GMLIB PROGRAMMING GUIDE

Programming Guide
Rev.: 1.10
Issue Date: December 2014

REVISION HISTORY

GMLIB Programming Guide

Date	Rev.	From	To
Mar. 2013	1.0	-	Original
Jul. 2013	1.1	-	Revised APIs, structure, and attribute to match the latest GM_LIB
Sept. 2013	1.2	-	Revised APIs, structure, and attribute to match the latest GM_LIB
Oct. 2013	1.3	-	• Added appendix for “/proc and debug” • Revised the structure for matching gmlib.h • Supported only one group for encoding multi-channel • Supported only the object of the capture for the OSD interface
Nov. 2013	1.4	-	• Added the notification API in Section 2.1.5 • Updated Sections 1.3.2, 1.3.3, 3.2.1, 3.2.37, 4.1.1, 4.6.1, 4.7.1, and 4.7.2
Dec. 2013	1.5	-	• Removed the support of marquee in Section 3.2.17
Jan. 2014	1.6	-	• Revised the descriptions in Sections 2.1.1, 2.1.2, 2.1.3, 2.1.5, 2.1.7, 2.1.10, 2.1.11, 2.1.14, 2.1.15, 2.2.4, 2.2.5, 2.2.6, 2.2.7, and 2.2.15 • Removed the unused H.264 motion detection in Sections 3.2.8, 3.2.9, and 4.2.1
Jan. 2014	1.7	-	To be suitable for GM series ICs
Mar. 2014	1.8	-	• Added gm_set_osd_font2() in Section 2.2.11 • Added gm_set_cap_tamper() in Section 2.2.18 • Synchronized with gmlib.h
Sept. 2014	1.9	-	• Updated Sections 2.1.1, 3.2.5, 3.2.6, 3.2.9, 3.2.11, 3.2.17, 3.2.18, 3.2.22, 3.2.23, 3.2.26, 3.2.30, 3.2.31, 3.2.33, 4.1.1, 4.2.1, 4.2.6, 4.3.1, 4.4.1, 4.6.1, and 4.7.1 • Added Sections 2.2.20, 2.2.21, 2.2.22, 3.2.39, 3.2.40, and 5.1.12 ~ 5.1.17
Dec. 2014	1.10		• Added Sections 3.2.42, 3.2.43, 3.2.44, 4.1.4, and 4.1.5 • Revised Sections 4.1.1, 4.1.2, 4.1.3, 4.2.1, 4.2.3, and 4.4.1 • Deleted Section 4.7.2

Printed in Taiwan 2014

Grain Media, Inc.
5F, No. 5, Li-Hsin Road III, Hsinchu Science Park, Hsinchu City, Taiwan 300, R.O.C.

Grain Media's home page can be found at:
http://www.grain-media.com

TABLE OF CONTENTS

LIST OF TABLE

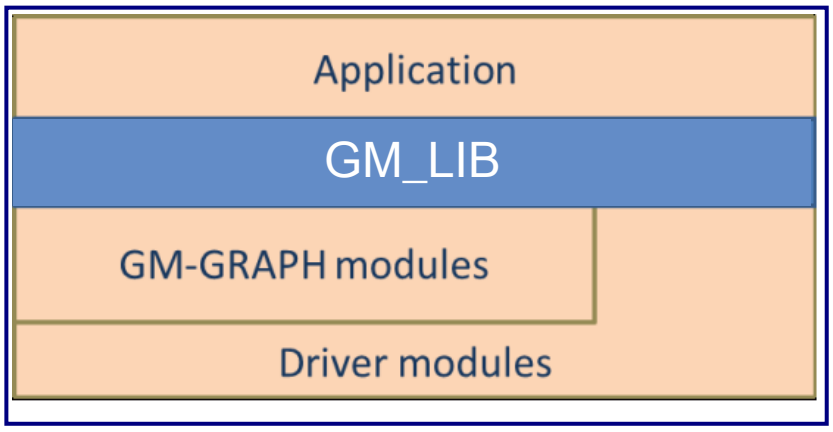
Chapter 1

Introduction

This chapter contains the following sections:

1.1 Introduction

GM_LIB is a library for users to access the Grain Media IC functionality from the user space. By using this library, users can easily implement many surveillance functions, such as Liveview, Record, and Playback. Applications can communicate with under layer via GM_LIB. In the Grain Media IC system, GM_LIB is part of GM-GRAPH (Please refer to the note listed below), which coordinates all modules to work together. From the following figure, users will see the position of GM_LIB in the whole software architecture.



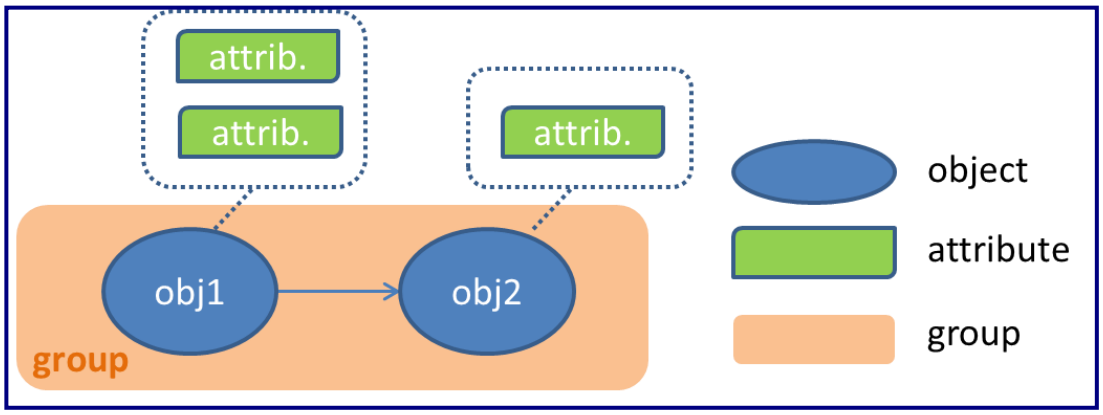
The files used in each layer are described below:

- Application: Users write to own application in the user level. Any communication is conducted via the GM_LIB interface.
- GM_LIB: It is responsible for receiving the request from the application layer; then translates the request and sends new command to the GM-GRAPH modules or drivers in the Linux kernel space.
- GM-GRAPH modules: This graph core is used to schedule tasks and prepare jobs for the hardware modules.
- Driver modules: The Linux drivers are used to control the hardware.

Note: In the previous SDK (Such as GM818x or GM812x), the software architecture is named as VideoGraph. It is renamed to GM-GRAPH from GM821x SDK.

1.2 Use GM_LIB

GM_LIB adopts a new design method. Unlike the previous SDK, new design is to treat each input or output as an independent object. Therefore, the video capture input or display window is an object. The file input or bitstream output is also an object. Users can set the attributes to an object and combine some of the attributes together to form the desired functions, and this is called a binding set 'group'. Please note that this group can be run by "apply".



Generally, users should follow the steps listed below to form the desired functions.

Start the feature

1. Create objects
2. Set the attribute to each object, if needed
3. Create a group
4. Bind objects to this group
5. Apply this group to be active
6. Send or receive data from the user functions, if needed

Change settings

1. Set the attribute to each object, if needed
2. Unbind objects, if needed
3. Apply the group

Stop

1. Stop sending or receiving data by user functions
2. Unbind objects
3. Apply the group
4. Delete objects and group

1.2.1 Object

gm_new_obj is used with a given object type to create an object. The object will be assigned with a channel number by the attribute (Described in the next section).

The following table lists the objects implemented by GM_LIB.

Object	Object name defined by GM_LIB	
Video capture	GM_CAP_OBJECT	Capture object is an input object for Liveview, which is bound with the window object. Capture object is an input object for Record, which is bound with the encoder object.
Video encoder	GM_ENCODER_OBJECT	Encoder object is an output object for Record, which is bound with the capture object.
Video window	GM_WIN_OBJECT	Window object is an output object for Liveview, which is bound with the capture object. Window object is an output object for Playback, which is bound with the file object.
Video bitstream	GM_FILE_OBJECT	File object is an input object for Playback, which is bound with the window object.
Audio grabber	GM_AUDIO_GRAB_OBJECT	Audio grab object is an input object for audio recorder, which is bound with the audio encoder object (GM_AUDIO_ENCODER_OBJECT).
Audio encoder	GM_AUDIO_ENCODER_OBJECT	Audio encoder object is an output object for audio recorder, which is bound with the audio encoder object.
Audio renderer	GM_AUDIO_RENDER_OBJECT	Audio render object is and output object for audio playback, which is bound with the audio file object.

1.2.2 Attribute

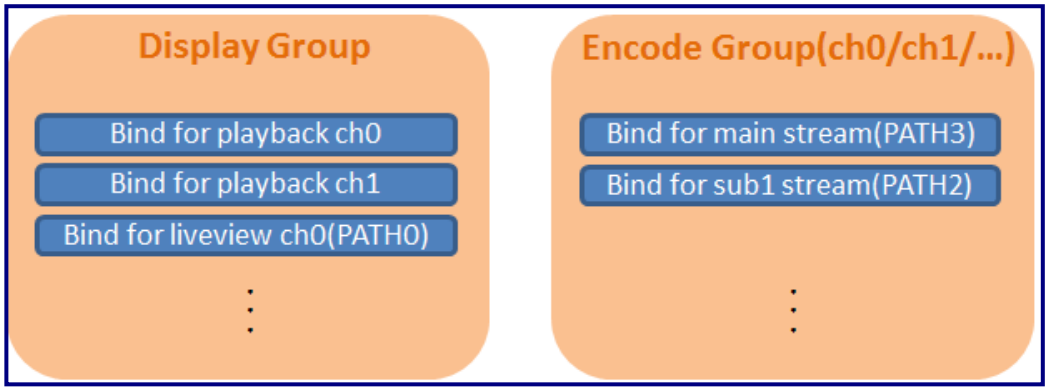
An attribute represents the property set for an object and it can be dynamically created by the DECLARE_ATTR macro with a given attribute type. Attributes have two types: Basic type and advanced type. The basic type means "must have arguments" for an object, while the advanced type is optional for the advanced features. Users cannot assign all types of attributes to one object. Please refer to the following table for correlation between the attribute types and objects.

Attribute Type	Valid Object
gm_cap_attr_t	GM_CAP_OBJECT
gm_3di_attr_t	GM_CAP_OBJECT
gm_crop_attr_t	GM_CAP_OBJECT/GM_FILE_OBJECT
gm_h264e_attr_t	GM_ENCODER_OBJECT
gm_mpeg4e_attr_t	GM_ENCODER_OBJECT
gm_mjpege_attr_t	GM_ENCODER_OBJECT
gm_enc_roi_attr_t	GM_ENCODER_OBJECT
gm_win_attr_t	GM_WIN_OBJECT
gm_win_aspect_ratio_attr_t	GM_WIN_OBJECT
gm_win_border_attr_t	GM_WIN_OBJECT
gm_file_attr_t	GM_FILE_OBJECT
gm_audio_grab_attr_t	GM_AUDIO_GRAB_OBJECT
gm_audio_enc_attr_t	GM_AUDIO_ENCODER_OBJECT
gm_audio_render_attr_t	GM_AUDIO_RENDER_OBJECT
gm_h264_watermark_attr_t	GM_ENCODER_OBJECT
gm_h264_vui_attr_t	GM_ENCODER_OBJECT
gm_h264_advanced_attr_t	GM_ENCODER_OBJECT
gm_h264_roiqp_attr_t	GM_ENCODER_OBJECT

1.2.3 Bind and Group

Before binding, a group must be created first. Users can select two objects and combine them by using the gm_bind function. The group means "things have relationship and will be treated as working together". Users should separate groups for different purposes in an application to have better performance. Grain Media suggests users arranging groups by following the rules listed below.

- PATH is a duplicate unit for a specific capture channel.
- Each PATH number has been assigned to a specific application. Capture path_0 (Liveview), path_1 (Reserved, do not use), path_2 (Substream), and path_3 (Mainstream)
- Display is a big group even if there are many different channels (Including Playback or Liveview).
- All Record channels (Including video or audio) belong to one group. The substream channel uses the same group.
- Each Playback channel (Including video or audio) belongs to own group.



1.2.4 Apply

A group can be created by calling the gm_apply function. Users can only call "apply" for one group for the multi-channel operations, such as the multi-channel record.

1.2.5 Samples

Sample of Liveview

```
#include "gmlib.h"

gm_init();
groupfd = gm_new_groupfd();
cap_obj = gm_new_obj(GM_CAP_OBJECT);
win_obj = gm_new_obj(GM_WIN_OBJECT);
…
gm_set_attr(cap_obj, &capAttr);
…
gm_set_attr(win_obj, &winAttr);
…
bindfd = gm_bind(groupfd, cap_obj, win_obj);
gm_apply(groupfd);
```

Sample of Record

```
#include "gmlib.h"

gm_init();
groupfd = gm_new_groupfd();
cap_obj = gm_new_obj(GM_CAP_OBJECT);
enc_obj = gm_new_obj(GM_ENCODER_OBJECT);
…
gm_set_attr(cap_obj, &capAttr);
…
gm_set_attr(enc_obj, &encAttr);
…
bindfd = gm_bind(groupfd, cap_obj, enc_obj);
gm_apply(groupfd);

while(…) {
    gm_poll(bindfd,…);
    ….
    gm_recv_multi_bitstreams(multi_enc, …);
```

Sample of Playback

```
#include "gmlib.h"

gm_init();
groupfd = gm_new_groupfd();
file_obj = gm_new_obj(GM_FILE_OBJECT);
win_obj = gm_new_obj(GM_WIN_OBJECT);
…
gm_set_attr(file_obj, &fileAttr);
…
gm_set_attr(win_obj, &winAttr);
…
bindfd = gm_bind(groupfd, file_obj, win_obj);
gm_apply(groupfd);

while(…) {
    gm_poll(bindfd, …);// Needn't use gm_poll in playback
    ….
    gm_send_multi_bitstreams(multi_dec, …); // when times up
    ….
}
```

1.3 Bitstream Operations

There are two kinds of flows that were called Playback and Record for handling bitstreams. The bitstream is the compressed frame data for storing media files or playing from media files. There will be more information about "how to put bitstream to GMLIB" or "how to get bitstream from GMLIB". Please refer to the following subsections.

1.3.1 Bitstream Thread

There is no limit to send or receive bitstream thread. In practice, it is common to create a bitstream thread. Users can create one thread to send/receive the multi-channel data or handle each channel in a private thread. For Record, gm_poll is used to monitor the availability of bitstream. Playback controls the time flow and does not need gm_poll (For more details, please refer to Section 1.3.3).

Sample of receiving bitstream

```
#define CH_CNT 16
gm_pollfd_t poll_fds[CH_CNT];
gm_enc_multi_bitstream_t multi_enc[CH_CNT];
….
while(continue receiving) {
    prepare for poll_fds[];
    ….
    gm_poll(poll_fds, CH_CNT, timeout value);
    ….
    for (i = 0; i< CH_CNT; i++ )
        gm_recv_multi_bitstreams(muti_enc, CH_CNT);
    ….
}
```

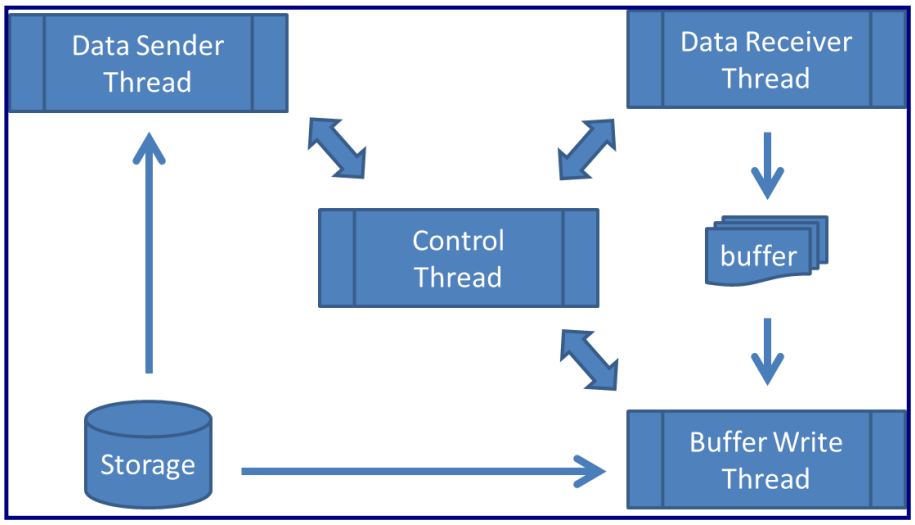
Sample of sending bitstream

```
#define USER_BUFFER_LEN 10000
char data[USER_BUFFER_LEN];
int length = USER_BUFFER_LEN;
gm_dec_multi_bitstream_t multi_dec[PB_MAX_CHANNEL];
….
while(continue sending) {

    ….
    gm_send_multi_bitstreams(multi_dec, PB_MAX_CHANNEL,
500);
    ….
}
```

1.3.2 Thread Arrangement

GM_LIB can be used in a multi-thread application, but it is not designed to run across processes. If the application is implemented in a multi-process design, it should always execute GM_LIB API within the specified process. Grain Media recommends users implementing the multi-thread application into the suggested design. When a user buffer queue mechanism is applied, this buffer queue will be used to avoid the frame being lost due to busy I/O. The following figure shows the operation of the thread arrangement.



Control Thread

This thread controls the whole system behavior, statemachine, and operations for feature start/update/stop by using gm_bind, gm_set_attr, gm_apply, and so on.

Data Receiver Thread

Generally, this thread is used for the frame encode. This thread may use "poll" to query data channels if the channels are available from under layer. After the data is ready, this thread will copy the bitstream data to a private user queue. They are operated by gm_poll, gm_recv_multi_bitstreams, and so on.

Buffer Write Thread

This thread is used to write data to the storage from the user buffer.

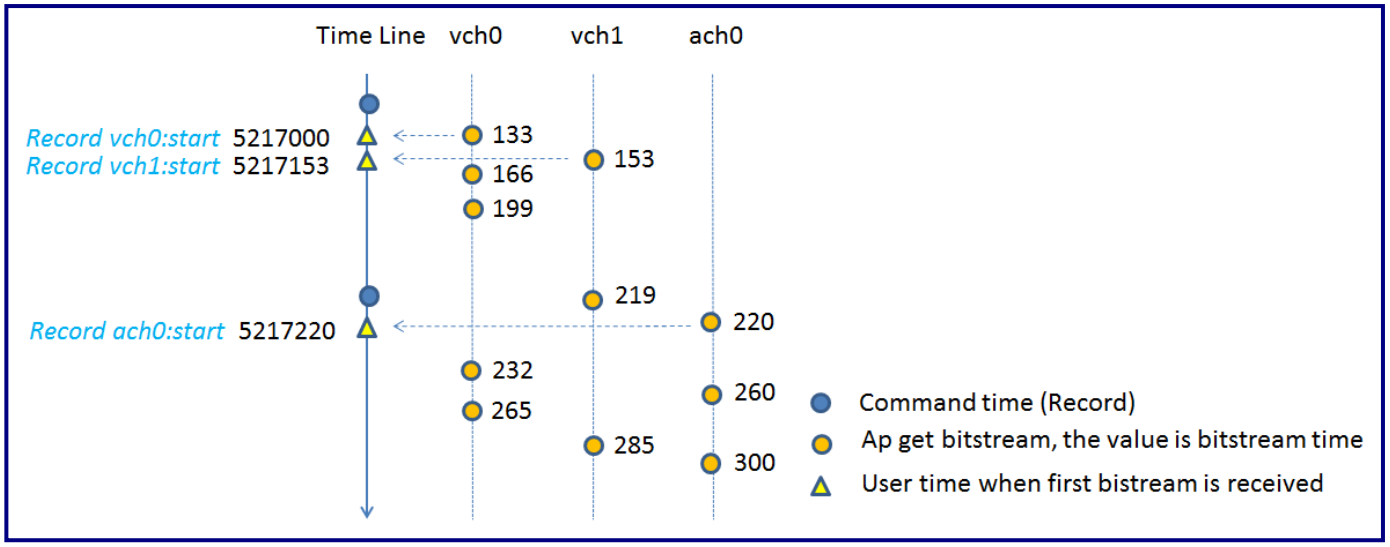
Data Sender Thread

Generally, this tread is used for the bitstream decoder. Each Playback channel has its own threads to control the playback speed according to the user requirements. Please note that Playback will not use gm_poll for waking up to send bitstreams.

1.3.3 Frame Time Control

In this section, users will know when and how to handle a bitstream by using additional timestamp information.

- Record - Receive bitstream from GM_LIB
  When users get a bitstream, the timestamp of this frame will also be filled in the same structure. The length of a timestamp is a 32bit integer and the timestamp is a reference time (Not an absolute time). Application should track the time from the first frame. The time of application that gets the first frame (Or called "user time", please refer to the note in the next page) is very important and application must keep this information. This is because that all later bitstreams in the same channel must refer to the information to identify the produced absolute time. Because the type of the timestamp is a 32bit unsigned integer with the resolution in millisecond, the application should carefully handle the overflow issue. Please note that the first frame may not start from zero. The picture in the next page is an example of this operation.

