

MI SED API

Version 2.04

© 2020 SigmaStar Technology Corp. All rights reserved.

SigmaStar Technology makes no representations or warranties including, for example but not limited to, warranties of merchantability, fitness for a particular purpose, non-infringement of any intellectual property right or the accuracy or completeness of this document, and reserves the right to make changes without further notice to any products herein to improve reliability, function or design. No responsibility is assumed by SigmaStar Technology arising out of the application or use of any product or circuit described herein; neither does it convey any license under its patent rights, nor the rights of others.

SigmaStar is a trademark of SigmaStar Technology Corp. Other trademarks or names herein are only for identification purposes only and owned by their respective owners.

REVISION HISTORY

Revision No.	Description	Date
2.0.4	<ul style="list-style-type: none">Initial release	01/20/2020

TABLE OF CONTENTS

REVISION HISTORY	i
TABLE OF CONTENTS.....	ii
1. API 参考	1
1.1. 概述.....	1
1.1.1 目前智能编码模块支持的图像检测算法	1
1.1.2 sed 的作用	2
1.1.3 SED 的数据来源	2
1.1.4 Sed 检测流程图如下	2
1.1.5 Sed 人非机动车检测功能支持的平台限制.....	2
1.2. 关键名词说明	2
1.2.1 VDF (Video detection function)	2
1.2.2 IVE (Intelligent video coding)	3
1.2.3 IPU (AI Process Unit)	3
1.2.4 ROI(Region Of Interest)	3
1.2.5 Motion 信息	4
1.2.6 QP (Quantization Parameter)	4
1.2.7 MD.....	4
1.2.8 OD	4
1.2.9 VG	4
1.2.10 SAD.....	4
1.2.11 CBR (Constant Bit Rate)	5
1.2.12 VBR (Variable Bit Rate)	5
1.2.13 CVBR (Constrained VariableBit Rate)	5
1.2.14 ABR (Average Bit Rate)	5
1.2.15 AVBR(Adaptive Variable Bit Rate)	5
1.2.16 IOU (Intersection Over Union)	6
2 功能模块 API	8
2.1 MI_SED_CreateChn.....	9
2.2 MI_SED_DestroyChn	9
2.3 MI_SED_StartDetector.....	11
2.4 MI_SED_StopDetector	13
2.5 MI_SED_AttachToChn	15
2.6 MI_SED_DetachFromChn.....	15
2.7 MI_SED_SetDbgLevel	16
2.8 MI_SED_GetRect.....	17
3 SED 数据类型.....	19
MI_SED_AlgoAttr_t.....	19
3.1 SED_MAX_CHN_NUM.....	20
3.2 SED_MAX_ROI_NUM_PER_CHN	20
3.3 SED_MAX_TARGET_CHN_NUM_PER_CHN	20

3.4	SED_MAX_CUS_DEF_ALGOPARAM_NUM	21
3.5	MI_SED_CHN	21
3.6	MI_SED_TARGET_CHN	21
3.7	MI_SED_AlgoType_e.....	22
3.8	MI_SED_MdMbMode_e	22
3.9	MI_SED_InputAttr_t.....	23
3.10	MI_SED_Default_AlgoParam_t.....	23
3.11	MI_SED_CusDef_AlgoParam_t.....	24
3.12	MI_SED_AlgoAttr_t	25
3.13	MI_SED_TargetAttr_t.....	25
3.14	MI_SED_DetectorAttr_t.....	26
4	错误码.....	27
5	How Build and Run Demo	28
5.1	How to Build Sed module and demo code:	28
5.2	How to Get sed bin & Demo Test bin.....	28
5.3	Sed Demo code 示例 :	28

1. API 参考

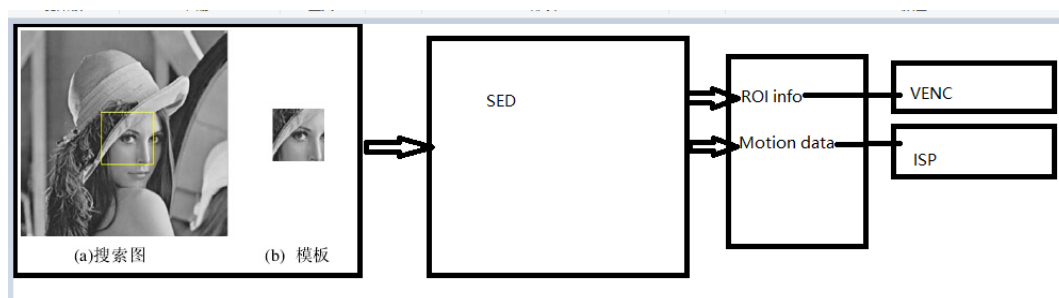
1.1. 概述

SED（智能编码）

智能编码模块主要提供智能编码通道的创建和销毁、开启和停止检测源图像、计算结果并关联到指定的编码通道等功能。

智能编码的核心功能是先做图像的识别（识别图像中的物体有物体运动，或者识别图像中物体种类），再设置编码参数到 venc module（当识别到物体有运动，可以控制 venc 模块降低 QP 值，提高清晰度（sharpness），当识别图像是静止状态的话，控制 venc 模块增加 QP 值，减少传输带宽等）

