Linux ISP
开发指南

版本号: 1.4
发布日期: 2025.03.20

深圳市宇芯数码技术有限公司yixin_01

版本历史

版本号	日期	制/修订人	内容描述
1.0	2022.07.14	AWA1456	初始版本
1.1	2023.03.08	AWA1833	更新文档中 API 接口的描述
1.2	2023.11.17	AWA1833	修正文档格式
1.3	2024.03.15	AWA1833	增加 ISP 工作模式、ISP 性能简介、3A 统计值描述、ISP 特效接口章节
1.4	2025.03.20	AWA2073	增加 ISP522 介绍

深圳市宇芯数码技术有限公司yxy_01

版权所有 © 珠海全志科技股份有限公司。保留一切权利

目　录

版权所有 © 珠海全志科技股份有限公司。保留一切权利

版权所有 © 珠海全志科技股份有限公司。保留一切权利

版权所有 © 珠海全志科技股份有限公司。保留一切权利

版权所有 © 珠海全志科技股份有限公司。保留一切权利

插　图

深圳市宇芯数码技术有限公司yxy_01

版权所有 © 珠海全志科技股份有限公司。保留一切权利

1 前言

1.1 文档简介

ISP 模块简介，3A 开发指南，API 接口，数据结构。

1.2 目标读者

ISP 模块维护/开发人员

1.3 适用范围

产品名称	硬件版本
T153/MR153	ISP522
V853/V851/R853	ISP600
A523/A527/T527/AI985/MR527	ISP601

版权所有 © 珠海全志科技股份有限公司。保留一切权利

2 模块介绍

2.1 模块功能介绍

ISP(Image Signal Processor) 即图像处理，主要作用是对前端图像传感器输出的信号做后期处理，主要功能黑电平校正、色差校正、颜色空间校正、颜色增强、色彩降噪、串音校正、二/三维降噪、去马赛克、坏点校正、动态范围压缩、镜头阴影校正、周边去噪、色调映射、宽动态范围、自动曝光、自动白平衡等，依赖于 ISP 才能在不同的光学条件下都能较好的还原现场细节，ISP 技术在很大程度上决定了摄像机的成像质量，它可以分为独立与集成两种形式。

其所处位置如下：

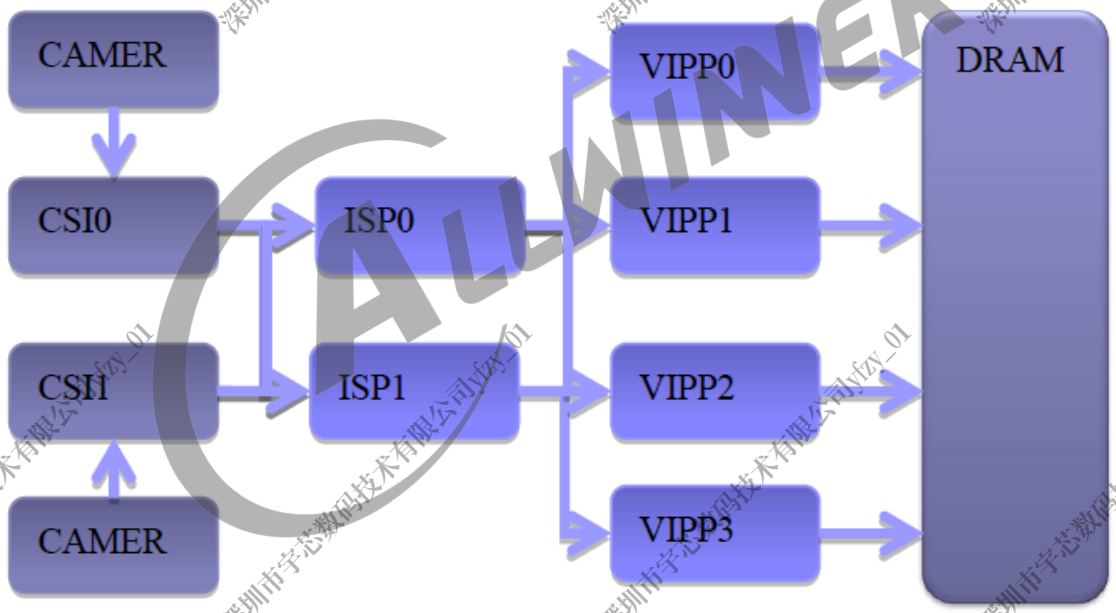

图 2-1: ISP 模块框架图

2.2 ISP 工作模式

ISP522V200、ISP600 及以上的 ISP 平台支持在线、离线两种工作模式，各个模式下数据流框架图如下图所示：

版权所有 © 珠海全志科技股份有限公司。保留一切权利

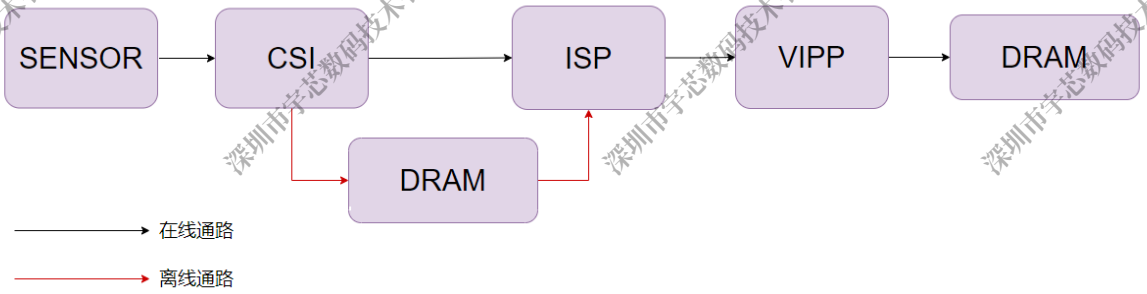

图 2-2: ISP 工作模式流程图

在线模式 (单路 CIS)：raw 数据经 CSI 后直通 ISP。

离线模式 (多路 CIS)：raw 数据经 CSI 后写到 DDR 中，ISP 从 DDR 回读 raw 数据。离线模式下，ISP 通过时分复用的方式最高支持 4 路线性 sensor 同时处理 (isp522 最高支持 2 路线性 sensor 同时处理)。

2.3 ISP 性能简介

各个 ISP 平台版本所支持最大的性能以及最大分辨率如下所示：

ISP 版本号	最大分辨率 (WxH)	最小分辨率 (WxH)	最大处理能力	数据位宽 (bit)
ISP522V200	1280x4224	320x192	1M@30fps	8/10/12
ISP600V100	3072x3072	320x192	5M@30fps	8/10/12/14/16
ISP600V200	3384x4224	320x192	8M@60fps	8/10/12/14/16/20
ISP601	3264x4224	320x192	8M@30fps	8/10/12/14/16
ISP602	3384x4224	320x192	8M@60fps	8/10/12/14/16
ISP603	1936x1936	320x192	2M@38fps	8/10/12
ISP606	2720x4224	320x192	5M@30fps	8/10/12/14/16

- 最大分辨率，最小分辨率限制为 ISP 在线模式下所有模块全开场景，关闭 pltm、D2D、D3D、MSC 模块最小支持 64x48 图像输入。离线模式下，ISP 可通过时分复用的方式处理更大分辨率的图像。
- 输入图像宽度需要 8 对齐，输入高度需要 2 对齐，支持 RGGB/BGGR/GRBG/GBRG/四种输入格式。
- ISP600 及以上的 ISP 平台支持 2fWDR 规格，ISP600V200 支持 3fWDR 规格。

2.4 ISP Server 简介

ISP Server 模块主要包括 ISP 算法库和 ISP 中间件部分：

版权所有 © 珠海全志科技股份有限公司。保留一切权利

- ISP 算法库：其主要用于在 ISP 运行时图像效果的处理，包括 3A 算法以及一系列 ISP 正常运行所需的基本算法。
- ISP 中间件：其主要用于控制 ISP 以及 Sensor 驱动、响应 Camera 应用以及 Tuning 工具命令、调度 ISP 相关算法等，包括事件管理、Pipeline 管理、Buffer 管理以及算法调度等模块。

ISP 算法库、中间件、驱动以及 Camera 应用相互关系如下图所示：

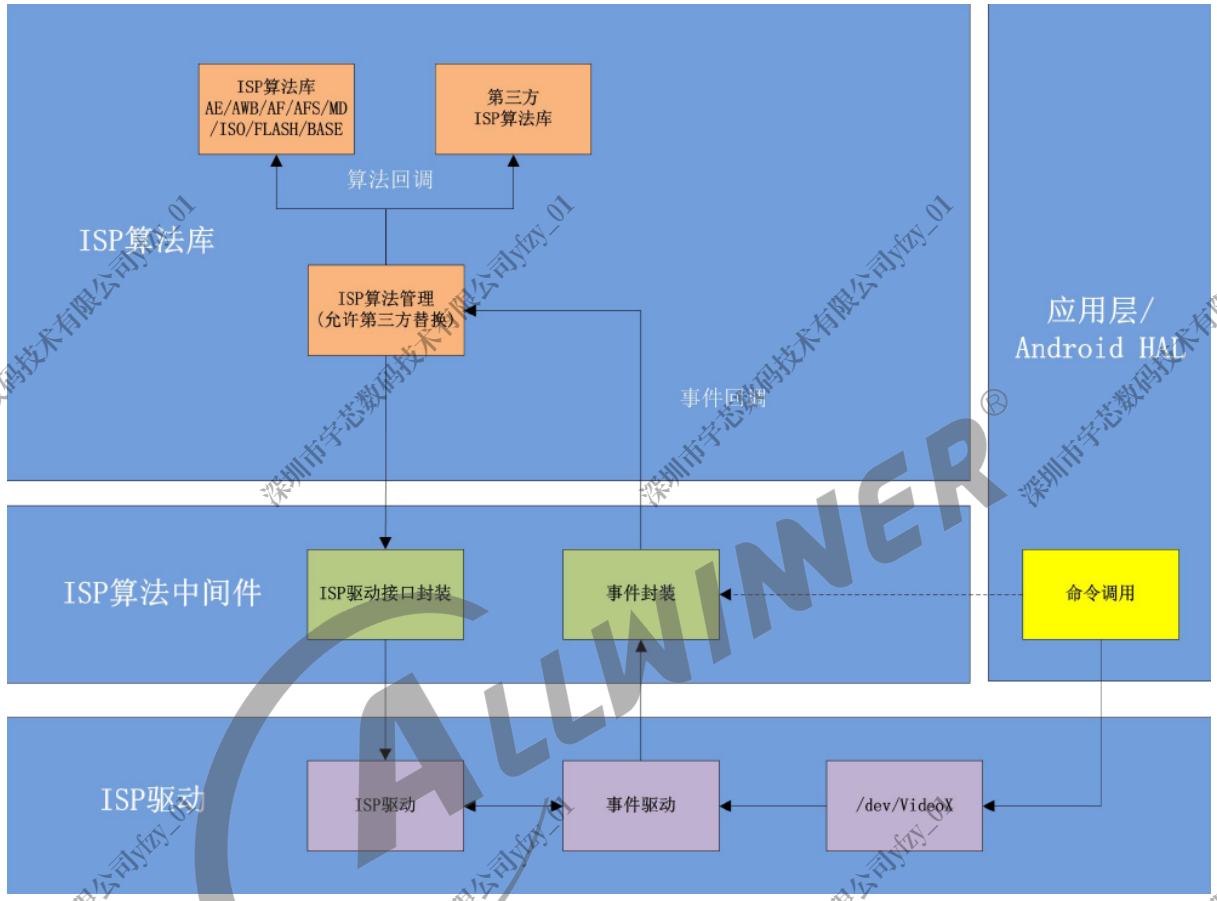

图 2-3: ISP Server 框架图

2.5 ISP Server 代码结构

ISP Server 开源部分代码结构如下：

```
lsp_server:
 | isp.c ;ISP 对外接口实现(对接调试工具、Camera应用)
 | isp.h ;ISP 对外接口头文件
 | Makefile
 ├──include
 |   isp_3a_ae.h ; 自动曝光算法头文件
 |   isp_3a_af.h ; 自动对焦算法头文件
 |   isp_3a_afs.h ; 工频检测算法头文件
 |   isp_3a_awb.h ; 自动白平衡算法头文件
 |   isp_3a_md.h ; 工频检测算法头文件
 |   isp_base.h ; 基本算法头文件
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
|   isp_cmd_intf.h ; 命令处理头文件
|   isp_comm.h ; 公共头文件
|  isp_debug.h ; debug头文件
|   isp_ini_parse.h ; ini文件解析头文件
|   isp_iso_config.h ; 动态参数设置头文件
|   isp_manage.h ; 算法管理模块头文件
|   isp_module_cfg.h ; 硬件模块头文件
|   isp_pltm.h ; 局部色调映射算法头文件
|   isp_rolloff.h ; 镜头阴影矫正算法头文件
|   isp_tone_mapping.h ; 色调映射算法头文件
|   isp_tuning.h ; ISP效果tuning接口头文件
|   isp_type.h ; 类型定义头文件
|
├──iniparser
|   ├──src
|    iniparser.c
|    iniparser.h
|    dictionary.c
|    dictionary.h
|
|  ├──isp_cfg
|    isp_ini_parse.c ; sensor 模组ISP配置文件解析
|   ├──——SENSOR_H ; sensor 模组ISP配置头文件
|     imx290_default_ini.h ; imx290 ISP效果参数配置
|     imx317_default_ini.h ; imx317 ISP效果参数配置
|     ar0238_default_ini.h ; ar0238 ISP效果参数配置
|     ov2718_wdr_ini.h ; ov2718 wdr ISP效果参数配置
|     Makefile
|
├──isp_dev
|   isp_dev.c ; ISP 设备封装，用于管理绑定相关设备
|   isp_v4l2_helper.c ; V4l2功能函数
|   isp_v4l2_helper.h ; V4l2功能函数头文件
|   media.c ; media框架帮助函数
|   media.h ; media框架帮助函数头文件
|   tools.h ; 工具文件
|  video.c ; 用于管理标准v4l2视频设备
|   Makefile
|
├──isp_events
|   events.c ; 事件管理模块, 用于监听分发驱动事件
|   events.h ; 事件管理模块头文件
|
├──isp_manage
|  isp_helper.c ; ISP算法库控制接口
|   isp_manage.c ; ISP算法管理模块
|  isp_otp_golden.h; sensor golden 模组 MSC table
|
└──isp_tuning
|   isp_tuning.c ;效果调试接口
|   isp_tuning_priv.h ;效果调试私有头文件
|
└──out
  libisp_ae.a ;自动曝光算法库
  libisp_af.a ;自动对焦算法库
  libisp_afs.a ;自动去工频算法库
  libisp_awb.a ;自动白平衡算法库
  libisp_base.a ;基础算法库
  libisp_ini.a ;ISP 配置参数获取库
  libisp_iso.a ;动态参数设置算法库
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
libisp_md.a ;运动检测算法库
libisp_gtm.a ;全局色调映射算法库
libisp_pltm.a ;局部色调映射算法库
libisp_rolloff.a ;自动color shading 矫正算法库
libisp_math.a ;自定义数学运算库
libmatrix.a ;矩阵运算库
```

ISP Server 开源代码可以概括为 3 部分：

- 设备管理部分，主要包括 isp_dev、isp_events 目录下的代码，其中 isp_dev 部分负责统一管理视频设备，CSI 设备、Sensor 设备、统计设备。在初始化时，其会建立好相应的 Sensor->CSI->ISP->Video 通路，对只需要图像数据的设备来说，仅需操作 Video 设备即可获取想要的数据。在设备运行时，isp_events 模块会根据 CSI 或者 ISP 返回来的事件来通知不同模块进行必要的事件处理，主要事件一般有 Frame Sync 事件 Vsync 事件，Video 命令事件以及统计值 Ready 事件等。
- sensor 配置管理部分，主要包括 iniparser、isp_cfg、isp_tuning 部分。

iniparser：标准 ini 文件解析库。

isp_cfg：sensor 配置文件。可通过读取 ini 文件或者头文件预定义方式获取效果参数配置。

isp_tuning：外部调整 ISP 效果提供接口。

- 算法管理部分，主要包括 isp_manage，其负责各个子算法的初始化、运行、退出等动作。

ISP Server 的构成结构如下图所示：

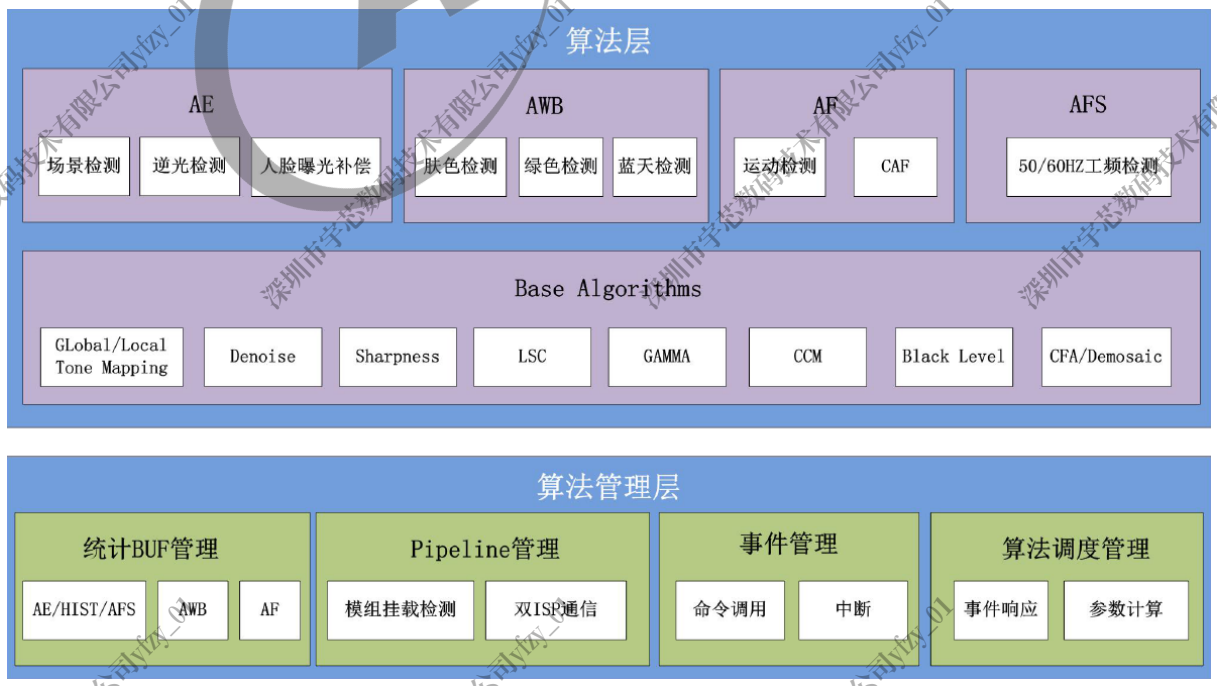

图 2-4: ISP 算法库结构图

版权所有 © 珠海全志科技股份有限公司。保留一切权利

3 使用指南

3.1 ISP 使用流程

- ISP Server 需要与 VI 采集协同工作，使用时应当先初始化 VI 采集相关配置，在初始化 ISP 相关配置，打开 VI 采集流之后运行 isp_run() 即可，ISP Server 此时会动态调整 Sensor、Lens、ISP 相关配置。
- ISP Server 使用方法非常简单，示例代码如下：

```
int main(int argc __attribute__((__unused__)), char *argv[] __attribute__((__unused__)))
{
    media_dev_init(); //初始化多媒体设备
    isp_init(0); //初始化isp0
    isp_run(0); //运行isp0 线程
    isp_pthread_join(0); //等待isp0 线程结束
    isp_exit(0); //退出isp0
    media_dev_exit(); //退出多媒体设备
    return 0;
}
```

