APPS019 Codensity T4xx Bitrate Reconfiguration Application Note

V3.0

09/29/2020

09/29/2020

09/29/2020

Table of Contents

1 Table of Abbreviations

Abbreviation:	Full Form:

2 References

Application Note: APPS008-netint_apps_note_reconfig.

3 Background

This document describes how to implement bitrate reconfiguration feature and its usage in operation/integrating with the NETINT T4xx transcoder.

3.1 Intended Audience

This document is intended to help engineers/technicians/developers integrating NETINT transcoders to reconfigure bitrate dynamically.

3.2 Compatibility

Software Compatibility

This guide is intended to be used with NETINT Codensity T4xx Video Transcoder software Release 2.1.3 and above.

Hardware Compatibility

Release 2.1.1 and above supports NETINT Codensity T4xx Video Transcoder hardware.

Operation system

All OS supporting T4xx Video Transcoder.

4 Bitrate reconfiguration

4.1 Implementation in libavcodec

A new AVFrameSideDataType named AV_FRAME_DATA_NETINT_BITRATE is introduced. Its payload is int32_t type data which represents the bitrate to set.

In function xcoder_send_frame in libavcodec/nienc.c, it will call av_frame_get_side_data(AV_FRAME_DATA_NETINT_BITRATE_SEI) to attempt to retrieve side data from AVFrame. If this side data exists, it will set reconfig structure ni_encoder_change_params_t enable_option field with the NI_SET_CHANGE_PARAM_RC_TARGET_RATE flag and set bitRate field to the side data's int32_t value. After that it follows reconfiguration call flow to send the reconfig structure down to firmware as metadata. Please refer to APPS008-netint_apps_note_reconfig for more details.

4.2 FFmpeg API usage

Since the new AVFrameSideDataType is NETINT proprietary, it is not supported by ffmpeg officially. There would be no command line options which can be used to exercise this feature.

To re-configure the bitrate dynamically during encoding, one can call APIs provided by ffmpeg libraries. Here is the sample code:

```
AVFrameSideData *sd;

AVFrame *frame;

…

sd = av_frame_new_side_data(frame, AV_FRAME_DATA_NETINT_BITRATE, sizeof(int32_t));

if (sd) {

    *((int32_t *)sd->data) = bitrate;

}
```

Once frame data is sent to fw, manually free the side data buffer by calling the following API function if needed:

```
av_frame_remove_side_data(frame, AV_FRAME_DATA_NETINT_BITRATE);
```

Note that bitrate reconfiguration works with rate control enabled. This means that xcoder-params needs to have "RcEnable=1" set.

The total implementation of reconfig bitrate can be separated into the following three steps:

Step 1:

It only needs to call the **av_frame_new_side_data** function, set the AVFrameSideDataType to **AV_FRAME_DATA_NETINT_BITRATE**, and then pass the value of bitrate to the sidedata.

```
AVFrameSideData *sd = av_frame_new_side_data(frame,
                              AV_FRAME_DATA_NETINT_BITRATE, sizeof(int32_t));
if(sd != NULL)
{
   *((int32_t*)sd->data) = bitrate_value;
}
```

The functon av_frame_new_side_data is a native function in ffmpeg. It is defined as follows:

```
/**
 * Add a new side data to a frame.
 *
 * @param frame a frame to which the side data should be added
 * @param type type of the added side data
 * @param size size of the side data
 *
 * @return newly added side data on success, NULL on error
 */
AVFrameSideData *av_frame_new_side_data(AVFrame *frame,
                              Enum AVFrameSideDataType type,
                              Int size);
```

Step 2:
After the above settings are completed, the reconfig bitrate will set as the aux_data. The implementation of this part is as follows:

```
side_data = av_frame_get_side_data(&ctx->buffered_fme, AV_FRAME_DATA_NETINT_BITRATE);
if (side_data && (side_data->size == sizeof(int32_t)))
{
   aux_data = ni_logan_frame_new_aux_data(&dec_frame, NI_FRAME_AUX_DATA_BITRATE,
                                                    sizeof(int32_t));
   if (aux_data)
   {
      memcpy(aux_data, side_data->data, side_data->size;);
   }
}
```

Step 3:

After completing the above steps,The reconfig bitrate has transferred to aux_data.In the libxcoder layer, get the aux_data and set the enable_option to NI_LOGAN_SET_CHANGE_PARAM_RC_TARGET_RATE and set the aux_data vault to bitrate param in enc_change_params structure. These code related to the reconfig bitrate function in libxcoder_logan/source/ni_av_codec_logan.c is as follows:

```
aux_data = ni_logan_frame_get_aux_data(p_dec_frame, NI_FRAME_AUX_DATA_BITRATE);
if (aux_data)
{
   int32_t bitrate = *((int32_t*)aux_data->data);
   if(!p_enc_ctx->enc_change_params)
   {
      p_enc_ctx->enc_change_params = calloc(1, sizeof(ni_logan_encoder_change_params_t));
   }
   p_enc_ctx->enc_change_params->enable_option |=
       NI_LOGAN_SET_CHANGE_PARAM_RC_TARGET_RATE;
   p_enc_ctx->enc_change_params->bitRate = bitrate;
}
```

After above process, the reconfig bitrate information will be passed to the encoder.

NOTE: Step2 and Step3 are implemented in the libavcodec and libxcoder layer, only the Step1 need to be implemented in the application layer.

4.3 Demo mode

A reconf demo mode **ReconfDemoMode=1** is used to demonstrate this feature. Another libxcoder parameter **ReconfFile** is used to specify a file containing a list of bitrates and frame numbers where the bitrate should be changed. The resulting FFmpeg command line options look like this:

sudo ./ffmpeg -i input.264 -c:v h265_ni_logan_enc -xcoder-params "RcEnable=1:bitrate=1000000:**ReconfDemoMode=1:ReconfFile=**reconf.txt" output.265

A sample reconfig file content is as follows:

300:5000000
600:2000000

On each line of the config file, the first column is the frame index and the second column the bitrate value. This sample reconfig file specifies a test that changes the bitrate to 5 Mbps at the 300th frame, then changes the bitrate to 2 Mbps at the 600th frame. Note that in order to change the bitrate, rate control needs to be enabled in the xcoder-params, and the initial bitrate set i.e., RcEnable=1:bitrate=1000000.

5 Revision History

Version	Change	Create Date	Author
1.0	Document Created	Sept 29,2020	Zheng Lv
2.0	Updated	Jan 29, 2021	Zhong Wang
3.0	Updated	Sep 07, 2022	Bing Li
3.1	Update	Sep 22,2022	Jack Wang

6 Legal Notice