AX MIPI 开发参考

文档版本：V2.1

发布日期：2024/03/25

目 录

权利声明

注意

您购买的产品、服务或特性等应受商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非商业合同另有约定，本公司对本文档内容不做任何明示或默示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

前 言

适用产品

AX650A/AX650N/AX630C/AX620Q

适读人群

- 技术支持工程师
- 软件开发工程师

符号与格式定义

符号/格式	说明
xxx	表示您可以执行的命令行。
说明/备注：	表示您在使用产品的过程中，我们向您说明的事项。
注意：	表示您在使用产品的过程中，需要您特别注意的事项。

修订历史

文档版本	发布时间	修订说明
V1.0	2022/11/17	文档初版
V1.1	2023/05/12	完善功能描述章节内容
V1.2	2023/07/18	完善 AX_MIPI_RX_SetLaneCombo 描述内容
V1.3	2024/01/05	添加 MIPI Proc 章节
V2.0	2024/02/22	下列文档合并： AX650A/AX650N《41 - AX MIPI 开发参考.docx》 AX630C/AX620Q《11 - AX MIPI 开发参考.docx》
V2.1	2024/03/25	错误码内容移动到《55 - AX 软件错误码文档》
V2.1	2024/09/01	更新 AX630C/AX620Q combo 组合 case

1 概述

1.1 概述

移动产业处理器接口（Mobile Industry Processor Interface，MIPI）为移动设备组件接口规范标准。MIPI PHY 为 DSI 和 CSI 提供物理层定义，描述源同步、高速、低功耗的物理层接口协议。根据应用需求，MIPI PHY 分为 RX 与 TX 两个部分，用于接收或发送符合 MIPI PHY 规范的数据。

AX650A/AX650N

- MIPI PHY 兼容 CPHY、DPHY 和 sub-LVDS。

AX620C/AX620Q

- MIPI PHY 兼容 DPHY 和 sub-LVDS。

MIPI Rx 通过低电压差分信号接收原始数据，将接收到的串行差分信号(serial differential signal)转化后传递给下一级模块 SIF（Sensor Interface），MIPI Rx 包含 PHY 和 Controller 两部分，功能框图及在系统中的位置如图 1：

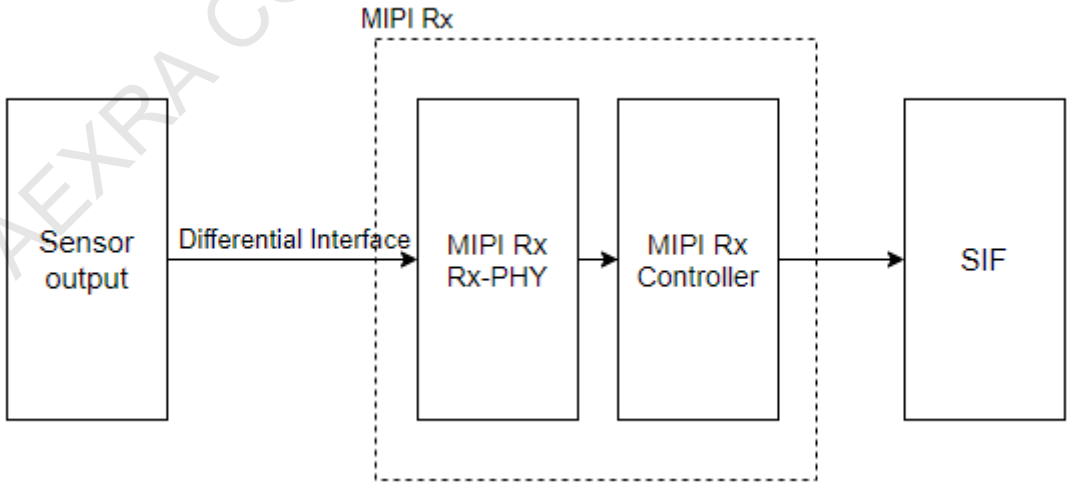

图1 MIPI Rx 功能框图及在系统中的位置

MIPI Rx 支持 MIPI、sub-LVDS 等视频传输接口输入，此类接口可以提供更高的传输带宽，增强传输的稳定性，另外 DVP、BT601、BT656、BT1120 可复用 PHY GPIO 完成数据传输。

1.2 重要概念

表1-1 重要概念

名称	介绍
MIPI	MIPI 的全称是 Mobile Industry Processor Interface（移动行业处理器接口），本文描述的 MIPI 接口指物理层使用 D-PHY/C-PHY 传输规范，协议层使用 CSI-2 的通信接口。
LVDS	LVDS 的全称是 Low Voltage differential Signaling（低压差分信号），通过同步码区分消隐区和有效数据。
SLVS-EC	SLVS-EC 的全称是 Scalable Low Voltage Signaling Embedded Clock，是与 MIPI 并列的接口，用于高帧率和高分辨率图像采集。
Lane	用于连接发送端和接收端的一对高速差分线，既可以是时钟 Lane，也可以是数据 Lane。
同步码	MIPI 接口使用 CSI-2 里面的短包进行同步，LVDS 使用同步码区分有效数据和消隐区。LVDS 有两种同步方式： -使用 SOF/EOF 表示帧起始和结束，使用 SOL/EOL 表示行起始和结束 -使用 SAV(Invalid)和 EAV(Invalid)标识消隐区的无效数据，使用 SAV(Valid)和 EAV(Valid)标识消隐区的有效数据

