UNIVERZITET U NOVOM SADU
FAKULTET TEHNIČKIH NAUKA
Odsek za elektrotehniku i računarstvo
Institut za računarstvo i automatiku
Katedra za računarsku tehniku i računarske komunikacije

Arhitekture i algoritmi DSP II

Projektni zadatak:

Realizacija algoritma kombinovanja kanala na Cirrus Logic DSP platformi

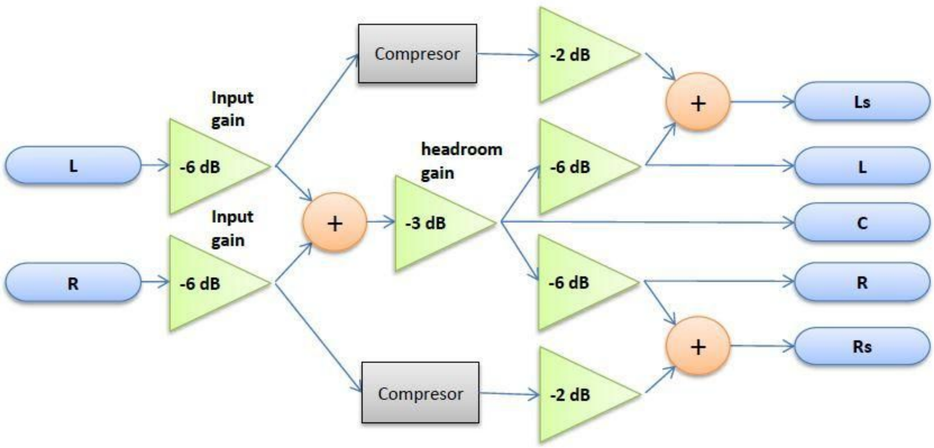
Student: Mladen Matić

Mentor: Jelena Kovačević

Novi Sad, 2018.

Sadržaj

1. Opis zadatka



control	Enable	Input gain	Headroom gain	Output Mode
values	On/Off	From 0 to -∞ dB	From 0 to -∞ dB	2_0_0, 2_0_0, 0_2_0, 0_2_0, 3_2_0
default value	On	-6 dB	-3 dB	2_0_0

Tabela 1 – Korisničke kontrole

U zadatku (slika iznad) je bilo potrebno doći do implementacije programske podrške za Cirrus Logic DSP kroz modele, od nultog do trećeg, i na kraju finalni kod integrisati na ciljnu platformu. Sama programska podrška izvršava datu funkciju za smanjenje tonova koji prelaze zadatu granicu određeni, racionalan, broj puta na određenim kanalima. Na početku, iz .wav datoteke se učitava šesnaest po šesnaest bita, a zatim se vrši obrada nad njima. Korisnik može uneti da li želi da se uopšte vrši bilo kakva obrada pomoću prvog dodatnog parametra komandne linije, zatim može uticati na količinu slabljenja signala unošenjem vrednosti, u decibelima, za "input_gain" i "headroom_gain", a takođe, može uticati i na to koliko i kojih kanala će biti na izlazu, prosleđivanjem nule, jedinice ili dvojke za, redom, modove "2_0_0", "0_2_0" i "3_2_0". Prvi mod označava dva izlazna kanala, levi i desni, drugi mod označava dva izlazna kanala, levi "surround" i desni "surround", a treći na izlazu daje pet kanala, levi, centralni, desni, levi "surround" i desni "surround".

Implementaciju treba početi od modela nula, koji sadrži osnovnu implementaciju bez preterane optimizacije. Cilj modela je da se napravi program koji jednostavno radi, da bi se na njegovom primeru napravio model jedan.

Model jedan treba da bude optimizovan na nivou “C” programskog jezika, osnovno, prelazeći sa pristupa indeksiranjem niza na pristup pokazivačima.

Model dva zahteva prelazak sa aritmetike pokretnog zareza na aritmetiku nepokretnog zareza. Pri toj optimizaciji je potrebno koristiti date referentne kodove u kojima su deklarisane promenljive “DSPint”, “DSPfract” i “DSPaccum”.

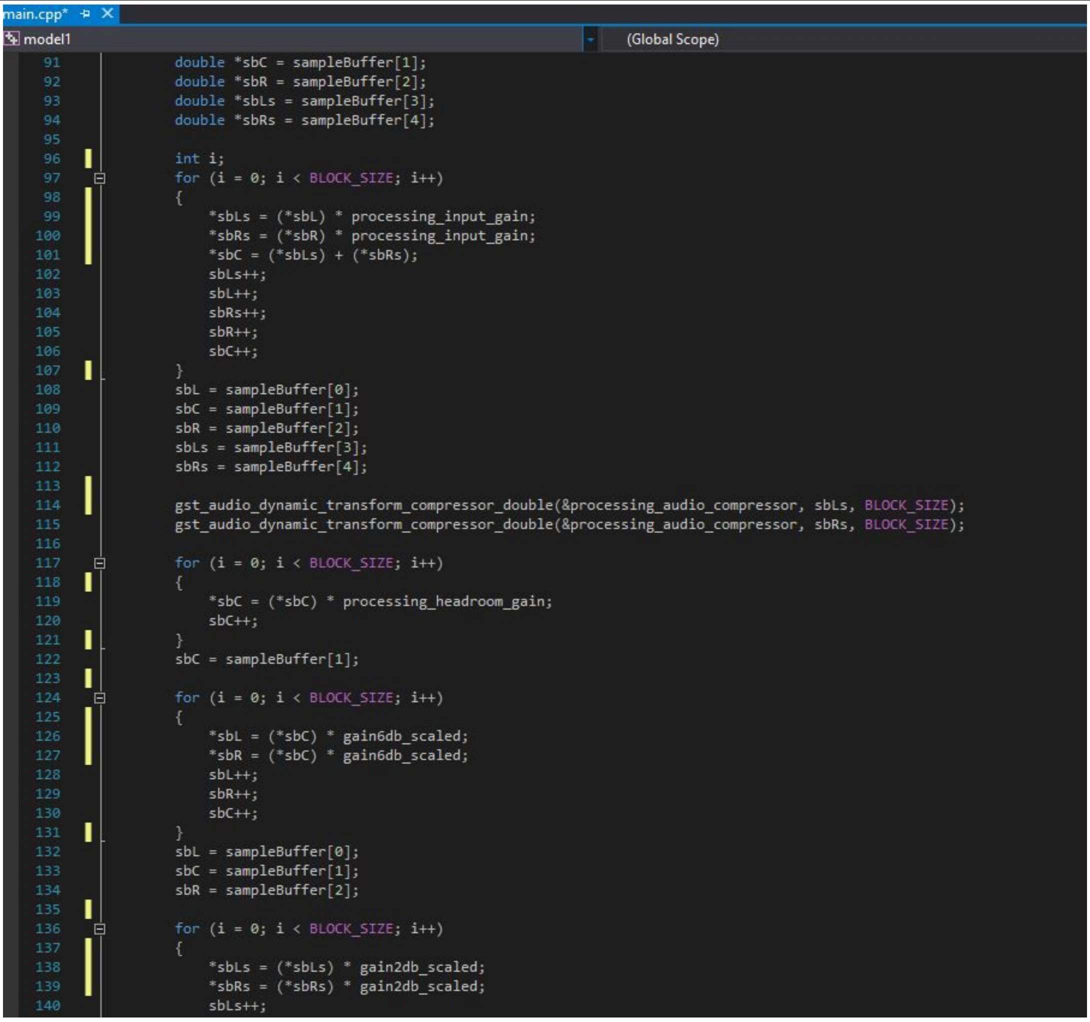
Treći model zahteva prelazak u “CLIDE” radno okruženje i rad sa simulatorom. Potrebno je elemente smestiti u određene memorijske zone. Ciljani digitalni signal procesor sadrži razdvojenu memoriju za podatke na “X” i “Y” u koje se ti elementi smeštaju.

