Ambarella

SDK6 UG MW UnitTest

Version 2.3

February 24, 2016

Confidentiality Notice:

The contents of this document are proprietary and confidential information of Ambarella, Inc.

The material in this document is for information only. Ambarella assumes no responsibility for errors or omissions and reserves the right to change, without notice, product specifications, operating characteristics, packaging, ordering, etc. Ambarella assumes no liability for damage resulting from the use of information contained in this document. All brands, product names and company names are trademarks of their respective owners.

For complete Ambarella contact information, please visit website: http://www.ambarella.com/.

I Contents

II Preface

This document provides technical details using a set of consistent typographical conventions to help the user differentiate key concepts at a glance.

Conventions include:

Example	Description
AmbaGuiGen, **DirectUSB** **Save, File > Save** **Power, Reset, Home**	Software names GUI commands and command sequences Computer / Hardware buttons
Flash_IO_control **da**, **status**, **enable**	Register names and register fields. For example, **Flash_IO_control** is the register for global control of Flash I/O, and bit 17 (**da)** is used for DMA acknowledgement.
GPIO81, **CLK_AU**	Hardware external pins
VIL, VIH, VOL, VOH	Hardware pin parameters
INT_O, RXDATA_I	Hardware pin signals
amb_performance_t ***amb_operating_mode_t*** **amb_set_operating_mode()**	API details (e.g., functions, structures, and type definitions)
`/usr/local/bin` `success = amb_set_operating_mode (amb_XXX_base_address and operating_mode)`	User entries into software dialogues and GUI windows File names and paths Command line scripting and Code

Table II-1. Typographical Conventions for Technical Documents.

Additional Ambarella typographical conventions include:

- Acronyms are given in UPPER CASE using the default font (e.g., AHB, ARM11 and DDRIO).
- Names of Ambarella documents and publicly available standards, specifications, and databooks appear in *italic* type.

1 Overview

1.1 Overview: Introduction

This document describes the procedures for running the Middleware unit tests for SDK6.

2 Hardware Platform

2.1 Hardware Platform: Overview

Platform	UnitTests
A9 Ginkgo EVK BSP name: a9evk_Ginkgo Sensor: Sony IMX117 sensor LCD: Wintek WDF9648W	All except MotorVu 360
A9-B5 BUB BSP name: b5bub Sensor: OmniVision OV10640 on B5F daughterboard x 4	MotorVu 360
A9S Cheetah EVK BSP name: A9SEVK_CHEETAH Sensor: Sony IMX377 sensor	All except MotorVu 360
A12 Dragonfly EVK BSP name: A12EVK_Dragonfly Sensor: OmniVision OV4689	All except MotorVu 360

Table 2-1. Platform.

3 Feature List

3.1 Feature List: Overview

This chapter provides the feature lists. It includes the following:

- (Section 3.2) Feature List: Camera Function Flow Control
- (Section 3.3) Feature List: Data Flow Control
- (Section 3.4) Feature List: Display Control
- (Section 3.5) Feature List: OSD control
- (Section 3.6) Feature List: Beep Sound Service

3.2 Feature List: Camera Function Flow Control

A. Encoding function

1. Liveview
   - Single channel input
   - MotorVu 360 (4-ch input) (B5 only)
   - Dual VIN (Hardware Switchable) (A12 HybridMode only)
   - Optical black level support (Based on sensor capability)

2. Video encoding
   - H.264, MJPEG spec
   - Maximum resolution: 2160P30 (UHD) or 1440P60 (WQHD)
   - Multi stream encoding up to
     - 2160P30 single stream [A9/A9S only]
     - 1440P60 single stream
     - 1080P120 single stream [A9/A9S only]
     - 720P240 single stream
     - 432x240P480 single stream
     - 1080P60 + 1080P30 dual stream
     - 1080P30 + 1080P30 dual stream
     - 1440P50 + 720P25 dual stream
   - Rotation
   - Date/time stamp
   - Time lapsed encoding
   - Runtime frame rate change [A9/A9S only]
   - Picture in View (PIV)

3. Still capture
    - Up to 16MP
    - Endless burst capture
    - Background raw2yuv, yuv2jpeg support [Except A9/A9S HISO, A12]
    - Digital effects [Not supported yet]
    - HDR [Not supported yet]
    - HISO
    - Mechanical shutter [Not supported yet]
    - Burst Capture
    - PES Capture
    - Pre-Capture [Except A12]

4. Audio encoding
    - AAC/PCM/OPUS spec
    - Multiple input (with mixing support)
    - Multiple output

5. 3A statistics
    - CFA statistics
    - RGB statistics
    - Motion vector histogram [Except A12]

6. Digital zoom
    - Initial zoom factor
    - Moving
    - Jumping

B. Decoding function

1. Video decoding
    - Trickplay
        - Forward, backward
        - 1x, 2x, 1/2x…
        - Pause/resume/step
        - Time search
    - Zoom in/out
    - Rotation
    - Playback capture
    - Fading effect

2. Video multichannel decoding
    - Trickplay
        - 1x Forward
        - Pause/resume/step
        - Time search
        - Zoom in/out
        - Rotation

3. Still decoding
    - Up to 16MP
    - Blending

- Zoom in/out
- Rotation
- Luma gain

4. Multiview decoding
   - Motion support
5. Audio decoding
   - AAC/PCM/MP3/OPUS
6. Video transcoding
   - H264 to H264/MJPEG (Decoding 2160P30, encoding up to 1080P30)
   - Frame rate conversion (1, 1/2, 1/4, 1/8)

3.3 Feature List: Data Flow Control

1. Format support
   - Muxer
     MP4, MOV, EXIF
   - Demuxer
     MP4, MOV, EXIF
   - Editor
     Crop2New, Dividing, Merging, Recovery
2. Stream support
   - File stream
   - External stream
3. Cached file system support
4. File naming support
   - DCF
5. Automotive camera features
   - Event recording
   - Emergency recording

3.4 Feature List: Display Control

1. LCD, CVBS, HDMI (fixed at 1920x1080)
2. Rotation, mirror
3. Layout management

3.5 Feature List: OSD control

1. Object management
2. Primitive drawing
3. BMP, TTF fonts

3.6 Feature List: Beep Sound Service

Please refer to (Section 6.18) Unit Tests: Beep.

4 How to Run the Unit Test

4.1 How to Run the Unit Test: Overview

This chapter provides the steps for running the unit test.

After unzipping the release package, please perform the following tasks (A12 is used as an example):

Step 1: Type ‘make a12_mw_unittest_defconfig’

Step 2: Type ‘make’

Step 3: Burn the file: out\fwprog\bst_bld_sys_dsp_rom_lnx_rfs.elf by Chameleon

Step 4: Reboot the hardware

Step 5: Type “t” to see a list of unit tests

Step 6: In A12, when the elf file contains xxx_lnx_xxx, the user shall reboot the system after first booting up when the console shows “Linux suspend AMBA_LINK_HIBER_TO_DISK is done”, this message depicts that Linux entering the hibernation mode needs to be resumed.

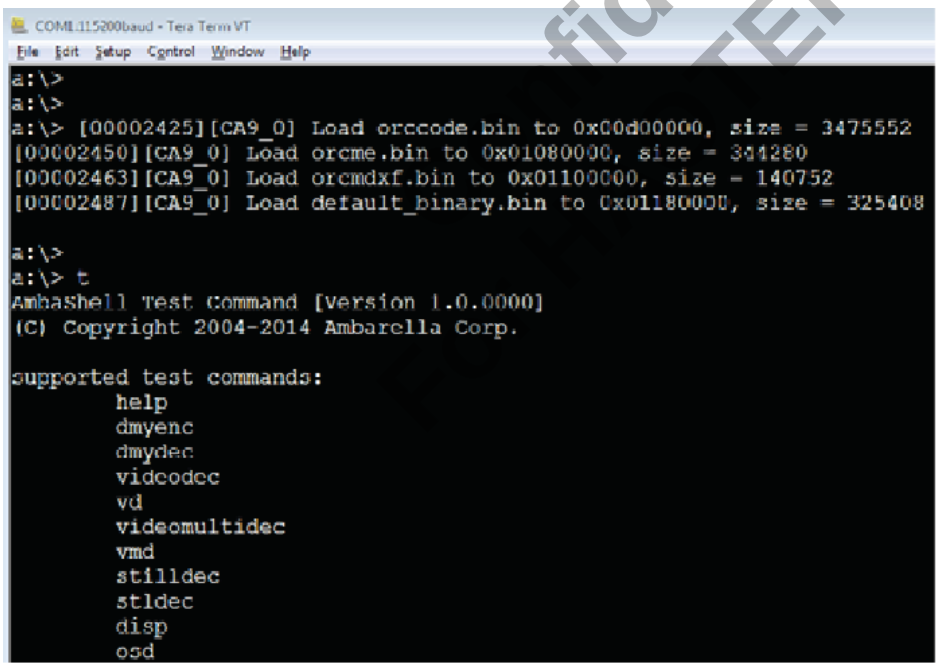

Figure 4-1. Type “t” to see a List of Unit Tests.

For detailed build steps, please refer to the document *Ambarella SDK6 AN Build Environment*.

5 List of Unit Tests

5.1 List of Unit Tests: Overview

This chapter provides the list of unit tests.

Test item	File name	Description
Audio encoder	AmpUT_AudioEnc.c	Test MW audio encoder
Display window	AmpUT_Display.c	Test MW display module
FIFO	AmpUT_DummyEnc.c	Fetch dummy data from FIFO
FIFO	AmpUT_DummyDec.c	Feed dummy data to FIFO
CFS	AmpUT_CFS.c	Test Cached file system API
Exif muxer	AmpUT_ExifMux.c	Test Exif muxer
Exif Demuxer	AmpUT_ExifDmx.c	Test Exif demuxer
Ext Muxer	AmpUT_ExtMux.c	Test External muxer
Mp4 Demuxer	AmpUT_Mp4Dmx.c	Test MP4 demuxer
	AmpUT_Mp4DmxAV.c	Test MP4 demuxer
Mp4 Muxer	AmpUT_Mp4Mux.c	Test MP4 muxer
	AmpUT_Mp4Mux2s.c	Test MP4 muxer
	AmpUT_Mp4Mux4s.c	Test MP4 muxer
	AmpUT_Mp4MuxAV.c	Test MP4 muxer
	AmpUT_Mp4MuxDual.c	Test MP4 muxer
Mov Demuxer	AmpUT_MovDmx.c	Test MOV demuxer
Mov Muxer	AmpUT_MovMux.c	Test MOV muxer
OSD	AmpUT_OSD.c	Test MW OSD API
Still decoder	AmpUT_StillDec.c	Test MW still decoder
Still encoder	AmpUT_StillEnc.c	Test MW still encoder
Video decoder	AmpUT_VideoDec.c	Test MW video decoder
	AmpUT_VideoMultiDec.c	Test MW video multi-channel decoder
Video encoder	AmpUT_VideoEnc.c	Test MW video encoder
MotorVu360 (B5 only)	AmpUT_MotorVu360.c	Test MotorVu 360
Audio decoder	AmpUT_AudioDec.c	Test audio decoder
Beep sound	AmpUT_Beep.c	Test beep sound
Digital zoom	AmpUT_VideoEnc.c	Test digital zoom
Transcoding	AmpUT_Transcode.c	Test transcoding
Video tuning	AmpUT_VideoTuning.c	Video tuning
Editor	AmpUT_Editor.c	Editor
Event recording	AmpUT_EventRecord.c	Event recording
	AmpUT_EventRecord_O.c	
	AmpUT_EventRecord_N.c	
Emergency recording	AmpUT_Emergency.c	Emergency recording
Linux virtual FIFO	AmpUT_NetFifo.c	Test Linux video streaming

Test item	File name	Description
Remote API	AmpUT_NetCtrl.c	Test Remote Control API
Linux Event Notifier	AmpUT_NetEventNotifier.c	Test event notification for Linux
AV Encode	AmpUT_AVEnc.c	Test Video and Audio enable at the same time
YUV Encode	AmpUT_YUVEnc.c	Test YUV input
DualVin Encode	AmpUT_DualVinEnc.c	Test dual VIN

Table 5-1. Unit Tests.

6 Unit Tests

6.1 Unit Tests: Overview

This chapter provides the unit tests. It includes unit tests for the video encoder, MotorVu 360, video decoder, video multi-channel decode, still encoder, still decoder, MP4 muxer, MP4 demuxer, EXIF muxer, EXIF demuxer, CFS, audio encoder, OSD, audio decoder, MOV muxer, MOV demuxer, Beep, transcode, video tuning, DCF, event recording, event recording – non-overlapped, event Recording – overlapped, emergency recording, editor and external muxer. Unit tests are used to verify and check different capabilities of the SDK6 MW using the command line interface. A unit test generally tests a single component without dependencies on the system.

Display is not a standalone unit test, but we still provide a test command “disp” to change display settings.

t disp cfg [channel] [enable] [mode]

channel: “dchan” or 0 means dchan, “fchan” or 1 means fchan
enable: 0 means disable, 1 means enable
mode: for dchan, 0 means WDF9648W, 1 means T20P52, 2 means T27P05, 3 means T30P61

For example, the user can type the below commands to ask the system to use T30P61 LCD in video encode unit test:

Step 1: t disp init
Step 2: t disp cfg dchan 1 3

Please note that the above commands must be executed before the video encode unit test is initiated.

6.2 Unit Tests: Video Encoder

The video encoder unit test contains following tests:

1. Liveview
2. Runtime sensor mode switch
3. Single stream encode
4. Dual stream encode
5. Bitrate control (runtime and static)
6. Picture in Video (PIV)
7. Bitstream specification selection
8. Video encode rotation
9. Time lapsed encoding
10. Dual HD encoding

11. Runtime framerate change (A9/A9S only)
12. Digital zoom (initial and runtime)
13. ExpressMode (SDK 6.2.001 or later supports it)
14. Hybrid Mode (SDK 6.2.002 or later supports it)

6.2.1 Video Encoder: Command List

On A9S, the resolutions below are run in Pipeline Mode 5, which does not support the pause, the time lapse encoding, the blending, and the runtime frame rate change:
(a) Frame rate > 60
(b) 1920x1080 P60 + 1920x1080 P30/P60
(c) 2704x1520 P60
On A12, Pause/Resume is not supported.

1. To initialize the video encoder unit test after booting

   t videoenc init [sensorID]
   sensorID: 0 – OV4689
   1 - IMX117
   2 - OV2710 (A9+B5)
   3 - OV10823
   4 - IMX206
   5 - AR0330_PARALLEL
   6 - AR0230
   7 - OV9750
   8 – MN34120
   9 - B5_OV4689
   10 - IMX290
   11 - MN34222
   12 - IMX377
   13 - IMX317 (A9S only, since 6.3.001u1)
   14 – B5_OV9750
   15 – B5_AR0230
   16 – B5_OV10640_R1E (A9S only)
   17 – YuvRawInput (A12 DualVin only)
   18 – AR0237
   19 - IMX277
   20 - IMX172
   21 – IMX183
   22 – IMX078
   23 – OV2718
   24 – TI5150
   25 – NVP6114A
   26 – IMX214
   27 – B5_IMX206
   28 – B5_AR0330

2. To enter Liveview

   t videoenc liveviewstart [modeID]
   modeID: Main/primary stream setting
   Setting for OV4689 :
   0 - 1920x1080@30fps

1 - 1920x1080@60fps
3 - 1920x1080@30fps HDR (on A12, HybridMode only)
5 - 1280x720@30fps
6 - 1280x720@60fps
11 - 2560x1440@30fps
12 - 2560x1440@60fps (on A12, ExpressMode only)
38 - 2560x1440@30fps HDR (on A12, HybridMode only)

For IMX117:

[A9/A9S]
0 - 1920x1080@30/25fps
1 - 1920x1080@60/50fps
2 - 1920x1080@120/100fps
5 - 1280x720@30/25fps
6 - 1280x720@60/50fps
7 - 1280x720@120/100fps
8 - 1280x720@240/200fps (On A9S, must disable TV)
11 - 2560x1440@30/25fps
12 - 2560x1440@60/50fps (On A9S, must disable TV)
13 - 3840x2160@30/25fps (On A9S, must disable TV)
15 - 1920x1440@30/25fps
20 - 1280x960@30/25fps
21 - 1280x960@60/50fps
22 - 1280x960@120/100fps
26 - 2560x1920@30/25fps (On A9S, not supported if LCD_T30P61 is used)
41 - 848x480@30/25fps
43 - 2704x1520@60fps
44 - 2688x1520@30/25fps

[A12]
0 - 1920x1080@30fps
1 - 1920x1080@60fps
2 - 1920x1080@120fps
5 - 1280x720@30fps
6 - 1280x720@60fps
7 - 1280x720@120fps
8 - 1280x720@240fps (on A12, ExpressMode NTSC mode can only run single stream)
11 - 2560x1440@30fps
12 - 2560x1440@60fps
13 – 3840X2160@30fps (On A12 HybridMode 4K mode)
15 - 1920x1440@30fps
26 - 2560x1920@30fps

For OV2710:

0 - 1920x1080@30fps
6 - 1280x720@60fps

For OV10823:

0 - 1920x1080@30fps

For IMX206 :

0 - 1920x1080@30fps
1 - 1920x1080@60fps
2 - 1920x1080@120fps (on A12, only ExpressMode PAL mode)
5 - 1280x720@30fps
6 - 1280x720@60fps
7 - 1280x720@120fps
8 - 1280x720@240fps
15 - 1920x1440@30fps
20 - 1280x960@30fps (A9 only)
21 - 1280x960@60fps (A9 only)
22 - 1280x960@120fps (A9 only)

For AR0330_PARALLEL:

0 - 1920x1080@30fps
5 - 1280x720@30fps
6 - 1280x720@60fps
20 - 1280x960@30fps (A9 only)

For AR0230:

0 - 1920x1080@30fps
1 - 1920x1080@60fps
3 - 1920x1080@30fps HDR (on A12, HybridMode only)
5 - 1280x720@30fps
6 - 1280x720@60fps
20 - 1280x960@30fps (A9 only)
21 - 1280x960@60fps (A9 only)

For MN34120:

0 - 1920x1080@30fps
1 - 1920x1080@60fps
5 - 1280x720@30fps
6 - 1280x720@60fps
7 - 1280x720@120fps
45 - 3840x2160@15fps (on A12, HybridMode 4K only)

For IMX290:

0 - 1920x1080@30fps
1 - 1920x1080@60fps
2 - 1920x1080@1200fps (on A12, only ExpressMode PAL mode)
3 - 1920x1080@30fps HDR (on A12, HybridMode only)
4 - 1920x1080@60fps HDR (on A12, HybridMode only)

For MN34229:

0 - 1920x1080@30fps
1 - 1920x1080@60fps
3 - 1920x1080@30fps HDR (on A12, HybridMode only)

For IMX377:

0 - 1920x1080@30/25fps
1 - 1920x1080@60/50fps
2 - 1920x1080@120/100fps
5 - 1280x720@30/25fps
6 - 1280x720@60/50fps
7 - 1280x720@120/100fps
8 - 1280x720@240/200fps (On A9S, must disable TV)

11 - 2560x1440@30/25fps
12 - 2560x1440@60/50fps (On A9S, must disable TV)
13 - 3840x2160@30/25fps (On A9S, must disable TV)
15 - 1920x1440@30fps (A9S only)
20 - 1280x960@30/25fps (A9S only)
21 - 1280x960@60/50fps (A9S only)
26 - 2560x1920@30fps (On A9S only, not supported if LCD_T30P61 is used)
43 - 2704x1520@60/50fps (A9S only)
44 - 2688x1520@30/25fps (A9S only)

For IMX317: (A9S only)
0 - 1920x1080@30/25fps
1 - 1920x1080@60/50fps
2 - 1920x1080@120/100fps
5 - 1280x720@30/25fps
6 - 1280x720@60/50fps
7 - 1280x720@120/100fps
11 - 2560x1440@30/25fps
12 - 2560x1440@60/50fps (On A9S, must disable TV)
13 - 3840x2160@30/25fps (On A9S, must disable TV)
43 - 2704x1520@60/50fps
44 - 2688x1520@30/25fps

For YuvRawInput: (A12 DualVin only)
5 - 1280x720@30/25fps
41 - 848x480@30/25fps

For AR0237:
0 - 1920x1080@30fps
3 - 1920x1080@30fps HDR (on A12, HybridMode only)

For IMX078:
0- 1920x1080@30fps
1 – 1920X1080@60fps
5 - 1280x720@30fps
6 - 1280x720@60fps
7 - 1280x720@120fps
11 – 2560X1440@30fps
12 – 2560X1440@60fps
13 – 3840X2160@30fps

For OV2718:
0- 1920x1080@30fps
30 - 1920x1080@ CFA HDR 30fps

For TI5150:
48 – 720x480@I60fps
49 - 720x576@I50fps

For NVP6114A:
5 - 1280x720@30fps

For IMX214:
0 - 1920x1080@30/25fps
1 - 1920x1080@60/50fps
5 - 1280x720@30/25fps
6 - 1280x720@60/50fps

15 - 1920x1440@30/25fps
16 - 1920x1440@60/50fps
20 - 1280x960@30/25fps
21 - 1280x960@60/50fps
26 - 2560x1920@30/25fps
28 - 3840x2880@30/25fps
41 - 848x480@30/25fps

For B5_IMX206:
50 - 1920x960@30/25fps
51 - 1920x960@60/50fps

For B5_AR0330:
0 - 1920x960@30/25fps
1 - 1920x960@60/50fps

3. Runtime change sensor mode

   t videoenc chg [modeID]
   modeID = Liveview's mode ID

4. Stop liveview

   t videoenc liveviewstop

5. Start video encoding

   t videoenc encstart [milliSeconds]
   milliSeconds: Duration of encoding in millisecond units. It can be absent while performing endless encoding.

