Technical Specification Group Services and System Aspects ***TSGS#15(02)0081***
Meeting #15, Cheju Island, Korea, 11-14 March 2002

Source: **TSG-SA WG4**

Title: CRs to TS 26.173 on "Correction of mode reading and memory usage", "Correction of pitch calculation of AMR-WB encoder", and "Error concealment of high band gain in 23.85 kbit/s mode" (Release 5)

Document for: **Approval**

Agenda Item: **7.4.3**

The following CRs, agreed at the TSG-SA WG4 meeting #20, are presented to TSG SA #15 for approval.

Spec	CR	Rev	Phase	Subject	Cat	Vers	WG	Meeting	S4 doc
26.173	011	2	REL-5	Correction of mode reading and memory usage	F	5.3.0	S4	TSG-SA WG4#20	S4-020175
26.173	012		REL-5	Correction of pitch calculation of AMR-WB encoder	F	5.3.0	S4	TSG-SA WG4#20	S4-020060
26.173	013		REL-5	Error concealment of high band gain in 23.85 kbit/s mode	F	5.3.0	S4	TSG-SA WG4#20	S4-020172

3GPP TSG-SA4 Meeting #20
Luleå, Sweden, 18-22nd February 2002

Tdoc S4-020060

CR-Form-v5

CHANGE REQUEST

⌘ **TS 26.173** **CR** **012** ⌘ **rev** **-** ⌘ Current version: **5.3.0** ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

Proposed change affects: ⌘ (U)SIM [ ] ME/UE [X] Radio Access Network [ ] Core Network [X]

Title:	⌘	Correction of pitch calculation of AMR-WB encoder
Source:	⌘	TSG SA WG4
Work item code:	⌘	AMRWB
Date:	⌘	2002-03-11
Category:	⌘	**F**
Release:	⌘	REL-5

Use one of the following categories:
F *(correction)*
A *(corresponds to a correction in an earlier release)*
B *(addition of feature),*
C *(functional modification of feature)*
D *(editorial modification)*
Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:
2 (GSM Phase 2)
R96 (Release 1996)
R97 (Release 1997)
R98 (Release 1998)
R99 (Release 1999)
REL-4 (Release 4)
REL-5 (Release 5)

Reason for change:	⌘	The calculated pitch value can saturate to minimum pitch lag value (34), when the real pitch period is below that value. This may occur when speaker is high pitched woman or child.
Summary of change:	⌘	In openloop pitch lag correlation computation "FOR-loop" execution is stopped when i > L_min, not when i >= L_min.
Consequences if not approved:	⌘	Pitch periods under the lower limit of 34, but very close to it, saturates into 34 and it causes constant pitch period of 34 to be generated in the decoder. This causes degradation in speech quality, because pitch value is not changing in time.

Clauses affected:	⌘	p_med_ol.c
Other specs affected:	⌘	[X] Other core specifications ⌘ TS 26.174 [ ] Test specifications [ ] O&M Specifications
Other comments:	⌘	

1. Background

Marginal problem with certain high pitch signals have been found from the AMR-WB codec. This may occur with certain woman or child speakers with high pitch.

In the AMR-WB codec open loop lag is searched for even integer values between [34,231] and the first phase of the closed loop search locates best integer lag around that. Lowest lag that can be quantised in the codec is 34. It may now occur that the true lag value of the coded signal is below the lower limit of 34, but very close to that. In that case the best correlation in the open loop lag search may give lag value of 34. However, due to lag quantisation of the codec, we can not perform closed loop search for the lag values below 34 and therefore lag search saturates in the lower (34). In the other words, this means that all the true lag values that lies below, but so close to the lower limit of 34 that open loop pitch search will choose that value, will be quantised as value of 34. However, if the true lag is well below the lower limit, the maximum correlation will, in many cases, be found in the value of twice the true lag value.

