Technical Specification Group Services and System Aspects
Meeting #23, Phoenix, Arizona (USA)
15-18 March 2004

TSGS#23(04)0071

Source: **TSG-SA WG4**

Title: Audio codec selection tests: Reports from "Host" and "Selection of items" Laboratories

Document for: **Approval**

Agenda Item: **7.4.3**

The following documents, agreed at the TSG-SA WG4 meeting #30, are presented to TSG SA #23 for approval.

S4-040022	Report on selection of items	France Telecom
S4-040026	Mirror Laboratory report on 3GPP Audio Experiment	ARL
S4-040036	PSS/MMS High Rate and AMR-WB+ and PSS/MMS Low Rate Audio Selection Test, Host Laboratory Report	T-Systems

Technical Specification Group Services and System Aspects **TSGS#23(04)0071**
Meeting #23, Phoenix, Arizona (USA)
15-18 March 2004

TSG-SA4#30 meeting Tdoc S4 (04)0022
February 23-27, 2004, Malaga, Spain

Source: TSG SA WG4 (France Telecom)

Title: Report on selection of items

Document for: Approval

Agenda Item: 7.4.3

---

1 Introduction

This document reports on item selection conducted by France Telecom for the PSS/MMS High-Rate Audio Selection test.
Files candidates for the PSS/MMS High-Rate Audio Selection test have been processed by host laboratory and the whole set of processed material corresponding to experiment 1, 2 and 3 have been received on November 28th.

2 Item selection

The whole set of candidate files comprises 105 files. Host laboratory informed selection entity that some of them have to be discarded due to strange behaviour such as clipping errors during the processing.

The received set was composed of music and speech items.
Classical music, pop music single instrument and vocal are classified under Music items and represent 56 items.
Speech, speech over music, speech with background noise is classified under speech items and represents 49 items.
The result of selection was 12 files:

- 7 music items (2 classical, 2 pop, 2 single instruments and 1 vocal)
- 5 speech items (2 speech over music, 2 speech with background noise and 1 speech)

Moreover 4 other files were selected to be used as training items (1 speech item, 1 classical, 1 pop, 1 single instrument)

3 Processing

Nearly all the items were processed in order to fit to the recommended lengths that were 5 seconds for speech items and 10 seconds for the others.
The processing involved as well some dynamic alignment in order to get more or less the same restitution sound level for all the items, and fade in and fade out at the beginning and end of the items.
The sampling frequency was checked as well as it had to be 48 Khz.

4 Conclusion

The selected items have been sent back to the host laboratory on December 3rd.

Technical Specification Group Services and System Aspects **TSGS#23(04)0071**
Meeting #23, Phoenix, Arizona (USA)
15-18 March 2004

TSG-SA4#30 meeting Tdoc S4 (04)0026
February 23-27, 2004, Malaga, Spain

Source: **TSG SA WG4 (Audio Research Labs)**
Title: **Mirror Lab Report on 3GPP Audio Codec Exercises**
Agenda Item: **7.4.3**
Document for: **Approval**

1 Introduction

3GPP has undertaken to test candidate codecs for their needs, as set forth in documents S4-030824 (Low-Rate Test) and S4-030821 (High-Rate Test). In order to insure that the processing, such as preparation of anchors and coding of candidate and reference codecs, was done correctly, 3GPP used two laboratories, the Host Laboratory and the Mirror Laboratory, to do duplicate processing. This is the report of the Mirror Laboratory.
In the remainder of this document, Sections 2 and 3 report on the processing done by the Mirror Lab, Annex 1 lists the training and test items used in the Low-Rate test, Annex 2 lists the scripts used for the processing and Annex 0 lists the cksum output for the Host Lab and Mirror Lab for each concatenated file used in the tests.

2 Low-Rate Test

The procedures for the Low-Rate test are specified in document S4-030824.
The tests are:

Table 2-1: Sub-experiments of experimental block A

Exp.	Operational mode	Audio Material	#Codecs under test	# reference codecs	#Anchors in test	#References	#items	Total
A1a	14 kbps, mono, use case A (PSS)	Set a	3	2	2	2	12	108
A1b		Set b						
A2a	18 kbps, stereo, use case A (PSS)	Set a	3	2	3	2	12	120
A2b		Set b						
A3a	24 kbps, mono, use case A (PSS)	Set a	3	2	2	2	12	108
A3b		Set b						
A4a	24 kbps, stereo, use case A (PSS)	Set a	3	2	3	2	12	120
A4b		Set b						

Table 2-2: Sub-experiments of experimental block B

Exp.	Operational condition	Audio Material	#Codecs under test	# reference codecs	#Anchors in test	#References	#items	Total
B1a	14 kbps, mono, use case B (MMS), 16 kHz input and output sampling rate	Set a	3	2	2	2	12	108
B1b		Set b						
B2a	18 kbps, stereo, use case B (MMS)	Set a	3	2	3	2	12	120
B2b		Set b						
B3a	14 kbps, mono, use case A (PSS), 3% FER	Set a	3	2	2	2	12	1848
B3b		Set b						
B4a	24 kbps, stereo, use case A (PSS), 3% FER	Set a	3	2	3	2	12	1860
B4b		Set b						

Note that the increased number of items in test 3 accounts for the fact that each listener and each listening lab receives a unique decoding of the test item due to its being processed with a distinct circular shift of the error pattern file.

2.1 Test Material

2.1.1 Signal categories

The test material is selected so as to be representative of the following four signal categories:

- Music
- Speech
- Speech over music (i.e. speech with background music)
- Speech between music (i.e. alternating speech and music segments)

2.1.2 Training Items

A single set of four training items are used for the eight tests, one item selected from each of the four stimulus categories. The four training items are shown in Annex 1.

2.1.3 Test Items

Eight sets of test items were used, one for each experiment. The four signal categories were represented within each set, specifically with four Music items, four Speech items, two Speech between Music items and two Speech over Music items. Due to limitations in the availability of test material, there some individual items appear in more than one set. The eight sets are shown in Annex 1.

2.1.4 Processing

All test material used in a given experiment was concatenated into a single file for processing. This included test material for both sub-experiments (a and b). The order of items in the concatenated file was:

- 4 Training items - order as shown in Annex 1, which is alphabetical by filename.
- 24 Test items - first the 12 for set a followed by the 12 for set b. Within each set, the order is by signal category (first Music, then Speech, Speech over music and finally Speech between music) and alphabetical by filename within each category. The order of files in each test is shown in Annex 1.

2.2 Processing Systems

The Mirror Lab has created scripts (that build upon scripts provided by 3GPP members) that automate all processing required in the Low-Rate Test, as specified in document S4-030694. Those scripts are shown in the ANNEX of this report. Portions of those scripts are reproduced below as a means to explain the processing done by the Mirror Lab.

In the scripts in this section, `EXP` (or `exp`) is a symbolic name for the experiment, and can be one of A1, A2, A3, A4, B1, B2, B3, or B4. In a similar way, `SET` (or `set`) is a symbolic name for the data set of the sub-experiment, and can be one of “a” or “b”.

2.2.1 Coding conditions under test

These are the candidate coding systems. In the discussion in this section they are given symbolic names that are identical to their directory names in the experiment file system. They are `aacPlus` `AMRWB+` and `ct`.

2.2.2 Reference coding conditions

In the discussion in this section, the reference coding systems are given symbolic names that are identical to their directory names in the experiment file system. They are AAC and AMR-WB.

2.2.3 Anchor conditions

The following anchors are created. This is discussed in more detail in the Section 2.7.2

```
opref          open reference
hidref         hidden reference
lp3500         3.5 kHz lowpassed signal
lp7000         7 kHz lowpassed signal
lp3500_s12     3.5 kHz lowpassed signal with narrowed stereo soundstage
lp7000_s12     3.5 kHz lowpassed signal with narrowed stereo soundstage
lp7000_s6      7 kHz lowpassed signal with narrowed stereo soundstage
```

Since all concatenated files have $EXP as part of their filename, this serves to disambiguate the stereo anchors from the mono anchors.

2.3 Directory Structure

Shown below is the directory structure (as listed by du) used in the Low-Rate experiment.

```
./AAC/bin
./AAC/processed
./AAC/tmp
./AAC              #reference codec
./aacPlus/bin
./aacPlus/processed
./aacPlus/tmp
./aacPlus          #proponent codec
./AMR-WB/bin
./AMR-WB/c-code
./AMR-WB/processed
./AMR-WB/testv
./AMR-WB/tmp
./AMR-WB           #reference codec
./AMRWB+/bin
./AMRWB+/processed
./AMRWB+/tmp
./AMRWB+           #proponent codec
./bin              #all scripts and programs (excepts codec-specific)
./ct/bin
./ct/processed
./ct/tmp
./ct               #proponent codec
./ep
./org/ClippedSpeech#results of clipping "Speech" dir signals
./org/Music
./org/Speech
./org/Speech between Music
./org/Speech over Music
./org              #low-rate test original signalsp
./output/exp_A1a   #low-Rate experiment "blinded" files
./output/exp_A1b
./output/exp_A2a
./output/exp_A2b
./output/exp_A3a
./output/exp_A3b
./output/exp_A4a
./output/exp_A4b
./output/exp_B1a
./output/exp_B1b
./output/exp_B2a
./output/exp_B2b
./output/exp_B3a
./output/exp_B3b
./output/exp_B4a
./output/exp_B4b
./output
```

```
./preproc          #low-rate test concatenated signal files
./ref/processed
./ref/tmp
./ref              #anchors
```

2.4 Processing procedure

2.5 Generation of error files

The seed was distributed by the SA4 secretary, and is show below along with the script fragment to generate the frame error rate pattern file.

```
SEED=171094
./genfer.csh $SEED .03; cp ferpat_FER.03_SEED$SEED.dat ../ep/epf_3.epf
```

2.5.1 Pre-processing

The following command was used to clip silence segments from the beginning and end of the speech signals, with results put in the `ClippedSpeech` directory.

```
./clip_silence.sh ../org/ClippedSpeech ../org/Speech/*.wav
```

In order to avoid any pathnames with embedded spaced, all signals were copied to the root of the ../org directory using the following script.

```
l_rename.sh
```

2.6 Concatenation of material

The original signals were concatenated with an additional 290ms of silence added at the end. Since the encoder/decoder chain typically has an inherent delay, or processing state, this insures that all of the signal material is present in the processed output file. The script `l_files.sh` insured that all files were in the correct order.

```
LIST=`l_files.sh $EXP`
./concat.sh $LIST
```

Make 24 kHz mono and stereo versions of the concatenated file, which are needed by the AMR-WB codec.

```
./anchor.sh -monoout -fsout24 ${CAT}48s.wav ${CAT}24m.wav
./anchor.sh -fsout24 ${CAT}48s.wav ${CAT}24s.wav
```

Make 16, and 48 kHz mono versions of the concatenated file.

```
./anchor.sh -monoout -fsout16 ${CAT}48s.wav ${CAT}16m.wav
./anchor.sh -monoout ${CAT}48s.wav ${CAT}48m.wav
```

Remove silence file from `all_cat48s.tim` file so that subsequent split-up does not extract the silence file.

```
mv ../preproc/all_cat48s.tim tmp
grep -v silence.wav tmp > ../preproc/all_cat48s.tim
```

2.7 Main processing

2.7.1 Coding of material

In the `bin` directory of each proponent and reference codec is the `l_encdecdly.sh` script which does all the codec-specific processing (i.e. encode, decode and delay compensation). Arguments `exp` and `set` are experiment (A1, A2, …) and set (a, b) and are used to set operating modes (e.g. bitrate and delay compensation), name output files and enable errored-channel processing. Upsampling of the concatenated and processed output file is done at this time instead of later in the processing chain.

```
for c in aacPlus AMRWB+ ct AAC AMR-WB
do
    (cd ../$c/bin; ./l_encdecdly.sh $exp $set)
    echo
    echo upsample
    echo
```

```
    for i in ../$c/tmp/*_${exp}.wav
    do
        base=${i##*/}
        ./upsamp48.sh $i ../$c/processed/$base
    done
done
```

2.7.2 Creation of anchors

In the following script fragment, argument EXP is experiment (A1, A2, ...) and is used to set parameters in creation of anchors, via calls to `anchor.sh`, and to name output files. In the case that the anchors are identical to the concatenated file (e.g. opref and hidref), a simple copy is used to create the anchor.

```
ANCHOR=./anchor.sh
CAT=../preproc/all_cat48s.wav
OUT=../ref/tmp

case $EXP in
    A1|A3|B1|B3)
    echo
    echo make mono anchors for test $EXP
    echo
    $ANCHOR -monoout $CAT $OUT/opref_${EXP}.wav
    cp $OUT/opref_${EXP}.wav $OUT/hidref_${EXP}.wav
    $ANCHOR -lp3500 -monoout $CAT $OUT/lp3500_${EXP}.wav
    $ANCHOR -lp7000 -monoout $CAT $OUT/lp7000_${EXP}.wav
        ;;

        A2|A4|B2|B4)
        echo
    echo make stereo anchors for test $EXP
    echo
    cp $CAT $OUT/opref_${EXP}.wav
    cp $CAT $OUT/hidref_${EXP}.wav
    $ANCHOR -lp3500 -s12 $CAT $OUT/lp3500_s12_${EXP}.wav
    $ANCHOR -lp3500 -s12 $CAT $OUT/lp3500_s12_${EXP}.wav
    $ANCHOR -lp7000 -s6 $CAT $OUT/lp7000_s6_${EXP}.wav
    ;;
esac
```

Upsampling of the concatenated and processed output file is done at this time instead of later in the processing chain.

```
for i in ../ref/tmp/*_${exp}.wav
do
    base=${i##*/}
    ./upsamp48.sh $i ../ref/processed/$base
done
```

2.7.3 Impaired channel processing

Below is a fragment of the l_encdecdly.sh script that shows how the FER pattern is shifted for each listener and listening laboratory prior to decoding. It also shows the delay compensation.

```
PRO=../tmp/<codec>
BIT=../tmp/${EXP}.3gp
CPY=../../bin/CopyAudio.exe


for i in 0 1
do
    for lis in 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15
    do
        echo "lab ${lab[$i]} lis $lis"
        OFILE="${PRO}_lab${lab[$i]}_lis${lis}_${EXP}.wav"
        ../../bin/shiftcirc.csh $lis ${lab[$i]} $exp \
            ../../ep/epf_3.epf tmp_fer_file
        $DEC $BIT tmp.wav tmp_fer_file
        $CPY -l ${dly}: tmp.wav $OFILE
    done
done
```

2.8 Post-processing

2.8.1 Up-sampling

Up-sampling is done as part of material coding and creation of anchors.

2.8.2 Split-up of processed material

This processing was not done by the Mirror Laboratory, although the suite of scripts provided by the Mirror Laboratory enabled the split-up. Clearly, this is required to obtain the files that are provided to the listening sites. Note that since all files are stereo at 48 kHz sampling rate, only one timing file, `all_cat48s.tim`, is required.

```
for exp in $EXPS
do
    for set in a b
    do
        SOUT=../output/exp_${EXP}${SET}

        TIM=../preproc/all_cat48s.tim

        for dir in aacPlus AMRWB+ ct AAC AMR-WB ref
        do
            for file in ../$dir/processed/*_${EXP}.wav
            do
                base=${file##*/}
                tag=${base%.wav}
                ./splitup.sh $file $SOUT/${tag} $TIM
            done
        done
    done
done
```

2.9 Report generation

2.9.1 Cksum

The most important role of the Mirror Lab is to cross-check the processing of the Host Lab. The cksum utility, which calculates a CRC on a file, was used to map the contents of a signal file to a unique CRC checksum sequence. Note that since WAVE file headers contain file creation date information, these headers had to be stripped prior to using cksum.

```
echo
echo run cksum on concatenated org files
echo
./Copyaudio.exe ../preproc/all_cat48s.wav temp.raw
ckout=`cksum temp.raw`
echo $ckout | sed "{s'temp.raw'$i'}"

for i in aacPlus AMRWB+ ct AAC AMR-WB ref
do
    echo
    echo cksum on concatenated and upsampled files in $i
    echo
    ./Copyaudio.exe $i temp.raw
    ckout=`cksum temp.raw`
    echo $ckout | sed "{s'temp.raw'$i'}"
done
```

2.9.2 Delay

The coder-specifc script `l_encdecdly.sh` compensated for the processing delay of each codec at the various operating parameters of the tests. The Mirror Lab provided a script that checked the delay of all processed concatenated files with respect to the concatenated original file. This used the utility `getdelay.exe`, which was kindly provided by FhG.

```
CK=../AAC/bin/getdelay.exe
```

```
for i in aacPlus AMRWB+ ct AAC AMR-WB ref
do
    echo
    echo $i
    echo
    for j in ../$i/processed/*[AB][1234].wav
    do
        delay=`$CK -if1 ../preproc/all_cat48s.wav  -if2 $j | grep delay`
        echo $j $delay
    done
done
```

3 High-Rate Test

The procedures for the High-Rate test are specified in document S4-030821.
The tests are:

Table 3-1: Experiments for high-rate codec selection

Exp.	Operational mode	#Codecs in test	# reference codecs	#Anchors in test	#References	#items	Total
1	32 kbps, stereo	2(use case B encoder)	2, incl. RealAudio @ 32 kbit/s stereo	2	2	12	96
2	48 kbps, stereo	2(use case B encoder)	2, incl. RealAudio @ 48 kbit/s stereo	2	2	12	96
3	32 kbps, stereo, 1% and 3% random frame loss	4 (2 candidates at 2 frame loss rates each)	2 (AAC-LC at 2 frame loss rates)	2	2	12	2208

Note that the increased number of items in test 3 accounts for the fact that each listener and each listening lab receives a unique decoding of the test item due to its being processed with a distinct circular shift of the error pattern file.

3.1 Test Material

3.1.1 Training Items

A single set of four training items are used for the three tests. They are:

```
c_09_org.wav
p_09_org.wav
si_09_org.wav
sp_09_org.wav
```

3.1.2 Test Items

A single set of twelve test items were used for the three experiments. They are:

```
c_01_org.wav
c_02_org.wav
p_01_org.wav
p_02_org.wav
si_01_org.wav
si_02_org.wav
sm_01_org.wav
sm_02_org.wav
sp_01_org.wav
sp_02_org.wav
sp_03_org.wav
v_01_org.wav
```

3.1.3 Processing

All test material used in a the experiments was concatenated into a single file for processing. This concatenated file consisted of:

- 4 Training items in alphabetical order by filename.
- 12 Test items in alphabetical order by filename

3.2 Processing Systems

In the scripts in this section, EXP (or exp) is a symbolic name for the experiment, and can be one of H1, H2 or H3.

3.2.1 Coding conditions under test

These are the candidate coding systems. In the discussion in this section they are given symbolic names that are identical to their directory names in the experiment file system. They are aacPlus and ct.

3.2.2 Reference coding conditions

In the discussion in this section, the reference coding systems are given symbolic names that are identical to their directory names in the experiment file system. They are AAC and RN.

3.2.3 Anchor conditions

The following anchors are created. This is discussed in more detail in the Section 3.7.2

```
opref          open reference
hidref         hidden reference
lp3500         3.5 kHz lowpassed signal
lp7000         7 kHz lowpassed signal
```

3.3 Directory Structure

Shown below is the directory structure (as listed by du) used in the High-Rate experiment. Note that the same directory tree is used for both the Low-Rate and High-Rate experiments, except that the High-Rate uses distinct directories for original signals (../h_org) and preprocessed signals (../h_preproc) and that different candidate and reference coders are used in the Low- and High-Rate experiments.

```
./AAC/bin
./AAC/processed
./AAC/tmp
./AAC              #reference codec
./aacPlus/bin
./aacPlus/processed
./aacPlus/tmp
./aacPlus          #proponent codec
./bin              #all scripts and programs (other than codecs)
./ct/bin
./ct/processed
./ct/tmp
./ct               #proponent codec
./ep
./h_org            #high-rate test original signals
./output/exp_1     #high-Rate experiment "blinded" files
./output/exp_2
./output/exp_3
./output
./h_preproc        #high-rate test concatenated signal files
./ref/processed
./ref/tmp
./ref              #anchors
./RN/bin
./RN/processed
./RN               #reference codec
```

3.4 Processing procedure

3.5 Generation of error files

Whereas only the 3% FER condition was used in the Low-Rate experiment, both a 1% and a 3% FER condition was used in the High-Rate test.

```
SEED=171094
./genfer.csh $SEED .01; cp ferpat_FER.01_SEED$SEED.dat ../ep/epf_1.epf
./genfer.csh $SEED .03; cp ferpat_FER.03_SEED$SEED.dat ../ep/epf_3.epf
```

3.5.1 Pre-processing

No pre-processing was done on the original signals.

3.6 Concatenation of material

The original signals were concatenated with an additional 290ms of silence added at the end. Since the encoder/decoder chain typically has an inherent delay, or processing state, this insures that all of the signal material is present in the processed output file

```
./concat.sh $ORG/Training/*.wav $ORG/Test_Items/*.wav ../org/Silence/silence.wav

mv ../preproc/all_cat48s.wav $PRE/all_cat48s.wav
mv ../preproc/all_cat48s.tim $PRE/all_cat48s.tim
```

Make 22.05 kHz and 44.1 kHz stereo versions of the concatenated file, which are needed by the RealNetworks codec.

```
./anchor.sh -fsout22 ${CAT}48s.wav ${CAT}22s.wav
./anchor.sh -fsout44 ${CAT}48s.wav ${CAT}44s.wav
```

Remove silence file from `all_cat48s.tim` file so that subsequent split-up does not extract the silence file.

```
mv $PRE/all_cat48s.tim tmp
grep -v silence.wav tmp > $PRE/all_cat48s.tim
```

3.7 Main processing

3.7.1 Coding of material

In the `bin` directory of each proponent and reference codec is the `l_encdecdly.sh` script which does all the codec-specific processing (i.e. encode, decode and delay compensation). Argument `exp` are experiment (H1, H2, ...) and are used to set operating modes (e.g. bitrate and delay compensation), name output files and enable errored-channel processing. Upsampling of the concatenated and processed output file is done at this time instead of later in the processing chain.

```
for c in aacPlus ct AAC RN
do
    (cd ../$c/bin; ./h_encdecdly.sh $exp $set)
    ls  ../$c/tmp

    for i in ../$c/tmp/*_${exp}.wav
    do
        base=${i##*/}
        ./upsamp48.sh $i ../$c/processed/$base
    done
done
```

3.7.2 Creation of anchors

In the following script fragment, agument EXP is experiment (H1, H2, …) and is used to set parameters in creation of anchors and to name output files. In the case that the anchors are identical to the concatenated file (e.g. opref and hidref), a simple copy is used to create the anchor. Note that, for the High-Rate test, all concatenated signals and the timing file are stored in

directory h_preproc, and that anchors are created once and copied to new names (e.g. per experiment) as required.

```
ANCHOR=./anchor.sh
CAT=../h_preproc/all_cat48s.wav
OUT=../ref/tmp

case $EXP in
    H1)
    echo
    echo make stereo anchors for test $EXP
    echo
    cp $CAT $OUT/opref_${EXP}.wav
    cp $CAT $OUT/hidref_${EXP}.wav
    $ANCHOR -lp3500  $CAT $OUT/lp3500_${EXP}.wav
    $ANCHOR -lp7000  $CAT $OUT/lp7000_${EXP}.wav
    ;;

    H2|H3)
    echo
    echo duplicate anchors for test $EXP
    echo
    for i in opref hidref lp35000 lp7000
    do
        cp $OUT/${i}_H1.wav $OUT/${i}_${EXP}.wav
    done
    ;;

esac
```

Upsampling of the concatenated and processed output file is done at this time instead of later in the processing chain

```
for i in ../ref/tmp/*_${exp}.wav
do
    base=${i##*/}
    ./upsamp48.sh $i ../ref/processed/$base
done
```

3.7.3 Impaired channel processing

Below is a fragment of the h_encdecdly.sh script that shows how the FER pattern is shifted for each listener and listening laboratory prior to decoding. It also shows the delay compensation.

```
PRO=../tmp/<codec>
BIT=../tmp/${EXP}.3gp
CPY=../../bin/CopyAudio.exe


for lab in 3 6
    do
        for fer in 1 3
        do
            for lis in 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15
            do
                echo "lab $lab lis $lis"
                OFILE="${PRO}_FER${fer}_lab${lab}_lis${lis}_${EXP}.wav"
                ../../bin/shiftcirc.csh $lis $lab 0 \
                    ../../ep/epf_${fer}.epf tmp_fer_file
                $DEC $BIT tmp.wav tmp_fer_file
                $CPY -l ${dly}: tmp.wav $OFILE
            done
        done
    done
```

3.8 Post-processing

3.8.1 Up-sampling

Up-sampling is done as part of material coding and creation of anchors.

3.8.2 Split-up of processed material

This processing was not done by the Mirror Laboratory, although the suite of scripts provided by the Mirror Laboratory enabled the split-up. Clearly, this is required to obtain the files that are provided to the listening sites. Note that since all files are stereo at 48 kHz sampling rate, only one timing file, `all_cat48s.tim`, is required.

```
for exp in H1 H2 H3
do
    SOUT=../output/exp_${EXP}
    TIM=../h_preproc/all_cat48s.tim

    for dir in aacPlus ct AAC RN ref
    do
        for file in ../$dir/processed/*_${EXP}.wav
        do
            base=${file##*/}
            tag=${base%.wav}
            ./splitup.sh $file $SOUT/${tag} $TIM"
        done
    done
done
```

3.9 Report generation

3.9.1 Cksum

The most important role of the Mirror Lab is to cross-check the processing of the Host Lab. The cksum utility, which calculates a CRC on a file, was used to map the contents of a signal file to a unique CRC checksum sequence. Note that since WAVE file headers contain file creation date information, these headers had to be stripped prior to using cksum.

```
echo
echo run cksum on concatenated org files
echo
./Copyaudio.exe ../h_preproc/all_cat48s.wav temp.raw
ckout=`cksum temp.raw`
echo $ckout | sed "{s'temp.raw'$i'}"

for i in aacPlus ct AAC RN ref
do
    echo
    echo cksum on concatenated and upsampled files in $i
    echo
    ./Copyaudio.exe ../${i}/processed/*.wav temp.raw
    ckout=`cksum temp.raw`
    echo $ckout | sed "{s'temp.raw'$i'}"
done
```

3.9.2 Delay

The coder-specifc script `h_encdecdly.sh` compensated for the processing delay of each codec at the various operating parameters of the tests. The Mirror Lab provided a script that checked the delay of all processed concatenated files with respect to the concatenated original file. This used the utility `getdelay.exe`, which was kindly provided by FhG.

```
CK=../AAC/bin/getdelay.exe
for i in aacPlus ct AAC RN ref
do
    for j in ../$i/processed/*H[123].wav
    do
        delay=`$CK -if1 ../preproc/all_cat48s.wav  -if2 $j | grep delay`
        echo $j $delay
    done
done
```

1 Annex - Low-Rate Experiment Training and Test Items

1.1 Training Items

m_vo_x_1_org.wav
s_no_ft_9_org.wav
sbm_fi_x_9_org.wav
som_ot_x_9_org.wav

1.2 Test Items

Test	Set	Item	Signal
A1	a	1	m_ot_x_8_org.wav
		2	m_ot_x_a_org.wav
		3	m_po_x_5_org.wav
		4	m_po_x_7_org.wav
		5	s_cl_2t_3_org.wav
		6	s_cl_2t_4_org.wav
		7	s_no_2t_1_org.wav
		8	s_no_ft_1_org.wav
		9	sbm_js_x_1_org.wav
		10	sbm_ms_x_1_org.wav
		11	som_fi_x_4_org.wav
		12	som_ot_x_4_org.wav
	b	1	m_ot_x_9_org.wav
		2	m_ot_x_b_org.wav
		3	m_po_x_6_org.wav
		4	m_si_x_3_org.wav
		5	s_cl_2t_5_org.wav
		6	s_cl_mt_2_org.wav
		7	s_no_2t_2_org.wav
		8	s_no_ft_2_org.wav
		9	sbm_sj_x_1_org.wav
		10	sbm_sm_x_6_org.wav
		11	som_ot_x_5_org.wav
		12	som_ot_x_6_org.wav
Test	Set	Item	Signal
A2	a	1	m_ot_x_4_org.wav
		2	m_ot_x_5_org.wav
		3	m_po_x_2_org.wav
		4	m_po_x_3_org.wav
		5	s_cl_2t_4_org.wav
		6	s_cl_ft_3_org.wav
		7	s_no_2t_3_org.wav
		8	s_no_mt_1_org.wav
		9	sbm_js_x_1_org.wav
		10	sbm_sm_x_4_org.wav
		11	som_fi_x_3_org.wav
		12	som_ot_x_2_org.wav
	b	1	m_ot_x_6_org.wav
		2	m_ot_x_7_org.wav
		3	m_po_x_4_org.wav
		4	m_si_x_2_org.wav
		5	s_cl_2t_5_org.wav
		6	s_cl_mt_2_org.wav
		7	s_no_ft_3_org.wav
		8	s_no_ft_4_org.wav
		9	sbm_js_x_2_org.wav
		10	sbm_sm_x_5_org.wav
		11	som_ad_x_1_org.wav
		12	som_ot_x_3_org.wav

Test	Set	Item	Signal
A3	a	1	m_ot_x_2_org.wav
		2	m_po_x_1_org.wav
		3	m_po_x_2_org.wav
		4	m_si_x_1_org.wav
		5	s_cl_2t_1_org.wav
		6	s_cl_ft_3_org.wav
		7	s_no_2t_1_org.wav
		8	s_no_mt_1_org.wav
		9	sbm_ms_x_1_org.wav
		10	sbm_sm_x_2_org.wav
		11	som_nt_x_1_org.wav
		12	som_ot_x_2_org.wav
	b	1	m_ot_x_3_org.wav
		2	m_po_x_3_org.wav
		3	m_po_x_4_org.wav
		4	m_si_x_2_org.wav
		5	s_cl_2t_2_org.wav
		6	s_cl_mt_2_org.wav
		7	s_no_2t_2_org.wav
		8	s_no_ft_4_org.wav
		9	sbm_js_x_2_org.wav
		10	sbm_sm_x_5_org.wav
		11	som_ot_x_1_org.wav
		12	som_ot_x_3_org.wav
Test	Set	Item	Signal
A4	a	1	m_ch_x_1_org.wav
		2	m_ot_x_2_org.wav
		3	m_po_x_1_org.wav
		4	m_si_x_1_org.wav
		5	s_cl_2t_1_org.wav
		6	s_cl_mt_1_org.wav
		7	s_no_2t_1_org.wav
		8	s_no_ft_2_org.wav
		9	sbm_ms_x_1_org.wav
		10	sbm_sm_x_2_org.wav
		11	som_fi_x_1_org.wav
		12	som_nt_x_1_org.wav
	b	1	m_cl_x_1_org.wav
		2	m_cl_x_2_org.wav
		3	m_ot_x_1_org.wav
		4	m_ot_x_3_org.wav
		5	s_cl_2t_2_org.wav
		6	s_cl_2t_3_org.wav
		7	s_no_2t_2_org.wav
		8	s_no_ft_1_org.wav
		9	sbm_sm_x_1_org.wav
		10	sbm_sm_x_3_org.wav
		11	som_fi_x_2_org.wav
		12	som_ot_x_1_org.wav

