



# Quadra™ Performance Test Report V5.3

## Contents

Contents .....	2
Environment Overview .....	3
Definitions .....	4
1. T1A – FFmpeg Throughput.....	5
2. T1A – Libxcodec Throughput.....	11
3. T1A – FFmpeg Latency .....	17
4. T1A – Decoder PPU Scaling .....	19
5. T1A – Streaming Ladder Generation.....	20
6. T1A – RGBA Encoding.....	21
7. T1A – Encoding EnableRdoQuant/rdoLevel/lookaheadDepth .....	23
8. T1A – Capped CRF .....	29
9. T1A – Inplace Overlay .....	35
10. 2x T2A – MultiThread P2P DMA on AMD GPU .....	37
11. T1A – AI .....	38
12. T1A – GStreamer XStack Throughput .....	42
13. T1A – GStreamer Ladder Generation.....	43
14. T1U – FFmpeg Throughput .....	44
15. T1U – Libxcodec Throughput .....	50
16. T1U – FFmpeg Latency.....	56
17. T1U – Decoder PPU Scaling.....	58
18. T1U – Streaming Ladder Generation .....	59
19. T1U – RGBA Encoding .....	60
20. T1U – Encoding EnableRdoQuant/rdoLevel/lookaheadDepth .....	62
21. T1U – Capped CRF.....	68
22. T1U – Inplace Overlay .....	74
Appendix A: GStreamer XStack Command .....	76
Appendix B: 7x7 Grid Layout.....	78
Appendix C: GStreamer Ladder Command .....	79

## Environment Overview

Revision: 5306sBr2

### Setup #1:

- Server: AMD Ryzen 5 5600 6-core Processor; CPU(s) 12; Motherboard MPG X570 GAMING EDGE WIFI (MS-7C37); Memory 16GiB System Memory 2x 8GiB DIMM DDR4 Synchronous Unbuffered (Unregistered) 2133 MHz (0.5 ns)
- DUT: 1x T1A or 1x T1U
- FFmpeg Version: 7.1
- Gstreamer Version: 1.22.2
- Tests:
  - FFmpeg Throughput
  - Libxcodec Throughput
  - FFmpeg Latency
  - Decoder PPU Scaling
  - Streaming Ladder Generation
  - Inplace Overlay
  - Encoding EnableRdoQuant/rdoLevel/lookaheadDepth (T1A only)
  - Gstreamer XStack Throughput (T1A only)
  - Gstreamer Ladder Generation (T1A only)

### Setup #2:

- Server: AMD EPYC 7763 64-Core Processor; CPU(s) 128; Motherboard OPYVT1; Memory 256GiB System Memory, 8x32GiB DIMM DDR4 Synchronous Registered (Buffered) 3200 MHz (0.3 ns)
- DUT: 2x T2A
- FFmpeg Version: 4.3.1
- Tests:
  - MultiThread P2P DMA on AMD GPU

### Setup #3:

- Server: AMD Ryzen 5 5600X 6-Core Processor; CPU(s) 12; Motherboard TUF GAMING X570-PLUS (WI-FI); Memory 16GiB System Memory, 2x8GiB DIMM DDR4 Synchronous Unbuffered (Unregistered) 2133 MHz (0.5 ns)
- DUT: 1x T1A
- FFmpeg Version: 4.3.1
- Tests:
  - AI

## Definitions

- CPU: Average per instance CPU usage.
  - $(\text{System-wide CPU usage} * \text{number of CPU}) / (\text{number of devices} * \text{number of instances per device})$
- FPS: Average of all FPS reported per process
- Jobs: Number of instances running concurrently
- HW Frame: Decoded YUV is kept on the device
- Bit: Input video's bit depth
- Resolution: Input video's resolution
- Load: Maximum load between FW Load and VPU Load during traffic

# 1. T1A – FFmpeg Throughput

## 1.1 Decoding

### 1.1.1 Description

Bitstream is read from an input file on ramdisk and then fed into hardware decoder through PCIe.

Bitstream is decoded by hardware decoder.

Decoded YUV frame is read out through PCIe and written into an output file.

### 1.1.2 Command Line

```
ffmpeg -nostdin -f concat -safe 0 -c:v <dec>_ni_quadra_dec -dec 0 -  
xcoder-params multicoreJointMode=<*> -i /media/ramdisk/input.list -f  
null /dev/null -
```

<dec> is the decoder codec. eg h264\_ni\_quadra\_dec, h265\_ni\_quadra\_dec, vp9\_ni\_quadra\_dec

<num\_jobs> is the number of instances running concurrently

<resolution> is resolution of input

<resolution> == 8k, multicoreJointMode = 1

<num\_jobs> == 1, multicoreJointMode = 1

## 1.2 Encoding

### 1.2.1 Description

YUV frame is read from an input file on ramdisk and then fed into hardware encoder through PCIe.

YUV frame is encoded by hardware encoder.

Encoded bitstream is read out through PCIe and written into an output file.

### 1.2.2 Command Line

```
ffmpeg -nostdin -f concat -safe 0 -i /media/ramdisk/input.list -c:v  
<enc>_ni_quadra_enc -enc 0 -xcoder-params  
intraPeriod=0:RcEnable=1:bitrate=<*>:multicoreJointMode=<*> -f null  
/dev/null -
```

<enc> is the encoder codec. eg h264\_ni\_quadra\_enc, h265\_ni\_quadra\_enc, av1\_ni\_quadra\_enc

<num\_jobs> is the number of instances running concurrently

<resolution> is resolution of input

<resolution> == 8k, multicoreJointMode = 1

<num\_jobs> == 1, multicoreJointMode = 1

<resolution> == 8k, bitrate = 50000000, framerate = 24

<resolution> == 4k, bitrate = 12000000, framerate = 30 (8bit) / 60 (10bit)

<resolution> == 1080p, bitrate = 3000000, framerate = 30 (8bit) / 60 (10bit)

<resolution> == 720p, bitrate = 1500000, framerate = 30 (8bit) / 60 (10bit)

## 1.3 Transcoding

### 1.3.1 Description

Bitstream is read from an input file on ramdisk and then fed into hardware decoder through PCIe. Bitstream is decoded by hardware decoder.

Decoded YUV frame is kept on device.

The YUV frame is encoded with hardware encoder.

The encoded bitstream is read out through PCIe and written into an output file.

### 1.3.2 Command line

```
ffmpeg -nostdin -f concat -safe 0 -c:v <dec>_ni_quadra_dec -dec 0 -  
xcoder-params out=hw:sempianar0=1:multicoreJointMode=<*> -i  
/media/ramdisk/input.list -c:v <enc>_ni_quadra_enc -enc 0 -xcoder-  
params intraPeriod=0:RcEnable=1:bitrate=<*>:multicoreJointMode=<*> -f  
null /dev/null -
```

<dec> is the decoder codec. ie h264\_ni\_quadra\_dec, h265\_ni\_quadra\_dec, vp9\_ni\_quadra\_dec

<enc> is the encoder codec. ie h264\_ni\_quadra\_enc, h265\_ni\_quadra\_enc, av1\_ni\_quadra\_enc

<num\_jobs> is the number of instances running concurrently

<resolution> is resolution of input

<resolution> == 8k, multicoreJointMode = 1

<num\_jobs> == 1, multicoreJointMode = 1

<resolution> == 8k, bitrate = 50000000, framerate = 24

<resolution> == 4k, bitrate = 12000000, framerate = 30 (8bit) / 60 (10bit)

<resolution> == 1080p, bitrate = 3000000, framerate = 30 (8bit) / 60 (10bit)

<resolution> == 720p, bitrate = 1500000, framerate = 30 (8bit) / 60 (10bit)

## 1.4 FFmpeg Throughput Performance Results

TYPE	RES	JOB	HW FRAME	Bit	Joint Mode	DEC_LOAD	ENC_LOAD	FPS	CPU
AVC to YUV	8k	1	0	8	1	95	0	115	8
HEVC to YUV	8k	1	0	8	1	96	0	115	9
VP9 to YUV	8k	1	0	8	1	24	0	38	3
YUV to AVC	8k	1	0	8	1	0	95	67	52
YUV to HEVC	8k	1	0	8	1	0	96	83	67
AVC to AVC	8k	1	1	8	1	68	99	54	3
AVC to HEVC	8k	1	1	8	1	80	98	71	4
HEVC to AVC	8k	1	1	8	1	61	99	53	4
HEVC to HEVC	8k	1	1	8	1	71	99	71	4
VP9 to AVC	8k	1	1	8	1	24	45	35	2
VP9 to HEVC	8k	1	1	8	1	24	43	37	2
AVC to YUV	8k	1	0	10	1	54	0	59	7
HEVC to YUV	8k	1	0	10	1	98	0	63	8
VP9 to YUV	8k	1	0	10	1	24	0	33	14
YUV to AVC	8k	1	0	10	1	0	89	48	76
YUV to HEVC	8k	1	0	10	1	0	73	59	94
AVC to YUV	4k	1	0	8	1	56	0	317	16
HEVC to YUV	4k	1	0	8	1	51	0	336	20
VP9 to YUV	4k	1	0	8	1	24	0	155	4
AVC to YUV	4k	16	0	8	0	99	0	482	3
HEVC to YUV	4k	16	0	8	0	99	0	506	3
VP9 to YUV	4k	16	0	8	0	99	0	488	1
YUV to AVC	4k	1	0	8	1	0	94	294	40
YUV to HEVC	4k	1	0	8	1	0	95	326	48
YUV to AV1	4k	1	0	8	1	0	94	281	40
YUV to AVC	4k	4	0	8	0	0	94	293	21
YUV to HEVC	4k	4	0	8	0	0	96	332	14
YUV to AV1	4k	4	0	8	0	0	96	284	20
YUV to AVC	4k	8	0	8	0	0	100	320	8
YUV to HEVC	4k	8	0	8	0	0	100	344	9
YUV to AV1	4k	8	0	8	0	0	100	296	9
AVC to AVC	4k	1	1	8	1	67	93	222	14
AVC to HEVC	4k	1	1	8	1	71	92	271	15
AVC to AV1	4k	1	1	8	1	61	93	262	13
HEVC to AVC	4k	1	1	8	1	55	92	220	16
HEVC to HEVC	4k	1	1	8	1	58	92	275	15
HEVC to AV1	4k	1	1	8	1	49	92	263	15
VP9 to AVC	4k	1	1	8	1	24	45	149	3
VP9 to HEVC	4k	1	1	8	1	24	43	149	7

TYPE	RES	JOB	HW FRAME	Bit	Joint Mode	DEC_LOAD	ENC_LOAD	FPS	CPU
VP9 to AV1	4k	1	1	8	1	24	50	149	4
AVC to AVC	4k	4	1	8	0	62	96	240	7
AVC to HEVC	4k	4	1	8	0	67	96	300	3
AVC to AV1	4k	4	1	8	0	57	95	272	3
HEVC to AVC	4k	4	1	8	0	57	96	236	8
HEVC to HEVC	4k	4	1	8	0	60	96	300	3
HEVC to AV1	4k	4	1	8	0	47	95	272	3
VP9 to AVC	4k	4	1	8	0	63	96	244	3
VP9 to HEVC	4k	4	1	8	0	65	96	300	3
VP9 to AV1	4k	4	1	8	0	55	96	264	3
AVC to AVC	4k	8	1	8	0	68	100	211	2
AVC to HEVC	4k	8	1	8	0	76	100	280	2
AVC to AV1	4k	8	1	8	0	67	100	272	3
HEVC to AVC	4k	8	1	8	0	64	99	215	2
HEVC to HEVC	4k	8	1	8	0	68	100	280	2
HEVC to AV1	4k	8	1	8	0	58	99	280	3
VP9 to AVC	4k	8	1	8	0	67	100	234	2
VP9 to HEVC	4k	8	1	8	0	72	100	296	2
VP9 to AV1	4k	8	1	8	0	63	100	280	2
AVC to YUV	4k	1	0	10	0	48	0	225	8
HEVC to YUV	4k	1	0	10	0	52	0	232	10
VP9 to YUV	4k	1	0	10	0	24	0	158	3
AVC to YUV	4k	16	0	10	0	98	0	284	0
HEVC to YUV	4k	16	0	10	0	98	0	282	0
VP9 to YUV	4k	16	0	10	0	100	0	501	0
YUV to AVC	4k	1	0	10	0	0	68	199	55
YUV to HEVC	4k	1	0	10	0	0	60	206	61
YUV to AV1	4k	1	0	10	0	0	66	197	57
YUV to AVC	4k	4	0	10	0	0	95	218	30
YUV to HEVC	4k	4	0	10	0	0	77	248	38
YUV to AV1	4k	4	0	10	0	0	81	239	42
AVC to YUV	1080p	1	0	8	1	41	0	861	24
HEVC to YUV	1080p	1	0	8	1	43	0	841	28
VP9 to YUV	1080p	1	0	8	1	22	0	557	6
AVC to YUV	1080p	40	0	8	0	89	0	1726	1
HEVC to YUV	1080p	40	0	8	0	93	0	1823	1
VP9 to YUV	1080p	40	0	8	0	81	0	1762	0
YUV to AVC	1080p	1	0	8	1	0	54	698	27
YUV to HEVC	1080p	1	0	8	1	0	50	701	30
YUV to AV1	1080p	1	0	8	1	0	54	639	24



TYPE	RES	JOB	HW FRAME	Bit	Joint Mode	DEC_LOAD	ENC_LOAD	FPS	CPU
YUV to AVC	1080p	32	0	8	0	0	99	1280	3
YUV to HEVC	1080p	32	0	8	0	0	100	1356	4
YUV to AV1	1080p	32	0	8	0	0	99	1180	3
AVC to AVC	1080p	1	1	8	1	66	79	956	24
AVC to HEVC	1080p	1	1	8	1	65	77	1014	25
AVC to AV1	1080p	1	1	8	1	53	79	893	21
HEVC to AVC	1080p	1	1	8	1	54	76	896	32
HEVC to HEVC	1080p	1	1	8	1	56	76	965	33
HEVC to AV1	1080p	1	1	8	1	47	75	848	31
VP9 to AVC	1080p	1	1	8	1	21	42	548	7
VP9 to HEVC	1080p	1	1	8	1	22	39	539	7
VP9 to AV1	1080p	1	1	8	1	21	46	540	6
AVC to AVC	1080p	32	1	8	0	75	99	938	2
AVC to HEVC	1080p	32	1	8	0	84	99	1058	1
AVC to AV1	1080p	32	1	8	0	76	99	1029	1
HEVC to AVC	1080p	32	1	8	0	69	99	992	1
HEVC to HEVC	1080p	32	1	8	0	76	100	1120	1
HEVC to AV1	1080p	32	1	8	0	70	99	1057	1
VP9 to AVC	1080p	32	1	8	0	70	99	1088	1
VP9 to HEVC	1080p	32	1	8	0	74	99	1216	1
VP9 to AV1	1080p	32	1	8	0	71	99	1120	1
AVC to YUV	1080p	1	0	10	0	30	0	648	9
HEVC to YUV	1080p	1	0	10	0	28	0	649	8
VP9 to YUV	1080p	1	0	10	0	22	0	455	6
AVC to YUV	1080p	40	0	10	0	66	0	1105	0
HEVC to YUV	1080p	40	0	10	0	63	0	1087	0
VP9 to YUV	1080p	40	0	10	0	72	0	1080	0
YUV to AVC	1080p	1	0	10	0	0	37	486	39
YUV to HEVC	1080p	1	0	10	0	0	35	486	39
YUV to AV1	1080p	1	0	10	0	0	38	452	34
YUV to AVC	1080p	32	0	10	0	0	62	798	6
YUV to HEVC	1080p	32	0	10	0	0	58	796	6
YUV to AV1	1080p	32	0	10	0	0	64	768	5
AVC to YUV	720p	1	0	8	1	42	0	1121	15
HEVC to YUV	720p	1	0	8	1	37	0	1116	24
VP9 to YUV	720p	1	0	8	1	33	0	1022	9
AVC to YUV	720p	100	0	8	0	100	0	2523	0
HEVC to YUV	720p	100	0	8	0	100	0	2907	1
VP9 to YUV	720p	100	0	8	0	100	0	2605	0
YUV to AVC	720p	1	0	8	1	0	30	856	20

TYPE	RES	JOB	HW FRAME	Bit	Joint Mode	DEC_LOAD	ENC_LOAD	FPS	CPU
YUV to HEVC	720p	1	0	8	1	0	30	860	19
YUV to AV1	720p	1	0	8	1	0	33	777	15
YUV to AVC	720p	64	0	8	0	0	94	2306	1
YUV to HEVC	720p	64	0	8	0	0	93	2326	2
YUV to AV1	720p	64	0	8	0	0	98	1998	1
AVC to AVC	720p	1	1	8	1	47	47	1245	20
AVC to HEVC	720p	1	1	8	1	47	47	1242	21
AVC to AV1	720p	1	1	8	1	40	50	1078	21
HEVC to AVC	720p	1	1	8	1	39	45	1199	27
HEVC to HEVC	720p	1	1	8	1	40	45	1219	25
HEVC to AV1	720p	1	1	8	1	34	48	1045	24
VP9 to AVC	720p	1	1	8	1	32	35	1003	12
VP9 to HEVC	720p	1	1	8	1	32	34	988	13
VP9 to AV1	720p	1	1	8	1	32	44	987	14
AVC to AVC	720p	64	1	8	0	95	100	2048	0
AVC to HEVC	720p	64	1	8	0	97	100	2113	0
AVC to AV1	720p	64	1	8	0	76	100	1792	0
HEVC to AVC	720p	64	1	8	0	84	100	2048	0
HEVC to HEVC	720p	64	1	8	0	84	100	2117	0
HEVC to AV1	720p	64	1	8	0	68	100	1792	0
VP9 to AVC	720p	64	1	8	0	96	100	2242	0
VP9 to HEVC	720p	64	1	8	0	100	100	2304	0
VP9 to AV1	720p	64	1	8	0	79	100	1920	0

## 2. T1A – Libxcoder Throughput

### 2.1 Decoding

#### 2.1.1 Description

Bitstream is read from an input file on ramdisk and then fed into hardware decoder through PCIe.

Bitstream is decoded by hardware decoder.

Decoded YUV frame is read out through PCIe and written into an output file.

#### 2.1.2 Command Line

```
./ni_xcoder_decode -c 0 -r 1000 -i /media/ramdisk/input.<ext> -m  
<test_type> -o /dev/null -d multicoreJointMode=<*>
```

<test\_type> = test codecs. ie. a (avc), h (hevc), etc

<num\_jobs> is the number of instances running concurrently

<resolution> is resolution of input

<resolution> == 8k, multicoreJointMode = 1

<num\_jobs> == 1, multicoreJointMode = 1

Note: Libxcoder decoding tests were run without multi-threading (but with multicoreJointMode enabled where noted)

### 2.2 Encoding

#### 2.2.1 Description

YUV frame is read from an input file on ramdisk and then fed into hardware encoder through PCIe.

YUV frame is encoded by hardware encoder.

Encoded bitstream is read out through PCIe and written into an output file.

#### 2.2.2 Command Line

```
./ni_xcoder_encode -c 0 -s <resolution> -r 1000 -i  
/media/ramdisk/input.yuv -m <test_type> -o /dev/null -e  
intraPeriod=0:RcEnable=1:bitrate=<*>:keepAliveTimeout=2:multicoreJointM  
ode=<*>
```

<test\_type> = test codecs. ie. a (avc), h (hevc), etc

<num\_jobs> is the number of instances running concurrently

<resolution> is resolution of input

<resolution> == 8k, multicoreJointMode = 1

<num\_jobs> == 1, multicoreJointMode = 1

<resolution> == 8k, bitrate = 50000000, framerate = 24

<resolution> == 4k, bitrate = 12000000, framerate = 30 (8bit) / 60 (10bit)

<resolution> == 1080p, bitrate = 3000000, framerate = 30 (8bit) / 60 (10bit)

<resolution> == 720p, bitrate = 1500000, framerate = 30 (8bit) / 60 (10bit)

Note: Libxcoder encoding tests were run without multi-threading (but with multicoreJointMode enabled where noted)

## 2.3 Transcoding

### 2.3.1 Description

Bitstream is read from an input file on ramdisk and then fed into hardware decoder through PCIe. Bitstream is decoded by hardware decoder.

Decoded YUV frame is kept on device.

The YUV frame is encoded with hardware encoder.

The encoded bitstream is read out through PCIe and written into an output file.

