



HiMPP IPC Media Processing Software

## **FAQs**

**Issue**            **05**

**Date**            **2016-10-28**

**Copyright © HiSilicon Technologies Co., Ltd. 2015-2016. All rights reserved.**

No part of this document may be reproduced or transmitted in any form or by any means without prior written consent of HiSilicon Technologies Co., Ltd.

## **Trademarks and Permissions**



**HISILICON**, and other HiSilicon icons are trademarks of HiSilicon Technologies Co., Ltd.

All other trademarks and trade names mentioned in this document are the property of their respective holders.

## **Notice**

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

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

## **HiSilicon Technologies Co., Ltd.**

Address: Huawei Industrial Base  
Bantian, Longgang  
Shenzhen 518129  
People's Republic of China

Website: <http://www.hisilicon.com>

Email: [support@hisilicon.com](mailto:support@hisilicon.com)



# About This Document

## Purpose

This document describes the solutions to the problems that may occur when you use the HiSilicon media processing platform (HiMPP).



### NOTE

This document uses the Hi3516A as an example. Unless otherwise specified, the descriptions of the Hi3516A also apply to the Hi3516D.

Unless otherwise specified, the descriptions of Hi3518E V200 also apply to Hi3518E V201 and Hi3516C V200.

## Related Versions

The following table lists the product versions related to this document.

Product Name	Version
Hi3516A	V100
Hi3516D	V100
Hi3518E	V200
Hi3518E	V201
Hi3516C	V200
Hi3519	V100
Hi3519	V101
Hi3516C	V300

## Intended Audience

This document is intended for:






- Technical support engineers
- Software development engineers



## Conventions

### Symbol Conventions

The symbols that may be found in this document are defined as follows.

Symbol	Description
 <b>DANGER</b>	Indicates a hazard with a high level of risk which, if not avoided, will result in death or serious injury.
 <b>WARNING</b>	Indicates a hazard with a medium or low level of risk that, if not avoided, could result in minor or moderate injury.
 <b>CAUTION</b>	Indicates a potentially hazardous situation, which if not avoided, could result in equipment damage, data loss, performance degradation, or unexpected results.
 <b>TIP</b>	Indicates a tip that may help you solve a problem or save time.
 <b>NOTE</b>	Provides additional information to emphasize or supplement important points of the main text.

### General Conventions

The general conventions that may be found in this document are defined as follows.

Convention	Description
Times New Roman	Normal paragraphs are in Times New Roman.
<b>Boldface</b>	Names of files, directories, folders, and users are in <b>boldface</b> . For example, log in as user <b>root</b> .
<i>Italic</i>	Book titles are in <i>italics</i> .
Courier New	Examples of information displayed on the screen are in Courier New.

## Change History

Changes between document issues are cumulative. Therefore, the latest document issue contains all changes made in previous issues.

### Issue 05 (2016-10-28)

This issue is the fifth official release, which incorporates the following changes:



## **Chapter 1 System Control**

Section 1.2 is modified.

Section 1.3.2 and section 1.3.4 are added.

## **Chapter 2 MIPI**

Section 2.1.3 is added.

Chapter 4, chapter 6, and chapter 7 are added.

## **Issue 04 (2016-02-15)**

This issue is the fourth official release, which incorporates the following changes:

Section 4.4 is added.

## **Issue 03 (2015-09-14)**

This issue is the third official release, which incorporates the following changes:

Section 4.3 and chapter 5 are added.

## **Issue 02 (2015-06-16)**

This issue is the second official release, which incorporates the following changes:

Chapter 4 is added.

## **Issue 01 (2015-02-10)**

This issue is the first official release.



# Contents

<b>About This Document.....</b>	<b>i</b>
<b>1 System Control .....</b>	<b>1</b>
1.1 Log Information .....	1
1.1.1 How Do I View HiMPP Logs? .....	1
1.2 Memory Usage .....	2
1.2.1 How Do I Adjust the Memories Occupied by Media Services? .....	2
1.3 Performance .....	5
1.3.1 What Do I Do If the VGS LDC Performance Does Not Meet Requirements? .....	5
1.3.2 What Do I Do If Image Transition Occurs When the Correction Ratio Is Configured Continuously in Fullall Mode for the VGS LDC? .....	6
1.3.3 What Do I Do If the CPU Usage Statistics Obtained by Running the Top Command Fluctuates? .....	6
1.3.4 How Do I Configure the Clock Frequency of Each Module? .....	6
1.4 Tailoring .....	7
1.4.1 How Do I Reduce the Application Size When the Static Libraries Are Used? .....	7
<b>2 MIPI .....</b>	<b>8</b>
2.1 MIPI Configuration .....	8
2.1.1 How Do I Configure lane_id? .....	8
2.1.2 How Do I Configure the Mode of sync_code? .....	9
2.1.3 What Do I Do If Exceptions Occur in the 2-Link Scenario of the LVDS Mode? .....	13
2.2 MIPI Frequency .....	14
2.2.1 How Do I Query the Relationship Between the Transfer Frequency of MIPI Lanes and VI Processing Frequency? .....	14
2.3 Sensor Reset .....	14
2.3.1 How Do I Change the Hi3516A Sensor Reset Pin? .....	14
<b>3 VI .....</b>	<b>16</b>
3.1 DIS .....	16
3.1.1 How Do I Use the DIS Function of the Hi3516A in the 5-Megapixel Scenario? .....	16
<b>4 Fisheye .....</b>	<b>18</b>
4.1 Fisheye Correction .....	18
4.1.1 How Do I Implement Fisheye Correction of More than Four Cells by Using Hi3519 V100/ Hi3519 V101? .....	18



<b>5 Audio .....</b>	<b>19</b>
5.1 How Do I Play the Audio Streams Encoded by HiSilicon on the PC? .....	19
5.1.1 How Do I Play G711/G726/ADPCM Audio Streams Encoded by HiSilicon on the PC? .....	19
5.2 How Do I Play Standard Audio Streams on HiSilicon Chips? .....	21
5.2.1 How Do I Play Standard G711/G726/ADPCM Audio Streams on HiSilicon Chips? .....	21
5.3 What Do I Do If High-Frequency Information Is Lost After VQE Is Enabled? .....	23
5.3.1 What Do I Do If High-Frequency Information Is Lost After VQE Is Enabled? .....	23
5.4 What Do I Do If Abnormal Amplitude Frequency Responses Occur During the Embedded Audio CODEC Output (AO Output) .....	24
<b>6 Low Power Consumption .....</b>	<b>26</b>
6.1 What Do I Do If the Frequency Is Frequently Modulated During Dynamic Frequency Modulation of the Low-Power Module? .....	26
<b>7 LCD Debugging .....</b>	<b>27</b>
7.1 Supported LCDs .....	27
7.2 LCD Debugging Sequence .....	27
7.2.1 Confirming Pin Multiplexing Configurations .....	27
7.2.2 Confirming User Timings .....	27
7.2.3 Confirming CRG Configurations .....	29
7.2.4 Confirming the Configuration of the LCD_CTRL Register .....	35
<b>8 Others .....</b>	<b>37</b>
8.1 Dynamic Library .....	37
8.1.1 What Do I Do If the Dynamic Libraries Cannot Be Used When the Application Is Statically Compiled? .....	37
8.1.2 What Do I Do If a Redefinition Error Occurs When libupvqe.a and libdnvqe.a Are Used for Dynamic Compilation? .....	37



# 1 System Control

## 1.1 Log Information

### 1.1.1 How Do I View HiMPP Logs?

[Symptom]

How do I view HiMPP logs and change the log level?

[Cause Analysis]

The logs record the error causes, error locations, and system running status during the running of the software development kit (SDK). Logs can help you locate errors.

Currently, the logs are classified into seven levels, and the default level is level 3. A higher log level indicates that more information is recorded. When the level is set to level 7, the information about the running status of the entire system is recorded in logs in real time. The mass information, however, significantly reduces the overall performance of the system. Typically, you are advised to set the log level to level 3. In this case, information is recorded in logs only when errors occur and most errors can be located.

[Solution]

You can run the following commands to obtain logs, and view or change the log level:

- To view the log level of each module, run the **cat /proc/umap/logmpp** command. Then, the log levels of all the modules are listed.
- To change the log level of a module, run the **echo "venc=4" > /proc/umap/logmpp** command. In this command, **venc** is a module name. This name must be the same as that displayed after the **cat** command is executed.
- To change the log levels of all the modules, run the **echo "all=4" > /proc/umap/logmpp** command.
- To obtain logs, run the **cat /dev/logmpp** command. Then, all the log information is displayed. If all the log information is read, the command is blocked until new log information is recorded. Press **Ctrl+C** to exit. To use the device node in **/dev/logmpp**, run **open** and **read** commands.





## 1.2 Memory Usage

### 1.2.1 How Do I Adjust the Memories Occupied by Media Services?

#### [Symptom]

Media services require memories for normal running. The memories mainly indicate the media memory zone (MMZ). The HiMPP allocates memories based on services. When the memories are insufficient, you can adjust the allocated memories.

#### [Cause Analysis]

The HiSilicon SDK allows you to adjust the allocated memories when the memories are insufficient. This section briefly describes the measures to minimize memory usage. For details, see related documents.

#### [Solution]

- Check the operating system (OS) memory and MMZ memory.

For details, see chapter 6 "Allocating and Using the Address Space" in the *Description of the Installation and Upgrade of the Hi35xx SDK* under **SDK\01.software\board\documents\_cn\**.

- Adjust the memories occupied by SDK services.

