MULTIMEDIA User Guide

SOPHGO

Jun 19, 2024

menu

CHAPTER 1

Disclaimer

Legal Disclaimer

Notice

The purchased products, services and features are stipulated by the contract made between SOPHGO and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied. The information in

this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Technical Support

Address Floor 6, Building 1, Yard 9, FengHao East Road, Haidian District, Beijing, 100094, China

Website https://www.sophgo.com/

Email sales@sophgo.com

Phone +86-10-57590723 +86-10-57590724

CHAPTER 2

Release note

Version	Date of Release	Description
V0.1.0	2022.08.10	First release, contains SOPHON ffmpeg and SOPHON opencv
V0.2.4	2022.08.30	Official release, supplement documents
V0.6.0	2023.02.28	Added new features: watermark, mosaic, unified memory allocation
V0.6.1	2023.03.31	Add sample installation and use method
V0.7.0	2023.05.16	1, bm_opencv support imshow; 2, fix bug.
V0.7.1	2023.07.11	1, bm_ffmpeg supports jpeg loop dec; 2, new operator bayer2rgb.
V0.7.3	2023.10.18	1, fix bug.
V0.8.0	2023.08.01	1, fix bug.
V0.10.0	2024.04.11	1, Open the bmvid interface; 2, VPU dec supports external allocation of physical memory; 3, Update the video codec firmware.

CHAPTER 3

Install sophon-mw

sophon-mw offers different types of installations on different Linux distributions. Please choose the corresponding installation according to your system, do not mix multiple installations on one machine. The “” in the following description is only an example, and may vary depending on the actual installed version.

Please replace $arch $system below with proper architecture on installation machine.

- If host processor is INTEL or AMD, replace $arch with amd64 and $system with x86_64
- If host processor is ARM64 or Phytium, replace $arch with arm64 and $system with aarch64
- If host processor is SG2042, replace $arch with riscv64 and $system with riscv_64

If using Debian/Ubuntu system:

The sophon-mw installation package consists of four files:

- sophon-mw-sophon-ffmpeg__$arch.deb
- sophon-mw-sophon-ffmpeg-dev__$arch.deb
- sophon-mw-sophon-opencv__$arch.deb
- sophon-mw-sophon-opencv-dev__$arch.deb

If using CentOS system:

The sophon-mw installation package consists of four files:

- sophon-mw-sophon-ffmpeg__$arch.rpm

- sophon-mw-sophon-ffmpeg-dev__$arch.rpm
- sophon-mw-sophon-opencv-abi0__$arch.rpm
- sophon-mw-sophon-opencv-abi0-dev__$arch.rpm

If using Fedora system:

The sophon-mw installation package consists of four files:

- sophon-mw-sophon-ffmpeg__$arch.rpm
- sophon-mw-sophon-ffmpeg-dev__$arch.rpm
- sophon-mw-sophon-opencv__$arch.rpm
- sophon-mw-sophon-opencv-dev__$arch.rpm

Of which:

> sophon-ffmpeg/sophon-opencv contains the ffmpeg/opencv runtime environment (libraries, tools, etc.); sophon-ffmpeg-dev/sophon-opencv-dev contains the development environment (headers, pkgconfig, cmake, etc.). If you’re just installing on a deployment environment, you don’t need to install sophon-ffmpeg-dev/sophon-opencv-dev.
>
> sophon-mw-sophon-ffmpeg depends on the sophon-libsophon package, and sophon-mw-sophon-opencv depends on sophon-mw-sophon-ffmpeg, so libsophon must be installed first in the installation order, then sophon-mw-sophon-ffmpeg, Finally install sophon-mw-sophon-opencv.
>
> the libstdc++ library used in CentOS system uses the old version ABI interface, please use the sophon-mw-sophon-opencv-abi0__$arch.rpm installation package to install sophon-mw-sophon-opencv.