6. Stop video encoding (when **[milliSeconds]** is not set)

   t videoenc encstop

7. Pause video encoding

   t videoenc pause

   Note: On A9S, pause is not supported in the resolutions run in Pipeline Mode 5.
   On A12, pause/resume is not supported.

8. Resume video encoding

   t videoenc resume

   Note: On A12, pause/resume is not supported.

9. Enable secondary stream

   t videoenc dual [enable]
   0: Disable (default) 1: Enable
   Note: Default setting will be (1) A9/A12: 720x480, (2) A9S: 848x480. The command could fail if the second stream's resolution exceeds the limit. (regardless of the resolution of the main stream)
   Note: On A9S, the following modes only support 432x240: (1) 1920x1080P120/100, (2) 1280x720P240/200.

10. Enable Dual HD stream

t videoenc dualhd [enable]

This cmd is replaced by " t videoenc dual" + "t videoenc secwin".

Note: Not recommend second stream's resolution exceeds the Pirmary stream resolutionlimit.

Sensor	Primary	Secondary	EIS (A12Mode5)
OV4689	1920X1080 P 30	1920X1080 P 30	
	1920X1080 P 60	1920X1080 P 30	
	1280X 720 P 30	1280X 720 P 30	
	1280X 720 P 60	1280X 720 P 30	
	2560X1440 P 30	1920x1080 P 30	
	2560X1440 P 50	1280x720 P 25	
	2560X1440 P 60 (on A12, Hybrid Mode only)	720x400 P 30	
	1920X1080 P 30 HDR	1920X1080 P 30	N/A
	2560X1440 P 30 HDR	1920X1080 P 30	N/A
IMX117	1920X1080 P 30	1920X1080 P 30	
	1920X1080 P 60 (on A12, ExpressMode PAL mode only)	1920X1080 P 30	
	1920X1080 P120 (on A12, ExpressMode PAL mode only)	1280X720 P 30 (No dual HD on A9S)	N/A
	1280X 720 P 30	1280X 720 P 30	
	1280X 720 P 60	1280X 720 P 30	
	1280X 720 P120	1280X 720 P 30	N/A
	1280X 720 P240 (on A12, ExpressMode PAL mode only) (On A9S, PAL mode only)	1280X 720 P 30 (No dual HD on A9S)	N/A
	2560X1440 P 30	1920x1080 P 30	
	2560X1440 P 60 (on A12, only PAL mode can support dual) (On A9S, must disable TV)	1280x720 P 30 (On A12, 720x400 P 30)	
	3840X2160 P 30 (On A12 HybridMode 4K mode)	720x400 P 30	
	2560X1920 P 30	1280x 960 P 30	
	1280X 960 P 30 (A9/A9S only)	1280X 960 P 30	
	1280X 960 P 60 (A9/A9S only)	1280X 960 P 30	
	1280X 960 P120 (A9/A9S only)	1280X 960 P 30	N/A
	2688X1520 P 30	1280X 720 P 30	
IMX206	1920X1080 P 30	1920X1080 P 30	TBD
	1920X1080 P 60	1920X1080 P 30	TBD
	1920X1080 P100 (on A12, ExpressMode only)	1920X1080 P 25	N/A
	1280X 720 P 30	1280X 720 P 30	TBD
	1280X 720 P 60	1280X 720 P 30	TBD
	1280X 720 P120 (A12 only)	1280X 720 P 30	N/A

	1280X 960 P 30	1280X 960 P 30	TBD
	1280X 960 P 60	1280X 960 P 30	TBD
	1280X 960 P120 (A12 only)	1280X 960 P 30	N/A
AR0330p	1920X1080 P 30	1920X1080 P 30	TBD
	1280X 720 P 30	1280X 720 P 30	TBD
	1280X 720 P 60	1280X 720 P 30	TBD
	1280X 960 P 30 (A9 only)	1280X 960 P 30	TBD
AR0230	1920X1080 P 30	1920X1080 P 30	TBD
	1920X1080 P 60	1920X1080 P 30	TBD
	1920X1080 P 30 HDR	1920X1080 P 30	N/A
	1280X 720 P 30	1280X 720 P 30	TBD
	1280X 720 P 60	1280X 720 P 30	TBD
	1280X 960 P 30 (A9 only)	1280X 960 P 30	TBD
	1280X 960 P 60 (A9 only)	1280X 960 P 30	TBD
MN34120	1920X1080 P 30	1920X1080 P 30	TBD
	1920X1080 P 60	1920X1080 P 30	TBD
	1280X 720 P 30	1280X 720 P 30	TBD
	1280X 720 P 60	1280X 720 P 30	TBD
	1280X 720 P 120 (A12 only)	1280X 720 P 30	N/A
	3840X2160 P 15(on A12, HybridMode 4K only)	720X400 P 15	TBD
IMX290	1920X1080 P 30	1920X1080 P 30	TBD
	1920X1080 P 60	1920X1080 P 30	TBD
	1920X1080 P 120 (on A12, Express Mode PAL mode only)	1920X1080 P 30	N/A
	1920X1080 P 30 HDR	1920X1080 P 30	N/A
	1920X1080 P 60 HDR	1920X1080 P 30	TBD
MN34229	1920X1080 P 30	1920X1080 P 30	TBD
	1920X1080 P 60	1920X1080 P 30	TBD
	1920X1080 P 30 HDR	1920X1080 P 30	N/A
	1920X1080 P 60 HDR	1920X1080 P 30	N/A
IMX377	1920X1080 P 30	1920X1080 P 30	
	1920X1080 P 60	1920X1080 P 30	
	1920X1080 P120 (on A12, Express Mode PAL mode only)	1280X 720 P 30	N/A
	1280X 720 P 30	1280X 720 P 30	
	1280X 720 P 60	1280X 720 P 30	
	1280X 720 P120	1280X 720 P 30	N/A
	1280X 720 P240 (on A12 only, Express- Mode PAL mode only) (On A9S, PAL mode only)	1280X 720 P 30 (No dual HD on A9S)	N/A
	2560X1440 P 30	1920X1080 P 30	
	2560X1440 P 60 (A12 only, only PAL mode can support dual stream)	(A12)720X400 P 30	

	3840X2160 P 30 (A12 only)	(A12)720X400 P 30	
	1280X 960 P 30(A9S only)	1280X 960 P 30	TBD
	1280X 960 P 60(A9S only)	1280X 960 P 30	TBD
	2688X1520 P 30	1280X 720 P 30	TBD
IMX317 (A9S only)	1920X1080 P 30	1920X1080 P 30	TBD
	1920X1080 P 60	1920X1080 P 60	TBD
	1280X 720 P 30	1280X 720 P 30	TBD
	1280X 720 P 60	1280X 720 P 30	TBD
	1280X 720 P120	1280X 720 P 30	TBD
	2688X1520 P 30	1280X 720 P 30	TBD
AR0237	1920X1080 P 30	1920X1080 P 30	
	1920X1080 P 30 HDR	1920X1080 P 30	N/A
IMX078	1920X1080 P 30	1920X1080 P 30	
	1920X1080 P 60 (on A12, Express Mode PAL mode only)	1920X1080 P 30	
	1280X 720 P 30	1280X 720 P 30	
	1280X 720 P 60	1280X 720 P 30	
	1280X 720 P120	1280X 720 P 30	N/A
	2560X1440 P 30	1920x1080 P 30	
	2560X1440 P 60 (on A12, only PAL mode can support dual stream)	720x400 P 30	
	3840X2160 P 30 (On A12 HybridMode 4K mode)	720x400 P 30	
OV2718	1920X1080 P 30	1920X1080 P 30	
	1920X 1080 P 30 CFA HDR	1920X1080 P 30	
TI5150	720X480 I 60	720X480 I 60	N/A
	720X576 I 50	720X576 I 50	N/A
NVP6114A	1280X 720 P30	1280X 720 P 30	N/A
B5_IMX206	1920X 960 P30	1920X 960 P 30	N/A
	1920X 960 P60	1920X 960 P 30	N/A
B5_AR0330	1920X 1080 P30	1920X 1080 P 30	N/A
	1920X 1080 P60	1920X 1080 P 30	N/A

11. Set the secondary stream encode window

    t videoenc secwin [Width] [Height]

    Note: Please always use this command to specify the 2nd stream's resolution if dual stream or dual HD is enabled.
    Note: For dual stream, the maximum resolution of the 2nd stream is (1) A9/A12: 720x480, (2) A9S: 848x480.
    Note: For dual HD, the maximum resolution is 1920x1080. But Ambarella does not suggest to use the ones whose width larger than 1280 because they are out-of-spec.
    Note: The minimum resolution is suggested to be no smaller than (MaxWidth / 8) x (MaxHeight / 8).
    Note: On A12, height must be 16-aligned.

12. Set secondary stream frame rate (A9S only)

t videoenc secfrate [TimeScale] [TimePerFrame]
Note: The frame rate of the 2nd stream cannot be larger than that os the 1st stream.
Note: Example: 30fps => TimeScale = 30000, TimePerFrame = 1001. 60fps => TimeScale = 60000, TimePerFrame = 1001.

13. Test the bitrate control setting

t videoenc brc [option]
option: config quality control or invoke
config - config quality control parameters
t videoenc brc config [mode][...]
[mode][Stream][...]
stream: 0: Primary stream, 1: Secondary stream
mode: Bitrate control option

Mode = 0 - Bitrate config
t videoenc brc config 0 [Stream][BrcMode][AvgBitRate][MaxBitRate]
[MinBitRate][iBeat]
BrcMode: 1 - CBR, 2 – SmartVBR
AvgBitRate: Average Bitrate. Unit: Mbps
MaxBitRate: Maximum Bitrate for SmartVBR only. Unit: Mbps (max: 54)
MinBitRate: Minimum Bitrate for SmartVBR only. Unit: Mbps (min: 6)

The default values are suggested as below:

	MinBitRate	MaxBitRate	AvgBitRate
1920X1080 P30	9	15	12
1920X1080 P60	13	21	18
1920X1080 P120	27	45	36
1280X720 P30	6	10	8
1280X720 P60	9	15	12
1280X720 P120	13	22	18
1280X720 P240	27	45	36
2560X1440 P30	18	30	24
2560X1440 P60	27	45	36
3840X2160 P30	45	75	60
3840X2160 P15	22	37	30
1920X1440 P30	13	22	18
1280X960 P30	10	15	13
1280X960 P60	12	18	15
1280X960 P120	18	27	24
2560X1920 P30	27	45	36
2560X1920 P60	27	45	36
2704X1520 P60	27	41	36
2688X1520 P30	27	41	36
720X400 P30	1.5	2.5	2

Table 6-1. Default Values.

The expected result are (1) for VBR clips, MinBitRate <= clip bit rate <= MaxBitRate, and (2) for CBR clips, (AvgBitRate * 0.9) <= clip bit rate <= (AvgBitRate * 1.1).

iBeat: 0 – Off
1 - M-1 B pictures after IDR picture requires special quant reduction
2 - The P picture just before IDR picture requires special quant reduction
4 - M-1 M-1 B pictures before IDR picture requires special quant reduction
8 - The P picture just before I picture requires special quant reduction
Note: Before invoking CBR, please turn off encode monitor first (refer to 28, 29)

Note: For engineer test only.

Mode = 1 - Bitrate change
t videoenc brc config 1 [Stream][AverBitRtate]
AverBitRtate: Unit in Mbps
Mode = 2 - Gop Change

Note: If AvgBitrate change is too low with original setting, it will easy hit the DESC_OVERRUN issue.

t videoenc brc config 2 [Stream][M][N][Idr]
M: Distance between in P frame(simple rc does not support M>1)
N: Distance between in I frame
Idr: Distance between in IDR frame
Mode = 3 - QP control
t videoenc brc config 3 [Stream][QpminI][QPmaxI][QPminP][QPmaxP][QPminB][QPmaxB]

QPmin: Minimum QP value
QPmax: Maximum QP value
Mode = 4 - QP model control
t videoenc brc config 4
Mode = 5 - ROI control
t videoenc brc config 5
Mode = 6 - HQP control
t videoenc brc config 6 [Qpmax][QPmin][QPreduce]
QPmin: Minimum QP value
QPmax: Maximum QP value
QPreduce: How much better to make Q QP relative to P QP, 1~10, default is 6
Mode = 7 - ZMV control
t videoenc brc config 7 [threshold]
threshold: ZMV threshold
Mode 8 = - Force Idr
t videoenc brc config 8
Mode = 9 - RESET ALL
t videoenc brc config 9
invoke - invoke command
t videoenc brc invoke [stream]
0: Primary stream
1: Secondary stream

14. Show current bitrate information

t videoenc brcshow
Note: **t videoenc brcshow reset** to reset bitrate data statistics

15. Set the bitstream specification

t videoenc spec [priSpec] [secSpec]

priSpec: Primary stream spec. 0: MJPEG, 1: H.264
secSpec: Secondary stream spec. 0: MJPEG, 1: H.264

Note: (1) HFR does not support MJPEG.
(2) A12 express mode does not support primary stream as MJPEG.
(3) This command must be called before starting liveview.

16. Set the encode rotation

t videoenc rotate [rotate]

rotate: Pre-defined rotation values
0: No rotation
1: Horizontal flip
2: 90 clockwise
3: 90 clockwise then vertical flip
4: 180 clockwise
5: 180 clockwise then horizontal flip
6: 270 clockwise
7: 270 clockwise then vertical flip

Note: 1. On A9S, rotation is not supported in HFR (frame rate > 60).
2. For A9/A9S, to rotate 90/270 degrees is only supported in 1920x1440P30, 1080P60/P30, 1280x960P60/P30, and 720P60/P30 (no dual).
3. For A12, 2560X1440P60 cannot support rotation.
4. A12 ExpressMode rotate result will show in Liveview and Encode stream; A12 Hybrid mode rotate result will show in Encode stream only.
5. A12 ExpressMode EIS only supports rotate = 4.
6. The effect only is shown in the resulted bitstream.

17. PIV during encoding

t videoenc piv [encW] [encH] [iso] [cmpr] [targetsize] [encodeLoop]
encW: PIV encode
width encW: PIV
encode height

Note: The list below shows the interpolation limitation of different recording sizes (a different AspectRatio is supported but not suggested).
Note: On A9S, the PIV size cannot be larger than the capture window size.
Note: On A9S, due to performance issue, PIV is not supported if background process mode is enabled.

[A12ExpressMode] Recording window	PIV interpolation limit
1920X1080 P 30	3840X2160
1920X1080 P 60	2560X1440
1920X1080 P100	1920X1080
1280X 720 P 30	3840X2160
1280X 720 P 60	2560X1920
1280X 720 P120	1280X720
1280X 720 P200	1280X720
2560X1440 P 30	3840X2160
2560X1440 P 50	3840X2160
2560X1440 P 60	2560X1440

2560X1920 P 30	2560X1920

[A12HybridMode] Recording window	PIV interpolation limit
1920X1080 P 30	3840X2160
1920X1080 P 60	2560X1440
1920X1080 P100	Not Support
1920X1080 P 30 HDR	3840X216
1280X 720 P 30	3840X216
1280X 720 P 60	2560X1440
1280X 720 P120	1280X720
1280X 720 P200	Not Support
2560X1440 P 30	3840X2160
2560X1440 P 30 HDR	2560X1440
2560X1440 P 50	Not Support
2560X1440 P 60	Not Support
3840X2160 P 30 (4K mode)	3840X2160

[A9S] Recording window	PIV interpolation limit
1920X1080 P 30	3840X2160
1920X1080 P 60	3840X2160
1280X 720 P 30	3840X2160
1280X 720 P 60	3840X2160
2560X1440 P 30	3840X2160
2560X1440 P 60	3840X2160
1920X1440 P 30	4000X3000
1280X 960 P 30	3840X2880 (up to 1322X998 on IMX377 due to support PAL mode)
1280X 960 P 60	2000X1500
2704x1520 P 60	3840X2160
2688x1520 P 30	3840X2160

ISO: Algo mode, 0: HISO 1: LISO 2: FastMode *

Note: A12ExpressMode and A12HybridMode support LISO only. A9S supports LISO only.
cmpr: Raw compressed or not
targetsize: Target encode size is in Kbyte unit. 0 implies “encW * encH / 2000”.
Note: On A12, targetsize must be set to 0, because A12 does not support BRC on PIV.

Note: On A9S, do not mix targetsize 0 and non-zero values in a record session. That is, if the first PIV command sets targetsize to 0, the following PIV commands must also set targetsize to 0. If the fist PIV command sets targetsize to a non-zero value, the following PIV commands must also set targetsize to non-zero values.

encodeLoop: Re-encode loop in BRC (max: 255)
If encodeLoop is not enabled, the resulted file size could be very different from targetsize.
A larger encodeLoop usually implies a smaller difference to targetsize.
On A12, encodeLoop must be set to 0, because A12 does not support BRC on PIV.

18. Time lapse encoding enable

t videoenc timelapse [enable]
enable - 0: Disable (default), 1: Enable time lapse mode

Note: On A9S, time lapse encoding is not supported in the resolutions run in Pipeline Mode 5.

19. Time lapse encoding capture one frame

t videoenc tcap
Capture a time lapsed video frame

20. Set the initial DZOOM

t videoenc initzoom [factorX] [factorY]
factorX: Horizontal DZOOM factor in decimal format. Ex. 1.0 (default)
factorY: Vertical DZOOM factor in decimal format. Ex. 1.0 (default)
The factorX and factorY Max values are 4.

Note: DZOOM cannot work with EIS.
Note: On A9S, DZOOM is not supported on 1280x720P240.

21. Digital zoom jump

t videoenc dzjump [Step] [shiftFactorX] [shiftFactorY]
Step - A12: 0~486, A9S:0~429, dzoom level.
shiftFactorX: -1.0~1.0, shifting factor in horizontal.
0 means the center
shiftFactorY: -1.0~1.0, shifting factor in vertical.
0 means the center

22. Digital zoom move

t videoenc dzmove [moveType] [speed] [distance] [shiftFactorX] [shiftFactorY]
moveType: 0~2, move type
0 means zoom in
1 means zoom out
2 means only shift
speed: 0~5, speed level
0 means the fastest
5 means the lowest
distance - A12: 0~486, A9S: 0~429
When moveType is 0, it means increasing the dzoom level.
When moveType is 1, it means decreasing the dzoom level.
When moveType is 2, it means shifting the Steps.
shiftFactorX: -1.0~1.0, shifting factor in horizontal
0 means the center.
shiftFactorY: -1.0~1.0, shifting factor in vertical
0 means the center
Note: The maximum valid shiftFactorX and shiftFactorY are different based on the dzoom level.

23. Dzoom stop

t videoenc dzstop
Force stopping dzjump or dzmove when it executes

24. Enable encode with thumbnail

t videoenc thm [enable]

enable - 0: Disable (default), 1: Enable thumbnail
Note: Thumbnail is implemented via PIV, so it has the same restrictions as PIV.
Note: On A12, please enable video thumbnail before Liveview starts.

25. Enable TV

t videoenc tv [enable]
enable - 0: Disable , 1: Enable TV F_chan (default)

26. Enable blend

t videoenc blend [streamID] [enable]
streamID: 0 - Primary stream, 1 - Secondary stream
enable: Enable blend
Note: On A9S, blending is not supported in (1) 3840x2160, and (2) the resolutions run in Pipeline Mode 5. Due to performance issue, TV must be disabled in 2560x1440P60.
Note: A12ExpressMode blending result will be shown in Liveview and Encode stream; A12 Hybridmode blending result will be shown in Encode stream only.

27. Set system type

t videoenc system [option]
enable - 0: NTSC(default) , 1: PAL

28. Encode bitrate monitor control

t videoenc encmonitor [control]
control - 0: Disable, 1: Enable

29. Encode aqp monitor control (A12 only)

t videoenc encmonitoraqp [control]
control - 0: Disable, 1: Enable

30. EIS demonstration control

t videoenc eis hook [EisDriverId] [SensorId]
EisDriverId - 0: DummyEis
SensorId - 0: sensor_0 (the 1st sensor in the system), 1: sensor_1 (the 2nd sensor in the system)

t videoenc eis algoen [enable]
enable - 0: Disable 1: Enable

Please note that the above commands must be executed before the unit test is initiated.

Note: Once EIS is enabled, the view angle of the resulted video/image would be slightly reduced(Compare to NonEisMode).
Note: A9S has the following limitations for EIS:
- TV must be disabled.
- Does not support HighFrameRate(>60fps).
- EIS cannot work with dzoom.

- Not supported on IMX117 (Vertical Blanking bound, if really need EIS on IMX117, please apply work-around by reducing FOV)
- Does not support 3840x2160P30 and 2704x1520P60.
- Does not support 2560x1440P60 (1) with blending enabled, or (2) in background process mode.
- Only supports 2688x1520P30 and 2560x1920P30 in the background process mode.
- Only supports 2560x1440P30 with dual stream in the background process mode.

Note: A12 Mode0 EIS supports limitation.

Item	Limitation
Resolution	<= 1920X1080 (IMX117 also can't suppot FHD size)
HFR (>60fps)	Not Support
PAL	Supported
Stamp	Not support in SGOP**
Digital Zoom	Not Support
SlowShutter	Not Support
LDC	Not Support
PIV	Supported
VideoThumbnail	Supported
Rotate	Not Support

Table 6-2. Items and Limitations.

Item	Limitation
Resolution	<= 3840X2160
HFR (>60fps)	Not Support
PAL	Supported
Stamp	Not support in SGOP**
HDR	Not Support
TA	Not Support
Digital Zoom	Supported but must disable EIS when Dzoom moving
SlowShutter	Not Support
LDC	Not Support
PIV	Supported but no EIS correction effect
VideoThumbnail	Supported
Rotate	Not support in SGOP**
DualVin	Not Support now

Table 6-3. Items and Limitations.