3.2 Sensor 设备操作

sensor 与 ISP 的对应关系在一般情况下可以由内核中的 Device tree 配置，执行 media_dev_init() 时，ISP Server 中的设备管理模块便可获取其相对应关系，应用操作 Sensor 无需直接找到对应 Sensor 设备，只需要操作 ISP 接口即可，如：

```
/*isp_dev.h 中定义的Sensor 相关的操作集*/
int isp_sensor_get_configs(struct hw_isp_device *isp, struct sensor_config *cfg);
int isp_sensor_set_exp_gain(struct hw_isp_device *isp, struct sensor_exp_gain *exp_gain);
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

4 3A 开发指南

4.1 3A 开发模式

ISP Server 框架可以为客户提供三种开发模式：

- 极简模式，使用全志提供的全套算法库，ISP Server 部分对客户不可见，客户只需要通过 MPI 操作相应视频设备节点即可获取图像数据，图像效果完全由 Tuning 工具给出的配置控制。
- 一般模式，使用全志提供的全套算法库，ISP Server 部分对客户不可见，客户可以通过 MPI 操作相应视频设备节点获取图像数据，同时可以操作 ISP、Sensor 设备，通过禁用全志 ISP 内部某些特定算法，然后通过 MPI 接口获取统计信息，再通过 MPI 接口设置给 ISP，可达到替换某些特定算法的目的。
- 高级模式，对于开发能力强的核心客户，可以部分替换 ISP Server 中的算法库，ISP Server 开源部分可以开放给这类客户。

4.2 3A 高级开发

对于高阶开发者，可以使用自己开发的算法替换 ISP 算法库 out 目录下相应的算法模块，主要涉及 AE、AWB、AF 部分，ISP Server 中每个独立算法的形式基本一致，因此可以提出一个固定模板来对接具体的算法, 通用模板格式如下：

```
/*
******************************************************************************
*/
#include "../include/isp_manage.h"
#include "../include/isp_3a_xxx.h"
#include "../isp_math/isp_math_util.h"
typedef struct isp_xxx_config_entity
{
}xxx_config_t;
typedef struct isp_xxx_core_entity
{
}xxx_core_entity_t;
int __IspxxxIsr(xxx_core_entity_t *entity, xxx_stats_t *stats, xxx_result_t *result)
{
    return 0;
}
xxx_core_entity_t *__IspAllocxxxEntity(void)
{
    xxx_core_entity_t *entity = NULL;
    entity = malloc(sizeof(*entity));
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
    if(entity == NULL) {
        ISP_ERR("xxx Entity No memory!");
        return NULL;
    }
    memset(entity, 0, sizeof(*entity));
    entity->busy_flag = 1;
    return entity;
}
void __IspFreexxxEntity(xxx_core_entity_t *xxx_core_obj)
{
    if(xxx_core_obj) {
        xxx_core_obj->busy_flag = 0;
        free(xxx_core_obj);
    }
}
#define ISP_xxx_SET_PARAMS(key)\\
{\\
    entity->config.key = param->u.key;\\
}
#define ISP_xxx_GET_PARAMS(key)\\
{\\
    param->u.key = entity->config.key;\\
}
int __AwxxxSetParams(void * xxx_core_obj, xxx_param_t *param, xxx_result_t *result)
{
    int ret = 0;
    xxx_core_entity_t *entity;
    if(xxx_core_obj && param)
    {
        entity = (xxx_core_entity_t *)xxx_core_obj;
    }
    else
    {
        ret = -1;
        goto set_param_end;
    }
    switch (param->type)
    {
        case ISP_xxx_PLATFORM_ID:
            ISP_xxx_SET_PARAMS(isp_platform_id);
            break;
        case ISP_xxx_FRAME_ID:
            ISP_xxx_SET_PARAMS(xxx_frame_id);
            break;
        default:
            ret = -1;
    }
set_param_end:
    return ret;
}
int __AwxxxGetParams(void *xxx_core_obj, xxx_param_t *param)
{
    int ret = 0;
    xxx_core_entity_t *entity;
    if(xxx_core_obj && param)
    {
        entity = (xxx_core_entity_t *)xxx_core_obj;
    }
    else
    {
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
    ret = -1;
    goto get_param_end;
  }
  ISP_LIB_LOG(ISP_LOG_AF, "aw_af_get_params param->type = %d\\n", param->type);
  switch (param->type)
  {
    case ISP_xxx_PLATFORM_ID:
     ISP_xxx_GET_PARAMS(isp_platform_id);
     break;
    case ISP_xxx_FRAME_ID:
     ISP_xxx_GET_PARAMS(xxx_frame_id);
     break;
    default:
     ret = -1;
  }
get_param_end:
  return ret;
}
int __AwxxxRun(void *xxx_core_obj, xxx_stats_t *stats, xxx_result_t *result)
{
  int ret = 0;
  xxx_core_entity_t *entity;
  if(xxx_core_obj &&stats && result)
  {
    entity = (xxx_core_entity_t *)xxx_core_obj;
  }
  ret = __IspxxxIsr(entity, stats, result);
  return ret;
}
static isp_xxx_core_ops_t AwxxxOps =
{
  .isp_xxx_set_params = __AwxxxSetParams,
  .isp_xxx_get_params = __AwxxxGetParams,
  .isp_xxx_run = __AwxxxRun,
};
void __xxxInitEntity(xxx_core_entity_t *entity)
{
  entity->xxx_detect_flicker_type = 0xff;
  entity->xxx_stat_cnt = 0;
  entity->xxx_weight[0] = 1;
  entity->xxx_weight[1] = -1;
  entity->xxx_weight[2] = 0;
  return;
}
void* xxx_init(isp_xxx_core_ops_t **xxx_core_ops)
{
  xxx_core_entity_t *entity;
  entity = __IspAllocxxxEntity();
  if(entity)
  {
    __xxxInitEntity(entity);
    *xxx_core_ops = &AwxxxOps;
    return (void *)entity;
  }
  ISP_ERR("xxx_init Error!\\n");
  return NULL;
}
void xxx_exit(void *xxx_core_obj)
{
  __IspFreexxxEntity((xxx_core_entity_t *)xxx_core_obj);
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
}
```

4.3 3A 算法库注册

所有 ISP 软件算法都有一组统一的注册接口，如 AE 算法：

```
/*isp_3a_ae.h 中定义的AE算法相关的操作集*/
typedef struct isp_ae_core_ops {
    HW_S32 (*isp_ae_set_params)(void * ae_core_obj, ae_param_t *param, ae_result_t *result);
    HW_S32 (*isp_ae_get_params)(void *ae_core_obj, ae_param_t *param);
    HW_S32 (*isp_ae_run)(void *ae_core_obj, ae_stats_t *stats, ae_result_t *result);
} isp_ae_core_ops_t;
void* ae_init(isp_ae_core_ops_t **ae_core_ops);
void ae_exit(void *ae_core_obj);
```

运行算法的通用基本流程为：

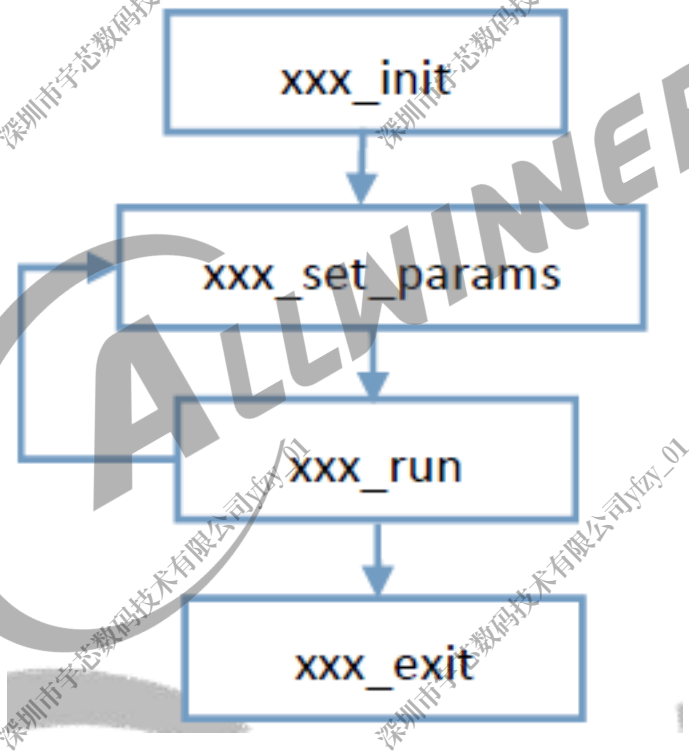

图 4-1: ISP 算法运行流程图

例如 AE 算法：

```
/***************AE init**************/
void __isp_ae_init(struct isp_lib_context *isp_gen)
{
   isp_gen->ae_entity_ctx.ae_entity = ae_init(&isp_gen->ae_entity_ctx.ops);
   if (isp_gen->ae_entity_ctx.ae_entity == NULL || NULL == isp_gen->ae_entity_ctx.ops) {
      ISP_ERR("AE Entity is BUSY or NULL!\\n");
      return -1;
   } else {
      clear(isp_gen->ae_entity_ctx.ae_param);
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
    isp_gen->ae_entity_ctx.ae_param.u.isp_platform_id = isp_gen->module_cfg.isp_platform_id;
    isp_ae_set_params_helper(ISP_AE_PLATFORM_ID);
  }
}
/***************AE set_params**************/
void __isp_ae_set_params(struct isp_lib_context *isp_gen)
{
  clear(isp_gen->ae_entity_ctx.ae_param);
  isp_gen->ae_entity_ctx.ae_param.u.ae_frame_id = isp_gen->ae_frame_cnt;
  isp_ae_set_params_helper(ISP_AE_FRAME_ID);
  clear(isp_gen->ae_entity_ctx.ae_param);
  isp_gen->ae_entity_ctx.ae_param.u.ae_setting = isp_gen->ae_settings;
  isp_ae_set_params_helper(ISP_AE_SETTINGS);
//ae_sensor_info.
  clear(isp_gen->ae_entity_ctx.ae_param);
  isp_gen->ae_entity_ctx.ae_param.u.ae_sensor_info = isp_gen->sensor_info;
  isp_ae_set_params_helper(ISP_AE_SENSOR_INFO);
}
/***************AE exit**************/
void __isp_ae_exit(struct isp_lib_context *isp_gen)
{
 ae_exit(isp_gen->ae_entity_ctx.ae_entity);
}
```

4.4 3A 统计值信息

4.4.1 统计值概述

ISP 支持对图像数据处理输出 AWB、AE、AF 控制算法需要的统计值信息，大致框图如下：

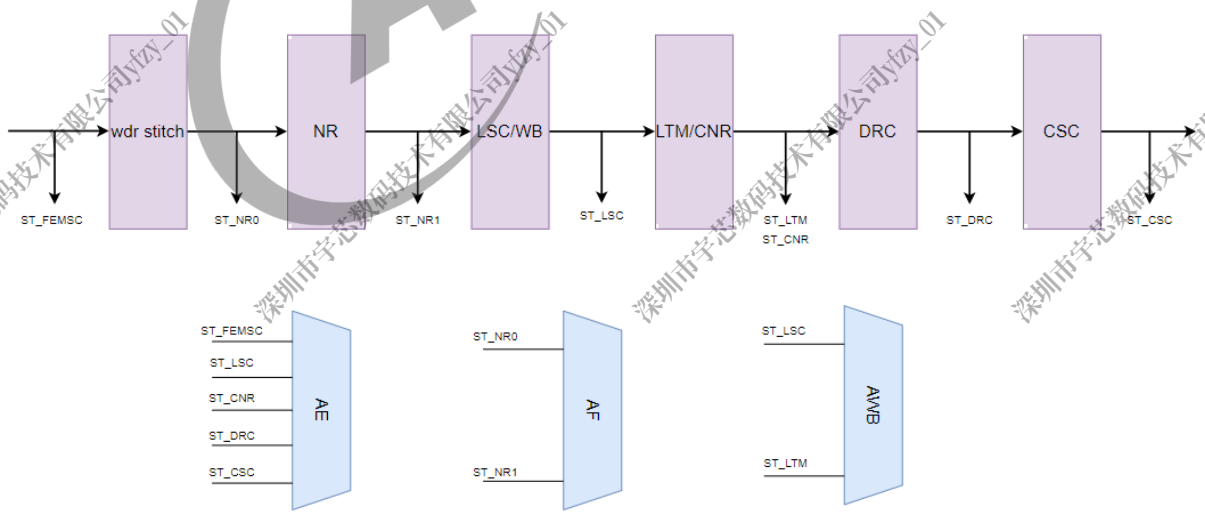

图 4-2: ISP 统计值框架图

版权所有 © 珠海全志科技股份有限公司。保留一切权利

4.4.2 AE 统计值信息

AE 统计值信息有亮度统计值和直方图两部分统计信息。AE 统计值在 pipeline 中的输出位置有 5 种模式可选择，亮度统计值与直方图信息统计值独立控制，可根据调试需求调节。各个模式具体信息如下所示：

ST_FEMSC: raw 域中经过 WDR 融合处理前统计输出

ST_LSC: raw 域中经过 LSC(镜头阴影矫正) 处理后统计输出

ST_CNR: rgb 域中经过 CNR(彩色降噪) 处理后统计输出

ST_DRC: rgb 域中经过 DRC(动态范围压缩) 处理后统计输出

ST_CSC: yuv 域中经过 CSC(颜色空间转换) 处理后输出

在对应 sensor 的效果文件中进行输出模式配置，如下所示：

```
.ae_hist0_sel = 0,
.ae_hist1_sel = 1,
.ae_stat_sel = 1,
```

直方图统计值输出模式 (ae_hist0_sel/ae_hist1_sel) 参数配置对应关系如下：

```
0: ST_LSC
1: ST_CNR
2: ST_DRC
3: ST_CSC
```

亮度统计值输出模式 (ae_stat_sel) 参数配置对应关系如下：

```
0: ST_FEMSC
1: ST_LSC
2: ST_CNR
3: ST_DRC
4: ST_CSC
```

亮度统计值：ISP600 及以上的 ISP 平台支持 24X18 窗口分块，各个分块输出 12bit 亮度均值统计值，选择 ST_CSC 统计值输出模式，亮度统计经过了 gamma 模块的处理。

直方图统计：根据分块数和对应分配的权重，进行 256 段 8bit 亮度统计，选择 ST_CSC 统计值输出模式，直方图统计经过了 gamma 模块的处理。

4.4.3 AF 统计值信息

AF 统计值在 pipeline 中的输出位置有 2 种模式可选择，模式选择暂不开放调节。各个模式具体信息如下所示：

ST_NR0: raw 域中经过 D2D(空间降噪) 处理前统计输出

ST_NR1: raw 域中经过 D2D(空间降噪) 处理后统计输出

版权所有 © 珠海全志科技股份有限公司。保留一切权利

AF 统计值输出模式配置默认配置为 ST_NR0，不开放输出模式的配置。ISP600 及以上的 ISP 平台 AF 统计值窗口横向支持 1-24 块配置，纵向支持 1-16 块配置，不开放用户配置，默认使用 24X16 窗口分块。

4.4.4 AWB 统计值信息

ISP600 及以上的 ISP 平台 AWB 统计值在 pipeline 中的输出位置有 2 种模式可选择，可根据调试需求调节。各个模式具体信息如下所示：

ST_LSC: raw 域中经过 MSC(镜头阴影矫正) 处理后统计输出

ST_LTM: rgb 域中经过 PLTM(局部色调映射) 处理后统计输出

在对应 sensor 的效果文件中进行输出模式配置，如下所示：

```
.awb_stat_sel = 1,
```

白平衡统计值输出模式 (awb_stat_sel) 参数配置对应关系如下：

```
0: ST_LSC
1: ST_LTM
```

ISP600 及以上的 ISP 平台 AWB 统计值为 32X24 窗口分块，每个窗口的数值为每个块内所有点的 R/G/B 均值。统计值输出经过了 WB Gain 的处理。

4.5 AE 算法开发

算法管理单元通过 ae_init 函数初始化 AE 算法结构体，并返回 isp_ae_core_ops_t 操作集合：

```
/*isp_3a_ae.h 中定义的AE 算法相关的操作集*/
typedef struct isp_ae_core_ops {
    HW_S32 (*isp_ae_set_params)(void * ae_core_obj, ae_param_t *param, ae_result_t *result);
    HW_S32 (*isp_ae_get_params)(void *ae_core_obj, ae_param_t *param);
    HW_S32 (*isp_ae_run)(void *ae_core_obj, ae_stats_t *stats, ae_result_t *result);
} isp_ae_core_ops_t;
void* ae_init(isp_ae_core_ops_t **ae_core_ops);
void ae_exit(void *ae_core_obj);
```

通过该操作集合，可以控制 AE 参数，调度 AE 算法，具体流程详见 3A 算法库注册部分，接口中用到主要结构体描述如下文所述。

4.5.1 isp_ae_stats_s

- PROTOTYPE

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
struct isp_ae_stats_s {
    HW_U32 win_pix_n;
    HW_U32 avg[ISP_AE_ROW*ISP_AE_COL];
    HW_U32 hist[ISP_HIST_NUM];
    HW_U32 hist1[ISP_HIST_NUM];
};
```

- MEMBERS

```
win_pix_n：每个窗口像素数
avg：分区域亮度平均值
hist：直方图统计值
hist1：直方图1统计值
```

- DESCRIPTION

```
isp_ae_stats_s：用于描述AE统计值的一个结构体。
```

4.5.2 ae_param_t

- PROTOTYPE

```
typedef struct isp_ae_param {
    ae_param_type_t type;
    HW_U32 current_frame_cnt;
    union {
        HW_S32 isp_platform_id;
        HW_S32 ae_frame_id;
        ae_ini_cfg_t ae_ini;
        isp_ae_settings_t ae_setting;
        HW_S32 ae_pline_index;
        HW_S32 sensor_update_done;
        struct isp_h3a_coor_win ae_coor;
        isp_sensor_info_t ae_sensor_info;
        ae_test_config_t test_cfg;
    } u;
} ae_param_t;
```

- MEMBERS

```
type：ISP AE命令类型，其每种类型与联合体u中参数一一对应，定义如下:
typedef enum isp_ae_param_type {
    ISP_AE_PLATFORM_ID,
    ISP_AE_FRAME_ID,
    ISP_AE_INI_DATA,
    ISP_AE_SETTINGS,
    ISP_AE_HDR_SETTINGS,
    ISP_AE_UPDATE_AE_TABLE,
    ISP_AE_SET_EXP_IDX,
    ISP_AE_BUILD_TOUCH_WEIGHT,
    ISP_AE_SENSOR_INFO,
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
    ISP_AE_TEST_CONFIG,
    ISP_AE_PARAM_TYPE_MAX,
} ae_param_type_t;
isp_platform_id：ISP平台ID
ae_frame_id：AE算法执行计数
ae_ini：AE算法初始化INI配置
ae_setting：AE算法运行时配置，包括曝光补偿、手动曝光，场景模式等
ae_pline_index：手动设置Pline索引
sensor_update_done：Sensor更新曝光设定完成
ae_coor：AE ROI窗口坐标
ae_sensor_info：Sensor信息，包括VTS，HTS以及输出宽高等
test_cfg：AE测试配置
```