1.3 功能介绍

在典型配置下，MIPI DPHY 包含 1 个时钟通道和 1~4 个数据通道。可配置数据通道 LANE 个数。时钟和数据通道可在 1.2V LVCMOS 信号或 SLVS-200 差分信号之间转换。

MIPI DPHY 支持以下两种数据传输模式：

- 高速（High-speed，HS）模式

- 低功耗（Low-power，LP）模式

在 HS 模式下，视频数据通过差分进行传递。如应用不同，可持续使用 HS 模式，亦可将高速差分通道转换为单端信号。当 DPHY 发送单端信号数据时，进入 LP 模式。

MIPI Rx 是一个支持多种视频差分输入的采集单元，主要的功能是接口时序的转换，通过 PHY 接收 MIPI、sub-LVDS 接口的数据，通过不同的功能模式配置，MIPI Rx 可以支持多种速度和分辨率的数据传输需求，支持多种外部输入设备。

AX650A/AX650N :

- DPHY 模式 LANE 最大速率 3.5Gbps，CPHY 模式 LANE 最大速率 3.5Gbps，sub-LVDS 模式 LANE 最大速率 1.0Gbps。
- MIPI Rx 最大对接 8 个 sensor，用户需要确定 MIPI Rx 的 LANE 分布模式。具体的 LANE 分布模式请参见下表：

AX650A/AX650N MIPI Rx LANE 分布模式

接口	模式	PHY0	PHY1	PHY2	PHY3	PHY4	PHY5	PHY6	PHY7
MIPI DPHY/ CPHY Sensor	0	sensor0(8L) eMipiDev = 0 nDevId = 0				sensor4(8L) eMipiDev = 4 nDevId = 4			
	1	sensor0(8L) eMipiDev = 0 nDevId = 0				sensor4(4L) eMipiDev = 4 nDevId = 4		sensor6(4L) eMipiDev = 6 nDevId = 6	
	2	sensor0(8L) eMipiDev = 0 nDevId = 0				sensor4(4L) eMipiDev = 4 nDevId = 4		sensor6(2L) eMipiDev = 6 nDevId = 7	sensor7(2L) eMipiDev = 7 nDevId = 6
	3	sensor0(8L) eMipiDev = 0 nDevId = 0				sensor4(2L) eMipiDev = 4 nDevId = 5	sensor5(2L) eMipiDev = 5 nDevId = 4	sensor6(2L) eMipiDev = 6 nDevId = 7	sensor7(2L) eMipiDev = 7 nDevId = 6
	4	sensor0(4L) eMipiDev = 0 nDevId = 0		sensor2(4L) eMipiDev = 2 nDevId = 2		sensor4(4L) eMipiDev = 4 nDevId = 4		sensor6(4L) eMipiDev = 6 nDevId = 6	
	5	sensor0(4L) eMipiDev = 0 nDevId = 0		sensor2(4L) eMipiDev = 2 nDevId = 2		sensor4(4L) eMipiDev = 4 nDevId = 4		sensor6(2L) eMipiDev = 6 nDevId = 7	sensor7(2L) eMipiDev = 7 nDevId = 6
	6	sensor0(4L) eMipiDev = 0 nDevId = 0		sensor2(4L) eMipiDev = 2 nDevId = 2		sensor4(2L) eMipiDev = 4 nDevId = 5	sensor5(2L) eMipiDev = 5 nDevId = 4	sensor7(2L) eMipiDev = 6 nDevId = 7	sensor7(2L) eMipiDev = 7 nDevId = 6
	7	sensor0(4L) eMipiDev = 0 nDevId = 0		sensor2(2L) eMipiDev = 2 nDevId = 2	sensor3(2L) eMipiDev = 3 nDevId = 3	sensor4(2L) eMipiDev = 4 nDevId = 5	sensor5(2L) eMipiDev = 5 nDevId = 4	sensor7(2L) eMipiDev = 6 nDevId = 7	sensor7(2L) eMipiDev = 7 nDevId = 6
	8	sensor0(2L) eMipiDev = 0 nDevId = 0	sensor1(2L) eMipiDev = 1 nDevId = 1	sensor2(2L) eMipiDev = 2 nDevId = 2	sensor3(2L) eMipiDev = 3 nDevId = 3	sensor4(2L) eMipiDev = 4 nDevId = 5	sensor5(2L) eMipiDev = 5 nDevId = 4	sensor7(2L) eMipiDev = 6 nDevId = 7	sensor7(2L) eMipiDev = 7 nDevId = 6
sub-LVDS/	9	sensor0(16ch/12ch/10ch)							

