Source: Editor[1]
Title: **Processing Functions for WB-AMR Subjective Experiments (WB-7a) version 0.11**
Document for: Review
Agenda Item:

---

[1] Contact: Paul Barrett
Tel: +44 1473 645655
Fax: +44 1473 644649
e-mail: paul.a.barrett@bt.com

THIS PAGE INTENTIONALLY LEFT BLANK

CONTENTS

Outstanding Issues

1. The add module must be provided (BT can provide C source code for this function).
2. An error insertion device must be provided that inserts frame erasures and residual bit errors into the WB-AMR bit stream. The format of the bit stream is defined in Annex B. Frame erasures should be taken from a 20ms FER file and implemented by setting the G.192 BFI flag. Residual bit errors will be taken from a RBER file organised into 20ms frames and mapped onto the X% least significant bits of the G.192 speech codec frame (the number of bits in 20ms is defined in the G.192 header). X is defined in the requirements for Application E as the percentage of bits subject to residual bit errors.

Introduction

This document defines how speech material must be prepared for the WB-AMR Qualification Phase subjective experiments, and provides details of the software modules and files required. It should be read in conjunction with the WB-AMR Test Plan (WB-8a).

1   Definitions

The following filenames are used in the command line descriptions:

input8   8kHz speech file that is the input to a processing module (or sequence of modules)
output8   8kHz speech file that is the output from a processing module (or sequence of modules)

input16   16kHz speech file that is the input to a processing module (or sequence of modules)
output16   16kHz speech file that is the output from a processing module (or sequence of modules)

noise16   16kHz background noise file

The format of the above files is headerless PCM with samples stored in 16-bit 2's complement format. Other command line filenames and variables will be described in the relevant part of the document.

2   Obtaining the Software Modules

2.1   ITU-T Software Tool Library 2000

Host laboratories <u>must</u> use the STL-2000 version of the ITU-T Software Tool Library (STL)[2]. This can be obtained the ITU-T WP3/16 informal ftp site using a TIES account (available to all ITU-T members). For further information please consult the TIES web site:

> http://www.itu.int/TIES/

The details for downloading the STL-2000 by FTP are as follows:

> Source: ftp.ties.itu.int
> Location: /u/tsg16/sg16/wp3/q22
> Filename: stl2000r1.zip
> Format: C source code

This zip file is password protected. Please contact the following person for a password:

> *Simao Campos Neto*
> ITU-T Q.22/16 (Software Tools) Rapporteur
> Tel: +1-301-428-4516
> Fax: +1-301-428-9287
> Email: simao.campos@comsat.com

2.2   Accessing the ETSI Server

Some modules and files are available from the ETSI ftp server, which has the following address:

> docbase.etsi.org

In order to access the ETSI server, the user must have an account with SMG11 access. For information on how to obtain an account contact:

> ETSI Help-Desk
> Phone: +33 (0)4 92 94 42 75
> Fax: +33 (0)4 92 96 03 07
> X.400:   c=FR; a=ATLAS; p=ETSI; s=HELPDESK;
> Internet: helpdesk@etsi.fr

The address of the ETSI web site is:

---

[2] Earlier versions of the STL do not provide the functionality required by this processing plan.

www.etsi.org

3   Processing Stages for WB-AMR Experiments

This Section defines, in the form of diagrams, the processing stages required by the AMR characterisation experiments.

The figures in Section 3.2 apply to the wideband experiments (Exps. 1a, 2a and 2b of the Qualification Phase); the figures in Section 3.3 apply to the narrowband (Exps. 1b, 2c and 2d of the Qualification Phase); the figure in Section 3.4 applies to all experiments. Note that the tandeming configurations in Figure 6 and Figure 13 are provided for future use.

Each listening laboratory has the responsibility to pre-process the speech samples it provides to the host laboratory according to Section 3.2.1 and Section 3.3.1, with the exception of the summation of the speech and background noise files (see Figure 2 and Figure 10), which is the responsibility of the host laboratory. All other processing tasks are the responsibility of the host laboratory.

3.1   File concatenation, separation and module initialisation

All speech files will be processed separately. Thus all processing functions must be reset to their initial state at the start of each 8-second sentence pair.