The installation steps are as follows:

install libsophon dependencies (refer to "Libsophon_user_guide")
install sophon-mw

If using Debian/Ubuntu system:

sudo dpkg -i sophon-mw-sophon-ffmpeg__$arch.deb sophon-mw-sophon-ffmpeg-dev__$arch.deb
sudo dpkg -i sophon-mw-sophon-opencv__$arch.deb sophon-mw-sophon-opencv-dev__$arch.deb

If using CentOS system:

sudo rpm -ivh sophon-mw-sophon-ffmpeg__$arch.rpm sophon-mw-sophon-ffmpeg-dev__$arch.rpm
sudo rpm -ivh sophon-mw-sophon-opencv-abi0__$arch.rpm sophon-mw-sophon-opencv-abi0-dev__$arch.rpm

If using Fedora system, please uninstall the installed package before installing:

The directory of installed packages can be enumerated by this command:
dnf list installed | grep sophon-ffmpeg

Then uninstall:
sudo rpm -e sophon-mw-sophon-opencv-dev
sudo rpm -e sophon-mw-sophon-opencv
sudo rpm -e sophon-mw-sophon-ffmpeg-dev
sudo rpm -e sophon-mw-sophon-ffmpeg

And install:
sudo rpm -ivh sophon-mw-sophon-ffmpeg__$arch.rpm sophon-mw-sophon-ffmpeg-dev__$arch.rpm
sudo rpm -ivh sophon-mw-sophon-opencv__$arch.rpm sophon-mw-sophon-opencv-dev__$arch.rpm

Execute the following command in the terminal, or logout and then login to the current user to use the installed tools:
source /etc/profile

Note: System in SOC mode is pre-installed with：

sophon-mw-soc-sophon-ffmpeg

sophon-mw-soc-sophon-opencv

Please follow the above steps to install：

sophon-mw-soc-sophon-ffmpeg-dev__arm64.deb

sophon-mw-soc-sophon-opencv-dev__arm64.deb

The installation location is:

```
/opt/sophon/
├──── libsophon-
├──── libsophon-current -> /opt/sophon/libsophon_|ver|
├──── sophon-ffmpeg_|ver|
│    ├──── bin
│    ├──── data
│    ├──── include
│    ├──── lib
│    │    ├──── cmake
│    │    └──── pkgconfig
│    └──── share
├──── sophon-ffmpeg-latest -> /opt/sophon/sophon-ffmpeg_|ver|
├──── sophon-opencv_|ver|
```

```
│   ├── bin
│   ├── data
│   ├── include
│   ├── lib
│   │   ├── cmake
│   │   │   └── opencv4
│   │   └── pkgconfig
│   ├── opencv-python
│   ├── share
│   └── test
└── sophon-opencv-latest -> /opt/sophon/sophon-opencv_|ver|
```

The deb package installation method does not allow you to install multiple different versions of the same package, but you may have placed several different versions under /opt/sophon in other ways. When installing with the deb package, /opt/sophon/sophon-ffmpeg-latest and /opt/sophon/sophon-opencv-latest will point to the last installed version. After uninstallation, it will point to the latest version remaining(if any).

The include and lib/cmake lib/pkgconfig directories are generated by the sophon-mw-sophon-ffmpeg-dev and sophon-mw-sophon-opencv-dev packages respectively.

How to uninstall:

If using Debian/Ubuntu system:

```
sudo apt remove sophon-mw-sophon-opencv-dev sophon-mw-sophon-opencv
sudo apt remove sophon-mw-sophon-ffmpeg-dev sophon-mw-sophon-ffmpeg
or:
sudo dpkg -r sophon-mw-sophon-opencv-dev
sudo dpkg -r sophon-mw-sophon-opencv
sudo dpkg -r sophon-mw-sophon-ffmpeg-dev
sudo dpkg -r sophon-mw-sophon-ffmpeg
```

If using CentOS system:

```
sudo rpm -e sophon-mw-sophon-opencv-abi0-dev
sudo rpm -e sophon-mw-sophon-opencv-abi0
sudo rpm -e sophon-mw-sophon-ffmpeg-dev
sudo rpm -e sophon-mw-sophon-ffmpeg
```

If using Fedora system:

```
sudo rpm -e sophon-mw-sophon-opencv-dev
sudo rpm -e sophon-mw-sophon-opencv
sudo rpm -e sophon-mw-sophon-ffmpeg-dev
sudo rpm -e sophon-mw-sophon-ffmpeg
```

