APPS012 Codensity™ T408 Low Latency Mode Application Note

V2.0

2021-12-02

Table of Contents

1 Table of Abbreviations

Abbreviation:	Full Form:
DTS	Decoding time stamp
PTS	Presentation time stamp
NVMe	Non-volatile Memory express (Transport protocol)
CQ	Constant Quality
GOP	Group of Pictures
VUI	Video Usability Information
AI	Artificial Intelligence
API	Application Programming Interface
AUD	Access Unit Delimiters
AV	Audio Video
AV1	Alliance Open Media Video 1 Codec
CABAC	Context Adaptive Binary Arithmetic Coding
CBR	Constant Bit Rate
CPU	Central Processing Unit
CRF	Constant Rate Factor
DSP	Digital Signal Processing
EOF	End Of File
FIXQB	Fixed Quantization Parameter
FF	Form Factor
FPS	Frames Per Second
GOP	Group of pictures
HDR	High Dynamic Range
HHHL	Half Height, Half Length
HHFL	Half Height, Full Length
HLG	Hybrid Log Gamma
HRD	Hypothetical Reference Decoder
IE	Inference Engine
Mbps	Mega Bit per second
MBps	Mega Byte per second
NLP	Natural Language Processing
NPU	Neural Processing Unit
NVMe	Non-Volatile Memory express
PCIe	Peripheral Component Interconnect express
PPS	Picture Parameter Set

QP	Quoted Printable
RGB	Red Green Blue
RGBA	Red Green Blue Alpha
RDO	Optimization
ROI	Region of interest
SDK	Software Development Kit
SEI	Supplemental Enhancement Information

2 References

Technical Notes:

Specifications:

3 Background

For some video applications such as cloud game and conferencing, latency is critical. This document explains the encode and decode latency modes for the T408.

3.1 Intended Audience

This document is intended to help engineers/technicians/developers operating NETINT transcoders troubleshoot common problems.

3.2 Compatibility

Software Compatibility

This guide is intended to be used with NETINT Codensity T408 Video Transcoder software Release 2.6.

Hardware Compatibility

NETINT Codensity T408 Video Transcoder hardware.

Operation system

Ubuntu 18.04, kernel, FFmpeg version, libxcoder version and etc.

4 Encoder Low latency mode

Video encode latency is defined as the delay between a video frame input to the encoder and the SAME frame output from the encoder.

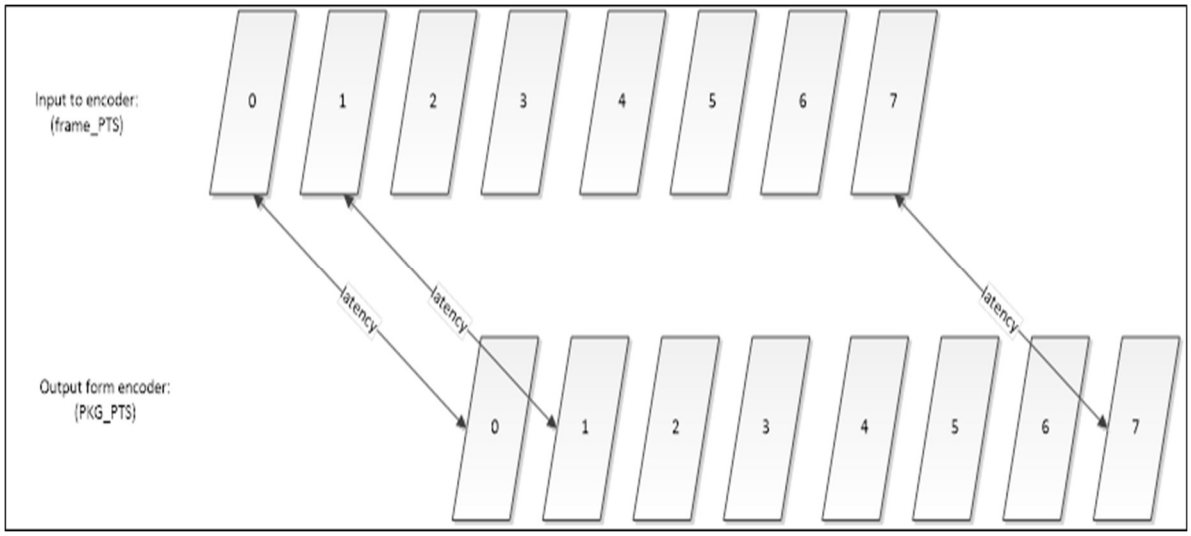

Figure 1: Encode Latency Diagram

In Figure 1, the order of display frames is increasing sequentially. When there is B frame, the DTS and PTS may be in different order.