**SGOP: 4Kp30/Compact4Kp30/4Mp60

31. Slow shutter

    t videoenc slowshutter [enable]

    Note: The speed can be set via the test command "t imgproc".
    Note: On A12, only supports 2x slowshutter at most, i.e, 30fps -> 15fps, 60fps->30fps... Currently, only the following frame rates support slow shutter: 1) 60fps (and its PAL) 2) 120fps (and its PAL) 3) 240fps (and its PAL).
    Note: A9S only supports 8x slowshutter at most. Due to the ucode limitation, for 120fps, only 4x slow-shutter is supported, and for 240fps, slowshutter is not supported.
    Note: On A9S, please disable TV in HFR mode. In HFR mode with dual stream, please set the 2nd stream to 432x240.

32. Optical black correction (A12 only)

    t videoenc VideoOB [enable]

    Note: For A12, currently, only IMX206 sensor supports OB mode.

33. Change Liveview process mode

t videoenc lvproc [pipe] [algo]
pipe - 0: ExpressMode, on A12, background process mode disabled on A9/A9S,
1: HybridMode on A12, background process mode enabled on A9/A9S.
algo - 0: FastMode, 1: LISO

Note: This command must be issued before the liveview start command.
Note: On A9S, the background process mode is always enabled in 4K mode, i.e., the parameter "pipe" will be ignored. Besides, the background process mode cannot be enabled in HFR modes, i.e., the parameter "pipe" can only be 0.

34. Enable Hybrid algo TA mode (A12 only)

t videoenc ta [en]
en - 0: Disable, 1: Enable
Note: 1. This command is only valid in the hybrid LISO mode.
2. For A12, HybridMode only
3. This command must be issued before the liveview start command.
4. The following sensors are supported now: OV4689/AR0230/MN34120

35. Enable low delay mode

([A9S] only)

t videoenc lowdelay [Mode] [DelayTime]
Mode - 0: disabled, 1: vin, 2: vin<-> vout0 (LCD), 3: vin<-> vout1 (TV)
DelayTime: in unit of 0.1 milliseconds

[A12] (HybridMode only)

t videoenc lowdelay [LiveviewLowDelay] [EncodeLowDelay]
LiveviewLowDelay - 0: Disabled, 1: VIN <-> vout0 (LCD), 2: VIN <-> vout1 (TV)
EncodeLowDelay - 0: Disabled, 1: Enable low delay on the whole encode stream

Note:

1. On A12, when enabling VOUT low delay, some limitation occur.

Vout delay type	Limitation
Tv low delay	Lcd or 2nd stream must disable

2. On A12, when encode low delay is enabled, encode performance will have impact.

Video Feature	Limitation
TBD	TBD

LowDelay verification

- VOUT Low Delay

Please aim the sensor at the StopWatch, then the user will observe LCD/TV/StopWatch have time difference if the VOUT low delay is enabled, the figure displaying below is TV low delay case.

Table 6-4. TV Low Delay Case.

- Encode Low Dealy

Type “t dsp pts on 3” before encode start and console will show the following message:

EncodeLowDelay Enable:

[00007142][CA9] ch[0]:91091

[00007142][CA9] PicInfo.CaptureTimeStamp(93949078)

[00007142][CA9] PicInfo.EncodeTimeStamp(94091122)

[00007142][CA9] Diff (EncodePTS-CapturePTS)(142044)

EncodeLowDelay Disable:

[00010421][CA9] ch[0]:97097

[00010421][CA9] PicInfo.CaptureTimeStamp(133421597)

[00010421][CA9] PicInfo.EncodeTimeStamp(134290525)

[00010421][CA9] Diff (EncodePTS-CapturePTS)(868928)

Encode improve (868928 - 142044)/12288 = 59.15 ms

36. Enable CVBS for fchan (A9S only)

    t videodevc cvbs [enable]
    enable - 0: Disable 1: Enable

37. Set Stream Encode Priority (A12 HybridMode only)

    t videoenc streampriority [PriStreamPriority] [SecStreamPriority]
    PriStreamPriority - 1: 1st priority, 2: 2nd priority…
    SecStreamPriority - 1: 1st priority, 2: 2nd priority…

    Note: *1. Must be set before liveview start.*
    2. Verify it by type "t dsp h264 on" to check encode frame output order.
    3. For example, for PriStrm slow than SecStrm, use t videoenc streampriority 2 1.

6.2.2 Video Encoder: Unit Test

On A12, please type "t ve perform 1 1 1" before "liveviewstart" in 3840x2160p30.

1. Liveview start/stop

   [Command sequence]
   Step 1: **t videoenc init [sensorID]**
   Step 2: **t videoenc liveviewstart [modeID]**
   Step 3: **t videoenc liveviewstop**

2. Change resolution

   [Command sequence]
   Step 1: **t videoenc init [sensorID]**
   Step 2: **t videoenc liveviewstart [modeID]**
   Step 3: **t videoenc chg [modeID]**

3. Video encoding start/stop

   [Command sequence]
   Step 1: **t videoenc init [sensorID]**
   Step 2: **t videoenc liveviewstart [modeID]**
   Step 3: **t videoenc encstart [milliSeconds]**
   Step 4: **t videoenc encstop** (if [milliSeconds] is not set)
   [Output]
   OUT_XXXX.h264
   OUT_XXXX.udta
   OUT_XXXX.nhnt

4. Dualstream encoding start/stop

   [Command sequence]
   Step 1: **t videoenc init [sensorID]**
   Step 2: **t videoenc dual [enable]**
   Step 3: **t videoenc liveviewstart [modeID]**
   Step 4: **t videoenc encstart [milliSeconds]**

[Output]
OUT_XXXX_s.h264
OUT_XXXX_s.udta
OUT_XXXX_s.nhnt

5. Initial zoom

[Command sequence]
Step 1: **t videoenc init [sensorID]**
Step 2: **t videoenc initzoom [factorX] [factorY]**
Step 3: **t videoenc liveviewstart [modeID]**

6. Test BRC settings

[Command sequence]
Step 1: **t videoenc init [sensorID]**
Step 2: **t videoenc liveviewstart [modeID]**
Step 3: **t videoenc brc config 0 [stream] [brcMode] [avgBitRate] [maxBitRate] [minBitRate] [iBeat]**
Step 4: **t videoenc encmonitor [control]**
Step 5: **t videoenc encmonitoraqp [control] (A12 only)**
Step 6: **t videoenc encstart [milliSeconds]**
Note: CBR does not support encode monitor.

7. Test bitstream spec (A12 only supports SecondStream MJpeg)

[Command sequence]
Step 1: **t videoenc init [sensorID]**
Step 2: **t videoenc spec [priSpec] [secSpec]**
Step 3: **t videoenc liveviewstart [modeID]**
Step 4: **t videoenc encstart [milliSeconds]**
[Output]
OUT_XXXX_s.h264 or OUT_XXXX.mjpeg
OUT_XXXX_s.udta
OUT_XXXX_s.nhnt
Note: Do not aim at a complex scene when testing MJPEG due to bad encoding efficiency of JPEG specification.

8. Change average bitrate during encoding

[Command sequence]
Step 1: **t videoenc init [sensorID]**
Step 2: **t videoenc liveviewstart [modeID]**
Step 3: **t videoenc encmonitor [control]**
Step 4: **t videoenc encmonitoraqp [control] (A12 only)**
Step 5: **t videoenc encstart**
Step 6: **t videoenc brc config 1 [stream] [avgBitrate]**
Step 7: **t videoenc brc invoke [stream]**
Step 8: **t videoenc encstop**
Note: Please note that once AverageBitRate is changed during encoding, the final average bitrate shall be between original BitRate and Changed BitRate.

9. Encode rotation

   [Command sequence]
   Step 1: **t videoenc init [sensorID]**
   Step 2: **t videoenc liveviewstart [modeID]**
   Step 3: **t videoenc rotate [rotate]**
   Step 4: **t videoenc encstart [milliSeconds**]
   0: Disable (default)

   Note: On A9S, rotation is not supported in HFR (frame rate > 60).
   Note: On A9S, to rotate 90/270 degrees is only supported in 1920x1440P30, 1080P60/P30, 1280x960P60/P30, and 720P60/P30 (no dual).

10. PIV during encoding

    [Command sequence]
    Step 1: **t videoenc init [sensorID]**
    Step 2: **t videoenc tv 0** (A9/A9S only)
    Step 3: **t videoenc liveviewstart [modeID]**
    Step 4: **t videoenc encstart**
    Step 5: **t videoenc piv [encW] [encH] [iso] [cmpr] [targetsize] [encodeLoop]**

    [Output]
    A set of JPEG files in the SD card root are named as follows:
    xxxx_m_nn.jpeg (main)
    xxxx_t_nn.jpeg (thumbnail)
    xxxx_s_nn.jpeg (screennail)
    Note: There is no way to see if cmpr really takes effect. Just make sure all test steps are passed.
    Note: If encodeLoop is set to 1 and targetsize is set to a very small value, the resulting JPEG must be smaller than the one under encodeLoop = 0 and targetsize = 0.

11. Time lapsed encoding

    [Command sequence]
    Step 1: **t videoenc init [sensorID]**
    Step 2: **t videoenc timelapse 1**
    Step 3: **t videoenc liveviewstart [modeID]**
    Step 4: **t videoenc encstart**
    Step 5: **t videoenc tcap**
    Step 6: Repeat Step 5 until user is satisfied
    Step 7: **t videoenc encstop**
    Note: Time lapsed encoding supports dual stream.
    Note: Need to issue Step 5 several times to let the first encoded frame out.
    Note: On A9S, time lapse encoding is not supported in the resolutions run in Pipeline Mode 5.

12. Enable secondary stream at HD resolution

    Note: Dual HD does not support TV liveview
    [Command sequence]
    Step 1: **t videoenc init [sensorID**]
    Step 2: **t videoenc tv 0**
    Step 3: **t videoenc dualhd [enable]**
    Step 4: **t videoenc secwin [encW] [encH] (to ensure the second one following the spec)**
    Step 5: **t videoenc liveviewstart [modeID]**
    Step 6: **t videoenc encstart [milliSeconds]**

[Output]

Resolution will be equal to main.
OUT_XXXX_s.h264
OUT_XXXX_s.udta
OUT_XXXX_s.nhnt

13. Change frame rate during encoding (for engineer test only)

[Command sequence]

Step 1: **t videoenc init [sensorID]**
Step 2: **t videoenc liveviewstart [modeID]**
Step 3: **t videoenc encstart**
Step 4: **t videoenc chgfrate [priStreamDivisor] [secStreamDivisor]**
Step 5: **t videoenc encstop**

Note: On A9S, runtime frame rate change is not supported in the resolutions run in Pipeline Mode 5.
On A12, Ambarella only supports this feature for streaming but not for encoding.

14. Digital zoom

[Command sequence]

Step 1: **t videoenc init [sensorID]**
Step 2: **t videoenc liveviewstart [modeID]**
Step 3: **t videoenc dzjump [Step] [shiftFactorX] [shiftFactorY]**
Step 4: **t videoenc dzmove [moveType] [speed] [distance] [shiftFactorX] [shiftFactorY]**
Step 5: **t videoenc dzstop**

15. EIS demonstration

[Command sequence]

Step 1: **t videoenc eis algoen [enable]**
Step 2: **t videoenc eis hook [driverId] [sensorId]**
Step 3: **t videoenc init [sensorID]**
Step 4: **t videoenc lvproc 0 0**
Step 5: **t videoenc liveviewstart [modeID]**
Step 6 (Optional): **t imgproc -eis 101 1 3**

Note: Liveview will shake periodically. With Step 6, liveview will have rotation effect. (Only A12 ExpressMode supports rotation effectively).
Note: For Step 2, "sensorId" 0 means the 1st sensor in the system, 1 means the 2nd sensor in the system.
Note: For Step 4, please check the rules listed in "Change Liveview process mode" in 6.2.1. For example, the parameter "pipe" can only be 0 in HFR modes.

16. Slow shutter

[Command sequence]

Step 1: **t videoenc init [sensorID]**
Step 2: **t videoenc liveviewstart [modeID]**
Step 3: **t videoenc slowshutter 1**

17. Optical black correction (A12 only)

[Command sequence]

Step 1: **t videoenc init [sensorID]**
Step 2: **t videoenc VideoOB [enable]**
Step 3: **t videoenc liveviewstart [modeID]**

18. Video pipeline change

    [Command sequence]
    Step 1: **t videoenc init [sensorID]**
    Step 2: **t videoenc lvproc [pipe] [algo]**
    Step 3: **t videoenc liveviewstart [modeID]**

19. Enter hybrid Algo TA mode (A12 Hybrid Mode only)

    [Command sequence]
    Step 1: **t videoenc init [sensorID]**
    Step 2: **t videoenc lvproc [pipe] [algo]**
    Step 2: **t videoenc ta [en]**
    Step 3: **t videoenc liveviewstart [modeID]**

20. CVBS test

    [Command sequence]
    Step 1: **t videoenc cvbs 1**
    Step 2: **t videoenc init [sensorID]**
    Step 3: **t videoenc liveviewstart [modeID]**

6.3 Unit Tests: MotorVu 360 (B5 only)

MotorVu 360 unit test provides the following function tests:

1. 10 FOV surrounding view on HDMI/CVBS
2. H264 encoding for four views

6.3.1 MotorVu360: Command List

1. Initialize MotorVu unit test and hook the sensor

   t mv init [sensorID]
   sensorID: 0 – OV10640@B5

2. Enter surround view

   t mv liveviewstart [modeID]
   modeID: 0 – Surround view

3. Stop surround view

   t mv liveviewstop

4. Start encoding

   t mv encstart [milliSeconds]
   milliSeconds: Time of encoding

5. Stop encoding

   t mv encstop

6.3.2 MotorVu360: Unit Test

1. Surround view start/stop

   [Command sequence]
   Step 1: **t mv init [sensorID]**
   Step 2: **t mv liveviewstart [modeID]**
   Step 3: **t mv liveviewstop**

2. Video encoding start/stop

   [Command sequence]
   Step 1: **t mv init [sensorID]**
   Step 2: **t mv liveviewstart [modeID]**
   Step 3: **t mv encstart [milliSeconds]**
   Step 4: **t mv encstop** (if [milliSeconds] is not set)

   [Output]
   OUT_XXXX.h264
   OUT_XXXX.udta
   OUT_XXXX.nhnt
   Content will be a 2560x1920, with a 2x2 grid showing 4 camera views.

6.4 Unit Tests: Video Decoder

SDK6 video decoder unit test provides the following function tests:

1. H264 video decode
2. H264 video trickplay (fast/slow forward, fast/slow backward, pause, Step, time search)
3. H264 video zoom
4. H264 video playback capture

6.4.1 Video Decoder: Command List

1. Init

   Enable LCD only
   [Command sequence]
   Step 1: **t videodec init 3**
   Step 2: **t videodec ch 0**

   Enable TV only
   [Command sequence]
   Step 1: **t videodec init 3**
   Step 2: **t videodec ch 1**

   Dual Vout method 1
   [Command Sequence]
   Step 1: **t videodec init 3**
   Step 2: **t videodec ch 0**
   Step 3: **t videodec ch 1**

Dual Vout method 2
[Command sequence]
Step 1: **t videodec init**

2. Start decoding

[Command sequence]
Step 1: **t videodec start [filePath] [videoWidth] [videoHeight] [startTime] [Speed] [Dir]**
filePath: the .nhnt/.h264/.udta file full path (excluding file extension)
videoWidth: Video width
videoHeight: Video height
startTime: time (in ms) of the first displayed video frame
Speed: 128 – 0.5x, 256 – 1x, 512 – 2x
Dir: 0: Forward, 1: Backward

3. Zoom during decoding

[Command sequence]
Step 1: **t videodec zoom [videoWidth] [videoHeight] [cropOffsetX] [cropOffsetY] [cropWidth] [cro pHeight]**
videoWidth: Video width
videoHeight: Video height
cropOffsetX: Cropping area horizontal offset
cropOffsetY: Cropping area vertical offset
cropWidth: Cropping area width
cropHeight: Cropping area height

Example: 2x center zoom of a 1080P video:
t videodec zoom 1920 1080 480 270 960 540

4. Pause during decoding

[Command sequence]
Step 1: **t videodec pause**

5. Resume during pause

[Command sequence]
Step 1: **t videodec resume**

6. Stop decoding

[Command sequence]
Step 1: **t videodec stop**

7. Capture YUV during pause

[Command sequence]
Step 1: **t videodec cap [width] [height]**
width: Output width
height: Output height

Note: Video must be in "pause" status before capturing YUV.

8. Fade-in effect

[Command sequence]
Step 1: **t videodec fade [enable]**
Enable: 0 to disable (default), 1 to enable

Note: The "Fade-in" effect must be set before starting the command. It will make a video fade from pink color in such a way that the user can see the effect of the fade-in more clearly.

9. Exit video decode

   [Command sequence]
   Step 1: **t videodec exit**
   Note: Exit video decode must be called before the mode is switched.

10. Enable CVBS for fchan

    [Command sequence]
    Step 1: **t videodec cvbs 1**

6.4.2 Video Decoder: Golden Data

Please check MWUnitTest\GoldenData\Decode\VideoDecode for golden data.
Clip resolution/frame rate is noted on the folder name.

6.4.3 Video Decoder: Unit Test

Note: Use 1920x1080 resolution's golden data as an example.

1. General forward 1x decode with zoom

   [Command sequences]
   Step 1: Put OUT_1000.nhnt/.h264/.udta files to SD root
   Step 2: **t videodec init**
   Step 3: **t videodec start** C:\OUT_1000 1920 1080 0 256 0
   Step 4: **t videodec stop**
   [Output]
   The video plays forward at normal speed in Step 3.

2. Forward 1/2x decode

   [Command sequences]
   Step 1: Put OUT_1000.nhnt/.h264/.udta files to SD root
   Step 2: **t videodec init**
   Step 3: **t videodec start** C:\OUT_1000 1920 1080 0 128 0
   Step 4: **t videodec stop**
   [Output]
   N/A

3. Forward 2x decode

   [Command sequences]
   Step 1: Put OUT_1000.nhnt/.h264/.udta files to SD root
   Step 2: **t videodec init**
   Step 3: **t videodec start** C:\OUT_1000 1920 1080 0 512 0
   Step 4: **t videodec stop**

   [Output]
   N/A

4. Backward 1x decode

   [Command sequences]
   Step 1: Put OUT_1000.nhnt/.h264/.udta files to SD root
   Step 2: **t videodec init**
   Step 3: **t videodec start** C:\OUT_1000 1920 1080 5000 256 1
   Step 4: **t videodec stop**

   [Output]
   N/A

5. Backward 1/2x decode

   [Command sequences]
   Step 1: Put OUT_1000.nhnt/.h264/.udta file to SD root
   Step 2: **t videodec init**
   Step 3: **t videodec start** C:\OUT_1000 1920 1080 5000 128 1
   Step 4: **t videodec stop**

   [Output]
   N/A

6. Backward 2x decode

   [Command sequences]
   Step 1: Put OUT_1000.nhnt/.h264/.udta files to SD root
   Step 2: **t videodec init**
   Step 3: **t videodec start** C:\OUT_1000 1920 1080 5000 512 1
   Step 4: **t videodec stop**

   [Output]
   N/A

7. Zoom

   [Command sequences]
   Step 1: Put OUT_1000.nhnt/.h264/.udta files to SD root
   Step 2: **t videodec init**
   Step 3: **t videodec start** C:\OUT_1000 1920 1080 0 256 0
   Step 4: **t videodec zoom** 1920 1080 480 270 960 540
   Step 5: **t videodec stop**

   [Output]
   Zoom effect at Step 4

8. Capture YUV (except A12)

   [Command sequences]
   Step 1: Put OUT_1000.nhnt/.h264/.udta files to SD root
   Step 2: **t videodec init**
   Step 3: (optional) **t videodec fade 1**
   Step 4: **t videodec start** C:\OUT_1000 1920 1080 0 256 0
   Step 5: **t videodec pause**
   Step 6: **t videodec cap** 1920 1080
   Step 7: **t videodec resume**
   Step 8: **t videodec stop**

   [Output]
   Dump.Y, Dump.UV at SD root at Step 6

9. Fading effect (Except A12)

   [Command sequences]
   Step 1: Put OUT_1000.nhnt/.h264/.udta files to SD root
   Step 2: **t videodec init**
   Step 3: **t videodec fade 1**
   Step 4: **t videodec start** C:\OUT_1000 1920 1080 0 256 0
   Step 5: **t videodec stop**
   [Output]
   Fade-in effect from original pink color in Step 4.

10. CVBS test

    [Command sequences]
    Step 1: Put OUT_1000.nhnt/.h264/.udta files to SD root
    Step 2: **t videodec cvbs 1**
    Step 3: **t videodec init**
    Step 4: **t videodec start** C:\OUT_1000 1920 1080 0 256 0
    Step 5: **t videodec stop**
    [Output]
    The video plays forward at normal speed in Step 4.

6.5 Unit Tests: Video Multi-channel Decode

Video multi-channel decoder unit test provides the following functions (except A12):

1. H264 4 videos decode
2. H264 4 videos trickplay (pause, resume, Step, time search)
3. H264 4 videos zoom

6.5.1 Video Multi-channel Decode: Command List

1. Init

   [Command sequence]
   Step 1: **t videomultidec init**

2. Start decoding

   [Command sequence]

   Rotation is supported in A9 and A9s, but not supported in A12.

