



CV184X IPCM 客制化通信使用手册

Version: 1.0.0

Release date: 2025-03-21

©2025 北京晶视智能科技有限公司
本文件所含信息归北京晶视智能科技有限公司所有。
未经授权，严禁全部或部分复制或披露该等信息。

目录

1	声明	2
2	IPCM_PORT_COMMON	3
2.1	功能概述	3
2.2	设计概述	3
2.3	API 参考	4
2.3.1	CVI_IPCM_Init	4
2.3.2	CVI_IPCM_Uninit	5
2.3.3	CVI_IPCM_InvData	5
2.3.4	CVI_IPCM_FlushData	6
2.3.5	CVI_IPCM_DataLock	7
2.3.6	CVI_IPCM_DataUnlock	7
2.3.7	CVI_IPCM_GetBuff	8
2.3.8	CVI_IPCM_ReleaseBuff	9
2.3.9	CVI_IPCM_GetUserAddr	10
2.3.10	CVI_IPCM_GetParamBinAddr	10
2.3.11	CVI_IPCM_GetParamBakBinAddr	11
2.3.12	CVI_IPCM_GetPQBinQddr	12
2.3.13	CVI_IPCM_CustInit	12
2.3.14	CVI_IPCM_CustUninit	13
2.3.15	CVI_IPCM_CustPoolReset	14
2.3.16	CVI_IPCM_RegisterCustHandle	14
2.3.17	CVI_IPCM_DeregisterCustHandle	15
2.3.18	CVI_IPCM_CustSendMsg	16
2.3.19	CVI_IPCM_CustSendParam	17
2.3.20	CVI_IPCM_SetRtosSysBootStat	17
2.3.21	CVI_IPCM_ClrRtosSysBootStat	18
2.3.22	CVI_IPCM_GetRtosBootStat	19
2.3.23	CVI_IPCM_GetRtosIpcmStat	19
2.3.24	CVI_IPCM_SetRtosBootLogoStat	20
2.3.25	CVI_IPCM_ClrRtosBootLogoStat	20
2.3.26	CVI_IPCM_GetRtosBootLogoStat	21
2.4	数据类型	22
2.4.1	CVI_IPCM_CUST_VUART_PORT	22
2.4.2	CVI_IPCM_CUST_MSG_PORT	22
2.4.3	CVI_IPCM_CUST_KER_PORT_ST	22
2.4.4	IPCM_MSG_TYPE_E	23
2.4.5	IPCM_CUST_SHM_DATA_S	24
2.4.6	IPCM_CUST_MSG_S	24
2.4.7	IPCM_CUST_MSGPROC_FN	25

2.5 错误码 26

修订记录

Revision	Date	Description
v1.0.0	2025/03/21	初稿

1 声明



法律声明

本数据手册包含北京晶视智能科技有限公司（下称“晶视智能”）的保密信息。未经授权，禁止使用或披露本数据手册中包含的信息。如您未经授权披露全部或部分保密信息，导致晶视智能遭受任何损失或损害，您应对因之产生的损失/损害承担责任。

本文件内信息如有更改，恕不另行通知。晶视智能不对使用或依赖本文件所含信息承担任何责任。本数据手册和本文件所含的所有信息均按“原样”提供，无任何明示、暗示、法定或其他形式的保证。晶视智能特别声明未做任何适销性、非侵权性和特定用途适用性的默示保证，亦对本数据手册所使用、包含或提供的任何第三方的软件不提供任何保证；用户同意仅向该第三方寻求与此相关的任何保证索赔。此外，晶视智能亦不对任何其根据用户规格或符合特定标准或公开讨论而制作的可交付成果承担责任。