HiSPi(SLVS) Sensor		eMipiDev = 0 nDevId = 0			
	0	sensor0(8ch/4ch/2ch) eMipiDev = 0 nDevId = 0		sensor4(8ch/4ch/2ch) eMipiDev = 4 nDevId = 2	
	1	sensor0(8ch/4ch/2ch) eMipiDev = 0 nDevId = 0		sensor4(4ch/2ch) eMipiDev = 4 nDevId = 2	sensor6(4ch/2ch) eMipiDev = 6 nDevId = 3
	4	sensor0(4ch/2ch) eMipiDev = 0 nDevId = 0	sensor2(4ch/2ch) eMipiDev = 2 nDevId = 1	sensor4(4ch/2ch) eMipiDev = 4 nDevId = 2	sensor6(4ch/2ch) eMipiDev = 6 nDevId = 3
DVP Sensor	0	sensor0 eMipiDev = 0 nDevId = 0			
BT Sensor	0	sensor0 eMipiDev = 0 nDevId = 4		sensor4 eMipiDev = 4 nDevId = 5	

☞ 说明：

PHY：AX650A 支持 PHY0~7, AX650N 支持 PHY0~3

AX630C/AX620Q:

- DPHY 模式 LANE 最大速率 2.5G bps，sub-LVDS 模式 LANE 最大速率 1.5G bps。
- MIPI Rx 最大对接 2 个 sensor，用户需要确定 MIPI Rx 的 LANE 分布模式。具体的 LANE 分布模式请参见下表：

AX630C/AX620Q MIPI Rx LANE 分布模式

接口	模式	PHY0		VI (BT/DVP)
MIPI DPHY Sensor	0	sensor0(4L) nDevId = 0		sensor2(BT/DVP) nDevId = 2
	1	sensor0(1/2L) nDevId = 0	sensor1(1/2L) nDevId = 1	sensor2(BT/DVP) nDevId = 2
sub-LVDS/ HiSPi(SLVS) Sensor	0	sensor0(4ch) nDevId = 0		sensor2(BT/DVP) nDevId = 2
	1	sensor0(2ch) nDevId = 0	sensor1(2ch) nDevId = 1	sensor2(BT/DVP) nDevId = 2
DVP Sensor	0	Sensor0 nDevId = 0		sensor2(DVP) nDevId = 2
	2	Sensor0(part0) nDevId = 0	Sensor1(1/2L) nDevId = 1	Sensor0(part1)
	3	Sensor0 nDevId = 0		

BT Sensor (BT656/BT601/BT1120)	0	sensor0 nDevId = 0		sensor2(BT601/BT656) nDevId = 2
	1	sensor0 nDevId = 0	sensor1 nDevId = 1	sensor2(BT601/BT656) nDevId = 2
	2	Sensor2(part1)	sensor1 nDevId = 1	sensor2(BT601/BT656 part0) nDevId = 2
	3	sensor2(BT1120) nDevId = 2		

☞ 说明：

- eMipiDev：《AX MIPI 开发参考》文档中 MIPI 相关接口使用
- nDevId：《AX VIN 开发参考》文档中 DEV 相关接口使用

➢ MIPI 库

用于设置/获取/复位 MIPI 相关参数和状态等，具体详见 API 参考部分，库的名称为 libax_mipi.so

2 API 参考

AX_MIPI_RX_Init

【描述】

初始化 MIPI Rx。

【语法】

AX_S32 AX_MIPI_RX_Init(void)

【参数】

无

【返回值】

返回值	描述
0	成功
非 0	失败

【需求】

- 头文件：ax_mipi_rx_api.h
- 库文件：libax_mipi.so

【注意】

无

【举例】

无

【相关主题】

无

AX_MIPI_RX_DeInit

【描述】

退出 MIPI Rx

【语法】

AX_S32 AX_MIPI_RX_DeInit(void)

【参数】

无

【返回值】

返回值	描述
0	成功
非 0	失败

【需求】

- 头文件：ax_mipi_rx_api.h
- 库文件：libax_mipi.so

【注意】

无

【举例】

无

【相关主题】

无

AX_MIPI_RX_SetLaneCombo

【描述】

设置 Date Lane 组合模式

【语法】

AX_S32 AX_MIPI_RX_SetLaneCombo(AX_LANE_COMBO_MODE_E eMode)

【参数】

参数名称	描述	输入/输出
eMode	Mode ID	输入

【返回值】

返回值	描述
0	成功
非 0	失败

【需求】

- 头文件：ax_mipi_rx_api.h
- 库文件：libax_mipi.so

【注意】

- 由于不同接口类型存在管脚复用关系，所以 MIPI、Sub-LVDS、DVP、BT 都需要调用该接口，具体配置的模式需要根据实际占用 Lane 的情况，可参考《AX MIPI 开发参考》文档内 1.3.1 功能描述。
- 设置的模式能够覆盖实际使用的接口 Lane 数即可，比如使用 4 Lane MIPI Sensor，设置模式 4/5/6/7 都可以工作。
- AX650N 仅支持 PHY0/PHY1/PHY2/PHY3。