If using other Linux systems:

The installation package consists of one file:

- sophon-mw__$system.tar.gz

It can be installed by the following steps:

Firstly, install the libsophon package according to the

“Libsophon_user_guide”, then:

```
tar -xzvf sophon-mw__$system.tar.gz
sudo cp -r sophon-mw__$system/* /
sudo ln -s /opt/sophon/sophon-ffmpeg_  /opt/sophon/sophon-ffmpeg-latest
sudo ln -s /opt/sophon/sophon-opencv_  /opt/sophon/sophon-opencv-latest
sudo ln -s /opt/sophon/sophon-sample_  /opt/sophon/sophon-sample-latest
sudo sed -i "s/usr\/local/opt\/sophon\/sophon-ffmpeg-latest/g"   /opt/sophon/sophon-
ffmpeg-latest/lib/pkgconfig/*.pc
sudo sed -i "s/^prefix=.*$/prefix=\/opt\/sophon\/sophon-opencv-latest/g"   /opt/
↪sophon/sophon-opencv-latest/lib/pkgconfig/opencv4.pc
```

Finally, install the bz2 libc6 libgcc library (this part needs to select the corresponding installation package according to the operating system, which is not uniformly introduced here)

Then some configuration work:

```
Add library and executable file path:
sudo cp /opt/sophon/sophon-ffmpeg-latest/data/01_sophon-ffmpeg.conf  /etc/ld.so.conf.d/
sudo cp /opt/sophon/sophon-opencv-latest/data/02_sophon-opencv.conf  /etc/ld.so.conf.d/
sudo ldconfig
sudo cp /opt/sophon/sophon-ffmpeg-latest/data/sophon-ffmpeg-autoconf.sh   /etc/profile.d/
sudo cp /opt/sophon/sophon-opencv-latest/data/sophon-opencv-autoconf.sh   /etc/profile.d/
sudo cp /opt/sophon/sophon-sample-latest/data/sophon-sample-autoconf.sh   /etc/profile.d/
source /etc/profile
```

How to uninstall:

```
sudo rm -f /etc/ld.so.conf.d/01_sophon-ffmpeg.conf
sudo rm -f /etc/ld.so.conf.d/02_sophon-opencv.conf
sudo ldconfig
sudo rm -f /etc/profile.d/sophon-ffmpeg-autoconf.sh
sudo rm -f /etc/profile.d/sophon-opencv-autoconf.sh
sudo rm -f /etc/profile.d/sophon-sample-autoconf.sh
sudo rm -f /opt/sophon/sophon-ffmpeg-latest
sudo rm -f /opt/sophon/sophon-opencv-latest
sudo rm -f /opt/sophon/sophon-sample-latest
sudo rm -rf /opt/sophon/sophon-ffmpeg_
sudo rm -rf /opt/sophon/sophon-opencv_
sudo rm -rf /opt/sophon/sophon-sample_
sudo rm -rf /opt/sophon/opencv-bmcpu_
```

note

- **if use python interface of sophon-opencv, set env by yourself:**

```
export PYTHONPATH=$PYTHONPATH:/opt/sophon/sophon-opencv-
latest/opencv-python
```

CHAPTER 4

Using sophon-sample

sophon-sample offers different types of installations on different Linux distributions. Please choose the corresponding installation according to your system, do not mix multiple installations on one machine. The “” in the following description is only an example, and may vary depending on the actual installed version.

Please replace $arch $system below with proper architecture on installation machine.

