MI SENSOR API

Version 2.03



REVISION HISTORY

Revision No.	Description	Date
2.03	Initial release	11/08/2018

TABLE OF CONTENTS

			DRY	
			NTS	
1.	API	参考		1
	1.1.	概述		1
	1.2.	模块 AP		1
		1.2.1	MI SNR Enable	2
		1.2.2	MI_SNR_Disable	
		1.2.3	MI_SNR_GetPadInfo	
		1.2.4	MI SNR GetPlaneInfo	
		1.2.5	MI_SNR_GetFps	6
		1.2.6	MI_SNR_SetFps	6
		1.2.7	MI_SNR_GetBT656SrcType	
		1.2.8	MI_SNR_QueryResCount	9
		1.2.9	MI_SNR_GetRes	10
		1.2.10	MI_SNR_GetCurRes	11
		1.2.11	MI_SNR_SetRes	12
		1.2.12	MI_SNR_SetOrien	14
		1.2.13	MI_SNR_GetOrien	16
		1.2.14	MI_SNR_SetPlaneMode	17
			MI_SNR_GetPlaneMode	
2.	SEN	SOR 数据	类型	20
	2.1.	MI_SNR	_MAX_PADNUM	21
	2.2.	MI_SNR	_MAX_PLANENUM	21
	2.3.	MI_SNR	_PAD_ID_e	21
	2.4.	MI_SNR	_HDRSrc_e	22
	2.5.	MI_SNR	_HDRHWMode_e	23
	2.6.	MI_SNR	_Anadec_SrcType_e	24
	2.7.	MI_SNR	_Res_t	24
	2.8.	MI_SNR	_Res_List_t	25
	2.9.	MI_SNR	_AttrParallel_t	26
	2.10.	MI_SNR	_MipiAttr_t	28
	2.11.	MI_SNR	_AttrBt656_t	29
	2.12.	MI_SNR	_IntfAttr_u	30
	2.13.	MI_SNR	_PADInfo_t	30
	2.14.	MI_SNR	_PlaneInfo_t	32
3	SEN	SOR 错误	码	34

1. API 参考

1.1. 概述

SNR(sensor)实现获取摄像头接口信息、调整分辨率和帧率等功能。

1.2. 模块 API

API名	功能
MI_SNR_Enable	设置Sensor使能
MI SNR Disable	设置Sensor失能
MI_SNR_GetPadInfo	获取Sensor设备信息
MI_SNR_GetPlaneInfo	获取Sensor通道信息
MI_SNR_GetFps	获取Sensor当前帧率
MI_SNR_SetFps	设置Sensor帧率
MI_SNR_GetBT656SrcType	获取BT656 Sensor源输入格式
MI SNR QueryResCount	获取Sensor支持分辨率的数量
MI_SNR_GetRes	获取索引对应的sensor分辨率
MI SNR GetCurRes	获取Sensor当前分辨率
MI_SNR_SetRes	设置Sensor分辨率
MI SNR GetOrien	获取Sensor翻转属性
MI_SNR_SetOrien	设置Sensor翻转
MI SNR SetPlaneMode	设置Sensor通道模式
MI_SNR_GetPlaneMode	获取设置的Sensor通道模式

1.2.1 MI_SNR_Enable

▶ 功能

设置 SENSOR 对应设备的使能

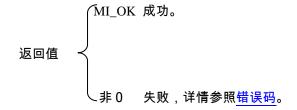
▶ 语法

MI_S32 MI_SNR_Enable(MI_SNR_PAD_ID_e ePADId);

▶ 形参

参数名称	描述	输入/输出
ePADId	SENSOR 设备号。	输入
	取值范围: [0, MI_SNR_MAX_PAD_NUM]。	

▶ 返回值



- ▶ 依赖
- 头文件: mi_sensor_datatype.h、mi_sensor.h
- 库文件: libmi_sensor.a
- ※ 注意
- ◆ 在调用前要保证 SENSOR设备处于未初始化状态。如果SENSOR设备已处于使能状态,可以使用MI SNR Disable来去初始化设备。
- ▶ 相关主题