### 2.3.2 Command line

```
./ni_xcoder_multithread_transcode -c 0 -r 1000 -i  
/media/ramdisk/input.<ext> -m <dec_test_type> -n <enc_test_type> -o  
/dev/null -e  
intraPeriod=0:RcEnable=1:bitrate=<*>:keepAliveTimeout=2:multicoreJointM  
ode=<*> -d out=hw:semiplanar0=1:multicoreJointMode=1
```

<dec\_test\_type> = decoding test codecs. ie. a (avc), h (hevc), etc

<enc\_test\_type> = encoding test codecs. ie. a (avc), h (hevc), etc

<num\_jobs> is the number of instances running concurrently

<resolution> is resolution of input

<resolution> == 8k, multicoreJointMode = 1

<num\_jobs> == 1, multicoreJointMode = 1

<resolution> == 8k, bitrate = 50000000, framerate = 24

<resolution> == 4k, bitrate = 12000000, framerate = 30 (8bit) / 60 (10bit)

<resolution> == 1080p, bitrate = 3000000, framerate = 30 (8bit) / 60 (10bit)

<resolution> == 720p, bitrate = 1500000, framerate = 30 (8bit) / 60 (10bit)

## 2.4 Libxcodec Throughput Performance Results

TYPE	RES	JOB	HW FRAME	Bit	Joint Mode	DEC_LOAD	ENC_LOAD	FPS	CPU
AVC to YUV	8k	1	0	8	1	67	0	92	7
HEVC to YUV	8k	1	0	8	1	66	0	96	6
VP9 to YUV	8k	1	0	8	1	24	0	38	3
YUV to AVC	8k	1	0	8	1	0	96	67	27
YUV to HEVC	8k	1	0	8	1	0	96	81	32
AVC to AVC	8k	1	1	8	1	72	99	54	5
AVC to HEVC	8k	1	1	8	1	81	99	71	5
HEVC to AVC	8k	1	1	8	1	64	100	53	4
HEVC to HEVC	8k	1	1	8	1	73	99	70	4
VP9 to AVC	8k	1	1	8	1	24	46	35	3
VP9 to HEVC	8k	1	1	8	1	24	43	37	4
AVC to YUV	8k	1	0	10	1	53	0	54	6
HEVC to YUV	8k	1	0	10	1	48	0	51	6
VP9 to YUV	8k	1	0	10	1	24	0	33	3
YUV to AVC	8k	1	0	10	1	0	65	40	35
YUV to HEVC	8k	1	0	10	1	0	46	39	35
AVC to YUV	4k	1	0	8	1	55	0	310	7
HEVC to YUV	4k	1	0	8	1	48	0	319	7
VP9 to YUV	4k	1	0	8	1	24	0	154	3
AVC to YUV	4k	16	0	8	0	99	0	486	1
HEVC to YUV	4k	16	0	8	0	99	0	511	0
VP9 to YUV	4k	16	0	8	0	99	0	492	0
YUV to AVC	4k	1	0	8	1	0	90	282	24
YUV to HEVC	4k	1	0	8	1	0	83	289	25
YUV to AV1	4k	1	0	8	1	0	91	270	23
YUV to AVC	4k	4	0	8	0	0	96	311	11
YUV to HEVC	4k	4	0	8	0	0	96	333	11
YUV to AV1	4k	4	0	8	0	0	96	288	12
YUV to AVC	4k	8	0	8	0	0	100	325	7
YUV to HEVC	4k	8	0	8	0	0	100	347	8
YUV to AV1	4k	8	0	8	0	0	100	301	9
AVC to AVC	4k	1	1	8	0	68	97	225	7
AVC to HEVC	4k	1	1	8	0	73	96	279	7
AVC to AV1	4k	1	1	8	0	64	96	265	8
HEVC to AVC	4k	1	1	8	0	58	97	225	6
HEVC to HEVC	4k	1	1	8	0	64	96	287	6
HEVC to AV1	4k	1	1	8	0	56	96	266	7
VP9 to AVC	4k	1	1	8	0	24	46	149	4

TYPE	RES	JOB	HW FRAME	Bit	Joint Mode	DEC_LOAD	ENC_LOAD	FPS	CPU
VP9 to HEVC	4k	1	1	8	0	24	43	149	4
VP9 to AV1	4k	1	1	8	0	24	50	149	4
AVC to AVC	4k	4	1	8	0	63	96	243	5
AVC to HEVC	4k	4	1	8	0	67	96	301	5
AVC to AV1	4k	4	1	8	0	56	95	277	5
HEVC to AVC	4k	4	1	8	0	57	96	240	5
HEVC to HEVC	4k	4	1	8	0	60	95	294	5
HEVC to AV1	4k	4	1	8	0	49	95	274	5
VP9 to AVC	4k	4	1	8	0	64	97	246	4
VP9 to HEVC	4k	4	1	8	0	67	95	303	4
VP9 to AV1	4k	4	1	8	0	56	95	277	4
AVC to AVC	4k	8	1	8	0	68	100	217	3
AVC to HEVC	4k	8	1	8	0	76	100	280	3
AVC to AV1	4k	8	1	8	0	67	100	273	4
HEVC to AVC	4k	8	1	8	0	64	100	218	3
HEVC to HEVC	4k	8	1	8	0	67	100	282	3
HEVC to AV1	4k	8	1	8	0	60	100	276	3
VP9 to AVC	4k	8	1	8	0	67	100	236	3
VP9 to HEVC	4k	8	1	8	0	71	100	304	3
VP9 to AV1	4k	8	1	8	0	60	99	288	3
AVC to YUV	4k	1	0	10	1	45	0	214	6
HEVC to YUV	4k	1	0	10	1	42	0	207	6
VP9 to YUV	4k	1	0	10	1	24	0	158	3
AVC to YUV	4k	16	0	10	0	99	0	283	0
HEVC to YUV	4k	16	0	10	0	90	0	284	0
VP9 to YUV	4k	16	0	10	0	99	0	505	0
YUV to AVC	4k	1	0	10	1	0	50	160	26
YUV to HEVC	4k	1	0	10	1	0	46	161	26
YUV to AV1	4k	1	0	10	1	0	50	153	26
YUV to AVC	4k	4	0	10	0	0	65	201	26
YUV to HEVC	4k	4	0	10	0	0	59	210	26
YUV to AV1	4k	4	0	10	0	0	59	186	28
AVC to YUV	1080p	1	0	8	1	39	0	805	13
HEVC to YUV	1080p	1	0	8	1	44	0	851	10
VP9 to YUV	1080p	1	0	8	1	22	0	556	5
AVC to YUV	1080p	40	0	8	0	81	0	1619	0
HEVC to YUV	1080p	40	0	8	0	84	0	1719	0
VP9 to YUV	1080p	40	0	8	0	70	0	1644	0
YUV to AVC	1080p	1	0	8	1	0	53	689	18

TYPE	RES	JOB	HW FRAME	Bit	Joint Mode	DEC_LOAD	ENC_LOAD	FPS	CPU
YUV to HEVC	1080p	1	0	8	1	0	50	691	16
YUV to AV1	1080p	1	0	8	1	0	50	596	16
YUV to AVC	1080p	32	0	8	0	0	99	1325	2
YUV to HEVC	1080p	32	0	8	0	0	99	1401	3
YUV to AV1	1080p	32	0	8	0	0	100	1210	2
AVC to AVC	1080p	1	1	8	0	68	84	999	17
AVC to HEVC	1080p	1	1	8	0	67	81	1052	18
AVC to AV1	1080p	1	1	8	0	54	81	924	17
HEVC to AVC	1080p	1	1	8	0	61	86	1004	13
HEVC to HEVC	1080p	1	1	8	0	64	85	1094	14
HEVC to AV1	1080p	1	1	8	0	51	82	931	14
VP9 to AVC	1080p	1	1	8	0	21	42	545	7
VP9 to HEVC	1080p	1	1	8	0	21	39	540	7
VP9 to AV1	1080p	1	1	8	0	22	46	539	7
AVC to AVC	1080p	32	1	8	0	75	99	977	1
AVC to HEVC	1080p	32	1	8	0	82	99	1104	1
AVC to AV1	1080p	32	1	8	0	79	99	1065	1
HEVC to AVC	1080p	32	1	8	0	69	99	1021	1
HEVC to HEVC	1080p	32	1	8	0	77	99	1156	1
HEVC to AV1	1080p	32	1	8	0	68	99	1090	1
VP9 to AVC	1080p	32	1	8	0	70	99	1115	1
VP9 to HEVC	1080p	32	1	8	0	73	99	1260	1
VP9 to AV1	1080p	32	1	8	0	71	99	1158	1
AVC to YUV	1080p	1	0	10	1	31	0	652	6
HEVC to YUV	1080p	1	0	10	1	28	0	640	6
VP9 to YUV	1080p	1	0	10	1	22	0	455	4
AVC to YUV	1080p	40	0	10	0	67	0	1078	0
HEVC to YUV	1080p	40	0	10	0	69	0	1073	0
VP9 to YUV	1080p	40	0	10	0	69	0	1032	0
YUV to AVC	1080p	1	0	10	1	0	37	477	24
YUV to HEVC	1080p	1	0	10	1	0	34	467	20
YUV to AV1	1080p	1	0	10	1	0	36	418	20
YUV to AVC	1080p	32	0	10	0	0	63	839	6
YUV to HEVC	1080p	32	0	10	0	0	58	826	5
YUV to AV1	1080p	32	0	10	0	0	63	781	8
AVC to YUV	720p	1	0	8	1	41	0	1090	12
HEVC to YUV	720p	1	0	8	1	38	0	1139	9
VP9 to YUV	720p	1	0	8	1	33	0	1019	6
AVC to YUV	720p	100	0	8	0	100	0	2745	0

TYPE	RES	JOB	HW FRAME	Bit	Joint Mode	DEC_LOAD	ENC_LOAD	FPS	CPU
HEVC to YUV	720p	100	0	8	0	95	0	2910	0
VP9 to YUV	720p	100	0	8	0	94	0	2652	0
YUV to AVC	720p	1	0	8	1	0	32	953	10
YUV to HEVC	720p	1	0	8	1	0	33	987	13
YUV to AV1	720p	1	0	8	1	0	33	793	14
YUV to AVC	720p	64	0	8	0	0	100	2642	2
YUV to HEVC	720p	64	0	8	0	0	99	2712	2
YUV to AV1	720p	64	0	8	0	0	100	2189	1
AVC to AVC	720p	1	1	8	0	50	48	1324	17
AVC to HEVC	720p	1	1	8	0	50	49	1329	17
AVC to AV1	720p	1	1	8	0	42	48	1058	16
HEVC to AVC	720p	1	1	8	0	45	49	1362	14
HEVC to HEVC	720p	1	1	8	0	45	50	1357	13
HEVC to AV1	720p	1	1	8	0	38	47	1057	12
VP9 to AVC	720p	1	1	8	0	32	35	1004	8
VP9 to HEVC	720p	1	1	8	0	32	34	991	8
VP9 to AV1	720p	1	1	8	0	31	44	989	9
AVC to AVC	720p	64	1	8	0	100	100	2178	0
AVC to HEVC	720p	64	1	8	0	100	100	2263	0
AVC to AV1	720p	64	1	8	0	93	100	1884	0
HEVC to AVC	720p	64	1	8	0	86	99	2186	0
HEVC to HEVC	720p	64	1	8	0	95	100	2268	0
HEVC to AV1	720p	64	1	8	0	87	100	1891	0
VP9 to AVC	720p	64	1	8	0	100	100	2419	0
VP9 to HEVC	720p	64	1	8	0	100	100	2492	0
VP9 to AV1	720p	64	1	8	0	98	100	2023	0



## 3. T1A – FFmpeg Latency

### 3.1 Encoding

#### 3.1.1 Description

Libxcodec is compiled and installed with parameter `--with-latency-display`

```
$ bash build.sh --with-latency-display
```

YUV frame is read from an input file on ramdisk and then fed into hardware encoder through PCIe.

YUV frame is encoded by hardware encoder.

Encoded bitstream is read out through PCIe and written into an output file.

For each frame, the encoder latency (eLat) value is provided in the output log.

All eLat values are parsed from the output log and the last 50% of frame data before killing ffmpeg instances is used to calculate the Average, Min, Max, and Variance.

The first 50% of frame data are ignored to reach stability while launching multiple jobs.

#### 3.1.2 Command Line

```
ffmpeg -re -loglevel info -f rawvideo -pix_fmt yuv420p -stream_loop  
1000 -s:v <resolution> -i /media/ramdisk/input.yuv -c:v  
<enc>_ni_quadra_enc -enc 0 -xcodec-params gopPresetIdx=9:lowDelay=1 -f  
null -
```

<enc> is the encoder codec. ie h264\_ni\_quadra\_enc, h265\_ni\_quadra\_enc, av1\_ni\_quadra\_enc

<resolution> is resolution of input

### 3.2 FFmpeg Latency Performance Results

TYPE	RESOLUTION	JOBS	ELAT_AVG (ms)	ELAT_MAX (ms)	ELAT_MIN (ms)	ELAT_VAR (ms)
YUV to AVC	8k	1	57.77	59.18	57.31	0.04
YUV to HEVC	8k	1	55.45	60.47	54.03	0.94
YUV to AVC	4k	1	15.33	17.34	15.05	0.06
YUV to HEVC	4k	1	16.7	18.66	15.64	0.11
YUV to AV1	4k	1	21.75	25.44	16.35	0.61
YUV to AVC	4k	4	17.01	22.08	15.03	2.75
YUV to HEVC	4k	4	17.69	21.4	15.6	1.41
YUV to AV1	4k	4	22.79	27.18	16.38	0.8
YUV to AVC	4k	8	18.19	23.35	15.05	3.2
YUV to HEVC	4k	8	22.95	38.51	15.64	25.48
YUV to AV1	4k	8	37.24	46.41	24.11	11.97
YUV to AVC	1080p	1	4.62	5.54	4.42	0.02
YUV to HEVC	1080p	1	4.99	5.55	4.7	0.03
YUV to AV1	1080p	1	6.67	7.4	5.04	0.04
YUV to AVC	1080p	32	6.19	13.95	4.63	0.44
YUV to HEVC	1080p	32	7.03	12.83	4.9	1.31
YUV to AV1	1080p	32	41.02	46.2	34.29	1.26
YUV to AVC	720p	1	2.85	3.51	2.49	0.01
YUV to HEVC	720p	1	2.91	3.38	2.71	0.02
YUV to AV1	720p	1	3.81	4.16	2.98	0.01
YUV to AVC	720p	64	5.38	10.5	2.85	1
YUV to HEVC	720p	64	5.54	9.08	3.03	1
YUV to AV1	720p	64	39.77	47.26	32.38	3.26

## 4. T1A – Decoder PPU Scaling

### 4.1 Decoding

#### 4.1.1 Description

Bitstream is read from an input file on ramdisk and then fed into hardware decoder through PCIe.

Bitstream is decoded by hardware decoder and scaled to 224x224 with decoder post processing unit.

Decoded YUV is kept on device.

The YUV frame is converted to RGBA format with 2D Engine.

The RGBA frame is read out through PCIe and written into an output file.

#### 4.1.2 Command Line

```
ffmpeg -vsync 0 -c:v <dec>_ni_quadra_dec -dec 0 -xcoder-params  
out=hw:scale0=224x224:multicoreJointMode=<resolution=8k?1:0> -f concat  
-safe 0 -i /media/ramdisk/input.list -vf  
ni_quadra_scale=iw:ih:format=rgba,hwdownload,format=rgba -f null -
```

<dec> is the decoder codec. ie h264\_ni\_quadra\_dec, h265\_ni\_quadra\_dec, vp9\_ni\_quadra\_dec

<resolution> is resolution of input

<resolution> == 8k, multicoreJointMode = 1

### 4.2 Decoder PPU Scaling Performance Results

TYPE	RESOLUTION	JOBS	DEC_LOAD	SCALER_LOAD	FPS	CPU
AVC to RGBA	8k	1	90	0	143	7
HEVC to RGBA	8k	1	90	0	154	8
VP9 to RGBA	8k	1	23	0	40	2
AVC to RGBA	4k	1	21	0	145	8
AVC to RGBA	4k	16	95	3	586	2
HEVC to RGBA	4k	1	22	0	174	10
HEVC to RGBA	4k	16	93	4	669	3
VP9 to RGBA	4k	1	22	0	166	4
VP9 to RGBA	4k	16	95	4	680	1
AVC to RGBA	1080p	40	93	14	1897	1
HEVC to RGBA	1080p	40	95	14	2013	1
VP9 to RGBA	1080p	40	94	17	2440	0
AVC to RGBA	720p	100	98	18	2600	0
HEVC to RGBA	720p	100	91	19	2827	0
VP9 to RGBA	720p	64	95	18	2752	0

## 5. T1A – Streaming Ladder Generation

### 5.1 Transcoding

#### 5.1.1 Description

Bitstream is read from an input file on ramdisk and then fed into hardware decoder through PCIe. Bitstream is decoded by hardware decoder split and scaled to smaller resolutions with decoder post processing unit or 2D Engine.

Decoded YUV frame is kept on device.

The YUV frames are encoded with hardware encoder.

The encoded bitstream is read out through PCIe and written into an output file.

#### 5.1.2 Command line

```
ffmpeg -vsync 0 -c:v <dec>_ni_quadra_dec -dec 0 -xcoder-params  
out=hw:sempianar0=1:enableOut1=1:sempianar1=1:scale1=1280x720:enableO  
ut2=1:sempianar2=1:scale2=960x540 -f concat -safe 0 -i  
/media/ramdisk/input.list -filter_complex  
'[0:v]ni_quadra_split=2:1:2[1080p][1080p_1][720p][540p][540p_1];[540p_1  
]ni_quadra_scale=640x360[360p]'-map [1080p] -xcoder-params  
RcEnable=1:bitrate=3500000 -c:v <enc>_ni_quadra_enc -enc 0 -f null - -  
map [1080p_1] -xcoder-params RcEnable=1:bitrate=1800000 -c:v  
<enc>_ni_quadra_enc -enc 0 -f null - -map [720p] -xcoder-params  
RcEnable=1:bitrate=1000000 -c:v <enc>_ni_quadra_enc -enc 0 -f null - -  
map [540p] -xcoder-params RcEnable=1:bitrate=800000 -c:v  
<enc>_ni_quadra_enc -enc 0 -f null - -map [360p] -xcoder-params  
RcEnable=1:bitrate=500000 -c:v <enc>_ni_quadra_enc -enc 0 -f null -
```

<dec> is the decoder codec. ie h264\_ni\_quadra\_dec, h265\_ni\_quadra\_dec, vp9\_ni\_quadra\_dec

<enc> is the encoder codec. ie h264\_ni\_quadra\_enc, h265\_ni\_quadra\_enc, av1\_ni\_quadra\_enc

Input: 1080p

Output: 1080p, 1080p, 720p(PPU Scale), 540p(PPU Scale), 360p(2D Scale)

### 5.2 Streaming Ladder Generation Performance Results

TYPE	JOBS	DEC_LOAD	ENC_LOAD	SCALER_LOAD	FPS	CPU
AVC to AVC	8	33	94	2	433	4
AVC to HEVC	8	32	93	2	468	4
AVC to AV1	8	22	93	2	392	3
HEVC to AVC	8	34	94	2	440	5
HEVC to HEVC	8	32	93	2	472	5
HEVC to AV1	8	24	96	2	400	4
VP9 to AVC	8	40	94	2	432	4
VP9 to HEVC	8	38	93	2	465	4
VP9 to AV1	8	29	96	2	395	3

## 6. T1A – RGBA Encoding

### 6.1 Encoding

#### 6.1.1 Description

RGBA frame is read from an input file on ramdisk and then fed into hardware encoder through PCIe.

RGBA frame is uploaded and encoded by hardware encoder.

Encoded bitstream is read out through PCIe and written into an output file.