Test	Set	Item	Signal
B1	a	1	m_ch_x_1_org.wav
		2	m_cl_x_1_org.wav
		3	m_po_x_2_org.wav
		4	m_po_x_3_org.wav
		5	s_cl_2t_4_org.wav
		6	s_cl_ft_3_org.wav
		7	s_no_2t_2_org.wav
		8	s_no_ft_2_org.wav
		9	sbm_js_x_2_org.wav
		10	sbm_ms_x_1_org.wav
		11	som_ad_x_1_org.wav
		12	som_ot_x_4_org.wav
	b	1	m_po_x_4_org.wav
		2	m_po_x_5_org.wav
		3	m_po_x_6_org.wav
		4	m_si_x_2_org.wav
		5	s_cl_2t_2_org.wav
		6	s_cl_mt_1_org.wav
		7	s_no_2t_3_org.wav
		8	s_no_ft_4_org.wav
		9	sbm_sj_x_1_org.wav
		10	sbm_sm_x_4_org.wav
		11	som_ot_x_2_org.wav
		12	som_ot_x_6_org.wav
Test	Set	Item	Signal
B2	a	1	m_cl_x_2_org.wav
		2	m_ot_x_1_org.wav
		3	m_ot_x_8_org.wav
		4	m_ot_x_a_org.wav
		5	s_cl_2t_4_org.wav
		6	s_cl_2t_5_org.wav
		7	s_no_2t_3_org.wav
		8	s_no_ft_2_org.wav
		9	sbm_js_x_1_org.wav
		10	sbm_sm_x_4_org.wav
		11	som_fi_x_1_org.wav
		12	som_ot_x_5_org.wav
	b	1	m_ch_x_1_org.wav
		2	m_cl_x_1_org.wav
		3	m_ot_x_9_org.wav
		4	m_ot_x_b_org.wav
		5	s_cl_2t_3_org.wav
		6	s_cl_mt_1_org.wav
		7	s_no_ft_1_org.wav
		8	s_no_ft_3_org.wav
		9	sbm_sm_x_1_org.wav
		10	sbm_sm_x_3_org.wav
		11	som_fi_x_2_org.wav
		12	som_ot_x_6_org.wav

Test	Set	Item	Signal
B3	a	1	m_cl_x_1_org.wav
		2	m_ot_x_1_org.wav
		3	m_ot_x_5_org.wav
		4	m_ot_x_7_org.wav
		5	s_cl_2t_1_org.wav
		6	s_cl_2t_3_org.wav
		7	s_no_2t_3_org.wav
		8	s_no_ft_4_org.wav
		9	sbm_js_x_2_org.wav
		10	sbm_ms_x_1_org.wav
		11	som_ad_x_1_org.wav
		12	som_fi_x_2_org.wav
	b	1	m_ch_x_1_org.wav
		2	m_cl_x_2_org.wav
		3	m_ot_x_4_org.wav
		4	m_ot_x_6_org.wav
		5	s_cl_2t_2_org.wav
		6	s_cl_mt_1_org.wav
		7	s_no_ft_3_org.wav
		8	s_no_mt_1_org.wav
		9	sbm_js_x_1_org.wav
		10	sbm_sm_x_1_org.wav
		11	som_fi_x_1_org.wav
		12	som_fi_x_3_org.wav
Test	Set	Item	Signal
B4	a	1	m_ot_x_4_org.wav
		2	m_ot_x_8_org.wav
		3	m_po_x_2_org.wav
		4	m_po_x_7_org.wav
		5	s_cl_2t_1_org.wav
		6	s_cl_2t_5_org.wav
		7	s_no_2t_1_org.wav
		8	s_no_ft_3_org.wav
		9	sbm_js_x_2_org.wav
		10	sbm_sm_x_2_org.wav
		11	som_fi_x_3_org.wav
		12	som_ot_x_5_org.wav
	b	1	m_ot_x_5_org.wav
		2	m_ot_x_9_org.wav
		3	m_po_x_3_org.wav
		4	m_si_x_3_org.wav
		5	s_cl_2t_3_org.wav
		6	s_cl_mt_2_org.wav
		7	s_no_ft_1_org.wav
		8	s_no_mt_1_org.wav
		9	sbm_sm_x_1_org.wav
		10	sbm_sm_x_6_org.wav
		11	som_fi_x_4_org.wav
		12	som_ot_x_3_org.wav

2 ANNEX – Scripts

2.1 Overview

The script structure for the Low-Rate and high-rate tests are very similar. The Low-Rate test is run by script `l_main.sh`, while the High-Rate test is run by `h_main.sh`. A schematic of the nesting of the Low-Rate scripts follows (all scripts are in directory `./bin`):

```
l_main.sh
  genfer.csh
  clip_silence.sh
    actspeech.awk
    frameinfo.awk
  l_files_clipped.sh
  concat.sh
  anchor.sh
  ../aacPlus/bin/l_encdecdly.sh
    l_shift.sh
    shiftcirc.csh
  upsamp48.sh
  l_split.sh
  splitup.sh
    astrip.exe
  cksum_raw.sh
  getdelay.exe
  (each codec followed by similar steps as above)
  ../AMRWB+/bin/l_encdecdly.sh
  ../ct/bin/l_encdecdly.sh
  ../AAC/bin/l_encdecdly.sh
  ../AMR-WB/bin/l_encdecdly.sh
  l_anchors.sh
    anchor.sh
  upsamp48.sh
  l_split.sh
    splitup.sh
      astrip.exe
  cksum_raw.sh
  getdelay.exe
  clean.sh
```

