Introduction

Pulse code modulation (PCM) is a common method of digitizing or quantizing an analog waveform. For any analog-to-digital conversion process, the quantization step produces an estimate of the waveform sample using a digital codeword. This digital estimate inherently contains some level of error due to the finite number of bits available. In practical terms, there is always tradeoff between the amount of error and the size of the digital data samples. The goal in any system design is quantizing the data in smallest number of bits that results in a tolerable level of error. In the case of speech coding, linear quantization with 13 bits sampled at 8 KHz is the minimum required to accurately produce a digital representation of the full range of speech signals. For many transmission systems, wired or wireless, 13 bits sampled at 8 KHz is an expensive proposition as far as bandwidth is concerned. To address this constraint, a companding system is often employed.

Companding is simply a system in which information is first compressed, transmitted through a bandwidth-limited channel, and expanded at the receiving end. It is frequently used to reduce the bandwidth requirements for transmitting telephone quality speech, by reducing the 13-bit codewords to 8-bit codewords. Two international standards for encoding signal data to 8-bit codes are A-law and μ-law. A-law is the accepted European standard, while μ-law is the accepted standard in the United States and Japan.

Speech Companding

The human auditory system is believed to be a logarithmic process in which high amplitude sounds do not require the same resolution as low amplitude sounds. The human ear is more sensitive to quantization noise in small signals than large signals. A-law and μ-law coding apply a logarithmic quantization function to adjust the data resolution in proportion to the level of the input signal. Smaller signals are represented with greater precision – more data bits – than larger signals. The result is fewer bits per sample to maintain an audible signal-to-noise ratio (SNR).

Rather than taking the logarithm of the linear input data directly, which can be computationally difficult, A-law/μ-law PCM matches the logarithmic curve with a piece-wise linear approximation. Eight straight-line segments along the curve produce a close approximation to the logarithm function. Each segment is known as a chord. Within each chord, the piece-wise linear approximation is divided into equally size quantization intervals called steps. The step size between adjacent codewords is doubled in each succeeding chord. Also encoded is the sign bit for the original integer. The result is an 8-bit logarithmic code composed of a 1-bit sign bit, a 3-bit chord, and a 4-bit step.

A-Law Compander

A-law is the CCITT recommended companding standard used across Europe. Limiting the linear sample values to 12 magnitude bits, the A-law compression is defined by Equation 1, where *A* is the compression parameter (*A*=87.7 in Europe), and *x* is the normalized integer to be compressed.

$$F(x) = \begin{cases} \dfrac{A*|x|}{1+\ln(A)} & 0 \le |x| < \dfrac{1}{A} \\[2ex] \dfrac{\operatorname{sgn}(x)*(1+\ln(A|x|))}{1+\ln(A)} & \dfrac{1}{A} \le |x| \le 1 \end{cases}$$

Equation 1, A-Law Definition

Table 1 illustrates an A-law encoding table. The sign bit of the linear input data is omitted from the table. The sign bit (*S*) for the 8-bit code is set to 1 if the input sample is negative, and is set to 0 if the input sample is positive.

Linear Input Data												A-law Encoded Output							
0	0	0	0	0	0	0	A	B	C	D	X	S	0	0	0	A	B	C	D
0	0	0	0	0	0	1	A	B	C	D	X	S	0	0	1	A	B	C	D
0	0	0	0	0	1	A	B	C	D	X	X	S	0	1	0	A	B	C	D
0	0	0	0	1	A	B	C	D	X	X	X	S	0	1	1	A	B	C	D
0	0	0	1	A	B	C	D	X	X	X	X	S	1	0	0	A	B	C	D
0	0	1	A	B	C	D	X	X	X	X	X	S	1	0	1	A	B	C	D
0	1	A	B	C	D	X	X	X	X	X	X	S	1	1	0	A	B	C	D
1	A	B	C	D	X	X	X	X	X	X	X	S	1	1	1	A	B	C	D

Table 1, A-Law Encoding

After the input data is encoded through the logic defined in the table, an inversion pattern is applied to the 8-bit code to increase the density of transitions on the transmission line, a benefit to hardware performance. The inversion pattern is applied by XOR'ing the 8-bit code with 0x55.

Decoding the A-law encoded data is essentially a matter of reversing the steps in the encoding. Table 2 illustrates the A-law decoding table, applied after reversing the inversion pattern. The least significant bits discarded in the encoding process are approximated by the median value of the interval. This is shown in the output section by the trailing 1..0 pattern after the *D* bit.

