1999

Design and implementation of a DSP based MPEG-1 audio encoder

Eric Hoekstra
Muhammad Shaaban
Roy Czernikowski

Follow this and additional works at: http://scholarworks.rit.edu/article

Recommended Citation

This Article is brought to you for free and open access by RIT Scholar Works. It has been accepted for inclusion in Articles by an authorized administrator of RIT Scholar Works. For more information, please contact ritscholarworks@rit.edu.
DESIGN AND IMPLEMENTATION OF A DSP BASED MPEG-1 AUDIO ENCODER
Eric Hoekstra, Muhammad Shaaban, Roy Czernikowski, Soheil Dianat

ABSTRACT - The speed of current PCs enable them to decode and play an MPEG bitstream in real-time. The encode process, however, cannot be done in real-time. This paper presents a low-cost real-time fixed point DSP implementation of an MPEG1 layer 1 audio encoder. The DSP transmits the encoded MPEG bitstream to a PC through a standard bi-directional parallel port, which the PC can immediately save to disk. The encoder is capable of transmitting bitstreams to the PC at bit-rates from 32kbps - 320 kbps.

I. INTRODUCTION

A Digital Signal Processor (DSP) has many advantages over fabricating an ASIC. It is cheaper and reduces the amount of hardware design that is required. Past work has been done with implementing a DSP based MPEG-1 encoder as in [1]. A single floating point TI DSP was used to implement an encoder and decoder. The analog input samples were encoded, decoded, and then output as an analog signal. During this process, the MPEG bitstream was never constructed or transmitted off chip. This paper presents an encoder constructed from a fixed point DSP that transmits the MPEG bitstream to a PC for immediate use.

II. MPEG ENCODE PROCESS

MPEG is a perceptually lossless subband based compression method. Since information is lost in the coding process, the original signal cannot be recovered from the compressed data. MPEG removes information from the audio signal that a listener would not be able to hear in the original signal.

The encoder block diagram for the MPEG ISO/IEC 11172-3 standard is shown in Figure 1. The filter bank divides the input signal into 32 subbands. The psychoacoustic model calculates the Signal to Mask Ratio (SMR) for each of the subbands. The bit and noise allocation stage uses this SMR to determine how each subband should be quantified. The final stage properly formats the information into an MPEG frame. The standard allows for three different layers of compression. Each layer has a different frame format and increasing encoder complexity. The audio data to be compressed can be originally sampled at 32kHz, 44.1kHz or 48kHz. For each layer, there are 14 possible bit-rates for the compressed data.

The audio data is broken down into 32 subbands of reduced frequency. These subbands are equally spaced in layer 1 and 2. In layer 3, the bandwidths of the subbands are more closely related to critical bandwidth. MPEG audio compression uses psychoacoustic principles to determine the signal to mask ratio for each subband. This information is used to dynamically distribute the available bits among the subbands.

There are 384 samples in an MPEG layer 1 frame. These 384 samples are critically filtered to produce 384 subband samples. Every 32 samples, the most recent 512 points are filtered by a polyphase filter. The polyphase filter contains 32 equally spaced non-overlapping bandpass filters. The filter produces one output for each of the subbands. This process repeats 12 times for each frame, to produce 12 filtered points for each subband. This effectively runs the 32 bandpass filters on the same data and downsamples the filtered data to produce critically sampled subbands. Figure 2 shows the method of calculating the subband samples as stated in the standard. It can be shown that each output S[i] can be represented by a 512 point FIR filter. [2]. The frequency response of the first four subband filters is shown in Figure 3. The top plot shows the complete frequency spectrum from 0 to π, and the bottom plot shows a close up in the area where the filters begin to overlap.
Figure 2: Subband Calculations

Shift 32 new samples into FIFO buffer X[i]

Window X by C[i] vector

\[ Z[i] = X[i] \times C[i] \]

for \( i = 0 \ldots 511 \)

Partial Calculations

\[ Y[i] = \sum_{j=0}^{64} Z[i+64j] \]

for \( i = 0 \ldots 63 \)

Calculate Subband Samples

\[ S[i] = \sum_{k=0}^{31} M[i,k] \times Y[j] \]

for \( i = 0 \ldots 31 \)

\[ M[i,k] = \cos \left( \frac{(2i+1)(k-16)\pi}{64} \right) \]

Besides validating the encoder algorithms, the PC encoder served as a testbench for observing the effects of 24 bit fixed point calculations on the encoding process. The DSP performs calculations with 24 fixed-point fractional number, while the PC performs calculations with floating point values. This enables the PC to produce results that contain more precision than the DSP. This allowed the PC encoder to analyze the effects of rounding errors during 24 bit FFT calculations. The PC encoder simulated the DSP's precision by using an optional mode of calculations where all floating point multiplies were immediately quantized to 24 bits and then converted back to floating point. The PC encoder showed that 24 bit fixed point calculations were sufficient for performing the encoding process.

