Implementation of a High-Quality Dolby* Digital Decoder Using MMX™ Technology

James C. Abel, Intel Microcomputer Labs, Intel Corp.
Michael A. Julier, Intel Microcomputer Labs, Intel Corp.

Index Words: MMX™ technology, Dolby Digital Decoder, audio, multimedia.

Abstract

Dolby Digital is a high-quality audio compression format widely used in feature films and, more recently, on DVD[1]. PCs now offer DVD drives, and providing a Dolby Digital decoder in software allows decoding of Dolby Digital to become a baseline capability on the PC. Intel's MMX™ technology provides instructions that can significantly speed up the execution of the Dolby Digital decoder, freeing up the processor to perform other tasks such as video decoding and/or audio enhancement.*

A simple port of Dolby Digital to MMX technology using only a 16-bit data type introduces quantization noise that makes the decoder unsatisfactory for high-quality audio. However, MMX technology provides additional flexibility through 32-bit operations which, combined with other software techniques, allows the implementer to increase the audio quality of the decoder while still providing a significant speedup over implementations that do not use MMX technology. Intel has worked closely with Dolby Laboratories to define an implementation of Dolby Digital based on MMX technology that has achieved Dolby's certification of quality. This paper describes the research performed and the resultant techniques Intel used in creating its Dolby Digital decoder.

Introduction

Dolby* Digital is a transform-based audio coding algorithm designed to provide data-rate reduction for wide-band signals while maintaining the high quality of the original content [1]. MMX™ technology can be used to provide a processor-efficient implementation of Dolby Digital for a PC based on a Pentium® processor with MMX technology. It is important to maintain high audio quality, and Dolby Laboratories has developed a stringent test suite to ensure that a certified decoder indeed provides high quality. In addition, trained listeners evaluate prospective decoders using both test and program material. Only after a decoder has passed both the analytical and subjective tests is the decoder certified.

Intel's MMX instructions operate on 8, 16, and 32 bits. The human ear has an overall dynamic range of 120 dB and an instantaneous dynamic range of 85 dB [2]. The dynamic range of a binary value is 6.0206 dB per bit. Eight bits (48 dB of dynamic range, about that of AM radio) is obviously insufficient for high-quality audio. Sixteen bits (96 dB of dynamic range, as is used on Compact Disks) is usually considered high-quality audio, and we will accept this notion for this paper. However, due to rounding errors during the intermediate calculations, the accuracy at the output of a Dolby Digital decoder is significantly less than the accuracy of the intermediate values (assuming a uniform accuracy throughout the algorithm). This is typical in signal processing algorithms. Using 16 bits of accuracy uniformly through a Dolby Digital decoder is insufficient to pass the test suite. The challenge was to obtain both good execution speed and good audio quality. 32-bit floating-point numbers could be used throughout the data path and only use MMX technology for bit manipulation, but this would not be the most processor-efficient method. MMX technology provides integer operations that are more processor-efficient than existing floating-point operations; we strove to use the MMX instructions as much as possible.

The goal of this investigation was to find a minimal CPU implementation at an acceptable audio quality level. If the CPU requirements could be made small enough, Dolby Digital decoding entirely in software

---

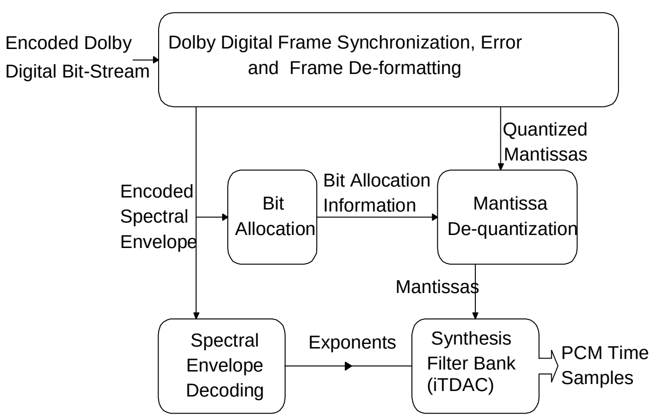
[1] DVD is often referred to as Digital Versatile Disk or Digital Video Disk.

would be feasible, even in combination with other operations (such as video playback). In order to do this, we had to determine the accuracy required in the various stages of the Dolby Digital decoder while maintaining effective use of MMX technology. We found that by using the flexibility of the 16-bit and 32-bit data types in the MMX instruction set, we were able to increase the accuracy of the Dolby Digital decoder significantly beyond that of a simple 16-bit approach with only a small impact on CPU performance. We also found that MMX technology can be used to speed up the bit manipulation, dithering, and downmix sections of the decoder.

An additional benefit of performing the audio decode in software is the resultant flexibility possible in the audio subsystem. If the Dolby Digital decoder is in software, it is easier to route the decoded audio to other audio subsystems. For example, simultaneous mixing of the PC's system sounds (i.e., via the Microsoft Windows Wave Device API) with the decoded audio is possible.

Dolby Digital Decoder

A block diagram of the Dolby Digital decoder is shown



in Figure 1 [3].

Figure 1. The Dolby Digital Decoder

The Dolby Digital bit stream contains Synchronization Information (SI), Bit Stream Information (BSI), Audio Block information (AB), Auxiliary (AUX) information, and Cyclic Redundancy Check (CRC). See Figure 2 for the Dolby Digital bit stream.