- If host processor is INTEL or AMD, replace $arch with amd64 and $system with x86_64
- If host processor is ARM64 or Phytium, replace $arch with arm64 and $system with aarch64
- If host processor is SG2042, replace $arch with riscv64 and $system with riscv_64

If using Debian/Ubuntu system:

The installation package consists of one file:

- sophon-mw-sophon-sample__$arch.deb

If using Fedora/CentOS system:

The installation package consists of one file:

- sophon-mw-sophon-sample__$arch.rpm

Of which：

- sophon-mw-sophon-sample contains applications for testing sophon-ffmpeg/sophon-opencv;

- sophon-mw-sophon-sample depends on sophon-ffmpeg/sophon-opencv package。

The installation steps are as follows:

```
install libsophon dependencies (refer to "Libsophon_user_guide")
install sophon-mw
install sophon-sample
```

If using Debian/Ubuntu system:

- sudo dpkg -i sophon-mw-sophon-sample__$arch.deb

If using Fedora/CentOS system, please uninstall the installed package before installing:

The directory of installed packages can be enumerated by this command:

- dnf list installed | grep sophon-ffmpeg

Then uninstall:

- sudo rpm -e sophon-mw-sophon-sample__$arch.rpm

And install:

- sudo rpm -ivh sophon-mw-sophon-sample__$arch.rpm

Installation location is:

```
/opt/sophon/
├── libsophon-  ├── libsophon-current -> /opt/sophon/libsophon  ├── sophon-ffmpeg  ├── sophon-ffmpeg-latest -> /opt/sophon/sophon-ffmpeg  ├── sophon-opencv  ├── sophon-opencv-latest -> /opt/sophon/sophon-opencv  ├── sophon-sample_  |   ├── bin
|   |   ├── test_bm_restart
|   |   ├── test_ff_bmcv_transcode
|   |   ├── test_ff_scale_transcode
|   |   ├── test_ff_video_encode
|   |   ├── test_ff_video_xcode
|   |   ├── test_ocv_jpubasic
|   |   ├── test_ocv_jpumulti
|   |   ├── test_ocv_vidbasic
|   |   ├── test_ocv_video_xcode
|   |   ├── test_ocv_vidmulti
|   |   └── water.bin
|   ├── data
|   └── sample
└── sophon-sample-latest -> /opt/sophon/sophon-sample_
```

The deb package installation method does not allow you to install multiple different versions of the same package, but you may have placed several different versions under

/opt/sophon in other ways. When installing with the deb package, /opt/sophon/sophon-ffmpeg-latest and /opt/sophon/sophon-opencv-latest will point to the last installed version. After uninstallation, it will point to the latest version remaining(if any).

Note: In SoC mode, the DEB package name is

sophon-mw-soc-sophon-sample__arm64.deb

The installation position is same.

Uninstall:

If using Debian/Ubuntu system:

- sudo apt remove sophon-mw-sophon-sample or:
- sudo dpkg -r sophon-mw-sophon-sample__$arch.deb

If using Fedora/CentOS system:

- sudo rpm -e sophon-mw-sophon-sample__$arch.rpm

Introduction:

Test case introduction:

This use case is mainly used to test the video decoding function and performance under the ffmpeg module, and supports multiplexing and disconnection reconnection. Users can monitor the decoding of video and code streams through use cases.

test_bm_restart [api_version] [yuv_format] [pre_allocation_frame] [codec_name] [sophon_idx] [zero_copy] [input_file/url] [input_file/url]

-api_version	Specify the version of ffmpegAPI used for the decoding process 0: Use the old version of decoding avcodec_decode_video2 interface 1: Use the new version of decoding avcodec_send_packet interface 2: Use av_parser_parse2 API for packet capture
-yuv_format	Whether to compress data, 0 means no compression, 1 means compression
-pre_allocation_frame	The number of cache frames allowed, up to a maximum of 64
-codec_name	Specify the decoder, select h264_bm/hevc_bm, no is not specified
-sophon_idx	In SOC mode, this option can be set arbitrarily (not null) and its value will be ignored
-zero_copy	If it is in SOC mode, 0 means that Host memory is enabled, 1 means that if it is not enabled if it is in PCIE mode, this

option can be set arbitrarily (not null), and its value will be ignored;

-input_fileorurl Enter the file path or bitstream address

e.g

test_bm_restart 1 0 1 no 0 1 ./example0.mp4 ./example1.mp4 ./example2.mp4

test_ff_bmcv_transcode

This use case is mainly used to test the video transcoding function and performance under the ffmpeg module, and realize the transcoding process of decoding first and then encoding by calling the ff_video_decode and ff_video_encode the data type and function in the use case, so as to ensure the correctness of the decoding and encoding functions. At the same time, this use case can also test the transcoding performance under ffmpeg, and the runtime program will output the instant transcoding frame rate for reference.