3.2   Wideband Experiments

3.2.1   Pre-Processing
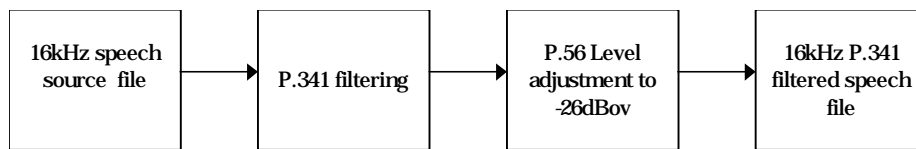
Figure 1: Pre-processing of P.341 filtered speech-only file.
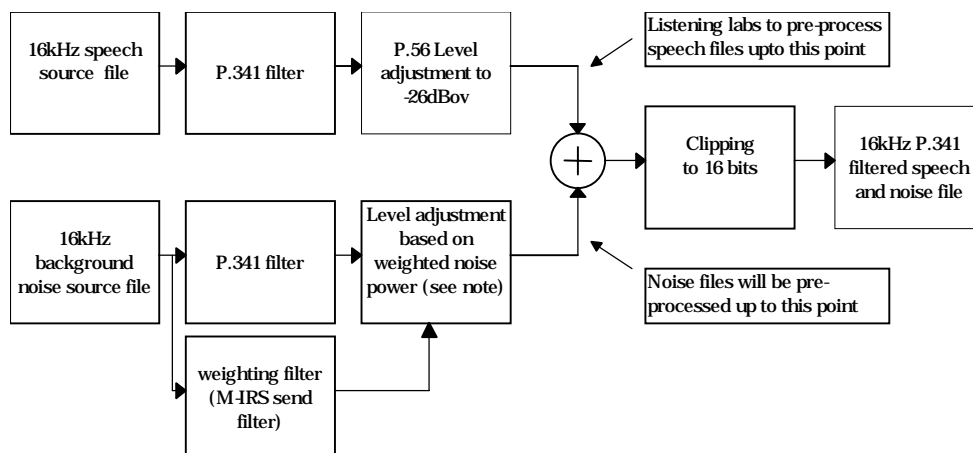
Figure 2: Pre-processing of P.341 filtered speech and noise file. (Note: the process for scaling the background noise is described in Annex A.)

3.2.2 Rounding and Scaling
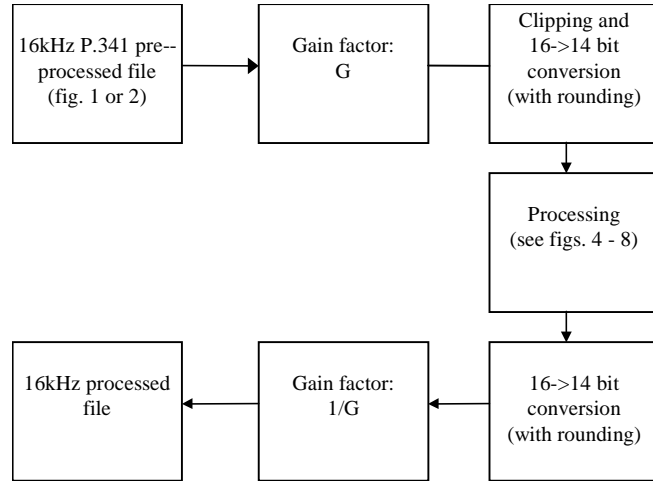
Figure 3: Rounding and scaling for wideband conditions.

3.2.3 Processing Stages
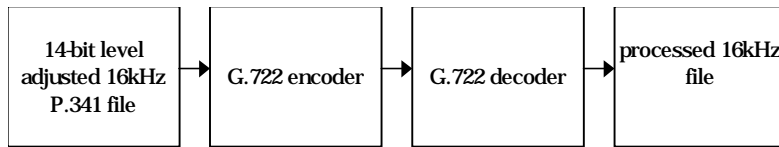
Figure 4: Processing of G.722 conditions.

Figure 5: Processing of WB-AMR conditions.