大概检测算法流程是比较前后 2 张 frame 的差异来检测画面有无移动，提取出画面 ROI 和 motion 信息，给编码器设置 ROI 指定区域的 QP 值来编码。检测算法有多种，下面会介绍我们平台使用到的有那些检测算法。



通过 SED 智能识别 ROI，动态设置图像指定位置 ROI 的 QP 值 可以有效的降低带宽，同时保证图像的质量。



1.1.1 目前智能编码模块支持的图像检测算法

- 1: E_MI_IVEOBJDETECT_ALGOPARAM 基于 VDF/MD mode 的运动物体检测,支持 ROI 对像追踪
- 2: E_MI_CNNOBJDETECT_ALGOPARAM 人非车检测可以检测的物体内容如下: bicycle, bus, car, motorbike, person) 支持 ROI 对像追踪
- 3: E_MI_MOTIONDETECT_ALGOPARAM 基于 IVE 的动静检测，只支持 for AVBR motion detection

以上三种检测算法可以同时支持，只需要创建不同的检测通道实例来实现即可。

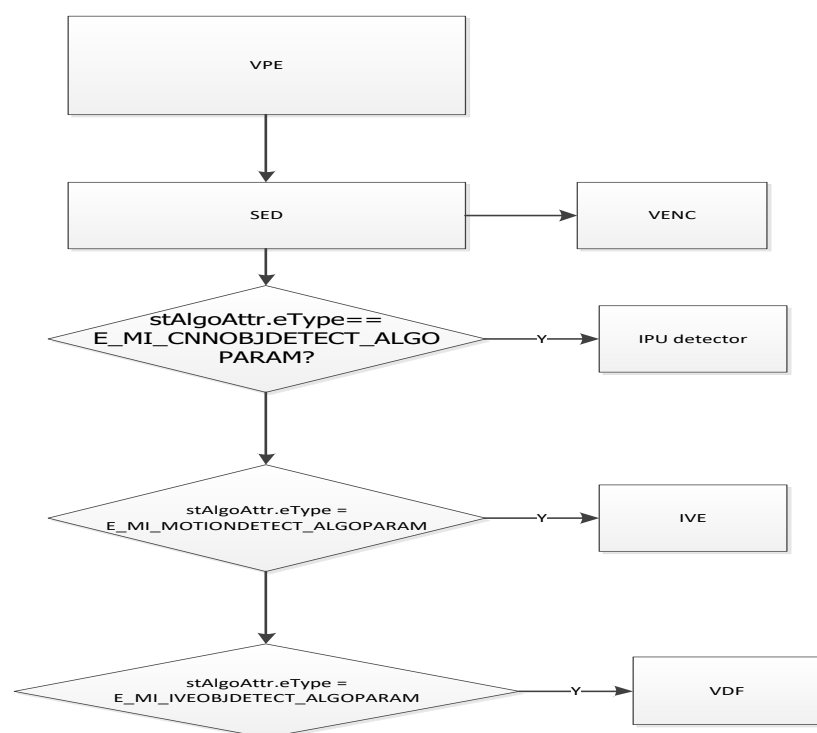
1.1.2 sed 的作用

负责使用 Sigmastar or 第三方智能算法，识别 ROI 及 motion 信息。ROI 信息设置给 VENC，motion 信息用于自动调整 ISP sharpness，etc

1.1.3 SED 的数据来源

是通过 VPE 获取 yuv data 来作智能检测（输入的 yuv 分辨率是 352*288）

1.1.4 Sed 检测流程图如下



1.1.5 Sed 人非车检测功能支持的平台限制

人非车检测是基 AI 训练模型通过 IPU 来作检测，具体检测的准确度依赖于 AI 模型训练的准确度。目前只有在 i6e 和 i6b0 平台才支持人非车检测,其他低阶平台目前还不支持。

1.2. 关键名词说明

1.2.1 VDF (Video detection function)

MI_VDF 实现 MD，OD，VG 视频通道的初始化，通道管理，视频检测结果的管理和通道销毁等功能。

1.2.2 IVE (Intelligent video coding)

IVE 模块负责图像数据的处理和计算，可以通过 MI_IVE_Sad 获取两张图片的 SAD 值，可以通过 MI_IVE_Hist 来执行直方图统计任务数据等做对笔，来做图像的动静检测

