Voice over IP application on TMS320C6701 EVM DSP Board

Isaac Skog, Gustav Almquist, Stefan Andersson, Petko Petkov

KTH Signals, Sensors and Systems, Royal Institute of Technology
SE–100 44 Stockholm, Sweden, Fax: +46 8 790 7260
E-mail: skog@kth.se

Abstract—Development of a low bit-rate voice over IP application is described. A speech coding algorithm is constructed and implemented on a digital signal processor (DSP). The algorithm is based on a sub-band mixed excitation LPC vocoder structure and operates at a rate of 2000 bits per second. Implementation is done on two TMS320C6701 floating point DSP’s each connected to a PC. On the PC, a graphical user interface (GUI) controls the data flow over the IP network. The application runs in full duplex and supports poor to fair sound quality depending on the used recording equipment.

INTRODUCTION

IP-telephony offers many benefits in comparison with the regular telephony service. The same network can be used for both data and speech transmissions, offering lower maintenance costs. It is possible to send both the coded speech as well as side information. This side information could for example be video streaming in a video conference application or a topographic map of the battlefield in a military communication system. Even though Internet wide band connections keeps getting wider, it is still important not to waste the available bandwidth. In military communication systems a low bit rate is often more important than perfect sound quality.

This paper will present the development and implementation of a low bit rate speech coding algorithm onto a DSP. The DSP together with a PC is set up to form a voice over IP telephony application. From the PC, the user can set up a call and control different features of the speech coding algorithm to hear the impact on the speech quality.

The work has been carried out as part of a project course in signal processing and digital communication at KTH Signals, Sensors and Systems, Royal Institute of Technology, Sweden [8].

THE SPEECH CODER

A mixed excitation linear prediction (MELP) algorithm utilize the fact that speech samples are highly correlated. To remove the redundancy in the sampled speech signal a linear prediction is done, trying to model the speech as an AR process. The parameters of the AR-process corresponds to filter coefficients of an all pole production filter, with a frequency response similar to the spectral envelope of the speech frame. The production filter will perfectly reconstruct the speech signal if fed with the prediction error. In a MELP algorithm the exaltation is modelled as a mixture of pulses and noise.

![Encoder](image)

Figure 1. Encoder.

Encoder

In the algorithm the most computational complex part is the encoder. The following section provides information on the different features of the encoder. A system overview is shown in Figure 1. The encoder operates on a frame-by-frame basis, where each speech frame is 20 ms, i.e., 160 samples at a sample rate of 8000 Hz. To remove DC bias and power supply disturbances which strongly effect
The linear prediction coefficient (LPC) analysis of the speech frame is high-pass filtered, using a 4th order Chebyshev 2 filter with a cut-off frequency of 60 Hz.

A 10th order linear prediction analysis is done on the speech frame to calculate the production filter. The characteristics of speech doesn't change significantly between contiguous frames. Therefore a Hamming window is used to smooth the LPC analysis and thereby reducing the spectral distortion between consecutive frames. The 220 point Hamming window is centered around the analysis frame and consists of 30 samples from the previous and next speech frame. The filter parameters can be found as the solution to the Yule-Walker equations. Solving this equation system requires the autocorrelation sequence of the speech. Since this is unknown an unbiased estimate is produced. The equation system is efficiently solved by utilizing the Levinson Durbin recursion.

As mention earlier, feeding the production filter with the prediction error will perfectly reconstruct the speech signal. In MELP algorithms this residual is modelled as a mixture of pulses and noise. Finding the parameters of the residual model can be divided into the following parts.

- Pitch search (Estimation of the pulse period).
- Finding the mixing ratio.