Figure 6: Processing of simulated tandem connection using G.722.1/WB-AMR codec.
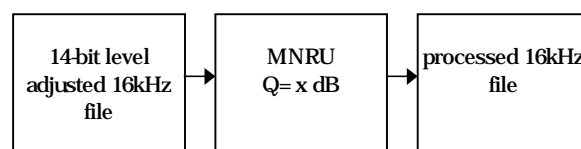
Figure 7: Processing of speech by wideband MNRU.

```
┌──────────────┐              ┌──────────────┐
│ 14-bit level │              │ processed    │
│adjusted 16kHz│─────────────▶│ 16kHz        │
│ file         │              │ file         │
└──────────────┘              └──────────────┘
```

Figure 8: Processing of speech for 'direct' condition.

3.3   Narrowband Experiments

3.3.1   Pre-Processing

```
┌──────────────┐   ┌──────────────┐   ┌──────────────┐   ┌──────────────┐
│ 16kHz speech │   │  GSM send    │   │ P.56 Level   │   │  16kHz GSM   │
│ source file  │──▶│characteristic│──▶│adjustment to │──▶│filtered      │
│              │   │              │   │ -26dBov      │   │ speech file  │
└──────────────┘   └──────────────┘   └──────────────┘   └──────────────┘
```
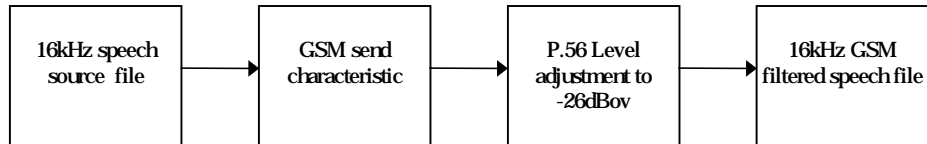
Figure 9: Pre-processing of GSM filtered speech-only file.
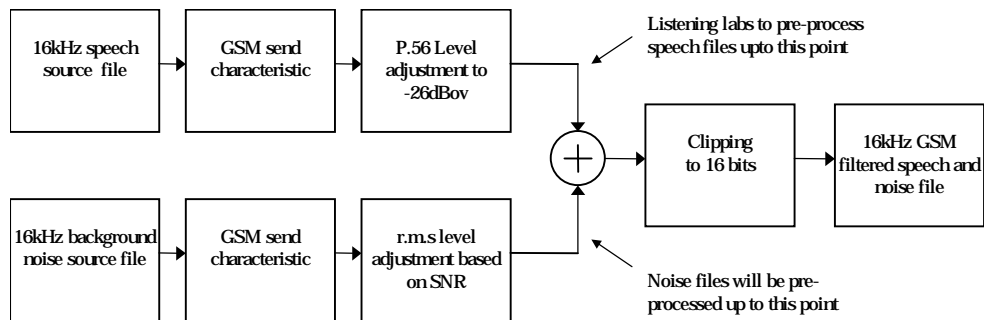
Figure 10: Pre-processing of GSM filtered speech and noise file.

3.3.2   Up and Down-Sampling, Rounding and Scaling
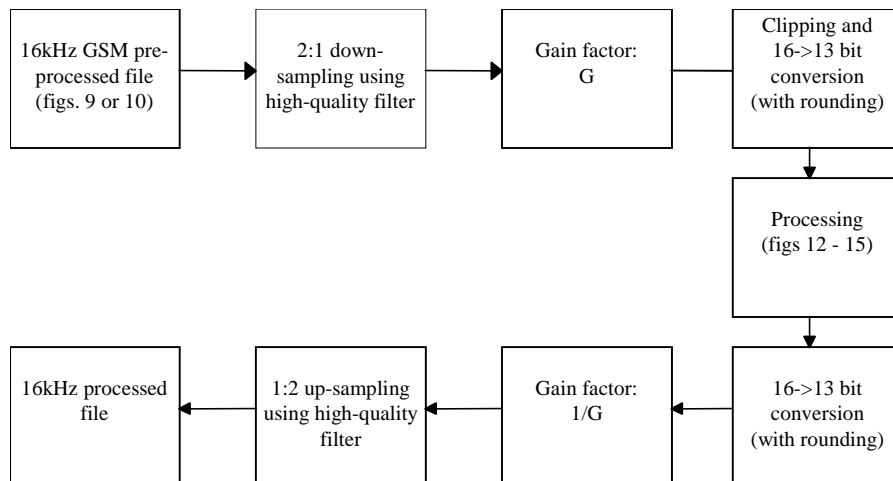
Figure 11: Sample-rate conversion, rounding and scaling for narrowband GSM filtered conditions.