2. Opis realizacije

Realizacija zadatka započeta je modelom nula:

```
main.cpp
model0                                        (Global Scope)

enum processing_output_mode_t { MODE2_0_0, MODE0_2_0, MODE3_2_0 };
processing_output_mode_t processing_output_mode = MODE3_2_0;

// TODO Monday: indexing with pointers.
void processing()
{
    int i;
    for (i = 0; i < BLOCK_SIZE; i++)
    {
        sampleBuffer[3][i] = sampleBuffer[0][i] * processing_input_gain;
        sampleBuffer[4][i] = sampleBuffer[2][i] * processing_input_gain;
        sampleBuffer[1][i] = sampleBuffer[3][i] + sampleBuffer[4][i];
    }

    gst_audio_dynamic_transform_compressor_double(&processing_audio_compressor, sampleBuffer[3], BLOCK_SIZE);
    gst_audio_dynamic_transform_compressor_double(&processing_audio_compressor, sampleBuffer[4], BLOCK_SIZE);

    for (i = 0; i < BLOCK_SIZE; i++)
    {
        sampleBuffer[1][i] = sampleBuffer[1][i] * processing_headroom_gain;
    }

    for (i = 0; i < BLOCK_SIZE; i++)
    {
        sampleBuffer[0][i] = sampleBuffer[1][i] * gain6db_scaled;
        sampleBuffer[2][i] = sampleBuffer[1][i] * gain6db_scaled;
    }

    for (i = 0; i < BLOCK_SIZE; i++)
    {
        sampleBuffer[3][i] = sampleBuffer[3][i] * gain2db_scaled;
        sampleBuffer[4][i] = sampleBuffer[4][i] * gain2db_scaled;
    }

    for (i = 0; i < BLOCK_SIZE; i++)
    {
        sampleBuffer[3][i] = sampleBuffer[0][i] + sampleBuffer[3][i];
        sampleBuffer[4][i] = sampleBuffer[2][i] + sampleBuffer[4][i];
    }
}
```

Gore prikazana fotografija, na sebi sadrži isečak “.cpp” koda modela nula. Tu se može uočiti pristup indeksiranjem, što ne predstavlja maksimalnu optimizaciju, iako je na optimizovanost koda vođeno računa od samog modela nula.




```
        double *sbC = sampleBuffer[1];
        double *sbR = sampleBuffer[2];
        double *sbLs = sampleBuffer[3];
        double *sbRs = sampleBuffer[4];

        int i;
        for (i = 0; i < BLOCK_SIZE; i++)
        {
            *sbLs = (*sbL) * processing_input_gain;
            *sbRs = (*sbR) * processing_input_gain;
            *sbC = (*sbLs) + (*sbRs);
            sbLs++;
            sbL++;
            sbRs++;
            sbR++;
            sbC++;
        }
        sbL = sampleBuffer[0];
        sbC = sampleBuffer[1];
        sbR = sampleBuffer[2];
        sbLs = sampleBuffer[3];
        sbRs = sampleBuffer[4];

        gst_audio_dynamic_transform_compressor_double(&processing_audio_compressor, sbLs, BLOCK_SIZE);
        gst_audio_dynamic_transform_compressor_double(&processing_audio_compressor, sbRs, BLOCK_SIZE);

        for (i = 0; i < BLOCK_SIZE; i++)
        {
            *sbC = (*sbC) * processing_headroom_gain;
            sbC++;
        }
        sbC = sampleBuffer[1];

        for (i = 0; i < BLOCK_SIZE; i++)
        {
            *sbL = (*sbC) * gain6db_scaled;
            *sbR = (*sbC) * gain6db_scaled;
            sbL++;
            sbR++;
            sbC++;
        }
        sbL = sampleBuffer[0];
        sbC = sampleBuffer[1];
        sbR = sampleBuffer[2];

        for (i = 0; i < BLOCK_SIZE; i++)
        {
            *sbLs = (*sbLs) * gain2db_scaled;
            *sbRs = (*sbRs) * gain2db_scaled;
            sbLs++;
```