test_ff_bmcv_transcode [platform] [src_filename] [output_filename] [encode_pixel_format] [codecer_name] [width] [height] [frame_rate] [bitrate] [thread_num] [zero_copy] [sophon_idx] <optional: enable_mosaic> <optional: enable_watermark>

parameters：

-platform platform: soc or pcie

-src_filename input file name x.mp4 x.ts…

-output_filename encode output file name x.mp4,x.ts…

-encode_pixel_format encode format I420.

-encoder_name encode h264_bm,h265_bm.

-width encode 32<width<=4096.

-height encode 32<height<=4096.

-frame_rate encode frame_rate.

-bitrate encode bitrate 500 < bitrate < 10000

-thread_num thread num

-zero_copy PCie platform: 0: copy host mem, 1: nocopy. SoC platform: any but invalid

-sophon_idx PCie platform: sophon devices idx SoC platform: any but invalid

-enable_mosaic Optional, add mosaic on the left_top corner, only bm1686 support now and [encode_pixel_format] need I420

-enable_watermark Optional, add watermark on video, only bm1686 support now and [encode_pixel_format] need I420

e.g

pcie mode example:

test_ff_bmcv_transcode pcie example.mp4 test.ts I420 h264_bm 800 400 25 3000 3 0 0

soc mode example:

test_ff_bmcv_transcode soc example.mp4 test.ts I420 h264_bm 800 400 25 3000 3 0 0

test_ff_scale_transcode

This use case is mainly used to test the function and performance of video transcoding under ffmpeg. This function is realized through the process of first decoding and then encoding, mainly calling the data types and functions in ff_video_decode and ff_video_encode.

test_ff_scale_transcode [src_filename] [output_filename] [encode_pixel_format] [codecer_name] [height] [width] [frame_rate] [bitrate] [thread_num] [zero_copy] [sophon_idx]

parameters:

-src_filename input file name x.mp4 x.ts…

-output_filename encode output file name x.mp4,x.ts…

-encode_pixel_format encode format I420.

-codecer_name encode h264_bm,hevc_bm,h265_bm.

-height encode height.

-width encode width

-frame_rate encode frame_rate

-bitrate encode bitrate

-thread_num thread num

-zero_copy 0: copy host mem,1: nocopy.

-sophon_idx sophon devices idx

e.g

test_ff_scale_transcode example.mp4 test.ts I420 h264_bm 800 400 25 3000 3 0 0

test_ff_video_encode

This use case is mainly used to test the encoding function of video under the ffmpeg module. The input video is limited to I420 and NV12 formats. By calling this use case, users can get encapsulated video files, which are supported by FFMPEG.

test_ff_video_encode <input file> <output file> <encoder> <width> <height> <roi_enable> <input pixel format> <bitrate(kbps)> <frame rate> <sophon device index>

-input_file Enter the file path or bitstream address

-output_file encode output file name

-encoder H264(default), H265.

-width video width, output and input need to be the same, 256 < = width < = 8192

-height video height, output and input must be the same, 128 < = height < = 8192

-roi_enable 0 disable(default), 1 enable roi.

-input_pixel_format I420(YUV, default), NV12. YUV

-bitrate Output bitrate, 10 < bitrate <= 100000, default frame rate x width x height/8

-framerate framerate,10 < framerate <= 60,default 30

-sophon_device_index used in PCIE mode. min valuse is 0.

e.g

- test_ff_video_encode <input file> <output file> H264 width height 0 I420 3000 30 2
- test_ff_video_encode <input file> <output file> H264 width height 0 I420 3000 30
- test_ff_video_encode <input file> <output file> H265 width height 0 I420
- test_ff_video_encode <input file> <output file> H265 width height 0 NV12
- test_ff_video_encode <input file> <output file> H265 width height 0

test_ff_video_xcode

This use case is mainly used to test the functionality and performance of video transcoding under FFMPEG. This function is achieved through a process of decoding and then encoding, mainly calling the data types and functions of the ff_video_decode, ff_video_encode. The resolution of the transcoded video is the same as that of the original video, and the bitrate cannot exceed 10000kbps or less than 500kbps, otherwise the default value of 3000kbps will be set. If the bitrate of the transcoded video is the same as the original video, the duration should also be the same. Some dropped frames are normal.

