HDMI

Development Reference

Issue **01**

Date **2018-05-15**

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

About This Document

Purpose

The embedded high-definition multimedia interface (HDMI) module of the Hi35*xx* supports the video HDMI output.

NOTE

Unless otherwise specified, this document applies to Hi3521D V100, Hi3520D V400, and Hi3536C V100.

Related Version

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

Product Name	Version
Hi3531D	V100
Hi3521D	V100
Hi3536C	V100
Hi3536D	V100
Hi3520D	V400

Intended Audience

This document is intended for:

- Technical support engineers
- Software development engineers

Change History

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

Issue 01(2018-05-15)

This issue is the first official release.

Issue 00B09 (2018-01-20)

This issue is the ninth draft release, which incorporates the following changes:

Chapter 5 is modified.

Issue 00B08 (2017-11-30)

This issue is the eighth draft release, which incorporates the following changes:

Chapter 2 API Reference

The descriptions in the **Note** field of HI_MPI_HDMI_GetInfoFrame are updated.

Chapter 3 Data Structures

HI_HDMI_AUDIO_SPEAKER_E is added. The descriptions in the **Syntax** field of HI_HDMI_CONTENT_TYPE_E are updated.

Issue 00B07 (2017-11-20)

This issue is the seventh draft release, which incorporates the following changes:

Chapter 2 API Reference

The descriptions in the **Note** field of HI_MPI_HDMI_SetDeepColor, HI_MPI_HDMI_SetInfoFrame, and HI_MPI_HDMI_GetInfoFrame are updated.

Chapter 3 Data Structures

The descriptions in the **Note** fields of the following interfaces are updated: HI_HDMI_DEEP_COLOR_E, HI_HDMI_BIT_DEPTH_E, HI_HDMI_INFOFRAME_TYPE_E, HI_HDMI_PIC_ASPECT_RATIO_E, HI_HDMI_PIXEL_REPETITION_E, HI_HDMI_YCC_QUAN_RAGE_E, HI_HDMI_LFE_PLAYBACK_LEVEL_E. The descriptions in the **Syntax** and **Note** fields of HI_HDMI_SAMPLE_RATE_E are updated.

In chapter 4 "Error Codes", the descriptions of the error codes of the HDMI APIs are updated.

Issue 00B06 (2017-10-18)

This issue is the sixth draft release, which incorporates the following changes:

Chapter 2 API Reference

The descriptions in the **Note** field of HI_MPI_HDMI_DeInit to HI_MPI_HDMI_GetInfoFrame are updated.

Chapter 3 Data Structures

The descriptions in the **Member** field of HI_HDMI_VIDEO_FMT_E and HI_HDMI_DEEP_COLOR_E are updated.

Chapter 4 Error Codes

The descriptions of the error codes of the HDMI APIs are updated.

Issue 00B05 (2017-09-08)

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

The description of the Hi3536D V100 is added.

Chapter 2 API Reference

The descriptions in the **Note** field of HI_MPI_HDMI_SetDeepColor are updated.

Chapter 4 Error Codes

The descriptions of the error codes of the HDMI APIs are updated.

Issue 00B04 (2017-08-15)

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

Chapter 3 Data Structures

The descriptions in the **Member** field of HI_HDMI_ATTR_S are updated.

Issue 00B03 (2017-07-26)

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

Chapter 2 API Reference

The descriptions in the **Example** field of HI_MPI_HDMI_Init is updated. The descriptions in the **Note** fields of HI_MPI_HDMI_SetAttr, HI_MPI_HDMI_Start, HI_MPI_HDMI_Stop, HI_MPI_HDMI_SetAVMute, and HI_MPI_HDMI_SetDeepColor are updated.

Chapter 3 Data Structures

The descriptions in the **Note** field of HI_HDMI_DEEP_COLOR_E are updated.

Chapter 5 Proc Debugging Information

The descriptions in the "HDMI Audio Debugging Information" are updated.

Issue 00B02 (2017-05-27)

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

Chapter 2 API Reference

Thedescriptions in the **Note** fields of HI_MPI_HDMI_DeInit, HI_MPI_HDMI_Close, HI_MPI_HDMI_GetSinkCapability and HI_MPI_HDMI_SetAVMute are updated.

Chapter 3 Data Structures

HI_HDMI_ATTR_S is updated. The descriptions in the **Note** fields of the following interfaces are updated: HI_HDMI_SND_INTERFACE_E, HI_HDMI_SAMPLE_RATE_E, I_HDMI_BIT_DEPTH_E, HI_HDMI_INFOFRAME_TYPE_E, HI_HDMI_EXT_COLORIMETRY_E, HI_HDMI_PIC_ASPECT_RATIO_E, HI_HDMI_ACT_ASPECT_RATIO_E, HI_HDMI_RGB_QUAN_RAGE_E, HI_HDMI_PIXEL_REPETITION_E, HI_HDMI_YCC_QUAN_RAGE_E, and HI_HDMI_LFE_PLAYBACK_LEVEL_E.

Issue 00B01 (2017-04-10)

This issue is the first draft release.

Contents

Tables

1 Overview

1.1 Concepts

The audio of the HDMI cannot be output independently, which depends on the video output. The HDMI clock is derived from the VO clock. Therefore, the interface invocation sequence must be as follows: enable the VO, call the HDMI, and configure the audio and video (AV) output. The embedded HDMI does not support the high-bandwidth digital content protection (HDCP) and CEC functions.

1.2 Specifications

The Hi3531D V100 supports the HDMI 2.0 specifications (compatible with HDMI1.4), and the Hi3521D V100/Hi3536C V100/Hi3536D V100/Hi3520D V100 supports only the HDMI 1.4 specifications.

CAUTION

All chips use the same API interfaces. However, the parameters related to HDMI 2.0 are not applicable to the chip only supporting the HDMI 1.4 APIs.

Unless otherwise specified, the products mentioned in this document support only the HDMI 1.4 specifications.

Table 1-1 lists the HDMI specifications.

Table 1-1 Specifications of the chip HDMI

Product Name	HDMI Specifications
Hi3531D V100	HDMI 2.0
Hi3521D V100	HDMI 1.4
Hi3536C V100	HDMI 1.4
Hi3536D V100	HDMI 1.4

Product Name	HDMI Specifications
Hi3520D V400	HDMI 1.4

2 API Reference

The HDMI provides the following MPIs:

- HI_MPI_HDMI_Init: Initializes the HDMI.
- HI_MPI_HDMI_DeInit: Deinitializes the HDMI.
- HI_MPI_HDMI_Open: Enables the HDMI.
- HI_MPI_HDMI_Close: Disables the HDMI.
- HI_MPI_HDMI_GetSinkCapability: Obtains the HDMI sink capability.
- HI_MPI_HDMI_SetAttr: Sets the HDMI attributes.
- HI_MPI_HDMI_GetAttr: Obtains the HDMI attributes.
- HI_MPI_HDMI_Start: Starts the HDMI output.
- HI_MPI_HDMI_Stop: Stops the HDMI output.
- HI_MPI_HDMI_SetAVMute: Sets the AV mute function of the HDMI.
- HI_MPI_HDMI_Force_GetEDID: Obtains the raw extended display identification data (EDID) information about the HDMI.
- HI_MPI_HDMI_RegCallbackFunc: Registers the callback function of an HDMI event.
- HI_MPI_HDMI_UnRegCallbackFunc: Deregisters the callback function of an HDMI event.
- HI_MPI_HDMI_SetDeepColor: Sets the deep color mode.
- HI_MPI_HDMI_GetDeepColor: Obtains the deep color mode.
- HI_MPI_HDMI_SetInfoFrame: Sets the HDMI information frame.
- HI_MPI_HDMI_GetInfoFrame: Obtains the HDMI information frame.

HI_MPI_HDMI_Init

[Description]

Initializes the HDMI.

[Syntax]

```
HI_S32 HI_MPI_HDMI_Init(HI_VOID);
```

[Parameter]

None

[Return Value]

Return Value	Description
0	Success
Other values	Failure. The value is an error code. For details, see section 4 "Error Codes."

[Requirement]

- Header files: mpi_hdmi.h, hi_comm_hdmi.h
- Library file: libhdmi.a

[Note]

By default, the HDMI provides the event handling function in the kernel state. If users do not need to specially respond to the events, the function pointer to the initialization parameter structure can be set to NULL.

[Example]

Note: In the following example, the MPI return value is not checked. However, you are advised to check the return value and take corresponding measures when using the APIs in this document.

```
HI_HDMI_ATTR_S     stAttr;

/*Initialize the HDMI.*/
HI_MPI_HDMI_Init();

/*Enable the HDMI.*/
HI_MPI_HDMI_Open(HI_HDMI_ID_0);

/*Obtain the HDMI attributes.*/
HI_MPI_HDMI_GetAttr(HI_HDMI_ID_0, &stAttr);
/*Set the HDMI attributes.*/
stAttr.bEnableHdmi =     HI_TRUE;
stAttr.bEnableVideo = HI_TRUE;
stAttr.enVideoFmt = HI_HDMI_VIDEO_FMT_720P_60;

stAttr.enVidOutMode = HI_HDMI_VIDEO_MODE_YCBCR444;
stAttr.enDeepColorMode = HI_HDMI_DEEP_COLOR_OFF
stAttr.bxvYCCMode = HI_FALSE;

stAttr.bEnableAudio = HI_TRUE;
stAttr.enSoundIntf = HI_HDMI_SND_INTERFACE_I2S;
stAttr.bIsMultiChannel = HI_FALSE;
stAttr.enSampleRate        = HI_HDMI_SAMPLE_RATE_48K;
stAttr.u8DownSampleParm    = 0;
```

```
stAttr.enBitDepth = HI_HDMI_BIT_DEPTH_16;
stAttr.u8I2SCtlVbit        = 0;

stAttr.bEnableAviInfoFrame = HI_TRUE;
stAttr.bEnableAudInfoFrame = HI_TRUE;
stAttr.bEnableSpdInfoFrame = HI_FALSE;
stAttr.bEnableMpegInfoFrame = HI_FALSE;

stAttr.bDebugFlag = HI_FALSE;
stAttr.bHDCPEnable = HI_FALSE;

stAttr.b3DEnable = HI_FALSE;
stAttr.u83DParam           = 9;
stAttr.enDefaultMode       = HI_HDMI_FORCE_HDMI;
HI_MPI_HDMI_SetAttr(0, &stAttr);

/*Start the HDMI.*/
HI_MPI_HDMI_Start(HI_HDMI_ID_0);

/*The following describes the exit process after the HDMI is used.*/
/*Stop the HDMI.*/
HI_MPI_HDMI_Stop(HI_HDMI_ID_0);

/*Disable the HDMI.*/
HI_MPI_HDMI_Close(HI_HDMI_ID_0);

/*Deinitialize the HDMI.*/
HI_MPI_HDMI_DeInit();
```

[See Also]

HI_MPI_HDMI_DeInit

HI_MPI_HDMI_DeInit

[Description]

Deinitializes the HDMI.

[Syntax]

```
HI_S32 HI_MPI_HDMI_DeInit(HI_VOID);
```

[Parameter]

None

[Return Value]

Return Value	Description
0	Success
Other values	Failure. The value is an error code. For details, see section 4 "Error Codes."

[Requirement]

- Header files: mpi_hdmi.h, hi_comm_hdmi.h
- Library file: libhdmi.a

[Note]

- If the initialization is successful, this MPI must be called when the program exits due to exceptions.
- Success is returned if the HDMI that has not been initialized is deinitialized or the HDMI is repeatedly deinitialized.

[Example]

See the examples of HI_MPI_HDMI_Init and HI_MPI_HDMI_RegCallbackFunc.

[See Also]

HI_MPI_HDMI_Init

HI_MPI_HDMI_Open

[Description]

Enables the HDMI.

[Syntax]

```
HI_S32 HI_MPI_HDMI_Open(HI_HDMI_ID_E enHdmi);
```

[Parameter]

Parameter	Description	Input/Output
enHdmi	HDMI ID The value is 0. For details, see HI_HDMI_ID_E, which is reserved for extension of HDMI devices.	Input

[Return Value]

Return Value	Description
0	Success
Other values	Failure. The value is an error code. For details, see section 4 "Error Codes."

[Requirement]

- Header files: mpi_hdmi.h, hi_comm_hdmi.h
- Library file: libhdmi.a

[Note]

- Ensure that the HDMI is initialized before HI_MPI_HDMI_Open is called. Otherwise, HI_ERR_HDMI_NOT_INIT is returned.
- The error code HI_SUCCESS is returned if the HDMI is enabled repeatedly.

[Example]

See the examples of HI_MPI_HDMI_Init and HI_MPI_HDMI_RegCallbackFunc.

[See Also]

HI_MPI_HDMI_Close

HI_MPI_HDMI_Close

[Description]

Disables the HDMI.

[Syntax]

```
HI_S32 HI_MPI_HDMI_Close(HI_HDMI_ID_E enHdmi);
```

[Parameter]

Parameter	Description	Input/Output
enHdmi	HDMI ID The value is 0.	Input

[Return Value]

Return Value	Description
0	Success
Other values	Failure. The value is an error code. For details, see section 4 "Error Codes."

[Requirement]

- Header files: mpi_hdmi.h, hi_comm_hdmi.h
- Library file: libhdmi.a

[Note]

- Before calling this API, ensure that the HDMI has been opened. Otherwise, HI_ERR_HDMI_DEV_NOT_OPEN is returned.

- After the HDMI is opened successfully, this MPI together with HI_MPI_HDMI_DeInit must be called to release HDMI resources when the application exits due to exceptions.

[Example]

See the examples of HI_MPI_HDMI_Init and HI_MPI_HDMI_RegCallbackFunc.

[See Also]

HI_MPI_HDMI_Open

HI_MPI_HDMI_SetAttr

[Description]

Sets the HDMI attributes.

[Syntax]

```
HI_S32 HI_MPI_HDMI_SetAttr(HI_HDMI_ID_E enHdmi, HI_HDMI_ATTR_S *pstAttr);
```

[Parameter]

Parameter	Description	Input/Output
enHdmi	HDMI ID The value is 0.	Input
pstAttr	Pointer to the HDMI attribute structure	Input

[Return Value]

Return Value	Description
0	Success
Other values	Failure. The value is an error code. For details, see section 4 "Error Codes."

[Requirement]

- Header files: mpi_hdmi.h, hi_comm_hdmi.h
- Library file: libhdmi.a

[Note]

- Before calling this API, ensure that the HDMI has been opened. Otherwise, HI_ERR_HDMI_DEV_NOT_OPEN is returned.
- Set the HDMI attributes before starting the HDMI. If the HDMI is started, stop it, set the attributes, and then restart the HDMI. The behavior that does not comply with this process is undefined.
- Some HDMI attributes are not supported by the Hi35xx currently. For details, see HI_HDMI_ATTR_S.

- If only some attributes need to be configured, obtain these attributes, assign values to them, and then configure them.
- The EDID of some display devices may be inaccurate. For example, some display devices have the display capacity of 4kp60. However, the EDID shows that they do not support this capacity and the SCDC. When this API is called to set the 4kp60 outputs, to prevent the compatibility issues, the 4kp60 outputs will not be supported forcibly by the HDMI driver, and pictures cannot be properly displayed on the TV set. The prompt "sink is not support scdc" is displayed in the serial port. For similar cases, if the display device has the 4kp60 output capacity, you can set **bAuthMode** to **HI_TRUE**. In this case, the HDMI driver will forcibly implement the 4kp60 outputs. However, various incompatibility risks may occur, such as artifacts, screen flickers, and even system crash.
- This MPI is a synchronous interface. During the invocation process, the driver may interact with the display device through the SCDC. This process takes a long time (usually about 3~5s).
- The Hi3536D V100 does not support the 36-bit HDMI deep color mode. HI_ERR_HDMI_FEATURE_NO_SUPPORT is returned when the HDMI deep color mode is set to HI_HDMI_DEEP_COLOR_36BIT.
- When **enVidOutMode** is HI_HDMI_VIDEO_MODE_YCBCR422, HI_ERR_HDMI_INVALID_PARA is returned if you set HI_HDMI_DEEP_COLOR_30BIT or HI_HDMI_DEEP_COLOR_36BIT.
- When the DVI output mode is used, the HDMI driver forcibly changes the configurations of ColorSpace into HI_HDMI_VIDEO_MODE_RGB444 and DeepColor into HI_HDMI_DEEP_COLOR_OFF. In this case, other configurations are invalid.

Note: When **bAuthMode** is set to **HI_TRUE**, the driver forces SCDC operations. If the display device neither supports HDMI cables nor has HDMI cables inserted, this MPI may take a longer time.

[Example]

See the examples of HI_MPI_HDMI_Init and HI_MPI_HDMI_RegCallbackFunc.

[See Also]

HI_MPI_HDMI_GetAttr

HI_MPI_HDMI_GetAttr

[Description]

Obtains the HDMI attributes.

[Syntax]

```
HI_S32 HI_MPI_HDMI_GetAttr(HI_HDMI_ID_E enHdmi, HI_HDMI_ATTR_S *pstAttr);
```

[Parameter]

Parameter	Description	Input/Output
enHdmi	HDMI ID The value is 0.	Input
pstAttr	Pointer to the HDMI attribute structure	Output

[Return Value]

Return Value	Description
0	Success
Other values	Failure. The value is an error code. For details, see section 4 "Error Codes."

[Requirement]

- Header files: mpi_hdmi.h, hi_comm_hdmi.h
- Library file: libhdmi.a

[Note]

- Before calling this API, ensure that the HDMI has been opened. Otherwise, HI_ERR_HDMI_DEV_NOT_OPEN is returned.
- If you need to configure some attributes, obtain the attributes, assign values to them before configuration.

[Example]

See the examples of HI_MPI_HDMI_Init and HI_MPI_HDMI_RegCallbackFunc.

[See Also]

HI_MPI_HDMI_SetAttr

HI_MPI_HDMI_Start

[Description]

Starts the HDMI output.

[Syntax]

```
HI_S32 HI_MPI_HDMI_Start(HI_HDMI_ID_E enHdmi);
```

[Parameter]

Parameter	Description	Input/Output
enHdmi	HDMI ID The value is 0.	Input

[Return Value]

Return Value	Description
0	Success
Other values	Failure. The value is an error code. For details, see section 4 "Error Codes."

[Requirement]

- Header files: mpi_hdmi.h, hi_comm_hdmi.h
- Library file: libhdmi.a

[Note]

- Before calling this API, ensure that the HDMI has been opened. Otherwise, HI_ERR_HDMI_DEV_NOT_OPEN is returned.
- This MPI sends the instruction to clear AVMute data packets before enabling the HDMI output.
- When no external HDMI device is connected, enabling the HDMI output will cause extra power consumption. It is advised to use the registered callback function to detect HDMI hot-plug events. Enable the HDMI output when hot plug is detected. Otherwise, keep the HDMI output disabled. For some display devices that do not have the hot-plug event, it is recommended that the HDMI output be enabled when needed.

[Example]

See the examples of HI_MPI_HDMI_Init and HI_MPI_HDMI_RegCallbackFunc.

[See Also]

HI_MPI_HDMI_Stop

HI_MPI_HDMI_Stop

[Description]

Stops the HDMI output.

[Syntax]

```
HI_S32 HI_MPI_HDMI_Stop(HI_HDMI_ID_E enHdmi);
```

[Parameter]

Parameter	Description	Input/Output
enHdmi	HDMI ID The value is 0.	Input

[Return Value]

Return Value	Description
0	Success
Other values	Failure. The value is an error code. For details, see section 4 "Error Codes."

[Requirement]

- Header files: mpi_hdmi.h, hi_comm_hdmi.h
- Library file: libhdmi.a

[Note]

- This MPI sends AVMute data packets before disabling the HDMI output.
- Before calling this API, ensure that the HDMI has been opened. Otherwise, HI_ERR_HDMI_DEV_NOT_OPEN is returned.

[Example]

See the examples of HI_MPI_HDMI_Init and HI_MPI_HDMI_RegCallbackFunc.