4.1 GOP Requirements to Minimize Encoder Latency

As discussed, when B frames are used, the frames may need to be encoded out of order which will increase latency, therefore, to minimize latency we need to use a low delay GOP. A low delay GOP is one in which all the frames are encoded in sequence. On the T408, the low delay GOPs are gopPresetIdx=1, 2, 3, 6, 7, and 9 or custom GOP with in-sequence frames, i.e., pocOffset increments by 1 for each frame.

gopPresetIdx	Description	GOP Size	Ref Frames	Encode Order	Max Frames out of Order	Low GOP
1	All I	1	2	I0-I1-I2…	0	✓
2	All P	1	2	P0-P1-P2…	0	✓

3	All B	1	2	B0-B1-B2…	0	✓
4	BP	2	2	B1-P0-B3-P2…	1	
5	BBBP	4	2	B2-B1-B3-P0	3	
6	Consecutive PPPP	4	2	P0-P1-P2-P3…	0	✓
7	Consecutive BBBB	4	2	B0-B1-B2-B3…	0	✓
8	BBBBBBBB	8	2	B3-B2-B4-B1-B6-B5-B7-B0…	7	
9	All P	1	1	P0-P2-P2	0	✓

Table 1Table 1 lists the details of the T408 GOP Presets. You can see that the low delay GOPs have all frames in sequence. If a low delay GOP is not used then the encoder must buffer incoming frames in order to encode them out of sequence which then adds to the encoder latency.

gopPresetIdx	Description	GOP Size	Ref Frames	Encode Order	Max Frames out of Order	Low GOP
1	All I	1	2	I0-I1-I2…	0	✓
2	All P	1	2	P0-P1-P2…	0	✓
3	All B	1	2	B0-B1-B2…	0	✓
4	BP	2	2	B1-P0-B3-P2…	1	
5	BBBP	4	2	B2-B1-B3-P0	3	
6	Consecutive PPPP	4	2	P0-P1-P2-P3…	0	✓
7	Consecutive BBBB	4	2	B0-B1-B2-B3…	0	✓
8	BBBBBBBB	8	2	B3-B2-B4-B1-B6-B5-B7-B0…	7	
9	All P	1	1	P0-P2-P2	0	✓

Table 1: List of T408 Gop Presets

4.2 Encoder Low Latency Mode

A second aspect of encode latency is the buffering of input frames for better performance. The encoder allocates additional source frame buffers to the minimum specified in Table 1 so that the libxcoder can be downloading the next frame while the encoder is encoding a previous frame. While this increases performance, it adds to the latency. Therefore, we have defined a special encoder low latency mode such that libxcoder only sends a single frame to the encoder at a time and does not send the next frame until it receives an encoded frame. While this reduces performance somewhat it ensures the lowest encoding latency possible.

The actual latency is primarily determined by the length of time to encode each frame and the number of streams being encoded. This is determined mainly by the picture size and the load on the T408. For example, if the T408 can encode 8 1080p30 streams in real time, then it takes roughly 1s/30/8=4.2ms to encode a single frame. If you are encoding 8 streams, it will take 8 times or 33.3ms on average to encode a frame.

The encoder low latency mode is enabled with encode parameter lowDelay=1.

For example, here is a H.264 to H.265 transcode with low latency mode enabled in the encoder and a low delay GOP (gopPresetIdx=2):

```
sudo ffmpeg -vsync 0 -c:v h264_ni_dec -i input.264 -c:v
h265_ni_enc -xcoder-params
"gopPresetIdx=2:lowDelay=1:RcEnable=1:bitrate=4000000" output.265
```

If we set lowDelay=0, or omit it from the command above, the encoder will encode using normal delay mode which is the default. Note that low delay mode requires a low delay GOP and if enabled with a high delay GOP, an error message will be returned.

Normally, FFmpeg only polls the encoder once to read an encoded frame when a new YUV frame is sent to encoder. The frame is sent to encoder at the interval of input FPS. For example, for 50Hz input, the interval is 20ms. This way, a frame sent to encoder for encoding will only be available after 20ms, so the latency is longer than interval of frames.