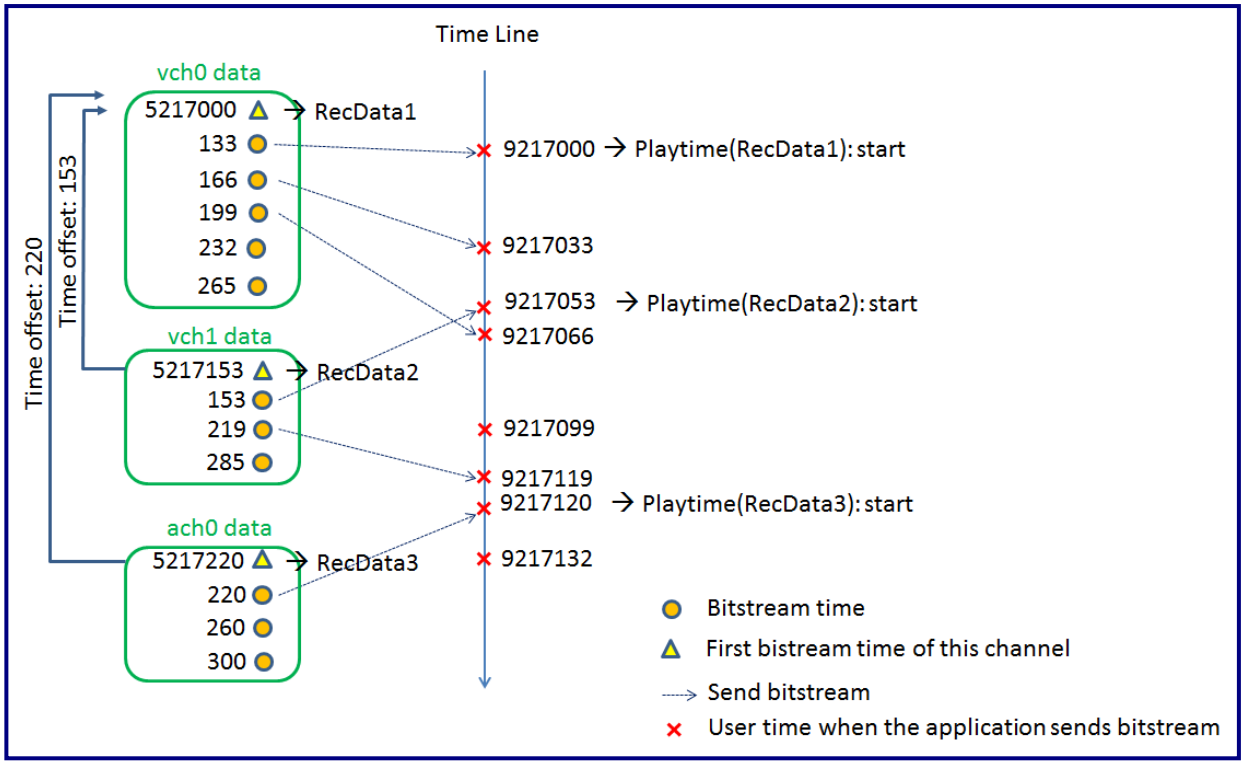


Users start the video channel 0 and channel 1 (vch0 and vch1) at different frame rates. At user time of 5217000, users can get frame1 of vch0 with the timestamp of 133. After frame1, users may get frame2 right away with the timestamp of 166. The absolute time of frame2 can be calculated as 5217033 (5217000 + 166 - 133). After that, video channel 1 (vch1) and new audio channel 0 (ach0) are the same. Users need to record the time of the first frame of each channel.

Note: User time can be obtained by the standard library, such as getting time of the day.

- Playback - Send bitstream to GM_LIB

  GM_LIB lets application to control the playback flow. Display is the target for the video frame and audio can specify the ID of display for synchronization. Application is responsible for calculating the interval between two adjacent frames and sending bitstream to GM_LIB. In other words, the sequence of the frames placed by application is in the sequence when they are rendered. The first frame on the display is treated as the base time of all playback channels, including video and audio. As mentioned in the previous section, an application has to use first frame time to derive the absolute playtime of the given frame. The picture in the next page is an example.



In this picture, there are three channels, vch0, vch1, and ach0. The playback begins from the first frame of vch0, and the timestamp is 133 relative to 5217000. Application is sending this frame to GM_LIB at the user time of 9217000. This means that playback is already started. By analyzing the data, the next frame is vch0 (166), and the timestamp is 33 after the previous vch0 frame. Therefore, application should place this frame at 9217033 (921700 + 33). The same operations are applied on all channels, including video and audio.

Chapter 2

GM_LIB API

This chapter contains the following sections:

Some of the strings are shown on the console by the API library. Sometimes, they are followed by short error messages and GM_LIB will inform users that error operations occurred. Software programmers should check and modify the software flow to avoid errors. Please note that when an error occurs, gm_lib will automatically recover for executing the next application. Sometimes, it needs to reboot the systems.

2.1 General API

General API is used to construct the surveillance video and audio flows. Users may refer to GM8xxx_sample_code.chm for more detailed usage.

2.1.1 gm_init

- Function prototype
  - int gm_init(void);
- Synopsis
  - Initialize the Grain Media library
- Returns

Value	Description
0	Success
Negative value (< 0)	Failure

- Error messages
  - Open "/dev/vpd" fail! Please check insert module driver flow.
  - Platform is not ready! Please check insert module driver flow.
  - gm_lib version error! Please check library version or re-compiler application program.
  - Platform type error! Please check "spec.cfg" or “gmlib.cfg” file.
- Sample code reference
  - Any sample

2.1.2 gm_release

- Function prototype
  - int gm_release(void);

- Synopsis
  - Release the Grain Media library

- Returns

Value	Description
0	Success
Negative value (< 0)	Failure

- Error messages
  - gm_lib not init! Please check "gm_init()".
  - groupfd exist! Please check "pif_delete_groupfd()".
  - Objects exist! Please check "gm_delete_obj()".

- Sample code reference
  - Any sample

2.1.3 gm_get_sysinfo

- Function prototype
  - int gm_get_sysinfo(gm_system_t *system_info);

- Synopsis
  - Get the system information

- Argument

Argument	Description	Input/Output
system_info	Pointer of system_info	Output

- Returns

Value	Description
0	Success
Negative value (< 0)	Failure

- Note
  - The function will get the system information when it returns successfully.

- Error message
  - Invalid system_info pointer!

- Sample code reference
  - Any sample

2.1.4 gm_new_obj

- Function prototype
  - void *gm_new_obj(gm_obj_type_t obj_type);

- Synopsis
  - Create a new object by obj_type

- Argument

Argument	Description	Input/Output
obj_type	Type of the object Please refer to gm_obj_type_t.	Input

- Return

Value	Description
obj	Pointer of an object

- Error messages
  - gm_lib not init! Please check "gm_init()".
  - Invalid object type!

- Sample code reference
  - Any sample

2.1.5 gm_delete_obj

- Function prototype
  - int gm_delete_obj(void *obj);

- Synopsis
  - Delete an exist object

- Argument

Argument	Description	Input/Output
obj	Pointer of the object	Input

- Returns

Value	Description
0	Success
Negative value (< 0)	Failure

- Error messages
  - Invalid object pointer
  - Invalid object type! Please check the object that may be invalid pointer.
  - Object is still working, need to "gm_unbind/gm_apply" first

- Sample code reference
  - Any sample

2.1.6 gm_bind

- Function prototype
  - void *gm_bind(void *groupfd, void *in_obj, void *out_obj);

- Synopsis
  - Bind in object to out object of groupfd

- Arguments

Argument	Description	Input/Output
groupfd	fd of the group	Input
in_obj	Pointer of in object	Input
out_obj	Pointer of out object	Input

- Return

Value	Description
fd	fd of the binding result

- Error messages
  - Invalid groupfd pointer
  - Invalid in object pointer! Please check "gm_new_obj()" flow.
  - Invalid out object pointer! Please check "gm_new_obj()" flow.
  - gm_lib not init! Please check "gm_init()".
  - Not support, in(In_Objec_Name) out(Out_Object_Name) binding!
  - Not support, exist same out objects(Objec_Name)!
  - In group, there is 2 cap objects attribute parameter, vch(%d) and path(%d) are the same value, please check gm_set_attr() flow.
  - Not support: One (Objec_Name) bind to multiple (Objec_Name) objects!
  - Not support: multiple (Objec_Name) objects bind to (Objec_Name)!
  - Not support, exist same vch(%d) of file object!

- Sample code reference
  - Any sample

2.1.7 gm_unbind

- Function prototype
  - int gm_unbind(void *bindfd);

- Synopsis
  - Unbind bindfd

- Argument

Argument	Description	Input/Output
bindfd	fd of gm_bind	Input

- Returns

Value	Description
0	Success
Negative value (< 0)	Failure

- Error messages
  - Invalid bindfd pointer
  - Bindfd is not existed
  - Bindfd already unbinded

- Sample code reference
  - Any sample

2.1.8 gm_new_groupfd

- Function prototype
  - void *gm_new_groupfd(void);

- Synopsis
  - Apply a groupfd

- Return

Value	Description
fd	fd of a group

- Error message
  - gm_lib not init, please use gm_init first.

- Sample code reference
  - Any sample

2.1.9 gm_delete_groupfd

- Function prototype
  - int gm_delete_groupfd(void *groupfd);

- Synopsis
  - Delete a group by groupfd

- Argument

Argument	Description	Input/Output
Groupfd	groupfd will be deleted.	Input

- Error messages
  - Invalid groupfd pointer
  - bindfd exist! please check gm_unbind flow!

- Sample code reference
  - Any sample

2.1.10 gm_apply

- Function prototype
  - int gm_apply(void *groupfd);

- Synopsis
  - Apply groupfd

- Argument

Argument	Description	Input/Output
groupfd	fd of a group	Input

- Returns

Value	Description
0	Success
Negative value (< 0)	Failure

- Error messages
  - Invalid groupfd pointer
  - Group is empty, It cannot use gm_apply without gm_bind.
  - "display" and "encode" should be in different groups, please check the gm_new_groupfd() flow.
  - "display" and "audio grab" should be in different groups, please check the gm_new_groupfd() flow.
  - "display" and "audio render" should be in different groups, please check the gm_new_groupfd() flow.
  - "encode" and "audio grab" should be in different groups, please check the gm_new_groupfd() flow.
  - "encode" and "audio render" should be in different groups, please check the gm_new_groupfd() flow.
  - "audio grab" and "audio render" should be in different groups, please check the gm_new_groupfd() flow.

- Sample code reference
  - Any sample

2.1.11 gm_set_attr

- Function prototype
  - int gm_set_attr(void *obj, void *attr);

- Synopsis
  - Set the object attribute of an object

- Arguments

Argument	Description	Input/Output
obj	Pointer of an object	Input
attr	Pointer of an attribute	Input

- Returns

Value	Description
0	Success
Negative value (< 0)	Failure

- Error messages
  - Invalid object pointer
  - Invalid attribute pointer
  - Invalid object type! It may be invalid pointer.
  - “attr_name” cannot be set to “obj_name” object!
  - “attr_para_name”(%d) not assigned

- Sample code reference
  - Any sample

2.1.12 gm_poll

- Function prototype
  - int gm_poll(gm_pollfd_t *poll_fds, int num_fds, int timeout_millisec);

- Synopsis
  - Block to poll a bitstream

- Arguments

Argument	Description	Input/Output
poll_fds	Poll fd sets	Input
num_fds	Number of fd sets	Input
timeout_millisec	Timeout value in millisecond	Input

- Returns

Value	Description
0	Success
-1	Failure
-4	Timeout

- Error messages
  - Invalid poll_fds pointer
  - Invalid value of num_fds

- Sample code reference
  - Please refer to encode_capture_substream.c/encode_scaler_substream.c/ encode_update_notification.c.

2.1.13 gm_send_multi_bitstreams

- Function prototype
  - int gm_send_multi_bitstreams(gm_dec_multi_bitstream_t *multi_bs, int num_bs, int timeout_millisec);
- Synopsis
  - Send multiple bitstreams to playback
- Arguments

Argument	Description	Input/Output
multi_bs	The information of multi-bitstreams Please refer to gm_dec_multi_bitstream_t.	Input/Output
num_bs	The number of bitstreams	Input
timeout_millisec	Timeout value in millisecond	Input

- Returns

Value	Description
0	Success
-1	Failure
-4	Timeout

- Error messages
  - Invalid bindfd pointer
  - Invalid send buf point
  - Invalid value of send buf length
- Sample code reference
  - Please refer to playback_1div_to_4div.c/playback_with_liveview.c.

2.1.14 gm_recv_multi_bitstreams

- Function prototype
  - int gm_recv_multi_bitstreams(gm_enc_multi_bitstream_t *multi_bs, int num_bs);
- Synopsis
  - Get multi encode bitstream and motion vector data

- Arguments

Argument	Description	Input/Output
multi_bs	The information of multi-bitstreams Please refer to gm_enc_multi_bitstream_t.	Input/Output
num_bs	The number of bitstreams	Input

- Returns

Value	Description
0	Success
-1	Failure
-2	Bitstream buffer is too small
-3	Motion vector buffer is too small
-4	Timeout

- Note
  - This function will get the bitstream and related information when it returns successful.

- Error messages
  - Invalid bindfd pointer
  - Invalid enc bitstream bs point

- Sample code reference
  - Please refer to encode_capture_substream.c/encode_scaler_substream.c/ encode_update_notification.c./encode_with_deinterlace.c/encode_with_roi.c/encode_with_watermark_and_vui.c.

2.1.15 gm_register_notify_handler

- Function prototype
  - typedef void (*gm_notify_handler_t)(gm_obj_type_t obj_type, int vch, gm_notify_t notify);
  - int gm_register_notify_handler(gm_notify_t notify, gm_notify_handler_t fn_notify_handler);
- Synopsis
  - Register a callback handler function when receiving under the layer notification

- Arguments of gm_register_notify_handler

Argument	Description	Input/Output
notify	The type of the notification that users want to monitor	Input
fn_notify_handler	Callback function for the notification 0: De-register Others: Register	Input

- Returns by gm_register_notify_handler

Value	Description
Positive value (> 0)	Success
-1	Failure

- Arguments of gm_notify_handler_t

Argument	Description	Input/Output
obj_type	The type of the object that sends the notification	Input
vch	The vch of the object that sends the notification	Input
notify	Type of the notification	Input

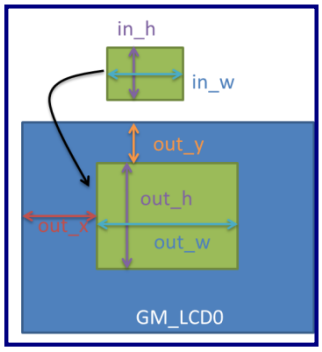
- Note
  - Users can register only some types of the notifications that users want to receive. Different types of the notifications can share the same callback handler and distinguish the type by obj_type.

- Sample code reference
  - Please refer to liveview_with_notification.c.

2.2 Additional Function

2.2.1 Clear Window

Clear window is used to show the pattern on the window of LCD vch (Virtual channel), such as the blank screen or logo. To provide the input pattern and show on the window of LCD vch, users need to handle the output position and size for the multiple divisions.



- Function prototype
  - int gm_clear_window(int lcd_vch, gm_clear_window_t *cw_str)
- Synopsis
  - Trigger clear window to display
- Arguments

Argument	Description	Input/Output
lcd_vch	vch of LCD (GM_LCD0, GM_LCD1, and so on)	Input
cw_str	Clear window command structure	Input

- Returns

Value	Description
0	Success
-1	Failure

- Sample code reference
  - Please refer to liveview_with_clearwin.c.

2.2.2 Capture Motion Data

- Function prototype
  - int gm_recv_multi_cap_md(gm_multi_cap_md_t *multi_cap_md, int num_cap_md);

- Synopsis
  - It is for detecting the moving object whose area was selected from users via parameters.

- Arguments

Argument	Description	Input/Output
multi_cap_md	Start the pointer of the array of cap_md Please refer to gm_multi_cap_md_t.	Input/Output
num_cap_md	Number of the array of cap_md	Input

- Returns

Value	Description
0	Success
-1	Failure
-2	Buffer is too small.

- Error message
  - Invalid cap_md_info point

- Note
  - Please refer to encode_with_capture_motion_detection.c for demonstration.

2.2.3 Capture Flip

- Function prototype
  - int gm_set_cap_flip(int cap_vch, gm_cap_flip_t *cap_flip);

- Synopsis
  - Set the capture flip mode

- Arguments

Argument	Description	Input/Output
cap_vch	The index of the specific capture virtual channel	Input
cap_flip	The flip mode Please refer to gm_cap_flip_t.	Input

- Returns

Value	Description
0	Success
-1	Failure

2.2.4 Display Snapshot

- Function prototype
  - int gm_request_disp_snapshot(disp_snapshot_t *disp_snapshot, int timeout_millisec)
- Synopsis
  - Request a snapshot JPEG from LCD
- Arguments

Argument	Description	Input/Output
disp_snapshot	disp_snapshot_t structure of a snapshot from LCD	Input
timeout_millisec	Timeout value to snapshot a JPEG	Input

- Returns

Value	Description
Positive value (> 0)	Success (Length of JPEG)
-1	Failure
-2	Buffer is too small.
-4	Timeout

- Sample code reference
  - Please refer to display_with_snapshot.c.

2.2.5 Encode Snapshot

- Function prototype
  - int gm_request_snapshot(snapshot_t *snapshot, int timeout_millisec);

- Synopsis
  - Request a snapshot from encoder

- Arguments

Argument	Description	Input/Output
snapshot	The information of snapshots Please refer to snapshot_t.	Input/Output
timeout_millisec	Timeout value to do snapshots	Input

- Returns

Value	Description
Positive value (> 0)	Success (The length of JPEG)
-1	Failure
-2	Buffer is too small.
-4	Timeout

- Sample code reference
  - Please refer to encode_with_snapshot.c.

2.2.6 Force Keyframe

- Function prototype
  - int gm_request_keyframe(void *bindfd);

- Synopsis
  - Encode I-frame as soon as possible, check gm_enc_bitstream_t -> keyframe if get I-frame

- Argument

Argument	Description	Input/Output
bindfd	fd of gm_bind	Input

- Returns

Value	Description
0	Success
Negative value (< 0)	Failure

- Sample code reference
  - Please refer to encode_force_keyframe.c.

2.2.7 H.264 Encode Get Raw Data

- Function prototype
  - int gm_get_rawdata(void *bindfd, region_rawdata_t *region_rawdata, int timeout_millisec)

- Synopsis
  - It is used to get raw data of the specific area of the capture frame.

- Arguments

Argument	Description	Input/Output
bindfd	fd of gm_bind	Input
region_rawdata	Please refer to region_rawdata_t.	Input
timeout_millisec	Timeout value in millisecond	Input

- Returns

Value	Description
Positive value (> 0)	Success
-1	Failure
-2	Buffer is too small
-4	Timeout

- Sample code reference
  - Please refer to encode_with_getraw.c.

2.2.8 H.264 Motion Data

Motion detection is used to automatically identify motions in the video bit stream. It can also identify motions in the specific area of a video.

- See also
  - gm_recv_multi_bitstreams and gm_enc_bitstream_t

- Sample code reference
  - Please refer to encode_with_h264_motion_detection.c.

2.2.9 H.264 Decode Single Keyframe

- Function prototype
  - int gm_decode_keyframe(decode_keyframe_t *dec)

- Synopsis
  - It is used to decode the given H.264 keyframe in the RGB buffer.

- Arguments

Argument	Description	Input/Output
dec	decode_keyframe_t structure for decoding the keyframe Please refer to decode_keyframe_t.	Input

- Returns

Value	Description
0	Success
-1	Failure
-2	Buffer is too small.

- Note
  - It only supports the RGB565 format

- Sample code reference
  - Please refer to decode_keyframe.c.

2.2.10 OSD Font

- Function prototype
  - int gm_set_osd_font(void *obj, gm_osd_font_t *osd_font)

- Synopsis
  - It shows the background area (Window) and foreground area (Font). The font is shown on the screen.

- Arguments

Argument	Description	Input/Output
obj	obj to show OSD	Input
osd_font	Parameters of osd_font Please refer to gm_osd_font_t.	Input

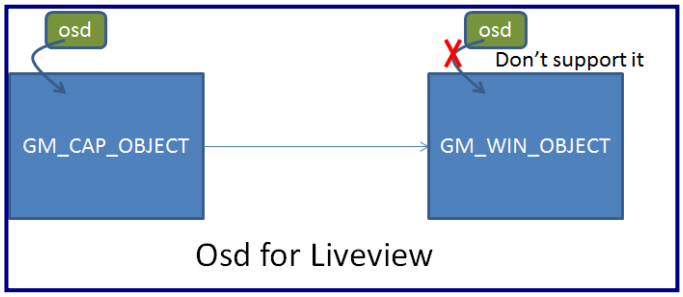
- Returns

Value	Description
0	Success
-1	Failure

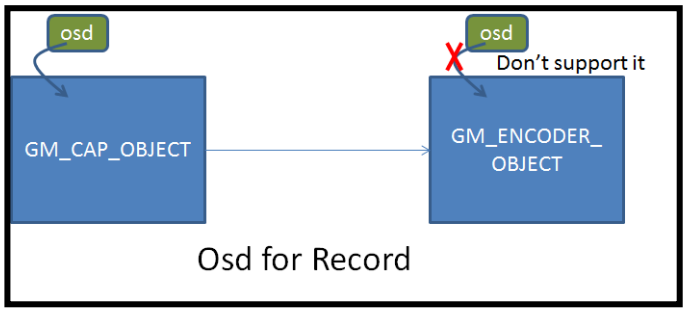
- Error messages
  - Failed to do osd_font.
  - api_osd_win_disable(ret=%d) error!

- api_osd_set_disp_string(ret=%d) error!
- api_osd_set_win_param(ret=%d) error!
- api_osd_set_font_zoom(ret=%d) error!
- api_osd_set_alpha(ret=%d) error!
- api_osd_set_color(ret=%d) error!
- api_osd_set_priority(ret=%d) error!
- api_osd_set_smooth(ret=%d) error!
- api_osd_set_border_disable(ret=%d) error!
- api_osd_set_border_param(ret=%d) error!
- api_osd_set_border_enable(ret=%d) error!
- api_osd_win_enable(ret=%d) error!

- Notes
  - Please refer to encode_with_osd.c for demonstration.
  - OSD for Liveview: It only supports GM_CAP_OBJECT.