[See Also]

HI_MPI_HDMI_Start

HI_MPI_HDMI_GetSinkCapability

[Description]

Obtains the HDMI sink capability.

[Syntax]

```
HI_S32 HI_MPI_HDMI_GetSinkCapability(HI_HDMI_ID_E enHdmi,
HI_HDMI_SINK_CAPABILITY_S*pstSinkCap);
```

[Parameter]

Parameter	Description	Input/Output
enHdmi	HDMI ID The value is 0.	Input
pstSinkCap	Pointer to the HDMI sink capability structure	Output

[Return Value]

Return Value	Description
0	Success
Other values	Failure. The value is an error code. For details, see section 4 "Error Codes."

[Requirement]

- Header files: mpi_hdmi.h, hi_comm_hdmi.h
- Library file: libhdmi.a

[Note]

- Before calling this API, ensure that the HDMI has been opened. Otherwise, HI_ERR_HDMI_DEV_NOT_OPEN is returned.

- Call this MPI after the HDMI is started and the HDMI cable is inserted.
- The driver reads the EDID of the display device after the application calls HI_MPI_HDMI_Open. The read and parsing process takes a certain period of time. Therefore, if the application registers the callback of the HDMI event, you are advised to call the application after cable insertion is detected. If the application does not register the callback, call this MPI one to two seconds after you open HDMI. If this MPI is called without delay, it may fail to obtain data.

[Example]

```
HI_HDMI_EDID_S stEdidData;
HI_HDMI_SINK_CAPABILITY_S stSinkCap;


/*Initialize the HDMI.*/
HI_MPI_HDMI_Init();

/*Enable the HDMI.*/
HI_MPI_HDMI_Open(HI_HDMI_ID_0);
/*Set the attributes.*/

…
/* Start the HDMI */
HI_MPI_HDMI_Start(HI_HDMI_ID_0);
…

sleep(2);
/* Obtain EDID */
HI_MPI_HDMI_Force_GetEDID(0, &stEdidData);


/*Obtain the capability set.*/
HI_MPI_HDMI_GetSinkCapability(HI_HDMI_ID_0,&stSinkCap);
```

[See Also]

HI_MPI_HDMI_Force_GetEDID

HI_MPI_HDMI_SetAVMute

[Description]

Sets the AV mute function of the HDMI.

[Syntax]

```
HI_S32 HI_MPI_HDMI_SetAVMute(HI_HDMI_ID_E enHdmi, HI_BOOL bAvMute);
```

[Parameter]

Parameter	Description	Input/Output
enHdmi	HDMI ID The value is 0.	Input
bAvMute	Whether to set the AV mute function of the HDMI Value range: • HI_TRUE: Set the AV mute function. • HI_FALSE: Disable the AV mute function.	Input

[Return Value]

Return Value	Description
0	Success
Other values	Failure. The value is an error code. For details, see section 4 "Error Codes."

[Requirement]

- Header files: mpi_hdmi.h, hi_comm_hdmi.h
- Library file: libhdmi.a

[Note]

- Before calling this API, ensure that the HDMI has been opened. Otherwise, HI_ERR_HDMI_DEV_NOT_OPEN is returned.
- Since the HI_MPI_HDMI_Start or HI_MPI_HDMI_Stop interface contains operations for the AVMute. If AVMute enabling is set through this interface, the AVMute will be cleared when a hot-plug event is generated or the HI_MPI_HDMI_Start interface is called in this case. Likewise, if the AVMute is disabled, it will be enabled if the HI_MPI_HDMI_Stop interface is called. Therefore, you are not advised to call this interface again.
- Some incompatible TVs cannot not respond to the AV mute function properly. In this case, disable HDMI output by calling HI_MPI_HDMI_Stop before setting the HDMI attributes. Then, enable HDMI output by calling HI_MPI_HDMI_Start after setting the HDMI attributes.

[Example]

```
/*Initialize the HDMI.*/
HI_MPI_HDMI_Init();

/*Enable the HDMI.*/
HI_MPI_HDMI_Open(HI_HDMI_ID_0);

/*Start the HDMI.*/
HI_MPI_HDMI_Start(HI_HDMI_ID_0);
```

```
…
/*Set AVMute.*/
HI_MPI_HDMI_SetAVMute(HI_HDMI_ID_0,HI_TRUE);
```

[See Also]

None

HI_MPI_HDMI_Force_GetEDID

[Description]

Obtains the raw EDID information about the HDMI.

[Syntax]

```
HI_S32 HI_MPI_HDMI_Force_GetEDID(HI_HDMI_ID_E enHdmi, HI_HDMI_EDID_S
*pstEdidData);
```

[Parameter]

Parameter	Description	Input/Output
enHdmi	HDMI ID The value is 0.	Input
pstEdidData	EDID information about the HDMI	Output

[Return Value]

Return Value	Description
0	Success
Other values	Failure. The value is an error code. For details, see section 4 "Error Codes."

[Requirement]

- Header files: mpi_hdmi.h, hi_comm_hdmi.h
- Library file: libhdmi.a

[Note]

- Before calling this API, ensure that the HDMI has been opened. Otherwise, HI_ERR_HDMI_DEV_NOT_OPEN is returned.
- The EDID is obtained from the sink after the HDMI cable is inserted. This MPI is used to forcibly obtain the EDID and is not used in general.

[Example]

See the example of HI_MPI_HDMI_GetSinkCapability.

[See Also]

HI_MPI_HDMI_GetSinkCapability

HI_MPI_HDMI_RegCallbackFunc

[Description]

Registers the callback function of an HDMI event.

[Syntax]

```
HI_S32 HI_MPI_HDMI_RegCallbackFunc(HI_HDMI_ID_E enHdmi,
HI_HDMI_CALLBACK_FUNC_S *pstCallbackFunc);
```

[Parameter]

Parameter	Description	Input/Output
enHdmi	HDMI ID The value is 0.	Input
pstCallbackFunc	Pointer to the HDMI callback function	Input

[Return Value]

Return Value	Description
0	Success
Other values	Failure. The value is an error code. For details, see section 4 "Error Codes."

[Requirement]

- Header files: mpi_hdmi.h, hi_comm_hdmi.h
- Library file: libhdmi.a

[Note]

- Before calling this API, ensure that the HDMI has been opened. Otherwise, HI_ERR_HDMI_DEV_NOT_OPEN is returned.
- You are advised to register the callback function of an HDMI event. When a hot plug event is generated, the registered callback function can be used to read the capability set information generated after the hot plug. The HDMI attributes can be modified based on the capability set information. Then you can restart the HDMI to enable the HDMI attributes to adapt to the peer-end monitor or television. If you do not register the callback function of an HDMI event, the event will be handled in the default way in the HDMI.
- If you have registered the callback function of the HDMI event, you need to deregister the callback function of the HDMI event before disabling the HDMI.
- If the same callback function or parameter is repeatedly registered, HI_ERR_HDMI_CALLBACK_ALREADY is returned.

[Example]

```
#define HDMI_PRINT printf("[HDMI]%s[%d]:\t",__func__,__LINE__);printf

#define HDMI_CHK_FAILURE_NORET(s32Ret) do{\
    if(s32Ret!=HI_SUCCESS)\
    {\
        HDMI_PRINT("s32Ret=%d is not expected HI_SUCCESS!\n",s32Ret);\
    }\
}while(0);

#define HDMI_CHK_FAILURE_RET(s32Ret)   do{\
    if(s32Ret!=HI_SUCCESS)\
    {\
        HDMI_PRINT("s32Ret=%d is not expected HI_SUCCESS!\n",s32Ret);\
        return HI_FAILURE;\
    }\
}while(0);

#define HDMI_CHK_FAILURE_GOTO(res,lable)   do{\
    if(HI_FAILURE==res)\
    {HDMI_PRINT("return failure!\n");goto lable;}\
}while(0);

typedef struct hiHDMI_ARGS_S
{
   HI_HDMI_ID_E    enHdmi;
    HI_HDMI_VIDEO_FMT_E eForceFmt;
}HDMI_ARGS_S;


static HI_S32 Hdmi_UnPlugProc(HI_VOID *pPrivateData)
{
    HI_S32         s32Ret = HI_SUCCESS;
   HDMI_ARGS_S     *pArgs  = (HDMI_ARGS_S*)pPrivateData;
   HI_HDMI_ID_E    hHdmi   =  pArgs->enHdmi;

   HDMI_PRINT("\n --- UnPlug event handling. --- \n");

   s32Ret = HI_MPI_HDMI_Stop(hHdmi);
    HDMI_CHK_FAILURE_RET(s32Ret);

   return s32Ret;
}

static HI_S32 Hdmi_HotPlugProc(HI_VOID *pPrivateData)
```

```
{
    HI_S32                  s32Ret = HI_SUCCESS;
    HDMI_ARGS_S             *pArgs  = (HDMI_ARGS_S*)pPrivateData;
    HI_HDMI_ID_E                hHdmi   =  pArgs->enHdmi;
    HI_HDMI_ATTR_S              stHdmiAttr;
    HI_HDMI_SINK_CAPABILITY_S   stSinkCap;

    HDMI_PRINT("\n --- hotplug event handling --- \n");

    s32Ret = HI_MPI_HDMI_GetAttr(hHdmi, &stHdmiAttr);
    HDMI_CHK_FAILURE_RET(s32Ret);

    s32Ret = HI_MPI_HDMI_GetSinkCapability(hHdmi, &stSinkCap);
    HDMI_CHK_FAILURE_RET(s32Ret);

    if (HI_FALSE == stSinkCap.bConnected )
    {
        HDMI_PRINT("stSinkCap.bConnected is HI_FALSE!\n");
        return HI_FAILURE;
    }

    if(HI_TRUE == stSinkCap.bSupportHdmi)
    {
        stHdmiAttr.bEnableHdmi = HI_TRUE;
        if(HI_TRUE != stSinkCap.bSupportYCbCr)
        {
            stHdmiAttr.enVidOutMode = HI_HDMI_VIDEO_MODE_RGB444;
        }
    }
    else
    {
        stHdmiAttr.enVidOutMode = HI_HDMI_VIDEO_MODE_RGB444;
        stHdmiAttr.bEnableHdmi = HI_FALSE;
    }

    if(HI_TRUE == stHdmiAttr.bEnableHdmi)
    {
        stHdmiAttr.bEnableAudio = HI_TRUE;
        stHdmiAttr.bEnableVideo = HI_TRUE;
        stHdmiAttr.bEnableAudInfoFrame = HI_TRUE;
        stHdmiAttr.bEnableAviInfoFrame = HI_TRUE;

        stHdmiAttr.enVidOutMode = HI_HDMI_VIDEO_MODE_YCBCR444;
        stHdmiAttr.enDeepColorMode = HI_HDMI_DEEP_COLOR_OFF;
```

```
        stHdmiAttr.bxvYCCMode = HI_FALSE;

        stHdmiAttr.bEnableAudio = HI_TRUE;
        stHdmiAttr.enSoundIntf = HI_HDMI_SND_INTERFACE_I2S;
        stHdmiAttr.bIsMultiChannel = HI_FALSE;

        stHdmiAttr.enBitDepth = HI_HDMI_BIT_DEPTH_16;

        stHdmiAttr.bDebugFlag = HI_FALSE;
        stHdmiAttr.bHDCPEnable = HI_FALSE;

        stHdmiAttr.b3DEnable = HI_FALSE;
        stHdmiAttr.enDefaultMode = HI_HDMI_FORCE_HDMI;
    }
    else
    {
        stHdmiAttr.bEnableAudio = HI_FALSE;
        stHdmiAttr.bEnableVideo = HI_TRUE;
        stHdmiAttr.bEnableAudInfoFrame = HI_FALSE;
        stHdmiAttr.bEnableAviInfoFrame = HI_FALSE;
        stHdmiAttr.bEnableAudio = HI_FALSE;
        stHdmiAttr.enVidOutMode =  HI_HDMI_VIDEO_MODE_RGB444;
        stHdmiAttr.enDefaultMode = HI_HDMI_FORCE_DVI;
    }


    if (    pArgs->eForceFmt >= HI_HDMI_VIDEO_FMT_1080P_60
        &&  pArgs->eForceFmt < HI_HDMI_VIDEO_FMT_BUTT
        &&  stSinkCap.bVideoFmtSupported[pArgs->eForceFmt] )
    {
        HDMI_PRINT("set force format=%d\n",pArgs->eForceFmt);
        stHdmiAttr.enVideoFmt = pArgs->eForceFmt;
    }
    else
    {
        HDMI_PRINT("not support expected format=%d, we set native
format=%d\n",pArgs->eForceFmt,stSinkCap.enNativeVideoFormat);
        stHdmiAttr.enVideoFmt = stSinkCap.enNativeVideoFormat;
    }

    s32Ret = HI_MPI_HDMI_SetAttr(hHdmi, &stHdmiAttr);
    HDMI_CHK_FAILURE_RET(s32Ret);

    /* HI_MPI_HDMI_SetAttr must before HI_MPI_HDMI_Start! */
```

```
    s32Ret = HI_MPI_HDMI_Start(hHdmi);
    HDMI_CHK_FAILURE_RET(s32Ret);
    return s32Ret;

}


HI_VOID HDMI_EventProc(HI_HDMI_EVENT_TYPE_E event, HI_VOID *pPrivateData)
{
    switch ( event )
    {
        case HI_HDMI_EVENT_HOTPLUG:
            Hdmi_HotPlugProc(pPrivateData);
            break;

        case HI_HDMI_EVENT_NO_PLUG:
            Hdmi_UnPlugProc(pPrivateData);
            break;

        case HI_HDMI_EVENT_EDID_FAIL:
            break;

        case HI_HDMI_EVENT_HDCP_FAIL:
            break;

        case HI_HDMI_EVENT_HDCP_SUCCESS:
            break;

        case HI_HDMI_EVENT_HDCP_USERSETTING:
            break;
        default:
             HDMI_PRINT("un-known event:%d\n",event);
            return;
    }

    return;
}

HI_HDMI_CALLBACK_FUNC_S g_stCallbackFunc;
HDMI_ARGS_S g_stHdmiArgs;
/*HDMI initialization process*/
HI_S32 HI_ADP_HDMIInit(HI_HDMI_ID_E enHDMIId,HI_HDMI_VIDEO_FMT_E eForceFmt)
{
    HI_S32 s32Ret = HI_FAILURE;
```

```
    s32Ret = HI_MPI_HDMI_Init();
    HDMI_CHK_FAILURE_RET(s32Ret);

    g_stHdmiArgs.enHdmi         = enHDMIId;
    g_stHdmiArgs.eForceFmt      = eForceFmt;

    g_stCallbackFunc.pfnHdmiEventCallback = HDMI_EventProc;
    g_stCallbackFunc.pPrivateData = &g_stHdmiArgs;

    s32Ret = HI_MPI_HDMI_Open(g_stHdmiArgs.enHdmi);
    HDMI_CHK_FAILURE_GOTO(s32Ret,ERROR2);

    s32Ret = HI_MPI_HDMI_RegCallbackFunc(g_stHdmiArgs.enHdmi,
&g_stCallbackFunc);
    HDMI_CHK_FAILURE_GOTO(s32Ret,ERROR1);

    return HI_SUCCESS;


ERROR1:
    s32Ret |= HI_MPI_HDMI_Close(g_stHdmiArgs.enHdmi);

ERROR2:
    s32Ret |= HI_MPI_HDMI_DeInit();

    return s32Ret;
}

/*Exit the HDMI initialization process.*/
HI_S32 HI_ADP_HDMIDeInit(HI_HDMI_ID_E enHDMIId)
{
    HI_S32 s32Ret = HI_FAILURE;

    s32Ret = HI_MPI_HDMI_Stop(enHDMIId);
    HDMI_CHK_FAILURE_NORET(s32Ret);

    g_stHdmiArgs.enHdmi = enHDMIId;
    g_stCallbackFunc.pfnHdmiEventCallback = HDMI_EventProc;
    g_stCallbackFunc.pPrivateData = &g_stHdmiArgs;
    s32Ret = HI_MPI_HDMI_UnRegCallbackFunc(enHDMIId, &g_stCallbackFunc);
    HDMI_CHK_FAILURE_NORET(s32Ret);

    s32Ret = HI_MPI_HDMI_Close(enHDMIId);
    HDMI_CHK_FAILURE_NORET(s32Ret);
```

```
    s32Ret = HI_MPI_HDMI_DeInit();
    HDMI_CHK_FAILURE_NORET(s32Ret);


    return s32Ret;
}
```

[See Also]

HI_MPI_HDMI_UnRegCallbackFunc

HI_MPI_HDMI_UnRegCallbackFunc

[Description]

Deregisters the callback function of an HDMI event.

[Syntax]

```
HI_S32 HI_MPI_HDMI_UnRegCallbackFunc(HI_HDMI_ID_E enHdmi,
HI_HDMI_CALLBACK_FUNC_S *pstCallbackFunc);
```

[Parameter]

Parameter	Description	Input/Output
enHdmi	HDMI ID The value is 0.	Input
pstCallbackFunc	Pointer to the HDMI callback function	Input

[Return Value]

Return Value	Description
0	Success
Other values	Failure. The value is an error code. For details, see section 4 "Error Codes."

[Requirement]

- Header files: mpi_hdmi.h, hi_comm_hdmi.h
- Library file: libhdmi.a

[Note]

- Before calling this API, ensure that the HDMI has been opened. Otherwise, HI_ERR_HDMI_DEV_NOT_OPEN is returned.
- If you have registered the callback function of the HDMI event, you need to deregister the callback function of the HDMI event before disabling the HDMI.

- If a callback function is repeatedly deregistered or an unregistered callback function is deregistered, HI_ERR_HDMI_CALLBACK_NOT_REGISTER is returned.

[Example]

See the example of HI_MPI_HDMI_RegCallbackFunc.

[See Also]

HI_MPI_HDMI_RegCallbackFunc

HI_MPI_HDMI_SetDeepColor

[Description]

Sets the deep color mode.

[Syntax]

```
HI_S32 HI_MPI_HDMI_SetDeepColor(HI_HDMI_ID_E enHdmi, HI_HDMI_DEEP_COLOR_E enDeepColor);
```

[Parameter]

Parameter	Description	Input/Output
enHdmi	HDMI ID The value is 0.	Input
enDeepColor	HDMI deep color mode	Input

[Return Value]

Return Value	Description
0	Success
Other values	Failure. The value is an error code. For details, see section 4 "Error Codes."

[Requirement]

- Header files: mpi_hdmi.h, hi_comm_hdmi.h
- Library file: libhdmi.a

[Note]

- HI_MPI_HDMI_SetDeepColor is an advanced interface. It is not called typically. If you need to set **Deep Color**, you are advised to use the **enDeepColorMode** parameter of the HI_MPI_HDMI_SetAttr interface.
- Before calling this API, ensure that the HDMI has been opened. Otherwise, HI_ERR_HDMI_DEV_NOT_OPEN is returned.

- Before invoking this interface to set the deep color mode, check whether the maximum sink capability is satisfied. If not, the sink in deep color mode is not supported. Therefore, do not use it.
- When the HI_MPI_HDMI_GetAttr interface is used to obtain **enVidOutMode** which is set to HI_HDMI_VIDEO_MODE_YCBCR422, only HI_HDMI_DEEP_COLOR_24BIT and HI_HDMI_DEEP_COLOR_OFF can be set for **enDeepColor.** When you set other parameters, HI_ERR_HDMI_INVALID_PARA is returned.

[Example]

```
/*Initialize the HDMI.*/
HI_MPI_HDMI_Init();

/*Enable the HDMI.*/
HI_MPI_HDMI_Open(HI_HDMI_ID_0);
/*Set the attributes.*/
…
/*Start the HDMI.*/
HI_MPI_HDMI_Start(HI_HDMI_ID_0);
…
HI_MPI_HDMI_SetDeepColor (HI_HDMI_ID_0, HI_HDMI_DEEP_COLOR_30BIT);
```

[See Also]

None

HI_MPI_HDMI_GetDeepColor

[Description]

Obtains the deep color mode.

