The encoding process for the μ law

As it is presented in Figure 1, the voice signal is applied to the input of a linear analog-digital converter on 14 bits, at the output of this converter being obtained the 14 bits codes of the input signal's samples. The digital compressor calculates, according to the compression characteristic, the 8 bits code which will be sent on the line; this 8 bits code corresponds to a 14 bits code generated by the linear converter.

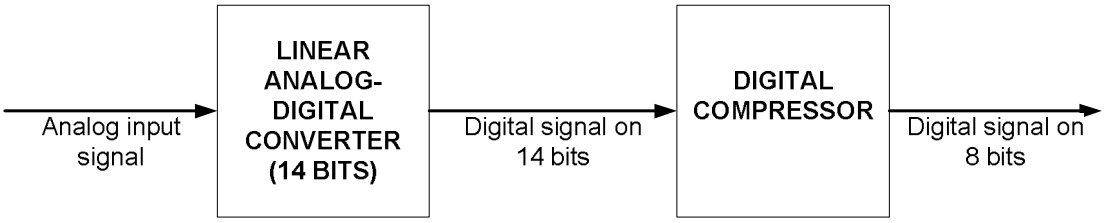

Figure 1 – The encoding process for the μ law

The dynamic range of the input signal can be computed based on footnote 1 (annex 1), which indicates the maximum power, *Tmax=3.17dBm0.* Due to the fact that the power computation uses the effective value of the signal, we will calculate the dynamic range of a sine signal. The value of the load impedance is considered $|Z| = 600\Omega$.

$$10 \cdot \lg(\frac{V_{max}^2}{2 \cdot 600 \cdot 10^{-3}}) = 3.17 \Rightarrow V_{max} = \sqrt{1.2 \cdot 10^{0.317}} = 1.57794V \quad (1)$$

The levels of the voice signal have a non-uniform distribution, the occurrence probability of low levels being significantly larger than that of large levels. Due to this reason the probability of an all zero code is large. The line code used in the case of μ law is B8ZS (Bipolar with 8 Zeros Substitution), a variant of the AMI code (Alternate Mark Inversion) which replaces the 8 consecutive zero bits sequences with other sequences including violations of the AMI coding rule. Because the AMI coding represents the zero bits by zero voltage levels the bit clock synchronization is affected. Even if the B8ZS code replaces the 8 bit long zero sequences, it still remains the possibility to have sequences of 7 consecutive zero bits. Due to this fact and tacking into consideration the large occurrence probability of sequences with many zero bits, the codes generated at the output of the digital compressor are inverted.

Can be noticed (see annex 1) that after inverting the code assigned to the first positive sub-segment (sign:0, segment:000, sub-segment:0000) is 11111111b (FFh), a code identical with the blue alarm indication signal is obtained; due to this fact instead the code of the first positive sub-segment is transmitted the code of the first negative sub-segment (sign:1, segment:000, sub-segment:0000), which is 01111111b (7Fh) after the bit level inversion. This code is transmitted on the line, when the line is not used. (The codes specified in the 6th column of the table from annex 1 are the inverted ones).

To simplify the implementation of the digital compressor, the value 33 is added to the binary codes applied to the input of this block. This operation has the purpose to make the end values of the segments power of two - (see column 3, annex 1: 31+33=64, 95+33=128, 223+33=256,...). As a result of this operation, the decision on the value of the

segment of the sample which is compressed/processed can be taken based on the position of the first 1 bit (starting from MSB) of the code applied to the digital compressor input; see Table 1, presenting the implementation of the digital compression process for the μ law.