MI SNR Disable

1.2.2 MI_SNR_Disable

▶ 功能

设置 SENSOR 对应设备失能。

▶ 语法

MI SENSOR API

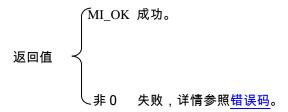
Version 2.03

MI_S32 MI_SNR_Disable(MI_SNR_PAD_ID_e ePADId);

▶ 形参

参数名称	描述	输入/输出
ePADId	SENSOR设备号。	输入
	取值范围: [0, MI SNR MAX PAD NUM)。	

▶ 返回值



▶ 依赖

• 头文件: mi_sensor_datatype.h、mi_sensor.h

• 库文件: libmi_sensor.a

▶ 相关主题

MI SNR Enable

1.2.3 MI_SNR_GetPadInfo

▶ 功能

获取 SENSOR 设备信息。

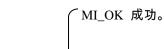
▶ 语法

MI_S32 MI_SNR_GetPadInfo(<u>MI_SNR_PAD_ID_e</u> ePADId, <u>MI_SNR_PADInfo_t</u> *pstPadInfo);

▶ 形参

参数名称	描述	输入/输出
ePADId	SENSOR设备号。	输入
	取值范围: [0, MI_SNR_MAX_PAD_NUM)。	
pstPadInfo	SENSOR设备属性指针	输出

▶ 返回值



返回值

非 0 失败,详情参照错误码。

▶ 依赖

• 头文件: mi_sensor_datatype.h、mi_sensor.h

• 库文件: libmi_sensor.a

▶ 相关主题

无。

1.2.4 MI_SNR_GetPlaneInfo

▶ 功能

获取 SENSOR 通道信息。

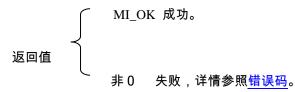
▶ 语法

MI_S32 MI_SNR_GetPadInfo(MI_SNR_PAD_ID_e ePADId, MI_SNR_PlaneInfo_t *pstPadInfo);

▶ 形参

参数名称	描述	输入/输出
ePADId	SENSOR设备号。	输入
	取值范围: [0, MI_SNR_MAX_PAD_NUM)。	
u32PlaneID	SENSOR通道号。	输入
	取值范围: [0, MI SNR MAX PLANE NUM)。	
pstChnInfo	SENSOR通道信息。	输出

▶ 返回值



▶ 依赖

• 头文件: mi_sensor_datatype.h、mi_sensor.h

• 库文件: libmi_sensor.a

▶ 相关主题

无。

1.2.5 MI_SNR_GetFps

▶ 功能

获取 SENSOR 帧率

▶ 语法

MI_S32 MI_SNR_GetFps(MI_SNR_PAD_ID_e ePADId, MI_U32 *pFps);

▶ 形参

参数名称	描述	输入/输出
ePADId	SENSOR设备号。 取值范围: [0, MI_SNR_MAX_PAD_NUM)。	输入
pFps	帧率指针	输出

▶ 返回值

▶ 依赖

• 头文件: mi_sensor_datatype.h、mi_sensor.h

• 库文件: libmi_sensor.a

▶ 相关主题

MI SNR SetFps

1.2.6 MI_SNR_SetFps

返回值

▶ 功能

设置 SENSOR 帧率。

▶ 语法

MI_S32 MI_SNR_GetFps(MI_SNR_PAD_ID_e_ePADId, MI_U32 *pFps);