[Syntax]

```
HI_S32 HI_MPI_HDMI_GetDeepColor(HI_HDMI_ID_E enHdmi, HI_HDMI_DEEP_COLOR_E
*penDeepColor);
```

[Parameter]

Parameter	Description	Input/Output
enHdmi	HDMI ID The value is 0.	Input
penDeepColor	Pointer to the deep color mode of the HDMI	Input

[Return Value]

Return Value	Description
0	Success

Return Value	Description
Other values	Failure. The value is an error code. For details, see section 4 "Error Codes."

[Requirement]

- Header files: mpi_hdmi.h, hi_comm_hdmi.h
- Library file: libhdmi.a

[Note]

- HI_MPI_HDMI_GetDeepColor is an advanced interface. It is not called typically.
- Before calling this API, ensure that the HDMI has been opened. Otherwise, HI_ERR_HDMI_DEV_NOT_OPEN is returned.

[Example]

None

[See Also]

HI_MPI_HDMI_SetDeepColor

HI_MPI_HDMI_SetInfoFrame

[Description]

Sets the HDMI information frame.

[Syntax]

```
HI_S32 HI_MPI_HDMI_SetInfoFrame(HI_HDMI_ID_E enHdmi, HI_HDMI_INFOFRAME_S
*pstInfoFrame);
```

[Parameter]

Parameter	Description	Input/Output
enHdmi	HDMI ID The value is 0.	Input
pstInfoFrame	Pointer to the HDMI information frame	Input

[Return Value]

Return Value	Description
0	Success
Other values	Failure. The value is an error code. For details, see section 4 "Error Codes."

[Requirement]

- Header files: mpi_hdmi.h, hi_comm_hdmi.h
- Library file: libhdmi.a

[Note]

- HI_MPI_HDMI_SetInfoFrame is an advanced interface. It is not called typically.
- Before calling this API, ensure that the HDMI has been opened. Otherwise, HI_ERR_HDMI_DEV_NOT_OPEN is returned.
- This MPI supports only the HI_INFOFRAME_TYPE_AVI and HI_INFOFRAME_TYPE_AUDIO information frames. For details of the return values, see HI_HDMI_INFOFRAME_TYPE_E.
- If this MPI is used, you need to configure the information frame based on the related audio/video attributes such as **enVideoFmt** and the standards including *High-Definition Multimedia Interface Specification Version 1.4b*, *High-Definition Multimedia Interface Specification Version 2.0*, *CEA-861-D*, and *CEA-861-F*. The behavior of transmitting the information frame that does not comply with the audio/video attributes and standards is undefined. Calling this API may cause display exceptions.

[Example]

```
HI_HDMI_INFOFRAME_S         stInfoFrame;

/*Initialize the HDMI.*/
HI_MPI_HDMI_Init();

/*Enable the HDMI.*/
HI_MPI_HDMI_Open(HI_HDMI_ID_0);
/*Set the attributes.*/
…
/*Start the HDMI.*/
HI_MPI_HDMI_Start(HI_HDMI_ID_0);
…
/*Set the valid aspect ratio of the auxiliary video information (AVI).*/
HI_MPI_HDMI_GetInfoFrame(HI_HDMI_ID_0,HI_INFOFRAME_TYPE_AVI,&stInfoFrame)
;
stInfoFrame.enInfoFrameType = HI_INFOFRAME_TYPE_AVI;
stInfoFrame.unInforUnit.stAVIInfoFrame.enActiveAspectRatio =
HI_HDMI_PIC_ASP_RATIO_16TO9;
HI_MPI_HDMI_SetInfoFrame(HI_HDMI_ID_0,&stInfoFrame);
```

[See Also]

HI_MPI_HDMI_GetInfoFrame

HI_MPI_HDMI_GetInfoFrame

[Description]

Obtains the HDMI information frame.

[Syntax]

```
HI_S32 HI_MPI_HDMI_GetInfoFrame(HI_HDMI_ID_E enHdmi,
HI_HDMI_INFOFRAME_TYPE_E enInfoFrameType, HI_HDMI_INFOFRAME_S
*pstInfoFrame);
```

[Parameter]

Parameter	Description	Input/Output
enHdmi	HDMI ID The value is 0.	Input
enInfoFrameType	Information frame type	Input
pstInfoFrame	Pointer to the HDMI information frame	Output

[Return Value]

Return Value	Description
0	Success
Other values	Failure. The value is an error code. For details, see section 4 "Error Codes."

[Requirement]

- Header files: mpi_hdmi.h, hi_comm_hdmi.h
- Library file: libhdmi.a

[Note]

- HI_MPI_HDMI_GetInfoFrame is an advanced interface. It is not called typically.
- Ensure that the HDMI is initialized before this MPI is called.
  - If the HDMI is not initialized, HI_ERR_HDMI_DEV_NOT_OPEN is returned.
  - If the HDMI is initialized and HI_MPI_HDMI_SetAttr has not been called, parameters obtained through this MPI may be invalid.
- This MPI supports only the HI_INFOFRAME_TYPE_AVI and HI_INFOFRAME_TYPE_AUDIO information frames. For details of the return values, see HI_HDMI_INFOFRAME_TYPE_E.

[Example]

See the example of HI_MPI_HDMI_SetInfoFrame.

[See Also]

HI_MPI_HDMI_SetInfoFrame

3 Data Structures

The data structures related to the HDMI are defined as follows:

- HI_HDMI_ID_E: Defines the HDMI ID.
- HI_HDMI_CallBack: Defines the pointer to the HDMI callback function.
- HI_HDMI_CALLBACK_FUNC_S: Defines the structure of the HDMI callback function.
- HI_HDMI_EVENT_TYPE_E: Defines the enumeration of the HDMI event notification.
- HI_HDMI_FORCE_ACTION_E: Defines the enumeration of the forcible output modes in the HDMI when the EDID fails to be read.
- HI_HDMI_ATTR_S: Defines the HDMI attribute structure.
- HI_HDMI_VIDEO_FMT_E: Defines the enumeration of HDMI video norms.
- HI_HDMI_VIDEO_MODE_E: Defines the enumeration of HDMI color space types.
- HI_HDMI_DEEP_COLOR_E: Defines the enumeration of HDMI deep color modes.
- HI_HDMI_SND_INTERFACE_E: Defines the enumeration of HDMI audio output (AO) interface types.
- HI_HDMI_SAMPLE_RATE_E: Defines the enumeration of HDMI AO sampling rates.
- HI_HDMI_BIT_DEPTH_E: Defines the enumeration of HDMI AO sampling bit widths.
- HI_HDMI_AUDIO_FORMAT_CODE_E: Defines the enumeration of HDMI audio formats.
- HI_HDMI_AUDIO_INFO_S: Defines the information about the HDMI audio capability set.
- HI_HDMI_SINK_CAPABILITY_S: Defines the structure of the HDMI sink capability.
- HI_HDMI_EDID_S: Defines the structure of the HDMI EDID information.
- HI_HDMI_INFOFRAME_TYPE_E: Defines the enumeration of HDMI information frame types.
- HI_HDMI_INFOFRAME_S: Defines the structure of the HDMI information frame.
- HI_HMDI_INFORFRAME_UNIT_U: Defines the structure of the HDMI information frame unit.
- HI_HDMI_AVI_INFOFRAME_VER2_S: Defines the structure of the HDMI AVI information frame (version 2) unit.
- HI_HDMI_AUD_INFOFRAME_VER1_S: Defines the structure of the HDMI audio information frame (version 1) unit.

- HI_HDMI_SPD_INFOFRAME_S: Defines the structure of the HDMI SPD information frame unit.
- HI_HDMI_MPEGSOURCE_INFOFRAME_S: Defines the structure of the HDMI MPEG information frame unit.
- HI_HDMI_VENDORSPEC_INFOFRAME_S: Defines the structure of the HDMI VS information frame unit.
- HI_HDMI_COLOR_SPACE_E: Defines the enumeration of the color space.
- HI_HDMI_BARINFO_E: Defines the enumeration of the bar information.
- HI_HDMI_SCANINFO_E: Defines the enumeration of the scan information.
- HI_HDMI_COLORIMETRY_E: Defines the enumeration of the colorimetry information.
- HI_HDMI_EXT_COLORIMETRY_E: Defines the enumeration of the extended colorimetry information.
- HI_HDMI_PIC_ASPECT_RATIO_E: Defines the enumeration of the picture aspect ratio.
- HI_HDMI_ACT_ASPECT_RATIO_E: Defines the enumeration of the actual picture aspect ratio.
- HI_HDMI_PICTURE_SCALING_E: Defines the enumeration of the picture scan information.
- HI_HDMI_RGB_QUAN_RAGE_E: Defines the enumeration of the RGB quantization range.
- HI_HDMI_PIXEL_REPETITION_E: Defines the enumeration of the pixel replication times.
- HI_HDMI_CONTENT_TYPE_E: Defines the enumeration of the content information.
- HI_HDMI_YCC_QUAN_RAGE_E: Defines the enumeration of the YCC quantization range.
- HI_HDMI_AUDIO_CHANEL_CNT_E: Defines the enumeration of the number of audio channels.
- HI_HDMI_CODING_TYPE_E: Define the enumeration of audio decoding types.
- HI_HDMI_AUDIO_SAMPLE_SIZE_E: Defines the enumeration of audio sampling sizes.
- HI_HDMI_AUDIO_SAMPLE_FREQ_E: Defines the enumeration of audio sampling frequencies.
- HI_HDMI_LEVEL_SHIFT_VALUE_E: Defines the enumeration of the audio shift information.
- HI_HDMI_LFE_PLAYBACK_LEVEL_E: Defines the enumeration of the audio playback information.
- HI_HDMI_QUANTIZATION_E: Defines the enumeration of the quantization range for the CSC output.
- HI_HDMI_AUDIO_SPEAKER_E: Defines the enumeration of the audio speaker capabilities.

Note: All important data structures are listed in this section. For details about other data structures, see **hi_comm_hdmi.h**

HI_HDMI_ID_E

[Description]

Defines the HDMI ID.

[Syntax]

```
typedef enum hiHDMI_ID_E
{
    HI_HDMI_ID_0        = 0,
    HI_HDMI_ID_BUTT
} HI_HDMI_ID_E;
```

[Member]

Member	Description
HI_HDMI_ID_0	HDMI 0

[Note]

- This MPI is reserved for extension of HDMI devices.
- HI_ERR_HDMI_INVALID_PARA is returned when this MPI is set to other values.

[See Also]

None

HI_HDMI_CallBack

[Description]

Defines the pointer to the HDMI callback function.

[Syntax]

```
typedef void (*HI_HDMI_CallBack)(HI_HDMI_EVENT_TYPE_E event, HI_VOID
*pPrivateData);
```

[Member]

Member	Description
HI_HDMI_EVENT_TYPE_E	Notification type of the HDMI event
pPrivateData	Private data of the event

[Note]

None

[See Also]

- HI_HDMI_CALLBACK_FUNC_S
- HI_MPI_HDMI_RegCallbackFunc
- HI_MPI_HDMI_UnRegCallbackFunc

HI_HDMI_CALLBACK_FUNC_S

[Description]

Defines the structure of the HDMI callback function.

[Syntax]

```
typedef struct hiHDMI_CALLBACK_FUNC_S
{
    HI_HDMI_CallBack    pfnHdmiEventCallback;
    HI_VOID             *pPrivateData;
}HI_HDMI_CALLBACK_FUNC_S;
```

[Member]

Member	Description
pfnHdmiEventCallback	Event handling callback function
pPrivateData	Private data of the callback function parameter

[Note]

- You are advised to register the callback function of the HDMI event. Otherwise, the default behavior is adopted in the HDMI. For details, see HI_MPI_HDMI_RegCallbackFunc.
- The embedded HDMI of the Hi35xx does not support the HDCP. The events related to the HDCP are invalid.

[See Also]

- HI_HDMI_CallBack
- HI_MPI_HDMI_RegCallbackFunc
- HI_MPI_HDMI_UnRegCallbackFunc

HI_HDMI_EVENT_TYPE_E

[Description]

Defines the enumeration of the HDMI event notification.

[Syntax]

```
typedef enum hiHDMI_EVENT_TYPE_E
{
    HI_HDMI_EVENT_HOTPLUG = 0x10,
    HI_HDMI_EVENT_NO_PLUG,
    HI_HDMI_EVENT_EDID_FAIL,
    HI_HDMI_EVENT_HDCP_FAIL,
    HI_HDMI_EVENT_HDCP_SUCCESS,
    HI_HDMI_EVENT_HDCP_USERSETTING = 0x17,
    HI_HDMI_EVENT_BUTT
```

```
}HI_HDMI_EVENT_TYPE_E;
```

[Member]

Member	Description
HI_HDMI_EVENT_HOTPLUG	HDMI cable insertion event
HI_HDMI_EVENT_NO_PLUG	HDMI cable removal event
HI_HDMI_EVENT_EDID_FAIL	HDMI EDID read failure event
HI_HDMI_EVENT_HDCP_FAIL	HDCP verification failure event
HI_HDMI_EVENT_HDCP_SUCCESS	HDCP verification success event
HI_HDMI_EVENT_HDCP_USERSETTING	HDMI reset event

[Note]

- The HDMI of the Hi35*xx* does not support the HDCP. The events related to the HDCP are invalid.
- The HDMI of the Hi35*xx* does not support the RSEN event. The events that are not listed in the preceding table are invalid.

[See Also]

- HI_MPI_HDMI_RegCallbackFunc
- HI_MPI_HDMI_UnRegCallbackFunc

HI_HDMI_FORCE_ACTION_E

[Description]

Defines the enumeration of the forcible output modes in the HDMI when the EDID fails to be read.

[Syntax]

```
typedef enum hiHDMI_FORCE_ACTION_E
{
    HI_HDMI_FORCE_NULL,
    HI_HDMI_FORCE_HDMI,
    HI_HDMI_FORCE_DVI,
    HI_HDMI_INIT_BOOT_CONFIG
}HI_HDMI_FORCE_ACTION_E;
```

[Member]

Member	Description
HI_HDMI_FORCE_NULL	Standard mode
HI_HDMI_FORCE_HDMI	Forcibly output data through the HDMI.

Member	Description
HI_HDMI_FORCE_DVI	Forcibly output data through the digital visual interface (DVI).
HI_HDMI_INIT_BOOT_CONFIG	Test only

[Note]

If you know that the sink does not support the HDMI, you can set **enForceMode** to **HI_HDMI_FORCE_DVI**. In this case, data is forcibly output through the DVI when the EDID fails to be read. Otherwise, you are advised to set **enForceMode** to **HI_HDMI_FORCE_HDMI**. In this case, data is forcibly output through the HDMI when the EDID fails to be read.

[See Also]

- HI_HDMI_ATTR_S
- HI_MPI_HDMI_SetAttr

HI_HDMI_ATTR_S

[Description]

Defines the HDMI attribute structure.

[Syntax]

```
typedef struct hiHDMI_ATTR_S
{
    HI_BOOL              bEnableHdmi;

    HI_BOOL              bEnableVideo;
    HI_HDMI_VIDEO_FMT_E  enVideoFmt;

    HI_HDMI_VIDEO_MODE_E  enVidOutMode;
    HI_HDMI_QUANTIZATION_E  eOutCscQuantization;
    HI_HDMI_DEEP_COLOR_E  enDeepColorMode;
    HI_BOOL              bxvYCCMode;

    HI_BOOL              bEnableAudio;
    HI_HDMI_SND_INTERFACE_E enSoundIntf;
    HI_BOOL              bIsMultiChannel;
    HI_HDMI_SAMPLE_RATE_E enSampleRate;
    HI_U8                u8DownSampleParm;

    HI_HDMI_BIT_DEPTH_E   enBitDepth;
    HI_U8                u8I2SCtlVbit;

    HI_BOOL              bEnableAviInfoFrame;
```

```
    HI_BOOL              bEnableAudInfoFrame;
    HI_BOOL              bEnableSpdInfoFrame;
    HI_BOOL              bEnableMpegInfoFrame;

    HI_BOOL               bDebugFlag;
    HI_BOOL              bHDCPEnable;

    HI_BOOL              b3DEnable;
    HI_U8               u83DParam;
    HI_HDMI_FORCE_ACTION_E enDefaultMode;
    HI_BOOL              bAuthMode;
    HI_BOOL              bEnableVidModeAdapt;
    HI_BOOL              bEnableDeepClrAdapt;
    HI_U32               u32PixClk;
} HI_HDMI_ATTR_S;
```

[Member]

Member	Description
bEnableHdmi	Whether to forcibly output the video through the HDMI HI_TRUE: Forcibly output the video through the HDMI. HI_FALSE: Output the video through the DVI.
bEnableVideo	Whether to output video HI_TRUE: normal picture output HI_FALSE: black screen output
enVideoFmt	Video norm. The value of **enVideoFmt** must be consistent with the norm of the video output. You are advised to set **enVideoFmt** to a video norm that is supported by the sink capability set.
enVidOutMode	HDMI video output mode HI_HDMI_VIDEO_MODE_RGB444, HI_HDMI_VIDEO_MODE_YCBCR422, HI_HDMI_VIDEO_MODE_YCBCR444, and HI_HDMI_VIDEO_MODE_YCBCR420 (supporting HDMI 2.0) are supported.
eOutCscQuantization	Quantification range for HDMI video output. The value can be **HDMI_QUANTIZATION_LIMITED_RANGE** or **HDMI_QUANTIZATION_FULL_RANGE**.

Member	Description
enDeepColorMode	DeepColor output mode • HI_HDMI_DEEP_COLOR_24BIT • HI_HDMI_DEEP_COLOR_30BIT • HI_HDMI_DEEP_COLOR_36BIT • HI_HDMI_DEEP_COLOR_OFF The default value is **HI_HDMI_DEEP_COLOR_24BIT**. Some sinks do not support the 30-bit and 36-bit HDMI deep color modes. If the deep color mode is set to **HI_HDMI_DEEP_COLOR_30BIT** or **HI_HDMI_DEEP_COLOR_36BIT**, exceptions may occur.
bxvYCCMode	Whether to enable the xvYCC output mode The default value is HI_FALSE, which is not supported by the Hi35xx currently.
bEnableAudio	Whether to enable the audio
enSoundIntf	HDMI audio source **enSoundIntf** must be set to **HI_HDMI_SND_INTERFACE_I2S** for the Hi35xx.
bIsMultiChannel	Multi-channel or stereo 0: stereo 1: eight-channel fixed for multi-channel
enSampleRate	Audio sampling rate. This parameter needs to be consistent with that of the AO. The Hi35xx does not support the sampling rate less than 32 kHz currently. You are advised to set **enSampleRate** to a sampling rate that is supported by the sink capability set.
u8DownSampleParm	Audio down sampling rate parameter The default value is 0. **u8DownSampleParm** must be set to **0** for the Hi35xx.
enBitDepth	Audio bit width. The default value is 16. This parameter needs to be consistent with that of the AO. You are advised to set **enBitDepth** to a bit width that is supported by the sink capability set.
u8I2SCtlVbit	Reserved. It is set to 0. **u8I2SCtlVbit** must be set to **0** for the Hi35xx.
bEnableAviInfoFrame	Whether to enable the AVI information frame You are advised to enable this function.
bEnableAudInfoFrame	Whether to enable audio information frame You are advised to enable this function. Audio InfoFrame cannot be enabled in DVI mode.