Solution to that problem is simply to limit the correlation search in the open loop pitch search routine between [36,231]. This modification still does not change the lower limit of the lag quantisation (34), because closed loop search will search around the open loop lag and will cover lag values down to 34. However, that modification helps to the observed problem, because it is very unlikely that true lag values below the lower limit of 34 will give maximum correlation in the open loop lag search with the lag values of 36. On the contrary, in the example case maximum correlation in the open loop lag search is most probably found in the double the true lag value, which is better in terms of speech quality. The figure 1 shows an example where the abnormal saturation of lag value happens and then shows the lag contour after the correction has been applied.

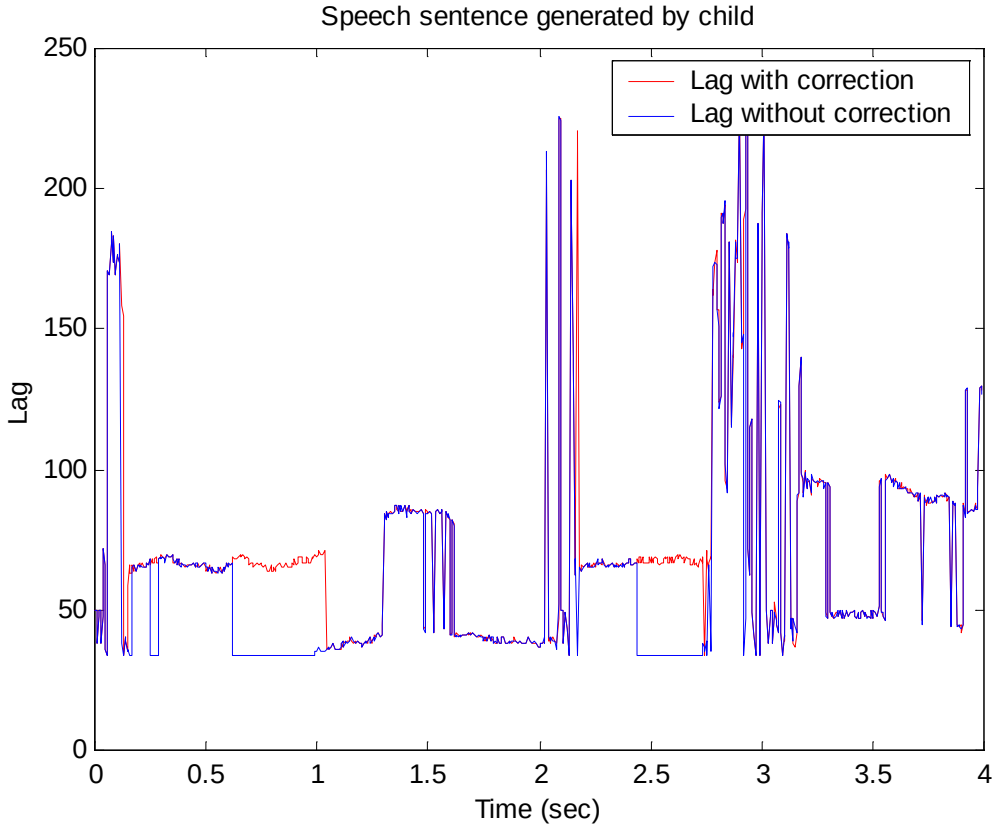

Figure 1 Speech sentence of child.

2. How 3GPP TS 26.173 V5.3.0 is changed

p_med_ol.c, line 43

Equal sign removed for correct operation of pitch search

```
for (i = L_max; i >= L_min; i--)
```

3GPP TSG-SA Meeting #20
Luleå, Sweden, 18-22nd February 2002

Tdoc S4-020175

CR-Form-v5

CHANGE REQUEST

⌘ **26.173** **CR** **011** ⌘ **rev** **2** ⌘ Current version: **5.3.0** ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

Proposed change affects: ⌘ (U)SIM [ ] ME/UE [X] Radio Access Network [ ] Core Network [X]