- DESCRIPTION

```
ae_param_t：用于描述AE命令参数的一个结构体。
```

4.5.3 isp_ae_result

- PROTOTYPE

```
typedef struct isp_ae_result {
    enum ae_status ae_status;
    sensor_setting_t sensor_set;
    sensor_setting_t sensor_set_short;
    HW_S32 BrightPixellValue;
    HW_S32 DarkPixelValue;
    HW_U32 ae_gain;
    HW_S32 ae_target;
    HW_S32 ae_avg_lum;
    HW_S32 ae_weight_lum;
    HW_S32 ae_wdr_ratio;
    HW_S32 wdr_hi_th;
    HW_S32 wdr_low_th;
    HW_U8 backlight;
} ae_result_t;
```

- MEMBERS

```
ae_status：AE状态
sensor_set：Sensor曝光设置
sensor_set_short：Sensor短曝光设置
BrightPixellValue：AE亮像素参考值
DarkPixelValue：AE暗像素参考值
ae_gain：AE调整变化率
ae_target：AE目标亮度
ae_avg_lum：AE平均亮度
ae_weight_lum：AE加权亮度
ae_wdr_ratio：AE WD 比率
wdr_hi_th：AE WDR高阈值
wdr_low_th：AE WDR低阈值
backlight：AE背光程度
```

- DESCRIPTION

版权所有 © 珠海全志科技股份有限公司。保留一切权利

ae_result_t：用于描述AE输出结果的一个结构体。

4.6 AWB 算法开发

算法管理单元通过 aw_init 函数初始化 AWB 算法结构体，并返回 isp_awb_core_ops_t 操作集合:

```
/*isp_3a_awb.h 中定义的AWB 算法相关的操作集*/
typedef struct isp_awb_core_ops {
    HW_S32 (*isp_awb_set_params)(void * awb_core_obj, awb_param_t *param, awb_result_t *result);
    HW_S32 (*isp_awb_get_params)(void *awb_core_obj, awb_param_t *param);
    HW_S32 (*isp_awb_run)(void *awb_core_obj, awb_stats_t *stats, awb_result_t *result);
} isp_awb_core_ops_t;
void* awb_init(isp_awb_core_ops_t **awb_core_ops);
void awb_exit(void *awb_core_obj);
```

通过该操作集合，可以控制 AWB 参数，调度 AWB 算法，具体流程详见 3A 算法库注册部分，接口中用到主要结构体描述如下:

4.6.1 isp_awb_stats_s

- PROTOTYPE

```
struct isp_awb_stats_s {
    HW_U32 awb_avg_r[ISP_AWB_ROW][ISP_AWB_COL];
    HW_U32 awb_avg_g[ISP_AWB_ROW][ISP_AWB_COL];
    HW_U32 awb_avg_b[ISP_AWB_ROW][ISP_AWB_COL];
    HW_U32 avg[ISP_AWB_ROW][ISP_AWB_COL];
};
```

- MEMBERS

```
awb_avg_r：分区域R像素平均值(归一化为0~255)
awb_avg_g：分区域G像素平均值(归一化为0~255)
awb_avg_b：分区域B像素平均值(归一化为0~255)
avg: 分区域加权平均值
```

- DESCRIPTION

```
isp_awb_stats_s：用于描述AWB统计值的一个结构体。
```

4.6.2 awb_param_t

- PROTOTYPE

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
typedef struct isp_awb_param {
  awb_param_type_t type;
  HW_U32 current_frame_cnt;
  union {
    HW_S32 isp_platform_id;
    HW_S32 awb_frame_id;
    isp_awb_setting_t awb_ctrl;
    awb_ini_cfg_t awb_ini;
    isp_sensor_info_t awb_sensor_info;
    awb_test_config_t test_cfg;
  } u;
} awb_param_t;
```

- MEMBERS

```
type：ISP AWB 命令类型，其每种类型与联合体u中参数一一对应，定义如下:
typedef enum isp_awb_param_type {
  ISP_AWB_PLATFORM_ID,
  ISP_AWB_FRAME_ID,
  ISP_AWB_CTRL_CFG,
  ISP_AWB_INI_DATA,
  ISP_AWB_SENSOR_INFO,
  ISP_AWB_TEST_CONFIG,
  ISP_AWB_PARAM_TYPE_MAX,
} awb_param_type_t;
isp_platform_id：ISP平台ID
awb_frame_id：AWB算法执行计数
awb_ctrl：AWB算法运行时配置
awb_ini：AWB算法初始化INI 配置
awb_sensor_info：Sensor信息，包括VTS，HTS以及输出宽高等
test_cfg：AWB测试配置
```

- DESCRIPTION

```
awb_param_t：用于描述AWB命令参数的一个结构体。
```

4.6.3 awb_result_t

- PROTOTYPE

```
typedef struct isp_awb_result {
  struct isp_wb_gain wb_gain_output;
  HW_S32 color_temp_output;
} awb_result_t;
```

- MEMBERS

```
wb_gain_output：白平衡输出增益
color_temp_output：色温输出参考值
```

- DESCRIPTION

版权所有 © 珠海全志科技股份有限公司。保留一切权利

awb_result_t：用于描述AWB输出结果的一个结构体。

4.7 AF 算法开发

算法管理单元通过 af_init 函数初始化 AF 算法结构体，并返回 isp_af_core_ops_t 操作集合:

```
/*isp_3a_af.h 中定义的AF 算法相关的操作集*/
typedef struct isp_af_core_ops {
    HW_S32 (*isp_af_set_params)(void * af_core_obj, af_param_t *param, af_result_t *result);
    HW_S32 (*isp_af_get_params)(void *af_core_obj, af_param_t *param);
    HW_S32 (*isp_af_run)(void *af_core_obj, af_stats_t *stats, af_result_t *result);
} isp_af_core_ops_t;
void* af_init(isp_af_core_ops_t **af_core_ops);
void af_exit(void *af_core_obj);
```

通过该操作集合，可以控制 AF 参数，调度 AF 算法，具体流程详见 3A 算法库注册部分，接口中用到主要结构体描述如下文所述。

4.7.1 isp_af_stats_s

- PROTOTYPE

```
struct isp_af_stats_s {
    HW_U64 af_iir[ISP_AF_ROW][ISP_AF_COL];
    HW_U64 af_fir[ISP_AF_ROW][ISP_AF_COL];
    HW_U64 af_iir_cnt[ISP_AF_ROW][ISP_AF_COL];
    HW_U64 af_fir_cnt[ISP_AF_ROW][ISP_AF_COL];
    HW_U64 af_hlt_cnt[ISP_AF_ROW][ISP_AF_COL];

    HW_U32 af_count[ISP_AF_ROW][ISP_AF_COL];
    HW_U32 af_h_d1[ISP_AF_ROW][ISP_AF_COL];
    HW_U32 af_h_d2[ISP_AF_ROW][ISP_AF_COL];
    HW_U32 af_v_d1[ISP_AF_ROW][ISP_AF_COL];
    HW_U32 af_v_d2[ISP_AF_ROW][ISP_AF_COL];
};
```

- MEMBERS

```
af_count：锐像素计数
af_h_d1：水平AF统计值1
af_h_d2：水平AF统计值2
af_v_d1：垂直AF统计值1
af_v_d2：垂直AF统计值2
```

- DESCRIPTION

isp_af_stats_s：用于描述AF统计值的一个结构体。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

4.7.2 af_param_t

- PROTOTYPE

```
typedef struct isp_af_param {
  af_param_type_t type;
  HW_U32 current_frame_cnt;
  union{
    HW_S32 isp_platform_id;
    HW_S32 af_frame_id;
    af_ini_cfg_t af_ini;
    HW_S32 focus_absolute;
    HW_S32 focus_relative;
    enum auto_focus_run_mode af_run_mode;
    enum auto_focus_metering_mode af_metering_mode;
    enum auto_focus_range_new af_range;
    struct vcm_para vcm;
    bool focus_lock;
    isp_sensor_info_t sensor_info;
    af_test_config_t test_cfg;
    HW_S32 auto_focus_trigger;
  }u;
} af_param_t;
```

- MEMBERS

```
type：ISP AF命令类型，其每种类型与联合体u中参数一一对应，定义如下:
typedef enum isp_af_param_type {
  ISP_AF_PLATFORM_ID,
  ISP_AF_FRAME_ID,
  ISP_AF_INI_DATA,
  ISP_AF_FOCUS_ABSOLUTE,
  ISP_AF_FOCUS_RELATIVE,
  ISP_AF_RUN_MODE,
  ISP_AF_METERING_MODE,
  ISP_AF_RANGE,
  ISP_AF_VCM_PARAM,
  ISP_AF_LOCK,
  ISP_AF_SENSOR_INFO,
  ISP_AF_TEST_CONFIG,
  ISP_AF_TRIGGER,
  ISP_AF_PARAM_TYPE_MAX,
} af_param_type_t;
isp_platform_id：ISP平台ID
af_frame_id：AF算法执行计数
af_ini：AF算法初始化INI 配置
focus_absolute：对焦位置绝对值
focus_relative：对焦位置相对值
af_run_mode：AF运行模式
af_metering_mode：AF测量模式
af_range：AF范围
vcm：AF VCM配置
focus_lock：焦距锁定
sensor_info：Sensor信息，包括VTS，HTS以及输出宽高等
test_cfg：AF测试配置
auto_focus_trigger：自动对焦触发
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

- DESCRIPTION

af_param_t：用于描述AF命令参数的一个结构体。

4.7.3 af_result_t

- PROTOTYPE

```
typedef struct isp_af_result {
  enum auto_focus_status af_status_output;
  HW_U32 last_code_output;
  HW_U32 real_code_output;
  HW_U32 std_code_output;
  HW_U16 af_sap_lim_output;
  HW_U32 af_sharp_output;
} af_result_t;
```

- MEMBERS

af_status_output：AF状态输出
last_code_output：VCM上一次Code值输出(归一化到0~1024)
real_code_output：VCM实际Code值输出(对应配置文件)
std_code_output：VCM标准Code值输出(归一化到0~1024)
af_sap_lim_output：AF锐度限制输出
af_sharp_output：清晰度参考值

- DESCRIPTION

af_result_t：用于描述AF输出结果的一个结构体。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

5 API 接口

5.1 ISP 运行接口

5.1.1 media_dev_init

【目的】

初始化 Media 相关设备，如 ISP、Sensor、CSI 等。

【语法】

int media_dev_init(void);

【参数】

参数	描述
void	无

【返回值】

返回值	描述
0	成功
-1	失败

【需求】

头文件: isp.h

库文件: libisp_dev.a

【注意】

在进行 ISP 相关操作前应该首先调用此接口，保证 ISP 设备已经初始化。

【举例】

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
/*declaration*/
int ret = 0;
/* init VI device*/
ret = media_dev_init();
if (ret)
{
    return -1;
}
```

5.1.2 media_dev_exit

【目的】

退出 Media 设备。

【语法】

void media_dev_exit(void);

【参数】

参数	描述
void	无

【返回值】

返回值	描述
void	无

【需求】

头文件：isp.h

库文件：libisp_dev.a

【注意】

调用 media_dev_exit 之前要确保设备已经初始化，与 media_dev_exit 调用次数对应。

【举例】

无。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

5.1.3 isp_init

【目的】

初始化 ISP 设备。

【语法】

int isp_init(int dev_id);

【参数】

参数	描述
dev_id	ISP 设备号

【返回值】

返回值	描述
0	成功
-1	失败

【需求】

头文件: isp.h

库文件: libisp_dev.a

【注意】

初始化 ISP 设备，分配 ISP 所需内存资源，初始化 ISP 配置参数。

【举例】

无。

5.1.4 isp_exit

【目的】

退出 ISP 设备。

【语法】

int isp_exit(int dev_id);

版权所有 © 珠海全志科技股份有限公司。保留一切权利

【参数】

参数	描述
dev_id	ISP 设备号

【返回值】

返回值	描述
0	成功
-1	失败

【需求】

头文件: isp.h

库文件: libisp_dev.a

【注意】

该操作用于退出 ISP 设备工作，并释放相关资源。

【举例】

无。

5.1.5 isp_run

【目的】

运行 ISP 设备。

【语法】

int isp_run(int dev_id);

【参数】

参数	描述
dev_id	ISP 设备号

【返回值】

版权所有 © 珠海全志科技股份有限公司。保留一切权利

返回值	描述
0	成功
-1	失败

【需求】

头文件: isp.h

库文件: libisp_dev.a

【注意】

该操作用于启动 ISP 处理线程，该线程用于监听 ISP 设备中断、调度 ISP 算法、配置 ISP 参数等工作。

【举例】

无。

5.1.6 isp_pthread_join

【目的】

等待 ISP 处理线程。

【语法】

int isp_pthread_join(int dev_id);

【参数】

参数	描述
dev_id	ISP 设备号

【返回值】

返回值	描述
0	成功
-1	失败

【需求】

头文件: isp.h

版权所有 © 珠海全志科技股份有限公司。保留一切权利

库文件: libisp_dev.a

【注意】

该操作用于等待 ISP 处理线程处理结束。

【举例】

无。

5.1.7 isp_stats_req

【目的】

获取 ISP 统计值。

【语法】

int isp_stats_reqs(int dev_id, struct isp_stats_context *stats_ctx);

【参数】

参数	描述
dev_id	ISP 设备号
stats_ctx	统计值结构体

【返回值】

返回值	描述
0	成功
-1	失败

【需求】

头文件: isp.h

库文件: libisp_dev.a

【注意】

该操作用来获取 ISP 统计值信息，包括 AE、AWB、AF、PLTM 等模块统计值。

【举例】

无。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

5.1.8 isp_get_cfg

【目的】

获取 ISP 配置参数。

【语法】

int isp_get_cfg(int dev_id, unsigned char group_id, unsigned int cfg_ids, void *cfg_data);

【参数】

参数	描述
dev_id	ISP 设备号
group_id	命令组 ID
cfg_ids	配置 ID
cfg_data	配置数据结构

【返回值】

返回值	描述
0	成功
-1	失败

【需求】

头文件: isp.h

库文件: libisp_dev.a

【注意】

该操作用来获取 ISP 模块参数，通过命令组配置方式定位具体模块参数，关于命令组与配置命令后面数据结构部分会有具体介绍。

【举例】

无。

5.1.9 isp_set_cfg

【目的】

设置 ISP 配置参数。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

【语法】

int isp_set_cfg(int dev_id, unsigned char group_id, unsigned int cfg_ids, void *cfg_data);

【参数】

参数	描述
dev_id	ISP 设备号
group_id	命令组 ID
cfg_ids	配置 ID
cfg_data	配置数据结构

【返回值】

返回值	描述
0	成功
-1	失败

【需求】

头文件: isp.h

库文件: libisp_dev.a

【注意】

该操作用来设置 ISP 模块参数，通过命令组配置方式定位具体模块参数，关于命令组与配置命令后面数据结构部分会有具体介绍。

【举例】

无。

5.2 ISP 特效接口

5.2.1 使用指南

ISP 特效接口是基于 V4L2 的 s_ctrl 框架来实现的。用户可以调用 ioctl(fd, VIDIOC_S_CTRL, &ctrl) 接口来设置实现自己需要的效果，使用实例如下文描述。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

5.2.1.1 设置命令参数值

```
int v4l2_set_control(struct media_entity *entity, unsigned int id, int32_t *value)
{
    struct v4l2_control ctrl;
    int ret;
    if (entity->fd == -1)
        return -1;
    ctrl.id = id;
    ctrl.value = *value;
    ret = ioctl(entity->fd, VIDIOC_S_CTRL, &ctrl);
    if (ret < 0) {
        ISP_ERR("unable to set control: %s (%d).\n", strerror(errno), errno);
        return -errno;
    }
    *value = ctrl.value;
    return 0;
}
```

5.2.1.2 获取命令参数值

```
int v4l2_get_control(struct media_entity *entity, unsigned int id, int32_t *value)
{
    struct v4l2_control ctrl;
    int ret;
    if (entity->fd == -1)
        return -1;
    ctrl.id = id;
    ret = ioctl(entity->fd, VIDIOC_G_CTRL, &ctrl);
    if (ret < 0) {
        ISP_ERR("unable to get control: %s (%d).\n", strerror(errno), errno);
        return -errno;
    }
    *value = ctrl.value; return 0;
}
```

5.2.1.3 查询命令取值范围

```
int v4l2_query_control(struct media_entity *entity, unsigned int id, struct v4l2_queryctrl *ctrl)
{
    int ret;
    if (entity->fd == -1)
        return -1;
    ctrl->id = id;
    ret = ioctl(entity->fd, VIDIOC_QUERYCTRL, ctrl);
    if (ret < 0) {
        ISP_ERR("unable to query control: %s (%d).\n", strerror(errno), errno);
        return -errno;
    }
    return 0;
}
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

5.2.2 结构体定义

s_ctrl 相关数据结构的变量意义如下:

```
struct v4l2_control
{
    __u32 id;     //命令号
    __s32 value;   //参数值
};

struct v4l2_queryctrl
{
    __u32 id;
    __u32 type; /* enum v4l2_ctrl_type */
    __u8 name[32]; /* Whatever */
    __s32 minimum; /* Note signedness */
    __s32 maximum;
    __s32 step;
    __s32 default_value;
    __u32 flags;
    __u32 reserved[2];
};
```

5.2.3 控制命令定义

支持用户通过以下 s_ctrl 命令对 ISP 效果参数进行个性化的调整

```
#define V4L2_CTRL_CLASS_USER            0x00980000  /* Old-style 'user' controls */
#define V4L2_CID_BASE                   (V4L2_CTRL_CLASS_USER | 0x900)
#define V4L2_CID_BRIGHTNESS             (V4L2_CID_BASE+0)
#define V4L2_CID_CONTRAST               (V4L2_CID_BASE+1)
#define V4L2_CID_SATURATION             (V4L2_CID_BASE+2)
#define V4L2_CID_HUE                    (V4L2_CID_BASE+3)
#define V4L2_CID_AUTO_WHITE_BALANCE     (V4L2_CID_BASE+12)
#define V4L2_CID_EXPOSURE               (V4L2_CID_BASE+17)
#define V4L2_CID_AUTOGAIN               (V4L2_CID_BASE+18)
#define V4L2_CID_GAIN                   (V4L2_CID_BASE+19)
#define V4L2_CID_POWER_LINE_FREQUENCY   (V4L2_CID_BASE+24)
#define V4L2_CID_WHITE_BALANCE_TEMPERATURE  (V4L2_CID_BASE+26)
#define V4L2_CID_SHARPNESS              (V4L2_CID_BASE+27)
#define V4L2_CID_AUTOBRIGHTNESS         (V4L2_CID_BASE+32)
#define V4L2_CID_BAND_STOP_FILTER       (V4L2_CID_BASE+33)
#define V4L2_CID_ILLUMINATORS_1         (V4L2_CID_BASE+37)
#define V4L2_CID_ILLUMINATORS_2         (V4L2_CID_BASE+38)
#define V4L2_CTRL_CLASS_CAMERA          0x009a0000  /* Camera class controls */
#define V4L2_CID_CAMERA_CLASS_BASE      (V4L2_CTRL_CLASS_CAMERA | 0x900)
#define V4L2_CID_EXPOSURE_AUTO          (V4L2_CID_CAMERA_CLASS_BASE+1)
#define V4L2_CID_EXPOSURE_ABSOLUTE      (V4L2_CID_CAMERA_CLASS_BASE+2)
#define V4L2_CID_FOCUS_ABSOLUTE         (V4L2_CID_CAMERA_CLASS_BASE+10)
#define V4L2_CID_FOCUS_RELATIVE         (V4L2_CID_CAMERA_CLASS_BASE+11)
#define V4L2_CID_FOCUS_AUTO             (V4L2_CID_CAMERA_CLASS_BASE+12)
#define V4L2_CID_AUTO_EXPOSURE_BIAS     (V4L2_CID_CAMERA_CLASS_BASE+19)
#define V4L2_CID_AUTO_N_PRESET_WHITE_BALANCE  (V4L2_CID_CAMERA_CLASS_BASE+20)
#define V4L2_CID_ISO_SENSITIVITY        (V4L2_CID_CAMERA_CLASS_BASE+23)
#define V4L2_CID_ISO_SENSITIVITY_AUTO   (V4L2_CID_CAMERA_CLASS_BASE+24)
#define V4L2_CID_EXPOSURE_METERING      (V4L2_CID_CAMERA_CLASS_BASE+25)
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
#define V4L2_CID_SCENE_MODE             (V4L2_CID_CAMERA_CLASS_BASE+26)
#define V4L2_CID_3A_LOCK                (V4L2_CID_CAMERA_CLASS_BASE+27)
#define V4L2_CID_AUTO_FOCUS_START       (V4L2_CID_CAMERA_CLASS_BASE+28)
#define V4L2_CID_AUTO_FOCUS_STOP        (V4L2_CID_CAMERA_CLASS_BASE+29)
#define V4L2_CID_AUTO_FOCUS_RANGE       (V4L2_CID_CAMERA_CLASS_BASE+31)
```