Member	Description
bEnableSpdInfoFrame	Whether to enable SPD information frame This function is disabled by default, and it cannot be enabled for the Hi35xx.
bEnableMpegInfoFrame	Whether to enable MPEG InfoFrame This function is disabled by default, and it cannot be enabled for the Hi35xx.
bDebugFlag	Whether to enable the debug information in the HDMI The default value is 0. This function cannot be enabled for the Hi35xx.
bHDCPEnable	Whether to enable the HDCP 0: disabled 1: enabled The default value is 0. This function cannot be enabled for the Hi35xx.
b3DEnable	Whether to enable 3D mode 0: disabled 1: enabled The default value is 0. This function cannot be enabled for the Hi35xx.
u83DParam	3D parameter The default value is 9. **u83DParam** must be set to **9** for the Hi35xx.
enDefaultMode	Forcible video output mode enumeration in the HDMI when the EDID fails to be read The default value is **HI_HDMI_FORCE_HDMI**.
bAuthMode	HDMI forcible output mode enable. If this mode is enabled, adaptive adjustment is not made based on the EDID information of the display or authentication device. This mode mainly applies to the authentication scenario. 0: disabled 1: enabled Note: It is used for debug and HDMI authentication test. The default value is **HI_FALSE**. You are advised to use the default value. For other precautions, see HI_MPI_HDMI_SetAttr.
bEnableVidModeAdapt	Whether to enable the drive color space adaptation policy You are advised to use the default value **HI_TRUE**.
bEnableDeepClrAdapt	Whether to enable the drive DeepColor adaptation policy You are advised to use the default value **HI_TRUE**.

Member	Description
u32PixClk	Pixel clock of the user-defined timing (unit: kHz) Note: This member must be set when you set the self-defined timing, and this member takes effect only when **enVideoFmt** is **HI_HDMI_VIDEO_FMT_VESA_CUSTOMER_DEFINE**.

[Note]

- You can set the HDMI attributes based on the recommended values.
- Set the attributes that are not supported currently to default values.
- The pixel clock (**u32PixClk**) must be set when you set the self-defined timing. (In other cases, the pixel clock does not need to be set, and will not take effect even if you have configured it.) If you do not set this member, the display device may work abnormally. You need to ensure the validity of this parameter because the driver cannot verify the validity of it. If the parameter is invalid, the display device may work abnormally.
- It is recommended that **enVidOutMode** be set to **YCBCR444**. In this way, the HDMI driver outputs data complying with the user configuration, and makes adjustments adaptively when the peer-end capability set does not meet the output requirements.

  For example, for the 4K@60 output configured by the user, if the maximum TMDS clock of the connected RX HDMI is 300 MHz, the HDMI driver adaptively adjusts the output to 4K@60 YCBCR420 based on the rules of giving priority to lightening up the peer-end device and improving user experience. Based on this, if the HDMI device with maximum 600 MHz TMDS clock is switched to again and the YCBCR444 output is required, users need to reconfigure the ATTR attribute.
- If **bEnableHdmi** is set to **HI_FALSE**, the HDMI driver works in DVI mode. In this mode, audio and all information frames (IFs) are not output. The **enVidOutMode**, **enDeepColorMode** and other parameters related to audio may not take effect after set.

[See Also]

- HI_MPI_HDMI_SetAttr
- HI_MPI_HDMI_GetAttr

HI_HDMI_VIDEO_FMT_E

[Description]

Defines the enumeration of HDMI video norms.

[Syntax]

```
typedef enum hiHDMI_VIDEO_FMT_E
{
    HI_HDMI_VIDEO_FMT_1080P_60 = 0,
    HI_HDMI_VIDEO_FMT_1080P_50,
    HI_HDMI_VIDEO_FMT_1080P_30,
    HI_HDMI_VIDEO_FMT_1080P_25,
    HI_HDMI_VIDEO_FMT_1080P_24,

    HI_HDMI_VIDEO_FMT_1080i_60,
```

```
    HI_HDMI_VIDEO_FMT_1080i_50,

    HI_HDMI_VIDEO_FMT_720P_60,
    HI_HDMI_VIDEO_FMT_720P_50,

    HI_HDMI_VIDEO_FMT_576P_50,
    HI_HDMI_VIDEO_FMT_480P_60,

    HI_HDMI_VIDEO_FMT_PAL,              /* B D G H I PAL */
    HI_HDMI_VIDEO_FMT_PAL_N,            /* (N)PAL       */
    HI_HDMI_VIDEO_FMT_PAL_Nc,           /* (Nc)PAL      */

    HI_HDMI_VIDEO_FMT_NTSC,             /* (M)NTSC      */
    HI_HDMI_VIDEO_FMT_NTSC_J,           /* NTSC-J       */
    HI_HDMI_VIDEO_FMT_NTSC_PAL_M,       /* (M)PAL       */

    HI_HDMI_VIDEO_FMT_SECAM_SIN,        /**< SECAM_SIN*/
    HI_HDMI_VIDEO_FMT_SECAM_COS,        /**< SECAM_COS*/

    HI_HDMI_VIDEO_FMT_861D_640X480_60,
    HI_HDMI_VIDEO_FMT_VESA_800X600_60,
    HI_HDMI_VIDEO_FMT_VESA_1024X768_60,
    HI_HDMI_VIDEO_FMT_VESA_1280X720_60,
    HI_HDMI_VIDEO_FMT_VESA_1280X800_60,
    HI_HDMI_VIDEO_FMT_VESA_1280X1024_60,
    HI_HDMI_VIDEO_FMT_VESA_1366X768_60,
    HI_HDMI_VIDEO_FMT_VESA_1440X900_60,
    HI_HDMI_VIDEO_FMT_VESA_1440X900_60_RB,
    HI_HDMI_VIDEO_FMT_VESA_1600X900_60_RB,
    HI_HDMI_VIDEO_FMT_VESA_1600X1200_60,
    HI_HDMI_VIDEO_FMT_VESA_1680X1050_60,
    HI_HDMI_VIDEO_FMT_VESA_1920X1080_60,
    HI_HDMI_VIDEO_FMT_VESA_1920X1200_60,
    HI_HDMI_VIDEO_FMT_VESA_2048X1152_60,

    HI_HDMI_VIDEO_FMT_2560x1440_30,
    HI_HDMI_VIDEO_FMT_2560x1440_60,
    HI_HDMI_VIDEO_FMT_2560x1600_60,
    HI_HDMI_VIDEO_FMT_1920x2160_30,

    HI_HDMI_VIDEO_FMT_3840X2160P_24,
    HI_HDMI_VIDEO_FMT_3840X2160P_25,
    HI_HDMI_VIDEO_FMT_3840X2160P_30,
    HI_HDMI_VIDEO_FMT_3840X2160P_50,
```

```
    HI_HDMI_VIDEO_FMT_3840X2160P_60,

    HI_HDMI_VIDEO_FMT_4096X2160P_24,
    HI_HDMI_VIDEO_FMT_4096X2160P_25,
    HI_HDMI_VIDEO_FMT_4096X2160P_30,
    HI_HDMI_VIDEO_FMT_4096X2160P_50,
    HI_HDMI_VIDEO_FMT_4096X2160P_60,

   HI_HDMI_VIDEO_FMT_VESA_CUSTOMER_DEFINE,

   HI_HDMI_VIDEO_FMT_BUTT
}HI_HDMI_VIDEO_FMT_E;
```

[Member]

None

[Note]

- The video output standards not supported are listed as follows:
  - HI_HDMI_VIDEO_FMT_PAL
  - HI_HDMI_VIDEO_FMT_PAL_N
  - HI_HDMI_VIDEO_FMT_PAL_Nc
  - HI_HDMI_VIDEO_FMT_NTSC
  - HI_HDMI_VIDEO_FMT_NTSC_J
  - HI_HDMI_VIDEO_FMT_NTSC_PAL_M
  - HI_HDMI_VIDEO_FMT_SECAM_SIN
  - HI_HDMI_VIDEO_FMT_SECAM_COS
  - HI_HDMI_VIDEO_FMT_VESA_1280X720_60
  - HI_HDMI_VIDEO_FMT_VESA_1440X900_60_RB
  - HI_HDMI_VIDEO_FMT_VESA_1600X900_60_RB
  - HI_HDMI_VIDEO_FMT_VESA_1920X1080_60
  - HI_HDMI_VIDEO_FMT_VESA_2048X1152_60
- You can set the HDMI norms based on the video output norms.
  - HI_HDMI_VIDEO_FMT_3840X2160P_50
  - HI_HDMI_VIDEO_FMT_3840X2160P_60
  - HI_HDMI_VIDEO_FMT_4096X2160P_50
  - HI_HDMI_VIDEO_FMT_4096X2160P_60

  The preceding four norms are the HDMI 2.0 specifications and products supporting only HDMI 1.4 do not support these norms.
- Hi3536D V100 supports HI_HDMI_VIDEO_FMT_2560x1440_30 at most. HI_HDMI_VIDEO_FMT_2560x1440_60, HI_HDMI_VIDEO_FMT_2560x1600_60, and HI_HDMI_VIDEO_FMT_3840X2160P_24 to HI_HDMI_VIDEO_FMT_4096X2160P_60 are not supported. If they are used, HI_ERR_HDMI_FEATURE_NO_SUPPORT is returned.

[See Also]

HI_MPI_HDMI_SetAttr

HI_HDMI_VIDEO_MODE_E

[Description]

Defines the enumeration of HDMI color space types.

[Syntax]

```
typedef enum hiHDMI_VIDEO_MODE
{
    HI_HDMI_VIDEO_MODE_RGB444,
    HI_HDMI_VIDEO_MODE_YCBCR422,
    HI_HDMI_VIDEO_MODE_YCBCR444,
    HI_HDMI_VIDEO_MODE_YCBCR420,
    HI_HDMI_VIDEO_MODE_BUTT
}HI_HDMI_VIDEO_MODE_E;
```

[Member]

Member	Description
HI_HDMI_VIDEO_MODE_RGB444	RGB444 output mode
HI_HDMI_VIDEO_MODE_YCBCR422	YCBCR422 output mode
HI_HDMI_VIDEO_MODE_YCBCR444	YCBCR444 output mode
HI_HDMI_VIDEO_MODE_YCBCR420	YCBCR420 output mode

[Note]

HI_HDMI_VIDEO_MODE_YCBCR420 is the HDMI 2.0 specifications and products supporting only HDMI 1.4 do not support this color space.

[See Also]

HI_MPI_HDMI_SetAttr

HI_HDMI_DEEP_COLOR_E

[Description]

Defines the enumeration of HDMI deep color modes.

[Syntax]

```
typedef enum hiHDMI_DEEP_COLOR_E
{
    HI_HDMI_DEEP_COLOR_24BIT = 0x00,
    HI_HDMI_DEEP_COLOR_30BIT,
    HI_HDMI_DEEP_COLOR_36BIT,
    HI_HDMI_DEEP_COLOR_OFF   = 0xff,
```

```
    HI_HDMI_DEEP_COLOR_BUTT
}HI_HDMI_DEEP_COLOR_E;
```

[Member]

Member	Description
HI_HDMI_DEEP_COLOR_24BIT	24-bit HDMI deep color mode
HI_HDMI_DEEP_COLOR_30BIT	30-bit HDMI deep color mode
HI_HDMI_DEEP_COLOR_36BIT	36-bit HDMI deep color mode
HI_HDMI_DEEP_COLOR_OFF	HDMI deep color disable mode (equivalent to the 24-bit HDMI deep color mode)

[Note]

- The Hi3536D V100 does not support the 36-bit HDMI deep color mode. HI_ERR_HDMI_FEATURE_NO_SUPPORT is returned when the HDMI deep color mode is set to HI_HDMI_DEEP_COLOR_36BIT.
- If the obtained peer end does not support 30-bit and 36-bit modes, it is recommended that you not set it. Otherwise, no information is likely to be displayed on the device.
- When **enVidOutMode** is HI_HDMI_VIDEO_MODE_YCBCR422, HI_HDMI_DEEP_COLOR_30BIT and HI_HDMI_DEEP_COLOR_36BIT cannot be set. Otherwise, HI_ERR_HDMI_INVALID_PARA is returned.
- HI_ERR_HDMI_INVALID_PARA is returned when HI_HDMI_DEEP_COLOR_BUTT or a value that is not enumerated is set.

[See Also]

HI_MPI_HDMI_SetAttr

HI_HDMI_SND_INTERFACE_E

[Description]

Defines the enumeration of HDMI AO interface types.

[Syntax]

```
typedef enum hiHDMI_SND_INTERFACE_E
{
    HI_HDMI_SND_INTERFACE_I2S,
    HI_HDMI_SND_INTERFACE_SPDIF,
    HI_HDMI_SND_INTERFACE_HBR,
    HI_HDMI_SND_INTERFACE_BUTT
}HI_HDMI_SND_INTERFACE_E;
```

[Member]

Member	Description
HI_HDMI_SND_INTERFACE_I2S	I^2S interface type
HI_HDMI_SND_INTERFACE_SPDIF	SPDIF interface type
HI_HDMI_SND_INTERFACE_HBR	HBR interface type

[Note]

- The HDMI audio output of the Hi35*xx* is I^2S interface type.
- When HI_HDMI_SND_INTERFACE_SPDIF and HI_HDMI_SND_INTERFACE_HBR are set, HI_ERR_HDMI_FEATURE_NO_SUPPORT is returned.
- When other parameters are set, HI_ERR_HDMI_INVALID_PARA is returned.

[See Also]

HI_MPI_HDMI_SetAttr

HI_HDMI_SAMPLE_RATE_E

[Description]

Defines the enumeration of HDMI AO sampling rates.

[Syntax]

```
typedef enum hiHDMI_SAMPLE_RATE_E
{
    HI_HDMI_SAMPLE_RATE_UNKNOWN=0,
    HI_HDMI_SAMPLE_RATE_8K    = 8000,
    HI_HDMI_SAMPLE_RATE_11K   = 11025,
    HI_HDMI_SAMPLE_RATE_12K   = 12000,
    HI_HDMI_SAMPLE_RATE_16K   = 16000,
    HI_HDMI_SAMPLE_RATE_22K   = 22050,
    HI_HDMI_SAMPLE_RATE_24K   = 24000,
    HI_HDMI_SAMPLE_RATE_32K   = 32000,
    HI_HDMI_SAMPLE_RATE_44K   = 44100,
    HI_HDMI_SAMPLE_RATE_48K   = 48000,
    HI_HDMI_SAMPLE_RATE_88K   = 88200,
    HI_HDMI_SAMPLE_RATE_96K   = 96000,
    HI_HDMI_SAMPLE_RATE_176K  = 176400,
    HI_HDMI_SAMPLE_RATE_192K  = 192000,
    HI_HDMI_SAMPLE_RATE_768K  = 768000,
    HI_HDMI_SAMPLE_RATE_BUTT
}HI_HDMI_SAMPLE_RATE_E;
```

[Member]

None

[Note]

- Only HI_HDMI_SAMPLE_RATE_32K, HI_HDMI_SAMPLE_RATE_44K and HI_HDMI_SAMPLE_RATE_48K are currently supported.
- HI_ERR_HDMI_INVALID_PARA is returned when HI_HDMI_SAMPLE_RATE_BUTT or a value that is not enumerated is set. HI_ERR_HDMI_FEATURE_NO_SUPPORT is set when other parameters are set.

[See Also]

HI_MPI_HDMI_SetAttr

HI_HDMI_BIT_DEPTH_E

[Description]

Defines the enumeration of HDMI AO sampling bit widths.

[Syntax]

```
typedef enum hiHDMI_BIT_DEPTH_E
{
    HI_HDMI_BIT_DEPTH_UNKNOWN =0,
    HI_HDMI_BIT_DEPTH_8  = 8,
    HI_HDMI_BIT_DEPTH_16 = 16,
    HI_HDMI_BIT_DEPTH_18 = 18,
    HI_HDMI_BIT_DEPTH_20 = 20,
    HI_HDMI_BIT_DEPTH_24 = 24,
    HI_HDMI_BIT_DEPTH_32 = 32,
    HI_HDMI_BIT_DEPTH_BUTT
}HI_HDMI_BIT_DEPTH_E;
```

[Member]

None

[Note]

HI_HDMI_BIT_DEPTH_UNKNOWN, HI_HDMI_BIT_DEPTH_8 and HI_HDMI_BIT_DEPTH_32 are currently not supported. When other parameters are set, HI_ERR_HDMI_FEATURE_NO_SUPPORT is returned. HI_ERR_HDMI_INVALID_PARA is returned when HI_HDMI_BIT_DEPTH_BUTT or a value that is not enumerated is set.

[See Also]

HI_MPI_HDMI_SetAttr

HI_HDMI_AUDIO_FORMAT_CODE_E

[Description]

Defines the enumeration of HDMI audio formats.

[Syntax]

```
typedef enum hiHDMI_AUDIO_FORMAT_CODE_E
```

```
{
    HI_HDMI_AUDIO_FORMAT_CODE_RESERVED  = 0x00,
    HI_HDMI_AUDIO_FORMAT_CODE_PCM,
    HI_HDMI_AUDIO_FORMAT_CODE_AC3,
    HI_HDMI_AUDIO_FORMAT_CODE_MPEG1,
    HI_HDMI_AUDIO_FORMAT_CODE_MP3,
    HI_HDMI_AUDIO_FORMAT_CODE_MPEG2,
    HI_HDMI_AUDIO_FORMAT_CODE_AAC,
    HI_HDMI_AUDIO_FORMAT_CODE_DTS,
    HI_HDMI_AUDIO_FORMAT_CODE_ATRAC,
    HI_HDMI_AUDIO_FORMAT_CODE_ONE_BIT,
    HI_HDMI_AUDIO_FORMAT_CODE_DDP,
    HI_HDMI_AUDIO_FORMAT_CODE_DTS_HD,
    HI_HDMI_AUDIO_FORMAT_CODE_MAT,
    HI_HDMI_AUDIO_FORMAT_CODE_DST,
    HI_HDMI_AUDIO_FORMAT_CODE_WMA_PRO,
    HI_HDMI_AUDIO_FORMAT_CODE_BUTT,
}HI_HDMI_AUDIO_FORMAT_CODE_E;
```

[Member]

None

[note]

None

[See Also]

HI_MPI_HDMI_SetAttr

HI_HDMI_AUDIO_INFO_S

[Description]

Defines the information about the HDMI audio capability set.

[Syntax]

```
typedef struct hiHDMI_AUDIO_INFO_S
{
    HI_HDMI_AUDIO_FORMAT_CODE_E     enAudFmtCode;
    HI_HDMI_SAMPLE_RATE_E
enSupportSampleRate[HI_HDMI_MAX_SAMPE_RATE_NUM];
    HI_U8                       u8AudChannel;
    HI_HDMI_BIT_DEPTH_E     bSupportBitDepth[HI_HDMI_MAX_BIT_DEPTH_NUM];
    HI_U32                      u32SupportBitDepthNum;
    HI_U32                      u32MaxBitRate;
}HI_HDMI_AUDIO_INFO_S;
```

[Member]

Member	Description
enAudFmtCode	Supported audio format
enSupportSampleRate	Supported audio sampling rate
u8AudChannel	Number of supported audio channels
bSupportBitDepth	Supported sampling size
u32SupportBitDepthNum	Number of supported sampling sizes
u32MaxBitRate	Maximum bit rate

[Note]

None

[See Also]

- HI_HDMI_SINK_CAPABILITY_S
- HI_MPI_HDMI_GetSinkCapability

HI_HDMI_SINK_CAPABILITY_S

[Description]

Defines the structure of the HDMI sink capability.