A-law Encoded Input								Linear Output Data											
S	0	0	0	A	B	C	D	0	0	0	0	0	0	0	A	B	C	D	1
S	0	0	1	A	B	C	D	0	0	0	0	0	0	1	A	B	C	D	1
S	0	1	0	A	B	C	D	0	0	0	0	0	1	A	B	C	D	1	0
S	0	1	1	A	B	C	D	0	0	0	0	1	A	B	C	D	1	0	0
S	1	0	0	A	B	C	D	0	0	0	1	A	B	C	D	1	0	0	0
S	1	0	1	A	B	C	D	0	0	1	A	B	C	D	1	0	0	0	0
S	1	1	0	A	B	C	D	0	1	A	B	C	D	1	0	0	0	0	0
S	1	1	1	A	B	C	D	1	A	B	C	D	1	0	0	0	0	0	0

Table 2, A-Law Decoding

m-Law Compander

The United States and Japan use μ-law companding. Limiting the linear sample values to 13 magnitude bits, the μ-law compression is defined by Equation 2, where *m* is the compression parameter (*m* =255 in the U.S. and Japan) and *x* is the normalized integer to be compressed.

$$F(x) = \frac{\operatorname{sgn}(x) * \ln(1 + m|x|))}{\ln(1 + m)} \qquad 0 \le |x| \le 1$$

Equation 2, m-Law Definition

The encoding and decoding process for μ-law is similar to that of A-law. There are, however, a few notable differences: 1) μ-law encoders typically operate on linear 13-bit magnitude data, as opposed to 12-bit magnitude data with A-law, 2) before chord determination a bias value of 33 is added to the absolute value of the linear input data to simplify the chord and step calculations, 3) the definition of the sign bit is reversed, and 4) the inversion pattern is applied to all bits in the 8-bit code.

Table 3 illustrates a μ-law encoding table. The sign bit of the linear input data is omitted from the table. The sign bit (*S*) for the 8-bit code is set to 1 if the input sample is positive, and is set to 0 if the input sample is negative.

Linear Input Data													m-law Encoded Output							
0	0	0	0	0	0	0	1	A	B	C	D	X	S	0	0	0	A	B	C	D
0	0	0	0	0	0	1	A	B	C	D	X	X	S	0	0	1	A	B	C	D
0	0	0	0	0	1	A	B	C	D	X	X	X	S	0	1	0	A	B	C	D
0	0	0	0	1	A	B	C	D	X	X	X	X	S	0	1	1	A	B	C	D
0	0	0	1	A	B	C	D	X	X	X	X	X	S	1	0	0	A	B	C	D
0	0	1	A	B	C	D	X	X	X	X	X	X	S	1	0	1	A	B	C	D
0	1	A	B	C	D	X	X	X	X	X	X	X	S	1	1	0	A	B	C	D
1	A	B	C	D	X	X	X	X	X	X	X	X	S	1	1	1	A	B	C	D

Table 3, m-Law Encoding

After the input data is encoded through the logic defined in the table, an inversion pattern is applied to the 8-bit code to increase the density of transitions on the transmission line, a benefit to hardware performance. The inversion pattern is applied by XOR'ing the 8-bit code with 0xFF.

Decoding the μ -law encoded data is essentially a matter of reversing the steps in the encoding. Table 4 illustrates the μ-law decoding table, applied after reversing the inversion pattern. The least significant bits discarded in the encoding process are approximated by the median value of the interval. This is shown in the output section by the trailing 1..0 pattern after the *D* bit.

m-law Encoded Input								Linear Output Data												
S	0	0	0	A	B	C	D	0	0	0	0	0	0	0	1	A	B	C	D	1
S	0	0	1	A	B	C	D	0	0	0	0	0	0	1	A	B	C	D	1	0
S	0	1	0	A	B	C	D	0	0	0	0	0	1	A	B	C	D	1	0	0
S	0	1	1	A	B	C	D	0	0	0	0	1	A	B	C	D	1	0	0	0
S	1	0	0	A	B	C	D	0	0	0	1	A	B	C	D	1	0	0	0	0
S	1	0	1	A	B	C	D	0	0	1	A	B	C	D	1	0	0	0	0	0
S	1	1	0	A	B	C	D	0	1	A	B	C	D	1	0	0	0	0	0	0
S	1	1	1	A	B	C	D	1	A	B	C	D	1	0	0	0	0	0	0	0

Table 4, m-Law Decoding

Summary

There is a wide array of audio transmission systems that employ A-law and/or µ-law companding for data rate reduction with good audio quality. The compression achieved by both A-law and µ-law coding is the result of utilizing the logarithmic characteristics of the human auditory system, where fewer bits of precision are required for larger signals than smaller ones. The logarithmic transfer function is implemented with a piece-wise linear approximation composed of a sign bit, a 3-bit chord, and a 4-bit segment. The encoding and decoding process is presented in table format, well suited for hardware or software implementation.