Title:	⌘	Correction of mode reading and memory usage
Source:	⌘	TSG SA WG4
Work item code:	⌘	AMRWB
Date:	⌘	2002-03-11
Category:	⌘	**F**
Release:	⌘	REL-5

Use one of the following categories:
F *(correction)*
A *(corresponds to a correction in an earlier release)*
B *(addition of feature),*
C *(functional modification of feature)*
D *(editorial modification)*
Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:
2 (GSM Phase 2)
R96 (Release 1996)
R97 (Release 1997)
R98 (Release 1998)
R99 (Release 1999)
REL-4 (Release 4)
REL-5 (Release 5)

Reason for change:	⌘	There are four following reasons for change: 1) Encoder reads mode information incorrectly from mode control file. 2) Some static memory is allocated, but not used. 3) All resources are not released when codec is closing. 4) Function 'round' is declared twice (math.h and basicop.h)
Summary of change:	⌘	1) Mode fetching operation is modified. (coder.c) 2) Allocated static memory is released after use (requires one new function). (bits.c, bits.h, coder.c, decoder.c) 3) Unused static vectors "mem_hf3" and "disp_mem" removed from encoder state structure 4) Function calls "Init_Filt_7k" and "Init_Phase_dispersion" are removed from function "Reset_encoder", because functions are not needed in the encoder. (cod_main.c, cod_main.h) 5) 'include <math.h>' is removed from three files (agc2.c, dtx.c, laqconc.c)
Consequences if not approved:	⌘	1) Incorrect mode information is used when mode file is used 2) codec has memory and resource leaks 3) 46 bytes of memory is allocated and initialised without any purpose 4) Code may not compile in some platforms

Clauses affected:	⌘	coder.c, bits.c, bits.h, decoder.c, cod_main.c, cod_main.h, agc2.c, dtx.c, laqconc.c		
Other specs affected:	⌘	[X] Other core specifications [ ] Test specifications [ ] O&M Specifications	⌘	TS 26.174
Other comments:	⌘			

How the code is changed

bits.c

New function definition is added to line 38

```
Word16 Close_write_serial(TX_State *st);
{
   /* allocate memory */
    test();
    if (st != NULL)
    {
        free(st);
        st = NULL;
        return 0;
    }
    return 1;
}
```

bits.h

New function reference is added to line 52

```
Word16 Close_write_serial(TX_State *st)
```

cod_main.h, lines (47 - 61)

Unused memory allocation removed from line 58 and 60

```
Word16 isfold[M];                       /* old isf (frequency domain) */
    Word32 L_gc_thres;                      /* threshold for noise enhancer */
    Word16 mem_syn_hi[M];                   /* modified synthesis memory (MSB) */
    Word16 mem_syn_lo[M];                   /* modified synthesis memory (LSB) */
    Word16 mem_deemph;                      /* speech deemph filter memory */
    Word16 mem_sig_out[6];                  /* hp50 filter memory for synthesis
*/
    Word16 mem_hp400[6];                    /* hp400 filter memory for synthesis
*/
    Word16 mem_oversamp[2 * L_FILT];        /* synthesis oversampled filter
memory */
    Word16 mem_syn_hf[M];                   /* HF synthesis memory */
    Word16 mem_hf[2 * L_FILT16k];           /* HF band-pass filter memory */
    Word16 mem_hf2[2 * L_FILT16k];          /* HF band-pass filter memory */
    Word16 mem_hf3[2 * L_FILT16k];          /* HF band-pass filter memory */
    Word16 seed2;                           /* random memory for HF generation */
    Word16 disp_mem[8];                     /* phase dispersion memory */
    Word16 vad_hist;
```

cod_main.c, lines 112 - 115

Unused phase dispersion initialisation is removed from line 115

```
    Init_gp_clip(cod_state->gp_clip);

    cod_state->L_gc_thres = 0;              move16();
    Init_Phase_dispersion(cod_state->disp_mem);
```