联系我们

地址 北京市海淀区丰豪东路 9 号院中关村集成电路设计园（ICPARK）1 号楼

深圳市宝安区福海街道展城社区会展湾云岸广场 T10 栋

电话 +86-10-57590723 +86-10-57590724

邮编 100094（北京）518100（深圳）

官方网站 <https://www.sophgo.com/>

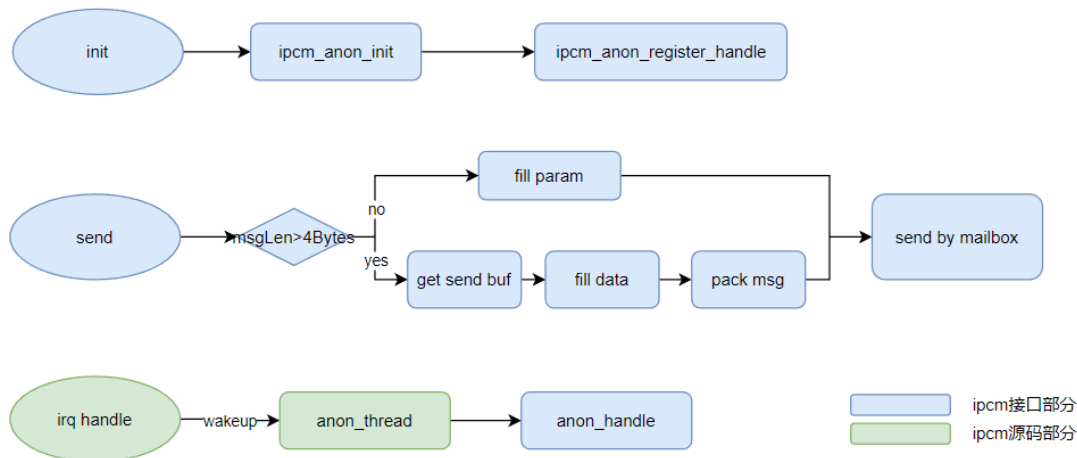
技术论坛 <https://developer.sophgo.com/forum/index.html>

2 IPCM_PORT_COMMON

2.1 功能概述

为 ipcm 提供访问 share memory、操作 cache、获取 param/param_bak/pqbin addr 的接口。
为 app 提供双核通信的 api 接口。

2.2 设计概述



- ipcm cust 初始化只需要调用 CVI_IPCM_CustInit 和调用 CVI_IPCM_RegisterCustHandle 注册 handle 即可
- 如果 msg 长度不超过 4byte，例如只传一下 command，则可以直接填充 msg 结构体的 param 域，然后通过 mailbox 发送到另一个 RISC-V
- 如果 msg 长度超过 4byte，则需要从 share memory 中申请一块 pool，然后填充 pool 内容和调用 CVI_IPCM_DataPacked 进行 msg 打包，最后再通过 mailbox 发送到另一个 RISC-V

注意： msg size 有长度限制

1. 最大不能超过 4095
2. 最大不能超过 CVI_IPCM_Init 时配置的最大 pool size

- 当 RISC-V 收到 cust msg 时, cust_thread 会被唤醒, 并调用 cust 初始化时注册的 handle, 在 handle 中可以通过 port_id 和 msg_id 区分不同的 cust msg