【举例】

- 接口组合：1 个 MIPI 8lane 接口 Sensor 和 2 个 MIPI 4lane 接口 Sensor。eMode 设置为 AX_LANE_COMBO_MODE_1。
- 接口组合：1 个 BT1120 接口 Sensor ，1 个 MIPI 4lane 接口 Sensor 和 2 个 MIPI 2lane 接口 Sensor。eMode 设置为 AX_LANE_COMBO_MODE_2。
- 接口组合：1 个 DVP 接口 Sensor 和 4 个 MIPI 2lane 接口 Sensor。eMode 设置为 AX_LANE_COMBO_MODE_3。
- 接口组合：4 个 MIPI 4lane Sensor，eMode 设置为 AX_LANE_COMBO_MODE_4。
- 接口组合：4 个 LVDS 4lane Sensor，eMode 设置为 AX_LANE_COMBO_MODE_4。

【相关主题】

无

AX_MIPI_RX_Reset

【描述】

MIPI Rx 复位。

【语法】

AX_S32 AX_MIPI_RX_Reset(AX_U32 eMipiDev)

【参数】

参数名称	描述	输入/输出
eMipiDev	MIPI Rx 设备号	输入

【返回值】

返回值	描述
0	成功
非 0	失败

【需求】

- 头文件：ax_mipi_rx_api.h
- 库文件：libax_mipi.so

【注意】

- 该接口必须在 AX_MIPI_RX_SetAttr 接口之后调用

【举例】

无

【相关主题】

无

AX_MIPI_RX_SetAttr

【描述】

设置 MIPI Rx 相关参数。

【语法】

AX_S32 AX_MIPI_RX_SetAttr(AX_U32 eMipiDev, AX_MIPI_RX_DEV_T *pDevAttr)

【参数】

参数名称	描述	输入/输出
nDevId	MIPI Rx 设备号	输入
pDevAttr	DevAttr 相关参数	输入

【返回值】

返回值	描述
0	成功
非 0	失败

【需求】

- 头文件：ax_mipi_rx_api.h
- 库文件：libax_mipi.so

【注意】

无

【举例】

无

【相关主题】

无

AX_MIPI_RX_GetAttr

【描述】

获取 MIPI Rx 相关参数。

【语法】

AX_S32 AX_MIPI_RX_GetAttr(AX_U32 nDevId, AX_MIPI_RX_ATTR_S *pMipiAttr)

【参数】

参数名称	描述	输入/输出
nDevId	MIPI Rx 设备号	输入
pMipiAttr	MIPI Rx 相关参数	输出

【返回值】

返回值	描述
0	成功
非 0	失败

【需求】

- 头文件：ax_mipi_rx_api.h
- 库文件：libax_mipi.so

【注意】

无

【举例】

无

【相关主题】

无

AX_MIPI_RX_Start

【描述】

MIPI Rx 启动，开始接收数据。

【语法】

AX_S32 AX_MIPI_RX_Start(AX_U32 nDevId)

【参数】

参数名称	描述	输入/输出
nDevId	MIPI Rx 设备号	输入

【返回值】

返回值	描述
0	成功
非 0	失败

【需求】

- 头文件：ax_mipi_rx_api.h
- 库文件：libax_mipi.so

【注意】

- 该函数会将 PIN 的复用改成与 eInputMode 对应功能，请注意 PHY PIN 上是否有 Sensor 以外的硬件设备。

【举例】

无

【相关主题】

无

AX_MIPI_RX_Stop

【描述】

MIPI Rx 关闭，停止接收数据。

【语法】

AX_S32 AX_MIPI_RX_Stop(AX_U32 nDevId)

【参数】

参数名称	描述	输入/输出
nDevId	MIPI Rx 设备号	输入

【返回值】

返回值	描述
0	成功
非 0	失败

【需求】

- 头文件：ax_mipi_rx_api.h
- 库文件：libax_mipi.so

【注意】

无

【举例】

无

【相关主题】

无

3 数据结构

AX_MIPI_CLK_LANE_MAX

【说明】

定义 mipi clk lane 的最大个数

【定义】

#define AX_MIPI_CLK_LANE_MAX (2)

【注意】

无

AX_MIPI_LANE_ID_MAX

【说明】

定义 mipi data lane 的最大个数

【定义】

AX650A/ AX650N

#define AX_MIPI_LANE_NUM_MAX (8)

AX630C/ AX620Q

#define AX_MIPI_LANE_NUM_MAX (4)

【芯片差异】

芯片差异	描述
AX650A/AX650N	支持 MIPI 设备最大 Data Lane 数目为 8
AX630C/AX620Q	支持 MIPI 设备最大 Data Lane 数目为 4

【注意】

无

AX_LVDS_LANE_NUM_MAX

【说明】

定义 lvds data lane 的最大个数

【定义】

AX650A/ AX650N

#define AX_LVDS_LANE_NUM_MAX (16)