1.2.3 IPU (AI Process Unit)

AI 模型处理和计算单元，MI IPU 模块实现了网络模型的快速推理功能，为每个通道独立配置网络模型，管理输入和输出数据的获取和释放。

1.2.4 ROI(Region Of Interest)

ROI 编码，感兴趣区域编码，启用 ROI 功能后，重要的或者移动的区域将会进行高质量无损编码，而对那些不移动，不被选择的区域降低其码率和图像质量，进行标准清晰度视频压缩，甚至是不传输这部分区域视频，从而最终达到节省网络带宽占用和视频存储空间，用户可以通过配置 ROI 区域，对该区域的图像 Qp 进行限制，从而实现图像中该区域的 Qp 与其他图像区域的差异化。系统支持 H264 和 H265 编码设置 ROI，且提供 16 个 ROI 区域供用户同时使用。



16 个 ROI 区域可以互相叠加，且叠加时的优先级按照 0~15 的索引号依次提高，也即叠加区域的 Qp 最终判定只按最高优先级的区域处理。ROI 区域可以配置绝对 Qp 与相对 Qp 两种模式。

绝对 Qp:ROI 区域的 Qp 为用户设定的 Qp 值

相对 Qp:ROI 区域的 Qp 为码率控制产生的 Qp 与用户设定的 Qp 偏移值的和

以下示例编码图像采用 FixQp 模式，设置图像 Qp 为 30，即图像所有宏块 Qp 值为 30。ROI 区域 0 设置为绝对 Qp 模式，Qp 值为 20，索引为 0；ROI 区域 1 设置为相对 Qp 模式，Qp 为-15，索引为 1。因为 ROI 区域 0 的 index 小于 ROI 区域 1 的 index，所以在发生重叠的图像区域按高优先级的 ROI 区域 1 的 Qp 设置。除了重叠部分的 ROI 区域 0 的 Qp 值为 20，区域 1 的 Qp 值为 30-15=15。



1.2.5 Motion 信息

包含图像的 SAD 信息。

1.2.6 QP (Quantization Parameter)

QP 值对应量化步长的序号，值越小，量化步长越小，量化的精度就越高，画质也就越好，编码出来的 size 也越大。

1.2.7 MD

运动检测 (Motion detect) 用在拍摄影片中检测物体移动，被实际应用安全到安全监控等。

1.2.8 OD

遮挡检测(Occlusion detection)功能用于检测接收到的影片是否出现遮挡，并输出遮挡检测结果。

1.2.9 VG

虚拟线段用于检测是否有物体穿越设置的警报线。

区域入侵用于检测是否有物体穿越设置的警报区域。

1.2.10 SAD

即 Sum of Absolute Differences，就是差的绝对值的和。此算法常用于图像块匹配，将每个像素对应数值之差的绝对值求和，据此评估两个图像块的相似度。可以看出这个算法很快速、但并不精确，通常用于多级处理的初步筛选。SAD 值的取值范围为[0, 255]，所有 SAD 值被分为 16 个区间：

[0,10), [10,15), [15,20), [20,25), [25,30), [30,40), [40,50), [50,60), [60,70), [70,80), [80,90), [90,100), [100,120), [120,140), [140,160), [160, 255]。

此处统计一帧中所有 MB(8*8)落入以上区间的占总的 MB 数量的比例。

值越大，表示画面越静止。

1.2.11 CBR (Constant Bit Rate)

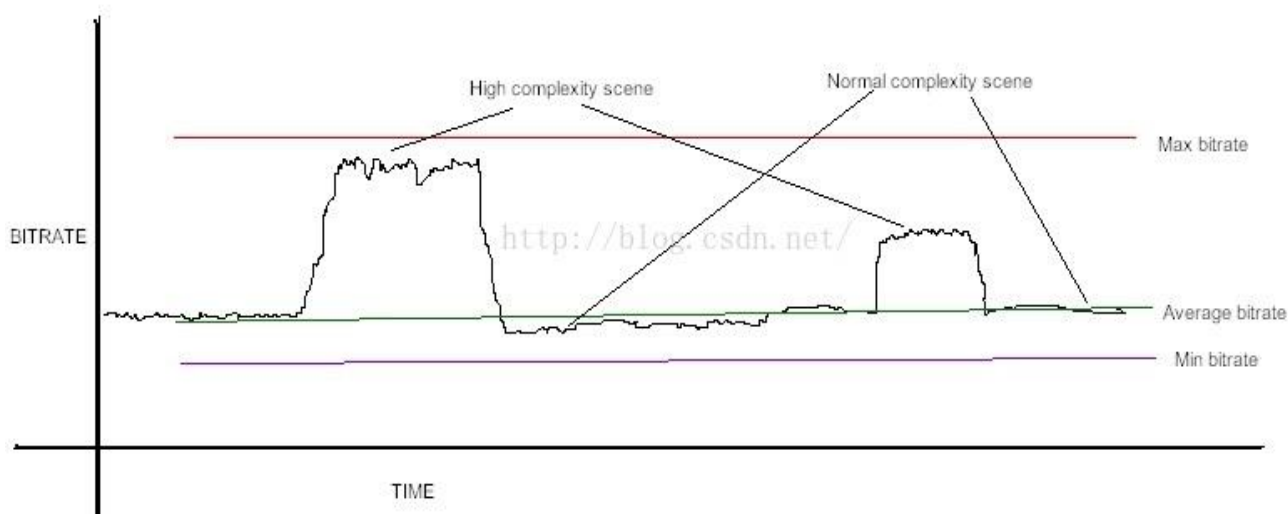
是以恒定比特率方式进行编码，有 **Motion** 发生时，由于码率恒定，只能通过增大 **QP** 来减少码字大小，图像质量变差，当场景静止时，图像质量又变好，因此图像质量不稳定。这种算法优先考虑码率(带宽)。

