INTERNATIONAL STANDARD ISO/IEC 13818-7:2006
TECHNICAL CORRIGENDUM 1

Published 2009-04-15

Information technology — Generic coding of moving pictures and associated audio information —

Part 7:
Advanced Audio Coding (AAC)

TECHNICAL CORRIGENDUM 1

Technologies de l'information — Codage générique des images animées et du son associé —

Partie 7: Codage du son avancé (AAC)

RECTIFICATIF TECHNIQUE 1

Technical Corrigendum 1 to ISO/IEC 13818-7:2006 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 29, *Coding of audio, picture, multimedia and hypermedia information*.

Page 54, 8.5.3.3, replace the first sentence:

If no explicit channel mapping is given, the following methods describe the implicit channel mapping:

with:

If no explicit channel mapping is given, the channel configuration is either known by the application (for examples of such application specific channel configurations see Annex H) or the following methods might be used to describe the implicit channel mapping:

Page 84, 12.3.3, replace:

```
   - decode spectral coefficients of embedded single_channel_element
     into buffer "cc_spectrum[]".

   /* Couple spectral coefficients onto target channels */
```

with:

```
   /*
      first:
         decode spectral coefficients of embedded single_channel_element
         into buffer "cc_spectrum[]"
         (no pseudo code is given for this task)

      second:
         Couple spectral coefficients onto target channels
         (according to the following pseudo code)
   */
```

Page 85, 12.3.3, replace:

```
couple_channel(source_spectrum[], dest_spectrum[], gain_list_index)
{
   idx = gain_list_index;
   a = 0;
   cc_scale = cc_scale_table[gain_element_scale];
   for (g = 0; g < num_window_groups; g++) {

      /* Decode coupling gain elements for this group */
      if (common_gain_element_present[idx]) {

         for (sfb = 0; sfb < max_sfb; sfb++) {
            cc_sign[idx][g][sfb] = 1;
            gain_element[idx][g][sfb] = common_gain_element[idx];
         }
      }
      else {
         for (sfb = 0; sfb < max_sfb; sfb++) {
            if (sfb_cb[g][sfb] == ZERO_HCB)
               continue;

            if (gain_element_sign) {
               cc_sign[idx][g][sfb] = 1 - 2*(dpcm_gain_element[idx][g][sfb] & 0x1);
               gain_element[idx][g][sfb] = a += (dpcm_gain_element[idx][g][sfb] >>
1);
            }
            else {
               cc_sign[idx][g][sfb] = 1;
               gain_element[idx][g][sfb] = a += dpcm_gain_element[idx][g][sfb];
            }
         }
      }
      /* Do coupling onto target channels */
      for (b = 0; b < window_group_length[b]; b++) {
         for (sfb = 0; sfb < max_sfb; sfb++) {

            if (sfb_cb[g][sfb] != ZERO_HCB) {
               cc_gain[idx][g][sfb] = cc_sign[idx][g][sfb] *
cc_scale^gain_element[idx][g][sfb];
                 for (i = 0; i<swb_offset[sfb+1]-swb_offset[sfb]; i++)
                 dest_spectrum[g][b][sfb][i] += cc_gain[idx][g][sfb] *
                 source_spectrum[g][b][sfb][i];
```

```
                }
            }
        }
    }
}
```

with (where changes to content are highlighted grey):

```
couple_channel(source_spectrum[], dest_spectrum[], gain_list_index) {
     idx = gain_list_index;
     a = 0;
     cc_scale = cc_scale_table[gain_element_scale];
     for (g = 0; g < num_window_groups; g++) {
          /* Decode coupling gain elements for this group */
          if (common_gain_element_present[idx]) {
               for (sfb = 0; sfb < max_sfb; sfb++) {
                    cc_sign[idx][g][sfb] = 1;
                    gain_element[idx][g][sfb] = common_gain_element[idx];
               }
          }
          else {
               for (sfb = 0; sfb < max_sfb; sfb++) {
                    if (sfb_cb[g][sfb] == ZERO_HCB)
                         continue;
                    if (gain_element_sign) {
                         a += (dpcm_gain_element[idx][g][sfb] >> 1);
                         cc_sign[idx][g][sfb] = 1 - 2*(a & 0x1);
                         gain_element[idx][g][sfb] = a;
                    }
                    else {
                         cc_sign[idx][g][sfb] = 1;
                         gain_element[idx][g][sfb] = a +=
dpcm_gain_element[idx][g][sfb];
                    }
               }
          }
          /* Do coupling onto target channels */
          for (b = 0; b < window_group_length[g]; b++) {
               for (sfb = 0; sfb < max_sfb; sfb++) {
                    if (sfb_cb[g][sfb] != ZERO_HCB) {
                         cc_gain[idx][g][sfb] = cc_sign[idx][g][sfb]
                                    * cc_scale^(-gain_element[idx][g][sfb]);
                              for (i = 0; i<swb_offset[sfb+1]-swb_offset[sfb]; i++)
                              dest_spectrum[g][b][sfb][i] += cc_gain[idx][g][sfb] *
                              source_spectrum[g][b][sfb][i];
                    }
               }
          }
     }
}
```

After Annex G, add the following new annex: