreme anti-intermod code is enabled at –ai=high and –ai=max.

Added stronger anti-hash code and a switch to disable (--ah=off) The stronger anti-hash code is most noticeable on strong female voices or male voices with some sibilance. The anti-hash code is normally enabled except when the anti-intermod is totally disabled (when the –ai=none or –ai=off switches are used, then the –ah switch doesn't do anything – the anti-hash is fully disabled when anti-intermod is fully disabled.)

BUGFIXES from 1.3 to 1.4:

Opened up some frequency ranges which were somewhat de-emphasized. The 3-9kHz band was being rolled off at 13.5kHz, which was too low because of the wide filters being used for DolbyA emulation. So, the filters for anti-aliasing in the 3-9kHz band were extended fully to 20.5kHz. There is still significant (and of critical importance) filtering on the low side of the band (both 3-9kHz and 9kHz-20.5kHz.)

This is NOT meant to be a replacement for a true DolbyA, but for convenience when material on a computer appears to be DolbyA encoded. The name Dolby is probably owned by Dolby Labs, and absolutely no proprietary material resides in the expander code base.

This design results from an initial inference as to what is necessary to implement, then further/exhaustive listening tests to narrow down the exact parameters. Thank God that my hearing is mostly intact (except for about about 13-14kHz), and enabled me (along with some wonderful help) to make the tuning as accurate as it is.

People using this commercially should donate a little to help with the effort. There is going to be a DolbySR effort soon, and need some more resources to complete that task. The level of complexity of DolbySR is much more complex than DolbyA, which is more complex than either DolbyC or DolbyB.

This program is currently command line only – a gui wrapper could be developed, and the processing code could be included into a plugin, but I haven't decided which platform where I want to do the plugin yet… This program is fairly CPU intensive, and goes far beyond a basic decoder WRT quality matters. After the usage info, there will be an explanation about what is going on in the decoder.

Program usage:

There are current two versions of the windows program: da-win and da-avx, both are 64 bit X86 only. The da-win version will work on most SSE2/3 cpus and da-avx will need AVX2. The da-win version has been tested on a Silvermont ATOM and the da-avx should work well on Haswell or better (that is a 4000 series CPU. Will probably work on a 3000 series.) The program is not multithreaded, so any more than one core gives only minor performance help.

Command line (all examples will be for the da-win version, but the da-avx version can be used on appropriate computers);

da-win –thresh=-15.50 <inputfile.wav >outputfile.wav

Note the simple stdin/stdout command line, but pipes can be used just as easily, and a few examples will be given below.

The –thresh=-15.50 is the decoder threshold (a few dB lower than the expander reference tone level), and for commercial recordings is usually in the range of -14.75 to -16.25, while -15.25 to -15.75 are most common. If the 'thresh' is wrong, then the sound will not be right (edgy super highs if too low, weird jerks in the level if too high, some surging if too low.)

On a Haswell running at 3.4GHz, it will run approx 3X real-time on a single core. There are modes where it can run faster, but a cost of quality.

File type: .wav only
Sample rates: 44.1kHz through 192kHz (decoder will attempt to work outside the range)
Data types: 16bit signed int, 24bit signed int(poorly tested), and 32bit floating

Here are the most useful 'switches':

```
--thresh=xxx          The threshold in dB.
--ingain=xxx          The input gain in dB.
--outgain=xxx         The output gain in dB.
--ai=none             Disable all anti-intermod. Equivalent to perfect normal
                      digital decoder without extraordinary techniques used in
                      this decoder.  Lots of intermod, much of the 'HF' is
                      actually intermod.
```

```
--ai=off              Disable all but the most simple anti-intermod.
--ai=med              (Default) The safest full anti-intermod, traditional
                      design on MF band.
--ai=high             Max anti-intermod/special MF band handling
--ai=max              Minor extra anti-intermod beyond –ai=high
--ah=on               Enable (default) anti-hash (only when anti-intermod is
                      functioning)
--ah=off              Disable anti-hash
--info                Provide running input/output levels
                           and per channel gain.
--floatout            Force 32bit floating point output,
                           even if integer input.
--lfoff               Disable LF (20-80Hz) expansion
--mfoff               Disable MF (80-3kHz) expansion
--hf0off              Disable HF0 (3k-9kHz) expansion
--hf1off              Disable HF1 (9k-20kHz) expansion
--wide                Change 21.5kHz limit to 0.46 sample rate
--hpf                 Make gain control less sensitive to LF
                      use of –hpf helps with intermod mitigation

The –lfoff,--mfoff,--hf0off,--hf1off switches each have a different meaning
available also:

--lfoff=xxx           Shift the LF threshold by xxx dB
--mfoff=xxx           Shift the MF threshold by xxx dB
--hf0off=xxx          Shift the HF0 threshold by xxx dB
--hf1off=xxx          Shift the HF1 threshold by xxx dB

A debugging/mode change switch is also available:

--raw                 Remove most of the anti-intermod filters.
--raw=xxx             Special changes to filtering scheme.
--raw=8192            Bare-bones – no special cleanup (crunchy sound.)
```

Typical usage with sox.