During our investigation, each block was inspected to determine if it could benefit from MMX technology. The following operations benefit significantly from MMX technology:

- Bit Stream Parsing
- Scaling
- TDAC Transform (DCT twiddles, FFT, Windowed-Overlapped-Add)
- Dithering
- Downmixing

We will now describe the five major operations from the input to the output (Bit Stream Parsing, Coefficient Extraction, TDAC Transform, Dithering, and Downmixing). We will also describe how MMX technology was used to provide a speedup. General precision and performance enhancements will also be discussed.

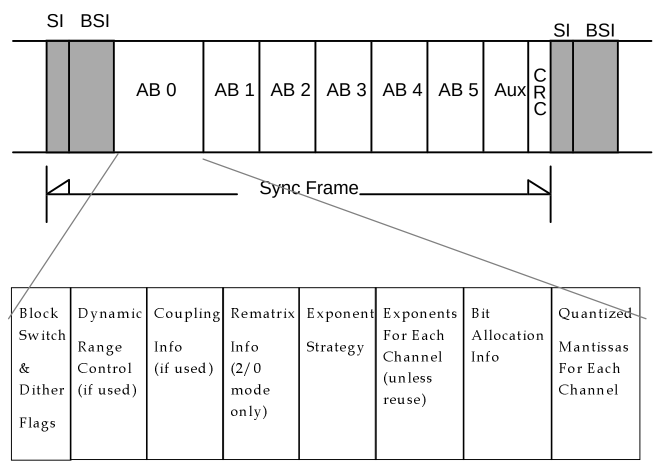


Figure 2. Dolby Digital Frame and Audio Block

Bit Stream Parsing

Each audio block (AB 0 through AB 5 in Figure 2) contains various pieces of information that tell the decoder how to decode the audio. These are bit fields that are extracted M bits at a time, where M is 0 to 16. MMX technology can be used to perform bit extraction [4], so we can efficiently parse the bit stream. From this information, we obtain the transform coefficients for the synthesis filter bank (TDAC transform).

Transform Coefficients Extraction

The audio block contains the information required to obtain the transform coefficients that will be sent to the synthesis filter bank. In Dolby Digital, the bit allocation, i.e., the number of bits used to represent a particular mantissa, is derived from the exponents (the spectral envelope). The mantissas are de-quantized and combined with the exponents in the denormalization process to create the transform coefficient values.

TDAC Transform

The Time Domain Aliasing Cancellation (TDAC) transform [5] converts the spectral information back to time domain, pulse-code modulated (PCM) samples. The TDAC provides perfect reconstruction (in the absence of quantization or other noise) and is critically sampled.

The TDAC transform is implemented as two DCT twiddle stages with an inverse Fast Fourier Transform (iFFT) in the middle [6]. A block diagram of this implementation of the TDAC transform is shown in Figure 3.

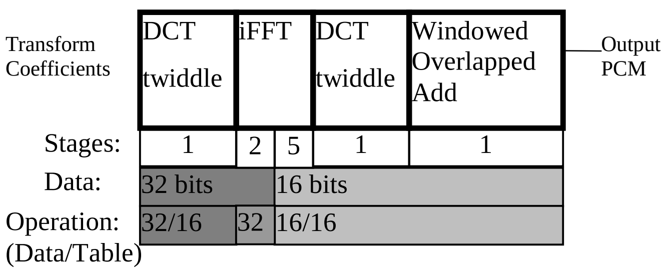


Figure 3. TDAC Transform Implementation

In our implementation, we created coefficient values with 24-bits of accuracy that are stored in 32-bit values. 24 bits of accuracy was chosen to prevent overflow during the intermediate denormalization and scaling processes. This 32-bit number was used in the first three stages of the TDAC transform. After the first two stages of the iFFT, the value was rounded to 16 bits of accuracy. The remainder of the operations were performed using pass-to-pass representations of 16 bits. MMX technology provides multiply accumulations to 32 bits, therefore many intermediate values were 32 bits.

The sine, cosine, and windowing values required in the TDAC transform were implemented via 16-bit lookup tables. Since these values are full-scale, 16 bits was sufficient for our needs. Errors introduced by imprecise coefficients are negligible compared to roundoff errors [7,8]. The technique of 32-bit data and 16-bit lookup tables has been shown to provide high-quality audio decoding [9].

Quantization errors introduced early in the transform process manifest themselves as tones in the output. Tonal noise is highly objectionable [10]. Output noise, if it must be present, should be broad-band or "white" noise. Therefore, the goal was to significantly reduce the peak spectral error. In a mixed-precision implementation, the question is how far into the TDAC transform do we need to carry 32 bits? In other words, where can we switch to 16 bits? Under subjective listening tests, we decided that performing the first three stages in 32 bits and the remainder in 16 bits reduced the tonal noise to a level of acceptability (see Figure 3). This also resulted in the decoder passing the measurement tests.

Multiplication in the MMX instruction set is 16 bits by 16 bits, yielding a 32-bit result. A 16-bit by 31-bit multiply is also possible in software, at a cost of at least five instructions and a pipelined output of two clocks per result [11]. Minimizing the number of 16-by-31 bit multiplies was important. It was discovered that the first two stages of the Decimation in Time (DIT) FFT contain only trivial coefficients, i.e., -1 and +1. This allowed these stages to be performed using only add and subtract instructions (no table lookup operations). These 32-bit operations are available in the MMX instruction set. This optimization allowed us to only use the more computationally intensive 16/31 bit operations only on the first DCT twiddle stage. The first two stages of the iFFT were performed with 32-bit adds and subtracts, which are efficient in the MMX instruction set.