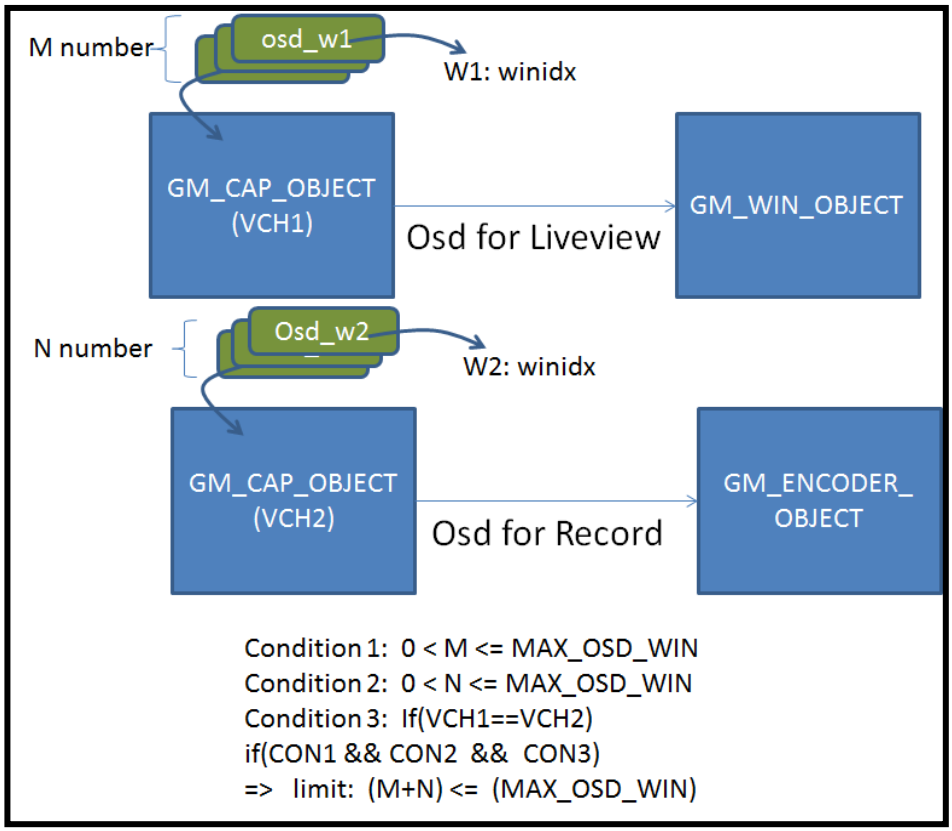
  - OSD for Record: It only supports GM_CAP_OBJECT.



  - OSD for Playback: It is not supported.

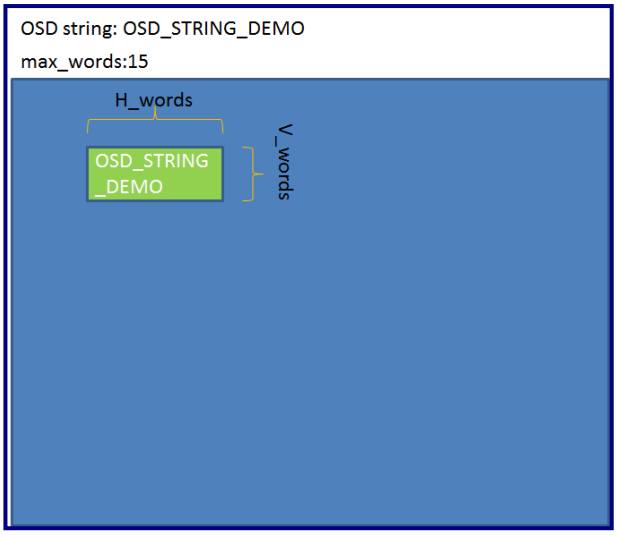
- OSD has limitation of capability on each object by following the rules defined below.

GM_CAP_OBJECT: Osd winidx



Horizontal number of the words for OSD window

- V_words: Vertical number of the words of the OSD window
- Max_words: Maximum number of the words of the OSD window
- ch_h_sp: Vertical space between two characters
- ch_v_sp: Horizontal space between two characters



- See also
  - The OSD mask and OSD mark

2.2.11 OSD Font 2

- Function prototype
  - int gm_set_osd_font2(void *obj, gm_osd_font2_t *osd_font)

- Synopsis
  - It shows the background area (Window) and foreground area (Font). The font is shown on the screen.
  - Font number can be more than GM_MAX_OSD_FONTS (31).

- Arguments

Argument	Description	Input/Output
obj	obj to show OSD	Input
osd_font	Parameters of osd_font Please refer to gm_osd_font2_t.	Input

- Returns

Value	Description
0	Success
-1	Failure

- Error messages
  - Failed to do osd_font.
  - api_osd_win_disable(ret=%d) error!
  - api_osd_set_disp_string(ret=%d) error!
  - api_osd_set_win_param(ret=%d) error!
  - api_osd_set_font_zoom(ret=%d) error!
  - api_osd_set_alpha(ret=%d) error!
  - api_osd_set_color(ret=%d) error!
  - api_osd_set_priority(ret=%d) error!
  - api_osd_set_smooth(ret=%d) error!
  - api_osd_set_border_disable(ret=%d) error!
  - api_osd_set_border_param(ret=%d) error!
  - api_osd_set_border_enable(ret=%d) error!
  - api_osd_win_enable(ret=%d) error!

2.2.12 OSD Mask

- osd_mark: It shows an image on the screen.
  - It shows the hollow or solid mask area with the specific color on the screen.

- Function prototype

- Synopsis
  - int gm_set_osd_mask(void *obj, gm_osd_mask_t *osd_mask, gm_path_mode_t path_mode)

- Arguments

Argument	Description	Input/Output
void *obj	obj to show OSD	Input
osd_mask	Parameters of osd_mask Please refer to gm_osd_mask_t.	Input
path_mode	Specify the osd_mask paste on this path or all paths GM_THIS_PATH/GM_ALL_PATH	Input

- Returns

Value	Description
0	Success
-1	Failure

- Error messages
  - Failed to do osd_mask.
  - api_osd_mask_win_disable(ret=%d) error!
  - api_osd_mask_set_win_param(ret=%d) error!
  - api_osd_mask_set_alpha(ret=%d) error!
  - api_osd_mask_win_enable(ret=%d) error!

- Note
  - Please refer to encode_with_osd.c for demonstration.

- See also
  - The OSD font and OSD mark

2.2.13 OSD Mark

- Function prototype
  - int gm_set_osd_mark(void *obj, gm_osd_mark_t *osd_mark)

- Synopsis
  - It shows an image received from the application on the screen.

- Arguments

Argument	Description	Input/Output
void *obj	obj to show OSD	Input
Osd_mark	Parameters of osd_mark Please refer to gm_osd_mark_t.	Input

- Returns

Value	Description
0	Success
-1	Failure

- Error messages
  - Failed to do osd_mark
  - api_osd_mark_disable(ret=%d) error!
  - api_osd_mark_set_win_param(ret=%d) error!
  - api_osd_mark_enable(ret=%d) error!

- Note
  - Please refer to encode_with_osd.c for demonstration.

- See also
  - The OSD font and OSD mask

2.2.14 OSD Mark Image

- Function prototype
  - int gm_set_osd_mark_image(gm_osd_mark_img_table_t *osd_mark_img);

- Synopsis
  - Pre-load the OSD mark image (The maximum is four images.)

- Argument

Argument	Description	Input/Output
osd_mark_img	Parameters for describing pre-load image Please refer to gm_osd_mark_img_table_t.	Input

- Returns

Value	Description
0	Success
-1	Failure

- Notes
  - The Pre-load images for osd_mark.
  - The total size of these four images has the maximum limit of (2048*8) bytes.

- Sample code reference
  - Please refer to encode_with_osd.c.

2.2.15 Palette Table

- Function prototype
  - int gm_set_palette_table(gm_palette_table_t *palette);

- Synopsis
  - Set the specific color to the palette table with the specific index.

- Arguments

Argument	Description	Input/Output
palette	Array element for storing the dedicated color Please refer to gm_palette_table_t.	Input

- Returns

Value	Description
0	Success
-1	Failure

- Note
  - Only the maximum depth of an array is defined in GM_MAX_OSD_PALETTE_IDX_NUM.

- Sample code reference
  - Please refer to encode_with_osd.c.

2.2.16 Update New Font

- Function prototype
  - int gm_update_new_font(gm_osd_font_update_t *new_font);

- Synopsis
  - Update the new font.

- Argument

Argument	Description	Input/Output
New_font	Please refer to gm_osd_font_update_t.	Input/Output

- Returns

Value	Description
Positive value (> 0)	Success
-1	Failure

- Sample code reference
  - Please refer to encode_with_osd.c.

2.2.17 Display Adjustment

- Function prototype
  - int gm_adjust_disp(int lcd_vch, int x, int y, int width, int height);

- Synopsis
  - Adjust the display window

- Arguments

Argument	Description	Input/Output
Lcd_vch	LCD channel index	Input
x	The x offset of the display frame on the target LCD window (4 bytes alignment)	Input
y	The y offset of the display frame on the target LCD window (2 bytes alignment)	Input
width	Shrink width on the target LCD window (4 bytes alignment)	Input
height	Shrink height on the target LCD window (2 bytes alignment)	Input

- Returns

Value	Description
0	Success
-1	Failure

- Sample code reference
  - Please refer to display_adjust.c.

2.2.18 Set Parameter for Capture Tamper Detection

- Function prototype
  - int gm_set_cap_tamper(int cap_vch, gm_cap_tamper_t *cap_tamper);

- Synopsis
  - Set parameters for the capture tamper detection

- Arguments

Argument	Description	Input/Output
cap_vch	Capture channel index	Input
cap_tamper	Parameters of the capture tamper detection Please refer to gm_cap_tamper_t.	Input

- Returns

Value	Description
0	Success
-1	Failure

- Sample code reference
  - Please refer to encode_with_capture_tamper_detection2.c and tamper_detection_introduction.

2.2.19 Set Parameter for Capture Motion Detection

- Function prototype
  - int gm_set_cap_motion(int cap_vch, gm_cap_motion_t *cap_motion);

- Synopsis
  - Set parameters for the capture motion detection

- Arguments

Argument	Description	Input/Output
cap_vch	Capture the channel index	Input
cap_motion	Parameters of the capture motion detection Please refer to gm_cap_motion_t.	Input

- Returns

Value	Description
0	Success
-1	Failure

- Sample code reference
  - Please refer to encode_with_capture_motion_detection.c and motion_detection_introduction.

2.2.20 Dynamically update attribute for the bindfd

- Function prototype

  int gm_apply_attr(void *bindfd, void *attr);

- Synopsis
  - Dynamically update the attribute for bindfd.

- Arguments

Argument	Description	Input/Output
bindfd	Fd of gm_bind Supports items of the input object of bindfd 1. Capture object 2. File object	Input
attr	Attribute Supports items of the attribute: 1. gm_crop_attr_t	Input

- Returns

Value	Description
0	Success
-1	Failure

- Sample code reference
  - Please refer to liveview_with_crop.c.

2.2.21 Set signal number for notify

- Function prototype
  - int gm_set_notify_sig(int sig);

- Synopsis
  - Set the signal number for notifying

- Arguments

Argument	Description	Input/Output
sig	Notify the signal number	Input

- Returns

Value	Description
Positive value	Success
-1	Failure

2.2.22 Clear bit-stream

- Function prototype
  - int gm_clear_bitstream(void *obj, gm_clear_window_t *cw_str);

- Synopsis
  - Send the specific pattern to be the input frame of playback for the purpose of clearing window when no playback bitstream input happened.

- Arguments

Argument	Description	Input/Output
obj	File object	Input
cw_str	Clear window command structure. Please refer to gm_clear_window_t.	Input

- Returns

Value	Description
0	Success
-1	Failure

Chapter 3

GM_LIB Control Structure

This chapter contains the following sections:

3.1 General Structure

3.1.1 gm_dim_t

- Definition

```
typedef struct gm_dim {
    int width;
    int height;
} gm_dim_t;
```

- Description
  - Dimension definition of resolution

- Members

Member	Description	Value
width	Width of the resolution	-
height	Height of the resolution	-

3.1.2 gm_obj_type_t

- Definition

```
typedef enum {
    GM_CAP_OBJECT = 0xFEFE0001,
    GM_ENCODER_OBJECT = 0xFEFE0002,
    GM_WIN_OBJECT = 0xFEFE0003,
    GM_FILE_OBJECT = 0xFEFE0004,
    GM_AUDIO_GRAB_OBJECT = 0xFEFE0005,
    GM_AUDIO_ENCODER_OBJECT = 0xFEFE0006,
    GM_AUDIO_RENDER_OBJECT = 0xFEFE0007,
} gm_obj_type_t;
```

- Description
  - Object type definition

- List

List	Description
GM_CAP_OBJECT	Video capture object
GM_ENCODER_OBJECT	Video encoder object
GM_WIN_OBJECT	LCD window object
GM_FILE_OBJECT	Video file object
GM_AUDIO_GRAB_OBJECT	Audio grab object
GM_AUDIO_ENCODER_OBJECT	Audio encoder object
GM_AUDIO_RENDER_OBJECT	Audio render object

3.1.3 gm_enc_ratecontrol_mode_t

- Definition

```
typedef enum {
    GM_CBR = 1,
    GM_VBR,
    GM_ECBR,
    GM_EVBR,
} gm_enc_ratecontrol_mode_t;
```

- Description
  - Encoder rate control mode

- Lists

List	Description
GM_CBR	Constant bitrate
GM_VBR	Variable bitrates or constant quality
GM_ECBR	Enhance constant bitrate
GM_EVBR	Enhance variable titrates

3.2 System Structure

3.2.1 gm_video_scan_method_t

- Definition

```
typedef enum {
    GM_INTERLACED,
    GM_PROGRESSIVE,
    GM_RGB888,
    GM_ISP,
} gm_video_scan_method_t;
```

- Description
  - Capture information of the system

- List

List	Description
GM_INTERLACED	Interlaced video
GM_PROGRESSIVE	Progressive video
GM_RGB888	RGB888
GM_ISP	CMOS sensor type

3.2.2 gm_cap_sys_info_t

- Definition

```
typedef struct {
    int valid;
    unsigned int number_of_path;
    gm_video_scan_method_t scan_method;
    int framerate;
    gm_dim_t dim;
    unsigned int chipid;
    int is_present;
    int reserved[8];
} gm_cap_sys_info_t;
```

- Description
  - Capture information of the system

- Members

Member	Description	Value
valid	Whether this item is available. -1: Non-available ≥ 0: Available	Output -1 or ≥ 0
number_of_path	Total available number of capture paths	Output (> 0)
scan_method	Video scan mode	Output Please refer to gm_video_scan_method_t.
framerate	Video maximum frame rate	Output (≥ 0)
dim	Video maximum resolution	Output (≥ 0)
chipid	Specific ID for identifying the cap_vch on which chip	Output (≥ 0)
Is_present	The present status for this cap_vch	Output 1: Presented 0: Not presented

- Note
  - It reports the capture information. The items are available when valid ≥ 0.

3.2.3 gm_lcd_sys_info_t

- Definition

```
typedef struct {
    int valid;
    int framerate;
    gm_dim_t dim;
    int duplicate;
    int reserved[10];
} gm_lcd_sys_info_t;
```

- Description
  - LCD information of the system

- Members

Member	Description	Value
valid	Whether this item is available -1: Non-available ≥ 0: Available	Output -1 or ≥ 0
framerate	LCD frame rate	Output (> 0)
dim	LCD resolution	Output (> 0)
duplicate	Duplicate = -1 (Not duplicate) Duplicate = vch (Duplicate of vch)	Output -1 or ≥ 0

- Notes
  - It reports the LCD information, and the items are available when "valid" ≥ 0.
  - When "duplicate" ≥ 0, "duplicate" means duplicating LCD. For example, if duplicate = '0', this LCD is the same with the LCD 0 video, and users should not do any functional setting for it.

3.2.4 gm_audio_channel_type_t

- Definition

```
typedef enum {
    GM_MONO = 1,
    GM_STEREO,
} gm_audio_channel_type_t;
```

- Description
  - This defines how many channels for this audio device.

- Lists

List	Description
GM_MONO	One channel
GM_STEREO	Two channels

3.2.5 gm_audio_grab_sys_info_t

- Definition

```
typedef struct {
    int valid;
    gm_audio_channel_type_t channel_type;
    int sample_rate;
    int sample_size;
    int ssp;
    int reserved[5];
} gm_audio_grab_sys_info_t;
```

- Description
  - Audio grab information of the system

- Members

Member	Description	Value
valid	Whether this item is available -1: Non-available ≥ 0: Available	Output -1 or ≥ 0
channel_type	Count for the audio channels defined by gm_audio_channel_type_t	Output Please refer to gm_audio_channel_type_t.
sample_rate	Audio sampling rate 8000/16000/32000/44100…	Output
sample_size	Audio sampling size 8bits/16bits...	Output
ssp	Hardware SSP ID from 0, 1, and so on	Output

- Note
  - It reports the audio information. The items are available, when the "valid" ≥ 0

3.2.6 gm_audio_render_sys_info_t

- Definition

```
typedef struct {
    int valid;
    gm_audio_channel_type_t channel_type;
    int sample_rate;
    int sample_size;
    int ssp;
    int reserved[5];
} gm_audio_render_sys_info_t;
```

- Description
  - Audio render information of the system

- Members

Members	Description	Value
Valid	Whether this item is available. -1: Non-available ≥ 0: Available	Output -1 or ≥ 0
channel_type	Count of audio channels defined by gm_audio_channel_type_t	Output Please refer to gm_audio_channel_type_t.
sample_rate	Audio sampling rate 8000/16000/32000/44100	Output
sample_size	Audio sampling size Bit number per sample: 8bits/16bits…	Output
ssp	Hardware SSP ID from 0, 1, and so on	Output

- Note
  - It reports the audio render information. The items are available, when "valid" ≥ 0

3.2.7 gm_system_t

- Definition

```
#define CAPTURE_VCH_NUMBER          128
#define LCD_VCH_NUMBER              6
#define AUDIO_GRAB_VCH_NUMBER       32
#define AUDIO_RENDER_VCH_NUMBER     32
typedef struct {
    unsigned int graph_type;
    unsigned int gm_lib_version;
    gm_cap_sys_info_t cap[CAPTURE_VCH_NUMBER];
    gm_lcd_sys_info_t lcd[LCD_VCH_NUMBER];
    gm_audio_grab_sys_info_t au_grab[AUDIO_GRAB_VCH_NUMBER];
    gm_audio_render_sys_info_t au_render[AUDIO_RENDER_VCH_NUMBER];
    int reserved[5];
} gm_system_t;
```

- Description
  - System information

- Members

Member	Description	Value
graph_type	Graph type	Output unsigned int
gm_lib_version	GM library version	Output unsigned int
cap	Capture information Please refer to gm_cap_sys_info_t.	Output structure
lcd	LCD information Please refer to gm_lcd_sys_info_t.	Output structure
au_grab	Audio grab information Please refer to gm_audio_grab_sys_info_t.	Output structure
au_render	Audio render information Please refer to gm_audio_render_sys_info_t.	Output structure

- Note
  - Array index of cap/lcd/au_grab/au_render is defined as vch (Virtual channel). vch is the number of the device virtual channels corresponding to a real entity in the system. For example, if users want to know the information about LCD vch = 1 resolution, users should check the following example:

```
gm_system_t gmSystem;
gm_init();
gm_get_sysinfo(&gmSystem);
if (gmSystem.lcd[1].valid >= 0 ) {
  printf("LCD1 resolution width=%d, height=%d", gmSystem.lcd[1].dim.width,
gmSystem.lcd[1].dim.height);
} else {
  printf("LCD_1 is non-available");
}
```

3.2.8 gm_pollfd_t

- Definition

```
#define GM_POLL_READ            0x1
#define GM_POLL_WRITE           0x2
#define GM_FD_MAX_PRIVATE_DATA  4

typedef struct {
    unsigned int event;
    unsigned int bs_len;
    unsigned int mv_len;
    unsigned int keyframe;
} gm_ret_events_t;

typedef struct {
    void *bindfd;
    unsigned int event;
    gm_ret_event_t revent;
    int fd_private[GM_FD_MAX_PRIVATE_DATA];
} gm_pollfd_t;
```

- Description
  - Poll event data structure

- Members

Member	Description	Value
bindfd	Return value of gm_bind	Input void point
event	GM_POLL_READ: Indicate an encode event GM_POLL_WRITE: Indicate a decode event	Input GM_POLL_READ or GM_POLL_WRITE

Member	Description	Value
revent.event	Return available events GM_POLL_READ: Encode event is available. GM_POLL_WRITE: Decode event is available.	Output GM_POLL_READ or GM_POLL_WRITE
revent.bs_len	Return the length of a bitstream of an available event	Output (> 0)
revent.mv_len	Reserved, do not use it.	Reserved
Revent.keyframe	Return the keyframe information 1: Keyframe 0: Non-keyframe	Output
fd_private	Library internal data, do not use it	None

3.2.9 gm_enc_multi_bitstream_t

- Definition

```
#define GM_FLAG_NEW_FRAME_RATE  (1 << 3)
#define GM_FLAG_NEW_GOP         (1 << 4)
#define GM_FLAG_NEW_DIM         (1 << 6)
#define GM_FLAG_NEW_BITRATE     (1 << 7)
#define GM_MAX_ROI_QP_NUMBER     8

typedef struct gm_enc_bitstream {
    /* provide by application */
    char            *bs_buf;
    unsigned int    bs_buf_len;
    char            *mv_buf;
    unsigned int    mv_buf_len;
    /* value return by gm_recv_bitstream() */
    unsigned int    bs_len;
    unsigned int    mv_len;
    int             keyframe;
    unsigned int    timestamp;
    unsigned int    newbs_flag;
    unsigned int    checksum;
    int             ref_frame;
    unsigned int    slice_offset[3];
    int reserved[5];
} gm_enc_bitstream_t;

typedef struct gm_enc_multi_bitstream {
    void *bindfd;
    gm_enc_bitstream_t bs;
    int retval;  ///< less than 0: recv bistream fail.
    int reserved[6];
#define GM_ENC_MAX_PRIVATE_DATA         28
    int enc_private[GM_ENC_MAX_PRIVATE_DATA]; ///< Library internal data, don't use it!
} gm_enc_multi_bitstream_t;
```

- Description
  - Encoder multi-bitstreams

- Members

Member	Description	Value
bindfd	fd of gm_bind	Input
retval	Returned status 0: Successful < 0: Failed	Output
bs.bs_buf	The pointer of the bitstream buffer for one channel	Input (NULL or character point)
bs.bs_buf_len	The maximum bitstream length for one channel	Input (> 0)
bs.mv_buf	Reserved, do not use it.	Reserved
bs.mv_buf_len	Reserved, do not use it.	Reserved
bs.bs_len	The real bitstream length from this bindfd	Output
bs.mv_len	Reserved, do not use it.	Reserved
bs.keyframe	Indicate keyframe or not 1: This bitstream is keyframe. 0: This bitstream is non-keyframe.	Output (1 or 0)
bs.timestamp	Encode the bitstream timestamp	Output
bs.newbs_flag	Flag notification of the new setting GM_FLAG_NEW_FRAME_RATE: Indicate the bitstream of the new frame rate setting GM_FLAG_NEW_GOP: Indicate the bitstream of the new GOP setting GM_FLAG_NEW_DIM: Indicate the bitstream of the new resolution setting GM_FLAG_NEW_BITRATE: Indicate the bitstream of the new bitrate setting	Output
bs.checksum	Checksum value	
bs.ref_frame	1: Reference frame, cannot skip 0: Not reference frame, can skip	Output
bs.slice_offset[]	Multi-slice offsets 1 ~ 3 (First offset is 0)	Output

- Note
  - newbs_flag: New bitstream reports. It tells users which kind of the new bitstream is coming in when the gm_apply function is applied.