2.2 ANNEX - Scripts used for the Low- and High-Rate tests

```
########################################################################
##
## AAC/bin/h_encdecdly.sh
##
########################################################################
#!/bin/bash
#
# run AAC encoder and decoder for 3GPP listening test
#

if [ $# -eq 0 ]; then
   echo "usage: `basename $0` <test condition>"
   echo ""
   exit
fi

EXP=$1  #H[123]
PRO=../tmp/AAC
ENCB=./mp4EvalCmdl.exe
DEC=./mp4dec.exe
BIT=../tmp/${EXP}.3gp
CPY=../../bin/CopyAudio.exe
CAT=../../h_preproc/all_cat
TMP1=../tmp/tmp1.wav
```

```
TMP2=../tmp/tmp2.wav

echo ""
echo "Shell script to run FhG AAC codec, test $EXP"
echo ""

case $EXP in
    H1) mode=s; rate=32; Fs=48; dly=3278;;
    H2) mode=s; rate=48; Fs=48; dly=3442;;
    H3) mode=s; rate=32; Fs=48; dly=3278;;
    *) echo "unknown test case!"; exit;;
esac

#encode
C="$ENCB -b ${rate}000 $CAT$Fs$mode.wav $BIT"; echo $C; $C

#decode and delay adjustment
if [[ ! $EXP = H3 ]]
then
    echo clear channel decode
    C="$DEC -if $BIT -of $TMP1"; echo $C; $C
    #remove encode-decode delay
    C="$CPY -l ${dly}: $TMP1 ${PRO}_${EXP}.wav"; echo $C; $C
else
    echo errored channel decode
    FER=../../bin/tmp_err_file.epf
    for lab in 3 6
    do
        for fer in 1 3
        do
            for lis in 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15
            do
                echo "lab $lab lis $lis"
                OFILE="${PRO}_FER${fer}_lab${lab}_lis${lis}_${EXP}.wav"
                    ../../bin/shiftcirc.csh $lis $lab 0 \
                    ../../ep/epf_${fer}.epf $FER
                C="$DEC -epf $FER -if $BIT -of $TMP1"; echo $C; $C
                C="$CPY -l ${dly}: $TMP1 $OFILE"; echo $C; $C
            done
        done
    done
fi

rm -f $TMP1 $TMP2 $FER



######################################################################
##
## AAC/bin/l_encdecdly.sh
##
######################################################################
#!/bin/bash
#
# run AAC encoder and decoder for 3GPP listening test
#

if [ $# -eq 0 ]; then
   echo "usage: `basename $0` <test condition>"
   echo ""
   exit
fi

EXP=$1
PRO=../tmp/AAC
ENCB=./mp4EvalCmdl.exe
DEC=./mp4dec.exe
BIT=../tmp/${EXP}.3gp
CPY=../../bin/CopyAudio.exe
CAT=../../preproc/all_cat
```

```
TMP1=../tmp/tmp1.wav
TMP2=../tmp/tmp2.wav

echo ""
echo "Shell script to run FhG AAC codec ..."
echo ""

case $EXP in
    A1) mode=m; rate=14; Fs=48; dly=3254;;
    A2) mode=s; rate=18; Fs=48; dly=3229;;
    A3) mode=m; rate=24; Fs=48; dly=3289;;
    A4) mode=s; rate=24; Fs=48; dly=3254;;

    B1) mode=m; rate=14; Fs=16; dly=3216;;
    B2) mode=s; rate=18; Fs=48; dly=3229;;
    B3) mode=m; rate=14; Fs=48; dly=3254;;
    B4) mode=s; rate=24; Fs=48; dly=3254;;
    *) echo "unknown test case!"; exit;;
esac

#encode
C="$ENCB -b ${rate}000 $CAT$Fs$mode.wav $BIT"; echo $C; $C

#decode and delay adjustment
if [[ ! $EXP = B3 && ! $EXP = B4 ]]
then
    echo clear channel decode
    C="$DEC -if $BIT -of $TMP1"; echo $C; $C
    #remove encode-decode delay
    C="$CPY -l ${dly}: $TMP1 ${PRO}_${EXP}.wav"; echo $C; $C
else
    echo errored channel decode
    FER=../../bin/tmp_err_file.epf
    for SET in a b
    do
        case $EXP$SET in
            B3a) lab[0]=1; lab[1]=5; exp=1;;
            B3b) lab[0]=2; lab[1]=6; exp=1;;
            B4a) lab[0]=3; lab[1]=7; exp=2;;
            B4b) lab[0]=4; lab[1]=8; exp=2;;
            *) echo "unknown experiment: $EXP$SET"; exit;;
        esac
        for i in 0 1
        do
            for lis in 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15
            do
                echo "lab ${lab[$i]} lis $lis"
                OFILE="${PRO}_lab${lab[$i]}_lis${lis}_${EXP}.wav"
                ../../bin/shiftcirc.csh $lis ${lab[$i]} $exp \
                    ../../ep/epf_3.epf $FER
                C="$DEC -epf $FER -if $BIT -of $TMP1"; echo $C; $C
                C="$CPY -l ${dly}: $TMP1 $OFILE"; echo $C; $C
            done
        done
    done
fi

rm -f $TMP1 $TMP2




######################################################################
##
## AMR-WB/bin/l_encdecdly.sh
##
######################################################################
#!/bin/bash

# run AMR-WB encoder and decoder for 3GPP listening test
#
```

```
if [ $# -eq 0 ]; then
   echo "usage: `basename $0` <test condition> <data set>"
   echo
   exit
fi

EXP=$1
PRO=../tmp/AMR-WB
FILT="../../bin/filter P341"
ENC="./coder.exe -mime"
EID=./AMR_MMS_EID.exe
DEC="./decoder.exe -mime"
BIT=../tmp/${EXP}.3gp
CPY=../../bin/CopyAudio.exe
PARAMS="-P integer16,0,16000.0,native,1,default"
CAT=../../preproc/all_cat16m.wav
TBIT=../tmp/tmp.3gp
TW1=../tmp/tmp1.wav
TW2=../tmp/tmp2.wav
TR1=../tmp/tmp1.raw
TR2=../tmp/tmp2.raw

echo ""
echo "Shell script to run AMR-WB codec ..."
echo ""

case $EXP in
    A1) mode=3; dly=390;;
    A2) mode=5; dly=390;;
    A3) mode=8; dly=390;;
    A4) mode=8; dly=390;;

    B1) mode=3; dly=390;;
    B2) mode=5; dly=390;;
    B3) mode=3; dly=390;;
    B4) mode=8; dly=390;;
    *) echo "unknown test case!"; exit;;
esac

#strip wav header, filter and encode
C="$CPY $CAT $TR1"; echo $C; $C
C="$FILT $TR1 $TR2 320"; echo $C; $C
C="$ENC $mode $TR2 $BIT"; echo $C; $C

#decode and delay adjustment
if [[ ! $EXP = B3 && ! $EXP = B4 ]]
then
    echo clear channel decode
    C="$DEC $BIT $TR2"; echo $C; $C
    #add wave header and remove encode-decode delay
    C="$CPY $PARAMS $TR2 $TW1"; echo $C; $C
    C="$CPY -l ${dly}: $TW1 ${PRO}_${EXP}.wav"; echo $C; $C
else
    echo errored channel decode
    FER=../../bin/tmp_err_file.epf
    for SET in a b
    do
        case $EXP$SET in
            B3a) lab[0]=1; lab[1]=5; exp=1;;
            B3b) lab[0]=2; lab[1]=6; exp=1;;
            B4a) lab[0]=3; lab[1]=7; exp=2;;
            B4b) lab[0]=4; lab[1]=8; exp=2;;
            *) echo "unknown experiment: $EXP$SET"; exit;;
        esac
        for i in 0 1
        do
            for lis in 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15
            do
                echo "lab ${lab[$i]} lis $lis"
                OFILE="${PRO}_lab${lab[$i]}_lis${lis}_${EXP}.wav"
```

```
                ../../bin/shiftcirc.csh $lis ${lab[$i]} $exp \
                    ../../ep/epf_3.epf $FER
                #error bitstream
                $EID $FER $BIT $TBIT
                #decode
                C="$DEC $TBIT $TR2"; echo $C; $C
                #remove encode-decode delay
                C="$CPY $PARAMS $TR2 $TW1"; echo $C; $C
                C="$CPY -l ${dly}: $TW1 $OFILE"; echo $C; $C
            done
        done
    done
fi

rm -f $TBIT
rm -f $TW1 $TW2
rm -f $TR1 $TR2



######################################################################
##
## AMRWB+/bin/l_encdecdly.sh
##
######################################################################
#!/bin/bash
#
# run AMRWB+ encoder and decoder for 3GPP low-rate listening test
#

if [ $# -eq 0 ]; then
   echo "usage: `basename $0` <test condition>"
   echo ""
   exit
fi

EXP=$1
PRO=../tmp/AMRWB+
ENC=./enc_wbplus.exe
EID=./EID.exe
FER=../../ep/epf_3.epf
DEC=./dec_wbplus.exe
BIT=../tmp/${EXP}.3gp
CPY=../../bin/CopyAudio.exe
CAT=../../preproc/all_cat
TMP1=../tmp/tmp1.wav
TMP2=../tmp/tmp2.wav

echo ""
echo "Shell script to run AMRWB+ codec ..."
echo ""

case $EXP in
    A1) mode=m; rate=14; Fs=24; opt=; dly=668;;
    A2) mode=s; rate=18; Fs=24; opt=; dly=885;;
    A3) mode=m; rate=24; Fs=24; opt=; dly=668;;
    A4) mode=s; rate=24; Fs=24; opt=; dly=885;;

    B1) mode=m; rate=14; Fs=16; opt="_lc"; dly=445;;
    B2) mode=s; rate=18; Fs=24; opt="_lc"; dly=885;;
    B3) mode=m; rate=14; Fs=24; opt=; dly=668;;
    B4) mode=s; rate=24; Fs=24; opt=; dly=885;;
    *) echo "unknown test case!"; exit;;
esac

#encode
C="$ENC -mode $rate$mode$opt -if $CAT$Fs$mode.wav -of $BIT"; echo $C; $C

#decode and delay adjustment
if [[ ! $EXP = B3 && ! $EXP = B4 ]]
```

```
then
    echo clear channel decode
    C="$DEC -mode $rate$mode -fs ${Fs}000 -if $BIT -of $TMP2"; echo $C; $C
    #remove encode-decode delay
    C="$CPY -l ${dly}: $TMP2 ${PRO}_${EXP}.wav"; echo $C; $C
else
    echo errored channel decode
    FER=../../bin/tmp_err_file.epf
    for SET in a b
    do
        case $EXP$SET in
            B3a) lab[0]=1; lab[1]=5; exp=1;;
            B3b) lab[0]=2; lab[1]=6; exp=1;;
            B4a) lab[0]=3; lab[1]=7; exp=2;;
            B4b) lab[0]=4; lab[1]=8; exp=2;;
            *) echo "unknown experiment: $EXP$SET"; exit;;
        esac
        for i in 0 1
        do
            for lis in 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15
            do
                echo "lab ${lab[$i]} lis $lis"
                OFILE="${PRO}_lab${lab[$i]}_lis${lis}_${EXP}.wav"
                ../../bin/shiftcirc.csh $lis ${lab[$i]} $exp \
                    ../../ep/epf_3.epf $FER
                C="$EID -mode $rate$mode -if $BIT -of $TMP1 -fer $FER"; echo $C; $C
                C="$DEC -mode $rate$mode -fs ${Fs}000 -if $TMP1 -of $TMP2"; echo $C; $C
                C="$CPY -l ${dly}: $TMP2 $OFILE"; echo $C; $C
            done
        done
    done
fi

rm -f $TMP1 $TMP2



######################################################################
##
## RN/bin/h_encdecdly.sh
##
######################################################################
#!/bin/bash
#
# run RN encoder and decoder for 3GPP listening test
#

if [ $# -eq 0 ]; then
   echo "usage: `basename $0` <test condition>"
   echo ""
   exit
fi

EXP=$1  #H[123]
PRO=../tmp/RN
COD="./coder.exe"
BIT=../tmp/${EXP}.3gp
CPY=../../bin/CopyAudio.exe
CAT=../../h_preproc/all_cat
TMP1=../tmp/tmp1.wav
TMP2=../tmp/tmp2.wav

echo ""
echo "Shell script to run RN codec, test $EXP"
echo ""

case $EXP in
    H1) mode=s; rate=32; Fs=22; dly=0;;
    H2) mode=s; rate=48; Fs=44; dly=0;;
    H3) mode=s; rate=32; Fs=22; dly=0;;
```

```
    *) echo "unknown test case!"; exit;;
esac

#encode and decode
C="$COD -b ${rate} $CAT$Fs$mode.wav $TMP1"; echo $C; $C
#remove encode-decode delay
C="$CPY -l ${dly}: $TMP1 ${PRO}_${EXP}.wav"; echo $C; $C



######################################################################
##
## aacPlus/bin/h_encdecdly.sh
##
######################################################################
#!/bin/bash
#
# run aacPlus encoder and decoder in course of the 3GPP listening test
#

if [ $# -eq 0 ]; then
   echo "usage: `basename $0` <test condition>"
   echo ""
   exit
fi

EXP=$1  #H[123]
PRO=../tmp/aacPlus
ENCB=./aacPlusenc.exe
DEC=./aacPlusdec.exe
BIT=../tmp/${EXP}.3gp
CPY=../../bin/CopyAudio.exe
CAT=../../h_preproc/all_cat
TMP1=../tmp/tmp1.wav
TMP2=../tmp/tmp2.wav

echo ""
echo "Shell script to run Coding Technologies' aacPlus codec, test $EXP"
echo ""

case $EXP in
    H1) mode=s; rate=32; Fs=48; dly=4166; opts=;;
    H2) mode=s; rate=48; Fs=48; dly=4166; opts=;;
    H3) mode=s; rate=32; Fs=48; dly=4166; opts=;;
    *) echo "unknown test case!"; exit;;
esac

#encode
C="$ENCB -if $CAT$Fs$mode.wav -of $BIT -br ${rate}000 $opts"; echo $C; $C

#decode and delay adjustment
if [[ ! $EXP = H3 ]]
then
    echo clear channel decode
    C="$DEC $BIT $TMP1"; echo $C; $C
    #remove encode-decode delay
    C="$CPY -l ${dly}: $TMP1 ${PRO}_${EXP}.wav"; echo $C; $C
else
    echo errored channel decode
    FER=../../bin/tmp_err_file.epf
    for lab in 3 6
    do
        for fer in 1 3
        do
            for lis in 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15
            do
                echo "lab $lab lis $lis"
                OFILE="${PRO}_FER${fer}_lab${lab}_lis${lis}_${EXP}.wav"
                    ../../bin/shiftcirc.csh $lis $lab 0 \
                    ../../ep/epf_${fer}.epf $FER
```

```
                C="$DEC $BIT $TMP1 $FER"; echo $C; $C
                C="$CPY -l ${dly}: $TMP1 $OFILE"; echo $C; $C
            done
        done
    done
fi

rm -f $TMP1 $TMP2 $FER



######################################################################
##
## aacPlus/bin/l_encdecdly.sh
##
######################################################################
#!/bin/bash
#
# run aacPlus encoder and decoder in course of the 3GPP listening test
#

if [ $# -eq 0 ]; then
   echo "usage: `basename $0` <test condition> <data set>"
   echo ""
   exit
fi

EXP=$1
PRO=../tmp/aacPlus
ENCB=./aacPlusenc.exe
DEC=./aacPlusdec.exe
BIT=../tmp/${EXP}.3gp
CPY=../../bin/CopyAudio.exe
CAT=../../preproc/all_cat
TMP1=../tmp/tmp1.wav
TMP2=../tmp/tmp2.wav

echo ""
echo "Shell script to run Coding Technologies' aacPlus codec, test $EXP$SET"
echo ""

case $EXP in
    A1) mode=m; rate=14; Fs=48; dly=4166; opts="-m";;
    A2) mode=s; rate=18; Fs=48; dly=4168; opts=;;
    A3) mode=m; rate=24; Fs=48; dly=4166; opts="-m";;
    A4) mode=s; rate=24; Fs=48; dly=4166; opts=;;

    B1) mode=m; rate=14; Fs=16; dly=2081; opts="-m";;
    B2) mode=s; rate=18; Fs=48; dly=4168; opts=;;
    B3) mode=m; rate=14; Fs=48; dly=4166; opts="-m";;
    B4) mode=s; rate=24; Fs=48; dly=4166; opts=;;
    *) echo "unknown test case!"; exit;;
esac

#encode
C="$ENCB -if $CAT$Fs$mode.wav -of $BIT -br ${rate}000 $opts"; echo $C; $C

#decode and delay adjustment
if [[ ! $EXP = B3 && ! $EXP = B4 ]]
then
    echo clear channel decode
    C="$DEC $BIT $TMP1"; echo $C; $C
    #remove encode-decode delay
    C="$CPY -l ${dly}: $TMP1 ${PRO}_${EXP}.wav"; echo $C; $C
else
    echo errored channel decode
    FER=../../bin/tmp_err_file.epf
    for SET in a b
    do
        case $EXP$SET in
```

```
            B3a) lab[0]=1; lab[1]=5; exp=1;;
            B3b) lab[0]=2; lab[1]=6; exp=1;;
            B4a) lab[0]=3; lab[1]=7; exp=2;;
            B4b) lab[0]=4; lab[1]=8; exp=2;;
            *) echo "unknown experiment: $EXP$SET"; exit;;
        esac
        for i in 0 1
        do
            for lis in 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15
            do
                echo "lab ${lab[$i]} lis $lis"
                OFILE="${PRO}_lab${lab[$i]}_lis${lis}_${EXP}.wav"
                ../../bin/shiftcirc.csh $lis ${lab[$i]} $exp \
                    ../../ep/epf_3.epf $FER
                C="$DEC $BIT $TMP1 $FER"; echo $C; $C
                C="$CPY -l ${dly}: $TMP1 $OFILE"; echo $C; $C
            done
        done
    done
fi

rm -f $TMP1 $TMP2
```

```
######################################################################
##
## bin/actspeech.awk
##
######################################################################
BEGIN {
  thresh = exp(log(10)*(90.3090 + lev)/10);
  c=0;
  energy = 0;
}
// { for (i=2;i<=NF;i++)
  {
    if (c++ < flen) {
      energy += $i*$i;
    }
    else {
      if (energy / flen > thresh) {
    print 1;
      }
      else {
    print 0;
      }
      energy = 0;
      c=0;
    }
  }
}
```

```
######################################################################
##
## bin/anchor.sh
##
######################################################################
#!/bin/bash
#
# anchor.sh according to Section 8 of S4-030677_AMRWB+TestPlan_v0.8.doc
#
# HP20031103 20031104
#

RES=./ResampAudio.exe
CPY=./CopyAudio.exe
```

```
INF=./InfoAudio.exe
TMPFILE=anchor_tmp_file.wav

if [ $# -lt 2 ]; then
    echo  "usage:  `basename  $0`  [-lp<cut-off>]  [-s<stereo_degree>]  [-fsout[16|22|24|44]]  [-
monoout] <infile.wav> <outfile> "
    echo ""
    exit
fi
echo $*

INARG=$(($# - 1))
OUTARG=$#
INFILE="${!INARG}"
OUTFILE="${!OUTARG}"
OUTEXT=${OUTFILE:(-4):4}

CUTOFF=0
STEREO=0
OUTFS=0
OUTMONO=0
for (( I=1 ; I<$#-1 ; I++ )) ; do
    OPT=${!I}
    if [[ ${OPT:0:3} == "-lp" ]] ; then
        CUTOFF=${OPT:3}
    elif [[ ${OPT:0:2} == "-s" ]] ; then
        STEREO=${OPT:2}
    elif [[ $OPT == "-fsout44" ]] ; then
        OUTFS=44100
    elif [[ $OPT == "-fsout24" ]] ; then
        OUTFS=24000
    elif [[ $OPT == "-fsout22" ]] ; then
        OUTFS=22050
    elif [[ $OPT == "-fsout16" ]] ; then
        OUTFS=16000
    elif [[ $OPT == "-monoout" ]] ; then
        OUTMONO=1
    else
        echo "ERROR: Unknown option "$OPT
        echo ""
        exit 1
    fi
done

RESOPTS=""
if [[ $CUTOFF == 3500 ]] ; then
    RESOPTS=$RESOPTS" -f cutoff=0.0729167"
elif [[ $CUTOFF == 7000 ]] ; then
    RESOPTS=$RESOPTS" -f cutoff=0.145833"
elif [[ $CUTOFF != 0 ]] ; then
    echo "ERROR: Wrong cut-off "$CUTOFF
    echo ""
    exit 1
fi
if [[ $OUTFS == 44100 ]] ; then
    RESOPTS=$RESOPTS" -s 44100"
elif [[ $OUTFS == 24000 ]] ; then
    RESOPTS=$RESOPTS" -s 24000"
elif [[ $OUTFS == 22050 ]] ; then
    RESOPTS=$RESOPTS" -s 22050"
elif [[ $OUTFS == 16000 ]] ; then
    RESOPTS=$RESOPTS" -s 16000"
else
    RESOPTS=$RESOPTS" -i 1"
fi

if [[ $OUTMONO != 0 && $STEREO != 0 ]] ; then
    echo "ERROR: Options -monoout and -s<stereo_degree> given simultaneously"
    echo ""
    exit 1
fi
```

```
CPYOPTS=""
if [[ $STEREO == 6 ]] ; then
    CPYOPTS=$CPYOPTS" -cA 3/4*A+1/4*B -cB 1/4*A+3/4*B"
elif [[ $STEREO == 12 ]] ; then
    CPYOPTS=$CPYOPTS" -cA 5/8*A+3/8*B -cB 3/8*A+5/8*B"
elif [[ $STEREO != 0 ]] ; then
    echo "ERROR: Wrong stereo-degree "$STEREO
    echo ""
    exit 1
fi
if [[ $OUTMONO == 1 ]] ; then
    CPYOPTS=$CPYOPTS" -cA 1/2*A+1/2*B"
    #use below for 2-channel mono presentation
    #CPYOPTS=$CPYOPTS" -cA 1/2*A+1/2*B -cB 1/2*A+1/2*B"
fi

if [[ $OUTEXT == ".raw" ]] ; then
    CPYOPTS=$CPYOPTS" -F noheader -D integer16"
fi

echo ""
echo "ANCHOR: checking input file "$INFILE
echo ""
if ! { $INF $INFILE | grep "Sampling frequency: 48000 Hz" ; } ; then
    echo "ERROR: Sampling frequency not 48000 Hz"
    echo ""
    exit 1
fi
if ! { $INF $INFILE | grep "Number of channels: 2" ; } ; then
    echo "ERROR: Number of channels not 2"
    echo ""
    exit 1
fi

if [[ $OUTFS != 0 || $CUTOFF != 0 ]] ; then
    echo ""
    echo "ANCHOR: filtering/resampling ..."
    echo ""
    echo $RES $RESOPTS $INFILE $TMPFILE
    $RES $RESOPTS $INFILE $TMPFILE
else
    TMPFILE=$INFILE
fi

echo ""
echo "ANCHOR: copying/downmixing ..."
echo ""
echo $CPY $CPYOPTS $TMPFILE $OUTFILE
$CPY $CPYOPTS $TMPFILE $OUTFILE

if [[ $TMPFILE != $INFILE ]] ; then
    rm $TMPFILE

fi

echo ""
echo "ANCHOR: done ..."
echo ""

exit 0
```

```
######################################################################
##
## bin/cksum_raw.sh
##
######################################################################
#!/bin/bash
```

```
case $# in
    0) echo "usage: $0 wavfile(s)"; exit;;
esac

for i in $*
do
    ./Copyaudio.exe $i temp.raw 1>/dev/null 2>/dev/null
    ckout=`cksum temp.raw`
    echo $ckout | sed "{s'temp.raw'$i'}"
done
```

```
######################################################################
##
## bin/clean.sh
##
######################################################################
#!/bin/bash

RM="rm -f"
#RM=ls
OUTDIR="exp_H1 exp_H2 exp_H3 \
exp_A1a exp_A1b exp_A2a exp_A2b exp_A3a exp_A3b exp_A4a exp_A4b \
exp_B1a exp_B1b exp_B2a exp_B2b exp_B3a exp_B3b exp_B4a exp_B4b"

echo
echo remove temp files
echo
(cd ..; find . -name "*~" -exec $RM {} \;);
(cd ..; find . -name "*.raw" -exec $RM {} \;);
(cd ..; find . -name "tmp?" -exec $RM {} \;);

if [[ $1 = clobber ]]
then
    echo
    echo revert back to initial state
    echo
    C="$RM ../org/ClippedSpeech/*.wav"; echo $C; $C
    C="$RM ../org/*.wav"; echo $C; $C
    $RM ../preproc/*.wav ../preproc/*.tim
    $RM ../h_preproc/*.wav ../h_preproc/*.tim
    CLEANDIR="AAC AMR-WB AMRWB+ aacPlus ct RN ref"
    for i in $CLEANDIR
    do
        C="$RM ../$i/processed/*.*"; echo $C; $C
        C="$RM ../$i/tmp/*.*"; echo $C; $C
    done
    for i in $OUTDIR
    do
        find ../output/$i  -name "*.wav" -exec $RM {} \;
    done
fi
```

```
######################################################################
##
## bin/clip_silence.sh
##
######################################################################
#!/bin/bash
if [ $# -eq 0 ]
then
    echo "Usage:                                                                    v. 0.1"
    echo "$0 <Outdir> <file_1.wav> [...<file_n.wav>]"
    echo " "
    echo " This script will clip leading and trailing inactivity of input speech files"
```

```
    echo "   required input file format: 2-channel 48000 Hz wav"
    echo " "
    echo " The clipped output files will be created in directory ./org"
    echo " "
    exit 1
fi

#### parameters #####################################################################
ACTLEV=-50   # threshold [dBov] below which a frame is considered inactive
FLEN=480     # frame length 10 ms, assuming sf = 48000 Hz
PREHANG=10   # (in frames) Allow 100 ms silence before active speech
POSTHANG=15  # (in frames) Allow 150 ms hang over after active speech

#### pathes to executables to be adjusted to local needs ############################
INFOAUDIO=./InfoAudio.exe
RESAMPAUDIO=./ResampAudio.exe
COPYAUDIO=./CopyAudio.exe
#####################################################################################

# initialization
outdir=$1; shift
mkdir -p $outdir

while [ $# -gt 0 ]
do
    infile=$1
    echo $infile
    outfile=$outdir/`basename $1`

    # check for valid WAV header
    if ! $INFOAUDIO $infile | grep "WAVE file" >/dev/null; then
        echo "Incorrect wav header in file $infile"
        exit 1
    fi
    # check sampling rate
    if ! $INFOAUDIO $infile | grep "Sampling frequency: 48000 Hz" > /dev/null; then
        echo "Incorrect sampling frequency in file $infile"
        exit 1
    fi
    # check input channels
    if ! $INFOAUDIO $infile | grep "Number of channels: 2" > /dev/null; then
        echo "Incorrect number of channels in file $infile"
        exit 1
    fi

    # convert to mono strip the wav header
    $COPYAUDIO --chanA="0.5*A+0.5*B" -F noheader $infile monofile.raw 1>/dev/null 2>/dev/null

    # Speech activity program
    od -t d2 -v monofile.raw | gawk -v lev=$ACTLEV -v flen=$FLEN -f actspeech.awk > frameinfo.txt

    # find first and last activity frame
    tmp=`cat frameinfo.txt | gawk -v flen=$FLEN -v prehang=$PREHANG -v posthang=$POSTHANG -f
frameinfo.awk`
    finfo=($tmp)

    echo "First sample: ${finfo[0]}"
    echo "Last sample:  ${finfo[1]}"

    # Do the clipping
    $COPYAUDIO --limits=${finfo[0]}\:${finfo[1]} $infile $outfile 1>/dev/null 2>/dev/null

    shift
done

# cleanup
rm -f monofile.raw frameinfo.txt
```

```
#####################################################################
##
## bin/concat.sh
##
#####################################################################
#!/bin/bash
if [ $# -eq 0 ]
then
    echo "Usage: "
    echo "$0 <file_1.wav> [...<file_n.wav>]"
    echo " "
    echo " This script will concatenate 2-channel 48000 Hz wav input files to the file"
    echo "   ../preproc/all_cat.wav"
    echo " it will also create a time file called"
    echo "   ../preproc/all_cat.tim"
    exit 1
fi

#### pathes to executables to be adjusted to local needs ##############################
RESAMPAUDIO=./ResampAudio.exe
COPYAUDIO=./CopyAudio.exe
INFOAUDIO=./InfoAudio.exe
#### pathes to files to be adjusted to local needs #####################################
CatFil=./catfile.raw
AllCat=../preproc/all_cat48s.wav
AllTim=../preproc/all_cat48s.tim
########################################################################################

# initialization
rm -f $CatFil
touch $CatFil
rm -f $AllTim
touch $AllTim

while [ $# -gt 0 ]
do
    infile=$1
    echo $infile

    # check for valid WAV header
    if ! $INFOAUDIO $infile | grep "WAVE file" >/dev/null; then
        echo "Incorrect wav header in file $infile"
        exit 1
    fi
    # check sampling rate
    if ! $INFOAUDIO $infile | grep "Sampling frequency: 48000 Hz" > /dev/null; then
        echo "Incorrect sampling frequency in file $infile"
        exit 1
    fi
    # check input channels
    if ! $INFOAUDIO $infile | grep "Number of channels: 2" > /dev/null; then
        echo "Incorrect number of channels in file $infile"
        exit 1
    fi

    # strip the wav header
    $COPYAUDIO -F noheader $infile tmpfile.raw 1>/dev/null 2>/dev/null

    # concatenate
    cat tmpfile.raw >> catfile.raw

    # derive length information in samples
    len=`ls -l tmpfile.raw | awk '{print $5}'`
    let len=len/2;
    echo `basename $infile` $len >> $AllTim
    shift
done

# regenerate wav file
$COPYAUDIO -t noheader -P "integer16,0,48000., native, 2, default" catfile.raw $AllCat
1>/dev/null 2>/dev/null
```

```
# clean up
rm -f catfile.raw tmpfile.raw



######################################################################
##
## bin/frameinfo.awk
##
######################################################################
BEGIN {begcnt=0; fcnt = 0 };
/0/ { fcnt++; if (notinit == 0) begcnt++  }
/1/ { fcnt++; notinit = 1; lastact=fcnt; }
END {
# apply some hangover
  firstsamp = (begcnt-prehang) * flen + 1;
  if (firstsamp < 1) firstsamp = 1;

  lastsamp = (lastact+posthang) * flen;
  if (lastsamp > fcnt * flen) lastsamp = fcnt * flen;

# return limits in samples
  print firstsamp " " lastsamp
}



######################################################################
##
## bin/genfer.csh
##
######################################################################
#!/bin/tcsh
#random generator seed
set SEED = $1
# FER
set FER=$2

# The length of the pattern shall cover the complete test+training material
# 8 different operational conditions
set NUM_CASES = 8
# 2 sub-experiments per op.cond.
set NUM_SUBEXP = 2
# 12 items per test
set NUM_TEST_ITEMS = 12
# 4 items for training
set NUM_TRAIN_ITEMS = 4
# The total number of items
set NUM_TOT_ITEMS = `echo "$NUM_CASES*$NUM_SUBEXP*$NUM_TEST_ITEMS+$NUM_TRAIN_ITEMS" | bc`

# The assumed maximum length of the items is 10s
set LEN_ITEM = 10
# The assumed maximum frame rate is 100, i.e. codec frame length of 10ms
set FRATE=100

# The length of the error pattern in frames
set LEN = `echo "$NUM_TOT_ITEMS * $LEN_ITEM * $FRATE" | bc`

set PATFILE=ferpat_FER$FER\_SEED$SEED.dat

echo "A pattern with $LEN frames is being created... name: $PATFILE"

gawk -v SEED=$SEED -v FER=$FER -v LEN=$LEN 'BEGIN { srand(SEED); for (i=0;i<LEN;i++) print
(rand() < FER) }' > $PATFILE
```

```
########################################################################
##
## bin/h_anchors.sh
##
########################################################################
#!/bin/bash
#make anchors for High-Rate test

EXP=$1
ANCHOR=./anchor.sh
CAT=../h_preproc/all_cat48s.wav
OUT=../ref/tmp

case $EXP in
    H1)
    echo
    echo make stereo anchors for test $EXP
    echo
    cp $CAT $OUT/opref_${EXP}.wav
    cp $CAT $OUT/hidref_${EXP}.wav
    $ANCHOR -lp3500  $CAT $OUT/lp3500_${EXP}.wav
    $ANCHOR -lp7000  $CAT $OUT/lp7000_${EXP}.wav
    ;;

    H2|H3)
    echo
    echo duplicate anchors for test $EXP
    echo
    for i in opref hidref lp3500 lp7000
    do
        cp $OUT/${i}_H1.wav $OUT/${i}_${EXP}.wav
    done
    ;;
esac
```

```
########################################################################
##
## bin/h_main.sh
##
########################################################################
#!/bin/bash

case $# in
    0) echo "usage: $0 [H1|H2|H3] [cat|split] [OutputPath]"; exit;;
esac
case $1 in
    H1|H2) EXP="$1"; shift; CDIR="aacPlus ct AAC RN";;
    H3) EXP="$1"; shift; CDIR="aacPlus ct AAC";;
    *) echo "unknown experiment"; exit;;
esac
case $1 in
    split) MODE=split; shift;;
    cat) MODE=cat; shift;;
    *) MODE=cat;;
esac
case $# in
    1) OUT=$1; shift; echo "setting output to $OUT";;
    *) OUT=../output;;
esac

LOGFILE=$EXP
RATELOG=${LOGFILE}_bitrate.txt
CKSUMLOG=${LOGFILE}_cksum.txt
DELAYLOG=${LOGFILE}_delay.txt
SEED=171094
CAT="../h_preproc/all_cat"
ORG="../h_org"
PRE="../h_preproc"
```

```
TIM="$PRE/all_cat48s.tim"
rm -f $RATELOG
rm -f $CKSUMLOG

echo
echo high-rate test for experiments $LOGFILE
echo

echo
echo generate random error files
echo
./genfer.csh $SEED .01; cp ferpat_FER.01_SEED$SEED.dat ../ep/epf_1.epf
./genfer.csh $SEED .03; cp ferpat_FER.03_SEED$SEED.dat ../ep/epf_3.epf

echo
echo process signals
echo

echo
echo concatenate original signals and silence signal
echo make 22.05 kHz and 44.1 kHz versions
echo
./concat.sh $ORG/Training/*.wav $ORG/Test_Items/*.wav ../org/Silence/silence.wav
mv ../preproc/all_cat48s.wav $PRE/all_cat48s.wav
mv ../preproc/all_cat48s.tim $TIM
./anchor.sh -fsout22 ${CAT}48s.wav ${CAT}22s.wav
./anchor.sh -fsout44 ${CAT}48s.wav ${CAT}44s.wav
./cksum_raw.sh ${CAT}48s.wav >> $CKSUMLOG
echo
echo remove silence signal from *.tim file
echo
mv $TIM tmp
grep -v silence.wav tmp > $TIM
rm tmp
ls $PRE

exp=$EXP
echo
echo proponent and reference codecs for experiment $exp
echo
for c in $CDIR
do
    (cd ../$c/bin; ./h_encdecdly.sh $exp)
    ls  ../$c/tmp

    echo
    echo "upsample (and split) coded signals, then cksum and check delay"
    echo
    for i in ../$c/tmp/*_${exp}.wav
    do
        base=${i##*/}
        j=../$c/processed/$base
        ./upsamp48.sh $i $j
        rm $i

        if [[ $MODE == cat ]]
        then
            ./cksum_raw.sh $j >> $CKSUMLOG
            delay=`./getdelay.exe -if1 ${CAT}48s.wav -if2 $j | grep delay`
            echo $j $delay >> $DELAYLOG
            if [[ $exp == H3 ]]
            then
                #don't have room to store it!
                rm $j
            fi
        else
            tag=${base%.wav}
            ./splitup.sh $j $OUT/exp_${exp}/${tag} $TIM
            rm $j
            ./cksum_raw.sh $OUT/exp_${exp}/*${base} >> $CKSUMLOG
        fi
```

```
    done
done

echo
echo make anchors
echo
./h_anchors.sh $exp
ls  ../ref/tmp

echo
echo "upsample (and split) anchors, then cksum and check delay"
echo
for i in ../ref/tmp/*_${exp}.wav
do
    base=${i##*/}
    j=../ref/processed/$base
    ./upsamp48.sh $i ../ref/processed/$base
    #don't delete anchors, since need for H2, H3

    if [[ $MODE == cat ]]
    then
        ./cksum_raw.sh $j >> $CKSUMLOG
        delay=`./getdelay.exe -if1 ${CAT}48s.wav -if2 $j | grep delay`
        echo $j $delay >> $DELAYLOG
    else
        tag=${base%.wav}
        ./splitup.sh $j $OUT/exp_${exp}/${tag} $TIM
        rm $j
        ./cksum_raw.sh $OUT/exp_${exp}/*${base} >> $CKSUMLOG
    fi
done

echo
echo cleanup
echo
./clean.sh
```

```
#########################################################################
##
## bin/h_split.sh
##
#########################################################################
#!/bin/bash

EXP=$1

DIRLIST="aacPlus ct AAC RN ref"
SPLIT=./splitup.sh
SOUT=../output/exp_${EXP}
TIM=../h_preproc/all_cat48s.tim

for dir in $DIRLIST
do
    for file in ../$dir/processed/*_${EXP}.wav
    do
        base=${file##*/}
        tag=${base%.wav}
        C="$SPLIT $file $SOUT/${tag} $TIM"; echo $C; $C
    done
done
```

```
#########################################################################
##
## bin/l_anchors.sh
##
```

```
#########################################################################
#!/bin/bash
#make anchors for Low-Rate test

EXP=$1
ANCHOR=./anchor.sh
CAT=../preproc/all_cat48s.wav
OUT=../ref/tmp

case $EXP in
    A1|A3|B1|B3)
    echo
    echo make mono anchors for test $EXP
    echo
    $ANCHOR -monoout $CAT $OUT/opref_${EXP}.wav
    cp $OUT/opref_${EXP}.wav $OUT/hidref_${EXP}.wav
    $ANCHOR -lp3500 -monoout $CAT $OUT/lp3500_${EXP}.wav
    $ANCHOR -lp7000 -monoout $CAT $OUT/lp7000_${EXP}.wav
    ;;

    A2|A4|B2|B4)
    echo
    echo make stereo anchors for test $EXP
    echo
    cp $CAT $OUT/opref_${EXP}.wav
    cp $CAT $OUT/hidref_${EXP}.wav
    $ANCHOR -lp3500 -s12 $CAT $OUT/lp3500_s12_${EXP}.wav
    $ANCHOR -lp7000 -s12 $CAT $OUT/lp7000_s12_${EXP}.wav
    $ANCHOR -lp7000 -s6  $CAT $OUT/lp7000_s6_${EXP}.wav
    ;;
esac



#########################################################################
##
## bin/l_files_clipped.sh
##
#########################################################################
exp=$1

ORG=../org

#training
echo $ORG/Training/m_vo_x_1_org.wav
echo $ORG/Training/s_no_ft_9_org.wav
echo $ORG/Training/sbm_fi_x_9_org.wav
echo $ORG/Training/som_ot_x_9_org.wav

#testing
for set in a b
do
SET="Set$exp$set"
case $SET in
SetA1a)
echo $ORG/clipped_items/m_ot_x_8_org.wav
echo $ORG/clipped_items/m_ot_x_a_org.wav
echo $ORG/clipped_items/m_po_x_5_org.wav
echo $ORG/clipped_items/m_po_x_7_org.wav
echo $ORG/Speech/s_cl_2t_3_org.wav
echo $ORG/Speech/s_cl_2t_4_org.wav
echo $ORG/Speech/s_no_2t_1_org.wav
echo $ORG/Speech/s_no_ft_1_org.wav
echo $ORG/clipped_items/sbm_js_x_1_org.wav
echo $ORG/clipped_items/sbm_ms_x_1_org.wav
echo $ORG/clipped_items/som_fi_x_4_org.wav
echo $ORG/clipped_items/som_ot_x_4_org.wav
;;
SetA1b)
echo $ORG/clipped_items/m_ot_x_9_org.wav
```

```
echo $ORG/clipped_items/m_ot_x_b_org.wav
echo $ORG/clipped_items/m_po_x_6_org.wav
echo $ORG/clipped_items/m_si_x_3_org.wav
echo $ORG/Speech/s_cl_2t_5_org.wav
echo $ORG/Speech/s_cl_mt_2_org.wav
echo $ORG/Speech/s_no_2t_2_org.wav
echo $ORG/Speech/s_no_ft_2_org.wav
echo $ORG/clipped_items/sbm_sj_x_1_org.wav
echo $ORG/clipped_items/sbm_sm_x_6_org.wav
echo $ORG/clipped_items/som_ot_x_5_org.wav
echo $ORG/clipped_items/som_ot_x_6_org.wav
;;
SetA2a)
echo $ORG/clipped_items/m_ot_x_4_org.wav
echo $ORG/clipped_items/m_ot_x_5_org.wav
echo $ORG/clipped_items/m_po_x_2_org.wav
echo $ORG/clipped_items/m_po_x_3_org.wav
echo $ORG/Speech/s_cl_2t_4_org.wav
echo $ORG/Speech/s_cl_ft_3_org.wav
echo $ORG/Speech/s_no_2t_3_org.wav
echo $ORG/Speech/s_no_mt_1_org.wav
echo $ORG/clipped_items/sbm_js_x_1_org.wav
echo $ORG/clipped_items/sbm_sm_x_4_org.wav
echo $ORG/clipped_items/som_fi_x_3_org.wav
echo $ORG/clipped_items/som_ot_x_2_org.wav
;;
SetA2b)
echo $ORG/clipped_items/m_ot_x_6_org.wav
echo $ORG/clipped_items/m_ot_x_7_org.wav
echo $ORG/clipped_items/m_po_x_4_org.wav
echo $ORG/clipped_items/m_si_x_2_org.wav
echo $ORG/Speech/s_cl_2t_5_org.wav
echo $ORG/Speech/s_cl_mt_2_org.wav
echo $ORG/Speech/s_no_ft_3_org.wav
echo $ORG/Speech/s_no_ft_4_org.wav
echo $ORG/clipped_items/sbm_js_x_2_org.wav
echo $ORG/clipped_items/sbm_sm_x_5_org.wav
echo $ORG/clipped_items/som_ad_x_1_org.wav
echo $ORG/clipped_items/som_ot_x_3_org.wav
;;
SetA3a)
echo $ORG/clipped_items/m_ot_x_2_org.wav
echo $ORG/clipped_items/m_po_x_1_org.wav
echo $ORG/clipped_items/m_po_x_2_org.wav
echo $ORG/clipped_items/m_si_x_1_org.wav
echo $ORG/Speech/s_cl_2t_1_org.wav
echo $ORG/Speech/s_cl_ft_3_org.wav
echo $ORG/Speech/s_no_2t_1_org.wav
echo $ORG/Speech/s_no_mt_1_org.wav
echo $ORG/clipped_items/sbm_ms_x_1_org.wav
echo $ORG/clipped_items/sbm_sm_x_2_org.wav
echo $ORG/clipped_items/som_nt_x_1_org.wav
echo $ORG/clipped_items/som_ot_x_2_org.wav
;;
SetA3b)
echo $ORG/clipped_items/m_ot_x_3_org.wav
echo $ORG/clipped_items/m_po_x_3_org.wav
echo $ORG/clipped_items/m_po_x_4_org.wav
echo $ORG/clipped_items/m_si_x_2_org.wav
echo $ORG/Speech/s_cl_2t_2_org.wav
echo $ORG/Speech/s_cl_mt_2_org.wav
echo $ORG/Speech/s_no_2t_2_org.wav
echo $ORG/Speech/s_no_ft_4_org.wav
echo $ORG/clipped_items/sbm_js_x_2_org.wav
echo $ORG/clipped_items/sbm_sm_x_5_org.wav
echo $ORG/clipped_items/som_ot_x_1_org.wav
echo $ORG/clipped_items/som_ot_x_3_org.wav
;;
SetA4a)
echo $ORG/clipped_items/m_ch_x_1_org.wav
echo $ORG/clipped_items/m_ot_x_2_org.wav
```

```
echo $ORG/clipped_items/m_po_x_1_org.wav
echo $ORG/clipped_items/m_si_x_1_org.wav
echo $ORG/Speech/s_cl_2t_1_org.wav
echo $ORG/Speech/s_cl_mt_1_org.wav
echo $ORG/Speech/s_no_2t_1_org.wav
echo $ORG/Speech/s_no_ft_2_org.wav
echo $ORG/clipped_items/sbm_ms_x_1_org.wav
echo $ORG/clipped_items/sbm_sm_x_2_org.wav
echo $ORG/clipped_items/som_fi_x_1_org.wav
echo $ORG/clipped_items/som_nt_x_1_org.wav
;;
SetA4b)
echo $ORG/clipped_items/m_cl_x_1_org.wav
echo $ORG/clipped_items/m_cl_x_2_org.wav
echo $ORG/clipped_items/m_ot_x_1_org.wav
echo $ORG/clipped_items/m_ot_x_3_org.wav
echo $ORG/Speech/s_cl_2t_2_org.wav
echo $ORG/Speech/s_cl_2t_3_org.wav
echo $ORG/Speech/s_no_2t_2_org.wav
echo $ORG/Speech/s_no_ft_1_org.wav
echo $ORG/clipped_items/sbm_sm_x_1_org.wav
echo $ORG/clipped_items/sbm_sm_x_3_org.wav
echo $ORG/clipped_items/som_fi_x_2_org.wav
echo $ORG/clipped_items/som_ot_x_1_org.wav
;;
SetB1a)
echo $ORG/clipped_items/m_ch_x_1_org.wav
echo $ORG/clipped_items/m_cl_x_1_org.wav
echo $ORG/clipped_items/m_po_x_2_org.wav
echo $ORG/clipped_items/m_po_x_3_org.wav
echo $ORG/Speech/s_cl_2t_4_org.wav
echo $ORG/Speech/s_cl_ft_3_org.wav
echo $ORG/Speech/s_no_2t_2_org.wav
echo $ORG/Speech/s_no_ft_2_org.wav
echo $ORG/clipped_items/sbm_js_x_2_org.wav
echo $ORG/clipped_items/sbm_ms_x_1_org.wav
echo $ORG/clipped_items/som_ad_x_1_org.wav
echo $ORG/clipped_items/som_ot_x_4_org.wav
;;
SetB1b)
echo $ORG/clipped_items/m_po_x_4_org.wav
echo $ORG/clipped_items/m_po_x_5_org.wav
echo $ORG/clipped_items/m_po_x_6_org.wav
echo $ORG/clipped_items/m_si_x_2_org.wav
echo $ORG/Speech/s_cl_2t_2_org.wav
echo $ORG/Speech/s_cl_mt_1_org.wav
echo $ORG/Speech/s_no_2t_3_org.wav
echo $ORG/Speech/s_no_ft_4_org.wav
echo $ORG/clipped_items/sbm_sj_x_1_org.wav
echo $ORG/clipped_items/sbm_sm_x_4_org.wav
echo $ORG/clipped_items/som_ot_x_2_org.wav
echo $ORG/clipped_items/som_ot_x_6_org.wav
;;
SetB2a)
echo $ORG/clipped_items/m_cl_x_2_org.wav
echo $ORG/clipped_items/m_ot_x_1_org.wav
echo $ORG/clipped_items/m_ot_x_8_org.wav
echo $ORG/clipped_items/m_ot_x_a_org.wav
echo $ORG/Speech/s_cl_2t_4_org.wav
echo $ORG/Speech/s_cl_2t_5_org.wav
echo $ORG/Speech/s_no_2t_3_org.wav
echo $ORG/Speech/s_no_ft_2_org.wav
echo $ORG/clipped_items/sbm_js_x_1_org.wav
echo $ORG/clipped_items/sbm_sm_x_4_org.wav
echo $ORG/clipped_items/som_fi_x_1_org.wav
echo $ORG/clipped_items/som_ot_x_5_org.wav
;;
SetB2b)
echo $ORG/clipped_items/m_ch_x_1_org.wav
echo $ORG/clipped_items/m_cl_x_1_org.wav
echo $ORG/clipped_items/m_ot_x_9_org.wav
```

```
echo $ORG/clipped_items/m_ot_x_b_org.wav
echo $ORG/Speech/s_cl_2t_3_org.wav
echo $ORG/Speech/s_cl_mt_1_org.wav
echo $ORG/Speech/s_no_ft_1_org.wav
echo $ORG/Speech/s_no_ft_3_org.wav
echo $ORG/clipped_items/sbm_sm_x_1_org.wav
echo $ORG/clipped_items/sbm_sm_x_3_org.wav
echo $ORG/clipped_items/som_fi_x_2_org.wav
echo $ORG/clipped_items/som_ot_x_6_org.wav
;;
SetB3a)
echo $ORG/clipped_items/m_cl_x_1_org.wav
echo $ORG/clipped_items/m_ot_x_1_org.wav
echo $ORG/clipped_items/m_ot_x_5_org.wav
echo $ORG/clipped_items/m_ot_x_7_org.wav
echo $ORG/Speech/s_cl_2t_1_org.wav
echo $ORG/Speech/s_cl_2t_3_org.wav
echo $ORG/Speech/s_no_2t_3_org.wav
echo $ORG/Speech/s_no_ft_4_org.wav
echo $ORG/clipped_items/sbm_js_x_2_org.wav
echo $ORG/clipped_items/sbm_ms_x_1_org.wav
echo $ORG/clipped_items/som_ad_x_1_org.wav
echo $ORG/clipped_items/som_fi_x_2_org.wav
;;
SetB3b)
echo $ORG/clipped_items/m_ch_x_1_org.wav
echo $ORG/clipped_items/m_cl_x_2_org.wav
echo $ORG/clipped_items/m_ot_x_4_org.wav
echo $ORG/clipped_items/m_ot_x_6_org.wav
echo $ORG/Speech/s_cl_2t_2_org.wav
echo $ORG/Speech/s_cl_mt_1_org.wav
echo $ORG/Speech/s_no_ft_3_org.wav
echo $ORG/Speech/s_no_mt_1_org.wav
echo $ORG/clipped_items/sbm_js_x_1_org.wav
echo $ORG/clipped_items/sbm_sm_x_1_org.wav
echo $ORG/clipped_items/som_fi_x_1_org.wav
echo $ORG/clipped_items/som_fi_x_3_org.wav
;;
SetB4a)
echo $ORG/clipped_items/m_ot_x_4_org.wav
echo $ORG/clipped_items/m_ot_x_8_org.wav
echo $ORG/clipped_items/m_po_x_2_org.wav
echo $ORG/clipped_items/m_po_x_7_org.wav
echo $ORG/Speech/s_cl_2t_1_org.wav
echo $ORG/Speech/s_cl_2t_5_org.wav
echo $ORG/Speech/s_no_2t_1_org.wav
echo $ORG/Speech/s_no_ft_3_org.wav
echo $ORG/clipped_items/sbm_js_x_2_org.wav
echo $ORG/clipped_items/sbm_sm_x_2_org.wav
echo $ORG/clipped_items/som_fi_x_3_org.wav
echo $ORG/clipped_items/som_ot_x_5_org.wav
;;
SetB4b)
echo $ORG/clipped_items/m_ot_x_5_org.wav
echo $ORG/clipped_items/m_ot_x_9_org.wav
echo $ORG/clipped_items/m_po_x_3_org.wav
echo $ORG/clipped_items/m_si_x_3_org.wav
echo $ORG/Speech/s_cl_2t_3_org.wav
echo $ORG/Speech/s_cl_mt_2_org.wav
echo $ORG/Speech/s_no_ft_1_org.wav
echo $ORG/Speech/s_no_mt_1_org.wav
echo $ORG/clipped_items/sbm_sm_x_1_org.wav
echo $ORG/clipped_items/sbm_sm_x_6_org.wav
echo $ORG/clipped_items/som_fi_x_4_org.wav
echo $ORG/clipped_items/som_ot_x_3_org.wav
;;
esac
done

#silence
echo $ORG/Silence/silence.wav
```

```
#########################################################################
##
## bin/l_main.sh
##
#########################################################################
#!/bin/bash

case $# in
    0) echo "usage: $0 [A1...B4] [cat|split] [OutputPath]"; exit;;
esac
case $1 in
    [AB][1234]) EXP=$1; shift;;
    *) echo "usage: $0 [A1...B4] [cat|split] [OutputPath]"; exit;;
esac
case $1 in
    split) MODE=split; shift;;
    cat) MODE=cat; shift;;
    *) MODE=cat;;
esac
case $# in
    1) OUT=$1; shift; echo "setting output to $OUT";;
    *) OUT=../output;;
esac

RATELOG=${EXP}_bitrate.txt
CKSUMLOG=${EXP}_cksum.txt
DELAYLOG=${EXP}_delay.txt
SEED=171094
CDIR="aacPlus AMRWB+ ct AAC AMR-WB"
CAT="../preproc/all_cat"
TIM="../preproc/all_cat48s.tim"
rm -f $RATELOG
rm -f $CKSUMLOG

echo
echo low-rate test for experiments $EXP
echo

echo
echo generate random error files
echo
./genfer.csh $SEED .03; cp ferpat_FER.03_SEED$SEED.dat ../ep/epf_3.epf

# do this as global pre-processing
#echo
#echo clip Speech items
#echo
#./clip_silence.sh ../org/ClippedSpeech ../org/Speech/*.wav
echo
echo "get list of files for this experiment (training and testing)"
echo
LIST=`l_files_clipped.sh $EXP`

echo
echo process signals
echo

echo
echo concatenate original signals and add 290ms of silence at end
echo make 24 kHz stereo version
echo make 16, 24 and 48 kHz mono version
echo
./concat.sh $LIST
./anchor.sh -fsout24 ${CAT}48s.wav ${CAT}24s.wav
./anchor.sh -monoout -fsout16 ${CAT}48s.wav ${CAT}16m.wav
./anchor.sh -monoout -fsout24 ${CAT}48s.wav ${CAT}24m.wav
./anchor.sh -monoout ${CAT}48s.wav ${CAT}48m.wav
```

```
./cksum_raw.sh ${CAT}48s.wav >> $CKSUMLOG
echo
echo remove silence file from *.tim file
echo
mv ../preproc/all_cat48s.tim tmp
grep -v silence.wav tmp > ../preproc/all_cat48s.tim
rm tmp
ls ../preproc

exp=$EXP
echo
echo proponent and reference codecs for experiment $exp
echo
for c in $CDIR
do
    (cd ../$c/bin; ./l_encdecdly.sh $exp)
    ls  ../$c/tmp

    echo
    echo "upsample (and split) coded signals, then cksum and check delay"
    echo
    for i in ../$c/tmp/*_${exp}.wav
    do
        base=${i##*/}
        j=../$c/processed/$base
        ./upsamp48.sh $i $j
        rm $i

        if [[ $MODE == cat ]]
        then
            ./cksum_raw.sh $j >> $CKSUMLOG
            delay=`./getdelay.exe -if1 ${CAT}48s.wav -if2 $j | grep delay`
            echo $j $delay >> $DELAYLOG
            if [[ $exp == B3 || $exp == B4 ]]
            then
                #don't have room to store it!
                rm $j
            fi
        else
            tag=${base%.wav}
            ./splitup.sh $j $OUT/exp_${exp}a/${tag} $TIM
            rm $j
            ./cksum_raw.sh $OUT/exp_${exp}a/*${base} >> $CKSUMLOG
        fi
    done
done

echo
echo make anchors
echo
./l_anchors.sh $exp
ls  ../ref/tmp

echo
echo "upsample (and split) anchors, then cksum and check delay"
echo
for i in ../ref/tmp/*_${exp}.wav
do
    base=${i##*/}
    j=../ref/processed/$base
    ./upsamp48.sh $i $j
    #don't delete anchors, since need for other exp's

    if [[ $MODE == cat ]]
    then
        ./cksum_raw.sh $j >> $CKSUMLOG
        delay=`./getdelay.exe -if1 ${CAT}48s.wav -if2 $j | grep delay`
        echo $j $delay >> $DELAYLOG
    else
        tag=${base%.wav}
        ./splitup.sh $j $OUT/exp_${exp}a/${tag} $TIM
```

```
        rm $j
        ./cksum_raw.sh $OUT/exp_${exp}a/*${base} >> $CKSUMLOG
    fi
done

echo
echo cleanup
echo
./clean.sh



######################################################################
##
## bin/l_shift.sh
##
######################################################################
#!/bin/bash

case $# in
    3) test=$1$2; error_rate=$3;;
    *) echo usage: $0 exp set error_rate; exit;;
esac

TEF=./tmp_err_file.epf

case $test in
    B3a) lab[0]=1; lab[1]=5; exp=1;;
    B3b) lab[0]=2; lab[1]=6; exp=1;;
    B4a) lab[0]=3; lab[1]=6; exp=2;;
    B4b) lab[0]=4; lab[1]=8; exp=2;;
    *) echo "unknown experiment: $test"; exit;;

esac

for i in 0 1
do
    let lis=1
    while [ $lis -le 15 ]
    do
        echo "listener $lis lab ${lab[$i]} exp $exp"
        ./shiftcirc.csh $lis ${lab[$i]} $exp ../ep/epf_${error_rate}.epf $TEF
    let lis=lis+1;
    done
done

rm -f $TEF



######################################################################
##
## bin/l_split.sh
##
######################################################################
#!/bin/bash

EXP=$1
SET="a" #split data into sets a and b later

DIRLIST="aacPlus AMRWB+ ct AAC AMR-WB ref"
SPLIT=./splitup.sh
SOUT=../output/exp_${EXP}${SET}
TIM=../preproc/all_cat48s.tim

for dir in $DIRLIST
do
    for file in ../$dir/processed/*_${EXP}.wav
    do
```

```
        base=${file##*/}
        tag=${base%.wav}
        C="$SPLIT $file $SOUT/${tag} $TIM"; echo $C; $C
    done
done



########################################################################
##
## bin/shiftcirc.csh
##
########################################################################
#!/bin/tcsh

if ( $# == 3 ) then
    set offs=$1
    set in=$2
    set out=$3
else if ( $# == 4 ) then
    set s=$1
    set p=$2
    set e=$3
    set in=$4
    set len = `wc -l $in`
    set offs=`echo "($s*12345 + $p*31415 + $e*27183) % $len[1]"|bc`
    echo "offs = $offs"
    exit
else if ( $# == 5 ) then
    set s=$1
    set p=$2
    set e=$3
    set in=$4
    set out=$5

    set len = `wc -l $in`
    set offs=`echo "($s*12345 + $p*31415 + $e*27183) % $len[1]"|bc`

else
    echo "Usage:                                                          v. 0.2"
    echo "$0 <listener_id> <lab_id> <exp_id> <inpat>"
    echo "     returning 'offs'"
    echo "or"
    echo "$0 <offs> <inpat> <outpat>"
    echo "or"
    echo "$0 <listener_id> <lab_id> <exp_id> <inpat> <outpat>"
    echo
    exit
endif

set len = `wc -l $in`
set len=$len[1]

echo $offs
if ( $offs >= $len[1] || $offs < 0 ) then
    echo "offset error: must be >= 0 and < length(inpat)"
    exit 1
endif

#do the shift:
@ offs++
sed -n -e $offs,$len\p $in > $out
@ offs--
if ( $offs > 0 ) then
sed -n -e 1,$offs\p $in >> $out
endif
```

```
######################################################################
##
## bin/splitup.sh
##
######################################################################
#!/bin/bash
if [ $# -ne 3 ]
then
    echo "Usage:                                                                v. 0.1"
    echo "$0 <infile> <proc_tag> <timfile>"
    echo " "
    echo " This script will de-concatenate a 48000 Hz wav input file"
    echo "   <infile>   : ../cand_1/processed/all_cat_cand_1_A1.wav (example)"
    echo "   <proc_tag> : <codec_id>_<exp_id>, e.g.: cand_1_A1"
    echo "   <timfile>  : ../preproc/all_cat.tim (typically)"
    echo " "
    echo " Note: mono-files will be converted to 2-channel files."
    exit 1
fi

#### pathes to executables to be adjusted to local needs ##############################
INFOAUDIO=./InfoAudio.exe
RESAMPAUDIO=./ResampAudio.exe
COPYAUDIO=./CopyAudio.exe
ASTRIP=./astrip.exe
########################################################################################

infile=$1
proctag=$2
timfile=$3

# check for valid WAV header
if ! $INFOAUDIO $infile | grep "WAVE file" >/dev/null; then
    echo "Incorrect wav header in file $infile"
    exit 1
fi
# check sampling rate
if ! $INFOAUDIO $infile | grep "Sampling frequency: 48000 Hz" > /dev/null; then
    echo "Incorrect sampling frequency in file $infile"
    exit 1
fi
# check input channels
if ! $INFOAUDIO $infile | grep "Number of channels: 2" > /dev/null; then
    echo "Incorrect number of channels in file $infile"
    exit 1
fi
# check input channels
nchan=`$INFOAUDIO $infile | grep "^No. channels" | awk '{print $3}'`

# make mono files to 2 channel files
if [ $nchan -eq 1 ]
then
    $COPYAUDIO  --chanA="A" --chanB="A" $infile tmpfile.wav 1>/dev/null 2>/dev/null
    nchan=2
else
    cp $infile tmpfile.wav
fi

# strip the wav header
$COPYAUDIO -F noheader tmpfile.wav catfile.raw  1>/dev/null 2>/dev/null

acclen=1
while read sigfile len
do
    # get length information
    #echo $acclen
    echo $len $sigfile

    # deconcatenate with smooth edges
    #   make the following simplification:
```

```
    #     treat stereo file as mono file and apply Hanning window
    #     of double length
    #   The window size is 100ms=4800samples (rather than 10 ms)
    let wlen=$((4800*$nchan))
    $ASTRIP -smooth -wlen $wlen -sample -start $acclen -n $len catfile.raw tmpfile.raw
    acclen=`echo $acclen + $len | bc`

    # regenerate wav file
    dir=${proctag%/*}
    sig=${sigfile%_org.wav}
    sys=${proctag##*/}
    outfile=$dir/${sig}_${sys}.wav
    $COPYAUDIO -t noheader -P "integer16,0,48000., native, 2, default" tmpfile.raw \
    $outfile 1>/dev/null 2>/dev/null
done < $timfile

# clean up
rm -f catfile.raw tmpfile.raw tmpfile.wav



######################################################################
##
## bin/upsamp48.sh
##
######################################################################
#!/bin/bash
#
# upsamp48.sh according to Section 8 of S4-030677_AMRWB+TestPlan_v0.8.doc
#
# HP20031103 20031104
#

if [ -f ./CopyAudio.exe ]
then
    BIN=.      #we are in ./bin
else
    BIN=../../bin  #we are in ../<codec_dir>/bin
fi
RES=$BIN/ResampAudio.exe
CPY=$BIN/CopyAudio.exe
INF=$BIN/InfoAudio.exe
TMPFILE=upsamp48_tmp_file.wav

ATTEN="0.93"

if [ $# -lt 2 ]; then
    echo "usage: `basename $0` [-fs<fsamp>] <infile> <outfile.wav> "
    echo ""
    exit
fi

INARG=$(($# - 1))
OUTARG=$#
INFILE="${!INARG}"
OUTFILE="${!OUTARG}"
INEXT=${INFILE:(-4):4}

INFS=0
for (( I=1 ; I<$#-1 ; I++ )) ; do
    OPT=${!I}
    if [[ ${OPT:0:3} == "-fs" ]] ; then
        INFS=${OPT:3}
    else
        echo "ERROR: Unknown option "$OPT
        echo ""
        exit 1
    fi
done
```

```
RAWOPTS=""
if [[ $INEXT == ".raw" ]] ; then
    if [[ $INFS == 0 ]] ; then
        echo "ERROR: Option -fs<fsamp> missing"
        echo ""
        exit 1
    fi
    RAWOPTS=" -P integer16,0,"$INFS",native,1,1/32768"
    INCH=1
else
    INFS=0
    if $INF $INFILE | grep "Sampling frequency: 48000 Hz" > /dev/null ; then
        INFS=48000
    fi
    INCH=1
    if $INF $INFILE | grep "Number of channels: 2" > /dev/null ; then
        INCH=2
    fi
fi

if [[ $INCH == 1 ]] ; then
    if [[ $INFS == 48000 ]] ; then
        CPYOPTS=" -cA "$ATTEN"*A -cB "$ATTEN"*A"$RAWOPTS
        echo ""
        echo "UPSAMPLE48: gain-adjustment/upmixing ..."
        echo ""
        echo $CPY $CPYOPTS $INFILE $OUTFILE
        $CPY $CPYOPTS $INFILE $OUTFILE
    else
        RESOPTS=" -s 48000 -g "$ATTEN""$RAWOPTS
        echo ""
        echo "UPSAMPLE48: gain-adjustment/resampling ..."
        echo ""
        echo $RES $RESOPTS $INFILE $TMPFILE
        $RES $RESOPTS $INFILE $TMPFILE
        echo ""
        echo "UPSAMPLE48: upmixing ..."
        echo ""
        CPYOPTS=" -cA A -cB A"
        echo $CPY $CPYOPTS $TMPFILE $OUTFILE
        $CPY $CPYOPTS $TMPFILE $OUTFILE
        rm $TMPFILE
    fi
else
    if [[ $INFS == 48000 ]] ; then
        CPYOPTS=" -g "$ATTEN
        echo ""
        echo "UPSAMPLE48: gain-adjustment ..."
        echo ""
        echo $CPY $CPYOPTS $INFILE $OUTFILE
        $CPY $CPYOPTS $INFILE $OUTFILE
    else
        RESOPTS=" -s 48000 -g "$ATTEN
        echo ""
        echo "UPSAMPLE48: gain-adjustment/resampling ..."
        echo ""
        echo $RES $RESOPTS $INFILE $OUTFILE
        $RES $RESOPTS $INFILE $OUTFILE
    fi
fi

echo ""
echo "UPSAMPLE48: done ..."
echo ""

exit 0



######################################################################
```

```
##
## ct/bin/h_encdecdly.sh
##
#######################################################################
#!/bin/bash
#
# run ct encoder and decoder in course of the 3GPP listening test
#

if [ $# -eq 0 ]; then
   echo "usage: `basename $0` <test condition>"
   echo ""
   exit
fi

EXP=$1  #H[123]
PRO=../tmp/ct
ENCB=./ctenc.exe
DEC=./ctdec.exe
BIT=../tmp/${EXP}.3gp
CPY=../../bin/CopyAudio.exe
CAT=../../h_preproc/all_cat
TMP1=../tmp/tmp1.wav
TMP2=../tmp/tmp2.wav

echo ""
echo "Shell script to run Coding Technologies' ct codec, test $EXP"
echo ""

case $EXP in
    H1) mode=s; rate=32; Fs=48; dly=5122; opts=;;
    H2) mode=s; rate=48; Fs=48; dly=4166; opts=;;
    H3) mode=s; rate=32; Fs=48; dly=5122; opts=;;
    *) echo "unknown test case!"; exit;;
esac

#encode
C="$ENCB -if $CAT$Fs$mode.wav -of $BIT -br ${rate}000 $opts"; echo $C; $C


#decode and delay adjustment
if [[ ! $EXP = H3 ]]
then
    echo clear channel decode
    C="$DEC $BIT $TMP1"; echo $C; $C
    #remove encode-decode delay
    C="$CPY -l ${dly}: $TMP1 ${PRO}_${EXP}.wav"; echo $C; $C
else
    echo errored channel decode
    FER=../../bin/tmp_err_file.epf
    for lab in 3 6
    do
        for fer in 1 3
        do
            for lis in 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15
            do
                echo "lab $lab lis $lis"
                OFILE="${PRO}_FER${fer}_lab${lab}_lis${lis}_${EXP}.wav"
                    ../../bin/shiftcirc.csh $lis $lab 0 \
                    ../../ep/epf_${fer}.epf $FER
                C="$DEC $BIT $TMP1 $FER"; echo $C; $C
                C="$CPY -l ${dly}: $TMP1 $OFILE"; echo $C; $C
            done
        done
    done
fi

rm -f $TMP1 $TMP2 $FER
```

```
######################################################################
##
## ct/bin/l_encdecdly.sh
##
######################################################################
#!/bin/bash
#
# run ct encoder and decoder in course of the 3GPP listening test
#

if [ $# -eq 0 ]; then
   echo "usage: `basename $0` <test condition>"
   echo ""
   exit
fi

EXP=$1
PRO=../tmp/ct
ENCB=./ctenc.exe
DEC=./ctdec.exe
BIT=../tmp/${EXP}.3gp
CPY=../../bin/CopyAudio.exe
CAT=../../preproc/all_cat
TMP1=../tmp/tmp1.wav
TMP2=../tmp/tmp2.wav

echo ""
echo "Shell script to run Coding Technologies' ct codec ..."
echo ""

case $EXP in
    A1) mode=m; rate=14; Fs=48; dly=4166; opts="-m";;
    A2) mode=s; rate=18; Fs=48; dly=5122; opts=;;
    A3) mode=m; rate=24; Fs=48; dly=4166; opts="-m";;
    A4) mode=s; rate=24; Fs=48; dly=5122; opts=;;

    B1) mode=m; rate=14; Fs=16; dly=2081; opts="-m";;
    B2) mode=s; rate=18; Fs=48; dly=5122; opts=;;
    B3) mode=m; rate=14; Fs=48; dly=4166; opts="-m";;
    B4) mode=s; rate=24; Fs=48; dly=5122; opts=;;
    *) echo "unknown test case!"; exit;;
esac

#encode
C="$ENCB -if $CAT$Fs$mode.wav -of $BIT -br ${rate}000 $opts"; echo $C; $C


#decode and delay adjustment
if [[ ! $EXP = B3 && ! $EXP = B4 ]]
then
    echo clear channel decode
    C="$DEC $BIT $TMP1"; echo $C; $C
    #remove encode-decode delay

    C="$CPY -l ${dly}: $TMP1 ${PRO}_${EXP}.wav"; echo $C; $C
else
    echo errored channel decode
    FER=../../bin/tmp_err_file.epf
    for SET in a b
    do
        case $EXP$SET in
            B3a) lab[0]=1; lab[1]=5; exp=1;;
            B3b) lab[0]=2; lab[1]=6; exp=1;;
            B4a) lab[0]=3; lab[1]=7; exp=2;;
            B4b) lab[0]=4; lab[1]=8; exp=2;;
            *) echo "unknown experiment: $EXP$SET"; exit;;
        esac
        for i in 0 1
        do
            for lis in 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15
```

```
            do
                echo "lab ${lab[$i]} lis $lis"
                OFILE="${PRO}_lab${lab[$i]}_lis${lis}_${EXP}.wav"
                ../../bin/shiftcirc.csh $lis ${lab[$i]} $exp \
                    ../../ep/epf_3.epf $FER
                C="$DEC $BIT $TMP1 $FER"; echo $C; $C
                C="$CPY -l ${dly}: $TMP1 $OFILE"; echo $C; $C
            done
        done
    done
fi

rm -f $TMP1 $TMP2
```

