



CV184X ISP 开发使用手册

Version: 0.1.0

Release date: 2025/05

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

目录

1	声明	2
2	概述	3
2.1	概述	3
2.2	功能描述	3
2.3	架构	4
2.4	开发模式	4
2.5	内部流程	5
2.6	软件流程	6
3	系统控制	7
3.1	功能概述	7
3.2	API 参考	7
3.2.1	CVI_ISP_MemInit	8
3.2.2	CVI_ISP_Init	9
3.2.3	CVI_ISP_Run	10
3.2.4	CVI_ISP_RunOnce	11
3.2.5	CVI_ISP_Exit	11
3.2.6	CVI_ISP_SetPubAttr	12
3.2.7	CVI_ISP_GetPubAttr	13
3.2.8	CVI_ISP_SetFMWState	14
3.2.9	CVI_ISP_GetFMWState	14
3.2.10	CVI_ISP_SetModuleControl	15
3.2.11	CVI_ISP_GetModuleControl	16
3.2.12	CVI_ISP_GetVDTimeOut	17
3.2.13	CVI_ISP_SensorRegCallBack	17
3.2.14	CVI_ISP_SensorUnRegCallBack	19
3.2.15	CVI_ISP_AELibRegCallBack	19
3.2.16	CVI_ISP_AELibUnRegCallBack	21
3.2.17	CVI_ISP_AWBLibRegCallBack	21
3.2.18	CVI_ISP_AWBLibUnRegCallBack	23
3.2.19	CVI_ISP_AFLibRegCallBack	23
3.2.20	CVI_ISP_AFLibUnRegCallBack	25
3.2.21	CVI_ISP_SetBindAttr	25
3.2.22	CVI_ISP_GetBindAttr	26
3.2.23	CVI_ISP_SetCtrlParam	27
3.2.24	CVI_ISP_GetCtrlParam	28
3.2.25	CVI_ISP_SetModParam	28
3.2.26	CVI_ISP_GetModParam	29
3.2.27	CVI_BIN_SetBinName	30
3.2.28	CVI_BIN_GetBinName	30

3.2.29	CVI_BIN_GetBinExtraAttr	30
3.2.30	CVI_BIN_GetBinTotalLen	30
3.2.31	CVI_BIN_ExportBinData	30
3.2.32	CVI_BIN_ImportBinData	31
3.2.33	CVI_BIN_SaveParamToBin	31
3.2.34	CVI_BIN_LoadParamFromBin	31
3.2.35	CVI_ISP_IrAutoRunOnce	31
3.2.36	CVI_ISP_SetSmartInfo	32
3.2.37	CVI_ISP_GetSmartInfo	33
3.2.38	CVI_ISP_SetStitchAttr	34
3.2.39	CVI_ISP_GetStitchAttr	35
3.2.40	CVI_ISP_StitchCalibartion	35
3.3	数据类型	36
3.3.1	RECT_S	36
3.3.2	SIZE_S	37
3.3.3	ISP_BAYER_FORMAT_E	37
3.3.4	WDR_MODE_E	38
3.3.5	ISP_PUB_ATTR_S	39
3.3.6	ISP_FMW_STATE_E	40
3.3.7	ISP_MODULE_CTRL_U	41
3.3.8	ISP_VD_TYPE_E	42
3.3.9	ISP_SNS_ATTR_INFO_S	43
3.3.10	ISP_CMOS_SENSOR_IMAGE_MODE_S	43
3.3.11	ISP_CMOS_BLACK_LEVEL_S	44
3.3.12	ISP_CMOS_DEFAULT_S	45
3.3.13	ALG_LIB_S	45
3.3.14	ISP_AE_REGISTER_S	46
3.3.15	ISP_AE_EXP_FUNC_S	46
3.3.16	ISP_AE_PARAM_S	47
3.3.17	ISP_SMART_INFO_S	48
3.3.18	ISP_AE_INFO_S	49
3.3.19	ISP_AE_RESULT_S	50
3.3.20	ISP_SENSOR_EXP_FUNC_S	52
3.3.21	ISP_SENSOR_REGISTER_S	53
3.3.22	ISP_AWB_EXP_FUNC_S	53
3.3.23	ISP_AWB_REGISTER_S	54
3.3.24	ISP_AWB_PARAM_S	54
3.3.25	ISP_AWB_STAT_1_S	55
3.3.26	ISP_AWB_STAT_RESULT_S	56
3.3.27	ISP_AWB_INFO_S	56
3.3.28	ISP_SMART_ROI_S	57
3.3.29	ISP_AWB_RAW_STAT_ATTR_S	58
3.3.30	ISP_AWB_RESULT_S	59
3.3.31	ISP_BIND_ATTR_S	60
3.3.32	ISP_CTRL_PARAM_S	60
3.3.33	ISP_MOD_PARAM_S	61
3.3.34	ISP_IR_AUTO_ATTR_S	62
3.3.35	ISP_STITCH_ATTR_S	63

4.1	概述	65
4.2	重要概念	66
4.3	功能描述	66
4.4	API 参考	67
4.4.1	AE 库接口	67
4.4.1.1	CVI_AE_Register	67
4.4.1.2	CVI_AE_UnRegister	68
4.4.1.3	CVI_AE_SensorRegCallBack	69
4.4.1.4	CVI_AE_SensorUnRegCallBack	70
4.4.2	AE 控制模块	71
4.4.2.1	CVI_ISP_SetExposureAttr	72
4.4.2.2	CVI_ISP_GetExposureAttr	76
4.4.2.3	CVI_ISP_SetWDRExposureAttr	77
4.4.2.4	CVI_ISP_GetWDRExposureAttr	78
4.4.2.5	CVI_ISP_SetAERouteAttr	79
4.4.2.6	CVI_ISP_GetAERouteAttr	81
4.4.2.7	CVI_ISP_SetAERouteAttrEx	82
4.4.2.8	CVI_ISP_GetAERouteAttrEx	83
4.4.2.9	CVI_ISP_QueryExposureInfo	84
4.4.2.10	CVI_ISP_SetAntiFlicker	85
4.4.2.11	CVI_ISP_GetAntiFlicker	86
4.4.2.12	CVI_ISP_QueryFps	86
4.4.2.13	CVI_ISP_GetCurrentLvX100	87
4.4.2.14	CVI_ISP_SetSmartExposureAttr	89
4.4.2.15	CVI_ISP_GetSmartExposureAttr	90
4.4.2.16	CVI_ISP_SetAERouteSFAttr	91
4.4.2.17	CVI_ISP_GetAERouteSFAttr	91
4.4.2.18	CVI_ISP_SetAERouteSFAttrEx	92
4.4.2.19	CVI_ISP_GetAERouteSFAttrEx	93
4.4.3	AI 控制模块	94
4.4.3.1	CVI_ISP_SetIrisAttr	94
4.4.3.2	CVI_ISP_GetIrisAttr	95
4.4.3.3	CVI_ISP_SetDcirisAttr	95
4.4.3.4	CVI_ISP_GetDcirisAttr	96
4.4.3.5	CVI_ISP_SetPirisAttr	97
4.4.3.6	CVI_ISP_GetPirisAttr	98
4.5	数据类型	99
4.5.1	Register	99
4.5.1.1	AE_SENSOR_REGISTER_S	99
4.5.1.2	AE_SENSOR_EXP_FUNC_S	99
4.5.1.3	AE_SENSOR_DEFAULT_S	101
4.5.1.4	AE_ACCURACY_E	103
4.5.1.5	AE_ACCURACY_S	104
4.5.2	AE	104
4.5.2.1	ISP_AE_MODE_E	105
4.5.2.2	ISP_AE_STRATEGY_E	105
4.5.2.3	ISP_AE_DELAY_S	106
4.5.2.4	ISP_AE_RANGE_S	107
4.5.2.5	ISP_ANTIFLICKER_MODE_E	107
4.5.2.6	ISP_ANTIFLICKER_S	108

4.5.2.7	ISP_SUBFLICKER_S	109
4.5.2.8	ISP_FSWDR_MODE_E	110
4.5.2.9	ISP_AE_GAIN_TYPE_E	110
4.5.2.10	ISP_AE_ATTR_S	111
4.5.2.11	ISP_ME_ATTR_S	116
4.5.2.12	ISP_EXPOSURE_ATTR_S	117
4.5.2.13	ISP_WDR_EXPOSURE_ATTR_S	119
4.5.2.14	ISP_AE_ROUTE_NODE_S	121
4.5.2.15	ISP_AE_ROUTE_S	122
4.5.2.16	ISP_AE_ROUTE_EX_NODE_S	122
4.5.2.17	ISP_AE_ROUTE_EX_S	123
4.5.2.18	ISP_EXP_INFO_S	124
4.5.2.19	ISP_SMART_EXPOSURE_ATTR_S	126
4.5.3	IRIS	128
4.5.3.1	ISP_IRIS_STATUS_E	128
4.5.3.2	ISP_IRIS_TYPE_E	129
4.5.3.3	ISP_IRIS_F_NO_E	129
4.5.3.4	ISP_MI_ATTR_S	130
4.5.3.5	ISP_DCIRIS_ATTR_S	131
4.5.3.6	ISP_PIRIS_ATTR_S	132
4.5.3.7	ISP_IRIS_ATTR_S	133
5	AWB	134
5.1	概述	134
5.2	重要概念	134
5.3	功能描述	134
5.4	API 参考	135
5.4.1	AWB 库接口	135
5.4.1.1	CVI_AWB_Register	135
5.4.1.2	CVI_AWB_UnRegister	136
5.4.1.3	CVI_AWB_SensorRegCallBack	137
5.4.1.4	CVI_AWB_SensorUnRegCallBack	138
5.4.2	AWB 控制模块	139
5.4.2.1	CVI_ISP_SetWBAttr	139
5.4.2.2	CVI_ISP_GetWBAttr	140
5.4.2.3	CVI_ISP_SetAWBAttrEx	141
5.4.2.4	CVI_ISP_GetAWBAttrEx	142
5.4.2.5	CVI_ISP_QueryWBInfo	143
5.4.2.6	CVI_ISP_SetAWBLogPath	143
5.4.2.7	CVI_ISP_SetAWBLogName	144
5.4.2.8	CVI_ISP_GetGrayWorldAwbInfo	145
5.5	数据类型	146
5.5.1	Register	146
5.5.2	WB	146
5.5.2.1	ISP_MWB_ATTR_S	147
5.5.2.2	ISP_AWB_CT_LIMIT_ATTR_S	147
5.5.2.3	ISP_AWB_ALG_E	148
5.5.2.4	ISP_AWB_ATTR_S	149
5.5.2.5	ISP_AWB_ALG_TYPE_E	150
5.5.2.6	ISP_AWB_CBCR_TRACK_ATTR_S	151

5.5.2.7	ISP_AWB_LUM_HISTGRAM_ATTR_S	151
5.5.2.8	ISP_WB_ATTR_S	152
5.5.2.9	ISP_AWB_ATTR_EX_S	153
5.5.2.10	ISP_AWB_EXTRA_LIGHTSOURCE_INFO_S	154
5.5.2.11	ISP_AWB_IN_OUT_ATTR_S	155
5.5.2.12	ISP_AWB_MULTI_LS_TYPE_E	157
5.5.2.13	ISP_AWB_INDOOR_OUTDOOR_STATUS_E	157
5.5.2.14	ISP_WB_INFO_S	158
6	AF	160
6.1	概述	160
6.2	重要概念	160
6.3	功能描述	161
6.4	API 参考	161
6.4.1	AF 库接口	161
6.4.1.1	CVI_AF_Register	161
6.4.1.2	CVI_AF_UnRegister	162
6.4.1.3	CVI_AF_MOTOR_Register	162
6.4.1.4	CVI_AF_MOTOR_UnRegister	163
6.4.2	AF 控制模块	164
6.4.2.1	CVI_ISP_AFAutoFocus	164
6.4.2.2	CVI_ISP_AFGetFv	165
6.4.2.3	CVI_ISP_SetAFAttr	166
6.4.2.4	CVI_ISP_GetAFAttr	167
6.4.2.5	CVI_ISP_SetAFVcmAttr	167
6.4.2.6	CVI_ISP_GetAFVcmAttr	168
6.4.2.7	CVI_ISP_SetAFStatisticsConfig	169
6.4.2.8	CVI_ISP_GetAFStatisticsConfig	170
6.4.2.9	CVI_ISP_SetAFZoneFocusAttr	170
6.4.2.10	CVI_ISP_GetAFZoneFocusAttr	171
6.4.2.11	CVI_ISP_AFQueryFocusInfo	172
6.4.2.12	CVI_ISP_AFSetFocus	172
6.4.2.13	CVI_ISP_AFSetFocusSpeed	173
6.4.2.14	CVI_ISP_AFSetZoom	174
6.4.2.15	CVI_ISP_AFSetZoomSpeed	174
6.4.2.16	CVI_AF_GetMotorCB	175
6.5	数据类型	176
6.5.1	枚举类型	176
6.5.1.1	AF_STATUS	176
6.5.1.2	AF_DIRECTION	177
6.5.1.3	AF_CHASINGFOCUS_MODE	177
6.5.1.4	AF_MANUAL_TYPE	178
6.5.1.5	ISP_AF_MOTOR_SPEED_E	178
6.5.1.6	AF_MOTOR_TYPE	179
6.5.2	结构体类型	180
6.5.2.1	ISP_FOCUS_ATTR_S	180
6.5.2.2	ISP_FOCUS_Q_INFO_S	182
6.5.2.3	ISP_FOCUS_MANUAL_ATTR_S	183
6.5.2.4	ISP_AF_VCM_ATTR_S	184
6.5.2.5	ISP_AF_ZONE_FOCUS_ATTR_S	184

6.5.2.6	ISP_AF_ZOOM_FOCUS_TAB	185
6.5.2.7	ISP_AF_LEN_INFO_S	186
6.5.2.8	ISP_AF_MOTOR_FUNC_S	187
6.5.2.9	ISP_FOCUS_STATISTICS_CFG_S	188
6.5.2.10	ISP_AF_CFG_S	189
6.5.2.11	ISP_AF_CROP_S	190
6.5.2.12	ISP_FOCUS_ZONE_S	191
6.5.2.13	ISP_AF_STATISTICS_S	191
7	IMP	193
8	BLC	194
8.1	功能描述	194
8.2	API 参考	194
8.2.1	CVI_ISP_SetBlackLevelAttr	194
8.2.2	CVI_ISP_GetBlackLevelAttr	195
8.3	数据类型	196
8.3.1	ISP_BLACK_LEVEL_MANUAL_ATTR_S	196
8.3.2	ISP_BLACK_LEVEL_AUTO_ATTR_S	197
8.3.3	ISP_BLACK_LEVEL_ATTR_S	198
9	DPC	200
9.1	功能描述	200
9.2	API 参考	200
9.2.1	CVI_ISP_SetDPDynamicAttr	200
9.2.2	CVI_ISP_GetDPDynamicAttr	201
9.3	数据类型	202
9.3.1	ISP_DP_DYNAMIC_MANUAL_ATTR_S	202
9.3.2	ISP_DP_DYNAMIC_AUTO_ATTR_S	203
9.3.3	ISP_DP_DYNAMIC_ATTR_S	204
10	LSC	206
10.1	功能描述	206
10.2	API 参考	206
10.2.1	CVI_ISP_SetMeshShadingAttr	206
10.2.2	CVI_ISP_GetMeshShadingAttr	207
10.2.3	CVI_ISP_SetMeshShadingGainLutAttr	208
10.2.4	CVI_ISP_GetMeshShadingGainLutAttr	209
10.3	数据类型	210
10.3.1	ISP_MESH_SHADING_ATTR_S	210
10.3.2	ISP_MESH_SHADING_GAIN_LUT_ATTR_S	211
10.3.3	ISP_MESH_SHADING_GAIN_LUT_S	211
11	RLSC	213
11.1	功能描述	213
11.2	API 参考	213
11.2.1	CVI_ISP_SetRadialShadingAttr	213
11.2.2	CVI_ISP_GetRadialShadingAttr	214
11.2.3	CVI_ISP_SetRadialShadingGainLutAttr	215
11.2.4	CVI_ISP_GetRadialShadingGainLutAttr	216
11.3	数据类型	217

11.3.1	ISP_RADIAL_SHADING_MANUAL_ATTR_S	217
11.3.2	ISP_RADIAL_SHADING_AUTO_ATTR_S	218
11.3.3	ISP_RADIAL_SHADING_ATTR_S	218
11.3.4	ISP_RADIAL_SHADING_GAIN_LUT_ATTR_S	219
12	CCM	221
12.1	功能描述	221
12.2	API 参考	221
12.2.1	CVI_ISP_SetCCMAAttr	221
12.2.2	CVI_ISP_GetCCMAAttr	222
12.2.3	CVI_ISP_SetCCMSaturationAttr	223
12.2.4	CVI_ISP_GetCCMSaturationAttr	224
12.2.5	CVI_ISP_SetSaturationAttr	225
12.2.6	CVI_ISP_GetSaturationAttr	226
12.3	数据类型	226
12.3.1	ISP_CCM_ATTR_S	227
12.3.2	ISP_CCM_MANUAL_ATTR_S	228
12.3.3	ISP_CCM_AUTO_ATTR_S	228
12.3.4	ISP_COLOMATRIX_ATTR_S	229
12.3.5	ISP_CCM_SATURATION_MANUAL_ATTR_S	230
12.3.6	ISP_CCM_SATURATION_AUTO_ATTR_S	230
12.3.7	ISP_CCM_SATURATION_ATTR_S	231
12.3.8	ISP_SATURATION_MANUAL_ATTR_S	232
12.3.9	ISP_SATURATION_AUTO_ATTR_S	232
12.3.10	ISP_SATURATION_ATTR_S	233
13	Noise profile	234
13.1	功能描述	234
13.2	API 参考	234
13.2.1	CVI_ISP_SetNoiseProfileAttr	234
13.2.2	CVI_ISP_GetNoiseProfileAttr	235
13.3	数据类型	236
13.3.1	ISP_CMOS_NOISE_CALIBRATION_S	236
14	BNR	237
14.1	功能描述	237
14.2	API 参考	237
14.2.1	CVI_ISP_SetBNRAAttr	237
14.2.2	CVI_ISP_GetBNRAAttr	238
14.2.3	CVI_ISP_SetBNRFilterAttr	239
14.2.4	CVI_ISP_GetBNRFilterAttr	240
14.3	数据类型	240
14.3.1	ISP_BNR_MANUAL_ATTR_S	241
14.3.2	ISP_BNR_AUTO_ATTR_S	242
14.3.3	ISP_BNR_ATTR_S	243
14.3.4	ISP_BNR_FILTER_MANUAL_ATTR_S	244
14.3.5	ISP_BNR_FILTER_AUTO_ATTR_S	244
14.3.6	ISP_BNR_FILTER_ATTR_S	245
15	CNR	247
15.1	功能描述	247