AX630C/ AX620Q

#define AX_LVDS_LANE_NUM_MAX (4)

【芯片差异】

芯片差异	描述
AX650A/ AX650N	支持 LVDS 设备最大 Data Lane 数目为 16
AX630C/ AX620Q	支持 LVDS 设备最大 Data Lane 数目为 4

【注意】

无

AX_LANE_COMBO_MODE_E

【说明】

PHY 组合模式配置

【定义】

AX650A/ AX650N

```
typedef enum {
    AX_LANE_COMBO_MODE_0      = 0x0,
    AX_LANE_COMBO_MODE_1      = 0x1,
    AX_LANE_COMBO_MODE_2      = 0x2,
    AX_LANE_COMBO_MODE_3      = 0x3,
    AX_LANE_COMBO_MODE_4      = 0x4,
    AX_LANE_COMBO_MODE_5      = 0x5,
    AX_LANE_COMBO_MODE_6      = 0x6,
    AX_LANE_COMBO_MODE_7      = 0x7,
    AX_LANE_COMBO_MODE_8      = 0x8,
    AX_LANE_COMBO_MODE_9      = 0x9,
    AX_LANE_COMBO_MODE_MAX
} AX_LANE_COMBO_MODE_E;
```

AX630C/ AX620Q

```
typedef enum {
    AX_LANE_COMBO_MODE_0      = 0x0,
    AX_LANE_COMBO_MODE_1      = 0x1,
```

AX_LANE_COMBO_MODE_MAX

} AX_LANE_COMBO_MODE_E;

【成员】

无

【芯片差异】

芯片差异	描述
AX650A/N	8 个 PHY，每个 PHY 1 个 clk lane & 2 个 data lane
AX630C	1 个 PHY，每个 PHY 2 个 clk lane & 4 个 data lane
AX620Q	1 个 PHY，每个 PHY 2 个 clk lane & 4 个 data lane

【注意】

- 参考《AX MIPI 开发参考》文档内 1.3.1 功能描述
- AX650A/N：单摄 mipi data lane 大于等于 4 时，是将多个 PHY 合成一个使用。
- AX630C/AX620Q：双摄 mipi data lane 小于等于 2 时，是将一个 PHY 分成 2 个使用。

【相关数据类型及接口】

无

AX_MIPI_PHY_TYPE_E

【说明】

MIPI RX CPHY/DPHY 配置

【定义】

AX650A/AX650N

```
typedef enum {
    AX_MIPI_PHY_TYPE_DPHY = 0,
    AX_MIPI_PHY_TYPE_CPHY = 1,
    AX_MIPI_PHY_TYPE_MAX,
} AX_MIPI_PHY_TYPE_E;
```

AX630Q/AX620C

```
typedef enum {
    AX_MIPI_PHY_TYPE_DPHY = 0,
    AX_MIPI_PHY_TYPE_MAX,
} AX_MIPI_PHY_TYPE_E;
```

【成员】

AX650A/AX650N

成员名称	描述
AX_MIPI_PHY_TYPE_DPHY	MIPI RX DPHY 模式
AX_MIPI_PHY_TYPE_CPHY	MIPI RX CPHY 模式

AX630C/AX620Q

成员名称	描述
AX_MIPI_PHY_TYPE_DPHY	MIPI RX DPHY 模式

【芯片差异】

芯片差异	描述
AX650A/N	PHY 兼容 CPHY&DPHY
AX630C	PHY 支持 DPHY
AX620Q	PHY 支持 DPHY

【注意】

- 用户需根据 sensor 的接口类型，适配这里的 CPHY 和 DPHY 的类型。

【相关数据类型及接口】

无

AX_MIPI_LANE_NUM_E

【说明】

Mipi Lane Num。

【定义】

AX650A/AX650N

```
typedef enum {
    AX_MIPI_DATA_LANE_1 = 1,
    AX_MIPI_DATA_LANE_2 = 2,
    AX_MIPI_DATA_LANE_4 = 4,
    AX_MIPI_DATA_LANE_8 = 8,
    AX_MIPI_DATA_LANE_MAX
} AX_MIPI_LANE_NUM_E;
```

AX630C/AX620Q

```
typedef enum {
    AX_MIPI_DATA_LANE_1 = 1,
    AX_MIPI_DATA_LANE_2 = 2,
    AX_MIPI_DATA_LANE_4 = 4,
    AX_MIPI_DATA_LANE_MAX
} AX_MIPI_LANE_NUM_E;
```

【成员】

AX650A/AX650N

成员名称	描述
AX_MIPI_DATA_LANE_1	MIPI 设备 Data Lane 数目为 1
AX_MIPI_DATA_LANE_2	MIPI 设备 Data Lane 数目为 2
AX_MIPI_DATA_LANE_4	MIPI 设备 Data Lane 数目为 4
AX_MIPI_DATA_LANE_8	MIPI 设备 Data Lane 数目为 8