#### 6.1.2 Command line

```
ffmpeg -nostdin -stream_loop -1 -f rawvideo -pix_fmt rgba -s:v  
<resolution> -r 30 -i /media/ramdisk/input.rgb -vf  
"ni_quadra_hwupload=0" -c:v <enc>_ni_quadra_enc -enc 0 -xcoder-params  
intraPeriod=0:RcEnable=1:bitrate=<*>:multicoreJointMode=<*> -f null  
/dev/null
```

<enc> is the encoder codec. eg h264\_ni\_quadra\_enc, h265\_ni\_quadra\_enc, av1\_ni\_quadra\_enc

<num\_jobs> is the number of instances running concurrently

<resolution> is resolution of input

<num\_jobs> == 1, multicoreJointMode = 1

<resolution> == 4k, bitrate = 12000000, framerate = 30

<resolution> == 1080p, bitrate = 3000000, framerate = 30

<resolution> == 720p, bitrate = 1500000, framerate = 30

## 6.2 RGBA Encoding Performance Results

TYPE	RES	JOB	Joint Mode	ENC_LOAD	FPS	CPU
RGBA to AVC	4k	1	1	51	161	64
RGBA to HEVC	4k	1	1	46	160	62
RGBA to AV1	4k	1	1	55	163	64
RGBA to AVC	4k	4	0	69	165	39
RGBA to HEVC	4k	4	0	61	168	38
RGBA to AV1	4k	4	0	62	172	43
RGBA to AVC	4k	8	0	62	176	31
RGBA to HEVC	4k	8	0	56	176	30
RGBA to AV1	4k	8	0	62	178	28
RGBA to AVC	1080p	1	1	32	426	46
RGBA to HEVC	1080p	1	1	31	426	43
RGBA to AV1	1080p	1	1	34	409	44
RGBA to AVC	1080p	16	0	59	644	14
RGBA to HEVC	1080p	16	0	54	648	14
RGBA to AV1	1080p	16	0	56	640	15
RGBA to AVC	1080p	32	0	55	665	8
RGBA to HEVC	1080p	32	0	52	670	8
RGBA to AV1	1080p	32	0	57	642	8
RGBA to AVC	720p	1	1	27	675	55
RGBA to HEVC	720p	1	1	28	773	66
RGBA to AV1	720p	1	1	35	684	56
RGBA to AVC	720p	16	0	47	1209	37
RGBA to HEVC	720p	16	0	47	1195	32
RGBA to AV1	720p	16	0	54	1123	31
RGBA to AVC	720p	32	0	46	1165	32
RGBA to HEVC	720p	32	0	45	1169	30
RGBA to AV1	720p	32	0	55	1173	28

## 7. T1A – Encoding EnableRdoQuant/rdoLevel/lookaheadDepth

### 7.1 Encoding

#### 7.1.1 Description

YUV frame is read from an input file on ramdisk and then fed into hardware encoder through PCIe.

YUV frame is encoded by hardware encoder with a mix of xcoder-params EnableRdoQuant, rdoLevel, and lookaheadDepth.

Encoded bitstream is read out through PCIe and written into an output file.

#### 7.1.2 Command line

```
ffmpeg -nostdin -f concat -safe 0 -i /media/ramdisk/input.list -c:v  
<enc>_ni_quadra_enc -enc 0 -xcoder-params  
intraPeriod=0:RcEnable=1:bitrate=<*>:lookaheadDepth=<*>:EnableRdoQuant=  
<*>:rdoLevel=<*> -f null /dev/null -
```

<enc> is the encoder codec. eg h264\_ni\_quadra\_enc, h265\_ni\_quadra\_enc, av1\_ni\_quadra\_enc

<num\_jobs> is the number of instances running concurrently

<resolution> is resolution of input

<resolution> == 4k, bitrate = 12000000, framerate = 30

<resolution> == 1080p, bitrate = 3000000, framerate = 30

## 7.2 Encoding EnableRdoQuant/rdoLevel/lookaheadDepth Performance Results

TYPE	RES	JOB	lookaheadDepth	enableRdoQuant	rdoLevel	ENC LOAD	FPS	CPU
YUV to AVC	4k	4	0	0	1	96	302	21
YUV to HEVC	4k	4	0	0	1	96	332	14
YUV to AV1	4k	4	0	0	1	96	288	12
YUV to AVC	4k	4	0	0	2	93	296	27
YUV to HEVC	4k	4	0	0	2	98	170	7
YUV to AV1	4k	4	0	0	2	98	140	7
YUV to AVC	4k	4	0	0	3	96	299	15
YUV to HEVC	4k	4	0	0	3	98	100	6
YUV to AV1	4k	4	0	0	3	97	76	4
YUV to AVC	4k	4	0	1	1	97	188	8
YUV to HEVC	4k	4	0	1	1	97	237	10
YUV to AVC	4k	4	0	1	2	98	188	8
YUV to HEVC	4k	4	0	1	2	99	108	5
YUV to AVC	4k	4	0	1	3	98	188	7
YUV to HEVC	4k	4	0	1	3	97	68	5
YUV to AVC	4k	4	4	0	1	100	196	9
YUV to HEVC	4k	4	4	0	1	100	228	16
YUV to AV1	4k	4	4	0	1	99	200	9
YUV to AVC	4k	4	4	0	2	100	196	10
YUV to HEVC	4k	4	4	0	2	100	136	6
YUV to AV1	4k	4	4	0	2	99	116	6
YUV to AVC	4k	4	4	0	3	99	197	9
YUV to HEVC	4k	4	4	0	3	99	88	4
YUV to AV1	4k	4	4	0	3	100	72	3
YUV to AVC	4k	4	4	1	1	99	140	6
YUV to HEVC	4k	4	4	1	1	100	180	8
YUV to AVC	4k	4	4	1	2	100	140	6
YUV to HEVC	4k	4	4	1	2	99	96	5
YUV to AVC	4k	4	4	1	3	99	140	6
YUV to HEVC	4k	4	4	1	3	98	64	3
YUV to AVC	4k	4	16	0	1	99	196	9
YUV to HEVC	4k	4	16	0	1	100	228	12
YUV to AV1	4k	4	16	0	1	99	200	11
YUV to AVC	4k	4	16	0	2	99	196	9
YUV to HEVC	4k	4	16	0	2	100	136	7
YUV to AV1	4k	4	16	0	2	100	116	5
YUV to AVC	4k	4	16	0	3	100	196	9
YUV to HEVC	4k	4	16	0	3	99	88	4



TYPE	RES	JOB	lookaheadDepth	enableRdoQuant	rdoLevel	ENC LOAD	FPS	CPU
YUV to AV1	4k	4	16	0	3	100	72	4
YUV to AVC	4k	4	16	1	1	100	140	6
YUV to HEVC	4k	4	16	1	1	100	180	7
YUV to AVC	4k	4	16	1	2	100	140	7
YUV to HEVC	4k	4	16	1	2	99	95	5
YUV to AVC	4k	4	16	1	3	100	140	7
YUV to HEVC	4k	4	16	1	3	100	64	3
YUV to AVC	4k	4	40	0	1	100	196	10
YUV to HEVC	4k	4	40	0	1	99	224	9
YUV to AV1	4k	4	40	0	1	100	196	8
YUV to AVC	4k	4	40	0	2	99	196	8
YUV to HEVC	4k	4	40	0	2	99	136	6
YUV to AV1	4k	4	40	0	2	100	112	6
YUV to AVC	4k	4	40	0	3	99	196	9
YUV to HEVC	4k	4	40	0	3	99	88	5
YUV to AV1	4k	4	40	0	3	100	72	3
YUV to AVC	4k	4	40	1	1	100	136	6
YUV to HEVC	4k	4	40	1	1	100	176	11
YUV to AVC	4k	4	40	1	2	100	136	6
YUV to HEVC	4k	4	40	1	2	100	92	5
YUV to AVC	4k	4	40	1	3	99	136	6
YUV to HEVC	4k	4	40	1	3	99	64	3
YUV to AVC	1080p	20	0	0	1	99	1280	5
YUV to HEVC	1080p	20	0	0	1	99	1360	5
YUV to AV1	1080p	20	0	0	1	100	1179	4
YUV to AVC	1080p	20	0	0	2	100	1280	4
YUV to HEVC	1080p	20	0	0	2	100	686	2
YUV to AV1	1080p	20	0	0	2	100	560	2
YUV to AVC	1080p	20	0	0	3	99	1280	4
YUV to HEVC	1080p	20	0	0	3	100	403	2
YUV to AV1	1080p	20	0	0	3	100	300	1
YUV to AVC	1080p	20	0	1	1	100	760	2
YUV to HEVC	1080p	20	0	1	1	99	960	3
YUV to AVC	1080p	20	0	1	2	99	760	2
YUV to HEVC	1080p	20	0	1	2	100	440	2
YUV to AVC	1080p	20	0	1	3	99	760	2
YUV to HEVC	1080p	20	0	1	3	99	280	1
YUV to AVC	1080p	20	4	0	1	100	700	2
YUV to HEVC	1080p	20	4	0	1	99	820	3

TYPE	RES	JOB	lookaheadDepth	enableRdoQuant	rdoLevel	ENC LOAD	FPS	CPU
YUV to AV1	1080p	20	4	0	1	99	720	2
YUV to AVC	1080p	20	4	0	2	99	700	2
YUV to HEVC	1080p	20	4	0	2	99	517	2
YUV to AV1	1080p	20	4	0	2	99	420	2
YUV to AVC	1080p	20	4	0	3	99	700	2
YUV to HEVC	1080p	20	4	0	3	99	340	2
YUV to AV1	1080p	20	4	0	3	99	267	1
YUV to AVC	1080p	20	4	1	1	99	500	2
YUV to HEVC	1080p	20	4	1	1	99	659	2
YUV to AVC	1080p	20	4	1	2	99	500	2
YUV to HEVC	1080p	20	4	1	2	100	360	1
YUV to AVC	1080p	20	4	1	3	100	501	2
YUV to HEVC	1080p	20	4	1	3	100	240	1
YUV to AVC	1080p	20	16	0	1	99	700	2
YUV to HEVC	1080p	20	16	0	1	99	820	3
YUV to AV1	1080p	20	16	0	1	100	719	2
YUV to AVC	1080p	20	16	0	2	99	700	3
YUV to HEVC	1080p	20	16	0	2	100	503	2
YUV to AV1	1080p	20	16	0	2	99	420	2
YUV to AVC	1080p	20	16	0	3	99	700	2
YUV to HEVC	1080p	20	16	0	3	100	340	1
YUV to AV1	1080p	20	16	0	3	99	264	1
YUV to AVC	1080p	20	16	1	1	100	500	2
YUV to HEVC	1080p	20	16	1	1	99	646	2
YUV to AVC	1080p	20	16	1	2	99	500	2
YUV to HEVC	1080p	20	16	1	2	100	360	1
YUV to AVC	1080p	20	16	1	3	99	500	2
YUV to HEVC	1080p	20	16	1	3	99	240	1
YUV to AVC	1080p	20	40	0	1	99	688	2
YUV to HEVC	1080p	20	40	0	1	99	800	3
YUV to AV1	1080p	20	40	0	1	99	703	3
YUV to AVC	1080p	20	40	0	2	99	688	2
YUV to HEVC	1080p	20	40	0	2	99	500	2
YUV to AV1	1080p	20	40	0	2	100	420	1
YUV to AVC	1080p	20	40	0	3	99	690	2
YUV to HEVC	1080p	20	40	0	3	99	340	1
YUV to AV1	1080p	20	40	0	3	100	260	1
YUV to AVC	1080p	20	40	1	1	99	500	2
YUV to HEVC	1080p	20	40	1	1	100	640	2

TYPE	RES	JOB	lookaheadDepth	enableRdoQuant	rdoLevel	ENC LOAD	FPS	CPU
YUV to AVC	1080p	20	40	1	2	99	500	2
YUV to HEVC	1080p	20	40	1	2	99	360	1
YUV to AVC	1080p	20	40	1	3	99	500	2
YUV to HEVC	1080p	20	40	1	3	100	240	1
YUV to AVC	720p	40	0	0	1	90	2284	2
YUV to HEVC	720p	40	0	0	1	88	2291	2
YUV to AV1	720p	40	0	0	1	95	2018	2
YUV to AVC	720p	40	0	0	2	92	2287	3
YUV to HEVC	720p	40	0	0	2	99	1520	1
YUV to AV1	720p	40	0	0	2	99	1202	1
YUV to AVC	720p	40	0	0	3	91	2284	3
YUV to HEVC	720p	40	0	0	3	99	920	1
YUV to AV1	720p	40	0	0	3	100	650	1
YUV to AVC	720p	40	0	1	1	99	1685	1
YUV to HEVC	720p	40	0	1	1	99	2080	2
YUV to AVC	720p	40	0	1	2	99	1692	1
YUV to HEVC	720p	40	0	1	2	99	960	1
YUV to AVC	720p	40	0	1	3	99	1692	1
YUV to HEVC	720p	40	0	1	3	99	611	1
YUV to AVC	720p	40	4	0	1	100	1440	1
YUV to HEVC	720p	40	4	0	1	100	1360	1
YUV to AV1	720p	40	4	0	1	100	1008	1
YUV to AVC	720p	40	4	0	2	100	1440	1
YUV to HEVC	720p	40	4	0	2	99	1080	1
YUV to AV1	720p	40	4	0	2	99	880	1
YUV to AVC	720p	40	4	0	3	100	1440	1
YUV to HEVC	720p	40	4	0	3	99	720	1
YUV to AV1	720p	40	4	0	3	100	560	1
YUV to AVC	720p	40	4	1	1	99	1079	1
YUV to HEVC	720p	40	4	1	1	100	1322	1
YUV to AVC	720p	40	4	1	2	99	1078	1
YUV to HEVC	720p	40	4	1	2	99	760	1
YUV to AVC	720p	40	4	1	3	99	1079	1
YUV to HEVC	720p	40	4	1	3	100	520	1
YUV to AVC	720p	40	16	0	1	100	1400	1
YUV to HEVC	720p	40	16	0	1	100	1357	1
YUV to AV1	720p	40	16	0	1	100	1060	1
YUV to AVC	720p	40	16	0	2	100	1400	1
YUV to HEVC	720p	40	16	0	2	99	1080	1

TYPE	RES	JOBS	lookaheadDepth	enableRdoQuant	rdoLevel	ENC LOAD	FPS	CPU
YUV to AV1	720p	40	16	0	2	100	880	1
YUV to AVC	720p	40	16	0	3	100	1400	1
YUV to HEVC	720p	40	16	0	3	99	720	1
YUV to AV1	720p	40	16	0	3	99	560	1
YUV to AVC	720p	40	16	1	1	99	1048	1
YUV to HEVC	720p	40	16	1	1	100	1321	1
YUV to AVC	720p	40	16	1	2	99	1051	1
YUV to HEVC	720p	40	16	1	2	99	760	1
YUV to AVC	720p	40	16	1	3	99	1050	1
YUV to HEVC	720p	40	16	1	3	100	520	1
YUV to AVC	720p	40	40	0	1	98	1360	1
YUV to HEVC	720p	40	40	0	1	100	1320	1
YUV to AV1	720p	40	40	0	1	100	1040	1
YUV to AVC	720p	40	40	0	2	100	1360	1
YUV to HEVC	720p	40	40	0	2	99	1040	1
YUV to AV1	720p	40	40	0	2	99	846	1
YUV to AVC	720p	40	40	0	3	100	1360	1
YUV to HEVC	720p	40	40	0	3	99	720	1
YUV to AV1	720p	40	40	0	3	99	560	1
YUV to AVC	720p	40	40	1	1	99	1040	1
YUV to HEVC	720p	40	40	1	1	100	1316	1
YUV to AVC	720p	40	40	1	2	99	1040	1
YUV to HEVC	720p	40	40	1	2	99	760	1
YUV to AVC	720p	40	40	1	3	99	1040	1
YUV to HEVC	720p	40	40	1	3	99	520	1

## 8. T1A – Capped CRF

### 8.1 Encoding with lookaheadDepth

#### 8.1.1 Description

YUV frame is read from an input file on ramdisk and fed into hardware encoder through PCIe.

YUV frame is encoded by hardware encoder with a mix of xcoder-params EnableRdoQuant, rdoLevel, lookaheadDepth, CRF, bitrate, and vbvBufferSize.

Encoded bitstream is read out through PCIe and written into an output file.

#### 8.1.2 Command line

```
ffmpeg -nostdin -f concat -safe 0 -i /media/ramdisk/input.list -c:v  
<enc>_ni_quadra_enc -enc 0 -xcoder-params  
intraPeriod=0:vbvBufferSize=1000:bitrate=<*>:lookaheadDepth=<*>:EnableR  
doQuant=<*>:rdoLevel=<*>:crf=<*> -f null /dev/null -
```

<enc> is the encoder codec. eg h264\_ni\_quadra\_enc, h265\_ni\_quadra\_enc, av1\_ni\_quadra\_enc

<num\_jobs> is the number of instances running concurrently

<resolution> is resolution of input

<resolution> == 4k, bitrate = 12000000, framerate = 30

<resolution> == 1080p, bitrate = 3000000, framerate = 30

## 8.2 Capped CRF Encoding with lookaheadDepth Performance Results

TYPE	RES	JOB	lookaheadDepth	enableRdoQuant	rdoLevel	CRF	ENC LOAD	FPS	CPU
YUV to AVC	1080p	20	0	0	1	19	100	700	2
YUV to HEVC	1080p	20	0	0	1	19	100	820	3
YUV to AV1	1080p	20	0	0	1	19	99	720	2
YUV to AVC	1080p	20	0	0	2	19	99	701	3
YUV to HEVC	1080p	20	0	0	2	19	100	520	2
YUV to AV1	1080p	20	0	0	2	19	99	440	2
YUV to AVC	1080p	20	0	0	3	19	99	701	2
YUV to HEVC	1080p	20	0	0	3	19	100	340	1
YUV to AV1	1080p	20	0	0	3	19	99	280	1
YUV to AVC	1080p	20	0	1	1	19	99	500	2
YUV to HEVC	1080p	20	0	1	1	19	99	660	2
YUV to AVC	1080p	20	0	1	2	19	99	501	2
YUV to HEVC	1080p	20	0	1	2	19	99	360	1
YUV to AVC	1080p	20	0	1	3	19	99	504	2
YUV to HEVC	1080p	20	0	1	3	19	100	240	1
YUV to AVC	1080p	20	4	0	1	19	99	700	3
YUV to HEVC	1080p	20	4	0	1	19	99	820	3
YUV to AV1	1080p	20	4	0	1	19	99	720	2
YUV to AVC	1080p	20	4	0	2	19	100	700	3
YUV to HEVC	1080p	20	4	0	2	19	99	520	2
YUV to AV1	1080p	20	4	0	2	19	100	420	1
YUV to AVC	1080p	20	4	0	3	19	99	700	2
YUV to HEVC	1080p	20	4	0	3	19	99	340	1
YUV to AV1	1080p	20	4	0	3	19	100	269	1
YUV to AVC	1080p	20	4	1	1	19	99	502	2
YUV to HEVC	1080p	20	4	1	1	19	99	660	2
YUV to AVC	1080p	20	4	1	2	19	99	500	2
YUV to HEVC	1080p	20	4	1	2	19	99	360	1
YUV to AVC	1080p	20	4	1	3	19	99	500	2
YUV to HEVC	1080p	20	4	1	3	19	99	240	1
YUV to AVC	1080p	20	16	0	1	19	99	700	3
YUV to HEVC	1080p	20	16	0	1	19	99	820	3
YUV to AV1	1080p	20	16	0	1	19	99	719	2
YUV to AVC	1080p	20	16	0	2	19	99	700	2
YUV to HEVC	1080p	20	16	0	2	19	100	503	2
YUV to AV1	1080p	20	16	0	2	19	99	420	1
YUV to AVC	1080p	20	16	0	3	19	99	700	3
YUV to HEVC	1080p	20	16	0	3	19	99	340	1