   Step 1: **t videomultidec start [filePath] [videoWidth] [videoHeight] [startTime] [speed] [dir] [rotate]**
   filePath: On A9, the full path of OUT_0000~OUT_0003.nhnt/.h264/.udta files.
   On A9s and A12, the full path of OUT_0000_01~OUT_0000_03.nhnt/.h264/.udta files.
   videoWidth: Video width (Note: Only supports 1280x720 clips)
   videoHeight: Video height (Note: Only supports 1280x720 clips)
   startTime: Time (in ms) of the first displayed video frame
   speed: 128 – 0.5x, 256 – 1x, 512 – 2x (Note: In A9 and A9s, only supports 1x speed. On A12, supports Nx speed)
   dir: 0: Forward (Note: Backward is not supported)
   rotate: 0 ~ 7, 0: No rotation, 1: Horizontal flip, ...etc

3. Zoom during decoding

   [Command sequence]

   Zooming is supported in A9 and A9s, but not supported on A12.

   Step 1: **t videomultidec zoom [videoWidth] [videoHeight] [cropOffsetX] [cropOffsetY] [cropWidth] [cropHeight]**
   videoWidth: Video width
   videoHeight: Video height
   cropOffsetX: Cropping area horizontal offset
   cropOffsetY: Cropping area vertical offset
   cropWidth: Cropping area width
   cropHeight: Cropping area height

4. Pause during decoding

   [Command sequence]
   Step 1: **t videomultidec pause**

5. Resume during pause

   [Command sequence]
   Step 1: **t videomultidec resume**

6. Step ahead during pause

   [Command sequence]
   Step 1: **t videomultidec step**

7. Stop decoding

   [Command sequence]
   Step 1: **t videomultidec stop**

8. Exit multiple channel video decode mode

   [Command sequence]
   Step 1: **t videomultidec exit**

6.5.2 Video Multi-channel Decode: Golden Data

Please check MWUnitTest\GoldenData\Decode\Video Multiple Decode for golden data.
Clip resolution/frame rate is noted on the folder name.

6.5.3 Video Multi-channel Decode: Unit Test

1. General forward 1x decode

   [Command sequences]
   Step 1: Put files at SD root.
   In A9 and A9s, only 720p files are supported.
   Step 2: **t videomultidec init**
   Step 3: **t videomultidec start** [FilePath] 1280 720 0 256 0 0 0 0 0
   Step 4: **t videomultidec stop**

[Output]

Four videos tiled on screen plays forward at normal speed in Step 3.

2. Time search

[Command sequences]

Step 1: Put files at SD root.
In A9 and A9s, only 720p files are supported.
Step 2: **t videomultidec init**
Step 3: **t videomultidec start** [FilePath] 1280 720 3000 256 0 0 0 0 0
Step 4: **t videomultidec stop**

[Output]

Four videos start from 5 seconds in Step 3.
It is normal that the videos delay for a while since the demux in unit test is not optimized.

3. Zoom

[Command sequences]

Step 1: Put files at SD root.
In A9 and A9s, only 720p files are supported.
Step 2: **t videomultidec init**
Step 3: **t videomultidec start** [FilePath] 1280 720 0 256 0 0 0 0 0
Step 4: **t videomultidec zoom** 1280 720 320 180 640 360
Step 5: t videomultidec stop

[Output]

Zoom effect in Step 4

4. Pause, Step and resume decoding

[Command sequences]

Step 1: Put files at SD root.
In A9 and A9s, only 720p files are supported.
Step 2: **t videomultidec init**
Step 3: **t videomultidec** start [FilePath] 1280 720 0 256 0 0 0 0 0
Step 4: **t videomultidec pause**
Step 5: **t videomultidec Step**
Step 6: **t videomultidec resume**
Step 7: **t videomultidec stop**

[Output]

Four videos pause in Step 4.
Four videos step ahead in Step 5.
Four videos resume in Step 6.

5. Rotate

[Command sequences]

Step 1: Put files at SD root.
In A9 and A9s, only 720p files are supported.
Step 2: **t videomultidec init**
Step 3: **t videomultidec start** [FilePath]1280 720 0 256 0 4 5 6 7
Step 4: **t videomultidec stop**

[Output]

Four videos rotating 90 degrees clockwise play forwards at normal speed in Step 3. The first video in the upper-left corner rotates 180 degrees clockwise. The second one in the upper-right corner rotates 180 degrees and flips horizontally. The third one in the lower-left corner rotates 270 degrees clockwise. The fourth one in the lower-left corner rotates 270 degrees clock wise and flips vertically.

6.6 Unit Tests: Still Encoder

The still encoder unit test contains the following tests:

1. Liveview
2. Runtime sensor mode switch
3. Single RAW capture
4. Single YUV conversion
5. Single JPEG encode
6. Single capture from RAW capture to JPEG out
7. Single raw 3A statistics
8. Quickview
9. Continuous single shot (Photo Every X Second)
10. Auto exposure bracket shot
11. Burst shot
12. Endless burst shot
13. Pre-capture
14. Digital zoom move
15. Digital zoom jump
16. Digital zoom stop
17. YUV interpolation

Note: Screennail's width is fixed at 960, its height is determined by the aspect ratio; and the thumbnail size is 160x120.

6.6.1 Still Encoder: Command List

1. Initialize still encoder unit test after booting

   t stillenc init [sensorID]

   Same as Section 6.2.1.1

2. Show encode table

   t stillenc encID

3. Liveview start

   t stillenc liveviewstart [EncID]

For OV4689 :
encID: 100 - 2016X1512 photo @ 960X720P30 Liveview
101 - 2688X1512 photo @ 960X54P30 Liveview

For IMX117 :
encID: 100 - 4000X3000 photo @ 960X720P30 Liveview
101 - 4096X2160 photo @ 960X540P30 Liveview

For IMX206 :
encID: 100 - 4608X3456 photo @ 960X720P30 Liveview
101 - 3840X2160 photo @ 960X540P30 Liveview
103 - 4608X2592 photo @ 960X540P30 Liveview

For AR0330_Parallel
encID: 101 – 2304X1296 photo @ 960X540P30 Liveview

For AR0230
encID: 101 – 1920X1080 photo @ 960X540P30 Liveview

For MN34120
encID: 100 – 4608X3456 photo @ 960X720P30 Liveview
encID: 101 – 4608X2592 photo @ 960X540P30 Liveview

For MN34229
encID: 100 – 1440X1080 photo @ 960X720P30 Liveview
encID: 101 – 1920X1080 photo @ 960X540P30 Liveview

For IMX377:
encID: 100 – 4000X3000 photo @ 960X720P30 Liveview
(A9S) 101 – 4096X2160 photo @ 960X540P30 Liveview
(A12)101 – 3840X2160 photo @ 960X540P30 Liveview

For IMX317: (A9S only)
encID: 101 – 4096X2160 photo @ 960X540P30 Liveview

For B5_OV9750
encID: 100 – 1280X960 photo @ 1280X960P30 Liveview
101 – 1280X720 photo @ 1280X720P30 Liveview

For IMX078
encID: 100 – 4000X3000 photo @ 960X720P30 Liveview
101 – 4096X2160 photo @ 960X540P30 Liveview

For OV2718
encID: 101 – 1920X1080 photo @ 960X540P30 Liveview

For AR0237
encID: 101 – 1920X1080 photo @ 960X540P30 Liveview

For IMX290
encID: 101 – 1920X1080 photo @ 960X540P30 Liveview

4. Runtime change liveview sensor mode

t stillenc modechange [encID]
Same as Liveview start

5. Single RAW capture

   t stillenc rawcap [encID] [iso] [cmpr]
   encID: Encode ID, please refer to **t stillenc encID**
   ISO: ISO mode, 0: HISO, 1: LISO, 3: MISO
   cmpr: RAW output is compressed or not
   Note: A9/A9S only supports HISO/LISO currently (A12 supports H/M/LISO).
   Note: A12 PipVin capture can only support uncompressed Raw.

6. Single RAW to YUV conversion

   t stillenc yuvconv [inputID] [encID] [iso] [cmpr][bits][bayer]
   inputID: RAW ID number, generated by RAW capture NNNN.raw
   encID: encode ID, please refer to "t stillenc encID"
   ISO - ISO mode, 0: HISO, 1: LISO, 3: MISO
   cmpr: RAW input is compressed or not
   bits: RAW input data bits
   0 means to use the default value 14 bits
   bayer: RAW input bayer pattern, RG(0) BG(1) GR(2) GB(3)
   Note: A9/A9S only supports HISO/LISO currently (A12 supports H/M/LISO).

7. Single YUV to JPEG encoding

   t stillenc jpgenc [inputID] [encID] [iso] [tsize] [lp]
   inputID: YUV ID number, please refer to SD card root:\NNNN_m.y
   encID: Encode ID, please refer to "t stillenc encID"
   ISO: ISO mode, 0 – HISO, 1 – LISO, 3: MISO
   tsize: Target encode size in Kbyte unit
   0 means no BRC in defaulting w*h/2000 JPEG size
   lp: Re-encode loop in BRC
   Note: A9/A9S only supports HISO/LISO currently. (A12 supports H/M/LISO)

8. Single RAW capture to JPEG in a row

   t stillenc singlecapture [encID] [iso] [cmpr] [tsize] [lp]
   encID: Encode ID, please refer to "**t stillenc encID**"
   ISO - ISO mode, 0: HISO 1: LISO 2: FastMode*, 3: MISO
   cmpr: RAW output is compressed or not
   tsize: target encode size in Kbyte unit
   0 means no BRC in defaulting w*h/2000 jpeg size
   lp: Re-encode loop in BRC
   Note: A9/A9S only supports HISO/LISO currently. (A12 supports H/M/LISO).
   Note: A12 PipVin capture can only support uncompressed Raw.

9. Single RAW 3A statistics generation

   t stillenc raw2raw [inputID] [encID] [mode] [cmpr] [bits][bayer]
   inputID: Raw ID number, please refer to MWUnitTest\GoldenData\Encode\StillEnc\0001.RAW
   encID: Encode ID, please refer to "t stillenc encID"
   mode: Raw2raw execute mode
   0X1: Raw2raw 3A
   0X2: Raw2raw compress/uncompress (Not supported yet)
   cmpr: RAW input is compressed or not
   bits: RAW input data bits
   bayer: RAW input bayer pattern, RG(0) BG(1) GR(2) GB(3)

10. Quickview Enable

t stillenc quickview [qvMode]

qvMode: Quickview display time

0: Disable quickview (default setting)
1: Quickview will show 1 second.
2: Quickview will show 3 seconds.
3: Quickview will show 5 seconds.
4: Quickview will show forever until a user types.

11. Quickview Stop

t stillenc quickviewstop

12. Continuous single shot

t stillenc singlecapturecont [encID] [capcnt] [timeLapse]

encID: Encode ID, please refer to **t stillenc encID**
capcnt: Desired capture number
timeLapse: Time interval in ms unit to trigger every shot (should >= 500)

13. Auto exposure bracket shot

t stillenc autoexposurebracketing [encID]

encID: Encode ID, please refer to **t stillenc encID**

14. Burst shot

t stillenc burstcapture [encID] [capcnt]
encID: Encode ID, please refer to **t stillenc encID**
capcnt: Desired capture number

15. Endless burst shot

t stillenc burstcapturecont [encID] [duration] [wb]

encID: Encode ID, please refer to **t stillenc encID**
durataion: Simulate how long shutter key is pressed(unit in ms, must be longer than raw cap time)
wb: Enable postWB or not

16. Pre-capture

t stillenc precapture [encID] [duration]

encID: Encode ID, please refer to **t stillenc encID**
duration: Simulate how long shutter key is pressed(unit in ms, must be longer than raw cap time)

17. Digital zoom jump in liveview

t stillenc dzjump [Step] [shiftFactorX] [shiftFactorY]
Step - A12: 0~486, A9S: 0~429, dzoom level
shiftFactorX: -1.0 ~ 1.0, shifting factor in horizontal. 0 means the center. shiftFactorY: -1.0 ~ 1.0, shifting factor in vertical. 0 means the center.

Note: On A12, still capture will follow liveview dzoom setting.

18. Digital zoom move in liveview

[Command sequence]
Step 1:
t stillenc dzmove [moveType] [speed] [distance] [shiftFactorX] [shiftFactorY]
moveType: 0~2, move type
0: Zoom in 1: Zoom out 2: Only shift
speed: 0~5, speed level
0: The fastest 5: The lowest distance - A12: 0 ~ 486, A9S: 0~429
When moveType is 0, it means increasing the dzoom level. When moveType is 1, it means decreasing the dzoom level. When moveType is 2, it means shifting the Steps.
shiftFactorX: -1.0~1.0, shifting factor in horizontal 0 means the center.
shiftFactorY: -1.0~1.0, shifting factor in vertical 0 means the center.

Note: The maximum valid shiftFactorX and shiftFactorY are different based on dzoom level.
Note: On A12, still capture will follow liveview dzoom setting.

19. Dzoom stop in liveview

t stillenc dzstop
Force stopping dzjump or dzmove when it executes

20. YUV interpolation

User can set the specific YUV window when YUV interpolation is needed, please make sure the aspect ratio is the same as the raw image.

t stillenc customyuv [mainW][mainH]
mainW: YUV width mainW: YUV height

Note: On A12, YUV interpolation only supported in single capture(6.6.2.5) and single YUV conversion(6.6.2.3).
Note: On A12, YUV interpolation has the limitation MIN (4xRawWidth, 4992).
*Note: On A9S, YUV interpolation can only be up to 2 * Width and 2* Height.*
Note: On A9S, please disable TV first due to performance limitation.

21. Encode rotation

t stillenc rotate [rotMode]
rotMode: Pre-defined rotation
0: No rotation
1: Horizontal flip
2: 90 clockwise
3: 90 clockwise then vertical flip
4: 180 clockwise
5: 180 clockwise then horizontal flip
6: 270 clockwise
7: 270 clockwise then vertical flip

Note: 1. For A12, encode rotation only takes effect on the final JPEG file.
2. For A12, please issue this command before capture.

22. Optical black correction (A12 only)

t stillenc ob [enable]
Note: Currently, only IMX206 sensor supports OB mode.

23. Select Vin (A12 DualChan HybridMode only)

    t stillenc vinsel [Vin_Ch]
    0: Main VIN
    1: Pip VIN

24. Enable low delay mode

    [A12]

    t stillenc lowdelay [LiveviewLowDelay]
    LiveviewLowDelay - 0: disabled, 1: vin<-> vout0 (LCD), 2: vin<-> vout1 (TV)
    Verification is same as VideoEnc

6.6.2 Still Encoder: Unit Test

1. Liveview start/stop & Runtime change resolution

   [Command sequence]
   Step 1: **t stillenc init [sensorID]**
   Step 2: **t stillenc liveviewstart [encID]**
   Step 3: **t stillenc modechange [encID]**
   Step 4: **t stillenc liveviewstop**

2. Single RAW capture

   [Command Sequence]
   Step 1: **t stillenc init [sensorID]**
   Step 2: **t stillenc liveviewstart [encID]**
   Step 3: **t stillenc rawcap [encID] [iso] [cmpr]**
   [Output]
   NNNN.raw at SD root

3. Single YUV conversion

   [Command Sequence]
   Step 1: **t stillenc init [sensorID]**
   Step 2: **t stillenc liveviewstart [encID]**
   Step 3: **t stillenc yuvconv [inputID] [encID] [iso] [cmpr] [bits][bayer]**

   [Output]
   A set of YUV file in SD card root naming is as follows:
   NNNN_m.y/NNNN_m.uv (main)
   NNNN_t.y/NNNN_t.uv (thumbnail)
   NNNN_s.y/NNNN_s.uv (screennail)
   Note: There is no way to see if cmpr really takes effect. Ensure that all test steps are passed.

4. Single JPEG encode

   [Command Sequence]
   Step 1: **t stillenc init [sensorID]**
   Step 2: **t stillenc liveviewstart [encID]**
   Step 3: **t stillenc jpgenc [inputID] [encID] [iso] [tsize] [lp]**
   [Output]
   A set of JPEG files in the SD card root naming are as follows:
   NNNN_m.jpeg (main)
   NNNN_t.jpeg (thumbnail)
   NNNN_s.jpeg (screennail)
   Note: A9/A9S only supports HISO/LISO currently. (A12 supports H/M/LISO).
   Note: If lp is set to 1 and tsize is set to a very small value, the resulting JPEG must be smaller than the one under lp = 0 and tsize = 0.

5. RAW capture to JPEG in a row

   [Command Sequence]
   Step 1: **t stillenc init [sensorID]**
   Step 2: **t stillenc liveviewstart [encID]**
   Step 3: **t stillenc singlecapture [encID] [iso] [cmpr] [tsize] [lp]**

   [Output]
   A set of JPEG files in the SD card root naming are as follows:
   xxxx_m.jpeg (main)
   xxxx_t.jpeg (thumbnail)
   xxxx_s.jpeg (screennail)

   Note: A9/A9S only supports HISO/LISO Currently. (A12 supports H/M/LISO).
   Note: There is no way to see if cmpr really takes effect. Ensure that all test steps are passed.
   Note: If lp is set to 1 and tsize is set to a very small value, the resulting JPEG must be smaller than the one under lp = 0 and tsize = 0.

6. Single raw 3A statistics

   [Command Sequence]
   Step 0: Put 0001.raw input SD card root
   Step 1: **t stillenc init [sensorID]**
   Step 2: **t stillenc liveviewstart [encID]**
   Step 3: **t stillenc raw2raw 0001 [encID] 1 1 0 0**
   [Output]
   A CFA ouput binary (C:\cfa_3a.bin)
   Note: This test sequence is only used to check if the flow is fine.

7. Quickview

   [Command Sequence]
   Step 1: **t stillenc init [sensorID]**
   Step 2: **t stillenc liveviewstart [encID]**
   Step 3: **t stillenc quickview [qvMode]**
   Step 4: **t stillenc singlecapture [encID] [iso] [cmpr] [tsize] [lp]**
   Step 5: **t stillenc quickviewstop**

   Note: There is no way to see if cmpr really takes effect. Ensure that all test steps are passed.
   Note: If lp is set to 1 and tsize is set to a very small value, the resulting JPEG must be smaller than the one under lp = 0 and tsize = 0.

8. Continuous single shot

   [Command Sequence]
   Step 1: **t stillenc init [sensorID]**
   Step 2: **t stillenc lvst [encID]**
   Step 3: **t stillenc singlecapturecont [encID] [capcnt] [timeLapse]**

   [Output]
   A group of JPEG files in the SD card that are named as follows:
   0001_m.jpg ~ 000X_m.jpg (main)
   0001_s.jpg ~ 000X_s.jpg (screennail)
   0001_t.jpg ~ 000X_t.jpg (thumbnail)
   X depends on [capcnt]

9. Auto exposure bracket shot

   [Command Sequence]
   Step 1: **t stillenc init [sensorID]**
   Step 2: **t stillenc lvst [encID]**
   Step 3: **t stillenc autoexposurebracketing [encID]**
   [Output]
   A group of jpeg files in SD card that are named as follows:
   0001_m.jpg ~ 0003_m.jpg (main)
   0001_s.jpg ~ 0003_s.jpg (screennail)
   0001_t.jpg ~ 0003_t.jpg (thumbnail)
   It will generate three groups of JPEG files.

10. Burst shot

    [Command Sequence]
    Step 1: **t stillenc init [sensorID]**
    Step 2: **t stillenc lvst [encID]**
    Step 3: **t stillenc dumpskip 14**
    Step 4: **t stillenc burstcapture [encID] [capcnt]**
    [Output]
    A group of JPEG files in the SD card that are named as follows:
    0001_m.jpg ~ 000X_m.jpg (main)
    0001_s.jpg ~ 000X_s.jpg (screennail)
    0001_t.jpg ~ 000X_t.jpg (thumbnail)
    X depends on [capcnt]

11. Endless burst shot

    [Command Sequence]
    Step 1: **t stillenc init [sensorID]**
    Step 2: **t stillenc lvst [encID]**
    Step 3: **t stillenc dumpskip 14**
    Step 4: **t stillenc burstcapturecont [encID][duration][wb]**
    [Output]
    A group of JPEG files in the SD card that is named as follows:
    0001_m.jpg ~ 000X_m.jpg (main)
    0001_s.jpg ~ 000X_s.jpg (screennail)
    0001_t.jpg ~ 000X_t.jpg (thumbnail)
    X depends on duration, if duration is larger, it will generate more JPEG files.

12. Pre-capture

[Command Sequence]
Step 1: **t stillenc init [sensorID]**
Step 2: **t stillenc lvst [encID]**
Step 3: **t stillenc precapture [encID][timeLapse]**
[Output]
A group of JPEG files in the SD card that is named as follows:
0001_m.jpg ~ 0005_m.jpg (main)
0001_s.jpg ~ 0005_s.jpg (screennail)
0001_t.jpg ~ 0005_t.jpg (thumbnail)

It will generate five JPEG files. Two of them are captured before the shutter is released and three of them after the shutter is released.

13. Still encode rotation

[Command Sequence]
Step 1: **t stillenc init [sensorID]**
Step 2: **t stillenc lvst [encID]**
Step 3: **t stillenc rotate [rotMode]**
Step 4: Perform any still capture (for example, single capture, please refer to Step 3 of Section 6.6.2.5)

[Output]
A pair of JPEG files will be produced.

14. Still encode optical black correction

[Command Sequence]
Step 1: **t stillenc init [sensorID]**
Step 2: **t stillenc ob [enable]**
Step 3: **t stillenc lvst [encID]**
Step 4: Perform any still capture (for example, single capture, please refer to Step 3 of Section 6.6.2.5)

[Output]
A pair of JPEGs will be produced.