1.2.12 VBR (Variable Bit Rate)

动态比特率其码率可以随着图像的复杂程度的不同而变化，因此其编码效率比较高，**Motion** 发生时，马赛克很少。码率控制算法根据图像内容确定使用的比特率，图像内容比较简单则分配较少的码率(似乎码字更合适)，图像内容复杂则分配较多的码字，这样既保证了质量，又兼顾带宽限制。这种算法优先考虑图像质量。

1.2.13 CVBR (Constrained VariableBit Rate)

它是 **VBR** 的一种改进方法。但是 **Constrained** 又体现在什么地方呢？这种算法对应的 **Maximum bitRate** 恒定或者 **Average BitRate** 恒定。这种方法的兼顾了以上两种方法的优点：在图像内容静止时，节省带宽，有 **Motion** 发生时，利用前期节省的带宽来尽可能的提高图像质量，达到同时兼顾带宽和图像质量的目的。这种方法通常会让用户输入最大码率和最小码率，静止时，码率稳定在最小码率，运动时，码率大于最小码率，但是又不超过最大码率。比较理想的模型如下：



1.2.14 ABR (Average Bit Rate)

在一定的时间内达到设定的码率，但是局部码率峰值可以超过设定的码率，平均码率恒定。

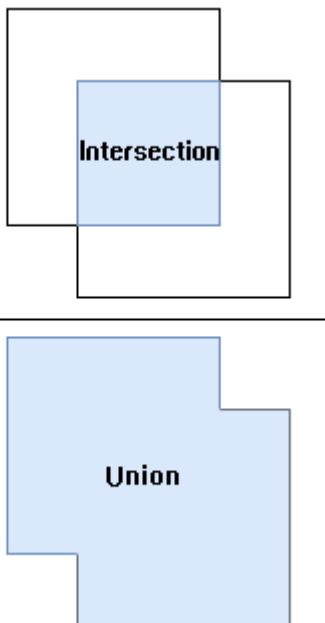
1.2.15 AVBR(Adaptive Variable Bit Rate)

AVBR (Adaptive Variable Bit Rate) 可变比特率，即允许在码率统计时间内编码码率波动，从而保证编码图像质量平稳。码率控制内部会检测当前场景的运动静止状态，在运动时用较高码率编码，在静止时主动降低目标码率。以H.264编码为例，**VENC**模块提供用户可设置**MaxBitrate**，**ChangePos** 和**MinStillPercent**。**MaxBitrate** 表示运动场景下的最大码率，**MaxBitrate*ChangePos*MinStillPercent** 表示静止情况下的最小码率。

根据运动程度的不同目标码率会在最大码率和最小码率间调整。MaxQp，MinQp用于控制图像的质量范围，码率控制以QP钳位为最高优先级，超出MinQp，MaxQp范围内码率控制将失效。

1.2.16 IOU (Intersection Over Union)

交并比 (Intersection-over-Union , IoU)，目标检测中使用的一个概念，是产生的候选框 (candidate bound) 与原标记框 (ground truth bound) 的交叠率，即它们的交集与并集的比值。(更详细的内容请参考：
<http://172.19.30.188:8090/pages/viewpage.action?pageId=1216493>)