5.2.3.1 亮度设置

设置亮度叠加值参数如下所示：

v4l2_control 命令号	V4L2_CID_BRIGHTNESS
v4l2_control 参数值	[0, 512]
默认值	256
步长	1

默认值为 256，设置参数小于 256 降低亮度，大于 256 为提高亮度。

5.2.3.2 对比度设置

设置对比度叠加值参数如下所示：

v4l2_control 命令号	V4L2_CID_CONTRAST
v4l2_control 参数值	[0, 512]
默认值	256
步长	1

默认值为 256，设置参数小于 256 降低对比度，大于 256 为提高对比度。

5.2.3.3 饱和度设置

设置饱和度叠加值参数如下所示：

v4l2_control 命令号	V4L2_CID_SATURATION
v4l2_control 参数值	[0, 512]
默认值	256
步长	1

默认值为 256，设置参数小于 256 降低饱和度，大于 256 为提高饱和度。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

5.2.3.4 色调设置

设置色调叠加值参数如下所示：

v4l2_control 命令号	V4L2_CID_HUE
v4l2_control 参数值	[-180, 180]
默认值	0
步长	1

默认值为 0。

5.2.3.5 白平衡模式设置

设置白平衡模式如下所示：

v4l2_control 命令号	V4L2_CID_AUTO_WHITE_BALANCE
v4l2_control 参数值	[0, 10)
默认值	1
步长	1

白平衡支持 10(0~9) 种模式设置，模式定义如下所示：

```
enum white_balance_mode {
  WB_MANUAL      = 0,
  WB_AUTO        = 1,
  WB_INCANDESCENT  = 2, //白炽灯 2800K
  WB_FLUORESCENT   = 3, //荧光灯 4000K
  WB_FLUORESCENT_H = 4, //荧光灯 5500K
  WB_HORIZON     = 5, //日出日落 2200K
  WB_DAYLIGHT    = 6, //日光灯 6500K
  WB_FLASH       = 7, //闪光灯 6800K
  WB_CLOUDY      = 8, //多云   7500K
  WB_SHADE       = 9, //阴天   2500K
  WB_TUNGSTEN    = 10, //钨丝灯
};
```

注意事项：该接口设置白平衡模式为 0 时需要手动设置白平衡增益，目前白平衡增益接口并未开放。

5.2.3.6 曝光模式设置

设置曝光模式如下所示：

版权所有 © 珠海全志科技股份有限公司。保留一切权利

v4l2_control 命令号	V4L2_CID_EXPOSURE_AUTO
v4l2_control 参数值	[1, 3]
默认值	0
步长	1

曝光模式支持 4(0~3) 种模式设置，模式定义如下所示：

```
enum v4l2_exposure_auto_type {
  V4L2_EXPOSURE_AUTO = 0,          //自动曝光模式
  V4L2_EXPOSURE_MANUAL = 1,        //手动曝光模式
  V4L2_EXPOSURE_SHUTTER_PRIORITY = 2, //快门优先模式
  V4L2_EXPOSURE_APERTURE_PRIORITY = 3 //光圈优先模式
};
```

5.2.3.7 曝光行设置

设置曝光行如下所示：

v4l2_control 命令号	V4L2_CID_EXPOSURE
v4l2_control 参数值	[1, 65536*16]
默认值	1
步长	1

注意事项：该接口需要在手动曝光模式或快门优先模式下使用，该接口设置的是 sensor 的曝光行，16 为 1 行，越大曝光时间越长。可通过 g_ctrl 获取 sensor 当前正在使用的曝光行（曝光行和曝光时间可以相互转化，sensor 驱动实际使用的是曝光行）。

5.2.3.8 曝光时间设置

设置曝光时间如下所示：

v4l2_control 命令号	V4L2_CID_EXPOSURE_ABSOLUTE
v4l2_control 参数值	[1, 30*1000000]
默认值	0
步长	1

注意事项：该接口需要在手动曝光或者快门优先时使用; 该接口设置的是 sensor 的曝光时间，单位为 us，越大曝光时间越长。可通过 g_ctrl 获取 sensor 当前正在使用的曝光时间（由曝光行转换得到）。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

5.2.3.9 自动增益模式设置

设置自动增益模式如下所示：

v4l2_control 命令号	V4L2_CID_AUTOGAIN
v4l2_control 参数值	[0, 1]
默认值	0
步长	1

注意事项：该接口设置为 0 时需要调用 V4L2_CID_GAIN 设置 sensor 增益，不能与 V4L2_CID_ISO_SENSITIVITY_AUTO 同时使用。

5.2.3.10 增益参数设置

设置增益参数模式如下所示：

v4l2_control 命令号	V4L2_CID_GAIN
v4l2_control 参数值	[116, 600016]
默认值	16
步长	1

注意事项：16 为一倍增益，当设置手动增益模式时需要调用此接口设置 sensor 的增益。可以通过 g_ctrl 获取 sensor 当前正在使用的增益值（该增益为 sensor 模拟增益与数字增益的乘积）。

5.2.3.11 flicker 模式设置

设置 flicker 模式如下所示：

v4l2_control 命令号	V4L2_CID_POWER_LINE_FREQUENCY
v4l2_control 参数值	[0, 3]
默认值	3
步长	1

flicker 支持 4(0~3) 种模式，模式定义如下所示：

```
enum v4l2_power_line_frequency {
    V4L2_CID_POWER_LINE_FREQUENCY_DISABLED = 0,  //关闭工频闪烁抑制
    V4L2_CID_POWER_LINE_FREQUENCY_50HZ = 1,      //50hz工频闪烁抑制
    V4L2_CID_POWER_LINE_FREQUENCY_60HZ = 2,      //60hz工频闪烁抑制
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
    V4L2_CID_POWER_LINE_FREQUENCY_AUTO = 3,    //自动工频闪烁抑制
};
```

5.2.3.12 锐化叠加强度设置

设置锐化叠加强度如下所示：

v4l2_control 命令号	V4L2_CID_SHARPNESS
v4l2_control 参数值	[1, 4095]
默认值	256
步长	1

锐化强度系数范围 [1, 4095]，默认值为 256，设置小于 256，为降低锐化强度，大于 256 提高锐化强度。

5.2.3.13 照明灯模式设置

设置照明灯如下所示：

v4l2_control 命令号	V4L2_CID_ILLUMINATORS_1
v4l2_control 参数值	0/2
默认值	0
步长	2

照明灯模式配置定义如下所示：

0：关闭照明灯

2：打开照明灯

5.2.3.14 对焦模式设置

设置对焦模式如下所示：

v4l2_control 命令号	V4L2_CID_FOCUS_AUTO
v4l2_control 参数值	[0, 1]
默认值	1
步长	1

版权所有 © 珠海全志科技股份有限公司。保留一切权利

对焦模式支持 4 种模式设置，模式定义如下所示：

```
enum auto_focus_run_mode {
    AUTO_FOCUS_MANUAL  = 0,
    AUTO_FOCUS_CONTINUEOUS = 1,
    AUTO_FOCUS_TOUCH   = 2,
    AUTO_FOCUS_SNAP    = 3,
};
```

注意事项：该接口设置为手动模式时需要调用 V4L2_CID_FOCUS_ABSOLUTE 与 V4L2_CID_FOCUS_RELATIVE 设置对焦值。

5.2.3.15 相对焦距设置

设置相对焦距如下所示：

v4l2_control 命令号	V4L2_CID_FOCUS_ABSOLUTE
v4l2_control 参数值	[0, 127]
默认值	0
步长	1

注意事项：该接口需要设置对焦为手动模式时才有效。

5.2.3.16 绝对焦距设置

设置绝对焦距如下所示：

v4l2_control 命令号	V4L2_CID_FOCUS_RELATIVE
v4l2_control 参数值	[-127, 127]
默认值	0
步长	1

注意事项：该接口需要设置对焦为手动模式时才有效。

5.2.3.17 曝光补偿设置

设置曝光补偿如下所示：

v4l2_control 命令号	V4L2_CID_AUTO_EXPOSURE_BIAS
v4l2_control 参数值	[-4, 4]

版权所有 © 珠海全志科技股份有限公司。保留一切权利

v4l2_control 命令号	V4L2_CID_AUTO_EXPOSURE_BIAS
默认值	4
步长	1

注意事项：无

5.2.3.18 ISO 模式设置

设置 ISO 模式如下所示：

v4l2_control 命令号	V4L2_CID_ISO_SENSITIVITY_AUTO
v4l2_control 参数值	[0, 1]
默认值	1
步长	1

ISO 模式支持 2 种模式设置，模式定义如下所示：

```
enum iso_mode {
  ISO_MANUAL      = 0,
  ISO_AUTO,
};
```

注意事项：该接口设置为手动模式时需要调用 V4L2_CID_ISO_SENSITIVITY 设置 ISO 值。

5.2.3.19 ISO 参数设置

设置 ISO 参数如下所示：

v4l2_control 命令号	V4L2_CID_ISO_SENSITIVITY
v4l2_control 参数值	[0, 6]
默认值	2
步长	1

ISO 支持 6 档设置，设置的参数值与挡位对应关系如下所示：

```
0: ISO100
1: ISO200
2: ISO400
3: ISO800
4: ISO1600
5: ISO3200
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
6: ISO6400
```

注意事项：该接口需要设置 ISO 为手动模式时才有效。

5.2.3.20 测光模式设置

设置测光模式如下所示：

v4l2_control 命令号	V4L2_CID_EXPOSURE_METERING
v4l2_control 参数值	[0, 3]
默认值	3
步长	1

测光模式支持 4 种模式设置，模式定义如下所示：

```
enum ae_metering_mode {
  AE_METERING_MODE_AVERAGE = 0,  //帧平均测光
  AE_METERING_MODE_CENTER = 1,  //帧中心测光
  AE_METERING_MODE_SPOT = 2,  //帧点测光
  AE_METERING_MODE_MATRIX = 3,  //帧矩阵
};
```

注意事项：AE_METERING_MODE_SPOT 需要设置测光点坐标，点坐标的。

5.2.3.21 锁定自动对焦设置

设置自动对焦锁定如下所示：

v4l2_control 命令号	V4L2_CID_AUTO_FOCUS_STOP
v4l2_control 参数值	[0, 1]
默认值	0
步长	1

该接口锁定自动对焦并固定焦距值。

注意事项：无。

5.2.3.22 启动自动对焦设置

设置自动对焦锁定如下所示：

版权所有 © 珠海全志科技股份有限公司。保留一切权利

v4l2_control 命令号	V4L2_CID_AUTO_FOCUS_START
v4l2_control 参数值	[0, 1]
默认值	0
步长	1

该接口与 V4L2_CID_AUTO_FOCUS_STOP 相对应，解锁自动对焦并触发对焦。

注意事项：无。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

6 数据结构

6.1 命令组

6.1.1 hw_isp_cfg_groups

- PROTOTYPE

```
typedef enum {
    HW_ISP_CFG_TEST = 0x01,
    HW_ISP_CFG_3A = 0x02,
    HW_ISP_CFG_TUNING = 0x03,
    HW_ISP_CFG_DYNAMIC = 0x04,
    HW_ISP_CFG_GROUP_COUNT
} hw_isp_cfg_groups;
```

- MEMBERS

```
HW_ISP_CFG_TEST：测试参数配置组
HW_ISP_CFG_3A：3A配置参数组
HW_ISP_CFG_TUNING：Tuning参数组
HW_ISP_CFG_DYNAMIC：动态参数配置组
HW_ISP_CFG_GROUP_COUNT：组计数
```

- DESCRIPTION

```
hw_isp_cfg_groups：用于枚举ISP参数命令组。
```

6.2 Test 命令组

6.2.1 hw_isp_cfg_test_ids

- PROTOTYPE

```
typedef enum {
    /* isp_test_param_cfg */
    HW_ISP_CFG_TEST_PUB = 0x00000001,
    HW_ISP_CFG_TEST_EXPTIME = 0x00000002,
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
    HW_ISP_CFG_TEST_GAIN = 0x00000004,
    HW_ISP_CFG_TEST_FOCUS = 0x00000008,
    HW_ISP_CFG_TEST_FORCED = 0x00000010,
    HW_ISP_CFG_TEST_ENABLE = 0x00000020,
} hw_isp_cfg_test_ids;
```

- MEMBERS

```
HW_ISP_CFG_TEST_PUB：测试参数公共属性
HW_ISP_CFG_TEST_EXPTIME：曝光时间测试配置
HW_ISP_CFG_TEST_GAIN：模拟增益测试配置
HW_ISP_CFG_TEST_FOCUS：对焦测试配置
HW_ISP_CFG_TEST_FORCED：强制AE测试配置
HW_ISP_CFG_TEST_ENABLE：模块使能配置
```

- DESCRIPTION

```
hw_isp_cfg_test_ids：用于枚举测试命令组中的相关测试命令。
```

6.2.2 isp_test_param_cfg

- PROTOTYPE

```
/* isp_test_param cfg */
struct isp_test_param_cfg {
    struct isp_test_pub_cfg test_pub; /* id: 0x0100000001 */
    struct isp_test_item_cfg test_exptime; /* id: 0x0100000002 */
    struct isp_test_item_cfg test_gain; /* id: 0x0100000004 */
    struct isp_test_item_cfg test_focus; /* id: 0x0100000008 */
    struct isp_test_forced_cfg test_forced; /* id: 0x0100000010 */
    struct isp_test_enable_cfg test_enable; /* id: 0x0100000020 */
};
```

- MEMBERS

```
test_pub：测试配置公共属性
test_exptime：测试曝光时间配置
test_gain：测试模拟增益配置
test_focus：测试对焦马达配置
test_forced：强制自动曝光配置
test_enable：模块使能配置
```

- DESCRIPTION

```
isp_test_param_cfg：用于设置测试模式相关配置，其与hw_isp_cfg_test_ids中所定义命令相对应。
```

6.2.3 isp_test_pub_cfg

- PROTOTYPE

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
struct isp_test_pub_cfg {
    HW_S32 test_mode;
    HW_S32 gain;
    HW_S32 exp_line;
    HW_S32 color_temp;
    HW_S32 log_param;
};
```

- MEMBERS

```
test_mode：测试模式；无效
gain：模拟增益，256代表1倍增益；range：16~65535
exp_line：曝光行数，16代表1行曝光时间；range：16~65535
color_temp：当前测试光源色温；range：1900K~13000K
log_param：ISP Debug 掩码，其每个bit 意义如下:
#define ISP_LOG_AE          (1 << 0)   //0x1
#define ISP_LOG_AWB         (1 << 1)   //0x2
#define ISP_LOG_AF          (1 << 2)   //0x4
#define ISP_LOG_ISO         (1 << 3)   //0x8
#define ISP_LOG_GAMMA       (1 << 4)   //0x10
#define ISP_LOG_COLOR_MATRIX  (1 << 5)   //0x20
#define ISP_LOG_AFS         (1 << 6)   //0x40
#define ISP_LOG_MOTION_DETECT  (1 << 7)   //0x80
#define ISP_LOG_GAIN_OFFSET   (1 << 8)   //0x100
#define ISP_LOG_DEFOG       (1 << 9)   //0x200
#define ISP_LOG_LSC         (1 << 10)  //0x400
#define ISP_LOG_GTM         (1 << 11)  //0x800
#define ISP_LOG_PLTM        (1 << 12)  //0x1000
#define ISP_LOG_SUBDEV      (1 << 13)  //0x2000
#define ISP_LOG_CFG         (1 << 14)  //0x4000
#define ISP_LOG_VIDEO       (1 << 15)  //0x8000
#define ISP_LOG_ISP         (1 << 16)  //0x10000
#define ISP_LOG_FLASH       (1 << 17)  //0x20000
```

- DESCRIPTION

```
isp_test_pub_cfg：用于描述测试参数公共属性，与HW_ISP_CFG_TEST_PUB命令相对应。
```

6.2.4 isp_test_item_cfg

- PROTOTYPE

```
struct isp_test_item_cfg {
    HW_S32 enable;
    HW_S32 start;
    HW_S32 step;
    HW_S32 end;
    HW_S32 change_interval;
};
```

- MEMBERS

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
enable：使能测试项；range：0~1
start：测试起始值；range：0~65535
step：测试步进值；range：0~65535
end ：测试结束值；range：0~65535
change_interval：测试间隔帧数；range：0~1024
```

- DESCRIPTION

```
isp_test_item_cfg：用于描述测试项目具体参数，与HW_ISP_CFG_TEST_EXPTIME、HW_ISP_CFG_TEST_GAIN、
    HW_ISP_CFG_TEST_FOCUS配置命令相对应。
```

6.2.5 isp_test_forced_cfg

- PROTOTYPE

```
struct isp_test_forced_cfg {
  HW_S32 ae_enable;
  HW_S32 lum;
};
```

- MEMBERS

```
ae_enable：强制AE使能；range：0~1
lum：强制AE目标亮度；range：0~255
```

- DESCRIPTION

```
isp_test_forced_cfg：用于配置强制AE测试模式，与HW_ISP_CFG_TEST_FORCED配置命令相对应。
```

6.2.6 isp_test_enable_cfg

- PROTOTYPE

```
struct isp_test_enable_cfg {
  HW_S8 manual;
  HW_S8 afs;
  HW_S8 ae;
  HW_S8 af;
  HW_S8 awb;
  HW_S8 hist;
  HW_S8 wdr_split;
  HW_S8 wdr_stitch;
  HW_S8 otf_dpc;
  HW_S8 ctc;
  HW_S8 gca;
  HW_S8 nrp;
  HW_S8 denoise;
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
    HW_S8 tdf;
    HW_S8 blc;
    HW_S8 wb;
    HW_S8 dig_gain;
    HW_S8 lsc;
    HW_S8 msc;
    HW_S8 pltm;
    HW_S8 cfa;
    HW_S8 lca;
    HW_S8 sharp;
    HW_S8 ccm;
    HW_S8 defog;
    HW_S8 cnr;
    HW_S8 drc;
    HW_S8 gtm;
    HW_S8 gamma;
    HW_S8 cem;
    HW_S8 encpp_en;
    HW_S8 enc_3dnr_en;
    HW_S8 enc_2dnr_en;
};
```

- MEMBERS

```
manual : 手动模块开关使能
afs  : 自动工频检测使能
ae   : 自动曝光算法使能
af   : 自动对焦算法使能
awb  : 自动白平衡算法使能
hist : 自动曝光算法直方图均衡使能
wdr_split: : 宽动态拆分使能
wdr_stitch : 宽动态合并使能
otf_dpc  : 坏点去除使能
ctc   : 迷宫纹理去除使能
gca   : 色差校正使能
nrp   : 降噪周边处理使能
denoise : 2D降噪使能
tdf   : 3D降噪使能
blc   : 黑电平校正使能
wb    : 白平衡校正使能
dig_gain : 数字增益使能
lsc   : 镜头阴影校正使能
msc   : 镜头阴影校正使能
pltm  : 局部色调映射使能
cfa   : cfa插值使能
lca   : 色差校正使能
sharp  : 锐化使能
ccm   : 色彩校正使能
defog  : 去雾算法使能
cnr   : 彩色降噪使能
drc   : 自适应直方图均衡使能
gtm   : 全局色调映射使能
gamma  : gamma 校正使能
cem   : 色彩增强使能
```