The Windowed-Overlapped-Add (WOLA) block also fits well into the MMX instruction set. To perform the WOLA, the current and previous output arrays from the last DCT twiddle stage are windowed and then added together [5]. The windowing and addition operations were implemented as two 16-bit by 16-bit multiplies (the windowing) and then added as 32-bit quantities. This is provided by the PMADDWD instruction. The 32-bit results were then rounded to 16 bits for the output.

Mantissa Dithering

Dithering is required in a Dolby Digital decoder. How dithering is used by a decoder is determined by the Dolby Digital encoder used to create the frame. Dithering is used when the encoder determines that a transform mantissa doesn't get any bits (only an exponent) *and* that it is best to dither the mantissa (as opposed to having a mantissa of zero). This is implemented as a pseudo-random number generator that is random to 14 bits (the Dolby Digital specification states that the random number generator must be random to 8 bits or greater [3], so we exceed that specification). The calculation is given in Listing 1.

Listing 1. Dither Generation

C code:

```
x(t) = (x(t-1) * 47989) & 0xffff;
```

MMX Technology Assembly Code:

```
// dither multiplier value is linear
// congruential multiplier ^ 4,
// i.e. 0x4f31, packed 4 times

Quadword DithMultVal = 0x4f314f314f314f31;
```

```
// [63:48] = linear congruential multiplier ^ 4
// [47:32] = linear congruential multiplier ^ 3
// [31:16] = linear congruential multiplier ^ 2
// [15:0]  = linear congruential multiplier ^ 1
Quadword DithregInit = 0x4f31994d2379bb75;

Initialization:
        ;4 16-bit packed values
        MOVQ    MM0, DithregInit

Generation Loop:
        ; dither register * dither multiplier
        ; to get next set of values in dither
        ; register
        PMULLW  MM0, DithMultVal
        ;result is 4 16-bit values
        MOVQ    [result64], MM0
```

PMULLW has a latency of 3 but a throughput of 1. This program can be pipelined to achieve one result per clock written out to memory (for example, on a Pentium processor PMULLW in the V Pipe, MOVQ in the U Pipe).

Calculating four dither values with a single PMULLW instruction provides a high throughput for this part of the decoder. This instruction multiplies two 16-bit values and provides the lower 16 bits of the result (four of these are performed per instruction).

Downmixing

Dolby Digital can contain up to six audio channels: five full-bandwidth channels and a low-frequency effect (LFE) channel. This mode is often referred to as 5.1 channels, where 5 is the number of full-bandwidth channels and .1 is the LFE. The vast majority of PCs have only two audio output channels, so downmixing is often used. Also, for our two-channel downmix, the LFE is discarded. Downmixing is generally an additive process. Scaling (which is discussed below, see "Early Scaling") is also part of downmixing in Dolby Digital. It is used to set relative levels between downmixed channels. Since we perform it up front as part of the denormalization process, downmixing becomes additive. MMX technology provides SIMD addition, which speeds up downmixing.

Precision Enhancements

To increase the audio quality, some precision enhancements were made. Even though these techniques increased the processing requirement slightly, they added a significant quality improvement and were judged to be worth the additional overhead.

Rounding

Rounding is important to perform every time a higher-precision number is being converted to a lower-precision number (e.g., 32 to 16 bits). This is encountered often in multiplications in the MMX instruction set. For example, the PMADDWD instruction (packed multiply-accumulate) multiplies 16-bit numbers, yielding a 32-bit result. If this 32-bit result is to be converted to a 16-bit value, rounding should be used. Rounding can provide a significantly reduced error compared to truncation [7]. While the MMX instruction set does not provide a rounding mode, it is easy to accomplish in software. Listing 2 provides an example.

Listing 2. Rounding Using MMX Technology

```
// Round 2.30 number
// RoundVal is ½ LSB of 16-bit result

RoundVal = 0x0000400000004000;
pmaddwd mm6,mm5                 ;2.30 number
paddd   mm6,RoundVal            ;round
psrad   mm6,15                  ;2.30 to 1.15
```

Since the values are represented in two's complement, this technique works with both positive and negative numbers. In our Dolby Digital decoder, rounding was used extensively.

Gain Ranging

Dolby Digital provides Gain Ranging [3], which allows block scaling for low-level signals. This enhances the dynamic range of the decoder and was used in our implementation. Gain Ranging can contribute to noise modulation as the gain ranging levels are crossed. For our decoder, we decided the benefit of the additional dynamic range outweighed the potential of discontinuous noise modulation.

Additional performance enhancements were made that are general to the implementation of a Dolby Digital decoder on a PC. These are included here, even though they are not unique to optimizations that utilize MMX technology.

Additional Performance Enhancements

Frequency Domain Downmixing

Since the TDAC transform is a linear process, downmixing can be accomplished in the frequency domain. This reduces the number of transforms from the number of input channels from the Dolby Digital stream (2 to 5) to the number of output channels (2). However, the transform block sizes in Dolby Digital can change from 512 to 256 in the presence of transients [3]. It is not possible to downmix in the frequency domain for differing block sizes, so in this case an additional downmix stage is required after the TDAC transform to perform the remainder of the downmix in the time domain. The transform coefficients are contained in 32-bits. Using the 32-bit adds in the MMX instruction set provides an efficient downmix.