▶ 形参

参数名称	描述	输入/输出
ePADId	SENSOR设备号。 取值范围: [0, MI_SNR_MAX_PAD_NUM)。	输入
pFps	帧率指针	输出

▶ 返回值

返回值 # 0 失败,详情参照错误码。

▶ 依赖

• 头文件: mi_sensor_datatype.h、mi_sensor.h

• 库文件: libmi_sensor.a

▶ 相关主题

MI SNR GetFps

1.2.7 MI_SNR_GetBT656SrcType

▶ 功能

获取 BT656 源输入格式。

▶ 语法

MI_S32 MI_SNR_GetBT656SrcType(MI_SNR_PAD_ID_e ePADId, MI_U32 u32PlaneID, MI_SNR_Anadec_SrcType_e *psttype);

▶ 形参

参数名称	描述	输入/输出
------	----	-------

ePADId	SENSOR设备号。	输入
	取值范围: [0, MI_SNR_MAX_PAD_NUM)。	
u32PlaneI D	SENSOR通道号。 取值范围: [0, MI_SNR_MAX_PLANE_NUM)。	输入
psttype	源输入格式	输出

▶ 返回值

返回值 《MI_OK 成功。 返回值 《非 0 失败,详情参照错误码。

▶ 依赖

• 头文件: mi_sensor_datatype.h、mi_sensor.h

• 库文件: libmi_sensor.a

※ 注意

在BT656 sensor上才可以用。

▶ 举例

无。

▶ 相关主题

MI SNR Anadec SrcType e

1.2.8 MI_SNR_QueryResCount

▶ 功能

获取 SENSOR 支持分辨率的数量。

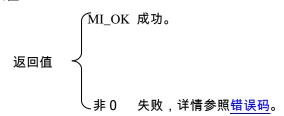
▶ 语法

MI_S32 MI_SNR_QueryResCount(<u>MI_SNR_PAD_ID_e</u> ePADId, MI_U32 *pu32ResCount);

▶ 形参

参数名称	描述	输入/输出
ePADId	SENSOR设备号。	输入
	取值范围: [0, MI_SNR_MAX_PAD_NUM)。	
*pu32ResCount	SENSOR设备支持的resolution 数量	输出

▶ 返回值



▶ 依赖

• 头文件: mi_sensor_datatype.h、mi_sensor.h

• 库文件: libmi_sensor.a

※ 注意

无。

▶ 举例

无。

▶ 相关主题

MI_SNR_GetRes

1.2.9 MI_SNR_GetRes

▶ 功能

获取 resolution 映射表中索引对应的分辨率。

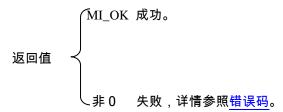
▶ 语法

MI_S32 MI_SNR_GetRes(<u>MI_SNR_PAD_ID_e</u> ePADId, MI_U8 u8ResIdx, <u>MI_SNR_Res_t</u> *pstRes);

▶ 形参

参数名称	描述	输入/输出
ePADId	SENSOR设备号。	输入
	取值范围: [0, MI_SNR_MAX_PAD_NUM)。	
u8ResIdx	分辨率映射表中的索引	输入
*pstRes	序号所对应的分辨率	输出