15.2	API 参考	247
15.2.1	CVI_ISP_SetCNRAttr	247
15.2.2	CVI_ISP_GetCNRAttr	248
15.2.3	CVI_ISP_SetCNRFilterAttr	249
15.2.4	CVI_ISP_GetCNRFilterAttr	250
15.3	数据类型	250
15.3.1	ISP_CNR_MANUAL_ATTR_S	251
15.3.2	ISP_CNR_AUTO_ATTR_S	253
15.3.3	ISP_CNR_ATTR_S	256
15.3.4	ISP_CNR_FILTER_MANUAL_ATTR_S	257
15.3.5	ISP_CNR_FILTER_AUTO_ATTR_S	260
15.3.6	ISP_CNR_FILTER_ATTR_S	262
16	TNR	264
16.1	功能描述	264
16.2	API 参考	264
16.2.1	CVI_ISP_SetTNRAttr	264
16.2.2	CVI_ISP_GetTNRAttr	265
16.2.3	CVI_ISP_SetTNRMvAttr	266
16.2.4	CVI_ISP_GetTNRMvAttr	267
16.2.5	CVI_ISP_SetTNRPsAttr	267
16.2.6	CVI_ISP_GetTNRPsAttr	268
16.2.7	CVI_ISP_SetTNRNrAttr	269
16.2.8	CVI_ISP_GetTNRNrAttr	270
16.3	数据类型	271
16.3.1	ISP_TNR_MANUAL_ATTR_S	271
16.3.2	ISP_TNR_AUTO_ATTR_S	272
16.3.3	ISP_TNR_ATTR_S	272
16.3.4	ISP_TNR_MV_MANUAL_ATTR_S	273
16.3.5	ISP_TNR_MV_AUTO_ATTR_S	275
16.3.6	ISP_TNR_MV_ATTR_S	277
16.3.7	ISP_TNR_PS_MANUAL_ATTR_S	277
16.3.8	ISP_TNR_PS_AUTO_ATTR_S	278
16.3.9	ISP_TNR_PS_ATTR_S	280
16.3.10	ISP_TNR_NR_MANUAL_ATTR_S	280
16.3.11	ISP_TNR_NR_AUTO_ATTR_S	283
16.3.12	ISP_TNR_NR_ATTR_S	286
17	Crosstalk	287
17.1	功能描述	287
17.2	API 参考	287
17.2.1	CVI_ISP_SetCrosstalkAttr	287
17.2.2	CVI_ISP_GetCrosstalkAttr	288
17.3	数据类型	289
17.3.1	ISP_CROSSTALK_MANUAL_ATTR_S	289
17.3.2	ISP_CROSSTALK_AUTO_ATTR_S	290
17.3.3	ISP_CROSSTALK_ATTR_S	290
18	DEMOSAIC	292
18.1	功能描述	292
18.2	API 参考	292

18.2.1	CVI_ISP_SetDemosaicAttr	292
18.2.2	CVI_ISP_GetDemosaicAttr	293
18.2.3	CVI_ISP_SetDemosaicDemoireAttr	294
18.2.4	CVI_ISP_GetDemosaicDemoireAttr	295
18.3	数据类型	296
18.3.1	ISP_DEMOSAIC_MANUAL_ATTR_S	296
18.3.2	ISP_DEMOSAIC_AUTO_ATTR_S	297
18.3.3	ISP_DEMOSAIC_ATTR_S	298
18.3.4	ISP_DEMOSAIC_DEMOIRE_MANUAL_ATTR_S	299
18.3.5	ISP_DEMOSAIC_DEMOIRE_AUTO_ATTR_S	302
18.3.6	ISP_DEMOSAIC_DEMOIRE_ATTR_S	305
19	SHARPEN	307
19.1	功能描述	307
19.2	API 参考	307
19.2.1	CVI_ISP_SetSharpenAttr	307
19.2.2	CVI_ISP_GetSharpenAttr	308
19.3	数据类型	309
19.3.1	ISP_SHARPEN_MANUAL_ATTR_S	309
19.3.2	ISP_SHARPEN_AUTO_ATTR_S	311
19.3.3	ISP_SHARPEN_ATTR_S	313
20	PRESHARPEN	315
20.1	功能描述	315
20.2	API 参考	315
20.2.1	CVI_ISP_SetPreSharpenAttr	315
20.2.2	CVI_ISP_GetPreSharpenAttr	316
20.2.3	CVI_ISP_SetPreSharpenRefineAttr	317
20.2.4	CVI_ISP_GetPreSharpenRefineAttr	318
20.2.5	CVI_ISP_SetPreSharpenEdgeExtAttr	319
20.2.6	CVI_ISP_GetPreSharpenEdgeExtAttr	320
20.3	数据类型	320
20.3.1	ISP_PRESHARPEN_MANUAL_ATTR_S	321
20.3.2	ISP_PRESHARPEN_AUTO_ATTR_S	323
20.3.3	ISP_PRESHARPEN_ATTR_S	325
20.3.4	ISP_PRESHARPEN_REFINE_MANUAL_ATTR_S	327
20.3.5	ISP_PRESHARPEN_REFINE_AUTO_ATTR_S	330
20.3.6	ISP_PRESHARPEN_REFINE_ATTR_S	334
20.3.7	ISP_PRESHARPEN_EDGE_EXT_MANUAL_ATTR_S	334
20.3.8	ISP_PRESHARPEN_EDGE_EXT_AUTO_ATTR_S	337
20.3.9	ISP_PRESHARPEN_EDGE_EXT_ATTR_S	339
21	RGBGAMMA	341
21.1	功能描述	341
21.2	API 参考	341
21.2.1	CVI_ISP_SetGammaAttr	341
21.2.2	CVI_ISP_GetGammaAttr	342
21.3	数据类型	343
21.3.1	ISP_GAMMA_ATTR_S	343
22	DCI	344

22.1	功能描述	344
22.2	API 参考	344
22.2.1	CVI_ISP_SetDCIAttr	344
22.2.2	CVI_ISP_GetDCIAttr	345
22.2.3	CVI_ISP_SetDciAutoGammaAttr	346
22.2.4	CVI_ISP_GetDciAutoGammaAttr	347
22.3	数据类型	347
22.3.1	ISP_DCI_ATTR_S	348
22.3.2	ISP_DCI_AUTO_GAMMA_ATTR_S	349
22.3.3	ISP_DCI_GAMMA_CURVE_ATTR_S	350
23	LDCI	351
23.1	功能描述	351
23.2	API 参考	351
23.2.1	CVI_ISP_SetLDCIAttr	351
23.2.2	CVI_ISP_GetLDCIAttr	352
23.3	数据类型	353
23.3.1	ISP_LDCI_MANUAL_ATTR_S	353
23.3.2	ISP_LDCI_AUTO_ATTR_S	354
23.3.3	ISP_LDCI_ATTR_S	355
24	ColorTone	356
24.1	功能描述	356
24.2	API 参考	356
24.2.1	CVI_ISP_SetColorToneAttr	356
24.2.2	CVI_ISP_GetColorToneAttr	357
24.3	数据类型	358
24.3.1	ISP_COLOR_TONE_ATTR_S	358
25	Saturation	359
25.1	功能描述	359
25.2	API 参考	359
25.2.1	CVI_ISP_SetSaturationAttr	359
25.2.2	CVI_ISP_GetSaturationAttr	360
25.3	数据类型	361
25.3.1	ISP_SATURATION_MANUAL_ATTR_S	361
25.3.2	ISP_SATURATION_AUTO_ATTR_S	361
25.3.3	ISP_SATURATION_ATTR_S	362
26	PFR	363
26.1	功能描述	363
26.2	API 参考	363
26.2.1	CVI_ISP_SetPFRAttr	363
26.2.2	CVI_ISP_GetPFRAttr	364
26.3	数据类型	365
26.3.1	ISP_PFR_MANUAL_ATTR_S	365
26.3.2	ISP_PFR_AUTO_ATTR_S	366
26.3.3	ISP_PFR_ATTR_S	366
27	FSHDR	370
27.1	功能描述	370

27.2	API 参考	370
27.2.1	CVI_ISP_SetFSHDRAttr	370
27.2.2	CVI_ISP_GetFSHDRAttr	371
27.3	数据类型	372
27.3.1	ISP_FSHDR_ATTR_S	372
28	DRC	377
28.1	功能描述	377
28.2	API 参考	377
28.2.1	CVI_ISP_SetDRCAAttr	377
28.2.2	CVI_ISP_GetDRCAAttr	378
28.3	数据类型	379
28.3.1	ISP_DRC_MANUAL_ATTR_S	379
28.3.2	ISP_DRC_AUTO_ATTR_S	380
28.3.3	ISP_DRC_ATTR_S	380
29	MONO	384
29.1	功能描述	384
29.2	API 参考	384
29.2.1	CVI_ISP_SetMonoAttr	384
29.2.2	CVI_ISP_GetMonoAttr	385
29.3	数据类型	386
29.3.1	ISP_MONO_ATTR_S	386
30	YCONTRAST	387
30.1	功能描述	387
30.2	API 参考	387
30.2.1	CVI_ISP_SetYContrastAttr	387
30.2.2	CVI_ISP_GetYContrastAttr	388
30.3	数据类型	389
30.3.1	ISP_YCONTRAST_MANUAL_ATTR_S	389
30.3.2	ISP_YCONTRAST_AUTO_ATTR_S	390
30.3.3	ISP_YCONTRAST_ATTR_S	390
31	CA	392
31.1	功能描述	392
31.2	API 参考	392
31.2.1	CVI_ISP_SetCAAttr	392
31.2.2	CVI_ISP_GetCAAttr	393
31.3	数据类型	394
31.3.1	ISP_CA_MANUAL_ATTR_S	394
31.3.2	ISP_CA_AUTO_ATTR_S	395
31.3.3	ISP_CA_ATTR_S	395
32	CA2	397
32.1	功能描述	397
32.2	API 参考	397
32.2.1	CVI_ISP_SetCA2Attr	397
32.2.2	CVI_ISP_GetCA2Attr	398
32.3	数据类型	399
32.3.1	ISP_CA2_MANUAL_ATTR_S	399

32.3.2	ISP_CA2_AUTO_ATTR_S	400
32.3.3	ISP_CA2_ATTR_S	400
33	CLUT	402
33.1	功能描述	402
33.2	API 参考	402
33.2.1	CVI_ISP_SetClutAttr	402
33.2.2	CVI_ISP_GetClutAttr	403
33.2.3	CVI_ISP_SetClutHslAttr	404
33.2.4	CVI_ISP_GetClutHslAttr	405
33.3	数据类型	405
33.3.1	ISP_CLUT_ATTR_S	406
33.3.2	ISP_CLUT_HSL_ATTR_S	407
34	CSC	408
34.1	功能描述	408
34.2	API 参考	408
34.2.1	CVI_ISP_SetCSCAttr	408
34.2.2	CVI_ISP_GetCSCAttr	409
34.3	数据类型	410
34.3.1	ISP_CSC_ATTR_S	410
34.3.2	ISP_CSC_MATRX_S	411
35	VC	412
35.1	功能描述	412
35.2	API 参考	412
35.2.1	CVI_ISP_SetVCAttr	412
35.2.2	CVI_ISP_GetVCAttr	413
35.3	数据类型	414
35.3.1	ISP_VC_ATTR_S	414
36	统计讯息	415
36.1	概述	415
36.2	API 参考	415
36.2.1	CVI_ISP_SetStatisticsConfig	415
36.2.2	CVI_ISP_GetStatisticsConfig	418
36.2.3	CVI_ISP_GetAESTatistics	418
36.2.4	CVI_ISP_GetWBStatistics	420
36.2.5	CVI_ISP_GetFocusStatistics	421
36.3	数据类型	421
36.3.1	ISP_STATISTICS_CTRL_U	421
36.3.2	ISP_AE_STATISTICS_CFG_S	422
36.3.3	ISP_AE_CROP_S	423
36.3.4	ISP_AE_FACE_CROP_S	424
36.3.5	ISP_WB_STATISTICS_CFG_S	424
36.3.6	ISP_AWB_CROP_S	425
36.3.7	ISP_WB_STATISTICS_S	426
36.3.8	ISP_AWB_GRID_INFO_S	427
36.3.9	ISP_FOCUS_STATISTICS_CFG_S	428
36.3.10	ISP_AF_CFG_S	429
36.3.11	ISP_AF_RAW_CFG_S	430

36.3.12	ISP_AF_PRE_FILTER_CFG_S	430
36.3.13	ISP_AF_CROP_S	431
36.3.14	ISP_AF_H_PARAM_S	432
36.3.15	ISP_AF_V_PARAM_S	432
36.3.16	ISP_STATISTICS_CFG_S	433
36.3.17	ISP_FOCUS_ZONE_S	433
36.3.18	ISP_FE_FOCUS_STATISTICS_S	434
36.3.19	ISP_AF_STATISTICS_S	434
37	查询内部状态消息	436
37.1	概述	436
37.2	API 参考	436
37.2.1	CVI_ISP_QueryInnerStateInfo	436
37.3	数据类型	437
37.3.1	ISP_INNER_STATE_INFO_S	437
38	Debug	439
39	错误码	440
39.1	Proc 调试信息说明	440
39.2	概述	440
39.3	使用方法	440
39.4	ISP	441
39.4.1	LEVEL1 级别调试信息分析	441
39.4.2	LEVEL2 级别调试信息分析	454
39.4.3	LEVEL3 级别调试信息分析	455
40	3A 开发指南概述	457
41	3A 开发用户指南	458
41.1	AF 统计信息使用说明	458
41.1.1	概述	458
41.1.2	输入图像的裁剪	458
41.1.3	Bayer 域的配置	459
41.1.4	抑制光源对于 FV 值的影响	459
41.1.5	统计信息配置注意事项	459
41.1.6	FV 值的获取	459
41.1.7	FV 值的计算	459
41.1.8	FV 计算参考代码	460
42	开发者指南	463
42.1	概述	463
42.2	API 参考	463
42.2.1	isp_sync_task_register	463
42.2.2	isp_sync_task_unregister	464
42.3	数据类型	465
42.3.1	isp_sync_tsk_method	465
42.3.2	isp_sync_task_node	466
43	附录	467
44	缩略语	468

修订记录

Revision	Date	Description
0.1.0	2025/04/25	初稿

1 声明



法律声明

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

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

联系我们

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

深圳市宝安区福海街道展城社区会展湾云岸广场 T10 栋

电话 +86-10-57590723 +86-10-57590724

邮编 100094（北京）518100（深圳）

官方网站 <https://www.sophgo.com/>

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

2 概述

2.1 概述

本文件主要介绍 ISP 的用户接口。可分为系统控制、3A、ISP 模块等三大份。第一部分为系统控制，说明如何控制 ISP middleware。第二部分为 3A，说明如何控制 AE、AWB、AF。本区功能大多可于 Cvi PqTool 中进行调试。第三部分为 ISP 各个模块，说明如何控制 Black Level Cancellation、Color Correction Matrix、Gamma、Noise Reduction、Sharpeness 等模块。本区功能大多可于 Cvi PqTool 中进行调试。

2.2 功能描述

ISP 的控制结构如 图 2.1 所示

1. Lens - 投聚焦光信号，射到 sensor 的感光区域
2. sensor - 经过光电转换，将 Bayer 格式的原始图像送给 ISP
3. ISP - 在此处理 ISP 通过运行在其上的 firmware 对 ISP 逻辑，lens 和 sensor 进行相应控制，进而完成自动光圈、自动曝光、自动白平衡等功能。其中，firmware 的运转靠视频采集单元的中断驱动。PQ Tools 工具通过网口或者串口完成对 ISP 的在线图像质量调节。输出 YUV 域的图像给后端的视频采集单元。

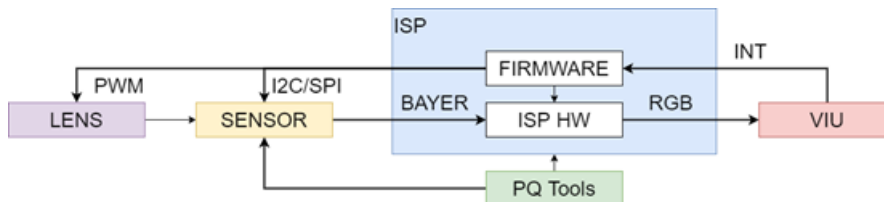


图 2.1: ISP 控制结构示意图

ISP 逻辑主要流程、具体概念和功能点请参见处理器手册。

2.3 架构

ISP 的 Firmware 可分为三部分。第一部分是 ISP 控制单元和基础算法库，第二部分是 3A 算法库，目前包括 AE 及 AWB。第三部分是 sensor 库。软件架构区分为此三大部分，并且透过注册函数回调以达到可以独立演进的目的。例如开发者自行实作 3A 函数，只要实作相同的接口，并且注册即可代换 Cvi 预设的 3A 库。ISP firmware 架构如 图 2.2 所示