- Entire system

Ensure that the size of a D1-series (D1, 2CIF, and CIF) picture is an integral multiple of the sizes of the other D1-series pictures. For example, the size of a D1 picture is 704 x 576, the size of a 2CIF picture is 352 x 576, and the size of a CIF picture is 352 x 288. If the size of a 2CIF picture is 352 x 576 and the size of a CIF picture is 360 x 288, the picture sizes do not meet requirements. In addition, if the size of the picture captured by a VI channel is 720 x 576 and the size of the picture encoded by a VENC channel is 704 x 576, the picture sizes do not meet requirements.

- Minimum buffer size for each module

- For the Linux version, see the *HiMPP IPC V2.0 Media Processing Software Development Reference* (applicable to the Hi3516A/Hi3518E V20X/Hi3519 V100), and the *HiMPP IPC V3.0 Media Processing Software Development Reference* (applicable to Hi3519 V101/Hi3516C V300).

- For the Huawei LiteOS version, see the *HiMPP IPC V3.0 Media Processing Software Development Reference*.

- Just enough public video buffers (VBs)

See **HI\_MPI\_VB\_SetConf**.

- For the Linux version, see chapter 2 "System Control" in the *HiMPP IPC V2.0 Media Processing Software Development Reference* (applicable to the Hi3516A/Hi3518E V20X/Hi3519 V100), and the *HiMPP IPC V3.0 Media Processing Software Development Reference* (applicable to Hi3519 V101/Hi3516C V300).

- For the Huawei LiteOS version, see the *HiMPP IPC V3.0 Media Processing Software Development Reference*.

In the proc information about the VB, if **IsComm** is **1**, the VB pool is a public VB pool. If **MinFree** of the public VB pool is **0** and there is no displayed information indicating that a module cannot obtain the VB in the logmpp, the public VBs are just enough.



- Video input unit (VIU)
  - For the Linux version, see chapter 3 "VI" in the *HiMPP IPC V2.0 Media Processing Software Development Reference* (applicable to the Hi3516A/Hi3518E V20X/Hi3519 V100), and the *HiMPP IPC V3.0 Media Processing Software Development Reference* (applicable to Hi3519 V101/Hi3516C V300).
  - For the Huawei LiteOS version, see the *HiMPP IPC V3.0 Media Processing Software Development Reference*.

Measure	MPI/Parameter	Benefit	Impact	Note
Add the switch for wide dynamic range (WDR) compression.	HI_MPI_VI_SetWDRAttr HI_MPI_VI_GetWDRAttr	One buffer (that stores one frame) is saved in non-compression mode (one buffer is required in non-compression mode, and two buffers are required in compression mode).	The bandwidth usage in non-compression mode is higher than that in compression mode.	This measure is supported only by the Hi3516A.

- Video processing subsystem (VPSS)
  - For the Linux version, see chapter 5 "VPSS" in the *HiMPP IPC V2.0 Media Processing Software Development Reference* (applicable to the Hi3516A/Hi3518E V20X/Hi3519 V100), and the *HiMPP IPC V3.0 Media Processing Software Development Reference* (applicable to Hi3519 V101/Hi3516C V300).
  - For the Huawei LiteOS version, see the *HiMPP IPC V3.0 Media Processing Software Development Reference*.

Measure	MPI/Parameter	Benefit	Impact	Note
Add the switch for 3DNR reference frame compression.	rfr_frame_comp	One buffer (that stores one frame) is saved in non-compression mode (one buffer is required in non-compression mode, and two buffers are required in compression mode).	The bandwidth usage in non-compression mode is higher than that in compression mode.	This measure is supported only by the Hi3516A.
Select CHN0 as the source of the reference frame.	Hi3516A, Hi3519V100, Hi3516CV300: HI_MPI_VPSS_SetRefSelect HI_MPI_VPSS_GetRefSelect Hi3519V101: HI_MPI_VPSS_SetGrpAttr, HI_MPI_VPSS_GetGrpAttr	The buffer for storing 3DNR reference frames is saved.	The 3DNR function does not take effect if the lens distortion correction (LDC), rotation, or mirror/flip function of CHN0 is enabled.	



– Video encoding (VENC)

- For the Linux version, see chapter 6 "VENC" in the *HiMPP IPC V2.0 Media Processing Software Development Reference* (applicable to the Hi3516A/Hi3518E V20X/Hi3519 V100), and the *HiMPP IPC V3.0 Media Processing Software Development Reference* (applicable to Hi3519 V101/Hi3516C V300).
- For the Huawei LiteOS version, see the *HiMPP IPC V3.0 Media Processing Software Development Reference*.

Measure	MPI/Parameter	Benefit	Impact	Note
Dynamically switch the encoding resolution.	HI_MPI_VENC_GetChnAttr HI_MPI_VENC_SetChnAttr	The VENC channel is not destroyed when the encoding resolution is switched, which reduces memory fragments.	None	After the encoding resolution is switched, all parameters are restored to default values.
Enable the reference frame to share the memory for storing the luminance data with the reconstruction frame.	H264eRcnEqualRef	The memory for storing the luminance data is saved for each channel.	<ul style="list-style-type: none"><li>• Only I frames can be inserted when the jumbo frame appears, bit rate overshoot occurs, or the stream buffer is full.</li><li>• One more memory for storing the luminance data is required in 2x frame skipping reference mode.</li></ul>	This measure is supported only by the Hi3516A and Hi3518E V200.
Allocate the encoding stream buffers in memory reduction mode.	H264eMiniBufMode H265eMiniBufMode JpegeMiniBufMode	The stream buffer size can be reduced.	If the memory reduction mode is used, the stream buffer size must be set to an appropriate size. Otherwise, re-encoding or frame discarding occurs due to insufficiency of the stream buffer.	-

– Video graphics subsystem (VGS)

- For the Linux version, see chapter 13 "VGS" in the *HiMPP IPC V2.0 Media Processing Software Development Reference* (applicable to the Hi3516A/Hi3518E V20X/Hi3519 V100), and the *HiMPP IPC V3.0 Media Processing Software Development Reference* (applicable to Hi3519 V101/Hi3516C V300).
- For the Huawei LiteOS version, see the *HiMPP IPC V3.0 Media Processing Software Development Reference*.



Measure	Parameter	Benefit	Impact	Note
Set the maximum number of jobs supported by the VGS.	max_vgs_job	The size of the memory occupied by a job is 192 bytes.	The VGS performance is restricted if the number of jobs is too small.	None
Set the maximum number of tasks supported by the VGS.	max_vgs_task	The size of the memory occupied by a task is 1256 bytes.	The VGS performance is restricted if the number of tasks is too small.	None
Set the maximum number of nodes supported by the VGS.	max_vgs_node	The size of the memory occupied by a node is 1104 bytes.	The VGS performance is restricted if the number of nodes is too small.	None

– HiFB

For details, see the *HiFB Development Guide*.

Measure	MPI/Parameter	Benefit	Impact	Note
Set an appropriate size for the physical display buffer at the graphics layer.	video	Setting an appropriate size for the physical display buffer at the graphics layer according to the actual resolution avoids memory waste.	None	None

## 1.3 Performance

### 1.3.1 What Do I Do If the VGS LDC Performance Does Not Meet Requirements?

[Symptom]

For the Hi3516A, the LDC performance cannot reach 5-megapixel@30 fps.

[Cause Analysis]

In the HiMPP, the functions provided by the VGS include scaling (supported by the extended channels), CoverEx overlaying, OverlayEx overlaying, rotation, and LDC. The VGS performance is the sum of the performance of each function. The functions are shared by other modules such as the VIU, VPSS, and VENC.

[Solution]

For the Hi3516A, the performance of the VGS LDC is restricted to 1080p@30 fps. You are advised to use the LDC function in offline mode.



### 1.3.2 What Do I Do If Image Transition Occurs When the Correction Ratio Is Configured Continuously in Fullall Mode for the VGS LDC?

[Symptom]

If the correction ratio is configured continuously when the LDC works in fullall mode, image transition occurs.

[Cause Analysis]

The LDC application is classified as follows:

- The machine connects to prime lens. For this type of machines, the LDC parameters are calibrated in the laboratory, and the correction ratio does not need to be adjusted during actual application. Therefore, the LDC functions properly in both crop and fullall modes.
- The machine connects to zoom lens. For this type of machines, multiple groups of parameters are calibrated based on the focal length. During actual application, the correction ratio is configured continuously based on the ratio parameters for different focal lengths during focus adjustment. In this case, the LDC functions properly in crop mode. However, the image transition is uneven in fullall mode. According to test results, the transition occurs only once or twice after all possible value ranges are traversed.

Currently this issue occurs only in the Hi3516A/Hi3516D, Hi3518E V200, and Hi3516C V200 solutions. It does not involve the Hi3518E V100, Hi3518C V100, Hi3518A V100, and Hi3516C V100 solutions.

### 1.3.3 What Do I Do If the CPU Usage Statistics Obtained by Running the Top Command Fluctuates?

[Symptom]

The CPU usage statistics obtained by running the **top** command is not accurate and fluctuates greatly in small service scenarios.

[Cause Analysis]

The default frequency of the Linux kernel in this version is 100 Hz. That is, the interval for calculating the CPU usage is 10 ms. The time granularity is coarse, which results in low statistical precision and large fluctuations in the statistics.

[Solution]

To improve the precision of the CPU usage statistics, you can change the kernel frequency to 1000 Hz.

### 1.3.4 How Do I Configure the Clock Frequency of Each Module?

- For Hi3519 V101, the default clock frequency of each module is as follows:  
IVE: 396 MHz; GDC: 475 MHz; VGS: 500 MHz; VEDU: 600 MHz; VPSS: 300 MHz;  
VIO: 300 MHz; ISP0: 300 MHz; ISP1: 300 MHz; VI1: 300 MHz
- Besides the default frequency, other frequencies of each module can also be configured. For details, see the *Hi3519 V101 HD IP Camera SoC Data Sheet*. For details about how to change the frequency of a module, see **clkcfg\_hi3519v101.sh**.