cod_main.c, lines 170 - 181

Unused filter initialisation is removed from line 175

```
cod_state->mem_deemph = 0;          move16();

        cod_state->seed2 = 21845;           move16();

        Init_Filt_6k_7k(cod_state->mem_hf2);
        Init_Filt_7k(cod_state->mem_hf3);
        cod_state->gain_alpha = 32767;      move16();

        cod_state->vad_hist = 0;

        wb_vad_reset(cod_state->vadSt);
        dtx_enc_reset(cod_state->dtx_encSt, isf_init);
```

coder.c, line 56-59 and lines 164 - 176

Mode vector removed (line 59), coding mode initialised (line 56)

```
    Word16 coding_mode = 0, nb_bits, allow_dtx, mode_file, mode = 0, i;
    Word16 reset_flag;
    long frame;
    char Mode[2] = "0";
```

Mode control file read operation changed (164-176)

```
if (mode_file)
        {
            if (fread(Mode, sizeof(char), 1, f_mode) != 1)
            if (fscanf(f_mode, "%hd", &mode) == EOF)
            {
                mode = coding_mode;
                fprintf(stderr, "\nend of mode control file reached\n");
                fprintf(stderr, "From now on using mode: %d kbit/s.\n",
nb_of_bits[mode]);
                fprintf(stderr, " From now on using mode: %hd\n", mode);
                mode_file = 0;
            }

            mode = (Word16) atoi(Mode);
            if ((mode < 0) || (mode > 8))
            {
               fprintf(stderr, " error in bit rate mode %hd: use 0 to 8\n",
mode);
               exit(0);
            }

        }
        coding_mode = mode;
```

```
        frame++;
        fprintf(stderr, " Frames processed: %ld\r", frame);
        fprintf(stderr, " Frames processed: %hd\r", frame);
```

coder.c, line 199

Code lines added to release the resources that were earlier left unreleased

```
/* free allocated memory */
    Close_coder(st);
    Close_write_serial(tx_state);
    fclose(f_speech);
    fclose(f_serial);
    if (f_mode != NULL)
    {
       fclose(f_mode);
    }
```

decoder.c, line 175

Code lines added to close the files earlier left open.

```
Close_decoder(st);
    fclose(f_serial);
    fclose(f_synth);
```

agc2.c, lines 7-9

Math library header include removed.

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
```

dtx.c, lines 7-9

Math library header include removed.

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
```

laqconc.c, lines 7-10

Math and float library header includes removed.

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <float.h>
```

3GPP TSG-SA Meeting #20
Luleå, Sweden, 18-22nd February 2002

Tdoc S4-020172

CR-Form-v5

CHANGE REQUEST

⌘ **TS 26.173** **CR** **013** ⌘ **rev** **-** ⌘ Current version: **5.3.0** ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

Proposed change affects: ⌘ (U)SIM ☐ ME/UE **X** Radio Access Network ☐ Core Network **X**

Title:	⌘	Error concealment of high-band gain in 23.85 kbit/s mode
Source:	⌘	TSG SA WG4
Work item code:	⌘	AMRWB
Date:	⌘	2002-03-11
Category:	⌘	**F**
Release:	⌘	REL-5

Use one of the following categories:
F *(correction)*
A *(corresponds to a correction in an earlier release)*
B *(addition of feature),*
C *(functional modification of feature)*
D *(editorial modification)*
Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:
2 (GSM Phase 2)
R96 (Release 1996)
R97 (Release 1997)
R98 (Release 1998)
R99 (Release 1999)
REL-4 (Release 4)
REL-5 (Release 5)

Reason for change:	⌘	There is no substitution procedure for high-band energy parameter (mode 23.85 kbit/s).
Summary of change:	⌘	Substitution procedure is added for high-band energy parameter.
Consequences if not approved:	⌘	No bad frame substitution is performed for the high-band gain parameter. That parameter is extracted from the erroneous/lost frame and used as such.

Clauses affected:	⌘	dec_main.c		
Other specs affected:	⌘	**X** Other core specifications ☐ Test specifications ☐ O&M Specifications	⌘	TS 26.191
Other comments:	⌘			

1. Background

The high-band gain parameter used in 23.85 kbit/s mode is decoded even the current frame is detected as an erroneous frame or the frame is lost in transmission. In these cases, high-band gain parameter is corrupted or undefined. Therefore, substitution procedure also for high-band energy parameter is needed.
According to this change, the received high-band gain parameter is not used in bad/lost frames. Instead, the calculated estimation value for the high-band gain is used (like it is used in all the other modes).

2. Changes to C-code

dec_main.c, lines 54-64

Input parameter bfi added into synthesis function introduction

