Computer Systems, Inc.
MERCURY

CVSD Encoder Component

FM3TR Waveform Reference Implementation

SDR Forum Contract

March 23, 2007

Revision 1.0

Table of Contents

1 Component Name

CVSD Encoder

2 Component Processing Summary

The continuously variable slope delta (CVSD) algorithm is a lossy audio encoding scheme that uses one bit per sample to compress the amount of data transmitted over a digital channel. Based upon delta modulation, CVSD continuously changes its slope to track to the input signal. If the reference value (CVSD output) is greater than the input value, a binary zero is recorded, and delta is subtracted from the reference; otherwise a binary one is recorded and delta is added. The past *N* bits are stored in a register. If all *N* bits are either zeros or ones, the value for delta is increased; otherwise delta is decreased. For this implementation, *N* was chosen to be three. Usually the value of delta has practical restrictions. The decoder simply reverses this process based upon the incoming bit stream. The resulting decoded waveform is usually low-pass filtered to remove high-frequency noise due to the lossy encoding process. This results in greater audio quality than conventional fixed-delta encoders, particularly at higher sample rates.
The CVSD encoder component uses an automatic gain control (AGC) block to reduce the dynamic range required by the encoding algorithm and further improve audio fidelity.

3 Where used

The CVSD encoder/decoder components are used in all encoded audio waveforms.

4 Data Input and Output Ports

The CVSD encoder has one uses and one provides data ports. The input data port (CVSDEncoderIn) accepts a sequence of signed short integer audio samples captured at an arbitrary frequency. After encoding, the component pushes a sequence of signed octets (one byte for encoded "bit") of the same length to the output port (CVSDEncoderOut).

5 Control Interfaces

The CVSD encoder inherits the control interfaces from CF::Resource.

6 Component SCA Properties

Aside from the DLL execparams, the CVSD encoder contains no additional SCA properties.

7 Component Attributes/Key Variables

Below is a list of several key variables to the CVSD encoding and decoding algorithms with a brief description of their purpose.

DA_MAX	Maximum level for the "digital-to-analog" converter.
AD_MAX	Maximum level for the "analog-to-digital" converter.
CVSD_STEP_MIN	Minimum step size for CVSD encoder/decoder delta value.
CVSD_STEP_MAX	Maximum step size for CVSD encoder/decoder delta value.
CVSD_DELTA_TC	Low-pass filter time constant for delta (???)
CVSD_REF_TC	Low-pass filter (integrator) time constant for input reference signal.
ATTACK_AGC_TC	AGC attack time constant. This value governs how quickly large audio signals are attenuated.
RELEASE_AGC_TC	AGC release time constant. This value governs how quickly the AGC releases attenuation after attacking.
MAX_AGC_RANGE	Maximum linear range for the AGC.
m_iOldbits	CVSD codec state for last *N* bits observed

8 Processing Details

The CVSD encoder waits for audio data in the Run() method. Once they are received, the data are passed through an automatic gain control processor before encoding. The encoded bits (each represented as a signed octet) are pushed to the next component.

8.1 Method: AutomaticGainControl()

Because CVSD encoding compresses each audio sample to a single bit, the algorithm is lossy. Furthermore, the encoder relies on the input signal having a relatively low dynamic range. Applying automatic gain control (AGC) helps to reduce the dynamic range of the input signal by maintaining a fixed power signal level. First, the signal root-mean square (RMS) value is calculated and compared to the desired level. If it is above a preset threshold, the gain is decreased. This is known as "attack" and usually occurs very quickly as determined by the variable ATTACK_AGC_TC. If, however, the RMS level is below the threshold, the gain is increased slightly. This is known as "release." The release reaction is determined by the variable RELEASE_AGC_TC and is usually much lower than the attack time constant.

8.2 Method: Encode()

The processing behind the CVSD encoding algorithm is handled by the Encode() method.