$$IOU = \frac{A \cap B}{A \cup B}$$


最理想情况是完全重叠，即比值为 1。

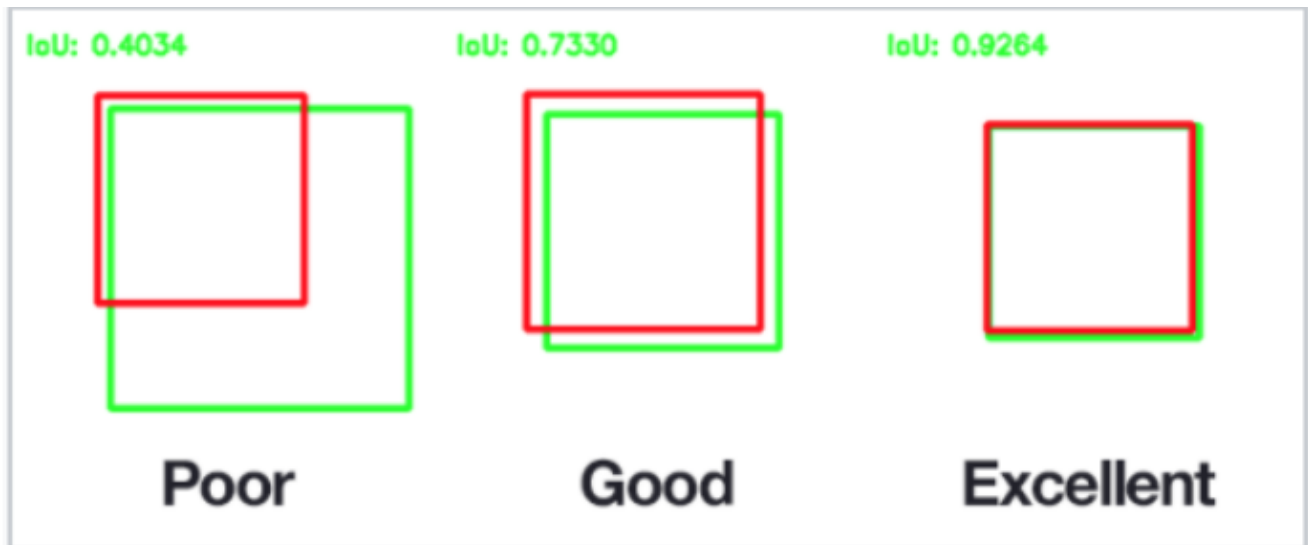


Diagram explaining IoU (from Wikipedia)

2. 功能模块 API

API 名	功能
MI_SED_CreateChn	创建 SED 通道
MI_SED_DestroyChn	销毁 SED 通道
MI_SED_StartDetector	开始检测指定的通道
MI_SED_StopDetector	停止检测
MI_SED_AttachToChn	关联到目标通道
MI_SED_DetachFromChn	停止关联

2.1. MI_SED_CreateChn

➤ 功能

创建 SED 通道。

➤ 语法

```
MI_S32 MI_SED_CreateChn(MI\_SED\_CHN SedChn, MI\_SED\_DetectorAttr\_t* pstAttr);
```

➤ 形参

参数名称	描述	输入/输出
SedChn	SED 通道号。 取值范围：[0, SED_MAX_CHN_NUM)。	输入
pstAttr	SED 检测属性指针。	输入

➤ 返回值

返回值 { 0 成功。
非 0 失败，参照[错误码](#)。

➤ 依赖

- 头文件：mi_common.h、mi_sed.h
- 库文件：libmi_sed.a

※ 注意

-

➤ 举例

无。

➤ 相关主题

无。

2.2. MI_SED_DestroyChn

➤ 功能

销毁 SED 通道。

➤ 语法

```
MI_S32 MI_SED_DestroyChn(MI\_SED\_CHN SedChn);
```


➤ 形参

参数名称	描述	输入/输出
SedChn	SED 通道号。 取值范围：[0, SED_MAX_CHN_NUM)。	输入

➤ 返回值

返回值 { 0 成功。
非 0 失败，参照[错误码](#)。

➤ 依赖

- 头文件：mi_common.h、mi_sed.h
- 库文件：libmi_sed.a

※ 注意

-

➤ 举例

无。

➤ 相关主题

无。

2.3. MI_SED_StartDetector

➤ 功能

开始检测指定通道。

➤ 语法

```
MI_S32 MI_SED_StartDetector(MI\_SED\_CHN SedChn);
```

➤ 参数

参数名称	描述	输入/输出
SedChn	SED 通道号。 取值范围：[0, SED_MAX_CHN_NUM)。	输入

➤ 返回值

返回值 { MI_OK 成功。

非 MI_OK 失败，参照[错误码](#)。

➤ 依赖

- 头文件：mi_common.h、mi_sed.h
- 库文件：libmi_sed.a

※ 注意

-

➤ 举例

无。

➤ 相关主题

无。

2.4. MI_SED_StopDetector

➤ 功能

停止检测指定通道。

➤ 语法

MI_S32 MI_SED_StopDetector([MI_SED_CHN](#) SedChn);