- DESCRIPTION

版权所有 © 珠海全志科技股份有限公司。保留一切权利

isp_test_enable_cfg：用于使能具体模块功能的开关，与HW_ISP_CFG_TEST_ENABLE配置命令相对应。

6.3 3A 命令组

6.3.1 hw_isp_cfg_3a_ids

- PROTOTYPE

```
typedef enum {
  /* isp_3a_param_cfg */
  /* ae */
  HW_ISP_CFG_AE_PUB          = 0x00000001,
  HW_ISP_CFG_AE_PREVIEW_TBL  = 0x00000002,
  HW_ISP_CFG_AE_CAPTURE_TBL  = 0x00000004,
  HW_ISP_CFG_AE_VIDEO_TBL    = 0x00000008,
  HW_ISP_CFG_AE_WIN_WEIGHT   = 0x00000010,
  HW_ISP_CFG_AE_DELAY        = 0x00000020,
  /* awb */
  HW_ISP_CFG_AWB_SPEED       = 0x00000040,
  HW_ISP_CFG_AWB_TEMP_RANGE  = 0x00000080,
  HW_ISP_CFG_AWB_DIST        = 0x00000100,
  HW_ISP_CFG_AWB_LIGHT_INFO  = 0x00000200,
  HW_ISP_CFG_AWB_EXT_LIGHT_INFO = 0x00000400,
  HW_ISP_CFG_AWB_SKIN_INFO   = 0x00000800,
  HW_ISP_CFG_AWB_SPECIAL_INFO  = 0x00001000,
  HW_ISP_CFG_AWB_PRESET_GAIN   = 0x00002000,
  HW_ISP_CFG_AWB_FAVOR       = 0x00004000,
  /* af */
  HW_ISP_CFG_AF_VCM_CODE     = 0x00008000,
  HW_ISP_CFG_AF_OTP          = 0x00010000,
  HW_ISP_CFG_AF_SPEED        = 0x00020000,
  HW_ISP_CFG_AF_FINE_SEARCH  = 0x00040000,
  HW_ISP_CFG_AF_REFOCUS      = 0x00080000,
  HW_ISP_CFG_AF_TOLERANCE    = 0x00100000,
  HW_ISP_CFG_AF_SCENE        = 0x00200000,
  /* wdr */
  HW_ISP_CFG_WDR_SPLIT       = 0x00400000,
  HW_ISP_CFG_WDR_STITCH      = 0x00800000,
} hw_isp_cfg_3a_ids
```

- MEMBERS

```
/* ae */
HW_ISP_CFG_AE_PUB     : AE公共属性
HW_ISP_CFG_AE_PREVIEW_TBL: AE预览曝光表
HW_ISP_CFG_AE_CAPTURE_TBL: AE拍照曝光表
HW_ISP_CFG_AE_VIDEO_TBL : AE视频曝光表
HW_ISP_CFG_AE_WIN_WEIGHT : AE窗口权重
HW_ISP_CFG_AE_DELAY    : AE曝光和增益延时配置
/* awb */
HW_ISP_CFG_AWB_SPEED      : AWB速度控制
HW_ISP_CFG_AWB_TEMP_RANGE   : AWB色温矫正范围
HW_ISP_CFG_AWB_DIST       : AWB特殊场景距离配置
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
HW_ISP_CFG_AWB_LIGHT_INFO  : AWB参考光源配置
HW_ISP_CFG_AWB_EXT_LIGHT_INFO: AWB扩展光源配置
HW_ISP_CFG_AWB_SKIN_INFO   : AWB肤色配置
HW_ISP_CFG_AWB_SPECIAL_INFO : AWB特殊光源配置
HW_ISP_CFG_AWB_PRESET_GAIN  : AWB预设增益配置
HW_ISP_CFG_AWB_FAVOR       : AWB整体偏色配置
/* af */
HW_ISP_CFG_AF_VCM_CODE : AF马达近端远端配置
HW_ISP_CFG_AF_OTP    : AF OTP 使能配置
HW_ISP_CFG_AF_SPEED   : AF速度配置
HW_ISP_CFG_AF_FINE_SEARCH: AF细搜索配置
HW_ISP_CFG_AF_REFOCUS : AF重新对焦判定配置
HW_ISP_CFG_AF_TOLERANCE: AF重对焦容忍度配置
HW_ISP_CFG_AF_SCENE   : AF场景配置
/* wdr */
HW_ISP_CFG_WDR_SPLIT  : WDR拆分模式配置
HW_ISP_CFG_WDR_STITCH : WDR合并模式配置
```

- DESCRIPTION

hw_isp_cfg_3a_ids：列出了3A相关的配置命令，这些配置一般为不随曝光变化的配置。

6.3.2 isp_3a_param_cfg

- PROTOTYPE

```
/* isp_3a_param cfg*/
struct isp_3a_param_cfg {
  /* ae */
  struct isp_ae_pub_cfg     ae_pub;          /* id: 0x0200000001 */
  struct isp_ae_table_cfg   ae_preview_tbl;    /* id: 0x0200000002 */
  struct isp_ae_table_cfg   ae_capture_tbl;    /* id: 0x0200000004 */
  struct isp_ae_table_cfg   ae_video_tbl;      /* id: 0x0200000008 */
  struct isp_ae_weight_cfg  ae_win_weight;     /* id: 0x0200000010 */
  struct isp_ae_delay_cfg   ae_delay;          /* id: 0x0200000020 */
  /* awb */
  struct isp_awb_speed_cfg      awb_speed;      /* id: 0x0200000040 */
  struct isp_awb_temp_range_cfg awb_temp_range;  /* id: 0x0200000080 */
  struct isp_awb_dist_cfg       awb_dist;       /* id: 0x0200000100 */
  struct isp_awb_temp_info_cfg  awb_light_info;  /* id: 0x0200000200 */
  struct isp_awb_temp_info_cfg  awb_ext_light_info; /* id: 0x0200000400 */
  struct isp_awb_temp_info_cfg  awb_skin_info;   /* id: 0x0200000800 */
  struct isp_awb_temp_info_cfg  awb_special_info; /* id: 0x0200001000 */
  struct isp_awb_preset_gain_cfg awb_preset_gain; /* id: 0x0200002000 */
  struct isp_awb_favor_cfg      awb_favor;      /* id: 0x0200004000 */
  /* af */
  struct isp_af_vcm_code_cfg    af_vcm_code;    /* id: 0x0200008000 */
  struct isp_af_otp_cfg         af_otp;         /* id: 0x0200010000 */
  struct isp_af_speed_cfg       af_speed;       /* id: 0x0200020000 */
  struct isp_af_fine_search_cfg af_fine_search;  /* id: 0x0200040000 */
  struct isp_af_refocus_cfg     af_refocus;     /* id: 0x0200080000 */
  struct isp_af_tolerance_cfg   af_tolerance;   /* id: 0x0200100000 */
  struct isp_af_scene_cfg       af_scene;       /* id: 0x0200200000 */
  /* wdr */
  struct isp_wdr_split_cfg      wdr_split;      /* id: 0x0200400000 */
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
    struct isp_wdr_comm_cfg       wdr_comm;       /* id: 0x0200800000 */
};
```

- MEMBERS

```
ae_pub      : AE公共属性
ae_preview_tbl: AE预览曝光表
ae_capture_tbl: AE拍照曝光表
ae_video_tbl  : AE视频曝光表
ae_win_weight : AE窗口权重
ae_delay    : AE曝光和增益延时配置

awb_speed       : AWB速度控制
awb_temp_range   : AWB色温矫正范围
awb_dist        : AWB特殊场景距离配置
awb_light_info   : AWB参考光源配置
awb_ext_light_info: AWB扩展光源配置
awb_skin_info    : AWB肤色配置
awb_special_info : AWB特殊光源配置
awb_preset_gain  : AWB预设增益配置
awb_favor       : AWB整体偏色配置

af_vcm_code  : AF马达近端远端配置
af_otp      : AF OTP 使能配置
af_speed    : AF速度配置
af_fine_search: AF细搜索配置
af_refocus   : AF重新对焦判定配置
af_tolerance : AF重对焦容忍度配置
af_scene    : AF场景配置

wdr_split;   : WDR 拆分模式配置
wdr_comm;    : WDR 公共配置
```

- DESCRIPTION

```
isp_3a_param_cfg：用于描述3A相关配置参数，其与hw_isp_cfg_3a_ids中定义的命令相对应。
```

6.3.3 AE 控制命令

6.3.3.1 isp_ae_pub_cfg

- PROTOTYPE

```
struct isp_ae_pub_cfg {
    HW_S32 define_table;
    HW_S32 max_lv;
    HW_S32 hist_mode_en;
    HW_S32 compensation_step;
    HW_S32 touch_dist_index;
    HW_S32 iso2gain_ratio;
    HW_S32 fno_table[16];
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
};
```

- MEMBERS

```
define_table：是否使用自定义AE table
max_lv：AE Table 第一项所对应的亮度值LV；range：0~2500，default：1600
hist_mode_en：是否启用直方图统计来计算AE；range：0~1，default：1
compensation_step：AE曝光补偿所跳表的步数，25为1倍；range：0~32，default：7
touch_dist_index：点对焦模式，相关点大小；range：0~5，default：2
iso2gain_ratio：ISO到Gain的转化系数，如果一倍增益对应ISO50，则该值为32，iso2gain_ratio = 1(倍)*16*100/50。
fno_table：光圈查找表
```

- DESCRIPTION

```
isp_ae_pub_cfg：用于描述AE的公共属性。
```

6.3.3.2 isp_ae_table_cfg

- PROTOTYPE

```
struct isp_ae_table_cfg {
  HW_S32 length;
  struct ae_table value[7];
};
```

- MEMBERS

```
length：每个表中的节点个数；range：1~7
value：Ae table中每个节点的描述，包括最大最小曝光，最大最小增益，最大最小光圈；range：0~65535，default：无
```

- DESCRIPTION

```
isp_ae_table_cfg：用于描述AE调整表的一个结构体。
```

6.3.3.3 isp_ae_weight_cfg

- PROTOTYPE

```
struct isp_ae_weight_cfg {
  HW_S32 weight[64];
};
```

- MEMBERS

版权所有 © 珠海全志科技股份有限公司。保留一切权利

weight：AE 窗口权重，分8x8个窗口，内部插值到48x32；range：0~128，default：无

- DESCRIPTION

isp_ae_weight_cfg：用于描述AE窗口权重的一个结构体。

6.3.3.4 isp_ae_delay_cfg

- PROTOTYPE

```
struct isp_ae_delay_cfg {
    HW_S32 ae_frame;
    HW_S32 exp_frame;
    HW_S32 gain_frame;
};
```

- MEMBERS

ae_frame：AE调整延时帧数；range：0~64，default：0
exp_frame：曝光生效延时帧数；range：1~2，default：2
gain_frame：增益生效延时帧数；range：1~2，default：2

- DESCRIPTION

isp_ae_delay_cfg：用于描述AE延时设置的一个结构体。

6.3.4 AWB 控制命令

6.3.4.1 isp_awb_speed_cfg

- PROTOTYPE

```
struct isp_awb_speed_cfg {
    HW_S32 interval;
    HW_S32 value;
};
```

- MEMBERS

interval：AWB调整间隔帧数；range：1~64，default：1
value：AWB调整速度，数字越大速度越慢；range：1~47，default：16

- DESCRIPTION

版权所有 © 珠海全志科技股份有限公司。保留一切权利

isp_awb_speed_cfg：用于描述AWB速度设置的一个结构体。

6.3.4.2 isp_awb_temp_range_cfg

- PROTOTYPE

```
struct isp_awb_temp_range_cfg {
  HW_S32 low;
  HW_S32 high;
  HW_S32 base;
};
```

- MEMBERS

low：AWB 校正最低色温；range：1000~13000，default：1900
high：AWB 校正最高色温；range：1000~13000，default：8500
base：AWB 校正基准色温；range：1000~13000，default：5500

- DESCRIPTION

isp_awb_temp_range_cfg：用于描述AWB色温限制的一个结构体。

6.3.4.3 isp_awb_dist_cfg

- PROTOTYPE

```
struct isp_awb_dist_cfg {
  HW_S32 green_zone;
  HW_S32 blue_sky;
};
```

- MEMBERS

green_zone：绿区距离限定，一般设置60~90 之间；range：0~256，default：75
blue_sky：蓝天距离限定，一般设置60~90 之间；range：0~256，default：75

- DESCRIPTION

isp_awb_dist_cfg：用于描述AWB特殊场景配置的一个结构体；

6.3.4.4 isp_awb_temp_info_cfg

- PROTOTYPE

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
struct isp_awb_temp_info_cfg {
    HW_S32 number;
    HW_S32 value[320];
};
```

- MEMBERS

```
number: 参考光源数量，不超过32个。
value: 参考光源描述。
```

图 6-1: AWB 参数图

- DESCRIPTION

```
isp_awb_temp_info_cfg：用于描述AWB参考光源信息的一个结构体。
```

6.3.4.5 isp_awb_preset_gain_cfg

- PROTOTYPE

```
struct isp_awb_preset_gain_cfg{
    HW_S32 value[22];
};
```

- MEMBERS

```
value：对应11种白平衡模式，每个模式需要两个值，前两种可填为256。
enum white_balance_mode
{
    WB_MANUAL      = 0,
    WB_AUTO        = 1,
    WB_INCANDESCENT  = 2,
    WB_FLUORESCENT   = 3,
    WB_FLUORESCENT_H = 4,
    WB_HORIZON     = 5,
    WB_DAYLIGHT    = 6,
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
  WB_FLASH      = 7,
  WB_CLOUDY     = 8,
  WB_SHADE      = 9,
  WB_TUNGSTEN   = 10,
};
```

- DESCRIPTION

```
isp_awb_preset_gain_cfg：用于描述AWB配置手动白平衡的的一个结构体。
```

6.3.4.6 isp_awb_favor_cfg

- PROTOTYPE

```
struct isp_awb_favor_cfg {
  HW_S32 rgain;
  HW_S32 bgain;
};
```

- MEMBERS

```
rgain：红色通道增益；range：0~4095，default：256
bgain：蓝色通道增益；range：0~4095，default：256
```

- DESCRIPTION

```
isp_awb_favor_cfg：用于描述AWB偏好配置的一个结构体。
```

6.3.5 AF 控制命令

6.3.5.1 isp_af_vcm_code_cfg

- PROTOTYPE

```
struct isp_af_vcm_code_cfg {
  HW_S32 min;
  HW_S32 max;
};
```

- MEMBERS

```
min：无限远端VCM code值，一般对准5 m；range：0~1023，default：100
max：微距端VCM code值，一般对准7~10 cm；range：0~1023，default：600
```

- DESCRIPTION

版权所有 © 珠海全志科技股份有限公司。保留一切权利

isp_af_vcm_code_cfg：用于描述AF VCM马达最小最大配置的一个结构体。

6.3.5.2 isp_af_otp_cfg

- PROTOTYPE

```
struct isp_af_otp_cfg {
  HW_S32 use_otp;
};
```

- MEMBERS

use_otp：是否使用OTP配置，目前该选项无效；range：0~1

- DESCRIPTION

isp_af_otp_cfg：用于配置AF算法是否使用OTP校准结构体。

6.3.5.3 isp_af_speed_cfg

- PROTOTYPE

```
struct isp_af_speed_cfg {
  HW_S32 interval_time;
  HW_S32 index;
};
```

- MEMBERS

interval_time：AF调整每步间隔时间单位为ms；range：0~1000，default：100
index：AF调整步长的大小；range：0~28，default：24

- DESCRIPTION

isp_af_speed_cfg：用于控制AF速度的一个结构体。

6.3.5.4 isp_af_fine_search_cfg

- PROTOTYPE

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
struct isp_af_fine_search_cfg {
    HW_S32 auto_en;
    HW_S32 single_en;
    HW_S32 step;
};
```

- MEMBERS

```
auto_en：连续自动对焦细搜索开关；range：0~1，default：0
single_en：单次对焦细搜索开关；range：0~1，default：1
step：细搜索步长，该值越小对焦精度越高，但对焦失败概率会上升；range：0~64，default：10
```

- DESCRIPTION

```
isp_af_fine_search_cfg：用于控制AF细搜索配置的一个结构体。
```

6.3.5.5 isp_af_refocus_cfg

- PROTOTYPE

```
struct isp_af_refocus_cfg {
    HW_S32 move_cnt;
    HW_S32 still_cnt;
    HW_S32 move_monitor_cnt;
    HW_S32 still_monitor_cnt;
};
```

- MEMBERS

```
move_cnt：运动帧计数阈值；range：0~64，default：4
still_cnt：静止帧计数阈值；range：0~64，default：2
move_monitor_cnt：运动帧检测数；range：0~64，default：6
still_monitor_cnt：静止帧检测数；range：0~64，default：3
```

- DESCRIPTION

```
isp_af_refocus_cfg：用于控制CAF重新对焦策略配置的一个结构体，当视频连续move_monitor_cnt帧出现多于move_cnt帧
    的运动帧，且之后连续still_monitor_cnt帧出现多于still_cnt静止帧时，则判定重新执行自动对焦。
```

6.3.5.6 isp_af_tolerance_cfg

- PROTOTYPE

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
struct isp_af_tolerance_cfg {
    HW_S32 near_distance;
    HW_S32 far_distance;
    HW_S32 offset;
    HW_S32 table_length;
    HW_S32 std_code_table[20];
    HW_S32 value[20];
};
```

- MEMBERS

```
near_distance：无限远端对焦检测灵敏度；range：0~32，default：8
far_distance：微距端灵敏度；range：0~32，default：10
offset：该值一般设置为0；range：0~32，default：0
table_length：灵敏度表长度；range：0~20，default：0
std_code_table：VCM code表；
value：以上VCM code表对应的期望灵敏度；range：0~32，default：8
```

- DESCRIPTION

```
isp_af_tolerance_cfg：用于控制CAF重新对焦容忍度配置的一个结构体，table_length、std_code_table、value属于细调节
    模式，当table_length=0时，使用near_distance、far_distance、offset所描述容忍度配置，当table_length>0，则需要
    配置std_code_table，这时会使用容忍度细调模式。
```

6.3.5.7 isp_af_scene_cfg

- PROTOTYPE

```
struct isp_af_scene_cfg {
    HW_S32 stable_min;
    HW_S32 stable_max;
    HW_S32 low_light_lv;
    HW_S32 peak_thres;
    HW_S32 direction_thres;
    HW_S32 change_ratio;
    HW_S32 move_minus;
    HW_S32 still_minus;
    HW_S32 scene_motion_thres;
};
```

- MEMBERS

```
stable_min：曝光稳定判定范围最小值，曝光不变为256，曝光减小1倍则为128；range：0~512，default：245
stable_max：曝光稳定判定范围最大值，曝光不变为256，曝光增加1倍则为512；range：0~512，default：265
low_light_lv：低照度判定条件，LV<low_light_lv 则判定为低照度；range：0~1000，default：200
peak_thres：峰值判定阈值；range：0~1023，default：100
direction_thres：方向判定阈值；range：0~1023，default：10
change_ratio：FV 变化阈值，当FV 变化比例超过该值，则判定为运动;range：0~100，default：20
move_minus：该值一般设置为0；range：0~10，default：0
still_minus：该值一般设置为0；range：0~10，default：0
scene_motion_thres：运动检测阈值；range：0~100，default：10
```

- DESCRIPTION

版权所有 © 珠海全志科技股份有限公司。保留一切权利

isp_af_scene_cfg：为AF判定场景运动所需的参数结构体。

6.4 Tuning 命令组

6.4.1 hw_isp_cfg_tuning_ids

- PROTOTYPE