The clock frequencies of the VI and ISP modules need to be configured based on the front-end sensors. For details, see **readme\_en.txt** in **mpp\_big-little/component/isp/sensor/**.

The default frequency of the VPSS module is 300 MHz. During debugging, you can change the VPSS frequency to 396 MHz if the VPSS performance is insufficient.

## 1.4 Tailoring

### 1.4.1 How Do I Reduce the Application Size When the Static Libraries Are Used?

#### [Symptom]

The application uses only a small part of the functions in the **libmpi.a** library. However, the application needs to link to the associated library files such as **vqev2**. As a result, the application size is too large.

#### [Cause Analysis]

The application needs to link to all the functions defined in the MPI libraries by default when it links to the MPI libraries. Therefore, the application needs to use other libraries associated with the MPI libraries.

#### [Solution]

When the libraries of the HiMPP are generated, add the **-ffunction-sections** compilation option to **Makefile.param**. When the application links to the MPI libraries during compilation, add the **-Wl,-gc-sections** compilation option to **Makefile**. This deletes the functions that are not used and reduces the application size significantly.



# 2 MIPI

## 2.1 MIPI Configuration

For details about the Hi3516A mobile industry processor interface (MIPI) specifications, see the *Hi3516A/Hi3516D HD IP Camera SoC Data Sheet* and *Features of the Video Interfaces of HiSilicon IP Cameras*.

### 2.1.1 How Do I Configure lane\_id?

[Symptom]

The Hi3516A MIPI provides two links, and each link contains four lanes. You can choose one or more lanes as required by setting lane\_id[8]. How do I configure lane\_id[8] when the Hi3516A is connected to a sensor?

[Solution]

Set the lane\_id of the lanes that are not used to -1 when the Hi3516A is connected to a sensor. The sequence of data channels can be adjusted by configuring the Lane\_id register of the MIPI. The value of the Lane\_id register is configured based on the mapping between the board and the actual output channel of the sensor.

The following section takes the MN34220 sensor as an example:

- Step 1** Check the mapping between the hardware connection and hardware. For example, the MN34220 of the low-voltage differential signaling (LVDS) interface uses four lanes. [Table 2-1](#) describes the hardware connection.

**Table 2-1** Hardware connection

MN34220 Pin	Hi3516A MIPI Pin
SENSOR_SDODA0M SENSOR_SDODA0P	MIPI0_D0M MIPI0_D0P
SENSOR_SDODA1M SENSOR_SDODA1P	MIPI0_D1M MIPI0_D1P
SENSOR_SDODB0M SENSOR_SDODB0P	MIPI1_D0M MIPI1_D0P

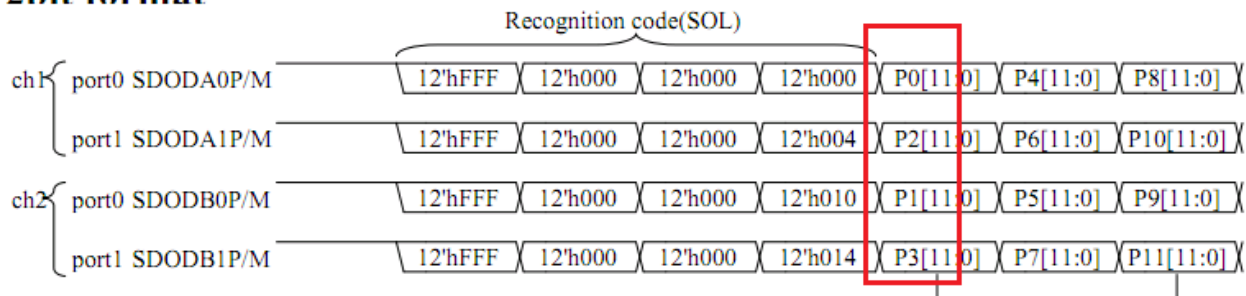


MN34220 Pin	Hi3516A MIPI Pin
SENSOR_SDODB1M SENSOR_SDODB1P	MIP11_D1M MIP11_D1P

**Step 2** Check the actual transfer relationship. For details, see the sensor data sheet. The actual transferred data is 0, 2, 1, 3, as shown in [Figure 2-1](#).

**Figure 2-1** Actual transfer relationship

## 12bit format



According to the results in step 1 and step 2, the configuration of lane\_id is as follows:

.lane\_id = {0, 2, -1, -1, 1, 3, -1, -1}

Whether the configuration of sync\_code takes effect depends on lane\_id. If lane\_id is set to -1, the configuration of the corresponding sync\_code does not take effect.

----End

## 2.1.2 How Do I Configure the Mode of sync\_code?

[Symptom]

The sync\_code of the LVDS/SUB\_LVDS interface of the sensor can be set to two modes: LVDS\_SYNC\_MODE\_SOL and LVDS\_SYNC\_MODE\_SAV. sync\_code needs to be configured based on the sensor type and transfer mode of the sensor.

[Solution]

You need to configure sync\_code[LVDS\_LANE\_NUM][WDR\_VC\_NUM][SYNC\_CODE\_NUM] in the MIPI according to the sensor data sheet.

[Table 2-2](#) describes the definitions of LVDS\_LANE\_NUM, WDR\_VC\_NUM, and SYNC\_CODE\_NUM.

**Table 2-2** Definitions of LVDS\_LANE\_NUM, WDR\_VC\_NUM, and SYNC\_CODE\_NUM

Field	Description
LVDS_LANE_NUM	Number of MIPI lanes that correspond to the LVDS physical





Field	Description
	channels. The Hi3516A MIPI provides eight lanes.
WDR_VC_NUM	Number of WDR channels. The value is 2 in 2-frame combination WDR mode, and the maximum value is 4. The Hi3516A supports only the 2-frame combination WDR mode.
SYNC_CODE_NUM	Number of code words. The sync_code of each lane consists of four code words. The meanings of the code words vary in different modes.

Table 2-3 describes the meanings of sync\_code[4] for each lane in two modes.

**Table 2-3** Meanings of sync\_code[4] for each lane in two modes

sync_mode	Meaning of sync_code[4]
LVDS_SYNC_MODE_SOL	SOF, EOF, SOL, EOL
LVDS_SYNC_MODE_SAV	Invalid sav, invalid eav, valid sav, valid eav



**NOTE**

The first three code words of the sync\_code for each SOF/EOF/SOL/EOL are fixed at 0xFFFF, 0x0000, 0x0000.

The sync\_code is configured in the normal sequence in one link regardless of the sequence in which hardware is connected to the sensor.

That is, the sync\_code is configured in the normal sequence in one link even if the two links are out of order. The two links of the Panasonic sensor are always out of order.

- Disorder in one link

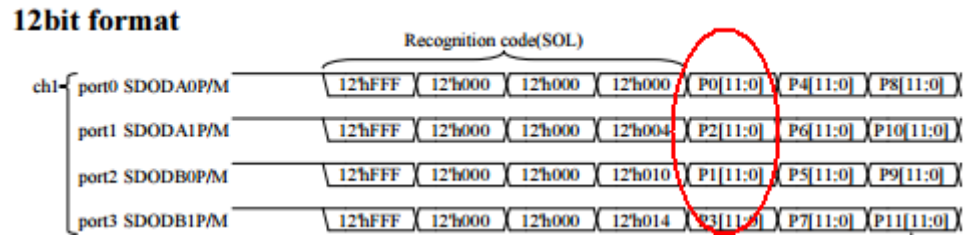
Figure 2-2 shows the port configurations when the MN34220 sync\_mode is **LVDS\_SYNC\_MODE\_SOL** and the sensor works in 12-bit 1-channel 4-port 4-lane mode.

**Figure 2-2** Port configurations when the MN34220 sync\_mode is LVDS\_SYNC\_MODE\_SOL and the sensor works in 12-bit 1-channel 4-port 4-lane mode

ch	port	Name	Code (12bit×4)				ch	port	Name	Code (12bit×4)			
ch1	Port0	SOF	FFFh	000h	000h	002h	ch1	Port2	SOF	FFFh	000h	000h	012h
		SOL	FFFh	000h	000h	000h			SOL	FFFh	000h	000h	010h
		EOL	FFFh	000h	000h	001h			EOL	FFFh	000h	000h	011h
		EOF	FFFh	000h	000h	003h			EOF	FFFh	000h	000h	013h
	Port1	SOF	FFFh	000h	000h	006h		Port3	SOF	FFFh	000h	000h	016h
		SOL	FFFh	000h	000h	004h			SOL	FFFh	000h	000h	014h
		EOL	FFFh	000h	000h	005h			EOL	FFFh	000h	000h	015h
		EOF	FFFh	000h	000h	007h			EOF	FFFh	000h	000h	017h

Figure 2-3 shows the sequence of pixels in each channel.