➤ 参数

参数名称	描述	输入/输出
SedChn	SED 通道号。 取值范围：[0, SED_MAX_CHN_NUM)。	输入

➤ 返回值

返回值 {
MI_OK 成功。
非 MI_OK 失败，参照[错误码](#)。

➤ 依赖

- 头文件：mi_common.h、mi_sed.h
- 库文件：libmi_sed.a

※ 注意

-

➤ 举例

- 相关主题
无。

2.5. MI_SED_AttachToChn

➤ 描述

关联到指定的编码通道。

➤ 语法

MI_S32 MI_SED_AttachToChn([MI_SED_CHN](#) SedChn, [MI_SED_TARGET_CHN](#) TargetChn);

➤ 参数

参数名称	描述	输入/输出
SedChn	SED 通道号。 取值范围：[0, SED_MAX_CHN_NUM)。	输入
TargetChn	目标编码通道号。 取值范围：[0, VENC_MAX_CHN_NUM)。	输入

➤ 返回值

返回值 {
MI_OK 成功。
非 MI_OK 失败，参照[错误码](#)。

➤ 依赖

- 头文件：mi_common.h、mi_sed.h
- 库文件：libmi_sed.a

※ 注意

-

➤ 举例

➤ 相关主题

无。

2.6. MI_SED_DetachFromChn

➤ 描述

取消对指定编码通道的关联。

➤ 语法

MI_S32 MI_SED_DetachFromChn([MI_SED_CHN](#) SedChn, [MI_SED_TARGET_CHN](#) TargetChn);

➤ 参数

参数名称	描述	输入/输出
SedChn	SED 通道号。 取值范围：[0, SED_MAX_CHN_NUM)。	输入
TargetChn	目标编码通道号。 取值范围：[0, VENC_MAX_CHN_NUM)。	输入

➤ 返回值

返回值 {
MI_OK 成功。
非 MI_OK 失败，参照[错误码](#)。

➤ 依赖

- 依赖头文件：mi_common.h、mi_sed.h
- 库文件：libmi_sed.a

※ 注意

-

➤ 举例

➤ 相关主题

无。

2.7. MI_SED_SetDbgLevel

➤ 描述

设置 SED Module 的 debug Level。

➤ 语法

MI_S32 MI_SED_SetDbgLevel(MI_DBG_LEVEL_e eLevel)

➤ 参数

参数名称	描述	输入/输出
eLevel	Debug Level 值。 取值范围：[0, MI_DBG_ALL]。	输入

➤ 返回值

返回值 {
MI_OK 成功。
非 MI_OK 失败，参照[错误码](#)。

- 依赖
 - 依赖头文件：mi_common.h、mi_sed.h
 - 库文件：libmi_sed.a

- ※ 注意
 -

- 举例

- 相关主题

无。

2.8. MI_SED_GetRect

- 描述

获取 SED 检测出的矩形框数据。

- 语法

MI_S32 MI_SED_GetRect([MI_SED_CHN](#) SedChn, [MI_SED_RectInfo_t](#) *pstRectInfo)

- 参数

参数名称	描述	输入/输出
SedChn	SED 通道号。 取值范围：[0, SED_MAX_CHN_NUM)。	输入
pstRectInfo	矩阵	输出

- 返回值

返回值 {

- MI_OK 成功。
- 非 MI_OK 失败，参照[错误码](#)。

- 依赖
 - 依赖头文件：mi_common.h、mi_sed.h
 - 库文件：libmi_sed.a

- ※ 注意
 -

- 举例

- 相关主题

无。

3. SED 数据类型

相关数据类型、数据结构定义如下：

SED_MAX_CHN_NUM	定义 SED 最大通道数
SED_MAX_ROI_NUM_PER_CHN	定义每路编码通道最多可设置 ROI 个数
SED_MAX_TARGET_CHN_NUM_PER_CHN	定义每个 SED 通道最多可同时作用编码通道数
SED_MAX_CUS_DEF_ALGOPARAM_NUM	定义 SED 最多支持的第三方算法数量
MI_SED_CHN	定义 SED 通道号
MI_SED_TARGET_CHN	定义 SED 通道映射对应的目标 VENC 编码通道号
MI_SED_AlgoType_e	定义 SED 算法属性
MI_SED_MdMbMode_e	定义 MD 宏块类型
MI_SED_InputAttr_t	定义 SED 输入端属性
MI_SED_Default_AlgoParam_t	定义 SDK 内置的算法参数
MI_SED_CusDef_AlgoParam_t	定义第三方算法的参数
MI_SED_AlgoAttr_t	定义 SED 算法属性
MI_SED_TargetAttr_t	定义最终生效的编码目标属性
MI_SED_DetectorAttr_t	定义 SED 检测属性