3.3.3   Processing Stages
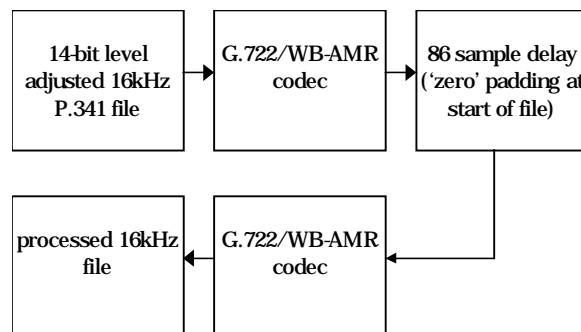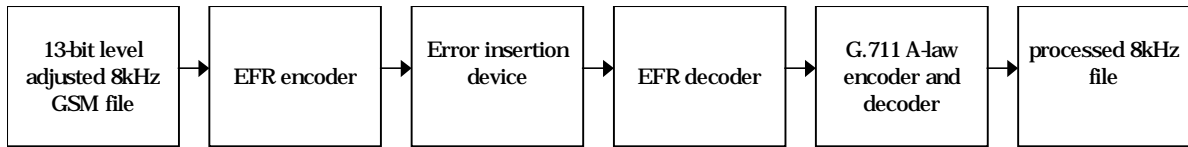
Figure 12: Processing of EFR conditions.
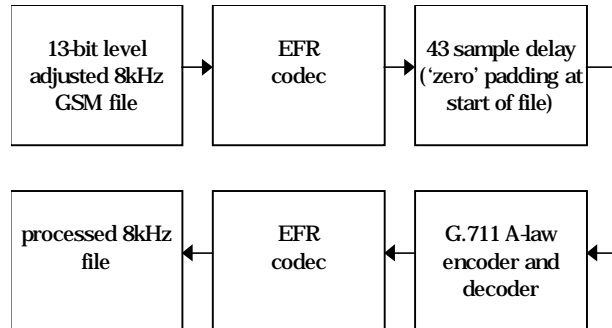
Figure 13: Processing of simulated tandem connection using EFR codec.
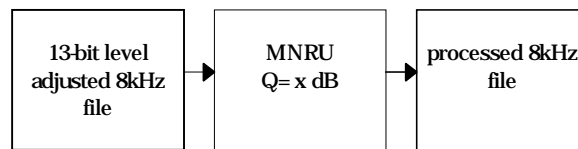
Figure 14: Processing of speech by narrow-band MNRU.

Figure 15: Processing of speech for 'direct' condition.

3.4 Post-processing
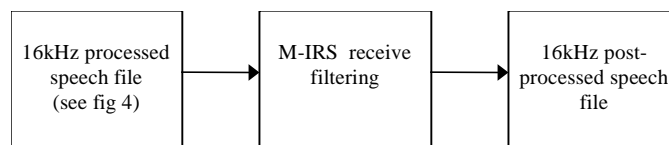
Figure 16: Post-processing of speech (all experiments).

3.5 Choice of AMR Mode

In Experiments 1a, 2a and 2b of the Qualification Phase the AMR mode will be selected by the candidate proponent within the constraints of the application.

4 Pre- and Post- Processing Modules

This section describes the modules that must be used in the pre- and post-processing of speech material.