[Syntax]

```
typedef struct hiHDMI_SINK_CAPABILITY_S
{
    HI_BOOL             bConnected;
    HI_BOOL             bSupportHdmi;
    HI_BOOL             bIsSinkPowerOn;
    HI_BOOL             bIsRealEDID;

    HI_HDMI_VIDEO_FMT_E enNativeVideoFormat;
    HI_BOOL             bVideoFmtSupported[HI_HDMI_VIDEO_FMT_BUTT];
    HI_BOOL             bSupportYCbCr;

    HI_BOOL             bSupportxvYCC601;
    HI_BOOL             bSupportxvYCC709;
    HI_U8               u8MDBit;

    HI_U32                u32AudioInfoNum;

    HI_HDMI_AUDIO_INFO_S  stAudioInfo[HI_HDMI_MAX_AUDIO_CAP_COUNT];
    HI_BOOL             bSpeaker[HDMI_AUDIO_SPEAKER_BUTT];
```

```
    HI_U8             u8IDManufactureName[4];
    HI_U32            u32IDProductCode;
    HI_U32            u32IDSerialNumber;
    HI_U32            u32WeekOfManufacture;
    HI_U32            u32YearOfManufacture;
    HI_U8             u8Version;
    HI_U8             u8Revision;
    HI_U8             u8EDIDExternBlockNum;

    HI_BOOL           bIsPhyAddrValid;
    HI_U8             u8PhyAddr_A;
    HI_U8             u8PhyAddr_B;
    HI_U8             u8PhyAddr_C;
    HI_U8             u8PhyAddr_D;
    HI_BOOL           bSupportDVIDual;
    HI_BOOL           bSupportDeepColorYCBCR444
    HI_BOOL           bSupportDeepColor30Bit;
    HI_BOOL           bSupportDeepColor36Bit;
    HI_BOOL           bSupportDeepColor48Bit;
    HI_BOOL           bSupportAI;
    HI_U32            u32MaxTMDSClock;
    HI_BOOL           bI_Latency_Fields_Present
    HI_BOOL           bLatency_Fields_Present;
    HI_BOOL           bHDMI_Video_Present;
    HI_U8             u8Video_Latency;
    HI_U8             u8Audio_Latency;
    HI_U8             u8Interlaced_Video_Latency;
    HI_U8             u8Interlaced_Audio_Latency;
    HI_BOOL           bSupportY420DC30Bit;
    HI_BOOL           bSupportY420DC36Bit;
    HI_BOOL           bSupportY420DC48Bit;
    HI_BOOL           bSupportHdmi_2_0;
    HI_BOOL           bSupportY420Format[HI_HDMI_VIDEO_FMT_BUTT];
HI_BOOL           bOnlySupportY420Format[HI_HDMI_VIDEO_FMT_BUTT];
} HI_HDMI_SINK_CAPABILITY_S;
```

[Member]

Member	Description
bConnected	Whether the devices are connected
bSupportHdmi	Whether the HDMI (HDMI 1.4 by default) is supported by the device. If the HDMI is not supported by the device, the device is DVI.

Member	Description
bIsSinkPowerOn	Whether the sink device is powered on
bIsRealEDID	Whether the EDID obtains the flag from the sink device HI_TRUE: The EDID information is correctly read. HI_FASE: default settings
enNativeVideoFormat	Physical resolution of the display device
bVideoFmtSupported	Video capability set HI_TRUE: This display format is supported. HI_FALSE: This display format is not supported.
bSupportYCbCr	Whether the YCBCR display is supported HI_TRUE: The YCBCR display is supported. HI_FALSE: Only red-green-blue (RGB) is supported.
bSupportxvYCC601	Whether the xvYCC601 color format is supported
bSupportxvYCC709	Whether the xvYCC709 color format is supported
u8MDBit	Transfer profile supported by xvYCC601. 1: P0; 2: P1; 4: P2
u32AudioInfoNum	Number of pieces of supported audio information. The value range is [1, 16].
stAudioInfo	Supported audio information. A maximum of 16 groups of data is supported. Each group of information contains the audio encoding format, sampling rate, number of channels, sampling size, number of sampling sizes, and maximum bit rate. For details, see section 7.5.2 "Audio Data Block" in the *EIA-CEA-861-F*.
u8Speaker	Speaker position. For details, see the definition of SpeakerDATABlock in the *EIA-CEA-861-F*.
u8IDManufactureName	Device vendor flag
u32IDProductCode	Device ID
u32IDSerialNumber	Device sequence number
u32WeekOfManufacture	Device production data (week)
u32YearOfManufacture	Set the production data (year)
u8Version	Device version number
u8Revision	Device sub version number
u8EDIDExternBlockNum	EDID extended block number
bIsPhyAddrValid	Valid flag of the consumer electronics control (CEC) physical address
u8PhyAddr_A	CEC physical address A

Member	Description
u8PhyAddr_B	CEC physical address B
u8PhyAddr_C	CEC physical address C
u8PhyAddr_D	CEC physical address D
bSupportDVIDual	Whether the DVI dual-link operation is supported
bSupportDeepColorYCBCR444	Whether the YCBCR 4:4:4 deep-color mode is supported
bSupportDeepColor30Bit	Whether the deep-color 30-bit mode is supported
bSupportDeepColor36Bit	Whether the deep-color 36-bit mode is supported
bSupportDeepColor48Bit	Whether the deep-color 48-bit mode is supported
bSupportAI	Whether the Supports_AI mode
u32MaxTMDSClock	Maximum TMDS clock
bI_Latency_Fields_Present	Delay flag bit
bLatency_Fields_Present	Whether Video_Latency and Audio_Latency fields exist
bHDMI_Video_Present	Special video format
u8Video_Latency	Video delay
u8Audio_Latency	Video delay
u8Interlaced_Video_Latency	Video delay in interlaced video mode
u8Interlaced_Audio_Latency	Audio delay in interlaced video mode
bSupportY420DC30Bit	Whether the YCbCr420 deep-color 30-bit mode is supported
bSupportY420DC36Bit	Whether the YCbCr420 deep-color 36-bit mode is supported
bSupportY420DC48Bit	Whether the YCbCr420 deep-color 48-bit mode is supported
bSupportHdmi_2_0	Whether HDMI 2.0 is supported
bSupportY420Format	The YCbCr420 video format is supported.
bOnlySupportY420Format	Only the YCbCr420 video format is supported.

[Note]

HDMI 1.4 and HDMI 2.0 are the same in capacity reporting.

[See Also]

HI_MPI_HDMI_GetSinkCapability

HI_HDMI_EDID_S

[Description]

Defines the structure of the HDMI EDID information.

[Syntax]

```
typedef struct hiHI_HDMI_EDID_S
{
    HI_BOOL              bEdidValid;
    HI_U32               u32Edidlength;
    HI_U8                u8Edid[512];
}HI_HDMI_EDID_S;
```

[Member]

Member	Description
bEdidValid	EDID information validity
u32Edidlength	EDID information length
u8Edid	EDID information

[Note]

None

[See Also]

HI_MPI_HDMI_Force_GetEDID

HI_HDMI_INFOFRAME_TYPE_E

[Description]

Defines the enumeration of HDMI information frame types.

[Syntax]

```
typedef enum tagHI_HDMI_INFOFRAME_TYPE_E
{
    HI_INFOFRAME_TYPE_AVI,
    HI_INFOFRAME_TYPE_SPD,
    HI_INFOFRAME_TYPE_AUDIO,
    HI_INFOFRAME_TYPE_MPEG,
    HI_INFOFRAME_TYPE_VENDORSPEC,
    HI_INFOFRAME_TYPE_BUTT
}HI_HDMI_INFOFRAME_TYPE_E;
```

[Member]

None

[Note]

Only HI_INFOFRAME_TYPE_AVI and HI_INFOFRAME_TYPE_AUDIO are supported. HI_ERR_HDMI_FEATURE_NO_SUPPORT is returned when HI_INFOFRAME_TYPE_SPD, HI_INFOFRAME_TYPE_MPEG, or HI_INFOFRAME_TYPE_VENDORSPEC is set.HI_ERR_HDMI_INVALID_PARA is returned when other parameters are set.

[See Also]

- HI_MPI_HDMI_SetInfoFrame
- HI_MPI_HDMI_GetInfoFrame

HI_HDMI_INFOFRAME_S

[Description]

Defines the structure of the HDMI information frame.

[Syntax]

```
typedef struct hiUNF_HDMI_INFOFRAME_S
{
    HI_HDMI_INFOFRAME_TYPE_E    enInfoFrameType;
    HI_HMDI_INFORFRAME_UNIT_U   unInforUnit;
}HI_HDMI_INFOFRAME_S;
```

[Member]

Member	Description
enInfoFrameType	Information frame type
unInforUnit	Information frame unit (content)

[Note]

None

[See Also]

- HI_MPI_HDMI_SetInfoFrame
- HI_MPI_HDMI_GetInfoFrame

HI_HMDI_INFORFRAME_UNIT_U

[Description]

Defines the structure of the HDMI information frame unit.

[Syntax]

```
typedef union hiHDMI_INFOFRAME_UNIT_U
{
    HI_HDMI_AVI_INFOFRAME_VER2_S   stAVIInfoFrame;
```

```
    HI_HDMI_AUD_INFOFRAME_VER1_S   stAUDInfoFrame;
    HI_HDMI_SPD_INFOFRAME_S        stSPDInfoFrame;
    HI_HDMI_MPEGSOURCE_INFOFRAME_S stMPEGSourceInfoFrame;
    HI_HDMI_VENDORSPEC_INFOFRAME_S stVendorSpecInfoFrame;
}HI_HMDI_INFORFRAME_UNIT_U;
```

[Member]

Member	Description
stAVIInfoFrame	AVI information frame unit
stAUDInfoFrame	Audio information frame unit
stSPDInfoFrame	SPD information frame unit
stMPEGSourceInfoFrame	MPEG information frame unit
stVendorSpecInfoFrame	Vendor-specific (VS) information frame unit

[Note]

None

[See Also]

- HI_MPI_HDMI_SetInfoFrame
- HI_MPI_HDMI_GetInfoFrame

HI_HDMI_AVI_INFOFRAME_VER2_S

[Description]

Defines the structure of the HDMI AVI information frame (version 2) unit.

[Syntax]

```
typedef struct hi_HDMI_AVI_INFOFRAME_VER2_S
{
    HI_HDMI_VIDEO_FMT_E       enTimingMode;
    HI_HDMI_COLOR_SPACE_E     enColorSpace;
    HI_BOOL                 bActive_Infor_Present;
    HI_HDMI_BARINFO_E         enBarInfo;
    HI_HDMI_SCANINFO_E        enScanInfo;

    HI_HDMI_COLORIMETRY_E     enColorimetry;
    HI_HDMI_EXT_COLORIMETRY_E  enExtColorimetry;
    HI_HDMI_PIC_ASPECT_RATIO_E enAspectRatio;
    HI_HDMI_ACT_ASPECT_RATIO_E enActiveAspectRatio;
    HI_HDMI_PICTURE_SCALING_E  enPictureScaling;

    HI_HDMI_RGB_QUAN_RAGE_E    enRGBQuantization;
    HI_BOOL                 bIsITContent;
```

```
    HI_HDMI_PIXEL_REPETITION_E enPixelRepetition;
    HI_HDMI_CONTENT_TYPE_E     enContentType;
    HI_HDMI_YCC_QUAN_RAGE_E    enYCCQuantization;

    HI_U16                  u16LineNEndofTopBar;
    HI_U16                  u16LineNStartofBotBar;
    HI_U16                  u16PixelNEndofLeftBar;
    HI_U16                  u16PixelNStartofRightBar;
}HI_HDMI_AVI_INFOFRAME_VER2_S;
```

[Member]

Member	Description
enTimingMode	Video timing
enColorSpace	Color space
bActive_Infor_Present	Whether the information is valid
enBarInfo	Bar information
enScanInfo	Scan information
enColorimetry	Color gamut
enExtColorimetry	Extended color gamut
enAspectRatio	Picture aspect ratio
enActiveAspectRatio	Valid aspect ratio
enPictureScaling	Picture equalization
enRGBQuantization	RGB quantization
bIsITContent	Whether the IT content is valid
enPixelRepetition	Pixel doubling
enContentType	IT content type
enYCCQuantization	YCC quantization
u16LineNEndofTopBar	Number of end lines for the top bar
u16LineNStartofBotBar	Number of start lines for the bottom bar
u16PixelNEndofLeftBar	Number of end pixels for the left bar
u16PixelNStartofRightBar	Number of start pixels for the right bar

[Note]

For details, see *High-Definition Multimedia Interface Specification Version 1.4b*, *High-Definition Multimedia Interface Specification Version 2.0*, *CEA-861-D*, and *CEA-861-F*.

[See Also]

- HI_MPI_HDMI_SetInfoFrame
- HI_MPI_HDMI_GetInfoFrame

HI_HDMI_AUD_INFOFRAME_VER1_S

[Description]

Defines the structure of the HDMI audio information frame (version 1) unit.

[Syntax]

```
typedef struct hiHDMI_AUD_INFOFRAME_VER1_S
{
    HI_HDMI_AUDIO_CHANEL_CNT_E        enChannelCount;
    HI_HDMI_CODING_TYPE_E              enCodingType;
    HI_HDMI_AUDIO_SAMPLE_SIZE_E       enSampleSize;
    HI_HDMI_AUDIO_SAMPLE_FREQ_E       enSamplingFrequency;
    HI_U8                             u8ChannelAlloc;
    HI_HDMI_LEVEL_SHIFT_VALUE_E       enLevelShift;
    HI_HDMI_LFE_PLAYBACK_LEVEL_E      enLfePlaybackLevel;
    HI_BOOL                           bDownmixInhibit;
}HI_HDMI_AUD_INFOFRAME_VER1_S;
```

[Member]

Member	Description
enChannelCount	Number of audio channels
enCodingType	Audio format
enSampleSize	Audio sampling depth (bit width)
enSamplingFrequency	Audio sampling rate
u8ChannelAlloc	Channel/Speaker allocation
enLevelShift	Left level shift value
enLfePlaybackLevel	LFE playback level information
bDownmixInhibit	Down mixing inhibit flag

[Note]

For details, see *High-Definition Multimedia Interface Specification Version 1.4b*, *High-Definition Multimedia Interface Specification Version 2.0*, *CEA-861-D*, and *CEA-861-F*.

[See Also]

- HI_MPI_HDMI_SetInfoFrame
- HI_MPI_HDMI_GetInfoFrame

HI_HDMI_SPD_INFOFRAME_S

[Description]

Defines the structure of the HDMI SPD information frame unit.

[Syntax]

```
typedef struct hiHDMI_SPD_INFOFRAME_S
{
    HI_U8                   u8VendorName[8];
    HI_U8                   u8ProductDescription[16];
}HI_HDMI_SPD_INFOFRAME_S;
```

[Member]

Member	Description
u8VendorName	Name of the source end vendor
u8ProductDescription	Description of the source end product

[Note]

For details, see *High-Definition Multimedia Interface Specification Version 1.4b*, *High-Definition Multimedia Interface Specification Version 2.0*, *CEA-861-D*, and *CEA-861-F*.

[See Also]

- HI_MPI_HDMI_SetInfoFrame
- HI_MPI_HDMI_GetInfoFrame

HI_HDMI_MPEGSOURCE_INFOFRAME_S

[Description]

Defines the structure of the HDMI MPEG information frame unit.

[Syntax]

```
typedef struct hiHDMI_MPEGSOURCE_INFOFRAME_S
{
    HI_U32                  u32MPEGBitRate;
    HI_BOOL                 bIsFieldRepeated;
}HI_HDMI_MPEGSOURCE_INFOFRAME_S;
```

[Member]

Member	Description
u32MPEGBitRate	MPEG bit rate
bIsFieldRepeated	Whether the current frame is a repeated frame

[Note]

For details, see *High-Definition Multimedia Interface Specification Version 1.4b*, *High-Definition Multimedia Interface Specification Version 2.0*, *CEA-861-D*, and *CEA-861-F*.

[See Also]

- HI_MPI_HDMI_SetInfoFrame
- HI_MPI_HDMI_GetInfoFrame

HI_HDMI_VENDORSPEC_INFOFRAME_S

[Description]

Defines the structure of the HDMI VS information frame unit.

[Syntax]

typedef struct hiHDMI_VENDORSPEC_INFOFRAME_S

{

HI_U32 u32RegistrationId;

}HI_HDMI_VENDORSPEC_INFOFRAME_S;

[Member]

Member	Description
u32RegistrationId	IEEE registration code

[Note]

For details, see *High-Definition Multimedia Interface Specification Version 1.4b*, *High-Definition Multimedia Interface Specification Version 2.0*, *CEA-861-D*, and *CEA-861-F*.

[See Also]

- HI_MPI_HDMI_SetInfoFrame
- HI_MPI_HDMI_GetInfoFrame

HI_HDMI_COLOR_SPACE_E

[Description]

Defines the enumeration of the color space.

[Syntax]

```
typedef enum hiHDMI_COLOR_SPACE_E
{
    HI_HDMI_COLOR_SPACE_RGB444,
    HI_HDMI_COLOR_SPACE_YCBCR422,
    HI_HDMI_COLOR_SPACE_YCBCR444,
```

```
    HI_HDMI_COLOR_SPACE_YCBCR420,
}HI_HDMI_COLOR_SPACE_E;
```

[Member]

None

[Note]

HI_HDMI_COLOR_SPACE_YCBCR420 is the HDMI 2.0 specifications and products supporting only HDMI 1.4 do not support this color space.

[See Also]

- HI_MPI_HDMI_SetInfoFrame
- HI_MPI_HDMI_GetInfoFrame

HI_HDMI_BARINFO_E

[Description]

Defines the enumeration of the bar information.

[Syntax]

```
typedef enum hiHDMI_BARINFO_E
{
   HDMI_BAR_INFO_NOT_VALID,
   HDMI_BAR_INFO_V,
   HDMI_BAR_INFO_H,
   HDMI_BAR_INFO_VH
}HI_HDMI_BARINFO_E;
```

[Member]

None

[Note]

None

[See Also]

- HI_MPI_HDMI_SetInfoFrame
- HI_MPI_HDMI_GetInfoFrame

HI_HDMI_SCANINFO_E

[Description]

Defines the enumeration of the scan information.

[Syntax]

```
typedef enum hiHDMI_SCANINFO_E
{
    HDMI_SCAN_INFO_NO_DATA      = 0,
```

```
    HDMI_SCAN_INFO_OVERSCANNED  = 1,
    HDMI_SCAN_INFO_UNDERSCANNED = 2,
    HDMI_SCAN_INFO_FUTURE
}HI_HDMI_SCANINFO_E;
```

[Member]

None

[Note]

None

[See Also]

- HI_MPI_HDMI_SetInfoFrame
- HI_MPI_HDMI_GetInfoFrame

HI_HDMI_COLORIMETRY_E

[Description]

Defines the enumeration of the colorimetry information.

[Syntax]

```
typedef enum hiHDMI_COLORIMETRY_E
{
    HDMI_COLORIMETRY_NO_DATA,
    HDMI_COLORIMETRY_ITU601,
    HDMI_COLORIMETRY_ITU709,
    HDMI_COLORIMETRY_EXTENDED,
} HI_HDMI_COLORIMETRY_E;
```

[Member]

None

[Note]

None

[See Also]

- HI_MPI_HDMI_SetInfoFrame
- HI_MPI_HDMI_GetInfoFrame

HI_HDMI_EXT_COLORIMETRY_E

[Description]

Defines the enumeration of the extended colorimetry information.

[Syntax]

```
typedef enum hiHDMI_EXT_COLORIMETRY_E
{
```

```
    HDMI_COLORIMETRY_XVYCC_601,
    HDMI_COLORIMETRY_XVYCC_709,
    HDMI_COLORIMETRY_S_YCC_601,
    HDMI_COLORIMETRY_ADOBE_YCC_601,
    HDMI_COLORIMETRY_ADOBE_RGB,
    HDMI_COLORIMETRY_2020_CONST_LUMINOUS,
    HDMI_COLORIMETRY_2020_NON_CONST_LUMINOUS,
    HDMI_COLORIMETRY_RESERVED
} HI_HDMI_EXT_COLORIMETRY_E;
```

[Member]

None

[Note]

HDMI_COLORIMETRY_RESERVED is currently not supported. When this parameter is set, HI_ERR_HDMI_INVALID_PARA is returned.

[See Also]

- HI_MPI_HDMI_SetInfoFrame
- HI_MPI_HDMI_GetInfoFrame

HI_HDMI_PIC_ASPECT_RATIO_E

[Description]

Defines the enumeration of the picture aspect ratio.

[Syntax]

```
typedef enum hiHDMI_PIC_ASPECT_RATIO_E
{
    HI_HDMI_PIC_ASP_RATIO_NO_DATA,
    HI_HDMI_PIC_ASP_RATIO_4TO3,
    HI_HDMI_PIC_ASP_RATIO_16TO9,
    HI_HDMI_PIC_ASP_RATIO_64TO27,
    HI_HDMI_PIC_ASP_RATIO_256TO135,
    HI_HDMI_PIC_ASP_RATIO_RESERVED,
}HI_HDMI_PIC_ASPECT_RATIO_E;
```

[Member]

None

[Note]

- HI_HDMI_PIC_ASP_RATIO_RESERVED is currently not supported. When this parameter is set, HI_ERR_HDMI_INVALID_PARA is returned.
- HI_ERR_HDMI_INVALID_PARA is returned when a value that is not enumerated is set.