test_ff_video_xcode <input file> <output file> encoder framerate bitrate(kbps) isdmabuffer pcie_no_copyback sophon_idx

-input_file Enter the file path or bitstream address

-output_file output file name

-encoder H264 or H265

-isdmabuffer 1, no memory copy in encoder. Or 0, has memory copy.

In PCIE module:

-pcie_no_copyback 1, do not copy decoded raw YUV to host. Or 0, copy decoded raw YUV to host.

-sophon_idx sophon devices idx

e.g

- test_ff_video_xcode ./file_example_MP4_1920_18MG.mp4 tran5.ts H264 30 3000 1 1 0
- test_ff_video_xcode ./file_example_MP4_1920_18MG.mp4 tran5.ts H264 30 3000 0 0 0
- test_ff_video_xcode ./file_example_MP4_1920_18MG.mp4 tran5.ts H264 30 3000 1 0 0

test_ocv_jpubasic

This use case is primarily used to test the basic functionality of image codecs.

test_ocv_jpubasic <file> <loop> <yuv_enable> <dump_enable> [card]

-file intput file

-loop loop number

-yuv_enable 0 decode output BGR; 1 decode output YUV.

-dump_enable 0 no dump file; 1 output dump file

-card Optional in PCIe mode. Specifies the card number of the run

e.g

test_ocv_jpubasic 1920x1080_gray.jpg 1000 1 1 0

test_ocv_jpumulti

This use case is primarily used to test the ability to record video in PNG or JPG format. If dump is enabled, raw data in BGR or YUV format can also be output.

test_ocv_jpumulti <test type> <inputfile> <loop> <num_threads> <outjpg> [card]

-inputfile input_file path

-test_type 1-only dec 2-only enc 3-mix

-loop loop number

-num_thread 1 <= num_threads <= 12

-outjpg 1 output jpg, 0 disable output jpg

-card Optional in PCIe mode. Specifies the card number of the run

e.g

test_ocv_jpumulti 3 1088test1_420.jpg 10 2 1 0

test_ocv_vidbasic

This use case is primarily used to test the ability to record video in PNG or JPG format. If dump is enabled, raw data in BGR or YUV format can also be output.

test_ocv_vidbasic <input_video> <output_name> <frame_num> <yuv_enable> [card] [WxH] [dump.BGR or dump.YUV]

-input_video	input_video
-output_name	Output file name prefix
-frame_num	frame rate of output
-yuv_enable	0 decode output BGR; 1 decode output YUV.
-card	Optional in PCIe mode. Specifies the card number of the run
-WxH	Optional parameter to specify the resolution of the output file, if not specified, it is consistent with the original video resolution
-dumpBGR_or_dumpYUV	Optional parameter, whether to output dump. BGR or dump. YUV file

e.g

test_ocv_vidbasic station.avi test_basic 10 1 0 1920x1080

test_ocv_video_xcode

This use case is mainly used for the all-round test of video transcoding under the OpenCV module, providing H264, H265 encoding, YUV format output, ROI region and other functions and multi-format output for video input in different formats supported by OpenCV Library, covering a wide range of formats and functions.

test_ocv_video_xcode <input> <code_type> <frame_num> <outputname> <yuv_enable> <roi_enable> [device_id] [encodeparams]

-input	input file
-code_type	H264enc is h264; H265enc is h265.
-frame_num	num of frame which to deal
-outputname	null or NULL output pkt.dump.
-yuv_enable	0 decode output bgr; 1 decode output yuv420.
-roi_enable	0 disable roi encoder; 1 enable roi encoder. if roi_enable is 1, you should set null/Null in outputname and set roi_enable=1 in encodeparams.