Sox is a very useful tool that makes using the decoder much easier. It provides input/output rate flexibility (decoupling it from the processing rate used by the program.) Also, it gives excellent file type flexibility.

Using the decoder with flac for example:

Almost any flac file can be used for input, and almost any flac type can be specified for output, so here is how you do it:

```
sox infile.flac –type=wav –encoding=floating-point –bits=32 – rate -v 96k | da-win –thresh=-15.50 –info |
      sox - outfile.flac
```

Notice that the sample rate is specified on the input sox command, and the data type is also specified. This is done to allow any acceptable flac file for input, while providing the program with the highest quality input/output & sample rate for its processing. (192k should work better, and it does in a way, but I compromised some of the filters so that the program performs better.) So, 96k is your best bet. 48K is faster and quite acceptable, but 44.1k will sometimes sound 'crunchy'.

I have sometimes found that recordings are either recorded with different thresholds for the MF frequencies, or the encoder was disabled entirely. I have some Carpenters and ABBA recordings which benefit from this modification:

```
sox infile.flac –type=wav –encoding=floating-point –bits=32 – rate -v 96k | da-win –thresh=-15.50 –mfoff –
       info | sox – outfile.flac
```

Note the use of the 'mfoff' switch. This is needed only once in a while, for example 2-3 of my 8 Carpenter's digital albums need the –mfoff switch, and some of my ABBA recordings (better than any normal CD – a 'Spanish' import that wasn't finalized) need the –mfoff switch. I suspect that the MF compressor wasn't fully disabled, but the levels were changed because of threshold sensitivity, and so the decoder also supports an option to –mfoff where you can change the threshold. I have found that on some recordings, --mfoff=-6 works great.

Usage notes:

I use the –info switch almost all of the time because the decoding does take quite a while (even if faster than real time), and it is nice to know that things are going well.

The anti-intermod code helps to remove the 'hash' and fake sibilance from the audio. The higher settings can cause some dips (order of 1dB) in the frequency response, but are very aggressive in filtering out errant hash. All gain control systems produce that hash, and it is bad that digital designs can be much worse than analog designs, but it is possible to filter out more of the hash in the digital domain – so in all, the digital approach can be much better. The –ai=med is max performance for now with no freq response impact and better than any practical analog design can be. For demo purposes, all special processing (including anti-intermod) can be disabled by –ai=none –raw=8192

Technology:

The expander is 'feed-forward', but implemented so that the effects of unfolding the feedback compressor are accurate. This expander is actually implemented as two separate devices running in parallel – one for the left channel and one for the right. If ever needed (unlikely) it would be easy to expand into a 5.1 expander if needed – but I believe that 5.1 came in after DolbyAs encoding days were long gone. (There is still a LOT of DolbyA material out there.)

The attack/decay curves are designed to be pretty close to what DolbyA decoding needs, and sometimes works poorly when attempting to decode non-DolbyA decoded signals. Specifically, the

decay curves are fixed/RC time constant (with a little frequency domain filtering to match the filtering for attack.)

The ‘attack’ curves are where all of the ‘meat’ is. Everything above and beyond the normal odd necessary attack curves is done to mitigate intermodulation effects – especially the first order effects between the gain control and the signal frequencies. The scenario is as follows: a two step attack curve, while the first and second phase time constants are not normal R/C but have some nonlinearity built in. This helps to maintain a continuous flow of different necessary attack schemes depending on a fast/high amplitude signal or a slow moving signal. Basically, there is NO NEED for a fast attack if the signal levels are moving slowly – so asymptotically, for slow input level changes, the attack and decay times are the same. However, for fast changing signal levels, the attack scheme has two phases (both are dynamic.) The scheme is meant to maintain an effectively fast attack time, while not wobbling about doing lots of evil intermodulation things… This is especially helpful in the digital world, where the nonlinear operations also inter-modulate with the sampling/distortion products wrapping around the Nyquist rate.