4.1 Pre-Processing Operations

The filtering operations are performed using the *filter* program from the ITU-T STL:

    Source: ITU-T
    Location: STL sub-directory: fir

```
Format:       C source code
Program:      filter, sv56demo

Source:       [tbd]
Format:       [tbd]
Program:      add
```

4.1.1   P.341 filtering

To produce a P.341 filtered speech file use:

```
filter P341 input16 output16 320
```

4.1.2   GSM filtering

To produce a GSM filtered speech file use:

```
filter MSIN input16 output16 320
```

4.1.3   P.56 speech file level adjustment

To produce a normalise the P.56 level of a speech file to -26dBov use:

```
sv56demo -lev -26 -sf 16000 input16 output16 320
```

4.1.4   Noise file level adjustment

The procedure for scaling a noise file to obtain the correct SNR is described in Annex A (normative).

4.1.5   Summation of a speech and noise file

To produce a 16kHz mixed speech and noise file called *output* by summing a 16kHz speech file called *input16* and a 16kHz background noise file called *noise16,* use:

```
add input16 noise16 output16
```

4.2   Down-sampling, scaling and rounding

```
Source:       ITU-T
Location:     STL sub-directory: fir, utl
Format:       C source code
Program:      filter, scaldemo
```

The gain values to be used in these processing stages are specified in Table 1.

Table 1: Gain values for processing stages.

Speech level for condition	Value of x for scaldemo (dB)	
	Prior to processing stage	After processing stage
-16dBov	10	-10
-26dBov	0	0
-36dBov	-10	10

4.2.1   Wideband scaling and rounding (prior to processing stage)

To scale a file by x dB and round to 14-bit resolution use:

```
scaldemo -dB -gain x -bits 14 -round -nopremask -blk 320 input16 output16
```

4.2.2 Wideband rounding and scaling (after processing stage)

To round a file to 14-bit resolution and scale it by *x* dB use:

```
scaldemo -lin -gain 1 -bits 14 -round -nopremask -blk 320 input16 temp
scaldemo -dB -gain x -bits 16 -round -nopremask -blk 320 temp output16
```

4.2.3 Narrowband down-sampling, scaling and rounding (prior to processing stage)

To down-sample a file using the high-quality filter characteristic, scale it by *x* dB and round to 13-bit resolution use:

```
filter -down HQ2 input16 temp 160
scaldemo -dB -gain x -bits 13 -round -nopremask -blk 160 temp output8
```

4.2.4 Narrowband rounding, scaling and up-sampling (after processing stage)

To round a file to 13-bit resolution, scale it by *x* dB, and up-sample it using the high-quality filter characteristic use:

```
scaldemo -lin -gain 1 -bits 13 -round -nopremask -blk 160 input8 temp1
scaldemo -dB -gain x -bits 16 -round -nopremask -blk 160 temp1 temp2
filter -up HQ2 temp2 output16 160
```

4.3 Post-processing

Source:	ITU-T
Location:	STL sub-directory: fir
Format:	C source code
Program:	*filter*

To apply the M-IRS receive filter characteristic to a processed 16kHz file use:

```
Filter RXIRS16 input16 output16 320
```

5   Codecs and Reference Processes

This section describes the speech and channel codecs that must be used in the preparation of speech material.

5.1   Enhanced Full-Rate GSM Speech and Channel Codec

Version 5.1 of the Enhanced Full-Rate (EFR) GSM speech codec is available from:

Source:	ETSI Server
Location:	/tech-org/smg/Document/smg11/phase2pl/0653-513/DISK
Format:	ANSI C-code
Programs:	*coder* and *decoder*

while the following program performs the EFR GSM channel encoding, error insertion, channel decoding and error concealment:

Source:	ETSI Server
Location:	/tech-org/smg/Document/smg11/ smg11_amr/SW_tool_library/EFR
Format:	PC command line executable
Programs:	*ed_iface*

Note that the input and output formats of *ed_iface* are different.

5.1.1 Error free processing

To process a speech file through the EFR GSM speech codec, use:

```
coder input8 bitstream dtx
ed_iface bitstream decoded
decoder decoded output8
```

where the *dtx* option has the following effect:

> *dtx* = dtx   enables DTX operation;
> *dtx* = nodtx   disables DTX operation;

bitstream and *decoded* are encoder and decoder EFR GSM format files, respectively.

5.1.2   Processing error patterns

To process a speech file through the EFR GSM speech codec inserting errors from the AEG format error file *pattern*, use:

```
coder input bitstream dtx
ed_iface bitstream decoded pattern -s x
decoder decoded output
```

where *-s x* is an option to skip **456*x*** bits in the error pattern. See Section 5.1.1 for definitions of *dtx, bitstream* and *decoded*.