Iznad prikazana slika dela “main.cpp” datoteke pokazuje deo optimizacije korišćenjem pokazivača umesto pristupa indeksiranjem. Takođe, jedna od bitnih optimizacija implementirana u ovom modelu jeste podela “processing()” funkcije da obrađuje odvojeno slučaj moda “2_0_0” od ostalih, radi korišćenja manje instrukcija pri samoj obradi.

```
compressor.cpp    main.cpp
model2                                  (Global Scope)
        DSPfract *sbL = sampleBuffer[0];
        DSPfract *sbC = sampleBuffer[1];
        DSPfract *sbR = sampleBuffer[2];
        DSPfract *sbLs = sampleBuffer[3];
        DSPfract *sbRs = sampleBuffer[4];
        DSPaccum tmp;

        DSPint i;
        for (i = 0; i < BLOCK_SIZE; i++)
        {
            *sbLs = (*sbL) * processing_input_gain;
            *sbRs = (*sbR) * processing_input_gain;
            tmp = *sbLs + *sbRs;
            *sbC = tmp;
            sbLs++;
            sbL++;
            sbRs++;
            sbR++;
            sbC++;
        }
        sbL = sampleBuffer[0];
        sbC = sampleBuffer[1];
        sbR = sampleBuffer[2];
        sbLs = sampleBuffer[3];
        sbRs = sampleBuffer[4];

        gst_audio_dynamic_transform_compressor_double(&processing_audio_compressor, sbLs, BLOCK_SIZE);

        gst_audio_dynamic_transform_compressor_double(&processing_audio_compressor, sbRs, BLOCK_SIZE);

        for (i = 0; i < BLOCK_SIZE; i++)
        {
            *sbC = (*sbC) * processing_headroom_gain;
            sbC++;
        }
        sbC = sampleBuffer[1];

        for (i = 0; i < BLOCK_SIZE; i++)
        {
            *sbL = (*sbC) * gain6db_scaled;
            *sbR = (*sbC) * gain6db_scaled;
            sbL++;
            sbR++;
            sbC++;
        }
        sbL = sampleBuffer[0];
```

Model dva, čiji deo je prikazan na slici iznad, unosi aritmetiku pokretnog zareza pomoću dobijenih datoteka. On predstavlja pripremu koda za mogućnosti ciljanog "DSP"-ja. Za razliku od prethodnih modela, u ovom je značajno trebalo izmeniti i samu "compressor.cpp" datoteku, čiji odlomak se može videti ispod:

```
void gst_audio_dynamic_transform_compressor_double(AudioCompressor_t * compressor, DSPfract * data, DSPfract num_samples)
{
    DSPaccum val, threshold = compressor->threshold;
    int i;

    /* Nothing to do for us if ratio == 1.0. */
    if (compressor->ratio == FRACT_NUM(1.0))
        return;

    for (i = 0; i < num_samples; i++) {
        val = data[i];
        DSPaccum negAccVal = val - threshold;
        DSPfract negVal = negAccVal;
        DSPaccum posAccVal = val + threshold;
        DSPfract posVal = posAccVal;

        if (val > threshold) {
            val = negVal * compressor->ratio;
            val = val + threshold;
        }
        else if (val < -threshold) {
            val = posVal * compressor->ratio;
            val = val - threshold;
        }
        data[i] = (DSPfract)val;
    }

}
```

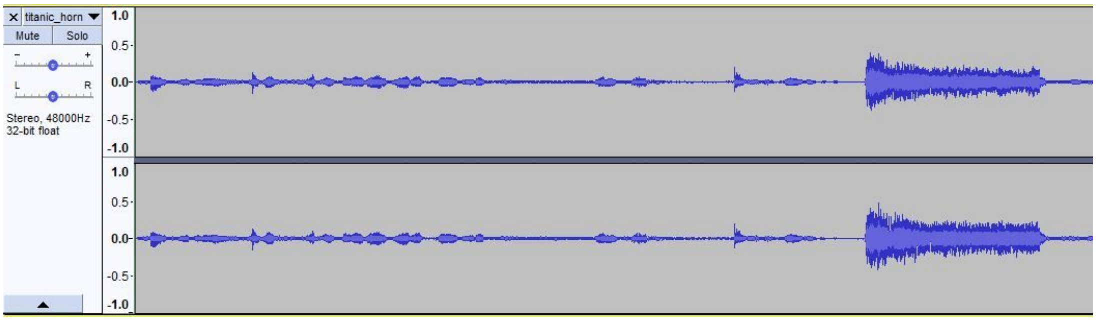
Potrebno je bilo i voditi računa o samoj veličini promenljivih i operacijama nad njima.