Newbs_flag	Description
GM_FLAG_NEW_FRAME_RATE	New frame rate setting applied
GM_FLAG_NEW_GOP	New GOP setting applied
GM_FLAG_NEW_DIM	New frame dimension
GM_FLAG_NEW_BITRATE	New bit rate setting applied

Example:

```
if ((ret = gm_recv_multi_bitstreams(multi_bs, MAX_BITSTREAM_NUM)) < 0) {
    printf("Error return value %d\n", ret);
} else {
    for (i = 0; i < MAX_BITSTREAM_NUM; i++) {
        if ((multi_bs[i].retval < 0) && multi_bs[i].bindfd) {
            printf("CH%d Error to receive bitstream. \n", i);
        } else if (multi_bs[i].retval == GM_SUCCESS) {
            if (multi_bs[i].bs.newbs_flag & GM_FLAG_NEW_BITRATE) {
                printf("<CH%d, newbsflag=0x%x detect bitrate change>\n", i,
                    multi_bs[i].bs.newbs_flag);
            }
            if (multi_bs[i].bs.newbs_flag & GM_FLAG_NEW_FRAME_RATE) {
                printf("<CH%d, newbsflag=0x%x detect framerate change>\n", i,
                    multi_bs[i].bs.newbs_flag);
            }
            if (multi_bs[i].bs.newbs_flag & GM_FLAG_NEW_GOP) {
                printf("<CH%d, newbsflag=0x%x detect GOP change>\n", i,
                    multi_bs[i].bs.newbs_flag);
            }
        }
    }
}
```

  - Timestamp: Applications should take the time for recording or RTSP streaming. The unit is milliseconds.

3.2.10 gm_multi_cap_md_t

- Definition

```
#define CAP_MOTION_SIZE     4096

typedef struct {
    /* provide md_buf by application, the max buf size is 4096*/
    char *md_buf;
    int  md_buf_len;
    /* value return by gm_recv_multi_cap_md() */
    int is_valid;
    int md_len;
    gm_dim_t md_dim;
    gm_dim_t md_mb;
} gm_cap_md_info_t;

typedef struct gm_multi_cap_md {
    void *bindfd;
    gm_cap_md_info_t cap_md_info;
    int retval;
#define GM_MD_MAX_PRIVATE_DATA         16
    int cap_md_private[GM_MD_MAX_PRIVATE_DATA];
} gm_multi_cap_md_t;
```

- Description
  - Capture motion detection data structure

- Members

Member	Description	Value
bindfd	fd of bind	Input
retval	The status of cap_md on this bindfd 0: Successful < 0: Failed	Output
cap_md_private [GM_MD_MAX_PRIVATE_DATA]	Library internal data and do not use it.	Don't care
cap_md_info.md_buf	Buffer for retrieving the motion detection data The size is CAP_MOTION_SIZE (4096).	Input character point
cap_md_info.md_buf_len	Specify the md_buf size The value should be CAP_MOTION_SIZE (4096).	Input (> 0)
cap_md_info.is_valid	Return this motion data is available or not 1: Valid 0: Invalid (It means that cap_md has not been ready on this bindfd.)	Output
cap_md_info.md_len	The real size of the motion data	Output

Member	Description	Value
cap_md_info.md_dim	The region dimension for algorithm	Output
cap_md_info.md_mb	The macro block dimension for algorithm	Output

3.2.11 gm_osd_align_type_t

- Definition

```
typedef enum {
    GM_OSD_ALIGN_TOP_LEFT = 0,
    GM_OSD_ALIGN_TOP_CENTER,
    GM_OSD_ALIGN_TOP_RIGHT,
    GM_OSD_ALIGN_BOTTOM_LEFT,
    GM_OSD_ALIGN_BOTTOM_CENTER,
    GM_OSD_ALIGN_BOTTOM_RIGHT,
    GM_OSD_ALIGN_CENTER,
    GM_OSD_ALIGN_NONE
} gm_osd_align_type_t;
```

- Description
  - OSD align type

- Lists

List	Description
GM_OSD_ALIGN_TOP_LEFT	Define the origin of coordinate
GM_OSD_ALIGN_TOP_CENTER	Define the origin of coordinate
GM_OSD_ALIGN_TOP_RIGHT	Define the origin of coordinate
GM_OSD_ALIGN_BOTTOM_LEFT	Define the origin of coordinate
GM_OSD_ALIGN_BOTTOM_CENTER	Define the origin of coordinate
GM_OSD_ALIGN_BOTTOM_RIGHT	Define the origin of coordinate
GM_OSD_ALIGN_CENTER	Define the origin of coordinate
GM_OSD_ALIGN_NONE	This mode is only for OSD_mask on liveview with crop enabled.

3.2.12 gm_osd_priority_t

- Definition

```
typedef enum {
    GM_OSD_PRIORITY_MARK_ON_OSD = 0,
    GM_OSD_PRIORITY_OSD_ON_MARK
} gm_osd_priority_t;
```

- Description
  - OSD priority data structure

- List

List	Description
GM_OSD_PRIORITY_MARK_ON_OSD	Mark above OSD window
GM_OSD_PRIORITY_OSD_ON_MARK	Mark below OSD window

3.2.13 gm_osd_smooth_t

- Definition

```
typedef enum {
    GM_OSD_FONT_SMOOTH_LEVEL_WEAK = 0,
    GM_OSD_FONT_SMOOTH_LEVEL_STRONG
} gm_osd_font_smooth_level_t;
typedef struct {
    int                          enabled;
    gm_osd_font_smooth_level_t   level;
    int reserved[5];
} gm_osd_smooth_t;
```

- Description
  - OSD font smooth level data structure

- List

List	Description
enabled	1: Enable 0: Disable
Level: GM_OSD_FONT_SMOOTH_LEVEL_WEAK	Font smooth with the weak level
Level: GM_OSD_FONT_SMOOTH_LEVEL_STRONG	Font smooth with the strong level

3.2.14 gm_osd_border_type_t

- Definition

```
typedef enum {
    GM_OSD_BORDER_TYPE_WIN = 0,
    GM_OSD_BORDER_TYPE_FONT
} gm_osd_border_type_t;
```

- Description
  - Decide the border type for osd_font.

- List

List	Description
GM_OSD_BORDER_TYPE_WIN	Treat the alpha of border as the window
GM_OSD_BORDER_TYPE_FONT	Treat the alpha of border as the font

3.2.15 gm_osd_border_param_t

- Definition

```
typedef struct {
    int enable;
    int width;
    gm_osd_border_type_t type;
    int palette_idx;
    int reserved[5];
} gm_osd_border_param_t;
```

- Description
  - OSD font border data structure

- Members

Member	Description	Value
enable	1: Enable the OSD font border function 0: Disable the OSD font border function	Input (0 or 1)
width	Border width of the OSD window	Input (0 ~ 7) Border width is 4 x (n + 1) pixels.
Type	OSD window border type Please refer to gm_osd_border_type_t.	Input
palette_idx	Border color of the OSD window	Input (0 ~ 15)

3.2.16 gm_osd_font_zoom_t

- Definition

```
typedef enum {
    GM_OSD_FONT_ZOOM_NONE = 0,
    GM_OSD_FONT_ZOOM_2X,
    GM_OSD_FONT_ZOOM_3X,
    GM_OSD_FONT_ZOOM_4X,
    GM_OSD_FONT_ZOOM_ONE_HALF,
    GM_OSD_FONT_ZOOM_H2X_V1X = 8,
    GM_OSD_FONT_ZOOM_H4X_V1X,
    GM_OSD_FONT_ZOOM_H4X_V2X,
    GM_OSD_FONT_ZOOM_H1X_V2X = 12,
    GM_OSD_FONT_ZOOM_H1X_V4X,
    GM_OSD_FONT_ZOOM_H2X_V4X,
} gm_osd_font_zoom_t;
```

- Description
  - Select the type of the font zoom

- Lists

List	Description
GM_OSD_FONT_ZOOM_NONE	No font zoom
GM_OSD_FONT_ZOOM_2X	Horizontalx2 and verticalx2
GM_OSD_FONT_ZOOM_3X	Horizontalx3 and verticalx3
GM_OSD_FONT_ZOOM_4X	Horizontalx4 and verticalx4
GM_OSD_FONT_ZOOM_ONE_HALF	Horizontal/2 and vertical/2
GM_OSD_FONT_ZOOM_H2X_V1X	Horizontalx2 and verticalx1
GM_OSD_FONT_ZOOM_H4X_V1X	Horizontalx4 and verticalx1
GM_OSD_FONT_ZOOM_H4X_V2X	Horizontalx4 and verticalx2
GM_OSD_FONT_ZOOM_H1X_V2X	Horizontalx1 and verticalx2
GM_OSD_FONT_ZOOM_H1X_V4X	Horizontalx1 and verticalx4
GM_OSD_FONT_ZOOM_H2X_V4X	Horizontalx2 and verticalx4

3.2.17 gm_osd_font_t

- Definition

```
typedef struct {
    int                     win_idx;
    int                     enabled;
    gm_osd_align_type_t     align_type;
    unsigned int            x;
    unsigned int            y;
    unsigned int            h_words;
    unsigned int            v_words;
    unsigned int            h_space;
    unsigned int            v_space;
    int                     font_index_len;
    unsigned short          font_index[GM_MAX_OSD_FONTS];
    gm_osd_font_alpha_t     font_alpha;
    gm_osd_font_alpha_t     win_alpha;
    int                     font_palette_idx;
    int                     win_palette_idx;
    gm_osd_priority_t       priority;
    gm_osd_smooth_t         smooth;
    gm_osd_marquee_param_t  marquee;
    gm_osd_border_param_t   border;
    gm_osd_font_zoom_t      font_zoom;
    int reserved[5];
} gm_osd_font_t;
```

- Description
  - OSD font data structure

- Members

Member	Description	Value
winidx	Index of the OSD window The paste order could be specified by win_idx. win_idx: 0 (Upper layer), 1, 2,...,7 (Lower layer)	Input (0 ~ 7)
enabled	1: Enable the OSD font function 0: Disable the OSD font function	Input (1 or 0)
align_type	Please refer to gm_osd_align_type_t.	Input
X	x coordinate of the OSD window	Input
Y	y coordinate of the OSD window	Input
h_words	Horizontal word number of the OSD window	Input
v_words	Vertical word number of the OSD window	Input
h_space	Vertical space between two characters	Input
v_space	Horizontal space between two characters	Input

Member	Description	Value
font_index_len	Maximum number of words from font_index to be shown on the OSD window	Input
font_index	The table for storing the index of the OSD font	Input
font_alpha	The alpha setting on the foreground 0: Alpha 0% 1: Alpha 25% 2: Alpha 37.5% 3: Alpha 50% 4: Alpha 62.5% 5: Alpha 75% 6: Alpha 87.5% 7: Alpha 100%	Input (0 ~ 7)
win_alpha	Alpha setting on the background The value definition is the same as font_alpha.	Input (0 ~ 7)
font_palette_idx	OSD font color	Input (0 ~ 15)
win_palette_idx	OSD window color	Input (0 ~ 15)
priority	Priority of OSD and the mark layer Please refer to gm_osd_priority_t.	Input (0 ~ 1)
smooth	Please refer to gm_osd_smooth_t.	Input
marquee	Reserved Please do not use it.	Input
border	Please refer to gm_osd_border_param_t.	Input
font_zoom	Decide the font zoom type Please refer to gm_osd_font_zoom_t.	Input (0 ~ 10)

3.2.18 gm_osd_font2_t

- Definition

```
typedef struct {
    int                     win_idx;
    int                     enabled;
    gm_osd_align_type_t     align_type;
    unsigned int            x;
    unsigned int            y;
    unsigned int            h_words;
    unsigned int            v_words;
    unsigned int            h_space;
    unsigned int            v_space;
    int                     font_index_len;
    unsigned short          *font_index;
    gm_osd_font_alpha_t     font_alpha;
    gm_osd_font_alpha_t     win_alpha;
```

```
    int                     font_palette_idx;
    int                     win_palette_idx;
    gm_osd_priority_t       priority;
    gm_osd_smooth_t         smooth;
    gm_osd_marquee_param_t  marquee;
    gm_osd_border_param_t   border;
    gm_osd_font_zoom_t      font_zoom;
    int reserved[5];
} gm_osd_font2_t;
```

- Description
  - OSD font2 data structure for font number of more than GM_MAX_OSD_FONTS (31)

- Members

Member	Description	Value
winidx	Index of the OSD window The paste order could be specified by win_idx. win_idx: 0 (Upper layer), 1, 2,...,7 (Lower layer)	Input (0 ~ 7)
enabled	1: Enable the OSD font function 0: Disable the OSD font function	Input (1 or 0)
align_type	Please refer to gm_osd_align_type_t.	Input
X	x coordinate of the OSD window	Input
Y	y coordinate of the OSD window	Input
h_words	Horizontal word number of the OSD window	Input
v_words	Vertical word number of the OSD window	Input
h_space	Vertical space between two characters	Input
v_space	Horizontal space between two characters	Input
font_index_len	Maximum number of the words from font_index to be shown on the OSD window	Input
*font_index	The pointer addresses which buffer size is allocated by users for storing the index of the OSD font.	Input
font_alpha	The alpha setting on the foreground 0: Alpha 0% 1: Alpha 25% 2: Alpha 37.5% 3: Alpha 50% 4: Alpha 62.5% 5: Alpha 75% 6: Alpha 87.5% 7: Alpha 100%	Input (0 ~ 7)
win_alpha	Alpha setting on the background The value definition is the same as font_alpha.	Input (0 ~ 7)

Member	Description	Value
font_palette_idx	OSD font color	Input (0 ~ 15)
win_palette_idx	OSD window color	Input (0 ~ 15)
priority	Priority of OSD and the mark layer Please refer to gm_osd_priority_t.	Input (0 ~ 1)
smooth	Please refer to gm_osd_smooth_t.	Input
marquee	Reserved Please do not use it.	Input
border	Please refer to gm_osd_border_param_t.	Input
font_zoom	Decide the font zoom type Please refer to gm_osd_font_zoom_t.	Input (0 ~ 10)

3.2.19 gm_osd_mask_alpha_t

- Definition

```
typedef enum {
    GM_OSD_MASK_ALPHA_0 = 0,
    GM_OSD_MASK_ALPHA_25,
    GM_OSD_MASK_ALPHA_37_5,
    GM_OSD_MASK_ALPHA_50,
    GM_OSD_MASK_ALPHA_62_5,
    GM_OSD_MASK_ALPHA_75,
    GM_OSD_MASK_ALPHA_87_5,
    GM_OSD_MASK_ALPHA_100
  } gm_osd_mask_alpha_t;
```

- Description
  - OSD mask alpha

- List

List	Description
GM_OSD_MASK_ALPHA_0	Alpha 0%
GM_OSD_MASK_ALPHA_25	Alpha 25%
GM_OSD_MASK_ALPHA_37_5	Alpha 37.5%
GM_OSD_MASK_ALPHA_50	Alpha 50%
GM_OSD_MASK_ALPHA_62_5	Alpha 62.5%
GM_OSD_MASK_ALPHA_75	Alpha 75%
GM_OSD_MASK_ALPHA_87_5	Alpha 87.5%
GM_OSD_MASK_ALPHA_100	Alpha 100%

3.2.20 gm_mask_border_type_t

- Definition

```
typedef enum {
    GM_OSD_MASK_BORDER_TYPE_HOLLOW = 0,
    GM_OSD_MASK_BORDER_TYPE_TRUE
} gm_osd_mask_border_type_t;
```

- Description
  - OSD masks the border type

- List

List	Description
GM_OSD_MASK_BORDER_TYPE_HOLLOW	Hollow border type
GM_OSD_MASK_BORDER_TYPE_TRUE	Solid border type

3.2.21 gm_osd_mask_border_t

- Definition

```
typedef struct {
    int width;
    gm_osd_mask_border_type_t type;
    int reserved[5];
} gm_osd_mask_border_t;
```

- Description
  - OSD mask border

- Members

Member	Description	Value
width	Border width of the mask window when hollow on the border width is 2 x (n + 1) pixels	Input (0 ~ 15)
type	Hollow/True of the mask window Please refer to gm_osd_mask_border_type_t.	Input

3.2.22 gm_osd_mask_t

- Definition

```
typedef struct {
    int mask_idx;
    int enabled;
    int x;
    int y;
    int width;
    int height;
    gm_osd_mask_alpha_t alpha;
    int palette_idx;
    gm_osd_mask_border_t border;
    gm_osd_align_type_t align_type;
    int reserved[5];
} gm_osd_mask_t;
```

- Description
  - OSD mask

- Members

Member	Description	Value
mask_idx	Index of the mark window GM_ALL_PATH: Apply in all capture paths, implemented by the capture mask engine mask_idx = 0 (Lower layer), 1, 2, ..., 7 (Upper layer) GM_THIS_PATH: Apply in the specified capture path, implemented by the capture OSD engine mask_idx occupies one of gm_osd_font_t->win_idx from 0 ~ 7. mask_idx = 0 (Upper layer), 1, 2, ..., 7 (Lower layer)	Input (0 ~ 7)
enabled	1: Enable the OSD mask function 0: Disable the OSD mask function	Input (1 or 0)
x	x coordinate of the OSD window	Input
y	y coordinate of the OSD window	Input
width	Width of the mask window	Input
height	Height of the mask window	Input
alpha	Please refer to gm_mask_alpha_t.	Input
palette_idx	The index for selecting color from the palette table path_mode==GM_THIS_PATH (Input: 0 ~ 3) path_mode==GM_ALL_PATH (Input: 0 ~ 15)	Input
border	Please refer to gm_mask_border_t.	Input
align_type	Please refer to gm_osd_align_type_t.	Input

3.2.23 gm_osd_mark_t

- Definition

```
typedef struct {
    int mark_idx;
    int enabled;
    int x;
    int y;
    gm_osd_mark_alpha_t alpha;
    gm_osd_mark_zoom_t  zoom;
    gm_osd_align_type_t align_type;
    unsigned short osg_mark_idx;
    char reserved[18];
} gm_osd_mark_t;
```

- Description
  - OSD mark

- Members

Member	Description	Value
mark_idx	Index of the mark window mark_idx ≥ 4 is for the osg mode. mark_idx = 0 (Upper layer), 1, 2, and 3 (Lower layer)	Input (0 ~ 3) Input (4 ~ 67) for OSG
enabled	1: Enable the OSD mark function 0: Disable the OSD mark function	Input (1 or 0)
x	x coordinate of the OSD window	Input
y	y coordinate of the OSD window	Input
alpha	0: Alpha 0% 1: Alpha 25% 2: Alpha 37.5% 3: Alpha 50% 4: Alpha 62.5% 5: Alpha 75% 6: Alpha 87.5% 7: Alpha 100%	Input (0 ~ 7)
zoom	0: Zoom x1 1: Zoom x2 2: Zoom x4	Input (0 ~ 2)
align_type	Please refer to gm_osd_align_type_t.	Input (0 ~ 6)
osg_mark_idx	The index is the ID of the specific mark image of osg.	Input (4 ~ 67)

3.2.24 gm_palette_table_t

- Definition

```
#define GM_MAX_PALETTE_IDX_NUM 16
typedef struct {
    int palette_table[GM_MAX_PALETTE_IDX_NUM];
} gm_palette_table_t;
```

- Description
  - The table is for palette setting.

- Members

Member	Description	Value
palette_table	The table for storing colors The maximum depth is GM_MAX_OSD_PALETTE_IDX_NUM.	Input (0 ~ 15)

3.2.25 gm_osd_mark_dim_t

- Definition

```
typedef enum {
    GM_OSD_MARK_DIM_16 = 0,
    GM_OSD_MARK_DIM_32,
    GM_OSD_MARK_DIM_64,
    GM_OSD_MARK_DIM_128,
    GM_OSD_MARK_DIM_256,
    GM_OSD_MARK_DIM_512,
    GM_OSD_MARK_DIM_MAX,
} gm_osd_mark_dim_t;
```

- Description
  - The dimension definition for mark image

- List

List	Description
GM_OSD_MARK_DIM_16	The pixel number is 16.
GM_OSD_MARK_DIM_32	The pixel number is 32.
GM_OSD_MARK_DIM_64	The pixel number is 64.
GM_OSD_MARK_DIM_128	The pixel number is 128.
GM_OSD_MARK_DIM_256	The pixel number is 256.
GM_OSD_MARK_DIM_512	The pixel number is 512.
GM_OSD_MARK_DIM_MAX	Reserved Do not use it.