5.2   G.711 A-law Codec

> Source: ITU-T
> Location: STL sub-directory: g711
> Format: C source code
> Program: *g711demo*

To process a speech file through the G.711 A-law encoder and decoder speech codec use:

```
g711demo A lili input8 output8 160
```

5.3   Modulated Noise Reference Unit (MNRU)

> Source: ITU-T
> Location: STL sub-directory: mnru
> Format: C source code
> Program: *mnrudemo*

5.3.1   Wideband experiments

To produce an MNRU processed file called *output* from a speech file called *input* use:

```
mnrudemo -Q x input16 output16 320
```

where *x* is the desired MNRU.

5.3.2   Narrowband experiments

To produce an MNRU processed file called *output* from a speech file called *input* use:

```
mnrudemo -Q x input8 output8 160
```

where *x* is the desired MNRU.

5.4   Tandeming delay

The insertion of delay in the tandem conditions will be accomplished by concatenating a file containing PCM silence with the subject file.

5.4.1 Wideband experiments

In UNIX and DOS this can be accomplished with the following command:

```
cat delay86 input16 > output16
```

delay86 is a PCM file containing 86 16-bit samples equal to zero.

5.4.2 Narrowband experiments

In UNIX and DOS this can be accomplished with the following command:

```
cat delay43 input8 > output8
```

delay43 is a PCM file containing 43 16-bit samples equal to zero.

6 Background Noise Files
[tba]

7 Error Insertion
[tba]

Annex A (normative) Noise scaling procedure

Source: ITU-T
Location: STL sub-directory: sv56
Format: C source code
Program: *filter, actlev, scaldemo*

Vehicle noise contains a large amount of energy at frequencies around 100Hz. Although audible to the human ear, this low frequency energy is perceived as being substantially quieter then signals in the telephone band (300 - 3400Hz). The P.341 filter characteristic pass-band extends from 50Hz to 7kHz, and therefore fails to remove this low frequency energy from vehicle noise signals.

If vehicle noise is scaled on the basis of the R.M.S. level after P.341 filtering, the perceived level of the scaled noise signal is very low. This is because the majority of the signal energy is present at frequencies where the ear is relatively insensitive. This finding can be easily verified using the O.41 psophometric weighting filter in the STL.

The following process scales the noise to a perceptually meaningful level by using the M-IRS filter[3] to remove perceptually unimportant frequencies from the SNR calculation:

Step 1:
The unfiltered 16kHz noise file *unfilt16* is filtered using the M-IRS send filter as follows:

```
filter -mod IRS16 unfilt16 mirs16 320
```

Step 2:
The scaling factor *F* (in dB) to be applied to the P.341 filtered noise signal is calculated as:

$$F = -(R + S + 26)dB \hspace{4cm} \text{Equation 1}$$

where *R* is the R.M.S. signal level of the M-IRS filtered noise signal expressed in dBov; *S* is the target SNR in dB. *R* is printed out as part of the output of the following command line argument:

```
actlev -sf 16000 -blk 320 mirs16
```

Step 3:
The noise can then be P.341 filtered and scaled to the desired level using:

```
filter P341 unfilt16 temp1 320
scaldemo -dB -gain F -bits 16 -round -nopremask -blk 320 temp1 noise16
```

where *F* is the value obtained in Step 2.

---

[3] Ideally, this should be a psophometric weighting filter, but the STL does not contain a version that operates at 16kHz.

Annex B (normative) Specification of candidate algorithm interface

This annex specifies the interface the candidate algorithm must comply with in order to facilitate and function correctly during the host-lab function of the selection test.

1. Command line call of the encoder
The command line call of the encoder must comply with the following:

```
encoder [-quiet] speech_file bitstream_file
```

where the speech_file contains the speech to be encoded and bitstream_file contains the WB-AMR bitstream produced by the encoder. The format of speech_file and bitstream_file is specified in item 4.

The -quiet option should disable any frame-by-frame screen output, however, the executable should always print the title and version number for log-file purposes.