Early Scaling

There are several factors in the scale factor of a particular channel: Dynamic Range Control, Gain Ranging, and Downmix Scaling. We found it computationally beneficial to perform this operation during denormalization, essentially combining scaling and denormalization into one operation. This is performed by adjusting all of the exponents and mantissas by a particular amount. We stored the exponents as 8-bit quantities (the range is only 5 bits in Dolby Digital) and used MMX technology 8-bit add instructions (PADDB) to scale 8 exponents at a time. The unpack instruction (PUNPCKLBW) was used to efficiently replicate the 8-bit scale value eight times across the 64-bit register.

When the values are scaled up front, then downmixing becomes a simple addition as opposed to a multiplication by a constant. Since the transform coefficients are represented in 32 bits, downmixing in the frequency domain is performed by 32-bit adds using the packed add (PADDD) instruction. This avoids 32-bit multiplies.

Exponent and Bit Allocation Reuse

A Dolby Digital stream only has exponents in an audio block when the encoder determines that they have changed enough to be resent. This is called exponent reuse. Therefore, if exponent reuse is in effect, it is more processor-efficient to save the exponents in an array and use the values from the array (as opposed to re-extracting the bits from the bit stream).

The bit allocation information is derived from the exponents. Therefore if exponent reuse is in effect, bit allocation may be also (depending on new bit allocation information, SNR offset information, delta bit allocation information and coupling information - see [3] for details). Since recalculating the bit allocation information is computationally expensive, the bit allocation information should be saved in an array and reused if possible. This does not benefit from MMX technology per se, but shows the advantages of decoding on a system that has a relatively large amount of cache memory as opposed to a DSP that may have to recalculate these values since it does not have sufficient memory for all of these arrays.

Results

Compared to an optimized version that does not use MMX technology, the processor speedup is about 1.5X for a two-channel, surround-compatible (also known as LtRt) downmix from 5.1 channel source material. For 5.1 channels of output, the speedup increases to about 1.8X. Typically, two TDAC transforms are performed for a two-channel downmix, and six are required for a full 5.1 channels of output. The greater speedup is due to the fact that the TDAC transform benefits greatly from MMX technology and the increased number of TDAC transforms performed for 5.1 channels (versus a two-channel downmix). One caveat is that six channels of audio output is not common on today's mainstream PC. However, sound cards with four channels of discrete audio output are on the market today, so six channels may become available in the future via analog outputs or the 1394 high-speed serial bus.

Intel's Dolby Digital decoder provides significantly better audio quality than a 16-bit only approach, while offering an efficient implementation. The included audio clips contrast the 16-bit only approach with the enhanced approach. Note that these are very low-level signals (you may have to increase the volume to hear them).

noisfl16.wav	Low-level noise decoded by a simple 16-bit implementation. Notice the tonal artifacts. (Sound file is only availiable in online HTML version.)
noisfl_m.wav	Low-level noise decoded by Intel's mixed 16/32-bit implementation. The noise is lower and broad-band (white). (Sound file is only available in online HTML version.)

Intel's Dolby Digital decoder compares favorably with floating-point based implementations. Typically Intel's decoder has about 5 to 10 dB of additional noise as compared to a floating-point implementation. The improvement over a simple 16-bit truncation model is approximately 5 to 15 dB, depending on the program material. The most striking improvement is the reduction in peak spectral error, or the "tonality."

Figures 4 through 10 show how Intel's decoder compares to the 16-bit truncation model and floating-point reference.

Figures 4 through 7 show a spectral plot of a 200 Hz sine wave at -60 dB. Figure 4 is a composite of Figures 5 through 7. These are separated out since, in the composite, it is difficult to distinguish between the three plots. This illustrates the peak spectral error (graphical peaks) in the 300 to 20 kHz region. These peaks show the presence of tonal noise. The 16-bit truncation decoder has by far the worst peaks, as high as -105 dB. The MMX technology decoder reduces these peaks by 13 dB to -118 dB.

Figure 8 shows the Total Harmonic Distortion (THD) vs. Frequency. The THD vs. Frequency is improved by about 10 dB over the 16-bit truncation decoder.

Figure 9 is the noise modulation plot. This is a plot of the output noise in a third octave band at 4 kHz as a function of the input level of a 41 Hz sinusoid decremented from 0 dBFS to -120 dBFS. The improved (lowered) noise level is between 15 dB for high-level signals and 5 dB for low-level signals.

Figure 10 is a noise plot of a 4 kHz sine wave reduced in level 1 dB per second from 0 dBFS to -120 dBFS, with the sine wave removed via a notch filter. This shows that the noise for a full-level signal is still small (-78 dB), going to -88 dB for a medium- to low-level signal. This is approximately a 12 dB improvement for high-level signals and approximately a 6 dB improvement for low-level signals.