3 Annex – Cross-check Results for the Low- and High-Rate tests

3.1 Low-Rate Test

A1

System	CkSum	Len	ARLck	ARLlen	TSck	TSlen
all_cat	OK	OK	1146076914	39612372	1146076914	39612372
aacPlus	OK	OK	2771181602	39583464	2771181602	39583464
AMRWB+	OK	OK	1837649494	39592732	1837649494	39592732
ct	OK	OK	513163774	39583464	513163774	39583464
AAC	OK	OK	792542946	39577456	792542946	39577456
AMR-WB	OK	OK	3727050234	39604912	3727050234	39604912

A2

System	CkSum	Len	ARLck	ARLlen	TSck	TSlen
all_cat	OK	OK	1960793198	39056316	1960793198	39056316
aacPlus	OK	OK	764355835	39013968	764355835	39013968
AMRWB+	OK	OK	3521327666	39038036	3521327666	39038036
ct	OK	OK	2992773628	39022584	2992773628	39022584
AAC	OK	OK	207362698	39015692	207362698	39015692
AMR-WB	OK	OK	3459182581	39048112	3459182581	39048112

A3

System	CkSum	Len	ARLck	ARLlen	TSck	TSlen
all_cat	OK	OK	1840173295	38307312	1840173295	38307312
aacPlus	OK	OK	2486367955	38280936	2486367955	38280936
AMRWB+	OK	OK	3713685785	38287132	3713685785	38287132
ct	OK	OK	1643134772	38280936	1643134772	38280936
AAC	OK	OK	1187385768	38288104	1187385768	38288104
AMR-WB	OK	OK	1301344462	38299312	1301344462	38299312

A4

System	CkSum	Len	ARLck	ARLlen	TSck	TSlen
all_cat	OK	OK	1468825302	39440272	1468825302	39440272
aacPlus	OK	OK	2389564964	39411432	2389564964	39411432
AMRWB+	OK	OK	1369891308	39422036	1369891308	39422036
ct	OK	OK	3483328899	39407608	3483328899	39407608
AAC	OK	OK	1021779504	39405424	1021779504	39405424
AMR-WB	OK	OK	1286611945	39432112	1286611945	39432112

B1

System	CkSum	Len	ARLck	ARLlen	TSck	TSlen
all_cat	OK	OK	530762056	39055024	530762056	39055024
aacPlus	OK	OK	2405483032	39013996	2405483032	39013996
AMRWB+	OK	OK	1970418011	39039772	1970418011	39039772
ct	OK	OK	1701197330	39013996	1701197330	39013996
AAC	OK	OK	1430233497	39024952	1430233497	39024952
AMR-WB	OK	OK	1155417054	39048112	1155417054	39048112

B2

System	CkSum	Len	ARLck	ARLlen	TSck	TSlen
all_cat	OK	OK	3071119152	40188656	3071119152	40188656
aacPlus	OK	OK	4130024015	40144464	4130024015	40144464
AMRWB+	OK	OK	2598746959	40159316	2598746959	40159316
ct	OK	OK	1650173120	40153080	1650173120	40153080
AAC	OK	OK	3772570668	40139168	3772570668	40139168
AMR-WB	OK	OK	2641254664	40180912	2641254664	40180912

B3

System	CkSum	Len	ARLck	ARLlen	TSck	TSlen
all_cat	OK	OK	1103744642	39827408	1103744642	39827408
aacPlus_lab1_lis01	OK	OK	1416904218	39796456	1416904218	39796456
aacPlus_lab1_lis02	OK	OK	3459925155	39796456	3459925155	39796456
aacPlus_lab1_lis03	OK	OK	1905151226	39796456	1905151226	39796456
aacPlus_lab1_lis04	OK	OK	2845734006	39796456	2845734006	39796456
aacPlus_lab1_lis05	OK	OK	1985732233	39796456	1985732233	39796456

aacPlus_lab1_lis06	OK	OK	276203926	39796456	276203926	39796456
aacPlus_lab1_lis07	OK	OK	2135874577	39796456	2135874577	39796456
aacPlus_lab1_lis08	OK	OK	3417039643	39796456	3417039643	39796456
aacPlus_lab1_lis09	OK	OK	55539287	39796456	55539287	39796456
aacPlus_lab1_lis10	OK	OK	2653664753	39796456	2653664753	39796456
aacPlus_lab1_lis11	OK	OK	2787797075	39796456	2787797075	39796456
aacPlus_lab1_lis12	OK	OK	2458899154	39796456	2458899154	39796456
aacPlus_lab1_lis13	OK	OK	224548490	39796456	224548490	39796456
aacPlus_lab1_lis14	OK	OK	1302308034	39796456	1302308034	39796456
aacPlus_lab1_lis15	OK	OK	1569086946	39796456	1569086946	39796456
aacPlus_lab2_lis01	OK	OK	1584819349	39796456	1584819349	39796456
aacPlus_lab2_lis02	OK	OK	651143026	39796456	651143026	39796456
aacPlus_lab2_lis03	OK	OK	265401244	39796456	265401244	39796456
aacPlus_lab2_lis04	OK	OK	1042609652	39796456	1042609652	39796456
aacPlus_lab2_lis05	OK	OK	2189425954	39796456	2189425954	39796456
aacPlus_lab2_lis06	OK	OK	1527659212	39796456	1527659212	39796456
aacPlus_lab2_lis07	OK	OK	1321905776	39796456	1321905776	39796456
aacPlus_lab2_lis08	OK	OK	1645003485	39796456	1645003485	39796456
aacPlus_lab2_lis09	OK	OK	3176295228	39796456	3176295228	39796456
aacPlus_lab2_lis10	OK	OK	898693219	39796456	898693219	39796456
aacPlus_lab2_lis11	OK	OK	917347801	39796456	917347801	39796456
aacPlus_lab2_lis12	OK	OK	2183483230	39796456	2183483230	39796456
aacPlus_lab2_lis13	OK	OK	3342191412	39796456	3342191412	39796456
aacPlus_lab2_lis14	OK	OK	290520075	39796456	290520075	39796456
aacPlus_lab2_lis15	OK	OK	3345241561	39796456	3345241561	39796456
aacPlus_lab5_lis01	OK	OK	240088691	39796456	240088691	39796456
aacPlus_lab5_lis02	OK	OK	1432553819	39796456	1432553819	39796456
aacPlus_lab5_lis03	OK	OK	2176346931	39796456	2176346931	39796456
aacPlus_lab5_lis04	OK	OK	3882387911	39796456	3882387911	39796456
aacPlus_lab5_lis05	OK	OK	2848762614	39796456	2848762614	39796456
aacPlus_lab5_lis06	OK	OK	2625100963	39796456	2625100963	39796456
aacPlus_lab5_lis07	OK	OK	4109565395	39796456	4109565395	39796456
aacPlus_lab5_lis08	OK	OK	2515035495	39796456	2515035495	39796456
aacPlus_lab5_lis09	OK	OK	1722329074	39796456	1722329074	39796456
aacPlus_lab5_lis10	OK	OK	3909719108	39796456	3909719108	39796456
aacPlus_lab5_lis11	OK	OK	2172964488	39796456	2172964488	39796456
aacPlus_lab5_lis12	OK	OK	1445892476	39796456	1445892476	39796456
aacPlus_lab5_lis13	OK	OK	971735068	39796456	971735068	39796456
aacPlus_lab5_lis14	OK	OK	463373900	39796456	463373900	39796456
aacPlus_lab5_lis15	OK	OK	698800540	39796456	698800540	39796456
aacPlus_lab6_lis01	OK	OK	3983778988	39796456	3983778988	39796456
aacPlus_lab6_lis02	OK	OK	1151351265	39796456	1151351265	39796456
aacPlus_lab6_lis03	OK	OK	2364138058	39796456	2364138058	39796456
aacPlus_lab6_lis04	OK	OK	4075943863	39796456	4075943863	39796456
aacPlus_lab6_lis05	OK	OK	1854325752	39796456	1854325752	39796456
aacPlus_lab6_lis06	OK	OK	2529839842	39796456	2529839842	39796456
aacPlus_lab6_lis07	OK	OK	1043362294	39796456	1043362294	39796456
aacPlus_lab6_lis08	OK	OK	2314779080	39796456	2314779080	39796456
aacPlus_lab6_lis09	OK	OK	2000694610	39796456	2000694610	39796456
aacPlus_lab6_lis10	OK	OK	683792710	39796456	683792710	39796456
aacPlus_lab6_lis11	OK	OK	1433604716	39796456	1433604716	39796456
aacPlus_lab6_lis12	OK	OK	1649305870	39796456	1649305870	39796456
aacPlus_lab6_lis13	OK	OK	1582376569	39796456	1582376569	39796456
aacPlus_lab6_lis14	OK	OK	3467568706	39796456	3467568706	39796456
aacPlus_lab6_lis15	OK	OK	726016217	39796456	726016217	39796456
AMRWB+_lab1_lis01	OK	OK	2164320714	39807772	2164320714	39807772
AMRWB+_lab1_lis02	OK	OK	1628484623	39807772	1628484623	39807772
AMRWB+_lab1_lis03	OK	OK	2546237325	39807772	2546237325	39807772
AMRWB+_lab1_lis04	OK	OK	1916419325	39807772	1916419325	39807772
AMRWB+_lab1_lis05	OK	OK	3146263597	39807772	3146263597	39807772
AMRWB+_lab1_lis06	OK	OK	3136905286	39807772	3136905286	39807772
AMRWB+_lab1_lis07	OK	OK	173927075	39807772	173927075	39807772
AMRWB+_lab1_lis08	OK	OK	1600828280	39807772	1600828280	39807772
AMRWB+_lab1_lis09	OK	OK	3566583077	39807772	3566583077	39807772
AMRWB+_lab1_lis10	OK	OK	1408689695	39807772	1408689695	39807772
AMRWB+_lab1_lis11	OK	OK	3143263520	39807772	3143263520	39807772
AMRWB+_lab1_lis12	OK	OK	4272890029	39807772	4272890029	39807772
AMRWB+_lab1_lis13	OK	OK	1915170318	39807772	1915170318	39807772
AMRWB+_lab1_lis14	OK	OK	674184728	39807772	674184728	39807772
AMRWB+_lab1_lis15	OK	OK	3840663115	39807772	3840663115	39807772
AMRWB+_lab2_lis01	OK	OK	3999638153	39807772	3999638153	39807772

AMRWB+_lab2_lis02	OK	OK	833997619	39807772	833997619	39807772
AMRWB+_lab2_lis03	OK	OK	1783963691	39807772	1783963691	39807772
AMRWB+_lab2_lis04	OK	OK	3698727602	39807772	3698727602	39807772
AMRWB+_lab2_lis05	OK	OK	4076969706	39807772	4076969706	39807772
AMRWB+_lab2_lis06	OK	OK	180634613	39807772	180634613	39807772
AMRWB+_lab2_lis07	OK	OK	1398095162	39807772	1398095162	39807772
AMRWB+_lab2_lis08	OK	OK	4093081078	39807772	4093081078	39807772
AMRWB+_lab2_lis09	OK	OK	3447166605	39807772	3447166605	39807772
AMRWB+_lab2_lis10	OK	OK	3166584517	39807772	3166584517	39807772
AMRWB+_lab2_lis11	OK	OK	1905702890	39807772	1905702890	39807772
AMRWB+_lab2_lis12	OK	OK	1930551463	39807772	1930551463	39807772
AMRWB+_lab2_lis13	OK	OK	2201876844	39807772	2201876844	39807772
AMRWB+_lab2_lis14	OK	OK	3967436854	39807772	3967436854	39807772
AMRWB+_lab2_lis15	OK	OK	2364393673	39807772	2364393673	39807772
AMRWB+_lab5_lis01	OK	OK	3534531947	39807772	3534531947	39807772
AMRWB+_lab5_lis02	OK	OK	8871255	39807772	8871255	39807772
AMRWB+_lab5_lis03	OK	OK	3797408005	39807772	3797408005	39807772
AMRWB+_lab5_lis04	OK	OK	589738035	39807772	589738035	39807772
AMRWB+_lab5_lis05	OK	OK	1139681052	39807772	1139681052	39807772
AMRWB+_lab5_lis06	OK	OK	3636481491	39807772	3636481491	39807772
AMRWB+_lab5_lis07	OK	OK	2860584612	39807772	2860584612	39807772
AMRWB+_lab5_lis08	OK	OK	702890318	39807772	702890318	39807772
AMRWB+_lab5_lis09	OK	OK	2380323786	39807772	2380323786	39807772
AMRWB+_lab5_lis10	OK	OK	645288112	39807772	645288112	39807772
AMRWB+_lab5_lis11	OK	OK	1402195555	39807772	1402195555	39807772
AMRWB+_lab5_lis12	OK	OK	2696599415	39807772	2696599415	39807772
AMRWB+_lab5_lis13	OK	OK	627763665	39807772	627763665	39807772
AMRWB+_lab5_lis14	OK	OK	1733401416	39807772	1733401416	39807772
AMRWB+_lab5_lis15	OK	OK	4025305652	39807772	4025305652	39807772
AMRWB+_lab6_lis01	OK	OK	893271618	39807772	893271618	39807772
AMRWB+_lab6_lis02	OK	OK	210821095	39807772	210821095	39807772
AMRWB+_lab6_lis03	OK	OK	3479345141	39807772	3479345141	39807772
AMRWB+_lab6_lis04	OK	OK	2586022934	39807772	2586022934	39807772
AMRWB+_lab6_lis05	OK	OK	1723991083	39807772	1723991083	39807772
AMRWB+_lab6_lis06	OK	OK	593910571	39807772	593910571	39807772
AMRWB+_lab6_lis07	OK	OK	3887768852	39807772	3887768852	39807772
AMRWB+_lab6_lis08	OK	OK	3095030862	39807772	3095030862	39807772
AMRWB+_lab6_lis09	OK	OK	3821959301	39807772	3821959301	39807772
AMRWB+_lab6_lis10	OK	OK	878066763	39807772	878066763	39807772
AMRWB+_lab6_lis11	OK	OK	2933218424	39807772	2933218424	39807772
AMRWB+_lab6_lis12	OK	OK	1258272945	39807772	1258272945	39807772
AMRWB+_lab6_lis13	OK	OK	4104680534	39807772	4104680534	39807772
AMRWB+_lab6_lis14	OK	OK	2086528817	39807772	2086528817	39807772
AMRWB+_lab6_lis15	OK	OK	3791616763	39807772	3791616763	39807772
ct_lab1_lis01	OK	OK	3601093761	39796456	3601093761	39796456
ct_lab1_lis02	OK	OK	2970541484	39796456	2970541484	39796456
ct_lab1_lis03	OK	OK	2462910429	39796456	2462910429	39796456
ct_lab1_lis04	OK	OK	1065467759	39796456	1065467759	39796456
ct_lab1_lis05	OK	OK	3679143247	39796456	3679143247	39796456
ct_lab1_lis06	OK	OK	3877526411	39796456	3877526411	39796456
ct_lab1_lis07	OK	OK	2765023385	39796456	2765023385	39796456
ct_lab1_lis08	OK	OK	2279928051	39796456	2279928051	39796456
ct_lab1_lis09	OK	OK	3816263374	39796456	3816263374	39796456
ct_lab1_lis10	OK	OK	3094440408	39796456	3094440408	39796456
ct_lab1_lis11	OK	OK	4119449757	39796456	4119449757	39796456
ct_lab1_lis12	OK	OK	490945008	39796456	490945008	39796456
ct_lab1_lis13	OK	OK	3063929492	39796456	3063929492	39796456
ct_lab1_lis14	OK	OK	502446879	39796456	502446879	39796456
ct_lab1_lis15	OK	OK	1395303724	39796456	1395303724	39796456
ct_lab2_lis01	OK	OK	1183497840	39796456	1183497840	39796456
ct_lab2_lis02	OK	OK	156843128	39796456	156843128	39796456
ct_lab2_lis03	OK	OK	3641314656	39796456	3641314656	39796456
ct_lab2_lis04	OK	OK	3561312400	39796456	3561312400	39796456
ct_lab2_lis05	OK	OK	2007958690	39796456	2007958690	39796456
ct_lab2_lis06	OK	OK	543883178	39796456	543883178	39796456
ct_lab2_lis07	OK	OK	3591914985	39796456	3591914985	39796456
ct_lab2_lis08	OK	OK	3742524415	39796456	3742524415	39796456
ct_lab2_lis09	OK	OK	988384204	39796456	988384204	39796456
ct_lab2_lis10	OK	OK	3214009036	39796456	3214009036	39796456
ct_lab2_lis11	OK	OK	167806598	39796456	167806598	39796456
ct_lab2_lis12	OK	OK	1433884647	39796456	1433884647	39796456