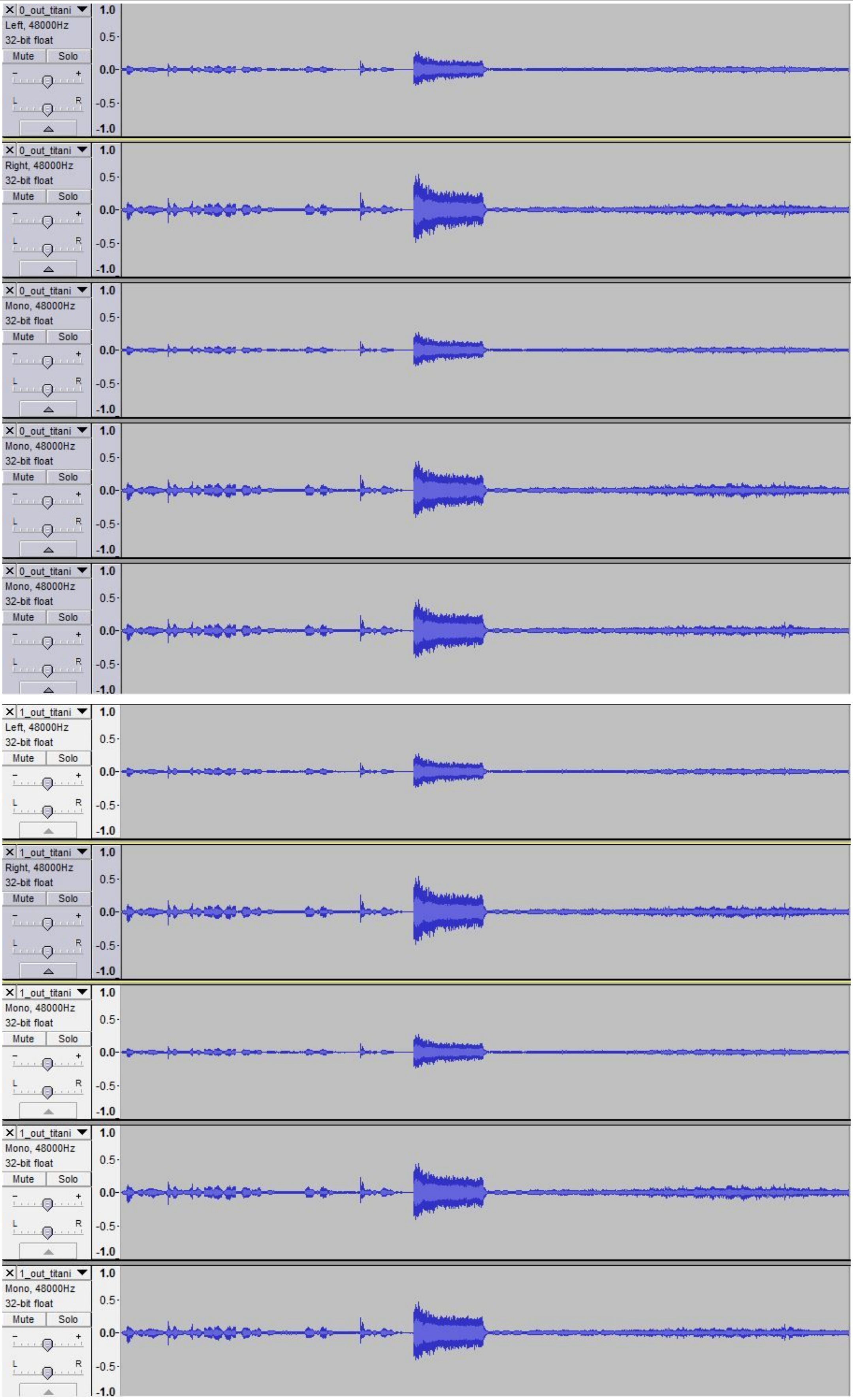
```c
__memY DSPfract sampleBuffer[MAX_NUM_CHANNEL][BLOCK_SIZE];

__memX DSPint output_channels[] = { 2, 2, 5 };

DSPfract gain6db_scaled = FRACT_NUM(0.501187);
DSPfract gain2db_scaled = FRACT_NUM(0.794328);
DSPfract processing_input_gain = FRACT_NUM(0.501187);
DSPfract processing_headroom_gain = FRACT_NUM(0.707946);

DSPint buffer_choice[3][5] = { { 0, 2, 0, 0, 0 }, { 3, 4, 0, 0, 0 }, { 0, 1, 2, 3, 4 } };

// Default processing compressor parameters for this project
DSPfract processing_compressor_threshold = FRACT_NUM(0.1);
DSPfract processing_compressor_ratio = FRACT_NUM(0.5);

__memX AudioCompressor_t processing_audio_compressor;

int processing_output_mode = MODE3_2_0;

void processing()
{
    // mode020
    if (processing_output_mode == MODE0_2_0)
    {
        __memY DSPfract *sbL = sampleBuffer[0];
        __memY DSPfract *sbC = sampleBuffer[1];
        __memY DSPfract *sbR = sampleBuffer[2];

        DSPint i;
        for (i = 0; i < BLOCK_SIZE; i++)
        {
            *sbC = (*sbL) * processing_input_gain + (*sbR) * processing_input_gain;
            sbR++;
            sbL++;
            sbC++;
        }
        sbL -= BLOCK_SIZE;
        sbC -= BLOCK_SIZE;
        sbR -= BLOCK_SIZE;

        for (i = 0; i < BLOCK_SIZE; i++)
        {
            *sbC = (*sbC) * processing_headroom_gain;
            sbC++;
        }
        sbC -= BLOCK_SIZE;

        for (i = 0; i < BLOCK_SIZE; i++)
        {
            *sbL = (*sbC) * gain6db_scaled;
```