[See Also]

- HI_MPI_HDMI_SetInfoFrame
- HI_MPI_HDMI_GetInfoFrame

HI_HDMI_ACT_ASPECT_RATIO_E

[Description]

Defines the enumeration of the actual picture aspect ratio.

[Syntax]

```
typedef enum hiHDMI_ACT_ASPECT_RATIO_E
{
    HI_HDMI_ACT_ASP_RATIO_RESERVED_0,
    HI_HDMI_ACT_ASP_RATIO_RESERVED_1,
    HI_HDMI_ACT_ASP_RATIO_16TO9_TOP,
    HI_HDMI_ACT_ASP_RATIO_14TO9_TOP,
    HI_HDMI_ACT_ASP_RATIO_16TO9_BOX_CENTER,
    HI_HDMI_ACT_ASP_RATIO_RESERVED_5,
    HI_HDMI_ACT_ASP_RATIO_RESERVED_6,
    HI_HDMI_ACT_ASP_RATIO_RESERVED_7,
   HI_HDMI_ACT_ASP_RATIO_SAME_PIC,
   HI_HDMI_ACT_ASP_RATIO_4TO3_CENTER,
   HI_HDMI_ACT_ASP_RATIO_16TO9_CENTER,
   HI_HDMI_ACT_ASP_RATIO_14TO9_CENTER,
   HI_HDMI_ACT_ASP_RATIO_RESERVED_12,
   HI_HDMI_ACT_ASP_RATIO_4TO3_14_9,
   HI_HDMI_ACT_ASP_RATIO_16TO9_14_9,
   HI_HDMI_ACT_ASP_RATIO_16TO9_4_3,
}HI_HDMI_ACT_ASPECT_RATIO_E;
```

[Member]

None

[Note]

The following parameters are currently not supported:

HI_HDMI_ACT_ASP_RATIO_RESERVED_0,
HI_HDMI_ACT_ASP_RATIO_RESERVED_1,
HI_HDMI_ACT_ASP_RATIO_RESERVED_5,
HI_HDMI_ACT_ASP_RATIO_RESERVED_6,
HI_HDMI_ACT_ASP_RATIO_RESERVED_7,
HI_HDMI_ACT_ASP_RATIO_RESERVED_12

When these parameters are set, HI_ERR_HDMI_INVALID_PARA is returned.

[See Also]

- HI_MPI_HDMI_SetInfoFrame
- HI_MPI_HDMI_GetInfoFrame

HI_HDMI_PICTURE_SCALING_E

[Description]

Defines the enumeration of the picture scan information.

[Syntax]

```
typedef enum hiHDMI_PICTURE_SCALING_E
{
    HDMI_PICTURE_NON_UNIFORM_SCALING,
    HDMI_PICTURE_SCALING_H,
    HDMI_PICTURE_SCALING_V,
    HDMI_PICTURE_SCALING_HV
 }HI_HDMI_PICTURE_SCALING_E;
```

[Member]

None

[Note]

None

[See Also]

- HI_MPI_HDMI_SetInfoFrame
- HI_MPI_HDMI_GetInfoFrame

HI_HDMI_RGB_QUAN_RAGE_E

[Description]

Defines the enumeration of the RGB quantization range.

[Syntax]

```
typedef enum hiHDMI_RGB_QUAN_RAGE_E
{
    HDMI_RGB_QUANTIZATION_DEFAULT_RANGE,
    HDMI_RGB_QUANTIZATION_LIMITED_RANGE,
    HDMI_RGB_QUANTIZATION_FULL_RANGE,
     HDMI_RGB_QUANTIZATION_FULL_RESERVED
}HI_HDMI_RGB_QUAN_RAGE_E;
```

[Member]

None

[Note]

HDMI_RGB_QUANTIZATION_FULL_RESERVED is currently not supported. When this parameter is set, HI_ERR_HDMI_INVALID_PARA is returned.

[See Also]

- HI_MPI_HDMI_SetInfoFrame

- HI_MPI_HDMI_GetInfoFrame

HI_HDMI_PIXEL_REPETITION_E

[Description]

Defines the enumeration of the pixel replication times.

[Syntax]

```
typedef enum hiHDMI_PIXEL_REPETITION_E
{
    HDMI_PIXEL_REPET_NO,
    HDMI_PIXEL_REPET_2_TIMES,
    HDMI_PIXEL_REPET_3_TIMES,
    HDMI_PIXEL_REPET_4_TIMES,
    HDMI_PIXEL_REPET_5_TIMES,
    HDMI_PIXEL_REPET_6_TIMES,
    HDMI_PIXEL_REPET_7_TIMES,
    HDMI_PIXEL_REPET_8_TIMES,
    HDMI_PIXEL_REPET_9_TIMES,
    HDMI_PIXEL_REPET_10_TIMES,
    HDMI_PIXEL_REPET_RESERVED_A,
    HDMI_PIXEL_REPET_RESERVED_B,
    HDMI_PIXEL_REPET_RESERVED_C,
    HDMI_PIXEL_REPET_RESERVED_D,
    HDMI_PIXEL_REPET_RESERVED_E,
    HDMI_PIXEL_REPET_RESERVED_F,
}HI_HDMI_PIXEL_REPETITION_E;
```

[Member]

None

[Note]

- HDMI_PIXEL_REPET_RESERVED_A to HDMI_PIXEL_REPET_RESERVED_F are currently not supported. When these parameters are set, HI_ERR_HDMI_INVALID_PARA is returned.
- HI_ERR_HDMI_INVALID_PARA is returned when a value that is not enumerated is set.

[See Also]

- HI_MPI_HDMI_SetInfoFrame
- HI_MPI_HDMI_GetInfoFrame

HI_HDMI_CONTENT_TYPE_E

[Description]

Defines the enumeration of the content information.

[Syntax]

```
typedef enum hiHDMI_CONTENT_TYPE_E
{
    HDMI_CONTNET_GRAPHIC,
    HDMI_CONTNET_PHOTO,
    HDMI_CONTNET_CINEMA,
    HDMI_CONTNET_GAME
}HI_HDMI_CONTENT_TYPE_E;
```

[Member]

None

[Note]

None

[See Also]

- HI_MPI_HDMI_SetInfoFrame
- HI_MPI_HDMI_GetInfoFrame

HI_HDMI_YCC_QUAN_RAGE_E

[Description]

Defines the enumeration of the YCC quantization range.

[Syntax]

```
typedef enum hiHDMI_YCC_QUAN_RAGE_E
{
    HDMI_YCC_QUANTIZATION_LIMITED_RANGE,
    HDMI_YCC_QUANTIZATION_FULL_RANGE,
    HDMI_YCC_QUANTIZATION_RESERVED_2,
    HDMI_YCC_QUANTIZATION_RESERVED_3
}HI_HDMI_YCC_QUAN_RAGE_E;
```

[Member]

None

[Note]

- HDMI_YCC_QUANTIZATION_RESERVED_2 and HDMI_YCC_QUANTIZATION_RESERVED_3 are currently not supported. When these parameters are set, HI_ERR_HDMI_INVALID_PARA is returned.
- HI_ERR_HDMI_INVALID_PARA is returned when a value that is not enumerated is set.

[See Also]

- HI_MPI_HDMI_SetInfoFrame
- HI_MPI_HDMI_GetInfoFrame

HI_HDMI_AUDIO_CHANEL_CNT_E

[Description]

Defines the enumeration of the number of audio channels.

[Syntax]

```
typedef enum hiHDMI_AUDIO_CHANEL_CNT_E
{
    HI_HDMI_AUDIO_CHANEL_CNT_STREAM,
    HI_HDMI_AUDIO_CHANEL_CNT_2,
    HI_HDMI_AUDIO_CHANEL_CNT_3,
    HI_HDMI_AUDIO_CHANEL_CNT_4,
    HI_HDMI_AUDIO_CHANEL_CNT_5,
    HI_HDMI_AUDIO_CHANEL_CNT_6,
    HI_HDMI_AUDIO_CHANEL_CNT_7,
    HI_HDMI_AUDIO_CHANEL_CNT_8,
}HI_HDMI_AUDIO_CHANEL_CNT_E;
```

[Member]

None

[Note]

None

[See Also]

- HI_MPI_HDMI_SetInfoFrame
- HI_MPI_HDMI_GetInfoFrame

HI_HDMI_CODING_TYPE_E

[Description]

Define the enumeration of audio decoding types.

[Syntax]

```
typedef enum hiHDMI_CODING_TYPE_E
{
    HDMI_AUDIO_CODING_REFER_STREAM_HEAD,
    HDMI_AUDIO_CODING_PCM,
    HDMI_AUDIO_CODING_AC3,
    HDMI_AUDIO_CODING_MPEG1,
    HDMI_AUDIO_CODING_MP3,
    HDMI_AUDIO_CODING_MPEG2,
    HDMI_AUDIO_CODING_AACLC,
    HDMI_AUDIO_CODING_DTS,
    HDMI_AUDIO_CODING_ATRAC,
    HDMI_AUDIO_CODIND_ONE_BIT_AUDIO,
```

```
    HDMI_AUDIO_CODING_ENAHNCED_AC3,
    HDMI_AUDIO_CODING_DTS_HD,
    HDMI_AUDIO_CODING_MAT,
    HDMI_AUDIO_CODING_DST,
    HDMI_AUDIO_CODING_WMA_PRO,
    HDMI_AUDIO_CODING_MAX
}HI_HDMI_CODING_TYPE_E;
```

[Member]

None

[Note]

None

[See Also]

- HI_MPI_HDMI_SetInfoFrame
- HI_MPI_HDMI_GetInfoFrame

HI_HDMI_AUDIO_SAMPLE_SIZE_E

[Description]

Defines the enumeration of audio sampling sizes.

[Syntax]

```
typedef enum hiHDMI_AUDIO_SAMPLE_SIZE_E
{
    HI_HDMI_AUDIO_SAMPLE_SIZE_STREAM,
    HI_HDMI_AUDIO_SAMPLE_SIZE_16,
    HI_HDMI_AUDIO_SAMPLE_SIZE_20,
    HI_HDMI_AUDIO_SAMPLE_SIZE_24,
}HI_HDMI_AUDIO_SAMPLE_SIZE_E;
```

[Member]

None

[Note]

None

[See Also]

- HI_MPI_HDMI_SetInfoFrame
- HI_MPI_HDMI_GetInfoFrame

HI_HDMI_AUDIO_SAMPLE_FREQ_E

[Description]

Defines the enumeration of audio sampling frequencies.

[Syntax]

```
typedef enum hiHDMI_AUDIO_SAMPLE_FREQ_E
{
    HI_HDMI_AUDIO_SAMPLE_FREQ_STREAM,
    HI_HDMI_AUDIO_SAMPLE_FREQ_32000,
    HI_HDMI_AUDIO_SAMPLE_FREQ_44100,
    HI_HDMI_AUDIO_SAMPLE_FREQ_48000,
    HI_HDMI_AUDIO_SAMPLE_FREQ_88200,
    HI_HDMI_AUDIO_SAMPLE_FREQ_96000,
    HI_HDMI_AUDIO_SAMPLE_FREQ_176400,
    HI_HDMI_AUDIO_SAMPLE_FREQ_192000,
}HI_HDMI_AUDIO_SAMPLE_FREQ_E;
```

[Member]

None

[Note]

None

[See Also]

- HI_MPI_HDMI_SetInfoFrame
- HI_MPI_HDMI_GetInfoFrame

HI_HDMI_LEVEL_SHIFT_VALUE_E

[Description]

Defines the enumeration of the audio shift information.

[Syntax]

```
typedef enum hiHDMI_LEVEL_SHIFT_VALUE_E
{
    HI_HDMI_LEVEL_SHIFT_VALUE_0_DB,
    HI_HDMI_LEVEL_SHIFT_VALUE_1_DB,
    HI_HDMI_LEVEL_SHIFT_VALUE_2_DB,
    HI_HDMI_LEVEL_SHIFT_VALUE_3_DB,
    HI_HDMI_LEVEL_SHIFT_VALUE_4_DB,
    HI_HDMI_LEVEL_SHIFT_VALUE_5_DB,
    HI_HDMI_LEVEL_SHIFT_VALUE_6_DB,
    HI_HDMI_LEVEL_SHIFT_VALUE_7_DB,
    HI_HDMI_LEVEL_SHIFT_VALUE_8_DB,
    HI_HDMI_LEVEL_SHIFT_VALUE_9_DB,
    HI_HDMI_LEVEL_SHIFT_VALUE_10_DB,
    HI_HDMI_LEVEL_SHIFT_VALUE_11_DB,
    HI_HDMI_LEVEL_SHIFT_VALUE_12_DB,
    HI_HDMI_LEVEL_SHIFT_VALUE_13_DB,
```

```
    HI_HDMI_LEVEL_SHIFT_VALUE_14_DB,
    HI_HDMI_LEVEL_SHIFT_VALUE_15_DB,
} HI_HDMI_LEVEL_SHIFT_VALUE_E;
```

[Member]

None

[Note]

None

[See Also]

- HI_MPI_HDMI_SetInfoFrame
- HI_MPI_HDMI_GetInfoFrame

HI_HDMI_LFE_PLAYBACK_LEVEL_E

[Description]

Defines the enumeration of the audio playback information.

[Syntax]

```
typedef enum hiHDMI_LFE_PLAYBACK_LEVEL_E
{
    HI_HDMI_LFE_PLAYBACK_NO,
    HI_HDMI_LFE_PLAYBACK_0_DB,
    HI_HDMI_LFE_PLAYBACK_10_DB,
    HI_HDMI_LFE_PLAYBACK_RESEVED,
}HI_HDMI_LFE_PLAYBACK_LEVEL_E;
```

[Member]

None

[Note]

- HI_HDMI_LFE_PLAYBACK_RESEVED is currently not supported. When this parameter is set, HI_ERR_HDMI_INVALID_PARA is returned.
- HI_ERR_HDMI_INVALID_PARA is returned when a value that is not enumerated is set.

[See Also]

- HI_MPI_HDMI_SetInfoFrame
- HI_MPI_HDMI_GetInfoFrame

HI_HDMI_QUANTIZATION_E

[Description]

Defines the enumeration of the quantization range for the CSC output.

[Syntax]

```
typedef enum hiHDMI_QUANTIZATION_E
{
    HDMI_QUANTIZATION_LIMITED_RANGE,
    HDMI_QUANTIZATION_FULL_RANGE,
    HDMI_QUANTIZATION_BUTT
}HI_HDMI_QUANTIZATION_E;
```

[Member]

None

[Note]

None

[See Also]

- HI_MPI_HDMI_SetAttr
- HI_MPI_HDMI_GetAttr

HI_HDMI_AUDIO_SPEAKER_E

[Description]

Defines the enumeration of the audio speaker capabilities.

[Syntax]

```
typedef enum hiHDMI_AUDIO_SPEAKER_E
{
    HDMI_AUDIO_SPEAKER_FL_FR   ,
    HDMI_AUDIO_SPEAKER_LFE     ,
    HDMI_AUDIO_SPEAKER_FC      ,
    HDMI_AUDIO_SPEAKER_RL_RR   ,
    HDMI_AUDIO_SPEAKER_RC      ,
    HDMI_AUDIO_SPEAKER_FLC_FRC ,
    HDMI_AUDIO_SPEAKER_RLC_RRC ,
    HDMI_AUDIO_SPEAKER_FLW_FRW ,
    HDMI_AUDIO_SPEAKER_FLH_FRH ,
    HDMI_AUDIO_SPEAKER_TC      ,
    HDMI_AUDIO_SPEAKER_FCH     ,
    HDMI_AUDIO_SPEAKER_BUTT    ,
} HI_HDMI_AUDIO_SPEAKER_E;
```

[Member]

None

[Note]

None

[See Also]

HI_MPI_HDMI_GetSinkCapability

4 Error Codes

Table 4-1 describes the error codes of the HDMI APIs

Table 4-1 Error codes for HDMI APIs

Error Code	Macro Definition	Description
0xA0288001	HI_ERR_HDMI_NOT_INIT	The HDMI is not initialized.
0xA0288002	HI_ERR_HDMI_INVALID_PARA	The parameter is invalid.
0xA0288003	HI_ERR_HDMI_NUL_PTR	The pointer is null.
0xA0288004	HI_ERR_HDMI_DEV_NOT_OPEN	The HDMI is disabled.
0xA0288005	HI_ERR_HDMI_DEV_NOT_CONNECT	The HDMI is disconnected.
0xA0288006	HI_ERR_HDMI_READ_SINK_FAILED	The HDMI fails to read the sink.
0xA0288007	HI_ERR_HDMI_INIT_ALREADY	The HDMI is initialized.
0xA0288008	HI_ERR_HDMI_CALLBACK_ALREADY	The HDMI callback is registered.
0xA0288009	HI_ERR_HDMI_INVALID_CALLBACK	The callback function is invalid.
0xA028800A	HI_ERR_HDMI_FEATURE_NO_SUPPORT	Not supported.
0xA028800B	HI_ERR_HDMI_BUS_BUSY	Bus busy flag.
0xA028800C	HI_ERR_HDMI_READ_EVENT_FAILED	The HDMI failed to read the event.
0xA028800D	HI_ERR_HDMI_NOT_START	The HDMI failed to be started.

Error Code	Macro Definition	Description
0xA028800E	HI_ERR_HDMI_READ_EDID_FAILED	The HDMI failed to read the EDID.
0xA028800F	HI_ERR_HDMI_INIT_FAILED	The HDMI failed to be initialized.
0xA0288010	HI_ERR_HDMI_CREATE_TESK_FAILED	The HDMI kernel failed to create tasks.
0xA0288011	HI_ERR_HDMI_MALLOC_FAILED	The HDMI failed to allocate the memory.
0xA0288012	HI_ERR_HDMI_FREE_FAILED	The HDMI failed to free the memory.
0xA0288013	HI_ERR_HDMI_PTHREAD_CREATE_FAILED	The HDMI failed to create the thread.
0xA0288014	HI_ERR_HDMI_PTHREAD_JOIN_FAILED	The HDMI failed to wait for the thread to finish.
0xA0288015	HI_ERR_HDMI_STRATEGY_FAILED	The HDMI kernel adaptive policy failed.
0xA0288016	HI_ERR_HDMI_SET_ATTR_FAILED	The HDMI failed to set attributes.
0xA0288017	HI_ERR_HDMI_CALLBACK_NOT_REGISTER	The HDMI callback function is not registered.
0xA0288018	HI_ERR_HDMI_CEC_CALLBACK_REREGISTER	The CEC callback function is repeatedly registered.
0xA0288019	HI_ERR_HDMI_UNKNOWN_COMMAND	Unknown HDMI command
0xA028801A	HI_ERR_HDMI_MUTEX_LOCK_FAILED	The HDMI failed to lock.