AX630C/AX620Q

成员名称	描述
AX_MIPI_DATA_LANE_1	MIPI 设备 Data Lane 数目为 1
AX_MIPI_DATA_LANE_2	MIPI 设备 Data Lane 数目为 2
AX_MIPI_DATA_LANE_4	MIPI 设备 Data Lane 数目为 4

【芯片差异】

芯片差异	描述
AX650A/N	支持 MIPI 设备最大 Data Lane 数目为 8
AX630C	支持 MIPI 设备最大 Data Lane 数目为 4
AX620Q	支持 MIPI 设备最大 Data Lane 数目为 4

【注意】

无

【相关数据类型及接口】

无

AX_SLVDS_LANE_NUM_E

【说明】

Lvds Lane Num。

【定义】

AX650A/AX650N

```
typedef enum {
    AX_SLVDS_DATA_LANE_2 = 2,
    AX_SLVDS_DATA_LANE_4 = 4,
    AX_SLVDS_DATA_LANE_8 = 8,
    AX_SLVDS_DATA_LANE_10 = 10,
    AX_SLVDS_DATA_LANE_12 = 12,
    AX_SLVDS_DATA_LANE_16 = 16,
    AX_SLVDS_DATA_LANE_MAX
} AX_SLVDS_LANE_NUM_E;
```

AX630C/AX620Q

```
typedef enum {
    AX_SLVDS_DATA_LANE_2 = 2,
    AX_SLVDS_DATA_LANE_4 = 4,
    AX_SLVDS_DATA_LANE_MAX
} AX_SLVDS_LANE_NUM_E;
```

【成员】

AX650A/AX650N

成员名称	描述
AX_SLVDS_DATA_LANE_2	LVDS 设备 Data Lane 数目为 2
AX_SLVDS_DATA_LANE_4	LVDS 设备 Data Lane 数目为 4
AX_SLVDS_DATA_LANE_8	LVDS 设备 Data Lane 数目为 8
AX_SLVDS_DATA_LANE_10	LVDS 设备 Data Lane 数目为 10
AX_SLVDS_DATA_LANE_12	LVDS 设备 Data Lane 数目为 12
AX_SLVDS_DATA_LANE_16	LVDS 设备 Data Lane 数目为 16

AX630C/AX620Q

成员名称	描述
AX_SLVDS_DATA_LANE_2	LVDS 设备 Data Lane 数目为 2
AX_SLVDS_DATA_LANE_4	LVDS 设备 Data Lane 数目为 4

【芯片差异】

芯片差异	描述
AX650A/N	支持 LVDS 设备最大 Data Lane 数目为 16
AX630C	支持 LVDS 设备最大 Data Lane 数目为 4
AX620Q	支持 LVDS 设备最大 Data Lane 数目为 4

【注意】

无

【相关数据类型及接口】

无

AX_INPUT_MODE_E

【说明】

MIPI RX 输入接口类型配置。

【定义】

```
typedef enum {
    AX_INPUT_MODE_MIPI = 0,
    AX_INPUT_MODE_SUBLVDS = 1,
    AX_INPUT_MODE_LVDS = 2,
    AX_INPUT_MODE_HISPI = 3,
    AX_INPUT_MODE_SLVS = 4,
    AX_INPUT_MODE_BT601 = 5,
    AX_INPUT_MODE_BT656 = 6,
    AX_INPUT_MODE_BT1120 = 7,
    AX_INPUT_MODE_DVP = 8,
    AX_INPUT_MODE_MAX
} AX_INPUT_MODE_E;
```

【成员】

成员名称	描述
AX_INPUT_MODE_MIPI	SENSOR 接口类型 MIPI
AX_INPUT_MODE_SUBLVDS	SENSOR 接口类型 SUBLVDS
AX_INPUT_MODE_LVDS	SENSOR 接口类型 LVDS

成员名称	描述
AX_INPUT_MODE_HISPI	SENSOR 接口类型 HISPI
AX_INPUT_MODE_SLVS	SENSOR 接口类型 SLVS
AX_INPUT_MODE_BT601	SENSOR 接口类型 BT601
AX_INPUT_MODE_BT656	SENSOR 接口类型 BT656
AX_INPUT_MODE_BT1120	SENSOR 接口类型 BT1120
AX_INPUT_MODE_DVP	SENSOR 接口类型 DVP

【注意】

无

【相关数据类型及接口】

无

AX_MIPI_RX_ATTR_T

【说明】

MIPI Rx mipi attr 相关参数。

【定义】

```
typedef struct {
    AX_MIPI_PHY_TYPE_E          ePhyMode;
    AX_MIPI_LANE_NUM_E          eLaneNum;
    AX_U32                      nDataRate;
    AX_U8                       nDataLaneMap [AX_MIPI_LANE_ID_MAX];
    AX_S8                       nClkLane [2];
} AX_MIPI_RX_ATTR_T;
```

【成员】

成员名称	描述
ePhyMode	Mipi cphy/dphy 配置
eLaneNum	Data Lane 个数
nDataRate	Data Rate
nLaneMap	Lane 映射表，数组长度为 8
nClkLane	Clk Lane

【注意】

- nDataRate 配置 DataRate 实际数值。

确定 nDataRate 可以借鉴以下方式：

1. 找 sensor AE 确定 DataRate，例如 1000Mbps，赋值 nDataRate = 1000.
2. 通过示波器量 mipi clk 频率确定，假设 mipi clk 为 500Mhz,对应 DataRate 为 1000Mbps，此时赋值 nDataRate = 1000.