```
typedef enum {
  /* isp_tuning_param_cfg */
  HW_ISP_CFG_TUNING_FLASH          = 0x00000001,
  HW_ISP_CFG_TUNING_FLICKER        = 0x00000002,
  HW_ISP_CFG_TUNING_VISUAL_ANGLE   = 0x00000004,
  HW_ISP_CFG_TUNING_GTM            = 0x00000008,
  HW_ISP_CFG_TUNING_CFA            = 0x00000010,
  HW_ISP_CFG_TUNING_CTC            = 0x00000020,
  HW_ISP_CFG_TUNING_DIGITAL_GAIN   = 0x00000040,
  HW_ISP_CFG_TUNING_CCM_LOW        = 0x00000080,
  HW_ISP_CFG_TUNING_CCM_MID        = 0x00000100,
  HW_ISP_CFG_TUNING_CCM_HIGH       = 0x00000200,
  HW_ISP_CFG_TUNING_PLTM           = 0x00000400,
  HW_ISP_CFG_TUNING_GCA            = 0x00000800,
  HW_ISP_CFG_TUNING_BDNF_COMM      = 0x00001000,
  HW_ISP_CFG_TUNING_TDNF_COMM      = 0x00002000,
  HW_ISP_CFG_TUNING_SHARP_COMM     = 0x00004000,
  HW_ISP_CFG_TUNING_DPC            = 0x00008000,
  HW_ISP_CFG_TUNING_ENCPP_SHARP_COMM = 0x00010000,
  HW_ISP_CFG_TUNING_SENSOR         = 0x00020000,
} hw_isp_cfg_tuning_ids;
```

- MEMBERS

```
HW_ISP_CFG_TUNING_FLASH       : 闪光灯配置
HW_ISP_CFG_TUNING_FLICKER     : 工频检测配置
HW_ISP_CFG_TUNING_VISUAL_ANGLE  : 视场角配置
HW_ISP_CFG_TUNING_GTM         : 全局色调映射配置
HW_ISP_CFG_TUNING_CFA         : CFA 插值配置
HW_ISP_CFG_TUNING_CTC         : 迷宫纹理去除配置
HW_ISP_CFG_TUNING_DIGITAL_GAIN  : 数字增益配置
HW_ISP_CFG_TUNING_CCM_LOW     : 低色温 CCM 配置
HW_ISP_CFG_TUNING_CCM_MID     : 中色温 CCM 配置
HW_ISP_CFG_TUNING_CCM_HIGH    : 高色温 CCM 配置
HW_ISP_CFG_TUNING_PLTM        : 局部色调映射配置
HW_ISP_CFG_TUNING_GCA         : 色差校正配置
HW_ISP_CFG_TUNING_BDNF_COMM   : 2D 降噪配置
HW_ISP_CFG_TUNING_TDNF_COMM   : 3D 降噪配置
HW_ISP_CFG_TUNING_SHARP_COMM  : 锐化降噪配置
HW_ISP_CFG_TUNING_DPC         : 坏点去除配置
HW_ISP_CFG_TUNING_ENCPP_SHARP_COMM: 编码锐化配置
HW_ISP_CFG_TUNING_SENSOR      : sensor 配置
```

- DESCRIPTION

版权所有 © 珠海全志科技股份有限公司。保留一切权利

hw_isp_cfg_tuning_ids：列出了一些基本模块的配置命令。

6.4.2 isp_tuning_param_cfg

- PROTOTYPE

```
/* isp_tuning_param cfg */
struct isp_tuning_param_cfg {
    struct isp_tuning_flash_cfg          flash;       /* id: 0x0300000001 */
    struct isp_tuning_flicker_cfg        flicker;     /* id: 0x0300000002 */
    struct isp_tuning_visual_angle_cfg   visual_angle; /* id: 0x0300000004 */
    struct isp_tuning_gtm_cfg            gtm;         /* id: 0x0300000008 */
    struct isp_tuning_cfa_cfg            cfa;         /* id: 0x0300000010 */
    struct isp_tuning_ctc_cfg            ctc;         /* id: 0x0300000020 */
    struct isp_tuning_blc_gain_cfg       digital_gain; /* id: 0x0300000040 */
    struct isp_tuning_ccm_cfg            ccm_low;     /* id: 0x0300000080 */
    struct isp_tuning_ccm_cfg            ccm_mid;     /* id: 0x0300000100 */
    struct isp_tuning_ccm_cfg            ccm_high;    /* id: 0x0300000200 */
    struct isp_tuning_pltm_cfg           pltm;        /* id: 0x0300000400 */
    struct isp_tuning_gca_cfg            gca;         /* id: 0x0300000800 */
    struct isp_tuning_bdnf_comm_cfg      bdnf_comm;   /* id: 0x0300001000 */
    struct isp_tuning_tdnf_comm_cfg      tdnf_comm;   /* id: 0x0300002000 */
    struct isp_tuning_sharp_comm_cfg     sharp_comm;  /* id: 0x0300004000 */
    struct isp_tuning_dpc_cfg            dpc;         /* id: 0x0300008000 */
    /* tuning tables */
    struct isp_tuning_lsc_table_cfg      lsc;         /* id: 0x0400000001 */
    struct isp_tuning_gamma_table_cfg    gamma;       /* id: 0x0400000002 */
    struct isp_tuning_bdnf_table_cfg     bdnf;        /* id: 0x0400000010 */
    struct isp_tuning_tdnf_table_cfg     tdnf;        /* id: 0x0400000020 */
    struct isp_tuning_sharp_table_cfg    sharp;       /* id: 0x0400000080 */
    struct isp_tuning_cem_table_cfg      cem;         /* id: 0x0400000100 */
    struct isp_tuning_cem_table_cfg      cem_1;       /* id: 0x0400000200 */
    struct isp_tuning_pltm_table_cfg     pltm_tbl;    /* id: 0x0400000400 */
    struct isp_tuning_wdr_table_cfg      wdr;         /* id: 0x0400000800 */
    struct isp_tuning_msc_table_cfg      msc;         /* id: 0x0400001000 */
    struct isp_tuning_lca_pf_satu_lut    lca_pf;      /* id: 0x0400002000 */
    struct isp_tuning_lca_gf_satu_lut    lca_gf;      /* id: 0x0400002000 */
};
```

- MEMBERS

```
flash     : 闪光灯配置
flicker   : 工频检测配置
visual_angle: 视场角配置
gtm       : 全局色调映射配置
cfa       : cfa插值配置
ctc       : 迷宫纹理去除配置
digital_gain: 数字增益配置
ccm_low   : 低色温ccm配置
ccm_mid   : 中色温ccm配置
ccm_high  : 高色温ccm配置
pltm      : 局部色调映射配置
gca       : 色差校正配置
bdnf_comm  : 2D 降噪配置
tdnf_comm  : 3D 降噪配置
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
sharp_comm ：锐化配置
dpc      ：坏点去除配置
/* tuning tables */
lsc    ：镜头阴影校正 table 配置
gamma  ：伽马校正    table 配置
bdnf   ：2D 降噪     table 配置
tdnf   ：3D 降噪     table 配置
sharp  ：锐化        table 配置
cem    ：色彩增强0   table 配置
cem_1  ：色彩增强1   table 配置
pltm_tbl: 局部色调映射 table 配置
wdr    ：宽动态      table 配置
msc    ：镜头阴影校正 table 配置
lca_pf ：色差校正    table 配置
lca_gf ：色差校正    table 配置
```

- DESCRIPTION

```
isp_tuning_param_cfg：用于描述ISP基本模块的配置参数。
```

6.4.3 isp_tuning_flash_cfg

- PROTOTYPE

```
struct isp_tuning_flash_cfg {
  HW_S32 flash_gain;
  HW_S32 delay_frame;
};
```

- MEMBERS

```
flash_gain：闪光比例；256*(预闪亮度/主闪亮度)；range：0~1023，default：64
delay_frame：预闪帧数；range：0~63，default：16
```

- DESCRIPTION

```
isp_tuning_flash_cfg：为描述闪光灯模块配置的结构体。
```

6.4.4 isp_tuning_flicker_cfg

- PROTOTYPE

```
struct isp_tuning_flicker_cfg {
  HW_S32 type;
  HW_S32 ratio;
};
```

- MEMBERS

版权所有 © 珠海全志科技股份有限公司。保留一切权利

type：工频检测类型；0为自动检测，1为50 HZ，2为60 HZ；range：0~3，default：0
ratio：检测变化比例；一般取10~30，该值越低检测越灵敏，误检率也会提高；range：0~100，default：16

- DESCRIPTION

isp_tuning_flicker_cfg：用于描述工频检测模块配置的结构体。

6.4.5 isp_tuning_visual_angle_cfg

- PROTOTYPE

```
struct isp_tuning_visual_angle_cfg {
    HW_S32 horizontal;
    HW_S32 vertical;
    HW_S32 focus_length;
};
```

- MEMBERS

horizontal：水平可视角度；range：0~360，default：60
vertical：垂直可视角度；range：0~360，default：40
focus_length：焦距值，例如焦距为28 mm，则该值为28x100；range：0~65535，default：300

- DESCRIPTION

isp_tuning_visual_angle_cfg：用于描述可视角和焦距等镜头规格的结构体。

6.4.6 isp_tuning_gtm_cfg

- PROTOTYPE

```
struct isp_tuning_gtm_cfg {
    HW_S32    gtm_hist_sel;
    HW_S32    type;
    HW_S32    gamma_type;
    HW_S32    auto_alpha_en;
    HW_S32    hist_pix_cnt;
    HW_S32    dark_minval;
    HW_S32    bright_minval;
    HW_S16    plum_var[GTM_LUM_IDX_NUM][GTM_VAR_IDX_NUM];
};
```

- MEMBERS

版权所有 © 珠海全志科技股份有限公司。保留一切权利

gtm_hist_sel：对比度调整直方图统计值输出位置选择
type：对比度调整类型；0: 按照预设亮度对比度来调整DRC 模块 1: 根据直方图自适应调整DRC模块
gamma_type：gamma类型；0: 使用标准gamma 1: 使用动态gamma
auto_alpha_en：是否根据图像内容自动调整亮暗阈值；0: 否 1: 是
hist_pix_cnt：统计值像素计数值
dark_minval：暗部最小值
bright_minval：亮部最大值

- DESCRIPTION

isp_tuning_gtm_cfg：用于描述全局色调映射(对比度调整)模块配置的结构体。

6.4.7 isp_tuning_cfa_cfg

- PROTOTYPE

```
struct isp_tuning_cfa_cfg {
  HW_S16    grad_th;
  HW_S16    dir_v_th;
  HW_S16    dir_h_th;
  HW_S16    res_smth_high;
  HW_S16    res_smth_low;
  HW_S16    res_high_th;
  HW_S16    res_low_th;
  HW_S16    res_dir_a;
  HW_S16    res_dir_d;
  HW_S16    res_dir_v;
  HW_S16    res_dir_h;
};
```

- MEMBERS

grad_th: 梯度阈值；range：0~255，default：128
dir_v_th: 水平边缘（垂直方向）色差阈值；range：0~4095，default：4095
dir_h_th: 垂直边缘（水平方向）色差阈值；range：0~4095，default：4095
res_smth_high: 解析度模糊阈值上限；range：0~64，default：0
res_smth_low: 解析度模糊阈值下限；range：0~64，default：0
res_high_th: 解析度模糊阈值上限的过渡步长 range：0~4095，default：4095
res_low_th: 解析度模糊阈值下限的过渡步长 range：0~4095，default：4095
res_dir_a: 对角（右上到左下方向）的解析度模糊程度 range：0~16，default：16
res_dir_d: 对角（左上到右下方向）的解析度模糊程度 range：0~16，default：16
res_dir_v: 垂直方向的解析度模糊程度；range：0~16，default：16
res_dir_h: 水平方向的解析度模糊程度；range：0~16，default：16

- DESCRIPTION

isp_tuning_cfa_cfg：用于描述CFA插值模块配置的结构体。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

6.4.8 isp_tuning_ctc_cfg

- PROTOTYPE

```
struct isp_tuning_ctc_cfg {
  HW_U16    min_thres;
  HW_U16    max_thres;
  HW_U16    slope_thres;
  HW_U16    dir_wt;
  HW_U16    dir_thres;
};
```

- MEMBERS

```
min_thres: 判断过渡边缘的下域值
max_thres: 判断过渡边缘的上域值
slope_thres: 判断过渡边缘的斜率
dir_wt: 判断边缘方向的权重
dir_thres: 判断边缘方向的阈值
```

- DESCRIPTION

```
isp_tuning_ctc_cfg：用于描述迷宫纹理去除模块配置的结构体。
```

6.4.9 isp_tuning_blc_gain_cfg

- PROTOTYPE

```
struct isp_tuning_blc_gain_cfg {
  HW_S32 value[ISP_RAW_CH_MAX];
};
```

- MEMBERS

```
value：所要设置的各个通道值，各通道意义如下：
enum isp_raw_ch {
  ISP_RAW_CH_R = 0,
  ISP_RAW_CH_GR,
  ISP_RAW_CH_GB,
  ISP_RAW_CH_G,
  ISP_RAW_CH_MAX,
};
```

- DESCRIPTION

```
isp_tuning_blc_gain_cfg：用于描述预校正增益以及黑电平模块配置的结构体。
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

6.4.10 isp_tuning_ccm_cfg

- PROTOTYPE

```
struct isp_tuning_ccm_cfg {
  HW_U16 temperature;
  struct isp_rgb2rgb_gain_offset  value;
};
```

- MEMBERS

```
temperature：色温值
value：色彩校正矩阵
struct isp_rgb2rgb_gain_offset {
  HW_S16 matrix[3][3];
  HW_S16 offset[3];
};
```

- DESCRIPTION

```
isp_tuning_ccm_cfg：用于描述色彩校正模块配置的结构体。
```

6.4.11 isp_tuning_pltm_cfg

- PROTOTYPE

```
struct isp_tuning_pltm_cfg {
  HW_U8 value[ISP_PLTM_MEM_SIZE];
};
enum isp_pltm_comm_cfg {
  ISP_PLTM_MODE          = 0,
  ISP_PLTM_SPEED         = 1,
  ISP_PLTM_TOLERANCE     = 2,
  ISP_PLTM_HDR_RATIO     = 3,
  ISP_PLTM_HDR_LOW       = 4,
  ISP_PLTM_HDR_HIGH      = 5,
  ISP_PLTM_GTM_EN        = 6,
  ISP_PLTM_LTF_EN        = 7,
  ISP_PLTM_DCC_EN        = 8,
  ISP_PLTM_DCC_SHF_BIT   = 9,
  ISP_PLTM_DCC_LW_TH     = 10,
  ISP_PLTM_DCC_HI_TH     = 11,
  ISP_PLTM_DSC_EN        = 12,
  ISP_PLTM_DSC_SHF_BIT   = 13,
  ISP_PLTM_DSC_LW_TH     = 14,
  ISP_PLTM_DSC_HI_TH     = 15,
  ISP_PLTM_DSC_LW_RT     = 16,
  ISP_PLTM_MGC_INT_SMTH0 = 17,
  ISP_PLTM_MGC_INT_SMTH1 = 18,
  ISP_PLTM_MGC_INT_SMTH2 = 19,
  ISP_PLTM_MGC_INT_SMTH3 = 20,
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
  ISP_PLTM_MGC_LUM_ADPT0 = 21,
  ISP_PLTM_MGC_LUM_ADPT1 = 22,
  ISP_PLTM_MGC_LUM_ADPT2 = 23,
  ISP_PLTM_MGC_LUM_ADPT3 = 24,
  ISP_PLTM_STS_CEIL_SLP0 = 25,
  ISP_PLTM_STS_CEIL_SLP1 = 26,
  ISP_PLTM_STS_CEIL_SLP2 = 27,
  ISP_PLTM_STS_CEIL_SLP3 = 28,
  ISP_PLTM_STS_FLOOR_SLP0 = 29,
  ISP_PLTM_STS_FLOOR_SLP1 = 30,
  ISP_PLTM_STS_FLOOR_SLP2 = 31,
  ISP_PLTM_STS_FLOOR_SLP3 = 32,
  ISP_PLTM_STS_GD_RT0   = 33,
  ISP_PLTM_STS_GD_RT1   = 34,
  ISP_PLTM_STS_GD_RT2   = 35,
  ISP_PLTM_STS_GD_RT3   = 36,
  ISP_PLTM_ADJK_CRCT_RT0 = 37,
  ISP_PLTM_ADJK_CRCT_RT1 = 38,
  ISP_PLTM_ADJK_CRCT_RT2 = 39,
  ISP_PLTM_ADJK_CRCT_RT3 = 40,
  ISP_PLTM_INTERVAL     = 41,
  ISP_PLTM_DARK_LOW_TH   = 42,
  ISP_PLTM_DARK_HIGH_TH  = 43,
  ISP_PLTM_DARKNEST_NUM  = 44,
  ISP_PLTM_MAX,
};
```

- MEMBERS

value：参考datasheet或者调试指南；range：0~4095

- DESCRIPTION

isp_tuning_pltm_cfg：为描述局部色调映射模块配置的结构体。

6.4.12 isp_tuning_gca_cfg

- PROTOTYPE

```
struct isp_tuning_gca_cfg {
  HW_S16    value[ISP_GCA_MAX];
};
enum isp_gca_cfg {
  ISP_GCA_CT_W = 0,
  ISP_GCA_CT_H = 1,
  ISP_GCA_R_PARA0 = 2,
  ISP_GCA_R_PARA1 = 3,
  ISP_GCA_R_PARA2 = 4,
  ISP_GCA_B_PARA0 = 5,
  ISP_GCA_B_PARA1 = 6,
  ISP_GCA_B_PARA2 = 7,
  ISP_GCA_INT_CNS = 8,
  ISP_GCA_MAX,
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
};
```

- MEMBERS

value：参考datasheet或者调试指南

- DESCRIPTION

isp_tuning_gca_cfg：用于描述色差校正模块配置的结构体。

6.4.13 isp_tuning_bdnf_comm_cfg

- PROTOTYPE

```
struct isp_tuning_bdnf_comm_cfg {
  HW_S16    value[ISP_DENOISE_COMM_MAX];
};

enum isp_denoise_comm_cfg {
  ISP_DENOISE_LYR0_CORE_RATIO = 0,
  ISP_DENOISE_LYR1_CORE_RATIO = 1,
  ISP_DENOISE_LYR2_CORE_RATIO = 2,
  ISP_DENOISE_LYR3_CORE_RATIO = 3,
  ISP_DENOISE_LYR0_SIDE_RATIO = 4,
  ISP_DENOISE_LYR1_SIDE_RATIO = 5,
  ISP_DENOISE_LYR2_SIDE_RATIO = 6,
  ISP_DENOISE_LYR3_SIDE_RATIO = 7,
  ISP_DENOISE_LYR0_DNR_CRATIO = 8,
  ISP_DENOISE_LYR1_DNR_CRATIO = 9,
  ISP_DENOISE_LYR2_DNR_CRATIO = 10,
  ISP_DENOISE_LYR3_DNR_CRATIO = 11,
  ISP_DENOISE_LYR0_GAUSS_TYPE = 12,
  ISP_DENOISE_LYR1_GAUSS_TYPE = 13,
  ISP_DENOISE_LYR2_GAUSS_TYPE = 14,
  ISP_DENOISE_LYR3_GAUSS_TYPE = 15,
  ISP_DENOISE_LYR0_GAUSS_WGTH = 16,
  ISP_DENOISE_LYR1_GAUSS_WGTH = 17,
  ISP_DENOISE_LYR2_GAUSS_WGTH = 18,
  ISP_DENOISE_LYR3_GAUSS_WGTH = 19,
  ISP_DENOISE_CNT_RATIO0 = 20,
  ISP_DENOISE_CNT_RATIO1 = 21,
  ISP_DENOISE_CNT_RATIO2 = 22,
  ISP_DENOISE_CNT_RATIO3 = 23,
  ISP_DENOISE_WDR_LM_HI_SLP = 24,
  ISP_DENOISE_WDR_MS_HI_SLP = 25,
  ISP_DENOISE_WDR_LM_LW_SLP = 26,
  ISP_DENOISE_WDR_MS_LW_SLP = 27,
  ISP_DENOISE_WDR_LM_MAX_CLP = 28,
  ISP_DENOISE_WDR_MS_MAX_CLP = 29,
  ISP_DENOISE_MSC_CMP_RATIO = 30,
  ISP_DENOISE_DTC_MG_MODE = 31,
  ISP_DENOISE_DTC_MG_CLIP = 32,
  ISP_DENOISE_COMM_MAX,
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
};
```