**Figure 2-3** Sequence of pixels in each channel



The sync\_code should be configured in the MIPI in the sequence of SDODA0, SDODB0, SDODA1, and SDODB1. The sync\_code configuration is as follows:

```
.sync_code = {
    {{0x002, 0x003, 0x000, 0x001}}, //PHY0_lane0
    {0x202, 0x203, 0x200, 0x201},
    {0x102, 0x103, 0x100, 0x101},
    {0x302, 0x303, 0x300, 0x301}},
    {{0x012, 0x013, 0x010, 0x011}}, //PHY0_lane1
    {0x212, 0x213, 0x210, 0x211},
    {0x112, 0x113, 0x110, 0x111},
    {0x312, 0x313, 0x310, 0x311}},
    {{0x006, 0x007, 0x004, 0x005}}, //PHY0_lane2
    {0x206, 0x207, 0x204, 0x205},
    {0x106, 0x107, 0x104, 0x105},
    {0x306, 0x307, 0x304, 0x305}},
    {{0x016, 0x017, 0x014, 0x015}}, //PHY0_lane3
    {0x216, 0x217, 0x214, 0x215},
    {0x116, 0x117, 0x114, 0x115},
    {0x316, 0x317, 0x314, 0x315}},
    {{0x00a, 0x00b, 0x008, 0x009}}, //PHY1_lane0
    {0x20a, 0x20b, 0x208, 0x209},
    {0x10a, 0x10b, 0x108, 0x109},
    {0x30a, 0x30b, 0x308, 0x309}},
    {{0x00a, 0x00b, 0x008, 0x009}}, //PHY1_lane1
    {0x20a, 0x20b, 0x208, 0x209},
    {0x10a, 0x10b, 0x108, 0x109},
    {0x30a, 0x30b, 0x308, 0x309}},
    {{0x01a, 0x01b, 0x018, 0x019}}, //PHY1_lane2
    {0x21a, 0x21b, 0x218, 0x219},
    {0x11a, 0x11b, 0x118, 0x119},
    {0x31a, 0x31b, 0x318, 0x319}},
    {{0x01a, 0x01b, 0x018, 0x019}}, //PHY1_lane3
    {0x21a, 0x21b, 0x218, 0x219},
    {0x11a, 0x11b, 0x118, 0x119},

```



```
{0x31a, 0x31b, 0x318, 0x319}}
}
```

- Disorder between two links

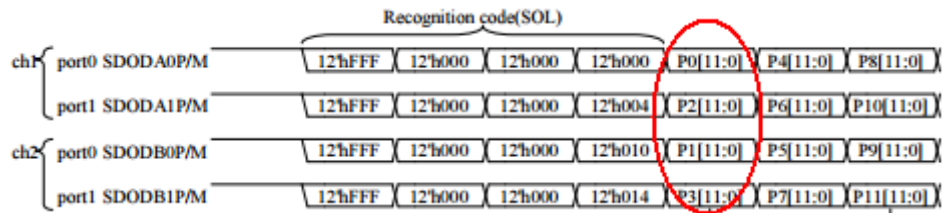
Figure 2-4 shows the port configurations when the MN34220 sync\_mode is LVDS\_SYNC\_MODE\_SOL and the sensor works in 12-bit 2-channel 2-port 4-lane mode.

**Figure 2-4** Port configurations when the MN34220 sync\_mode is LVDS\_SYNC\_MODE\_SOL and the sensor works in 12-bit 2-channel 2-port 4-lane mode

ch	port	Name	Code (12bit×4)				ch	port	Name	Code (12bit×4)			
ch1	Port0	SOF	FFFh	000h	000h	002h	ch2	Port0	SOF	FFFh	000h	000h	012h
		SOL	FFFh	000h	000h	000h			SOL	FFFh	000h	000h	010h
		EOL	FFFh	000h	000h	001h			EOL	FFFh	000h	000h	011h
		EOF	FFFh	000h	000h	003h			EOF	FFFh	000h	000h	013h
	Port1	SOF	FFFh	000h	000h	006h		Port1	SOF	FFFh	000h	000h	016h
		SOL	FFFh	000h	000h	004h			SOL	FFFh	000h	000h	014h
		EOL	FFFh	000h	000h	005h			EOL	FFFh	000h	000h	015h
		EOF	FFFh	000h	000h	007h			EOF	FFFh	000h	000h	017h

Figure 2-5 shows the sequence of pixels in each channel.

**Figure 2-5** Sequence of pixels in each channel



The line sequences of the two links may overlap. However, only the configuration of sync\_code in one link is considered. The final sync\_code configuration is as follows:

```
.sync_code = {
    {{0x002, 0x003, 0x000, 0x001}, //PHY0_lane0
    {0x202, 0x203, 0x200, 0x201},
    {0x102, 0x103, 0x100, 0x101},
    {0x302, 0x303, 0x300, 0x301}},
    {{0x006, 0x007, 0x004, 0x005}, //PHY0_lane1
    {0x206, 0x207, 0x204, 0x205},
    {0x106, 0x107, 0x104, 0x105},
    {0x306, 0x307, 0x304, 0x305}},
    {{0x00a, 0x00b, 0x008, 0x009}, //PHY0_lane2
    {0x20a, 0x20b, 0x208, 0x209},
    {0x10a, 0x10b, 0x108, 0x109},
    {0x30a, 0x30b, 0x308, 0x309}},
    {{0x00a, 0x00b, 0x008, 0x009}, //PHY0_lane3
    {0x20a, 0x20b, 0x208, 0x209},
```



```
{0x10a, 0x10b, 0x108, 0x109},
{0x30a, 0x30b, 0x308, 0x309}},
{{0x012, 0x013, 0x010, 0x011}, //PHY1_lane0
{0x212, 0x213, 0x210, 0x211},
{0x112, 0x113, 0x110, 0x111},
{0x312, 0x313, 0x310, 0x311}},
{{0x016, 0x017, 0x014, 0x015}, //PHY1_lane1
{0x216, 0x217, 0x214, 0x215},
{0x116, 0x117, 0x114, 0x115},
{0x316, 0x317, 0x314, 0x315}},
{{0x01a, 0x01b, 0x018, 0x019}, //PHY1_lane2
{0x21a, 0x21b, 0x218, 0x219},
{0x11a, 0x11b, 0x118, 0x119},
{0x31a, 0x31b, 0x318, 0x319}},
{{0x01a, 0x01b, 0x018, 0x019}, //PHY1_lane3
{0x21a, 0x21b, 0x218, 0x219},
{0x11a, 0x11b, 0x118, 0x119},
{0x31a, 0x31b, 0x318, 0x319}}
}
```

### 2.1.3 What Do I Do If Exceptions Occur in the 2-Link Scenario of the LVDS Mode?

#### [Symptom]

When two links are used to transfer data, data errors occur when there is interference (such as ESD), which cause exceptions such as picture artifacts. After the interference is eliminated and the front-end (sensor output) data becomes normal again, the system cannot be restored to the normal status.

#### [Solution]

1. Set bit[19:16] of the interrupt mask registers INT\_MASK\_LINK0 (offset address: 0x1004) and INT\_MASK\_LINK1 (offset address: 0x1404) to 0.
2. Delete the int\_raw and int2\_raw conditional judgment from mipi\_int\_statics of the MIPI interrupt handling function mipi\_interrupt\_route. That is, the MIPI controller needs to be reset as long as the MIPI interrupt handling function is entered.
3. Set the width to the actual output width of the sensor when configuring the attributes of the LVDS mode.

The following takes the 3-megapixel (2048 x 1536) linear scenario of the IMX123 sensor as an example, in which the actual output width is 2064:

- Change the macro definition in hi\_mipi.h from #define COMBO\_LINK\_INT\_MASK ~(0x300000) to #define COMBO\_LINK\_INT\_MASK ~(0x3f0000).
- Delete the conditional judgment if (int\_raw || int2\_raw) from mipi\_int\_statics of the interrupt function mipi\_interrupt\_route.
- The user program needs to set **img\_size.width** and **img\_size.height** in the LVDS attribute **lvds\_dev\_attr\_t** to **2064** and **1536**, respectively.



#### NOTE

The preceding modification applies only to the 2-link scenario of the LVDS mode. Do not modify the raw code in other cases.

## 2.2 MIPI Frequency

### 2.2.1 How Do I Query the Relationship Between the Transfer Frequency of MIPI Lanes and VI Processing Frequency?

#### [Symptom]

When multiple MIPI lanes are used for data transfer, how do I query the relationship between the transfer frequency of MIPI lanes and VI processing frequency, and how do I calculate the maximum transfer frequency of each lane?

#### [Solution]

The Hi3516A receives data from multiple lanes over the MIPI, converts data into the internal timing, and transmits the timing to the VIU for processing. The total amount of data transferred by multiple lanes remains unchanged, as shown in the following equation:

$$\text{VI\_Freq} \times \text{Pix\_Width} = \text{Lane\_Num} \times \text{MIPI\_Freq}$$

Where **VI\_Freq** indicates the frequency of the VI working clock, **Pix\_Width** indicates the pixel bit width, **Lane\_Num** indicates the number of lanes used for data transfer, and **MIPI\_Freq** indicates the maximum RX frequency of each lane.

**MIPI\_Freq** is calculated as follows:  $\text{MIPI\_Freq} = \text{VI\_Freq} \times \text{Pix\_Width} / \text{Lane\_Num}$ . For example, if the frequency of the VI working clock is 250 MHz, the MIPI data is in RAW12 format, and four lanes are used for data transfer, **MIPI\_Freq** is 750 (250 x 12/4).

That is, the maximum transfer frequency of each lane is 750 MHz.

## 2.3 Sensor Reset

### 2.3.1 How Do I Change the Hi3516A Sensor Reset Pin?

#### [Symptom]

The Hi3516A has one dedicated sensor reset pin (SENSOR\_RSTN). You are advised to use it by default. If you need to modify the sensor reset pin, modify the MIPI driver accordingly.

#### [Cause Analysis]

Sensor reset/unreset operations are implemented in the MIPI driver. There are requirements on the sequence of the sensor reset/unreset and MIPI configuration.

#### [Solution]

Modify the `mipi_reset_sensor/mipi_unreset_sensor` function in the MIPI driver based on the actually used pin.



# 3 VI

## 3.1 DIS

### 3.1.1 How Do I Use the DIS Function of the Hi3516A in the 5-Megapixel Scenario?

#### [Symptom]

In the 1080p scenario, digital image stabilization (DIS) is implemented as follows: If the resolution of the input picture for the VIU is greater than 1920 x 1080, for example, 2176 x 1336, the media processing platform (MPP) algorithm is used to calculate the offset due to picture flicker and then the input picture is cropped to a 1080p picture by a VPSS group. However, the DIS function cannot be implemented in the 5-megapixel scenario because the maximum width of the pictures that can be captured by the VIU is 2592 pixels.