The encoder low latency mode changes this behavior. If enabled, it polls the encoder at a 200us intervals until a frame is available to read. The latency will then be encode time plus 200us (worst case).

Note that the latency of first frame can be a little higher than other frames due to additional time for buffer allocation. Also note that the latency can vary from frame to frame slightly based on the complexity of the frame.

Figure 2 shows the measured latency for encoding a single 1080p H.265 bitstream with low delay mode enabled.

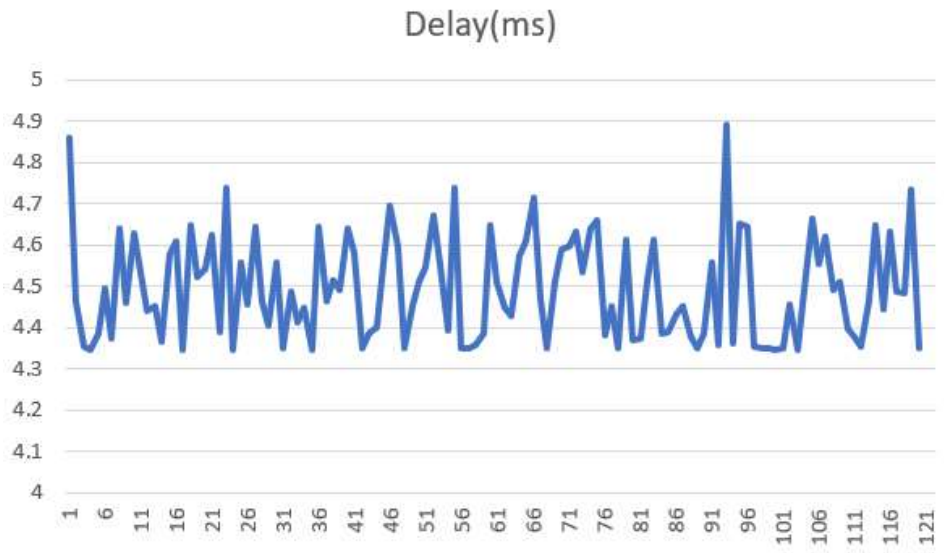

Figure 2: Measured latency of a single 1080p H.265 encode with low delay mode enabled.

5 Decoder Low latency mode

The decoder latency is defined similarly to the encoder and is the time between when the host writes an encode packet to the decoder to the time it receives back the same decoded YUV frame from the decoder. Just as with encoding, if the frames in the bitstream are out of sequence, additional delay is required to reorder them.

5.1 GOP Requirements to Minimize Decoding Delay

To minimize latency while decoding, we need to ensure first that we use a low delay GOP during encoding and we need to signal the use of the low delay GOP to the decoder.

Both H.264 and H.265 can contain information that the decoder can use to know how much reordering is required and how many input buffers it will need for decoding. For H.265, this information is stored in the SPS in sps_max_dec_pic_buffering_minus1 and sps_max_num_reorder_pics. For H.264, this information is stored in the VUI bitstream restriction flags, max_num_reorder_frames and max_dec_frame_buffering.

For H.265, this information is mandatory and so all H.265 bitstreams decode with the minimum delay possible. For H.264 unfortunately, the bitstream restriction flags are optional, and are often not included by many encoders.

H.264 also has an encoding mode that doesn't support out of sequence encoding, and so bitstreams encoded in this mode can always be decoded with minimum latency. This mode is indicated in the SPS as pic_order_cnt_type=2. So for H.264, the decoding delay is minimized if the encoder uses either pic_order_cnt_type=2 or includes VUI containing bitstream restriction flags. If neither is present, then the T408 decoder will normally add 5 additional frames of delay when decoding.

If you use the T408 encoder it will always include the bitstream restriction flags when encoding H.264, and it also supports encoding with pic_order_cnt_type=2 (by enabling useLowDelayPocType). Note that useLowDelayPocType is only supported for low delay GOPs while the bitstream restriction flags will minimize the delay for any GOP, but you still want to use a low delay GOP to achieve the lowest delay.

Figure 3 shows a partial SPS decode showing pic_order_cnt_type=2. Figure 4 shows a decode of the H.264 bitstream restriction flags for a low delay GOP (gopPresetIdx=2). Note that max_num_reorder_frames=0 meaning that the frames are in sequence and do not require reordering, and max_dec_frame_buffering=1 meaning that only 1 frame buffer is required for decoding.