ct_lab2_lis13	OK	OK	3614739886	39796456	3614739886	39796456
ct_lab2_lis14	OK	OK	642233934	39796456	642233934	39796456
ct_lab2_lis15	OK	OK	82471528	39796456	82471528	39796456
ct_lab5_lis01	OK	OK	3092797301	39796456	3092797301	39796456
ct_lab5_lis02	OK	OK	90893025	39796456	90893025	39796456
ct_lab5_lis03	OK	OK	1028169494	39796456	1028169494	39796456
ct_lab5_lis04	OK	OK	3141792857	39796456	3141792857	39796456
ct_lab5_lis05	OK	OK	3303529501	39796456	3303529501	39796456
ct_lab5_lis06	OK	OK	266075019	39796456	266075019	39796456
ct_lab5_lis07	OK	OK	115249358	39796456	115249358	39796456
ct_lab5_lis08	OK	OK	737223240	39796456	737223240	39796456
ct_lab5_lis09	OK	OK	1813068793	39796456	1813068793	39796456
ct_lab5_lis10	OK	OK	639290305	39796456	639290305	39796456
ct_lab5_lis11	OK	OK	3550398455	39796456	3550398455	39796456
ct_lab5_lis12	OK	OK	3983865594	39796456	3983865594	39796456
ct_lab5_lis13	OK	OK	2506049324	39796456	2506049324	39796456
ct_lab5_lis14	OK	OK	3237113500	39796456	3237113500	39796456
ct_lab5_lis15	OK	OK	665222709	39796456	665222709	39796456
ct_lab6_lis01	OK	OK	3620724370	39796456	3620724370	39796456
ct_lab6_lis02	OK	OK	2916850221	39796456	2916850221	39796456
ct_lab6_lis03	OK	OK	3120880642	39796456	3120880642	39796456
ct_lab6_lis04	OK	OK	2986195757	39796456	2986195757	39796456
ct_lab6_lis05	OK	OK	3643162478	39796456	3643162478	39796456
ct_lab6_lis06	OK	OK	2099387380	39796456	2099387380	39796456
ct_lab6_lis07	OK	OK	4283196340	39796456	4283196340	39796456
ct_lab6_lis08	OK	OK	2353311204	39796456	2353311204	39796456
ct_lab6_lis09	OK	OK	2731736697	39796456	2731736697	39796456
ct_lab6_lis10	OK	OK	3621376891	39796456	3621376891	39796456
ct_lab6_lis11	OK	OK	2053769662	39796456	2053769662	39796456
ct_lab6_lis12	OK	OK	2833906502	39796456	2833906502	39796456
ct_lab6_lis13	OK	OK	1693294676	39796456	1693294676	39796456
ct_lab6_lis14	OK	OK	3462847667	39796456	3462847667	39796456
ct_lab6_lis15	OK	OK	1135321601	39796456	1135321601	39796456
AAC_lab1_lis01	OK	OK	202707342	39798640	202707342	39798640
AAC_lab1_lis02	OK	OK	2294614148	39798640	2294614148	39798640
AAC_lab1_lis03	OK	OK	1978167095	39798640	1978167095	39798640
AAC_lab1_lis04	OK	OK	1747646803	39798640	1747646803	39798640
AAC_lab1_lis05	OK	OK	2762463698	39798640	2762463698	39798640
AAC_lab1_lis06	OK	OK	2680202738	39798640	2680202738	39798640
AAC_lab1_lis07	OK	OK	3128443152	39798640	3128443152	39798640
AAC_lab1_lis08	OK	OK	3217048897	39798640	3217048897	39798640
AAC_lab1_lis09	OK	OK	636921891	39798640	636921891	39798640
AAC_lab1_lis10	OK	OK	2215335688	39798640	2215335688	39798640
AAC_lab1_lis11	OK	OK	2006267404	39798640	2006267404	39798640
AAC_lab1_lis12	OK	OK	3475205253	39798640	3475205253	39798640
AAC_lab1_lis13	OK	OK	2377760943	39798640	2377760943	39798640
AAC_lab1_lis14	OK	OK	3217146979	39798640	3217146979	39798640
AAC_lab1_lis15	OK	OK	2500808852	39798640	2500808852	39798640
AAC_lab2_lis01	OK	OK	393455698	39798640	393455698	39798640
AAC_lab2_lis02	OK	OK	145368680	39798640	145368680	39798640
AAC_lab2_lis03	OK	OK	1203283166	39798640	1203283166	39798640
AAC_lab2_lis04	OK	OK	3739583373	39798640	3739583373	39798640
AAC_lab2_lis05	OK	OK	1412116591	39798640	1412116591	39798640
AAC_lab2_lis06	OK	OK	3059253608	39798640	3059253608	39798640
AAC_lab2_lis07	OK	OK	2283417266	39798640	2283417266	39798640
AAC_lab2_lis08	OK	OK	2783075238	39798640	2783075238	39798640
AAC_lab2_lis09	OK	OK	1856554874	39798640	1856554874	39798640
AAC_lab2_lis10	OK	OK	3830005473	39798640	3830005473	39798640
AAC_lab2_lis11	OK	OK	2667403657	39798640	2667403657	39798640
AAC_lab2_lis12	OK	OK	1298807278	39798640	1298807278	39798640
AAC_lab2_lis13	OK	OK	3023048836	39798640	3023048836	39798640
AAC_lab2_lis14	OK	OK	918962013	39798640	918962013	39798640
AAC_lab2_lis15	OK	OK	2021720854	39798640	2021720854	39798640
AAC_lab5_lis01	OK	OK	1617505785	39798640	1617505785	39798640
AAC_lab5_lis02	OK	OK	4048232237	39798640	4048232237	39798640
AAC_lab5_lis03	OK	OK	125949045	39798640	125949045	39798640
AAC_lab5_lis04	OK	OK	2963637084	39798640	2963637084	39798640
AAC_lab5_lis05	OK	OK	1262094602	39798640	1262094602	39798640
AAC_lab5_lis06	OK	OK	3421286819	39798640	3421286819	39798640
AAC_lab5_lis07	OK	OK	97416968	39798640	97416968	39798640
AAC_lab5_lis08	OK	OK	3133078860	39798640	3133078860	39798640

AAC_lab5_lis09	OK	OK	908866063	39798640	908866063	39798640	
AAC_lab5_lis10	OK	OK	732466962	39798640	732466962	39798640	
AAC_lab5_lis11	OK	OK	1439086655	39798640	1439086655	39798640	
AAC_lab5_lis12	OK	OK	3744128812	39798640	3744128812	39798640	
AAC_lab5_lis13	OK	OK	2735992902	39798640	2735992902	39798640	
AAC_lab5_lis14	OK	OK	3715483387	39798640	3715483387	39798640	
AAC_lab5_lis15	OK	OK	2991712479	39798640	2991712479	39798640	
AAC_lab6_lis01	OK	OK	215433099	39798640	215433099	39798640	
AAC_lab6_lis02	OK	OK	2866346288	39798640	2866346288	39798640	
AAC_lab6_lis03	OK	OK	797462337	39798640	797462337	39798640	
AAC_lab6_lis04	OK	OK	1415323239	39798640	1415323239	39798640	
AAC_lab6_lis05	OK	OK	4740137	39798640	4740137	39798640	
AAC_lab6_lis06	OK	OK	865003695	39798640	865003695	39798640	
AAC_lab6_lis07	OK	OK	3109944245	39798640	3109944245	39798640	
AAC_lab6_lis08	OK	OK	3603581016	39798640	3603581016	39798640	
AAC_lab6_lis09	OK	OK	227569128	39798640	227569128	39798640	
AAC_lab6_lis10	OK	OK	1982931731	39798640	1982931731	39798640	
AAC_lab6_lis11	OK	OK	2578882430	39798640	2578882430	39798640	
AAC_lab6_lis12	OK	OK	3329980258	39798640	3329980258	39798640	
AAC_lab6_lis13	OK	OK	6815871	39798640	6815871	39798640	
AAC_lab6_lis14	OK	OK	1056800734	39798640	1056800734	39798640	
AAC_lab6_lis15	OK	OK	2566590667	39798640	2566590667	39798640	
AMR-WB_lab1_lis01	OK	OK	4187272821	39819952	4187272821	39819952	
AMR-WB_lab1_lis02	OK	OK	3314063277	39819952	3314063277	39819952	
AMR-WB_lab1_lis03	OK	OK	2727410506	39819952	2727410506	39819952	
AMR-WB_lab1_lis04	OK	OK	1017853144	39819952	1017853144	39819952	
AMR-WB_lab1_lis05	OK	OK	84330523	39819952	84330523	39819952	
AMR-WB_lab1_lis06	OK	OK	2006441794	39819952	2006441794	39819952	
AMR-WB_lab1_lis07	OK	OK	1395386596	39819952	1395386596	39819952	
AMR-WB_lab1_lis08	OK	OK	3588699150	39819952	3588699150	39819952	
AMR-WB_lab1_lis09	OK	OK	1000283492	39819952	1000283492	39819952	
AMR-WB_lab1_lis10	OK	OK	1536367353	39819952	1536367353	39819952	
AMR-WB_lab1_lis11	OK	OK	482610192	39819952	482610192	39819952	
AMR-WB_lab1_lis12	OK	OK	3788112291	39819952	3788112291	39819952	
AMR-WB_lab1_lis13	OK	OK	3129660701	39819952	3129660701	39819952	
AMR-WB_lab1_lis14	OK	OK	2286372067	39819952	2286372067	39819952	
AMR-WB_lab1_lis15	OK	OK	2038508564	39819952	2038508564	39819952	
AMR-WB_lab2_lis01	OK	OK	3966289677	39819952	3966289677	39819952	
AMR-WB_lab2_lis02	OK	OK	84175265	39819952	84175265	39819952	
AMR-WB_lab2_lis03	OK	OK	2322275250	39819952	2322275250	39819952	
AMR-WB_lab2_lis04	OK	OK	774583565	39819952	774583565	39819952	
AMR-WB_lab2_lis05	OK	OK	3247182853	39819952	3247182853	39819952	
AMR-WB_lab2_lis06	OK	OK	1294649845	39819952	1294649845	39819952	
AMR-WB_lab2_lis07	OK	OK	2335745076	39819952	2335745076	39819952	
AMR-WB_lab2_lis08	OK	OK	833061591	39819952	833061591	39819952	
AMR-WB_lab2_lis09	OK	OK	249838818	39819952	249838818	39819952	
AMR-WB_lab2_lis10	OK	OK	3244004637	39819952	3244004637	39819952	
AMR-WB_lab2_lis11	OK	OK	2859941383	39819952	2859941383	39819952	
AMR-WB_lab2_lis12	OK	OK	608637432	39819952	608637432	39819952	
AMR-WB_lab2_lis13	OK	OK	3340379323	39819952	3340379323	39819952	
AMR-WB_lab2_lis14	OK	OK	3047882127	39819952	3047882127	39819952	
AMR-WB_lab2_lis15	OK	OK	2755554162	39819952	2755554162	39819952	
AMR-WB_lab5_lis01	OK	OK	352742622	39819952	352742622	39819952	
AMR-WB_lab5_lis02	OK	OK	3089461248	39819952	3089461248	39819952	
AMR-WB_lab5_lis03	OK	OK	1078105460	39819952	1078105460	39819952	
AMR-WB_lab5_lis04	OK	OK	2291794590	39819952	2291794590	39819952	
AMR-WB_lab5_lis05	OK	OK	2588694732	39819952	2588694732	39819952	
AMR-WB_lab5_lis06	OK	OK	994557527	39819952	994557527	39819952	
AMR-WB_lab5_lis07	OK	OK	4063932876	39819952	4063932876	39819952	
AMR-WB_lab5_lis08	OK	OK	3904471126	39819952	3904471126	39819952	
AMR-WB_lab5_lis09	OK	OK	2322098400	39819952	2322098400	39819952	
AMR-WB_lab5_lis10	OK	OK	3812322077	39819952	3812322077	39819952	
AMR-WB_lab5_lis11	OK	OK	2449212688	39819952	2449212688	39819952	
AMR-WB_lab5_lis12	OK	OK	2315312258	39819952	2315312258	39819952	
AMR-WB_lab5_lis13	OK	OK	1593960324	39819952	1593960324	39819952	
AMR-WB_lab5_lis14	OK	OK	1513178890	39819952	1513178890	39819952	
AMR-WB_lab5_lis15	OK	OK	899266184	39819952	899266184	39819952	
AMR-WB_lab6_lis01	OK	OK	1262658653	39819952	1262658653	39819952	
AMR-WB_lab6_lis02	OK	OK	1478201584	39819952	1478201584	39819952	
AMR-WB_lab6_lis03	OK	OK	483551829	39819952	483551829	39819952	
AMR-WB_lab6_lis04	OK	OK	585389590	39819952	585389590	39819952	

AMR-WB_lab6_lis05	OK	OK	1918016820	39819952	1918016820	39819952
AMR-WB_lab6_lis06	OK	OK	3762675048	39819952	3762675048	39819952
AMR-WB_lab6_lis07	OK	OK	1258800843	39819952	1258800843	39819952
AMR-WB_lab6_lis08	OK	OK	3152107389	39819952	3152107389	39819952
AMR-WB_lab6_lis09	OK	OK	2190605476	39819952	2190605476	39819952
AMR-WB_lab6_lis10	OK	OK	1774890623	39819952	1774890623	39819952
AMR-WB_lab6_lis11	OK	OK	2606609120	39819952	2606609120	39819952
AMR-WB_lab6_lis12	OK	OK	1085864280	39819952	1085864280	39819952
AMR-WB_lab6_lis13	OK	OK	4236384757	39819952	4236384757	39819952
AMR-WB_lab6_lis14	OK	OK	1799055946	39819952	1799055946	39819952
AMR-WB_lab6_lis15	OK	OK	3215511504	39819952	3215511504	39819952
hidref	OK	OK	3054436269	39827408	3054436269	39827408
lp3500	OK	OK	3041777297	39827408	3041777297	39827408
lp7000	OK	OK	3474530659	39827408	3474530659	39827408
opref	OK	OK	3054436269	39827408	3054436269	39827408

B4

System	CkSum	Len	ARLck	ARLlen	TSck	TSlen
all_cat	OK	OK	613201085	39025080	613201085	39025080
aacPlus_lab3_lis01	OK	OK	3658374069	38993640	3658374069	38993640
aacPlus_lab3_lis02	OK	OK	2680453761	38993640	2680453761	38993640
aacPlus_lab3_lis03	OK	OK	477268611	38993640	477268611	38993640
aacPlus_lab3_lis04	OK	OK	365436264	38993640	365436264	38993640
aacPlus_lab3_lis05	OK	OK	4161046340	38993640	4161046340	38993640
aacPlus_lab3_lis06	OK	OK	1109004917	38993640	1109004917	38993640
aacPlus_lab3_lis07	OK	OK	923650024	38993640	923650024	38993640
aacPlus_lab3_lis08	OK	OK	3906742244	38993640	3906742244	38993640
aacPlus_lab3_lis09	OK	OK	429183542	38993640	429183542	38993640
aacPlus_lab3_lis10	OK	OK	1016608038	38993640	1016608038	38993640
aacPlus_lab3_lis11	OK	OK	656587746	39001832	656587746	39001832
aacPlus_lab3_lis12	OK	OK	3719203633	38993640	3719203633	38993640
aacPlus_lab3_lis13	OK	OK	2327012369	38993640	2327012369	38993640
aacPlus_lab3_lis14	OK	OK	2699178340	38993640	2699178340	38993640
aacPlus_lab3_lis15	OK	OK	58137927	38993640	58137927	38993640
aacPlus_lab4_lis01	OK	OK	872479463	38993640	872479463	38993640
aacPlus_lab4_lis02	OK	OK	2233223637	38993640	2233223637	38993640
aacPlus_lab4_lis03	OK	OK	1965069247	38993640	1965069247	38993640
aacPlus_lab4_lis04	OK	OK	2920697888	39001832	2920697888	39001832
aacPlus_lab4_lis05	OK	OK	2436293055	38993640	2436293055	38993640
aacPlus_lab4_lis06	OK	OK	547950481	38993640	547950481	38993640
aacPlus_lab4_lis07	OK	OK	3953080291	38993640	3953080291	38993640
aacPlus_lab4_lis08	OK	OK	2101525211	39001832	2101525211	39001832
aacPlus_lab4_lis09	OK	OK	911523624	38993640	911523624	38993640
aacPlus_lab4_lis10	OK	OK	837417425	38993640	837417425	38993640
aacPlus_lab4_lis11	OK	OK	2314033837	38993640	2314033837	38993640
aacPlus_lab4_lis12	OK	OK	3063627327	38993640	3063627327	38993640
aacPlus_lab4_lis13	OK	OK	3080552670	38993640	3080552670	38993640
aacPlus_lab4_lis14	OK	OK	4164434681	38993640	4164434681	38993640
aacPlus_lab4_lis15	OK	OK	2227707407	38993640	2227707407	38993640
aacPlus_lab7_lis01	OK	OK	3503899590	38993640	3503899590	38993640
aacPlus_lab7_lis02	OK	OK	757667652	38993640	757667652	38993640
aacPlus_lab7_lis03	OK	OK	1721488286	38993640	1721488286	38993640
aacPlus_lab7_lis04	OK	OK	1732321740	38993640	1732321740	38993640
aacPlus_lab7_lis05	OK	OK	262969553	38993640	262969553	38993640
aacPlus_lab7_lis06	OK	OK	480907629	38993640	480907629	38993640
aacPlus_lab7_lis07	OK	OK	3287417742	38993640	3287417742	38993640
aacPlus_lab7_lis08	OK	OK	3186158390	38993640	3186158390	38993640
aacPlus_lab7_lis09	OK	OK	229668086	38993640	229668086	38993640
aacPlus_lab7_lis10	OK	OK	3543845161	38993640	3543845161	38993640
aacPlus_lab7_lis11	OK	OK	311500708	38993640	311500708	38993640
aacPlus_lab7_lis12	OK	OK	1070658123	38993640	1070658123	38993640
aacPlus_lab7_lis13	OK	OK	1071436153	38993640	1071436153	38993640
aacPlus_lab7_lis14	OK	OK	11523989	38993640	11523989	38993640
aacPlus_lab7_lis15	OK	OK	1906468181	38993640	1906468181	38993640
aacPlus_lab8_lis01	OK	OK	2294101789	38993640	2294101789	38993640
aacPlus_lab8_lis02	OK	OK	3999628929	38993640	3999628929	38993640
aacPlus_lab8_lis03	OK	OK	784030153	38993640	784030153	38993640
aacPlus_lab8_lis04	OK	OK	3714309919	38993640	3714309919	38993640
aacPlus_lab8_lis05	OK	OK	4044023774	38993640	4044023774	38993640
aacPlus_lab8_lis06	OK	OK	3991540549	38993640	3991540549	38993640

aacPlus_lab8_lis07	OK	OK	3585754242	38993640	3585754242	38993640
aacPlus_lab8_lis08	OK	OK	2168205883	39001832	2168205883	39001832
aacPlus_lab8_lis09	OK	OK	3920861820	38993640	3920861820	38993640
aacPlus_lab8_lis10	OK	OK	2889066636	38993640	2889066636	38993640
aacPlus_lab8_lis11	OK	OK	1040228491	38993640	1040228491	38993640
aacPlus_lab8_lis12	OK	OK	3018011837	38993640	3018011837	38993640
aacPlus_lab8_lis13	OK	OK	3380354992	38993640	3380354992	38993640
aacPlus_lab8_lis14	OK	OK	1113547421	38993640	1113547421	38993640
aacPlus_lab8_lis15	OK	OK	3339173775	38993640	3339173775	38993640
AMRWB+_lab3_lis01	OK	OK	2537762488	39007316	2537762488	39007316
AMRWB+_lab3_lis02	OK	OK	3339910187	39007316	3339910187	39007316
AMRWB+_lab3_lis03	OK	OK	1178433251	39007316	1178433251	39007316
AMRWB+_lab3_lis04	OK	OK	3375975606	39007316	3375975606	39007316
AMRWB+_lab3_lis05	OK	OK	3969596723	39007316	3969596723	39007316
AMRWB+_lab3_lis06	OK	OK	4015132388	39007316	4015132388	39007316
AMRWB+_lab3_lis07	OK	OK	1641647237	39007316	1641647237	39007316
AMRWB+_lab3_lis08	OK	OK	4091442629	39007316	4091442629	39007316
AMRWB+_lab3_lis09	OK	OK	911362316	39007316	911362316	39007316
AMRWB+_lab3_lis10	OK	OK	592615319	39007316	592615319	39007316
AMRWB+_lab3_lis11	OK	OK	2161040289	39007316	2161040289	39007316
AMRWB+_lab3_lis12	OK	OK	2079006524	39007316	2079006524	39007316
AMRWB+_lab3_lis13	OK	OK	1295945261	39007316	1295945261	39007316
AMRWB+_lab3_lis14	OK	OK	3905662967	39007316	3905662967	39007316
AMRWB+_lab3_lis15	OK	OK	510859158	39007316	510859158	39007316
AMRWB+_lab4_lis01	OK	OK	2259923647	39007316	2259923647	39007316
AMRWB+_lab4_lis02	OK	OK	678989870	39007316	678989870	39007316
AMRWB+_lab4_lis03	OK	OK	1323496816	39007316	1323496816	39007316
AMRWB+_lab4_lis04	OK	OK	3326076089	39007316	3326076089	39007316
AMRWB+_lab4_lis05	OK	OK	1069486926	39007316	1069486926	39007316
AMRWB+_lab4_lis06	OK	OK	3821171788	39007316	3821171788	39007316
AMRWB+_lab4_lis07	OK	OK	3898108637	39007316	3898108637	39007316
AMRWB+_lab4_lis08	OK	OK	1745925328	39007316	1745925328	39007316
AMRWB+_lab4_lis09	OK	OK	2271712631	39007316	2271712631	39007316
AMRWB+_lab4_lis10	OK	OK	1757924412	39007316	1757924412	39007316
AMRWB+_lab4_lis11	OK	OK	275040682	39007316	275040682	39007316
AMRWB+_lab4_lis12	OK	OK	309357359	39007316	309357359	39007316
AMRWB+_lab4_lis13	OK	OK	3747223034	39007316	3747223034	39007316
AMRWB+_lab4_lis14	OK	OK	1847966406	39007316	1847966406	39007316
AMRWB+_lab4_lis15	OK	OK	109667778	39007316	109667778	39007316
AMRWB+_lab7_lis01	OK	OK	3550613199	39007316	3550613199	39007316
AMRWB+_lab7_lis02	OK	OK	2223948519	39007316	2223948519	39007316
AMRWB+_lab7_lis03	OK	OK	1890129962	39007316	1890129962	39007316
AMRWB+_lab7_lis04	OK	OK	1394004759	39007316	1394004759	39007316
AMRWB+_lab7_lis05	OK	OK	4173706058	39007316	4173706058	39007316
AMRWB+_lab7_lis06	OK	OK	1687778165	39007316	1687778165	39007316
AMRWB+_lab7_lis07	OK	OK	346616558	39007316	346616558	39007316
AMRWB+_lab7_lis08	OK	OK	2708444254	39007316	2708444254	39007316
AMRWB+_lab7_lis09	OK	OK	1886442888	39007316	1886442888	39007316
AMRWB+_lab7_lis10	OK	OK	76874335	39007316	76874335	39007316
AMRWB+_lab7_lis11	OK	OK	1369811193	39007316	1369811193	39007316
AMRWB+_lab7_lis12	OK	OK	769310641	39007316	769310641	39007316
AMRWB+_lab7_lis13	OK	OK	2054383968	39007316	2054383968	39007316
AMRWB+_lab7_lis14	OK	OK	1718153841	39007316	1718153841	39007316
AMRWB+_lab7_lis15	OK	OK	4186701300	39007316	4186701300	39007316
AMRWB+_lab8_lis01	OK	OK	1057127583	39007316	1057127583	39007316
AMRWB+_lab8_lis02	OK	OK	4036393036	39007316	4036393036	39007316
AMRWB+_lab8_lis03	OK	OK	324353872	39007316	324353872	39007316
AMRWB+_lab8_lis04	OK	OK	2825661108	39007316	2825661108	39007316
AMRWB+_lab8_lis05	OK	OK	2977859262	39007316	2977859262	39007316
AMRWB+_lab8_lis06	OK	OK	197229364	39007316	197229364	39007316
AMRWB+_lab8_lis07	OK	OK	2179137275	39007316	2179137275	39007316
AMRWB+_lab8_lis08	OK	OK	1450969274	39007316	1450969274	39007316
AMRWB+_lab8_lis09	OK	OK	602588847	39007316	602588847	39007316
AMRWB+_lab8_lis10	OK	OK	2213273170	39007316	2213273170	39007316
AMRWB+_lab8_lis11	OK	OK	3400597754	39007316	3400597754	39007316
AMRWB+_lab8_lis12	OK	OK	2795426979	39007316	2795426979	39007316
AMRWB+_lab8_lis13	OK	OK	3790448677	39007316	3790448677	39007316
AMRWB+_lab8_lis14	OK	OK	2477436808	39007316	2477436808	39007316
AMRWB+_lab8_lis15	OK	OK	401943637	39007316	401943637	39007316
ct_lab3_lis01	OK	OK	3242377789	38989816	3242377789	38989816
ct_lab3_lis02	OK	OK	4243582940	38989816	4243582940	38989816

ct_lab3_lis03	OK	OK	2879232160	38989816	2879232160	38989816
ct_lab3_lis04	OK	OK	194127239	38989816	194127239	38989816
ct_lab3_lis05	OK	OK	1643976335	38989816	1643976335	38989816
ct_lab3_lis06	OK	OK	948205457	38989816	948205457	38989816
ct_lab3_lis07	OK	OK	1687848429	38989816	1687848429	38989816
ct_lab3_lis08	OK	OK	3415549650	38989816	3415549650	38989816
ct_lab3_lis09	OK	OK	817120291	38989816	817120291	38989816
ct_lab3_lis10	OK	OK	3238429321	38989816	3238429321	38989816
ct_lab3_lis11	OK	OK	91045223	38998008	91045223	38998008
ct_lab3_lis12	OK	OK	1603168101	38989816	1603168101	38989816
ct_lab3_lis13	OK	OK	737895022	38989816	737895022	38989816
ct_lab3_lis14	OK	OK	771033502	38989816	771033502	38989816
ct_lab3_lis15	OK	OK	3198486066	38989816	3198486066	38989816
ct_lab4_lis01	OK	OK	4184453721	38989816	4184453721	38989816
ct_lab4_lis02	OK	OK	2777849710	38989816	2777849710	38989816
ct_lab4_lis03	OK	OK	3848830684	38989816	3848830684	38989816
ct_lab4_lis04	OK	OK	2495861721	38998008	2495861721	38998008
ct_lab4_lis05	OK	OK	4278563585	38989816	4278563585	38989816
ct_lab4_lis06	OK	OK	1830538637	38989816	1830538637	38989816
ct_lab4_lis07	OK	OK	3061316209	38989816	3061316209	38989816
ct_lab4_lis08	OK	OK	1634867734	38998008	1634867734	38998008
ct_lab4_lis09	OK	OK	3084829727	38989816	3084829727	38989816
ct_lab4_lis10	OK	OK	52486156	38989816	52486156	38989816
ct_lab4_lis11	OK	OK	3537915724	38989816	3537915724	38989816
ct_lab4_lis12	OK	OK	432377049	38989816	432377049	38989816
ct_lab4_lis13	OK	OK	445957849	38989816	445957849	38989816
ct_lab4_lis14	OK	OK	149006370	38989816	149006370	38989816
ct_lab4_lis15	OK	OK	3618654833	38989816	3618654833	38989816
ct_lab7_lis01	OK	OK	637637726	38989816	637637726	38989816
ct_lab7_lis02	OK	OK	3736460254	38989816	3736460254	38989816
ct_lab7_lis03	OK	OK	3092275960	38989816	3092275960	38989816
ct_lab7_lis04	OK	OK	2474561741	38989816	2474561741	38989816
ct_lab7_lis05	OK	OK	2188359407	38989816	2188359407	38989816
ct_lab7_lis06	OK	OK	1813552410	38989816	1813552410	38989816
ct_lab7_lis07	OK	OK	771960452	38989816	771960452	38989816
ct_lab7_lis08	OK	OK	108169698	38989816	108169698	38989816
ct_lab7_lis09	OK	OK	480208049	38989816	480208049	38989816
ct_lab7_lis10	OK	OK	4043859013	38989816	4043859013	38989816
ct_lab7_lis11	OK	OK	3856201822	38989816	3856201822	38989816
ct_lab7_lis12	OK	OK	3441288934	38989816	3441288934	38989816
ct_lab7_lis13	OK	OK	3710828186	38989816	3710828186	38989816
ct_lab7_lis14	OK	OK	1098997599	38989816	1098997599	38989816
ct_lab7_lis15	OK	OK	2967943897	38989816	2967943897	38989816
ct_lab8_lis01	OK	OK	1532423029	38989816	1532423029	38989816
ct_lab8_lis02	OK	OK	3482052807	38989816	3482052807	38989816
ct_lab8_lis03	OK	OK	1647763820	38989816	1647763820	38989816
ct_lab8_lis04	OK	OK	2679162899	38989816	2679162899	38989816
ct_lab8_lis05	OK	OK	3625265524	38989816	3625265524	38989816
ct_lab8_lis06	OK	OK	1911311061	38989816	1911311061	38989816
ct_lab8_lis07	OK	OK	3867282864	38989816	3867282864	38989816
ct_lab8_lis08	OK	OK	130969984	38998008	130969984	38998008
ct_lab8_lis09	OK	OK	1468541477	38989816	1468541477	38989816
ct_lab8_lis10	OK	OK	1661026824	38989816	1661026824	38989816
ct_lab8_lis11	OK	OK	4180059204	38989816	4180059204	38989816
ct_lab8_lis12	OK	OK	937669369	38989816	937669369	38989816
ct_lab8_lis13	OK	OK	2773507100	38989816	2773507100	38989816
ct_lab8_lis14	OK	OK	2283011003	38989816	2283011003	38989816
ct_lab8_lis15	OK	OK	3666806426	38989816	3666806426	38989816
AAC_lab3_lis01	OK	OK	4168580511	38987632	4168580511	38987632
AAC_lab3_lis02	OK	OK	612734854	38987632	612734854	38987632
AAC_lab3_lis03	OK	OK	2541462991	38987632	2541462991	38987632
AAC_lab3_lis04	OK	OK	2588486939	38987632	2588486939	38987632
AAC_lab3_lis05	OK	OK	1153057362	38987632	1153057362	38987632
AAC_lab3_lis06	OK	OK	1713872636	38987632	1713872636	38987632
AAC_lab3_lis07	OK	OK	3402963511	38987632	3402963511	38987632
AAC_lab3_lis08	OK	OK	2846044960	38987632	2846044960	38987632
AAC_lab3_lis09	OK	OK	4231292444	38987632	4231292444	38987632
AAC_lab3_lis10	OK	OK	2522713224	38987632	2522713224	38987632
AAC_lab3_lis11	OK	OK	3663237022	38987632	3663237022	38987632