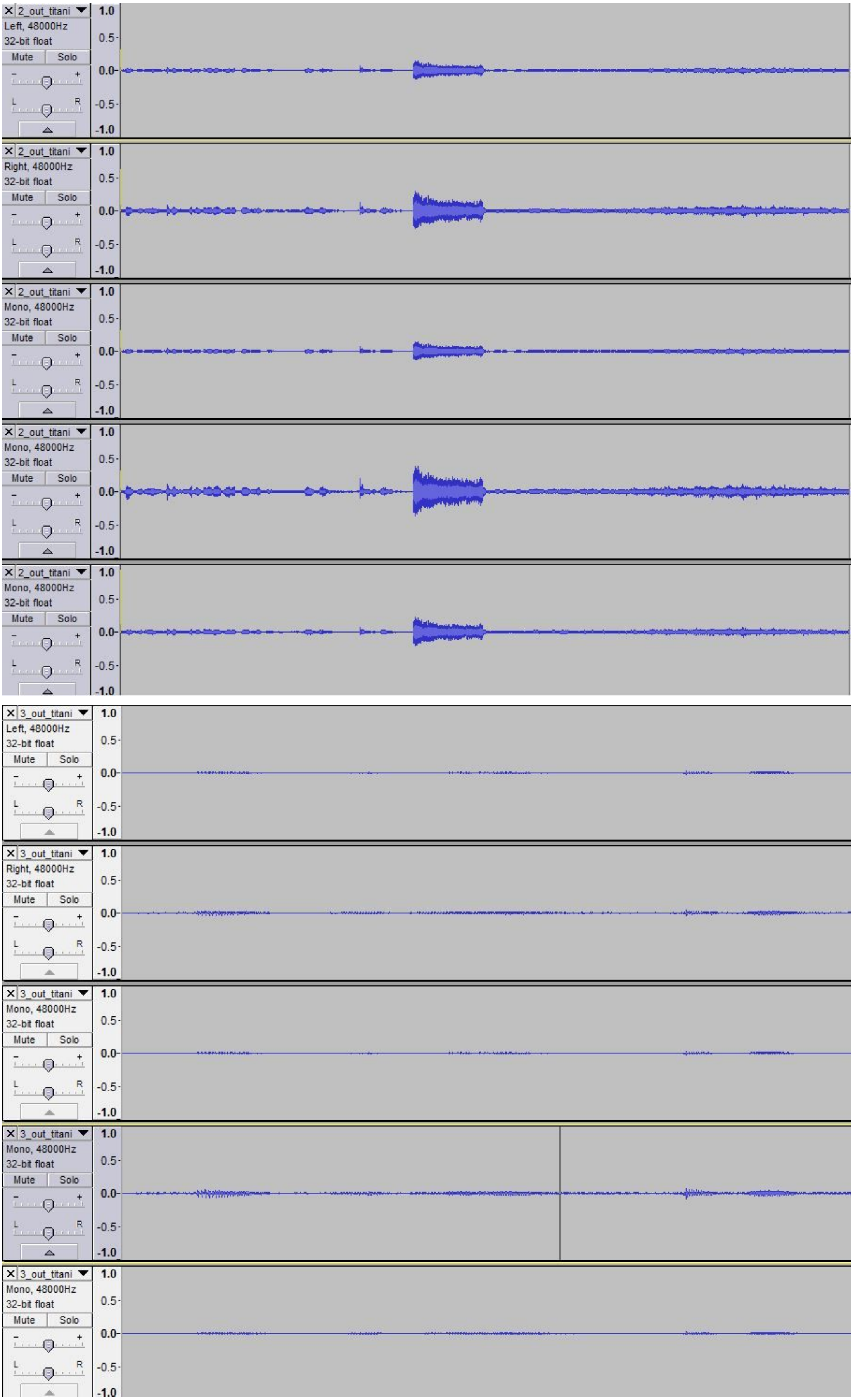
Iznad prikazan, odsečak modela tri, pokazuje deo optimizacije podelom na različite memorijske zone.

3. Ispitivanje i verifikacija

Za ispitivanje zadatka, prvo je korišćen alat “audacity”, koji prikazuje izgled “.wav” datoteke i omogućava njeno reprodukovanje. Ispod su, redom, prikazani osnovni signal i izlazi iz modela nula, jedan, dva i tri, testirani na ulaznoj datoteci “titanic_horn.wav” koja ima dva ulazna kanala, a za izlaz je uzet podrazumevani model “3_2_0” sa pet kanala, “enable” podešen na jedan, “gain”-ovi na po minus dva, a “ratio” kompresora na 0.5 i “threshold” na 0.1:




0_out_titani
Left, 48000Hz
32-bit float
Mute
Solo
0_out_titani
Right, 48000Hz
32-bit float
Mute
Solo
0_out_titani
Mono, 48000Hz
32-bit float
Mute
Solo
0_out_titani
Mono, 48000Hz
32-bit float
Mute
Solo
0_out_titani
Mono, 48000Hz
32-bit float
Mute
Solo
1_out_titani
Left, 48000Hz
32-bit float
Mute
Solo
1_out_titani
Right, 48000Hz
32-bit float
Mute
Solo
1_out_titani
Mono, 48000Hz
32-bit float
Mute
Solo
1_out_titani
Mono, 48000Hz
32-bit float
Mute
Solo
1_out_titani
Mono, 48000Hz
32-bit float
Mute
Solo
1.0
0.5
0.0
-0.5
-1.0



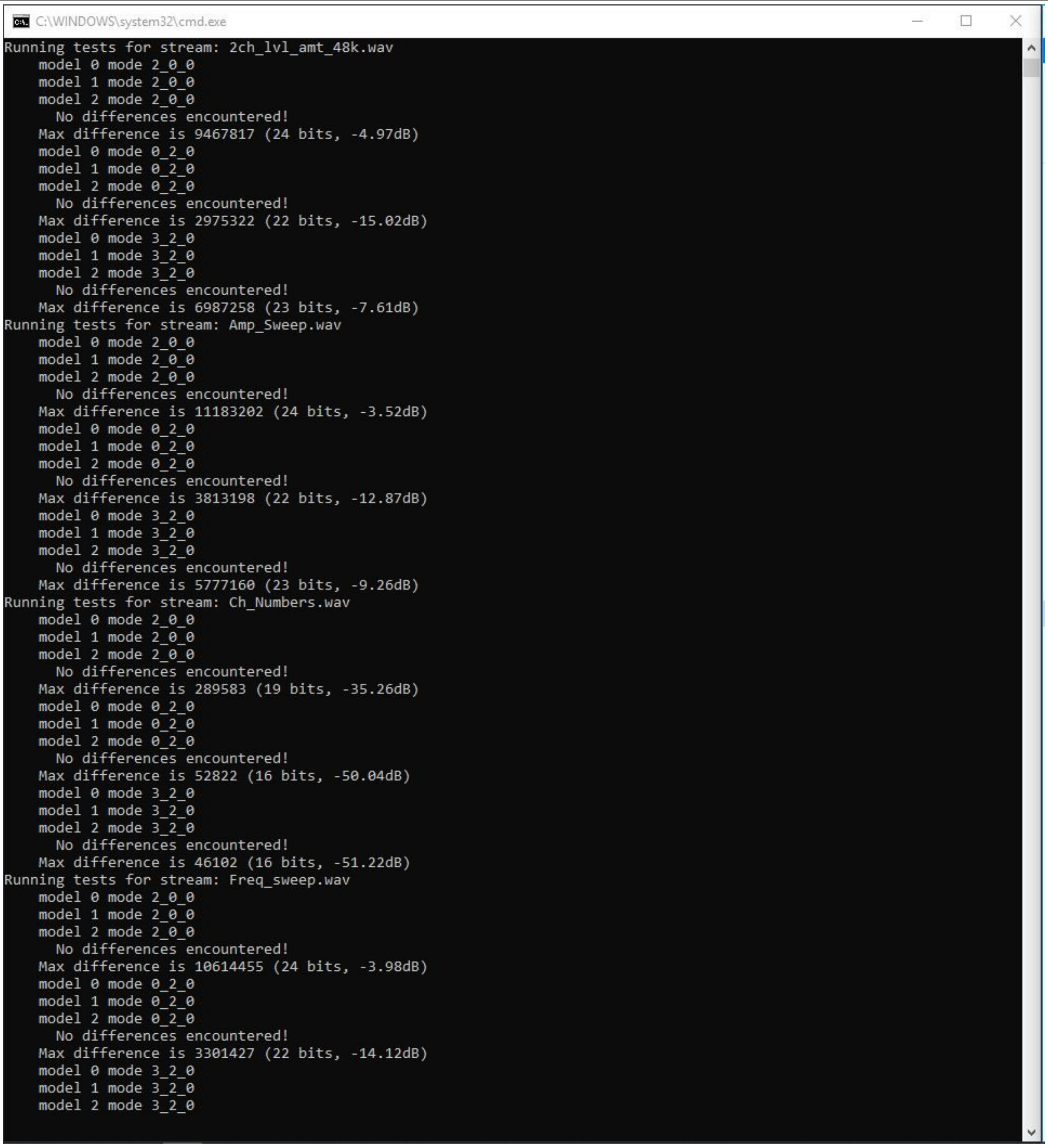
Kao što se iznad može primetiti, druga dva modela ne daju potpuno željeni rezultat, međutim samo razumevanje govora je moguće bez ikakvih poteškoća uz malu količinu pucketanja u pozadini. Treći model je prošao kroz, grubo, četrdeset i pet miliona ciklusa dok nije nasilno zaustavljen u izvršavanju.