```
static void synthesis(
     Word16 Aq[],                          /* A(z)  : quantized Az               */
     Word16 exc[],                         /* (i)   : excitation at 12kHz        */
     Word16 Q_new,                         /* (i)   : scaling performed on exc   */
     Word16 synth16k[],                    /* (o)   : 16kHz synthesis signal     */
     Word16 prms,                          /* (i)   : parameter                  */
     Word16 HfIsf[],
     Word16 nb_bits,
     Word16 newDTXState,
     Decoder_State * st                    /* (i/o) : State structure            */
     Word16 bfi                            /* (i)   : bad frame indicator        */
);
```

dec_main.c, lines 325-335

Input parameter bfi added into synthesis function call

```
for (i_subfr = 0; i_subfr < L_FRAME; i_subfr += L_SUBFR)
        {
            j = shr(i_subfr, 6);
            for (i = 0; i < M; i++)
            {
                L_tmp = L_mult(isf_tmp[i], sub(32767, interpol_frac[j]));
                L_tmp = L_mac(L_tmp, isf[i], interpol_frac[j]);
                HfIsf[i] = round(L_tmp);   move16();
            }
            synthesis(Aq, &exc2[i_subfr], 0, &synth16k[i_subfr * 5 / 4], (short) 1,
HfIsf, nb_bits, newDTXState, st, bfi);
        }
```

dec_main.c, lines 912-917

Input parameter bfi added into synthesis function call

```
if (sub(nb_bits, NBBITS_24k) >= 0)
        {
            corr_gain = Serial_parm(4, &prms);
            synthesis(p_Aq, exc2, Q_new, &synth16k[i_subfr * 5 / 4], corr_gain, HfIsf,
nb_bits, newDTXState, st, bfi);
        } else
            synthesis(p_Aq, exc2, Q_new, &synth16k[i_subfr * 5 / 4], 0, HfIsf, nb_bits,
newDTXState, st, bfi);
```

dec_main.c, lines 949-959

Input parameter bfi added into synthesis function definition

before change

```
static void synthesis(
     Word16 Aq[],                          /* A(z)  : quantized Az               */
     Word16 exc[],                         /* (i)   : excitation at 12kHz        */
     Word16 Q_new,                         /* (i)   : scaling performed on exc   */
     Word16 synth16k[],                    /* (o)   : 16kHz synthesis signal     */
     Word16 prms,                          /* (i)   : parameter                  */
     Word16 HfIsf[],
     Word16 nb_bits,
     Word16 newDTXState,
     Decoder_State * st                    /* (i/o) : State structure            */
     Word16 bfi                            /* (i)   : bad frame indicator        */
)
```

dec_main.c, lines 1093-1111

Bad frame substitution added for the mode 23.85 kbit/s

```
    if (sub(nb_bits, NBBITS_24k) >= 0 && (bfi == 0))
    {
        /* HF correction gain */
        HF_gain_ind = prms;
        HF_corr_gain = HP_gain[HF_gain_ind];

        /* HF gain */
        for (i = 0; i < L_SUBFR16k; i++)
        {
            HF[i] = shl(mult(HF[i], HF_corr_gain), 1);  move16();
        }
    } else
    {
        for (i = 0; i < L_SUBFR16k; i++)
        {
            HF[i] = mult(HF[i], tmp);      move16();
        }
    }
```