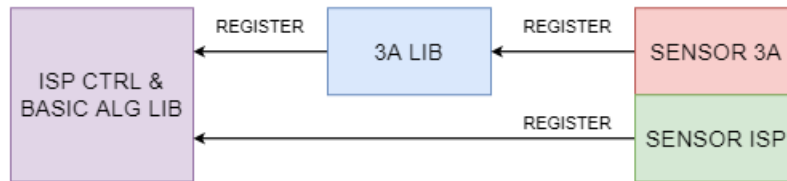


图 2.2: ISP firmware 架构

不同的 sensor 都以回调函数的形式，向 ISP 算法库注册控制函数。ISP 控制单元调度基础算法库和 3A 算法库时，将通过这些回调函数获取初始化参数，并控制 sensor，如调节模拟增益、数字增益、曝光时间。

2.4 开发模式

SDK 支持用户使用多种开发模式：

用户使用晶视智能的 3A 算法库。这时用户需要根据 ISP 基础算法库和 3A 算法库给出的 sensor 适配接口去适配不同的 sensor。每款 sensor 对应一个文件夹，文件夹中包含两个主要文件：

- sensor_cmos.c

该主要实现 ISP 需要的回调函数，包含了 sensor 的适配算法。不同的 sensor 可能有所不同。

- sensor_ctrl.c

sensor 的底层控制驱动，主要实现 sensor 的读写和初始化动作。

为了可以同时兼容多个 sensor，所以以上两个文件档名会加入 sensor 型号，例如 Sony imx307 会命名为 imx307_cmos.c、imx307_ctrl.c。

用户可以根据 sensor 的 datasheet 进行这两个文件的开发，必要的时候可以向 sensor 厂家寻求支持。

用户可根据 ISP 库提供的 3A 算法注册接口，实现自己的 3A 算法库开发。这时用户需要根据 ISP 基础算法库和用户的 3A 算法库给出的 sensor 适配接口去适配不同的 sensor。用户可以部分使用晶视智能 3A 算法库，部分实现自己的 3A 算法库。例如 AE 使用晶视智能 AE 算法库 libae.a，AWB 使用自己的 libawb.a 算法库。

2.5 内部流程

Firmware 内部流程分两部分，如 图 2.3 所示。一部分是初始化任务，主要完成 ISP 控制单元的初始化、ISP 基础算法库的初始化、3A 算法库的初始化，包括调用 sensor 的回调获取 sensor 差异化的初始化参数；另一部分是动态调节过程，在这个过程中，firmware 中的 ISP 控制单元调度 ISP 基础算法库和 3A 算法库，实时计算并进行相应控制。Firmware 的软件结构如 图 2.4 所示。

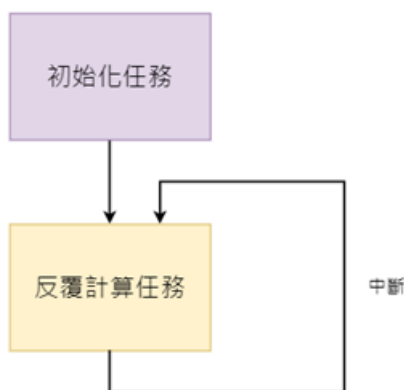


图 2.3: ISP firmware 内部流程

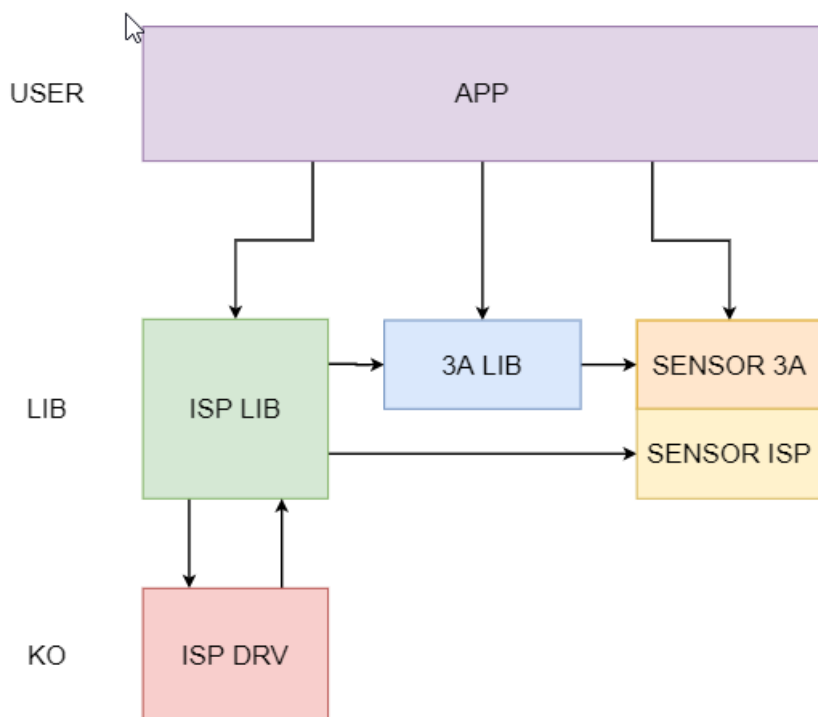


图 2.4: ISP firmware 软件结构

2.6 软件流程

软件使用流程如 图 2.5 所示。

PQ Tools 工具主要完成在 PC 端进行动态图像质量调节，可以调节多个影响图像质量的因子，如去噪强度、色彩转换矩阵、饱和度等。

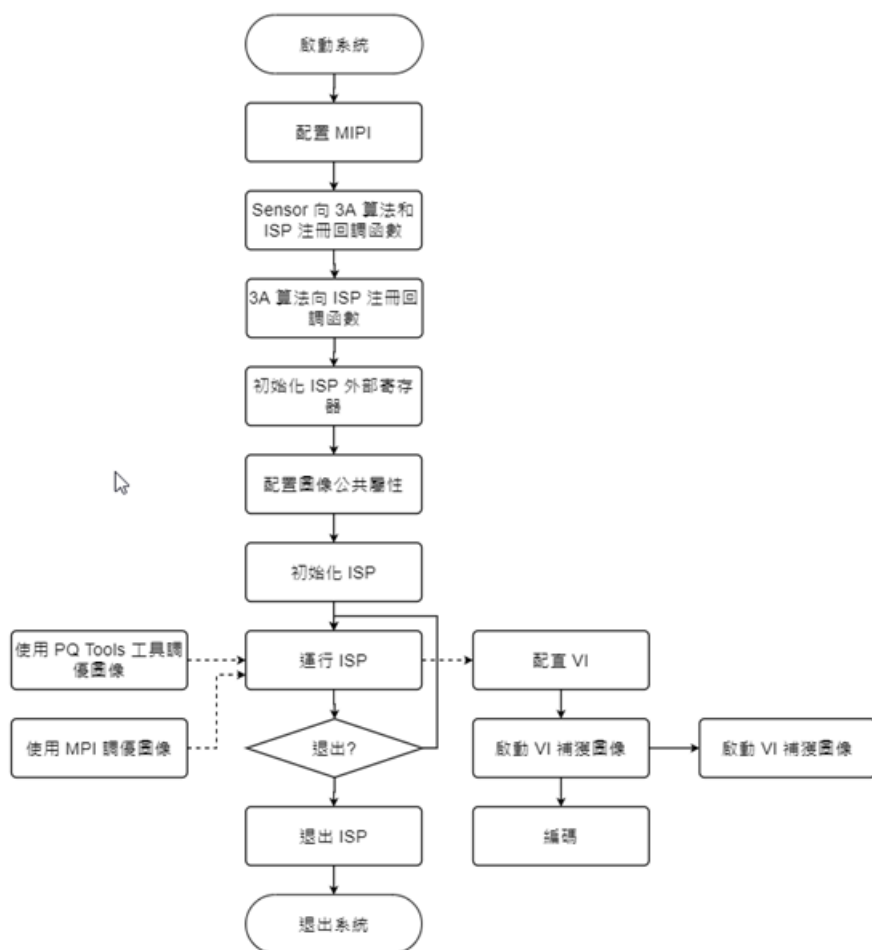


图 2.5: 软件使用流程

如果用户调试好图像效果后，可以使用 PQ Tools 工具提供的配置文件保存功能进行配置参数的保存。在下次启动时系统可以使用 PQ Tools 工具提供的配置文件加载功能加载已经调节好的图像参数。

3 系统控制

3.1 功能概述

系统控制部分包含了 ISP 公共属性配置，初始化 ISP Firmware、运行 ISP firmware、退出 ISP firmware，设置 ISP 各模块等功能。

3.2 API 参考

本文中接口，如无特殊说明，不支持多进程。

- `CVI_ISP_MemInit`：初始化 ISP 外部寄存器。
- `CVI_ISP_Init`：初始化 ISP firmware。
- `CVI_ISP_Run`：运行 ISP firmware。
- `CVI_ISP_RunOnce`：运行 ISP firmware 一次。
- `CVI_ISP_Exit`：退出 ISP firmware。
- `CVI_ISP_SetPubAttr`：设置 ISP 公共属性。
- `CVI_ISP_GetPubAttr`：获取 ISP 公共属性。
- `CVI_ISP_SetFMWState`：设置 ISP firmware 状态。
- `CVI_ISP_GetFMWState`：获取 ISP firmware 状态。
- `CVI_ISP_SetModuleControl`：设定 ISP 功能模块的控制。
- `CVI_ISP_GetModuleControl`：获取 ISP 功能模块的控制。
- `CVI_ISP_GetVDTimeOut`：获取 ISP 中断信息。
- `CVI_ISP_SensorRegCallBack`：ISP 提供的 sensor 注册的回调接口。
- `CVI_ISP_SensorUnRegCallBack`：ISP 提供的 sensor 反注册的回调接口。
- `CVI_ISP_AELibRegCallBack`：ISP 提供的 AE 库注册的回调接口。
- `CVI_ISP_AELibUnRegCallBack`：ISP 提供的 AE 库反注册的回调接口。
- `CVI_ISP_AWBLibRegCallBack`：ISP 提供的 AWB 库注册的回调接口。
- `CVI_ISP_AWBLibUnRegCallBack`：ISP 提供的 AWB 库反注册的回调接口。

- `CVI_ISP_AFLibRegCallBack` : ISP 提供的 AF 库注册的回调接口。
- `CVI_ISP_AFLibUnRegCallBack` : ISP 提供的 AF 库反注册的回调接口。
- `CVI_ISP_SetBindAttr` : 设置 ISP 库与 3A 库、sensor 的绑定关系。
- `CVI_ISP_GetBindAttr` : 获取 ISP 库与 3A 库、sensor 的绑定关系。
- `CVI_ISP_SetCtrlParam` : 设置 ISP 的控制参数。
- `CVI_ISP_GetCtrlParam` : 获取 ISP 的控制参数。
- `CVI_ISP_SetModParam` : 设置 ISP 模块参数。
- `CVI_ISP_GetModParam` : 获取 ISP 模块参数。
- `CVI_BIN_SetBinName` : 设置 PQBIN 存放的路径和文件名
- `CVI_BIN_GetBinName` : 获取 PQBIN 存放的路径和文件名
- `CVI_BIN_GetBinExtraAttr` : 获取 bin 头数据信息。
- `CVI_BIN_GetBinTotalLen` : 获取 bin 数据的总长度。
- `CVI_BIN_ExportBinData` : 将参数存入 PQBin 档。
- `CVI_BIN_ImportBinData` : 从 PQBin 中解析所有模块的数据。
- `CVI_BIN_SaveParamToBin` : 将参数存入 PQBin 档。
- `CVI_BIN_LoadParamFromBin` : 从 PQBin 中解析所有模块的数据。
- `CVI_ISP_IrAutoRunOnce` : 运行红外自动切换功能。
- `CVI_ISP_SetSmartInfo` : 设置智能识别区域信息。
- `CVI_ISP_GetSmartInfo` : 获取智能识别区域信息。
- `CVI_ISP_SetStitchAttr` : 设置 ISP 拼接参数。
- `CVI_ISP_GetStitchAttr` : 获取 ISP 拼接参数。
- `CVI_ISP_StitchCalibartion` : ISP 拼接标定。

3.2.1 CVI_ISP_MemInit

【描述】

初始化 ISP 外部寄存器。

【语法】

```
CVI_S32 CVI_ISP_MemInit(VI_PIPE ViPipe);
```

【参数】

参数名称	描述
ViPipe	ViPipe 号

[返回值]

返回值	描述
0	成功。
非 0	失败，请参见 错误码 。

【需求】

- 头文件：cvi_isp.h
- 库文件：libisp.so

【注意】

- 此接口在双系统 SDK 的 linux 侧暂不支持。

【举例】

无。

【相关主题】

- [CVI_ISP_Exit](#)

3.2.2 CVI_ISP_Init

【描述】

初始化 ISP firmware。

【语法】

```
CVI_S32 CVI_ISP_Init(VI_PIPE ViPipe);
```

【参数】

参数名称	描述
ViPipe	ViPipe 号

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 错误码 。

【需求】

- 头文件：cvi_isp.h
- 库文件：libisp.a

【注意】

无。

【举例】

无。

【相关主题】

- [CVI_ISP_Exit](#)

3.2.3 CVI_ISP_Run

【描述】

运行 ISP firmware。

【语法】

```
CVI_S32 CVI_ISP_Run(VI_PIPE ViPipe);
```

【参数】

参数名称	描述
ViPipe	ViPipe 号

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_isp.h
- 库文件：libisp.a

【注意】

- 此接口在双系统 SDK 的 linux 侧暂不支持。

【举例】

无。

【相关主题】

无。

3.2.4 CVI_ISP_RunOnce

【描述】

运行 ISP firmware 一次。

【语法】

```
CVI_S32 CVI_ISP_RunOnce(VI_PIPE ViPipe);
```

【参数】

参数名称	描述
ViPipe	ViPipe 号

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_isp.h
- 库文件：libisp.a

【注意】

- 此接口在双系统 SDK 的 linux 侧暂不支持。

【举例】

无。

【相关主题】

无。

3.2.5 CVI_ISP_Exit

【描述】

退出 ISP firmware。

【语法】

```
CVI_S32 CVI_ISP_Exit(VI_PIPE ViPipe);
```

【参数】

参数名称	描述
ViPipe	ViPipe 号

返回值】

返回值	描述
0	成功。
非 0	失败，请参见 错误码 。

【需求】

- 头文件：cvi_isp.h
- 库文件：libisp.a

【注意】

无。

【举例】

无。

【相关主题】

- [CVI_ISP_Init](#)

3.2.6 CVI_ISP_SetPubAttr

【描述】

设置 ISP 公共属性。

【语法】

```
CVI_S32 CVI_ISP_SetPubAttr(VI_PIPE ViPipe, const ISP_PUB_ATTR_S *pstPubAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstPubAttr	ISP 公共属性。	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 错误码 。

【需求】

- 头文件：cvi_isp.h

- 库文件: libisp.a

【注意】

无。

【举例】

无。

【相关主题】

- [CVI_ISP_GetPubAttr](#)

3.2.7 CVI_ISP_GetPubAttr

【描述】

获取 ISP 公共属性。

【语法】

```
CVI_S32 CVI_ISP_GetPubAttr(VI_PIPE ViPipe, ISP_PUB_ATTR_S *pstPubAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstPubAttr	ISP 公共属性。	输出

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 错误码 。

【需求】

- 头文件: cvi_isp.h
- 库文件: libisp.a

【注意】

无。

【举例】

无。

【相关主题】

- [CVI_ISP_SetPubAttr](#)

3.2.8 CVI_ISP_SetFMWState

【描述】

设置 ISP firmware 状态。

【语法】

```
CVI_S32 CVI_ISP_SetFMWState(VI_PIPE ViPipe, const ISP_FMW_STATE_E enState);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
enState	ISP firmware 状态。	输入

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见错误码。

【需求】

- 头文件: cvi_isp.h
- 库文件: libisp.a

【注意】

无。

【举例】

无。

【相关主题】

- CVI_ISP_GetFMWState

3.2.9 CVI_ISP_GetFMWState

【描述】

获取 ISP firmware 状态。

【语法】

```
CVI_S32 CVI_ISP_GetFMWState(VI_PIPE ViPipe, ISP_FMW_STATE_E *penState);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
enState	ISP firmware 状态。	输出

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_isp.h
- 库文件：libisp.a

【注意】

无。

【举例】

无。

【相关主题】

- [CVI_ISP_SetFMWState](#)

3.2.10 CVI_ISP_SetModuleControl

【描述】

设定 ISP 功能模块的控制。

【语法】

```
CVI_S32 CVI_ISP_SetModuleControl(VI_PIPE ViPipe, const ISP_MODULE_CTRL_U_
↪ *punModCtrl);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
punModCtrl	ISP 功能模块的控制	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件: `cvi_isp.h`
- 库文件: `libisp.a`

【注意】

- 此接口在双系统 SDK 的 linux 侧暂不支持。

【举例】

无。

【相关主题】

- [CVI_ISP_GetModuleControl](#)

3.2.11 CVI_ISP_GetModuleControl

【描述】

获取 ISP 功能模块的控制。

【语法】

```
CVI_S32 CVI_ISP_GetModuleControl(VI_PIPE ViPipe, ISP_MODULE_CTRL_U *punModCtrl);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
punModCtrl	ISP 功能模块的控制。	输出

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 错误码 。

【需求】

- 头文件: `cvi_isp.h`
- 库文件: `libisp.a`

【注意】

- 此接口在双系统 SDK 的 linux 侧暂不支持。

【举例】

无。

【相关主题】

- [CVI_ISP_SetModuleControl](#)

3.2.12 CVI_ISP_GetVDTimeOut

【描述】

获取 ISP 中断信息。

【语法】

```
CVI_S32 CVI_ISP_GetVDTimeOut(VI_PIPE ViPipe, ISP_VD_TYPE_E_ enIspVDType, CVI_
→ U32 u32MilliSec);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
enIspVDType	场同步信号。	输入/输出
u32MilliSec	超时时间，单位 ms。	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_isp.h
- 库文件：libisp.a

【注意】

无。

【举例】

无。

【相关主题】

无。

3.2.13 CVI_ISP_SensorRegCallBack

【描述】

ISP 提供的 sensor 注册的回调接口。

【语法】