IV. DSP IMPLEMENTATION

With a low cost solution in mind, the MPEG encoder was implemented using a DSP56302EVM. The EVM contains a DSP, 32KB of external SRAM, and a 16 bit stereo codec. The 66MHz DSP has 20KB of on-chip program RAM and 14KB of on-chip data RAM. This is large enough so that all program and data can be stored in the on-chip memories. The DSP contains many features such as: single cycle 24 bit multiply, nested hardware DO loops, 56 bit barrel shifter, 8 bit parallel host interface, triple timer, SCI, two ESSI, and 6 channel DMA support.

The DSP encoder works in a pipelined manner. There are three levels to the pipeline. The subband samples are calculated in the first stage. The frame processing and bitstream formatting are done in the second stage, and the bitstream is transmitted in the final stage.

The current implementation of the encoder does not perform any psychoacoustic calculations for determining the SMR. With the psychoacoustic model removed from the DSP encoder, the bit allocation stage does not receive the SMR that the psychoacoustic model normally produces. The bit allocation stage needs the SMR to calculate the Mask-to-Noise Ratio (MNR) for all the subbands. Bits are allocated to the subbands to minimize the MNR.

A heuristic was needed to generate a SMR that represents the energy present in the signal without performing the FFT calculations. The SMR is calculated by subtracting the mask level from the signal level. In psychoacoustic calculations, the FFT is used to determine the masking levels for each subband. With the FFT removed, an expression for

III. PC ENCODER

A PC MPEG encoder was written in C to examine the facets of the encoding process and test the functionality of the DSP algorithms. The PC encoder was written using the standard [3,4] as a guide. Using a high level language, such as C, allowed for quick changes to the encoder algorithm. All code was first tested for functionality in C before being ported to DSP assembly.
the masking level for each subband needs to be determined. The solution is to set the masking level for all subbands equal. The masking level can then be removed from the SMR calculations because the SMR is now independent of the masking level.

Equation 1: Determination of Sound Pressure Level
\[ L_{sb} = \max \left[ X(k), 20 \log_{10} (\text{scalefactor}(n) \cdot 32768) - 10 \right] \]
for \( X(n) \) in subband \( n \)

The signal power is calculated using the expression in Equation 1. The signal power is the maximum of the scalefactor energy and the spectral power. With equal masking for all the subbands, the SMR can be approximated by Equation 2.

Equation 2: Approximation of SMR
\[ \text{SMR}(sb) = 20 \log_{10} (\text{scalefactor}(sb) \cdot 32768) \]

V. DSP - PC TRANSFER

One of the major goals is to provide a means to transmit the MPEG bitstream to a PC in real time. The bit-rates for MPEG-1 layer 1 range from 32kbps to 448kbps. The serial port of the PC does not have sufficient bandwidth to receive the amount of data transmitted by the encoder. The parallel port was used in the bi-directional mode to perform the transfers. The eight-bit data bus of the printer port was used for the bitstream data and the control lines used for handshaking.

- ACK
- DATA
- STR

Figure 4: Double Edge Acknowledge

The devised protocol for the transmission of the bitstream, shown in Figure 4, uses both edges of the control lines for acknowledgement. A control signal (INIT) is required to synchronize both the ACK and the STR lines to the high state. From the initialization state, the transmitter sets the data line and then toggles the state of the ACK line. Once the receiver sees that the ACK line has changed state, it reads the data and toggles the STR line to acknowledge receiving the data.

A PC program receives data from the DSP through the parallel port and writes the MPEG bitstream to disk. The program continuously polls the parallel port for new data. Once new data is available, it adds the data to a buffer in memory. Periodically, the buffer is written to disk. The PC program is in control of the INIT line. A simple method of detecting a synchronization error was implemented with counters. The INIT signal is set high whenever a timeout occurs.

The state of the PC receiver is based on the previous ACK signal read from the status register. Table 1 shows the state table used by the PC program. The DSP toggles the value of ACK whenever new data is available. The PC program saves the contents of the data register whenever it detects a change of the ACK line, and sends back a response by setting the STR signal to the new value of ACK.