2.3 API 参考

2.3.1 CVI_IPCM_Init

【描述】

ipcm 初始化。

【语法】

```
CVI_S32 CVI_IPCM_Init(void);
```

【参数】

无

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见错误码。

【需求】

- 头文件: cvi_type.h, cvi_ipcm.h

【注意】

无

【举例】

无

【相关主题】

无

2.3.2 CVI_IPCM_Uninit

【描述】

ipcm 去初始化。

【语法】

```
CVI_S32 CVI_IPCM_Uninit(void);
```

【参数】

无

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见错误码。

【需求】

- 头文件: cvi_type.h, cvi_ipcm.h

【注意】

无

【举例】

无

【相关主题】

无

2.3.3 CVI_IPCM_InvData

【描述】

使指定地址的 cache 无效。

【语法】

```
CVI_S32 CVI_IPCM_InvData(CVI_VOID *pData, CVI_U32 u32Size);
```

【参数】

参数名称	描述	输入/输出
pData	数据地址, 需要 64byte 对齐	输入
u32Size	数据长度, 需要 64byte 对齐	输入

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见错误码。

【需求】

- 头文件: cvi_type.h, cvi_ipcm.h

【注意】

无

【举例】

无

【相关主题】

无

2.3.4 CVI_IPCM_FlushData

【描述】

将指定地址的内容 flush 到 ddr。

【语法】

```
CVI_S32 CVI_IPCM_FlushData(CVI_VOID *pData, CVI_U32 u32Size);
```

【参数】

参数名称	描述	输入/输出
pData	数据地址, 需要 64byte 对齐	输入
u32Size	数据长度, 需要 64byte 对齐	输入

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见错误码。

【需求】

- 头文件: cvi_type.h, cvi_ipcm.h

【注意】

无

【举例】

无

【相关主题】

无

2.3.5 CVI_IPCM_DataLock

【描述】

大小核访问互斥锁上锁，同一个 lock id，只有拿到 lock 的核才能访问。

【语法】

```
CVI_S32 CVI_IPCM_DataLock(CVI_U8 u8LockID);
```

【参数】

参数名称	描述	输入/输出
u8LockID	锁 id 号；当前支持 5 组 data lock，lock id 的范围是 [0,4]	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件: cvi_type.h, cvi_ipcm.h