```
CVI_S32 CVI_ISP_SensorRegCallBack(VI_PIPE ViPipe, ISP_SNS_ATTR_INFO_S_
→ *pstSnsAttrInfo, ISP_SENSOR_REGISTER_S *pstRegister);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstSnsAttrInfo	向 ISP 注册的 Sensor 的属性	输入/输出
pstRegister	Sensor 注册结构体指针	输入/输出

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_isp.h
- 库文件：libisp.a

【注意】

- 此接口在双系统 SDK 的 linux 侧暂不支持。

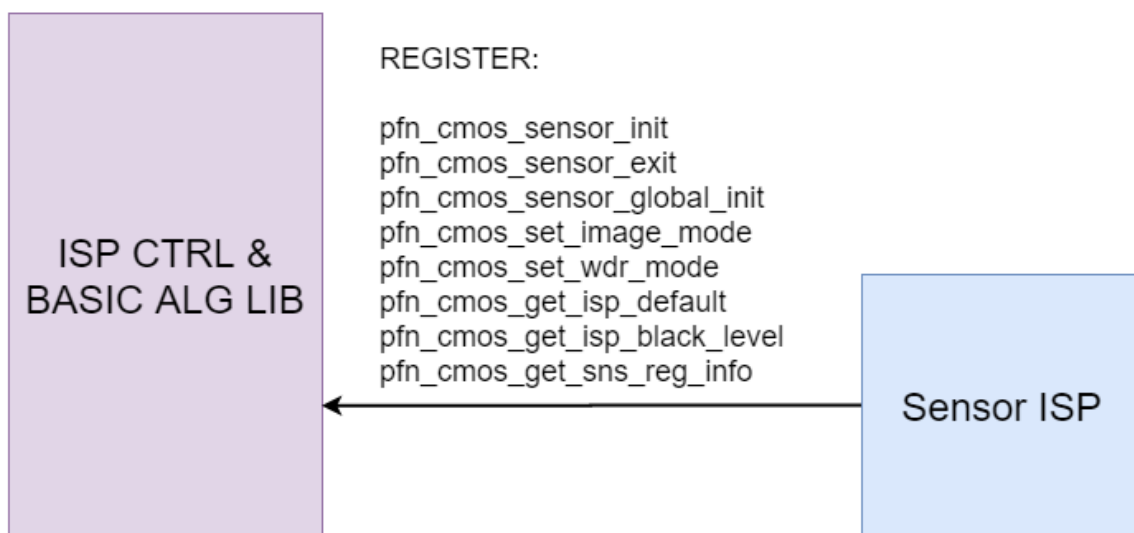


图 3.1: ISP 库与 sensor 库间的接口

【举例】

无。

【相关主题】

- [CVI_ISP_SensorUnRegCallback](#)

3.2.14 CVI_ISP_SensorUnRegCallBack

【描述】

ISP 提供的 sensor 反注册的回调接口。

【语法】

```
CVI_S32 CVI_ISP_SensorUnRegCallBack(VI_PIPE ViPipe, SENSOR_ID SensorId);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
SensorId	向 ISP 注册的 Sensor 的 Id。	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_isp.h
- 库文件：libisp.a

【注意】

- 此接口在双系统 SDK 的 linux 侧暂不支持。

【举例】

无。

【相关主题】

- [CVI_ISP_SensorRegCallBack](#)

3.2.15 CVI_ISP_AELibRegCallBack

【描述】

ISP 提供的 AE 库注册的回调接口。

【语法】

```
CVI_S32 CVI_ISP_AELibRegCallBack(VI_PIPE ViPipe, ALG_LIB_S *pstAeLib, ISP_AE_
→REGISTER_S *pstRegister);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstAeLib	AE 库结构体指针	输入
pstRegister	AE 库注册结构体指针	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 错误码 。

【需求】

- 头文件：cvi_isp.h
- 库文件：libisp.a

【注意】

- 此接口在双系统 SDK 的 linux 侧暂不支持。

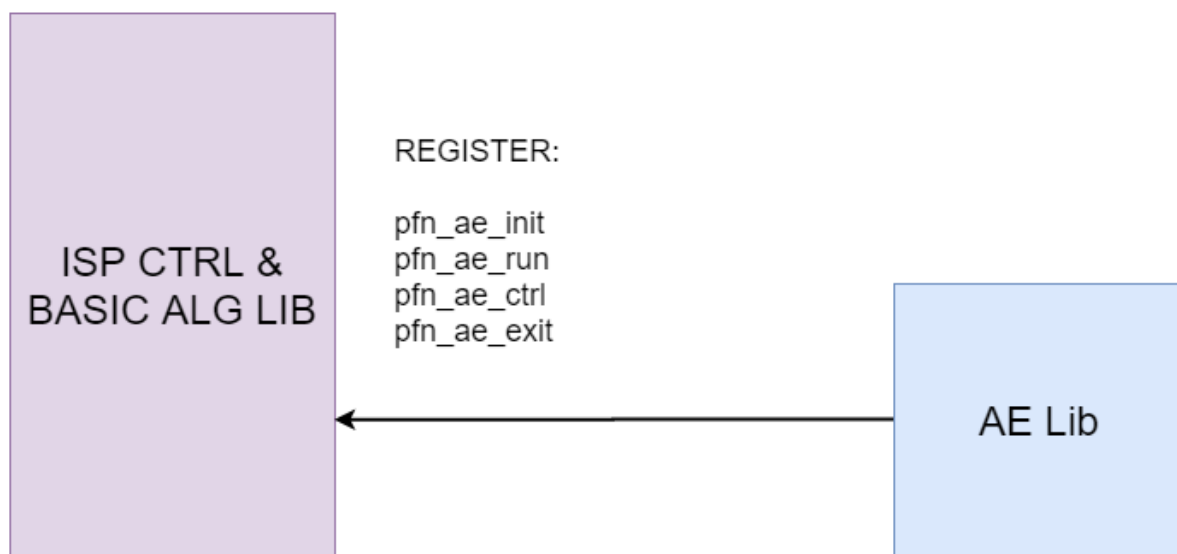


图 3.2: ISP 库与 AE 库间的接口

【举例】

无。

【相关主题】

- [CVI_ISP_AELibUnRegCallBack](#)

3.2.16 CVI_ISP_AELibUnRegCallBack

【描述】

ISP 提供的 AE 库反注册的回调接口。

【语法】

```
CVI_S32 CVI_ISP_AELibUnRegCallBack(VI_PIPE ViPipe, ALG_LIB_S *pstAeLib);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstAeLib	AE 库结构体指针	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_isp.h
- 库文件：libisp.a

【注意】

- 此接口在双系统 SDK 的 linux 侧暂不支持。

【举例】

无。

【相关主题】

- [CVI_ISP_AELibRegCallBack](#)

3.2.17 CVI_ISP_AWBLibRegCallBack

【描述】

ISP 提供的 AWB 库注册的回调接口。

【语法】

```
CVI_S32 CVI_ISP_AWBLibRegCallBack(VI_PIPE ViPipe, ALG_LIB_S *pstAwbLib, ISP_AWB_
→REGISTER_S *pstRegister);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstAwbLib	AWB 库结构体指针	输入
pstRegister	AWB 库注册结构体指针	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 错误码 。

【需求】

- 头文件：cvi_isp.h
- 库文件：libisp.a

【注意】

- 此接口在双系统 SDK 的 linux 侧暂不支持。

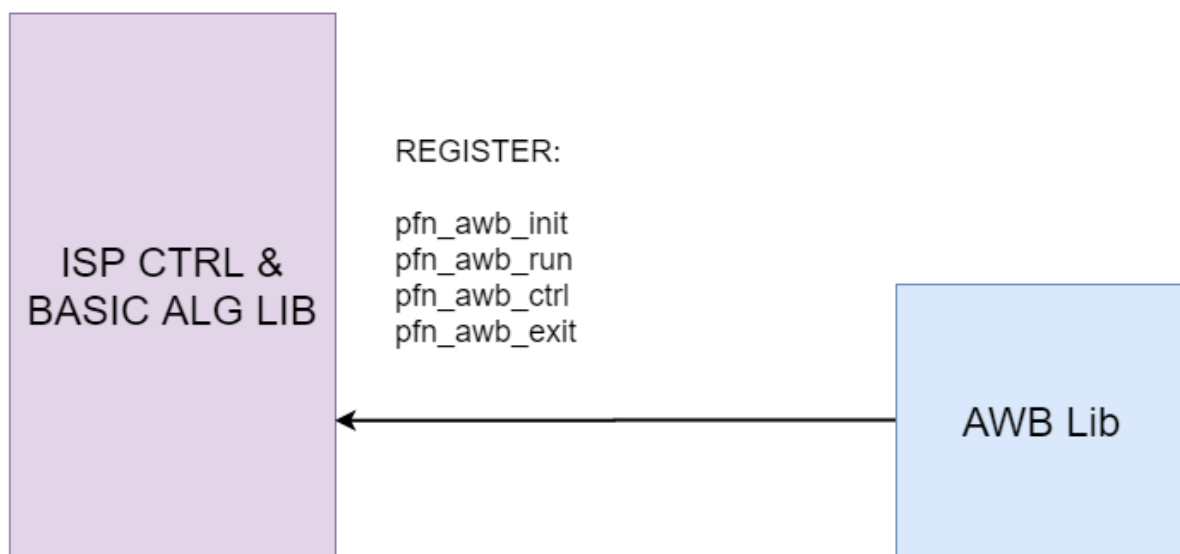


图 3.3: ISP 库与 AWB 库间的接口

【举例】

无。

【相关主题】

- CVI_ISP_AWBLibUnRegCallBack

3.2.18 CVI_ISP_AWBLibUnRegCallBack

【描述】

ISP 提供的 AWB 库反注册的回调接口。

【语法】

```
CVI_S32 CVI_ISP_AWBLibUnRegCallBack(VI_PIPE ViPipe, ALG_LIB_S *pstAwbLib);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstAwbLib	AWB 库结构体指针	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_isp.h
- 库文件：libisp.a

【注意】

- 此接口在双系统 SDK 的 linux 侧暂不支持。

【举例】

无。

【相关主题】

- [CVI_ISP_AWBLibRegCallBack](#)

3.2.19 CVI_ISP_AFLibRegCallBack

【描述】

ISP 提供的 AF 库注册的回调接口。

【语法】

```
CVI_S32 CVI_ISP_AFLibRegCallBack(VI_PIPE ViPipe, ALG_LIB_S *pstAfLib, ISP_AF_
↪REGISTER_S *pstRegister);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstAfLib	AF 库结构体指针	输入
pstRegister	AF 库注册结构体指针	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_isp.h
- 库文件：libisp.a

【注意】

- 此接口在双系统 SDK 的 linux 侧暂不支持。

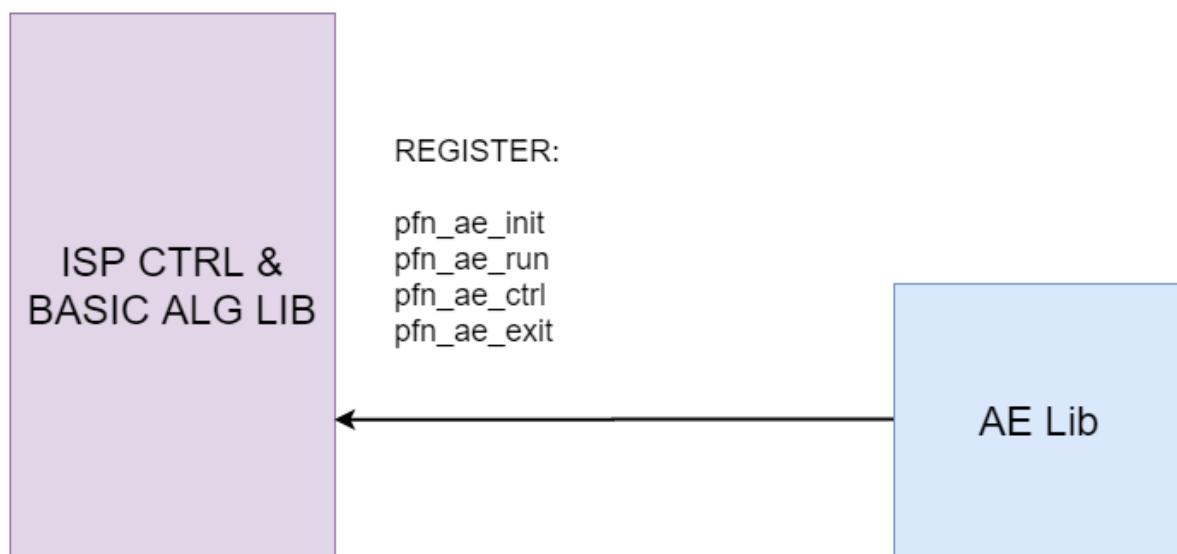


图 3.4: ISP 库与 AF 库间的接口

【举例】

无。

【相关主题】

- CVI_ISP_AFLibUnRegCallBack

3.2.20 CVI_ISP_AFLibUnRegCallBack

【描述】

ISP 提供的 AF 库反注册的回调接口。

【语法】

```
CVI_S32 CVI_ISP_AFLibUnRegCallBack(VI_PIPE ViPipe, ALG_LIB_S *pstAfLib);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstAfLib	AF 库结构体指针	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_isp.h
- 库文件：libisp.a

【注意】

- 此接口在双系统 SDK 的 linux 侧暂不支持。

【举例】

无。

【相关主题】

- [CVI_ISP_AFLibRegCallBack](#)

3.2.21 CVI_ISP_SetBindAttr

【描述】

设置 ISP 库与 3A 库、sensor 的绑定关系。

【语法】

```
CVI_S32 CVI_ISP_SetBindAttr(VI_PIPE ViPipe, const ISP_BIND_ATTR_S *pstBindAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstBindAttr	绑定结构体指针	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 错误码 。

【需求】

- 头文件：cvi_isp.h
- 库文件：libisp.a

【注意】

- 此接口在双系统 SDK 的 linux 侧暂不支持。

【举例】

无。

【相关主题】

- [CVI_ISP_GetBindAttr](#)

3.2.22 CVI_ISP_GetBindAttr

【描述】

获取 ISP 库与 3A 库、sensor 的绑定关系。

【语法】

```
CVI_S32 CVI_ISP_GetBindAttr(VI_PIPE ViPipe, ISP_BIND_ATTR_S *pstBindAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstBindAttr	绑定结构体指针	输出

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 错误码 。

【需求】

- 头文件：cvi_isp.h

- 库文件: libisp.a

【注意】

- 此接口不支持多进程操作。
- 此接口在双系统 SDK 的 linux 侧暂不支持。

【举例】

无。

【相关主题】

- [CVI_ISP_SetBindAttr](#)

3.2.23 CVI_ISP_SetCtrlParam

【描述】

设置 ISP 控制参数。

【语法】

```
CVI_S32 CVI_ISP_SetCtrlParam(VI_PIPE ViPipe, const ISP_CTRL_PARAM_S_
↪ *pstIspCtrlParam);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstIspCtrlParam	ISP 控制参数结构体指针	输入

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见 错误码 。

【需求】

- 头文件: cvi_isp.h
- 库文件: libisp.a

【注意】

无。

【举例】

无。

【相关主题】

- [CVI_ISP_GetCtrlParam](#)

3.2.24 CVI_ISP_GetCtrlParam

【描述】

获取 ISP 控制参数。

【语法】

```
CVI_S32 CVI_ISP_GetCtrlParam(VI_PIPE ViPipe, ISP_CTRL_PARAM_S *pstIspCtrlParam);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstIspCtrlParam	ISP 控制参数结构体指针。	输出

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_isp.h
- 库文件：libisp.a

【注意】

无。

【举例】

无。

【相关主题】

- [CVI_ISP_SetCtrlParam](#)

3.2.25 CVI_ISP_SetModParam

【描述】

设置 ISP 模块参数。

【语法】

```
CVI_S32 CVI_ISP_SetModParam(const ISP_MOD_PARAM_S *pstModParam);
```

【参数】

参数名称	描述	输入/输出
pstIspModParam	ISP 模块参数结构体指针	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 错误码 。

【需求】

- 头文件：cvi_isp.h
- 库文件：libisp.a

【注意】

无。

【举例】

无。

【相关主题】

- [CVI_ISP_GetModParam](#)

3.2.26 CVI_ISP_GetModParam

【描述】

获取 ISP 模块参数。

【语法】

```
CVI_S32 CVI_ISP_GetModParam( ISP_MOD_PARAM_S *pstModParam);
```

【参数】

参数名称	描述	输入/输出
pstIspCtrlParam	ISP 模块参数结构体指针	输出

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 错误码 。

【需求】

- 头文件：cvi_isp.h
- 库文件：libisp.a

【注意】

无。

【举例】

无。

【相关主题】

· [CVI_ISP_SetModParam](#)

3.2.27 CVI_BIN_SetBinName

【说明】

详情请参见文档《图像质量调试工具使用指南》的 4.1.2 章节。

3.2.28 CVI_BIN_GetBinName

【说明】

详情请参见文档《图像质量调试工具使用指南》的 4.1.2 章节。

3.2.29 CVI_BIN_GetBinExtraAttr

【说明】

详情请参见文档《图像质量调试工具使用指南》的 4.1.2 章节。

3.2.30 CVI_BIN_GetBinTotalLen

【说明】

详情请参见文档《图像质量调试工具使用指南》的 4.1.2 章节。

3.2.31 CVI_BIN_ExportBinData

【说明】

详情请参见文档《图像质量调试工具使用指南》的 4.1.2 章节。

3.2.32 CVI_BIN_ImportBinData

【说明】

详情请参见文档《图像质量调试工具使用指南》的 4.1.2 章节。

3.2.33 CVI_BIN_SaveParamToBin

【说明】

详情请参见文档《图像质量调试工具使用指南》的 4.1.2 章节。

3.2.34 CVI_BIN_LoadParamFromBin

【说明】

详情请参见文档《图像质量调试工具使用指南》的 4.1.2 章节。

3.2.35 CVI_ISP_IrAutoRunOnce

【描述】

运行红外自动切换功能。

【语法】