▶ 返回值

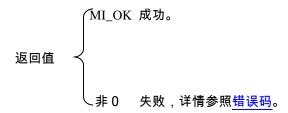


```
依赖
                     头文件: mi_sensor_datatype.h、mi_sensor.h
                     库文件: libmi sensor.a
   注意
※
                  无
    举例
         MI U32 u32ResCount =0;
         MI_U8 u8ResIndex =0;
         MI_U8 u8ChocieRes =0;
         MI_SNR_QueryResCount(E_MI_SNR_PAD_ID_0, &u32ResCount);
         for(u8ResIndex=0; u8ResIndex < u32ResCount; u8ResIndex++)</pre>
         {
             MI_SNR_GetRes(E_MI_SNR_PAD_ID_0, u8ResIndex, &stRes);
             printf("index %d, Crop(%d,%d,%d), outputsize(%d,%d), maxfps %d, minfps %d,
                       ResDesc %s\n",u8ResIndex, stRes.stCropRect.u16X, stRes.stCropRect.u16Y,
                  stRes.stCropRect.u16Width,stRes.stCropRect.u16Height,stRes.stOutputSize.u16Width,
                  stRes.stOutputSize.u16Height, stRes.u32MaxFps,stRes.u32MinFps, stRes.strResDesc);
         }
         printf("select res\n");
         scanf("%c", &select);
         MI_SNR_SetPlaneMode(E_MI_SNR_PAD_ID_0, FALSE);
         MI SNR SetRes(E_MI_SNR_PAD_ID_0,u8ResIdx);
         MI SNR Enable(E_MI_SNR_PAD_ID_0);
   相关主题
                  MI SNR QueryResCount
                  MI SNR Res t
1.2.10 MI_SNR_GetCurRes
    功能
                 获取 sensor 当前分辨率和在分辨率映射表中的位置。
    语法
                  MI_S32 MI_SNR_GetCurRes(MI_SNR_PAD_ID_e ePADId, MI_U8 *pu8CurResIdx,
                  MI_SNR_Res_t *pstCurRes);
```

▶ 形参

参数名称	描述	输入/输出
ePADId	SENSOR设备号。	输入
	取值范围: [0, MI SNR MAX PAD NUM)。	
*pu8CurResIdx	当前分辨率的索引	输出
*pstCurRes	当前分辨率信息	输出

▶ 返回值



▶ 依赖

• 头文件: mi_sensor_datatype.h、mi_sensor.h

• 库文件: libmi_sensor.a

※ 注意

无。

▶ 举例

参考 MI SNR GetRes 举例

▶ 相关主题

MI SNR Res t

1.2.11 MI_SNR_SetRes

▶ 功能

设置 sensor 设备输出分辨率

▶ 语法

MI_S32 MI_SNR_SetRes(MI_SNR_PAD_ID_e ePADId, MI_U8 u8ResIdx);

▶ 形参

参数名称	描述	输入/输出
stVifDevMap	Dev 和SensorPad Maping 关系	输入
u8Length	Dev Num	输入

▶ 返回值



▶ 依赖

• 头文件: mi_sensor_datatype.h、mi_sensor.h

• 库文件: libmi_sensor.a

※ 注意

在默认情况下 vif Dev 和 SensorPad 对应关系是 vif Dev0 -> SensorPad0, vif Dev1 -> SensorPad1.

▶ 举例

参考 MI_SNR_GetRes 举例

▶ 相关主题

MI SNR GetRes MI SNR Res t

1.2.12 MI_SNR_SetOrien

▶ 功能

设置 sensor 图象翻转属性

▶ 语法

MI_S32 MI_SNR_SetOrien(<u>MI_SNR_PAD_ID_e</u> ePADId, MI_BOOL bMirror, MI_BOOL bFlip);

▶ 形参

参数名称	描述	输入/输出
ePADId	SENSOR设备号。	输入
	取值范围: [0, MI SNR MAX PAD NUM)。	
bMirror	使能竖直镜像翻转	输入
bFlip	使能水平镜像翻转	输入

▶ 返回值

MI SENSOR API

Version 2.03

▶ 依赖

• 头文件: mi_sensor.h

• 库文件: libmi_sensor.a

※ 注意

无。

▶ 举例

无。

> 相关主题

MI SNR GetOrien

1.2.13 MI_SNR_GetOrien

▶ 功能

获取 sensor 图象翻转属性

▶ 语法

 $\label{eq:mi_snr_pad_id} MI_S32\ MI_SNR_GetOrien(\mbox{\underline{MI} SNR PAD ID e} \mbox{ ePADId, MI_BOOL *pbMirror, } MI_BOOL *pbFlip);$

▶ 形参

参数名称	描述	输入/输出
ePADId	SENSOR设备号。	输入
	取值范围: [0, MI SNR MAX PAD NUM)。	
*pbMirror	使能竖直镜像翻转	输出
*pbFlip	使能水平镜像翻转	输出