15. Select Vin source for different vin still capture

[Command Sequence]
Step 1: **t stillenc vinsel**
Step 2: **t stillenc init [sensorID]**
Step 3: **t stillenc lvproc 1 1**
Step 4: **t stillenc liveviewstart [encID]**
Step 5: Preform still capture

[Output]
A pair of JPEGs will be produced.

6.7 Unit Tests: Still Decoder

SDK6 still decoder provides the following functions:

1. JPEG decode
2. H264 I frame decode
3. Buffer rescale
4. Buffer blending
5. Buffer display
6. Digital filters

6.7.1 Still Decoder: Command List

1. To initialize still decoder after booting

   Enable LCD only
   t stilldec init 3
   t stilldec ch 0
   Enable TV only
   t stilldec init 3
   t stilldec ch 1
   Dual Vout method 1
   t stilldec init 3
   t stilldec ch 0
   t stilldec ch 1
   Dual Vout method 2
   t stilldec init

2. To feed data

   t stilldec feed [fileName] [mediaType]
   fileName: full path of the file
   mediaType: 0 – JPEG, 1 – H264

3. To start decoding to a target cache

   t stilldec dec [mediaWidth] [mediaHeight] [cacheId]
   mediaWidth: Media width
   mediaHeight: Media height
   cacheId: Target cache ID, 0 ~ 31

4. To show motion on the screen

   t stilldec motion_test [cacheId]
   cacheId: Cache ID to demonstrate motion, 0 ~ 31

   Note: Only supported on LCD.

5. To show image on screen

   t stilldec disp [cacheId] [channelId]
   cacheId: Cache ID to be displayed, 0 ~ 31
   channelId - Channel ID to be displayed on, 0: LCD, 1: TV
   Note: The image is fitted to the screen regardless of its aspect ratio.

6. To test the blend and show the image on the screen

   t stilldec blend_test [srcCacheId1] [srcCacheId2] [destCacheId1] [alpha]
   srcCacheId1: Source cache ID #1, 0 ~ 31
   srcCacheId2: Source cache ID #2, 0 ~ 31
   destCacheId1: Target cache ID for storing blend result, 0 ~ 31
   alpha: Alpha value used to blend (0~255)
   Note: Only supported on LCD. The image is fitted to the screen regardless of its aspect ratio.
   Note: Only supported in A9/A9S

7. To test the blend by alpha map and show the image on the screen

   t stilldec map_blend_test [srcCacheId1] [srcCacheId2] [destCacheId1] [alphaMapCacheId]
   srcCacheId1: Source cache ID #1, 0 ~ 31
   srcCacheId2: Source cache ID #2, 0 ~ 31
   destCacheId1: Target cache ID for storing blend result, 0 ~ 31
   alphaMapCacheId: Cache ID for storing alpha map, 0 ~ 31
   Note: Only supported on LCD. The image is fitted to the screen regardless of its aspect ratio.

8. To enable CVBS

   t stilldec cvbs 1 (must send before init)

6.7.2 Still Decoder: Golden Data

1. MWUnitTest\GoldenData\Decode\StillDecode\

   File: 1.jpg
   Width: 4000
   Height: 3000
   File: 2.jpg
   Width: 240
   Height: 180

6.7.3 Still Decoder: Unit Test

1. Normal decode on LCD

   [Command Sequence]
   Step 1: **t stilldec init 3**
   Step 2: **t stilldec ch 0**
   Step 3: **t stilldec feed [fileName] [mediaType]**
   Step 4: **t stilldec dec [mediaWidth] [mediaHeight] [cacheId]**
   Step 5: **t stilldec disp [cacheId] 0**
   [Output]:
   Input image is shown on screen at Step 5

2. Normal decode on TV

   [Command Sequence]
   Step 1: **t stilldec init 3**
   Step 2: **t stilldec ch 1**
   Step 3: **t stilldec feed [fileName] [mediaType]**
   Step 4: **t stilldec dec [mediaWidth] [mediaHeight] [cacheId]**
   Step 5: **t stilldec disp [cacheId] 1**

   [Output]:
   Input image is shown on screen at Step 5.

3. Normal decode on dual VOUT

   [Command Sequence]
   Step 1: **t stilldec init**
   Step 2: **t stilldec feed [fileName] [mediaType]**
   Step 3: **t stilldec dec [mediaWidth] [mediaHeight] [cacheId]**
   Step 4: **t stilldec disp [cacheId] 0**
   Step 5: **t stilldec disp [cacheId] 1**

   [Output]:
   Input image is shown on screen at Step 5

4. Normal decode with motion

   [Command Sequence]
   Step 1: **t stilldec init 3**
   Step 2: **t stilldec ch 0**
   Step 3: **t stilldec feed [fileName] [mediaType]**
   Step 4: **t stilldec dec [mediaWidth] [mediaHeight] [cacheId]**
   Step 5: **t stilldec motion_test [cacheId]**
   [Output]:
   Input image is shown on motion on screen.

5. Blend function

   [Command Sequence]
   Step 1: **t stilldec init 3**
   Step 2: **t stilldec ch 0**
   Step 3: **t stilldec feed [fileName_1] [mediaType]**
   Step 4: **t stilldec dec [mediaWidth] [mediaHeight] [cacheId_1]**
   Step 5: **t stilldec feed [fileName_2] [mediaType]**
   Step 6: **t stilldec dec [mediaWidth] [mediaHeight] [cacheId_2]**
   Step 7: **t stilldec blend_test [cacheId_1] [cacheId_2] [cacheId_3] [alpha]**

   [Output]:
   Show the blended image on screen at Step 7.

6. Blend function by alpha map

   [Command Sequence]
   Step 1: **t stilldec init 3**
   Step 2: **t stilldec ch 0**
   Step 3: **t stilldec feed [fileName_1] [mediaType]**
   Step 4: **t stilldec dec [mediaWidth] [mediaHeight] [cacheId_1]**
   Step 5: **t stilldec feed [fileName_2] [mediaType]**
   Step 6: **t stilldec dec [mediaWidth] [mediaHeight] [cacheId_2]**

Step 7: **t stilldec map_blend_test [cacheId_1] [cacheId_2] [cacheId_3] [cacheId_4]**

[Output]:
Show the blended image on screen at Step 7

6.8 Unit Tests: MP4 Muxer

The MP4 muxer unit test contains following tests:

1. Video only MP4 muxing
2. Mux two streams at the same time (using the same source)
3. Mux four streams at the same time (using the same source)
4. A/V MP4 muxing
5. Mux two different video streams into one file

6.8.1 MP4 Muxer: Unit Test

1. Video only MP4 Muxing

   [Command sequence]
   Step 0: Prepare related files to SD card root:
   MWUnitTest\GoldenData\Muxer\Mp4Mux\OUT_0000.udta
   MWUnitTest\GoldenData\Muxer\Mp4Mux\OUT_0000.nhnt
   MWUnitTest\GoldenData\Muxer\Mp4Mux\OUT_0000.h264
   Step 1: Initialize MP4 Muxer. Mp4 Muxer can be initialized only once after booting.
   t mp4mux init
   If the test passed, the console will show "[SUCCESS] AmpUT_Mp4Mux_init".
   Step 2: Start the unit test of MP4 Muxer
   t mp4mux start [frameRateDivisor]
   (The frameRateDivisor is a parameter for converting a high frame rate bitstream to a low frame rate one. For example, a 60fps clip with frameRateDivisor 2 implies a 30fps resulted clip, i.e, playing the resulted clip would see that it is slow-motioned and its length is doubled.)
   If the test passed, the console will show "[SUCCESS] AmpUT_Mp4Mux : End".

   [Output]
   Test.mp4. Golden data is
   MWUnitTest\GoldenData\Muxer\Mp4Mux\TEST.MP4

2. Mux two streams at the same time

   [Command sequence]
   Step 0: Prepare related files to SD card root
   MWUnitTest\GoldenData\Muxer\Mp4Mux2s\OUT_0000.udta
   MWUnitTest\GoldenData\Muxer\Mp4Mux2s\OUT_0000.nhnt
   MWUnitTest\GoldenData\Muxer\Mp4Mux2s\OUT_0000.h264
   Step 1: Initiate mp4 mux unit test. Mp4 Muxer can be initialized only once after booting.
   t mp4mux2s init
   If the test passed, the console will show "[SUCCESS] AmpUT_Mp4Mux2s_init".
   Step 2: Start the unit test of MP4 Muxer
   t mp4mux2s start
   If the test passed, the console will show "[SUCCESS] AmpUT_Mp4Mux2s : End".

[Output]

TEST.MP4, TEST2.MP4. Golden data is at
MWUnitTest\GoldenData\Muxer\Mp4Mux2s\TEST.MP4
MWUnitTest\GoldenData\Muxer\Mp4Mux2s\TEST2.MP4

3. Mux four streams at the same time

[Command sequence]

Step 0: Prepare related files to SD card root
MWUnitTest\GoldenData\Muxer\Mp4Mux4s\OUT_0000.udta
MWUnitTest\GoldenData\Muxer\Mp4Mux4s\OUT_0000.nhnt
MWUnitTest\GoldenData\Muxer\Mp4Mux4s\OUT_0000.h264
Step 1: Initiate mp4 mux unit test. Mp4 Muxer can be initialized only once after booting.
t mp4mux4s init
If the test passed, the console will show “[SUCCESS] AmpUT_Mp4Mux4s_init”.
Step 2: Start the unit test of MP4 Muxer
t mp4mux4s start
If the test passed, the console will show “[SUCCESS] AmpUT_Mp4Mux4s: End”.

[Output]

TEST.MP4, TEST2.MP4, TEST3.MP4, TEST4.MP4. Golden data is at:
MWUnitTest\GoldenData\Muxer\Mp4Mux4s\TEST.MP4
MWUnitTest\GoldenData\Muxer\Mp4Mux4s\TEST2.MP4
MWUnitTest\GoldenData\Muxer\Mp4Mux4s\TEST3.MP4
MWUnitTest\GoldenData\Muxer\Mp4Mux4s\TEST4.MP4

4. A/V MP4 muxing

[Command sequence]

Step 0: Prepare related files to SD card root:
MWUnitTest\GoldenData\Muxer\Mp4MuxAV\OUT_0000.udta
MWUnitTest\GoldenData\Muxer\Mp4MuxAV\OUT_0000.nhnt
MWUnitTest\GoldenData\Muxer\Mp4MuxAV\OUT_0000.h264
MWUnitTest\GoldenData\Muxer\Mp4MuxAV\Pri_0000.aac
Step 1: Initiate mp4 mux unit test. Mp4 Muxer can be initialized only once after booting.
t mp4muxav init
If the test passed, the console will show “[SUCCESS] AmpUT_Mp4MuxAV_init”.
Step 2: Start the unit test of MP4 Muxer
t mp4muxav start
If the test passed, the console will show “[SUCCESS] AmpUT_Mp4MuxAV : End”.

[Output]

TEST.MP4, Golden data is at:
MWUnitTest\GoldenData\Muxer\Mp4MuxAV\TEST.MP4

5. Mux two different video streams into one file.

[Command sequence]

Step 0: Prepare related files to SD card root:
MWUnitTest\GoldenData\Muxer\Mp4MuxDual\OUT_0000.udta
MWUnitTest\GoldenData\Muxer\Mp4MuxDual\OUT_0000.nhnt
MWUnitTest\GoldenData\Muxer\Mp4MuxDual\OUT_0000.h264
MWUnitTest\GoldenData\Muxer\Mp4MuxDual\OUT_0002.udta
MWUnitTest\GoldenData\Muxer\Mp4MuxDual\OUT_0002.nhnt
MWUnitTest\GoldenData\Muxer\Mp4MuxDual\OUT_0002h264
Step 1: Initiate mp4 mux unit test. Mp4 Muxer can be initialized only once after booting.
t mp4muxdual init

If the test passed, the console will show “[SUCCESS] AmpUT_Mp4MuxDual_init”.

Step 2: Start the unit test of MP4 Muxer

t mp4muxdual start

If the test is passed, the console will show “[SUCCESS] AmpUT_Mp4MuxDual : End”.

[Output]

TEST.MP4. Golden data is at:
MWUnitTest\GoldenData\Muxer\Mp4MuxDual\TEST.MP4

6.9 Unit Tests: MP4 Demuxer

The MP4 demuxer unit test contains the following tests:

1. Video only Mp4 file demuxing
2. A/V Mp4 file demuxing

6.9.1 MP4 Demuxer: Unit Test

1. Video only MP4 file demuxing

[Command sequence]

Step 0: Prepare following file to SD root
MWUnitTest\GoldenData\Demuxer\Mp4Dmx\TEST.MP4

Step 1: Initialize Mp4 Demuxer. Mp4 Demuxer can be initialized only once after booting.

t mp4dmx init

If the test passed, the console will show “[SUCCESS] AmpUT_Mp4Dmx_DmxInit”.

Step 2: Start the unit test of MP4 Demuxer

t mp4dmx start 0 0 0

If the test passed, the console will show “[SUCCESS] AmpUT_Mp4Dmx : End”.

Step 3: Exit the unit test of MP4 Demuxer

t mp4dmx exit

If the test passed, the console will show “[SUCCESS] AmpUT_Mp4Dmx_exit”.

[Output]

OUT_0001.H264 and OUT_0001.nhnt
Golden data is at:
MWUnitTest\GoldenData\Demuxer\Mp4Dmx\OUT_0001.H264
MWUnitTest\GoldenData\Demuxer\Mp4Dmx\OUT_0001.NHNT

2. A/V MP4 file demuxing

[Command sequence]

Step 0: Prepare related files to SD card root:
MWUnitTest\GoldenData\Demuxer\Mp4DmxAV\TEST.MP4

Step 1: Initialize Mp4 Demuxer. Mp4 Demuxer can be initialized only once after booting.

t mp4dmxav init

If the test passed, the console will show “[SUCCESS] AmpUT_Mp4DmxAV_DmxInit”.

Step 2: Start the unit test of MP4 Demuxer

t mp4dmxav start 0 0 0

If the test passed, the console will show “[SUCCESS] AmpUT_Mp4DmxAV : End”.

Step 3: Exit the unit test of MP4 Demuxer

t mp4dmxav exit

If the test passed, the console will show “[SUCCESS] AmpUT_Mp4DmxAV_exit”.

[Output]

OUT_0001.H264, OUT_0001.nhnt, and AUDIO.raw

Golden data is at

MWUnitTest\GoldenData\Demuxer\Mp4DmxAV\OUT_0001.H264

MWUnitTest\GoldenData\Demuxer\Mp4DmxAV\OUT_0001.NHNT

MWUnitTest\GoldenData\Demuxer\Mp4DmxAV\AUDIO.raw

6.10 Unit Tests: EXIF Muxer

Exifmux unit test contains the following tests:

1. Mux thumbnail, screennail, main JPEG into one EXIF compliant JPEG file.

6.10.1 EXIF Muxer: Unit Test

[Command sequence]

Step 0: Prepare related file to SD card root:

MWUnitTest\GoldenData\Muxer\ExifMux\AMBA_MAIN.jpg

MWUnitTest\GoldenData\Muxer\ExifMux\AMBA_SCRN.jpg

MWUnitTest\GoldenData\Muxer\ExifMux\AMBA_THMB.jpg

Step 1: Initialize exifmux. Exif Muxer can be initialized only once after booting.

t exifmux init

If the test passed, the console will show “[SUCCESS] AmpUT_ExifMux_init”.

Step 2: Start processing mux image

t exifmux start

If the test passed, the console will show “[SUCCESS] AmpUT_ExifMux : End”.

[Output]

Compare TEST1.JPG to

MWUnitTest\GoldenData\Muxer\ExifMux\TEST1.JPG

6.11 Unit Tests: EXIF Demuxer

The Exifdemux unit test contains the following tests:

1. Demux an exif-jpg file

6.11.1 EXIF Demuxer: Unit Tests

1. JPEG demuxing

[Command sequences]

Step 1: Prepare the image file TEST.jpg to SD card root

MWUnitTest\GoldenData\Demuxer\ExifDmx\TEST.JPG

Step 2: To initialize exifdmx. Exif Demuxer can be initialized only once after booting.

t exifdmx init

Step 3: To start parsing TEST.jpg:

t exifdmx feed [frame_no]

frame_no: 0 – Fullview, 1 – Thumbnail, 2 – Screennail

Step 4: To get specified target to DEC.jpg:
t exifdmx dec

[Output]
Compare DEC.JPG to
MWUnitTest\GoldenData\Demuxer\ExifDmx\0\DEC.JPG (fullview)
MWUnitTest\GoldenData\Demuxer\ExifDmx\1\DEC.JPG (thumbnail)
MWUnitTest\GoldenData\Demuxer\ExifDmx\2\DEC.JPG (screennail)

6.12 Unit Tests: CFS

Cached file system unit test contains following tests:

1. Synchronous mode test
2. Asynchronous mode test
3. Stress test

They will test fopen, fwrite, fsync, ftell, flen, fseek, fread, fclose operations for write mode (“w+”) and fopen, ftell, flen, fseek, fread, fclose operations for read mode (“r”).

6.12.1 CFS: Command List

1. CFS test

   t cfs sync_auto|async_auto |stress_test
   sync_auto: Synchronous mode auto test
   async_auto: Asynchronous mode auto test
   stress_test: **Stress test**

6.12.2 CFS: Unit Test

1. Synchronous mode test

   [Command sequence]
   Step 1: **t cfs sync_auto**

   [Output]
   AMBA_SYNC1.txt (16 bytes) contains “This is a test”.
   AMBA_SYNC2.txt (2097152 bytes) contains binary codes: 00 01 … FE FF 00 01 … FE FF
   If CFS passes the test, the console will show “[SUCCESS] CFS sync_auto” at the end of all message lines; otherwise, the console will show “[FAIL] CFS sync_auto”.

2. Asynchronous mode test

   [Command sequence]
   Step 1: **t cfs async_auto**

   [Output]
   AMBA_ASYNC1.txt (16 bytes) contains “This is a test.”
   AMBA_ASYNC2.txt (2097152 bytes) contains binary codes: 00 01 … FE FF 00 01 … FE FF
   If CFS passes the test, the console will show “[SUCCESS] CFS async_auto” at the end of all message lines; otherwise, the console will show “[FAIL] CFS async_auto”.

3. Stress test

   [Command sequence]
   Step 1: **t cfs stress_test [FileSize(1-1024MiB)]**

   [Output]
   UT_CFS_STRESS_SYNC.txt and UT_CFS_STRESS_ASYNC.txt, both are composed by multiple consecutive binary codes from 00 to FA, for example, “00 01 ... F9 FA 00 01 ... F9 FA 00 01 02”.
   If CFS passes the test, the console will show “[SUCCESS] CFS stress_test” at the end of all message lines; otherwise; the console will show “[FAIL] CFS stress_test”.

6.13 Unit Tests: Audio Encoder

The audio encoder unit test contains the following tests:

1. Create Audio encoder instance
2. Delete Audio encoder instance
3. Encoding a voice clip
4. Multiple input encoding
5. Multiple stream encoding

6.13.1 Audio Encoder: Command List

1. Initialize audio encoder unit test after boot:

   t audioenc init

2. Create an audio encoder instance that only supports one instance in UnitTest

   t audioenc create [priEncType] [secEncType]
   EncType: encode type
   0: AAC
   1: PCM
   2: AAC_PLUS (except A12)
   3: AAC_PLUS_V2 (except A12)
   4: ADPCM (except A12)
   5: AC3 (except A12)
   6: MPEG3 (except A12)
   7: OPUS (except A12)
   priEncType: The first audio clip’s type. 0 – AAC, 1 – PCM
   secEncType: The second audio clip’s type. 0 – AAC, 1 – PCM

3. Start encoding

   t audioenc encstart [time]
   time: Encode time in msec unit, 0 means continuous encode

4. Stop encoding

   t audioenc encstop

5. Delete an audio encoder instance

   t audioenc delete

6. Multiple input/stream enable

   t audioenc multi [mode]

   mode: 0X0: Single input/single stream encode
   0X1: Multi-input encoding (current UnitTest supports 2 inputs)
   0X2: Multi-stream encoding (current UnitTest supports 2 streams)

6.13.2 Audio Encoder: Unit Test

1. Basic audio encoding

   [Command sequence]
   Step 1: **t audioenc init**
   Step 2: **t audioenc create [priEncType] 0**
   Step 3: **t audioenc encstart [time]**
   Step 4: **t audioenc encstop** (if [time] is not set)
   Step 5: **t audioenc delete**

   [Output]
   Pri_XXXX.aac/pcm file in SD root

2. Multi-input encode

   The test simulates encoder mixing two audio inputs into one audio clip.
   [Command Sequence]
   Step 1: **t audioenc init**
   Step 2: **t audioenc multi 1**
   Step 3: **t audioenc create [priEncType]**
   Step 4: **t audioenc encstart [time]**
   [Output]
   Pri_xxxx.aac/pcm in SD card root with higher volume than basic encoding test clip

3. Multi-stream encode

   The test simulates encoder encoding two audio inputs into two audio clips at the same time.
   [Command Sequence]
   Step 1: **t audioenc init**
   Step 2: **t audioenc multi 2**
   Step 3: **t audioenc create [priEncType] [secEncType]**
   Step 4: **t audioenc encstart [time]**
   [Output]
   Pri_xxxx.aac/pcm and Sec_xxxx.aac/pcm in SD root

6.14 Unit Tests: OSD

OSD unit test is used to verify the OSD function.
In OSD Unit test, the user supports the OSD paint (with test pattern) and window setup.

6.14.1 OSD: Command List

1. Initialize OSD, two buffers are allocated at this stage

   t osd init [osd bits] [rgb mode]
   OSD bits: The value should be 8/16/32 to indicate the OSD format.
   Rgb mode: Using RGB OSD or YUV OSD

2. Set and activate OSD on given channels

   t osd setBuf [channel] [bufIdx]
   channel: The channel OSD display on. 0: LCD, 1: TV
   bufIdx: The buffer index (0/1) for default OSD, buffer 0 is empty and buffer 1 is light gray.