```
CVI_S32 CVI_ISP_IrAutoRunOnce(ISP_DEV IspDev, ISP_IR_AUTO_ATTR_S *pstIrAttr);
```

【参数】

参数名称	描述	输入/输出
IspDev	ISP 设备号	输入
pstIrAttr	红外自动切换属性	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_comm_isp.h, cvi_isp.h
- 库文件：libisp.so

【注意】

无。

【举例】

```
ISP_DEV IspDev = 0;
ISP_IR_AUTO_ATTR_S stIrAttr;

stIrAttr.bEnable = 1;
stIrAttr.u32Normal2IrIsoThr = 3200;
stIrAttr.u32Ir2NormalIsoThr = 100;
stIrAttr.u32RGMin = 256;
stIrAttr.u32RGMax = 512;
stIrAttr.u32BGMin = 256;
stIrAttr.u32BGMax = 512;
CVI_ISP_IrAutoRunOnce(IspDev, &stIrAttr);
```

【相关主题】

无。

3.2.36 CVI_ISP_SetSmartInfo

【描述】

设置 AI 辨识出的（人脸、人形、物品）坐标给 AE 进行测光。

【语法】

```
CVI_S32 CVI_ISP_SetSmartInfo(VI_PIPE ViPipe, const ISP_SMART_INFO_S *pstSmartInfo,
↪ CVI_U8 TimeOut);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ISP 设备号	输入
pstSmartInfo	AE AI 识别坐标信息结构指针	输入
TimeOut	识别坐标信息未更新帧数，超过后恢复正常 AE 模式	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_ae.h
- 库文件：libae.a

【注意】

无。

【举例】

```
//设定人脸在 raw domain坐标位置 (X, Y , W, H) =(0, 0, 100, 100)
//及 frame 的宽/高 1920/1080信息给 AE
//4帧未更新识别结果后恢复正常AE
VI_PIPE ViPipe = 0;
ISP_SMART_INFO_S stSmartInfo;

CVI_ISP_GetSmartInfo(ViPipe, &stSmartInfo);
stSmartInfo.stROI[0].bEnable = 1;
stSmartInfo.stROI[0].bAvailable = 1;
stSmartInfo.stROI[0].u8Num = 1;
stSmartInfo.stROI[0].u16PosX[0] = 0;
stSmartInfo.stROI[0].u16PosY[0] = 0;
stSmartInfo.stROI[0].u16Width[0] = 100;
stSmartInfo.stROI[0].u16Height[0] = 100;
stSmartInfo.stROI[0].u16FrameWidth = 1920;
stSmartInfo.stROI[0].u16FrameHeight = 1080;
CVI_ISP_SetSmartInfo(ViPipe, &stSmartInfo, 4);
```

【相关主题】

- [CVI_ISP_GetSmartInfo](#)

3.2.37 CVI_ISP_GetSmartInfo

【描述】

获取 AI 辨识出的（人脸、人形、物品）坐标。

【语法】

```
CVI_S32 CVI_ISP_GetSmartInfo(VI_PIPE ViPipe, ISP_SMART_INFO_S *pstSmartInfo);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ISP 设备号	输入
pstSmartInfo	AE AI 识别坐标信息结构指针	输出

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_ae.h
- 库文件：libae.a

【注意】

无。

【举例】

无。

【相关主题】

- [CVI_ISP_SetSmartInfo](#)

3.2.38 CVI_ISP_SetStitchAttr

【描述】

设置 ISP 拼接的通道信息、需要组合统计值信息以及标定信息。

【语法】

```
CVI_S32 CVI_ISP_SetStitchAttr(VI_PIPE ViPipe, ISP_STITCH_ATTR_S *pstStitchAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ISP 设备号	输入
pstStitchAttr	ISP 拼接信息结构指针	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_isp.h
- 库文件：libisp.a

【注意】

无。

【举例】

无。

【相关主题】

- [CVI_ISP_GetStitchAttr](#)

3.2.39 CVI_ISP_GetStitchAttr

【描述】

获取 ISP 拼接的通道信息、需要组合统计值信息以及标定信息。

【语法】

```
CVI_S32 CVI_ISP_GetStitchAttr(VI_PIPE ViPipe, ISP_STITCH_ATTR_S *pstStitchAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ISP 设备号	输入
pstStitchAttr	ISP 拼接信息结构指针	输出

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_isp.h
- 库文件：libisp.a

【注意】

无。

【举例】

无。

【相关主题】

- [CVI_ISP_SetStitchAttr](#)

3.2.40 CVI_ISP_StitchCalibartion

【描述】

标定获取相同拼接 group 各通道的 sensor 和镜头差异比例

【语法】

```
CVI_S32 CVI_ISP_StitchCalibartion(VI_PIPE ViPipe, ISP_STITCH_ATTR_S *pstAttr, CVI_U8 ↵  
↵chnNum);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ISP 设备号	输入
pstAttr	ISP 拼接信息结构指针	输入 & 输出
chnNum	ISP 拼接标定的通道总数	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 错误码 。

【需求】

- 头文件：cvi_isp.h
- 库文件：libisp.a

【注意】

无。

【举例】

无。

【相关主题】

- [CVI_ISP_SetStitchAttr](#)
- [CVI_ISP_GetStitchAttr](#)

3.3 数据类型

本文档中变量，如未明确指定取值范围，则默认是数据类型对应的取值范围。例如 CVI_U8 数据类型的变量取值范围为 [0, 255]。本文档中变量，如未明确指定数据精度，则默认是 1。

3.3.1 RECT_S

【说明】

定义裁剪窗口起始位置和图像宽高

【定义】

xxx

【成员】

成员名称	描述
s32X	水平方向起始位置

下页继续

表 3.33 – 续上页

成员名称	描述
s32Y	垂直方向起始位置
u32Width	图像宽度
u32Height	图像高度

【注意事项】

无。

【相关数据类型及接口】

无。

3.3.2 SIZE_S

【说明】

定义 sensor 输出的宽高。

【定义】

```
typedef struct _SIZE_S {  
    CVI_U32 u32Width;  
    CVI_U32 u32Height;  
} SIZE_S;
```

【成员】

成员名称	描述
u32Width	Sensor 输出宽度
u32Height	Sensor 输出高度

【注意事项】

无。

【相关数据类型及接口】

无。

3.3.3 ISP_BAYER_FORMAT_E

【说明】

定义了输入 ISP 图像的拜尔阵列的格式类型

【定义】

```
typedef enum _ISP_BAYER_FORMAT_E {  
    BAYER_BGGR,  
    BAYER_GBRG,
```

(下页继续)

(续上页)

```

BAYER_GRBG,
BAYER_RGGB,
//for RGBIR sensor
BAYER_GRGBI = 8,
BAYER_RGBGI,
BAYER_GBGR,
BAYER_BGRGI,
BAYER_IGRGB,
BAYER_IRGBG,
BAYER_IBGRG,
BAYER_IGBGR,
BAYER_BUTT
} ISP_BAYER_FORMAT_E;

```

【成员】

成员名称	描述
BAYER_XX	各种 bayer 阵列的格式类型, 名称标示了 pixel 以何种方式排列

【注意事项】

无。

【相关数据类型及接口】

无。

3.3.4 WDR_MODE_E

【说明】

定义了输入 ISP 运行时 sensor 的运行方式,

【定义】

```

typedef enum WDR_MODE_E {
    WDR_MODE_NONE = 0,
    WDR_MODE_BUILT_IN,
    WDR_MODE_QUDRA,

    WDR_MODE_2To1_LINE,
    WDR_MODE_2To1_FRAME,
    WDR_MODE_2To1_FRAME_FULL_RATE,

    WDR_MODE_3To1_LINE,
    WDR_MODE_3To1_FRAME,
    WDR_MODE_3To1_FRAME_FULL_RATE,
    WDR_MODE_4To1_LINE,
    WDR_MODE_4To1_FRAME,
    WDR_MODE_4To1_FRAME_FULL_RATE,
}

```

(下页继续)

(续上页)

```
WDR_MODE_MAX,
} WDR_MODE_E;
```

【成员】

成员名称	描述
WDR_MODE_NONE	线性模式。
WDR_MODE_BUILT_IN	Sensor 合成 WDR 模式。
WDR_MODE_QUDRA	Qudra 模式
WDR_MODE_2To1_LINE	2 帧合成行 WDR 模式。
WDR_MODE_2To1_FRAME	2 帧合成帧 WDR 模式。
WDR_MODE_2To1_FRAME_FULL_RATE	2 帧合成帧 WDR 全帧率模式。
WDR_MODE_3To1_LINE	3 帧合成行 WDR 模式。
WDR_MODE_3To1_FRAME	3 帧合成帧 WDR 模式。
WDR_MODE_3To1_FRAME_FULL_RATE	3 帧合成帧 WDR 全帧率模式。
WDR_MODE_4To1_LINE	4 帧合成行 WDR 模式。
WDR_MODE_4To1_FRAME	4 帧合成帧 WDR 模式。
WDR_MODE_4To1_FRAME_FULL_RATE	4 帧合成帧 WDR 全帧率模式。

【注意事项】

无。

【相关数据类型及接口】

无。

3.3.5 ISP_PUB_ATTR_S

【说明】

定义 ISP 公共属性

【定义】

```
typedef struct _ISP_PUB_ATTR_S {
    RECT_S stWndRect;
    SIZE_S stSnsSize;
    CVI_FLOAT f32FrameRate;
    ISP_BAYER_FORMAT_E enBayer;
    WDR_MODE_E enWDRMode;
    CVI_U8 u8SnsMode;
} ISP_PUB_ATTR_S;
```

【成员】

成员名称	描述
stWndRect	ISP 输出的 rect
stSnsSize	Sensor 的 image size

下页继续

表 3.37 – 续上页

成员名称	描述
f32FrameRate	Sensor 的帧率
enBayer	Sensor 的 Bayer 格式
enWDRMode	宽动态模式
u8SnsMode	用于进行 Sensor 初始化序列的选择，在分辨率和帧率相同时，配置不同的 u8SnsMode 对应不同的初始化序列； 其他情况，u8SnsMode 默认配置为 0， 可通过 stSnsSize 和 f32FrameRate 进行初始化序列的选择。

【注意事项】

无。

【相关数据类型及接口】

- RECT_S
- SIZE_S
- ISP_BAYER_FORMAT_E
- WDR_MODE_E
- CVI_ISP_SetPubAttr
- CVI_ISP_GetPubAttr

3.3.6 ISP_FMW_STATE_E

【说明】

定义 ISP firmware 状态。

【定义】

```
typedef enum _ISP_FMW_STATE_E {  
    ISP_FMW_STATE_RUN,  
    ISP_FMW_STATE_FREEZE,  
    ISP_FMW_STATE_BUTT  
} ISP_FMW_STATE_E;
```

【成员】

成员名称	描述
ISP_FMW_STATE_RUN	Firmware 正常运行状态
ISP_FMW_STATE_FREEZE	Firmware 冻结状态

【注意事项】

无。

【相关数据类型及接口】

- CVI_ISP_SetFMWState
- CVI_ISP_GetFMWState

3.3.7 ISP_MODULE_CTRL_U

【说明】

定义 ISP 功能模块的控制。

【定义】

```
typedef union _ISP_MODULE_CTRL_U {
    CVI_U64 u64Key;
    struct {
        CVI_U64 bitBypassBlc : 1; /* RW:[0] */
        CVI_U64 bitBypassRlsc : 1; /* RW:[1] */
        CVI_U64 bitBypassFpn : 1; /* RW:[2] */
        CVI_U64 bitBypassDpc : 1; /* RW:[3] */
        CVI_U64 bitBypassCrosstalk : 1; /* RW:[4] */
        CVI_U64 bitBypassWBGain : 1; /* RW:[5] */
        CVI_U64 bitBypassDis : 1; /* RW:[6] */
        CVI_U64 bitBypassBnr : 1; /* RW:[7] */
        CVI_U64 bitBypassDemosaic : 1; /* RW:[8] */
        CVI_U64 bitBypassRbgcac : 1; /* RW:[9] */
        CVI_U64 bitBypassMlsc : 1; /* RW:[10] */
        CVI_U64 bitBypassCcm : 1; /* RW:[11] */
        CVI_U64 bitBypassFusion : 1; /* RW:[12] */
        CVI_U64 bitBypassDrc : 1; /* RW:[13] */
        CVI_U64 bitBypassGamma : 1; /* RW:[14] */
        CVI_U64 bitBypassDehaze : 1; /* RW:[15] */
        CVI_U64 bitBypassClut : 1; /* RW:[16] */
        CVI_U64 bitBypassCsc : 1; /* RW:[17] */
        CVI_U64 bitBypassDci : 1; /* RW:[18] */
        CVI_U64 bitBypassCa : 1; /* RW:[19] */
        CVI_U64 bitBypassPreyee : 1; /* RW:[20] */
        CVI_U64 bitBypassMotion : 1; /* RW:[21] */
        CVI_U64 bitBypass3dnr : 1; /* RW:[22] */
        CVI_U64 bitBypassYnr : 1; /* RW:[23] */
        CVI_U64 bitBypassCnr : 1; /* RW:[24] */
        CVI_U64 bitBypassCac : 1; /* RW:[25] */
        CVI_U64 bitBypassCa2 : 1; /* RW:[26] */
        CVI_U64 bitBypassYee : 1; /* RW:[27] */
        CVI_U64 bitBypassYcontrast : 1; /* RW:[28] */
        CVI_U64 bitBypassMono : 1; /* RW:[29] */
        CVI_U64 bitRsv : 34; /* H ; [30:63] */
    };
} ISP_MODULE_CTRL_U;
```

【成员】

成员名称	描述
u64Key	结构体枚举的整形值

下页继续

表 3.39 – 续上页

成员名称	描述
bitBypassXXX	各 module 功能控制 bit

【注意事项】

无。

【相关数据类型及接口】

- CVI_ISP_SetModuleControl
- CVI_ISP_GetModuleControl

3.3.8 ISP_VD_TYPE_E

【说明】

定义与 ISP 的同步讯号

【定义】

```
typedef enum _ISP_VD_TYPE_E {  
    ISP_VD_FE_START = 0,  
    ISP_VD_FE_END,  
    ISP_VD_BE_END,  
    ISP_VD_MAX  
} ISP_VD_TYPE_E;
```

【成员】

成员名称	描述
ISP_VD_FE_START	FE 帧的起始讯号
ISP_VD_FE_END	FE 帧的结束讯号
ISP_VD_BE_END	BE 帧的结束讯号

【注意事项】

无。

【相关数据类型及接口】

- CVI_ISP_GetVDTimeOut

3.3.9 ISP_SNS_ATTR_INFO_S

【说明】

定义 ISP sensor 属性。

【定义】

```
typedef struct _ISP_SNS_ATTR_INFO_S {  
    CVI_U32 eSensorId;  
} ISP_SNS_ATTR_INFO_S;
```

【成员】

成员名称	描述
eSensorId	Sensor ID 号。

【注意事项】

无。

【相关数据类型及接口】

· CVI_ISP_SensorRegCallBack

3.3.10 ISP_CMOS_SENSOR_IMAGE_MODE_S

【说明】

定义 ISP sensor 图像模式信息属性。

【定义】

```
typedef struct _ISP_CMOS_SENSOR_IMAGE_MODE_S {  
  
    CVI_U16 u16Width;  
  
    CVI_U16 u16Height;  
  
    CVI_FLOAT f32Fps;  
  
    CVI_U8 u8SnsMode;  
  
    CVI_U8 u8LaneNum;  
  
    CVI_U8 u8EnableMaster;  
  
} ISP_CMOS_SENSOR_IMAGE_MODE_S;
```

【成员】

成员名称	描述
u16Width	图像宽度
u16Height	图像高度
f32Fps	帧率
u8SnsMode	传感器工作模式
u8LaneNum	MIPI 通道数
u8EnableMaster	主从使能标志

【注意事项】

无。

【相关数据类型及接口】

· `ISP_SENSOR_EXP_FUNC_S`

3.3.11 ISP_CMOS_BLACK_LEVEL_S

【说明】

定义 ISP sensor 黑电平信息属性。

【定义】

```
typedef struct _ISP_CMOS_BLACK_LEVEL_S {  
    CVI_BOOL bUpdate;  
    ISP_BLACK_LEVEL_ATTR_S blcAttr;  
} ISP_CMOS_BLACK_LEVEL_S;
```

【成员】

成员名称	描述
bUpdate	更新标志
blcAttr	黑电平属性

【注意事项】

无。

【相关数据类型及接口】

· `ISP_SENSOR_EXP_FUNC_S`

3.3.12 ISP_CMOS_DEFAULT_S

【说明】

定义 ISP sensor 默认噪声校准属性。

【定义】

```
typedef struct _ISP_CMOS_DEFAULT_S {  
    ISP_CMOS_NOISE_CALIBRATION_S stNoiseCalibration;  
} ISP_CMOS_DEFAULT_S;
```

【成员】

成员名称	描述
stNoiseCalibration	噪声校准配置

【注意事项】

无。

【相关数据类型及接口】

· [ISP_SENSOR_EXP_FUNC_S](#)

3.3.13 ALG_LIB_S

【说明】

所用库的信息。

【定义】

```
typedef struct _ALG_LIB_S {  
    CVI_S32 s32Id;  
    CVI_CHAR acLibName[ALG_LIB_NAME_SIZE_MAX];  
} ALG_LIB_S;
```

【成员】

成员名称	描述
s32Id	算法库实例的 Id。
acLibName	标识算法库名称的字符数组。

【注意事项】

无。

【相关数据类型及接口】

· [CVI_ISP_SensorRegCallBack](#)

- CVI_ISP_SensorUnRegCallBack
- CVI_ISP_AELibRegCallBack
- CVI_ISP_AELibUnRegCallBack
- CVI_ISP_AWBLibRegCallBack
- CVI_ISP_AWBLibUnRegCallBack

3.3.14 ISP_AE_REGISTER_S

【说明】

ISP 提供的 AE 库注册的回调接口。

【定义】

```
typedef struct ISP_AE_REGISTER_S {
    ISP_AE_EXP_FUNC_S stAeExpFunc;
} ISP_AE_REGISTER_S;
```