Input of the digital compressor (after the offset of 33 is added)														Compressed code							
															Segment			Sub-segment			
bit:	12	11	10	9	8	7	6	5	4	3	2	1	0	bit:	6	5	4	3	2	1	0
	0	0	0	0	0	0	0	1	a	b	c	d	x		0	0	0	a	b	c	d
	0	0	0	0	0	0	1	a	b	c	d	x	x		0	0	1	a	b	c	d
	0	0	0	0	0	1	a	b	c	d	x	x	x		0	1	0	a	b	c	d
	0	0	0	0	1	a	b	c	d	x	x	x	x		0	1	1	a	b	c	d
	0	0	0	1	a	b	c	d	x	x	x	x	x		1	0	0	a	b	c	d
	0	0	1	a	b	c	d	x	x	x	x	x	x		1	0	1	a	b	c	d
	0	1	a	b	c	d	x	x	x	x	x	x	x		1	1	0	a	b	c	d
	1	a	b	c	d	x	x	x	x	x	x	x	x		1	1	1	a	b	c	d

Table 1 – Encoding table for the μ law

We can notice in table 1 that the position of the first 1 bit indicates the code of the segment.

SEGMENT CODE$_{10}$=7–*no.of zeros located on the front of the first 1 bi t*=-5+*postion of the first 1 bit* (2)

The code of the sub-segment is represented by the next 4 bits located after the first 1 bit.

For example if at the input of the digital compressor, after the addition of the 33 offset, we have the code 1000011101 1100 we can use relation (2) to compute the code of the segment:

$$\text{SEGMENT CODE}_{10} = 7\text{-}4=3 \Rightarrow \text{SEGMENT CODE}_2=011$$

and the code of the sub-segment is represented by the next 4 bits :

$$\text{SUBSEGMENT CODE}_2=1101$$

The sign bit remains unchanged:

$$\text{SIGN}_2=1$$

The binary code generated is 10111101_2. This code is inverted and is transmitted on the line. At the destination the code received from the line is inverted again and it is generated the code 10111101_2. Based on the segment code we calculate the position of the first 1 bit and we get the code 00001xxxxxxxx; after that the next 4 positions are replaced with the sub-segment code, being obtained the code 000011101xxxx. We have to fill in the last 4 positions without having any information about these bits. In order to minimize the quantization error, the last unknown positions are replaced with the code representing the middle (approximately) of the sub-segment. In such a way the maximum quantization error is half of the quantization interval (see Figure 2 for supplementary explanation). The code generated is 0000111011000_2, and after adding the sign bit we get 10000111011000_2.

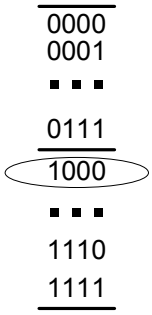

Figure 2 – Replacing of the suppressed bits at reception

The encoding process for the A law

In the case of the A compression law, the length of the segments is changed, and the first segment has 32 sub-segments, being obtained a total number of only 7 segments, both for the positive and the negative values. The first segment can be divided into two smaller segments, each of them having 16 sub-segments and in such a way the structure of the code remains the same as that of the μ law. The uniform quantization is performed only on 13 bits and it is not necessary to add any offset to the input values, the segment ends being power of 2 (column 3, annex 2). The coding is performed according to the encoding table (Table 2) and only the even bits of the obtained code are inverted. The line code used is HDB3 (High Density Bipolar 3 Zeros), which is also a modified AMI code, which replaces sequences of 4 consecutive zero bits with violations of the AMI coding rule.

Input of the digital compressor													Segment			Sub-segment			
bit:	11	10	9	8	7	6	5	4	3	2	1	0	bit:	6	5	4	3	2	1	0
	0	0	0	0	0	0	0	a	b	c	d	x		0	0	0	a	b	c	d
	0	0	0	0	0	0	1	a	b	c	d	x		0	0	1	a	b	c	d
	0	0	0	0	0	1	a	b	c	d	x	x		0	1	0	a	b	c	d
	0	0	0	0	1	a	b	c	d	x	x	x		0	1	1	a	b	c	d
	0	0	0	1	a	b	c	d	x	x	x	x		1	0	0	a	b	c	d
	0	0	1	a	b	c	d	x	x	x	x	x		1	0	1	a	b	c	d
	0	1	a	b	c	d	x	x	x	x	x	x		1	1	0	a	b	c	d
	1	A	b	c	d	x	x	x	x	x	x	x		1	1	1	a	b	c	d

Table 2 – Encoding table for the A law

It can be noticed that the difference which appears in the coding table, relatively to the μ law, is only in the first two segments.