TYPE	RES	JOB	lookaheadDepth	enableRdoQuant	rdoLevel	CRF	ENC LOAD	FPS	CPU
YUV to AV1	1080p	20	16	0	3	19	100	261	1
YUV to AVC	1080p	20	16	1	1	19	99	500	2
YUV to HEVC	1080p	20	16	1	1	19	100	644	2
YUV to AVC	1080p	20	16	1	2	19	99	500	2
YUV to HEVC	1080p	20	16	1	2	19	99	360	1
YUV to AVC	1080p	20	16	1	3	19	99	500	2
YUV to HEVC	1080p	20	16	1	3	19	99	240	1
YUV to AVC	1080p	20	40	0	1	19	99	685	2
YUV to HEVC	1080p	20	40	0	1	19	99	800	3
YUV to AV1	1080p	20	40	0	1	19	99	700	2
YUV to AVC	1080p	20	40	0	2	19	99	690	2
YUV to HEVC	1080p	20	40	0	2	19	99	500	2
YUV to AV1	1080p	20	40	0	2	19	100	420	1
YUV to AVC	1080p	20	40	0	3	19	99	691	2
YUV to HEVC	1080p	20	40	0	3	19	99	340	1
YUV to AV1	1080p	20	40	0	3	19	99	260	1
YUV to AVC	1080p	20	40	1	1	19	99	500	2
YUV to HEVC	1080p	20	40	1	1	19	99	640	2
YUV to AVC	1080p	20	40	1	2	19	100	500	2
YUV to HEVC	1080p	20	40	1	2	19	100	359	1
YUV to AVC	1080p	20	40	1	3	19	99	500	2
YUV to HEVC	1080p	20	40	1	3	19	100	240	1
YUV to AVC	1080p	20	0	0	1	23	99	700	2
YUV to HEVC	1080p	20	0	0	1	23	99	820	3
YUV to AV1	1080p	20	0	0	1	23	100	720	2
YUV to AVC	1080p	20	0	0	2	23	99	703	2
YUV to HEVC	1080p	20	0	0	2	23	99	520	2
YUV to AV1	1080p	20	0	0	2	23	99	440	1
YUV to AVC	1080p	20	0	0	3	23	99	700	2
YUV to HEVC	1080p	20	0	0	3	23	99	340	1
YUV to AV1	1080p	20	0	0	3	23	100	280	1
YUV to AVC	1080p	20	0	1	1	23	100	501	2
YUV to HEVC	1080p	20	0	1	1	23	100	660	2
YUV to AVC	1080p	20	0	1	2	23	100	501	2
YUV to HEVC	1080p	20	0	1	2	23	100	360	1
YUV to AVC	1080p	20	0	1	3	23	99	501	2
YUV to HEVC	1080p	20	0	1	3	23	100	240	1
YUV to AVC	1080p	20	4	0	1	23	99	700	2
YUV to HEVC	1080p	20	4	0	1	23	99	820	3

TYPE	RES	JOB	lookaheadDepth	enableRdoQuant	rdoLevel	CRF	ENC LOAD	FPS	CPU
YUV to AV1	1080p	20	4	0	1	23	99	720	2
YUV to AVC	1080p	20	4	0	2	23	99	700	2
YUV to HEVC	1080p	20	4	0	2	23	99	520	2
YUV to AV1	1080p	20	4	0	2	23	99	420	2
YUV to AVC	1080p	20	4	0	3	23	99	700	3
YUV to HEVC	1080p	20	4	0	3	23	100	340	1
YUV to AV1	1080p	20	4	0	3	23	99	267	1
YUV to AVC	1080p	20	4	1	1	23	99	500	2
YUV to HEVC	1080p	20	4	1	1	23	99	660	2
YUV to AVC	1080p	20	4	1	2	23	100	502	2
YUV to HEVC	1080p	20	4	1	2	23	100	360	1
YUV to AVC	1080p	20	4	1	3	23	99	500	2
YUV to HEVC	1080p	20	4	1	3	23	99	240	1
YUV to AVC	1080p	20	16	0	1	23	99	700	2
YUV to HEVC	1080p	20	16	0	1	23	99	818	3
YUV to AV1	1080p	20	16	0	1	23	99	720	2
YUV to AVC	1080p	20	16	0	2	23	99	700	2
YUV to HEVC	1080p	20	16	0	2	23	100	501	2
YUV to AV1	1080p	20	16	0	2	23	100	420	1
YUV to AVC	1080p	20	16	0	3	23	99	700	2
YUV to HEVC	1080p	20	16	0	3	23	99	340	1
YUV to AV1	1080p	20	16	0	3	23	100	262	1
YUV to AVC	1080p	20	16	1	1	23	100	500	2
YUV to HEVC	1080p	20	16	1	1	23	99	641	2
YUV to AVC	1080p	20	16	1	2	23	100	500	2
YUV to HEVC	1080p	20	16	1	2	23	100	360	1
YUV to AVC	1080p	20	16	1	3	23	99	500	2
YUV to HEVC	1080p	20	16	1	3	23	100	240	1
YUV to AVC	1080p	20	40	0	1	23	99	690	2
YUV to HEVC	1080p	20	40	0	1	23	99	801	3
YUV to AV1	1080p	20	40	0	1	23	99	700	2
YUV to AVC	1080p	20	40	0	2	23	100	692	2
YUV to HEVC	1080p	20	40	0	2	23	100	500	2
YUV to AV1	1080p	20	40	0	2	23	99	420	1
YUV to AVC	1080p	20	40	0	3	23	99	689	2
YUV to HEVC	1080p	20	40	0	3	23	99	339	1
YUV to AV1	1080p	20	40	0	3	23	100	260	1
YUV to AVC	1080p	20	40	1	1	23	99	500	2
YUV to HEVC	1080p	20	40	1	1	23	99	640	2



TYPE	RES	JOB	lookaheadDepth	enableRdoQuant	rdoLevel	CRF	ENC LOAD	FPS	CPU
YUV to AVC	1080p	20	40	1	2	23	99	500	2
YUV to HEVC	1080p	20	40	1	2	23	100	360	1
YUV to AVC	1080p	20	40	1	3	23	100	500	2
YUV to HEVC	1080p	20	40	1	3	23	100	240	1
YUV to AVC	1080p	20	0	0	1	27	99	700	2
YUV to HEVC	1080p	20	0	0	1	27	99	820	3
YUV to AV1	1080p	20	0	0	1	27	99	720	2
YUV to AVC	1080p	20	0	0	2	27	99	703	2
YUV to HEVC	1080p	20	0	0	2	27	99	520	2
YUV to AV1	1080p	20	0	0	2	27	99	439	2
YUV to AVC	1080p	20	0	0	3	27	99	700	3
YUV to HEVC	1080p	20	0	0	3	27	100	340	1
YUV to AV1	1080p	20	0	0	3	27	99	280	1
YUV to AVC	1080p	20	0	1	1	27	100	501	2
YUV to HEVC	1080p	20	0	1	1	27	99	660	2
YUV to AVC	1080p	20	0	1	2	27	100	501	2
YUV to HEVC	1080p	20	0	1	2	27	100	360	1
YUV to AVC	1080p	20	0	1	3	27	99	502	2
YUV to HEVC	1080p	20	0	1	3	27	99	240	1
YUV to AVC	1080p	20	4	0	1	27	99	700	2
YUV to HEVC	1080p	20	4	0	1	27	99	820	3
YUV to AV1	1080p	20	4	0	1	27	99	720	2
YUV to AVC	1080p	20	4	0	2	27	100	700	2
YUV to HEVC	1080p	20	4	0	2	27	100	520	2
YUV to AV1	1080p	20	4	0	2	27	99	420	1
YUV to AVC	1080p	20	4	0	3	27	99	700	2
YUV to HEVC	1080p	20	4	0	3	27	100	340	1
YUV to AV1	1080p	20	4	0	3	27	99	263	1
YUV to AVC	1080p	20	4	1	1	27	99	501	2
YUV to HEVC	1080p	20	4	1	1	27	99	660	2
YUV to AVC	1080p	20	4	1	2	27	100	500	2
YUV to HEVC	1080p	20	4	1	2	27	100	360	1
YUV to AVC	1080p	20	4	1	3	27	99	500	2
YUV to HEVC	1080p	20	4	1	3	27	100	240	1
YUV to AVC	1080p	20	16	0	1	27	99	700	2
YUV to HEVC	1080p	20	16	0	1	27	99	820	3
YUV to AV1	1080p	20	16	0	1	27	99	720	2
YUV to AVC	1080p	20	16	0	2	27	99	700	2
YUV to HEVC	1080p	20	16	0	2	27	100	501	2

TYPE	RES	JOB	lookaheadDepth	enableRdoQuant	rdoLevel	CRF	ENC LOAD	FPS	CPU
YUV to AV1	1080p	20	16	0	2	27	99	420	2
YUV to AVC	1080p	20	16	0	3	27	99	700	2
YUV to HEVC	1080p	20	16	0	3	27	99	340	1
YUV to AV1	1080p	20	16	0	3	27	100	261	1
YUV to AVC	1080p	20	16	1	1	27	99	500	2
YUV to HEVC	1080p	20	16	1	1	27	100	645	2
YUV to AVC	1080p	20	16	1	2	27	99	500	2
YUV to HEVC	1080p	20	16	1	2	27	99	360	1
YUV to AVC	1080p	20	16	1	3	27	99	500	2
YUV to HEVC	1080p	20	16	1	3	27	100	240	1
YUV to AVC	1080p	20	40	0	1	27	100	688	2
YUV to HEVC	1080p	20	40	0	1	27	99	802	3
YUV to AV1	1080p	20	40	0	1	27	99	701	2
YUV to AVC	1080p	20	40	0	2	27	100	691	2
YUV to HEVC	1080p	20	40	0	2	27	100	500	2
YUV to AV1	1080p	20	40	0	2	27	99	420	2
YUV to AVC	1080p	20	40	0	3	27	99	692	2
YUV to HEVC	1080p	20	40	0	3	27	100	339	2
YUV to AV1	1080p	20	40	0	3	27	99	261	1
YUV to AVC	1080p	20	40	1	1	27	100	500	2
YUV to HEVC	1080p	20	40	1	1	27	99	640	2
YUV to AVC	1080p	20	40	1	2	27	99	500	2
YUV to HEVC	1080p	20	40	1	2	27	100	360	1
YUV to AVC	1080p	20	40	1	3	27	99	500	2
YUV to HEVC	1080p	20	40	1	3	27	99	240	1

## 9. T1A – Inplace Overlay

### 9.1 Transcoding

#### 9.1.1 Description

A bitstream is read from an input file on ramdisk and then fed into the hardware decoder through PCIe. The bitstream is decoded by the hardware decoder. The decoded YUV frame is kept on the device.

An RGBA image is also uploaded to the device and overlayed onto the video stream via the 2D Engine. The overlayed YUV frames are encoded with the hardware encoder. The encoded bitstream is then read out through PCIe and written into an output file.

#### 9.1.2 Command line

```
ffmpeg -c:v <dec>_ni_quadra_dec -dec 0 -xcoder-params "out=hw" -f
concat -safe 0 -i /media/ramdisk/input.list -f rawvideo -s:v 128x128 -
pix_fmt rgba -i /media/ramdisk/img.rgb -filter_complex
"[1:v]format=rgba,ni_quadra_hwupload=0[a];[0:v][a]ni_quadra_overlay=0:0
:alpha=1:inplace=1[b]" -c:a copy -map "[b]" -c:v <enc>_ni_quadra_enc -
enc 0 -xcoder-params "RcEnable=1:bitrate=2000000" -f null -
```

<dec> is the decoder codec. ie h264\_ni\_quadra\_dec, h265\_ni\_quadra\_dec, vp9\_ni\_quadra\_dec

<enc> is the encoder codec. ie h264\_ni\_quadra\_enc, h265\_ni\_quadra\_enc, av1\_ni\_quadra\_enc

Input Video: 1080p

Input Image: 128x128

## 9.2 Inplace Overlay Performance Results

TYPE	JOBS	FPS	CPU	DEC_LOAD	ENC_LOAD	SCALER_LOAD
AVC to AVC	1	288	11	14	20	5
AVC to HEVC	1	307	12	14	20	5
AVC to AV1	1	262	11	13	20	4
HEVC to AVC	1	288	15	12	21	4
HEVC to HEVC	1	306	16	13	21	5
HEVC to AV1	1	262	14	12	21	4
VP9 to AVC	1	288	11	17	21	5
VP9 to HEVC	1	305	12	17	20	5
VP9 to AV1	1	262	11	15	20	4
AVC to AVC	16	1045	3	71	92	24
AVC to HEVC	16	1159	3	77	92	27
AVC to AV1	16	1110	3	71	93	25
HEVC to AVC	16	1088	4	71	93	24
HEVC to HEVC	16	1216	4	77	94	27
HEVC to AV1	16	1136	4	69	94	25
VP9 to AVC	16	1040	3	83	95	24
VP9 to HEVC	16	1168	3	89	92	26
VP9 to AV1	16	1120	3	83	93	25
AVC to AVC	32	960	1	73	93	23
AVC to HEVC	32	1078	1	82	95	27
AVC to AV1	32	1056	1	76	92	25
HEVC to AVC	32	993	1	71	92	24
HEVC to HEVC	32	1138	2	78	92	27
HEVC to AV1	32	1089	2	72	93	26
VP9 to AVC	32	992	1	81	93	23
VP9 to HEVC	32	1120	1	89	93	26
VP9 to AV1	32	1088	1	84	94	25

## 10. 2x T2A – MultiThread P2P DMA on AMD GPU

### 10.1 Encoding

#### 10.1.1 Description

GPU renders frames in its video memory and will convert it from RGB to YUV.

YUV is transferred directly to Quadra device through peer-to-peer DMA without host PC intervention.

The YUV frame is encoded with hardware encoder.

The encoded bitstream is read out through PCIe and written into an output file.

#### 10.1.2 Command line

```
sudo python3 ~/FFmpegXcoder/amd-multi-  
thread/scripts/run_multiple_encoding.py --frames 1000 --instance  
<num_jobs> --codec 0
```

<num\_jobs> = number of instances running concurrently

### 10.2 Multi Thread P2P DMA on AMD GPU Performance Results

TYPE	RES	JOBS	Frames	FPS	CPU	Enc Load	P2P MEM	Latency Avg	Latency Dev
P2A	720p	1	1001	29	1.8	0.25	0.5	2	0
P2H	720p	1	1001	30	1.2	0.25	0.5	2	0.01
P2A	720p	180	180180	29.7	0.77	46	56	2.36	0.12
P2H	720p	180	180180	29.3	0.77	45	56	4.2	0.13
P2A	1080p	1	1001	29	1.2	0.5	0.75	3.72	0.01
P2H	1080p	1	1001	30	1.2	0.5	0.75	3.5	0
P2A	1080p	80	80080	29.7	0.89	46	50	4.09	0.54
P2H	1080p	80	80080	29.4	0.97	43	50	3.7	0.12

## 11. T1A – AI

### 11.1 AI Model

#### 11.1.1 Description

aiperf reads the network binary file provided on the command line and sends the data to the device through the PCIe bus. At the device side, the network binary is unfolded into memory and initializes the AI hardware.

aiperf sends and receives the model input and output parameters from the device through the PCIe bus.

aiperf performs any data format conversion expected by the hardware.

After conversion, aiperf writes the input data to the device through the PCIe bus. The device receives the input data buffer address, then trigger the hardware to start the inference.

When the device has completed the inference, aiperf then reads the output data from the device through the PCIe bus.

The output data is converted to tensor data or binary data, based on the hardware and model format.

#### 11.1.2 Command line

```
sudo ./aiperf -conf_file config_example.json
```

In config\_example.json, user need to specify the following arguments

```
{
    "nb": "/path/to/network_binary_0.nb",
    "dataset": "/path/to/dataset0.txt",
    "outdir": "/path/to/output0",
    "format": "nchw",
    "order": "rgb",
    "devid": "0",
    "loop": "10000"
}
```

In dataset0.txt, user need to specify the path to the input batch (image or tensor)  
/path/to/image.png

### 11.1.3 AI Model Performance Results

Model	Session Number	Loops	Channel Order	File Format	File Type	Input Size	FPS per session
yolov5s_640	8	10000	rgb	nchw	image	640x640x3	78
yolov5s_320	8	10000	rgb	nchw	image	320x320x3	278
deeplabv3_FRP	8	10000	rgb	nchw	image	257x257x3	359
resnet50	8	10000	rgb	nchw	image	224x224x3	228
mobilenetv2	8	10000	rgb	nchw	image	224x224x3	1107
deeplabv3	8	10000	rgb	nchw	image	257x257x3	160
yolov4	8	10000	bgr	nchw	image	416x416x3	258
fsrcnn	8	2000	bgr	nchw	image	360x640x1	31
BiSeNetv1	8	10000	rgb	nchw	image	512x512x3	76
HrNet	8	10000	rgb	nchw	image	256x192x3	74
usm_1656x1920	8	10000	rgb	nchw	image	1920x1656x1	250
usm_3240x3840	8	10000	rgb	nchw	image	3840x3240x1	61
lpips	8	2000	rgb	nchw	image	720x480x3	1
PaddleOCR-512_onnx	8	10000	rgb	nchw	image	512x48x3	16
segm32_tflite_kl_mle	8	10000	rgb	nchw	image	256x144x3	836
mobilenetv2_nchw_keras_96x160	1	10000	rgb	nchw	image	96x160x3	2347
mobilenetv2_nchw_keras_96x160	8	10000	rgb	nchw	image	96x160x3	2261
mobilenetv2_nchw_keras_96x160	16	10000	rgb	nchw	image	96x160x3	2281

## 11.2 AI Encoding with 2D Engine

### 11.2.1 Description BG Filter

The FFmpeg Background Removal filter analyses input frames, infers these input images using the AI module (segm32), segments the foreground and background of the input images, and then removes the background.

With the features of 2D scale, AI inference, alpha merge, and 2D overlay, the background removal filter can remove the background of the input frame.

### 11.2.2 Description ROI Filter

The FFmpeg ROI filter makes inferences from input frames using the AI module in Quadra. It identifies the bounding coordinates of chosen objects and classes within the images, and then wraps the coordinates into ROI side data.

All ROI side data within an image is appended to, then passed down to the encoder along with the actual images themselves.

### 11.2.3 Description PRE Filter

The FFMPEG PRE filter makes YUV previous processing by custom AI model. The input and output are both a single Quadra HW Frame. The actual effect is determined by the AI model.

### 11.2.4 Command line BG

```
ffmpeg -vsync 0 -init_hw_device ni_quadra=foo:0 -dec 0 -c:v
h264_ni_quadra_dec -xcoder-params 'out=hw' -i bg_1920x1080.h264 -
filter_hw_device foo -vf
'ni_quadra_bg=nb=segm32_tflite_nchw_bgr.nb:use_default_bg=1' -enc 0 -
c:v h264_ni_quadra_enc -xcoder-params "RcEnable=1" -f null -
```

### 11.2.5 Command line ROI

```
ffmpeg -vsync 0 -init_hw_device ni_quadra=foo:0 -dec 0 -c:v
h264_ni_quadra_dec -xcoder-params 'out=hw' -i cr7_1920x1080.h264 -
filter_hw_device foo -vf
'ni_quadra_roi=nb=network_binary_yolov4_head.nb:qpoffset=-0.3' -enc 0 -
c:v h264_ni_quadra_enc -xcoder-params 'roiEnable=1:RcEnable=1' -f null
-
```

### 11.2.6 Command line PRE

```
ffmpeg -vsync 0 -dec 0 -c:v h264_ni_quadra_dec -xcoder-params 'out=hw'
-f concat -safe 0 -i pre_1920x1080.h264.list -vf
ni_quadra_ai_pre=nb=<hw_*_network_binary>:width=1280:height=720 -enc 0
-c:v h265_ni_quadra_enc -xcoder-params RcEnable=1:bitrate=1000000 -f
null -
```



### 11.2.7 AI Encoding with 2D Engine Performance Results

Filter	Model	Resolution	Session Number	Average FPS per session
ROI	network_binary_yolov4_head	1920x1080	1	81
ROI	network_binary_yolov4_head	1920x1080	8	24
ROI	network_binary_yolov4_head	1920x1080	32	5
BG	segm32_tflite_nchw_bgr	1920x1080	1	68
BG	segm32_tflite_nchw_bgr	1920x1080	8	40
BG	segm32_tflite_nchw_bgr	1920x1080	32	14
PRE	hw_lanczos_network_binary	1920x1080	8	66
PRE	hw_lanczos_network_binary	1920x1080	16	34
PRE	hw_bicubic_network_binary	1920x1080	8	66
PRE	hw_bicubic_network_binary	1920x1080	16	34

## 12. T1A – GStreamer XStack Throughput

### 12.1 Transcoding

#### 12.1.1 Description

Bitstreams are read from multiple input files on ramdisk and then fed into hardware decoder through PCIe. Bitstreams are decoded by hardware decoder.

Decoded YUV frames are all kept on device and are sent through the ni\_quadra\_xstack filter to produce a single YUV output.

The YUV frame is encoded with hardware encoder.

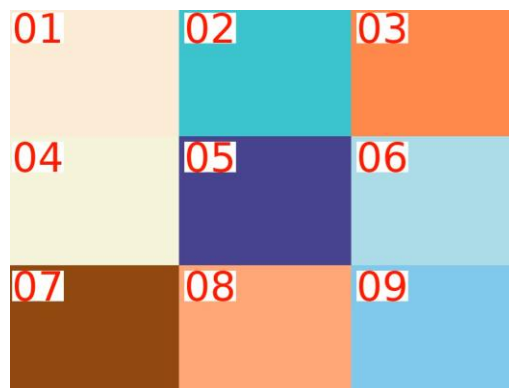
The encoded bitstream is read out through PCIe and written into an output file.