【注意】

无

【举例】

无

【相关主题】

无

2.3.6 CVI_IPCM_DataUnlock

【描述】

大小核访问互斥锁解锁。

【语法】

```
CVI_S32 CVI_IPCM_DataUnlock(CVI_U8 u8LockID);
```

【参数】

参数名称	描述	输入/输出
u8LockID	锁 id 号；当前支持 5 组 data lock, lock id 的范围是 [0,4]	输入

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见错误码。

【需求】

- 头文件: cvi_type.h, cvi_ipcm.h

【注意】

无

【举例】

无

【相关主题】

无

2.3.7 CVI_IPCM_GetBuff

【描述】

从 share memory 中申请一块空闲 pool, 返回 pool 的地址。

【语法】

```
CVI_VOID *CVI_IPCM_GetBuff(CVI_U32 u32Size);
```

【参数】

参数名称	描述	输入/输出
u32Size	申请 pool 的长度	输入

【返回值】

返回值	描述
NULL	失败
其他	返回 pool 的地址。

【需求】

- 头文件: cvi_type.h, cvi_ipcm.h

【注意】

无

【举例】

无

【相关主题】

无

2.3.8 CVI_IPCM_ReleaseBuff

【描述】

释放指定地址的 share memory pool。

【语法】

```
CVI_S32 CVI_IPCM_ReleaseBuff(CVI_VOID *pData);
```

【参数】

参数名称	描述	输入/输出
pData	pool 的地址	输入

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见错误码。

【需求】

- 头文件: cvi_type.h, cvi_ipcm.h

【注意】

无

【举例】

无

【相关主题】

无

2.3.9 CVI_IPCM_GetUserAddr

【描述】

返回对应物理地址的虚拟地址，物理地址的范围必须是 alios 管理的地址。

在 **linux** 侧，将从 alios 拿到的物理地址转成虚拟地址；

在 **alios** 侧，因为 alios 没有开 mmc，所以直接返回 alios 的物理地址。

【语法】

```
CVI_VOID *CVI_IPCM_GetUserAddr(CVI_U32 u32PAddr);
```

【参数】

参数名称	描述	输入/输出
u32PAddr	待转换的物理地址	输入

【返回值】

返回值	描述
NULL	失败，物理地址不在 alios 管理的范围
其他	成功，返回对应物理地址的虚拟地址

【需求】

- 头文件: cvi_type.h, cvi_ipcm.h

【注意】

无

【举例】

无

【相关主题】

无

2.3.10 CVI_IPCM_GetParamBinAddr

【描述】

获取参数区域的物理地址。

【语法】

```
CVI_U32 CVI_IPCM_GetParamBinAddr(void);
```

【参数】

无

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见错误码。

【需求】

- 头文件: `cvi_type.h`, `cvi_ipcm.h`

【注意】

无

【举例】

无

【相关主题】

无

2.3.11 CVI_IPCM_GetParamBakBinAddr

【描述】

获取参数备份区域的物理地址。

【语法】

```
CVI_U32 CVI_IPCM_GetParamBakBinAddr(void);
```

【参数】

无

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见错误码。

【需求】

- 头文件: `cvi_type.h`, `cvi_ipcm.h`

【注意】

无

【举例】

无

【相关主题】

无

2.3.12 CVI_IPCM_GetPQBinQddr

【描述】

获取 pq 参数的物理地址。

【语法】

```
CVI_U32 CVI_IPCM_GetPQBinQddr(void);
```

【参数】

无

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见错误码。

【需求】

- 头文件: cvi_type.h, cvi_ipcm.h

【注意】

无

【举例】

无

【相关主题】

无

2.3.13 CVI_IPCM_CustInit

【描述】

ipcm cust 初始化。

【语法】

```
CVI_S32 CVI_IPCM_CustInit(void);
```

【参数】

无

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见错误码。

【需求】

- 头文件: `cvi_type.h`, `cvi_ipcm.h`

【注意】

无

【举例】

无

【相关主题】

无

2.3.14 CVI_IPCM_CustUninit

【描述】

ipcm cust 去初始化。

【语法】

```
CVI_S32 CVI_IPCM_CustUninit(void);
```

【参数】

无

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见错误码。

【需求】

- 头文件: `cvi_type.h`, `cvi_ipcm.h`

【注意】

无

【举例】

无

【相关主题】

无

2.3.15 CVI_IPCM_CustPoolReset

【描述】

将 share memory pool 全部复位为 free 状态，一般在系统第一次启动时 ipcm 驱动会调用。

【语法】

```
CVI_S32 CVI_IPCM_CustPoolReset(void);
```

【参数】

无

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见错误码。

【需求】

- 头文件: cvi_type.h, cvi_ipcm.h

【注意】

无

【举例】

无

【相关主题】

无

2.3.16 CVI_IPCM_RegisterCustHandle

【描述】

注册 ipcm cust handler。

【语法】

```
CVI_S32 CVI_IPCM_RegisterCustHandle(CVI_U8 u8PortID, IPCM_CUST_MSGPROC_FN_
↪ pfnHandler, CVI_VOID *pData);
```

【参数】

参数名称	描述	输入/输出
u8PortID	port id, 当前 0 和 1 分别给虚拟串口、参数解析使用, 用户 port id 建议从 2 开始	输入
pfnHandle	cust handler	输入
pData	handler 应用数据	输入

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见 错误码 。

【需求】

- 头文件: `cvi_type.h`, `cvi_ipcm.h`, `cvi_comm_ipcm.h`

【注意】

port id, 当前 0 和 1 分别给虚拟串口、参数解析使用, 用户 port id 建议从 2 开始

【举例】

无

【相关主题】

无

2.3.17 CVI_IPCM_DeregisterCustHandle

【描述】

注销 ipcm cust handler。

【语法】

```
CVI_S32 CVI_IPCM_DeregisterCustHandle(CVI_U8 u8PortID);
```

【参数】

参数名称	描述	输入/输出
u8PortID	port id	输入

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见 错误码 。

【需求】

- 头文件: `cvi_type.h`, `cvi_ipcm.h`

【注意】

无

【举例】

无

【相关主题】

无

2.3.18 CVI_IPCM_CustSendMsg

【描述】

将 share memory 的数据通过 mailbox 发送到另一个核

【语法】

```
CVI_S32 CVI_IPCM_CustSendMsg(CVI_U8 u8PortID, CVI_U8 u8MsgID, CVI_VOID *pData, CVI_U32 u32Len);
```

【参数】

参数名称	描述	输入/输出
u8PortID	port id 号	输入
u8MsgID	msg id, 范围是 [0,127]	输入
pData	data 指针, 该 data 需要通过 CVI_IPCM_GetBuff 从 share memory 中获取	输入
u32Len	data 的长度	输入

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见错误码。

【需求】

- 头文件: cvi_type.h, cvi_ipcm.h

【注意】

无

【举例】

无

【相关主题】

CVI_IPCM_GetBuff

2.3.19 CVI_IPCM_CustSendParam

【描述】

当 msg 不超过 4byte 时，可以通过该接口发送 4byte 的 param。

【语法】

```
CVI_S32 CVI_IPCM_CustSendParam(CVI_U8 u8PortID, CVI_U8 u8MsgID, CVI_U32 u32Param);;
```