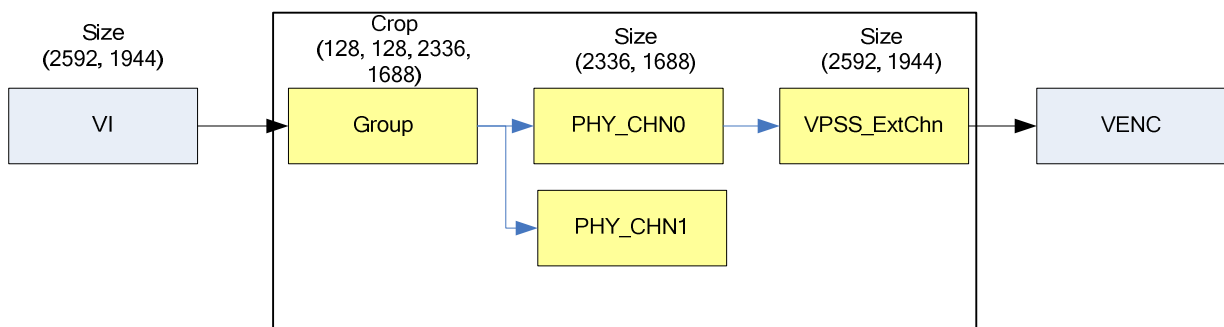
#### [Cause Analysis]

The maximum width of the pictures that can be captured by the VIU is 2592 pixels in the 5-megapixel scenario. To implement DIS when the picture width is greater than 2592 pixels, the picture must be cropped to the picture with the resolution less than 2592 x 1944 (5 megapixels), such as 2336 x 1688, and then scaled to a 5-megapixel picture.

#### [Solution]

After the picture is cropped, VPSS channel 0 is bound to an extended channel and the picture is scaled to a 5-megapixel picture, as shown in [Figure 3-1](#).

**Figure 3-1** Picture processing procedure





[Note]

In the preceding example, the height (in pixel) of the input picture is 1944 in the 5-megapixel scenario. The height needs to be configured according to the sensor timing and can be greater than 1944. The maximum jitter range allowed by the Hi3516A is  $\pm 128$  (in pixel), and the jitter range can be tuned by adjusting the start position of the region to be cropped.



# 4 Fisheye

## 4.1 Fisheye Correction

### 4.1.1 How Do I Implement Fisheye Correction of More than Four Cells by Using Hi3519 V100/ Hi3519 V101?

#### [Symptom]

In the system binding channel, the VI/VPSS interface HI\_MPI\_VI\_SetFisheyeAttr/HI\_MPI\_VPSS\_SetFisheyeAttr allows you to combine only two to four cells (at most four cells).

#### [Cause Analysis]

The implementation of the VI/VPSS interface for combining more than four cells complicates the system. In addition, the GDC performance of Hi3519 V100/Hi3519 V101 is insufficient.

#### [Solution]

Obtain pictures from the VI/VPSS module, implement fisheye correction by calling HI\_MPI\_FISHEYE\_AddCorrectionTask, and then transfer the pictures back to the system channel (VI/VPSS/VO module). The configuration of the LMF parameters can also be implemented by calling the fisheye interface HI\_MPI\_FISHEYE\_SetConfig.



#### NOTE

For details, see the fourth fisheye sample. The GDC performance may be insufficient.





# 5 Audio

## 5.1 How Do I Play the Audio Streams Encoded by HiSilicon on the PC?

### 5.1.1 How Do I Play G711/G726/ADPCM Audio Streams Encoded by HiSilicon on the PC?

#### [Symptom]

The G711/G726/ADPCM audio streams encoded by HiSilicon cannot be played directly by using software on the PC.

#### [Cause Analysis]

A HiSilicon voice frame header is added at the beginning of each frame in the audio streams encoded by HiSilicon.

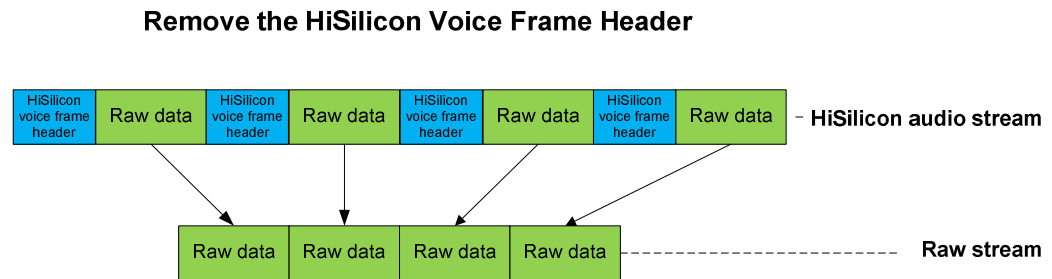
- For the Linux version, see section 9.2.2.3 in the *HiMPP IPC V2.0 Media Processing Software Development Reference* (applicable to the Hi3516A/Hi3518E V20X/Hi3519 V100), and the *HiMPP IPC V3.0 Media Processing Software Development Reference* (applicable to Hi3519 V101/Hi3516C V300).
- For the Huawei LiteOS version, see the *HiMPP IPC V3.0 Media Processing Software Development Reference*. The HiSilicon voice frame header cannot be identified by software on the PC.

#### [Solution]

Remove the HiSilicon voice frame header at the beginning of each frame, add the WAV header to the frames in the raw stream, and play the streams using software on the PC. [Figure 5-1](#) shows how to remove the HiSilicon voice frame header.



**Figure 5-1** Remove the HiSilicon voice frame header



The reference code for removing the HiSilicon voice frame header is as follows:

```
int HisiVoiceGetRawStream(short *HisiVoicedata, short *outdata, int
hisisamplelen)
{
    int len = 0, outlen = 0;
    short *copyHisiData, *copyOutdata;
    int copySamplelen = 0;
    copySamplelen = hisisamplelen;
    copyHisiData = HisiVoicedata;
    copyOutdata = outdata;
    while(copySamplelen > 2)
    {
        len = copyHisiData[1]&0x00ff;
        copySamplelen -= 2;
        copyHisiData += 2;
        if(copySamplelen < len)
        {
            break;
        }
        memcpy(copyOutdata, copyHisiData, len * sizeof(short));
        copyOutdata += len;
        copyHisiData += len;
        copySamplelen -= len;
        outlen += len;
    }
    return outlen;
}
```



**NOTE**

- The audio streams in ADPCM\_DVI4 or ADPCM\_ORG\_DVI4 format are used for network transfer over the Real-time Transport Protocol (RTP) and cannot be played by the client programs on the PC. For details, see the RFC3551 standard.



- The method of adding the WAV header is not provided in this document. You can add the WAV header by following the WAV header standard. For details, see the reference links [https://msdn.microsoft.com/en-us/library/dd390970\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/dd390970(v=vs.85).aspx) and <http://www.moon-soft.com/program/FORMAT/windows/wavec.htm>.

## 5.2 How Do I Play Standard Audio Streams on HiSilicon Chips?

### 5.2.1 How Do I Play Standard G711/G726/ADPCM Audio Streams on HiSilicon Chips?

[Symptom]

The standard G711/G726/ADPCM audio streams cannot be played directly on HiSilicon chips.

[Cause Analysis]

To ensure that the previous-generation chips are compatible, the audio streams can be played on HiSilicon chips only after the HiSilicon voice frame header is added at the beginning of each frame in the raw audio streams.

[Solution]

To play G711/G726/ADPCM audio streams on HiSilicon chips, obtain the raw stream data, add the HiSilicon voice frame header at the beginning of each frame based on the frame data length **PerSampleLen**.

1. Obtain the raw stream data. Remove the WAV header if the WAV header is added to the frame.
2. Obtain the data length of each frame (**PersampleLen**, a short number).

**Table 5-1** Data length of each frame

Encoding Format	Data Length of Each Frame	Remarks
G711	$N \times 40$	$N$ is a positive integer ranging from 1 to 5.
G726 (16 kbit/s)	$N \times 10$	$N$ is a positive integer ranging from 1 to 5.
G726 (24 kbit/s)	$N \times 15$	$N$ is a positive integer ranging from 1 to 5.
G726 (32 kbit/s)	$N \times 20$	$N$ is a positive integer ranging from 1 to 5.
G726 (40 kbit/s)	$N \times 25$	$N$ is a positive integer ranging from 1 to 5.
IMA ADPCM	Number of bytes in each block/2	The number of bytes in each block indicates the number of bytes in the encoded IMA ADPCM data of each block, corresponding to <b>nblockalign</b> (0x20–0x21, 2-byte) of the IMA ADPCM WAV header.

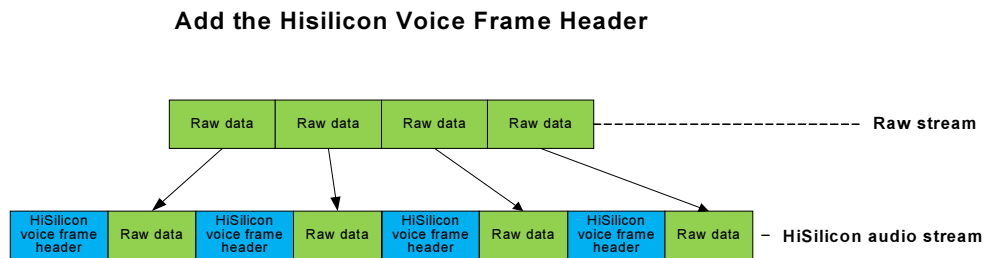


**NOTE**

- Of all the ADPCM formats, only the IMA ADPCM format is supported. The number of bytes in each sampling point (**whitspersample**) must be 4.
- If the WAV header is added to the frames in ADPCM streams, the number of bytes in each block can be obtained from the WAV header. For the raw ADPCM streams, the number of bytes in each block must be obtained from the provider of the streams.
- Only the mono-channel encoding format is supported.