In this test, XStack will generate a single video output in a 3x3, 4x4, or 4x8 grid format generated from 9, 16, or 32 inputs, respectively.

Each input will scale to a cell size and be placed in the grid layout.

The grid layout and cell size will determine the output resolution.

This test is HEVC to AVC only.



*Example output in a 3x3 layout with 9 inputs*

#### 12.1.2 Command line

See Appendix A: GStreamer XStack Command

## 12.2 GStreamer XStack Performance Results

Input Res	Grid	Output Res	Cell Size	FPS	CPU	Dec Load	Enc Load	Scaler Load
1920x1080	3x3	1920x1080	640x360	159.75	33	70	15	13
1920x1080	4x4	1920x1080	480x270	105.29	43	82	10	11
1920x1080	4x8	1920x1080	480x135	56.8	54	87	5	6
1920x1080	3x3	3840x2160	1280x720	70.08	17	31	21	11
1920x1080	4x4	3840x2160	960x540	60.67	30	49	21	14
1920x1080	4x8	3840x2160	960x270	46.72	46	76	17	14
1920x1080	3x3	7680x4320	2560x1440	19.88	11	8	22	9
1920x1080	4x4	7680x4320	1920x1080	19.16	13	15	22	10
1920x1080	4x8	7680x4320	1920x540	19.32	22	32	22	11
1920x1080	6x6	1920x1080	320x180	50.17	59	88	5	8
1920x1080	7x7	1920x1080	274x154 276x154 274x156 276x156*	37.19	60	87	4	7

\*7x7 uses multiple cell sizes. See Appendix B: 7x7 Grid Layout for a visual

## 13. T1A – GStreamer Ladder Generation

### 13.1 Transcoding

#### 13.1.1 Description

Bitstream is read from an input file on ramdisk and then fed into hardware decoder through PCIe. Bitstream is decoded by hardware decoder.

Decoded YUV is split to multiple pads.

The YUV frames are encoded with hardware encoder.

The encoded bitstream is read out through PCIe and written into an output file.

This test will generate 64 outputs of 1080p from a single 1080p input

This test is AVC to HEVC only

#### 13.1.2 Command line

See Appendix C: GStreamer Ladder Command

## 13.2 GStreamer Ladder Performance Results

Jobs	Outputs	FPS	CPU	Dec Load	Enc Load
1	64	18.95	58	1	90

## 14. T1U – FFmpeg Throughput

### 14.1 Decoding

#### 14.1.1 Description

Bitstream is read from an input file on ramdisk and then fed into hardware decoder through PCIe.

Bitstream is decoded by hardware decoder.

Decoded YUV frame is read out through PCIe and written into an output file.

#### 14.1.2 Command Line

```
ffmpeg -nostdin -f concat -safe 0 -c:v <dec>_ni_quadra_dec -dec 0 -  
xcoder-params multicoreJointMode=<*> -i /media/ramdisk/input.list -f  
null /dev/null -
```

<dec> is the decoder codec. eg h264\_ni\_quadra\_dec, h265\_ni\_quadra\_dec, vp9\_ni\_quadra\_dec

<num\_jobs> is the number of instances running concurrently

<resolution> is resolution of input

<resolution> == 8k, multicoreJointMode = 1

<num\_jobs> == 1, multicoreJointMode = 1

### 14.2 Encoding

#### 14.2.1 Description

YUV frame is read from an input file on ramdisk and then fed into hardware encoder through PCIe.

YUV frame is encoded by hardware encoder.

Encoded bitstream is read out through PCIe and written into an output file.

#### 14.2.2 Command Line

```
ffmpeg -nostdin -f concat -safe 0 -i /media/ramdisk/input.list -c:v  
<enc>_ni_quadra_enc -enc 0 -xcoder-params  
intraPeriod=0:RcEnable=1:bitrate=<*>:multicoreJointMode=<*> -f null  
/dev/null -
```

<enc> is the encoder codec. eg h264\_ni\_quadra\_enc, h265\_ni\_quadra\_enc, av1\_ni\_quadra\_enc

<num\_jobs> is the number of instances running concurrently

<resolution> is resolution of input

<resolution> == 8k, multicoreJointMode = 1

<num\_jobs> == 1, multicoreJointMode = 1

<resolution> == 8k, bitrate = 50000000, framerate = 24

<resolution> == 4k, bitrate = 12000000, framerate = 30 (8bit) / 60 (10bit)

<resolution> == 1080p, bitrate = 3000000, framerate = 30 (8bit) / 60 (10bit)

<resolution> == 720p, bitrate = 1500000, framerate = 30 (8bit) / 60 (10bit)

## 14.3 Transcoding

### 14.3.1 Description

Bitstream is read from an input file on ramdisk and then fed into hardware decoder through PCIe. Bitstream is decoded by hardware decoder.

Decoded YUV frame is kept on device.

The YUV frame is encoded with hardware encoder.

The encoded bitstream is read out through PCIe and written into an output file.

### 14.3.2 Command line

```
ffmpeg -nostdin -f concat -safe 0 -c:v <dec>_ni_quadra_dec -dec 0 -  
xcoder-params out=hw:sempianar0=1:multicoreJointMode=<*> -i  
/media/ramdisk/input.list -c:v <enc>_ni_quadra_enc -enc 0 -xcoder-  
params intraPeriod=0:RcEnable=1:bitrate=<*>:multicoreJointMode=<*> -f  
null /dev/null -
```

<dec> is the decoder codec. ie h264\_ni\_quadra\_dec, h265\_ni\_quadra\_dec, vp9\_ni\_quadra\_dec

<enc> is the encoder codec. ie h264\_ni\_quadra\_enc, h265\_ni\_quadra\_enc, av1\_ni\_quadra\_enc

<num\_jobs> is the number of instances running concurrently

<resolution> is resolution of input

<resolution> == 8k, multicoreJointMode = 1

<num\_jobs> == 1, multicoreJointMode = 1

<resolution> == 8k, bitrate = 50000000, framerate = 24

<resolution> == 4k, bitrate = 12000000, framerate = 30 (8bit) / 60 (10bit)

<resolution> == 1080p, bitrate = 3000000, framerate = 30 (8bit) / 60 (10bit)

<resolution> == 720p, bitrate = 1500000, framerate = 30 (8bit) / 60 (10bit)

#### 14.4 FFmpeg Throughput Performance Results

TYPE	RES	JOB	HW FRAME	Bit	Joint Mode	DEC_LOAD	ENC_LOAD	FPS	CPU
AVC to YUV	8k	1	0	8	1	93	0	99	14
HEVC to YUV	8k	1	0	8	1	89	0	97	18
VP9 to YUV	8k	1	0	8	1	24	0	30	6
YUV to AVC	8k	1	0	8	1	0	96	57	80
YUV to HEVC	8k	1	0	8	1	0	97	70	103
AVC to AVC	8k	1	1	8	1	64	99	48	1
AVC to HEVC	8k	1	1	8	1	81	99	64	10
HEVC to AVC	8k	1	1	8	1	56	99	47	4
HEVC to HEVC	8k	1	1	8	1	69	100	63	10
VP9 to AVC	8k	1	1	8	1	25	43	29	1
VP9 to HEVC	8k	1	1	8	1	24	40	30	1
AVC to YUV	8k	1	0	10	1	61	0	55	6
HEVC to YUV	8k	1	0	10	1	91	0	56	7
VP9 to YUV	8k	1	0	10	1	24	0	28	6
YUV to AVC	8k	1	0	10	1	0	78	39	117
YUV to HEVC	8k	1	0	10	1	0	60	39	121
AVC to YUV	4k	1	0	8	1	64	0	274	19
HEVC to YUV	4k	1	0	8	1	53	0	283	16
VP9 to YUV	4k	1	0	8	1	24	0	108	9
AVC to YUV	4k	16	0	8	0	98	0	420	0
HEVC to YUV	4k	16	0	8	0	100	0	448	2
VP9 to YUV	4k	16	0	8	0	98	0	401	0
YUV to AVC	4k	1	0	8	1	0	95	252	40
YUV to HEVC	4k	1	0	8	1	0	96	278	43
YUV to AV1	4k	1	0	8	1	0	94	240	46
YUV to AVC	4k	4	0	8	0	0	96	264	13
YUV to HEVC	4k	4	0	8	0	0	96	284	13
YUV to AV1	4k	4	0	8	0	0	96	248	12
YUV to AVC	4k	8	0	8	0	0	100	280	9
YUV to HEVC	4k	8	0	8	0	0	99	296	8
YUV to AV1	4k	8	0	8	0	0	100	256	9
AVC to AVC	4k	1	1	8	1	67	94	199	12
AVC to HEVC	4k	1	1	8	1	72	93	247	14
AVC to AV1	4k	1	1	8	1	61	94	233	9
HEVC to AVC	4k	1	1	8	1	54	93	199	15
HEVC to HEVC	4k	1	1	8	1	57	92	248	13
HEVC to AV1	4k	1	1	8	1	49	93	227	14
VP9 to AVC	4k	1	1	8	1	24	38	108	6
VP9 to HEVC	4k	1	1	8	1	24	36	108	4

TYPE	RES	JOB	HW FRAME	Bit	Joint Mode	DEC_LOAD	ENC_LOAD	FPS	CPU
VP9 to AV1	4k	1	1	8	1	24	41	108	6
AVC to AVC	4k	4	1	8	0	62	95	220	2
AVC to HEVC	4k	4	1	8	0	69	93	259	3
AVC to AV1	4k	4	1	8	0	55	95	232	6
HEVC to AVC	4k	4	1	8	0	54	97	216	3
HEVC to HEVC	4k	4	1	8	0	56	96	264	1
HEVC to AV1	4k	4	1	8	0	46	95	236	7
VP9 to AVC	4k	4	1	8	0	64	96	216	1
VP9 to HEVC	4k	4	1	8	0	70	96	264	2
VP9 to AV1	4k	4	1	8	0	61	93	236	7
AVC to AVC	4k	8	1	8	0	68	99	195	1
AVC to HEVC	4k	8	1	8	0	75	99	255	1
AVC to AV1	4k	8	1	8	0	67	99	240	6
HEVC to AVC	4k	8	1	8	0	61	99	195	1
HEVC to HEVC	4k	8	1	8	0	64	99	250	2
HEVC to AV1	4k	8	1	8	0	55	99	240	7
VP9 to AVC	4k	8	1	8	0	66	99	197	1
VP9 to HEVC	4k	8	1	8	0	73	100	256	2
VP9 to AV1	4k	8	1	8	0	66	99	247	4
AVC to YUV	4k	1	0	10	0	46	0	189	10
HEVC to YUV	4k	1	0	10	0	47	0	189	10
VP9 to YUV	4k	1	0	10	0	24	0	126	4
AVC to YUV	4k	16	0	10	0	99	0	252	0
HEVC to YUV	4k	16	0	10	0	100	0	251	0
VP9 to YUV	4k	16	0	10	0	98	0	434	0
YUV to AVC	4k	1	0	10	0	0	65	164	61
YUV to HEVC	4k	1	0	10	0	0	57	169	62
YUV to AV1	4k	1	0	10	0	0	67	167	56
YUV to AVC	4k	4	0	10	0	0	90	189	30
YUV to HEVC	4k	4	0	10	0	0	77	216	40
YUV to AV1	4k	4	0	10	0	0	86	214	38
AVC to YUV	1080p	1	0	8	1	46	0	745	21
HEVC to YUV	1080p	1	0	8	1	50	0	715	26
VP9 to YUV	1080p	1	0	8	1	22	0	446	6
AVC to YUV	1080p	40	0	8	0	98	0	1521	2
HEVC to YUV	1080p	40	0	8	0	99	0	1518	2
VP9 to YUV	1080p	40	0	8	0	92	0	1609	0
YUV to AVC	1080p	1	0	8	1	0	55	620	24
YUV to HEVC	1080p	1	0	8	1	0	53	617	30
YUV to AV1	1080p	1	0	8	1	0	57	571	21

TYPE	RES	JOB	HW FRAME	Bit	Joint Mode	DEC_LOAD	ENC_LOAD	FPS	CPU
YUV to AVC	1080p	32	0	8	0	0	99	1106	3
YUV to HEVC	1080p	32	0	8	0	0	99	1173	3
YUV to AV1	1080p	32	0	8	0	0	99	1024	2
AVC to AVC	1080p	1	1	8	1	67	82	864	19
AVC to HEVC	1080p	1	1	8	1	66	78	905	24
AVC to AV1	1080p	1	1	8	1	54	81	774	20
HEVC to AVC	1080p	1	1	8	1	59	77	806	31
HEVC to HEVC	1080p	1	1	8	1	62	76	857	27
HEVC to AV1	1080p	1	1	8	1	54	80	739	27
VP9 to AVC	1080p	1	1	8	1	22	40	445	4
VP9 to HEVC	1080p	1	1	8	1	22	38	444	6
VP9 to AV1	1080p	1	1	8	1	22	44	441	4
AVC to AVC	1080p	32	1	8	0	76	99	864	0
AVC to HEVC	1080p	32	1	8	0	84	99	967	0
AVC to AV1	1080p	32	1	8	0	77	99	928	2
HEVC to AVC	1080p	32	1	8	0	71	99	896	3
HEVC to HEVC	1080p	32	1	8	0	77	99	995	1
HEVC to AV1	1080p	32	1	8	0	68	99	938	3
VP9 to AVC	1080p	32	1	8	0	69	99	992	0
VP9 to HEVC	1080p	32	1	8	0	74	99	1091	1
VP9 to AV1	1080p	32	1	8	0	68	99	992	1
AVC to YUV	1080p	1	0	10	0	30	0	470	6
HEVC to YUV	1080p	1	0	10	0	28	0	471	9
VP9 to YUV	1080p	1	0	10	0	22	0	454	4
AVC to YUV	1080p	40	0	10	0	70	0	1040	0
HEVC to YUV	1080p	40	0	10	0	81	0	1040	0
VP9 to YUV	1080p	40	0	10	0	92	0	1640	0
YUV to AVC	1080p	1	0	10	0	0	36	403	37
YUV to HEVC	1080p	1	0	10	0	0	34	399	37
YUV to AV1	1080p	1	0	10	0	0	37	378	36
YUV to AVC	1080p	32	0	10	0	0	72	768	6
YUV to HEVC	1080p	32	0	10	0	0	67	772	7
YUV to AV1	1080p	32	0	10	0	0	75	749	8
AVC to YUV	720p	1	0	8	1	44	0	1083	15
HEVC to YUV	720p	1	0	8	1	38	0	1077	21
VP9 to YUV	720p	1	0	8	1	29	0	845	12
AVC to YUV	720p	100	0	8	0	100	0	2310	0
HEVC to YUV	720p	100	0	8	0	95	0	2623	0
VP9 to YUV	720p	100	0	8	0	100	0	2400	0
YUV to AVC	720p	1	0	8	1	0	33	817	15



TYPE	RES	JOB	HW FRAME	Bit	Joint Mode	DEC_LOAD	ENC_LOAD	FPS	CPU
YUV to HEVC	720p	1	0	8	1	0	32	810	22
YUV to AV1	720p	1	0	8	1	0	34	722	16
YUV to AVC	720p	64	0	8	0	0	98	2196	2
YUV to HEVC	720p	64	0	8	0	0	99	2273	1
YUV to AV1	720p	64	0	8	0	0	98	1856	1
AVC to AVC	720p	1	1	8	1	47	48	1163	21
AVC to HEVC	720p	1	1	8	1	48	47	1167	18
AVC to AV1	720p	1	1	8	1	40	50	1002	16
HEVC to AVC	720p	1	1	8	1	40	45	1118	24
HEVC to HEVC	720p	1	1	8	1	39	44	1126	26
HEVC to AV1	720p	1	1	8	1	35	48	985	20
VP9 to AVC	720p	1	1	8	1	29	33	836	8
VP9 to HEVC	720p	1	1	8	1	28	33	833	8
VP9 to AV1	720p	1	1	8	1	29	40	827	8
AVC to AVC	720p	64	1	8	0	95	100	1815	1
AVC to HEVC	720p	64	1	8	0	97	100	1917	0
AVC to AV1	720p	64	1	8	0	77	100	1608	0
HEVC to AVC	720p	64	1	8	0	84	100	1846	2
HEVC to HEVC	720p	64	1	8	0	86	100	1920	0
HEVC to AV1	720p	64	1	8	0	67	100	1619	0
VP9 to AVC	720p	64	1	8	0	99	100	2048	0
VP9 to HEVC	720p	64	1	8	0	99	100	2112	0
VP9 to AV1	720p	64	1	8	0	75	98	1728	0

## 15. T1U – Libxcoder Throughput

### 15.1 Decoding

#### 15.1.1 Description

Bitstream is read from an input file on ramdisk and then fed into hardware decoder through PCIe.

Bitstream is decoded by hardware decoder.

Decoded YUV frame is read out through PCIe and written into an output file.

#### 15.1.2 Command Line

```
./ni_xcoder_decode -c 0 -r 1000 -i /media/ramdisk/input.<ext> -m  
<test_type> -o /dev/null -d multicoreJointMode=<*>
```

<test\_type> = test codecs. ie. a (avc), h (hevc), etc

<num\_jobs> is the number of instances running concurrently

<resolution> is resolution of input

<resolution> == 8k, multicoreJointMode = 1

<num\_jobs> == 1, multicoreJointMode = 1

Note: Libxcoder decoding tests were run without multi-threading (but with multicoreJointMode enabled where noted)

### 15.2 Encoding

#### 15.2.1 Description

YUV frame is read from an input file on ramdisk and then fed into hardware encoder through PCIe.

YUV frame is encoded by hardware encoder.

Encoded bitstream is read out through PCIe and written into an output file.

#### 15.2.2 Command Line

```
./ni_xcoder_encode -c 0 -s <resolution> -r 1000 -i  
/media/ramdisk/input.yuv -m <test_type> -o /dev/null -e  
intraPeriod=0:RcEnable=1:bitrate=<*>:keepAliveTimeout=2:multicoreJointM  
ode=<*>
```

<test\_type> = test codecs. ie. a (avc), h (hevc), etc

<num\_jobs> is the number of instances running concurrently

<resolution> is resolution of input

<resolution> == 8k, multicoreJointMode = 1

<num\_jobs> == 1, multicoreJointMode = 1

<resolution> == 8k, bitrate = 50000000, framerate = 24

<resolution> == 4k, bitrate = 12000000, framerate = 30 (8bit) / 60 (10bit)

<resolution> == 1080p, bitrate = 3000000, framerate = 30 (8bit) / 60 (10bit)

<resolution> == 720p, bitrate = 1500000, framerate = 30 (8bit) / 60 (10bit)

Note: Libxcoder encoding tests were run without multi-threading (but with multicoreJointMode enabled where noted)

## 15.3 Transcoding

### 15.3.1 Description

Bitstream is read from an input file on ramdisk and then fed into hardware decoder through PCIe. Bitstream is decoded by hardware decoder.

Decoded YUV frame is kept on device.

The YUV frame is encoded with hardware encoder.

The encoded bitstream is read out through PCIe and written into an output file.

