

MDEC Data Compression Overview



What Does The MDEC Do?

- ▶ The PlayStation MDEC is used to quickly decode graphics images compressed using the MDEC format.
 - Still images.
 - Individual frames of video sequences.

MDEC Graphics Compression Format

- ▶ MDEC is a “lossy” format, very similar to JPEG.
 - Not quite identical to JPEG.
 - Called “lossy” because the picture is altered somewhat throughout the compression process.
 - Encoding method is designed to maximize compression ratios while still maintaining image quality.

MDEC Encoding Format

- ▶ Macroblock Division
- ▶ Color Space Conversion
- ▶ JPEG Downsampling Step
- ▶ Discrete Cosine Transform
- ▶ Quantization
- ▶ Run Length Encoding
- ▶ Huffman (VLC) Encoding

MDEC Encoding Format

▶ Macroblock Division.

- Picture is divided into 16x16 “macroblocks”.
 - JPEG uses 8x8 blocks
- Incomplete macroblocks not allowed.
 - Image size must be multiple of 16 pixels tall and 16 pixels wide.
 - Image size can be padded during encoding process and excess discarded after decoding.
 - Movie Converter pads image size if necessary.

MDEC Encoding Format

- ▶ Color Space Conversion.
 - Convert pixels from RGB to YCbCr color space.
 - YCbCr has a “brightness” value and two color vector values.
 - Allows brightness and color to be quantized separately.
 - Human eye perceives brightness better than color, so color data can be quantized more without visible loss of quality.

MDEC Encoding Format

- ▶ JPEG Chroma Downsampling Step.
 - Not done for MDEC encoding.
 - Averages color values for groups of pixels together.
 - This is part of the “lossy” stage for JPEG.

MDEC Encoding Format

- ▶ Discrete Cosine Transform.
 - Translates pixel data into a frequency map that describes how the pixel values change from one pixel to the next.
 - Not “Lossy” except for round-off error.

MDEC Encoding Format

- ▶ 8x8 macroblock source, brightness channel only
- ▶ 8x8 macroblock after Discrete Cosine Transform

| | | | | | | | |
|------|-----|-----|-----|-----|-----|-----|-----|
| 128 | 128 | 128 | 128 | 128 | 128 | 128 | 128 |
| 128 | 128 | 128 | 128 | 128 | 68 | 51 | 39 |
| 128 | 128 | 128 | 83 | 59 | 42 | 28 | 17 |
| 128 | 128 | 83 | 56 | 38 | 23 | 11 | 0 |
| 128 | 128 | 59 | 38 | 22 | 8 | 5 | 14 |
| -128 | -68 | -42 | -23 | -8 | 6 | 18 | 28 |
| -128 | -51 | -28 | -11 | 5 | 18 | 30 | 40 |
| -128 | -39 | -17 | 0 | 14 | 28 | 40 | 51 |

| | | | | | | | |
|------|------|-----|-----|-----|-----|----|----|
| -229 | -155 | -23 | -21 | -10 | -4 | -1 | -4 |
| -155 | 42 | 34 | 23 | 12 | 12 | 7 | 0 |
| 23 | 34 | 7 | -6 | -13 | -10 | -7 | -5 |
| -21 | 23 | -6 | -5 | -2 | 1 | 3 | 3 |
| -10 | 12 | -13 | -2 | 5 | 4 | 5 | 6 |
| -4 | 12 | -10 | 1 | 4 | -4 | -4 | 1 |
| -1 | 7 | -7 | 3 | 5 | -4 | -6 | -1 |
| -4 | 0 | -5 | 3 | 6 | 1 | -1 | 2 |

MDEC Encoding Format

▶ Quantization

- Reduces the number of discrete levels of intensity.
 - Eliminates small changes in brightness and/or color.
- Color and brightness are quantized separately.
 - Eye is less sensitive to changes in color, so greater quantization can be done without visible loss of quality.
- Non-Linear
 - High-frequency values quantized more heavily than low-frequency values.

MDEC Encoding Format

▶ Quantization (continued)

- “Lossy” stage.
 - Only “lossy” stage for MDEC.
 - 2nd “lossy” stage for JPEG.
 - Amount of quantization (and therefore the amount of loss) is specified by the “quality” setting of the compressor program.