3. Add the HiSilicon voice frame header, as shown in [Figure 5-2](#).

**Figure 5-2** Adding the HiSilicon voice frame header



The reference code for adding the HiSilicon voice frame header is as follows:

```
int HisiVoiceAddHisiHeader(short *inputdata, short *Hisivoicedata, int
PersampleLen,int inputsamplelen)
{
    int len = 0, outlen = 0;
    short HisiHeader[2];
    short *copyHisidata, *copyinputdata;
    int copysamplelen = 0;
    HisiHeader[0] = (short)(0x001<<8) & (0x0300);
    HisiHeader[1] = PersampleLen & 0x00ff;
    copysamplelen = inputsamplelen;
    copyHisidata = Hisivoicedata;
    copyinputdata = inputdata;
    while(copysamplelen >= PersampleLen)
    {
        memcpy(copyHisidata, HisiHeader, 2 * sizeof(short));
        outlen += 2;
        copyHisidata += 2;
        memcpy(copyHisidata, copyinputdata, PersampleLen * sizeof(short));
        copyinputdata += PersampleLen;
        copyHisidata += PersampleLen;
        copysamplelen -= PersampleLen;
        outlen += PersampleLen;
    }
}
```



```
        return outlen;  
    }
```

## 5.3 What Do I Do If High-Frequency Information Is Lost After VQE Is Enabled?

### 5.3.1 What Do I Do If High-Frequency Information Is Lost After VQE Is Enabled?

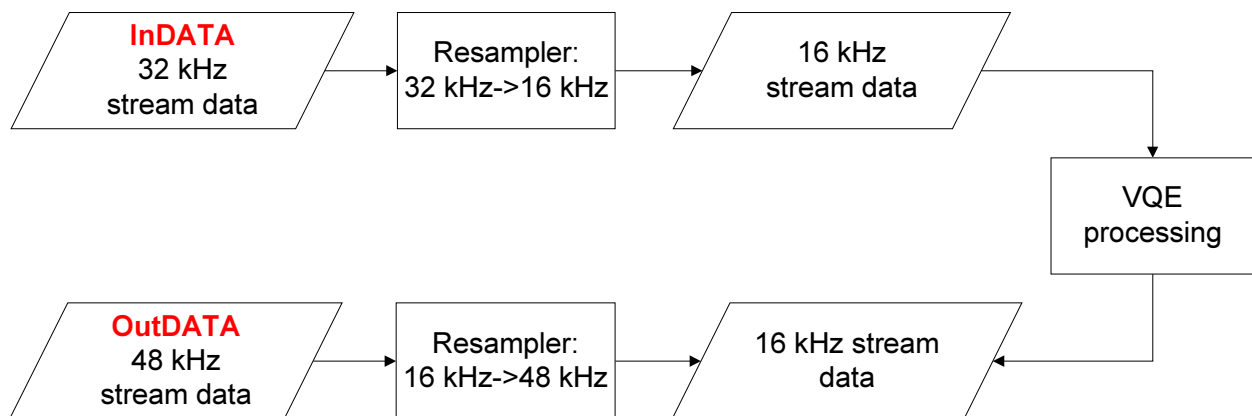
[Symptom]

When the AI sampling rate (**AI SampleRate**) is 32 kHz, the voice quality enhancement (VQE) working sampling rate (**VQE WorkSampleRate**) is 16 kHz, output sampling rate (**ResOutSampleRate**) is 48 kHz, and the VQE and resampling functions are enabled, the 8 kHz or higher-frequency information is lost according to the analysis result of the output sequence.

[Cause Analysis]

In actual applications, the HiVQE supports only the 8 kHz and 16 kHz working sampling rates. To meet customer requirements, the resampling layer is encapsulated in the HiVQE to support any standard sampling rate from 8 kHz to 48 kHz. When **AI SampleRate**, **VQE WorkSampleRate**, and **ResOutSampleRate** are set to **48 kHz**, **16 kHz**, and **48 kHz** respectively, the resampling layer resamples data from 32 kHz to 16 kHz, and then resamples data from 16 kHz to 48 kHz for output after VQE processing. [Figure 5-3](#) shows the VQE processing flow.

**Figure 5-3** VQE processing flow



When data is resampled from 32 kHz to 16 kHz, 8 kHz or higher-frequency information is lost.

In the current application scenario, the frequency band information of the output sequence is as follows:



- When VQE and resampling are disabled, the information within the frequency band of 0 to  $(\text{AISampleRate}/2)$  is output. For example, if the AI sampling rate (**AISampleRate**) is 48 kHz, the frequency band of the output information is 0 kHz to 24 kHz.
- When VQE is disabled and resampling is enabled, the information within the frequency band of 0 to  $\min(\text{AISampleRate}, \text{ResOutSampleRate})/2$  is output. For example, if the AI sampling rate (**AISampleRate**) is 16 kHz and the output sampling rate (**ResOutSampleRate**) is 32 kHz,  $\min(\text{AISampleRate}, \text{ResOutSampleRate})$  is 16 kHz. Therefore, the frequency band of the output information is 0 kHz to 8 kHz.
- When VQE is enabled and resampling is disabled, the information within the frequency band of 0 to  $\min(\text{AISampleRate}, \text{VQEWorkSampleRate})/2$  is output. For example, if the AI sampling rate (**AISampleRate**) is 32 kHz and the VQE working sampling rate (**VQEWorkSampleRate**) is 16 kHz,  $\min(\text{AISampleRate}, \text{VQEWorkSampleRate})$  is 16 kHz. Therefore, the frequency band of the output information is 0 kHz to 8 kHz.
- When VQE and resampling are enabled, the information within the frequency band of 0 to  $\min(\text{AISampleRate}, \text{VQEWorkSampleRate}, \text{ResOutSampleRate})/2$  is output. For example, if the AI sampling rate (**AISampleRate**) is 32 kHz, the VQE working sampling rate (**VQEWorkSampleRate**) is 16 kHz, and the output sampling rate (**ResOutSampleRate**) is 48 kHz,  $\min(\text{AISampleRate}, \text{VQEWorkSampleRate}, \text{ResOutSampleRate})$  is 16 kHz. Therefore, the frequency band of the output information is 0 kHz to 8 kHz.



#### NOTE

- After the sampling rate is configured, the maximum frequency of the output information is half of the sampling rate.
- The standard sampling rates from 8 kHz to 48 kHz are supported, including 8 kHz, 11.025 kHz, 12 kHz, 16 kHz, 22.05 kHz, 32 kHz, 44.1 kHz, and 48 kHz.
- The processing flow of the audio output unit (AOU) is similar to that of the audio input unit (AIU).

## 5.4 What Do I Do If Abnormal Amplitude Frequency Responses Occur During the Embedded Audio CODEC Output (AO Output)

### [Symptom]

During the testing of the amplitude frequency responses for the embedded audio CODEC (DAC) output (AO output), the amplitude frequency response is attenuated significantly in the frequency bands higher than 2 kHz.

### [Cause Analysis]

This case is caused by the enabled de-emphasis function of the `dac1_deemph[22:21]` and `dacr_deemph[20:19]` bits for the `MISC_CTRL52` control register of audio CODEC. The de-emphasis function is relative to the pre-emphasis function, and is modification on pre-emphasis. If the audio signal input to the AO channel is pre-emphasized, the signal can be restored to the normal amplitude frequency response when the de-emphasis function is enabled. If the audio signal input to the AO channel is not pre-emphasized, the amplitude frequency response will be affected when the de-emphasis function is enabled in bit [22:21] and bit [20:19] of the `MISC_CTRL52` register (the bits are not set to 00). This test is performed when the HiVQE function is disabled. During the test, the data input to the AO channel is not pre-emphasized. However, the de-emphasis function is enabled in the `dac1_deemph[22:21]` and `dacr_deemph[20:19]` bits of the `MISC_CTRL52` control register for the audio CODEC (the bits are not set to 00). As a result, this issue occurs.



[Solution]

For the AI channel, the pre-emphasis function of the audio CODEC control register is disabled. Therefore, the de-emphasis function needs to be disabled by default in the `dac1_deemph[22:21]` and `dacr_deemph[20:19]` bits of the `MISC_CTRL52` control register and the two fields are set to 00. Note that the pre-emphasis function and the de-emphasis function should be used in pairs.



**NOTE**

When bit [22:21] and bit [20:19] of the `MISC_CTRL52` control register are reserved, the de-emphasis function is not supported. In this case, the default configuration should be applied and additional configuration is not allowed.



# 6 Low Power Consumption

---

## 6.1 What Do I Do If the Frequency Is Frequently Modulated During Dynamic Frequency Modulation of the Low-Power Module?

### [Symptom]

When the dynamic frequency modulation policy (such as the on demand policy) is used after the low-power module **hi35xx\_pm.ko** is loaded, the frequency is frequently modulated.

### [Cause Analysis]

The Linux kernel uses the 100 Hz frequency by default, that is, 10 ms statistical period. The statistical time granularity is coarse, which leads to low precision in statistics. In this case, the CPU load statistics fluctuates significantly. Linux implements dynamic voltage and frequency scaling (DVFS) in each statistical cycle based on the CPU load statistics, which results in frequent frequency modulation of the low-power module.