3. Paint on OSD buffer

   t osd paintBuf [bufIdx] [test color]
   bufIdx: The buffer index (0/1)
   test color: The color will be draw on part of the OSD buffer.
   The Step will paint test pattern on buffer (4 color bar GRGB)

4. Use the setBuf command to show the painted buffer

   t osd setBuf [channel] [bufIdx]
   channel: The channel OSD display on. 0 – LCD, 1 – TV
   bufIdx: The buffer index (0/1)

5. Change source/target OSD window configuration

   t osd setWindow [channel] [crop x] [crop y] [crop w] [crop h] [display x]
   [display y] [display w] [display h]
   channel - The window to configure. 0: LCD, 1: TV
   crop x/y/w/h: Part to be cropped from buffer
   display x/y/w/h: Target location on output. The size of the target area cannot be larger than the size of output devices.
   For example, **t osd setWindow 0 240 0 480 480 0 0 480 240** will crop a part of buffer (which is 960x480) and scale it to left top with 1/4 size.

6. Enable CVBS for fchan (A9S only)

   t osd cvbs [enable]
   enable - 0: Disable 1: Enable

6.14.2 OSD: Unit Test

For LCD WDF9648W (960X480)]

Step 0: Boot to Dec or Enc, please refer to 6.2.2 or 6.4.3 (when cvbs is used, please type “t osd cvbs 1” at the beginning)
t vd init
Step 1: Initialize OSD module
t osd init 16 1
Step 2: Paint Buf
t osd paintBuf 0 0xFFFF

Step 3: Show the OSD on the LCD
t osd start 0
Step 4: Test same size with offset
t osd setWindow 0 240 0 480 240 60 60 480 240
t osd setWindow 0 240 0 480 240 120 120 480 240
Step 5: Test scale down
t osd setWindow 0 0 0 960 480 60 60 480 240
Step 6: Test the scale up
t osd setWindow 0 240 0 480 240 0 0 960 480

[For LCD T30P61 (960X240)]

Step 0: Init the Display module and change the LCD configuration
t disp init
t disp dchan 1 3
Step 1: Boot to Dec or Enc, please refer to 6.2.2 or 6.4.3 (when cvbs is used, please type “t osd cvbs 1” before the step)
t vd init
Step 2: Initialize the OSD module
t osd init 16 1
Step 3: Show the OSD buffer on the LCD
t osd start 0
Step 4: Paint the OSD buffer
t osd paintBuf 0 0xFFFF
Step 5: Change the window offset
t osd setWindow 0 240 0 480 120 60 60 480 120
t osd setWindow 0 240 0 480 120 120 120 480 120
Step 6: Scale down the window
t osd setWindow 0 0 0 960 240 0 0 480 120
Step 7: Scale up the window
t osd setWindow 0 0 0 480 120 0 0 960 240

[For TV CVBS (720X480)]

Step 0: Enable CVBS
t osd cvbs 1
Step 1: Boot to Dec or Enc, please refer to 6.2.2 or 6.4.3 (when cvbs is used, please type “t osd cvbs 1” before the step)
t vd init
Step 2: Initialize the OSD module
t osd init 16 1
Step 3: Show the OSD buffer on the LCD
t osd start 1
Step 4: Paint the OSD buffer
t osd paintBuf 1 0xFFFF
Step 5: Change the window offset
t osd setWindow 1 240 0 480 240 45 60 360 240
t osd setWindow 1 240 0 480 240 90 120 360 240
Step 6: Scale down the window
t osd setWindow 1 0 0 960 480 0 0 360 240
Step 7: Scale up the window
t osd setWindow 1 240 0 480 240 0 0 720 480

6.15 Unit Tests: Audio Decoder

The audio decoder unit test contains following test:

1. Audio decoding

6.15.1 Audio Decoder: Command List

1. Initialize audio decoder

   t audiodec init

2. Audio decoding start

   t audiodec start [filename] [fileType] [sampleRate]
   filename: The full file path to the file
   fileType: pcm/aac
   sampleRate: Sample rate of file

3. Audio decoding stop

   t audiodec stop

6.15.2 Audio Decoder: Unit Test

1. Audio decoding

   [Command sequence]
   Step 0: Put the audio file in the SD card from MWUnitTest\GoldenData\Decode\AudioDecode.
   Step 1: **t audiodec init**
   Step 2: **t audiodec start [filename] [fileType] [sampleRate]**
   Step 3: **t audiodec stop**

6.16 Unit Tests: MOV Muxer

The Mov mux unit test contains the following tests:

1. MOV file muxing

6.16.1 MOV Muxer: Unit Test

1. Mov file muxing

   [Command sequence]
   Step 0: Prepare related file to SD root:
   MWUnitTest\GoldenData\Muxer\MovMux\OUT_0000.udta
   MWUnitTest\GoldenData\Muxer\MovMux\OUT_0000.nhnt
   MWUnitTest\GoldenData\Muxer\MovMux\OUT_0000.h264

Step 1: Initialize MOV Muxer. MOV Muxer can be initialized only once after booting.
t movmux init
If the test passed, the console will show “[SUCCESS] AmpUT_MovMux_init”.
Step 2: Start the unit test of the MOV Muxer
t movmux start
If the test passed, the console will show “[SUCCESS] AmpUT_MovMux : End”.

[Output]
Test.MOV. Golden data is:
MWUnitTest\GoldenData\Muxer\MovMux\TEST.MOV

2. Mov + LPCM file muxing

[Command sequence]
Step 0: Prepare related file to SD root:
MWUnitTest\GoldenData\Muxer\MovMux\LPCM\OUT_0000.udta
MWUnitTest\GoldenData\Muxer\MovMux\LPCM\OUT_0000.nhnt
MWUnitTest\GoldenData\Muxer\MovMux\LPCM\OUT_0000.h264
MWUnitTest\GoldenData\Muxer\MovMux\LPCM\Pri_0000.udta
MWUnitTest\GoldenData\Muxer\MovMux\LPCM\Pri_0000.nhnt
MWUnitTest\GoldenData\Muxer\MovMux\LPCM\Pri_0000.pcm
Step 1: Enable the audio type as LPCM
t movmux audio [type]
Type : 4 - LPCM
Then the console will show “MOV + LPCM”.
Step 2: Initialize MOV Muxer. MOV Muxer can be initialized only once after booting.
t movmux init
If the test passed, the console will show “[SUCCESS] AmpUT_MovMux_init”.
Step 3: Start the unit test of the MOV Muxer.
t movmux start
If the test passed, the console will show “[SUCCESS] AmpUT_MovMux : End”.

[Output]
Test.MOV. Golden data is: MWUnitTest\GoldenData\Muxer\MovMux\LPCM\TEST.MOV

6.17 Unit Tests: MOV Demuxer

The Mov demuxer unit test contains the following tests:
1. MOV file demuxing

6.17.1 MOV Demuxer: Unit test

1. Mov file demuxing

[Command sequence]
Step 0: Prepare following file to SD root
MWUnitTest\GoldenData\Demuxer\MovDmx\TEST.MOV
Step 1: Initialize MOV Demuxer. MOV Demuxer can be initialized only once after booting.
t movdmx init
If the test passed, the console will show “[SUCCESS] AmpUT_MovDmx_DmxInit”.
Step 2: Start the unit test of MOV Demuxer
t movdmx start 0 0 0
If the test passed, the console will show “[SUCCESS] AmpUT_MovDmx : End”.

Step 3: Exit the unit test of MOV Demuxer

t movdmx exit

If the test passed, the console will show “[SUCCESS] AmpUT_MovDmx_exit”.

[Output]

OUT_0001.H264 and OUT_0001.nhnt at SD root
Golden data is at:
MWUnitTest\GoldenData\Demuxer\MovDmx\OUT_0001.H264
MWUnitTest\GoldenData\Demuxer\MovDmx\OUT_0001.NHNT

6.18 Unit Tests: Beep

Beep is a simple function to play beep sound.

6.18.1 Beep: Unit Test

1. Beep sound test

   [Command sequence]
   Step 0: Put MWUnitTest\GoldenData\Decode\AudioDecode\48k.pcm in the SD root
   Step 1: **t be init**
   Step 2: **t be b C:\48k.pcm**

6.19 Unit Tests: Transcode (Supported only on A9/A9s)

The transcode (Xcode) unit test provides a test to transcode from a given h264 file to output h264 file in different resolutions.

6.19.1 Transcode: Command List

1. Initialization

   t xcode init

2. Start transcoding

   t xcode start [idx] [clipWidth] [clipHeight]
   idx: Source file index. Eg. 0000
   clipWidth: Video clip width
   clipHeight: Video clip height

3. Stop transcoding

   t xcode stop

6.19.2 Transcode: Unit Test

[Command sequence]

Step 0: Prepare golden out_0000.h264, out_0000.nhnt in SD root
Step 1: **t xcode init**
Step 2: **t xcode start**
Step 3: **t xcode stop**

[Output]

Output filename will be idx + 1000: out_1000.h264, out_1000.nhnt. Compare it with the golden result.

6.20 Unit Tests: Video Tuning

Video tuning unit test contains the following functions for testing:

1. Liveview
2. Runtime sensor mode switch
3. Liveview RAW capture
4. Liveview RAW capture and YUV conversion
5. Video RAW encode (RAW to H.264)

6.20.1 Video Tuning: Command list

1. Initialize unit test

   t videotuning init [sensorID]
   Same as Section 6.2.2.1

2. Liveview start

   t videotuning liveviewstart [modeID]
   modeID:
   Please refer to Section 6.2.1.2.

3. Liveview capture configuration

   t videotuning liveviewcapture config [enable]
   0: Disable, 1: Enable

4. Liveview capture start

   t videotuning liveviewcapture start [num][cmpr][yuv]
   num: Number of raw to capture, 0: Let MW decide
   cmpr: Capture compressed RAW or uncompressed
   yuv: Perform YUV conversion for each raw or not

5. Video RAW encode configuration

   t videotuning videorawenc config [inputID]
   inputID: ID of RAW capture

6. Video RAW encode start

   t videotuning videorawenc start

7. Video RAW encode feed the next raw file

   t videotuning videorawenc next [raw_path]
   raw_path: Setup the next raw file path and read it.
   If it is NULL, it will read the raw file path from the previous path.

8. Video RAW encode feeds the next ituner file

   t videotuning videorawenc ikne [ituner_path]
   ituner_path: Setup the next ituner file path and read it.
   If it is NULL, it will read the ituner file from the previous path.

9. Video RAW encode end

   t videotuning videorawenc end

10. Select VIN in the dual channel case when the user wants to select certain vin for raw capture.

    t videotuning vinsel [ChanIdx]
    ChanIdx: 0 is MainVin, 1 is PipVin

6.20.2 Video Tuning: Unit Test

1. Video RAW capture

   [Command sequence]
   Step 0: Boot up
   Step 1: **t videotuning init [sensorID]**
   Step 1.1: (Optional in DualChan) **t videotuning vinsel [ChanIdx]**
   Step 2: **t videotuning liveviewcapture config 1**
   Step 3: **t videotuning liveviewstart 13 (4K30P only)**
   Step 4: **t videotuning liveviewcapture start 0 0 0**

   [Output]
   LvCap_0001.bin
   LvCap_0001~00XX.RAW
   *XX is calculated based on the memory available

2. Video RAW capture and YUV conversion:

   [Command sequence]
   Step 0: Boot up
   Step 1: **t videotuning init [sensorID]**
   Step 1.1: (Optional in DualChan) **t videotuning vinsel [ChanIdx]**
   Step 2: **t videotuning liveviewcapture config 1**
   Step 3: **t videotuning liveviewstart 13 (4K30P only)**
   Step 4: **t videotuning liveviewcapture start 0 0 1**

   [Output]
   LvCap_0001.bin
   LvCap_0001~00XX.RAW
   LvCap_0001~00XX.y and LvCap_0001~00XX.uv are based on the memory available
   *XX is calculated based on the memory available.

3. Video RAW encode

   [A9/A9S]
   [Command sequence]
   Step 0: Reboot
   Step 1: **t videotuning init [sensorID]**
   Step 2: **t videotuning liveviewstart 13 (4K30P only)**
   Step 3: **t videotuning videorawenc config 0001 (0001 means LvCap_0001.bin)**
   Step 4: **t videotuning videorawenc lv 13**
   Step 5: **t videotuning videorawenc start**
   Step 6: **t videotuning videorawenc next 0**
   Step 7: loop Step 6 until LvCap_0001_0001.RAW ~ LvCap_0001_0010.RAW are all pushed
   Step 8: **t videotuning videorawenc end**
   Step 9: **t videotuning liveviewstop**
   Step 10: **t videotuning videorawenc config 0**
   Step 11: **t videotuning liveviewstart 13** (4K30P only)

   [Output]
   OUT_0000.H264
   Compare player decoded result and RAW picture's content

   [A12]

   [Command sequence]
   Step 0: Reboot
   Step 1: **t videotuning init [sensorID]**
   Step 2: **t videotuning lvproc 1 2 0**
   Step 3: **t videotuning videorawenc ikne [ituner_path]**
   Step 4: **t videotuning liveviewstart 0**
   Step 5: **t videotuning videorawenc start**
   Step 6: **t videotuning videorawenc next [raw_path]**
   Step 7: Loop Step 6 until the entire raw pictures are pushed
   Step 8: **t videotuning videorawenc end**
   Step 9: **t videotuning liveviewstop**
   Step 10: **t videotuning videorawenc config 0**
   Step 11: **t videotuning liveviewstart 1**
   [Output]
   OUT_0000.H264
   Compare player decoded result and RAW picture's content

6.21 Unit Tests: DCF

The DCF unit test contains the following test:

1. Auto: It will test (1) scan c:\DCIM to put all valid files to DCF, (2) add directories/files to DCF, and (3) remove directories/files from DCF

6.21.1 DCF: Unit Test

1. Auto

   [Command sequence]
   Step 0: Copy MWUnitTest\GoldenData\DCF\DCIM to the SD card root
   Step 1: **t dcf auto** c:\DCIM

[Output]

Result will show on console. Compare it with MWUnitTest\GoldenData\DCF\log.txt. If DCF passes the test, the console will show “[SUCCESS] DCF auto” at the end of all message lines. If DCF fails the test, the console will show “[FAIL] DCF auto”.

6.22 Unit Tests: Event Recording

The Event Recording unit test contains the following tests:

1. Event Recording (parking mode)

6.22.1 Event Recording: Unit Test

1. Event Recording

[Command sequence]

Step 1: Prepare the following to SD root
MWUnitTest\GoldenData\Muxer\EventRecoed\OUT_0000.udta
MWUnitTest\GoldenData\Muxer\EventRecoed\OUT_0000.nhnt
MWUnitTest\GoldenData\Muxer\EventRecoed\OUT_0000.h264

Step 2: Initialize the test of Event Recording. Event Recording can be initialized only once after booting.
t evtrec init
If the test passes, the console will show “[SUCCESS] AmpUT_EventRecord_init”.

Step 3: Start the unit test of Event Recording
t evtrec start
If the test passes, the console will show “[SUCCESS] AmpUT_EventRecord_start.

Step 4: Trigger Event Record before the unit test stops (supports multiple events, but the interval of two events must be longer than 1 minute)
t evtrec event
If an event command completes successfully, the console will show “[SUCCESS] TES TXX. MP4 is closed”. If the test passes, the console will show “[SUCCESS] AmpUT_EventRecord : End”.

[Output]

Every event generates a MP4 in the SD root with the name TESTXX.MP4. Play it with any player and check whether each one is 13 seconds long or not.

6.23 Unit Tests: Event Recording – Non-Overlapped

The event-recording of non-overlapped mode contains the following tests:

1. Event-recording of non-overlapped mode

6.23.1 Event Recording – Non-Overlapped: Unit Test

1. Event-recording of non-overlapped mode

[Command sequences]

Step 0 Prepare related files to SD card root
MWUnitTest\GoldenData\Muxer\EventRecoed_N\OUT_0000.udta
MWUnitTest\GoldenData\Muxer\EventRecoed_N\OUT_0000.nhnt
MWUnitTest\GoldenData\Muxer\EventRecoed_N\OUT_0000.h264

Step 1 Initialize EVENT-RECORD function. Event recording can be initialized only once after booting.
t evtrec_n init
If the test passed, the console will show "[SUCCESS] AmpUT_EventRecord_init".

Step 2 Record start:
t evtrec_n start
(The time interval between Step 2 and Step 3 should be longer than 10 seconds)
If the test passed, the console will show "[SUCCESS] AmpUT_EventRecord_start".

Step 3 Trigger Event Record before the unit test stops (supports multiple events, but the interval of the two events must be longer than 1 minute).
t evtrec_n event
Note: The event command will be skipped if it shows the log below.
AmpUT_EventRecord_event: sytem time < delay time
If an event command completes successfully, the console will show "[SUCCESS] EVNT0X. MP4 is closed". If the test passed, the console will show "[SUCCESS] AmpUT_EventRecord : End".

[Output]

EVNT0x.MP4
The duration of event files EVNT0x.Mp4 is 13 seconds.

6.24 Unit Tests: Event Recording – Overlapped

The event-recording of overlapped mode contains the following tests:

1. Event-recording of overlapped mode

6.24.1 Event Recording - Overlapped: Unit Test

[Command sequences]

Step 0: Prepare related files to SD card root
MWUnitTest\GoldenData\Muxer\EventRecoed_O\OUT_0000.udta
MWUnitTest\GoldenData\Muxer\EventRecoed_O\OUT_0000.nhnt
MWUnitTest\GoldenData\Muxer\EventRecoed_O\OUT_0000.h264

Step 1: Initialize EVENT-RECORD function. Event recording can be initialized only once after booting.
If the test passed, the console will show "[SUCCESS] AmpUT_EventRecord_init".
t evtrec_o init

Step 2: Record start

t evtrec_o start

If the test passed, the console will show “[SUCCESS] AmpUT_EventRecord_start”.

Step 3: Trigger Event Record before the unit test stops (support multiple events, but the interval of the two events must be longer than 1 minute)

t evtrec_o event

If an event command completes successfully, the console will show “[SUCCESS] EVNT0X. MP4 is closed”. If the test passed, the console will show “[SUCCESS] AmpUT_EventRecord : End”.

[Output]

EVNT0x.MP4.

The duration of event file EVNT0x.Mp4 is 13 seconds.

6.25 Unit Tests: Emergency Recording

The Emergency recording unit test contains the following tests:

1. Emergency recording. When starting emergency recording process, an emergency clip will be generated. The clip will have 5 seconds before and 5 seconds after emergency recording process starts.

6.25.1 Emergency Recording: Unit Test

1. Emergency recording

[Command sequence]

Step 0: Before the unit test, prepare related files within the SD card:
MWUnitTest\GoldenData\Muxer\Emergency\OUT_0000.udta
MWUnitTest\GoldenData\Muxer\Emergency \OUT_0000.nhnt
MWUnitTest\GoldenData\Muxer\Emergency \OUT_0000.h264

Step 1: Initial Emergency record. Emergency record can be initialized only once after booting.

t em init

If the test passed, the console will show “[SUCCESS] AmpUT_Em_init”.

Step 2: Start Emergency record mux

t em start

If the test passed, the console will show “[SUCCESS] AmpUT_Em_start”.

Step 3: Start Emergency record process

t em em

If an event command completes successfully, the console will show “[SUCCESS] EM.MP4 is closed”. If the test passed, the console will show “[SUCCESS] AmpUT_Em : End”.

[Output]

Please wait until the following console message shows up:
AmpMp4Edt_Merge End
File will be:
EM.MP4

Note: Please be aware that the raw H264 bitstream only has 6 minutes. Please do not type the Step 3 command too late. Emergency recording could also take some time to copy and merge files.

6.26 Unit Tests: Editor

The editor unit test contains the following tests:

1. Edit function - crop2new
2. Edit function - merge
3. Edit function - recover MP4
4. Edit function - recover MOV
5. Edit function - divide

Note: Before testing edit functions every time, BUB needs to be reset.

6.26.1 Editor: Unit Test

1. Edit function - crop2new

   [Command sequences]

   Step 0: Prepare related files to SD card root
   MWUnitTest\GoldenData\Editor\Crop\Ori\TEST.MP4

   Step 1: Initiate editor unit test. Editor can be initialized only once after booting.
   t edt init
   If the test passed, the console will show "[SUCCESS] AmpUT_Editor_Init".
   Step 2: Start testing the edit function
   t edt crop 1000 3000
   If the test passed, the console will show "[SUCCESS] AmpUT_Mp4Edt : End".

   [Output]

   Compare the TEST2.MP4 with
   MWUnitTest\GoldenData\Editor\Crop\TEST2.MP4

2. Edit function - merge

   [Command sequences]

   Step 0: Prepare related files to SD card root
   MWUnitTest\GoldenData\Editor\Merge\Ori\TEST.MP4
   MWUnitTest\GoldenData\Editor\Merge\Ori\TEST2.MP4
   Step 1: Initiate editor unit test. Editor can be initialized only once after booting.
   t edt init
   If the test passed, the console will show "[SUCCESS] AmpUT_Editor_Init".
   Step 2: Start testing the edit function
   t edt merge
   If the test passed, the console will show "[SUCCESS] AmpUT_Mp4Edt : End".