AAC_lab3_lis12	OK	OK	3622060666	38987632	3622060666	38987632
AAC_lab3_lis13	OK	OK	511088819	38987632	511088819	38987632
AAC_lab3_lis14	OK	OK	746137416	38987632	746137416	38987632
AAC_lab3_lis15	OK	OK	1437954881	38987632	1437954881	38987632
AAC_lab4_lis01	OK	OK	2789303944	38987632	2789303944	38987632
AAC_lab4_lis02	OK	OK	231559578	38987632	231559578	38987632
AAC_lab4_lis03	OK	OK	1239096714	38987632	1239096714	38987632
AAC_lab4_lis04	OK	OK	3881043654	38987632	3881043654	38987632
AAC_lab4_lis05	OK	OK	3595361778	38987632	3595361778	38987632
AAC_lab4_lis06	OK	OK	1323444156	38987632	1323444156	38987632
AAC_lab4_lis07	OK	OK	698882494	38987632	698882494	38987632
AAC_lab4_lis08	OK	OK	335479284	38987632	335479284	38987632
AAC_lab4_lis09	OK	OK	409227055	38987632	409227055	38987632
AAC_lab4_lis10	OK	OK	3675021161	38987632	3675021161	38987632
AAC_lab4_lis11	OK	OK	659418374	38987632	659418374	38987632
AAC_lab4_lis12	OK	OK	3014475821	38987632	3014475821	38987632
AAC_lab4_lis13	OK	OK	2934022118	38987632	2934022118	38987632
AAC_lab4_lis14	OK	OK	2063281976	38987632	2063281976	38987632
AAC_lab4_lis15	OK	OK	203894854	38987632	203894854	38987632
AAC_lab7_lis01	OK	OK	1051353285	38987632	1051353285	38987632
AAC_lab7_lis02	OK	OK	159359002	38987632	159359002	38987632
AAC_lab7_lis03	OK	OK	3845608145	38987632	3845608145	38987632
AAC_lab7_lis04	OK	OK	2115707699	38987632	2115707699	38987632
AAC_lab7_lis05	OK	OK	3986982174	38987632	3986982174	38987632
AAC_lab7_lis06	OK	OK	4026267689	38987632	4026267689	38987632
AAC_lab7_lis07	OK	OK	1902899373	38987632	1902899373	38987632
AAC_lab7_lis08	OK	OK	2186296140	38987632	2186296140	38987632
AAC_lab7_lis09	OK	OK	438812629	38987632	438812629	38987632
AAC_lab7_lis10	OK	OK	3960422139	38987632	3960422139	38987632
AAC_lab7_lis11	OK	OK	2802776856	38987632	2802776856	38987632
AAC_lab7_lis12	OK	OK	2615976462	38987632	2615976462	38987632
AAC_lab7_lis13	OK	OK	2282108065	38987632	2282108065	38987632
AAC_lab7_lis14	OK	OK	3761610767	38987632	3761610767	38987632
AAC_lab7_lis15	OK	OK	3585081512	38987632	3585081512	38987632
AAC_lab8_lis01	OK	OK	3205447950	38987632	3205447950	38987632
AAC_lab8_lis02	OK	OK	1851949435	38987632	1851949435	38987632
AAC_lab8_lis03	OK	OK	2885677961	38987632	2885677961	38987632
AAC_lab8_lis04	OK	OK	2182936089	38987632	2182936089	38987632
AAC_lab8_lis05	OK	OK	3196817420	38987632	3196817420	38987632
AAC_lab8_lis06	OK	OK	2100388539	38987632	2100388539	38987632
AAC_lab8_lis07	OK	OK	3903484516	38987632	3903484516	38987632
AAC_lab8_lis08	OK	OK	4025274130	38987632	4025274130	38987632
AAC_lab8_lis09	OK	OK	3611204155	38987632	3611204155	38987632
AAC_lab8_lis10	OK	OK	3148026988	38987632	3148026988	38987632
AAC_lab8_lis11	OK	OK	27330726	38987632	27330726	38987632
AAC_lab8_lis12	OK	OK	3808526129	38987632	3808526129	38987632
AAC_lab8_lis13	OK	OK	3186048451	38987632	3186048451	38987632
AAC_lab8_lis14	OK	OK	3476566035	38987632	3476566035	38987632
AAC_lab8_lis15	OK	OK	1129126078	38987632	1129126078	38987632
AMR-WB_lab3_lis01	OK	OK	4096586183	39017392	4096586183	39017392
AMR-WB_lab3_lis02	OK	OK	3717091898	39017392	3717091898	39017392
AMR-WB_lab3_lis03	OK	OK	3099707267	39017392	3099707267	39017392
AMR-WB_lab3_lis04	OK	OK	1318116916	39017392	1318116916	39017392
AMR-WB_lab3_lis05	OK	OK	1323106270	39017392	1323106270	39017392
AMR-WB_lab3_lis06	OK	OK	2767424982	39017392	2767424982	39017392
AMR-WB_lab3_lis07	OK	OK	2762499626	39017392	2762499626	39017392
AMR-WB_lab3_lis08	OK	OK	501328113	39017392	501328113	39017392
AMR-WB_lab3_lis09	OK	OK	1825771139	39017392	1825771139	39017392
AMR-WB_lab3_lis10	OK	OK	3827739324	39017392	3827739324	39017392
AMR-WB_lab3_lis11	OK	OK	491452483	39017392	491452483	39017392
AMR-WB_lab3_lis12	OK	OK	4045500679	39017392	4045500679	39017392
AMR-WB_lab3_lis13	OK	OK	3131701150	39017392	3131701150	39017392
AMR-WB_lab3_lis14	OK	OK	540327290	39017392	540327290	39017392
AMR-WB_lab3_lis15	OK	OK	2475192882	39017392	2475192882	39017392
AMR-WB_lab4_lis01	OK	OK	2727568068	39017392	2727568068	39017392
AMR-WB_lab4_lis02	OK	OK	714363472	39017392	714363472	39017392
AMR-WB_lab4_lis03	OK	OK	1520451695	39017392	1520451695	39017392
AMR-WB_lab4_lis04	OK	OK	2785946643	39017392	2785946643	39017392
AMR-WB_lab4_lis05	OK	OK	1560664001	39017392	1560664001	39017392
AMR-WB_lab4_lis06	OK	OK	3267055743	39017392	3267055743	39017392

AMR-WB_lab4_lis07	OK	OK	968677452	39017392	968677452	39017392
AMR-WB_lab4_lis08	OK	OK	2208557631	39017392	2208557631	39017392
AMR-WB_lab4_lis09	OK	OK	3281161749	39017392	3281161749	39017392
AMR-WB_lab4_lis10	OK	OK	815686118	39017392	815686118	39017392
AMR-WB_lab4_lis11	OK	OK	716482502	39017392	716482502	39017392
AMR-WB_lab4_lis12	OK	OK	1198063078	39017392	1198063078	39017392
AMR-WB_lab4_lis13	OK	OK	1480189521	39017392	1480189521	39017392
AMR-WB_lab4_lis14	OK	OK	1176534177	39017392	1176534177	39017392
AMR-WB_lab4_lis15	OK	OK	2610982779	39017392	2610982779	39017392
AMR-WB_lab7_lis01	OK	OK	3215684546	39017392	3215684546	39017392
AMR-WB_lab7_lis02	OK	OK	2126293435	39017392	2126293435	39017392
AMR-WB_lab7_lis03	OK	OK	922513646	39017392	922513646	39017392
AMR-WB_lab7_lis04	OK	OK	2145119980	39017392	2145119980	39017392
AMR-WB_lab7_lis05	OK	OK	4019689984	39017392	4019689984	39017392
AMR-WB_lab7_lis06	OK	OK	1869514521	39017392	1869514521	39017392
AMR-WB_lab7_lis07	OK	OK	121618863	39017392	121618863	39017392
AMR-WB_lab7_lis08	OK	OK	1736710192	39017392	1736710192	39017392
AMR-WB_lab7_lis09	OK	OK	3935729428	39017392	3935729428	39017392
AMR-WB_lab7_lis10	OK	OK	2062876184	39017392	2062876184	39017392
AMR-WB_lab7_lis11	OK	OK	1244927456	39017392	1244927456	39017392
AMR-WB_lab7_lis12	OK	OK	1021461777	39017392	1021461777	39017392
AMR-WB_lab7_lis13	OK	OK	3843783283	39017392	3843783283	39017392
AMR-WB_lab7_lis14	OK	OK	305370524	39017392	305370524	39017392
AMR-WB_lab7_lis15	OK	OK	2304539598	39017392	2304539598	39017392
AMR-WB_lab8_lis01	OK	OK	4062706840	39017392	4062706840	39017392
AMR-WB_lab8_lis02	OK	OK	517015501	39017392	517015501	39017392
AMR-WB_lab8_lis03	OK	OK	1157687678	39017392	1157687678	39017392
AMR-WB_lab8_lis04	OK	OK	3561550451	39017392	3561550451	39017392
AMR-WB_lab8_lis05	OK	OK	1272632407	39017392	1272632407	39017392
AMR-WB_lab8_lis06	OK	OK	2993146919	39017392	2993146919	39017392
AMR-WB_lab8_lis07	OK	OK	2179347515	39017392	2179347515	39017392
AMR-WB_lab8_lis08	OK	OK	2016428377	39017392	2016428377	39017392
AMR-WB_lab8_lis09	OK	OK	783102575	39017392	783102575	39017392
AMR-WB_lab8_lis10	OK	OK	4258758198	39017392	4258758198	39017392
AMR-WB_lab8_lis11	OK	OK	2352463800	39017392	2352463800	39017392
AMR-WB_lab8_lis12	OK	OK	3865207087	39017392	3865207087	39017392
AMR-WB_lab8_lis13	OK	OK	857034679	39017392	857034679	39017392
AMR-WB_lab8_lis14	OK	OK	1034940926	39017392	1034940926	39017392
AMR-WB_lab8_lis15	OK	OK	3558904135	39017392	3558904135	39017392
hidref		OK	1868767381	39025080	1868767381	39025080
lp3500_s12		OK	991955621	39025080	991955621	39025080
lp7000_s12		OK	3132004107	39025080	3132004107	39025080
lp7000_s6		OK	587142961	39025080	587142961	39025080
opref		OK	1868767381	39025080	1868767381	39025080

3.2 High-Rate Test

H1

System	CkSum	Len	ARLck	ARLlen	TSck	TSlen
all_cat	OK	OK	3164833053	26781948	3164833053	26781948
aacPlus	OK	OK	2853830041	26754792	2853830041	26754792
ct	OK	OK	3758885945	26750968	3758885945	26750968
AAC	OK	OK	1551757251	26761612	1551757251	26761612
RN	OK	OK	3912230100	26767216	3912230100	26767216
hidref	OK	OK	2014698721	26781948	2014698721	26781948
lp3500	OK	OK	60306597	26781948	60306597	26781948
lp7000	OK	OK	2435172842	26781948	2435172842	26781948
opref	OK	OK	2014698721	26781948	2014698721	26781948

H2

System	CkSum	Len	ARLck	ARLlen	TSck	TSlen
all_cat	OK	OK	3164833053	26781948	3164833053	26781948
aacPlus	OK	OK	3181743050	26754792	3181743050	26754792
ct	OK	OK	3181743050	26754792	3181743050	26754792
AAC	OK	OK	2642424272	26761044	2642424272	26761044
RN	OK	OK	1455418496	26767220	1455418496	26767220
hidref	OK	OK	2014698721	26781948	2014698721	26781948
lp3500	OK	OK	60306597	26781948	60306597	26781948
lp7000	OK	OK	2435172842	26781948	2435172842	26781948
opref	OK	OK	2014698721	26781948	2014698721	26781948

H3

System	CkSum	Len	ARLck	ARLlen	TSck	TSlen
all_cat	OK	OK	3164833053	26781948	3164833053	26781948
aacPlus_FER1_lab3_lis01	OK	OK	3548610842	26754792	3548610842	26754792
aacPlus_FER1_lab3_lis02	OK	OK	230919893	26754792	230919893	26754792
aacPlus_FER1_lab3_lis03	OK	OK	19977230	26754792	19977230	26754792
aacPlus_FER1_lab3_lis04	OK	OK	3524278133	26754792	3524278133	26754792
aacPlus_FER1_lab3_lis05	OK	OK	2025188597	26754792	2025188597	26754792
aacPlus_FER1_lab3_lis06	OK	OK	1157542472	26754792	1157542472	26754792
aacPlus_FER1_lab3_lis07	OK	OK	971946427	26754792	971946427	26754792
aacPlus_FER1_lab3_lis08	OK	OK	754798582	26754792	754798582	26754792
aacPlus_FER1_lab3_lis09	OK	OK	1976202386	26754792	1976202386	26754792
aacPlus_FER1_lab3_lis10	OK	OK	3594110410	26754792	3594110410	26754792
aacPlus_FER1_lab3_lis11	OK	OK	1702620420	26754792	1702620420	26754792
aacPlus_FER1_lab3_lis12	OK	OK	991029139	26754792	991029139	26754792
aacPlus_FER1_lab3_lis13	OK	OK	3410978821	26754792	3410978821	26754792
aacPlus_FER1_lab3_lis14	OK	OK	855137367	26754792	855137367	26754792
aacPlus_FER1_lab3_lis15	OK	OK	2498653381	26754792	2498653381	26754792
aacPlus_FER1_lab6_lis01	OK	OK	3173557258	26754792	3173557258	26754792
aacPlus_FER1_lab6_lis02	OK	OK	765784618	26762984	765784618	26762984
aacPlus_FER1_lab6_lis03	OK	OK	3444558806	26754792	3444558806	26754792
aacPlus_FER1_lab6_lis04	OK	OK	3807464857	26754792	3807464857	26754792
aacPlus_FER1_lab6_lis05	OK	OK	2337071747	26754792	2337071747	26754792
aacPlus_FER1_lab6_lis06	OK	OK	1820357858	26754792	1820357858	26754792
aacPlus_FER1_lab6_lis07	OK	OK	3099719281	26754792	3099719281	26754792
aacPlus_FER1_lab6_lis08	OK	OK	1079367903	26754792	1079367903	26754792
aacPlus_FER1_lab6_lis09	OK	OK	4149668353	26754792	4149668353	26754792
aacPlus_FER1_lab6_lis10	OK	OK	834355501	26754792	834355501	26754792
aacPlus_FER1_lab6_lis11	OK	OK	1533654551	26754792	1533654551	26754792
aacPlus_FER1_lab6_lis12	OK	OK	90166317	26754792	90166317	26754792
aacPlus_FER1_lab6_lis13	OK	OK	361809098	26754792	361809098	26754792
aacPlus_FER1_lab6_lis14	OK	OK	1709980125	26754792	1709980125	26754792
aacPlus_FER1_lab6_lis15	OK	OK	1103118042	26754792	1103118042	26754792
aacPlus_FER3_lab3_lis01	OK	OK	2846671316	26754792	2846671316	26754792
aacPlus_FER3_lab3_lis02	OK	OK	1390949373	26754792	1390949373	26754792
aacPlus_FER3_lab3_lis03	OK	OK	1920203693	26754792	1920203693	26754792
aacPlus_FER3_lab3_lis04	OK	OK	3789391358	26754792	3789391358	26754792
aacPlus_FER3_lab3_lis05	OK	OK	3349600771	26754792	3349600771	26754792
aacPlus_FER3_lab3_lis06	OK	OK	2847966045	26754792	2847966045	26754792
aacPlus_FER3_lab3_lis07	OK	OK	4049456712	26754792	4049456712	26754792
aacPlus_FER3_lab3_lis08	OK	OK	2025678954	26754792	2025678954	26754792
aacPlus_FER3_lab3_lis09	OK	OK	1511098233	26754792	1511098233	26754792
aacPlus_FER3_lab3_lis10	OK	OK	76662166	26754792	76662166	26754792
aacPlus_FER3_lab3_lis11	OK	OK	3415032953	26754792	3415032953	26754792
aacPlus_FER3_lab3_lis12	OK	OK	2591649971	26754792	2591649971	26754792
aacPlus_FER3_lab3_lis13	OK	OK	2322177573	26754792	2322177573	26754792
aacPlus_FER3_lab3_lis14	OK	OK	2914703695	26754792	2914703695	26754792
aacPlus_FER3_lab3_lis15	OK	OK	2615329849	26754792	2615329849	26754792
aacPlus_FER3_lab6_lis01	OK	OK	4237063404	26754792	4237063404	26754792
aacPlus_FER3_lab6_lis02	OK	OK	1750907866	26762984	1750907866	26762984
aacPlus_FER3_lab6_lis03	OK	OK	2329761071	26754792	2329761071	26754792
aacPlus_FER3_lab6_lis04	OK	OK	890723663	26754792	890723663	26754792
aacPlus_FER3_lab6_lis05	OK	OK	3667230761	26754792	3667230761	26754792
aacPlus_FER3_lab6_lis06	OK	OK	2387294795	26754792	2387294795	26754792
aacPlus_FER3_lab6_lis07	OK	OK	2666918204	26754792	2666918204	26754792
aacPlus_FER3_lab6_lis08	OK	OK	2398338681	26754792	2398338681	26754792
aacPlus_FER3_lab6_lis09	OK	OK	1144170418	26754792	1144170418	26754792
aacPlus_FER3_lab6_lis10	OK	OK	3846455027	26754792	3846455027	26754792
aacPlus_FER3_lab6_lis11	OK	OK	2771567841	26754792	2771567841	26754792
aacPlus_FER3_lab6_lis12	OK	OK	3327856338	26754792	3327856338	26754792
aacPlus_FER3_lab6_lis13	OK	OK	1286579674	26754792	1286579674	26754792
aacPlus_FER3_lab6_lis14	OK	OK	2106895072	26754792	2106895072	26754792
aacPlus_FER3_lab6_lis15	OK	OK	471951166	26754792	471951166	26754792
ct_FER1_lab3_lis01	OK	OK	2710196854	26750968	2710196854	26750968
ct_FER1_lab3_lis02	OK	OK	3190308896	26750968	3190308896	26750968
ct_FER1_lab3_lis03	OK	OK	2360323530	26750968	2360323530	26750968
ct_FER1_lab3_lis04	OK	OK	3314949772	26750968	3314949772	26750968
ct_FER1_lab3_lis05	OK	OK	916230285	26750968	916230285	26750968
ct_FER1_lab3_lis06	OK	OK	4252697283	26750968	4252697283	26750968
ct_FER1_lab3_lis07	OK	OK	255207898	26750968	255207898	26750968

ct_FER1_lab3_lis08	OK	OK	939712618	26750968	939712618	26750968
ct_FER1_lab3_lis09	OK	OK	2821690121	26750968	2821690121	26750968
ct_FER1_lab3_lis10	OK	OK	496877080	26750968	496877080	26750968
ct_FER1_lab3_lis11	OK	OK	2382492109	26750968	2382492109	26750968
ct_FER1_lab3_lis12	OK	OK	1088097214	26750968	1088097214	26750968
ct_FER1_lab3_lis13	OK	OK	1821245623	26750968	1821245623	26750968
ct_FER1_lab3_lis14	OK	OK	2062663777	26750968	2062663777	26750968
ct_FER1_lab3_lis15	OK	OK	3883734575	26750968	3883734575	26750968
ct_FER1_lab6_lis01	OK	OK	3412249162	26750968	3412249162	26750968
ct_FER1_lab6_lis02	OK	OK	2508392897	26759160	2508392897	26759160
ct_FER1_lab6_lis03	OK	OK	3713911617	26750968	3713911617	26750968
ct_FER1_lab6_lis04	OK	OK	3282728844	26750968	3282728844	26750968
ct_FER1_lab6_lis05	OK	OK	3898014792	26750968	3898014792	26750968
ct_FER1_lab6_lis06	OK	OK	758294126	26750968	758294126	26750968
ct_FER1_lab6_lis07	OK	OK	3184162720	26750968	3184162720	26750968
ct_FER1_lab6_lis08	OK	OK	3419862204	26750968	3419862204	26750968
ct_FER1_lab6_lis09	OK	OK	195304254	26750968	195304254	26750968
ct_FER1_lab6_lis10	OK	OK	4199690198	26750968	4199690198	26750968
ct_FER1_lab6_lis11	OK	OK	4003467654	26750968	4003467654	26750968
ct_FER1_lab6_lis12	OK	OK	2678635546	26750968	2678635546	26750968
ct_FER1_lab6_lis13	OK	OK	572001385	26750968	572001385	26750968
ct_FER1_lab6_lis14	OK	OK	975169268	26750968	975169268	26750968
ct_FER1_lab6_lis15	OK	OK	3635901348	26750968	3635901348	26750968
ct_FER3_lab3_lis01	OK	OK	1702643257	26750968	1702643257	26750968
ct_FER3_lab3_lis02	OK	OK	1100162795	26750968	1100162795	26750968
ct_FER3_lab3_lis03	OK	OK	868354206	26750968	868354206	26750968
ct_FER3_lab3_lis04	OK	OK	1266589175	26750968	1266589175	26750968
ct_FER3_lab3_lis05	OK	OK	4079020512	26750968	4079020512	26750968
ct_FER3_lab3_lis06	OK	OK	1379713600	26750968	1379713600	26750968
ct_FER3_lab3_lis07	OK	OK	1041508067	26750968	1041508067	26750968
ct_FER3_lab3_lis08	OK	OK	3666891694	26750968	3666891694	26750968
ct_FER3_lab3_lis09	OK	OK	3881281785	26750968	3881281785	26750968
ct_FER3_lab3_lis10	OK	OK	1856269082	26750968	1856269082	26750968
ct_FER3_lab3_lis11	OK	OK	516070730	26750968	516070730	26750968
ct_FER3_lab3_lis12	OK	OK	337154459	26750968	337154459	26750968
ct_FER3_lab3_lis13	OK	OK	4030378155	26750968	4030378155	26750968
ct_FER3_lab3_lis14	OK	OK	2547978000	26750968	2547978000	26750968
ct_FER3_lab3_lis15	OK	OK	1543461300	26750968	1543461300	26750968
ct_FER3_lab6_lis01	OK	OK	644994390	26750968	644994390	26750968
ct_FER3_lab6_lis02	OK	OK	1775438167	26759160	1775438167	26759160
ct_FER3_lab6_lis03	OK	OK	3569789744	26750968	3569789744	26750968
ct_FER3_lab6_lis04	OK	OK	3059649512	26750968	3059649512	26750968
ct_FER3_lab6_lis05	OK	OK	345838880	26750968	345838880	26750968
ct_FER3_lab6_lis06	OK	OK	3299905512	26750968	3299905512	26750968
ct_FER3_lab6_lis07	OK	OK	1096067786	26750968	1096067786	26750968
ct_FER3_lab6_lis08	OK	OK	4267274526	26750968	4267274526	26750968
ct_FER3_lab6_lis09	OK	OK	4135568521	26750968	4135568521	26750968
ct_FER3_lab6_lis10	OK	OK	2438720487	26750968	2438720487	26750968
ct_FER3_lab6_lis11	OK	OK	1135224564	26750968	1135224564	26750968
ct_FER3_lab6_lis12	OK	OK	1684532984	26750968	1684532984	26750968
ct_FER3_lab6_lis13	OK	OK	3569257738	26750968	3569257738	26750968
ct_FER3_lab6_lis14	OK	OK	2544874218	26750968	2544874218	26750968
ct_FER3_lab6_lis15	OK	OK	1312325671	26750968	1312325671	26750968
AAC_FER1_lab3_lis01	OK	OK	3514729623	26761612	3514729623	26761612
AAC_FER1_lab3_lis02	OK	OK	2315404033	26761612	2315404033	26761612
AAC_FER1_lab3_lis03	OK	OK	4182342219	26761612	4182342219	26761612
AAC_FER1_lab3_lis04	OK	OK	4037701316	26761612	4037701316	26761612
AAC_FER1_lab3_lis05	OK	OK	2687934731	26761612	2687934731	26761612
AAC_FER1_lab3_lis06	OK	OK	1703842287	26761612	1703842287	26761612
AAC_FER1_lab3_lis07	OK	OK	2937247487	26761612	2937247487	26761612
AAC_FER1_lab3_lis08	OK	OK	3861372079	26761612	3861372079	26761612
AAC_FER1_lab3_lis09	OK	OK	2435156999	26761612	2435156999	26761612
AAC_FER1_lab3_lis10	OK	OK	2264510640	26761612	2264510640	26761612
AAC_FER1_lab3_lis11	OK	OK	3759606181	26761612	3759606181	26761612
AAC_FER1_lab3_lis12	OK	OK	177203968	26761612	177203968	26761612
AAC_FER1_lab3_lis13	OK	OK	2200875729	26761612	2200875729	26761612
AAC_FER1_lab3_lis14	OK	OK	2077072537	26761612	2077072537	26761612
AAC_FER1_lab3_lis15	OK	OK	1176319502	26761612	1176319502	26761612
AAC_FER1_lab6_lis01	OK	OK	2515870187	26761612	2515870187	26761612
AAC_FER1_lab6_lis02	OK	OK	1502931402	26761612	1502931402	26761612
AAC_FER1_lab6_lis03	OK	OK	1187615095	26761612	1187615095	26761612

AAC_FER1_lab6_lis04	OK	OK	1616068630	26761612	1616068630	26761612
AAC_FER1_lab6_lis05	OK	OK	1419291596	26761612	1419291596	26761612
AAC_FER1_lab6_lis06	OK	OK	3864798629	26761612	3864798629	26761612
AAC_FER1_lab6_lis07	OK	OK	202761818	26761612	202761818	26761612
AAC_FER1_lab6_lis08	OK	OK	4071261471	26761612	4071261471	26761612
AAC_FER1_lab6_lis09	OK	OK	3177557483	26761612	3177557483	26761612
AAC_FER1_lab6_lis10	OK	OK	2640859588	26761612	2640859588	26761612
AAC_FER1_lab6_lis11	OK	OK	72326515	26761612	72326515	26761612
AAC_FER1_lab6_lis12	OK	OK	2578396911	26761612	2578396911	26761612
AAC_FER1_lab6_lis13	OK	OK	2236440431	26761612	2236440431	26761612
AAC_FER1_lab6_lis14	OK	OK	1517741018	26761612	1517741018	26761612
AAC_FER1_lab6_lis15	OK	OK	2436651062	26761612	2436651062	26761612
AAC_FER3_lab3_lis01	OK	OK	878802055	26761612	878802055	26761612
AAC_FER3_lab3_lis02	OK	OK	3668788947	26761612	3668788947	26761612
AAC_FER3_lab3_lis03	OK	OK	4081735751	26761612	4081735751	26761612
AAC_FER3_lab3_lis04	OK	OK	2978313307	26761612	2978313307	26761612
AAC_FER3_lab3_lis05	OK	OK	1964519746	26761612	1964519746	26761612
AAC_FER3_lab3_lis06	OK	OK	2938641539	26761612	2938641539	26761612
AAC_FER3_lab3_lis07	OK	OK	2001289512	26761612	2001289512	26761612
AAC_FER3_lab3_lis08	OK	OK	3619517064	26761612	3619517064	26761612
AAC_FER3_lab3_lis09	OK	OK	616400307	26761612	616400307	26761612
AAC_FER3_lab3_lis10	OK	OK	2667414856	26761612	2667414856	26761612
AAC_FER3_lab3_lis11	OK	OK	2351961471	26761612	2351961471	26761612
AAC_FER3_lab3_lis12	OK	OK	1415741462	26761612	1415741462	26761612
AAC_FER3_lab3_lis13	OK	OK	173084395	26761612	173084395	26761612
AAC_FER3_lab3_lis14	OK	OK	4075977604	26761612	4075977604	26761612
AAC_FER3_lab3_lis15	OK	OK	1533326939	26761612	1533326939	26761612
AAC_FER3_lab6_lis01	OK	OK	3649707613	26761612	3649707613	26761612
AAC_FER3_lab6_lis02	OK	OK	3936911225	26761612	3936911225	26761612
AAC_FER3_lab6_lis03	OK	OK	2536659742	26761612	2536659742	26761612
AAC_FER3_lab6_lis04	OK	OK	2072497752	26761612	2072497752	26761612
AAC_FER3_lab6_lis05	OK	OK	965796038	26761612	965796038	26761612
AAC_FER3_lab6_lis06	OK	OK	2262994680	26761612	2262994680	26761612
AAC_FER3_lab6_lis07	OK	OK	1288290613	26761612	1288290613	26761612
AAC_FER3_lab6_lis08	OK	OK	2181383779	26761612	2181383779	26761612
AAC_FER3_lab6_lis09	OK	OK	3395581562	26761612	3395581562	26761612
AAC_FER3_lab6_lis10	OK	OK	487654947	26761612	487654947	26761612
AAC_FER3_lab6_lis11	OK	OK	1198126450	26761612	1198126450	26761612
AAC_FER3_lab6_lis12	OK	OK	3428580752	26761612	3428580752	26761612
AAC_FER3_lab6_lis13	OK	OK	4022669344	26761612	4022669344	26761612
AAC_FER3_lab6_lis14	OK	OK	2862329171	26761612	2862329171	26761612
AAC_FER3_lab6_lis15	OK	OK	1919939465	26761612	1919939465	26761612
hidref	OK	OK	2014698721	26781948	2014698721	26781948
lp3500	OK	OK	60306597	26781948	60306597	26781948
lp7000	OK	OK	2435172842	26781948	2435172842	26781948
opref	OK	OK	2014698721	26781948	2014698721	26781948

Technical Specification Group Services and System Aspects **TSGS#23(04)0071**
Meeting #23, Phoenix, Arizona (USA)
15-18 March 2004

3GPP TSG-SA WG4 Meeting #30 S4-040036
Malaga, Spain, 23rd - 27th February 2004.

Source:	**TSG SA WG4 (T-Systems)**
Title:	**PSS/MMS High Rate and AMR-WB+ and PSS/MMS Low Rate Audio Selection Test, Host Laboratory Report**
Agenda item:	**7.4.3**
Document for:	**APPROVAL**

---

1 Introduction

This document reports about the processing of the audio material for the Low Rate and High Rate Audio Selection tests.

The whole processing was done parallel at the host lab (T-Systems) and at the mirror host lab (Audio Research Labs). Every step was cross checked between these two labs in order to get reliable results.

The main processing was done on a Windows PC with Intel Pentium 4 Processor and Cygwin 1.5.5-1. After some problems while running the AAC+ Codec on a Windows 2000 machine, Windows XP Professional was used for processing. Additional Tools were the AFsp Library [4] and the STL2000 Library [5]. The processing was done script based. Most of the scripts were provided by several organizations as described in the test plans. These scripts as well as the scripts which were produced by the host lab, are available on request and are not part of this report. Only those which are necessary to reproduce the results of processing are appended in the annex.

2 Low-Rate Test

2.1 Test Items

The test items were selected at the Selection Entity (France Telecom). In a first round of this selection process there were provided:

- 4 training items and
- 12 test items

In a second selection step the number of items was increased to a much higher value. Then, the total number of items was 62. The number of training items remains 4. These items are belonging to 4 different genres which are of interest for future applications:

- Music
- Speech
- Speech over Music
- Speech between Music

The exact description of the naming of the items can be found in the test plan [2] in Annex B-4.

A complete list of these 62 items as well as the exact assignment of the audio files to the several experiments can be found in Annex 1 .