- MEMBERS

thres：2D降噪配置表，用来配置不同像素亮度下的降噪阈值；range：0~4095

- DESCRIPTION

isp_tuning_bdnf_cfg：用于描述2D降噪模块配置的结构体。

6.4.14 isp_tuning_tdnf_comm_cfg

- PROTOTYPE

```
struct isp_tuning_tdnf_comm_cfg {
  HW_S16    value[ISP_TDF_COMM_MAX];
};

enum isp_tdf_comm_cfg {
  ISP_TDF_DIFF_INTRA_AMP = 0,
  ISP_TDF_DIFF_INTER_AMP = 1,
  ISP_TDF_THR_INTRA_AMP  = 2,
  ISP_TDF_THR_INTER_AMP  = 3,
  ISP_TDF_DIFF_MIN_CLP   = 4,
  ISP_TDF_WDR_LM_HI_SLP  = 5,
  ISP_TDF_WDR_MS_HI_SLP  = 6,
  ISP_TDF_WDR_LM_LW_SLP  = 7,
  ISP_TDF_WDR_MS_LW_SLP  = 8,
  ISP_TDF_WDR_LM_MAX_CLP = 9,
  ISP_TDF_WDR_MS_MAX_CLP = 10,
  ISP_TDF_MSC_CMP_RATIO  = 11,
  ISP_TDF_STL_STG_CTH_0  = 12,
  ISP_TDF_STL_STG_CTH_1  = 13,
  ISP_TDF_STL_STG_CTH_2  = 14,
  ISP_TDF_STL_STG_CTH_3  = 15,
  ISP_TDF_STL_STG_CTH_4  = 16,
  ISP_TDF_STL_STG_CTH_5  = 17,
  ISP_TDF_STL_STG_CTH_6  = 18,
  ISP_TDF_STL_STG_CTH_7  = 19,
  ISP_TDF_STL_STG_KTH_0  = 20,
  ISP_TDF_STL_STG_KTH_1  = 21,
  ISP_TDF_STL_STG_KTH_2  = 22,
  ISP_TDF_STL_STG_KTH_3  = 23,
  ISP_TDF_STL_STG_KTH_4  = 24,
  ISP_TDF_STL_STG_KTH_5  = 25,
  ISP_TDF_STL_STG_KTH_6  = 26,
  ISP_TDF_STL_STG_KTH_7  = 27,
  ISP_TDF_MOT_STG_CTH_0  = 28,
  ISP_TDF_MOT_STG_CTH_1  = 29,
  ISP_TDF_MOT_STG_CTH_2  = 30,
  ISP_TDF_MOT_STG_CTH_3  = 31,
  ISP_TDF_MOT_STG_CTH_4  = 32,
  ISP_TDF_MOT_STG_CTH_5  = 33,
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
  ISP_TDF_MOT_STG_CTH_6 = 34,
  ISP_TDF_MOT_STG_CTH_7 = 35,
  ISP_TDF_MOT_STG_KTH_0 = 36,
  ISP_TDF_MOT_STG_KTH_1 = 37,
  ISP_TDF_MOT_STG_KTH_2 = 38,
  ISP_TDF_MOT_STG_KTH_3 = 39,
  ISP_TDF_MOT_STG_KTH_4 = 40,
  ISP_TDF_MOT_STG_KTH_5 = 41,
  ISP_TDF_MOT_STG_KTH_6 = 42,
  ISP_TDF_MOT_STG_KTH_7 = 43,
  ISP_TDF_FILT_OUT_SEL  = 44,
  ISP_TDF_COMM_MAX,
};
```

- MEMBERS

value：参考datasheet

- DESCRIPTION

isp_tuning_tdnf_cfg：用于描述3D降噪模块配置的结构体。

6.4.15 isp_tuning_sharp_cfg

- PROTOTYPE

```
struct isp_tuning_sharp_comm_cfg {
  HW_S16    value[ISP_SHARP_COMM_MAX];
};
enum isp_sharp_comm_cfg {
  ISP_SHARP_SS_SHP_RATIO   = 0,
  ISP_SHARP_SS_DIR_RATIO   = 1,
  ISP_SHARP_SS_CRC_STREN   = 2,
  ISP_SHARP_SS_CRC_MIN     = 3,
  ISP_SHARP_LS_SHP_RATIO   = 4,
  ISP_SHARP_LS_DIR_RATIO   = 5,
  ISP_SHARP_WHT_SAT_RATIO  = 6,
  ISP_SHARP_BLK_SAT_RATIO  = 7,
  ISP_SHARP_WHT_SLP_BT     = 8,
  ISP_SHARP_BLK_SLP_BT     = 9,
  ISP_SHARP_COMM_MAX,
};
```

- MEMBERS

value：参考datasheet或者调试指南；range：0~4095

- DESCRIPTION

版权所有 © 珠海全志科技股份有限公司。保留一切权利

isp_tuning_sharp_cfg：用于描述锐化模块配置的结构体。

6.4.16 isp_tuning_dpc_cfg

- PROTOTYPE

```
struct isp_tuning_dpc_cfg {
  HW_S8 dpc_remove_mode;
  HW_S8 dpc_sup_twinkle_en;
};
```

- MEMBERS

dpc_remove_mode：坏点去除模式
dpc_sup_twinkle_en：闪烁抑制使能

- DESCRIPTION

isp_tuning_dpc_cfg：为描述坏点去除模块配置的结构体。

6.5 Dynamic 命令组

6.5.1 hw_isp_cfg_dynamic_ids

- PROTOTYPE

```
typedef enum {
  /* isp_dynamic_cfg */
  HW_ISP_CFG_DYNAMIC_LUM_POINT          = 0x00000001,
  HW_ISP_CFG_DYNAMIC_GAIN_POINT         = 0x00000002,
  HW_ISP_CFG_DYNAMIC_SHARP              = 0x00000004,
  HW_ISP_CFG_DYNAMIC_DENOISE            = 0x00000010,
  HW_ISP_CFG_DYNAMIC_BLACK_LV           = 0x00000040,
  HW_ISP_CFG_DYNAMIC_DPC                = 0x00000080,
  HW_ISP_CFG_DYNAMIC_PLTM               = 0x00000100,
  HW_ISP_CFG_DYNAMIC_DEFOG              = 0x00000200,
  HW_ISP_CFG_DYNAMIC_HISTOGRAM          = 0x00000400,
  HW_ISP_CFG_DYNAMIC_CEM                = 0x00001000,
  HW_ISP_CFG_DYNAMIC_TDF                = 0x00002000,
  HW_ISP_CFG_DYNAMIC_AE                 = 0x00004000,
  HW_ISP_CFG_DYNAMIC_GTM                = 0x00008000,
  HW_ISP_CFG_DYNAMIC_LCA                = 0x00010000,
  HW_ISP_CFG_DYNAMIC_CFA                = 0x00020000,
  HW_ISP_CFG_DYNAMIC_ENCPP_SHARP        = 0x00040000,
  HW_ISP_CFG_DYNAMIC_WDR                = 0x00080000,
  HW_ISP_CFG_DYNAMIC_ENCODER_DENOISE    = 0x00100000,
  HW_ISP_CFG_DYNAMIC_SHADING            = 0x00200000,
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
} hw_isp_cfg_dynamic_ids;
```

- MEMBERS

```
HW_ISP_CFG_DYNAMIC_LUM_POINT    : 设置亮度映射点(一般不用设置)
HW_ISP_CFG_DYNAMIC_GAIN_POINT   : 设置增益映射点(一般不用设置)
HW_ISP_CFG_DYNAMIC_SHARP        : 设置不同 ISO 下的锐化参数
HW_ISP_CFG_DYNAMIC_DENOISE      : 设置不同 ISO 下的2D降噪参数
HW_ISP_CFG_DYNAMIC_BLACK_LV     : 设置不同 ISO 下的黑电平校正参数
HW_ISP_CFG_DYNAMIC_DPC          : 设置不同 ISO 下的坏点去除参数
HW_ISP_CFG_DYNAMIC_PLTM         : 设置不同 ISO 下的局部色调映射参数
HW_ISP_CFG_DYNAMIC_DEFOG        : 设置不同 ISO 下的去雾算法参数
HW_ISP_CFG_DYNAMIC_HISTOGRAM    : 设置不同 ISO 下的参数
HW_ISP_CFG_DYNAMIC_CEM          : 设置不同 ISO 下的色彩增强参数
HW_ISP_CFG_DYNAMIC_TDF          : 设置不同 ISO 下的3D降噪参数
HW_ISP_CFG_DYNAMIC_AE           : 设置不同 ISO 下的自动曝光算法参数
HW_ISP_CFG_DYNAMIC_GTM          : 设置不同 ISO 下的全局色调映射参数
HW_ISP_CFG_DYNAMIC_LCA          : 设置不同 ISO 下的塞色差校正参数
HW_ISP_CFG_DYNAMIC_CFA          : 设置不同 ISO 下的CFA插值参数
HW_ISP_CFG_DYNAMIC_ENCPP_SHARP  : 设置不同 ISO 下的锐化参数
HW_ISP_CFG_DYNAMIC_WDR          : 设置不同 ISO 下的WDR参数
HW_ISP_CFG_DYNAMIC_ENCODER_DENOISE: 设置不同 ISO 下的降噪参数
HW_ISP_CFG_DYNAMIC_SHADING      : 设置不同 ISO 下的镜头阴影校正参数
```

- DESCRIPTION

```
hw_isp_cfg_dynamic_ids：用于枚举ISP动态参数设置命令。
```

6.5.2 isp_dynamic_param_cfg

- PROTOTYPE

```
struct isp_dynamic_param_cfg {
    struct isp_dynamic_single_cfg    lum_mapping_point;  /* id: 0x0500000001 */
    struct isp_dynamic_single_cfg    gain_mapping_point; /* id: 0x0500000002 */
    struct isp_dynamic_sharp_cfg     sharp;          /* id: 0x0500000004 */
    struct isp_dynamic_denoise_cfg   denoise;        /* id: 0x0500000010 */
    struct isp_dynamic_black_level_cfg black_level;  /* id: 0x0500000040 */
    struct isp_dynamic_dpc_cfg       dpc;            /* id: 0x0500000080 */
    struct isp_dynamic_pltm_cfg      pltm;           /* id: 0x0500000100 */
    struct isp_dynamic_defog_cfg     defog;          /* id: 0x0500000200 */
    struct isp_dynamic_histogram_cfg histogram;      /* id: 0x0500000400 */
    struct isp_dynamic_cem_cfg       cem;            /* id: 0x0500001000 */
    struct isp_dynamic_tdf_cfg       tdf;            /* id: 0x0500002000 */
    struct isp_dynamic_ae_cfg        ae;             /* id: 0x0500004000 */
    struct isp_dynamic_gtm_cfg       gtm;            /* id: 0x0500008000 */
    struct isp_dynamic_lca_cfg       lca;            /* id: 0x0500010000 */
    struct isp_dynamic_cfa_cfg       cfa;            /* id: 0x0500020000 */
};
```

- MEMBERS

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
lum_mapping_point : 设置亮度映射点(一般不用设置)
gain_mapping_point : 设置增益映射点(一般不用设置)
sharp          : 锐化ISO联动参数
denoise        : 2D降噪ISO联动参数
black_level    : 黑电平校正ISO联动参数
dpc            : 坏点去除ISO联动参数
pltm           : 局部色调映射ISO联动参数
defog          : 去雾算法ISO联动参数
histogram      : 亮度/对比度参数
cem            : 色彩增强ISO联动参数
tdf            : 3D降噪ISO联动参数
ae             : 自动曝光算法ISO联动参数
gtm            : 全局色调映射ISO联动参数
lca            : 色差校正ISO联动参数
cfa            : CFA插值ISO联动参数
```

- DESCRIPTION

```
isp_dynamic_param_cfg：用于描述ISP动态参数设置的结构体。
```

6.5.3 isp_dynamic_single_cfg

- PROTOTYPE

```
struct isp_dynamic_single_cfg {
  HW_S32 value[ISP_DYNAMIC_GROUP_COUNT];
};
```

- MEMBERS

```
value: 映射点配置
```

- DESCRIPTION

```
isp_dynamic_single_cfg：用于描述ISP映射点配置的结构体。
```

6.5.4 isp_dynamic_sharp_cfg

- PROTOTYPE

```
struct isp_dynamic_sharp_cfg {
  HW_S16 trigger;
  struct isp_dynamic_sharp_ss_item tuning_ss_cfg[ISP_DYNAMIC_GROUP_COUNT];
  struct isp_dynamic_sharp_ls_item tuning_ls_cfg[ISP_DYNAMIC_GROUP_COUNT];
  struct isp_dynamic_sharp_hfr_item tuning_hfr_cfg[ISP_DYNAMIC_GROUP_COUNT];
  struct isp_dynamic_sharp_comm_item tuning_comm_cfg[ISP_DYNAMIC_GROUP_COUNT];
};
```

- MEMBERS

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
trigger：锐化触发选择；0：gain触发 1：lum触发
tuning_ss_cfg：小尺度锐化参数配置
struct isp_dynamic_sharp_ss_item {
  HW_S16    value[ISP_SHARP_LS_NS_LW];
};
tuning_ls_cfg：大尺度锐化参数配置
struct isp_dynamic_sharp_ls_item {
  HW_S16    value[ISP_SHARP_HFR_SMTH_RATIO - ISP_SHARP_LS_NS_LW];
};
tuning_hfr_cfg：高频锐化参数配置
struct isp_dynamic_sharp_hfr_item {
  HW_S16    value[ISP_SHARP_DIR_SMTH0 - ISP_SHARP_HFR_SMTH_RATIO];
};
tuning_comm_cfg：锐化参数配置
struct isp_dynamic_sharp_comm_item {
  HW_S16    value[ISP_SHARP_MAX - ISP_SHARP_DIR_SMTH0];
};

enum isp_sharp_cfg {
  ISP_SHARP_SS_NS_LW     = 0,//SS
  ISP_SHARP_SS_NS_HI     = 1,
  ISP_SHARP_SS_LW_COR    = 2,
  ISP_SHARP_SS_HI_COR    = 3,
  ISP_SHARP_SS_BLK_STREN   = 4,
  ISP_SHARP_SS_WHT_STREN   = 5,
  ISP_SHARP_SS_AVG_SMTH    = 6,
  ISP_SHARP_SS_DIR_SMTH    = 7,
  ISP_SHARP_LS_NS_LW     = 8,//LS
  ISP_SHARP_LS_NS_HI     = 9,
  ISP_SHARP_LS_LW_COR    = 10,
  ISP_SHARP_LS_HI_COR    = 11,
  ISP_SHARP_LS_BLK_STREN   = 12,
  ISP_SHARP_LS_WHT_STREN   = 13,
  ISP_SHARP_HFR_SMTH_RATIO  = 14,//HFR
  ISP_SHARP_HFR_HF_WHT_STREN = 15,
  ISP_SHARP_HFR_HF_BLK_STREN = 16,
  ISP_SHARP_HFR_MF_WHT_STREN = 17,
  ISP_SHARP_HFR_MF_BLK_STREN = 18,
  ISP_SHARP_HFR_RD_WHT_STREN = 19,
  ISP_SHARP_HFR_RD_BLK_STREN = 20,
  ISP_SHARP_HFR_HF_COR_RATIO = 21,
  ISP_SHARP_HFR_MF_COR_RATIO = 22,
  ISP_SHARP_HFR_RD_COR_RATIO = 23,
  ISP_SHARP_HFR_HF_MIX_RATIO = 24,
  ISP_SHARP_HFR_MF_MIX_RATIO = 25,
  ISP_SHARP_HFR_RD_MIX_RATIO = 26,
  ISP_SHARP_HFR_HF_MIX_MIN_RATIO = 27,
  ISP_SHARP_HFR_MF_MIX_MIN_RATIO = 28,
  ISP_SHARP_HFR_RD_MIX_MIN_RATIO = 29,
  ISP_SHARP_HFR_HF_WHT_CLP   = 30,
  ISP_SHARP_HFR_HF_BLK_CLP   = 31,
  ISP_SHARP_HFR_MF_WHT_CLP   = 32,
  ISP_SHARP_HFR_MF_BLK_CLP   = 33,
  ISP_SHARP_HFR_RD_WHT_CLP   = 34,
  ISP_SHARP_HFR_RD_BLK_CLP   = 35,
  ISP_SHARP_DIR_SMTH0     = 36,//COMM
  ISP_SHARP_DIR_SMTH1     = 37,
  ISP_SHARP_DIR_SMTH2     = 38,
  ISP_SHARP_DIR_SMTH3     = 39,
  ISP_SHARP_WHT_CLP_PARA    = 40,
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
  ISP_SHARP_BLK_CLP_PARA    = 41,
  ISP_SHARP_WHT_CLP_SLP     = 42,
  ISP_SHARP_BLK_CLP_SLP     = 43,
  ISP_SHARP_MAX_CLP_RATIO   = 44,
  ISP_SHARP_MAX,
};
```

- DESCRIPTION

isp_dynamic_sharp_cfg：用于描述ISP动态锐化参数配置的结构体。

6.5.5 isp_dynamic_denoise_cfg

- PROTOTYPE

```
struct isp_dynamic_denoise_cfg {
  HW_S16 trigger;
  HW_S16 color_trigger;
  struct isp_dynamic_denoise_dnr_item tuning_dnr_cfg[ISP_DYNAMIC_GROUP_COUNT];
  struct isp_dynamic_denoise_dtc_item tuning_dtc_cfg[ISP_DYNAMIC_GROUP_COUNT];
  struct isp_dynamic_denoise_wdr_item tuning_wdr_cfg[ISP_DYNAMIC_GROUP_COUNT];
};
```