3.2.26 gm_osd_img_param_t

- Definition

```
#define GM_MAX_OSD_MARK_IMG_NUM 4
typedef struct {
    int mark_exist;

    char *mark_yuv_buf;
    unsigned int mark_yuv_buf_len;

    gm_osd_mark_dim_t mark_width;
    gm_osd_mark_dim_t mark_height;
    int osg_tp_color;
    unsigned short osg_mark_idx;
    char reserved[14];
} gm_osd_img_param_t;

typedef struct {
    gm_osd_img_param_t mark_img[GM_MAX_OSD_MARK_IMG_NUM];
    int reserved[5];
} gm_osd_mark_img_table_t;
```

- Description
  - Structure for describing the pre-load image.

- Members

Member	Description	Value
mark_img[]	There are four pre-load images.	Input
mark_img[].mark_exist	The image is existed or not. 1: Exist 0: Non-exist	Input (0 or 1)
mark_img[].mark_yuv_buf	The pointer for storing image contents with the YUV format	Input
mark_img[].mark_yuv_buf_len	The image size	Input (> 0)
mark_img[].mark_width	Width of an image Please refer to gm_osd_mark_dim_t. OSG mode: mark_img[0].mark_width > 5 to select the osg mode (Be careful that the array index must be 0.) The max. size is capture output width. The width must be multiple of 2.	Input (0 ~ 5) Input (6 ~ capture output width) for OSG

Member	Description	Value
mark_img[].mark_height	Height of an image Please refer to gm_osd_mark_dim_t. OSG mode: mark_img[0].mark_height > 5 to select osg mode (Be careful that the array index must be 0) The max. size is capture output height.	Input (0 ~ 5) Input (6 ~ capture output height) for OSG
mark_img[].osg_tp_color	Reserved Do not use it.	-
mark_img[].osg_mark_idx	mark_img[0].mark_idx for the osg mode (Be careful that the array index must be 0.)	Input (4 ~ 64*(channel number)*4)

3.2.27 gm_dec_multi_bitstream_t

- Definition

```
typedef struct gm_dec_multi_bitstream {
    void *bindfd;
    char *bs_buf;
    unsigned int bs_buf_len;
    int retval;
    time_algin_t time_align;
    int reserved[5];
#define GM_DEC_MAX_PRIVATE_DATA         22
    int dec_private[GM_DEC_MAX_PRIVATE_DATA];
} gm_dec_multi_bitstream_t;
```

- Description
  - The information of multi-bitstreams for playback

- Members

Member	Description	Value
bindfd	fd of gm_bind	Input void point
bs_buf	The pointer of the bitstream buffer for playback	Input character point
bs_buf_len	Length of the bitstream buffer	Input
retval	Returned status 0: Successful < 0: Failed	Output

Member	Description	Value
Time_align	time_align (μs): Playback interval time by micro-second TIME_ALIGN_ENABLE (Default): Playback time is aligned by the LCD period (ex. 60HZ is 33333us) TIME_ALIGN_DISABLE: Play timestamp by gm_send_multi_bitstreams called N (Others): Start to play at previous play point + N(μs)	Input
dec_private[GM_DEC_MAX_PRIVATE_DATA]	Library internal data, do not use it.	None

- Note
  - The information of multi-bitstream for playback

3.2.28 gm_cap_flip_t

- Definition

```
typedef struct  {
    int h_flip_enabled;
    int v_flip_enabled;
    int reserved[5];
} gm_cap_flip_t;
```

- Description
  - The setting for capture flip

- Members

Member	Description	Value
h_flip_enable	Flip with the horizontal direction 1: On 0: Off	Input (0 or 1)
v_flip_enable	Flip with the vertical direction 1: On 0: Off	Input (0 or 1)

3.2.29 gm_osd_font_update_t

- Definition

```
#define GM_OSD_FONT_MAX_ROW 18
typedef struct {
    int   font_idx;
    unsigned short  bitmap[GM_OSD_FONT_MAX_ROW];
    int reserved[5];
} gm_osd_font_update_t;
```

- Description
  - The parameters for updating the new font.

- Members

Member	Description	Value
font_idx	Specify the font index for updating	Input (0 ~ 65535)
bitmap	The bitmap for storing the font format with (12 bits (MSB) + 4bits reserved (LSB))x18 bits, that is called the 12x18 font format. Bit 0 means that it does not need to fill the foreground color or font color. Bit 1 means that it fills the foreground color or font color.	Input

3.2.30 gm_clear_window_t

- Definition

```
typedef enum {
    GM_ACTIVE_BY_APPLY,
    GM_ACTIVE_IMMEDIATELY,
} gm_clear_window_mode_t;

typedef enum {
    GM_FMT_YUV422,
} gm_clear_window_yuv_t;

typedef struct {
    int in_w;
    int in_h;
    gm_clear_window_yuv_t in_fmt;
    unsigned char *buf;
    int out_x;
    int out_y;
    int out_w;
    int out_h;
    gm_clear_window_mode_t  mode;
    int reserved[5];
} gm_clear_window_t;
```

- Description
  - Trigger the clear window to display the data structure

- Members

Member	Description	Value
in_w	Input buffer width	Input (≥ 0)
in_h	Input buffer height	Input (≥ 0)
in_fmt	Pattern buffer format Please refer to gm_clear_window_yuv_t.	Input (GM_FMT_YUV422t)
*buf	Pointer of the pattern buffer	Input point
out_x	x coordinate of the OSD window	Input (≥ 0)
out_y	y coordinate of the OSD window	Input (≥ 0)
out_w	Width of the display window	Input (≥ 0)
out_h	Height of the display window	Input (≥ 0)
Mode	Clear window active mode Please refer to gm_clear_window_mode_t. “Don’t care” when clear bit-stream	Input (GM_ACTIVE_BY_APPLY or GM_ACTIVE_IMMEDIATELY)

- Note
  - Please refer to liveview_with_clearwin.c.

3.2.31 region_rawdata_t

- Definition

```
Typedef_enum {
    GM_PARTIAL_FROM_CAP_OBJ = 0xFBFB9933,
    GM_WHOLE_FROM_ENC_OBJ = 0xFBFC9934,
}gm_rawdata_mode_t;
typedef struct rawdata {
    gm_rect_t region;
    char *yuv_buf;
    unsigned int yuv_buf_len;
    gm_rawdata_mode_t rawdata_mode;
    int reserved[4];
} region_rawdata_t;
```

- Description
  - The parameters for getting capture raw data with regions.

- Members

Member	Description	Value
region	Specify area of a region includs the (x, y) width and height	Input x: 0 ~ (gm_system.cap[cap_vch].dim.width - 1) y: 0 ~ (gm_system.cap[cap_vch].dim.height - 1) Width: 1 ~ (gm_system.cap[cap_vch].dim.width - x) Height: 1 ~ (gm_system.cap[cap_vch].dim.height - y)
yuv_buf	The pointer of the buffer for storing the raw data	Input
yuv_buf_len	Buffer length equals to (region.width * region.height * 2).	Input (> 0)
rawdata_mode	Specify the position of the graph link for getting the raw data. (Default: GM_PARTIAL_FROM_CAP_OBJ) GM_PARTIAL_FROM_CAP_OBJ: From capture output frame buffer. (to use region.x / region.y / region.width / region.height to crop the partial area) GM_WHOLE_FROM_ENC_OBJ: From encode input frame buffer. (To use region.width / region.height to output the whole dimension with scaling-down.)	Input

- Note
  - Please follow this flow to get gm_system:

    gm_system_t gm_system;

    gm_init();

    gm_get_sysinfo(&gm_system);

3.2.32 disp_snapshot_t

- Definition

```
typedef enum {
    GM_SNAPSHOT_P_FRAME = 0,
    GM_SNAPSHOT_I_FRAME = 2,
} slice_type_t;
typedef struct disp_snapshot {
    int  lcd_vch;
    int  image_quality;
    char *bs_buf;
    unsigned int bs_buf_len;
    int bs_width;
    int bs_height;
    int bs_type;
    int slice_type;
    int reserved[4];
} disp_snapshot_t;
```

- Description
  - Display the snapshot data structure

- Members

Member	Description	Value
lcd_vch	vch of the LCD display (It only supports LCD0.) Please refer to gm_lcd_sys_info_t.	Input (≥ 0)
image_quality	Value of the image quality	Input 1 (Worst) ~ 100 (Best)
bs_buf	Pointer of the bitstream buffer	Input character point
Bs_buf_len	Users prepared bs_buf length	Input (> 0)
bs_width	Bitstream width	Output
bs_height	Bitstream height	Output
bs_type	Bitstream type 0: SNAPSHOT_JPEG 1: SNAPSHOT_H264	Output
slice_type	Specify the frame type for the snapshot GM_SNAPSHOT_P_FRAME or GM_SNAPSHOT_I_FRAME	Input
reserved	Reserved	None

- Note
  - lcd_vch: lcd_vch must be available in the system. Please refer to gm_lcd_sys_info_t.

3.2.33 snapshot_t

- Definition

```
typedef struct snapshot {
    void *bindfd;
    int  image_quality;
    char *bs_buf;
    unsigned int bs_buf_len;
    int bs_width;
    int bs_height;
    unsigned int extra_mode;
    int  reserved[2];
} snapshot_t;
```

- Description
  - Encoder snapshot data structure

- Members

Member	Description	Value
bindfd	fd of gm_bind	Input void point
image_quality	Value of the image quality	Input 1 (Worst) ~ 100 (Best)
bs_buf	Pointer of the bitstream buffer	Input character point
bs_buf_len	Users prepared bs_buf length	Input (≥ 0)
bs_width	The width of the bitstream content	Output
bs_height	The height of the bitstream content	Output
extra_mode	0x2694: Interference (JPEG only for cap vch0 without scaling)	Input
reserved	Reserved	None

- Note
  - Encoder snapshot

3.2.34 decode_keyframe_t

- Definition

```
typedef struct decode_keyframe {
   int  bs_width;
   int  bs_height;
   char *bs_buf;
   int  bs_buf_len;
   int  rgb_width;
   int  rgb_height;
   char *rgb_buf;
   int  rgb_buf_len;
   int reserved[5];
} decode_keyframe_t;
```

- Description
  - Decode a keyframe into the RGB buffer data structure

- Members

Member	Description	Value
bs_width	Bitstream width	Input (> 0)
bs_height	Bitstream height	Input (> 0)
bs_buf	Pointer of the bitstream buffer	Input character point
bs_buf_len	Bitstream length	Input (> 0)
rgb_width	RGB width	Input (> 0)
rgb_height	RGB height	Input (> 0)
rgb_buf	Pointer of the RGB buffer	Input character point
rgb_buf_len	RGB length	Input (> 0)

3.2.35 gm_notify_t

- Definition

```
typedef enum {
   GM_NOTIFY_SIGNAL_LOSS = 0,
   GM_NOTIFY_SIGNAL_PRESENT,
   GM_NOTIFY_FRAMERATE_CHANGE,
   GM_NOTIFY_HW_CONFIG_CHANGE,
   GM_NOTIFY_TAMPER_ALARM,
   GM_NOTIFY_TAMPER_ALARM_RELEASE,
   GM_NOTIFY_AUDIO_BUFFER_UNDERRUN,
   GM_NOTIFY_AUDIO_BUFFER_OVERRUN,
   MAX_GM_NOTIFY_COUNT,
} gm_notify_t;
```

- Description
  - The event of notification

- Lists

List	Description
GM_NOTIFY_SIGNAL_LOSS	Capture signal loss
GM_NOTIFY_SIGNAL_PRESENT	Capture signal present
GM_NOTIFY_FRAMERATE_CHANGE	Frame rate changed by capture itself
GM_NOTIFY_HW_CONFIG_CHANGE	Hardware CONFIG change such as the below cases 1. Capture signal from PAL to NTSC or from NTSC to PAL 2. LCD input resolution change
GM_NOTIFY_TAMPER_ALARM	Alarm from the tamper detection
GM_NOTIFY_TAMPER_ALARM_RELEASE	Release alarm from the tamper detection
GM_NOTIFY_AUDIO_BUFFER_UNDERRUN	Audio internal buffer is underrun when playback
GM_NOTIFY_AUDIO_BUFFER_OVERRUN	Audio internal buffer is overrun when record
MAX_GM_NOTIFY_COUNT	Library internal definition, do not use it

3.2.36 gm_cap_tamper_t

- Definition

```
typedef struct {
   unsigned short tamper_sensitive_b;
   unsigned short tamper_threshold;
   unsigned short tamper_sensitive_h;
   int reserved[5];
} gm_cap_tamper_t;
```

- Description
  - The parameters of the capture tamper detection

- Members

Member	Description	Value
tamper_sensitive_b	Sensitive level for the black scene 0: Disable	Input 0: Disable 1 (Less) ~ 100 (More)
Tamper_threshold	The index of the luminance threshold	Input (1~255)
tamper_sensitive_h	Sensitive level for the homogenous scene 0: Disable	Input 0: Disable 1 (Less) ~ 100 (More)

3.2.37 Time_align_t

- Definition

```
typedef enum {
   TIME_ALIGN_ENABLE = 0xFEFE01FE,
   TIME_ALIGN_ENABLE = 0xFEFE07FE
} time_align_t;
```

- Description
  - The event of notification

- List

List	Description
TIME_ALIGN_ENABLE	Enable time alignment
TIME_ALIGN_ENABLE	Disable time alignment

3.2.38 gm_h264e_profile_t

- Definition

```
typedef enum {
   GM_H264E_DEFAULT_PROFILE = 0,
   GM_H264E_BASELINE_PROFILE = 66,
   GM_H264E_MAIN_PROFILE = 77,
   GM_H264E_HIGH_PROFILE = 100
} gm_h264e_profile_t;
```

- Description
  - The event of notification
- List

List	Description
GM_H264E_DEFAULT_PROFILE	Specific default profile mode for h264e
GM_H264E_BASELINE_PROFILE	Specific baseline profile mode for h264e
GM_H264E_MAIN_PROFILE	Specific main profile mode for h264e
GM_H264E_HIGH_PROFILE	Specific high profile mode for h264e

3.2.39 gm_cap_motion_t

- Definition

```
typedef struct {
   unsigned int id;
   unsigned int value;
} gm_cap_motion_t;
```

- Description
  - The parameters of the capture motion detection

- Members

Member	Description	Value
id	Specific the parameter ID for the setting of the capture motion Please refer to motion_detection_introduction for the definition of the parameter ID.	Input
value	Specific parameter value for the setting of the capture motion	Input

3.2.40 gm_checksum_type_t

- Definition

```
typedef enum  {
    GM_CHECKSUM_NONE = 0x0,
    GM_CHECKSUM_ALL_CRC = 0x101,
    GM_CHECKSUM_ALL_SUM = 0x0102,
    GM_CHECKSUM_ALL_4_BYTE = 0x103,
    GM_CHECKSUM_ONLY_I_CRC = 0x201,
    GM_CHECKSUM_ONLY_I_SUM = 0x0202,
    GM_CHECKSUM_ONLY_I_4_BYTE = 0x203
} gm_checksum_type_t;
```

- Description
  - The type of checksum

- List

List	Description
GM_CHECKSUM_NONE	No checksum
GM_CHECKSUM_ALL_CRC	All frames use the crc method
GM_CHECKSUM_ALL_SUM	All frames use the sum method
GM_CHECKSUM_ALL_4_BYTE	All frames use the 4-byte sum method
GM_CHECKSUM_ONLY_I_CRC	Only I frames use the crc method

List	Description
GM_CHECKSUM_ONLY_I_SUM	Only I frames use the sum method
GM_CHECKSUM_ONLY_I_4_BYTE	Only I frames use the 4-byte sum method

3.2.41 gm_fast_forward_t

- Definition

```
typedef enum  {
    GM_FASTFORWARD_NONE = 0,
    GM_FASTFORWARD_1_FRAME = 2,
    GM_FASTFORWARD_3_FRAMES = 4
} gm_fast_forward_t;
```

- Description
  - The mode of fast forward

- List

List	Description
GM_FASTFORWARD_NONE	No fastforward
GM_FASTFORWARD_1_FRAME	Skip one frame
GM_FASTFORWARD_3_FRAMES	Skip three frames

3.2.42 gm_h264e_level_t

- Definition

```
typedef enum {
    GM_H264E_DEFAULT_LEVEL = 0,
    GM_H264E_LEVEL_1 = 10,
    GM_H264E_LEVEL_1_1 = 11,
    GM_H264E_LEVEL_1_2 = 12,
    GM_H264E_LEVEL_1_3 = 13,
    GM_H264E_LEVEL_2 = 20,
    GM_H264E_LEVEL_2_1 = 21,
    GM_H264E_LEVEL_2_2 = 22,
    GM_H264E_LEVEL_3 = 30,
    GM_H264E_LEVEL_3_1 = 31,
    GM_H264E_LEVEL_3_2 = 32,
    GM_H264E_LEVEL_4 = 40,
    GM_H264E_LEVEL_4_1 = 41,
    GM_H264E_LEVEL_4_2 = 42,
    GM_H264E_LEVEL_5 = 50,
    GM_H264E_LEVEL_5_1 = 51
} gm_h264e_level_t;
```

- Description
  - The level of profile of h264e

- List

List	Description
GM_H264E_DEFAULT_LEVEL	Default Level
GM_H264E_LEVEL_1	Level 1.0
GM_H264E_LEVEL_1_1	Level 1.1
GM_H264E_LEVEL_1_2	Level 1.2
GM_H264E_LEVEL_1_3	Level 1.3
GM_H264E_LEVEL_2	Level 2.0
GM_H264E_LEVEL_2_1	Level 2.1
GM_H264E_LEVEL_2_2	Level 2.2
GM_H264E_LEVEL_3	Level 3.0
GM_H264E_LEVEL_3_1	Level 3.1
GM_H264E_LEVEL_3_2	Level 3.2
GM_H264E_LEVEL_4	Level 4.0
GM_H264E_LEVEL_4_1	Level 4.1
GM_H264E_LEVEL_4_2	Level 4.2
GM_H264E_LEVEL_5	Level 5.0
GM_H264E_LEVEL_5_1	Level 5.1

3.2.43 gm_h264e_config_t

- Definition

```
typedef enum {
  GM_H264E_DEFAULT_CONFIG = 0,
  GM_H264E_PERFORMANCE_CONFIG = 1,
  GM_H264E_LIGHT_QUALITY_CONFIG = 2,
  GM_H264E_QUALITY_CONFIG = 3
} gm_h264e_config_t;
```

- Description
  - The config of profile of h264e

- List

List	Description
GM_H264E_DEFAULT_CONFIG	Default config
GM_H264E_PERFORMANCE_CONFIG	Config for performance
GM_H264E_LIGHT_QUALITY_CONFIG	Config for light quality
GM_H264E_QUALITY_CONFIG	Config for quality

3.2.44 gm_h264e_coding_t

- Definition

```
typedef enum {
    GM_H264E_DEFAULT_CODING = 0,
    GM_H264E_CABAC_CODING = 1,
    GM_H264E_CAVLC_CODING = 2,
} gm_h264e_coding_t;
```

- Description
  - The coding method of profile of h264e

- List

List	Description
GM_H264E_DEFAULT_CODING	Default conding
GM_H264E_CABAC_CODING	Specify CABAC coding
GM_H264E_CAVLC_CODING	Specify CAVLC coding

Chapter 4

GM_LIB Attribute Structure

This chapter contains the following sections:

4.1 Capture Attribute

4.1.1 gm_cap_attr_t

- Definition

```
typedef struct  {
    gm_priv_t priv;
    int cap_vch;
    int path;
    int enable_mv_data;
    int I2P_threshold;
    unsigned int dma_path;
    unsigned short prescale_reduce_width;
    unsigned short prescale_reduce_height;
    int reserved[2];
} gm_cap_attr_t;
```

- Description
  - Capture basic attribute

- Members

Member	Description	Value
Priv	Library internal data, do not use it	None
cap_vch	The index of the specific virtual channel	Input 0 ~ (CAPTURE_VCH_NUMBER - 1)
path	The index of the specific path of the channel which was specified by cap_vch. The path is the duplicate unit from this channel. GM8210: Path usage rule PATH0: Liveview PATH1: CVBS PATH2: Can scaling PATH3: Can scaling GM8139/GM8287: Path usage rule PATH0: Liveview PATH1: CVBS PATH2: Can scaling PATH3: Cannot scaling	Input 0 ~ (number_of_path - 1)
enable_mv_data	It turns on the capture motion data.	Input (1 or 0)

Member	Description	Value
I2P_threshold	If dest_w < (src_w * I2P_threshold), transfer interlace to progressive by capture, and then without 3DI action. (Unit is percentage) Src_w: gmSystem.cap[].dim.width Dest_w: winAttrs.basic.rect.width	Input (0 ~100, value = N means N%)
dma_path	DMA path 0: DMA0 1: DMA1	Input (0 or 1)
prescale_reduce_width	It is used for capture prescale reduce width adjust	(< gm_system.cap[cap_vch].dim.width)
prescale_reduce_height	It is used for capture prescale reduce height adjust	(< gm_system.cap[cap_vch].dim.height)

- Notes
  - Capture the basic attribute. It needs to be set to capture an object by using the gm_set_attr() function.
  - cap_vch/path: cap_vch/path must be available in the system. Please refer to gm_cap_sys_info_t.
  - Example: To get cap_vch and number_of_path from available captures:

```
gm_system_t gmSystem;
gm_init();
gm_get_sysinfo(&gmSystem);
for (i = 0; i < CAPTURE_VCH_NUMBER; i++) {
    if (gmSystem.cap[cap_vch].valid >= 0) {
        printf("cap_vch = %d is available. and the number of path is %d\n", cap_vch,
gmSystem.cap[cap_vch].number_of_path);
    } else {
    printf("cap_vch = %d is non-available\n");
}
```

4.1.2 gm_crop_attr_t

- Definition

```
typedef struct  {
    gm_priv_t priv;
    int enabled;
    gm_rect_t src_crop_rect;
    char cvbs_quality;
    char reserved[19];
} gm_crop_attr_t;
```

- Description
  - Capture crop attribute

- Members

Member	Description	Value
Priv	Library internal data, do not use it	None
enabled	Enable this feature or not 1: Enable 0: Disable	Input (0 or 1)
src_crop_rect	Specify the area of the crop which includes the (x, y) width and height	Input
cvbs_quality	Crop from the original capture source or scaling-down(D1) capture source. 0: Low quality (Scaling-down source) 1: High quality (Original source)	Input (0 or 1)

- Notes
  - This attribute is for GM_CAP_OBJECT and GM_FILE_OBJECT.