5 Proc Debugging Information

[General Status Debugging Information About HDMI Software and Hardware]

```
# cat /proc/umap/hdmi0
[HDMI] Version:[Hi35xx_MPP_V1.0.0.0 B010 Debug] Build Time:[Sep 14 2016,
12:42:29]
HDMI Version:  2.0.0.201600910.0
-------------------------- APPAttr ----------------------------
HDMIEnable    : YES               DefaultAction  : HDMI
VideoEnable   : YES               AudioEnable    : YES
AviInfoEnable  : YES              AudioInfoEnable: YES
xvYCCMode     : NO                HDCPEnable    : NO
DeepColorMode  : 24               SpdInfoEnable  : NO
OutColorSpace  : YCbCr444          MpegInfoEnable : NO
ColorSpaceAdapt: YES              DeepColorAdapt : YES
DebugEnable   : NO                CtsAuthEnable  : NO
enHDCPMode    : AUTO
-------------------------- SWStatus ---------------------------
ThreadRun     : YES               RunStatus     : OPEN START
TMDSMode      : HDMI1.4
KernelCnt     : 0                 UserCnt       : 1
KCallBack     : YES               UCallBackCnt   : 0
TransitState   : BOOT->APP
EmiEnable     : NO
-------------------------- HWStatus ---------------------------
HotPlug       : YES               Rsen          : NO
PhyOutputEnable: YES               PhyPowerEnable : YES
TMDSMode      : HDMI1.4            AvMute        : NO
-------------------------- Detect Timming ---------------------
SyncSwEnable   : NO               HsyncPolarity  : N
Progressive    : NO               VsyncPolarity  : N
HsyncTotal    : 1650              HactiveCnt    : 1280
VsyncTotal    : 762               VactiveCnt    : 720
```

```
EmiEnable     : NO
EmiDebugEnable : NO
----------------- TaskID=1918 Event Pool[0] Status ------------
CNT|ErrTotal|HPD|UnHPD|EdidFail|HdcpFail|HdcpSucc|RsenCon|RsenDis|HdcpUsr
WR:|0       |0  |1    |0       |0       |0       |0      |0      |0
RD:|0       |0  |1    |0       |0       |0       |0      |0      |0
Memory[WkFlg=0 |RdAble= 0| RdPtr=1 | WrPtr=1 ]:
```

[Analysis]

Records information about the HDMI output management module.

[Parameter Description]

Parameter		Description
APPAttr	HDMIEnable	Whether to enable the HDMI mode Value: {YES, NO}
	DefaultAction	Default working mode Value: {NONE, HDMI, DVI, UNKNOWN}
	VideoEnable	Whether the user enables the video output Value: {YES, NO}
	AudioEnable	Whether the user enables the audio Value: {YES, NO}
	AudioInfoEnable	Whether the user enables the audio information frame Value: {YES, NO}
	xvYCCMode	Whether the user enables the xvYCC output Value: {YES, NO} Note: The Hi35xx does not support the xvYCC.
	bHDCPEnable	Whether the user enables the HDCP Value: {YES, NO} Note: The Hi35xx does not support the HDCP.
	DeepColorMode	Picture color depth configured by the user Value: {24, 30, 36, 48, OFF, UNKNOWN}
	SpdInfoEnable	Whether the user enables the SPD information frame Value: {YES, NO} Note: The Hi35xx does not support the SPD information frame.
	OutColorSpace	Output color space configured by the user Value: {RGB, YCbCr422, YCbCr444, YCbCr420, BUTT}

Parameter		Description
	MpegInfoEnable	Whether the user enables the output of the MPEG information frame Value: {YES, NO} Note: The Hi35xx does not support the MPEG information frame.
	ColorSpaceAdapt	Whether the user enables the color space adaptation policy Value: {YES, NO}
	DeepColorAdapt	Whether the user enables the DeepColor adaptation policy Value: {YES, NO}
	DebugEnable	Whether the user enables the debug mode Value: {YES, NO} Note: The Hi35xx does not support the debug mode.
	CtsAuthEnable	Whether the user enables the CTS authentication mode Value: {YES, NO}
	enHDCPMode	HDCP mode configured by the user Value: {AUTO, HDCP1.4, HDCP2.2, UNKNOWN} Note: The Hi35xx does not support the HDCP.
	DrmInfoEnable	Whether the user enables the DRM information frame Value: {YES, NO}
SWStatus	ThreadRun	Whether the HDMI drive thread is running Value: {YES, NO}
	RunStatus	HDMI running status Value: {NONE, OPEN, START, STOP, CLOSE}
	TMDSMode	TMDS working mode Value: {NONE, DVI, HDMI1.4, HDMI2.0, AUTO, UNKNOWN}
	KernelCnt	Number of HDMI devices enabled by the kernel
	UserCnt	Number of HDMI devices enabled by the user
	KCallBack	Whether the HDMI callback is registered Value: {YES, NO}
	UCallBackCnt	Number of user callback times
	TransitState	Record of states during the boot process Value: { BOOT->MCE, MCE->APP, BOOT->APP}

Parameter		Description
	EmiEnable	Whether the spread spectrum function is enabled Value: {YES, NO}
HWStatus	HotPlug	HotPlug state of the hardware Value: {YES, NO}
	Rsen	Rsen state of the hardware Value: {YES, NO}
	PhyOutputEnable	Whether the HDMI PHY is enabled Value: {YES, NO}
	PhyPowerEnable	Whether the HDMI PHY is powered on Value: {YES, NO}
	TMDSMode	Current TMDS working mode of the hardware Value: {NONE, DVI, HDMI1.4, HDMI2.0, AUTO, UNKNOWN}
	AvMute	Whether to enable audio and video mute (display device) Value: {YES, NO}
Detect Timming	SyncSwEnable	Whether to use the software polar configuration Value: {YES, NO}
	HsyncPolarity	Hsync polarity logical detection value Value: {P, N}
	Progressive	Whether to output line by line Value: {YES, NO}
	VsyncPolarity	Vsync polarity logical detection value Value: {P, N}
	HsyncTotal	Logical detection value of the total number of pixels in a line
	HactiveCnt	Logical detection value of the number of valid pixels in a line
	VsyncTotal	Logical detection value of the total number of lines in a field
	VactiveCnt	Logical detection value of the number of valid lines in a field
	EmiEnable	Spread spectrum enable flag Value: {YES, NO}
	EmiDebugEnable	Debugging mode enable flag of the spread spectrum Value: {YES, NO}

Parameter		Description
TaskID Event Pool Status	ErrTotal	Total number of failed events
	HPD	Number of insertion times for the hot plug event
	UnHPD	Number of removal times for the hot plug event
	EdidFail	Number of times that the EDID fails to be read
	HdcpFail	Number of times that the HDCP authentication fails
	HdcpSucc	Number of times that the HDCP authentication is successful
	RsenCon	Number of times that Rsen is connected
	RsenDis	Number of times that Rsen is disconnected
	HdcpUsr	Number of times that HDCP is configured by the user
	WkFlg	Wakeup flag
	RdAble	Number of readable events in the event pool
	RdPtr	Read pointer to the events in the event pool
	WrPtr	Write pointer to the events in the event pool

Note: **N/A**, **NONE**, or **UNKNOWN** indicates unknown or invalid; **Reserved** indicates reserved or unknown; **ERROR** indicates erroneous.

[HDMI Audio Debugging Information]

```
# cat /proc/umap/hdmi0_vo
[HDMI] Version:[Hi35xx_MPP_V1.0.0.0 B010 Debug] Build Time:[Sep 14 2016,
12:42:29]
HDMI Version:  2.0.0.201600910.0
------------ AudioAttr -------------------- AudioIfno ---------
SoundIntf     : I2S               |AudioInfoEnable: YES
CodeType      : STREAM            |CodeType      : STREAM
ChannelCnt    : 2_CH              |ChannelCnt    : 2_CH
SampleFreq    : 48000             |SampleFreq    : STR_HEADER
SampleDepth   : 16                |SampleDepth   : 16
DownSample    : NO                |SampleSize    : STR_HEADER
------------ AudioPath ----------|DownMixInhibit : NO
AudioEnable   : YES               |LevelShiftValue: 0
AudioMute     : NO                |LFEPlayBack   : UNKNOWN
SoundIntf     : I2S               |Channel/SpeakerAlloc: 0x00 (0)
ChannelCnt    : 2_CH              |AudioInfoRawData:
```

```
SampleFreq     : 48000              | 84 01 0a 70  01 00 00 00
SampleDepth    : 16                 | 00 00 00 00  00 00
DownSample     : NO                 |
Ref_CTS        : 74250              |
Reg_CTS        : 120733             |
Ref_N          : 6144               |
Reg_N          : 6144               |
```

[Analysis]

Records the current HDMI audio working state.

[Parameter Description]

Parameter		Description
AudioAttr	SoundIntf	Type of the HDMI audio interface configured by the user Value: {I2S, SPDIF, HBRA, UNKNOWN}
	CodeType	Audio encoding type Value: {STREAM, L-PCM, AC3, MPEG1, MP3, MPEG2, AAC_LC, DTS, ATRAC, OneBitAudio, EAC3, DTS-HD, MAT, DST, WMA_PRO, Reserved, UNKNOWN} Note: For details, see the *EIA-CEA-861-D (F)*.
	ChannelCnt	Number of audio channels Value: {STR_HEADER, 2_CH, 3_CH, 4_CH, 5_CH, 6_CH, 7_CH, 8_CH} Note: For details, see the *EIA-CEA-861-D (F)*.
	SampleFreq	Audio sampling rate Note: The Hi35xx only supports sampling rates lower than 48000 Hz.
	SampleDepth	Sampling depth (bit width)
	DownSample	Whether down sampling is performed Value: {YES, NO}
AudioIfno	AudioInfoEnable	Whether the audio information frame is enabled Value: {YES, NO}
	CodeType	Audio encoding type Value: {STREAM, L-PCM, AC3, MPEG1, MP3, MPEG2, AAC_LC, DTS, ATRAC, OneBitAudio, EAC3, DTS-HD, MAT, DST, WMA_PRO, Reserved, UNKNOWN} Note: For details, see the *EIA-CEA-861-D (F)*.
	ChannelCnt	Number of audio channels Value: {STR_HEADER, 2_CH, 3_CH, 4_CH, 5_CH, 6_CH, 7_CH, 8_CH}

Parameter		Description
		Note: For details, see the *EIA-CEA-861-D (F)*.
	SampleFreq	Audio sampling rate Value: {STR_HEADER, 32 kHz, 44.1 kHz, 48 kHz, 88.2 kHz, 96 kHz, 176.4 kHz, 192 kHz} Note: The Hi35xx does not support sampling rates lower than 32 kHz currently.
	SampleDepth	Sampling depth
	SampleSize	Sampling size (bit width) Value: {STR_HEADER, 16 bits, 20 bits, 24 bits}
	DownMixInhibit	Down mixing inhibit flag Value: {YES, NO} Note: For details, see the *EIA-CEA-861-D (F)*.
	LevelShiftValue	Level shift value Value: 0 dB−15 dB Note: For details, see the *EIA-CEA-861-D (F)*.
	LFEPlayBack	LFE playback level information Value: {UNKNOWN, 0 dB, +10 dB, Reserved} Note: For details, see the *EIA-CEA-861-D (F)*.
	Channel/SpeakerAlloc	Channel/Speaker allocation Value: 0x00−0xff Note: This parameter is expressed in hexadecimal or decimal. For details, see the *EIA-CEA-861-D (F)*.
	AudioInfoRawData	Raw data of the audio information frame
AudioPath	AudioEnable	Whether to enable the audio of the HDMI hardware Value: {YES, NO}
	AudioMute	Whether to enable the audio mute function Value: {YES, NO}
	SoundIntf	Audio interface type Value: {I2S, SPDIF, HBRA, UNKNOWN} Note: The Hi35xx supports only the I^2S audio interface currently.
	ChannelCnt	Number of audio output channels Value: {STR_HEADER, 2_CH, 3_CH, 4_CH, 5_CH, 6_CH, 7_CH, 8_CH} Note: For details, see the *EIA-CEA-861-D (F)*.

Parameter		Description
	SampleFreq	Current audio sampling rate Value: {STR_HEADER, 32 kHz, 44.1 kHz, 48 kHz, 88.2 kHz, 96 kHz, 176.4 kHz, 192 kHz} Note: The Hi35xx does not support sampling rates higher than 48000 Hz.
	SampleDepth	Current audio sampling depth
	DownSample	Whether audio down sampling is performed Value: {YES, NO}
	Ref_CTS	Theoretic CTS value specified by the protocol
	Reg_CTS	CTS value that is output Note: For details, see the *EIA-CEA-861-D (F)*.
	Ref_N	Theoretic N value specified by the protocol
	Reg_N	N value that is output Note: For details, see the *EIA-CEA-861-D (F)*.

Note: **N/A**, **NONE**, or **UNKNOWN** indicates unknown or invalid; **Reserved** indicates reserved or unknown; **ERROR** indicates erroneous.

[HDMI Video Debugging Information]

```
------------ VideoAttr --------------------------- AVIIfno --------
VideoTiming    : 1920*1080p60 16:9    |AVIInfoEnable  : YES
DispFmt        : 1080P@60             |CurrentFormat  : 1920*1080p60
16:9(VIC=16)
PixelClk       : 148500               |VSIFormat      : (HDMI_VIC= 0)
InBitDepth     : 10 Bit               |BarDataPresent : NONE
InColorSpace   : YCbCr444             |ColorSpace     : YCbCr444
Colorimetry    : ITU-R BT.709         |Colorimetry    : ITU-R BT.709
PicAspectRatio : 16:9                 |PicAspectRatio : 16:9
ActAspectRatio : PICTURE              |ActAspectRatio : PICTURE
PixelRepeat    : 1                    |PixelRepeat    : No Repetition
YCCQuantization: LIMITED              |YCCQuantization: LIMITED
RGBQuantization: DEFAULT              |RGBQuantization: DEFAULT
ExtColorimetry : XV_YCC601            |ExtColorimetry : XV_YCC601
StereoMode     : NONE                 |ItContentValid : NO
bVSyncPol      : 0                    |bHSyncPol      : 0
------------ VedioPath ---------------|ITContentType  : GRAPHICS
VideoMute      : NO                   |PicScaling     : UNKNOWN
OutBitDepth    : 08 Bit               |ActFmtPresent  : YES
OutColorSpace  : YCbCr444             |ScanInfo       : NONE
```

```
YCbCr420_422   : NO                |AVIInfoRawData :
YCbCr422_444   : NO                | 82 02 0d 67  50 a8 00 10
YCbCr444_422   : NO                | 00 00 00 00  00 00 00 00
YCbCr422_420   : NO                | 00
RGB2YCbCr      : NO                |VSInfoRawData  :
YCbCr2RGB      : NO                | 81 01 07 68  03 0c 00 00
Dither        : 10_8               | 00 00 00
DeepColorMode  : 24 Bit(OFF)       |
```

[Analysis]

Records the current HDMI video working state.

[Parameter Description]

Parameter		Description
VideoAttr	VideoTiming	Current video timing Note: For details, see the *EIA-CEA-861-D (F)* and *VESA Display Monitor Timing Standard*.
	DispFmt	Current video standard Note: For details, see the *EIA-CEA-861-D (F)* and *VESA Display Monitor Timing Standard*.
	PixelClk	Pixel clock Note: For details, see the *EIA-CEA-861-D (F)* and *VESA Display Monitor Timing Standard*.
	InBitDepth	Color depth output by the VO to the HDMI Value: {8 bits, 10 bits, 12 bits, 16 bits, UNKNOWN} Note: **UNKNOWN** indicates that the color depth is unknown.
	InColorSpace	Color space output by the VO to the HDMI Value: {RGB, YCbCr422, YCbCr444, YCbCr420, UNKNOWN} Note: **UNKNOWN** indicates that the color space is unknown.
	Colorimetry	Color gamut output by the VO to the HDMI Value: {No Data, SMPTE 170M, ITU-R BT.709, Extended}
	PicAspectRatio	Aspect ratio of the input video Value: {NONE, 4:3, 16:9, FUTURE, UNKNOWN} Note: **UNKNOWN** indicates that the aspect ratio is unknown.
	ActAspectRatio	Aspect ratio of the valid picture of the input video Value: {Reserved, 16:9_TOP, 14:9_TOP, 16:9_CENTER, PICTURE, 4:3, 16:9, 14:9,

Parameter		Description
		4:3_SP_14_9, 16:9_SP_14_9, 16:9_SP_4_3, UNKNOWN} Note: **UNKNOWN** indicates that the aspect ratio is unknown.
	PixelRepeat	Number of times that the pixels are repeated
	YCCQuantization	YCC quantization range Value: {LIMITED, FULL, UNKNOWN} Note: **UNKNOWN** indicates that the range is unknown.
	RGBQuantization	RGB quantization range Value: {DEFAULT, LIMITED, FULL, UNKNOWN} Note: **UNKNOWN** indicates that the range is unknown.
	ExtColorimetry	Extended color gamut of the input picture Value: {XV_YCC601, XV_YCC709, S_YCC601, ADOBE_YCC601, ADOBE_RGB, BT2020_YCC, BT2020_RGB/cYCC, UNKNOWN}
	StereoMode	Stereo mode Value: {FRAME_PACK, FIELD_ALTER, LINE_ALTERN, SBS_FULL, L_DEPTH, L_DEPTH_GGD, TAndB, Reserved, SByS_HALF, NONE} Note: **NONE** indicates that the mode is unknown.
	HvSyncPol	Horizontal/Vertical synchronization polarity Value: {HPVP, HPVN, HNVP, HNVN, UNKNOWN} Note: **H** indicates horizontal; **V** indicates vertical; **P** indicates positive; **N** indicates negative; **UNKNOWN** indicates unknown.
	bVSyncPol	Whether the HDMI enables vertical reverse Value: {0, 1}
VedioPath	VideoMute	Whether the HDMI hardware enables video mute Value: {YES, NO}
	OutBitDepth	Color depth that is output Value: {8bit, 10bit, 12bit, 16bit, UNKNOWN}
	OutColorSpace	Image color space that is output Value: {RGB, YCbCr422, YCbCr444, YCbCr420, UNKNOWN}
	YCbCr420_422	Whether the color space is converted from YCbCr420 into YCbCr422 Value: {YES, NO}
	YCbCr422_444	Whether the color space is converted from YCbCr422

Parameter		Description
		into YCbCr444 Value: {YES, NO}
	YCbCr444_422	Whether the color space is converted from YCbCr444 into YCbCr422 Value: {YES, NO}
	YCbCr422_420	Whether the color space is converted from YCbCr422 into YCbCr420 Value: {YES, NO}
	RGB2YCbCr	Whether the color space is converted from RGB into YCbCr Value: {YES, NO}
	YCbCr2RGB	Whether the color space is converted from YCbCr into RGB
	Dither	Dither working mode Value: {12_10, 12_8, 10_8, disable}
	DeepColorMode	Deep Color mode for logic working Value: {24 Bit, 30 Bit, 36 Bit, 48 Bit, 24 Bit(OFF), UNKNOWN}
AVIIfno	AVIInfoEnable	Whether the AVI information frame is enabled Value: {YES, NO}
	CurrentFormat	Current video standard/VIC code Note: For details, see the *EIA-CEA-861-D (F)*.
	VSIFormat	4K non-3D standard/VIC code Note: For details, see the *HDMI Specification 2.0*.
	BarDataPresent	Bar information Value: {NONE, HnVp, HpVn, HpVp} Note: **H** indicates horizontal; **V** indicates vertical; **P** indicates positive; **N** indicates negative. For details, see the *EIA-CEA-861-D (F)*.
	ColorSpace	Video color space Value: {RGB, YCbCr422, YCbCr444, YCbCr420, UNKNOWN} Note: **UNKNOWN** indicates that the color space is unknown.
	Colorimetry	Color gamut Value: {No Data, SMPTE 170M, ITU-R BT.709, Extended}
	PicAspectRatio	Video aspect ratio

Parameter		Description
		Value: {NONE, 4:3, 16:9, FUTURE, UNKNOWN} Note: **UNKNOWN** indicates that the aspect ratio is unknown.
	ActAspectRatio	Aspect ratio of the valid video picture Value: {Reserved, 16:9_TOP, 14:9_TOP, 16:9_CENTER, PICTURE, 4:3, 16:9, 14:9, 4:3_SP_14_9, 16:9_SP_14_9, 16:9_SP_4_3, UNKNOWN} Note: **UNKNOWN** indicates that the aspect ratio is unknown.
	PixelRepeat	Number of times that the pixels are repeated
	YCCQuantization	YCC quantization range Value: {LIMITED, FULL, UNKNOWN} Note: **UNKNOWN** indicates that the range is unknown.
	RGBQuantization	RGB quantization range Value: {DEFAULT, LIMITED, FULL, UNKNOWN} Note: **UNKNOWN** indicates that the range is unknown.
	ExtColorimetry	Extended color gamut of the picture Value: {XV_YCC601, XV_YCC709, S_YCC601, ADOBE_YCC601, ADOBE_RGB, BT2020_YCC, BT2020_RGB/YCC, UNKNOWN} Note: **UNKNOWN** indicates that the color gamut is unknown.
	ItContentValid	Whether the IT content is valid Value: {YES, NO} Note: For details, see the *EIA-CEA-861-D (F)*.
	bHSyncPol	Whether to enable the horizontal reverse Value: {0, 1}
	ITContentType	IT content type Value: {GRAPHICS, PHOTO, CINEMA, GAME}
	PicScaling	Picture equalization Value: {UNKNOWN, HpVn, HnVp, HpVp} Note: **H** indicates horizontal; **V** indicates vertical; **P** indicates positive; **N** indicates negative. For details, see the *EIA-CEA-861-D (F)*.
	ActFmtPresent	Whether the valid information is displayed Value: {YES, NO} Note: For details, see the *EIA-CEA-861-D (F)*.
	ScanInfo	Scan flag bit

Parameter		Description
		Value: {NONE, OVER_SCAN, UNDERS_SCAN, Reserved}
	AVIInfoRawData	Raw data of the AVI information frame
	VSInfoRawData	Raw data of the VSIF information frame

Note: **N/A**, **NONE**, or **UNKNOWN** indicates unknown or invalid; **Reserved** indicates reserved or unknown; **ERROR** indicates erroneous.