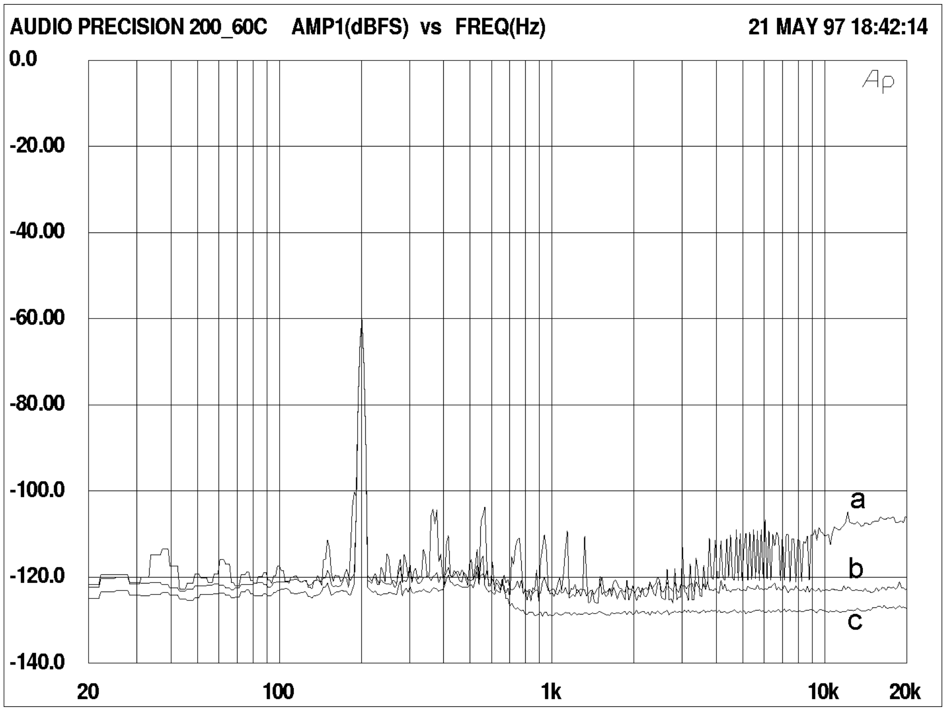


Figure 4. 200 Hz at -60 dB. a) 16-Bit Truncation, b) MMX Technology, c) Dolby Reference Decoder