4.1.3 gm_3di_attr_t

- Definition

```
typedef struct  {
    gm_priv_t priv;
    int deinterlace;
    int denoise;
    int stand_alone;
   int reserved[4];
} gm_3di_attr_t;
```

- Description
  - Capture temporal de-interlace/de-noise attribute

- Members

Member	Description	Value
priv	Library internal data, do not use it	None
deinterlace	Control the capture temporal de-Interlace 1: Enable 0: Disable	Input (0 or 1)
denoise	Control the capture temporal de-noise 1: Enable 0: Disable	Input (0 or 1)
stand_alone	1: External dedicated 3di 0: Embedded 3di if it is available by codec	Input (0 or 1)

- Notes
  - The attribute is optional.
  - Only for the record function. Liveview is not needed.
  - Please refer to encode_with_deinterlace.c for the demonstration.

4.1.4 gm_3dnr_attr_t

- Definition

```
typedef struct  {
    gm_priv_t priv;
    int enabled;
    int reserved[5];
} gm_3dnr_attr_t;
```

- Description
  - 3D de-noise feature

- Members

Member	Description	Value
priv	Library internal data, do not use it	None
enable	Enable/Disable the feature of 3D de-noise 1: Enable 0: Disable	Input (0 or 1)

- Notes
  - The attribute is optional.
  - Only for the GM813x streaming function. The others are not available.
  - Please refer to rtspd.c for the demonstration.

4.1.5 gm_rotation_attr_t

- Definition

```
typedef struct  {
    gm_priv_t priv;
    int enabled;
    int clockwise;
    int reserved[4];
} gm_rotation_attr_t;
```

- Description
  - Encode streaming rotation feature

- Members

Member	Description	Value
priv	Library internal data, do not use it	None
enable	Enable/Disable the feature of rotation 1: Enable 0: Disable	Input (0 or 1)
clockwise	Rotation clockwise setting 1: Clockwise 90 degrees 0: Counterclockwise 90 degrees	Input (0 or 1)

- Notes
  - The attribute is optional.
  - Only for the GM813x streaming function. The others are not available.
  - Please refer to rtspd.c for the demonstration.

4.2 Video Encoder Attribute

4.2.1 gm_h264e_attr_t

- Definition

```
typedef struct {
    gm_priv_t priv;
    gm_dim_t dim;
    union {
        int framerate;
        struct {
            int numerator:16;
            int denominator:16;
        } fps_ratio;
    } frame_info;
    gm_enc_ratecontrol_t ratectl;
    int     b_frame_num;
    int     enable_mv_data;
    struct {
        gm_h264e_profile_t profile:8;
        gm_h264e_level_t level:8;
        gm_h264e_config_t config:8;
        gm_h264e_coding_t coding:8;
    } profile_setting;
    struct {
        char ip_offset:8;
        char roi_delta_qp:8;
        char reserved1:8;
        char reserved2:8;
    } qp_offset;
    gm_checksum_type_t checksum_type;
    gm_fast_forward_t fast_forward;
    int     reserved[1];
} gm_h264e_attr_t;
```

- Description
  - Basic attribute of H264 encoder

- Members

Member	Description	Value
Priv	Library internal data, do not use it	None
Dim	Dimension width and height of the encode region	Input (> 0)
framerate	Frame rate	Input (1 ~ max_fps)
fps_ratio.numerator	Numerator for the frame rate ratio	Input (1 ~ 65535, numerator <= denominator)

Member	Description	Value
fps_ratio.denominator	Denominator for the frame rate ratio	Input (1 ~ 65535, numerator <= denominator)
ratectl	Bit rate control Please refer to gm_enc_ratecontrol_t.	Input structure
b_frame_num	The number of B frames	Input (> 0, It depends on the memory size.)
enable_mv_data	Reserved, do not use it. (0: Disable)	Input (0)
profile_setting	Please refer to gm_h264e_profile_t. Please refer to gm_h264e_level_t. Please refer to gm_h264e_config_t. Please refert to gm_h264e_coding_t.	Input
qp_offset	Image quality difference: ip_offset: QP offsets between I frame and P frame roi_delta_qp: QP offsets between the specific regions and other regions	Input
checksum_type	The method of checksum which is applied for errors detected from bitstream of encode.	Input
fast_forward	The method of “fast_forward” which is applied for the bitstream of encode.	Input

- Notes
  - H264 encoder basic attributes. It needs to be set to encode an object by using the gm_set_attr() function.
  - If the basic attributes of H264 encoder are selected to encode an object, users cannot choose another codec, such as the mpeg4 or mjpeg basic attributes.
  - fps_ratio: The frame rate is expressed as a ratio. The formula is “framerate = fps_ratio * src_framerate”. For example, if framerate = 15, fps_ratio is 15/30 and src_framerate = 30fps, and can be expressed as fps_ratio.numerator = 15, fps_ratio.denominator = 30.
  - How to get the max_fps? Please follow this rule.

    gm_system_t gm_system;

    gm_init(); // GM library initial

    gm_get_sysinfo(&gm_system);

    printf(“framerate=%d”,gm_system.cap[cap_vch].framerate);
  - The frame rate can be set by framerate or fps_ratio. Users can choose one of two methods for the frame rate setting. For example,

- If the frame rate is 15 fps, framerate = 15.
- If the frame rate is 7.5 fps (src_framerate = 30fps), fps_ration.numerator = 15 and fps_ratio.denominator = 60.

4.2.2 gm_mpeg4e_attr_t

- Definition

```
typedef struct {
    gm_priv_t priv;
    gm_dim_t dim;
    union {
        int framerate;
        struct {
            int numerator:16;
            int denominator:16;
        } fps_ratio;
    } frame_info;
    gm_enc_ratecontrol_t ratectl;
    int reserved[5];
} gm_mpeg4e_attr_t;
```

- Description
  - Basic attribute of MPEG4 encoder

- Members

Member	Description	Value
priv	Library internal data, do not use it	None
dim	Dimension width and height of the encode region	Input (> 0)
framerate	Frame rate	Input (> 0)
fps_ratio.numerator	Numerator for the frame rate ratio	Input (> 0)
fps_ratio.denominator	Denominator for the frame rate ratio	Input (> 0)
ratectl	Bit rate control Please refer to gm_enc_ratecontrol_t.	Input structure

- Notes
  - The basic attributes of the MPEG4 encoder. It needs to be set to encode an object by using the gm_set_attr() function.
  - If the basic attributes of the MPEG4 encoder are selected to encode an object, users cannot choose another codec, such as the basic attributes of the H264 or mjpeg encoder.

- fps_ratio: The frame rate is expressed as a ratio. For example, if framerate = 15, fps_ratio is 15/30, and expressed as fps_ratio.numerator = 15, fps_ratio.denominator = 30.
- The frame rate can be set by framerate or fps_ratio. Users can choose one of the both methods for setting.
- For example,
  - If the frame rate is 15 fps, framerate = 15.
  - If the frame rate is 7.5 fps, fps_ration.numerator = 15 and fps_ratio.denominator = 60.

4.2.3 gm_mjpege_attr_t

- Definition

```
typedef struct {
    gm_priv_t priv;
    gm_dim_t dim;
    union {
        int framerate;
        struct {
            int numerator:16;
            int denominator:16;
        } fps_ratio;
    } frame_info;
    int quality;
    gm_enc_ratecontrol_mode_t mode;
    int bitrate;
    int bitrate_max;
    int reserved[2];
} gm_mjpege_attr_t;
```

- Description
  - Basic attributes of the mjpeg encoder

- Members

Member	Description	Value
priv	Library internal data, do not use it	None
dim	Width and height of a encode region	Input (> 0)
framerate	Frame rate	Input (> 0)
fps_ratio.numerator	Frame rate ratio numerator	Input (> 0)
fps_ratio.denominator	Frame rate ratio denominator	Input (> 0)
quality	Image quality	Input 1 (Worst) ~ 100 (Best)

Member	Description	Value
mode	Rate control mode	Input Please refer to gm_enc_ratecontrol_mode_t.
Bitrate	It is the average number of bits per second.	Input (> 0)
bitrate_max	Maximum bitrate	Input (> 0)

- Notes
  - Do not support the rate control because it is the JPEG format.
  - The basic attributes of the mjpeg encoder. It needs to be set to encode an object by using the gm_set_attr() function.
  - If the mjpeg encoder basic attributes are selected to encode an object, users cannot choose another codec, such as the MPEG4 or H264 basic attributes.
  - fps_ratio: The frame rate is expressed as a ratio. For example, if framerate = 15, fps_ratio is 15/30, and expressed as fps_ratio.numerator = 15, fps_ratio.denominator = 30.
  - The frame rate can be set by framerate or fps_ratio. Users can choose one of both methods for setting.
  - For example,
    - If frame rate is 15 fps, framerate = 15.
    - If frame rate is 7.5 fps, fps_ration.numerator = 15 and fps_ratio.denominator = 60.

4.2.4 gm_enc_ratecontrol_t

- Definition

```
typedef struct {
    gm_enc_ratecontrol_mode_t mode;
    int gop;
    int init_quant;
    int min_quant;
    int max_quant;
    int bitrate;
    int bitrate_max;
    int reserved[5];
} gm_enc_ratecontrol_t;
```

- Description
  - Encode the rate control attribute

- Members

Member	Description	Value
mode	Rate control mode	Input Please refer to gm_enc_ratecontrol_mode_t.
gop	It will be the period when an I-frame occurs.	Input (> 0)
init_quant	Init quant	Input H264: 1 ~ 51 MPEG4: 1 ~ 31
min_quant	Minimum quant	Input H264: 1 ~ 51 MPEG4: 1 ~ 31
max_quant	Maximum quant	Input H264: 1 ~ 51 MPEG4: 1 ~ 31
Bitrate	It is the average number of bits per second.	Input (> 0)
bitrate_max	Maximum bitrate	Input (> 0)

- Notes
  - Only for the H264 and MPEG4 attributes
  - min_quant <= init_quant <= max_quant

4.2.5 gm_enc_roi_attr_t

- Definition

```
typedef struct {
    gm_priv_t priv;
    int enabled;
    gm_rect_t rect;
    int reserved[5];
} gm_enc_roi_attr_t;
```

- Description
  - Encode the ROI attribute

- Members

Member	Description	Value
priv	Library internal data, do not use it	None
enabled	Enable or disable the ORI function	Input (1 or 0)
rect	Specify a region for encoding. The region includes (x, y) width and height.	Input x: 0 ~ (gm_system.cap[cap_vch].dim.width - 128) y: 0 ~ (gm_system.cap[cap_vch].dim.height - 96) Width: (128 ~ 4096) Height: (96 ~ 4096)

- Notes
  - The attribute is optional.
  - Please follow this flow to get gm_system:

    gm_system_t gm_system;

    gm_init();

    gm_get_sysinfo(&gm_system);

4.2.6 gm_h264_advanced_attr_t

- Definition

```
typedef struct  {
     gm_priv_t priv;
     int multi_slice;
     int field_coding;
     int gray_scale;
     int reserved[5];
} gm_h264_advanced_attr_t;
```

- Description
  - It is for the advanced parameters of the H264 encoder.

- Members

Member	Description	Value
priv	Library internal data, do not use it	None
multi_slice	Number of MB rows of one slice 0: Single slice > 0: Multiple slices More multi-slice number could reduce the number of error areas from decoding because bitstream has an error, especially bitstream transfers via network. But users should pay attention to the performance drop in case of more multi-slice usages.	Input 0 ~ 4
field_coding	Field coding or frame coding 0: Frame coding 1: Field coding	Input (0 or 1)
gray_scale	Chroma sampling relative to luma sampling 0: 4:2:0 chroma format 1: Mono	Input (0 or 1)

4.2.7 gm_h264_roiqp_attr_t

- Definition

```
typedef struct  {
     gm_priv_t priv;
     int enabled;
     gm_rect_t  rect[GM_MAX_ROI_QP_NUMBER];
     int reserved[5];
} gm_h264_roiqp_attr_t;
```

- Description
  - It is for the H264 object to enhance the picture quality of a region.

- Members

Member	Description	Value
priv	Library internal data, do not use it	None
enabled	Enable or disable the ORI qp function	Input 0: Disable 1: Enable

Member	Description	Value
rect	Select the enhanced specific region picture quality. The region parameter includes (x, y) width and height	Input x: 0 ~ (gm_system.cap[cap_vch].dim.width - 1) y: 0 ~ (gm_system.cap[cap_vch].dim.height - 1) Width: 1 ~ (gm_system.cap[cap_vch].dim.width - x) Height: 1 ~ (gm_system.cap[cap_vch].dim.height - y)

- Notes

  Please follow this flow to get gm_system:

  gm_system_t gm_system;

  gm_init();

  gm_get_sysinfo(&gm_system);

4.2.8 gm_h264_watermark_attr_t

- Definition

```
typedef struct  {
     gm_priv_t priv;
     int pattern;
     int reserved[5];
} gm_h264_watermark_attr_t;
```

- Description
  - It is for the H264 object to setup the watermark pattern.

- Members

Member	Description	Value
priv	Library internal data, do not use it	None
pattern	Key to encrypt an image	Input Integer

4.2.9 gm_h264_vui_attr_t

- Definition

```
typedef struct  {
    gm_priv_t priv;
    union {
        int value;
        struct {
            char matrix_coefficient;
            char transfer_characteristics;
            char colour_primaries;
            char full_range : 1;
            char video_format : 3;
            char timing_info_present_flag : 1;
        } param;
    } param_info;
    union {
        int value;
        struct {
            unsigned short sar_width;
            unsigned short sar_height;
        } sar;
    }sar_info;
    int reserved[5];
}gm_h264_vui_attr_t;
```

- Description
  - It is for the H264 object to setup the encoder parameter.

- Members

Member	Description	Value
priv	Library internal data, do not use it	None
param.matrix_coefficient	Martrix coefficients are used to derive the luma and chroma signals from green, blue, and red primaries.	Input (0 ~ 255)
param.transfer_characteristics	The opto-electronic transfers characteristic of the source pictures	Input (0 ~ 255)
param.colour_primaries	Chromaticity coordinates the source primaries	Input (0 ~ 255)
param.full_range	Indicate the black level and range of the luma and chroma signals 0: Not full range 1: Full range	Input (0 or 1)

Member	Description	Value
param.video_format	Indicate the representation of pictures 0: Component 1: PAL 2: NTSC 3: SECAM 4: MAC 5: Unexpected video format 6 and 7: Reserved	Input (0 ~ 7)
param.timing_info_present_flag	Specify whether the frame rate is presented in the bitstream 0: Not present 1: Present	Input (0 or 1)
sar.sar_width	Horizontal size of the sample aspect ratio	Input (0 ~ 65535)
sar.sar_height	Vertical size of the sample aspect ratio	Input (0 ~ 65535)

4.2.10 gm_raw_attr_t

- Definition

```
typedef struct {
    gm_priv_t priv;
    gm_dim_t dim;
    union {
        int framerate;
        struct {
            int numerator:16;
            int denominator:16;
        } fps_ratio;
    } frame_info;
    int reserved[5];
} gm_raw_attr_t;
```

- Description
  - Get capture raw data without encoding

- Members

Member	Description	Value
Priv	Library internal data, do not use it	None
dim	Specify the output dimension that must be equal to or smaller than the capture source.	Input (dim.width <= gm_system.cap[vch].dim.width) (dim.height <= gm_system.cap[vch].dim.height)
framerate	Frame rate	Input (1 ~ max_fps)

Member	Description	Value
fps_ratio.numerator	Numerator for the frame rate ratio	Input (1 ~ 65535, numerator <= denominator)
fps_ratio.denominator	Denominator for the frame rate ratio	Input (1 ~ 65535, numerator <= denominator)

- Notes
  - The attribute is optional.
  - Please follow this flow to get gm_system:

    gm_system_t gm_system;

    gm_init();

    gm_get_sysinfo(&gm_system);

4.3 File Attribute

4.3.1 gm_file_attr_t

- Definition

```
typedef struct {
    gm_priv_t priv;
    int vch;
    int max_width;
    int max_height;
    int max_fps;
    unsigned short sample_rate;
    unsigned short sample_size;
    gm_audio_channel_type_t channel_type;
    int reserved[3];
} gm_file_attr_t;
```

- Description
  - Capture the basic attributes

- Members

Member	Description	Value
priv	Library internal data, do not use it	None
vch	File virtual channels (Any value given by application, which must be unique)	Input
max_width	For video data, specify the max. width	Input (> 0)
max_height	For video data, specify the max. height	Input (> 0)
max_fps	For video data, specify the max. framerate	Input (1 ~ display_rate)

Member	Description	Value
sample_rate	For audio data, specify the sample rate	Input
sample_size	For audio data, specify the sample size	Input
channel_type	For audio data, specify the channel counts	Input

4.4 Window Attribute

4.4.1 gm_win_attr_t

- Definition

```
typedef enum {
    WIN_LAYER1 = 0,
    WIN_LAYER2,
    COUNT_OF_WIN_LAYERS
} gm_win_layer_t;

typedef enum {
    DISABLE_WIN_TO_ENC = 0,
    FROM_MULTIWIN = 1,
    FROM_MAIN_DISPLAY = 2,
} gm_win_to_enc_t;


typedef struct {
    gm_priv_t priv;
    int lcd_vch;
    int visible;
    gm_rect_t rect;
   gm_win_to_enc_t win_to_enc;
   gm_win_layer_t layer;
   int bg_dim_width;
   int bg_dim_height;
    unsigned int feature_bmp;
} gm_win_attr_t;
```

- Description
  - the basic attributes of window

- Members

Member	Description	Value
priv	Library internal data, do not use it	None
lcd_vch	The index of display LCD	Input (≥ 0)
visible	0: Video shows in the background. 1: Video shows in the foreground.	Input (1 or 0)

Member	Description	Value
rect	Specify the display region on the display LCD The region parameter includes the (x, y) width and height.	Input x: 0 ~ (gm_system.lcd[lcd_vch].dim.width - 1) y: 0 ~ (gm_system.lcd[lcd_vch].dim.height - 1) Width: 1 ~ (gm_system.lcd[lcd_vch].dim.width - x) Height: 1 ~ (gm_system.lcd[lcd_vch].dim.height - y)
win_to_enc	Enable multi-cap to the H264 record function. 0: DISABLE_WIN_TO_ENC: Window object will not bind to the encode object for the display case. 1: FROM_MULTIWIN: Window object will bind to the encode object for recording and not to be displayed. 2: FROM_MAIN_DISPLAY: Window object will be displayed (e.g. liveview or playback) and display the encoded buffer at the same time.	Input(0 / 1 / 2)
layer	The layer of this window. If users do not fill it, this variable is WIN_LAYER1 in default. Layer config 0: Background 1: Second layer	Input (0 / 1)
bg_dim_width	In the multi_win record mode, specify the width of the H264 encode Other modes: Do not care.	Input (>= 128)
bg_dim_height	In the multi_win record mode, specify the height of the H264 encode. Other modes: Do not care.	Input (>= 96)
feature_bmp	Bit 0 : Methods 0 and 4, enable interlaced video 60i display Bit 31: Reserved, please don't use it	Input bitmap

- Notes
  - lcd_vch: lcd_vch must be available in the system. Please refer to gm_lcd_sys_info_t.
  - Please follow this flow to get gm_system:

    ```
    gm_system_t gm_system;
    gm_init();
    gm_get_sysinfo(&gm_system);
    ```

- Please refer to liveview_1div_to_4_div.c for the demonstration.