▶ 返回值

返回值 《MI_OK 成功。 非 0 失败,详情参照错误码。 ▶ 依赖

• 头文件: mi_sensor.h

• 库文件: libmi_sensor.a

※ 注意

无。

▶ 举例

无。

▶ 相关主题

MI SNR SetOrien

1.2.14 MI_SNR_SetPlaneMode

▶ 功能

设置 sensor Plane 模式。

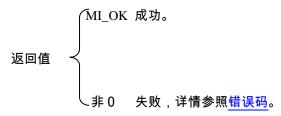
▶ 语法

MI_S32 MI_SNR_SetPlaneMode(MI_SNR_PAD_ID_e_ ePADId, MI_BOOL bEnable);

▶ 形参

参数名称	描述	输入/输出
ePADId	SENSOR设备号。	输入
	取值范围: [0, MI SNR MAX PAD NUM)。	
bEnable	开HDR 需要置为TRUE, 否则为FALSE	输入

▶ 返回值



▶ 依赖

• 头文件: mi_sensor.h

• 库文件: libmi_sensor.a

※ 注意

无。

▶ 举例

无。

▶ 相关主题

MI SNR GetPlaneMode

1.2.15 MI_SNR_GetPlaneMode

▶ 功能

获取上层设置的 Sensor Plane 模式。

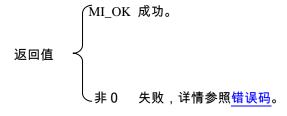
▶ 语法

MI_S32 MI_SNR_GetPlaneMode(MI_SNR_PAD_ID_e ePADId, MI_BOOL *pbEnable);

▶ 形参

参数名称	描述	输入/输出
ePADId	SENSOR设备号。	输入
	取值范围: [0, MI SNR MAX PAD NUM)。	
*pbEnable	开HDR 需要置为TRUE, 否则为FALSE	输出

▶ 返回值



▶ 依赖

• 头文件: mi_sensor.h

• 库文件: libmi_sensor.a

※ 注意

无。

▶ 举例

无。

> 相关主题

MI SNR SetPlaneMode

2. SENSOR 数据类型

视频输入相关数据类型定义如下:

MI_SNR_MAX_PADNUM	定义支持Sensor最大数量。
MI SNR MAX PLANENUM	定义每一个Sensor Pad 支持通道数量
MI SNR PAD ID e	定义Sensor Pad枚举类型
MI_SNR_HDRSrc_e	定义Sensor HDR 通道序号
MI SNR HDRHWMode e	定义 HDR 硬件设置模式
MI_SNR_Anadec_SrcType_e	定义Bt656 sensor 输入源格式
MI_SNR_Res_t	定义 Sensor分辨率属性
MI_SNR_Res_List_t	定义 Sensor分辨率映射表
MI_SNR_AttrParallel_t	定义 Parallel Sensor属性
MI_SNR_MipiAttr_t	定义Mipi Sensor属性
MI_SNR_AttrBt656_t	定义Bt656 Sensor属性
MI SNR IntfAttr u	定义Sensor接口联合体
MI SNR PADInfo t	定义Sensor Pad信息
MI_SNR_PlaneInfo_t	定义Sensor 通道信息