【参数】

参数名称	描述	输入/输出
u8PortID	port id 号	输入
u8MsgID	msg id, 范围是 [0,127]	输入
u32Param	param 内容	输入

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见错误码。

【需求】

· 头文件: cvi_type.h, cvi_ipcm.h

【注意】

无

【举例】

无

【相关主题】

无

2.3.20 CVI_IPCM_SetRtosSysBootStat

【描述】

将 rtos 启动状态位设为 1；只在 rtos 端有效

【语法】

```
CVI_S32 CVI_IPCM_SetRtosSysBootStat(void);
```

【参数】

无

【返回值】

无

【需求】

· 头文件: `cvi_type.h`, `cvi_ipcm.h`

【注意】

该接口一般在 rtos 启动时调用, 且仅在 rtos 端调用有效

【举例】

无

【相关主题】

无

2.3.21 CVI_IPCM_ClrRtosSysBootStat

【描述】

将 rtos 启动状态位设为 0; 只在 rtos 端有效

【语法】

<code>CVI_S32 CVI_IPCM_ClrRtosSysBootStat(void);</code>

【参数】

无

【返回值】

无

【需求】

· 头文件: `cvi_type.h`, `cvi_ipcm.h`

【注意】

无

【举例】

无

【相关主题】

无

2.3.22 CVI_IPCM_GetRtosBootStat

【描述】

获取 rtos 启动状态

【语法】

```
CVI_S32 CVI_IPCM_GetRtosBootStat(void);
```

【参数】

无

【返回值】

无

【需求】

- 头文件: cvi_type.h, cvi_ipcm.h

【注意】

无

【举例】

无

【相关主题】

无

2.3.23 CVI_IPCM_GetRtosIpcmStat

【描述】

获取 ipcm 启动状态

【语法】

```
CVI_S32 CVI_IPCM_GetRtosIpcmStat(void);
```

【参数】

无

【返回值】

无

【需求】

- 头文件: cvi_type.h, cvi_ipcm.h

【注意】

无

【举例】

无

【相关主题】

无

2.3.24 CVI_IPCM_SetRtosBootLogoStat

【描述】

将 rtos boot logo 状态位设为 1

【语法】

```
CVI_S32 CVI_IPCM_SetRtosBootLogoStat(void);
```

【参数】

无

【返回值】

无

【需求】

- 头文件: cvi_type.h, cvi_ipcm.h

【注意】

无

【举例】

无

【相关主题】

无

2.3.25 CVI_IPCM_ClrRtosBootLogoStat

【描述】

将 rtos boot logo 状态位设为 0

【语法】

```
CVI_S32 CVI_IPCM_ClrRtosBootLogoStat(void);
```

【参数】

无

【返回值】

无

【需求】

· 头文件: cvi_type.h, cvi_ipcm.h

【注意】

无

【举例】

无

【相关主题】

无

2.3.26 CVI_IPCM_GetRtosBootLogoStat

【描述】

获取 rtos boot logo 态位

【语法】

CVI_S32 CVI_IPCM_GetRtosBootLogoStat(void);