Potpuno testiranje izvršeno je pomoću “batch” skripte, čiji kod je dat u prilog na slici dole:

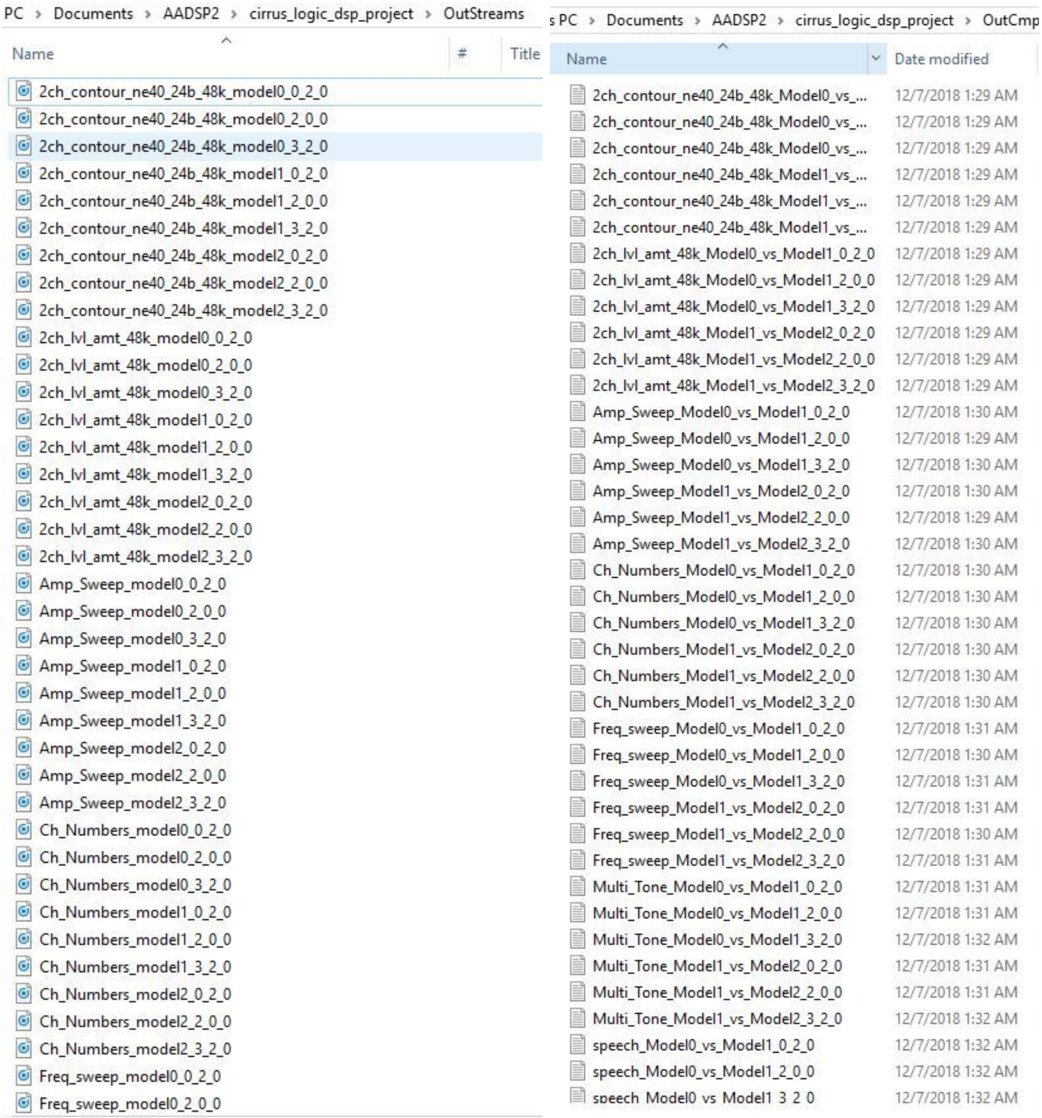
Test.bat

```
SET ADDITIONAL_ARGS_2_0_0=" 1 -4 -4 0
SET ADDITIONAL_ARGS_0_2_0=" 1 -4 -4 1 "
SET ADDITIONAL_ARGS_3_2_0=" 1 -4 -4 2 "

SET SIMULATOR=C:\CirrusDSP\bin\cmdline_simulator.exe -silent
SET COMPARE="PCMCompare.exe"


: Delete log files first.
del /Q OutCmp\*
del /Q OutStreams\*
: For each Test stream
cd TestStreams
for %%g in ("*.*") do ( echo Running tests for stream: %%~ng%%~xg
    echo     model 0 mode 2_0_0
    "..\%CONFIGURATION%\model0.exe" "%%g" "..\OutStreams//%%~ng_model0_2_0_0.wav %ADDITIONAL_ARGS_2_0_0%"
    echo     model 1 mode 2_0_0
    "..\%CONFIGURATION%\model1.exe" "%%g" "..\OutStreams//%%~ng_model1_2_0_0.wav %ADDITIONAL_ARGS_2_0_0%"
    echo     model 2 mode 2_0_0
    "..\%CONFIGURATION%\model2.exe" "%%g" "..\OutStreams//%%~ng_model2_2_0_0.wav %ADDITIONAL_ARGS_2_0_0%"

    cd ..\

    %COMPARE% OutStreams//%%~ng_model0_2_0_0.wav OutStreams//%%~ng_model1_2_0_0.wav 2> OutCmp//%%~ng_Model0_vs_Model1_2_0_0.txt
    %COMPARE% OutStreams//%%~ng_model1_2_0_0.wav OutStreams//%%~ng_model2_2_0_0.wav 2> OutCmp//%%~ng_Model1_vs_Model2_2_0_0.txt
    cd TestStreams

    echo     model 0 mode 0_2_0
    "..\%CONFIGURATION%\model0.exe" "%%g" "..\OutStreams//%%~ng_model0_0_2_0.wav %ADDITIONAL_ARGS_0_2_0%"
    echo     model 1 mode 0_2_0
    "..\%CONFIGURATION%\model1.exe" "%%g" "..\OutStreams//%%~ng_model1_0_2_0.wav %ADDITIONAL_ARGS_0_2_0%"
    echo     model 2 mode 0_2_0
    "..\%CONFIGURATION%\model2.exe" "%%g" "..\OutStreams//%%~ng_model2_0_2_0.wav %ADDITIONAL_ARGS_0_2_0%"

    cd ..\

    %COMPARE% OutStreams//%%~ng_model0_0_2_0.wav OutStreams//%%~ng_model1_0_2_0.wav 2> OutCmp//%%~ng_Model0_vs_Model1_0_2_0.txt
    %COMPARE% OutStreams//%%~ng_model1_0_2_0.wav OutStreams//%%~ng_model2_0_2_0.wav 2> OutCmp//%%~ng_Model1_vs_Model2_0_2_0.txt
    cd TestStreams

    echo     model 0 mode 3_2_0
    "..\%CONFIGURATION%\model0.exe" "%%g" "..\OutStreams//%%~ng_model0_3_2_0.wav %ADDITIONAL_ARGS_3_2_0%"
    echo     model 1 mode 3_2_0
    "..\%CONFIGURATION%\model1.exe" "%%g" "..\OutStreams//%%~ng_model1_3_2_0.wav %ADDITIONAL_ARGS_3_2_0%"
    echo     model 2 mode 3_2_0
    "..\%CONFIGURATION%\model2.exe" "%%g" "..\OutStreams//%%~ng_model2_3_2_0.wav %ADDITIONAL_ARGS_3_2_0%"

    cd ..\

    %COMPARE% OutStreams//%%~ng_model0_3_2_0.wav OutStreams//%%~ng_model1_3_2_0.wav 2> OutCmp//%%~ng_Model0_vs_Model1_3_2_0.txt
```