[EDID Debugging Information]

```
# cat /proc/umap/hdmi0_sink
[HDMI] Version:[Hi35xx_MPP_V1.0.0.0 B010 Debug] Build Time:[Sep 14 2016,
12:42:29]
HDMI Version:  2.0.0.201600910.0
----------------------- EDIDRawData -----------------------
/*00H:*/ 0x00,0xff,0xff,0xff,  0xff,0xff,0xff,0x00,  0x4d,0xd9,0x03,0xc8,
0x01,0x01,0x01,0x01,
/*0fH:*/ 0x01,0x19,0x01,0x03,  0x80,0x90,0x51,0x78,  0x0a,0x0d,0xc9,0xa0,
0x57,0x47,0x98,0x27,
/*1fH:*/ 0x12,0x48,0x4c,0x21,  0x08,0x00,0x81,0x80,  0xa9,0xc0,0x71,0x4f,
0xb3,0x00,0x01,0x01,
/*2fH:*/ 0x01,0x01,0x01,0x01,  0x01,0x01,0x02,0x3a,  0x80,0x18,0x71,0x38,
0x2d,0x40,0x58,0x2c,
/*3fH:*/ 0x45,0x00,0x9f,0x29,  0x53,0x00,0x00,0x1e,  0x01,0x1d,0x00,0x72,
0x51,0xd0,0x1e,0x20,
/*4fH:*/ 0x6e,0x28,0x55,0x00,  0x9f,0x29,0x53,0x00,  0x00,0x1e,0x00,0x00,
0x00,0xfc,0x00,0x53,
/*5fH:*/ 0x4f,0x4e,0x59,0x20,  0x54,0x56,0x20,0x20,  0x2a,0x30,0x32,0x0a,
0x00,0x00,0x00,0xfd,
/*6fH:*/ 0x00,0x30,0x3e,0x0e,  0x46,0x3c,0x00,0x0a,  0x20,0x20,0x20,0x20,
0x20,0x20,0x01,0xdf,
/*7fH:*/ 0x02,0x03,0x60,0xf0,  0x5b,0x61,0x60,0x5d,  0x5e,0x5f,0x62,0x1f,
0x10,0x14,0x05,0x13,
/*8fH:*/ 0x04,0x20,0x22,0x3c,  0x3e,0x12,0x16,0x03,  0x07,0x11,0x15,0x02,
0x06,0x01,0x65,0x66,
/*9fH:*/ 0x29,0x0d,0x7f,0x07,  0x15,0x07,0x50,0x3d,  0x07,0xbc,0x83,0x0f,
0x00,0x00,0x78,0x03,
/*afH:*/ 0x0c,0x00,0x10,0x00,  0xb8,0x3c,0x2f,0xd0,  0x8a,0x01,0x02,0x03,
0x04,0x01,0x40,0x1f,
/*bfH:*/ 0xc0,0x80,0x90,0xd0,  0xe0,0xf0,0xd6,0x67,  0xd8,0x5d,0xc4,0x01,
0x78,0x80,0x01,0xe2,
/*cfH:*/ 0x00,0xf9,0xe3,0x05,  0xff,0x01,0xe5,0x0f,  0x03,0x00,0x00,0x06,
```

```
0xe3,0x06,0x05,0x01,
/*dfH:*/ 0x01,0x1d,0x80,0x18,  0x71,0x1c,0x16,0x20,  0x58,0x2c,0x25,0x00,
0x9f,0x29,0x53,0x00,
/*efH:*/ 0x00,0x9e,0x00,0x00,  0x00,0x00,0x00,0x00,  0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x6a,
------------------------ SWStatus -----------------------------
CapFromSink        : YES               RawUpdateErrCnt     : 3
CapIsValid         : YES               ParseErrorType      : 0
RawIsValid         : YES               ParseWarnType       : 0x00000000
RawGetErrCnt       : 0                 RawLength           : 256
-------------------------- BasicCap -------------------------
HDMI1.4Support      : YES               1stBlockVersion     : 1.3
HDMI2.0Support      : YES               ManufacturerName    : SNY
MaxTMDSClock(MHz)   : 600               ProductCode         : 51203
SerialNumber        : 16843009          WeekOfManufacture   : 1
MaxDispWidth        : 144               MaxDispHeight       : 81
SCDCSupport         : YES               YearOfManufacture   : 2015
DVIDualSupport      : NO                CECAddrIsValid      : YES
AISupport          : YES               CECAddr            : 01.00.00.00
ExtBlockCnt         : 1                 SpeakerSupport      : FL_FR LFE FC
RL_RR
RgbQuanSelectable   : NO                YccQuanSelectable   : NO
--------------------------- VidoCap -----------------------
NativeFormat        : 3840X2160P60 16:9(VIC 97)
ColorSpace         : RGB444 YCbCr444 YCbCr422 YCbCr420
DeepColor          : RGB_30Bit RGB_36Bit YCbCr444_SameRGB
YCbCr420DeepColor   : 30Bit
YCbCr420[Also]      : 97 96 101 102
YCbCr420[Olny]      :
Colorimetry         : xvYCC601 xvYCC709 sYCC601 AdobleYCC601 AdobleRGB
BT2020cYCC BT2020YCC BT2020RGB
--------------------------- FormatCap ----------------------
3840X2160P60 16:9   3840X2160P50 16:9   3840X2160P24 16:9   3840X2160P25
16:9
3840X2160P30 16:9   4096X2160P24 256:135 1080P_50 16:9       1080P_60 16:9
1080i_50 16:9       1080i_60 16:9       720P_50 16:9        720P_60 16:9
1080P_24 16:9       1080P_30 16:9       720P_24 16:9        720P_30 16:9
576P_50 16:9        PAL 16:9            480P_60 16:9        NTSC 16:9
576P_50 4:3         PAL 4:3             480P_60 4:3         NTSC 4:3
640X480_60 4:3      4096X2160P50 256:135 4096X2160P60 256:135 3840X2160P30
16:9
3840X2160P25 16:9   3840X2160P24 16:9   4096X2160P24 256:135 1280x1024
1600x900           1152x864           1680x1050           V800X600_60
V800X600_56         V640X480_75
```

```
---------------------------- 3DCap -------------------------
3DSupport          : YES               3DOsdDisparity     : NO
3DDualView         : NO                3DIndepView        : NO
3DTypeSupport      : TAndB SByS_HALF
---------------------------- AudioCap ----------------------
NO.0:
CodeType           : L-PCM             MaxChannelNum      : 6
MaxBitRate(KHz)    : N/A               BitDepth           : 16 20 24
SampleRate(Hz)     : 32000 44100 48000 88200 96000 176400 192000
NO.1:
CodeType           : AC3               MaxChannelNum      : 6
MaxBitRate(KHz)    : 640               BitDepth           : N/A
SampleRate(Hz)     : 32000 44100 48000
NO.2:
CodeType           : DTS               MaxChannelNum      : 6
MaxBitRate(KHz)    : 1504              BitDepth           : N/A
SampleRate(Hz)     : 32000 44100 48000
---------------------------- HdrCap ------------------------
HdrEotfSdr         : NO                HdrEotfHdr         : NO
HdrEotfSt2084      : NO                HdrEotfHLG         : NO
MaxLum             : 0                 AvgLum             : 0
MinLum             : 0
---------------------------- DolbyCap ----------------------
DolbyOUI           : 0x0               DolbyCapsVer       : 0
DolbySu_Y422       : NO                DolbySu_2160P60    : NO
DolbyRed_X         : 0                 DolbyRed_Y         : 0
DolbyGreen_X       : 0                 DolbyGreen_Y       : 0
DolbyBlue_X        : 0                 DolbyBlue_Y        : 0
DolbyMinLum        : 0                 DolbyMaxLum        : 0
DolbyWhite_X       : 0                 DolbyWhite_Y       : 0
DMmajorVer         : 0                 DMminorVer         : 0
---------------------------- DetailTiming ------------------
[NO.]:HACT|VACT|P/I |PClk|AspW|AspH|HFB |HPW |HBB |VFB |VPW |VBB
|ImgW|ImgH|IHS |IVS |IDV
[  0]:1920|1080|P   |148M|0   |0   |88  |44  |192 |4   |5   |41  |1439|809
|YES |YES |NO
[  1]:257 |1920|P   |12 M|0   |0   |284 |278 |3045|34  |0   |246 |44  |37  |NO
|NO  |NO
```

[Analysis]

Records the EDID of the current monitor.

[Parameter Description]

Parameter		Description
EDIDRawData		Raw EDID data (256 bytes)
SWStatus	CapFromSink	Whether the capability set is from the monitor (sink) Value: {YES, NO} Note: The capability set may be from data used for tests. In this case, the parameter value is **NO**.
	RawUpdateErrCnt	Number of raw data update errors
	CapIsValid	Whether the capability set is valid Value: {YES, NO}
	ParseErrorType	EDID parsing error flag • 0: No error occurs. • 1: Check errors occur. • 2: The data header is incorrect. • 3: The basic data block is not the 1.3 version. • 4: The tag value of the extended data block is unknown. • 5: The CEA value is invalid. • 6 or larger value: reserved
	RawIsValid	Whether the raw data is valid Value: {YES, NO}
	ParseWarnType	EDID parsing warning flag Value range: 0x00000000−0xffffffff Each bit corresponds to a warning. • Bit 0: No warning is reported. • Bit 1: The vendor block is invalid. • Bit 2: The number of DTD blocks exceeds the threshold. • Bit 3: The DTD block is invalid. • Bit 4: There is no extended block. • Bit 5: The number of extended blocks is greater than 4. • Bit 6: The number of audio blocks exceeds the threshold. • Bit 7: The number of VICs exceeds the threshold. • Bit 8: The VIC is invalid. • Bit 9: The VSDB is invalid. • Bit 10: The HFVSDB is invalid. • Bit 11: The SPEAKER DB is invalid. • Bit 12: The number of YCBCR420 VICs exceeds the threshold.

Parameter		Description
		• Bit 13: The data block length is invalid. • Bits 14−31: reserved
	RawGetErrCnt	Number of times that the raw data fails to be obtained
	RawLength	Raw data length
BasicCap	HDMI1.4Support	Whether the sink supports HDMI 1.4 Value: {YES, NO}
	1stBlockVersion	Version number of the first EDID block
	HDMI2.0Support	Whether HDMI 2.0 is supported Value: {YES, NO}
	ManufacturerName	Sink vendor name
	MaxTMDSClock(MHz)	Maximum TMDS clock Note: For details, see the *EIA-CEA-861-D (F)*.
	ProductCode	Sink product number
	SerialNumber	Sink product serial number
	WeekOfManufacture	Week of manufacture for the sink product Note: For details, see the *EIA-CEA-861-D (F)*.
	SCDCSupport	Whether the sink supports SCDC Value: {YES, NO}
	YearOfManufacture	Year of manufacture for the sink product
	DVIDualSupport	Whether the sink supports DVI Dual
	CECAddrIsValid	Whether the sink CEC address is valid Value: {YES, NO}
	AISupport	Whether the sink supports AI Value: {YES, NO}
	CECAddr	CEC physical address
	ExtBlockCnt	Number of EDID extended blocks
	SpeakerSupport	Speaker supported by the sink Value: {FL_FR, LFE, FC, RL_RR, RC, FLC_FRC, RLC_RRC, FLW_FRW, FLH_FRH, TC, FCH, UNKNOWN} Note: For details, see the *EIA-CEA-861-D (F)*.
	RgbQuanSelectable	Whether the YCbCr quantization range of the display device can be selected. Value: {YES, NO}

Parameter		Description
	YccQuanSelectable	Whether the RGB quantization range of the display device can be selected. Value: {YES, NO}
VidoCap	NativeFormat	Best standard/VIC code of the monitor. Note: For details, see the *EIA-CEA-861-D (F)*.
	ColorSpace	Color space supported by the monitor Value: {RGB444, YCbCr444, YCbCr422, YCbCr420} Note: For details, see the *EIA-CEA-861-D (F)*.
	DeepColor	Number of deep color bits Value: {RGB_30Bit, RGB_36Bit, RGB_48Bit, YCbCr444_SameRGB} Note: For details, see the *EIA-CEA-861-D (F)*.
	YCbCr420DeepColor	Number of deep color bits that support YCbCr420 Value: {30 bits, 36 bits, 48 bits} Note: For details, see the *EIA-CEA-861-D (F)*.
	YCbCr420[Also]	The VIC codes of RGB, YCbCr444, YCbCr422, and YCbCr420 are supported. Note: For details, see the *EIA-CEA-861-D (F)*.
	YCbCr420[Olny]	Only the VIC code of YCbCr420 is supported. Note: For details, see the *EIA-CEA-861-D (F)*.
	Colorimetry	Colorimetry Value: {xvYCC601, xvYCC709, sYCC601, AdobleYCC601, AdobleRGB, BT2020cYCC, BT2020YCC, BT2020RGB} Note: For details, see the *EIA-CEA-861-D (F)*.
FormatCap		Standards supported by the sink Note: For details, see the *EIA-CEA-861-D (F)* and *VESA Display Monitor Timing Standard*.
3DCap	3DSupport	Whether the sink supports 3D Value: {YES, NO}
	3DDualView	Whether the sink supports 3D dual view Value: {YES, NO}
	3DTypeSupport	3D display type supported by the sink Value: {FRAME_PACK, FIELD_ALTER, LINE_ALTERN, SBS_FULL, L_DEPTH, L_DEPTH_GGD, TAndB, Reserved, SByS_HALF, NONE}
	3DOsdDisparity	Whether the sink supports 3D OSD disparity

Parameter		Description
		Value: {YES, NO}
	3DIndepView	Whether the sink supports 3D independent view Value: {YES, NO}
AudioCap	NO.x:	Number x
	CodeType	Encoding type Value: {STREAM, L-PCM, AC3, MPEG1, MP3, MPEG2, AAC_LC, DTS, ATRAC, OneBitAudio, EAC3, DTS-HD, MAT, DST, WMA_PRO, Reserved, UNKNOWN} Note: For details, see the *EIA-CEA-861-D (F)*.
	MaxChannelNum	Maximum number of channels
	MaxBitRate(KHz)	Maximum bit rate
	BitDepth	Supported bit depth Value: {0, 8, 16, 18, 20, 24, 32, N/A }
	SampleRate(Hz)	Sampling rate Value: {0, 8000, 11025, 12000, 16000, 22050, 24000, 32000, 44100, 48000, 88200, 96000, 176400, 192000, 768000}
HdrCap	HdrEotfSdr	Whether the monitor supports the SDR EOTF type Note: For details, see the *CEA-861.3*.
	HdrEotfHdr	Whether the monitor supports the HDR EOTF type Note: For details, see the *CEA-861.3*.
	HdrEotfSt2084	Whether the monitor supports the SMPTE ST 2084 Note: For details, see the *CEA-861.3*.
	HdrEotfHLG	Whether the monitor supports the HLG Note: For details, see the *CEA-861.3*.
	MaxLum	Maximum luminance supported by the monitor Note: For details, see the *CEA-861.3*.
	AvgLum	Average luminance supported by the monitor Note: For details, see the *CEA-861.3*.
	MinLum	Minimum luminance supported by the monitor Note: For details, see the *CEA-861.3*.
DolbyCap	DolbyOUI	IEEE OUI value of the Dolby Vision EDID Vendor Specific Video Data Block Note: For details, see the *Dolby Vision HDMI Transmission Specification*.
	DolbyCapsVer	Version number of the Dolby Vision EDID Vendor

Parameter		Description
		Specific Video Data Block
	DolbySu_Y422	Whether the monitor supports Dolby Vision signals in YCbCr422 format Note: For details, see the *Dolby Vision HDMI Transmission Specification*.
	DolbySu_2160P60	Whether the monitor supports 2160P60-standard Dolby Vision signals Note: For details, see the *Dolby Vision HDMI Transmission Specification*.
	DolbyRed_X	X value of the red coordinate on the monitor Note: For details, see the *Dolby Vision HDMI Transmission Specification*.
	DolbyRed_Y	Y value of the red coordinate on the monitor Note: For details, see the *Dolby Vision HDMI Transmission Specification*.
	DolbyGreen_X	X value of the green coordinate on the monitor Note: For details, see the *Dolby Vision HDMI Transmission Specification*.
	DolbyGreen_Y	Y value of the green coordinate on the monitor Note: For details, see the *Dolby Vision HDMI Transmission Specification*.
	DolbyBlue_X	X value of the blue coordinate on the monitor Note: For details, see the *Dolby Vision HDMI Transmission Specification*.
	DolbyBlue_Y	Y value of the blue coordinate on the monitor Note: For details, see the *Dolby Vision HDMI Transmission Specification*.
	DolbyMinLum	Minimum display luminance of the monitor Note: For details, see the *Dolby Vision HDMI Transmission Specification*.
	DolbyMaxLum	Maximum display luminance of the monitor Note: For details, see the *Dolby Vision HDMI Transmission Specification*.
	DolbyWhite_X	X value of the white coordinate on the monitor Note: For details, see the *Dolby Vision HDMI Transmission Specification*. Note: This parameter is supported only when **DolbyCapsVer** is set to **0**.
	DolbyWhite_Y	Y value of the white coordinate on the monitor Note: For details, see the *Dolby Vision HDMI*

Parameter		Description
		Transmission Specification. Note: This parameter is supported only when **DolbyCapsVer** is set to **0**.
	DMmajorVer	Major version number of Dolby Vision Sink device display management Note: This parameter is supported only when **DolbyCapsVer** is set to **0**.
	DMminorVer	Minor version number of Dolby Vision Sink device display management Note: This parameter is supported only when **DolbyCapsVer** is set to **0**.
	DMVersion	Version number of Dolby Vision Sink device display management Note: This parameter is supported only when **DolbyCapsVer** is set to **1**.
	Colorimetry	Version number of Dolby Vision Sink device display management Note: This parameter is supported only when **DolbyCapsVer** is set to **1**.
DetailTiming	NO.	Number
	HACT	Horizontal active pixel
	VACT	Vertical active pixel
	P/I	Progressive/Interlaced mode
	PClk	Pixel clock
	AspW	Width in the aspect ratio
	AspH	Height in the aspect ratio
	HFB	Horizontal front blanking
	HPW	Horizontal pulse width
	HBB	Horizontal back blanking
	VFB	Vertical front blanking
	VPW	Vertical pulse width
	VBB	Vertical back blanking
	ImgW	Picture width
	ImgH	Picture height
	IHS	Whether the horizontal sync pulse is inverted
	IVS	Whether the vertical sync pulse is inverted

Parameter		Description
	IDV	Whether the valid signal is inverted

Note: **N/A**, **NONE**, or **UNKNOWN** indicates unknown or invalid; **Reserved** indicates reserved or unknown; **ERROR** indicates erroneous.