2.1. MI_SNR_MAX_PADNUM

▶ 说明

定义支持 Sensor 最大数量。

▶ 定义

#define MI_SNR_MAX_PADNUM 4

※ 注意事项

无

▶ 相关数据类型及接口

无。

2.2. MI_SNR_MAX_PLANENUM

▶ 说明

定义每一个 Sensor Pad 支持通道数量。

▶ 定义

#define MI_SNR_MAX_PLANENUM 2

※ 注意事项

无。

▶ 相关数据类型及接口

无。

2.3. MI_SNR_PAD_ID_e

▶ 说明

Sensor Pad Id 枚举。

```
typedef enum
             {
                 E_MI_SNR_PAD_ID_0 = 0,
                 E_MI_SNR_PAD_ID_1 = 1,
                 E_MI_SNR_PAD_ID_2 = 2,
                 E_MI_SNR_PAD_ID_3 = 3,
                 E_MI_SNR_PAD_ID_MAX = 3,
                 E_MI_SNR_PAD_ID_NA = 0xFF,
              } MI_SNR_PAD_ID_e;
※
  注意事项
                无。
   相关数据类型及接口
                无。
2.4. MI_SNR_HDRSrc_e
   说明
                Sensor HDR 通道号枚举。
   定义
                typedef enum
              {
                 E_MI_SNR_HDR_SOURCE_VC0,
                 E_MI_SNR_HDR_SOURCE_VC1,
                 E_MI_SNR_HDR_SOURCE_VC2,
                 E_MI_SNR_HDR_SOURCE_VC3,
                 E_MI_SNR_HDR_SOURCE_MAX
```

} MI_SNR_HDRSrc_e;

※ 注意事项

定义

无。

▶ 相关数据类型及接口

MI_SNR_PlaneInfo_t

2.5. MI_SNR_HDRHWMode_e

▶ 说明

定义 Sensor 硬件设置 HDR 模式。

▶ 定义

```
typedef enum

{
    E_MI_SNR_HDR_HW_MODE_NONE = 0,
    E_MI_SNR_HDR_HW_MODE_SONY_DOL = 1,
    E_MI_SNR_HDR_HW_MODE_DCG = 2,
    E_MI_SNR_HDR_HW_MODE_EMBEDDED_RAW8 = 3,
    E_MI_SNR_HDR_HW_MODE_EMBEDDED_RAW10 = 4,
    E_MI_SNR_HDR_HW_MODE_EMBEDDED_RAW12 = 5,
    E_MI_SNR_HDR_HW_MODE_EMBEDDED_RAW16 = 6, //Only for OV2718?

} MI_SNR_HDRHWMode_e;
```

▶ 成员

成员名称	描述
E_MI_SNR_HDR_HW_MODE_NONE	没有开HDR模式
E_MI_SNR_HDR_HW_MODE_SONY_DOL	Digital Overlap High Dynamic Range
E_MI_SNR_HDR_HW_MODE_DCG	
E_MI_SNR_HDR_HW_MODE_EMBEDDED _RAW8	8bit 压缩模式
E_MI_SNR_HDR_HW_MODE_EMBEDDED _RAW10	10bit 压缩模式
E_MI_SNR_HDR_HW_MODE_EMBEDDED _RAW12	12bit 压缩模式
E_MI_SNR_HDR_HW_MODE_EMBEDDED _RAW16	16bit 压缩模式

※ 注意事项

无。

▶ 相关数据类型及接口

MI SNR MipiAttr t

2.6. MI_SNR_Anadec_SrcType_e

▶ 说明

定义BT656 sensor 输入源格式。

▶ 定义

```
typedef enum

{

E_MI_SNR_ANADEC_SRC_NO_READY = 0,

E_MI_SNR_ANADEC_SRC_PAL,

E_MI_SNR_ANADEC_SRC_NTSC,

E_MI_SNR_ANADEC_SRC_HD,

E_MI_SNR_ANADEC_SRC_FHD,

E_MI_SNR_ANADEC_SRC_DISCNT,

E_MI_SNR_ANADEC_SRC_DISCNT,
```

※ 注意事项

无。

▶ 相关数据类型及接口

MI SNR GetBT656SrcType

} MI_SNR_Anadec_SrcType_e;

2.7. MI_SNR_Res_t

▶ 说明

定义 Sensor 分辨率属性

▶ 定义

```
typedef struct MI_SNR_Res_s

{

MI_SYS_WindowRect_t stCropRect;

MI_SYS_WindowSize_t stOutputSize; /**< Sensor actual output size */

MI_U32 u32MaxFps; /**< Max fps in this resolution */

MI_U32 u32MinFps; /**< Min fps in this resolution*/

MI_S8 strResDesc[32]; // Need to put "HDR" here if the resolution is for HDR

} __attribute__((packed, aligned(4))) MI_SNR_Res_t;
```