### 15.3.2 Command line

```
./ni_xcoder_multithread_transcode -c 0 -r 1000 -i  
/media/ramdisk/input.<ext> -m <dec_test_type> -n <enc_test_type> -o  
/dev/null -e  
intraPeriod=0:RcEnable=1:bitrate=<*>:keepAliveTimeout=2:multicoreJointM  
ode=<*> -d out=hw:semiplanar0=1:multicoreJointMode=1
```

<dec\_test\_type> = decoding test codecs. ie. a (avc), h (hevc), etc

<enc\_test\_type> = encoding test codecs. ie. a (avc), h (hevc), etc

<num\_jobs> is the number of instances running concurrently

<resolution> is resolution of input

<resolution> == 8k, multicoreJointMode = 1

<num\_jobs> == 1, multicoreJointMode = 1

<resolution> == 8k, bitrate = 50000000, framerate = 24

<resolution> == 4k, bitrate = 12000000, framerate = 30 (8bit) / 60 (10bit)

<resolution> == 1080p, bitrate = 3000000, framerate = 30 (8bit) / 60 (10bit)

<resolution> == 720p, bitrate = 1500000, framerate = 30 (8bit) / 60 (10bit)

## 15.4 Libxcode Throughput Performance Results

TYPE	RES	JOB	HW FRAME	Bit	Joint Mode	DEC_LOAD	ENC_LOAD	FPS	CPU
AVC to YUV	8k	1	0	8	1	75	0	85	8
HEVC to YUV	8k	1	0	8	1	73	0	87	7
VP9 to YUV	8k	1	0	8	1	24	0	30	2
YUV to AVC	8k	1	0	8	1	0	92	55	25
YUV to HEVC	8k	1	0	8	1	0	96	69	30
AVC to AVC	8k	1	1	8	1	73	99	48	4
AVC to HEVC	8k	1	1	8	1	80	99	64	4
HEVC to AVC	8k	1	1	8	1	64	100	47	3
HEVC to HEVC	8k	1	1	8	1	69	99	62	3
VP9 to AVC	8k	1	1	8	1	24	42	29	2
VP9 to HEVC	8k	1	1	8	1	24	39	30	2
AVC to YUV	8k	1	0	10	1	62	0	50	7
HEVC to YUV	8k	1	0	10	1	55	0	47	6
VP9 to YUV	8k	1	0	10	1	25	0	28	5
YUV to AVC	8k	1	0	10	1	0	58	33	34
YUV to HEVC	8k	1	0	10	1	0	46	34	34
AVC to YUV	4k	1	0	8	1	53	0	239	7
HEVC to YUV	4k	1	0	8	1	47	0	246	6
VP9 to YUV	4k	1	0	8	1	24	0	108	3
AVC to YUV	4k	16	0	8	0	98	0	419	0
HEVC to YUV	4k	16	0	8	0	100	0	447	0
VP9 to YUV	4k	16	0	8	0	99	0	402	0
YUV to AVC	4k	1	0	8	1	0	80	218	23
YUV to HEVC	4k	1	0	8	1	0	77	225	22
YUV to AV1	4k	1	0	8	1	0	83	212	23
YUV to AVC	4k	4	0	8	0	0	96	267	11
YUV to HEVC	4k	4	0	8	0	0	96	283	12
YUV to AV1	4k	4	0	8	0	0	95	244	10
YUV to AVC	4k	8	0	8	0	0	86	243	17
YUV to HEVC	4k	8	0	8	0	0	94	289	14
YUV to AV1	4k	8	0	8	0	0	99	257	9
AVC to AVC	4k	1	1	8	0	69	97	202	6
AVC to HEVC	4k	1	1	8	0	71	96	250	6
AVC to AV1	4k	1	1	8	0	63	97	233	7
HEVC to AVC	4k	1	1	8	0	56	97	202	4
HEVC to HEVC	4k	1	1	8	0	62	96	252	4
HEVC to AV1	4k	1	1	8	0	52	97	233	5
VP9 to AVC	4k	1	1	8	0	24	38	107	2

TYPE	RES	JOB	HW FRAME	Bit	Joint Mode	DEC_LOAD	ENC_LOAD	FPS	CPU
VP9 to HEVC	4k	1	1	8	0	24	36	109	2
VP9 to AV1	4k	1	1	8	0	24	43	109	2
AVC to AVC	4k	4	1	8	0	62	96	223	4
AVC to HEVC	4k	4	1	8	0	68	96	270	4
AVC to AV1	4k	4	1	8	0	54	95	243	4
HEVC to AVC	4k	4	1	8	0	55	95	218	3
HEVC to HEVC	4k	4	1	8	0	58	96	263	3
HEVC to AV1	4k	4	1	8	0	48	96	240	3
VP9 to AVC	4k	4	1	8	0	63	97	218	3
VP9 to HEVC	4k	4	1	8	0	71	96	266	3
VP9 to AV1	4k	4	1	8	0	58	96	241	3
AVC to AVC	4k	8	1	8	0	67	100	197	3
AVC to HEVC	4k	8	1	8	0	76	99	256	3
AVC to AV1	4k	8	1	8	0	65	99	246	3
HEVC to AVC	4k	8	1	8	0	63	99	198	2
HEVC to HEVC	4k	8	1	8	0	65	99	254	3
HEVC to AV1	4k	8	1	8	0	54	99	244	3
VP9 to AVC	4k	8	1	8	0	66	100	202	3
VP9 to HEVC	4k	8	1	8	0	75	100	261	3
VP9 to AV1	4k	8	1	8	0	68	99	249	3
AVC to YUV	4k	1	0	10	1	45	0	176	6
HEVC to YUV	4k	1	0	10	1	40	0	169	6
VP9 to YUV	4k	1	0	10	1	24	0	125	3
AVC to YUV	4k	16	0	10	0	95	0	255	0
HEVC to YUV	4k	16	0	10	0	100	0	253	0
VP9 to YUV	4k	16	0	10	0	99	0	434	0
YUV to AVC	4k	1	0	10	1	0	44	123	24
YUV to HEVC	4k	1	0	10	1	0	42	125	26
YUV to AV1	4k	1	0	10	1	0	46	119	24
YUV to AVC	4k	4	0	10	0	0	49	142	23
YUV to HEVC	4k	4	0	10	0	0	47	142	24
YUV to AV1	4k	4	0	10	0	0	53	134	24
AVC to YUV	1080p	1	0	8	1	41	0	654	10
HEVC to YUV	1080p	1	0	8	1	47	0	696	8
VP9 to YUV	1080p	1	0	8	1	22	0	443	4
AVC to YUV	1080p	40	0	8	0	90	0	1469	0
HEVC to YUV	1080p	40	0	8	0	98	0	1535	0
VP9 to YUV	1080p	40	0	8	0	81	0	1494	0
YUV to AVC	1080p	1	0	8	1	0	55	610	14

TYPE	RES	JOB	HW FRAME	Bit	Joint Mode	DEC_LOAD	ENC_LOAD	FPS	CPU
YUV to HEVC	1080p	1	0	8	1	0	52	609	15
YUV to AV1	1080p	1	0	8	1	0	50	511	13
YUV to AVC	1080p	32	0	8	0	0	99	1134	3
YUV to HEVC	1080p	32	0	8	0	0	95	1150	3
YUV to AV1	1080p	32	0	8	0	0	99	1046	2
AVC to AVC	1080p	1	1	8	0	69	84	897	18
AVC to HEVC	1080p	1	1	8	0	69	83	952	20
AVC to AV1	1080p	1	1	8	0	55	84	807	18
HEVC to AVC	1080p	1	1	8	0	68	87	890	12
HEVC to HEVC	1080p	1	1	8	0	69	84	945	12
HEVC to AV1	1080p	1	1	8	0	57	85	803	11
VP9 to AVC	1080p	1	1	8	0	22	40	447	4
VP9 to HEVC	1080p	1	1	8	0	22	38	446	3
VP9 to AV1	1080p	1	1	8	0	22	44	442	5
AVC to AVC	1080p	32	1	8	0	76	100	891	1
AVC to HEVC	1080p	32	1	8	0	85	99	1001	1
AVC to AV1	1080p	32	1	8	0	76	99	961	1
HEVC to AVC	1080p	32	1	8	0	73	99	923	0
HEVC to HEVC	1080p	32	1	8	0	74	99	1036	0
HEVC to AV1	1080p	32	1	8	0	70	99	971	1
VP9 to AVC	1080p	32	1	8	0	70	99	1013	0
VP9 to HEVC	1080p	32	1	8	0	74	99	1134	0
VP9 to AV1	1080p	32	1	8	0	71	99	1022	0
AVC to YUV	1080p	1	0	10	1	29	0	462	6
HEVC to YUV	1080p	1	0	10	1	27	0	460	5
VP9 to YUV	1080p	1	0	10	1	22	0	453	4
AVC to YUV	1080p	40	0	10	0	67	0	1013	0
HEVC to YUV	1080p	40	0	10	0	76	0	1004	0
VP9 to YUV	1080p	40	0	10	0	81	0	1519	0
YUV to AVC	1080p	1	0	10	1	0	32	356	21
YUV to HEVC	1080p	1	0	10	1	0	30	358	20
YUV to AV1	1080p	1	0	10	1	0	32	319	18
YUV to AVC	1080p	32	0	10	0	0	58	666	4
YUV to HEVC	1080p	32	0	10	0	0	54	677	4
YUV to AV1	1080p	32	0	10	0	0	58	642	7
AVC to YUV	720p	1	0	8	1	42	0	1036	11
HEVC to YUV	720p	1	0	8	1	39	0	1086	7
VP9 to YUV	720p	1	0	8	1	29	0	839	4
AVC to YUV	720p	100	0	8	0	100	0	2514	0

TYPE	RES	JOB	HW FRAME	Bit	Joint Mode	DEC_LOAD	ENC_LOAD	FPS	CPU
HEVC to YUV	720p	100	0	8	0	93	0	2597	0
VP9 to YUV	720p	100	0	8	0	94	0	2430	0
YUV to AVC	720p	1	0	8	1	0	35	876	12
YUV to HEVC	720p	1	0	8	1	0	35	882	14
YUV to AV1	720p	1	0	8	1	0	31	677	11
YUV to AVC	720p	64	0	8	0	0	96	2386	2
YUV to HEVC	720p	64	0	8	0	0	94	2399	1
YUV to AV1	720p	64	0	8	0	0	100	2036	1
AVC to AVC	720p	1	1	8	0	50	49	1211	18
AVC to HEVC	720p	1	1	8	0	50	49	1222	18
AVC to AV1	720p	1	1	8	0	43	49	993	17
HEVC to AVC	720p	1	1	8	0	44	50	1246	11
HEVC to HEVC	720p	1	1	8	0	44	50	1253	11
HEVC to AV1	720p	1	1	8	0	39	48	1010	11
VP9 to AVC	720p	1	1	8	0	28	33	830	7
VP9 to HEVC	720p	1	1	8	0	29	33	832	6
VP9 to AV1	720p	1	1	8	0	29	39	828	7
AVC to AVC	720p	64	1	8	0	100	100	1964	0
AVC to HEVC	720p	64	1	8	0	99	99	2047	0
AVC to AV1	720p	64	1	8	0	95	100	1722	0
HEVC to AVC	720p	64	1	8	0	96	100	1960	0
HEVC to HEVC	720p	64	1	8	0	96	100	2051	0
HEVC to AV1	720p	64	1	8	0	88	100	1723	0
VP9 to AVC	720p	64	1	8	0	100	100	2217	0
VP9 to HEVC	720p	64	1	8	0	100	100	2284	0
VP9 to AV1	720p	64	1	8	0	99	100	1873	0

## 16. T1U – FFmpeg Latency

### 16.1 Encoding

#### 16.1.1 Description

Libxcodec is compiled and installed with parameter `--with-latency-display`

```
$ bash build.sh --with-latency-display
```

YUV frame is read from an input file on ramdisk and then fed into hardware encoder through PCIe.

YUV frame is encoded by hardware encoder.

Encoded bitstream is read out through PCIe and written into an output file.

For each frame, the encoder latency (eLat) value is provided in the output log.

All eLat values are parsed from the output log and the last 50% of frame data before killing ffmpeg instances is used to calculate the Average, Min, Max, and Variance.

The first 50% of frame data are ignored to reach stability while launching multiple jobs.

#### 16.1.2 Command Line

```
ffmpeg -re -loglevel info -f rawvideo -pix_fmt yuv420p -stream_loop  
1000 -s:v <resolution> -i /media/ramdisk/input.yuv -c:v  
<enc>_ni_quadra_enc -enc 0 -xcodec-params gopPresetIdx=9:lowDelay=1 -f  
null -
```

<enc> is the encoder codec. ie h264\_ni\_quadra\_enc, h265\_ni\_quadra\_enc, av1\_ni\_quadra\_enc

<resolution> is resolution of input



## 16.2 FFmpeg Latency Performance Results

TYPE	RESOLUTION	JOBS	ELAT_AVG (ms)	ELAT_MAX (ms)	ELAT_MIN (ms)	ELAT_VAR (ms)
YUV to AVC	8k	1	66.31	67.13	65.11	0.13
YUV to HEVC	8k	1	63.59	69.56	61.93	1.45
YUV to AVC	4k	1	17.77	20.02	17.54	0.06
YUV to HEVC	4k	1	19.17	21.46	18.13	0.11
YUV to AV1	4k	1	25.1	29.43	18.91	0.83
YUV to AVC	4k	4	17.92	22.57	17.04	0.18
YUV to HEVC	4k	4	19.23	22.62	17.53	0.81
YUV to AV1	4k	4	25.18	31.77	18.8	0.95
YUV to AVC	4k	8	20.82	31.24	17.05	20.77
YUV to HEVC	4k	8	25.79	35.79	17.71	27.95
YUV to AV1	4k	8	42.81	53.98	28.36	6.28
YUV to AVC	1080p	1	5.44	6.37	5.26	0.01
YUV to HEVC	1080p	1	5.74	8.88	5.38	0.04
YUV to AV1	1080p	1	7.75	9.46	5.82	0.07
YUV to AVC	1080p	32	8.96	15.24	6.08	2.14
YUV to HEVC	1080p	32	10.42	15.08	6.39	2.39
YUV to AV1	1080p	32	47.14	54.8	39.07	2.25
YUV to AVC	720p	1	3.13	3.81	2.77	0.01
YUV to HEVC	720p	1	3.18	3.56	2.84	0.01
YUV to AV1	720p	1	4.2	4.59	3.37	0.02
YUV to AVC	720p	64	7.9	14.7	4.73	1.01
YUV to HEVC	720p	64	8.44	12.75	5.18	1.05
YUV to AV1	720p	64	45.7	52.9	36.46	3.49

## 17. T1U – Decoder PPU Scaling

### 17.1 Decoding

#### 17.1.1 Description

Bitstream is read from an input file on ramdisk and then fed into hardware decoder through PCIe.

Bitstream is decoded by hardware decoder and scaled to 224x224 with decoder post processing unit.

Decoded YUV is kept on device.

The YUV frame is converted to RGBA format with 2D Engine.

The RGBA frame is read out through PCIe and written into an output file.

#### 17.1.2 Command Line

```
ffmpeg -vsync 0 -c:v <dec>_ni_quadra_dec -dec 0 -xcoder-params  
out=hw:scale0=224x224:multicoreJointMode=<resolution=8k?1:0> -f concat  
-safe 0 -i /media/ramdisk/input.list -vf  
ni_quadra_scale=iw:ih:format=rgba,hwdownload,format=rgba -f null -
```

<dec> is the decoder codec. ie h264\_ni\_quadra\_dec, h265\_ni\_quadra\_dec, vp9\_ni\_quadra\_dec

<resolution> is resolution of input

<resolution> == 8k, multicoreJointMode = 1

### 17.2 Decoder PPU Scaling Performance Results

TYPE	RESOLUTION	JOBS	DEC_LOAD	SCALER_LOAD	FPS	CPU
AVC to RGBA	8k	1	88	0	113	4
HEVC to RGBA	8k	1	89	0	117	5
VP9 to RGBA	8k	1	22	0	31	1
AVC to RGBA	4k	1	21	0	111	6
AVC to RGBA	4k	16	92	2	461	1
HEVC to RGBA	4k	1	22	0	135	8
HEVC to RGBA	4k	16	93	3	551	2
VP9 to RGBA	4k	1	22	0	111	5
VP9 to RGBA	4k	16	95	2	463	1
AVC to RGBA	1080p	40	92	12	1573	1
HEVC to RGBA	1080p	40	93	11	1592	1
VP9 to RGBA	1080p	40	91	15	1999	0
AVC to RGBA	720p	100	93	17	2326	0
HEVC to RGBA	720p	100	87	18	2510	0
VP9 to RGBA	720p	64	90	17	2432	0

## 18. T1U – Streaming Ladder Generation

### 18.1 Transcoding

#### 18.1.1 Description

Bitstream is read from an input file on ramdisk and then fed into hardware decoder through PCIe. Bitstream is decoded by hardware decoder split and scaled to smaller resolutions with decoder post processing unit or 2D Engine.

Decoded YUV frame is kept on device.

The YUV frames are encoded with hardware encoder.

The encoded bitstream is read out through PCIe and written into an output file.

#### 18.1.2 Command line

```
ffmpeg -vsync 0 -c:v <dec>_ni_quadra_dec -dec 0 -xcoder-params  
out=hw:semiplanar0=1:enableOut1=1:semiplanar1=1:scale1=1280x720:enableO  
ut2=1:semiplanar2=1:scale2=960x540 -f concat -safe 0 -i  
/media/ramdisk/input.list -filter_complex  
'[0:v]ni_quadra_split=2:1:2[1080p][1080p_1][720p][540p][540p_1];[540p_1  
]ni_quadra_scale=640x360[360p]' -map [1080p] -xcoder-params  
RcEnable=1:bitrate=3500000 -c:v <enc>_ni_quadra_enc -enc 0 -f null - -  
map [1080p_1] -xcoder-params RcEnable=1:bitrate=1800000 -c:v  
<enc>_ni_quadra_enc -enc 0 -f null - -map [720p] -xcoder-params  
RcEnable=1:bitrate=1000000 -c:v <enc>_ni_quadra_enc -enc 0 -f null - -  
map [540p] -xcoder-params RcEnable=1:bitrate=800000 -c:v  
<enc>_ni_quadra_enc -enc 0 -f null - -map [360p] -xcoder-params  
RcEnable=1:bitrate=500000 -c:v <enc>_ni_quadra_enc -enc 0 -f null -
```

<dec> is the decoder codec. ie h264\_ni\_quadra\_dec, h265\_ni\_quadra\_dec, vp9\_ni\_quadra\_dec

<enc> is the encoder codec. ie h264\_ni\_quadra\_enc, h265\_ni\_quadra\_enc, av1\_ni\_quadra\_enc

Input: 1080p

Output: 1080p, 1080p, 720p(PPU Scale), 540p(PPU Scale), 360p(2D Scale)

### 18.2 Streaming Ladder Generation Performance Results

TYPE	JOBS	DEC_LOAD	ENC_LOAD	SCALER_LOAD	FPS	CPU
AVC to AVC	8	29	90	2	374	3
AVC to HEVC	8	28	89	2	394	3
AVC to AV1	8	21	90	1	343	3
HEVC to AVC	8	31	90	2	376	4
HEVC to HEVC	8	29	89	2	400	4
HEVC to AV1	8	22	90	1	337	4
VP9 to AVC	8	37	91	2	375	3
VP9 to HEVC	8	37	91	2	392	3
VP9 to AV1	8	29	90	1	336	3

## 19. T1U – RGBA Encoding

### 19.1 Encoding

#### 19.1.1 Description

RGBA frame is read from an input file on ramdisk and then fed into hardware encoder through PCIe.

RGBA frame is uploaded and encoded by hardware encoder.

Encoded bitstream is read out through PCIe and written into an output file.

#### 19.1.2 Command line

```
ffmpeg -nostdin -stream_loop -1 -f rawvideo -pix_fmt rgba -s:v  
<resolution> -r 30 -i /media/ramdisk/input.rgb -vf  
"ni_quadra_hwupload=0" -c:v <enc>_ni_quadra_enc -enc 0 -xcoder-params  
intraPeriod=0:RcEnable=1:bitrate=<*>:multicoreJointMode=<*> -f null  
/dev/null
```

<enc> is the encoder codec. eg h264\_ni\_quadra\_enc, h265\_ni\_quadra\_enc, av1\_ni\_quadra\_enc

<num\_jobs> is the number of instances running concurrently

<resolution> is resolution of input

<num\_jobs> == 1, multicoreJointMode = 1

<resolution> == 4k, bitrate = 12000000, framerate = 30

<resolution> == 1080p, bitrate = 3000000, framerate = 30

<resolution> == 720p, bitrate = 1500000, framerate = 30

## 19.2 RGBA Encoding Performance Results

TYPE	RES	JOB	Joint Mode	ENC_LOAD	FPS	CPU
RGBA to AVC	4k	1	1	52	136	78
RGBA to HEVC	4k	1	1	47	141	68
RGBA to AV1	4k	1	1	56	140	69
RGBA to AVC	4k	4	0	74	148	36
RGBA to HEVC	4k	4	0	64	152	41
RGBA to AV1	4k	4	0	66	156	41
RGBA to AVC	4k	8	0	69	157	33
RGBA to HEVC	4k	8	0	62	161	35
RGBA to AV1	4k	8	0	67	165	35
RGBA to AVC	1080p	1	1	33	366	40
RGBA to HEVC	1080p	1	1	31	369	45
RGBA to AV1	1080p	1	1	34	348	45
RGBA to AVC	1080p	16	0	60	582	20
RGBA to HEVC	1080p	16	0	59	592	21
RGBA to AV1	1080p	16	0	62	582	21
RGBA to AVC	1080p	32	0	61	609	11
RGBA to HEVC	1080p	32	0	58	623	13
RGBA to AV1	1080p	32	0	64	610	14
RGBA to AVC	720p	1	1	29	621	55
RGBA to HEVC	720p	1	1	28	622	54
RGBA to AV1	720p	1	1	32	606	61
RGBA to AVC	720p	16	0	48	1138	36
RGBA to HEVC	720p	16	0	48	1121	33
RGBA to AV1	720p	16	0	55	1066	32
RGBA to AVC	720p	32	0	47	1109	31
RGBA to HEVC	720p	32	0	46	1108	31
RGBA to AV1	720p	32	0	58	1115	31

## 20. T1U – Encoding EnableRdoQuant/rdoLevel/lookaheadDepth

### 20.1 Encoding

#### 20.1.1 Description

YUV frame is read from an input file on ramdisk and then fed into hardware encoder through PCIe.