Skripta prolazi kroz prva tri modela, kroz sva tri moda, izvršava ih, i na kraju poredi, a rezultat poređenja ispisuje u “OutCmp” direktorijum. Primer pokrenute skripte i rezultata je dat na slikama ispod:



```
C:\WINDOWS\system32\cmd.exe
Running tests for stream: 2ch_lvl_amt_48k.wav
    model 0 mode 2_0_0
    model 1 mode 2_0_0
    model 2 mode 2_0_0
      No differences encountered!
    Max difference is 9467817 (24 bits, -4.97dB)
    model 0 mode 0_2_0
    model 1 mode 0_2_0
    model 2 mode 0_2_0
      No differences encountered!
    Max difference is 2975322 (22 bits, -15.02dB)
    model 0 mode 3_2_0
    model 1 mode 3_2_0
    model 2 mode 3_2_0
      No differences encountered!
    Max difference is 6987258 (23 bits, -7.61dB)
Running tests for stream: Amp_Sweep.wav
    model 0 mode 2_0_0
    model 1 mode 2_0_0
    model 2 mode 2_0_0
      No differences encountered!
    Max difference is 11183202 (24 bits, -3.52dB)
    model 0 mode 0_2_0
    model 1 mode 0_2_0
    model 2 mode 0_2_0
      No differences encountered!
    Max difference is 3813198 (22 bits, -12.87dB)
    model 0 mode 3_2_0
    model 1 mode 3_2_0
    model 2 mode 3_2_0
      No differences encountered!
    Max difference is 5777160 (23 bits, -9.26dB)
Running tests for stream: Ch_Numbers.wav
    model 0 mode 2_0_0
    model 1 mode 2_0_0
    model 2 mode 2_0_0
      No differences encountered!
    Max difference is 289583 (19 bits, -35.26dB)
    model 0 mode 0_2_0
    model 1 mode 0_2_0
    model 2 mode 0_2_0
      No differences encountered!
    Max difference is 52822 (16 bits, -50.04dB)
    model 0 mode 3_2_0
    model 1 mode 3_2_0
    model 2 mode 3_2_0
      No differences encountered!
    Max difference is 46102 (16 bits, -51.22dB)
Running tests for stream: Freq_sweep.wav
    model 0 mode 2_0_0
    model 1 mode 2_0_0
    model 2 mode 2_0_0
      No differences encountered!
    Max difference is 10614455 (24 bits, -3.98dB)
    model 0 mode 0_2_0
    model 1 mode 0_2_0
    model 2 mode 0_2_0
      No differences encountered!
    Max difference is 3301427 (22 bits, -14.12dB)
    model 0 mode 3_2_0
    model 1 mode 3_2_0
    model 2 mode 3_2_0
```