### [Solution]

Change the frequency of the Linux kernel to 1000 Hz to improve the statistical precision, or increase the statistical cycle of the low-power module. For example, you can run the following command (the unit of the statistical cycle is  $\mu$ s) to change the statistical cycle of the on demand policy to 1s:

```
echo 1000000 >/sys/devices/system/cpu/cpufreq/ondemand/sampling_rate
```





# 7 LCD Debugging

## 7.1 Supported LCDs

The LCD compatibility depends on whether the LCD interface type matches the chip capability.

Table 7-1 lists the LCD interface types supported by the current IPC chips.

Table 7-1 Supported LCD interfaces

Interface	Chip		
	Hi3518E V200	Hi3519 V100	Hi3519 V101
6-bit serial	Supported	Supported	Supported
8-bit serial	Supported	Supported	Supported
16-bit parallel	Not supported	Supported	Supported
24-bit parallel	Not supported	Not supported	Supported

## 7.2 LCD Debugging Sequence

### 7.2.1 Confirming Pin Multiplexing Configurations

You need to confirm that all pins for connecting the LCD are correctly configured as VO-related functions, and the driving capabilities of these pins are properly configured. For details about the pin configurations, see the *Hi35xx HD IP Camera SoC Data Sheet*. For the Linux version, you can also see the script `pinmux_hi35xx.sh` in the SDK; for HuaweiLiteOS, you can see the pin configuration part in `sdk_init.c`.

### 7.2.2 Confirming User Timings

Currently the SDK provides only one kind of timing for each interface type. For example, for the VO\_INTF\_LCD\_8BIT interface, only VO\_OUTPUT\_320X240\_60 is provided. In addition, the timing is valid for only a specific LCD model. For example,



VO\_OUTPUT\_320X240\_60 applies to the LCDs with the driver IC OTA5182. Therefore, user timings are required when LCDs are debugged in most cases.

When configuring the public attributes of the output by calling HI\_MPI\_VO\_SetPubAttr, select the correct LCD interface type and the VO\_OUTPUT\_USER interface timing, and then configure the user timing structure based on the requirements of the LCD.

```
typedef struct tagVO_SYNC_INFO_S
{
    HI_BOOL bSynm;      /* sync mode(0:timing,as BT.656; 1:signal,as LCD)
    */
    HI_BOOL bIop;       /* interlaced or progressive display(0:i; 1:p) */
    HI_U8 u8Intfb;      /* interlace bit width while output */
    HI_U16 u16Vact;     /* vertical active area */
    HI_U16 u16Vbb;      /* vertical back blank porch */
    HI_U16 u16Vfb;      /* vertical front blank porch */
    HI_U16 u16Hact;     /* horizontal active area */
    HI_U16 u16Hbb;      /* horizontal back blank porch */
    HI_U16 u16Hfb;      /* horizontal front blank porch */
    HI_U16 u16Hmid;     /* bottom horizontal active area */
    HI_U16 u16Bvact;    /* bottom vertical active area */
    HI_U16 u16Bvbb;     /* bottom vertical back blank porch */
    HI_U16 u16Bvfb;     /* bottom vertical front blank porch */
    HI_U16 u16Hpw;      /* horizontal pulse width */
    HI_U16 u16Vpw;      /* vertical pulse width */
    HI_BOOL bIdv;       /* inverse data valid of output */
    HI_BOOL bIhs;       /* inverse horizontal synch signal */
    HI_BOOL bIvs;       /* inverse vertical synch signal */
} VO_SYNC_INFO_S;
```

Table 7-2 describes the parameters.

**Table 7-2** Parameter description

Parameter	Description
bSynm	Sync mode. Set it to <b>1</b> for LCDs, indicating signal synchronization.
bIop	<b>0</b> indicates interlaced, and <b>1</b> indicates progressive. Set it to <b>1</b> for LCDs typically.
u8Intfb	Invalid parameter, which can be ignored
u16Vact	Vertical active region. It indicates the vertical active region of the top field in interlaced output mode. The unit is row.
u16Vbb	Vertical blank back porch. It indicates the vertical blank back porch of the top field in interlaced output mode. The unit is row.
u16Vfb	Vertical blank front porch. It indicates the vertical blank front porch of the top field in interlaced output mode. The unit is row.



Parameter	Description
u16Hact	Horizontal active region. The unit is pixel.
u16Hbb	Horizontal blank back porch. The unit is pixel.
u16Hfb	Horizontal blank front porch. The unit is pixel.
u16Hmid	Valid pixel value of bottom field vertical synchronization
u16Bvact	Vertical active region of the bottom field, which is valid in interlaced mode. The unit is row.
u16Bvbb	Vertical blank back porch of the bottom field, which is valid in interlaced mode. The unit is row.
u16Bvfb	Vertical blank front porch of the bottom field, which is valid in interlaced mode. The unit is row.
u16Hpw	Width of the horizontal sync signal. The unit is pixel.
u16Vpw	Width of the vertical sync signal. The unit is row.
bl dv	Polarity of the data validity signal, which is active high by default and cannot be adjusted currently
blhs	Polarity of the horizontal validity signal. <b>0</b> indicates active high, and <b>1</b> indicates active low.
blvs	Polarity of the vertical validity signal. <b>0</b> indicates active high, and <b>1</b> indicates active low.

For details about how to configure the user timings, see the related LCD document. Note that the unit of each value is consistent with the requirement.

## 7.2.3 Confirming CRG Configurations

Besides the user timings, you need to configure the VDP and LCD CRG clock registers to output correct clocks to the LCD.

**Table 7-3** VDP CRG register PERI\_CRG13 (0x20030034) of Hi3518E V200

Bits	Name	Description	Suggestion
[31:21]	reserved	Reserved	-
[20]	lcd_cksel	LCD frequency-division clock select 0: divide-by-3 clock of the LCD 1: divide-by-4 clock of the LCD	For details, see the description of the configuration of the frequency divider in the following section.
[19]	reserved	Reserved	-
[18]	lcd_cken	LCD working clock gating	1



Bits	Name	Description	Suggestion
		0: disabled 1: enabled	
[17]	hd_lcd_cksel	HD_LCD working clock select 0: The HD frequency-division clock works as the clock of the DHD1 channel. 1: The LCD frequency-division clock works as the clock of the DHD1 channel.	1
[16:15]	reserved	Reserved	
[14]	vo_out_cksel	VO OUT_CLK frequency select 0: The HD clock functions as the echo clock of VDP outputs. 1: The LCD clock functions as the echo clock of VDP outputs.	1
[13:12]	hd_cksel	DHD frequency Bit[12]: 0: HD clock frequency 27 MHz 1: HD clock frequency 74.25 MHz Bit[13]: 0: HD frequency-division clock, frequency multiplication of 2 1: HD frequency-division clock, frequency multiplication of 1	-
[11:10]	reserved	Reserved	-
[9]	vou_hd_cken	VOU HD clock gating 0: disabled 1: enabled	1
[8]	vou_acken	VOU AXI bus clock gating 0: disabled 1: enabled	1
[7]	vo_out_cken	VO_CLKOUT clock gating 0: disabled 1: enabled	1
[6]	vou_pcken	VOU APB clock gating 0: disabled 1: enabled	1
[5]	vou_ppc_cken	VO PPC clock gating 0: disabled	1



Bits	Name	Description	Suggestion
		1: enabled	
[4]	vou_cfg_cken	VO CFG (internal configuration) clock gating 0: disabled 1: enabled	1
[3]	reserved	Reserved	-
[2]	vo_out_pctrl	Phase of the VOU HD output echo clock, reversed by default 0: normal 1: reversed	Adjust the value based on the actual effect.
[1]	reserved	Reserved	-
[0]	vo_srst_req	VOU soft reset 0: reset deasserted 1: reset	0

**Table 7-4** LCD CRG register PERI\_CRG26 (0x20030068) of Hi3518E V200

Bits	Name	Description	Suggestion
[31:27]	reserved	Reserved	-
[26:0]	lcd_mclk_div	LCD clock, configurable	Assume that the target frequency is X (MHz). $lcd\_mclk\_div = (X/594) \times 2^{27}$

**Table 7-5** VDP CRG register PERI\_CRG17 (0x12010044) of Hi3519 V100

Bits	Name	Name	Suggestion
[31:17]	reserved	Reserved	-
[16:14]	vo_out_cksel	VO_OUT_CLK frequency 000: 148.5 MHz 001: 74.25 MHz 010: 37.125 MHz 011: 107 MHz 100: 54 MHz 101: 27 MHz 110: LCD frequency divider clock 111: reserved	110



Bits	Name	Name	Suggestion
[13:12]	hd_div_mode	VO_OUT_CLK and DHD channel clock frequency divider 00: not divided 01: divided by 2 10: divided by 3 11: divided by 4	For details, see the description of the configuration of the frequency divider in the following section.
[11]	vdac_cken	VDACH clock gating 0: disabled 1: enabled	1
[10]	vou_sd_cken	VOU SD clock gating 0: disabled 1: enabled	1
[9]	vou_hd_cken	VOU HD clock gating 0: disabled 1: enabled	1
[8]	vo_out_cken	VO_CLKOUT clock gating 0: disabled 1: enabled	1
[7]	vou_ackn	VOU AXI bus clock gating 0: disabled 1: enabled	1
[6]	vou_pcken	VOU APB clock gating 0: disabled 1: enabled	1
[5]	vou_ppc_cken	VOU PPC clock gating 0: disabled 1: enabled	1
[4]	vou_cfg_cken	VOU CFG (internal configuration) clock gating 0: disabled 1: enabled	1
[3]	vdac_pctrl	VDAC clock phase 0: normal 1: reversed	1
[2]	vo_out_pctrl	Phase of the VOU HD output echo clock, reversed by default	Adjust the value based on the actual effect.



