README.TXT for AMR speech encoder and decoder

===============================================================================

GSM AMR-NB speech codec R98 Version 7.6.0 December 12, 2001

R99 Version 3.3.0 REL-4 Version 4.1.0

REL-5 version 5.3.0 REL-6 version 6.0.0 REL-7 version 7.0.0

REL-8 version 8.0.0 REL-9 version 9.0.0 REL-10 version 10.0.0 REL-11 version 11.0.0 REL-12 version 12.0.0 REL-13 version 13.0.0 REL-14 version 14.0.0 REL-15 version 15.0.0

Installation

------------

Assuming all delivered files are in the same directory, issuing "make"

or "make VAD=VAD1" on a Unix system compiles the source code and

builds two standalone programs called "encoder" and "decoder" which

use the ENS VAD option. To compile with the Motorola VAD, use

"make VAD=VAD2"

To compile with weighted MOPS count enabled use

"make [VAD=VAD#] MODE=WMOPS".

To verify the correct installation the files spch\_unx.inp,

spch\_unx.cod, and spch\_unx.out are provided for the ENS VAD option and

the files spch\_un2.cod and spch\_un2.out for the Motorola VAD option.

These files contain data in MSBbyte first order. The corresponding

files spch\_dos.\* and spch\_do2.\* contain the same data in LSByte first

order. A mode control file (allmodes.txt) is also supplied.

To verify the encoder, use

encoder -dtx -modefile=allmodes.txt spch\_unx.inp tmp.cod

or

encoder -dtx -modefile=allmodes.txt spch\_un2.inp tmp.cod

for the ENS or the Motorola VAD option, respectively, and compare the

output tmp.cod to spch\_unx.cod or spch\_un2.cod.

To verify the decoder, use

decoder spch\_un2.cod tmp.out

or

decoder spch\_unx.cod tmp.out

for the ENS or the Motorola VAD option, respectively, and compare the

output tmp.out to spch\_unx.out or spch\_un2.out.

On LSByte first systems, spch\_dos.\*/spch\_do2.\* must be used instead of

spch\_unx.\*/spch\_un2.\*

A shell script "amr\_chk.csh" is provided for Unix systems which

performs these tests automatically. The command line syntax is as

follows:

amr\_chk.csh [-vad2] unix

for MSB first systems (such as SUN workstations), or

amr\_chk.csh [-vad2] dos

for LSB first systems (such as Linux on a PC).

By default, it is assumed that the executable was compiled with

the ENS VAD option. To check the Motorola VAD version, the option

"-vad2" must be given on the command line.

Note that this verification procedure only performs a very basic

installation verification and is not a proof of overall bitexact

operation.

MMS compatible encoder output / decoder input

---------------------------------------------

The encoder output / decoder input can be optionally formatted according to the MIME file storage format used e.g. by the MMS service (referred to as MMS format from this point on). The specification of this format can be found in the RFC 3267 "Real-Time Transport Protocol (RTP) Payload Format and File Storage Format for the Adaptive Multi-Rate (AMR) and Adaptive Multi-Rate Wideband (AMR-WB) Audio Codecs", sections 5.1 and 5.3. The MMS format is turned on by defining "MMS\_IO" during compiling.

Note that using MMS format option does not have any effect on the speech encoding or decoding algorithms but only the format of the encoded bitstream is affected. However, the simple verification based on encoder output using the provided test samples is not possible when using this format.

Command line options for encoder program

----------------------------------------

The general syntax is

encoder [options] amr\_mode input\_file bitstream\_file

or

encoder [options] -modefile=mode\_file input\_file bitstream\_file

In the first case, amr\_mode must be one of MR475, MR515, MR59, MR67,

MR74, MR795, MR102, or MR122. In the second case, the text file

mode\_file must contain the mode names to be used (one line per

speech frame, each line containing one of the mode names mentioned

above). An example file iterating trhough all mode names is supplied

(allmodes.txt).

The following options are recognized:

-dtx enables DTX operation

Command line options for decoder program

----------------------------------------

The general syntax is

decoder [option] bitstream\_file output\_file

The following option is recognized:

-rxframetype expect RX frame type instead of TX frame type in

input file. This only useful for simulations with

other components between encoder and decoder.

NOTE that the option introduced above is not needed and therefore not available if the code is compiled with definition "MMS\_IO" to enable MMS format in encoder output / decoder input.

File formats

------------

The encoder input and decoder output files contain are 8 kHz, 16

bit/sample data. The output file length is always a multiple of 320

bytes because of the frame length of 160 samples. The encoder will

only process whole input frames, i.e. for input files whose length

is not a multiple of 160 samples the last <size> - n\*160 samples are

not processed.

The encoder output/decoder input files ("bitstream files") contain

frames which are structured as follows:

1 word frame type

244 words encoded speech parameter bitstream (one bit per word,

each word contains either 0x0001 or 0x0000), unused

bits written as 0x0000 for modes < MR122

1 word mode information

4 words unused (written as 0x0000 by encoder)

The frame type written by the encoder (TX frame type) can have one

of the following values:

0x0000 TX\_SPEECH\_GOOD (normal speech frame)

0x0001 TX\_SID\_FIRST (first SID frame after speech)

0x0002 TX\_SID\_UPDATE (SID update frame in DTX)

0x0003 TX\_NO\_DATA (empty frame in DTX)

If invoked without the -dtx option, the encoder only produces type

0x0000 (TX\_SPEECH) frames.