▶ 成员

成员名称	描述
stCropRect	在output size上裁剪的区域
stOutputSize	Sensor 输出大小范围
u32MaxFps	当前分辨率下最大帧率
u32MinFps	当前分辨率下最小帧率
strResDesc	Resolution string

※ 注意事项

sensor 实际输出给后端的 size 是 stCropRect 的 width/height。

▶ 相关数据类型及接口

MI_SNR_GetRes MI_SNR_GetCurRes MI_SNR_SetRes

2.8. MI_SNR_Res_List_t

▶ 说明

定义 Sensor 分辨率映射表。

▶ 定义

```
typedef struct MI_SNR_Res_List_s
{
    MI_U32 u32NumRes;/**< number of sensor resolution in list */
    MI_U32 u32CurResIndex;/**< current sensor resolution*/
    MI_SNR_Res_t stRes[12]; /**< resolution list */
} __attribute__((packed, aligned(4))) MI_SNR_Res_List_t;</pre>
```

▶ 成员

成员名称	描述
u32NumRes	支持的resolution 数量
u32CurResIndex	当前resolution 索引
stRes	Resolution 映射表

※ 注意事项

此结构体为了之后扩展用,目前没有用到

▶ 相关数据类型及接口

MI_SNR_Res_t

2.9. MI_SNR_AttrParallel_t

▶ 说明

定义parallel sensor 属性。

▶ 定义

```
typedef struct MI_SNR_AttrParallel_s
{
     MI_VIF_SyncAttr_t stSyncAttr;
} MI_SNR_AttrParallel_t;
```

▶ 成员

成员名称	描述

Version 2.03

|--|

注意事项 Ж

无。

相关数据类型及接口

MI SNR IntfAttr u

MI SNR MipiAttr t 2.10.

{

说明

定义Mipi sensor 属性。

}MI_SNR_MipiAttr_t;

定义

```
typedef struct MI_SNR_MipiAttr_s
   MI_U32 u32LaneNum; // 几条信号同时传
   MI_U32 u32DataFormat;
                               //0: YUV 422 format. 1: RGB pattern.
   MI_VIF_DataYuvSeq_e
                           eDataYUVOrder;
   MI_U32 u32HsyncMode; //每条 line 的前面 hsync 还是后一般后
   MI_U32 u32Sampling_delay;
   /** < MIPI start sampling delay */ /*bit 0~7: clk_skip_ns. bit 8~15: data_skip_ns*/
   MI_SNR_HDRHWMode_e eHdrHWmode;
   MI_U32 u32Hdr_Virchn_num;
   MI_U32 u32Long_packet_type[2];
// [0]Null [1]blinking [2]embedded [14]yuv422_8b [26]RAW8 [27]RAW10 [28]RAW12 [32]UD1
   [33]UD2 [34]UD3 [35]UD4 [36]UD5 [37]UD6 [38]UD7 [39]UD8
```

成员

成员名称	描述	
u32LaneNum	支持同时传输数据的信号线数量	
u32DataFormat	0: YUV 422 format. 1: RGB pattern	
eDataYUVOrder	YUV 排列顺序	
u32HsyncMode	同步前一条或者后一条line的hsync 信号	

成员名称	描述	
u32Sampling_delay	延时跳过数据头部分	
eHdrHWmode	Sensor支持的HDR mode	
u32Hdr_Virchn_num	Sensor支持的HDR 虚拟通道数量	
u32Long_packet_type[2]	Sensor支持的数据打包格式	

