INTERNATIONAL STANDARD ISO/IEC 23003-3:2012
TECHNICAL CORRIGENDUM 3

Published 2015-04-01

Corrected version 2016-10-01

Information technology — MPEG audio technologies —

Part 3: **Unified speech and audio coding**

TECHNICAL CORRIGENDUM 3

Technologies de l'information — Technologies audio MPEG —

Partie 3: Discours unifié et codage audio

RECTIFICATIF TECHNIQUE 3

Technical Corrigendum 3 to ISO/IEC 23003-3:2012 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 29, *Coding of audio, picture, multimedia and hypermedia information*.

This corrected version of Technical Corrigendum 3 to ISO/IEC 23003-3:2012 contains 2 replacement formulae in the text concerning 7.17 to improve legibility.

ICS 35.040 **Ref. No. ISO/IEC 23003-3:2012/Cor.3:2015(E)**

Published in Switzerland

In 5.2, Table 12, replace:

Syntax	No. of bits	Mnemonic
SbrDfltHeader()		
{		
dflt_start_freq;	**4**	**uimsbf**
dflt_stop_freq;	**4**	**uimsbf**
dflt_header_extra1;	**1**	**uimsbf**
dflt_header_extra2;	**1**	**uimsbf**
if (dflt_header_extra1 == 1) {		
dflt_freq_scale;	**2**	**uimsbf**
dflt_alter_scale;	**1**	**uimsbf**
dflt_noise_bands;	**2**	**uimsbf**
}		
if (dflt_header_extra2 == 1) {		
dflt_limiter_bands;	**2**	**uimsbf**
dflt_limiter_gains;	**2**	**uimsbf**
dflt_interpol_freq;	**1**	**uimsbf**
dflt_smoothing_mode;	**1**	**uimsbf**
}		
}		

with:

Syntax	No. of bits	Mnemonic
SbrDfltHeader()		
{		
dflt_start_freq;	**4**	**uimsbf**
dflt_stop_freq;	**4**	**uimsbf**
dflt_header_extra1;	**1**	**uimsbf**
dflt_header_extra2;	**1**	**uimsbf**
if (dflt_header_extra1 == 1) {		
dflt_freq_scale;	**2**	**uimsbf**
dflt_alter_scale;	**1**	**uimsbf**
dflt_noise_bands;	**2**	**uimsbf**
} else {		
dflt_freq_scale = 2;		
dflt_alter_scale = 1;		
dflt_noise_bands = 2;		
}		
if (dflt_header_extra2 == 1) {		
dflt_limiter_bands;	**2**	**uimsbf**
dflt_limiter_gains;	**2**	**uimsbf**
dflt_interpol_freq;	**1**	**uimsbf**
dflt_smoothing_mode;	**1**	**uimsbf**
} else {		
dflt_limiter_bands = 2;		
dflt_limiter_gains = 2;		
dflt_interpol_freq = 1;		
dflt_smoothing_mode = 1;		
}		
}		

In 5.3.2 replace:

Table 27 – Syntax of tw_data()

Syntax	No. of bits	Mnemonic
tw_data( )		
{		
tw_data_present;	**1**	**uimsbf**
if (tw_data_present == 1) {		
for (i = 1 ; i < NUM_TW_NODES ; i++) {		
tw_ratio[i];	**3**	**uimsbf**
}		
}		
}		

with

Table 27 – Syntax of tw_data()

Syntax	No. of bits	Mnemonic
tw_data( )		
{		
tw_data_present;	**1**	**uimsbf**
if (tw_data_present == 1) {		
for (i = 0 ; i < NUM_TW_NODES ; i++) {		
tw_ratio[i];	**3**	**uimsbf**
}		
}		
}		

In 5.3.3, Table 45 replace:

```
        sbr_grid(0, 0);
        sbr_dtdf(0,0, indepFlag);
        sbr_dtdf(1,0, indepFlag);
        sbr_invf(0);
```

[…]

```
    }
}
```

with:

```
            sbr_grid(0, 0);                                   NOTE 1
            if(!bs_coupling) sbr_grid(1, 0);
            sbr_dtdf(0, 0, indepFlag);
            sbr_dtdf(1, 0, indepFlag);
            sbr_invf(0);
            if(!bs_coupling) sbr_invf(1);                     NOTE 1
```

[…]

```
      }
}
```

NOTE 1: In coupling mode only the data for channel 0 is read. This data shall be copied to the same syntax element for channel 1.

In 5.3.3, Table 46 replace:

```
            switch (bs_frame_class) {                        2          uimsbf
```

with:

```
            switch (bs_frame_class[ch]) {                    2          uimsbf
```

In 5.3.3, Table 46, delete:

```
                        if (bs_num_env[ch] == 1)
                            bs_amp_res = 0;
```

In 5.3.3, Table 47, replace:

```
Syntax                                                    No. of bits    Mnemonic
sbr_envelope(ch, bs_coupling, bs_amp_res)
{
    if (bs_coupling) {
        if (ch) {
            if (bs_amp_res) {
```

[…]

with:

```
Syntax                                                              No. of bits   Mnemonic
sbr_envelope(ch, bs_coupling, bs_amp_res)
{
    amp_res = bs_amp_res;
    if (bs_frame_class[ ch ] == FIXFIX && bs_num_env[ ch ] == 1) {
        amp_res = 0;
    }
    if (bs_coupling) {
        if (ch) {
```

[…]

Further, replace "bs_amp_res" with "amp_res" in the rest of the syntax element sbr_envelope().