The pitch period extraction plays an important roll in LPC coders. Inaccurate pitch tracking results in robotic and unnatural sounding speech. A reliable and often used method for pitch tracking is peak picking in the autocorrelation (ACF) sequence calculated based on the ideal residual. The reliability of this method is however poor in regions of varying pitch. The pitch extracting algorithm used in this MELP is based upon the work of Kwon et al [1]. An estimate of the pitch frequency can be obtained by picking the lag $M$ that maximizes (1) below, where $r_{ee}$ is the autocorrelation of the normalized residual.

$$K(M) = r_{ee}^2(M-1)+r_{ee}^2(M)-2r_{ee}(1)r_{ee}(M-1)r_{ee}(M)$$

(1)

Pitch detection through the peak picking in $K(M)$ requires extra computation but increases the reliability of the pitch search. By utilizing the fact that the pitch does not dramatically change between to concatenating speech frames the calculated pitch values are smoothed, to avoid unrealistic changes between the pitch estimates.

The proper mixing ratio of pulses and noise in the residual is obtained by a method suggested by Makhoul et al [2], where the speech frame is divided into five frequency sub-bands. In each sub-band a voiced or unvoiced decision is done. From these decisions an excitation consisting of a proper mix of low-pass filtered pulses and high-pass filtered noise is determined.

The calculation of the voicing decision is related to that of [3], where the degree of periodicity in the pitch period is estimated. With the difference that each sub-band is processed individually.

At first, a fractional pitch analysis is performed on the lowest frequency band (0 - 2000 Hz). The correlation coefficients are calculated for lags between 19 and 140, where the peak value corresponds to the fractional pitch estimate. The fractional pitch estimate is then used to calculate the voicing “value” for all sub-bands. The voicing value is the sum of normalized correlation coefficients around the fractional pitch. If the value exceeds a set threshold, the sub-band is assumed voiced. The threshold is different for each sub-band and has been optimized individually using the frequency voicing distribution [4].

Before the final voicing decision is made a “peakiness” calculation is also performed [5]. The peakiness value $p$ is defined as the ratio of the RMS power to the average value of the full-wave rectified LPC residual signal, see equation (2) below, where $e(n)$ is the prediction error (residual). If $p$ exceeds 1.6, the previous voiced or unvoiced decision will be over written and the lowest three sub-bands (0 - 2000 Hz) are forced to voiced. If $p$ exceeds 1.34, the lowest sub-band (0 - 500 Hz) is forced to voiced.

$$p = \frac{\sqrt[\frac{1}{N}]{\sum_{n=0}^{N-1} e(n)^2}}{\frac{1}{N} \sum_{n=0}^{N-1} |e(n)|}$$

(2)

Before the parameters of the processed speech frame are send to the decoder they must be quantized. LPC coefficients are extremely sensitive to quantization errors. Small changes in the LPC coefficients may result in large changes in the filter spectrum and possibly an unstable LPC synthesis filter. By converting the LPC coefficient into line spectral frequencies (LSF), an alternative and more robust representation of the synthesis filter is achieved. The LSF is then quantized using a 2-stage vector quantization. Where each code book has 1024 vector entries, giving a total of 10 bits for each of the separate quantization stages. In Table 1 the parameters of the coded speech frame is shown, with the corresponding bit allocation.

<table>
<thead>
<tr>
<th>PARAMETERS</th>
<th>BITS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pitch period</td>
<td>7</td>
</tr>
<tr>
<td>Energy</td>
<td>8</td>
</tr>
<tr>
<td>LSF’s (first stage)</td>
<td>10</td>
</tr>
<tr>
<td>LSF’s (second stage)</td>
<td>10</td>
</tr>
<tr>
<td>Band-pass voicing strengths</td>
<td>5</td>
</tr>
<tr>
<td>TOTAL bits / 20 ms</td>
<td>40</td>
</tr>
</tbody>
</table>

Table 1

Bit allocation.
Decoder

The purpose of the decoder is to reproduce the speech given the parameters from the encoder. A system overview is shown in Figure 3.

The parameters of each speech frame is received on a package-by-package basis. There exists three types of packages: empty package, signaling that there is no speech to reproduce, package including voice information and lost package.

Since no re-sending of lost packages is possible due to the real times constrains, the lost information has to be compensated for. In the implemented algorithm this is done by substituting the lost frame with a reduced version of the previous synthesized speech frame. Even though this is a very simple method, it has proven to be efficient. Receiving packages with speech information, speech is reproduced by generating an excitation according to voicing and pitch information, which then is filtered through a synthesis filter.