2.2 Processing Systems

2.2.1 Coding conditions under test

2.2.1.1 aacPlus

- Encoder: Coding Technologies' aacPlus 3GPP Reference Encoder
  Build Oct 30 2003, 17:50:34
  (c) 2000 - 2003 Coding Technologies
  (c) 1999 Fraunhofer Gesellschaft IIS-A
- Decoder: Coding Technologies' aacPlus 3GPP Reference Decoder
  Build Oct 30 2003, 17:57:00
  (c) 2000 - 2003 Coding Technologies
  (c) 1999 Fraunhofer Gesellschaft IIS-A

2.2.1.2 AMRWB+

- Encoder: AMRWB+ Encoder for 3GPP PSS/MMS low-rate audio codec selection
  Copyright 031030 Ericsson, Nokia, VoiceAge
- Decoder: AMRWB+ Decoder for 3GPP PSS/MMS low-rate audio codec selection
  Copyright 031030 Ericsson, Nokia, VoiceAge

2.2.1.3 ct

- Encoder: Coding Technologies' proprietary 3GPP Reference Encoder
  Build Oct 30 2003, 17:48:19
  (c) 2000 - 2003 Coding Technologies
  (c) 1999 Fraunhofer Gesellschaft IIS-A
- Decoder: Coding Technologies' proprietary 3GPP Reference Decoder
  Build Oct 30 2003, 17:54:37
  (c) 2000 - 2003 Coding Technologies
  (c) 1999 Fraunhofer Gesellschaft IIS-A

2.2.2 Reference coding conditions

2.2.2.1 AAC

- Encoder: MPEG-4 Audio Encoder (build Aug 13 2003),
  (C) 1997-2003 by Fraunhofer IIS-A,
  AudioLibDate: Aug 13 2003,
  AudioLibVersion: 1.1
- Decoder: MPEG-4 Audio Decoder (build Aug 8 2003),
  (C) 1997-2003 by Fraunhofer IIS-A

2.2.2.2 AMR-WB

- Encoder: AMR Wideband Codec 3GPP TS26.190 / ITU-T G.722.2, Aug 25, 2003. Version 5.8.0.
- Decoder: AMR Wideband Codec 3GPP TS26.190 / ITU-T G.722.2, Aug 25, 2003. Version 5.8.0.

2.2.3 Anchor conditions

2.2.3.1 Mono anchors

The mono anchors had to be low pass filtered with cutoff frequencies of 3.5 and 7 kHz. The script “anchor.sh”, which was provided by Coding Technologies was used for that purpose. The script can be found in the Annex 3.

2.2.3.2 Stereo anchors

In the experiments testing stereo signals three anchors will be used, low pass filtered and with reduced stereo image. The cutoff frequencies are the same as for mono tests 3.5 and 7.0 kHz.

2.3 Generation of error files

The error files were generated by using a tool which was provided by Ericsson. The tool was a shell script “genfer.csh“. The seed value was provided by ETSI and had a value of “171094“. In order to get individual error pattern files for each listener, the output file of the genfer.csh was processed with the “shiftcirc.csh” script which can be found in Annex 6.

2.4 Pre-processing

The clip silence utility was applied to all speech files in order to prevent unnecessary silence within the speech items. The “clip_silence.csh” script was provided by Ericsson.

For the mono experiments a mixing from stereo to mono was applied to all input files by invoking following command:

```
./anchor.sh -monoout ${CAT}48s.wav ${CAT}48m.wav
```

Sample rate conversion down to 16 kHz and 24 kHz for those codecs which require these lower input sampling rates i.e. the AMR codec.

```
./anchor.sh -monoout -fsout16 ${CAT}48s.wav ${CAT}16m.wav
./anchor.sh -monoout -fsout24 ${CAT}48s.wav ${CAT}24m.wav
./anchor.sh -fsout24 ${CAT}48s.wav ${CAT}24s.wav
```

2.5 Concatenation of material

Before main processing, all files for an experiment were concatenated to one big wav file. the order of the items within the several experiments was

- 4 training files in categorized order (m_, s_, sbm_, som_) followed by
- music files (m_) for sub-experiment a
- speech files (s_) for sub-experiment a
- speech between music files (sbm_) for sub-experiment a
- speech over music files (som_) for sub-experiment a
- music files (m_) for sub-experiment b
- speech files (s_) for sub-experiment b
- speech between music files (sbm_) for sub-experiment b
- speech over music files (som_) for sub-experiment b
- additional silence file with a length of 290 ms to provide buffer for codec delays.

All subcategories were arranged in alphabetical order of the filenames. As an example, the time file for Experiment A1 looks as follows:

```
m_vo_x_1_org.wav 960000
s_no_ft_9_org.wav 960002
sbm_fi_x_9_org.wav 960002
som_ot_x_9_org.wav 480002
m_ot_x_8_org.wav 960002
m_ot_x_a_org.wav 960002
m_po_x_5_org.wav 960002
m_po_x_7_org.wav 960002
s_cl_2t_3_org.wav 478080
s_cl_2t_4_org.wav 469440
s_no_2t_1_org.wav 478080
s_no_ft_1_org.wav 478080
sbm_js_x_1_org.wav 960002
sbm_ms_x_1_org.wav 960002
som_fi_x_4_org.wav 480002
som_ot_x_4_org.wav 480002
m_ot_x_9_org.wav 960002
m_ot_x_b.org.wav 960002
m_po_x_6_org.wav 960002
m_si_x_3_org.wav 960002
s_cl_2t_5_org.wav 465600
s_cl_mt_2_org.wav 454080
s_no_2t_2_org.wav 478080
s_no_ft_2_org.wav 478080
sbm_sj_x_1_org.wav 960002
sbm_sm_x_6_org.wav 960002
som_ot_x_5_org.wav 480002
som_ot_x_6_org.wav 480002
```

```
silence.wav 27838
```

The sampling rate for each concatenated file was 48 kHz, stereo. These files were used as input for the further processing steps. The input sampling rates and channel numbers for some codecs and mono experiments had to be changed. Therefore, also the “anchor.sh” script was used.

2.6 Main processing

2.6.1 Creation of anchors

The low frequency anchors were produced using the anchors.sh script. The processing for the mono sessions included:

- low pass filtering with cutoff frequencies of 3500 Hz and 7000 Hz.

```
./anchor.sh -lp3500 -monoout ${CAT}48s.wav ${CAT}_anchor_35_m.wav
./anchor.sh -lp7000 -monoout ${CAT}48s.wav ${CAT}_anchor_7_m.wav
```

For the stereo experiments 3 low pass filtered anchors were needed:

- 3.5 kHz Low pass, significantly reduced stereo image (12dB attenuated side channel)

```
./anchor.sh -lp3500 -s12 ${CAT}48s.wav ${CAT}_anchor_35_s12.wav
```

- 7.0 kHz Low pass, significantly reduced stereo image (12 dB attenuated side channel)

```
./anchor.sh -lp7000 -s12 ${CAT}48s.wav ${CAT}_anchor_7_s12.wav
```

- 7.0 kHz Low pass, slightly reduced stereo image (6 dB attenuated side channel)

```
./anchor.sh -lp7000 -s6 ${CAT}48s.wav ${CAT}_anchor_7_s6.wav
```

2.6.2 Coding of material

The coding for the several sub-experiments was done according the bitrates in the test plan [2].

Exp.	Operational mode	Audio Material
A1a	14 kbps, mono, use case A (PSS)	Set a
A1b		Set b
A2a	18 kbps, stereo, use case A (PSS)	Set a
A2b		Set b
A3a	24 kbps, mono, use case A (PSS)	Set a
A3b		Set b
A4a	24 kbps, stereo, use case A (PSS)	Set a
A4b		Set b
B1a	14 kbps, mono, use case B (MMS), 16 kHz input and output sampling rate	Set a
B1b		Set b
B2a	18 kbps, stereo, use case B (MMS)	Set a
B2b		Set b
B3a	14 kbps, mono, use case A (PSS), 3% FER	Set a
B3b		Set b
B4a	24 kbps, stereo, use case A (PSS), 3% FER	Set a
B4b		Set b

Each codec was used as command line utility with file input and file output. After decoding, all files were upsampled to 48 kHz and attenuated to 0.93 by using the “upsamp48.sh” script, which can be found in Annex 4.

For each codec an individual script was used, which called the encoder, then the decoder and afterwards shifted the output signal in a way that the codec delay was removed.

In the following paragraphs only the important calls with relevant parameters are listed in order to enable better reading. Especially for sub-experiments B3 and B4 the loops were not included here, even though they were processed.

2.6.2.1 AAC

LBR-A1:

```
mp4EvalCmdl -v -b 14000 a1_all_cat48m.wav out_14m_a1.mp4
mp4dec -if out_14m_a1.mp4 -of out_14m_a1.wav
CopyAudio -l3254: out_14m_a1.wav dly_${MODE}.wav
```

LBR-A2

```
mp4EvalCmdl.exe -v -b 18000 a2_all_cat48s.wav out_18s_a2.mp4
mp4dec.exe -if out_18s_a2.mp4 -of out_18s_a2.wav
CopyAudio.exe -l3229: out_18s_a2.wav dly_${MODE}.wav
```

LBR-A3

```
mp4EvalCmdl -v -b 24000 a3_all_cat48m.wav out_24m_a3.mp4
mp4dec -if out_24m_a3.mp4 -of out_24m_a3.wav
CopyAudio -l3289: out_24m_a3.wav dly_${MODE}.wav
```

LBR-A4

```
mp4EvalCmdl -v -b 24000 a4_all_cat48s.wav out_24s_a4.mp4
mp4dec -if out_24s_a4.mp4 -of out_24s_a4.wav
CopyAudio -l3254: out_24s_a4.wav dly_${MODE}.wav
```

LBR-B1

```
mp4EvalCmdl -v -b 14000 b1_all_cat16m.wav out_14m_b1.mp4
mp4dec -if out_14m_b1.mp4 -of out_14m_b1.wav
CopyAudio -l3216: out_14m_b1.wav dly_${MODE}.wav
```

LBR-B2

```
mp4EvalCmdl -v -b 18000 preproc/b2_all_cat48s.wav out_18s_b2.mp4
mp4dec -if out_18s_b2.mp4 -of out_18s_b2.wav
CopyAudio -l3229: out_18s_b2.wav dly_${MODE}.wav
```

LBR-B3

```
mp4EvalCmdl -v -b 14000 b3_all_cat48m.wav out_14m_b3.mp4
mp4dec -if out_14m_b3.mp4 -of out_14m.wav –epf epf3_lab${i}_lis${j}_b3.epf
CopyAudio -l3254: out_14m.wav dly_lab${i}_lis${j}_${MODE}.wav
```

LBR-B4

```
mp4EvalCmdl -v -b 24000 b4_all_cat48s.wav out_24s_b4.mp4
mp4dec -if out_24s_b4.mp4 -of out_24s_b4.wav -epf epf3_lab${i}_lis${j}_b4.epf
CopyAudio -l3254: out_24s_b4.wav dly_lab${i}_lis${j}_${MODE}.wav
```

2.6.2.2 aacPlus

LBR-A1

```
aacPlusenc-if a1_all_cat48m.wav -of out_14m_a1.3gp -br 14000 -m
aacPlusdec out_14m_a1.3gp out_14m_a1.wav
CopyAudio-l4166: out_14m_a1.wav dly_${MODE}.wav
```

LBR-A2

```
aacPlusenc-if a2_all_cat48s.wav -of out_18s_a2.3gp -br 18000
aacPlusdec out_18s_a2.3gp out_18s_a2.wav
CopyAudio-l4168: out_18s_a2.wav dly_${MODE}.wav
```

LBR-A3

```
aacPlusenc-if a3_all_cat48m.wav -of out_24m_a3.3gp -br 24000 -m
aacPlusdec out_24m_a3.3gp out_24m_a3.wav
```

```
CopyAudio-l4166: out_24m_a3.wav dly_${MODE}.wav
```

LBR-A4

```
aacPlusenc-if a4_all_cat48s.wav -of out_24s_a4.3gp -br 24000
aacPlusdec out_24s_a4.3gp out_24s_a4.wav
CopyAudio-l4166: out_24s_a4.wav dly_${MODE}.wav
```

LBR-B1

```
aacPlusenc-if b1_all_cat16m.wav -of out_14m_b1.3gp -br 14000 -m
aacPlusdec out_14m_b1.3gp out_14m_b1.wav
CopyAudio-l2081: out_14m_b1.wav dly_${MODE}.wav
```

LBR-B2

```
aacPlusenc-if b2_all_cat48s.wav -of out_18s_b2.3gp -br 18000
aacPlusdec out_18s_b2.3gp out_18s_b2.wav
CopyAudio-l4168: out_18s_b2.wav dly_${MODE}.wav
```

LBR-B3

```
aacPlusenc-if preproc/b3_all_cat48m.wav -of out_14m_b3.3gp -br 14000 -m
aacPlusdec out_14m_b3.3gp out_14m.wav epf3_lab${i}_lis${j}_b3.epf
CopyAudio-l4166: out_14m.wav dly_lab${i}_lis${j}_${MODE}.wav
```

LBR-B4

```
aacPlusenc-if b4_all_cat48s.wav -of out_24s_b4.3gp -br 24000
aacPlusdec out_24s_b4.3gp out_24s_b4.wav epf3_lab${i}_lis${j}_b4.epf
CopyAudio-l4166: out_24s_b4.wav dly_lab${i}_lis${j}_${MODE}.wav
```

2.6.2.3 ct

LBR-A1

```
ctenc-if a1_all_cat48m.wav -of out_14m.3gp -br 14000 -m
ctdec../tmp/out_14m.3gp ../tmp/out_14m.wav
CopyAudio -l4166: out_14m.wav dly_${MODE}.wav
```

LBR-A2

```
ctenc-if preproc/a2_all_cat48s.wav -of out_18s.3gp -br 18000
ctdec out_18s.3gp out_18s.wav
CopyAudio -l5122: out_18s.wav dly_${MODE}.wav
```

LBR-A3

```
ctenc-if a3_all_cat48m.wav -of out_24m.3gp -br 24000 -m
ctdec out_24m.3gp out_24m.wav
CopyAudio -l4166: out_24m.wav dly_${MODE}.wav
```

LBR-A4

```
ctenc-if a4_all_cat48s.wav -of out_24s.3gp -br 24000
ctdec out_24s.3gp out_24s.wav
CopyAudio -l5122: out_24s.wav dly_${MODE}.wav
```

LBR-B1

```
ctenc-if b1_all_cat16m.wav -of out_14m_16kHz.3gp -br 14000 -m
ctdec out_14m_16kHz.3gp out_14m_16kHz.wav
CopyAudio -l2081: out_14m_16kHz.wav dly_${MODE}.wav
```

LBR-B2

```
ctenc-if b2_all_cat48s.wav -of out_18s.3gp -br 18000
ctdec out_18s.3gp out_18s.wav
CopyAudio -l5122: out_18s.wav dly_${MODE}.wav
```

LBR-B3

```
ctenc-if b3_all_cat48m.wav -of out_14m_FER3.3gp -br 14000 -m
ctdec out_14m_FER3.3gp out_14m_FER3.wav epf3_lab${i}_lis${j}_b3.epf
CopyAudio -l4166: out_14m_FER3.wav dly_lab${i}_lis${j}_${MODE}.wav
```

LBR-B4

```
ctenc-if b4_all_cat48s.wav -of out_24s_FER3.3gp -br 24000
ctdec out_24s_FER3.3gp out_24s_FER3.wav epf3_lab${i}_lis${j}_b4.epf
CopyAudio -l5122: out_24s_FER3.wav dly_lab${i}_lis${j}_${MODE}.wav
```

2.6.2.4 AMRWB+

LBR-A1

```
enc_wbplus-mode 14m -if a1_all_cat24m.wav -of out_14m_a1.wb+
dec_wbplus -mode 14m -fs 24000 -if out_14m_a1.wb+ -of out_14m_a1.wav
CopyAudio -l668: out_14m_a1.wav dly_${MODE}.wav
```

LBR-A2

```
enc_wbplus-mode 18s -if a2_all_cat24s.wav -of out_18s_a2.wb+
dec_wbplus -mode 18s -fs 24000 -if out_18s_a2.wb+ -of out_18s_a2.wav
CopyAudio -l885: out_18s_a2.wav dly_${MODE}.wav
```

LBR-A3

```
enc_wbplus -mode 24m -if a3_all_cat24m.wav -of out_24m_a3.wb+
dec_wbplus -mode 24m -fs 24000 -if out_24m_a3.wb+ -of out_24m_a3.wav
CopyAudio -l668: out_24m_a3.wav dly_${MODE}.wav
```

LBR-A4

```
enc_wbplus -mode 24s -if a4_all_cat24s.wav -of out_24s_a4.wb+
dec_wbplus -mode 24s -fs 24000 -if out_24s_a4.wb+ -of out_24s_a4.wav
CopyAudio -l885: out_24s_a4.wav dly_${MODE}.wav
```

LBR-B1

```
enc_wbplus -mode 14m_lc -if b1_all_cat16m.wav -of out_14m_b1.wb+
dec_wbplus -mode 14m -fs 16000 -if out_14m_b1.wb+ -of out_14m_b1.wav
CopyAudio -l445: out_14m_b1.wav dly_${MODE}.wav
```

LBR-B2

```
enc_wbplus -mode 18s_lc -if b2_all_cat24s.wav -of out_18s_b2.wb+
dec_wbplus -mode 18s -fs 24000 -if out_18s_b2.wb+ -of out_18s_b2.wav
CopyAudio -l885: out_18s_b2.wav dly_${MODE}.wav
```

LBR-B3

```
enc_wbplus -mode 14m -if preproc/b3_all_cat24m.wav -of out_14m_b3_clean.wb+
eid -mode 14m -if out_14m_b3_clean.wb+ -of out_14m_b3_fer3.wb+ -fer epf3_lab${i}_lis${j}_b3.epf
dec_wbplus -mode 14m -fs 24000 -if out_14m_b3_fer3.wb+ -of out_14m_b3_fer3.wav
CopyAudio -l668: out_14m_b3_fer3.wav dly_lab${i}_lis${j}_${MODE}.wav
```

LBR-B4

```
enc_wbplus -mode 24s -if b4_all_cat24s.wav -of out_24s_b4_clean.wb+
eid -mode 24s -if out_24s_b4_clean.wb+ -of out_24s_b4_fer3.wb+ -fer epf3_lab${i}_lis${j}_b4.epf
dec_wbplus -mode 24s -fs 24000 -if out_24s_b4_fer3.wb+ -of out_24s_b4_fer3.wav
CopyAudio -l885: out_24s_b4_fer3.wav dly_lab${i}_lis${j}_${MODE}.wav
```

2.6.2.5 AMR-WB

LBR-A1

```
CopyAudio -F noheader a1_all_cat16m.wav a1_raw.wav
filterP341 a1_raw.wav a1_flt.wav 320
coder 3 a1_flt.wav out_14m_a1.amr
decoder out_14m_a1.amr out_14m_a1.wav
CopyAudio -l390: -t noheader -P "integer16,0,16000,little-endian,1,default" -F WAVE out_14m_a1.wav
dly_${MODE}.wav
```

LBR-A2

```
CopyAudio -F noheader a2_all_cat16m.wav a2_raw.wav
filterP341 a2_raw.wav a2_flt.wav 320
coder 5 a2_flt.wav out_18m_a2.amr
decoder out_18m_a2.amr out_18m_a2.wav
CopyAudio -l390: -t noheader -P "integer16,0,16000,little-endian,1,default" -F WAVE out_18m_a2.wav
dly_${MODE}.wav
```

LBR-A3

```
CopyAudio -F noheader a3_all_cat16m.wav a3_raw.wav
filterP341 a3_raw.wav a3_flt.wav 320
coder 8 a3_flt.wav out_24m_a3.amr
decoder out_24m_a3.amr out_24m_a3.wav
CopyAudio -l390: -t noheader -P "integer16,0,16000,little-endian,1,default" -F WAVE out_24m_a3.wav
dly_${MODE}.wav
```

LBR-A4

```
CopyAudio -F noheader a4_all_cat16m.wav a4_raw.wav
filterP341 a4_raw.wav a4_flt.wav 320
coder 8 a4_flt.wav out_24m_a4.amr
```

```
decoder out_24m_a4.amr out_24m_a4.wav
CopyAudio -l390: -t noheader -P "integer16,0,16000,little-endian,1,default" -F WAVE out_24m_a4.wav
dly_${MODE}.wav
```

LBR-B1

```
CopyAudio -F noheader b1_all_cat16m.wav b1_raw.wav
filterP341 b1_raw.wav b1_flt.wav 320
coder 3 b1_flt.wav out_14m_b1.amr
decoder out_14m_b1.amr out_14m_b1.wav
CopyAudio -l390: -t noheader -P "integer16,0,16000,little-endian,1,default" -F WAVE out_14m_b1.wav
dly_${MODE}.wav
```

LBR-B2

```
CopyAudio -F noheader b2_all_cat16m.wav b2_raw.wav
filterP341 b2_raw.wav b2_flt.wav
coder 5 b2_flt.wav out_18m_b2.amr
decoder out_18m_b2.amr out_18m_b2.wav
CopyAudio -l390: -t noheader -P "integer16,0,16000,little-endian,1,default" -F WAVE out_18m_b2.wav
dly_${MODE}.wav
```

LBR-B3

```
CopyAudio -F noheader b3_all_cat16m.wav b3_raw.wav
filterP341 b3_raw.wav b3_flt.wav 320
coder -mime 3 b3_flt.wav out_14m_b3.amr
amr_mms_eid epf3_lab${i}_lis${j}_b3.epf out_14m_b3.amr out_14m_b3_temp.amr
decoder -mime out_14m_b3_temp.amr out_14m_b3.wav
CopyAudio -l390: -t noheader -P "integer16,0,16000,little-endian,1,default" -F WAVE out_14m_b3.wav
dly_lab${i}_lis${j}_${MODE}.wav
```

LBR-B4

```
CopyAudio -F noheader b4_all_cat16m.wav b4_raw.wav
filterP341 b4_raw.wav b4_flt.wav 320
coder -mime 8 b4_flt.wav out_24m_b4.amr
amr_mms_eid epf3_lab${i}_lis${j}_b4.epf out_24m_b4.amr out_24m_b4_temp.amr
decoder -mime out_24m_b4_temp.amr out_24m_b4.wav
CopyAudio -l390: -t noheader -P "integer16,0,16000,little-endian,1,default" -F WAVE out_24m_b4.wav
dly_lab${i}_lis${j}_${MODE}.wav
```

2.6.3 Impaired channel processing

According the test plan, the impaired channel conditions for experiments B3 and B4 were simulated by applying error patterns to the bit streams during or before the decoding process. In the previous chapter this procedure (which is slightly different for the several codecs) can be seen. The error patterns were individual for each listener.

2.7 Post-processing

2.7.1 Up-sampling

The script “upsamp48.sh” was applied to all output files. This script performs

- Up-sampling to 48 kHz (if necessary)
- Conversion from Mono to Dual Mono (for the mono experiments)
- Level attenuation to a value of 0.93 for all files.

Because of the level attenuation, the open and hidden reference files were produced also by that script.

2.7.2 Split-up of processed material

The output files were split into separate files by using the “splitup.csh” tool which was provided by Ericsson. This script deconcatenates the files with smooth edges and makes the following simplification: treat stereo file as mono file and apply Hanning window of double length. The window size is 100ms=4800samples.

2.7.3 Blinding of Audio files

The blinding of the de-concatenated audio files was done in order to avoid identification of the files based on the filename. Only the open reference was to identify. All other versions of an item including the hidden reference and the anchors have been got a new filename where the condition was coded as “cond1“ to “cond9“.

The assignment of coding condition to the labels “cond1“ to “cond9“ was done different for each item. For that job the Host Lab created a script, which did the renaming process automatically. To resolve the blinding for later

analysis, a log file was written for each experiment, that contains information about the renaming. These files were kept secret and provided only to the analysis lab.

2.7.4 Cross-Checks

The concatenated audio files, as well as the output files after the encoding/decoding process were checked by calculating the checksums for all audio files. These checksums were compared with the results of the mirror host lab and the codec proponents. Only, if there were no differences the processing was continued.

For calculation of the checksums the raw data of an audio file were used. For calculation a script was used, which was provided by the mirror host lab:

```
#!/bin/bash

case $# in
        0) echo "usage: $0 wavfile(s)"; exit;;
esac

for i in $*
do
        ./Copyaudio.exe $i temp.raw 1>/dev/null 2>/dev/null
        ckout=`cksum temp.raw`
        echo $ckout | sed "{s'temp.raw'$i'}"
done
```

Some subjective spot checks were done for the single items after split-up process.

2.8 Generation of Session Files

The host lab has generated session files for all listeners at all test labs. These session files could be uses direct on the CRC-SEAQ MUSHRA software. The presentation order of the items was randomized as well as the assignment of the conditions to the buttons on the screen. The session file was a text file which could be easily adapted to other test systems. Here is an example for a training session file for CRC software:

```
[CRC-SEAQ]
Number of Trials:=4
Scale:=MUSHRA
[Trial 1]
Reference:=c:\etsi2003\exp_A1_Training\sbm_fi_x_9_A1_opref.wav
Sample 1:=c:\etsi2003\exp_A1_Training\sbm_fi_x_9_A1_cond6.wav
Sample 2:=c:\etsi2003\exp_A1_Training\sbm_fi_x_9_A1_cond5.wav
Sample 3:=c:\etsi2003\exp_A1_Training\sbm_fi_x_9_A1_cond4.wav
Sample 4:=c:\etsi2003\exp_A1_Training\sbm_fi_x_9_A1_cond7.wav
Sample 5:=c:\etsi2003\exp_A1_Training\sbm_fi_x_9_A1_cond2.wav
Sample 6:=c:\etsi2003\exp_A1_Training\sbm_fi_x_9_A1_cond3.wav
Sample 7:=c:\etsi2003\exp_A1_Training\sbm_fi_x_9_A1_cond8.wav
Sample 8:=c:\etsi2003\exp_A1_Training\sbm_fi_x_9_A1_cond1.wav
Sample 9:=
Sample 10:=
Sample 11:=
Sample 12:=
Trial Name:=Trial 1
Sample Rate:=48000
Channels:=2
BitDepth:=16
Byte Order:=Intel
[Trial 2]
Reference:=c:\etsi2003\exp_A1_Training\s_no_ft_9_A1_opref.wav
Sample 1:=c:\etsi2003\exp_A1_Training\s_no_ft_9_A1_cond1.wav
Sample 2:=c:\etsi2003\exp_A1_Training\s_no_ft_9_A1_cond2.wav
Sample 3:=c:\etsi2003\exp_A1_Training\s_no_ft_9_A1_cond3.wav
Sample 4:=c:\etsi2003\exp_A1_Training\s_no_ft_9_A1_cond7.wav
Sample 5:=c:\etsi2003\exp_A1_Training\s_no_ft_9_A1_cond4.wav
Sample 6:=c:\etsi2003\exp_A1_Training\s_no_ft_9_A1_cond6.wav
Sample 7:=c:\etsi2003\exp_A1_Training\s_no_ft_9_A1_cond8.wav
Sample 8:=c:\etsi2003\exp_A1_Training\s_no_ft_9_A1_cond5.wav
Sample 9:=
Sample 10:=
Sample 11:=
Sample 12:=
Trial Name:=Trial 2
Sample Rate:=48000
Channels:=2
```

```
BitDepth:=16
Byte Order:=Intel
[Trial 3]
Reference:=c:\etsi2003\exp_A1_Training\m_vo_x_1_A1_opref.wav
Sample 1:=c:\etsi2003\exp_A1_Training\m_vo_x_1_A1_cond6.wav
Sample 2:=c:\etsi2003\exp_A1_Training\m_vo_x_1_A1_cond4.wav
Sample 3:=c:\etsi2003\exp_A1_Training\m_vo_x_1_A1_cond8.wav
Sample 4:=c:\etsi2003\exp_A1_Training\m_vo_x_1_A1_cond5.wav
Sample 5:=c:\etsi2003\exp_A1_Training\m_vo_x_1_A1_cond7.wav
Sample 6:=c:\etsi2003\exp_A1_Training\m_vo_x_1_A1_cond1.wav
Sample 7:=c:\etsi2003\exp_A1_Training\m_vo_x_1_A1_cond2.wav
Sample 8:=c:\etsi2003\exp_A1_Training\m_vo_x_1_A1_cond3.wav
Sample 9:=
Sample 10:=
Sample 11:=
Sample 12:=
Trial Name:=Trial 3
Sample Rate:=48000
Channels:=2
BitDepth:=16
Byte Order:=Intel
[Trial 4]
Reference:=c:\etsi2003\exp_A1_Training\som_ot_x_9_A1_opref.wav
Sample 1:=c:\etsi2003\exp_A1_Training\som_ot_x_9_A1_cond2.wav
Sample 2:=c:\etsi2003\exp_A1_Training\som_ot_x_9_A1_cond6.wav
Sample 3:=c:\etsi2003\exp_A1_Training\som_ot_x_9_A1_cond1.wav
Sample 4:=c:\etsi2003\exp_A1_Training\som_ot_x_9_A1_cond4.wav
Sample 5:=c:\etsi2003\exp_A1_Training\som_ot_x_9_A1_cond5.wav
Sample 6:=c:\etsi2003\exp_A1_Training\som_ot_x_9_A1_cond3.wav
Sample 7:=c:\etsi2003\exp_A1_Training\som_ot_x_9_A1_cond8.wav
Sample 8:=c:\etsi2003\exp_A1_Training\som_ot_x_9_A1_cond7.wav
Sample 9:=
Sample 10:=
Sample 11:=
Sample 12:=
Trial Name:=Trial 4
Sample Rate:=48000
Channels:=2
BitDepth:=16
Byte Order:=Intel
```

The randomization in presentation order and the assignment of conditions to the buttons was done using a script, which can be found in the Annex.

2.9 Distribution of the Test files to the Listening Labs

The Files for the Listening Labs were distributed via FTP. The single sound files were zipped to a big file using WinZip without compression. Especially the Sessions B3, B4 and H3 have produced a big number of output files with an overall size of about 3 GB. Because of the 2GB limit in sending files via FTP, the zip-files were split into smaller 1 GB parts. These files could be reconstructed by using the cat command of Cygwin.

3 High-Rate Test

The processing for the high rate test was done in a very similar way as for the low rate test. The processing environment was the same, Cygwin on a Windows XP machine. Some of the processing steps were exactly the same as in the low rate test, which is indicated in the respective chapters.