※ 注意事项

无。

相关数据类型及接口

MI_SNR_IntfAttr_u

2.11. MI_SNR_AttrBt656_t

▶ 说明

定义BT656 sensor 属性。

▶ 定义

```
typedef struct MI_SNR_AttrBt656_s

{

MI_U32 u32Multiplex_num;

MI_VIF_SyncAttr_t stSyncAttr;

MI_VIF_ClkEdge_e eClkEdge;

MI_VIF_BitOrder_e eBitSwap;

} MI_SNR_AttrBt656_t;
```

▶ 成员

成员名称	描述
u32Multiplex_num	复合模式的路数
stSyncAttr	同步信号属性
eClkEdge	采样时钟模式
eBitSwap	数据排列方向

※ 注意事项

无。

▶ 相关数据类型及接口

MI SNR IntfAttr u

2.12. MI_SNR_IntfAttr_u

▶ 说明

定义 sensor 接口类型联合体。

▶ 定义

typedef union {
 MI_SNR_AttrParallel_t stParallelAttr;
 MI_SNR_MipiAttr_t stMipiAttr;
 MI_SNR_AttrBt656_t stBt656Attr;
} MI_SNR_IntfAttr_u;

▶ 成员

成员名称	描述	
stParallelAttr	Parallel sensor 属性	
stMipiAttr	Mipi sensor属性	
stBt656Attr	Bt656 sensor属性	

※ 注意事项

无。

▶ 相关数据类型及接口

MI_SNR_PADInfo_t

2.13. MI_SNR_PADInfo_t

▶ 说明

MI SENSOR API Version 2.03

定义 sensor Pad 信息属性。

▶ 定义

```
typedef struct MI_SNR_PADInfo_s

{

MI_U32     u32PlaneCount;

//It is different expo number for HDR. It is mux number for BT656. //??

MI_VIF_IntfMode_e    eIntfMode;

MI_VIF_HDRType_e    eHDRMode;

MI_SNR_IntfAttr_u    unIntfAttr;

} MI_SNR_PADInfo_t;
```

▶ 成员

成员名称	描述	
u32PlaneCount	BT656 sensor 代表最大复合路数 Mipi sensor 代表长短曝数量	
eIntfMode	Sensor 接口枚举	
eHDRMode	Hdr 模式	
unIntfAttr	Sensor 接口属性联合体	

※ 注意事项

无。

▶ 相关数据类型及接口

MI SNR GetPadInfo

2.14. MI_SNR_PlaneInfo_t

▶ 说明

定义 sensor 通道 信息属性。

▶ 定义

▶ 成员

成员名称	描述
u32PlaneID	开HDR代表当前Plane是长曝还是短曝 BT656 代表代表当前Plane在复合路数中的通道 id
s8SensorName	Sensor name字符串
stCapRect	在sensor 数据上裁剪的位置
eBayerId	RGB 排列顺序
ePixPrecision	RGB 压缩模式
eHdrSrc	HDR 通道号

※ 注意事项

无。

▶ 相关数据类型及接口

 $MI_SNR_GetPadInfo$

3. SENSOR 错误码

视频输入 API 错误码如表 所示。

Sensor API 错误码

错误代码	宏定义	描述
0xA0032001	MI_ERR_SNR_INVALID_DEVID	设备号无效
0xA0032002	MI_ERR_SNR_INVALID_CHNID	通道号无效
0xA0032003	MI_ERR_SNR_INVALID_PARA	参数设置无效
0xA0032006	MI_ERR_SNR_INVALID_NULL_PTR	输入参数空指针错误
0xA0032007	MI_ERR_SNR_FAILED_NOTCONFIG	设备或通道属性未配置
0xA0108008	MI_ERR_SNR_NOT_SUPPORT	操作不支持
0xA0108009	MI_ERR_SNR_NOT_PERM	操作不允许
0xA0108010	MI_ERR_SNR_SYS_NOTREADY	系统未初始化
0xA0108012	MI_ERR_SNR_BUSY	系统忙
0xA0032080	MI_ERR_SNR_FAIL	端口无效