【成员】

成员名称	描述
stAeExpFunc	AE 注册的回调函数结构体。

【注意事项】

无。

【相关数据类型及接口】

- ISP_AE_EXP_FUNC_S
- CVI_ISP_AELibRegCallBack

3.3.15 ISP_AE_EXP_FUNC_S

【说明】

定义 AE 回调函数结构体。

【定义】

```
typedef struct ISP_AE_EXP_FUNC_S {
    CVI_S32 (*pfn_ae_init)(VI_PIPE ViPipe, const ISP_AE_PARAM_S *pstAeParam);
    CVI_S32 (*pfn_ae_run)(VI_PIPE ViPipe, const ISP_AE_INFO_S *pstAeInfo, ISP_AE_
    →RESULT_S *pstAeResult, CVI_S32 s32Rsv);
    CVI_S32 (*pfn_ae_ctrl)(VI_PIPE ViPipe, CVI_U32 u32Cmd, void *pValue);
    CVI_S32 (*pfn_ae_exit)(VI_PIPE ViPipe);
} ISP_AE_EXP_FUNC_S;
```

【成员】

成员名称	描述
pfn_ae_init	初始化 AE 的回调函数指针。
pfn_ae_run	运行 AE 的回调函数指针。
pfn_ae_ctrl	控制 AE 内部状态的回调函数指针。
pfn_ae_exit	销毁 AE 的回调函数指针。

【注意事项】

- 调用 CVI_ISP_Init 时将调用 pfn_ae_init 回调函数，以初始化 AE 算法库。
- 调用 CVI_ISP_Run 时将调用 pfn_ae_run 回调函数，以运行 AE 算法库，计算得到 sensor 的曝光时间和增益、ISP 的数字增益。
- 调用 CVI_ISP_Exit 时将调用 pfn_ae_exit 回调函数，以销毁 AE 算法库。

【相关数据类型及接口】

- ISP_AE_REGISTER_S

3.3.16 ISP_AE_PARAM_S

【说明】

定义 ISP 提供给 AE 的初始化参数结构体。

【定义】

```
typedef struct _ISP_AE_PARAM_S{  
    SENSOR_ID SensorId;  
    CVI_U8 u8WDRMode;  
    CVI_U8 u8HDRMode;  
    CVI_U16 u16BlackLevel;  
    CVI_FLOAT f32Fps;  
    ISP_BAYER_FORMAT_E enBayer;  
    ISP_STITCH_ATTR_S stStitchAttr;  
    CVI_S32 s32Rsv;  
    ISP_3AWIN_CONFIG_S aeLEWinConfig[AE_MAX_NUM];  
    ISP_3AWIN_CONFIG_S aeSEWinConfig;  
} ISP_AE_PARAM_S;
```

【成员】

成员名称	描述
SensorId	所注册 Sensor 的 id。
u8WDRMode	宽动态模式，ISP 向 AE 提供宽动态模式信息。
u8HDRMode	HDR 模式，ISP 向 AE 提供 HDR 模式信息。
u16BlackLevel	黑电平值，12bit 精度，ISP 向 AE 提供黑电平信息。
f32Fps	帧率，ISP 向 AE 提供帧率信息。

下页继续

表 3.48 – 续上页

成员名称	描述
enBayer	Sensor Bayer Pat tern, 包括 RGGB、GRBG、GBRG、BGGR 四种格式。
stStitchAttr	拼接模式, ISP 向 AE 提供拼接模式信息。
aeLEWinConfig	ISP 向 AE 提供 WDR 长帧的 window 配置信息。
aeSEWinConfig	ISP 向 AE 提供 WDR 短帧的 window 配置信息。

【注意事项】

无。

【相关数据类型及接口】

无。

3.3.17 ISP_SMART_INFO_S

【说明】

定义 ISP 提供给 AE 的统计信息结构体。

【定义】

```
typedef struct _ISP_SMART_INFO_S
{
    ISP_SMART_ROI_S stROI[SMART_CLASS_MAX];
} ISP_SMART_INFO_S;
```

【成员】

成员名称	描述
stROI	人脸识别或人形检测区域信息

【注意事项】

- 当前仅支持人脸模型与人形模型检测结果, stROI 下标为 0 时为人脸检测结果, 下标为 1 时为人形检测结果
- AE 会根据提供的人脸在 raw 的位置来找出人脸位置的 AE window, 并使用此 AE window 的亮度来执行 face AE 的测光策略

【相关数据类型及接口】

无。

3.3.18 ISP_AE_INFO_S

【说明】

定义 ISP 提供给 AE 的统计信息结构体。

【定义】

```
typedef struct _ISP_AE_INFO_S {
    CVI_U32 u32FrameCnt;
    ISP_SMART_ROI_S stSmartInfo;
    ISP_FE_AE_STAT_1_S *pstFEAeStat1[AE_MAX_NUM];
    ISP_FE_AE_STAT_2_S *pstFEAeStat2[AE_MAX_NUM];
    ISP_FE_AE_STAT_3_S *pstFEAeStat3[AE_MAX_NUM];
    ISP_FE_AE_STITCH_STAT_3_S *pstFEAeStiStat;
    ISP_BE_AE_STAT_1_S *pstBEAeStat1;
    ISP_BE_AE_STAT_2_S *pstBEAeStat2;
    ISP_BE_AE_STAT_3_S *pstBEAeStat3;
    ISP_BE_AE_STITCH_STAT_3_S *pstBEAeStiStat;
    ISP_3AWIN_CONFIG_S aeLEWinConfig[AE_MAX_NUM];
    ISP_3AWIN_CONFIG_S aeSEWinConfig;
} ISP_AE_INFO_S;
```

【成员】

成员名称	子成员名称	描述
u32FrameCnt		帧的累加计数
stSmartInfo		请参考 ISP_SMART_INFO_S 接口说明
pstFEAeStat1	u32PixelCount	统计的像素点总个数。
	u32PixelWeight	统计的带权重像素点总个数。
	au32HistogramMemArray	256 段直方图的统计信息数组
pstFEAeStat2	u16GlobalAvgR	全局 R 分量平均值
	u16GlobalAvgGr	全局 Gr 分量平均值
	u16GlobalAvgGb	全局 Gb 分量平均值
	u16GlobalAvgB	全局 B 分量平均值
pstFEAeStat3	au16ZoneAvg	分区间 R、Gr、Gb、B 分量平均值
pstFEAeStiStat	au16ZoneAvg	拼接模式下拼接后分区间 R、Gr、Gb、B 分量平均值
pstBEAeStat1	bStable	目前不使用
pstBEAeStat2	bStable	目前不使用
pstBEAeStat3	bStable	目前不使用
pstBEAeStiStat	bStable	目前不使用
aeLEWinConfig	winWidth	AE window 宽
	winHeight	AE window 高
	winXOffset	AE window 起始位置水平方向偏移值
	winYOffset	AE window 起始位置垂直方向偏移值
	winXNum	AE 水平方向 window 数
	winYNum	AE 垂直方向 window 数

【注意事项】

无。

【相关数据类型及接口】

无。

3.3.19 ISP_AE_RESULT_S

【说明】

定义 AE 库返回给 ISP 的配置寄存器结构体。

【定义】

```
typedef struct _ISP_AE_STAT_RESULT_S {  
    CVI_U32 u32IntTime[VI_MAX_PIPE_NUM];  
    CVI_U32 u32IspDgain;  
    CVI_U32 u32Again;  
    CVI_U32 u32Dgain;  
    CVI_U32 u32Iso;  
    CVI_U8 u8AERunInterval;  
    CVI_BOOL bPirisValid;  
    CVI_S32 s32PirisPos;  
    CVI_U32 u32PirisGain;  
    ISP_FSWDR_MODE_E enFSWDRMode;  
    CVI_U32 au32WDRGain[4];  
    CVI_U32 u32HmaxTimes;  
    ISP_AE_STAT_ATTR_S stStatAttr;  
    ISP_DCF_UPDATE_INFO_S stUpdateInfo;  
    CVI_U32 u32ExpRatio;  
    CVI_S16 s16CurrentLV;  
    CVI_U32 u32AvgLuma;  
    CVI_U8 u8MeterFramePeriod;  
    CVI_BOOL bStable;  
    CVI_FLOAT fBvStep;  
    CVI_U32 u32BlcIso;  
    CVI_U32 u32IspDgainSF;  
    CVI_U32 u32AgainSF;  
    CVI_U32 u32DgainSF;  
    CVI_U32 u32IsoSF;  
    CVI_U32 u32BlcIsoSF;  
    CVI_FLOAT fEvRatio[2];  
} ISP_AE_RESULT_S;
```

【成员】

成员名称	描述
u32IntTime	曝光时间
u32IspDgain	ISP 数字增益, 10 bit 精度
u32Again	Sensor 模拟增益, 10 bit 精度
u32Dgain	Sensor 数字增益, 10 bit 精度
u32Iso	AE 总增益值, 2 倍时为 $100 * 2 = 200$
u8AERunInterval	AE 算法运行的间隔

下页继续

表 3.51 – 续上页

成员名称	描述
bPirisValid	Piris 是否有效的标志
s32PirisPos	Piris 步进电机的位置，取值范围与具体 Piris 镜头相关
u32PirisGain	Piris 光圈等效增益，取值范围与具体 Piris 镜头相关
enFSWDRMode	WDR 合成模式。 0 表示普通多帧合成 WDR 模式； 1 表示长帧模式； 2 表示自动长帧模式。
au32WDRGain	兼容参数，目前不使用
u32HmaxTimes	Sensor 对应读出一行的时间，单位：ns
stStatAttr	兼容参数，目前不使用
stUpdateInfo	用于传递 AE 相关 DCF 信息
u32ExpRatio	WDR 模式时，长/短帧的曝光比值，64 为 1 倍
s16CurrentLV	AE 目前估算的环境亮度，数值越大表示环境越亮
u32AvgLuma	目前画面的亮度
u8MeterFramePeriod	AE 的曝光生效周期
bStable	当前 AE 收敛状态是否稳定
fBvStep	目前画面的亮度距离目标亮度的步幅
u32BlcIso	长曝对应 BLC 参考的总增益值，此增益仅包含 sensor 的仿真/数字增益，不包含 ISP 数字增益
u32IspDgainSF	WDR 模式，短帧 ISP 数字增益，10 bit 精度
u32AgainSF	WDR 模式，短帧 Sensor 模拟增益，10 bit 精度
u32DgainSF	WDR 模式，短帧 Sensor 数字增益，10 bit 精度
u32IsoSF	WDR 模式，AE 总增益值
u32BlcIsoSF	短曝 BLC 参考的总增益值，此增益仅包含 sensor 的仿真/数字增益，不包含 ISP 数字增益
fEvRatio	AE 当前帧与前一帧的曝光差异比率

【注意事项】

无。

【相关数据类型及接口】

无。

3.3.20 ISP_SENSOR_EXP_FUNC_S

【说明】

定义 sensor 回调函数结构体。

【定义】

```
typedef struct ISP_SENSOR_EXP_FUNC_S {
    CVI_VOID (*pfn_cmos_sensor_init)(VI_PIPE ViPipe);
    CVI_VOID (*pfn_cmos_sensor_exit)(VI_PIPE ViPipe);
    CVI_VOID (*pfn_cmos_sensor_global_init)(VI_PIPE ViPipe);
    CVI_S32 (*pfn_cmos_set_image_mode)(VI_PIPE ViPipe, ISP_CMOS_SENSOR_IMAGE_
    ↪MODE_S *pstSensorImageMode);
    CVI_S32 (*pfn_cmos_set_wdr_mode)(VI_PIPE ViPipe, CVI_U8 u8Mode);

    /* the algs get data which is associated with sensor, except 3a */
    CVI_S32 (*pfn_cmos_get_isp_default)(VI_PIPE ViPipe, ISP_CMOS_DEFAULT_S *pstDef);
    CVI_S32 (*pfn_cmos_get_isp_black_level)(VI_PIPE ViPipe, ISP_CMOS_BLACK_LEVEL_S_
    ↪*pstBlackLevel);
    CVI_S32 (*pfn_cmos_get_sns_reg_info)(VI_PIPE ViPipe, ISP_SNS_SYNC_INFO_S_
    ↪*pstSnsRegsInfo);

    /* the function of sensor set pixel detect */
    //CVI_VOID (*pfn_cmos_set_pixel_detect)(VI_PIPE ViPipe, bool bEnable);
} ISP_SENSOR_EXP_FUNC_S;
```

【成员】

成员名称	描述
pfn_cmos_sensor_init	初始化 sensor 的回调函数指针。
pfn_cmos_sensor_exit	sensor 的回调退出函数指针。
pfn_cmos_sensor_global_init	初始化全局变量的回调函数指针。
pfn_cmos_set_image_mode	设置分辨率和帧率切换的回调函数指针。
pfn_cmos_set_wdr_mode	设置 wdr 模式的回调函数指针。
pfn_cmos_get_isp_default	获取 ISP 基础算法的初始值的回调函数指针。
pfn_cmos_get_isp_black_level	获取 sensor 的黑电平值的回调函数指针，支持根据 sensor 增益动态调整黑电平值
pfn_cmos_get_sns_reg_info	获取 sensor 寄存器信息的回调函数指针，用于实现内核态配置 AE 信息
pfn_cmos_set_pixel_detect	设置坏点校正开关的回调函数指针

【注意事项】

无。

【相关数据类型及接口】

无。

3.3.21 ISP_SENSOR_REGISTER_S

【说明】

ISP 提供的 sensor 注册的回调接口。

【定义】

```
typedef struct bmISP_SENSOR_REGISTER_S {
    ISP_SENSOR_EXP_FUNC_S stSnsExp;
} ISP_SENSOR_REGISTER_S;
```

【成员】

成员名称	描述
stSnsExp	Sensor 注册的回调函数结构体。

【注意事项】

无。

【相关数据类型及接口】

- ISP_SENSOR_EXP_FUNC_S
- CVI_ISP_SensorRegCallBack

3.3.22 ISP_AWB_EXP_FUNC_S

【说明】

定义 AWB 回调函数结构体。

【定义】

```
typedef struct _ISP_AWB_EXP_FUNC_S {
    CVI_S32 (*pfn_awb_init)(VI_PIPE ViPipe, const ISP_AWB_PARAM_S *pstAwbParam);
    CVI_S32 (*pfn_awb_run)(VI_PIPE ViPipe, const ISP_AWB_INFO_S *pstAwbInfo, ISP_AWB_
→RESULT_S *pstAwbResult, CVI_S32 s32Rsv);
    CVI_S32 (*pfn_awb_ctrl)(VI_PIPE ViPipe, CVI_U32 u32Cmd, CVI_VOID * pValue);
    CVI_S32 (*pfn_awb_exit)(VI_PIPE ViPipe);
} ISP_AWB_EXP_FUNC_S;
```

【成员】

成员名称	描述
pfn_awb_init	初始化 AWB 的回调函数指针
pfn_awb_run	运行 AWB 的回调函数指针。
pfn_awb_ctrl	控制 AWB 内部状态的回调函数指针。
pfn_awb_exit	销毁 AWB 的回调函数指针

【注意事项】

无。

【相关数据类型及接口】

无。

3.3.23 ISP_AWB_REGISTER_S

【说明】

定义 AWB 注册结构体。

【定义】

```
typedef struct _ISP_AWB_REGISTER_S {  
    ISP_AWB_EXP_FUNC_S stAwbExpFunc;  
} ISP_AWB_REGISTER_S;
```

【成员】

成员名称	描述
stAwbExpFunc	AWB 注册回调函数的结构体

【注意事项】

无。

【相关数据类型及接口】

无。

3.3.24 ISP_AWB_PARAM_S

【说明】

定义 ISP 给 AWB 初始化参数结构

【定义】

```
typedef struct _ISP_AWB_PARAM_S {  
    SENSOR_ID SensorId;  
    CVI_U8 u8WDRMode;  
    CVI_U8 u8AWBZoneRow;  
    CVI_U8 u8AWBZoneCol;  
    CVI_U8 u8AWBZoneBin;  
    ISP_STITCH_ATTR_S stStitchAttr;  
    CVI_U16 u16AWBWidth;  
    CVI_U16 u16AWBHeight;  
    CVI_S8 s8Rsv;  
} ISP_AWB_PARAM_S;
```

【成员】

成员名称	描述
SensorId	向 ISP 注册的 Sensor ID
u8WDRMode	宽动态模式
u8AWBZoneRow	AWB 统计行数
u8AWBZoneCol	AWB 统计列数
u8AWBZoneBin	AWB 统计亮度数
stStitchAttr	拼接讯息结构体
u16AWBWidth	AWB 算法的图像宽度
u16AWBHeight	AWB 算法的图像高度
s8Rsv	保留参数

【注意事项】

无。

【相关数据类型及接口】

无。

3.3.25 ISP_AWB_STAT_1_S

【说明】

定义 AWB 统计值讯息结构体

【定义】

```
typedef struct _ISP_AWB_STAT_1_S {  
    CVI_U16 u16MeteringAwbAvgR;  
    CVI_U16 u16MeteringAwbAvgG;  
    CVI_U16 u16MeteringAwbAvgB;  
    CVI_U16 u16MeteringAwbCountAll;  
} ISP_AWB_STAT_1_S;
```

【成员】

成员名称	描述
u16MeteringAwbAvgR	Bayer 域全局统计中白点的 R 分量
u16MeteringAwbAvgG	Bayer 域全局统计中白点的 G 分量
u16MeteringAwbAvgB	Bayer 域全局统计中白点的 B 分量
u16MeteringAwbCountAll	Bayer 域全局统计中白点的个数

【注意事项】

无。

【相关数据类型及接口】

无。

3.3.26 ISP_AWB_STAT_RESULT_S

【说明】

定义 AWB 统计值讯息结构体

【定义】

```
typedef struct _ISP_AWB_STAT_RESULT_S {  
    CVI_U16 *pau16ZoneAvgR;  
    CVI_U16 *pau16ZoneAvgG;  
    CVI_U16 *pau16ZoneAvgB;  
    CVI_U16 *pau16ZoneCount;  
} ISP_AWB_STAT_RESULT_S;
```