2. Command line call of the decoder
The command line call of the decoder must comply with the following:

```
decoder [-quiet] bitstream_file speech_file
```

where the bit_stream file contains the 4 kbit/s bitstream to be decoded and the speech_file contains the speech synthesised by the decoder. The format of speech_file and bitstream_file is specified in item 4.

3. Platform for the encoder and decoder executables
The executables of the encoder and decoder must be compiled for the platform agreed to between the proponent and the host-lab. Executables for UNIX platforms should be compiled with statically linked libraries.

4. Format of input and output for the encoder and decoder
- The format of the speech_file, i.e. input to encoder and output from decoder, must be 16-bit linear PCM in native byte order.
- The format of the bitstream_file, i.e. output from encoder and input to decoder, must comply with G.192 format (see Figure 1 taken from the ITU-T Software Tool Library User's Manual).
- Bad frames will be marked by setting the frame synchronisation word to 0x6B20 and the payload bits to all zeroes.
- The speech codec bits must be ordered based on their sensitivity to bit errors with the most sensitive bits placed first in the frame.
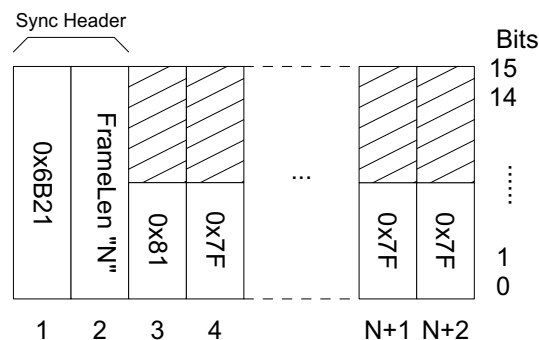

Figure 1:
Frame format for ITU-T 4-kbit/s candidate executables. A binary '0' in the payload is indicated by 0x007F; a binary '1' is indicated by 0x0081; the shaded areas represent zeros. N specifies the number of bits in the frame.

5. Frames size
If the WB-AMR speech codec has a frame size of 10ms, the bits from two successive speech frames should be packed into a single G.192 speech frame. If WB-AMR speech codec has a frame size of 5ms, the bits

from four successive speech frames should be packed into a single G.192 speech frame. The value of N in the G.192 header must be set to the number of bits in the composite G.192 frame (not the number of bits in a 5 or 10ms sub-frame). The decoder must apply the frame erasure flag in the G.192 header to each of the 5 or 10ms sub-frames.

The bits must be interleaved to maintain the correct bit sensitivity ordering. Thus for a 10ms frame size, the first bit in the G.192 payload will be the most sensitive bit of the first 10ms frame; the second bit the most sensitive bit of the second 10ms frame; the third bit the second most sensitive bit of the first 10ms frame; the fourth bit the second most sensitive bit of the second 10ms frame; etc.

6. Alignment of coding frame
The frame alignment in the encoder must be done in such a way that the first frame to be encoded is aligned with the first frame of the input speech signal. This procedure ensures that the occurrence of frame erasures is aligned identically with the input speech signal for all candidate algorithms.

Note that this procedure requires any look-ahead buffers in the encoder and decoder to be initialised, since the first frame of the input speech signal is copied to the frame to be encoded. For example, the encoder look-ahead buffer can be initialised with the future of the input speech signal.

Annex C (informative) AEG error pattern format

AEG format error patterns are organised as 512 byte records. The first 456 bytes of each record contain the soft decision values from four bursts; the remaining 56 bytes of each record are unused. The error patterns contain bursts from a single traffic channel. Therefore frames 12 and 25 (SACCH and idle frames) are not included, and the contents of a multiframe are held in three consecutive 512 byte records. AEG format error pattern files include a 512 byte header.

The AEG error patterns represent soft decision values as signed bytes. A negative soft decision value indicates that the chip was received in error; a positive soft decision value indicates that the chip was received correctly. The modulus of each soft decision value *sd* is equal to:

$$sd = 8\ln\left(\frac{1 - p_e}{p_e}\right)$$

where p_e is the probability that the chip was received in error.