3.1. SED_MAX_CHN_NUM

➤ 说明

定义最大 SED 通道个数。

➤ 定义

```
#define SED_MAX_CHN_NUM    (4)
```

※ 注意事项

无。

➤ 相关数据类型及接口

无。

3.2. SED_MAX_ROI_NUM_PER_CHN

➤ 说明

定义每路编码通道可以最多使用 ROI 的个数。

➤ 定义

```
#define SED_MAX_ROI_NUM_PER_CHN    (16)
```

※ 注意事项

无。

➤ 相关数据类型及接口

无。

3.3. SED_MAX_TARGET_CHN_NUM_PER_CHN

➤ 说明

定义每个 SED 通道最多可同时作用编码通道数。

➤ 定义

```
#define SED_MAX_TARGET_CHN_NUM_PER_CHN    (8)
```

※ 注意事项

无。

➤ 相关数据类型及接口

无。

3.4. SED_MAX_CUS_DEF_ALGOPARAM_NUM

➤ 说明

定义 SED 最多支持的第三方算法数量。

➤ 定义

```
#define SED_MAX_CUS_DEF_ALGOPARAM_NUM    (10)
```

※ 注意事项

无。

➤ 相关数据类型及接口

无。

3.5. MI_SED_CHN

➤ 说明

定义 SED 通道号。

➤ 定义

```
typedef MI_S32 MI_SED_CHN;
```

※ 注意事项

无。

➤ 相关数据类型及接口

无。

3.6. MI_SED_TARGET_CHN

➤ 说明

定义 SED 通道映射对应的目标 VENC 编码通道号。

➤ 定义

```
typedef MI_S32 MI_SED_TARGET_CHN;
```

※ 注意事项

无。

➤ 相关数据类型及接口

无。

3.7. MI_SED_AlgoType_e

➤ 说明

定义 SED 算法类型。

➤ 定义

```
typedef enum
{
    E_MI_DEFAULT_ALGOPARAM = 0x0,
    E_MI_CUSDEF_ALGOPARAM = 0x1,
    E_MI_DEFAULT_ALGOPARAM_MAX,
} MI_SED_AlgoType_e;
```

➤ 成员

成员名称	描述
E_MI_DEFAULT_ALGOPARAM	使用 SDK 内置的算法。
E_MI_CUSDEF_ALGOPARAM	使用第三方的算法。

※ 注意事项

无。

➤ 相关数据类型及接口

无。

3.8. MI_SED_MdMbMode_e

➤ 说明

定义 MD 宏块类型。

➤ 定义

```
typedef enum
{
    E_MI_MDMB_MODE_MB_4x4 = 0x0,
    E_MI_MDMB_MODE_MB_8x8 = 0x1,
    E_MI_MDMB_MODE_MB_16x16 = 0x2,
    E_MI_MDMB_MODE_MAX
} MI_SED_MdMbMode_e;
```

➤ 成员

成员名称	描述
E_MI_MDMB_MODE_MB_4x4	使用 4×4 的宏块。
E_MI_MDMB_MODE_MB_8x8	使用 8×8 的宏块。
E_MI_MDMB_MODE_MB_16x16	使用 16×16 的宏块。

※ 注意事项
无。

➤ 相关数据类型及接口
无

3.9. MI_SED_InputAttr_t

➤ 说明
定义 SED 输入端属性。

➤ 定义

```
typedef struct MI_SED_InputAttr_s
{
    MI_U32 u32Width;
    MI_U32 u32Height;
    MI_U32 u32FrameRateNum;
    MI_U32 u32FrameRateDen;
    MI_SYS_ChnPort_t stInputPort;
} MI_SED_InputAttr_t;
```

➤ 成员

成员名称	描述
u32Width	输入 YUV 的宽
u32Height	输入 YUV 的高
u32FrameRateNum	输入 YUV 的帧率分子部分
u32FrameRateDen	输入 YUV 的帧率分母部分
stInputPort	输入 YUV 的通道属性

※ 注意事项
无。

➤ 相关数据类型及接口

MI_SED_CreateChn

MI_SED_DetectorAttr_t

3.10. MI_SED_Default_AlgoParam_t

➤ 说明
定义 SDK 内置的算法参数。

➤ 定义

```
typedef struct MI_SED_Default_AlgoParam_s
{
    MI_U32          u32VdfChn;
    MI_U8           u8Sensitivity;
    MI_SED_MdMbMode_e eMdMbMode;
} MI_SED_Default_AlgoParam_t;
```

➤ 成员

成员名称	描述
u32VdfChn	SED 内部使用的 VDF 通道号
u8Sensitivity	SED 算法灵敏度。
eMdMbMode	MD 使用的宏块类型