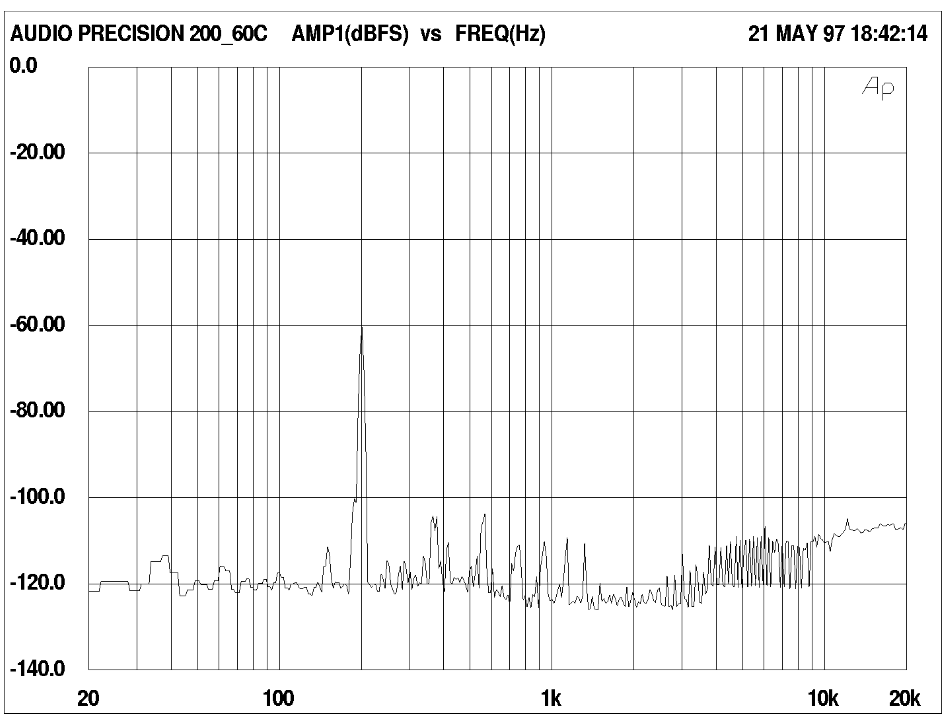


Figure 5. 200 Hz at -60 dB. 16-Bit Truncation Decoder

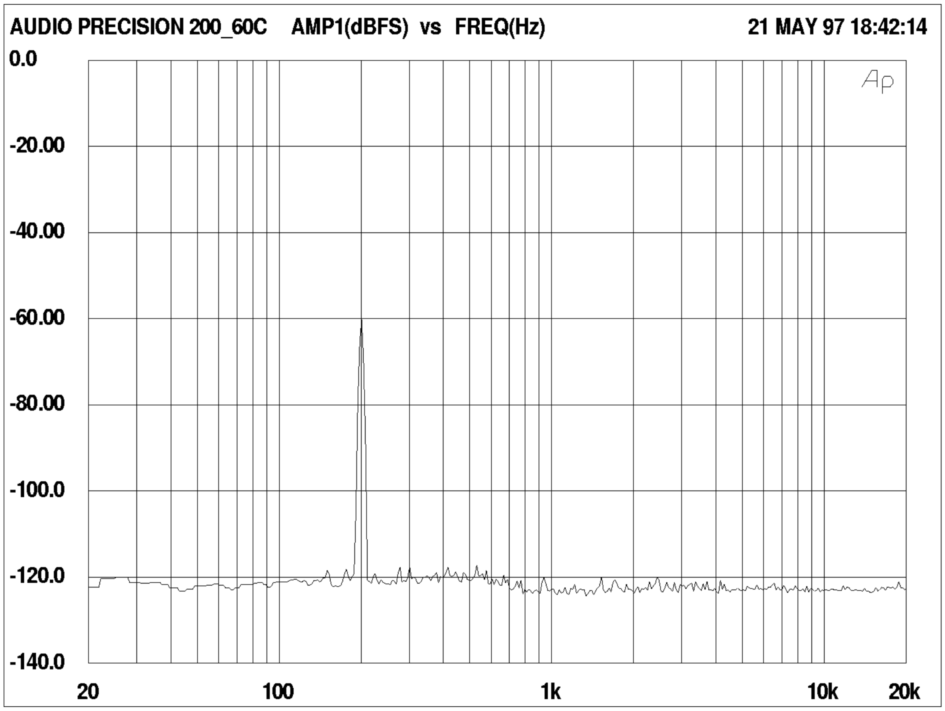


Figure 6. 200 Hz at -60 dB. MMX Technology Decoder

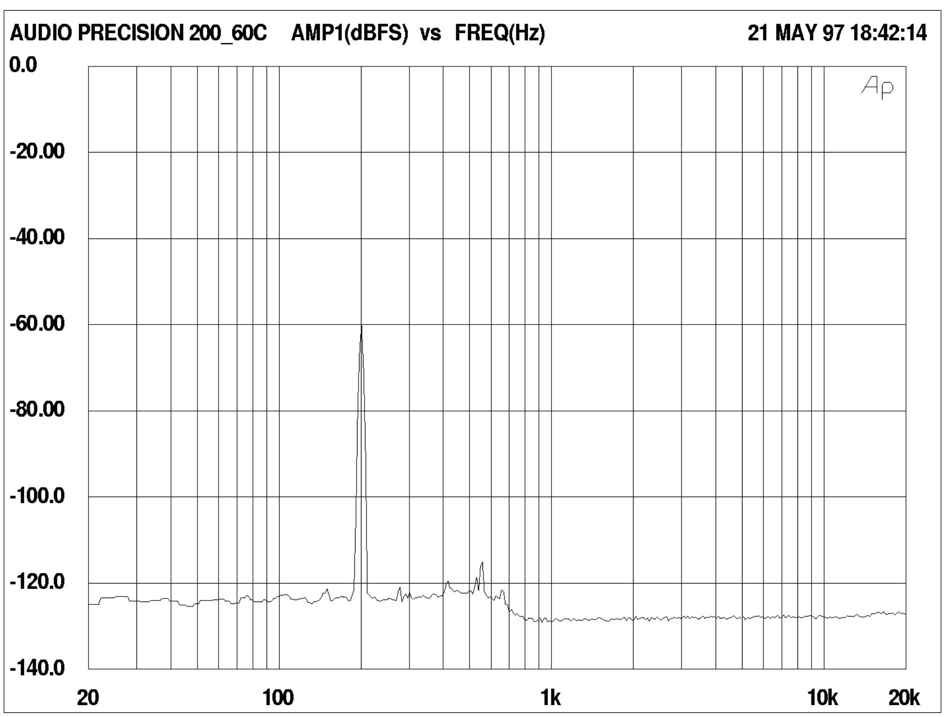


Figure 7. 200 Hz at -60 dB. Dolby Reference Decoder

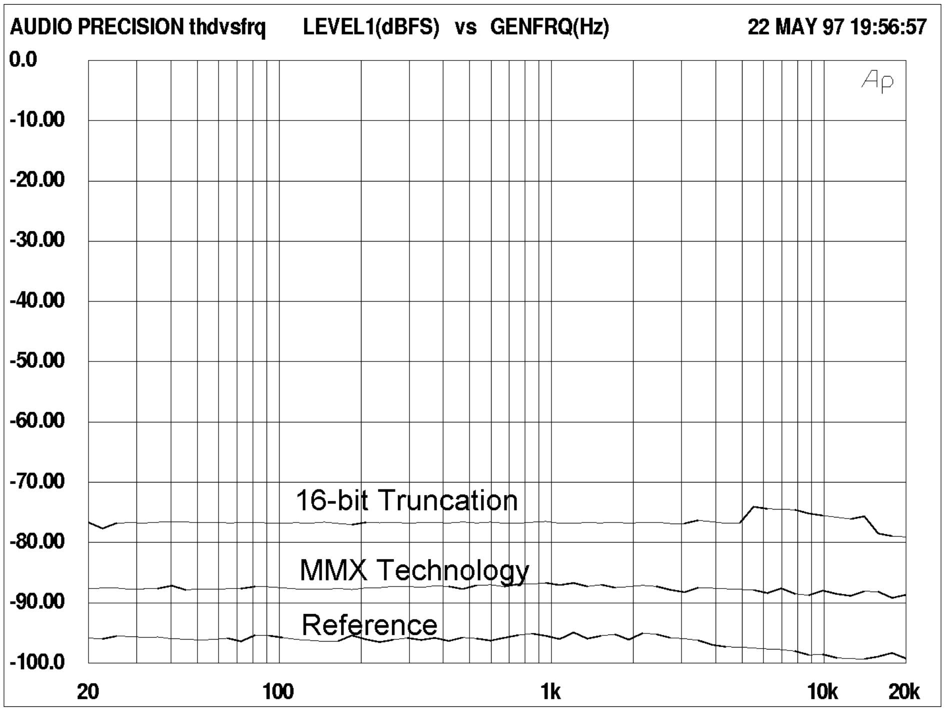


Figure 8. THD vs. Frequency

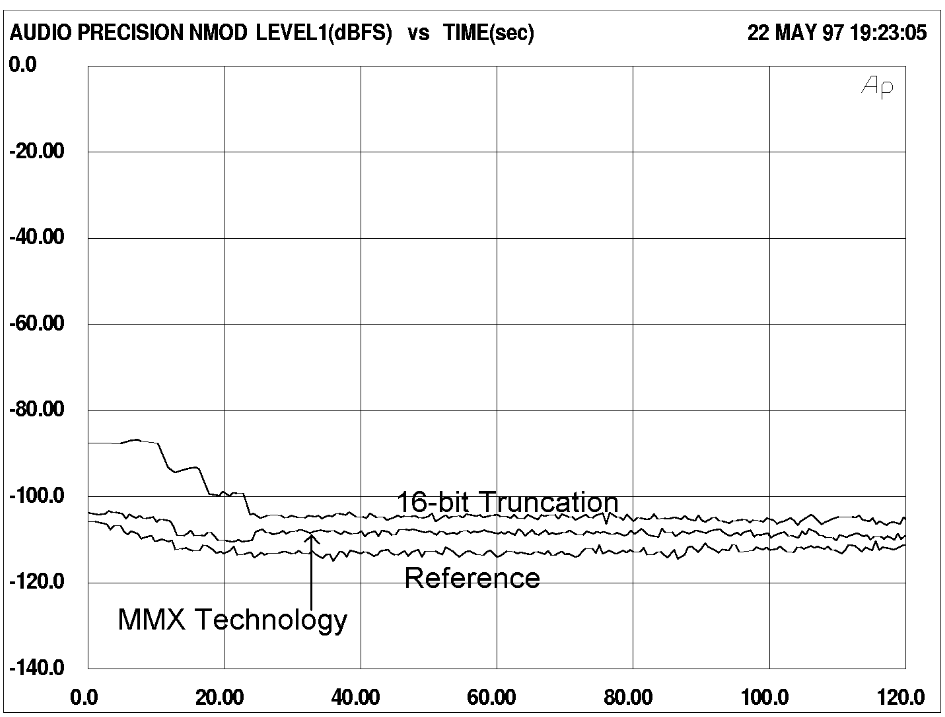


Figure 9. Noise Modulation at 4 kHz, 41 Hz Input

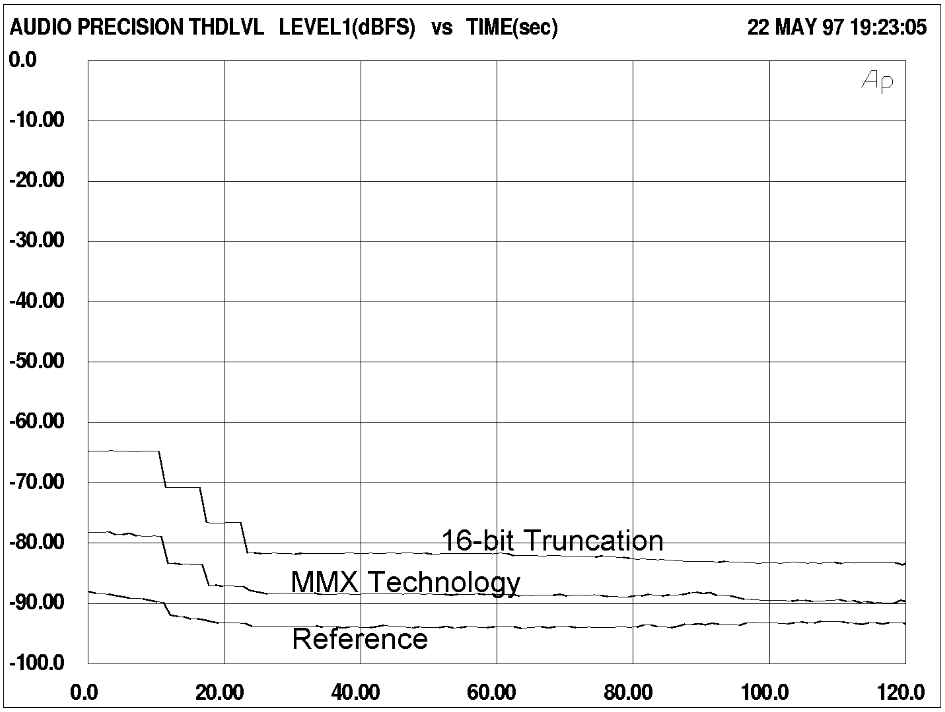


Figure 10. THD vs. Level, 4 kHz Input

Decoding a Dolby Digital stream consumes less than 8% of a Pentium® II processor running at 233 MHz. Figure 11 shows the processor requirements for several DVD audio tracks (5.1 channels, 384K bits/second, 48K samples/second, downmixed to LtRt, except for TWISTER which is two channels, 192K bits/second). Clearly, this is small enough to make software Dolby decoding quite feasible in real-world applications. The remaining 92% of the processor can be used for other things, such as a software MPEG 2 video decoder for a software DVD player.