In 7.5.5.2, add to the requirements:

- The largest interval from the $\mathbf{f}_{\mathrm{Master}}$, i.e. $\mathbf{f}_{\mathrm{Master}}(\mathrm{N}_{\mathrm{Master}}) - \mathbf{f}_{\mathrm{Master}}(\mathrm{N}_{\mathrm{Master}} - 1)$ shall satisfy $\mathbf{f}_{\mathrm{Master}}(\mathrm{N}_{\mathrm{Master}}) - \mathbf{f}_{\mathrm{Master}}(\mathrm{N}_{\mathrm{Master}} - 1) \le \mathrm{k}_0 - 2$

In 7.13.3 replace:

In addition to the 1 to 4 LPC filters of the superframe, an optional LPC0 is transmitted for the first super-frame of each segment encoded using the LPD core codec. This is indicated to the LPC decoding procedure by a flag first_lpd_flag set to 1.

with:

In addition to the 1 to 4 LPC filters of the superframe, an optional LPC0 is transmitted for the first super-frame of each segment encoded using the LPD core codec. This is indicated to the LPC decoding procedure by a flag first_lpd_flag set to 1. In case of first_lpd_flag==0, LPC0 shall be equal to LPC4 of the previous super frame.

In 7.14.4, replace:

In case of a transition from FD to ACELP, the past excitation buffer u'(n) and the buffer containing the past pre-emphasized synthesis $\hat{s}(n)$ are updated using the past FD synthesis (including FAC) and LPC0 prior to the decoding of the ACELP excitation.

with:

In case of a transition from FD to LPD, the past excitation buffer u'(n) and the buffer containing the past pre-emphasized synthesis $\hat{s}(n)$ are updated using the past FD synthesis (including FAC or the overlapped TCX-signal) and LPC0 prior to the decoding of the ACELP excitation.

In 7.14.5.1, replace:

When the pitch value is encoded on 6 bits, a pitch resolution of 1/4 is always used in the range [T1-8, T1+7¾], where T1 is nearest integer to the fractional pitch lag of the previous subframe.

With:

When the pitch value is encoded with 6 bits, a pitch resolution of 1/4 is always used in the range [T1-8, T1+7¾], where T1 is the rounded down integer of the fractional pitch lag of the previous subframe. To be able to use as many different pitch lags as possible T1 has to be between TMIN+8 and TMAX-7. So in case T1 < TMIN+8 set T1=TMIN+8, just as if T1 > TMAX-7 set T1=TMAX-7.

In 7.14.6.3, replace:

$$c'(n) = c(n) - c_{pe}(c(n+1) + c(n-1))$$

(…)

$$u(n) = \hat{g}_p v(n) + \hat{g}_{sc} c(n) - \hat{g}_{sc} c_{pe}(c(n+1) + c(n-1))$$

With:

$$c'(n) = \begin{cases} c(0) - c_{pe} c(1) & \text{if } n = 0 \\ c(n) - c_{pe}(c(n+1) + c(n-1)) & \text{if } 0 < n < 63 \\ c(63) - c_{pe} c(62) & \text{if } n = 63 \end{cases}$$

(…)

$$u(n) = \begin{cases} \hat{g}_p v(0) + \hat{g}_{sc} c(0) - \hat{g}_{sc} c_{pe} c(1) & \text{if } n = 0 \\ \hat{g}_p v(n) + \hat{g}_{sc} c(n) - \hat{g}_{sc} c_{pe}(c(n+1) + c(n-1)) & \text{if } 0 < n < 63 \\ \hat{g}_p v(63) + \hat{g}_{sc} c(63) - \hat{g}_{sc} c_{pe} c(62) & \text{if } n = 63 \end{cases}$$

After the following paragraph in 7.17:

After LP synthesis, the reconstructed signal can be post-processed using low-frequency pitch enhancement. The received bass-post filter control information controls whether bass-post filtering which results in a pitch enhancement in the low frequency range is enabled or not. For speech signals, the post processing filter reduces inter-harmonic noise in the decoded signal, which leads to an improved quality. However, for music signals, which are commonly of multi-pitch nature, the post filtering may suppress signal components that reside below the dominating pitch frequency or between its harmonics. For the post filtering a two-band decomposition is used and adaptive filtering is applied only to the lower band. This results in a total post-processing that is mostly targeted at frequencies near the first harmonics of the synthesized signal.

Add:

To avoid additional delay due to bass-post filtering, bass-post filter operation is modified for high values of T. Therefore, $T_{\lim}$ is defined as follows.
In case of LPD:

- For the first $\frac{M}{2}+64$ samples of a superframe:

$$T_{\lim} = M - L_{fac} - N_z$$

- For the last $\frac{M}{2}-64$ samples of a superframe:

$$T_{\lim} = 2M - L_{fac\_next} - N_z$$

In case of FD (the FAC-area):

$$T_{\lim} = \frac{M}{2} - N_z$$

Where *M*=coreCoderFrameLength, L_{fac} is the length of the FAC area from the last frame of the current superframe. With $L_{fac} = 0$ for ACELP and $L_{fac} = 96/128$ for TCX (coreCoderFrameLength 768/1024). L_{fac_next} is the length of the FAC area from the last frame of the next superframe. N_z is the number of samples of the superframe up to and including the sample currently being bass post filtered.

And in chapter 7.17 replace:

[…] where $P_{LT}(z)$ is the transfer function of the long-term predictor filter given by

$$P_{LT}(z) = 1 - 0.5z^{T} - 0.5z^{-T}$$

with:

[…] where $P_{LT}(z)$ is the transfer function of the long-term predictor filter given by

$$P_{LT}(z) = \begin{cases} 1 - 0.5z^{T} - 0.5z^{-T} & ,if\ T \le T_{\lim} \\ 1 - z^{-T} & ,if\ T > T_{\lim} \end{cases}$$