【参数】

无

【返回值】

无

【需求】

· 头文件: cvi_type.h, cvi_ipcm.h

【注意】

无

【举例】

无

【相关主题】

无

2.4 数据类型

2.4.1 CVI_IPCM_CUST_VUART_PORT

【说明】

虚拟串口使用的客制化通信 port 号

【定义】

```
#define CVI_IPCM_CUST_VUART_PORT 0
```

【注意事项】

无

【相关数据类型及接口】

无

2.4.2 CVI_IPCM_CUST_MSG_PORT

【说明】

参数解析使用的客制化通信 port 号

【定义】

```
#define CVI_IPCM_CUST_MSG_PORT 1
```

【注意事项】

无

【相关数据类型及接口】

无

2.4.3 CVI_IPCM_CUST_KER_PORT_ST

【说明】

linux 端内核态可以使用的起始 port 号，支持内核态注册；

在 linux 端，port 号大于或等于 CVI_IPCM_CUST_KER_PORT_ST 的消息会被内核态先接收，调用内核态的 callback；

存在以下情况：

1. 用户态和内核态都注册了 callback，则 alios 的 msg 会先到内核态；

内核态处理完毕后，如果 callback 返回 1，则用户态不会再收到 msg，即用户态 callback 不会被调用

内核态处理完毕后，如果 callback 返回 0，则 msg 会继续往用户态发送，即用户态 callback 也会被调用

2. 内核态注册了 callback，用户态没有注册 callback，则 msg 只会在内核态处理
3. 内核态没有注册 callback，用户态注册了 callback，则 msg 只会在用户态处理
4. 用户态和内核态都没有注册 callback，则 msg 不会被处理，会打印 err

【定义】

```
#define CVI_IPCM_CUST_KER_PORT_ST 64
```

【注意事项】

无

【相关数据类型及接口】

无

2.4.4 IPCM_MSG_TYPE_E

【说明】

定义 msg 的类别

【定义】

```
typedef enum _IPCM_MSG_TYPE_E {
    IPCM_MSG_TYPE_SHM = 0,    // msg_param is share memory addr
    IPCM_MSG_TYPE_RAW_PARAM,  // msg_param is the param
} IPCM_MSG_TYPE_E;
```

【成员】

成员名称	描述
IPCM_MSG_TYPE_SHM	msg 长度超过 4byte，callback 返回的是 IPCM_CUST_SHM_DATA_S(share memory pool 的 addr 和 size)
IPCM_MSG_TYPE_RAW_PARAM	msg 内容不超过 4bytes，callback 返回的是 msg param

【注意事项】

无

【相关数据类型及接口】

无

2.4.5 IPCM_CUST_SHM_DATA_S

【说明】

当 msg 长度超过 4byte, 定义 share memory pool 的 addr 和 size

【定义】

```
typedef struct _IPCM_CUST_SHM_DATA_S {  
    CVI_VOID *pData;  
    CVI_U32 u32Size;  
} IPCM_CUST_SHM_DATA_S;
```

【成员】

成员名称	描述
pData	share memory 的地址
u32Size	share memory 数据的 size

【注意事项】

无

【相关数据类型及接口】

无

2.4.6 IPCM_CUST_MSG_S

【说明】

定义 ipcm cust 消息结构体, 注册 cust handler 后, 如果收到 cust 消息, handler 会返回该结构体的指针

【定义】

```
typedef struct _IPCM_CUST_MSG_S {  
    CVI_U8 u8PortID;  
    CVI_U8 u8MsgID : 7;  
    CVI_U8 u8DataType : 1;  
    union {  
        IPCM_CUST_SHM_DATA_S stData; // while u8DataType is 0  
        CVI_U32 u32Param; // while u8DataType is 1  
    };  
} IPCM_CUST_MSG_S;
```

【成员】

成员名称	描述
u8PortID;	port 号
u8MsgID	msg 号
u8DataType	msg 数据类型 0: 返回 stData 1: 返回 u32Param

【注意事项】

无

【相关数据类型及接口】

无

2.4.7 IPCM_CUST_MSGPROC_FN

【说明】

定义 ipcm cust 的 handler

【定义】

```
typedef CVI_S32 (*IPCM_CUST_MSGPROC_FN)(CVI_VOID *pPriv, IPCM_CUST_MSG_S_
↪ *pstData);
```

【成员】

成员名称	描述
pPriv	用户数据指针；注册 handle 时传入的数据指针
pstData	接收到的 msg 指针

【注意事项】

无

【相关数据类型及接口】

无

2.5 错误码

错误代码	宏定义	描述
-200	-EQFULL	msg 队列消息未及时处理，msg 队列已满
-201	-EQEMPTY	msg 队列中不存在 msg，msg 队列为空
-202	-EMLOCK	mailbox 上锁失败
-203	-EMBUSY	mailbox 通道已全部使用
-204	-EQNULL	msg 队列未初始化