- nLaneMap & nClkLane：AX650A&&AX650N 不支持 lane map 配置，该参数可以不配置。

【相关数据类型及接口】

无

AX_LVDS_ATTR_T

【说明】

MIPI Rx lvds attr 相关参数。

【定义】

```
typedef struct {
    AX_SLVDS_LANE_NUM_E     eLaneNum;
    AX_U32                  nDataRate;
    AX_S8                   nDataLaneMap[AX_LVDS_LANE_NUM_MAX];
    AX_S8                   nClkLane[AX_MIPI_CLK_LANE_MAX];
} AX_LVDS_ATTR_T;
```

【成员】

成员名称	描述
eLaneNum	Lane 个数
nDataRate	Lvds 速率
nDataLaneMap	Lane 映射表，数组长度为 16
nClkLane [2]	4 字节对齐预留字段

【注意】

- nDataRate 配置 DataRate 实际数值。

【相关数据类型及接口】

无

AX_MIPI_RX_DEV_T

【说明】

MIPI Rx Dev 相关参数。

【定义】

```
typedef struct {
    AX_INPUT_MODE_E          eInputMode;
    union {
        AX_MIPI_RX_ATTR_T    tMipiAttr;
        AX_LVDS_ATTR_T       tLvdsAttr;
    };
} AX_MIPI_RX_DEV_T;
```

【成员】

成员名称	描述
eInputMode	Sensor 接口类型 mipi / lvds / dvp / bt 等
tMipiAttr	MIPI 数据结构
tLvdsAttr	Lvds 数据结构

【注意】

- eInputMode 为 DVP/BT 类接口时，不需要配置数据结构，但是也需要调用该接口配置输入模式。

【相关数据类型及接口】

无

【注意】

无

【相关数据类型及接口】

无

4 错误码

错误码详见《55 - AX 软件错误码文档》

5 Proc 信息说明

Proc 中记录了属性和统计信息，提供调试时使用。

文件目录：/proc/ax_proc/mipi_rx

文件列表：

文件名称	描述
attr	属性
status	状态信息

5.1 attr

参数名称		描述
DEV ATTR	LaneComboMode	Lane 的组合模式 1.3.1 功能描述。
	MipiDev	Mipi Dev Id 号
	InputMode	输入接口类型
	PhyMode	Phy 类型
	LaneNum	Data Lane 数
	DataRate	Data Lane 数据传输速率
	DataLaneMap	Data Lane map 配置
	ClkLane	Clk Lane map 配置

备注：
DataLaneMap：AX650A/AX650N 不支持 lane map 配置，内部配置不参考 DataLaneMap，可以默认配置为全 0。
ClkLane：AX650A/AX650N 不支持 lane map 配置，内部配置不参考 ClkLane，可以默认配置为全 0。

5.2 status

5.2.1 AX650A/AX650N status

属性：

参数名称		描述
MIPI STATUS	MIPIDEV	Mipi Dev Id 号
	CtrlId	Mipi Ctrl Id 号
	ErrorStatus0	Mipi 解析数据包头检查，（工作正常 0x0）
	ErrorStatus1	Mipi 解析数据 CRC 校验，（工作正常 0x0）
	PhyId	Mipi Phy Id 号
	PhyStatus	Phy 状态检查，(工作正常 0x1）
	ClkLaneStatus	Clk lane 传输数据状态，（重复多次查看在 0 和 1 来回跳动，表示 Mipi Clk Lane 在传输数据和 LP11 状态来回切换。如果 Mipi Clk 配置为 continue 模式，ClkLaneStatus 固定为 0。）
	DataLaneStatus	Data lane 传输数据状态，（重复多次查看在 0 和 1 来回跳动，表示 Mipi Data Lane 在传输数据和 LP11 状态来回切换）

5.2.2 AX630C/AX620Q status

参数名称		描述
MIPI STATUS	MIPIDEV	Mipi Dev Id 号
	CtrlId	Mipi Ctrl Id 号
	ErrorStatus0	Mipi 数据错误信息（0x0 工作正常， 其他值代表数据有错误）

参数名称		描述
	ErrorStatus1	Mipi phy 错误状态信息（0x0 工作正常，其他值代表 Phy 状态有错误）
	PhyStatus	Phy 状态检查 ● 2lane 下的值会在 0x2206 和 0x3307 之间变化 ● 4lane 下的值会在 0x222206 和 0x333307 之间变化 ● “2222”/“3333”代表每一对 data lane 的变化情况，“06”/“07”代表 clk lane 的变化情况
	ResetCount	出现 mipi 数据错误时，触发 reset mipi 的次数
	ErrorCount	出现 mipi 数据错误的总次数

6 FAQ

6.1 MIPI 配置流程说明

MIPI 模式下需要配置 PHY 的工作模式、数据传输所用的通道数、数据类型等参数。

推荐 MIPI 模式软件配置流程如下：

步骤 1. 上电启动；

步骤 2. Sensor Reset.