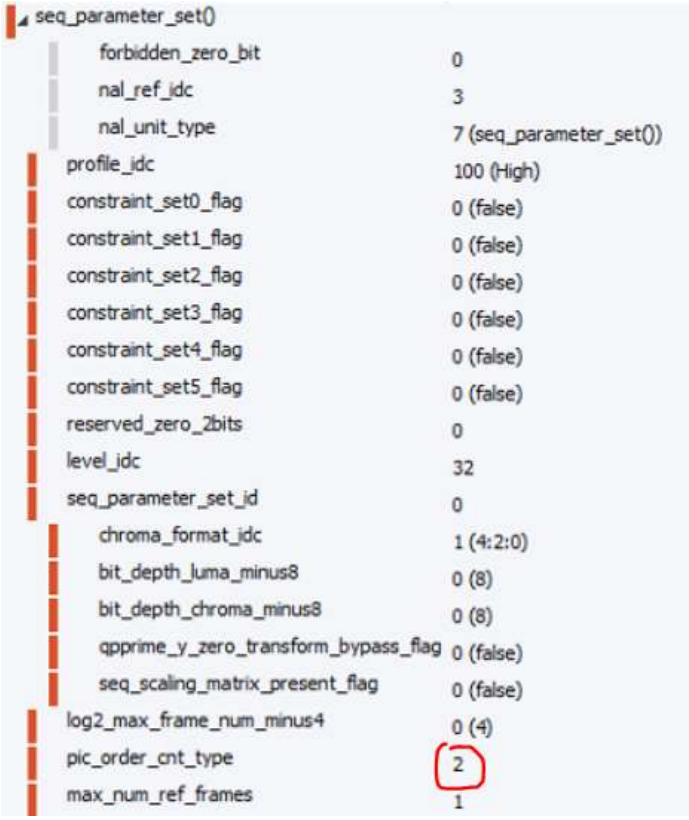

Figure 3: Partial H.264 SPS decode showing pic_order_cnt_type=2

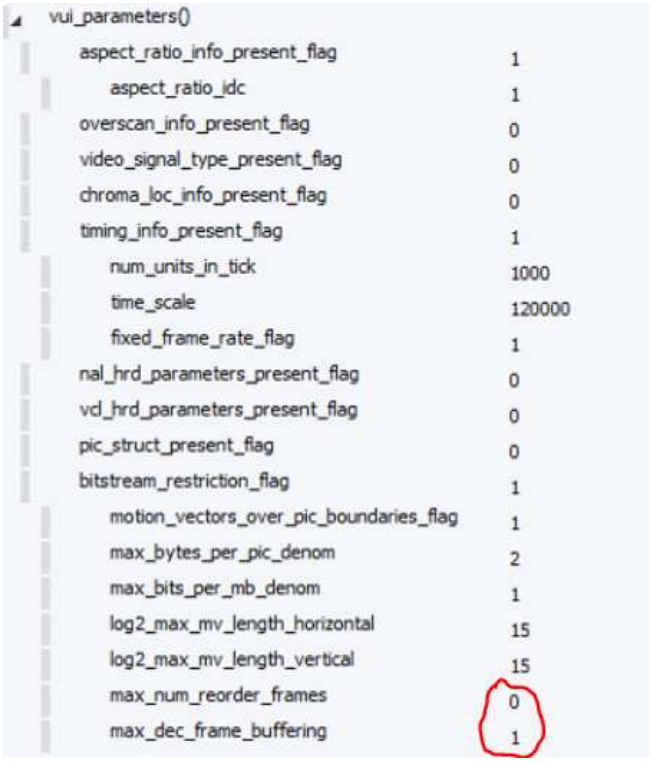

Figure 4: H.264 Bitstream Restriction Flags for low delay GOP gopPresetIdx=2

5.2 Decoder Low Latency Mode

Like the encoder, the decoder also has a low latency mode. This mode does 2 things, the first is to minimize buffering delay libxcoder sends only 1 frame at a time to the decoder and waits for a decoded frame to come back before sending the next frame. In order for this to work, the input stream needs to use a low delay GOP. To resolve the issue with H.264 bitstreams that do not indicate their reorder and buffering requirements (i.e., no bitstream restriction flags and no pic_order_cnt_type=2), libxcoder disables reordering on the decoder in order to force the decoder to return a decoded frame for each input frame.