Bits	Name	Name	Suggestion
		0: normal 1: reversed	
[1]	reserved	Reserved	-
[0]	vo_srst_req	VOU soft reset 0: reset deasserted 1: reset	0

**Table 7-6** LCD CRG register PERI\_CRG18 (0x12010048) of Hi3519 V100

Bits	Name	Description	Suggestion
[31:28]	reserved	Reserved	-
[27]	lcd_cken	LCD frequency divider clock gating 0: disabled 1: enabled	1
[26:0]	lcd_mclk_div	LCD frequency division clock, configurable	Assume that the target frequency is X (MHz). $lcd\_mclk\_div = (X/1188) \times 2^{27}$

**Table 7-7** VDP CRG register PERI\_CRG17 (0x12010044) of Hi3519 V101

Bits	Name	Description	Suggestion
[31:18]	reserved	Reserved	-
[17]	vdp_core_clkssel	VDP working clock select 0: 300 MHz 1: 198 MHz	0
[16:14]	vdp_out_clkssel	VDP output clock (chip output clock) 000: 148.5 MHz 001: 74.25 MHz 010: 37.125 MHz 011: 107 MHz 100: 54 MHz 101: 27 MHz 110: LCD frequency divider clock 111: reserved	110



Bits	Name	Description	Suggestion
[13:12]	hd_div_mode	VDP output clock (chip output clock) and HD channel clock frequency divider 00: not divided 01: divided by 2 10: divided by 3 11: divided by 4	For details, see the description of the configuration of the frequency divider in the following section.
[11]	vdac_clken	VDAC clock gating 0: disabled 1: enabled	1
[10]	vdp_sd_clken	VDP SD clock gating 0: disabled 1: enabled	1
[9]	vdp_hd_clken	VDP HD clock gating 0: disabled 1: enabled	1
[8]	vdp_out_clken	VDP output clock (chip output clock) gating 0: disabled 1: enabled	1
[7]	vdp_aclken	VDP AXI bus clock gating 0: disabled 1: enabled	1
[6]	vdp_pclken	VDP APB clock gating 0: disabled 1: enabled	1
[5]	vdp_core_clken	VDP working clock gating 0: disabled 1: enabled	1
[4]	vdp_cfg_clken	VDP CFG (internal configuration) clock gating 0: disabled 1: enabled	1
[3]	vdac_pctrl	VDAC clock phase 0: normal 1: reversed	1





Bits	Name	Description	Suggestion
[2]	vdp_out_pctrl	Phase of the VDP output clock (chip output clock), reversed by default 0: normal 1: reversed	Adjust the value based on the actual effect.
[1]	reserved	Reserved	-
[0]	vdp_srst_req	VDP soft reset 0: reset deasserted 1: reset	0

**Table 7-8** LCD CRG register PERI\_CRG18 (0x12010048) of Hi3519 V101

Bits	Name	Description	Suggestion
[31:28]	reserved	Reserved	-
[27]	lcd_clken	LCD frequency divider clock gating 0: disabled 1: enabled	1
[26:0]	lcd_mclk_div	LCD frequency division clock, configurable	Assume that the target frequency is X (MHz). $\text{lcd\_mclk\_div} = (X/1188) \times 2^{27}$

The VDP CRG configurations of each chip involve the configuration of a frequency divider, which is the ratio of the VDP output clock (chip output clock) to the HD channel clock. The HD channel outputs one pixel during one clock beat. However, the LCD requires multiple clock beats to output a pixel. For example:

- If the required data sequence of an 8-bit serial LCD is the RGB sequence, that is, three clock beats are required to transfer the R, G, and B data for one pixel, the frequency divider is 3.
- For a 16-bit serial LCD, one clock beat is required to output one pixel. In this case, the frequency is not divided.

## 7.2.4 Confirming the Configuration of the LCD\_CTRL Register

When debugging the LCD, you also need to configure a VDP register LCD\_CTRL with the offset address of 0xD400.



**Table 7-9** LCD\_CTRL register configuration

Bits	Name	Description	Suggestion
[31]	reserved	Reserved	-
[30]	reserved	Reserved	-
[29]	lcd_serial_mode	LCD interface mode 0: parallel mode 1: serial mode	Set the mode based on the LCD interface type.
[28]	lcd_serial_perd	Number of cycles of the LCD serial output single-pixel clock 0: three cycles 1: four cycles	Set this bit when the serial output mode is used. The value is consistent with the frequency divider described in the preceding section.
[27]	lcd_parallel_order	LCD parallel output sequence 0: RGB 1: BGR	Set the parallel output sequence based on the hardware connection. Ignore this bit if the serial mode is used.
[26]	lcd_data_inv	LCD output bit phase inversion 0: from upper bits to lower bits (bit 15 to bit 0 or bit 23 to bit 0) 1: from lower bits to upper bits (bit 0 to bit 15 or bit 0 to bit 23)	Set this bit based on the hardware connection.
[25]	lcd_parallel_mode	Bit width mode of LCD parallel output 0: RGB565 1: RGB888	Set this bit to <b>0</b> for the 16-bit parallel output, and set it to <b>1</b> for the 24-bit parallel output. Ignore this bit when the serial output is used.
[24]	reserved	Reserved	-
[23:0]	reserved	Reserved	-



# 8 Others

## 8.1 Dynamic Library

### 8.1.1 What Do I Do If the Dynamic Libraries Cannot Be Used When the Application Is Statically Compiled?

[Symptom]

The file systems and executable programs of customer A are statically compiled. As a result, the dynamic libraries in the SDK cannot be used.

[Cause Analysis]

The current arm-linux-gcc version supports the static compilation, dynamic compilation, and semi-static compilation.

- In static compilation mode (compilation options: **-static**, **-pthread**, **-lrt**, and **-ldl**), the **libc**, **libpthread**, **librt**, and **libdl** libraries are all compiled to the executable program. The static compilation does not depend on any system dynamic library and is implemented independently. However, the dynamic libraries cannot be used in this mode.
- In dynamic compilation mode (common compilation), the system dynamic libraries under **/lib** are linked. Therefore, the compiled program depends on the system dynamic libraries. The advantage of the dynamic compilation is that the system dynamic libraries can be shared by multiple executable programs such as the BusyBox and Himount under **/bin**.
- In semi-static compilation mode (compilation options: **-static-libgcc**, **-static-libstdc++**, **-L**, **-pthread**, **-lrt**, and **-ldl**), the **libgcc** and **libstdc++** libraries are compiled to the executable program. Other system libraries still depend on the system dynamic libraries. In this mode, the dynamic libraries can be used, but the **libc**, **libpthread**, **librt**, and **libdl** files still need to be placed under the system directory.

[Solution]

Adopt the dynamic compilation and place the system files that the dynamic libraries depend on (including **ld-uClibc.so.0**, **libc.so.0**, **libpthread.so.0**, **librt.so.0**, and **libdl.so.0**) under **/lib**.

### 8.1.2 What Do I Do If a Redefinition Error Occurs When libupvqe.a and libdnvqe.a Are Used for Dynamic Compilation?

[Symptom]



A redefinition error occurred when customer B compiled the audio component libraries **libupvqe.a** and **libdnvqe.a** into a dynamic library. The compilation statement is as follows:

```
$(CC) -shared -o $@ -L. -Wl,--whole-archive libupvqe.a libdnvqe.a -Wl,--no-whole-archive
```

[Cause Analysis]

**libupvqe.a** and **libdnvqe.a** share some functional modules to implement code reuse and modularization, and save the file space when ELF files are generated after compilation.

The static libraries can be compiled into the dynamic library in any of the following ways:

- Directly use the **-l** compilation option. The compilation statement is as follows:

```
$(CC) -shared -o libshare.so -L. -lupvqe -ldnvqe
```

This is a link compilation method. The function symbols of the static libraries are not linked to the **libshare.so** library generated after compilation.

- Use the **-Wl,--whole-archive** compilation option. The compilation statement is as follows:

```
$(CC) -shared -o $@ -L. -Wl,--whole-archive libupvqe.a libdnvqe.a -Wl,--no-whole-archive
```

In this method, the function symbols of the static libraries are compiled to **libshare.so**. However, **libupvqe.a** and **libdnvqe.a** cannot have functions with the same name.

- Split the **.a** files into multiple **.o** files respectively, and then compile the **.o** files into the **.so** file. The compilation statement is as follows:

```
LIB_PATH = ./
EXTERN_OBJ_DIR = ./EXTERN_OBJ
LIBUPVQE_NAME = libupvqe.a
LIBDNVQE_NAME = libdnvqe.a
EXTERN_OBJ = $(EXTERN_OBJ_DIR)/*.o
all: pre_mk $(TARGET) pre_clr
pre_mk:
    @mkdir -p $(EXTERN_OBJ_DIR);
    @cp $(LIB_PATH)$(LIBUPVQE_NAME) $(EXTERN_OBJ_DIR);
    @cd $(EXTERN_OBJ_DIR); $(AR) -x $(LIBUPVQE_NAME);
    @cp $(LIB_PATH)$(LIBDNVQE_NAME) $(EXTERN_OBJ_DIR);
    @cd $(EXTERN_OBJ_DIR); $(AR) -x $(LIBDNVQE_NAME);

$(TARGET):
    #$(CC) -shared -o $@ -L. libupvqe.so libdnvqe.so
    $(CC) -shared -o $@ -L. $(EXTERN_OBJ)

pre_clr:
    @rm -rf $(EXTERN_OBJ_DIR);
```



**NOTE**

In this method, **libupvqe.a** and **libdnvqe.a** are split into .o files respectively, and then the .o files are compiled into the .so file. The function symbols of the static libraries are compiled to the .so file, and no function name conflict occurs.

[Solution]

Customer B used the second compilation method. A redefinition error occurred because **libupvqe.a** and **libdnvqe.a** have functions with the same name. To solve this issue, customer B can use the first or third compilation method to generate the **libshare.so** file.