Example

In the case of the A compression law and a -1.5V - 1.5V dynamic range of the input signal, compute the code generated at the output of the digital compressor for an input voltage Vin=-0.45779V.

Coding

Method 1 – using the compression characteristic:

- Because the compression characteristics are normalized, the first step is to normalize the input voltage:

$$Vin_{normalized} = \frac{-0.45779}{1.5} = -0.305193$$

- The next step is to establish the value of the sign bit, $b_{sign}=0$, and after that we will use only the absolute value of the normalized input voltage $|Vin_{normalized}| = 0.305193$;
- The next step is to calculate the segment which includes the input voltage. To perform this operation we consider the X axis of the compression characteristic (A law) and we can notice that the normalized input value is located in segment 6, identified by the edge values 0.25, 0.5. The code of the segment obtained: 110_2;
- The last step is the computation of the sub-segment which includes the value of the input voltage. To perform this operation has to be computed the quantization step of segment 6, $q_6 = \frac{0.5-0.25}{16} = 0.015625$. Using this value the sub-segment which includes the input voltage can be computed as:

$$sub-segment = \frac{0.305193-0.25}{0.015625} = 3.53 \Rightarrow sub-segment\_code = 0011_2$$

- Finally the code obtained at the output of the compressor is: 01100011;

Method 2 – using the encoding table

- In the first step we have to calculate the linear code applied to the input of the compressor. For this we have to compute the elementary quantization step: $q_e = \frac{1}{2^{12}}$, and after that the code generated by the linear converter is computed as:

$$value = \frac{|Vin_{normalized}|}{q_e} = 0.305193 \cdot 2^{12} = 1250.070528 \Rightarrow value = 1250,$$ value represented as the binary stream: 010011100010_2

- The sign bit is $b_{sign}=0$, and using relation (2) or Table 2 we can calculate the SEGMENT CODE = 110_2, and SUB-SEGMENT CODE =0011_2;
- The code obtained at the output of the digital compressor is the same as that obtained with method 1: 01100011;

Decoding

Method 1 – using the compression characteristic

At destination the value of the output voltage can be calculated as:

$$V_{out} = Inf\_limit_{seg\_i} + sub - segment\_code \cdot q_i + \frac{1}{2} \cdot q_i$$

and we obtain: $|V_{out}| = 0.25 + 3 \cdot 0.015625 + \frac{1}{2} \cdot 0.015625 = 0.3046875$, the quantization error being: $quant\_error = 0.305193 - 0.3046875 = 0.0005055$.

Method 2 – using the encoding table

Based on the received code and using Table 2 we can get the sign, the position of first 1 bit and the next 4 bits, located right after the mentioned 1 bit: 0010011xxxxxx. We replace the last 6 bits and we obtained 0010011100000_2. By converting this code in decimal we obtain the value 1248 and the output voltage is:

$|V_{out}| = 1248 \cdot q_e = 1248 \cdot 2^{-12} = 0.3046875$.