Since it is possible under conditions of packet loss for the decoder to sometimes not be able to decode a frame, the decoder low latency feature has a configurable timeout. If libxcoder does not receive a decoded frame within the timeout, it will send another frame and hopefully continue decoding in low delay mode normally.

The decoder low latency mode is enabled by a decoder parameter -low_delay <timeout in ms> placed in the decoder section of the ffmpeg command..

For example, here is a command to decode an H.264 input to YUV with the low latency feature enabled with a 100ms timeout:

```
sudo ffmpeg -vsync 0 -c:v h264_ni_dec -low_delay 100 -i
lowDelay.264 output1280x720.yuv
```

When integrating directly with libxcoder, the decoder low latency mode is enabled by setting decoder session context *ni_session_context_t*'s attribute *decoder_low_delay* to the desired timeout in ms.

Note that the decoder low latency feature requires that the input bitstream use a low delay GOP. If the input is not low delay, i.e, it has out of sequence frames, the decoder will decode and return the frames out of sequence.

Figure 5 shows the measured latency of a single 1080p h.264 decode with the low delay feature enabled.

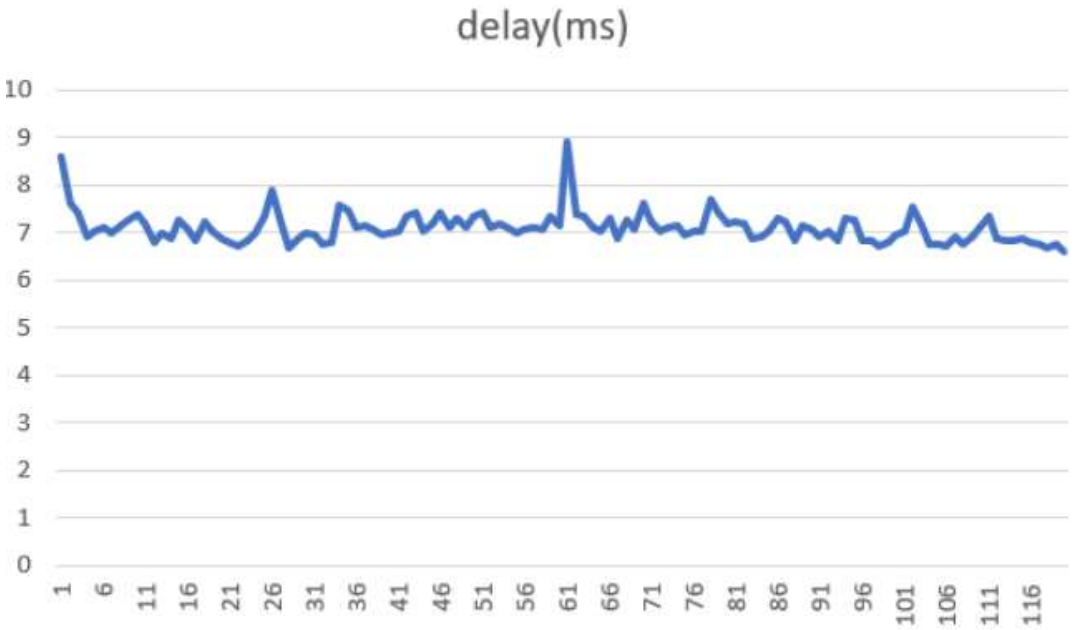

Figure 5: measured latency of a single 1080p H.264 decode with low delay feature enabled.

6 Measuring Encoder and Decoder Latency

Libxcoder can be compiled to generate latency information for the encoder and decoder in the FFmpeg log. This will enable plotting of the latency as shown in Figure 2 and Figure 5. Details on how to do this are contained in APP Note 32, T408 Latency Reporting Application Note.

7 Summary

To minimize encoder and decoder latency do the following:

1. Use low delay GOP with in sequence frames for encoding
2. For H.264, ensure that the encoder supports bitstream restriction flags or pic_order_type_cnt=2 mode. The Netint encoder supports both.
3. Enable low delay mode in the Netint encoder and decoder.

8 Revision History

Version	Change	Create Date	Author
1.0	Document Created	Feb 18.2020	Tom Han
1.1	Update	May 11, 2021	Zhong Wang, Neil Gunn
2.0	Updated per new app note template	December 2, 2021	Balwinder Sidhu

9 Legal Notice