【成员】

成员名称	描述
pau16ZoneAvgR	Bayer 域分区统计值中白点 R 分量数组的起始位置
pau16ZoneAvgG	Bayer 域分区统计值中白点 G 分量数组的起始位置
pau16ZoneAvgB	Bayer 域分区统计值中白点 B 分量数组的起始位置
pau16ZoneCount	Bayer 域分区统计值中白点个数数组的起始位置

【注意事项】

无。

【相关数据类型及接口】

无。

3.3.27 ISP_AWB_INFO_S

【说明】

定义 ISP 提供给 AWB 统计值讯息的结构体

【定义】

```
typedef struct _ISP_AWB_INFO_S {  
    CVI_U32 u32FrameCnt;  
    ISP_SMART_ROI_S stSmartInfo;  
    ISP_AWB_STAT_1_S *pstAwbStat1[ISP_CHANNEL_MAX_NUM];  
    ISP_AWB_STAT_RESULT_S stAwbStat2[ISP_CHANNEL_MAX_NUM];  
    CVI_U8 u8AwbGainSwitch;  
    CVI_U32 au32WDRWBGain[ISP_BAYER_CHN_NUM];  
    CVI_U32 u32IsoNum;  
    CVI_S16 s16LVx100;
```

(下页继续)

(续上页)

```
CVI_FLOAT fBVstep;
}ISP_AWB_INFO_S;
```

【成员】

成员名称	描述
u32FrameCnt	帧的累加计数
stSmartInfo	特殊区域的信息 (给车牌或是人脸使用), 请参考 ISP_SMART_INFO_S 接口说明
pstAwbStat1	AWB 统计讯息 1
stAwbStat2	AWB 统计讯息 2
u8AwbGainSwitch	保留, 无作用
au32WDRWBGain	保留, 无作用
u32IsoNum	当前画面的 ISO 增益
s16LVx100	当前画面的亮度值 x100
fBVstep	当前画面 AE 收敛的差值

【注意事项】

无。

【相关数据类型及接口】

无。

3.3.28 ISP_SMART_ROI_S

【说明】

定义 ISP 提供给 AE 人脸识别或人形识别区域统计信息的结构体

【定义】

```
typedef struct _ISP_SMART_ROI_S {
    CVI_BOOL bEnable;
    CVI_BOOL bAvailable;
    CVI_U8 u8Luma;
    CVI_U8 u8Num;
    CVI_U16 u16PosX[SMART_MAX_NUM];
    CVI_U16 u16PosY[SMART_MAX_NUM];
    CVI_U16 u16Width[SMART_MAX_NUM];
    CVI_U16 u16Height[SMART_MAX_NUM];
    CVI_U16 u16FrameWidth;
    CVI_U16 u16FrameHeight;
} ISP_SMART_ROI_S;
```

【宏】

```
# define SMART_MAX_NUM (3)
```

【成员】

成员名称	描述
bEnable	模型是否使能
bAvailable	模型是否有可用的检测结果
u8Luma	兼容参数, 目前不使用
u16PosX[SMART_MAX_NUM]	检测出的人脸位置对应应在 raw 上的 X 坐标
u16PosY[SMART_MAX_NUM]	检测出的人脸位置对应应在 raw 上的 Y 坐标
u16Width[SMART_MAX_NUM]	检测出的人脸位置对应应在 raw 上的宽
u16Height[SMART_MAX_NUM]	检测出的人脸位置对应应在 raw 上的高
u16FrameWidth	Raw 的 frame 宽
u16FrameHeight	Raw 的 frame 高

【注意事项】

- 当前仅支持人脸模型与人形模型检测结果, stROI 下标为 0 时为人脸检测结果, 下标为 1 时为人形检测结果
- AE 会根据提供的人脸在 raw 的位置来找出人脸位置的 AE window, 并使用此 AE window 的亮度来执行 face AE 的测光策略

【相关数据类型及接口】

无。

3.3.29 ISP_AWB_RAW_STAT_ATTR_S

【说明】

定义 AWB lib 返回给 ISP 的配置缓存器结构体

【定义】

```
typedef struct ISP_AWB_RAW_STAT_ATTR_S {
    CVI_BOOL bStatCfgUpdate;
    CVI_U16 u16MeteringWhiteLevelAwb;
    CVI_U16 u16MeteringBlackLevelAwb;
    CVI_U16 u16MeteringCrRefMaxAwb;
    CVI_U16 u16MeteringCbRefMaxAwb;
    CVI_U16 u16MeteringCrRefMinAwb;
    CVI_U16 u16MeteringCbRefMinAwb;
} ISP_AWB_RAW_STAT_ATTR_S;
```

【成员】

成员名称	描述
bStatCfgUpdate	保留, 无作用
u16MeteringWhiteLevelAwb	保留, 无作用
u16MeteringBlackLevelAwb	保留, 无作用
u16MeteringCrRefMaxAwb	保留, 无作用
u16MeteringCbRefMaxAwb	保留, 无作用
u16MeteringCrRefMinAwb	保留, 无作用

下页继续

表 3.61 – 续上页

成员名称	描述
u16MeteringCbRefMinAwb	保留, 无作用

【注意事项】

无。

【相关数据类型及接口】

无。

3.3.30 ISP_AWB_RESULT_S

【说明】

定义 AWB lib 返回给 ISP 配置缓存器的结构体

【定义】

```
typedef struct ISP_AWB_RESULT_S {  
    CVI_U32 au32WhiteBalanceGain[ISP_BAYER_CHN_NUM];  
    CVI_U16 au16ColorMatrix[CCM_MATRIX_SIZE];  
    CVI_U32 u32ColorTemp;  
    CVI_U8 u8Saturation[4];  
    ISP_AWB_RAW_STAT_ATTR_S stRawStatAttr;  
    CVI_BOOL bStable;  
    CVI_U8 u8AdjCASaturation;  
    CVI_U8 u8AdjCASatLuma;  
} ISP_AWB_RESULT_S;
```

【成员】

成员名称	描述
au32WhiteBalanceGain	AWB lib 算出来的 R,Gr,Gb,B 颜色通道的增益
au16ColorMatrix	保留, 无作用
u32ColorTemp	现在 AWB 估算出的色温
u8Saturation	当前饱和度
stRawStatAttr	保留, 无作用
bStable	是否收敛
u8AdjCASaturation	饱和度计算结果
u8AdjCASatLuma	亮度计算结果

【注意事项】

无。

【相关数据类型及接口】

无。

3.3.31 ISP_BIND_ATTR_S

【说明】

定义 ISP 库与 Sensor、3A 库之间绑定关系的结构体。

【定义】

```
typedef struct _ISP_BIND_ATTR_S {  
    CVI_S32 sensorId;  
    ALG_LIB_S stAeLib;  
    ALG_LIB_S stAfLib;  
    ALG_LIB_S stAwbLib;  
} ISP_BIND_ATTR_S;
```

【成员】

成员名称	描述
SensorId	注册 Sensor 的 Id。
stAeLib	AE 库结构体。
stAwbLib	AF 库结构体。
stAfLib	AWB 库结构体。

【注意事项】

无。

【相关数据类型及接口】

- CVI_ISP_SetBindAttr
- CVI_ISP_GetBindAttr

3.3.32 ISP_CTRL_PARAM_S

【说明】

定义 ISP 控制参数结构体。

【定义】

```
typedef struct _ISP_CTRL_PARAM_S {  
    CVI_U32 u32AESTatIntvl;  
    CVI_U32 u32AWBStatIntvl;  
    CVI_U32 u32AFStatIntvl;  
    CVI_U32 u32ProcParam;  
    CVI_U32 u32ProcLevel;  
    CVI_U32 u32UpdatePos;  
    CVI_U32 u32IntTimeOut;  
    CVI_U32 u32PwmNumber;  
    CVI_U32 u32PortIntDelay;  
} ISP_CTRL_PARAM_S;
```

【成员】

成员名称	描述
u32AStatIntvl	ISP 3A AE 统计信息更新频率，单位为帧 取值范围：(0,0xffffffff]
u32AWBStatIntvl	ISP 3A AWB 统计信息更新频率，单位为帧 取值范围：(0,0xffffffff]
u32AFStatIntvl	ISP 3A AF 统计信息更新频率，单位为帧 取值范围：(0,0xffffffff]
u32ProcParam	ISP 收集信息的收集频率，单位为帧，默认值为 30，这个值越高，ISP 的 CPU 占用率就越低 取值范围：(0,0xffffffff]
u32ProcLevel	ISP 的 proc 打印 Level，Level 为 0,proc 功能关闭，level 为 1，精简信息，level 为 2，较多信息，level 为 3，非常多信息（会把 3A 统计值也打印出来）
u32UpdatePos	目前 cvitek 仅支持在帧开始配置 sensor 寄存器，默认值为 0
u32IntTimeOut	表示中断超时的时间（ms）。目前 cvitek 没有使用
u32PwmNumber	表示 PWM 号。目前 cvitek 没有使用
u32PortIntDelay	表示 Port 中断延时时间

【注意事项】

无。

【相关数据类型及接口】

- [CVI_ISP_SetCtrlParam](#)
- [CVI_ISP_GetCtrlParam](#)

3.3.33 ISP_MOD_PARAM_S

【说明】

定义 ISP 模块参数结构体。

【定义】

```
typedef struct _ISP_MOD_PARAM_S {  
    CVI_U32 u32IntBotHalf;  
} ISP_MOD_PARAM_S;
```

【成员】

成员名称	描述
u32IntBotHalf	表示 ISP 中断处理是否采用下半部机制，默认值为 0, 目前 cvitek 仅支持（读统计信息和配置 sensor 和 ISP 同步寄存器）在中断服务程序中完成；

【注意事项】

无。

【相关数据类型及接口】

- CVI_ISP_SetModParam
- CVI_ISP_GetModParam

3.3.34 ISP_IR_AUTO_ATTR_S

【说明】

定义红外自动切换属性。

【定义】

```
typedef struct _ISP_IR_AUTO_ATTR_S
{
    CVI_BOOL bEnable;
    CVI_U32 u32Normal2IrIsoThr;
    CVI_U32 u32Ir2NormalIsoThr;
    CVI_U32 u32RGMax;
    CVI_U32 u32RGMin;
    CVI_U32 u32BGMax;
    CVI_U32 u32BGMin;
    ISP_IR_STATUS_E enIrStatus;
    ISP_IR_SWITCH_STATUS_E enIrSwitch;
} ISP_IR_AUTO_ATTR_S;
```

【成员】

成员名称	描述
bEnable	红外自动切换使能。 CVI_FALSE: 关闭 CVI_TRUE: 使能。
u32Normal2IrIsoThr	从普通状态切换到红外状态的 ISO 阈值。当实际生效的 ISO 大于此阈值时，系统需要切换到红外状态。 取值范围：[0, 0xFFFFFFFF]。
u32Ir2NormalIsoThr	从红外状态切换到普通状态的 ISO 阈值。当实际生效的 ISO 小于此阈值时，系统需要切换到普通状态。 取值范围：[0, 0xFFFFFFFF]。

下页继续

表 3.66 – 续上页

成员名称	描述
u32RGMax	红外状态下的 R/G 最大值。实际图像的 R/G 大于此参数时，系统需要切换到普通状态。4.8 格式。 取值范围：[0, 0xFF FFF]。
u32RGMin	红外状态下的 R/G 最小值。实际图像的 R/G 小于此参数时，系统需要切换到普通状态。4.8 格式。 取值范围：[0, u32RGMax]。
u32BGMax	红外状态下的 B/G 最大值。实际图像的 B/G 大于此参数时，系统需要切换到普通状态。4.8 格式。 取值范围：[0, 0xFF FFF]。
u32BGMin	红外状态下的 B/G 最小值。实际图像的 B/G 小于此参数时，系统需要切换到普通状态。4.8 格式。 取值范围：[0, u32BGMax]。
enIrStatus	设备当前的红外状态。应配置为设备实际的红外状态，需要用户保证状态的正确性。
enIrSwitch	设备的红外切换状态，为只读。

【注意事项】

无。

【相关数据类型及接口】

无。

3.3.35 ISP_STITCH_ATTR_S

【说明】

定义 ISP 拼接参数结构体。

【定义】

```
typedef struct ISP_STITCH_ATTR_T {  
    CVI_BOOL enable;  
    CVI_BOOL bMainPipe;  
    CVI_BOOL bCalibEnable;  
    CVI_BOOL bCombineSts;  
    CVI_U8 u8CombChnSum;  
    CVI_U8 u8CombChn;  
    CVI_U8 u8Group;  
    CVI_U32 u32CalibLumaRatio;  
    CVI_U32 u32CalibRGainRatio;  
    CVI_U32 u32CalibBGainRatio;  
} ISP_STITCH_ATTR_S;
```

【成员】

成员名称	描述
enable	拼接功能使能 CVI_FALSE: 关闭 CVI_TRUE: 使能
bMainPipe	是否为主通道, 当 ISP 拼接使能后, 从通道 AE、AWB 结果参数与主通道保持一致。 CVI_FALSE: 从通道 CVI_TRUE: 主通道
bCombineSts	是否组合统计值, 组合统计值后 AE、AWB 计算时可以兼顾设置组合的通道。 CVI_FALSE: 不组合 CVI_TRUE: 组合
bCalibEnable	是否使能标定参数。 CVI_FALSE: 不使用标定参数 CVI_TRUE: 使能标定参数
u8CombChnSum	组合统计值通道总数。 取值范围: [0, 0x6]
u8CombChn	当前组合统计值通道号 取值范围: [0, 0x6]
u8Group	拼接组数。 取值范围: [0, 0x2]
u32CalibLumaRatio	当前通道与其它通道亮度差异比例 (1024 为 1) 取值范围: [0, 0x1000]
u32CalibRGainRatio	当前通道与其它通道 RGain 差异比例 (1024 为 1) 取值范围: [0, 0x1000]
u32CalibBGainRatio	当前通道与其它通道 BGain 差异比例 (1024 为 1) 取值范围: [0, 0x1000]

【注意事项】

无。

【相关数据类型及接口】

无。

4 AE

4.1 概述

CVI AE 模块实现的功能是：根据自动测光系统获得当前图像的曝光量，再自动配置镜头光圈、sensor 快门及增益来获得最佳的图像质量。

自动曝光的算法主要分光圈优先、快门优先、增益优先。

光圈优先时算法会优先调整光圈到合适的位置，再分配曝光时间和增益，只适合 p-iris 镜头，这样能均衡噪声和景深。

快门优先时算法会优先分配曝光时间，再分配 sensor 增益和 ISP 增益，这样拍摄的图像噪声会比较小。

增益优先则是优先分配 sensor 增益和 ISP 增益，再分配曝光时间，适合拍摄运动物体的场景。

当前 AE 算法也支持客户设定更灵活的曝光分配策略，AE 模块的工作流程如图 4.1 所示

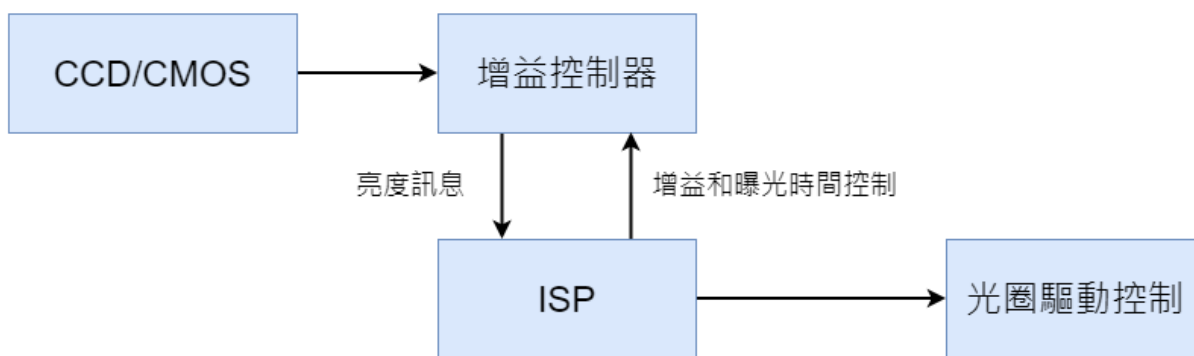


图 4.1: AE 模块工作流程图

4.2 重要概念

- 曝光时间：sensor 积累电荷的时间，是 sensor pixel 从开始曝光到电量被读出的这段时间。
- 曝光增益：对 sensor 输出电荷的总放大系数，一般有数字增益和模拟增益，模拟增益引入的噪声会稍小，所以一般优先用模拟增益。
- 光圈：光圈是镜头中可以改变中间孔大小的机械装置。
- 抗闪烁：由于电灯电源工频与 sensor 的帧率不匹配而导致的画面闪烁，一般通过限定曝光时间和修改 sensor 的帧率来达到抗闪烁的效果。

4.3 功能描述

AE 模块主要有 ISP 的 AE 统计信息模块及 AE 控制策略的 AE 算法 Firmware 两部分组成。ISP 的 AE 统计信息模块主要是提供 sensor 输入数据的亮度信息统计。其提供的统计信息包括直方图和平均值，可同时提供整幅图像的 256 段的直方图和 R/Gr/Gb/B 四分量平均值统计信息，还可提供将整幅图像分成 MxN 区块，每个区块的 R/Gr/Gb/B 四分量平均值统计信息，具体如图 4.2 所示