MDEC Encoding Format

- ▶ 8x8 macroblock after DCT, brightness channel only

| | | | | | | | |
|------|------|-----|-----|-----|-----|----|----|
| -229 | -155 | -23 | -21 | -10 | -4 | -1 | -4 |
| -155 | 42 | 34 | 23 | 12 | 12 | 7 | 0 |
| 23 | 34 | 7 | -6 | -13 | -10 | -7 | -5 |
| -21 | 23 | -6 | -5 | -2 | 1 | 3 | 3 |
| -10 | 12 | -13 | -2 | 5 | 4 | 5 | 6 |
| -4 | 12 | -10 | 1 | 4 | -4 | -4 | 1 |
| -1 | 7 | -7 | 3 | 5 | -4 | -6 | -1 |
| -4 | 0 | -5 | 3 | 6 | 1 | -1 | 2 |

- ▶ 8x8 macroblock after Quantization

| | | | | | | | |
|------|-----|-----|----|---|----|----|----|
| -229 | -19 | -19 | -2 | 5 | -2 | -2 | 3 |
| 3 | -2 | -1 | 2 | 1 | 2 | -1 | 0 |
| 1 | 0 | 0 | 1 | 0 | 0 | 1 | -1 |
| 0 | -1 | 1 | 0 | 0 | 0 | -1 | 0 |
| 0 | -1 | 1 | 0 | 0 | -1 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

MDEC Encoding Format

▶ Run Length Encoding (RLE)

- After quantization, the DCT encoded data has many repeat values and many zeros, making it ideal for RLE compression.
- Non-lossy compression.
 - This is the main data-compression stage for both MDEC and JPEG. Prior to this stage, we have the same amount of data as we started with.
 - The previous stages are designed to convert the image data to something that will RLE compress more efficiently.

MDEC Encoding Format

- ▶ The 8x8 macroblock after quantization has many repeating zero values and is ideal for RLE compression.

| | | | | | | | |
|------|-----|-----|----|---|----|----|----|
| -229 | -19 | -19 | -2 | 5 | -2 | -2 | 3 |
| 3 | -2 | -1 | 2 | 1 | 2 | -1 | 0 |
| 1 | 0 | 0 | 1 | 0 | 0 | 1 | -1 |
| 0 | -1 | 1 | 0 | 0 | 0 | -1 | 0 |
| 0 | -1 | 1 | 0 | 0 | -1 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

MDEC Encoding Format

- ▶ Huffman (VLC) Encoding.
 - Non-lossy compression.
 - Achieves about 50% additional compression
 - VLC = Variable Length Coding.
 - Converts most commonly used sequences into shorter codes of varying lengths.
 - Most common sequence is stored using 2 bits, other sequences stored using 3 bits, 4 bits, etc.
 - Uses look-up table to decode bit codes into original sequences.

MDEC Encoding Using Movie Converter

▶ Image Conversions.

- Converts AVI video to STR format.
- Converts TIM files to BS format.
 - BS format is single-frame MDEC image.
 - “BS” stands for “bitstream”, referring to Huffman encoding.

▶ Audio Conversions.

- Converts AVI audio to XA format.
- Converts WAV audio to XA format.

MDEC Encoding Using Movie Converter

- ▶ Movie Converter converts AVI files to STR format.
 - AVI files must be uncompressed.
 - Specify fixed single-speed or double-speed data rate.
 - AVI files can be video-only, video+audio, or audio only.

MDEC Encoding Using Movie Converter

- ▶ Movie Converter converts sequence of still images into STR movie format.
 - Last 4 chars of filename specify position in sequence.

| Format | TIM files (PlayStation bitmap) | RGB files (raw RGB pixel data) | YUV files (raw YUV pixel data) |
|---------------------|---|---|---|
| File 1 | fr0001.tim | fr0001.rgb | fr0001.yuv |
| File 2 | fr0002.tim | fr0002.rgb | fr0002.yuv |
| File 3 | fr0003.tim | fr0003.rgb | fr0003.yuv |
| (more files) | ... | ... | ... |

Combining STR Files and XA Files with Movie Pack

- ▶ MovPack combines STR movie files and XA audio files to create new STR files with interleaved audio.