※ 注意事项

无。

➤ 相关数据类型及接口

MI_SED_CreateChn
MI_SED_AlgoAttr_t

3.11. MI_SED_CusDef_AlgoParam_t

➤ 说明

定义第三方算法的参数。

➤ 定义

```
typedef struct MI_SED_CusDef_AlgoParam_s
{
    MI_U32 u32ParamNum;
    MI_U32 u32CusDefAlgoParam[SED\_MAX\_CUS\_DEF\_ALGOPARAM\_NUM];
} MI_SED_CusDef_AlgoParam_t;
```

➤ 成员

成员名称	描述
u32ParamNum	有效参数数量。
u32CusDefAlgoParam	参数数组。

※ 注意事项

无。

➤ 相关数据类型及接口

MI_SED_CreateChn
MI_SED_AlgoAttr_t

3.12. MI_SED_AlgoAttr_t

➤ 说明

定义 SED 算法属性。

➤ 定义

```
typedef struct MI_SED_CusAlgoAttr_s
{
    MI_SED_AlgoType_e eType;
    union
    {
        MI_SED_Default_AlgoParam_t    stDefaultAlgoParam;
        MI_SED_CusDef_AlgoParam_t    stCusDefAlgoParam;
    };
} MI_SED_AlgoAttr_t;
```

➤ 成员

成员名称	描述
eType	SED 使用的算法类型
stDefaultAlgoParam	默认算法参数
stCusDefAlgoParam	第三方算法参数

➤ 注意事项

目前暂不支持第三方算法

➤ 相关数据类型及接口

MI_SED_CreateChn
MI_SED_DetectorAttr_t

3.13. MI_SED_TargetAttr_t

➤ 说明

定义最终生效的编码目标属性。

➤ 定义

```
typedef struct MI_SED_TargetAttr_s
{
    MI_S32 s32RltQp;
} MI_SED_TargetAttr_t;
```

➤ 成员

成员名称	描述
s32RltQp	ROI 相对 Qp 值

※ 注意事项

无。

➤ 相关数据类型及接口

MI_SED_CreateChn
MI_SED_DetectorAttr_t

3.14. MI_SED_DetectorAttr_t

➤ 说明

定义 SED 检测属性。

➤ 定义

```
typedef struct MI_SED_DetectorAttr_s
{
    MI_SED_InputAttr_t stInputAttr;
    MI_SED_AlgoAttr_t stAlgoAttr;
    MI_SED_TargetAttr_t stTargetAttr;
} MI_SED_DetectorAttr_t;
```

➤ 成员

成员名称	描述
stInputAttr	SED 输入部分的属性
stAlgoAttr	SED 算法相关的属性
stTargetAttr	SED 最终生效的目标控制属性

※ 注意事项

➤ 相关数据类型及接口

MI_SED_CreateChn

4. 错误码

视频编码 API 错误码如表 3-1 所示。

表 3-1 视频编码 API 错误码

错误代码	宏定义	描述
0x00000000	MI_SUCCESS	success
0xA01E2002	MI_ERR_SED_INVALID_CHNID	invalid channel ID
0xA01E2003	MI_ERR_SED_ILLEGAL_PARAM	at lease one parameter is illegal
0xA01E2004	MI_ERR_SED_EXIST	channel exists
0xA01E2005	MI_ERR_SED_UNEXIST	channel unexist
0xA01E2006	MI_ERR_SED_NULL_PTR	using a NULL point
0xA01E200c	MI_ERR_SED_NOMEM	failure caused by malloc memory
0xA01E2013	MI_ERR_SED_CHN_NOT_STARTED	channel not start
0xA01E2014	MI_ERR_SED_CHN_NOT_STOPPED	channel not stop
0xA01E2017	MI_ERR_SED_NOT_ENABLE	Channel not enable

5. HOW BUILD AND RUN DEMO

5.1. How to Build Sed module and demo code:

Build Demo code:

```
cd sdk/verify/mi_demo/alderaan$  
make
```

5.2. How to Get sed bin & Demo Test bin

最后生成的 mixer bin 会在如下目录下:

```
sdk\verify\mi_demo\out\demo\app\prog_sed
```

参考如下: 标红部分 mac 地址设置为你自己的分机号, ip 地址请替换为自己的测试 IP 地址

```
ifconfig eth0 hw ether 00:70:08:00:00:01; ifconfig eth0 172.19.24.178 netmask 255.255.255.0;  
route add default gw 172.19.24.254;  
mount -t nfs -o nolock 172.19.24.221:/Home/gavin.yuan/debug /mnt;  
export LD_LIBRARY_PATH=/mnt:$LD_LIBRARY_PATH  
cd /mnt  
./prog_sed
```

5.3. Sed Demo code 示例

具体参考如下:

<http://hcg04-master:9080/#/c/mstar/alkaid/sdk/+68242/>