So – for the attack, there is some frequency domain filtering of both the pre-detection signal level, and less drastic filtering of the after-nonlinear-transfer-function. Since after the partial/weak second order curve, there is still some nonlinearity splattering around the signal (up past Nyquist), then there is some pre-filtering meant to keep most of the stronger sidebands below the Nyquist rate. It is important to be very careful doing two stages of filtering like this, because it can mess up the dynamics, but there are some rules – if followed, the dynamics will be minimally affected. I made such choices so that the gain control is fast (fast enough), but not noticeably slow in any way.

Additionally, due to the clipping in a DolbyA encoder compressor, there is some clipping in the encoded signal. There is a very minimal amount of compensation for the natural clipping in the true DolbyA also.

The general set of time constants is quite dynamic because of the unfolding from feedback to feed-forward, so the extra time domain ‘boost’ caused by the feedback scheme doesn’t normally happen in a feed-forward scheme. This is all compensated for quite accurately – so the dynamics in the decoding should be nearly perfect. Also, the time constants for the two HF channels are different than the LF and MF channels.

Another aspect of the decoder are the gain curves. Looking at the design from the outside – must ignore the actual design parameters – it seems like there might be the same gain curve for each channel, but that is not true. When looking at the output of a real DolbyA, no matter the internal reasons, there are 4 different curves. Also, due to the way that the two channels (HF0, HF1) work together, the HF1 channel has a totally different curve. The HF0 detection channel is 3-20kHz, while the HF1 detection channel is 9-20Khz. On the other hand, the HF0 gain control channel is 3-9kHz and the HF1 gain control channel is 9-20kHz, so that causes some interesting issues. Firstly, in the feedback connection, the HF0,HF1 compressors outputs mix together with the rest, while the inputs are overlapping. This means that the two outputs (and the two gains) need to be merged. This is NOT a simple problem, even though the results are simple. Suffice to say that the method that is used in the decoder is totally accurate.

Music (from my collection) that benefits from the expander/decoder. No time to look up all of the CDs and compare or the download provenance – so will try to describe when needed. The real vinyl albums will NOT need decoding because they were processed correctly. This only applies to the digital copies from CDs, online, Hdtracks, etc. This is only a portion of my collection, and I have tried to represent both encoded and non-encoded materials.

These settings were determined from a previous version of the decoder – the new values will probably be approximately 1.0dB higher. So, for the Carpenters 1969-1971, -14.65 will be better than the original -15.65.

ABBA Gold and ABBA Gold 2 Spanish CD
    Requires using the –mfoff switch, and the –thresh=-14.50. Tried different MF offsets, and the best appears to simply disable MF. On Carpenters below, it appears that there was a level offset – odd. Don't really know what is going on.
ABBA (vast collection of various Polar, Polydor,Atlantic, etc versions)
    decoding is useless due to further compression/processing/etc.
Carpenters 1969-1971
    --thresh=-15.65
Carpenters 1972,1973,1975-1977
    --thresh=-14.50 (Previous version of the decoder had troubles with these recordings, but now the problem is fixed.)
Herb Alpert – The Very Best
    --thresh=-15.6 (really hard to determine!!!)
Olivia, Olivia!
    --thresh=-15.50
Olivia, Let me be there (US)
    --thresh=-15.50
Olivia, If you love me
    --thresh=-15.7
Olivia, Let me be there (music makes my day)
    (probably not encoded)
Olivia, Long live love
    (probably not encoded)
Olivia, Clearly love
    --thresh=-15.75
Olivia, Have you never been mellow
    --thresh=-15.75
Suzanne Vega (with Luka, the mp3 song – the diner song at the beginning)
    --thresh=-15.75
Cars – (a greatest hits type collection) (1: just what I needed, 2: my best friends girl)
    --thresh=-15.75
Simon&Garfunkel – Parsely, Sage, Rosemary, Thyme
    --thresh=-15.65
Simon&Garfunkel – Bridge Over Troubled Water
    --thresh=-15.75
Simon&Garfunkel – Sounds of Silence
    --thresh=-15.65
Queen – Greatest Hits

--thresh=-15.50
Carly Simon – Greatest Hits
--thresh=-15.60

Chicago – Greatest Hits (cheap bargain CD)
--thresh=-15.50
Sergio Mendes – Brasil’66
--thresh=-15.75
Burt Bacharach – Greatest Hits (2 disk)
--thresh=15.55
Dionne Warwick – Greatest Hits (2 disk)
--thresh=15.65
Carpenters – from Hdtracks
--thresh=-15.7
Paul McCartney/Wings unlimited from Hdtracks
--thresh=-15.6

In all of the above – the music is enjoyable whether or not decoded, but there is additional (real technical) quality that can be extracted upon proper decoding!!!