-encodeparams Optional parameters, you can set roi_enable, bitrate, min_qp, GOP and other parameters

-device_id Optional in PCIe mode. Specifies the card number of the run

e.g

test_ocv_video_xcode rtsp_url H265enc 10000 encoder_test265.ts 1 0 0 bitrate=1000

test_ocv_vidmulti This use case is mainly used for the encoding and decoding performance test of video and code streams under the OpenCV module. Test the performance of the device when transcoding video multiplexing, The terminal can output the number of each thread, input video resolution, number of reconnections, The current number of decoded frames, real-time frame rate, total average frame rate, number of dropped frames, etc., as well as information about the encoding function.

test_ocv_vidmulti <thread_num> <input_video> [card] [enc_enable] <input_video> [card] [enc_enable]

-thread_num number of thread,less than 512

-enc_enable Whether encoding is turned on. 0 means no on, 1 means on

-card Optional in PCIe mode. Specifies the card number of the run

Environment variables:

export VIDMULTI_DISPLAY_FRAMERATE=0 display frame number only

export VIDMULTI_DISPLAY_FRAMERATE=1 display dtailed information of each channel, can see the dropped frames, etc

e.g

test_ocv_vidmulti 3 a.264 0 1 b.264 1 1 c.264 0 0

CHAPTER 5

Develop with sophon-mw

After installing sophon-mw, users can link sophon-mw’ s libraries into their own compiled programs in two ways.

** If using Make compile system **

It is recommended that users use pkgconfig to find the sophon-mw library.

In the previous installation, we have added the pkgconfig path of sophon-mw to the environment variable PKG_CONFIG_PATH. Therefore, users can add the following statement to the Makefile:

```
CFLAGS   = -std=c++11

# add libsophon dependency because sophon-ffmpeg rely on it
CFLAGS  += -I/opt/sophon/libsophon-current/include/
LDFLAGS += -L/opt/sophon/libsophon-current/lib -lbmcv -lbmlib -lbmvideo -lbmvpuapi -
↪lbmvpulite -lbmjpuapi -lbmjpulite -lbmion

# add sophon-ffmpeg
CFLAGS  += $(shell pkg-config --cflags libavcodec libavformat libavfilter libavutil libswscale)
LDFLAGS += $(shell pkg-config --libs libavcodec libavformat libavfilter libavutil libswscale)

# add sophon-opencv
CFLAGS  += $(shell pkg-config --cflags opencv4)
LDFLAGS += $(shell pkg-config --libs opencv4)
```

Then you can use the sophon-ffmpeg and sophon-opencv libraries in the Makefile.

Note: When another copy of ffmpeg or opencv is installed under /usr/lib or /usr/local/lib, pay attention to check whether the correct sophon-ffmpeg/sophon-opencv path is

found. If the search is not correct, you need to specify the header file location and library file location explicitly

** If using CMake compile system **

Users can add the following statements in CMakeLists.txt:

```
# add libsophon
find_package(libsophon REQUIRED)
include_directories(${LIBSOPHON_INCLUDE_DIRS})
link_directories(${LIBSOPHON_LIB_DIRS})

# add sophon-ffmpeg
set(FFMPEG_DIR      /opt/sophon/sophon-ffmpeg-latest/lib/cmake)
find_package(FFMPEG REQUIRED NO_DEFAULT_PATH)
include_directories(${FFMPEG_INCLUDE_DIRS})
link_directories(${FFMPEG_LIB_DIRS})

# add sophon-opencv
set(OpenCV_DIR /opt/sophon/sophon-opencv-latest/lib/cmake/opencv4)
find_package(OpenCV REQUIRED NO_DEFAULT_PATH)
include_directories(${OpenCV_INCLUDE_DIRS})

add_executable(${YOUR_TARGET_NAME} ${YOUR_SOURCE_FILES})

target_link_libraries(${YOUR_TARGET_NAME} ${FFMPEG_LIBS} ${OpenCV_LIBS})
```

The functions in sophon-ffmpeg and sophon-opencv can be called in the user’ s code:

```
#include <opencv2/opencv.hpp>

int main(int argc, char const *argv[])
{
  cv::Mat img = cv::imread(argv[1]);

  return 0;
}
```