   [Output]

   Check the clip TEST.MP4: Play it with any player and check if the length is 16 seconds or not

3. Edit function - recover MP4

   Step 0: Prepare related files within SD card
   MWUnitTest\GoldenData\Editor\Recover\MP4\Ori\TEST.MP4 (Bad MP4 file, it cannot be played).
   MWUnitTest\GoldenData\Editor\Recover\MP4\Ori\TEST.IDX

Step 1: Initiate editor unit test. The editor can be initialized only once after booting.
t edt init
If the test passed, the console will show “[SUCCESS] AmpUT_Editor_Init”.
Step 2: Start testing the edit function
t edt recover
If the test passed, the console will show “[SUCCESS] AmpUT_Mp4Edt : End”.

[Output]
Step 4: Compare the TEST.MP4 with
MWUnitTest\GoldenData\Editor\Recover\MP4\TEST.MP4

4. Edit function – recover MOV

Step 0: Prepare related files within the SD card
MWUnitTest\GoldenData\Editor\Recover\MOV\Ori\TEST.MOV (Bad MOV file, it can’t be played).
MWUnitTest\GoldenData\Editor\Recover\MOV\Ori\TEST.IDX
Step 1: Initiate editor unit test. Editor can be initialized only once after booting.
t edt init
If the test passed, the console will show “[SUCCESS] AmpUT_Editor_Init”.
Step 2: Start testing the edit function
t edt recover
If the test passed, the console will show “[SUCCESS] AmpUT_MovEdt : End”.

[Output]
Step 4: Compare the TEST.MOV with MWUnitTest\GoldenData\Editor\Recover\MOV\TEST.MOV

5. Edit function – divide

Step 0: Prepare related files to the SD card root MWUnitTest\GoldenData\Editor\Divide\Ori\TEST.MP4
MWUnitTest\GoldenData\Editor\Divide\Ori\TEST2.MP4

Step 1: Initiate editor unit test. Editor can be initialized only once after booting.
t edt init
If the test passed, the console will show “[SUCCESS] AmpUT_Editor_Init”.
Step 2: Start testing the edit function
t edt divide 10000
If the test passed, the console will show “[SUCCESS] AmpUT_Mp4Edt : End”.

[Output] TEST.MP4 TEST2.MP4

Check clip TEST.MP4: Play it with any player and check the length is 10 seconds or not.
Check clip TEST2MP4: Play it with any player and check the length is 1min 29 seconds or not.

6.27 Unit Tests: External Muxer

The external mux unit test contains the following tests:

1. Mux the raw bitstream and pass the frame information via stream command

6.27.1 External Muxer: Unit Test

[Command sequence]

Step 0: Prepare related files within the SD card
MWUnitTest\GoldenData\Muxer\ExtMux\OUT_0000.udta
MWUnitTest\GoldenData\Muxer\ExtMux\OUT_0000.nhnt
MWUnitTest\GoldenData\Muxer\ExtMux\OUT_0000.h264

Step 1: Initiate external mux unit test. External mux can be initialized only once after booting.
t extmux init
If the test passed, the console will show “[SUCCESS] AmpUT_ExtMux_init”.

Step 2: Start mux external unit test
t extmux start
If the test passed, the console will show “[SUCCESS] AmpUT_ExtMux : End”.

[Output]

TEST.EXT
TEST.EXT_HDR
Compare with MWUnitTest\GoldenData\Muxer\ExtMux\TEST.EXT
Compare with MWUnitTest\GoldenData\Muxer\ExtMux\TEST.EXT_HDR

6.28 Unit Tests: External Demuxer

The external demux unit test contains the following tests:

- Demux the external file via external header file

6.28.1 External Demuxer: Unit Test

[Command sequence]

Step 0: Prepare related files within the SD card MWUnitTest\GoldenData\Demuxer\ExtDmx\
TEST.EXT MWUnitTest\GoldenData\Demuxer\ExtDmx\TEST.EXT_HDR

Step 1: Initiate external demux unit test. External demux can be initialized only once after booting.
t extdmx init
If the test passed, the console will show “[SUCCESS] AmpUT_ExtDmx_DmxInit”.

Step 2: Start demux external unit test
t extdmx start 0 0 0
If the test passed, the console will show “[SUCCESS] AmpUT_ExtDmx : End”.

Step 3: Exit demux external unit test
t extdmx exit
If the test passed, the console will show “[SUCCESS] AmpUT_ExtDmx_exit”.

[Output]

Compare with golden data
MWUnitTest\GoldenData\Demuxer\ExtDmx\OUT_0001.h264 MWUnitTest\GoldenData\
Demuxer\ExtDmx\OUT_0001.nhnt

6.29 Unit Tests: Linux Virtual FIFO

The NetFIFO unit test contains the following tests:

- Stream out video to PC-VLC via RTSP

6.29.1 Linux Virtual FIFO: Unit Test

[Command sequence]

Step 0: Insert WIFI card into the SD port-1 and prepare the related files within the SD card:
MWUnitTest\GoldenData\Net\NetFIFO\OUT_0000.udta
MWUnitTest\GoldenData\Net\NetFIFO\OUT_0000.nhnt
MWUnitTest\GoldenData\Net\NetFIFO\OUT_0000.h264

Step 1: Initiate the NetFIFO unit test (At ThreadX console)
t netfifo init

Step 2: Start the dummy encode (At ThreadX console)
t netfifo rec_start

Step 3: Start the RTSP server encode (At ThreadX console)
t netfifo rtsp_start

Step 4: On the PC, connect to the camera's AP and use VLC to open "rtsp://192.168.42.1/live".

Step 5: Stop the RTSP server (At ThreadX console)
t netfifo rtsp_stop

Step 6: Stop the dummy video encode (At ThreadX console)
t netfifo rec_stop

[Output]

1. VLC should play the live steam successfully after the RTSP server started.
2. VLC should automatically stop after the RTSP server is stopped.

6.30 Unit Tests: Remote API

The Remote API unit test contains the following tests:

1. Test Remote API interface with AmbaRemoteCam

6.30.1 Remote API: Unit Test

[Command sequence]

Step 0: Insert the WiFi card into the SD port-1 and connect the PC to the camera's Access Point (AP)

Step 1: Set the mode (ThreadX console)
t netctrl set_mode wifi_cmd_data
After the above command completed, the correct result is shown on the console as below:
Current set mode: 0: (Bluetooth Low Energy) BLE, 1: WiFi, 2: null -> cmd_mode: 1, data_mode: 1

Step 2: Initialize the Remote API control unit test (ThreadX console).
t netctrl init

Step 3: From the PC, use AmbaRemoteCam to connect with the camera and perform the test operations, such as LS, CD, PWD, Get_file and Put_file.

[Output]

AmbaRemoteCam should get response for every button pressed.
After the file download is complete, please ensure that the downloaded file is exactly the same as the source file.

6.30.2 Remote API: Unit Test for transmitting the command using RTOS-BLE

[Command sequence]

Step 1: Set the mode (ThreadX console)
t netctrl set_mode ble_cmd
After the above command is completed, the correct result is shown on the console as below:
Current set mode: 0: Able, 1: WiFi, 2: null -> cmd_mode: 0, data_mode: 2

Step 2: Initialize the Remote API control unit test (ThreadX console)
t netctrl init

Step 3: Install and connect BLE with the IOS application "Ambarella Remote Camera" called

Note1: The following commands are not supported in pure-RTOS:
AMBA_PWD, **AMBA_LS**, **AMBA_CD**, **AMBA_GET_FILE**, **AMBA_GET_THUMB**, **AMBA_PUT_FILE**, **AMBA_GET_WIFI_SETTING**, **AMBA_SET_WIFI_SETTING**, **AMBA_GET_WIFI_STATUS**.

6.30.3 Remote API: Unit Test for Transmitting the Command Using RTOS-BLE and Transmitting the File Using WIFI

Step 1: Set the mode (ThreadX console)

t netctrl set_mode ble_cmd_wifi_data

After the above command is completed, the correct result is shown on the console as below:

Current set mode: 0: BLE, 1: WiFi, 2: null -> cmd_mode: 0, data_mode: 1

Step 2: Initialize the Remote API control unit test (ThreadX console)

t netctrl init

Step 3: Install and connect BLE with the IOS application “Ambarella Remote Camera” called

If the user wants to change the mode, please call the command **t netctrl end** and then set the mode and initialize the netctrl framework again.

6.31 Unit Tests: Event Notifier

The Event Notifier unit test contains the following tests:

1. Send event to Linux and get the echo back

6.31.1 Event Notifier: Unit Test

[Command sequence]

Step 0: Initiate Linux service (At Linux console)
RecvFromRTOS

Step 1: Initiate Event Notifier unit test: (At ThreadX console)
t netevent init

Step 2: Send event to Linux (At ThreadX console)
t netevent send 0x12345678

[Output]

Linux console will show as below:
cbEvent: unknown event 0x12345678
data length: 0

6.32 Unit Tests: AV Encode

AV Encode unit test provides the following function tests:

1. Single stream + audio encode
2. Dual stream + audio encode
3. DualHD stream + audio encode

6.32.1 AV Encode: Command List

1. Initialize AV encode unit test and hook the sensor

   t avenc init [sensorID]
   Same as Section 6.2.1.1

2. Enter liveview view

   t avenc create [modeID] [audioType]
   modeID: Please refer to Section 6.2.1.
   audioType: 0 - AAC
   1 - PCM
   . . . . .
   Note : For audioType index, please refer to the audio unit test.

3. Stop liveview view

   t avenc delete

4. Start encoding

   t avenc encstart [milliSeconds]
   milliSeconds: Time of encoding

5. Stop encoding

   t avenc encstop

6. Pause encoding

   t avenc encpause
   Note: On A12, pause/resume is not supported.

7. Resume encoding

   t avenc encresume
   Note: On A12, pause/resume is not supported.

8. Set second stream encode window

   t avenc secwin [width][height]
   width: Encode width
   height: Encode height

9. Dual stream recorder enable

   t avenc dual [on/off]
   on/off - 0: Disable
   1: Enable

10. Dual HD stream recorder enable

    t avenc dualhd [on/off]
    This cmd is replaced by " t avenc dual" + "t avenc secwin".
    Note: Secondary stream encode limitation is the same as Video Encode, see Table in the Video Encode UT Guide.

11. Set the system type

    t avenc system [option]
    enable - 0: NTSC(default), 1: PAL

12. Enable Hybrid algo TA mode (A12 only)

    t avenc ta [en]
    en - 0: Disable, 1: Enable

    Note: 1. This command is only valid in the hybrid LISO mode.
    2. For A12, HybridMode only
    3. This command must be issued before the liveview start command.
    The following sensors are supported now: OV4689/AR0230/MN34120

13. Enable low delay mode

    [A12]

    t avenc lowdelay [LiveviewLowDelay] [EncodeLowDelay]
    LiveviewLowDelay - 0: Disabled, 1: VIN <-> vout0 (LCD), 2: VIN <-> vout1 (TV)
    EncodeLowDelay - 0: Disabled, 1: Enable low delay on whole encode stream
    Verification is the same as VideoEnc.

6.32.2 AV Encode: Unit Test

On A12, please type “t avenc perform 1 1 1” before “create” in 3840x2160p30.

1. AVencode view start/stop - single stream + audio encode

   [Command sequence]
   Step 1: **t avenc init [sensorID][LCD_ID]**
   Step 2: **t avenc create [modelID][audioType]**
   Step 3: **t avenc encodestart 10000**

   [Output]
   Audio file
   PriXXX.bin,PriXXX.aac, PriXXX.nhnt
   Video file
   OUTXXX.h264,OUTXXX.nhnt, OUTXXX.udta
2. AVencode view start/stop - dual stream and 2nd_stream window as 720X400+ audio encode

   [Command sequence]
   Step 1: **t avenc init [sensorID][LCD_ID]**
   Step 2: **t avenc dual 1**
   Step 3: **t avenc secwin 720 400**
   Step 4: **t avenc create [modelID][audioType]**
   Step 5: **t avenc encodestart 10000**

   [Output]
   Audio file
   PriXXX.bin,PriXXX.aac, PriXXX.nhnt

Video file

1. Main stream
OUTXXX.h264,OUTXXX.nhnt, OUTXXX.udta
2. Second stream
OUTXXX_s.h264,OUTXXX_s.nhnt, OUTXXX_S.udta

3. AVencode view start/stop - dual HD stream and 2nd_stream window as 1280 720+ audio encode

[Command sequence]

Step 1: **t avenc init [sensorID][LCD_ID]**
Step 2: **t avenc dualhd 1**
Step 3: **t avenc secwin 1280 720**
Step 4: **t avenc create [modeID][audioType]**
Step 5: **t avenc encodestart 10000**

[Output]

Audio file

PriXXX.bin,PriXXX.aac, PriXXX.nhnt

Video file

1. Main stream
OUTXXX.h264,OUTXXX.nhnt, OUTXXX.udta
2. Second stream
OUTXXX_s.h264,OUTXXX_s.nhnt, OUTXXX_S.udta

6.33 Unit Tests: YUV Input Encode (A12 only)

YUV input Encode unit test provides the following function tests:

1. YUV input liveview

2. YUV input encode

6.33.1 Yuv Input Encode: Command List

1. Initialize YUV input encode unit test and hook the device

   t yuvenc init [yuvID]
   yuvID: 0 – TI5150

2. Enter liveview view

   t yuvenc liveviewstart [modeID]
   modeID: 0 – 480I60
   1 – 567I50

3. Stop liveview view

   t yuvenc delete

4. Start encoding

   t yuvenc encstart [milliSeconds]
   milliSeconds: Time of encoding

5. Stop encoding

   t yuvenc encstop

6. Pause encoding

   t yuvenc encpause
   Note: On A12, pause/resume is not supported.

7. Resume encoding

   t yuvenc encresume

6.33.2 YUV Input Encode: Unit Test

1. Yuvencode view start/stop

   [Command sequence]
   Step 1: **t yuvenc init [yuvID]**
   Step 2: **t yuvenc liveviewstart [modeID]**
   Step 3: **t yuvenc encodestart**
   Step 4: **t yuvenc encodestop**

   [Output]
   Video file
   OUTXXX.h264,OUTXXX.nhnt, OUTXXX.udta

6.34 Unit Tests: Dual VIN Encode (A12 only)

Dual VIN Encode unit test provides the following function tests:

1. Main or PIP switchable
2. Main and PIP dual vin (Hybrid Mode only)
3. Different FrameRate for separate VIN
   - PIV is forbid.
   - MainVin must have Higher FrameRate.

6.34.1 Dual VIN Encode: Command List

1. Initialize Dual VIN encode unit test and hook the sensor

   t dualvinenc init [MainSensorID][PipSensorID]
   MainSensorID: Same as Section 6.2.1.1 (Only supports OV4689/AR0230 now)
   PipSensorID: Same as Section 6.2.1.1 (Only supports B5_OV4689/B5_OV9750 now)

2. Select VIN source

   t dualvinenc dualvin [Vinsource]
   VinSource: 0: MainVin Only
   1: PipVin Only
   2: Main and PIP Both (Hybrid Mode only)

3. Enter liveview view

   t dualvinenc liveviewstart [MainModeID][PipModeID]
   MainModeID: Please refer to Section 6.2.1
   PipModeID: Please refer to Section 6.2.1

OV4689

NTSC		PIP Stream of VINChannel 1		
		2560X1440P30	1920X1080P30	1280X720P30
Main stream of Vin-Channel_0	2560X1440P30 HDR	O	O	O
	2560X1440P30	O	O	O
	1920X1080P30 HDR	O	O	O
	1920X1080P30	O	O	O

[AR0230 + OV9750]

NTSC		PIP Stream of VINChannel 1		
		2560X1440P30	1920X1080P30	1280X720P30
Main stream of Vin-Channel_0	2560X1440P30 HDR	X	X	X
	2560X1440P30	X	X	X
	1920X1080P30 HDR	X	O	O
	1920X1080P30	X	O	O

[OV4689 + AR0230]

NTSC		PIP Stream of VINChannel 1		
		2560X1440P30	1920X1080P30	1280X720P30
Main stream of Vin-Channel_0	2560X1440P30 HDR	X	O	O
	2560X1440P30	X	O	O
	1920X1080P30 HDR	X	O	O
	1920X1080P30	X	O	O

4. Stop liveview view

   t dualvinenc liveviewstop

5. Start encoding

   t dualvinenc encstart [milliSeconds]
   milliSeconds: Time of encoding

6. Stop encoding

   t dualvinenc encstop

7. Pause encoding

   t dualvinenc encpause

Note: On A12, pause/resume is not supported.

8. Resume encoding

   t dualvinenc encresume

   Note: On A12, pause/resume is not supported.

9. Enable dual channel dual stream encode (A total of 4 streams)

   t duve dual 1

10. Set small stream enc window of dual channel

    t duve secwin [Width][Height]

11. PIV (currently only support 1x PIV interpolation)

    t duve piv [Width][Height] 1 0 0 0
    Note: This feature is forbid when DualVin has different FrameRates.

12. Select PIV source, this command must be set before 6.34.1.11

    t duve stillvin [source]
    Source: 0 – Main, 1 – Pip.

13. Enable Hybrid algo TA mode

    t duve ta [en] [VinId]
    VinId: 0 is MainVin, 1 is PipVin

14. Enable encode with thumbnail (Currently, only supports MainVin Video thumbnail)

    t duve thm [en] [VinId]
    VinId: 0 is MainVin, 1 is PipVin
    Note: Thumbnail is implemented via PIV, so it has the same restrictions as PIV.
    Note: On A12, please enable video thumbnail before Liveview starts.

15. Test the bitrate control setting

    t duve brc [option]
    option: Config quality control or invoke
    config - config quality control parameters **t duve brc config [mode][...] [mode][Stream][...]**
    stream: 0: Primary stream, 1: Secondary stream mode: Bitrate control option

    Mode = 0 - Bitrate config
    t duve brc config 0 [Stream][BrcMode][AvgBitRate][MaxBitRate]
    [MinBitRate][iBeat]
    BrcMode: 1 - CBR, 2 – SmartVBR AvgBitRate: Average Bitrate. Unit: Mbps
    MaxBitRate: Maximum Bitrate for SmartVBR only. Unit: Mbps (max: 54)
    MinBitRate: Minimum Bitrate for SmartVBR only. Unit: Mbps (min: 6)

The default values are suggested as below:

	MinBitRate	MaxBitRate	AvgBitRate
1920X1080P30	9	15	12
1920X1080P60	13	21	18
1920X1080P120	27	45	36
1280X720P30	6	10	8
1280X720P60	9	15	12
1280X720P120	13	22	18
1280X720P240	27	45	36
2560X1440P30	18	30	24
2560X1440P60	27	45	36
3840X2160P30	45	75	60
1920X1440P30	13	22	18
1280X960P30	10	15	13
1280X960P60	12	18	15
1280X960P120	18	27	24
2560X1920P30	27	45	36
2560X1920P60	27	45	36
2704X1520P60	27	41	36

Table 6-5. Default Values.

The expected result is that (1) for VBR clips, MinBitRate <= clip bit rate <= MaxBitRate, and (2) for CBR clips, (AvgBitRate * 0.9) <= clip bit rate <= (AvgBitRate * 1.1).

iBeat: 0 – Off
1 - M-1 B pictures after IDR picture requires special quant reduction.
2 - The P picture just before IDR picture requires special quant reduction.
4 - M-1 M-1 B pictures before IDR picture requires special quant reduction.
8 - The P picture just before I picture requires special quant reduction.
Note: Before invoking CBR, please turn off encode monitor first (refer to 28, 29)
Note: For engineer test only.

Mode = 1 - Bitrate change
t duve brc config 1 [Stream][AverBitRtate]
AverBitRtate: Unit in Kbps Mode = 2 - Gop Change
t duve brc config 2 [Stream][M][N][Idr]

M: Distance between in P frame(simple rc does not support M>1) N: Distance between in I frame
Idr: Distance between in IDR frame Mode = 3 - QP control
t duve brc config 3 [Stream][QpminI][QPmaxI][QPminP][QPmaxP]
[QPminB][QPmaxB]
QPmin: Minimum QP value
QPmax: Maximum QP value
Mode = 4 - QP model control
t videoenc brc config 4
Mode = 5 - ROI control
t duve brc config 5
Mode = 6 - HQP control
t duve brc config 6 [Qpmax][QPmin][QPreduce]

QPmin: Minimum QP value
QPmax: Maximum QP value
QPreduce: How much better to make Q QP relative to P QP, 1~10, default is 6.
Mode = 7 - ZMV control
t duve brc config 7 [threshold]
threshold: ZMV threshold Mode 8 = - Force Idr
t duve c brc config 8
Mode = 9 - RESET ALL
t duve brc config 9
invoke - Invoke command
t duve brc invoke [stream]
0: 1st stream (MainVin Primary stream)
1: 2nd stream (MainVin Secondary stream)
2: 3rd stream (PipVin Primary stream)
3: 4th stream (PipVin Secondary stream)

16. Enable blend

t dualvinenc blend [streamID] [enable]
For DualVin case
streamID: 0 – MainVin Primary stream, 1 – PipVin primary stream
2– MainVin secondary stream, 3 – PipVin secondary stream
For SingleVin case
streamID: 0 – MainVin(PipVin) Primary stream, 1 – MainVin(PipVin) secondary stream

enable: Enable blend
Note: A12ExpressMode blending result will be shown in Liveview and Encode stream; A12 Hybridmode blending result will be shown in the Encode stream only.
Note: On A9S, blending is not supported in the resolutions run in Pipeline Mode 5.