YUV frame is encoded by hardware encoder with a mix of xcoder-params EnableRdoQuant, rdoLevel, and lookaheadDepth.

Encoded bitstream is read out through PCIe and written into an output file.

#### 20.1.2 Command line

```
ffmpeg -nostdin -f concat -safe 0 -i /media/ramdisk/input.list -c:v  
<enc>_ni_quadra_enc -enc 0 -xcoder-params  
intraPeriod=0:RcEnable=1:bitrate=<*>:lookaheadDepth=<*>:EnableRdoQuant=  
<*>:rdoLevel=<*> -f null /dev/null -
```

<enc> is the encoder codec. eg h264\_ni\_quadra\_enc, h265\_ni\_quadra\_enc, av1\_ni\_quadra\_enc

<num\_jobs> is the number of instances running concurrently

<resolution> is resolution of input

<resolution> == 4k, bitrate = 12000000, framerate = 30

<resolution> == 1080p, bitrate = 3000000, framerate = 30

## 20.2 Encoding EnableRdoQuant/rdoLevel/lookaheadDepth Performance Results

TYPE	RES	JOB	lookaheadDepth	enableRdoQuant	rdoLevel	ENC LOAD	FPS	CPU
YUV to AVC	4k	4	0	0	1	96	264	18
YUV to HEVC	4k	4	0	0	1	96	284	12
YUV to AV1	4k	4	0	0	1	97	247	12
YUV to AVC	4k	4	0	0	2	96	264	14
YUV to HEVC	4k	4	0	0	2	98	148	9
YUV to AV1	4k	4	0	0	2	98	120	5
YUV to AVC	4k	4	0	0	3	96	261	14
YUV to HEVC	4k	4	0	0	3	99	88	4
YUV to AV1	4k	4	0	0	3	98	64	5
YUV to AVC	4k	4	0	1	1	98	164	8
YUV to HEVC	4k	4	0	1	1	98	204	9
YUV to AVC	4k	4	0	1	2	98	164	8
YUV to HEVC	4k	4	0	1	2	99	96	4
YUV to AVC	4k	4	0	1	3	97	164	7
YUV to HEVC	4k	4	0	1	3	99	60	3
YUV to AVC	4k	4	4	0	1	99	172	9
YUV to HEVC	4k	4	4	0	1	100	196	9
YUV to AV1	4k	4	4	0	1	99	172	9
YUV to AVC	4k	4	4	0	2	100	172	10
YUV to HEVC	4k	4	4	0	2	100	120	6
YUV to AV1	4k	4	4	0	2	99	100	7
YUV to AVC	4k	4	4	0	3	99	172	9
YUV to HEVC	4k	4	4	0	3	100	76	5
YUV to AV1	4k	4	4	0	3	99	64	3
YUV to AVC	4k	4	4	1	1	100	120	6
YUV to HEVC	4k	4	4	1	1	100	156	9
YUV to AVC	4k	4	4	1	2	100	120	5
YUV to HEVC	4k	4	4	1	2	100	84	5
YUV to AVC	4k	4	4	1	3	99	120	6
YUV to HEVC	4k	4	4	1	3	100	56	3
YUV to AVC	4k	4	16	0	1	99	172	9
YUV to HEVC	4k	4	16	0	1	99	196	10
YUV to AV1	4k	4	16	0	1	99	172	9
YUV to AVC	4k	4	16	0	2	99	172	11
YUV to HEVC	4k	4	16	0	2	100	120	6
YUV to AV1	4k	4	16	0	2	99	100	5

TYPE	RES	JOB	lookaheadDepth	enableRdoQuant	rdoLevel	ENC LOAD	FPS	CPU
YUV to AVC	4k	4	16	0	3	100	172	8
YUV to HEVC	4k	4	16	0	3	99	76	5
YUV to AV1	4k	4	16	0	3	97	64	4
YUV to AVC	4k	4	16	1	1	100	120	8
YUV to HEVC	4k	4	16	1	1	99	156	7
YUV to AVC	4k	4	16	1	2	99	120	6
YUV to HEVC	4k	4	16	1	2	100	80	4
YUV to AVC	4k	4	16	1	3	100	120	5
YUV to HEVC	4k	4	16	1	3	100	56	3
YUV to AVC	4k	4	40	0	1	99	168	9
YUV to HEVC	4k	4	40	0	1	100	196	8
YUV to AV1	4k	4	40	0	1	99	172	9
YUV to AVC	4k	4	40	0	2	100	169	9
YUV to HEVC	4k	4	40	0	2	99	116	5
YUV to AV1	4k	4	40	0	2	99	100	5
YUV to AVC	4k	4	40	0	3	100	169	9
YUV to HEVC	4k	4	40	0	3	100	76	5
YUV to AV1	4k	4	40	0	3	100	60	4
YUV to AVC	4k	4	40	1	1	100	120	7
YUV to HEVC	4k	4	40	1	1	99	152	9
YUV to AVC	4k	4	40	1	2	99	120	6
YUV to HEVC	4k	4	40	1	2	100	80	4
YUV to AVC	4k	4	40	1	3	99	120	6
YUV to HEVC	4k	4	40	1	3	100	54	3
YUV to AVC	1080p	20	0	0	1	99	1100	4
YUV to HEVC	1080p	20	0	0	1	99	1160	3
YUV to AV1	1080p	20	0	0	1	99	1000	3
YUV to AVC	1080p	20	0	0	2	99	1100	4
YUV to HEVC	1080p	20	0	0	2	99	600	2
YUV to AV1	1080p	20	0	0	2	99	481	1
YUV to AVC	1080p	20	0	0	3	99	1100	4
YUV to HEVC	1080p	20	0	0	3	99	360	1
YUV to AV1	1080p	20	0	0	3	100	260	1
YUV to AVC	1080p	20	0	1	1	100	660	2
YUV to HEVC	1080p	20	0	1	1	100	840	3
YUV to AVC	1080p	20	0	1	2	99	660	2
YUV to HEVC	1080p	20	0	1	2	99	380	1
YUV to AVC	1080p	20	0	1	3	99	660	2
YUV to HEVC	1080p	20	0	1	3	99	240	1



TYPE	RES	JOB	lookaheadDepth	enableRdoQuant	rdoLevel	ENC LOAD	FPS	CPU
YUV to AVC	1080p	20	4	0	1	99	613	2
YUV to HEVC	1080p	20	4	0	1	99	720	2
YUV to AV1	1080p	20	4	0	1	99	620	2
YUV to AVC	1080p	20	4	0	2	99	617	2
YUV to HEVC	1080p	20	4	0	2	99	440	1
YUV to AV1	1080p	20	4	0	2	99	369	1
YUV to AVC	1080p	20	4	0	3	99	614	2
YUV to HEVC	1080p	20	4	0	3	99	300	1
YUV to AV1	1080p	20	4	0	3	99	240	1
YUV to AVC	1080p	20	4	1	1	99	440	1
YUV to HEVC	1080p	20	4	1	1	99	564	2
YUV to AVC	1080p	20	4	1	2	99	440	1
YUV to HEVC	1080p	20	4	1	2	100	320	1
YUV to AVC	1080p	20	4	1	3	99	440	1
YUV to HEVC	1080p	20	4	1	3	100	220	1
YUV to AVC	1080p	20	16	0	1	99	603	2
YUV to HEVC	1080p	20	16	0	1	99	710	3
YUV to AV1	1080p	20	16	0	1	99	620	2
YUV to AVC	1080p	20	16	0	2	99	604	2
YUV to HEVC	1080p	20	16	0	2	100	440	1
YUV to AV1	1080p	20	16	0	2	100	361	1
YUV to AVC	1080p	20	16	0	3	99	604	2
YUV to HEVC	1080p	20	16	0	3	99	300	1
YUV to AV1	1080p	20	16	0	3	100	238	1
YUV to AVC	1080p	20	16	1	1	100	440	1
YUV to HEVC	1080p	20	16	1	1	99	560	2
YUV to AVC	1080p	20	16	1	2	100	440	1
YUV to HEVC	1080p	20	16	1	2	100	320	1
YUV to AVC	1080p	20	16	1	3	99	440	1
YUV to HEVC	1080p	20	16	1	3	99	220	1
YUV to AVC	1080p	20	40	0	1	99	600	2
YUV to HEVC	1080p	20	40	0	1	99	700	2
YUV to AV1	1080p	20	40	0	1	99	620	2
YUV to AVC	1080p	20	40	0	2	99	600	2
YUV to HEVC	1080p	20	40	0	2	99	440	1
YUV to AV1	1080p	20	40	0	2	100	360	1
YUV to AVC	1080p	20	40	0	3	99	600	2
YUV to HEVC	1080p	20	40	0	3	99	283	1
YUV to AV1	1080p	20	40	0	3	100	228	1

TYPE	RES	JOB	lookaheadDepth	enableRdoQuant	rdoLevel	ENC LOAD	FPS	CPU
YUV to AVC	1080p	20	40	1	1	99	438	1
YUV to HEVC	1080p	20	40	1	1	99	560	2
YUV to AVC	1080p	20	40	1	2	99	440	1
YUV to HEVC	1080p	20	40	1	2	99	305	1
YUV to AVC	1080p	20	40	1	3	99	440	1
YUV to HEVC	1080p	20	40	1	3	99	201	1
YUV to AVC	720p	40	0	0	1	93	2161	2
YUV to HEVC	720p	40	0	0	1	90	2178	2
YUV to AV1	720p	40	0	0	1	98	1890	2
YUV to AVC	720p	40	0	0	2	90	2159	2
YUV to HEVC	720p	40	0	0	2	99	1320	1
YUV to AV1	720p	40	0	0	2	99	1040	1
YUV to AVC	720p	40	0	0	3	93	2159	2
YUV to HEVC	720p	40	0	0	3	100	800	1
YUV to AV1	720p	40	0	0	3	100	560	1
YUV to AVC	720p	40	0	1	1	99	1480	1
YUV to HEVC	720p	40	0	1	1	99	1800	1
YUV to AVC	720p	40	0	1	2	99	1480	1
YUV to HEVC	720p	40	0	1	2	99	840	1
YUV to AVC	720p	40	0	1	3	99	1480	1
YUV to HEVC	720p	40	0	1	3	100	525	0
YUV to AVC	720p	40	4	0	1	99	1244	1
YUV to HEVC	720p	40	4	0	1	100	1241	1
YUV to AV1	720p	40	4	0	1	100	957	0
YUV to AVC	720p	40	4	0	2	99	1251	1
YUV to HEVC	720p	40	4	0	2	99	920	1
YUV to AV1	720p	40	4	0	2	99	760	1
YUV to AVC	720p	40	4	0	3	99	1246	1
YUV to HEVC	720p	40	4	0	3	99	640	0
YUV to AV1	720p	40	4	0	3	99	480	0
YUV to AVC	720p	40	4	1	1	99	920	1
YUV to HEVC	720p	40	4	1	1	99	1160	1
YUV to AVC	720p	40	4	1	2	99	920	1
YUV to HEVC	720p	40	4	1	2	99	652	0
YUV to AVC	720p	40	4	1	3	99	920	1
YUV to HEVC	720p	40	4	1	3	99	443	0
YUV to AVC	720p	40	16	0	1	99	1240	1
YUV to HEVC	720p	40	16	0	1	100	1268	1
YUV to AV1	720p	40	16	0	1	100	1000	1

TYPE	RES	JOBS	lookaheadDepth	enableRdoQuant	rdoLevel	ENC LOAD	FPS	CPU
YUV to AVC	720p	40	16	0	2	99	1240	1
YUV to HEVC	720p	40	16	0	2	99	920	1
YUV to AV1	720p	40	16	0	2	99	760	0
YUV to AVC	720p	40	16	0	3	99	1240	1
YUV to HEVC	720p	40	16	0	3	99	640	1
YUV to AV1	720p	40	16	0	3	99	480	0
YUV to AVC	720p	40	16	1	1	99	920	1
YUV to HEVC	720p	40	16	1	1	99	1160	1
YUV to AVC	720p	40	16	1	2	99	920	1
YUV to HEVC	720p	40	16	1	2	99	643	0
YUV to AVC	720p	40	16	1	3	99	920	1
YUV to HEVC	720p	40	16	1	3	99	440	0
YUV to AVC	720p	40	40	0	1	99	1240	1
YUV to HEVC	720p	40	40	0	1	100	1225	1
YUV to AV1	720p	40	40	0	1	100	965	1
YUV to AVC	720p	40	40	0	2	99	1239	1
YUV to HEVC	720p	40	40	0	2	99	920	1
YUV to AV1	720p	40	40	0	2	99	760	1
YUV to AVC	720p	40	40	0	3	99	1238	1
YUV to HEVC	720p	40	40	0	3	99	622	0
YUV to AV1	720p	40	40	0	3	99	480	0
YUV to AVC	720p	40	40	1	1	99	916	1
YUV to HEVC	720p	40	40	1	1	99	1136	1
YUV to AVC	720p	40	40	1	2	99	918	1
YUV to HEVC	720p	40	40	1	2	99	640	1
YUV to AVC	720p	40	40	1	3	99	917	1
YUV to HEVC	720p	40	40	1	3	99	440	0

## 21. T1U – Capped CRF

### 21.1 Encoding with lookaheadDepth

#### 21.1.1 Description

YUV frame is read from an input file on ramdisk and fed into hardware encoder through PCIe.

YUV frame is encoded by hardware encoder with a mix of xcoder-params EnableRdoQuant, rdoLevel, lookaheadDepth, CRF, bitrate, and vbvBufferSize.

Encoded bitstream is read out through PCIe and written into an output file.

#### 21.1.2 Command line

```
ffmpeg -nostdin -f concat -safe 0 -i /media/ramdisk/input.list -c:v  
<enc>_ni_quadra_enc -enc 0 -xcoder-params  
intraPeriod=0:vbvBufferSize=1000:bitrate=<*>:lookaheadDepth=<*>:EnableR  
doQuant=<*>:rdoLevel=<*>:crf=<*> -f null /dev/null -
```

<enc> is the encoder codec. eg h264\_ni\_quadra\_enc, h265\_ni\_quadra\_enc, av1\_ni\_quadra\_enc

<num\_jobs> is the number of instances running concurrently

<resolution> is resolution of input

<resolution> == 4k, bitrate = 12000000, framerate = 30

<resolution> == 1080p, bitrate = 3000000, framerate = 30

## 21.2 Capped CRF Encoding with lookaheadDepth Performance Results

TYPE	RES	JOB	lookaheadDepth	enableRdoQuant	rdoLevel	CRF	ENC LOAD	FPS	CPU
YUV to AVC	1080p	20	0	0	1	19	99	619	2
YUV to HEVC	1080p	20	0	0	1	19	99	717	2
YUV to AV1	1080p	20	0	0	1	19	99	621	2
YUV to AVC	1080p	20	0	0	2	19	99	619	2
YUV to HEVC	1080p	20	0	0	2	19	99	440	1
YUV to AV1	1080p	20	0	0	2	19	99	380	1
YUV to AVC	1080p	20	0	0	3	19	99	619	2
YUV to HEVC	1080p	20	0	0	3	19	99	300	1
YUV to AV1	1080p	20	0	0	3	19	100	240	1
YUV to AVC	1080p	20	0	1	1	19	99	440	1
YUV to HEVC	1080p	20	0	1	1	19	99	573	1
YUV to AVC	1080p	20	0	1	2	19	99	440	1
YUV to HEVC	1080p	20	0	1	2	19	99	320	1
YUV to AVC	1080p	20	0	1	3	19	100	440	1
YUV to HEVC	1080p	20	0	1	3	19	100	220	1
YUV to AVC	1080p	20	4	0	1	19	99	614	2
YUV to HEVC	1080p	20	4	0	1	19	99	717	2
YUV to AV1	1080p	20	4	0	1	19	99	620	3
YUV to AVC	1080p	20	4	0	2	19	99	616	2
YUV to HEVC	1080p	20	4	0	2	19	99	440	1
YUV to AV1	1080p	20	4	0	2	19	100	364	1
YUV to AVC	1080p	20	4	0	3	19	99	618	2
YUV to HEVC	1080p	20	4	0	3	19	99	300	1
YUV to AV1	1080p	20	4	0	3	19	100	240	1
YUV to AVC	1080p	20	4	1	1	19	99	440	1
YUV to HEVC	1080p	20	4	1	1	19	99	573	2
YUV to AVC	1080p	20	4	1	2	19	99	440	1
YUV to HEVC	1080p	20	4	1	2	19	100	320	1
YUV to AVC	1080p	20	4	1	3	19	99	440	1
YUV to HEVC	1080p	20	4	1	3	19	99	220	1
YUV to AVC	1080p	20	16	0	1	19	99	603	2
YUV to HEVC	1080p	20	16	0	1	19	99	712	3
YUV to AV1	1080p	20	16	0	1	19	99	620	2
YUV to AVC	1080p	20	16	0	2	19	99	601	2
YUV to HEVC	1080p	20	16	0	2	19	99	440	1
YUV to AV1	1080p	20	16	0	2	19	99	360	1
YUV to AVC	1080p	20	16	0	3	19	99	602	2
YUV to HEVC	1080p	20	16	0	3	19	100	300	1

TYPE	RES	JOB	lookaheadDepth	enableRdoQuant	rdoLevel	CRF	ENC LOAD	FPS	CPU
YUV to AV1	1080p	20	16	0	3	19	99	238	1
YUV to AVC	1080p	20	16	1	1	19	99	440	1
YUV to HEVC	1080p	20	16	1	1	19	99	560	2
YUV to AVC	1080p	20	16	1	2	19	99	440	2
YUV to HEVC	1080p	20	16	1	2	19	99	318	1
YUV to AVC	1080p	20	16	1	3	19	99	440	1
YUV to HEVC	1080p	20	16	1	3	19	100	220	1
YUV to AVC	1080p	20	40	0	1	19	99	600	2
YUV to HEVC	1080p	20	40	0	1	19	99	700	2
YUV to AV1	1080p	20	40	0	1	19	99	619	2
YUV to AVC	1080p	20	40	0	2	19	99	600	2
YUV to HEVC	1080p	20	40	0	2	19	99	440	1
YUV to AV1	1080p	20	40	0	2	19	100	360	1
YUV to AVC	1080p	20	40	0	3	19	99	600	2
YUV to HEVC	1080p	20	40	0	3	19	100	285	1
YUV to AV1	1080p	20	40	0	3	19	99	229	1
YUV to AVC	1080p	20	40	1	1	19	100	440	1
YUV to HEVC	1080p	20	40	1	1	19	99	560	2
YUV to AVC	1080p	20	40	1	2	19	99	440	1
YUV to HEVC	1080p	20	40	1	2	19	99	302	1
YUV to AVC	1080p	20	40	1	3	19	99	440	1
YUV to HEVC	1080p	20	40	1	3	19	99	200	1
YUV to AVC	1080p	20	0	0	1	23	99	618	2
YUV to HEVC	1080p	20	0	0	1	23	99	718	2
YUV to AV1	1080p	20	0	0	1	23	99	625	2
YUV to AVC	1080p	20	0	0	2	23	99	618	2
YUV to HEVC	1080p	20	0	0	2	23	99	440	1
YUV to AV1	1080p	20	0	0	2	23	99	380	1
YUV to AVC	1080p	20	0	0	3	23	99	619	2
YUV to HEVC	1080p	20	0	0	3	23	99	300	1
YUV to AV1	1080p	20	0	0	3	23	99	240	1
YUV to AVC	1080p	20	0	1	1	23	100	440	1
YUV to HEVC	1080p	20	0	1	1	23	99	577	2
YUV to AVC	1080p	20	0	1	2	23	99	440	1
YUV to HEVC	1080p	20	0	1	2	23	99	320	1
YUV to AVC	1080p	20	0	1	3	23	99	440	1
YUV to HEVC	1080p	20	0	1	3	23	99	220	1
YUV to AVC	1080p	20	4	0	1	23	99	618	2
YUV to HEVC	1080p	20	4	0	1	23	99	717	3