PC > Documents > AADSP2 > cirrus_logic_dsp_project > OutStreams
Name
#
Title
2ch_contour_ne40_24b_48k_model0_0_2_0
2ch_contour_ne40_24b_48k_model0_2_0_0
2ch_contour_ne40_24b_48k_model0_3_2_0
2ch_contour_ne40_24b_48k_model1_0_2_0
2ch_contour_ne40_24b_48k_model1_2_0_0
2ch_contour_ne40_24b_48k_model1_3_2_0
2ch_contour_ne40_24b_48k_model2_0_2_0
2ch_contour_ne40_24b_48k_model2_2_0_0
2ch_contour_ne40_24b_48k_model2_3_2_0
2ch_lvl_amt_48k_model0_0_2_0
2ch_lvl_amt_48k_model0_2_0_0
2ch_lvl_amt_48k_model0_3_2_0
2ch_lvl_amt_48k_model1_0_2_0
2ch_lvl_amt_48k_model1_2_0_0
2ch_lvl_amt_48k_model1_3_2_0
2ch_lvl_amt_48k_model2_0_2_0
2ch_lvl_amt_48k_model2_2_0_0
2ch_lvl_amt_48k_model2_3_2_0
Amp_Sweep_model0_0_2_0
Amp_Sweep_model0_2_0_0
Amp_Sweep_model0_3_2_0
Amp_Sweep_model1_0_2_0
Amp_Sweep_model1_2_0_0
Amp_Sweep_model1_3_2_0
Amp_Sweep_model2_0_2_0
Amp_Sweep_model2_2_0_0
Amp_Sweep_model2_3_2_0
Ch_Numbers_model0_0_2_0
Ch_Numbers_model0_2_0_0
Ch_Numbers_model0_3_2_0
Ch_Numbers_model1_0_2_0
Ch_Numbers_model1_2_0_0
Ch_Numbers_model1_3_2_0
Ch_Numbers_model2_0_2_0
Ch_Numbers_model2_2_0_0
Ch_Numbers_model2_3_2_0
Freq_sweep_model0_0_2_0
Freq_sweep_model0_2_0_0
s PC > Documents > AADSP2 > cirrus_logic_dsp_project > OutCmp
Name
Date modified
2ch_contour_ne40_24b_48k_Model0_vs_... 12/7/2018 1:29 AM
2ch_contour_ne40_24b_48k_Model0_vs_... 12/7/2018 1:29 AM
2ch_contour_ne40_24b_48k_Model0_vs_... 12/7/2018 1:29 AM
2ch_contour_ne40_24b_48k_Model1_vs_... 12/7/2018 1:29 AM
2ch_contour_ne40_24b_48k_Model1_vs_... 12/7/2018 1:29 AM
2ch_contour_ne40_24b_48k_Model1_vs_... 12/7/2018 1:29 AM
2ch_lvl_amt_48k_Model0_vs_Model1_0_2_0 12/7/2018 1:29 AM
2ch_lvl_amt_48k_Model0_vs_Model1_2_0_0 12/7/2018 1:29 AM
2ch_lvl_amt_48k_Model0_vs_Model1_3_2_0 12/7/2018 1:29 AM
2ch_lvl_amt_48k_Model1_vs_Model2_0_2_0 12/7/2018 1:29 AM
2ch_lvl_amt_48k_Model1_vs_Model2_2_0_0 12/7/2018 1:29 AM
2ch_lvl_amt_48k_Model1_vs_Model2_3_2_0 12/7/2018 1:29 AM
Amp_Sweep_Model0_vs_Model1_0_2_0 12/7/2018 1:30 AM
Amp_Sweep_Model0_vs_Model1_2_0_0 12/7/2018 1:29 AM
Amp_Sweep_Model0_vs_Model1_3_2_0 12/7/2018 1:30 AM
Amp_Sweep_Model1_vs_Model2_0_2_0 12/7/2018 1:30 AM
Amp_Sweep_Model1_vs_Model2_2_0_0 12/7/2018 1:29 AM
Amp_Sweep_Model1_vs_Model2_3_2_0 12/7/2018 1:30 AM
Ch_Numbers_Model0_vs_Model1_0_2_0 12/7/2018 1:30 AM
Ch_Numbers_Model0_vs_Model1_2_0_0 12/7/2018 1:30 AM
Ch_Numbers_Model0_vs_Model1_3_2_0 12/7/2018 1:30 AM
Ch_Numbers_Model1_vs_Model2_0_2_0 12/7/2018 1:30 AM
Ch_Numbers_Model1_vs_Model2_2_0_0 12/7/2018 1:30 AM
Ch_Numbers_Model1_vs_Model2_3_2_0 12/7/2018 1:30 AM
Freq_sweep_Model0_vs_Model1_0_2_0 12/7/2018 1:31 AM
Freq_sweep_Model0_vs_Model1_2_0_0 12/7/2018 1:30 AM
Freq_sweep_Model0_vs_Model1_3_2_0 12/7/2018 1:31 AM
Freq_sweep_Model1_vs_Model2_0_2_0 12/7/2018 1:31 AM
Freq_sweep_Model1_vs_Model2_2_0_0 12/7/2018 1:30 AM
Freq_sweep_Model1_vs_Model2_3_2_0 12/7/2018 1:31 AM
Multi_Tone_Model0_vs_Model1_0_2_0 12/7/2018 1:31 AM
Multi_Tone_Model0_vs_Model1_2_0_0 12/7/2018 1:31 AM
Multi_Tone_Model0_vs_Model1_3_2_0 12/7/2018 1:32 AM
Multi_Tone_Model1_vs_Model2_0_2_0 12/7/2018 1:31 AM
Multi_Tone_Model1_vs_Model2_2_0_0 12/7/2018 1:31 AM
Multi_Tone_Model1_vs_Model2_3_2_0 12/7/2018 1:32 AM
speech_Model0_vs_Model1_0_2_0 12/7/2018 1:32 AM
speech_Model0_vs_Model1_2_0_0 12/7/2018 1:32 AM
speech_Model0_vs_Model1_3_2_0 12/7/2018 1:32 AM