- MEMBERS

```
trigger：2D降噪触发选择；0：gain触发 1：lum触发
color_trigger：2D降噪触发选择；0：gain触发 1：lum触发
tuning_dnr_cfg：2D降噪dnr参数配置
struct isp_dynamic_denoise_dnr_item {
  HW_S16   value[ISP_DENOISE_HF_SMTH_RATIO];
  HW_S16   color_denoise;
};
tuning_dtc_cfg：2D降噪dtc参数配置
struct isp_dynamic_denoise_dtc_item {
  HW_S16   value[ISP_DENOISE_LYR0_DNR_LM_AMP - ISP_DENOISE_HF_SMTH_RATIO];
};
tuning_dtc_cfg：2D降噪wdr参数配置
struct isp_dynamic_denoise_wdr_item {
  HW_S16   value[ISP_DENOISE_MAX - ISP_DENOISE_LYR0_DNR_LM_AMP];
};

enum isp_denoise_cfg {
  ISP_DENOISE_BLACK_GAIN   = 0,//DNR
  ISP_DENOISE_BLACK_OFFSET  = 1,
  ISP_DENOISE_WHITE_GAIN   = 2,
  ISP_DENOISE_WHITE_OFFSET  = 3,
  ISP_DENOISE_LYR0_DNR_YRATIO = 4,
  ISP_DENOISE_LYR1_DNR_YRATIO = 5,
  ISP_DENOISE_LYR2_DNR_YRATIO = 6,
  ISP_DENOISE_LYR3_DNR_YRATIO = 7,
  ISP_DENOISE_HF_SMTH_RATIO = 8,//DTC
  ISP_DENOISE_DTC_HF_WHT_STR = 9,
  ISP_DENOISE_DTC_HF_BLK_STR = 10,
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
  ISP_DENOISE_DTC_HF_WHT_CLP = 11,
  ISP_DENOISE_DTC_HF_BLK_CLP = 12,
  ISP_DENOISE_DTC_HF_COR_RT = 13,
  ISP_DENOISE_DTC_MF_WHT_STR = 14,
  ISP_DENOISE_DTC_MF_BLK_STR = 15,
  ISP_DENOISE_DTC_MF_WHT_CLP = 16,
  ISP_DENOISE_DTC_MF_BLK_CLP = 17,
  ISP_DENOISE_DTC_MF_COR_RT = 18,
  ISP_DENOISE_LYR0_DNR_LM_AMP = 19,//WDR
  ISP_DENOISE_LYR1_DNR_LM_AMP = 20,
  ISP_DENOISE_LYR2_DNR_LM_AMP = 21,
  ISP_DENOISE_LYR3_DNR_LM_AMP = 22,
  ISP_DENOISE_LYR0_DNR_LMS_AMP = 23,
  ISP_DENOISE_LYR1_DNR_LMS_AMP = 24,
  ISP_DENOISE_LYR2_DNR_LMS_AMP = 25,
  ISP_DENOISE_LYR3_DNR_LMS_AMP = 26,
  ISP_DENOISE_MAX,
};
```

- DESCRIPTION

```
isp_dynamic_denoise_cfg：用于描述ISP 2D降噪参数配置的结构体。
```

6.5.6 isp_dynamic_black_level_cfg

- PROTOTYPE

```
struct isp_dynamic_black_level_cfg {
  HW_S16 trigger;
  struct isp_dynamic_black_level_item tuning_cfg[ISP_DYNAMIC_GROUP_COUNT];
};
```

- MEMBERS

```
trigger：BLC触发选择；0：gain触发 1：lum触发
tuning_cfg：BLC参数配置
struct isp_dynamic_black_level_item {
  HW_S16 value[ISP_BLC_MAX];
};
enum black_level {
  ISP_BLC_R_OFFSET = 0,
  ISP_BLC_GR_OFFSET = 1,
  ISP_BLC_GB_OFFSET = 2,
  ISP_BLC_B_OFFSET = 3,
  ISP_BLC_MAX,
};
```

- DESCRIPTION

```
isp_dynamic_black_level_cfg：用于描述ISP动态黑电平校正参数配置的结构体。
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

6.5.7 isp_dynamic_dpc_cfg

- PROTOTYPE

```
struct isp_dynamic_dpc_cfg {
  HW_S16 trigger;
  struct isp_dynamic_dpc_item tuning_cfg[ISP_DYNAMIC_GROUP_COUNT];
};
```

- MEMBERS

```
trigger：DPC触发选择；0：gain触发 1：lum触发
tuning_cfg：DPC参数配置
struct isp_dynamic_dpc_item {
  HW_S16    value[ISP_DPC_MAX];
};

enum dpc_cfg {
  ISP_DPC_HOT_RATIO = 0,
  ISP_DPC_COLD_RATIO = 1,
  ISP_DPC_NBHD_SMAD_RATIO = 2,
  ISP_DPC_NEAREST_SMAD_RATIO = 3,
  ISP_DPC_SLOPE_TH = 4,
  ISP_DPC_COLD_ABS_TH_COEFF = 5,
  ISP_DPC_SUP_TWINKLE_THR_H = 6,
  ISP_DPC_SUP_TWINKLE_THR_L = 7,
  ISP_DPC_MAX,
};
```

- DESCRIPTION

```
isp_dynamic_dpc_cfg：用于描述ISP动态坏点去除参数配置的结构体。
```

6.5.8 isp_dynamic_pltm_cfg

- PROTOTYPE

```
struct isp_dynamic_pltm_cfg {
  HW_S16 trigger;
  struct isp_dynamic_pltm_item   tuning_cfg[ISP_DYNAMIC_GROUP_COUNT];
};
```

- MEMBERS

```
trigger：PLTM参数设置触发选择；0：gain触发 1：lum触发
tuning_cfg：PLTM参数配置
struct isp_dynamic_pltm_item {
  HW_S16    value[ISP_PLTM_DYNAMIC_MAX];
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
};
enum pltm_dynamic_cfg {
    ISP_PLTM_DYNAMIC_AUTO_STREN = 0,
    ISP_PLTM_DYNAMIC_MANUL_STREN = 1,
    ISP_PLTM_DYNAMIC_LUM_SCALE = 2,
    ISP_PLTM_DYNAMIC_LUM_RATIO = 3,
    ISP_PLTM_DYNAMIC_DCC_LW_RT = 4,
    ISP_PLTM_DYNAMIC_MIN_STREN_STEP = 5,
    ISP_PLTM_DYNAMIC_MAX_STREN_CLIP = 6,
    ISP_PLTM_DYNAMIC_SHP_SS_COMP = 7,
    ISP_PLTM_DYNAMIC_SHP_LS_COMP = 8,
    ISP_PLTM_DYNAMIC_D2D_COMP = 9,
    ISP_PLTM_DYNAMIC_D3D_COMP = 10,
    ISP_PLTM_DYNAMIC_DARKNEST_RT = 11,
    ISP_PLTM_DYNAMIC_MAX,
};
```

- DESCRIPTION

isp_dynamic_pltm_cfg：用于描述ISP局部色调映射参数配置的结构体。

6.5.9 isp_dynamic_defog_cfg

- PROTOTYPE

```
struct isp_dynamic_defog_cfg {
    HW_S16 trigger;
    struct isp_dynamic_defog_item   tuning_cfg[ISP_DYNAMIC_GROUP_COUNT];
};
```

- MEMBERS

```
trigger：去雾算法触发选择；0：gain触发 1：lum触发
tuning_cfg：去雾算法配置
struct isp_dynamic_defog_item {
    HW_S16     value;
};
```

- DESCRIPTION

isp_dynamic_defog_cfg：用于描述ISP去雾算法参数配置的结构体。

6.5.10 isp_dynamic_histogram_cfg

- PROTOTYPE

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
struct isp_dynamic_histogram_cfg {
    HW_S16 brightness_trigger;
    HW_S16 contrast_trigger;
    struct isp_dynamic_histogram_item   tuning_cfg[ISP_DYNAMIC_GROUP_COUNT];
};
```

- MEMBERS

```
brightness_trigger：亮度设置触发选择；0：gain触发 1：lum触发
contrast_trigger：对比度设置触发选择；0：gain触发 1：lum触发
tuning_cfg：亮度/对比度配置
struct isp_dynamic_histogram_item {
    HW_S16     brightness;
    HW_S16     contrast;
};
```

- DESCRIPTION

```
isp_dynamic_histogram_cfg：用于描述ISP亮度、对比度参数配置的结构体。
```

6.5.11 isp_dynamic_cem_cfg

- PROTOTYPE

```
struct isp_dynamic_cem_cfg {
    HW_S16 trigger;
    struct isp_dynamic_cem_item tuning_cfg[ISP_DYNAMIC_GROUP_COUNT];
};
```

- MEMBERS

```
trigger：CEM 参数触发选择；0：gain触发 1：lum触发
tuning_cfg：CEM table0/table1插值比例配置
struct isp_dynamic_cem_item {
    HW_S16     value[ISP_CEM_MAX];
};
```

- DESCRIPTION

```
isp_dynamic_cem_cfg：用于描述ISP色彩增强参数配置的结构体。
```

6.5.12 isp_dynamic_tdf_cfg

- PROTOTYPE

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
struct isp_dynamic_tdf_cfg {
  HW_S16 trigger;
  struct isp_dynamic_tdf_dnr_item tuning_dnr_cfg[ISP_DYNAMIC_GROUP_COUNT];
  struct isp_dynamic_tdf_mtd_item tuning_mtd_cfg[ISP_DYNAMIC_GROUP_COUNT];
  struct isp_dynamic_tdf_dtc_item tuning_dtc_cfg[ISP_DYNAMIC_GROUP_COUNT];
  struct isp_dynamic_tdf_srd_item tuning_srd_cfg[ISP_DYNAMIC_GROUP_COUNT];
};
```

- MEMBERS

```
trigger：3D降噪触发选择；0：gain触发 1：lum触发
tuning_dnr_cfg：3D降噪dnr配置
struct isp_dynamic_tdf_dnr_item {
  HW_S16    value[ISP_TDF_DIFF_INTRA_SENS];
};
tuning_mtd_cfg：3D降噪mtd配置
struct isp_dynamic_tdf_mtd_item {
  HW_S16    value[ISP_TDF_DTC_HF_COR - ISP_TDF_DIFF_INTRA_SENS];
};
tuning_dtc_cfg：3D降噪dtc配置
struct isp_dynamic_tdf_dtc_item {
  HW_S16    value[ISP_TDF_D2D0_CNR_STREN - ISP_TDF_DTC_HF_COR];
};
tuning_srd_cfg：3D降噪srd配置
struct isp_dynamic_tdf_srd_item {
  HW_S16    value[ISP_TDF_MAX - ISP_TDF_D2D0_CNR_STREN];
};

enum isp_tdf_cfg {
  ISP_TDF_BLACK_GAIN     = 0,//DNR
  ISP_TDF_WHITE_GAIN     = 1,
  ISP_TDF_FLT1_THR_GAIN  = 2,
  ISP_TDF_SS_MV_DNR      = 3,
  ISP_TDF_SS_STL_DNR     = 4,
  ISP_TDF_LS_MV_DNR      = 5,
  ISP_TDF_LS_STL_DNR     = 6,
  ISP_TDF_NR_LM_AMP      = 7,
  ISP_TDF_NR_LMS_AMP     = 8,
  ISP_TDF_DIFF_INTRA_SENS = 9,//MTD
  ISP_TDF_DIFF_INTER_SENS = 10,
  ISP_TDF_THR_INTRA_SENS  = 11,
  ISP_TDF_THR_INTER_SENS  = 12,
  ISP_TDF_STL_CYC_VAL     = 13,
  ISP_TDF_MOT_CYC_VAL     = 14,
  ISP_TDF_CYC_DEC_SLP     = 15,
  ISP_TDF_DTC_HF_COR      = 16,//DTC
  ISP_TDF_DTC_HF_BLK_CLP  = 17,
  ISP_TDF_DTC_HF_WHT_CLP  = 18,
  ISP_TDF_DTC_MF_COR      = 19,
  ISP_TDF_DTC_MF_BLK_CLP  = 20,
  ISP_TDF_DTC_MF_WHT_CLP  = 21,
  ISP_TDF_SS_MV_SHP       = 22,
  ISP_TDF_SS_STL_SHP      = 23,
  ISP_TDF_LS_MV_SHP       = 24,
  ISP_TDF_LS_STL_SHP      = 25,
  ISP_TDF_D2D0_CNR_STREN  = 26,//SRD
  ISP_TDF_D2D1_CNR_STREN  = 27,
  ISP_TDF_MV_SATU         = 28,
  ISP_TDF_MV_K_MIN        = 29,
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
  ISP_TDF_MV_K_RATIO    = 30,
  ISP_TDF_MV_R_MIN      = 31,
  ISP_TDF_MV_R_RATIO    = 32,
  ISP_TDF_DIFF_CV_CLP   = 33,
  ISP_TDF_NPU_FACE_NR   = 34,
  ISP_TDF_MAX,
};
```

- DESCRIPTION

isp_dynamic_tdf_cfg：用于描述ISP 3D降噪参数配置的结构体。

6.5.13 isp_dynamic_ae_cfg

- PROTOTYPE

```
struct isp_dynamic_ae_cfg {
  HW_S16 trigger;
  struct isp_dynamic_ae_item  tuning_cfg[ISP_DYNAMIC_GROUP_COUNT];
};
```

- MEMBERS

```
trigger：AE触发选择；0：gain触发 1：lum触发
tuning_cfg：AE配置
struct isp_dynamic_ae_item {
  HW_S16    value[ISP_EXP_CFG_MAX];
};
enum exposure_cfg_type {
  ANTI_EXP_WIN_OVER    = 0,
  ANTI_EXP_WIN_UNDER = 1,
  ANTI_EXP_HIST_OVER = 2,
  ANTI_EXP_HIST_UNDER = 3,

  AE_PREVIEW_SPEED = 4,
  AE_CAPTURE_SPEED = 5,
  AE_VIDEO_SPEED = 6,
  AE_TOUCH_SPEED = 7,
  AE_TOLERANCE = 8,
  AE_TARGET = 9,

  AE_HIST_DARK_WEIGHT_MIN = 10,
  AE_HIST_DARK_WEIGHT_MAX = 11,
  AE_HIST_BRIGHT_WEIGHT_MIN = 12,
  AE_HIST_BRIGHT_WEIGHT_MAX = 13,

  AE_WDR_RATIO_SPEED = 14,
  AE_DYNAMIC_RESERVE_0 = 15,
  AE_DYNAMIC_RESERVE_1 = 16,
  AE_DYNAMIC_RESERVE_2 = 17,
  AE_DYNAMIC_RESERVE_3 = 18,
  ISP_EXP_CFG_MAX,
};
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

- DESCRIPTION

isp_dynamic_ae_cfg：用于描述ISP自动曝光算法参数配置的结构体。

6.5.14 isp_dynamic_gtm_cfg

- PROTOTYPE

```
struct isp_dynamic_gtm_cfg {
  HW_S16 trigger;
  struct isp_dynamic_gtm_item tuning_cfg[ISP_DYNAMIC_GROUP_COUNT];
};
```

- MEMBERS

```
trigger：GTM触发选择；0：gain触发 1：lum触发
tuning_cfg：GTM配置
struct isp_dynamic_gtm_item{
  HW_S16    value[ISP_GTM_HEQ_MAX];
};
enum isp_gtm_comm_cfg {
  ISP_GTM_GAIN = 0,
  ISP_GTM_EQ_RATIO = 1,
  ISP_GTM_EQ_SMOOTH = 2,
  ISP_GTM_BLACK = 3,
  ISP_GTM_WHITE = 4,
  ISP_GTM_BLACK_ALPHA = 5,
  ISP_GTM_WHITE_ALPHA = 6,
  ISP_GTM_GAMMA_IND = 7,
  ISP_GTM_GAMMA_PLUS = 8,
  ISP_GTM_HEQ_MAX,
};
```

- DESCRIPTION

isp_dynamic_gtm_cfg：用于描述ISP全局色调映射参数配置的结构体。

6.5.15 isp_dynamic_lca_cfg

- PROTOTYPE

```
struct isp_dynamic_lca_cfg {
  HW_S16 trigger;
  struct isp_dynamic_lca_item    tuning_cfg[ISP_DYNAMIC_GROUP_COUNT];
};
```

- MEMBERS

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
trigger：LCA触发选择；0：gain触发 1：lum触发
tuning_cfg：LCA配置
struct isp_dynamic_lca_item {
  HW_S16    value[ISP_LCA_MAX];
};
```

- DESCRIPTION

```
isp_dynamic_lca_cfg：用于描述ISP色差校正参数配置的结构体。
```

6.5.16 isp_dynamic_cfa_cfg

- PROTOTYPE

```
struct isp_dynamic_cfa_cfg {
  HW_S16 trigger;
  struct isp_dynamic_cfa_item   tuning_cfg[ISP_DYNAMIC_GROUP_COUNT];
};
```

- MEMBERS

```
trigger：CFA触发选择；0：gain触发 1：lum触发
tuning_cfg：CFA配置
struct isp_dynamic_lca_item {
  HW_S16 value[ISP_LCA_MAX];
};
```

- DESCRIPTION

```
isp_dynamic_cfa_cfg：用于描述ISP去马赛克插值参数配置的结构体。
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

7 错误码

错误码	宏定义	描述
-1	AW_ERR_VI_INVALID_PARA	无效参数
-2	AW_ERR_VI_INVALID_DEVID	无效 VI 设备号
-3	AW_ERR_VI_INVALID_CHNID	无效 VI 通道号
-4	AW_ERR_VI_INVALID_NULL_PTR	空指针
-5	AW_ERR_VI_FAILED_NOTCONFIG	模块未配置
-6	AW_ERR_VI_SYS_NOTREADY	系统未准备好
-7	AW_ERR_VI_BUF_EMPTY	缓存为空
-8	AW_ERR_VI_BUF_FULL	缓存为满
-9	AW_ERR_VI_NOMEM	无可用内存
-10	AW_ERR_VI_NOT_SUPPORT	设备不支持
-11	AW_ERR_VI_BUSY	设备忙
-12	AW_ERR_VI_FAILED_NOTENABLE	未使能
-13	AW_ERR_VI_FAILED_NOTDISABLE	未去使能
-14	AW_ERR_VI_CFG_TIMEOUT	配置超时
-15	AW_ERR_VI_NORM_UNMATCH	视频没有禁止使能
-16	AW_ERR_VI_INVALID_PHYCHNID	无效物理通道号
-17	AW_ERR_VI_FAILED_NOTBIND	设备未绑定
-18	AW_ERR_VI_FAILED_BINDED	设备已绑定

版权所有 © 珠海全志科技股份有限公司。保留一切权利

著作权声明

版权所有 ©2025 珠海全志科技股份有限公司。保留一切权利。

本文档及内容受著作权法保护，其著作权由珠海全志科技股份有限公司（“全志”）拥有并保留一切权利。

本文档是全志的原创作品和版权财产，未经全志书面许可，任何单位和个人不得擅自摘抄、复制、修改、发表或传播本文档内容的部分或全部，且不得以任何形式传播。

商标声明





（不完全列举）均为珠海全志科技股份有限公司的商标或者注册商标。在本文档描述的产品中出现的其它商标，产品名称，和服务名称，均由其各自所有人拥有。

免责声明

您购买的产品、服务或特性应受您与珠海全志科技股份有限公司（“全志”）之间签署的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所购买或使用的范围内。使用前请认真阅读合同条款和相关说明，并严格遵循本文档的使用说明。您将自行承担任何不当使用行为（包括但不限于如超压，超频，超温使用）造成的不利后果，全志概不负责。

本文档作为使用指导仅供参考。由于产品版本升级或其他原因，本文档内容有可能修改，如有变更，恕不另行通知。全志尽全力在本文档中提供准确的信息，但并不确保内容完全没有错误，因使用本文档而发生损害（包括但不限于间接的、偶然的、特殊的损失）或发生侵犯第三方权利事件，全志概不负责。本文档中的所有陈述、信息和建议并不构成任何明示或暗示的保证或承诺。

本文档未以明示或暗示或其他方式授予全志的任何专利或知识产权。在您实施方案或使用产品的过程中，可能需要获得第三方的权利许可。请您自行向第三方权利人获取相关的许可。全志不承担也不代为支付任何关于获取第三方许可的许可费或版税（专利税）。全志不对您所使用的第三方许可技术做出任何保证、赔偿或承担其他义务。