TYPE	RES	JOB	lookaheadDepth	enableRdoQuant	rdoLevel	CRF	ENC LOAD	FPS	CPU
YUV to AV1	1080p	20	4	0	1	23	99	620	2
YUV to AVC	1080p	20	4	0	2	23	99	617	2
YUV to HEVC	1080p	20	4	0	2	23	99	440	2
YUV to AV1	1080p	20	4	0	2	23	99	363	1
YUV to AVC	1080p	20	4	0	3	23	99	614	2
YUV to HEVC	1080p	20	4	0	3	23	99	300	1
YUV to AV1	1080p	20	4	0	3	23	100	240	1
YUV to AVC	1080p	20	4	1	1	23	100	440	1
YUV to HEVC	1080p	20	4	1	1	23	99	569	2
YUV to AVC	1080p	20	4	1	2	23	99	440	1
YUV to HEVC	1080p	20	4	1	2	23	99	320	1
YUV to AVC	1080p	20	4	1	3	23	99	440	1
YUV to HEVC	1080p	20	4	1	3	23	99	220	1
YUV to AVC	1080p	20	16	0	1	23	99	604	2
YUV to HEVC	1080p	20	16	0	1	23	99	715	2
YUV to AV1	1080p	20	16	0	1	23	99	620	2
YUV to AVC	1080p	20	16	0	2	23	99	603	2
YUV to HEVC	1080p	20	16	0	2	23	99	440	1
YUV to AV1	1080p	20	16	0	2	23	99	362	1
YUV to AVC	1080p	20	16	0	3	23	99	601	2
YUV to HEVC	1080p	20	16	0	3	23	99	300	1
YUV to AV1	1080p	20	16	0	3	23	99	238	1
YUV to AVC	1080p	20	16	1	1	23	99	440	1
YUV to HEVC	1080p	20	16	1	1	23	99	560	2
YUV to AVC	1080p	20	16	1	2	23	99	440	2
YUV to HEVC	1080p	20	16	1	2	23	100	319	1
YUV to AVC	1080p	20	16	1	3	23	99	440	1
YUV to HEVC	1080p	20	16	1	3	23	100	220	1
YUV to AVC	1080p	20	40	0	1	23	99	600	2
YUV to HEVC	1080p	20	40	0	1	23	99	700	2
YUV to AV1	1080p	20	40	0	1	23	99	618	2
YUV to AVC	1080p	20	40	0	2	23	99	600	2
YUV to HEVC	1080p	20	40	0	2	23	100	440	1
YUV to AV1	1080p	20	40	0	2	23	99	360	1
YUV to AVC	1080p	20	40	0	3	23	99	600	2
YUV to HEVC	1080p	20	40	0	3	23	99	288	1
YUV to AV1	1080p	20	40	0	3	23	99	229	1
YUV to AVC	1080p	20	40	1	1	23	99	440	1
YUV to HEVC	1080p	20	40	1	1	23	99	560	2

TYPE	RES	JOB	lookaheadDepth	enableRdoQuant	rdoLevel	CRF	ENC LOAD	FPS	CPU
YUV to AVC	1080p	20	40	1	2	23	99	440	1
YUV to HEVC	1080p	20	40	1	2	23	99	300	1
YUV to AVC	1080p	20	40	1	3	23	99	439	1
YUV to HEVC	1080p	20	40	1	3	23	99	203	1
YUV to AVC	1080p	20	0	0	1	27	99	620	2
YUV to HEVC	1080p	20	0	0	1	27	99	719	3
YUV to AV1	1080p	20	0	0	1	27	99	626	2
YUV to AVC	1080p	20	0	0	2	27	99	620	2
YUV to HEVC	1080p	20	0	0	2	27	99	440	1
YUV to AV1	1080p	20	0	0	2	27	99	380	1
YUV to AVC	1080p	20	0	0	3	27	99	620	2
YUV to HEVC	1080p	20	0	0	3	27	100	300	1
YUV to AV1	1080p	20	0	0	3	27	99	240	1
YUV to AVC	1080p	20	0	1	1	27	99	440	1
YUV to HEVC	1080p	20	0	1	1	27	99	570	1
YUV to AVC	1080p	20	0	1	2	27	99	440	1
YUV to HEVC	1080p	20	0	1	2	27	99	320	1
YUV to AVC	1080p	20	0	1	3	27	99	440	1
YUV to HEVC	1080p	20	0	1	3	27	99	220	1
YUV to AVC	1080p	20	4	0	1	27	99	617	2
YUV to HEVC	1080p	20	4	0	1	27	99	718	3
YUV to AV1	1080p	20	4	0	1	27	99	620	2
YUV to AVC	1080p	20	4	0	2	27	99	615	2
YUV to HEVC	1080p	20	4	0	2	27	99	441	1
YUV to AV1	1080p	20	4	0	2	27	99	363	1
YUV to AVC	1080p	20	4	0	3	27	99	612	2
YUV to HEVC	1080p	20	4	0	3	27	100	300	1
YUV to AV1	1080p	20	4	0	3	27	100	240	1
YUV to AVC	1080p	20	4	1	1	27	99	440	1
YUV to HEVC	1080p	20	4	1	1	27	99	573	2
YUV to AVC	1080p	20	4	1	2	27	99	440	1
YUV to HEVC	1080p	20	4	1	2	27	99	320	1
YUV to AVC	1080p	20	4	1	3	27	99	440	1
YUV to HEVC	1080p	20	4	1	3	27	99	220	1
YUV to AVC	1080p	20	16	0	1	27	99	605	2
YUV to HEVC	1080p	20	16	0	1	27	99	705	2
YUV to AV1	1080p	20	16	0	1	27	99	621	2
YUV to AVC	1080p	20	16	0	2	27	99	601	2
YUV to HEVC	1080p	20	16	0	2	27	99	440	1



TYPE	RES	JOBS	lookaheadDepth	enableRdoQuant	rdoLevel	CRF	ENC LOAD	FPS	CPU
YUV to AV1	1080p	20	16	0	2	27	99	360	2
YUV to AVC	1080p	20	16	0	3	27	99	606	2
YUV to HEVC	1080p	20	16	0	3	27	99	299	1
YUV to AV1	1080p	20	16	0	3	27	99	239	1
YUV to AVC	1080p	20	16	1	1	27	100	440	2
YUV to HEVC	1080p	20	16	1	1	27	99	560	1
YUV to AVC	1080p	20	16	1	2	27	99	440	1
YUV to HEVC	1080p	20	16	1	2	27	100	320	1
YUV to AVC	1080p	20	16	1	3	27	99	440	1
YUV to HEVC	1080p	20	16	1	3	27	99	220	1
YUV to AVC	1080p	20	40	0	1	27	99	600	1
YUV to HEVC	1080p	20	40	0	1	27	99	700	2
YUV to AV1	1080p	20	40	0	1	27	99	620	2
YUV to AVC	1080p	20	40	0	2	27	99	600	2
YUV to HEVC	1080p	20	40	0	2	27	100	440	1
YUV to AV1	1080p	20	40	0	2	27	100	360	1
YUV to AVC	1080p	20	40	0	3	27	99	600	2
YUV to HEVC	1080p	20	40	0	3	27	99	283	1
YUV to AV1	1080p	20	40	0	3	27	99	231	1
YUV to AVC	1080p	20	40	1	1	27	99	440	2
YUV to HEVC	1080p	20	40	1	1	27	100	560	2
YUV to AVC	1080p	20	40	1	2	27	100	440	2
YUV to HEVC	1080p	20	40	1	2	27	100	300	1
YUV to AVC	1080p	20	40	1	3	27	99	440	1
YUV to HEVC	1080p	20	40	1	3	27	100	202	1

## 22. T1U – Inplace Overlay

### 22.1 Transcoding

#### 22.1.1 Description

A bitstream is read from an input file on ramdisk and then fed into the hardware decoder through PCIe. The bitstream is decoded by the hardware decoder. The decoded YUV frame is kept on the device.

An RGBA image is also uploaded to the device and overlayed onto the video stream via the 2D Engine. The overlayed YUV frames are encoded with the hardware encoder. The encoded bitstream is then read out through PCIe and written into an output file.

#### 22.1.2 Command line

```
ffmpeg -c:v <dec>_ni_quadra_dec -dec 0 -xcoder-params "out=hw" -f
concat -safe 0 -i /media/ramdisk/input.list -f rawvideo -s:v 128x128 -
pix_fmt rgba -i /media/ramdisk/img.rgb -filter_complex
"[1:v]format=rgba,ni_quadra_hwupload=0[a];[0:v][a]ni_quadra_overlay=0:0
:alpha=1:inplace=1[b]" -c:a copy -map "[b]" -c:v <enc>_ni_quadra_enc -
enc 0 -xcoder-params "RcEnable=1:bitrate=2000000" -f null -
```

<dec> is the decoder codec. ie h264\_ni\_quadra\_dec, h265\_ni\_quadra\_dec, vp9\_ni\_quadra\_dec

<enc> is the encoder codec. ie h264\_ni\_quadra\_enc, h265\_ni\_quadra\_enc, av1\_ni\_quadra\_enc

Input Video: 1080p

Input Image: 128x128

## 22.2 Inplace Overlay Performance Results

TYPE	JOB5	FPS	CPU	DEC_LOAD	ENC_LOAD	SCALER_LOAD
AVC to AVC	1	240	8	13	20	4
AVC to HEVC	1	256	9	14	19	4
AVC to AV1	1	219	9	12	19	3
HEVC to AVC	1	240	12	13	19	4
HEVC to HEVC	1	255	12	14	19	4
HEVC to AV1	1	218	12	12	19	3
VP9 to AVC	1	240	9	17	19	4
VP9 to HEVC	1	254	9	19	19	4
VP9 to AV1	1	218	8	16	19	3
AVC to AVC	16	941	2	70	91	23
AVC to HEVC	16	1024	3	77	91	26
AVC to AV1	16	960	2	67	90	23
HEVC to AVC	16	960	3	71	92	23
HEVC to HEVC	16	1056	3	75	90	25
HEVC to AV1	16	979	3	67	92	23
VP9 to AVC	16	944	2	85	91	23
VP9 to HEVC	16	1023	2	90	88	24
VP9 to AV1	16	976	3	84	91	23
AVC to AVC	32	864	1	74	94	23
AVC to HEVC	32	960	1	80	91	26
AVC to AV1	32	928	1	75	93	25
HEVC to AVC	32	896	1	72	92	24
HEVC to HEVC	32	992	1	79	91	26
HEVC to AV1	32	960	1	71	92	24
VP9 to AVC	32	896	1	86	93	23
VP9 to HEVC	32	998	1	89	88	25
VP9 to AV1	32	961	1	86	92	24

## Appendix A: GStreamer XStack Command

Example of a 4x8 grid with 32 inputs and an output resolution of 1920x1080 with each cell 480x135.

[illegible]

```
xcoder-params='out=hw' ! xstack. multifilesrc  
location=/media/ramdisk/input.h265 loop=true ! h265parse ! niquadrah265dec  
xcoder-params='out=hw' ! xstack. multifilesrc  
location=/media/ramdisk/input.h265 loop=true ! h265parse ! niquadrah265dec  
xcoder-params='out=hw' ! xstack. multifilesrc  
location=/media/ramdisk/input.h265 loop=true ! h265parse ! niquadrah265dec  
xcoder-params='out=hw' ! xstack. multifilesrc  
location=/media/ramdisk/input.h265 loop=true ! h265parse ! niquadrah265dec  
xcoder-params='out=hw' ! xstack. multifilesrc  
location=/media/ramdisk/input.h265 loop=true ! h265parse ! niquadrah265dec  
xcoder-params='out=hw' ! xstack. multifilesrc  
location=/media/ramdisk/input.h265 loop=true ! h265parse ! niquadrah265dec  
xcoder-params='out=hw' ! xstack. multifilesrc  
location=/media/ramdisk/input.h265 loop=true ! h265parse ! niquadrah265dec  
xcoder-params='out=hw' ! xstack. multifilesrc  
location=/media/ramdisk/input.h265 loop=true ! h265parse ! niquadrah265dec  
xcoder-params='out=hw' ! xstack. multifilesrc  
location=/media/ramdisk/input.h265 loop=true ! h265parse ! niquadrah265dec  
xcoder-params='out=hw' ! xstack. multifilesrc  
location=/media/ramdisk/input.h265 loop=true ! h265parse ! niquadrah265dec  
xcoder-params='out=hw' ! xstack. multifilesrc  
location=/media/ramdisk/input.h265 loop=true ! h265parse ! niquadrah265dec  
xcoder-params='out=hw' ! xstack. multifilesrc  
location=/media/ramdisk/input.h265 loop=true ! h265parse ! niquadrah265dec  
xcoder-params='out=hw' ! xstack. multifilesrc  
location=/media/ramdisk/input.h265 loop=true ! h265parse ! niquadrah265dec  
xcoder-params='out=hw' ! xstack. multifilesrc  
location=/media/ramdisk/input.h265 loop=true ! h265parse ! niquadrah265dec  
xcoder-params='out=hw' ! xstack.  
location=/media/ramdisk/input.h265 loop=true ! h265parse ! niquadrah265dec
```

## Appendix B: 7x7 Grid Layout

Size of each cell in a 7x7 grid with 49 outputs. Overall output resolution is 1080p

	274x154		274x154		274x154		274x154		274x154		274x154		276x154	
	274x154		274x154		274x154		274x154		274x154		274x154		276x154	
	274x154		274x154		274x154		274x154		274x154		274x154		276x154	
	274x154		274x154		274x154		274x154		274x154		274x154		276x154	
	274x154		274x154		274x154		274x154		274x154		274x154		276x154	
	274x154		274x154		274x154		274x154		274x154		274x154		276x154	
	274x156		274x156		274x156		274x156		274x156		274x156		276x156	

## Appendix C: GStreamer Ladder Command

### Example of single input with 64 outputs

```
gst-launch-1.0 multiqueue sync-by-running-time=TRUE max-size-bytes=0 max-
size-buffers=0 max-size-time=0 name=mq multifilesrc
location=/media/ramdisk/input.h264 loop=true ! h264parse ! niquadrah264dec !
tee name=t ! mq. mq. ! niquadrah265enc ! fpsdisplaysink video-sink=fakesink
sync=false -v t. ! mq. mq. ! niquadrah265enc ! fpsdisplaysink video-
sink=fakesink sync=false -v t. ! mq. mq. ! niquadrah265enc ! fpsdisplaysink
video-sink=fakesink sync=false -v t. ! mq. mq. ! niquadrah265enc !
fpsdisplaysink video-sink=fakesink sync=false -v t. ! mq. mq. !
niquadrah265enc ! fpsdisplaysink video-sink=fakesink sync=false -v t. ! mq.
mq. ! niquadrah265enc ! fpsdisplaysink video-sink=fakesink sync=false -v t.
! mq. mq. ! niquadrah265enc ! fpsdisplaysink video-sink=fakesink sync=false -
v t. ! mq. mq. ! niquadrah265enc ! fpsdisplaysink video-sink=fakesink
sync=false -v t. ! mq. mq. ! niquadrah265enc ! fpsdisplaysink video-
sink=fakesink sync=false -v t. ! mq. mq. ! niquadrah265enc ! fpsdisplaysink
video-sink=fakesink sync=false -v t. ! mq. mq. ! niquadrah265enc !
fpsdisplaysink video-sink=fakesink sync=false -v t. ! mq. mq. !
niquadrah265enc ! fpsdisplaysink video-sink=fakesink sync=false -v t. ! mq.
mq. ! niquadrah265enc ! fpsdisplaysink video-sink=fakesink sync=false -v t.
! mq. mq. ! niquadrah265enc ! fpsdisplaysink video-sink=fakesink sync=false -
v t. ! mq. mq. ! niquadrah265enc ! fpsdisplaysink video-sink=fakesink
sync=false -v t. ! mq. mq. ! niquadrah265enc ! fpsdisplaysink video-
sink=fakesink sync=false -v t. ! mq. mq. ! niquadrah265enc ! fpsdisplaysink
video-sink=fakesink sync=false -v t. ! mq. mq. ! niquadrah265enc !
fpsdisplaysink video-sink=fakesink sync=false -v t. ! mq. mq. !
niquadrah265enc ! fpsdisplaysink video-sink=fakesink sync=false -v t. ! mq.
mq. ! niquadrah265enc ! fpsdisplaysink video-sink=fakesink sync=false -v t.
! mq. mq. ! niquadrah265enc ! fpsdisplaysink video-sink=fakesink sync=false -
v t. ! mq. mq. ! niquadrah265enc ! fpsdisplaysink video-sink=fakesink
sync=false -v t. ! mq. mq. ! niquadrah265enc ! fpsdisplaysink video-
sink=fakesink sync=false -v t. ! mq. mq. ! niquadrah265enc ! fpsdisplaysink
video-sink=fakesink sync=false -v t. ! mq. mq. ! niquadrah265enc !
fpsdisplaysink video-sink=fakesink sync=false -v t. ! mq. mq. !
niquadrah265enc ! fpsdisplaysink video-sink=fakesink sync=false -v t. ! mq.
mq. ! niquadrah265enc ! fpsdisplaysink video-sink=fakesink sync=false -v t.
! mq. mq. ! niquadrah265enc ! fpsdisplaysink video-sink=fakesink sync=false -
v t. ! mq. mq. ! niquadrah265enc ! fpsdisplaysink video-sink=fakesink
sync=false -v t. ! mq. mq. ! niquadrah265enc ! fpsdisplaysink video-
sink=fakesink sync=false -v t. ! mq. mq. ! niquadrah265enc ! fpsdisplaysink
video-sink=fakesink sync=false -v t. ! mq. mq. ! niquadrah265enc !
fpsdisplaysink video-sink=fakesink sync=false -v t. ! mq. mq. !
niquadrah265enc ! fpsdisplaysink video-sink=fakesink sync=false -v t. ! mq.
```

mq. ! niquadrah265enc ! fpsdisplaysink video-sink=fakesink sync=false -v t.  
! mq. mq. ! niquadrah265enc ! fpsdisplaysink video-sink=fakesink sync=false -  
v t. ! mq. mq. ! niquadrah265enc ! fpsdisplaysink video-sink=fakesink  
sync=false -v t. ! mq. mq. ! niquadrah265enc ! fpsdisplaysink video-  
sink=fakesink sync=false -v t. ! mq. mq. ! niquadrah265enc ! fpsdisplaysink  
video-sink=fakesink sync=false -v t. ! mq. mq. ! niquadrah265enc !  
fpsdisplaysink video-sink=fakesink sync=false -v t. ! mq. mq. !  
niquadrah265enc ! fpsdisplaysink video-sink=fakesink sync=false -v t. ! mq.  
mq. ! niquadrah265enc ! fpsdisplaysink video-sink=fakesink sync=false -v t.  
! mq. mq. ! niquadrah265enc ! fpsdisplaysink video-sink=fakesink sync=false -  
v t. ! mq. mq. ! niquadrah265enc ! fpsdisplaysink video-sink=fakesink  
sync=false -v t. ! mq. mq. ! niquadrah265enc ! fpsdisplaysink video-  
sink=fakesink sync=false -v t. ! mq. mq. ! niquadrah265enc ! fpsdisplaysink  
video-sink=fakesink sync=false -v t. ! mq. mq. ! niquadrah265enc !  
fpsdisplaysink video-sink=fakesink sync=false -v t. ! mq. mq. !  
niquadrah265enc ! fpsdisplaysink video-sink=fakesink sync=false -v t. ! mq.  
mq. ! niquadrah265enc ! fpsdisplaysink video-sink=fakesink sync=false -v t.  
! mq. mq. ! niquadrah265enc ! fpsdisplaysink video-sink=fakesink sync=false -  
v t. ! mq. mq. ! niquadrah265enc ! fpsdisplaysink video-sink=fakesink  
sync=false -v