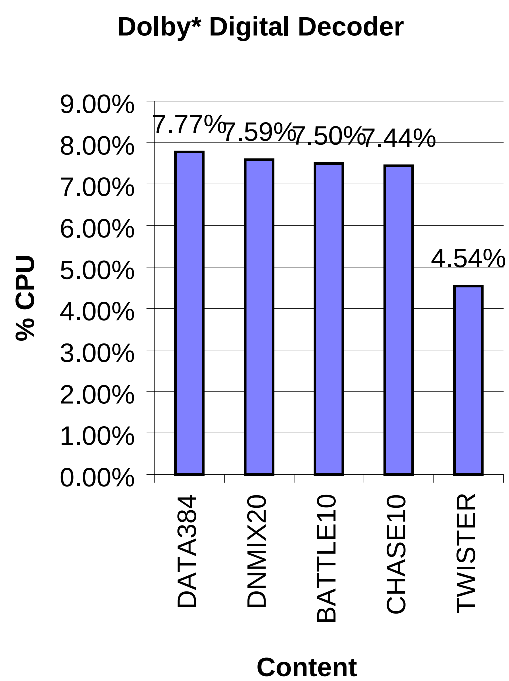


Figure 11. Processor Requirements. Note - DATA384 and DNMIX20 are test materials. BATTLE10 and CHASE10 are from the movie *Outbreak*. TWISTER is from the movie *Twister*.

Discussion

Making intelligent use of MMX technology requires a good understanding of the algorithm being coded. By understanding the strengths and flexibility of MMX technology, many clever techniques can be devised. While high-quality audio is a subjective term, we believe this decoder lives up to the name.

Table 1 shows the CPU breakdown for each part of the Dolby Digital decoder. After the data path has been sped up by MMX technology, the Bit Unpacking section becomes the next major consumer of the CPU. This is mainly due to the sequential nature of extracting variable-length bit fields from the bit stream.

Table 1. CPU Breakdown

Processing Block	% of Full Decoder
Bit Unpacking	28.3
TDAC/WOLA/Downmix	27.7
Scaling/Denormalization	27.2
Bit Allocation	10.2
Miscellaneous	6.6

Based on measurements (see Figure 10), the Intel decoder has a Signal-to-Noise Ratio (SNR) for a full-scale signal of about 78 dB. This compares reasonably well to the instantaneous sensitivity of the ear of about 85 dB [2]. The Dynamic Range (maximum output level vs. noise floor for a low-level signal) is about 88 dB. This compares reasonably well to a consumer CD player, which is typically at about 95 dB.

Conclusion

Intel's Dolby Digital decoder provides a processor-efficient implementation that meets a high-quality standard. By offering this decoder as a baseline capability on PCs with MMX technology, decoding and playback of compressed audio is possible with no additional hardware cost. The low processor usage allows additional features such as software video decoding and audio enhancement to occur concurrently.

Acknowledgment

The writers thank Eric Benjamin, Charles Seagrave, Uttam Suri, and Steve Vernon of Dolby Laboratories for their suggestions on quality improvement and the help they provided in the analysis of the Intel decoder.

Authors

James Abel is currently a Software Development Engineering Manager in the Desktop Performance Lab at Intel Corporation in Chandler, Arizona. James obtained a Bachelor's Degree in Engineering from Bradley University in Peoria, Illinois in 1983 and a Master's Degree in Computer Science from Arizona State University in 1991. His interests include signal processing, computer architectures, software tools, and audio algorithms. James' email address is jabel@inside.intel.com.

Mike Julier is currently a Sr. Software Development Engineer in the Desktop Performance Lab at Intel Corporation in Chandler, Arizona. He holds a BS in Computer Engineering from the University of Michigan, Ann Arbor. Mike's technical interests include performance optimizations of code from C++ to assembly, 3D graphics, and ISAs. Mike's email address is Michael_A_Julier@ccm.sc.intel.com.

References

[1] Fielder, L., Bosi, M., Davidson, G., Davis, M., Todd, C., and Vernon, S., "AC-2 and AC-3: Low-Complexity Transform-Based Audio Coding," Collected Papers on Digital Audio Bit-Rate Reduction, Audio Engineering Society, New York, New York, pp. 54-72.

[2] Harris, S., "Understanding, enhancing, and measuring PC-audio quality," EDN, Vol. 42, Number 8, April 10, 1997, p. 173.

[3] Advanced Television Systems Committee, "Digital Audio Compression Standard (AC-3)," Revision A/52, 20 December 1995.

[4] "Using MMX™ Instructions to Get Bits From a Data Stream," http://developer.intel.com/drg/mmx/appnotes.

[5] Princen, J. and Bradley, A., "Analysis/Synthesis Filter Back Design Based on Time Domain Aliasing Cancellation," IEEE Transactions on ASSP, Vol. 34, pp. 1153-1161, October 1986.

[6] Sevic, D., Popovic, M., "A New Efficient Implementation of the Oddly Stacked Princen-Bradley Filter Bank," IEEE Signal Processing Letters, Vol. 1, No. 11, pp. 166-168, November 1994.

[7] Thong, T., "Fixed-Point Fast Fourier Transform Error Analysis," IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. ASSP-24, No. 6, December 1976.

[8] Weinstein, C., "Quantization Effects in Digital Filters," MIT Lincoln Laboratories Technical Report 468, ASTIA Doc. DDC AD-706862, November 21, 1969.

[9] Davidson, G., Anderson, W., Lovrich, A., "A Low-Cost Adaptive Transform Decoder Implementation for High-Quality Audio," IEEE International Conference on

Acoustics, Speech, and Signal Processing, March 23-26, 1992.

[10] Dolby Laboratories, "Test Procedure for AC-3 Decoders Using Test Files," Dolby Document S96/11283.

[11] "Using MMX™ Instructions to Perform 16-Bit x 31-Bit Multiplication," http://developer.intel.com/drg/mmx/appnotes.