MDEC Decoding

- ▶ Reset the MDEC.
- ▶ Determine location of compressed data.
- ▶ Get size of RLE data with ***DecDCTBufSize***.
- ▶ Decode Huffman-encoded data
- ▶ Send RLE data to MDEC.
- ▶ Get back decoded macroblocks from MDEC.
- ▶ Use LoadImage to move image data to VRAM.

MDEC Decoding

- ▶ Reset the MDEC.
 - ***DecDCTReset(0)***
 - Cancels any ongoing decoding process.
 - Clears any DMA operation into or out of MDEC buffers.
 - Resets quantization tables to default.
 - ***DecDCTReset(1)***
 - Cancels any ongoing decoding process.
 - Clears any DMA operation into or out of MDEC buffers.
 - Keeps current quantization tables.

MDEC Decoding

- ▶ Determine location of compressed data.
 - Still image is location of BS file.
 - Location where loaded from CD or where embedded in data segment.
 - Movie frame is location within ring buffer.
 - Location returned by *StGetNext*.

MDEC Decoding

- ▶ Determine size of RLE data to be decoded with ***DecDCTBufSize***.
 - $buffersizeneeded = \mathbf{DecDCTBufSize}(framedata)$
 - *framedata* is raw compressed frame data.
 - Allocate buffer space specified by *buffersizeneeded*.

MDEC Decoding

- ▶ Decode Huffman-encoded data
 - ***DecDCTvlc(framedata, buffer);***
 - *framedata* is raw compressed frame data
 - *buffer* is pointer to buffer that will receive the RLE-encoded DCT data, must be at least the size returned by ***DecDCTBufSize*(framedata);**

MDEC Decoding

- ▶ Send RLE data to MDEC
 - ***DecDCTin***(*buffer*, *mode*);
 - *mode* parameter controls 16-bit mode or 24-bit mode. For 16-bit mode, also controls transparency (the STP bit of each pixel).
 - MDEC accepts a maximum of 128k at once.
 - Break larger images into segments.
 - 128k of RLE-encoded MDEC data should be a larger than screen-sized picture in the first place, even at maximum quality.

MDEC Decoding

- ▶ MDEC decodes RLE, performs inverse-DCT, and converts color space back to RGB.
 - You specify 16-bit or 24-bit output mode when sending data to MDEC with ***DecDCTin***.

MDEC Decoding

- ▶ Get back decoded macroblocks from MDEC.
 - **DecDCTout**(*mbimage*, *numlongs*);
 - *mbimage* is address in RAM where data will be placed.
 - *numlongs* is the number of long words to retrieve from the MDEC.
 - A 16-bit 16x16 macroblock is 512 longwords (2048 bytes).
 - A 24-bit 16x16 macroblock is 768 longwords (3096 bytes).

MDEC Decoding

- ▶ Wait for each macroblock to be transferred before doing the next one.
 - Use ***DecDCToutSync*** or ***DecDCToutCallback***.
 - ***DecDCToutSync*** either polls current status or waits for transfer from MDEC to complete.
 - ***DecDCToutCallback*** allows you to specify function to be executed when transfer from MDEC is completed.
 - Use callback and continue processing non-MDEC related code
 - Callback typically copies segment to image buffer in RAM or else calls LoadImage to move segment to VRAM

MDEC Decoding

- ▶ Use ***LoadImage*** to move image data to VRAM.
 - Typically done after ***DecDCToutSync(0)*** or in routine specified to ***DecDCToutCallback***.
 - Use ***DrawSync*** first to make sure previous operation is finished.

MDEC Decoding

- ▶ Image is decoded into vertical strips of 16-pixel width.
 - For a 96x64 image, the 16x16 pixel macroblocks would be decoded in the order shown below.
 - Make sure you specify the proper coordinates with LoadImage.

| | | | | | |
|---|---|----|----|----|----|
| 1 | 5 | 9 | 13 | 17 | 21 |
| 2 | 6 | 10 | 14 | 18 | 22 |
| 3 | 7 | 11 | 15 | 19 | 23 |
| 4 | 8 | 12 | 16 | 20 | 24 |

The End