步骤 3. 初始化： AX_MIPI_RX_Init；

步骤 4. 设置 Lane 分布模式： AX_MIPI_RX_SetLaneCombo；

步骤 5. 设置参数： AX_MIPI_RX_SetAttr；

步骤 6. 复位： AX_MIPI_RX_Reset；

步骤 7. 使能： AX_MIPI_RX_Start；

步骤 8. 准备接收数据（使能 DEV 和 PIPE）

步骤 9. Sensor stream on.

推荐 MIPI 模式软件退出流程如下：

步骤 1. Sensor 关流：Sensor stream off

步骤 2. Mipi 停止接收：AX_MIPI_RX_Stop；

步骤 3. Mipi 去初始化: AX_MIPI_RX_DeInit。

！ 注意：

sensor 的 reset 需要在 mipi rx reset 之前配置。

6.2 MIPI LANE ID 如何配置：

6.2.1 AX630C/AX620Q mipi 单摄 4lane

PHY 硬件 lane 编号	推荐连接方式	Lane map 配置参考
Lane0	Data lane 0	.nDataLaneMap[0] = 0, .nDataLaneMap[1] = 1, .nDataLaneMap[2] = 3, .nDataLaneMap[3] = 4, .nClkLane[0] = 2, .nClkLane[1] = 5, 这里 lane5 没用，可以配置为-1. 也可以配置为 5
Lane1	Data lane 1	
Lane2	Clock lane	
Lane3	Data lane 2	
Lane4	Data lane 3	
Lane5	不使用	

6.2.2 AX630C/AX620Q mipi 双摄 2lane

PHY 硬件 lane 编号	推荐连接方式	Lane map 配置参考
Lane0	Data lane 0	.nDataLaneMap[0] = 0, .nDataLaneMap[1] = 1, .nDataLaneMap[2] = 3, .nDataLaneMap[3] = 4, .nClkLane[0] = 2, .nClkLane[1] = 5
Lane1	Data lane 1	
Lane2	Clock lane 0	
Lane3	Data lane 2	
Lane4	Data lane 3	
Lane5	Clock lane 1	

6.2.3 AX630C/AX620Q mipi 单摄 1lane

PHY 硬件 lane 编号	推荐连接方式	Lane map 配置参考
Lane0	Data lane 0	.nDataLaneMap[0] = 0,

Lane1	不使用	.nDataLaneMap[1] = -1, .nDataLaneMap[2] = -1, .nDataLaneMap[3] = -1, .nClkLane[0] = 2, .nClkLane[1] = -1, 不使用 PHY 硬件 Lane1、Lane3、Lane4、Lane5
Lane2	Clock lane 0	
Lane3	不使用	
Lane4	不使用	
Lane5	不使用	

6.2.4 AX630C/AX620Q mipi 双摄 1lane

PHY 硬件 lane 编号	推荐连接方式	Lane map 配置参考
Lane0	Data lane 0	.nDataLaneMap[0] = 0, .nDataLaneMap[1] = -1, .nDataLaneMap[2] = 3, .nDataLaneMap[3] = -1, .nClkLane[0] = 2, .nClkLane[1] = 5, 不使用 PHY 硬件 Lane1、Lane4
Lane1	不使用	
Lane2	Clock lane 0	
Lane3	Data lane 2	
Lane4	不使用	
Lane5	Clock lane 1	

6.3 AX630C/AX620Q MIPI 双摄配置 PHY 的限制

当 PHY 采用 2+2 双摄连接方式时。PHY 分成两组

分组	PHY 硬件 lane 编号	说明
0	Lane0	组内 3 个 Lane 之间是可以做 swap, Clock Lane 范围 0~2,
	Lane1	

	Lane2	Data Lane 范围 0~2
1	Lane3	组内 3 个 Lane 之间是可以做 swap, Clock Lane 范围 3~5, Data Lane 范围 3~5
	Lane4	
	Lane5	

6.4 MIPI CLK 配置

Mipi clock 分为连续模式和非连续模式，这个模式由 sensor 或者输入信号决定。

平台端建议统一配置为非连续模式，原因如下：

Mipi 接收数据过程不可避免会收到外界或者线路方面的干扰，会导致 mipi 接收数据异常，需要 mipi reset 进行 Mipi 的复位操作来恢复正常数据接收。

当 sensor 输入为连续模式，Mipi 中途复位会导致接收端无法正常接收数据，而非连续模式就不会有这个问题，所以建议申请 sensor setting 时，配置 mipi clk 模式为非连续模式。