The excitation is constructed in the following manner. A pulse train of 160 samples is generated based on the pitch period. A noise sequence of the same length is also generated. The pulse train is filtered through sub-bands with the voicing strength set to one, and the opposite for the noise sequence. See Figure 4. Reconstruction of the synthesis filter is done by conversion of the de-quantized and interpolated LFS’s into LPC coefficients.

Despite the mixed excitation model, the synthetic speech has a somewhat unnatural sound. Post processing of the speech waveform, can reduce this anomaly. An adaptive spectral enhancement filter [6] is applied on the excitation before it is filtered through the synthesis filter. This adaptive filter helps the generated synthetic speech to match natural speech in the format regions by enhancing these regions, thus removing unnatural sharpness. Finally, before the speech is sent to the output device the energy of the frame is corrected.

IMPLEMENTATION

A Texas Instruments TMS320C6701 Evaluation Module Board (EVM) together with a PC was used for implementation of the voice over IP application. The PC was responsible for the data exchange over the IP-network, while the DSP handled the speech coding.

The software of the DSP can be divided into two parts: a backbone providing the data flow between the codec and the PCI, and the signal processing part, representing the speech coding.

The DSP has to communicate with the codec on one side and the host PC, through the host port interface (HPI), on the other. Communication with the codec in both directions should be interminable. That implies that a double buffering scheme is used. This is illustrated in Figure 4. Two DMA channels are utilized to communicate between the codec and the buffers. DMA channel one, which is used to fetch data from the direct receive register (DDR), generates an interrupt once a whole speech frame has been fed into the corresponding buffer. That event triggers the processing module, and also defines the real-time deadlines that needs to be met. Communication with the host PC is performed through the HPI using the so-called mailbox functionality. The sequence of operations performed by the application can be synthesized in the following manner:

1) Get interrupt from DMA channel (DRR).
2) Read input buffers on both sides.
3) Send message to PC via mailbox.
4) Perform processing. Simultaneously the PC reads from the PCI output buffer and writes to the PCI input buffer.
5) Write to output buffers on both sides. Stay in idle loop until next DMA interrupt.

Once the algorithm was designed and its operability verified in MatLab, it was ported to C and tested on a function-by-function basis to assure that it gives the same output as the MatLab prototype. Next, the functions where optimized with respect to speed to fulfil the real time constrains.

The main bottlenecks encountered during the implementation of the algorithm were the size of the on-chip data and program memory, each 64 KB. With regards to the data memory a problem occurred when trying to fit the code
books that are used in the two-stage vector quantization of the line spectral frequencies (LSF). Each code book occupies 40 KB of memory, which makes it impossible to keep them both in the internal memory. The solution to the problem, was to allocate two segments in the on-chip memory. The codebooks, on the other hand, were divided into four sections each and placed in the external memory. DMA channel three was programmed to fetch sections of the codebook and copy them to one of the two allocated segments. Thus, while the codebook search function is looking for the best candidate in a certain section of the codebook, another section is being simultaneously copied by the DMA channel to the other segment.

![Diagram](image)

**Figure 4.** Data flow chart.

The development and implementation of the voice over IP application was successful, resulting in that full duplex IP telephony conversation could take place over the local IP network at a bit rate of 2000 bps. However the sound quality was quite poor and worse than during the tests done in MatLab. This was due to the degradation of the recording quality compared to the sounds used in MatLab.

A full documentation of the work (DSP and PC source code, MatLab code and Project rapport) can be found in [8].

**RESULTS**

The development and implementation of the voice over IP application was successful, resulting in that full duplex IP telephony conversation could take place over the local IP network at a bit rate of 2000 bps. However the sound quality was quite poor and worse than during the tests done in MatLab. This was due to the degradation of the recording quality compared to the sounds used in MatLab.

A full documentation of the work (DSP and PC source code, MatLab code and Project rapport) can be found in [8].

**References**