The decoder normally accepts the output format and frame type

produced by the encoder. However, using the option -rxframetype, it

can be forced to expect frames with RX frame type instead of TX

frame type. In that case, it can NOT directly read files produced by

the encoder because the frame types do not match. The RX frame type

coding is as shown below:

0x0000 RX\_SPEECH\_GOOD (normal speech frame)

0x0001 RX\_SPEECH\_DEGRADED (degraded speech frame)

0x0002 RX\_ONSET (ONSET frame)

0x0003 RX\_SPEECH\_BAD (bad speech frame)

0x0004 RX\_SID\_FIRST (first SID frame after speech)

0x0005 RX\_SID\_UPDATE (SID update frame in DTX)

0x0006 RX\_SID\_BAD (corrupted SID frame)

0x0007 RX\_NO\_DATA (no data received)

The coding of the mode information field in the bitstream files is

as follows:

0x0000 MR475 4.75 kbit/s

0x0001 MR515 5.15 kbit/s

0x0002 MR59 5.90 kbit/s

0x0003 MR67 6.70 kbit/s

0x0004 MR74 7.40 kbit/s

0x0005 MR795 7.95 kbit/s

0x0006 MR102 10.20 kbit/s

0x0007 MR122 12.20 kbit/s

Motorola VAD

------------

Summary: Version 2.0.1 incorporates the Voice Activity Detector option 2.

Either option 1 or option 2 can be compiled via the respective

command "make VAD=VAD1" or "make VAD=VAD2". This package has

been verified to be backward compatible with version 2.0.0. That

is, version 2.0.0 encoder is bit-exact to version 2.0.1 when

compiled with the VAD=VAD1 argument. It has also been verified

that version 2.0.1 compiled with the VAD=VAD1 argument is

bit-exact to version 2.0.1 compiled with the VAD=VAD2 argument

when run in non-DTX mode. Only the vad\_flag output of the vad2()

function will affect the subsequent state of the encoder process.

No other outputs are generated, nor are any encoder state

variables directly modified by the use of the VAD2 option.

The co-existence of the two VAD options is achieved through the

use of conditional compiler directives. That is, for every

instance where the existing code has been modified, there is

a corresponding "#ifdef VAD2" where code has been added, and a

corresponding "#ifndef VAD2" where code related to the VAD1

option is to be omitted. It should be noted that the conditional

compile was purposefully triggered from a single define to avoid,

for example, the cases where both (or neither) a VAD1 and/or VAD2

compiler directives were defined.

Furthermore, the integration impact of the VAD2 option was kept

to a minimum by allowing the typedef for the vadState structure

to be dependent on the compiler directive. This was accomplished

by "typedef-ing" vadState1 for VAD1, and vadState2 for VAD2, and

then further defining the typedef to be used as:

#ifndef VAD2

#define vadState vadState1

#else

#define vadState vadState2

#endif

The result is that very few modules are affected by the change

because the functions passing the state pointers do not care

how the structure is defined. Only the low level routines which

use the specific elements of the state structures need to care

about the elements of the vadState structure.

C code changes since 7.3.0

--------------------------

One change request on the AMR C-code reference was approved by the joint

SMG11/TSG-SA4 meeting #14 / #9 in Puerto Vallarta, Mexico, January 24-28 2000.

CR ID:

A021

Subject of change:

Avoidance of pulse cancellation in FCB excitation

Files affected:

C code files c2\_11pf.c, d2\_11pf.c

For details of the changes, see the CR.

C code changes since 7.4.0

--------------------------

CR ID:

A023

Subject of change:

Correction of potential bug in AMR decoder due to the usage of standard

C abs() function.

File affected:

dtx\_dec.c

For details of the changes, see the CR.

CR ID:

A024

Subject of change:

Correction of comfort noise parameter interpolation bug of AMR decoder

File affected:

dtx\_dec.c

For details of the changes, see the CR

CR ID:

A025

Subject of change:

Correction of mode state bug in AMR decoder

File affected:

decoder.c, sp\_dec.c, sp\_dec.h, coder.c

For details of the changes, see the CR

CR ID:

A026 rev 1

Subject of change:

Correction of the TX\_Type and RX\_Type identifiers

File affected:

frame.h, sid\_sync.c, decoder.c, dec\_amr.c, dtx\_dec.c, strfunc.c

For details of the changes, see the CR

CR ID:

A027

Subject of change:

Correction of potential bug in AMR due to the usage of standard

C abs() function (VAD2).

File affected:

vad2.c

For details of the changes, see the CR

C code changes since 7.5.0

--------------------------

CR ID:

A028

Subject of change:

Correction of RX-DTX handling of NO\_DATA frames in AMR decoder

File affected:

dtx\_dec.c

For details of the changes, see the CR.

CR ID:

A029

Subject of change:

Correction in AMR decoder to avoid division by zero in RX-DTX Handling

File affected:

dtx\_dec.c

For details of the changes, see the CR.

$Id $