4.4.2 gm_win_aspect_ratio_attr_t

- Definition

```
typedef struct {
    gm_priv_t priv;
    int enabled;
    int palette_idx;
    int reserved[5];
} gm_win_aspect_ratio_attr_t;
```

- Description
  - The aspect-ratio attributes of window

- Members

Member	Description	Value
priv	Library internal data, do not use it	None
enabled	Enable/Disable the feature of the aspect ratio	Input (0 or 1)
palette_idx	The palette index for specifying the background color to fill the un-covered area inside the window when the aspect ratio is enabled.	Input (0 ~ 15)

4.4.3 gm_win_border_attr_t

- Definition

```
typedef struct {
    gm_priv_t priv;
    int enabled;
    int width;
    int palette_idx;
    int reserved[5];
} gm_win_border_attr_t;
```

- Description
  - The border attributes of window

- Members

Member	Description	Value
priv	Library internal data, do not use it	None
enabled	Enable/Disable the feature of the border	Input (0 or 1)
width	Width of the border	Input
palette_idx	Specify colors for the border	Input (0 ~ 15)

4.5 Audio Grab Attribute

4.5.1 gm_audio_grab_attr_t

- Definition

```
typedef struct  {
    gm_priv_t priv;
    int vch;
    int sample_rate;
    int sample_size;
    gm_audio_channel_type_t channel_type;
    int reserved[5];
} gm_audio_grab_attr_t;
```

- Description
  - Audio grabber basic attributes

- Members

Member	Description	Value
Priv	Library internal data, do not use it	None
vch	Number of the audio virtual channels	Input (> 0)
sample_rate	Audio sampling rate	input (> 0)
sample_size	The bits number per sample	Input (8 or 16)
channel_type	Channel type (GM_MONO/GM_STEREO)	Input (GM_MONO or GM_STEREO)

4.6 Audio Encoder Attribute

4.6.1 gm_audio_enc_attr_t

- Definition

```
typedef enum {
    GM_PCM = 1,
    GM_AAC,
    GM_ADPCM,
    GM_G711_ALAW,
    GM_G711_ULAW
} gm_audio_encode_type_t;

typedef struct  {
    gm_priv_t priv;
    gm_audio_encode_type_t encode_type;
    int bitrate;
    int block_count;
    int frame_samples;
    int reserved[4];
} gm_audio_enc_attr_t;
```

- Description
  - Audio encoder basic attributes

- Members

Member	Description	Value
priv	Library internal data, do not use it	None
encode_type	Audio codec type GM_PCM/GM_AAC/GM_ADPCM/ GM_G711_ALAW/GM_G711_ULAW	Input Please refer to gm_audio_encode_type_t.
bitrate	It is the average number of bits per second.	Input (> 0)
block_count	The count of the audio frames received once in a single received operation PCM: 250 ~ 2048 AAC: 1024 * n (Mono) 2048 * n (Stereo) G711: 320 * n (Mono) 640 * n (Stereo) (n: 1, 2, ….)	Input
frame_samples	The count of the samples as an audio sample This member only takes effects on PCM.	Input (> 0 && < 2048)

4.7 Audio Render Attribute

4.7.1 gm_audio_render_attr_t

- Definition

```
#define SYNC_LCD_DISABLE 0xFEFEFEFE

typedef struct  {
    gm_priv_t priv;
    int vch;
    gm_audio_encode_type_t encode_type;
    int block_size;
    int sync_with_lcd_vch;
    int reserved[4];
} gm_audio_render_attr_t;
```

- Description
  - Audio renderer attributes

- Members

Member	Description	Value
Priv	Library internal data, do not use it	None
vch	Number of the audio virtual channels	Input Integer
encode_type	Audio codec type GM_PCM/GM_AAC/GM_ADPCM	Input Please refer to gm_audio_encode_type_t.
block_size	Specify the size of one encoded (Or not encoded) audio frame, especially used for the audio bitstream without the header information The default block_size of the ADPCM bitstream encoded by GM_LIB is 256. PCM: 1 ~ 2048 AAC: 1024 ADPCM: 256 G711: 320	Input (> 0)
sync_with_lcd_vch	Specify the LCD channel index for the time sync SYNC_LCD_DISABLE/GM_LCD0/GM_LCD1…	Input (> 0 && < LCD_VCH_NUMBER)

Chapter 5

Appendix

This chapter contains the following section:

- 5.1 /proc and Debug

5.1 /proc and Debug

The purpose of this chapter is used to describe how to use "proc" for debugging and logging to system information. The subsections summarize all useful commands.

5.1.1 Video Graph View

The video graph was dynamically built according to the condition of the object binds and attribute settings. The purpose is for debugging the correctness of AP setting.

For example: (Slave console)

Entity name:
vcap0_135_0_1/vcap0_134_0_2/vcap0_133_0_3/3DI_0_0_0/scaler_0_0_0/lcd0_vg_0_0_0
buf/pool: Buffer management name
ratio(IN:OUT): Frame ratio for each entity

[root@GM]# cat /proc/videograph/gs/graph

```
================ jiffies:0xa3110 ==============
##### Graph [Disp_0:3] #####
[vcap0_135_0_1]: ratio(1:1)
    ->[3DI_0_0_0]: buf(disp_0) pool(disp0_in) ratio(1:1)
        ->[scaler_0_0_0]: buf(disp_0) pool(disp0_in) ratio(1:1)
            ->[lcd0_vg_0_0_0]: buf(disp_0) pool(disp0_in) ratio(30:30)
[vcap0_134_0_2]: ratio(1:1)
     >> [3DI_0_0_0]
[vcap0_133_0_3]: ratio(1:1)
     >> [3DI_0_0_0]
```

5.1.2 Grab Frame Buffer

According to the description of the video graph view, users can grab the frame buffer from each entity, such as capture/scaler/3di for debugging. Please follow this command flow.

i: Input buffer
o: Output buffer
b: Both input buffer and output buffer
Command: echo [entity name] i/o/b [path] > /proc/videograph/gs/dump

For example: (Slave console)
[root@GM]# echo vcap0_135_0_1 o /mnt/nfs > /proc/videograph/gs/dump

Then, users can see the file, "vcap0_135_0_1_outfix_bid277829_btype264_idx0_w1920_h1080.yuv", was stored in the folder of /mnt/nfs. This file is a kind of raw data file into yuv format.

5.1.3 Frame Rate Check for Playback

In the playback channel, Grain Media provides this command for debugging the real frame rate after applying this graph.

For example: (Slave console)
fd: Channel number defined by the driver
fps: Frame rate
max_cnt: Maximum buffer count
buf_nr: Available buffer count

```
[root@GM]# cat /proc/videograph/datain/fps
fd              fps    max_cnt    buf_nr
0x70030000      30     9              9
0x70040001      30     9              8
```

5.1.4 Skip Frame Check for Encode

In the encode channel, users can use this command to watch the frame skip status for debugging. “total” represents the total number of the frames from the typing time of the last command to now. “skips” represents the skip number of the frames between the periods.

For example: (Slave console)

Engine+Minor: The driver defined channel number

Skips: The skip number of the frames

Total: The total number of the frames from the typing time of the last command to now

```
[root@GM]# cat /proc/videograph/dataout/statistic

Engine  Minor          Skips     Total
00          00          00000    00000016
00          01          00000    00000016
```

5.1.5 Performance Statistic for Encode and Playback

In the encode or playback channel, users can use this command for observing the performance during the specific period set when the unit of the first argument is second.

For example: (Master console)

```
[root@GM]# echo 3 > /proc/videograph/vpd/perf
 bindfd(0x20004): 32 frames in 3081 ms
 bindfd(0x20005): 31 frames in 3000 ms
 bindfd(0x10014): 32 frames in 3081 ms
 bindfd(0x10015): 31 frames in 3001 ms
 bindfd(0x20001): 5 frames in 3958 ms
 bindfd(0x20003): 16 frames in 3006 ms
 bindfd(0x10013): 16 frames in 3007 ms
 bindfd(0x10012): 16 frames in 3024 ms
 bindfd(0x2000e): 90 frames in 3001 ms
```

5.1.6 LCD Information

The detailed LCD information for confirming the LCD status are extracted to useful information into gm_lcd_sys_info_t of gm_system_t gotten by calling the function of gm_get_sysinfo().

For example: (Master console)

[root@GM]#cat /proc/videograph/vpd/lcd

LCD Info:

vch: User channel number (lcd_vch)

id: The channel number defined by the driver (0: LCD0, 2: LCD2)

cip: Chip ID

fps: Frames per second

dup: Duplication vch (-1: No duplication, others: Duplicate vch)

input: Input resolution

output: Output resolution

```
-------------------------------------------------------
vch    id   cid   fps   dup   input          output
0      0    01    60    -1    1920x1080      1920x1080
1      2    01    30    0     720x480        720x480
```

5.1.7 Capture Information

It includes the basic characteristic OSD ability and scaling ability of capture. They extract useful information into gm_cap_sys_info_t of gm_system_t that are obtained by calling the function of gm_get_sysinfo().

For example: (Master console)

```
[root@GM]#cat /proc/videograph/vpd/cap
Cap Info:
vch: user channel number (cap_vch)
id: the driver defined channel number
   nrS: number of split
   nrP: number of path
   scM: scan method, 0->interlace, 1->progressive, 2->rgb888
vi: vi mode
fps: frames per second
cct: cascade count
mwt: mask win count
owt: font win count
mrt: mark win count
pXu: path X scaling-up ability, 1:Yes, 0:No
pXd: path X scaling-down ability, 1:Yes, 0:No
-------------------------------------------------------
vch    vcap_ch  cid  nrS   nrP   vi  fps  scM    resolution
0      7        01   4     2     2   30   0      704x480
1      6        01   4     2     2   30   0      704x480
2      5        01   4     2     2   30   0      704x480
3      4        01   4     2     2   30   0      704x480
4      3        01   4     2     2   30   0      704x480
5      2        01   4     2     2   30   0      704x480
……
vch    cct   mwt     owt     mrt   p0u   p1u   p2u   p3u   p0d   p1d   p2d   p3d
0      1     8       8       4     0     0     0     0     1     1     1     1
1      1     8       8       4     0     0     0     0     1     1     1     1
2      1     8       8       4     0     0     0     0     1     1     1     1
3      1     8       8       4     0     0     0     0     1     1     1     1
4      1     8       8       4     0     0     0     0     1     1     1     1
5      1     8       8       4     0     0     0     0     1     1     1     1
......
```

5.1.8 Set Debug Level and Dump Log

The function is used to enable the debug level to gain more detailed log information for debugging. To select the suitable level for logging file is the most important; otherwise, the debug log cannot be tracked due to the redundant information.

Level: 1 is suitable for all most cases.
Level: 0 is used to disable the log information.

```
<Set debug level>
For example: (Master console)
[root@GM]#echo 1 > /proc/videograph/gmlib/dbglevel
[root@GM]#echo 1 > /proc/videograph/vpd/dbglevel

For example: (Slave console)
[root@GM]#echo 1 > /proc/videograph/em/dbglevel
[root@GM]#echo 1 > /proc/videograph/ms/dbglevel
[root@GM]#echo 1 > /proc/videograph/gs/dbglevel
[root@GM]#echo 1 > /proc/videograph/datain/dbglevel
[root@GM]#echo 1 > /proc/videograph/dataout/dbglevel

<Dump log>
For example: (Master or Slave console)
[root@GM]# echo YOUR_FOLDER  >  /proc/videograph/dumplog
[root@GM]# cat /proc/videograph/dumplog
```

5.1.9 Print Parameter on Console When Attributions Updated by Application

This command is used to print the attribution on the console for debugging when the application updates the attributions.

For example: (Master console)

```
 [root@GM]#echo 1 > /proc/videograph/gmlib/dbglevel
```

Then, the master console would print the following information after executing application:

```
/samples # ./liveview_1div_to_4div
----------- graphidx(1) [START] -------
==[ATTR]: bindfd(0) State(BIND_INCOMPLETE)
  cap: cap_vch(0) path(0) enable_mv_data(0)
  win: visible(1) lcd_vch(0) rect(0 0 960 540)
==[LINK]
  CAP(0) ->   3DI(0) ->   SCL(0x30000000) -> DISP(0)
----------- graphidx(1) [START] -------
==[ATTR]: bindfd(1) State(BIND_INCOMPLETE)
  cap: cap_vch(1) path(0) enable_mv_data(0)
  win: visible(1) lcd_vch(0) rect(960 0 960 540)
==[LINK]
  CAP(0x10000) ->   3DI(0) ->   SCL(0x30000000) -> DISP(0)
----------- graphidx(1) [START] -------
==[ATTR]: bindfd(2) State(BIND_INCOMPLETE)
  cap: cap_vch(2) path(0) enable_mv_data(0)
  win: visible(1) lcd_vch(0) rect(0 540 960 540)
==[LINK]
  CAP(0x20000) ->   3DI(0) ->   SCL(0x30000000) -> DISP(0)
```

5.1.10 Get Audio Channel Number of HDMI

Users can use this command for getting the audio channel number of HDMI.

For example: (Master console)

vch: User channel number

ssp: Channel number defined by the driver

```
 [root@GM]#cat /proc/videograph/vpd/au_render
VCH#    SSP#   Description
--------------------------
   0         0          frond-end dec
   1         1          frond-end dec
   2         2          HDMI out
```

5.1.11 Mulit_win Record Function

Users can get multiple capture outputs to one H264 encoder, and then get the multi_win encoded file. To enable the multi_win record, users need to bind the capture objects to the window objects to indicate the capture output dimension and position. Therefore, users need to set win_to_enc of the window attribute, and then bind any window object to encode an object.

The following example will do four captures to one H264 record:

```
#define MAX_MULTIWIN_NUM    4
for (ch = 0; ch < MAX_MULTIWIN_NUM; ch++) {
        cap_attr.cap_vch = ch;
        cap_attr.path = 2;
        cap_attr.enable_mv_data = 0;
        gm_set_attr(sub_capture_object[ch], &cap_attr);

        win_attr.lcd_vch=0;
        win_attr.win_to_enc = 1;
        win_attr.rect.x = (ch % 2) * (MULTIWIN_ENC_WIDTH / 2);
        win_attr.rect.y = (ch / 2) * (MULTIWIN_ENC_HEIGHT / 2);
        win_attr.rect.width = MULTIWIN_ENC_WIDTH / 2;
        win_attr.rect.height = MULTIWIN_ENC_HEIGHT / 2;
        gm_set_attr(multiwin_object[ch], &win_attr);
        multiwin_bindfd[ch] = gm_bind(rec_groupfd, sub_capture_object[ch],
                                      multiwin_object[ch]);

        h264e_attr.ratectl.mode = GM_CBR;
        h264e_attr.ratectl.gop = 30;
        h264e_attr.dim.width = MULTIWIN_ENC_WIDTH;
        h264e_attr.dim.height = MULTIWIN_ENC_HEIGHT;
        h264e_attr.ratectl.bitrate = 512;
        h264e_attr.frame_info.framerate = 10;
        h264e_attr.b_frame_num = 0;
        h264e_attr.enable_mv_data = 0;
        gm_set_attr(encode_object[MAX_MAIN_NUM], &h264e_attr);
        sub_bindfd[ch] = gm_bind(rec_groupfd, multiwin_object[ch],
                                 encode_object[MAX_MAIN_NUM]);
    }
```

5.1.12 How to Setup LCD + Duplicated_CVBS + Spot Monitor

[Abbreviation]

- Duplicated_CVBS: D_CVBS
- CVBS only: CVBS
- LCD_A only: LCD_A
- LCD_B only: LCD_B
- Spot Monitor: S

Table 5-1. Display Feature Table

Method / Product	0	1	2	3	4
GM813x	CVBS	N/A	N/A	N/A	N/A
GM8210_DUAL_16HD_1080P	LCD_A CVBS (S)	N/A	N/A	N/A	LCD_A D_CVBS LCD_B (S)
GM8210_8HD_1080P	LCD_A CVBS (S)	N/A	N/A	N/A	LCD_A D_CVBS
GM8210_16CH_960H_1080P	N/A	N/A	LCD_A CVBS (S)	LCD_A D_CVBS	N/A
GM8210_32D1_1080P	N/A	N/A	LCD_A CVBS (S)	LCD_A D_CVBS	N/A
GM8287_4HD_1080P / GM8286_4HD_SXGA	LCD_A CVBS (S)	N/A	N/A	N/A	LCD_A D_CVBS
GM8287_16D1_1080P / GM8286_16CH_SXGA	N/A	N/A	LCD_A CVBS (S)	LCD_A D_CVBS	N/A
GM8282_16CH_960H_1080P	N/A	N/A	LCD_A CVBS (S)	LCD_A D_CVBS	N/A
GM8283_8CH_960H_1080P	N/A	N/A	LCD_A CVBS (S)	LCD_A D_CVBS	N/A

How to configure vg_boot.sh / gmlib.cfg

1. To set CVBS only

   [vg_boot.sh]

   ```
   #output_type=0 (NTSC), output_type=1 (PAL)
   /sbin/insmod /lib/modules/flcd200-pip2.ko input_res=0 output_type=0 bf=0 d1_3frame=1
                                                    fb0_fb1_share=1 lcd_disable=0
   ```

   [gmlib.cfg]

   ```
   #Lcd_vch: GM_LCD0 is GROUP0,  GM_LCD1 is GROUP1 and so on.
   #The first inserted lcd or cvbs module is VCH(0). The second one is VCH(1) and so on.
   #VCH(0):CVBS
   GROUP0 = 0
   METHOD0 = 0
   ```

2. To set LCD only

   [vg_boot.sh]

   ```
   #output_type=51(1920x1080)
   /sbin/insmod /lib/modules/flcd200-pip.ko input_res=15 output_type=51 bf=0
   ```

   fb0_fb1_share=1 [gmlib.cfg]

   ```
   #Lcd_vch: GM_LCD0 is GROUP0, GM_LCD1 is GROUP1, and so on.
   #The first inserted lcd or cvbs module is VCH(0). The second one is VCH(1) and so on.
   #VCH(0):LCD
   GROUP0 = 0
   METHOD0 = 0
   ```

3. To set LCD_A + LCD_B (Spot_Monitor)

   [vg_boot.sh]

   ```
   #output_type=51(HD), output_type=15(VGA)
   /sbin/insmod /lib/modules/flcd200-pip.ko input_res=15 output_type=51 bf=0
                                                    fb0_fb1_share=1
   /sbin/insmod /lib/modules/flcd200-pip1.ko input_res=15 output_type=11 bf=0
                                                    fb0_fb1_share=1
   ```

   [gmlib.cfg]

   ```
   #Lcd_vch: GM_LCD0 is GROUP0, GM_LCD1 is GROUP1, and so on.
   #The first inserted lcd or cvbs module is VCH(0). The second one is VCH(1) and so on.
   ```

#VCH(0):LCD0, VCH(1): LCD1
GROUP0 = 0
METHOD0 = 4
GROUP1 = 1
METHOD1 = 4

4. To set LCD_A + CVBS (Spot_Monitor)

[vg_boot.sh]

#output_type=0(NTSC), output_type=1(PAL), output_type=51(1920x1080)
/sbin/insmod /lib/modules/flcd200-pip.ko input_res=15 output_type=51 bf=0 fb0_fb1_share=1
/sbin/insmod /lib/modules/flcd200-pip2.ko input_res=0 output_type=0 bf=0 d1_3frame=1 fb0_fb1_share=1 lcd_disable=0

[gmlib.cfg]

#Lcd_vch: GM_LCD0 is GROUP0, GM_LCD1 is GROUP1, and so on.
#The first inserted lcd or cvbs module is VCH(0). The second one is VCH(1) and so on.
#VCH(0):LCD, VCH(1): CVBS
GROUP0 = 0
METHOD0 = 2
GROUP1 = 1
METHOD1 = 2

5. To set LCD_A + Duplicated_CVBS

[vg_boot.sh]

#output_type=0(NTSC), output_type=1(PAL), output_type=51(1920x1080)
/sbin/insmod /lib/modules/flcd200-pip.ko input_res=15 output_type=51 bf=0 fb0_fb1_share=1
/sbin/insmod /lib/modules/flcd200-pip2.ko input_res=0 output_type=0 bf=0 d1_3frame=1 fb0_fb1_share=1 lcd_disable=0

[gmlib.cfg]

#Lcd_vch: GM_LCD0 is GROUP0, GM_LCD1 is GROUP1, and so on.
#The first inserted lcd or cvbs module is VCH(0). The second one is VCH(1) and so on.

#VCH(0):LCD, VCH(1): CVBS

GROUP0 = 0, 1

METHOD0 = 3

6. To set LCD_A + Duplicated_CVBS + LCD_B (Spot_Monitor)

[vg_boot.sh]

#output_type=0(NTSC), output_type=1(PAL), output_type=51(1920x1080), output_type=11(VGA)

/sbin/insmod /lib/modules/flcd200-pip.ko input_res=15 output_type=51 bf=0 fb0_fb1_share=1

/sbin/insmod /lib/modules/flcd200-pip1.ko input_res=15 output_type=11 bf=0 fb0_fb1_share=1

/sbin/insmod /lib/modules/flcd200-pip2.ko input_res=0 output_type=0 bf=0 d1_3frame=1 fb0_fb1_share=1 lcd_disable=0

[gmlib.cfg]

#Lcd_vch: GM_LCD0 is GROUP0, GM_LCD1 is GROUP1, and so on.

#The first inserted lcd or cvbs module is VCH(0). The second one is VCH(1) and so on.

#VCH(0):LCD0, VCH(1):LCD1, VCH(2):CVBS

GROUP0 = 0, 2

METHOD0 = 4

GROUP1=1

METHOD1=4

For example:

To set CVBS duplicated LCD1 and use LCD2 as the spot monitor (Item 6)

A. Modify liveview/playback setup as follows:

```
int ch;
DECLARE_ATTR(win_attr, gm_win_attr_t);
DECLARE_ATTR(cap_attr, gm_cap_attr_t);

      cap_attr.cap_vch = 0;//use capture 0 as input
      cap_attr.path = 0;//use path 0
      cap_attr.enable_mv_data = 0;
      gm_set_attr(liveview_capture_object[0], &cap_attr);
      win_attr.lcd_vch = GM_LCD0; //Use LCD0 and duplicate to CVBS
      win_attr.rect.x = 0;
```

```
win_attr.rect.y = 0;
win_attr.rect.width = gm_system.lcd[ch].dim.width;
win_attr.rect.height = gm_system.lcd[ch].dim.height;
gm_set_attr(win_object[0], &win_attr);
liveview_bindfd = gm_bind(display_groupfd[0], liveview_capture_object[0],
                          win_object[0]);
gm_apply(display_groupfd[0]);

cap_attr.cap_vch = 1; //use capture 1 as input
cap_attr.path = 0;//use path 0
cap_attr.enable_mv_data = 0;
gm_set_attr(liveview_capture_object[1], &cap_attr);
win_attr.lcd_vch = GM_LCD1; //Use LCD2 as spot monitor
win_attr.rect.x = 0;
win_attr.rect.y = 0;
win_attr.rect.width = gm_system.lcd[ch].dim.width ;
win_attr.rect.height = gm_system.lcd[ch].dim.height;
gm_set_attr(win_object[1], &win_attr);
liveview_bindfd_spot = gm_bind(display_groupfd[1], liveview_capture_object[1],
                               win_object[1]);
gm_apply(display_groupfd[1]);
```

To set LCD1 and use CVBS as the spot monitor (Item 4.)