3.1 Test Items

The items for the high rate test were selected by France Telecom. The item list consisted from 12 test and 4 additional training items:

No	Item	Description
1	c_01	classical
2	c_02	classical
3	p01	pop
4	p02	pop
5	si_01	single instrument
6	si_02	single instrument
7	sm_01	Speech with music
8	sm_02	Speech with music
9	sp_01	Speech
10	sp_02	Speech
11	sp_03	Speech
12	v_01	Vocal Music
13	c_09	Classical Music
14	p_09	Pop Music
15	si_09	Single Instrument
16	sp_09	Speech

3.2 Processing Systems

3.2.1 Coding conditions under test

3.2.1.1 aacPlus

- Encoder: Coding Technologies' aacPlus 3GPP Reference Encoder
  Build Oct 30 2003, 17:50:34
  (c) 2000 - 2003 Coding Technologies
  (c) 1999 Fraunhofer Gesellschaft IIS-A
- Decoder: Coding Technologies' aacPlus 3GPP Reference Decoder
  Build Oct 30 2003, 17:57:00
  (c) 2000 - 2003 Coding Technologies
  (c) 1999 Fraunhofer Gesellschaft IIS-A

3.2.1.2 ct

- Encoder: Coding Technologies' proprietary 3GPP Reference Encoder
  Build Oct 30 2003, 17:48:19
  (c) 2000 - 2003 Coding Technologies
  (c) 1999 Fraunhofer Gesellschaft IIS-A

- Decoder: Coding Technologies' proprietary 3GPP Reference Decoder
  Build Oct 30 2003, 17:54:3
  (c) 2000 - 2003 Coding Technologies
  (c) 1999 Fraunhofer Gesellschaft IIS-A

3.2.2 Reference coding conditions

3.2.2.1 AAC

- Encoder: MPEG-4 Audio Encoder (build Aug 13 2003)
  (C) 1997-2003 by Fraunhofer IIS-A
- Decoder: MPEG-4 Audio Decoder (build Aug 8 2003)
  (C) 1997-2003 by Fraunhofer IIS-A

3.2.2.2 RN

- Encoder: RealNetworks RealAudio8 low bitrate test executable,
  Oct 14th, 2003
- Decoder:RealNetworks RealAudio8 low bitrate test executable,
  Oct 14th, 2003

The coder.exe was used in encoder mode and afterwards in decoder mode. The bitstream file was checked in length to be sure that the average bitrate matches the requirements.

3.2.3 Anchor conditions

The two low pass filtered anchors have cutoff frequencies of 3.5 kHz and 7.0 kHz. They were produced using the "anchor.sh" script, which was provided by Ericsson.

3.2.4 Generation of error files

The error files were generated by using a tool which was provided by Ericsson. The tool was a shell script "genfer.csh". The seed value was provided by ETSI and had a value of "171094". In order to get individual error pattern files for each listener, the output file of the "genfer.csh" was processed with the "shiftcirc.csh" script.

```
./genfer.csh $SEED .01; cp ferpat_FER.01_SEED$SEED.dat ../ep/epf_1.epf
./genfer.csh $SEED .03; cp ferpat_FER.03_SEED$SEED.dat ../ep/epf_3.epf
```

3.3 Pre-processing

3.3.1 Sample rate conversions

For the sample rate conversions the anchor.sh script was used. For the Real Networks encoder, 22.05 kHz and 44.1 kHz sampling frequencies were required.

```
./resampaudio.exe -s 44100 ${CAT}48s.wav ${CAT}44s.wav
./resampaudio.exe -s 22050 ${CAT}48s.wav ${CAT}22s.wav
```

3.4 Concatenation of material

Before main processing, all files for an experiment were concatenated to one big wav file. the order of the items within the several experiments was

- 4 training files in alphabetical order followed by
- 12 test item files in alphabetical order
- additional silence file with a length of 290 ms to provide buffer for codec delays.

As an example, the time file for Experiment H1 looks as follows:

```
c_09_org.wav 931576
p_09_org.wav 960000
si_09_org.wav 934234
sp_09_org.wav 478852
c_01_org.wav 960002
c_02_org.wav 779004
p_01_org.wav 960002
p_02_org.wav 960002
si_01_org.wav 908738
```

```
si_02_org.wav 824948
sm_01_org.wav 831962
sm_02_org.wav 493482
sp_01_org.wav 960000
sp_02_org.wav 960002
sp_03_org.wav 480184
v_01_org.wav 940148
silence.wav 27838
```

The sampling rate for each concatenated file was 48 kHz, stereo. These files were used as input for the further processing steps. The input sampling rates for one of the reference codecs Therefore, the script “anchor.sh”, provided by Coding Technologies was used.

3.5 Main processing

3.5.1 Creation of anchors

The low frequency anchors were produced using the “anchors.sh” script. The processing for the was done by invoking the script with these parameters:

- low pass filtering with cutoff frequencies of 3500 Hz and 7000 Hz.

```
./anchor.sh -lp3500 ${CAT}48s.wav ${CAT}_anchor_35.wav
./anchor.sh -lp7000 ${CAT}48s.wav ${CAT}_anchor_7.wav
```

3.5.2 Coding of material

According the test plan, there were following bitrates to code:

Exp.	Operational mode	#Codecs in test	# reference codecs
1	32 kbps, stereo	2 (use case B encoder)	2, incl. RealAudio @ 32 kbit/s stereo
2	48 kbps, stereo	2 (use case B encoder)	2, incl. RealAudio @ 48 kbit/s stereo
3	32 kbps, stereo, 1% and 3% random frame loss	4 (2 candidates at 2 frame loss rates each)	2 (AAC-LC at 2 frame loss rates)

This was done using scripts which were calling the command line encoders and decoders as described in the following section.

3.5.2.1 AAC

```
mp4EvalCmdl.exe -b 32000 ${CAT}48s.wav ${aac_t}out_32.mp4
mp4dec.exe -if ${aac_t}out_32.mp4 -of ${aac_t}out_32.wav
CopyAudio -l3278: ${aac_t}out_32.wav ${aac}ref_1_32.wav

mp4EvalCmdl.exe -b 48000 ${CAT}48s.wav ${aac_t}out_48.mp4
mp4dec.exe -if ${aac_t}out_48.mp4 -of ${aac_t}out_48.wav
$CPY -l3442: ${aac_t}out_48.wav ${aac}ref_1_48.wav
```

3.5.2.2 aacPlus

```
aacPlusenc.exe -if ${CAT}48s.wav -of ${aacplus_t}out_32.3gp -br 32000
aacPlusdec.exe ${aacplus_t}out_32.3gp ${aacplus_t}out_32.wav
$CPY -l4166: ${aacplus_t}out_32.wav ${aacplus}cand_1_32.wav

aacPlusenc.exe -if ${CAT}48s.wav -of ${aacplus_t}out_48.3gp -br 48000
aacPlusdec.exe ${aacplus_t}out_48.3gp ${aacplus_t}out_48.wav
$CPY -l4166: ${aacplus_t}out_48.wav ${aacplus}cand_1_48.wav
```

3.5.2.3 ct

```
ctenc.exe -if ${CAT}48s.wav -of ${ct_t}out_32.3gp -br 32000
ctdec.exe ${ct_t}out_32.3gp ${ct_t}out_32.wav
$CPY -l5122: ${ct_t}out_32.wav ${ct}cand_2_32.wav

ctenc.exe -if ${CAT}48s.wav -of ${ct_t}out_48.3gp -br 48000
ctdec.exe ${ct_t}out_48.3gp ${ct_t}out_48.wav
$CPY -l4166: ${ct_t}out_48.wav ${ct}cand_2_48.wav
```

3.5.2.4 RealNetworks

```
coder -b 32 -e ${CAT}22s.wav ${real_t}out_32.bit
coder -b 32 -d ${real_t}out_32.bit ${real}ref_2_32.wav

coder -b 48 -e ${CAT}44s.wav ${real_t}out_48.bit
coder -b 48 -d ${real_t}out_48.bit ${real}ref_2_48.wav
```

For the Real Networks codec, there was no need for delay compensation, because the input and output files were time aligned.

3.5.3 Creation of anchors

Two low pass filtered anchors were produced by invoking the anchor script with these parameters:

```
./anchor.sh -lp3500 ${CAT}48s.wav ${CAT}_anchor_35.wav
./anchor.sh -lp7000 ${CAT}48s.wav ${CAT}_anchor_7.wav
```

3.5.4 Impaired channel processing

For processing the impaired versions of the signals, the following script sample was used, which produced individual audio files for each listener

```
for i in 3 6 #lab
do
	for j in 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 #lis
	do
		# aac @32 1%
		mp4dec -if ${aac_t}out_32.mp4 -of ${aac_t}out_32.wav -epf
		epf1_lab${i}_lis${j}_H3.epf
		$CPY -l3278: ${aac_t}out_32.wav ${aac}ref_1_32.wav

		# aac+ @32 1%
		aacPlusdec ${aacplus_t}out_32.3gp ${aacplus_t}out_32.wav
		epf1_lab${i}_lis${j}_H3.epf
		$CPY -l4166: ${aacplus_t}out_32.wav ${aacplus}cand_1_32.wav

		# ct @32 1%
		ctdec ${ct_t}out_32.3gp ${ct_t}out_32.wav epf1_lab${i}_lis${j}_H3.epf
		$CPY -l5122: ${ct_t}out_32.wav ${ct}cand_2_32.wav

		# aac @32 3%
		mp4dec -if ${aac_t}out_32.mp4 -of ${aac_t}out_32.wav -epf
		epf3_lab${i}_lis${j}_H3.epf
		$CPY -l3278: ${aac_t}out_32.wav ${aac}ref_1_32.wav

		# aac+ @32 3%
		aacPlusdec ${aacplus_t}out_32.3gp ${aacplus_t}out_32.wav
		epf3_lab${i}_lis${j}_H3.epf
		$CPY -l4166: ${aacplus_t}out_32.wav ${aacplus}cand_1_32.wav

		# ct @32 3%
		ctdec ${ct_t}out_32.3gp ${ct_t}out_32.wav
		epf3_lab${i}_lis${j}_H3.epf
		$CPY -l5122: ${ct_t}out_32.wav ${ct}cand_2_32.wav

	done
done
```

3.6 Post-processing

3.6.1 Up-sampling

The script upsamp48.sh was applied to all output files. This script performs

- Up-sampling to 48 kHz (if necessary)
- Level attenuation to a value of 0.93 for all files.

Because of the level attenuation, the open and hidden reference files were produced also by that script in order to get equal level for all versions of a signal.

3.6.2 Split-up of processed material

The output files were split into separate files by using the “splitup.csh” tool (can be found in Annex 5) which was provided by Ericsson. This script deconcatenates the files with smooth edges and makes the following simplification: treat stereo file as mono file and apply Hanning window of double length. The window size is 100ms=4800samples.

3.6.3 Blinding of Audio files

The blinding of the de-concatenated audio files was done in order to avoid identification of the files based on the filename. Only the open reference was to identify. All other versions of an item including the hidden reference and the anchors have been got a new filename where the condition was coded as “cond1“ to “cond9“.

The assignment of coding condition to the labels “cond1“ to “cond9“ was done different for each item. For that job the Host Lab created a script, which did the renaming process automatically. To resolve the blinding for later analysis, a log file was written for each experiment, that contains information about the renaming. These files were kept secret and provided only to the analysis lab.

3.6.4 Cross-Checks

see 2.7.4

3.7 Generation of Session Files

see 2.8

3.8 Distribution of the Test files to the Listening Labs

see 2.9

4 References

[1] 3GPP TDoc S4-030821 PSS/MMS High-Rate Audio Selection Test and Processing Plan Version 2.2.0
[2] 3GPP Tdoc S4-030824 AMR-WB+ and PSS/MMS Low-Rate Audio Selection Test and Processing Plan Version 2.2
[3] RECOMMENDATION ITU-R BS.1534, Method for the subjective assessment of intermediate quality level of coding systems
[4] AFsp version 7r1, Peter Kabal, Electrical & Computer Engineering, McGill University
[5] ITU T Recommendation G.191, Software tools for speech and audio coding standardization (STL2000)

Annex

1 Low Rate Items

No.	item	genre
1	m_ch_x_1	music, choir
2	m_cl_x_1	music, classical
3	m_cl_x_2	music, classical
4	m_ot_x_1	music, other
5	m_ot_x_2	music, other
6	m_ot_x_3	music, other
7	m_ot_x_4	music, other
8	m_ot_x_5	music, other
9	m_ot_x_6	music, other
10	m_ot_x_7	music, other
11	m_ot_x_8	music, other
12	m_ot_x_9	music, other
13	m_ot_x_a	music, other
14	m_ot_x_b	music, other
15	m_po_x_1	music, pop
16	m_po_x_2	music, pop
17	m_po_x_3	music, pop
18	m_po_x_4	music, pop
19	m_po_x_5	music, pop
20	m_po_x_6	music, pop
21	m_po_x_7	music, pop
22	m_si_x_1	music, single instr.
23	m_si_x_2	music, single instr.
24	m_si_x_3	music, single instr.
25	sbm_js_x_1	speech between music
26	sbm_js_x_2	speech between music
27	sbm_ms_x_1	speech between music
28	sbm_sj_x_1	speech between music
29	sbm_sm_x_1	speech between music
30	sbm_sm_x_2	speech between music
31	sbm_sm_x_3	speech between music
32	sbm_sm_x_4	speech between music

No.	item	genre
33	sbm_sm_x_5	speech between music
34	sbm_sm_x_6	speech between music
35	som_ad_x_1	speech over music
36	som_fi_x_1	speech over music
37	som_fi_x_2	speech over music
38	som_fi_x_3	speech over music
39	som_fi_x_4	speech over music
40	som_nt_x_1	speech over music
41	som_ot_x_1	speech over music
42	som_ot_x_2	speech over music
43	som_ot_x_3	speech over music
44	som_ot_x_4	speech over music
45	som_ot_x_5	speech over music
46	som_ot_x_6	speech over music
47	s_cl_2t_1	speech, clean
48	s_cl_2t_2	speech, clean
49	s_cl_2t_3	speech, clean
50	s_cl_2t_4	speech, clean
51	s_cl_2t_5	speech, clean
52	s_cl_ft_3	speech, clean
53	s_cl_mt_1	speech, clean
54	s_cl_mt_2	speech, clean
55	s_no_2t_1	speech, noisy
56	s_no_2t_2	speech, noisy
57	s_no_2t_3	speech, noisy
58	s_no_ft_1	speech, noisy
59	s_no_ft_2	speech, noisy
60	s_no_ft_3	speech, noisy
61	s_no_ft_4	speech, noisy
62	s_no_mt_1	speech, noisy

2 Low Rate Items - Assignment to Experiments

Experiment A1

SetA1a	m_ot_x_8_org m_ot_x_a_org m_po_x_5_org m_po_x_7_org	Set A1b	m_ot_x_9_org m_ot_x_b.org m_po_x_6_org m_si_x_3_org
	s_cl_2t_4_org s_cl_2t_3_org s_no_2t_1_org s_no_ft_1_org		s_cl_2t_5_org s_cl_mt_2_org s_no_2t_2_org s_no_ft_2_org
	sbm_js_x_1_org sbm_ms_x_1_org		sbm_sj_x_1_org sbm_sm_x_6_org
	som_fi_x_4_org som_ot_x_4_org		som_ot_x_5_org som_ot_x_6_org

Experiment A2

Set A2a	m_ot_x_4_org m_ot_x_5_org m_po_x_2_org m_po_x_3_org	SetA2 b	m_ot_x_6_org m_ot_x_7.org m_po_x_4_org m_si_x_2_org
	s_cl_2t_4_org s_cl_ft_3_org s_no_2t_3_org s_no_mt_1_org		s_cl_2t_5_org s_cl_mt_2_org s_no_ft_4_org s_no_ft_3_org
	sbm_js_x_1_org sbm_sm_x_4_org		sbm_js_x_2_org sbm_sm_x_5_org
	som_fi_x_3_org som_ot_x_2_org		som_ot_x_3_org som_ad_x_1_org

Experiment A3

SetA3 a	m_ot_x_2_org m_si_x_1_org m_po_x_2_org m_po_x_1_org	SetA3 b	m_ot_x_3_org m_po_x_3.org m_po_x_4_org m_si_x_2_org
	s_cl_2t_1_org s_cl_ft_3_org s_no_2t_1_org s_no_mt_1_org		s_cl_2t_2_org s_cl_mt_2_org s_no_2t_2_org s_no_ft_4_org
	sbm_ms_x_1_org sbm_sm_x_2_org		sbm_js_x_2_org sbm_sm_x_5_org
	som_nt_x_1_org som_ot_x_2_org		som_ot_x_3_org som_ot_x_1_org

Experiment A4

SetA4 a	m_ot_x_2_org m_si_x_1_org m_ch_x_1_org m_po_x_1_org	SetA4 b	m_ot_x_3_org m_ot_x_1.org m_cl_x_1_org m_cl_x_2_org
	s_cl_2t_1_org s_no_2t_1_org s_cl_mt_1_org s_no_ft_2_org		s_cl_2t_2_org s_no_ft_1_org s_no_2t_2_org s_cl_2t_3_org
	sbm_ms_x_1_org sbm_sm_x_2_org		sbm_sm_x_1_org sbm_sm_x_3_org
	som_nt_x_1_org som_fi_x_1_org		som_fi_x_2_org som_ot_x_1_org

Experiment B1

Set B1 a	m_ch_x_1_org m_cl_x_1_org m_po_x_2_org m_po_x_3_org	SetB1 b	m_po_x_5_org m_po_x_6.org m_po_x_4_org m_si_x_2_org
	s_cl_2t_4_org s_cl_ft_3_org s_no_2t_2_org s_no_ft_2_org		s_cl_2t_2_org s_cl_mt_1_org s_no_2t_3_org s_no_ft_4_org
	sbm_js_x_2_org sbm_ms_x_1_org		sbm_sj_x_1_org sbm_sm_x_4_org
	som_ad_x_1_org som_ot_x_4_org		som_ot_x_2_org som_ot_x_6_org

Experiment B2

SetB2 a	m_ot_x_1_org m_ot_x_8_org m_ot_x_a_org m_cl_x_2_org	SetB2 b	m_ot_x_9_org m_ot_x_b.org m_cl_x_1_org m_ch_x_1_org
	s_cl_2t_4_org s_cl_2t_5_org s_no_2t_3_org s_no_ft_2_org		s_cl_2t_3_org s_cl_mt_1_org s_no_ft_1_org s_no_ft_3_org
	sbm_js_x_1_org sbm_sm_x_4_org		sbm_sm_x_1_org sbm_sm_x_3_org
	som_fi_x_1_org som_ot_x_5_org		som_ot_x_6_org som_fi_x_2_org

Experiment B3

SetB3 a	m_ot_x_1_org m_ot_x_5_org m_ot_x_7_org m_cl_x_1_org	SetB3 b	m_ot_x_4_org m_ot_x_6.org m_ch_x_1_org m_cl_x_2_org
	s_cl_2t_1_org s_cl_2t_3_org s_no_2t_3_org s_no_ft_4_org		s_cl_2t_2_org s_cl_mt_1_org s_no_ft_3_org s_no_mt_1_org
	sbm_ms_x_1_org sbm_js_x_2_org		sbm_js_x_1_org sbm_sm_x_1_org
	som_ad_x_1_org som_fi_x_2_org		som_fi_x_3_org som_fi_x_1_org

Experiment B4

SetB4 a	m_ot_x_4_org m_ot_x_8_org m_po_x_2_org m_po_x_7_org	SetB4 b	m_ot_x_5_org m_ot_x_9.org m_po_x_3_org m_si_x_3_org
	s_cl_2t_1_org s_cl_2t_5_org s_no_2t_1_org s_no_ft_3_org		s_cl_2t_3_org s_cl_mt_2_org s_no_ft_1_org s_no_mt_1_org
	sbm_js_x_2_org sbm_sm_x_2_org		sbm_sm_x_1_org sbm_sm_x_6_org
	som_ot_x_5_org som_fi_x_3_org		som_fi_x_4_org som_ot_x_3_org

3 Script Anchor.sh

```
#!/bin/bash
#
# anchor.sh according to Section 8 of S4-030677_AMRWB+TestPlan_v0.8.doc
#
# HP20031103 20031104
#

RES=./ResampAudio.exe
CPY=./CopyAudio.exe
INF=./InfoAudio.exe
TMPFILE=anchor_tmp_file.wav

if [ $# -lt 2 ]; then
    echo "usage: `basename $0` [-lp<cut-off>] [-s<stereo_degree>] [-fsout[16|24]] [-
monooout] <infile.wav> <outfile> "
    echo ""
    exit
fi
echo $*

INARG=$(($# - 1))
OUTARG=$#
INFILE="${!INARG}"
OUTFILE="${!OUTARG}"
OUTEXT=${OUTFILE:(-4):4}

CUTOFF=0
STEREO=0
OUTFS=0
OUTMONO=0
for (( I=1 ; I<$#-1 ; I++ )) ; do
    OPT=${!I}
    if [[ ${OPT:0:3} == "-lp" ]] ; then
        CUTOFF=${OPT:3}
    elif [[ ${OPT:0:2} == "-s" ]] ; then
        STEREO=${OPT:2}
    elif [[ $OPT == "-fsout24" ]] ; then
        OUTFS=24000
    elif [[ $OPT == "-fsout16" ]] ; then
        OUTFS=16000
    elif [[ $OPT == "-monoout" ]] ; then
        OUTMONO=1
    else
        echo "ERROR: Unknown option "$OPT
        echo ""
        exit 1
    fi
done

RESOPTS=""
if [[ $CUTOFF == 3500 ]] ; then
    RESOPTS=$RESOPTS" -f cutoff=0.0729167"
elif [[ $CUTOFF == 7000 ]] ; then
    RESOPTS=$RESOPTS" -f cutoff=0.145833"
elif [[ $CUTOFF != 0 ]] ; then
    echo "ERROR: Wrong cut-off "$CUTOFF
    echo ""
    exit 1
fi
if [[ $OUTFS == 24000 ]] ; then
    RESOPTS=$RESOPTS" -s 24000"
elif [[ $OUTFS == 16000 ]] ; then
    RESOPTS=$RESOPTS" -s 16000"
else
    RESOPTS=$RESOPTS" -i 1"
fi

if [[ $OUTMONO != 0 && $STEREO != 0 ]] ; then
```

```
    echo "ERROR: Options -monoout and -s<stereo_degree> given simultaneously"
    echo ""
    exit 1
fi

CPYOPTS=""
if [[ $STEREO == 6 ]] ; then
    CPYOPTS=$CPYOPTS" -cA 3/4*A+1/4*B -cB 1/4*A+3/4*B"
elif [[ $STEREO == 12 ]] ; then
    CPYOPTS=$CPYOPTS" -cA 5/8*A+3/8*B -cB 3/8*A+5/8*B"
elif [[ $STEREO != 0 ]] ; then
    echo "ERROR: Wrong stereo-degree "$STEREO
    echo ""
    exit 1
fi
if [[ $OUTMONO == 1 ]] ; then
    CPYOPTS=$CPYOPTS" -cA 1/2*A+1/2*B"
    #use below for 2-channel mono presentation
    #CPYOPTS=$CPYOPTS" -cA 1/2*A+1/2*B -cB 1/2*A+1/2*B"
fi

if [[ $OUTEXT == ".raw" ]] ; then
    CPYOPTS=$CPYOPTS" -F noheader -D integer16"
fi

echo ""
echo "ANCHOR: checking input file "$INFILE
echo ""
if ! { $INF $INFILE | grep "Sampling frequency: 48000 Hz" ; } ; then
    echo "ERROR: Sampling frequency not 48000 Hz"
    echo ""
    exit 1
fi
if ! { $INF $INFILE | grep "Number of channels: 2" ; } ; then
    echo "ERROR: Number of channels not 2"
    echo ""
    exit 1
fi

if [[ $OUTFS != 0 || $CUTOFF != 0 ]] ; then
    echo ""
    echo "ANCHOR: filtering/resampling ..."
    echo ""
    echo $RES $RESOPTS $INFILE $TMPFILE
    $RES $RESOPTS $INFILE $TMPFILE
else
    TMPFILE=$INFILE
fi

echo ""
echo "ANCHOR: copying/downmixing ..."
echo ""
echo $CPY $CPYOPTS $TMPFILE $OUTFILE
$CPY $CPYOPTS $TMPFILE $OUTFILE

if [[ $TMPFILE != $INFILE ]] ; then
    rm $TMPFILE
fi

echo ""
echo "ANCHOR: done ..."
echo ""

exit 0
```

4 Script upsamp48.sh

```
#!/bin/bash
#
# upsamp48.sh according to Section 8 of S4-030677_AMRWB+TestPlan_v0.8.doc
#
# HP20031103 20031104
#

RES=./ResampAudio.exe
CPY=./CopyAudio.exe
INF=./InfoAudio.exe
TMPFILE=upsamp48_tmp_file.wav

ATTEN="0.93"

if [ $# -lt 2 ]; then
    echo "usage: `basename $0` [-fs<fsamp>] <infile> <outfile.wav> "
    echo ""
    exit
fi

INARG=$(($# - 1))
OUTARG=$#
INFILE="${!INARG}"
OUTFILE="${!OUTARG}"
INEXT=${INFILE:(-4):4}

INFS=0
for (( I=1 ; I<$#-1 ; I++ )) ; do
    OPT=${!I}
    if [[ ${OPT:0:3} == "-fs" ]] ; then
        INFS=${OPT:3}
    else
        echo "ERROR: Unknown option "$OPT
        echo ""
        exit 1
    fi
done

RAWOPTS=""
if [[ $INEXT == ".raw" ]] ; then
    if [[ $INFS == 0 ]] ; then
        echo "ERROR: Option -fs<fsamp> missing"
        echo ""
        exit 1
    fi
    RAWOPTS=" -P integer16,0,"$INFS",native,1,1/32768"
    INCH=1
else
    INFS=0
    if $INF $INFILE | grep "Sampling frequency: 48000 Hz" > /dev/null ; then
        INFS=48000
    fi
    INCH=1
    if $INF $INFILE | grep "Number of channels: 2" > /dev/null ; then
        INCH=2
    fi
fi

if [[ $INCH == 1 ]] ; then
    if [[ $INFS == 48000 ]] ; then
        CPYOPTS=" -cA "$ATTEN"*A -cB "$ATTEN"*A"$RAWOPTS
        echo ""
        echo "UPSAMPLE48: gain-adjustment/upmixing ..."
        echo ""
        echo $CPY $CPYOPTS $INFILE $OUTFILE
        $CPY $CPYOPTS $INFILE $OUTFILE
    else
        RESOPTS=" -s 48000 -g "$ATTEN""$RAWOPTS
```

```
        echo ""
        echo "UPSAMPLE48: gain-adjustment/resampling ..."
        echo ""
        echo $RES $RESOPTS $INFILE $TMPFILE
        $RES $RESOPTS $INFILE $TMPFILE
        echo ""
        echo "UPSAMPLE48: upmixing ..."
        echo ""
        CPYOPTS=" -cA A -cB A"
        echo $CPY $CPYOPTS $TMPFILE $OUTFILE
        $CPY $CPYOPTS $TMPFILE $OUTFILE
        rm $TMPFILE
    fi
else
    if [[ $INFS == 48000 ]] ; then
        CPYOPTS=" -g "$ATTEN
        echo ""
        echo "UPSAMPLE48: gain-adjustment ..."
        echo ""
        echo $CPY $CPYOPTS $INFILE $OUTFILE
        $CPY $CPYOPTS $INFILE $OUTFILE
    else
        RESOPTS=" -s 48000 -g "$ATTEN
        echo ""
        echo "UPSAMPLE48: gain-adjustment/resampling ..."
        echo ""
        echo $RES $RESOPTS $INFILE $OUTFILE
        $RES $RESOPTS $INFILE $OUTFILE
    fi
fi

echo ""
echo "UPSAMPLE48: done ..."
echo ""

exit 0
```

5 Script splitup.csh

```
#!/bin/csh
if ( $# != 3 ) then
    echo "Usage:                                                                v. 0.1"
    echo "$0 <infile> <proc_tag> <timfile>"
    echo " "
    echo " This script will de-concatenate a 48000 Hz wav input file"
    echo "   <infile>   : ./cand_1/processed/all_cat_cand_1_A1.wav (example)"
    echo "   <proc_tag> : <codec_id>_<exp_id>, e.g.: cand_1_A1"
    echo "   <timfile>  : ./preproc/all_cat.tim (typically)"
    echo " "
    echo " Note: this script is to be executed from the processing root!"
    echo "       mono-files will be converted to 2-channel files."
    exit -1
endif

#### pathes to executables to be adjusted to local needs ##############################
set RESAMPAUDIO = ResampAudio.exe
set COPYAUDIO = CopyAudio.exe
set ASTRIP = astrip
########################################################################################

set infile = $1
set proctag = $2
set timfile = $3

# set outdir
set base = `basename $infile`
set outdir = `echo $infile | sed s/$base//`

# check sampling rate
set info = `file $infile`
while ( "$info[3]" != "Hz" )
    shift info
    if ( $#info < 3 ) then
       echo "incorrect wav header in file $infile"
       exit -1
    endif
end
if ( $info[2] != 48000) then
    echo "Wrong sampling rate of $infile. Must be 48000 Hz."
    exit -1
endif

# check input channels
if ( $info[1] == stereo ) then
    set channels = 2
else
    set channels = 1
endif

# make mono files to 2 channel files
if ( $channels == 1 ) then
    $COPYAUDIO  --chanA="A" --chanB="A" $infile tmpfile.wav
    set channels = 2
else
    cp $infile tmpfile.wav
endif


# strip the wav header
$COPYAUDIO -F noheader tmpfile.wav catfile.raw

set acclen=1
@ i=1
set line=`sed -n $i\p $timfile`
while ("$line" != "")
    @ i++
    echo $line
```

```
    # get length information
    set len = $line[2]
    echo $len

    # deconcatenate with smooth edges
    #   make the following simplification:
    #     treat stereo file as mono file and apply Hanning window
    #     of double length
    #   The window size is 100ms=4800samples (rather than 10 ms)
    @ wlen = 4800 * $channels
    $ASTRIP -smooth -wlen $wlen -sample -start $acclen -n $len catfile.raw tmpfile.raw
    set acclen = `echo $acclen + $len | bc`

    # regenerate wav file
    set outname = `echo $line[1] | sed s/.wav/_$proctag.wav/`
    set outfile = $outdir/$outname
    $COPYAUDIO -t noheader -P "integer16,0,48000., native, 2, default" tmpfile.raw
$outfile

    # next
    set line=`sed -n $i\p $timfile`
end

# clean up
rm -f catfile.raw tmpfile.raw tmpfile.wav
```

6 Script Shiftcirc.csh

```
#!/bin/tcsh

if ( $# == 3 ) then
    set offs=$1
    set in=$2
    set out=$3
else if ( $# == 4 ) then
    set s=$1
    set p=$2
    set e=$3
    set in=$4
    set len = `wc -l $in`
    set offs=`echo "($s*12345 + $p*31415 + $e*27183) % $len[1]"|bc`
    echo "offs = $offs"
    exit
else if ( $# == 5 ) then
    set s=$1
    set p=$2
    set e=$3
    set in=$4
    set out=$5

    set len = `wc -l $in`
    set offs=`echo "($s*12345 + $p*31415 + $e*27183) % $len[1]"|bc`

else
    echo "Usage:                                                          v. 0.2"
    echo "$0 <listener_id> <lab_id> <exp_id> <inpat>"
    echo "     returning 'offs'"
    echo "or"
    echo "$0 <offs> <inpat> <outpat>"
    echo "or"
    echo "$0 <listener_id> <lab_id> <exp_id> <inpat> <outpat>"
    echo
    exit
endif

set len = `wc -l $in`
set len=$len[1]

echo $offs
if ( $offs >= $len[1] || $offs < 0 ) then
    echo "offset error: must be >= 0 and < length(inpat)"
    exit 1
endif

#do the shift:
@ offs++
sed -n -e $offs,$len\p $in > $out
@ offs--
if ( $offs > 0 ) then
sed -n -e 1,$offs\p $in >> $out
endif
```