[OV4689+B5OV4689]

NTSC		PIP Stream of VINChannel 1		
		2560X1440P30	1920X1080P30	1280X720P30
Main stream of Vin-Channel_0	2560X1440P30 HDR	X	480p30 2nd stream	720p30 2nd stream
	2560X1440P30	X	720p30 2nd stream	720p30 2nd stream
	1920X1080P30 HDR	480p30 2nd stream	720p30 2nd stream	720p30 2nd stream
	1920X1080P30	720p30 2nd stream	720p30 2nd stream	720p30 2nd stream

[AR0230 + OV OV9750]

NTSC		PIP Stream of VINChannel 1
		1280X720P30
Main stream of VinChannel_0	1920X1080P30 HDR	720p30 2nd stream
	1920X1080P30	720p30 2nd stream

[OV4689+B5AR0230]

NTSC		PIP Stream of VINChannel 1
		1920X1080P30
Main stream of VinChannel_0	2560X1440P30 HDR	720p30 2nd stream
	2560X1440P30	720p30 2nd stream
	1920X1080P30 HDR	720p30 2nd stream
	1920X1080P30	720p30 2nd stream

[OV4689+UsbCam]

NTSC		PIP Stream of VINChannel 1
		1280X720P30
Main stream of VinChannel_0	2560X1440P30 HDR	720p30 2nd stream
	2560X1440P30	720p30 2nd stream
	1920X1080P30 HDR	720p30 2nd stream
	1920X1080P30	720p30 2nd stream

[AR0230+UsbCam]

NTSC		PIP Stream of VINChannel 1
		1280X720P30
Main stream of VinChannel_0	1920X1080P30 HDR	720p30 2nd stream
	1920X1080P30	720p30 2nd stream

Table 6-6. Capability of Four Stream Encode Blending.

17. Enable Dual(HD) stream

    t dualvinenc dual [enable]
    0: Disable (default), 1: Enable
    Note: Default setting is 720*400. Please use secwin test cmd to set desired second stream window.

18. Config second stream window

    t dualvinenc secwin [width] [height]

19. Config YuvRawInput setting

    t dualvinenc yuv [enable][Ch][Fmt][Source][Offset]
    Enable: 0 - Disable (default), 1 – Enable
    Ch: 0 - MainVin, 1 – PipVin (Current only supports PipVin)
    Fmt: 0 – Yuv420, 1 – Yuv422 (Current only supports Yuv422)
    Source: 0 – Memory(Debugging usage), 1 – USBCam
    Offset: Tune this value 0,1,2... if the preview is pink when mounting YUVinput Device.
    Note: Currently, DualVin+USBCam can only support the following features.

Feature	Support
Encode Start/Stop	O
ResolutionSwitch(MainChan)	O
DualStream	O
DualHD	O

Table 6-7. Supporting Features.

20. Config VOUT Main or PIP layout before liveview starts or resolution changes. (A12 dualVin only)

    t dualvinenc DispLayout [VoutIdx][VidSelect][Offset X][Offset Y][Width][Height]
    VoutIdx: 0 - LCD, 1 – Tv
    VinSelect: 0 - Main, 1 – Pip
    OffsetX: offset of X-axis
    OffsetY: offset of Y-axis
    Width: Width
    Height: Height

21. Set Stream Encode Priority (A12 HybridMode only)

    t dualvinenc streampriority [MainPriStreamPriority] [PipPriStreamPriority] [MainSecStreamPriority] [PipSecStreamPriority]
    *PriStreamPriority - 1: 1st priority, 2: 2nd priority…
    *SecStreamPriority - 1: 1st priority, 2: 2nd priority…

22. Enable low delay mode

    [A12]

    t dualvinenc lowdelay [LiveviewLowDelay] [EncodeLowDelay]
    LiveviewLowDelay - 0: Disabled, 1: VIN <-> vout0 (LCD), 2: VIN <-> vout1 (TV)
    EncodeLowDelay - 0: Disabled, 1: Enable low delay on the whole encode stream.
    Verification is the same as VideoEnc.

6.34.2 Dual VIN Encode: USB Cam Commnd List

This section decribe USB Cam command list for user to active it

1. Init the USB cam

   t usbcamenc init

2. Start the USB cam

   t usbcamenc start [Fmt][Frm][Fps][Alt]
   Fmt: Encode format
   Frm: Encode window size
   Fps: Frame rate
   Alt: 0 (default), should not change it.
   Note: The setting above is variant in different USBCams, please init USBCam first then type "t usbcamenc info" to get the desired information. For recommended device, please use "t usbcamenc start 2 12 30 0".

3. Recieve USB cam data

   t usbcamenc rcv

6.34.3 Dual VIN Encode: Unit Test

1. Start MainVin only at 1080p30

   [Command sequence]
   Step 1: **t dualvinenc dualvin 0**
   Step 2: **t dualvinenc lvproc 1 1**
   Step 3: **t dualvinenc init 0 9**
   Step 4 : **t dualvinenc liveviewstart 0 0**

2. Start PipVin only at 1080p30

   [Command sequence]
   Step 1: **t dualvinenc dualvin 1**
   Step 2: **t dualvinenc lvproc 1 1**
   Step 3: **t dualvinenc init 0 9**
   Step 4: **t dualvinenc liveviewstart 0 0**

3. Start MainVin only at 1080p30, then switch to PipVin

   [Command sequence]
   Step 1: **t dualvinenc dualvin 0**
   Step 2: **t dualvinenc lvproc 1 1**
   Step 3: **t dualvinenc init 0 9**
   Step 4: **t dualvinenc liveviewstart 0 0**
   Step 5: **t dualvinenc liveviewstop**
   Step 6: **t dualvinenc dualvin 1**
   Step 7: **t dualvinenc liveviewstart 0 0**

4. Start Dual Vin at 1080P30 and Encode

   [Command sequence]
   Step 1: **t dualvinenc dualvin 2**
   Step 2: **t dualvinenc lvproc 1 1**
   Step 3: **t dualvinenc init 0 9**
   Step 4: **t dualvinenc liveviewstart 0 0**
   Step 5: **t dualvinenc encstart**
   Step 6: **t dualvinenc encstop**
   [Output] Generate two h264 clips.

5. Start Dual VIN Dual stream at 1080P30

   [Command sequence]
   Step 1: **t dualvinenc dualvin 2**
   Step 2: **t dualvinenc lvproc 1 1**
   Step 3: **t dualvinenc init 0 9**
   Step 4: **t dualvinenc dual 1**
   Step 5: **t dualvinenc secwin 576 320** (small stream window, can not exceed 720 width)
   Step 6: **t dualvinenc liveviewstart 0 0**
   Step 7: **t dualvinenc encstart**
   Step 8: **t dualvinenc encstop**
   [Output] Generate two h264 clips.

6. Illegal signal detect

   [Command sequence]
   Step 1: Start liveview in the dual VIN, please refer to 6.34.2.4.
   Step 2: Unplug PIPVin cable.
   Step 3: The system automatically switches to Single VIN, the user will see MainVin in LCD/TV.

7. UsbCam as PipChan and encode clip

   [Command sequence]
   Step 1: **t usbcamenc init**
   Step 2: **t usbcamenc start 2 12 30 0**
   Step 3: **t usbcamenc rcv**
   Step 4: **t duve yuv 1 1 1 1 2**
   Step 5: **t duve dualvin 2**
   Step 6: **t duve init 0 17 OV4689 + UsbCam**
   Step 7: **t duve lvproc 1 1**
   Step 8: **t duve tv 0**
   Step 9: **t duve lvst 0 5**
   Step 10: **t duve enst**
   Step 11: **t duve ensp**

8. UsbCam as PipChan and encode clip (only single PipVin)

   [Command sequence]
   Step 1: **t usbcamenc init**
   Step 2: **t usbcamenc start 2 12 30 0**
   Step 3: **t usbcamenc rcv**
   Step 4: **t duve yuv 1 0 1 1 2**
   Step 5: **t duve dualvin 0**
   Step 6: **t duve init 17 0 OV4689 + UsbCam**
   Step 7: **t duve lvproc 1 1**
   Step 8: **t duve tv 0**
   Step 9: **t duve lvst 5**
   Step 9: **t duve enst**
   Step 10: **t duve ensp**

9. Config vout Pip of LCD to let Pip show in Left-Top side

   [Command sequence]
   Step 1: **t duve dualvin 2**
   Step 2: **t duve init 0 9**
   Step 3: **t duve lvproc 1 1**
   Step 4: **t duve PipDispLayout 0 0 480 180**
   Step 5: **t duve lvst 0 0**

6.35 Unit Tests: Color Space Conversion (Csc) Control

[Command sequence]

1. Enable VOUT and display image on VOUT vid decode unit test. Please refer to Sections 6.4 and 6.7.
2. Set CSC [ch] [parameters] for LCD/HDMI

   Normal display
   t csc set 1 1.00000 -0.33666 -0.69822 1.00000 1.73225 0.00000 1.00000 0.00000 1.37068 132.46 -221.73 -175.45 255 0 255 0 255 0

Red

t csc set 1 1.00000 -0.33666 -0.69822 1.00000 1.73225 0.00000 1.00000 0.00000 2.37068 132.46 -221.73 -175.45 255 0 255 0 255 0

Green

t csc set 1 1.00000 0.33666 -0.69822 1.00000 1.73225 0.00000 1.00000 0.00000 1.37068 132.46 -221.73 -175.45 255 0 255 0 255 0

Blue

t csc set 1 1.00000 -0.33666 -0.69822 1.00000 2.73225 0.00000 1.00000 0.00000 1.37068 132.46 -221.73 -175.45 255 0 255 0 255 0

Black

t csc set 1 0 0 0 0 0 0 0 0 0 0 0 0 255 0 255 0 255 0

3. Set CSC [ch] [parameters] for CVBS

   Currently, it is not supported, TBD.

7 Regression

7.1 Regression: Overview

This chapter describes basic cross-unit test regressions.

7.2 Regression: Mode Switching

1. Boot to liveview first
2. Boot to video decode first
3. Boot to still decode first

7.2.1 Mode Switching: Test Flow

1. Boot to liveview first

   [Loop command sequence]
   Step 1: **t videoenc init 0**
   Step 2: **t videoenc liveviewstart 0**
   Step 3: **t videoenc encodestart 2000**
   Step 4: **t videoenc liveviewstop**
   Step 5: **t videodec init**
   Step 6: **t videodec start 0000**
   Step 7: **t videodec stop**
   Step 8: Go to Step 2

   [Variations]
   - Single/dual VOUT
   - Dzoom liveview/encode
   - Various video resolutions

2. Boot to decode first

   [Loop command sequence]
   Step 0: Prepare OUT_0000.h264/nhnt/udta
   Step 1: **t videodec init**
   Step 2: **t videodec start 0000**
   Step 3: **t videodec stop**
   Step 4: **t videoenc init 1**
   Step 5: **t videoenc liveviewstart 1**
   Step 6: **t videoenc encodestart 2000**
   Step 7: **t videoenc liveviewstop**
   Step 8: Go to Step 2

[Variations]
- Single/dual vout
- Dzoom liveview/encode
- Various video resolutions
- Boot to still decode first

[Loop command sequence]

Step 0: prepare xxx.jpg in SD card
Step 1: **t stilldec init**
Step 2: **t stilldec feed xxx.jpg 0**
Step 3: **t stilldec dec [width] [height] 0**
Step 4: **t videoenc init 1**
Step 5: **t videoenc liveviewstart 1**
Step 6: **t videoenc encodestart 2000**
Step 7: **t videoenc liveviewstop**
Step 8: Go to Step 2

[Variations]

1. Single/dual vout
2. Dzoom liveview/encode
3. Various video resolutions
4. Decode .jpg or .h264

7.3 Regression: Standby Mode

There is only one test regression that is carried out in the Standby Mode:

1. Boot to liveview first
2. Boot to decode first

7.3.1 Standby Mode: Test Flow

1. Boot to liveview first (Assume imx117 sensor)

Step 1: **t videoenc init 1**
Step 2: **t videoenc liveviewstart 0**
Step 3: **t videoenc suspend 1**
Step 4: **t videoenc liveviewstop**
Step 5: **t videoenc standby**
Step 6: Wait for a while, hit any key on keyboard
Step 7: **t videoenc liveviewstart 0**
Step 8: Loop Step 4 to Step 7. Liveview should always come out in Step 7.

2. Boot to decode first

   Step 1: **t videodec init**
   Step 2: **t videodec suspend 1**
   Step 3: **t videodec start** (Please refer to 6.4.1)
   Step 4: **t videodec stop**
   Step 5: **t videodec standby**
   Step 6: Wait for a while, hit any key on the keyboard
   Step 7: Loop Step 3 to Step 5. The clip should be played normally.

7.4 Regression: Image Quality

1. Still Mode
2. Video Mode

7.4.1 Image Quality: Still Regression Test

Assumed Environment: A12 EVK Target Board, OV4689 Sensor, MW UnitTest App

Do the following procedures:

Step 1. Copy test data Still_Regression_Test\SD_card* to your Secure Digital(SD) card
Step 2. After system booting is complete, type command "**BUB_test_a12.ash**"
Step 3. Mount your Secure Digital(SD) card to PC
Step 4. Run `Still_Regression_Test\PC\1.CopyResultsFromSD.bat`. It can help the user to copy YUV results to `Still_Regression_Test\PC\hw\`.
Step 5. Run `Still_Regression_Test\PC\2.CompareResults.bat`. The user should get the following messages and it means that the test passed:

```
HDF:  640x480 to 640x480
    0
    0
LI1:  640x480 to 640x480
    0
    0
LI2: 640x480 to 640x480
    0
    0
LI3:  1920x1088 to 1920x1088
    0
    0
LI3_2X:  1920x1088 to 1920x1088
    0
    0
LI3_CA:  1920x1088 to 1920x1088
    0
    0
LI3_WARP:  1920x1088 to 1920x1088
    0
    0
LI3_VIG:  1920x1088 to 1920x1088
    0
    0
```

LI3_FPN: 1920x1088 to 1920x1088
    0
    0
LI3_STAT:
    0
LI5: 640x480 to 640x480
    0
    0
MI1: 640x480 to 640x480
    0
    0
MI1_2X: 640x480 to 640x480
    0
    0
MI1_CA: 640x480 to 640x480
    0
    0
MI1_WARP: 640x480 to 640x480
    0
    0
MI1_FPN: 640x480 to 640x480
    0
    0
MI1_VIG: 640x480 to 640x480
    0
    0
MI2: 640x480 to 640x480
    0
    0
MI3: 4608x3456 to 4608x3456
    0
    0
MI3_2X: 4608x3456 to 4608x3456
    0
    0
MI3_WARP: 4608x3456 to 4608x3456
    0
    0
MI3_FPN: 4608x3456 to 4608x3456
    0
    0
MI3_VIG: 4608x3456 to 4608x3456
    0
    0
MI4: 640x480 to 640x480
    0
    0
MI5: 640x480 to 640x480
    0
    0
MI6: 640x480 to 640x480
    0
    0
MI7: 640x480 to 640x480
    0
    0
HI10: 640x480 to 640x480

```
    0
    0
HI10_2X:  640x480 to 640x480
    0
    0
HI10_CA:  640x480 to 640x480
    0
    0
HI10_WARP:  640x480 to 640x480
    0
    0
HI10_VIG:  640x480 to 640x480
    0
    0
HI10_FPN:  640x480 to 640x480
    0
    0
HI20:  3640x480 to 3640x480
    0
    0
HI40:  4416x3288 to 4416x3288
    0
    0
HI40_2X:  4416x3288 to 4416x3288
    0
    0
HI40_CA:  4416x3288 to 4416x3288
    0
    0
HI40_WARP:  4416x3288 to 4416x3288
    0
    0
HI40_VIG:  4416x3288 to 4416x3288
    0
    0
HI40_FPN:  4416x3288 to 4416x3288
    0
    0
HI40_STAT:
    0
Press any key to continue . . .
```


7.4.2 Image Quality: Video Regression Test

Assumed Environment: A12 EVK Target Board, OV4689 Sensor, MW UnitTest App

Do the following procedures:

Step 1. Copy test data `Video_Regression_Test\SD_card\*` to the user's Secure Digital(SD) card

Step 2. After system boot up is complete, type "**HybridVideoLiso.ash**"

Step 3. Mount the user's Secure Digital(SD) card to PC

Step 4. Run `Video_Regression_Test\PC\HybridVideoLiso_1_CopyResultsFromSD.bat`. It can help the user to copy YUV results to `Still_Regression_Test\PC\hw\`.

Step 5. Run `Video_Regression_Test\PC\HybridVideoLiso_2_CompareResults.bat`. The user should get the following messages and it means that the test passed:

```
FRAME0:
    0
    0
FRAME1:
    0
    0
FRAME2:
    0
    0
FRAME3:
    0
    0
FRAME4:
    0
    0
FRAME5:
    0
    0
FRAME6:
    0
    0
FRAME7:
    0
    0
FRAME8:
    0
    0
Press any key to continue . . .
```

Appendix 1 Additional Resources

Please contact an Ambarella representative for digital copies.

Appendix 2 Important Notice

All Ambarella design specifications, datasheets, drawings, files, and other documents (together and separately, “materials”) are provided on an “*as is*” basis, and Ambarella makes no warranties, expressed, implied, statutory, or otherwise with respect to the materials, and expressly disclaims all implied warranties of noninfringement, merchantability, and fitness for a particular purpose. The information contained herein is believed to be accurate and reliable. However, Ambarella assumes no responsibility for the consequences of use of such information.

Ambarella Incorporated reserves the right to correct, modify, enhance, improve, and otherwise change its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete.

All products are sold subject to Ambarella’s terms and conditions of sale supplied at the time of order acknowledgment. Ambarella warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with its standard warranty. Testing and other quality control techniques are used to the extent Ambarella deems necessary to support this warranty.

Ambarella assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using Ambarella components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

Ambarella does not warrant or represent that any license, either expressed or implied, is granted under any Ambarella patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which Ambarella products or services are used. Information published by Ambarella regarding third-party products or services does not constitute a license from Ambarella to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from Ambarella under the patents or other intellectual property of Ambarella.

Reproduction of information from Ambarella documents is not permissible without prior approval from Ambarella.

Ambarella products are not authorized for use in safety-critical applications (such as life support) where a failure of the product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Customers acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of Ambarella products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by Ambarella. Further, Customers must fully indemnify Ambarella and its representatives against any damages arising out of the use of Ambarella products in such safety-critical applications.

Ambarella products are neither designed nor intended for use in automotive and military/aerospace applications or environments. Customers acknowledge and agree that any such use of Ambarella products is solely at the Customer’s risk, and they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

Appendix 3 Revision History

NOTE: Page numbers for previous drafts may differ from page numbers in the current version.

Version	Date	Comments
1.0	22 September 2014	Formatted to SDK6. Update VideoDec/StillDec test items. Add Emergency Recording unit test. Rename to SDK6. Add StillEnc unit tests. Add MotorVu360. Add Video multi-channel decoding. Remove CALIB. Add EventRecord Add EventRecord-Overlapped Add EventRecord-NonOverlapped Remove StillTuning
1.1	25 September 2014	Update in Section 6.12.2 CFS: Unit Test.
1.2	11 November 2014	Update in Sections 6.2.1, 6.28, 6.29, 6.30, 6.4.1, 6.7.1 and 6.7.3.
1.3	20 November 2014	Update in Section 6.29.1.
1.4	11 December 2014	Update in Chapter 4.
1.5	18 December 2014	Update in Sections 6.8.1, 6.9.1, 6.10.1, 6.11.1, 6.16.1, 6.17.1, 6.22.1, 6.23.1, 6.24.1, 6.25.1, 6.26.1 and 6.27.1. Added Section 6.31 Unit Test: AV Encode.
1.6	24 December 2014	Update in Sections 6.2.1, 6.6.1 and 6.20.1.
1.7	26 December 2014	Update in Sections 6.20, 6.20.1 and 6.20.2.
1.8	13 January 2015	Update in Sections 6.2.1 and 6.27.1. Added 6.28 Unit Tests: External Demuxer.
1.9	5 February 2015	Update in Sections 3.2, 3.3, 4.1, 5.1, 6.2.1, 6.2.2, 6.4.1, 6.4.2, 6.4.3, 6.5, 6.5.2, 6.6, 6.6.1, 6.6.2, 6.7.1, 6.8, 6.8.1, 6.13.1, 6.20.1, 6.22.1, 6.23.1, 6.24.1, 6.25.1, 6.26.1, 6.27.1, 6.28.1, 6.32.1, 7.2.1. Added Sections 6.33 & 7.3.
2.0	11 March 2015	Update in Sections 3.3, 6.2.1, 6.4.2, 6.5.2, 6.6.1, 6.6.2, 6.23.1, 6.26, 6.26.1, & 6.32.1. Added Sections 7.4, 7.4.1. & 7.4.2.
2.1	26 May 2015	Update in Sections 3.2, 5.1, 6.2.1, 6.2.2, 6.6.1, 6.8.1, 6.9.1, 6.10.1, 6.12.2, 6.16.1, 6.17.1, 6.21.1, 6.22.1, 6.23.1, 6.24.1, 6.25.1, 6.26.1, 6.27.1, 6.28.1, 6.30.1, 6.33, 7.3 and 7.3.1. Added 6.30.2, 6.30.3, 6.34, 6.34.1 and 6.34.2.
2.2	15 July 2015	Update in Sections 2.1, 3.2, 6.2, 6.2.1, 6.2.2, 6.3, 6.4.3, 6.5, 6.6, 6.6.1, 6.7.1, 6.13.1, 6.19, 6.20.1, 6.33 and 6.34. Update in Section 3.2.
2.3	24 February 2016	Update in Sections 3.2, 3.4, 6.1, 6.2.1, 6.2.2, 6.4.1, 6.5.1, 6.5.2, 6.5.3, 6.6.1, 6.6.2, 6.7.1, 6.14.1, 6.14.2, 6.15.2, 6.16.1, 6.18.1, 6.19.1, 6.20.2, 6.32.1, 6.32.2, 6.33.1, 6.34, 6.34.1 and 6.35. Add Section 6.34.2.

Table A3-1. Revision History