<table>
<thead>
<tr>
<th>State</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timeout Occurred on read</td>
<td>INIT='1'</td>
</tr>
<tr>
<td>Previous ACK high</td>
<td>Add data to buffer</td>
</tr>
<tr>
<td>Previous ACK low</td>
<td>Add data to buffer</td>
</tr>
</tbody>
</table>

The DSP transmitter is a state machine based on the current value of ACK. The INIT signal synchronizes the transmitter with the receiver, and causes the DSP to initialize ACK to '1'. The transmitter places the first byte on the data lines and lowers the ACK signal to indicate the data is valid. After that point, the transmitter checks until the STR line is equal to the new state of ACK. The transmitter then knows the receiver has acknowledged the data. The transmitter can then transmit the next byte of the bitstream and toggle the state of the ACK. Table 2 and Figure 5 show the state table and the state diagram for the DSP transmitter.
Table 2: State Table for DSP Transmitter

<table>
<thead>
<tr>
<th>State</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>INIT high</td>
<td>Data = '0'</td>
</tr>
<tr>
<td>ACK high</td>
<td>ACK = '1'</td>
</tr>
</tbody>
</table>
| ACK high   | If STR = '1'
|            | Data = next data
|            | ACK = '0'
| ACK low    | If STR = '0'
|            | Data = next data
|            | ACK = '1'

VI. CONCLUSIONS

The increasing amount of multimedia content demands the need for real-time processing of video and audio signals. MPEG audio provides high compression of an audio signal with little removal of audio quality. The general-purpose processors in today's PCs are not suited for the real-time encoding of an audio source. Producing a custom ASIC to perform the encoding requires extensive design and higher cost for small production volumes. The DSP is quickly becoming the tool used to process data in real-time. In addition to being optimized to process data in real-time, development on a DSP can be done with assembly or high level languages. Both the DSP hardware and the development tools are inexpensive.

This paper presents a low cost real-time MPEG audio encoder that transmits the MPEG bitstream to a PC. An additional concern was to minimize the amount of additional hardware needed to interface with the PC. Requiring the interface to use additional PC hardware makes the encoder less portable. A standard bi-directional parallel port was the only PC hardware requirement to interface with the DSP encoder.

The DSP was able to encode MPEG bitstreams between 32kbps and 320kbps. The DSP-PC transfer limited the maximum bit-rate to be 320kbps. Higher bit-rates could be encoded by the DSP, but the host transfer would have to be improved. A counter was used to count the number of idle cycles available for future expansion of the encoder. The results from the idle counters indicate that the DSP is idle from approximately 44% to 67% of the time.

Table 3: Idle times as a function of bit-rate

<table>
<thead>
<tr>
<th>Bit-rate</th>
<th>% idle</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td>67.6%</td>
</tr>
<tr>
<td>64</td>
<td>65.3%</td>
</tr>
<tr>
<td>96</td>
<td>62.8%</td>
</tr>
<tr>
<td>128</td>
<td>60.2%</td>
</tr>
<tr>
<td>160</td>
<td>57.5%</td>
</tr>
<tr>
<td>192</td>
<td>54.6%</td>
</tr>
<tr>
<td>224</td>
<td>51.9%</td>
</tr>
<tr>
<td>256</td>
<td>49.3%</td>
</tr>
<tr>
<td>288</td>
<td>46.8%</td>
</tr>
<tr>
<td>320</td>
<td>44.0%</td>
</tr>
</tbody>
</table>

REFERENCES


Eric Hoekstra is a MS/BS student in Computer Engineering at Rochester Institute of Technology, Rochester, New York. His research interest include data compression and digital signal processing.

Muhammad Shaaban is currently an assistant professor in the Department of Computer Engineering at Rochester Institute of Technology. He received his BS and MS in electrical engineering from the University of Petroleum and Minerals, Saudi Arabia and his Ph.D. in computer engineering from the University of Southern California.
Roy S. Czernikowski is currently the department head of Computer Engineering at Rochester Institute of Technology. He received his BS degree from The Catholic University of America, and his ME and Ph.D. degrees from Rensselaer Polytechnic Institute, all in Electrical Engineering. His specialization is in Computer Architecture, Realtime Systems, Digital Systems, Operating Systems, and Computer Communications.

Soheil A. Dianat is currently a professor in the Department of Electrical Engineering at Rochester Institute of Technology. He received his Ph.D. from George Washington University. He specializes in Control Systems and Signal Processing.