Annex 1. μ law encoding/decoding table (source: Recommendation G.711)

TABLEAU 2a / G.711

μ-law, positive input values

1	2	3	4	5	6	7	8
Segment number	Number of intervals × interval size	Value at segment end points	Decision value number n	Decision value x_n (see Note 1)	Character signal Bit number 1 2 3 4 5 6 7 8	Quantized value (value at decoder output) y_n	Decoder output value number
		8159	(128)	(8159)			
8	16 × 256		127	7903	1 0 0 0 0 0 0 0 (see Note 2)	8031	127
			113	4319	1 0 0 0 1 1 1 1	4191	112
		4063	112	4063			
7	16 × 128				(see Note 2)		
			97	2143	1 0 0 1 1 1 1 1	2079	96
		2015	96	2015			
6	16 × 64				(see Note 2)		
			81	1055	1 0 1 0 1 1 1 1	1023	80
		991	80	991			
5	16 × 32				(see Note 2)		
			65	511	1 0 1 1 1 1 1 1	495	64
		479	64	479			
4	16 × 16				(see Note 2)		
			49	239	1 1 0 0 1 1 1 1	231	48
		223	48	223			
3	16 × 8				(see Note 2)		
			33	103	1 1 0 1 1 1 1 1	99	32
		95	32	95			
2	16 × 4				(see Note 2)		
			17	35	1 1 1 0 1 1 1 1	33	16
		31	16	31			
	15 × 2				(see Note 2)		
			2	3	1 1 1 1 1 1 1 0	2	1
1			1	1	1 1 1 1 1 1 1 1	0	0
↓	1 × 1		0	0			

Note 1 – 8159 normalized value units correspond to T_{max} = 3.17 dBm0.

Note 2 – The character signal corresponding to positive input values between two successive decision values numbered n and $n + 1$ (see column 4) is $(255 - n)$ expressed as a binary number.

Note 3 – The value at the decoder output is $y_0 = x_0 = 0$ for $n = 0$, and $y_n = \frac{x_n + x_{n+1}}{2}$ for $n = 1, 2, ..., 127$.

Note 4 – x_{128} is a virtual decision value.

Note 5 – **In Tables 1/G.711 and 2/G.711 the values of the uniform code are given in columns 3, 5 and 7.**

Annex 2. A law encoding/decoding table (source: Recommendation ITU-T G.711)

TABLE 1a/G.711

A-law, positive input values

1	2	3	4	5	6	7	8
Segment number	Number of intervals × interval size	Value at segment end points	Decision value number n	Decision value x_n (see Note 1)	Character signal before inversion of the even bits Bit number 1 2 3 4 5 6 7 8	Quantized value (value at decoder output) y_n	Decoder output value number
		4096	(128)	(4096)			
					1 1 1 1 1 1 1 1	4032	128
7	16 × 128		127	3968	(see Note 2)		
			113	2176	1 1 1 1 0 0 0 0	2112	113
		2048	112	2048	(see Note 2)		
6	16 × 64		97	1088	1 1 1 0 0 0 0 0	1056	97
		1024	96	1024	(see Note 2)		
5	16 × 32		81	544	1 1 0 1 0 0 0 0	528	81
		512	80	512	(see Note 2)		
4	16 × 16		65	272	1 1 0 0 0 0 0 0	264	65
		256	64	256	(see Note 2)		
3	16 × 8		49	136	1 0 1 1 0 0 0 0	132	49
		128	48	128	(see Note 2)		
2	16 × 4		33	68	1 0 1 0 0 0 0 0	66	33
		64	32	64	(see Note 2)		
1 ↓	32 × 2		1	2	1 0 0 0 0 0 0 0	1	1
			0	0			

Note 1 – 4096 normalized value units correspond to T_{max} = 3.14 dBm0.

Note 2 – The character signals are obtained by inverting the even bits of the signals of column 6. Before this inversion, the character signal corresponding to positive input values between two successive decision values numbered n and $n + 1$ (see column 4) is $(128 + n)$ expressed as a binary number

Note 3 – The value at the decoder output is $y_n = \frac{x_{n-1} + x_n}{2}$ for $n = 1, ..., 127, 128$.

Note 4 – x_{128} is a virtual decision value.

Note 5 – **In Tables 1/G.711 and 2/G.711 the values of the uniform code are given in columns 3, 5 and 7.**