A. Modify liveview/playback setup as follows:

```
int ch;
DECLARE_ATTR(win_attr, gm_win_attr_t);
DECLARE_ATTR(cap_attr, gm_cap_attr_t);

cap_attr.cap_vch = 0; //use capture 0 as input
cap_attr.path = 0;
cap_attr.enable_mv_data = 0;
gm_set_attr(liveview_capture_object[0], &cap_attr);
win_attr.lcd_vch = GM_LCD0; //Use LCD0
win_attr.rect.x = 0;
win_attr.rect.y = 0;
win_attr.rect.width = gm_system.lcd[ch].dim.width;
win_attr.rect.height = gm_system.lcd[ch].dim.height;
gm_set_attr(win_object[0], &win_attr);
liveview_bindfd = gm_bind(display_groupfd[0], liveview_capture_object[0],
                          win_object[0]);
gm_apply(display_groupfd[0]);

cap_attr.cap_vch = 1; //use capture 1 as input
cap_attr.path = 0;
cap_attr.enable_mv_data = 0;
gm_set_attr(liveview_capture_object[1], &cap_attr);
win_attr.lcd_vch = GM_LCD1; //Use CVBS as spot monitor
```

```
win_attr.rect.x = 0;
win_attr.rect.y = 0;
win_attr.rect.width = gm_system.lcd[ch].dim.width ;
win_attr.rect.height = gm_system.lcd[ch].dim.height;
gm_set_attr(win_object[1], &win_attr);
liveview_bindfd_spot = gm_bind(display_groupfd[1], liveview_capture_object[1],
                               win_object[1]);
gm_apply(display_groupfd[1]);
```

5.1.13 OSD Font Number Assignment

Users can use this inserted parameter for OSD font number assignment per window. The total font number among windows per channel is the same.

For example:

```
/sbin/insmod /lib/modules/vpd_slave.ko font_num=31,30,29,28,0,0,0,0
win_0 max 31 words
win_1 max 30 words
win_2 max 29 words
win_3 max 28 words
win_4 max  0 words
win_5 max  0 words
win_6 max  0 words
win_7 max  0 words
```

5.1.14 gmlib_setting Usage Guide

Print the last setting of gmlib on the master console for user debugging.

The information includes the parameter of the bind_fd/input object/output object/attributes.

For example:

```
cat  /proc/videograph/gmlib_setting
```

5.1.15 gmlib_flow Usage Guide

Print the recent setting flow of gmlib on the master console for user debugging.

The information includes the recent called parameter of gmlib and attributes.

For example:

```
cat  /proc/videograph/gmlib_flow
```

5.1.16 OSG Mark User Guide

OSG Mark is an alternative one of osd_font with 2-color (Black/White) font and more font database stored on ddr ram. It was implemented in HW or SW according to the product. Below are the descriptions of the detailed specification.

[Definition]

1. Osg_mark_idx

   image index or name

2. Mark_idx

   It is window index per channel. Window is a management unit for specific x,y,align…, but the content on this widow must be specified by osg_mark_idx.

[Usage Flow]

1. Load image

   API:

   int gm_set_osd_mark_image(gm_osd_mark_img_table_t *osd_mark_img);

   structure:

   osd_mark_img.mark_img[0].osg_mark_idx <= Use this parameter to set the name or index of the image

2. Show image

   API:

   int gm_set_osd_mark(void *obj, gm_osd_mark_t *osd_mark);

   structure:

   osd_mark.mark_idx <= window index

   osd_mark.osg_mark_idx <= Use this parameter to decide which image (Content) would be shown on the specific window.

[OSG limits]

1. 64 windows (mark_idx) for each path of one channel (The recommendation is 4 window.)
2. The max osg_mark_idx is 64*(Channel number)*4
3. The width and height cannot be smaller than 6. It cannot be bigger than the capture output size.
4. Width must be multiples of 2.

[OSG Hardware / Software supported guide]

<GM8210>

1. OSG_SW

*Build command of sw_osg.ko: make //no extra parameter

*Module inserted flow: Insert sw_osg.ko at slave site. //only sw_osg.ko

*Notification:

a.ft2dge.ko is only for GUI at master site.

b. sw_osg.ko must be built with GM8210 Kernel environment.

2. OSG_HW

*Build command of sw_osg.ko: make 8210_hw=y

*Module inserted flow:

a. Cancel the ft2dge.ko inserted at master site.

b. Insert ft2dge.ko before sw_osg.ko at slave site.

*Notification:

a. ft2dge.ko is only supported for GUI or OSG. Both usages at the same time are not allowed.

b. sw_osg.ko must be built with GM8210 kernel environment.

<GM8287>

1. OSG_SW (Don't support)

2. OSG_HW

*Build command of sw_osg.ko: make //no extra parameter

*Module inserted flow:

a. Insert think2d.ko before sw_osg.ko.

*Notification:

a.Think2d.ko is only supported for GUI or OSG. Both usages at the same time are not allowed.

b. sw_osg.ko must be built with GM8287 kernel environment.

<GM8139>

1. OSG_SW

*Build command of sw_osg.ko: make //no extra parameter

*Module inserted flow:

a. Insert sw_osg.ko. //only sw_osg.ko.

*Notification:

a. sw_osg.ko must be built with GM8139 kernel environment.

2. OSG_HW (Don't support)

[How to use static memory allocated of osg]

1. Inserted parameter for swosg

   If users need 128 osg_mark_idx, users need to add plus 4 (128+4 = 132). The total memories need to be assigned, and the unit is Kbyte (e.g. 8192 is equal to 8Mbytes).

   For example: Insert sw_osg.ko idn=132 mem=8192

2. After swosg module is successfully inserted, users could set the static memory size for certain osg_mark_idx by "echo" command.

For example:

Please use "echo 4 64 > /proc/swosg/canvas_static_info " // to set osg_mark_idx 4 as 64KByte

Please use "echo 5 128 > /proc/swosg/canvas_static_info " // to set osg_mark_idx 5 as 128KByte

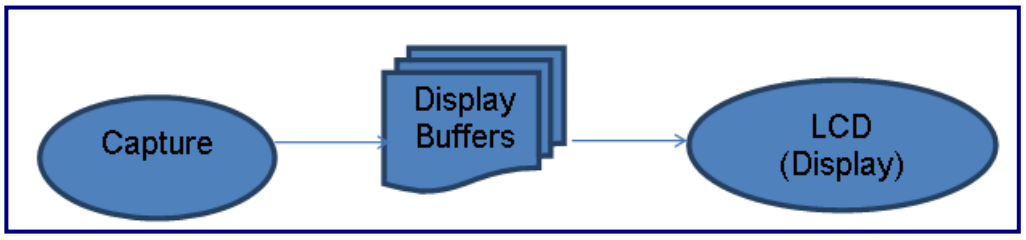
<Warning>

If users have to set osg_mark_idx 4 as 64kByte once time, it cannot be modified again by any value, such as "echo 4 32 > /proc/swosg/canvas_static_info"

5.1.17 Definition of gmlib.cfg

5.1.17.1 Buffer Definition

5.1.17.1.1 Liveview Display Buffers



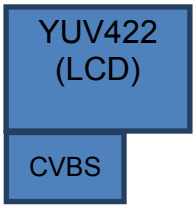
Display buffer count is specified by section [BUFFER_DISPLAY] of gmlib.cfg.

The display buffer format is indicated by [DISPLAY_GROUP], such as:

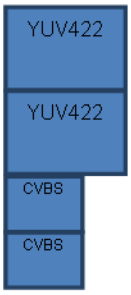
FRAME (Display 30p):

2FRAME (Display 60p):

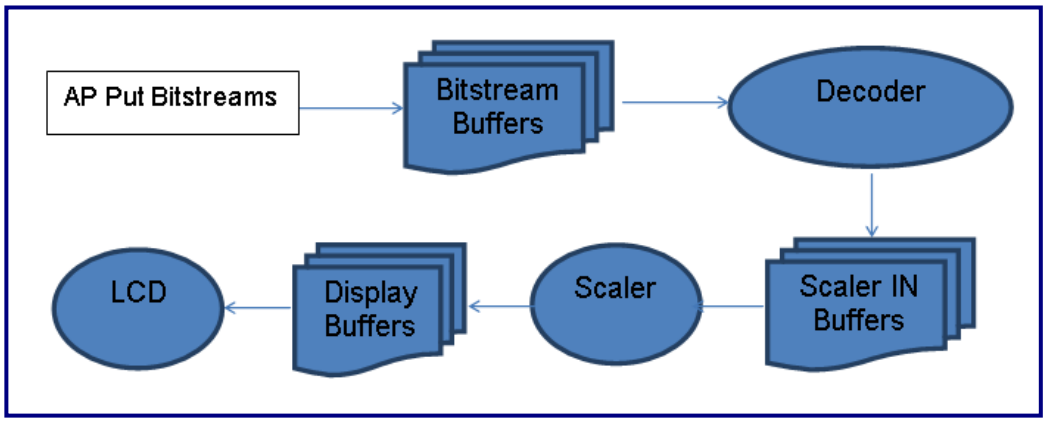
FRAME_FRAME (Display 30p with CVBS):



2FRAME_2FRAME (Display 60p with CVBS):



5.1.17.1.2 Playback Buffers

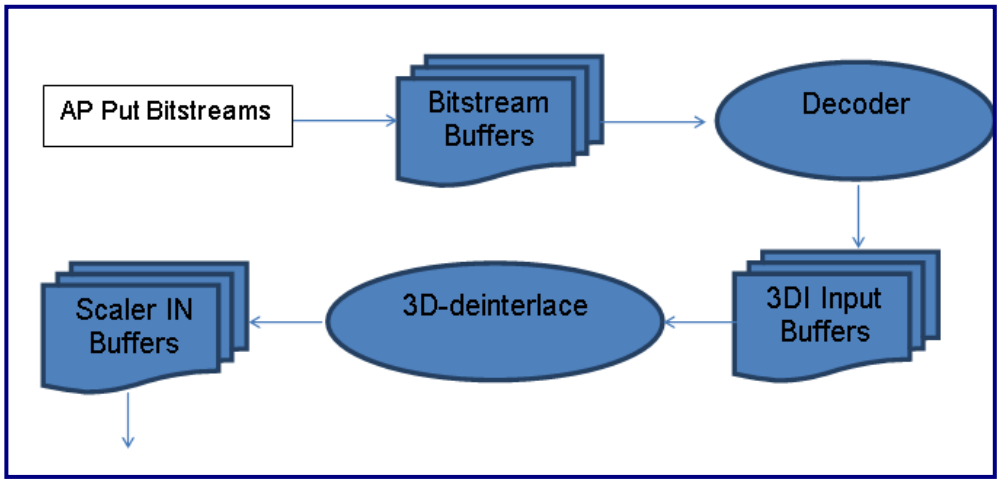


Bitstream buffers are indicated in [PLAYBACK] and [BUFFER_BITSTREAM] sections of gmlig.cfg

Scaler IN buffers are indicated in [PLAYBACK] and [BUFFER_DISPLAY] sections of gmlib.cfg

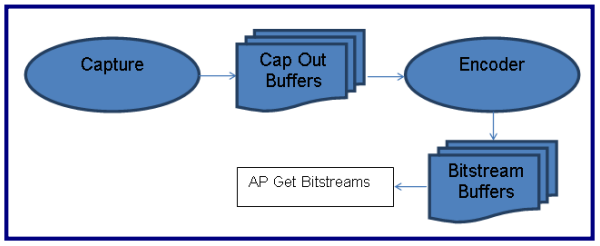
5.1.17.1.3 Playback with 3D-deinterlace

Under the usage of fielding coding feature, bitstream is decoded by the field interlaced frame, it may need to do 3D-deinterlace before scaling.



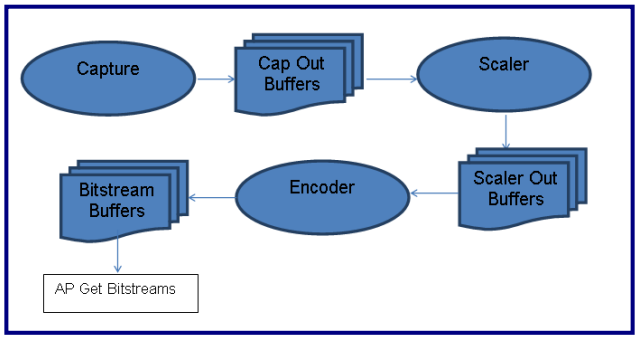
Decode 3DI input buffers are indicated in [PLAYBACK_3DI] and [BUFFER_DISPLAY] sections of gmlig.cfg

5.1.17.1.4 Mainstream Encode Buffers



Capture output buffers are indicated in [ENCODE_CAPTURE] and [BUFFER_CAPTURE] sections of gmlib.cfg
Encode output bitstream buffers are indicated by [ENCODE_STREAMS] and [BUFFER_BITSTREAM] sections of gmlig.cfg

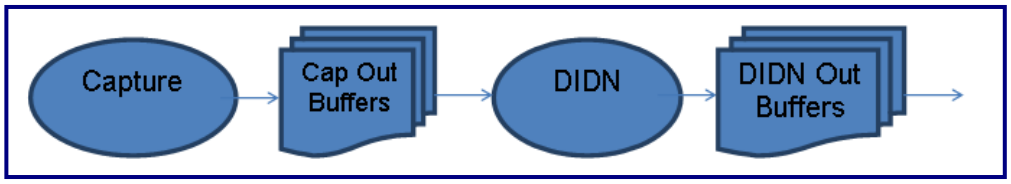
5.1.17.1.5 Scaler Substream Encode Buffers



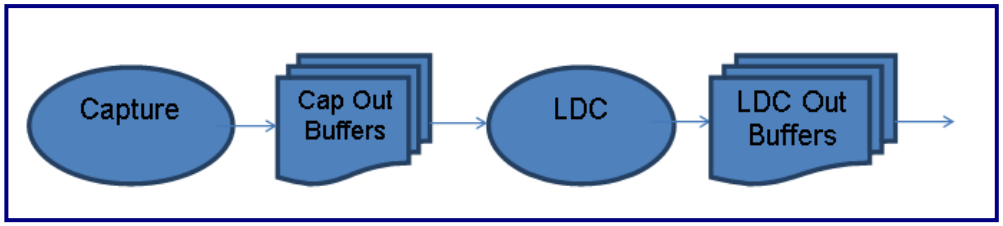
Scaler output buffers are indicated in [ENCODE_SCALER] and [BUFFER_SUBSTREAM] sections of gmlib.cfg

5.1.17.1.6 DIDN & LDC output

Encode DIDN only supports in GM8139, GM8138 series.
LDC feature is not provided under the current IC capability.



DIDN output buffers are indicated in [ENCODE_DIDN] and [BUFFER_ENCODE_DIDN] sections of gmlib.cfg



LDC output buffers are indicated in [LDC_OUT] and [BUFFER_LDC] sections of gmlib.cfg

5.1.17.2 RAW Output Buffer

RAW output buffers are indicated by [RAW_OUT] and [BUFFER_RAW] sections of gmlib.cfg.

5.1.17.3 Description of gmlib.cfg

```
================================================================
====
; Product Configuration

[VERSION]
VERSION = 0.1 ➔ indicates the version number of config, and this is v0.1

[GRAPH_TYPE]
; graph type: 1(DVR with display 3DI), 2(IPCAM)
graph_type = 1 ➔ Indicates the graph type mode about 1(DVR) and 2(IPCAM)

[RESOLUTION] ➔ Setups YUV resolution infomation
CONFIG1 = 2M/2048/960 ➔ Indicates keyword "2M" resolution with 2048x960 value

[DISPLAY_GROUP]
; GROUP0 = lcd_vch0, lcd_vch1
; Display group method:
;   0(FRAME), 2(2FRAME), 3(2FRAME_2FRAME), 4(FRAME_FRAME)
GROUP0 = 0, 1 ➔ Means display 0, 1 and it displays the same content by method 3
METHOD0 = 3 ➔ method 3 is (2frames + 2frames buffer mode)
```

[BITRATE]

min_compressed_ratio = 20 ➔ The minimal compressed rate is 20%, and it means compressed from 0%~20% that is used to calculate bitstream buffer.

max_bitrate = 1080P/9216K, D1/4096K ➔ Means 1080P with max. 9216Kbps, which used to indicate the max. bitrate to calculate all bitstream buffers for different resolutions.

[PLAYBACK] ➔ Playback section, indicates the playback specification

; CONFIG1 = resolution_keywords/channels/total_fps/ddr_channel,...

; default resolution keywords: 8M, 5M, 4M, 3M, 2M, 1.3M, 1M, nHD, 1080P, 720P, D1, VGA, CIF, QCIF

CONFIG1 = D1/16/480/0 ➔ Means D1 16 channels with 480fps under DDR0 specification to do playback

CONFIG2 = 960H/16/240/0 ➔ Means 960H 16 channels with 240fp under DDR0 specification to do playback. System will select the max. requirement buffer size from CONFIG1 or CONFIG2 automatically.

[PLAYBACK_3DI]

; CONFIG1 = resolution_keywords/channels/total_fps/ddr_channel,...

[ENCODE_STREAMS] ➔ Indicates the encode bitstream out specification

; CONFIG1 = resolution_keywords/channels/total_fps/ddr_channel,...

CONFIG1 = D1/16/480/0, CIF/16/480/0 ➔ CONFIG1 is D1/16channels/480fps/DDR0 + CIF/16channels/480fps/DDR0 for encode streams

[ENCODE_CAPTURE] ➔ Indicates the capture output buffer specification

; CONFIG1 = resolution_keywords/channels/total_fps/ddr_channel,...

CONFIG1 = D1/16/480/0, CIF/16/480/0 ➔ CONFIG 1 is D1/16channels/DDR0 + CIF/16channels/DDR0 to allocate capture output buffers

[ENCODE_SCALER] ➔ Scaler substeram, it indicates the scaler output buffer specification

; CONFIG1 = resolution keywords = channels/total_fps/ddr_channel,...

CONFIG1 = CIF/16/480/0 ➔ CONFIG1 is CIF/16channels/480fps/DDR0 for encode substream by scaling, and this indicates the scaler output buffer specification.

[ENCODE_DIDN] ➔ Indicates the encode DIDN out buffer specification

; CONFIG1 = resolution_keywords/channels/total_fps/ddr_channel,...

CONFIG1 = CIF/16/480/0 ➔ CONFIG1 is CIF/16channels/480fps/DDR0 for encode substream by DIDN.

[RAW_OUT] ➔ Indicates RAW(YUV422) out buffer specification

; CONFIG1 = channels/total_fps/ddr_channel,...

CONFIG1 = CIF/1/30/0 ➔CONFIG1 is CIF/1channel/30fps/DDR0 for the raw out buffer setting

[LDC_OUT] ➔ Lens distortion correction (LDC) output buffer specification

; CONFIG1 = channels/total_fps/ddr_channel,...

CONFIG1 = CIF/1/30/0 ➔CONFIG1 is CIF/1channel/30fps/DDR0 for the raw out buffer setting

[AUDIO]

; CONFIG1 = audio_keyword/channels/sample_rate/stereo/bits/ddr_vch

CONFIG1 = au_grab/32/16K/mono/16/0, au_render/4/16K/mono/16/0 ➔ Indicates the audio grab 32channels/16K/mono/16bits/DDR0 specification + audio render 4channels/16K/mono/16bits/DDR0 specification

max_frame_samples = 2048 ➔ Maximum frame is 2048 samples.

grab_buffer_time = 768 ➔ Indicates that the audio grab buffer latency time is 768ms.

render_buffer_time = 768 ➔ Indicates that audio render buffer time is 768ms.

[BUFFER_DISPLAY] ➔ Display buffer configuration

; Display count lcd_vch/count/ddr_channel, lcd_vch/count/ddr_channel,...

display_count = 0/9/0 ➔ LCD vch0 with 9 buffers (DDR0)

max_resolution = 0/1920/1080 ➔ Indicates LCD vch0 with max resolution 1920x1080 (please note bootup LCD resolution may be setup to 1280x720 or others)

; Playback 3DI input buffer count per channel

playback_3di_input_count = 3 ➔ Indicates the playback 3di input buffer count per channels (Field coding encode mode)

; Playback scaler input buffer count per channel

playback_scaler_input_count = 3 ➔ Indicates the playback scaler input buffer count per channels

[BUFFER_BITSTREAM] ➔ Indicates the buffer bitstream buffer time

; Playback bitstream data buffer: time/ddr_vch

datain_buffer_time = 198/0 ➔ Playback data input buffer time 198ms/DDR0. If 33ms display rate, it means 6 bitstream buffers.

; Record bitstream data buffer: time/ddr_vch

dataout_buffer_time = 330/0 ➔ Encode data output buffer time 330ms/DDR0. If 33ms display rate, it means 10 bitstream buffers.

[BUFFER_CAPTURE] ➔ Indicates the capture output buffer configuration

; Encode capture buffer three slop: fps/count, fps/count, fps/count

CONFIG1 = D1/30/6, D1/1/2 ➔ Indicates that capture D1 30fps uses 6 buffer counts, 1fps uses 2 buffer counts

CONFIG2 = CIF/30/6, CIF/1/2 ➔ Indicates that capture CIF 30fps uses 6 buffers, 1fps uses 2 buffers

capture_max_fps = 30 ➔ Indicates the max. frame rate of the capture input

[BUFFER_ENCODE_DIDN] ➔ Encodes DIDN out buffer configuration (GM813x only)

; Encode substream scaler out buffer time

encode_didn_out_buffer_count = 0 ➔Indicates the encode didn output buffer count per channel

[BUFFER_SUBSTREAM] ➔ Scaler substream buffer configuration

; Encode substream scaler out buffer count

encode_scaler_out_buffer_count = 3 ➔ Indicates the encode substream scaler output buffer count per channel

[BUFFER_RAW]

raw_out_buffer_count = 2 ➔ Indicates that the raw out buffer count is 2 per channels.

[BUFFER_LDC]

ldc_out_buffer_count = 2 ➔ LDC (Lens Distortion Correction) buffer count per channel.

[ENCODE_MOTION] ➔ Indicates the encode motion info buffer specification

; Enable motion data under specified resolution by "motion_resolution = D1, CIF..."

; motion_resolution = CIF ➔ Indicates the resolution needs to output motion info data

[SNAPSHOT]

yuv_max_width = 960 ➔ Indicates the maximum width of the YUV buffer

yuv_max_height = 540 ➔ Indicates the maximum height of the YUV buffer

max_bitstream_size = 1048544 ➔ Indicates the maximum buffer size of the snapshot bitstream

[RESERVED_BUFFER]

; Reserved buffer ratio for seamless switching usage, such as 4CH 720P->4CH 1080P without any stop

reserved_ratio = 10 ➔ Indicates 10% reserved buffer for switching to make it seamless, such as playback switch or encode switch without stop