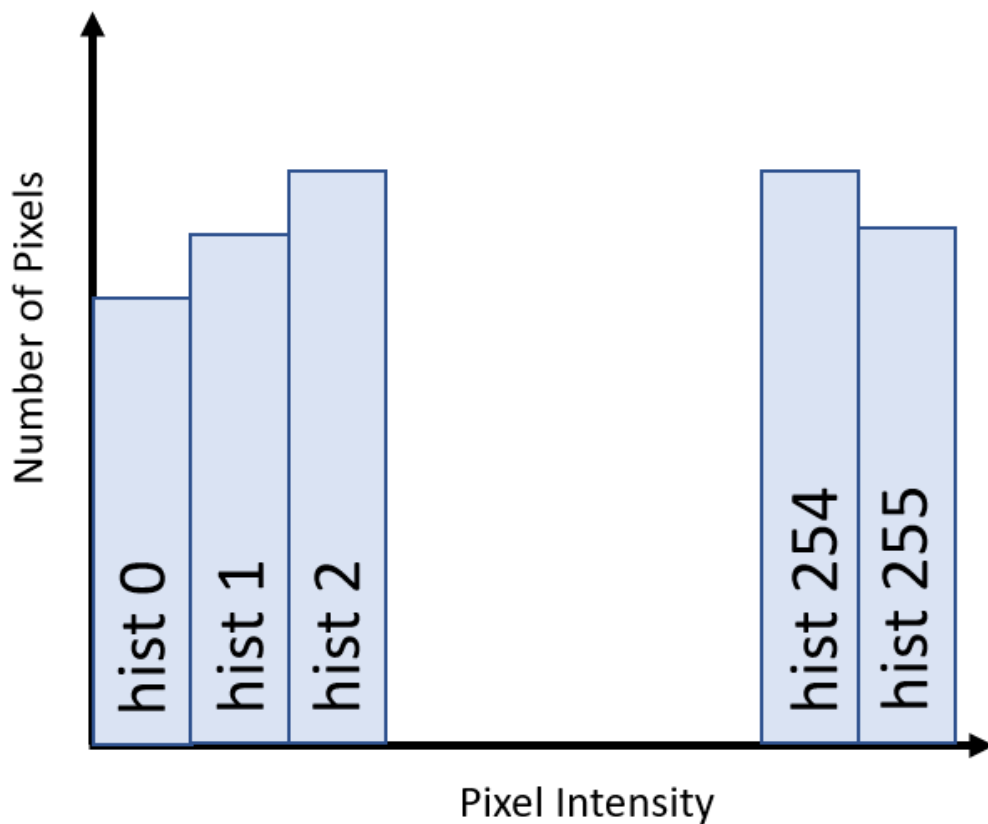


图 4.2: AE 256 段统计信息直方图

AE 算法的主要工作原理是即时获取输入图像的统计信息并与设定目标亮度进行比较，而动态调

节 sensor 的曝光时间和增益以及镜头光圈大小以达到实际亮度与设定目标亮度接近。其工作原理如 图 4.3 所示。

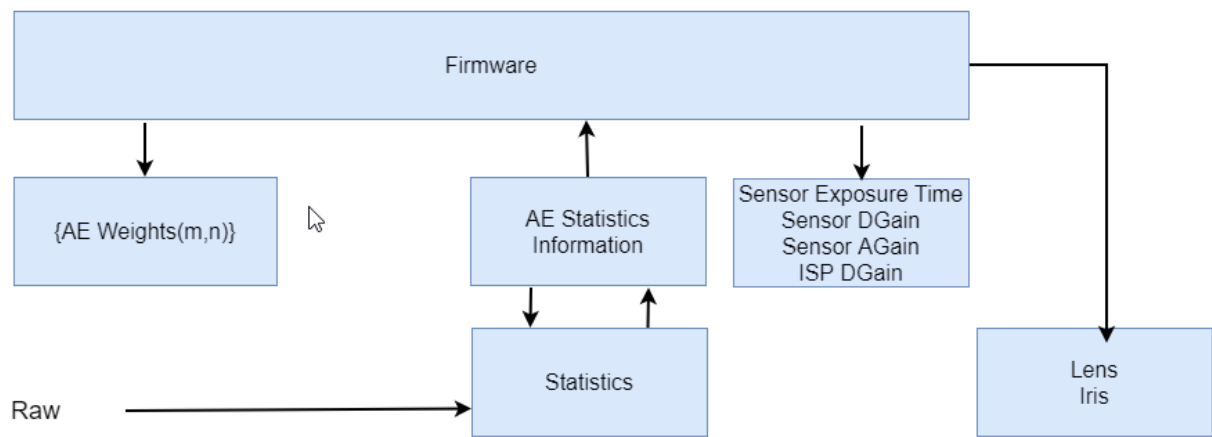


图 4.3: AE 工作原理图

4.4 API 参考

4.4.1 AE 库接口

所有 AE 库接口都只是针对 CVI AE 库，如果客户自己实现 AE 库，不需要关注这些接口，且无法使用这些接口。

- CVI_AE_Register：向 ISP 注册 AE 库。
- CVI_AE_UnRegister：向 ISP 反注册 AE 库。
- CVI_AE_SensorRegCallBack：AE 库提供的 sensor 注册的回调接口。
- CVI_AE_SensorUnRegCallBack：AE 库提供的 sensor 反注册的回调接口。

4.4.1.1 CVI_AE_Register

【描述】

向 ISP 注册 AE 库。

【语法】

```
CVI_S32 CVI_AE_Register(VI_PIPE ViPipe, ALG_LIB_S *pstAeLib);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstAeLib	AE 算法库结构体指针	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 错误码 。

【需求】

- 头文件：cvi_ae.h
- 库文件：libae.a

【注意】

无。

【举例】

无。

【相关主题】

- [CVI_AE_UnRegister](#)

4.4.1.2 CVI_AE_UnRegister**【描述】**

向 ISP 反注册 AE 库。

【语法】

```
CVI_S32 CVI_AE_UnRegister(VI_PIPE ViPipe, ALG_LIB_S *pstAeLib);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstAeLib	AE 算法库结构体指针	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 错误码 。

【需求】

- 头文件：cvi_ae.h
- 库文件：libae.a

【注意】

- 该接口调用了 ISP 库提供的 AE 反注册回调接口，以实现 AE 向 ISP 库反注册的功能。

- 此接口不支持多进程操作。

无。

【举例】

无。

【相关主题】

- CVI_AE_Register

4.4.1.3 CVI_AE_SensorRegCallBack

【描述】

AE 库提供的 sensor 注册的回调接口。

【语法】

```
CVI_S32 CVI_AE_SensorRegCallBack(VI_PIPE ViPipe, ALG_LIB_S *pstAeLib, ISP_SNS_
→ATTR_INFO_S *pstSnsAttrInfo, AE_SENSOR_REGISTER_S *pstRegister);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstAeLib	AE 算法库结构体指针	输入
pstSnsAttrInfo	向 AE 注册的 Sensor 的属性	输入
pstRegister	Sensor 注册结构体指针	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_ae.h
- 库文件：libae.a

【注意】

- SensorId 是 sensor 库中自定义的值，主要用于校对向 ISP 注册的 sensor 和向 3A 注册的 sensor 是否为同一个 sensor。
- AE 通过 sensor 注册的一系列回调接口，获取差异化的初始化参数，并控制 sensor。
- 此接口不支持多进程操作。

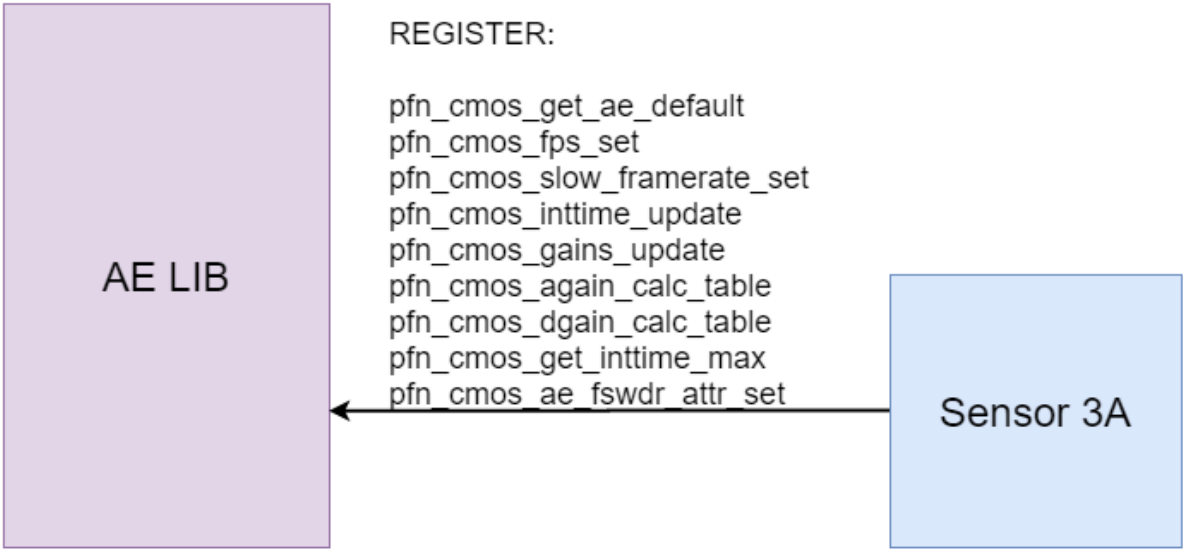


图 4.4: AE 库与 sensor 库间的接口

【举例】

无。

【相关主题】

- CVI_AE_SensorUnRegCallBack

4.4.1.4 CVI_AE_SensorUnRegCallBack

【描述】

AE 库提供的 sensor 反注册的回调接口。

【语法】

```
CVI_S32 CVI_AE_SensorUnRegCallBack(VI_PIPE ViPipe, ALG_LIB_S *pstAeLib, SENSOR_ID_
↪SensorId);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstAeLib	AE 算法库结构体指针	输入
SensorId	向 AE 反注册的 Sensor 的 Id	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件: `cvi_ae.h`
- 库文件: `libae.a`

【注意】

- `SensorId` 是 `sensor` 库中自定义的值, 主要用于校对向 ISP 注册的 `sensor` 和向 3A 注册的 `sensor` 是否为同一个 `sensor`。
- 此接口不支持多进程操作。

【举例】

无。

【相关主题】

- `CVI_AE_SensorRegCallBack`

4.4.2 AE 控制模块

曝光控制接口:

- `CVI_ISP_SetExposureAttr`: 设置 AE 曝光属性。
- `CVI_ISP_GetExposureAttr`: 获取 AE 曝光属性。
- `CVI_ISP_SetWDRExposureAttr`: 设置 WDR 模式下的 AE 曝光属性。
- `CVI_ISP_GetWDRExposureAttr`: 获取 WDR 模式下的 AE 曝光属性。
- `CVI_ISP_SetAERouteAttr`: 设置 AE 路由属性。
- `CVI_ISP_GetAERouteAttr`: 获取 AE 路由属性。
- `CVI_ISP_SetAERouteAttrEx`: 设置 AE 曝光分配扩展属性, 支持分别设置 AE 分配策略中的 `sensor` 模拟增益, `sensor` 数字增益和 ISP 数字增益。
- `CVI_ISP_GetAERouteAttrEx`: 获取 AE 曝光分配策略扩展属性。
- `CVI_ISP_QueryExposureInfo`: 获取 AE 内部状态信息。
- `CVI_ISP_SetAntiFlicker`: 设置 AE anti flicker 功能。
- `CVI_ISP_GetAntiFlicker`: 获取 AE anti flicker 设定。
- `CVI_ISP_QueryFps`: 获取 AE 当前的 fps。
- `CVI_ISP_GetCurrentLvX100`: 获取当前环境亮度的 LV 值。
- `CVI_ISP_SetSmartExposureAttr`: 设置智能模式下的 AE 曝光属性。
- `CVI_ISP_GetSmartExposureAttr`: 获取智能模式下的 AE 曝光属性。
- `CVI_ISP_SetAERouteSFAttr`: 设置 AE WDR 短帧曝光分配策略。
- `CVI_ISP_GetAERouteSFAttr`: 获取 AE WDR 短帧曝光分配策略。
- `CVI_ISP_SetAERouteSFAttrEx`: 设置 AE WDR 短帧曝光分配扩展属性。
- `CVI_ISP_GetAERouteSFAttrEx`: 获取 AE WDR 短帧曝光分配扩展属性。

4.4.2.1 CVI_ISP_SetExposureAttr

【描述】

设定 AE 曝光属性。

【语法】

```
CVI_S32 CVI_ISP_SetExposureAttr(VI_PIPE ViPipe, const ISP_EXPOSURE_ATTR_S_
→*pstExpAttr)
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstExpAttr	AE 曝光属性结构体指针	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_ae.h
- 库文件：libae.a

【注意】

- AE 曝光控制类型为自动时，曝光时间，曝光增益都由 AE 算法自动控制，可以通过配置自动曝光属性结构体 stAuto 里面的参数得到不同的曝光效果。
- AE 曝光控制类型为手动时，可以通过配置手动曝光属性结构体 stManual 控制使能类型（曝光时间使能、ISO num 使能、sensor 模拟增益使能、sensor 数字增益使能、ISP 数字增益使能）及相应的曝光参数（曝光时间、ISO Num、sensor 模拟增益、sensor 数字增益、ISP 数字增益）。
- AE 曝光控制类型为自动时，配置手动曝光属性的参数无效。同理，AE 曝光控制类型为手动时，配置自动曝光属性的参数无效。
- AE 曝光控制类型为手动时，若曝光参数设置超出最大（小）值，将使用 sensor 支持的最大（小）值代替。
- 无论是自动曝光还是手动曝光，曝光时间的单位为微秒 (us)，曝光增益的单位为 10bit 精度的倍数，即 1024 代表 1 倍，2048 代表 2 倍等。

【举例】

```
// 自动曝光属性设置
VI_PIPE ViPipe = 0;
ISP_EXPOSURE_ATTR_S stExpAttr;
CVI_ISP_GetExposureAttr(ViPipe, &stExpAttr);
stExpAttr.bByPass = CVI_FALSE;
```

(下页继续)

(续上页)

```
stExpAttr.enOpType = OP_TYPE_AUTO;
stExpAttr.stAuto.stExpTimeRange.u32Max = 40000;
stExpAttr.stAuto.stExpTimeRange.u32Min = 10;
CVI_ISP_SetExposureAttr(ViPipe, &stExpAttr);

stExpAttr.stAuto.u8Speed = 0x40;
CVI_ISP_SetExposureAttr(ViPipe, &stExpAttr);

stExpAttr.stAuto.enAESTrategyMode = AE_EXP_HIGHLIGHT_PRIOR;
stExpAttr.stAuto.u16HistRatioSlope = 0x8;
stExpAttr.stAuto.u8MaxHistOffset = 0x10;
CVI_ISP_SetExposureAttr(ViPipe, &stExpAttr);

stExpAttr.stAuto.stAntiflicker.bEnable = CVI_TRUE;
stExpAttr.stAuto.stAntiflicker.enFrequency = AE_FREQUENCY_50HZ;
stExpAttr.stAuto.stAntiflicker.enMode = ISP_ANTIFLICKER_NORMAL_MODE;
CVI_ISP_SetExposureAttr(ViPipe, &stExpAttr);

stExpAttr.stAuto.stAEDelayAttr.u16BlackDelayFrame = 10;
stExpAttr.stAuto.stAEDelayAttr.u16WhiteDelayFrame = 0;
CVI_ISP_SetExposureAttr(ViPipe, &stExpAttr);

//手动曝光属性设置(使用 gain 控制):
VI_PIPE ViPipe = 0;
ISP_EXPOSURE_ATTR_S stExpAttr;
CVI_ISP_GetExposureAttr(ViPipe, &stExpAttr);
stExpAttr.bByPass = CVI_FALSE;
stExpAttr.enOpType = OP_TYPE_MANUAL;
stExpAttr.stManual.enExpTimeOpType = OP_TYPE_MANUAL;
stExpAttr.stManual.enAGainOpType = OP_TYPE_MANUAL;
stExpAttr.stManual.enDGainOpType = OP_TYPE_MANUAL;
stExpAttr.stManual.enISPDGainOpType = OP_TYPE_MANUAL;

stExpAttr.stManual.enGainType = AE_TYPE_GAIN;
stExpAttr.stManual.u32AGain = 0x400;
stExpAttr.stManual.u32DGain = 0x400;
stExpAttr.stManual.u32ISPDGain = 0x400;
stExpAttr.stManual.u32ExpTime = 0x40000;
CVI_ISP_SetExposureAttr(ViPipe, &stExpAttr);

//手动曝光属性设置(使用 ISO Num 控制):
VI_PIPE ViPipe = 0;
ISP_EXPOSURE_ATTR_S stExpAttr;
CVI_ISP_GetExposureAttr(ViPipe, &stExpAttr);
stExpAttr.bByPass = CVI_FALSE;
stExpAttr.enOpType = OP_TYPE_MANUAL;
stExpAttr.stManual.enExpTimeOpType = OP_TYPE_MANUAL;
stExpAttr.stManual.enISONumOpType = OP_TYPE_MANUAL;

stExpAttr.stManual.enGainType = AE_TYPE_ISO;
stExpAttr.stManual.u32ISONum = 1600;
stExpAttr.stManual.u32ExpTime = 0x40000;
CVI_ISP_SetExposureAttr(ViPipe, &stExpAttr);
```

(下页继续)

(续上页)

```
//自动模式设定最大增益 32x(使用 gain 控制)
VI_PIPE ViPipe = 0;
ISP_EXPOSURE_ATTR_S stExpAttr;
CVI_ISP_GetExposureAttr(ViPipe, &stExpAttr);
stExpAttr.bByPass = CVI_FALSE;
stExpAttr.enOpType = OP_TYPE_AUTO;
stExpAttr.stAuto.enGainType = AE_TYPE_GAIN;
stExpAttr.stAuto.stSysGainRange.u32Max = 32767;
CVI_ISP_SetExposureAttr(ViPipe, &stExpAttr);

//自动模式设定最大增益 32x(使用 ISO Num 控制)
VI_PIPE ViPipe = 0;
ISP_EXPOSURE_ATTR_S stExpAttr;
CVI_ISP_GetExposureAttr(ViPipe, &stExpAttr);
stExpAttr.bByPass = CVI_FALSE;
stExpAttr.enOpType = OP_TYPE_AUTO;
stExpAttr.stAuto.enGainType = AE_TYPE_ISO;
stExpAttr.stAuto.stISONumRange.u32Max = 3200;
CVI_ISP_SetExposureAttr(ViPipe, &stExpAttr);

//快门模式(33333 us)设定最大增益 32x
VI_PIPE ViPipe = 0;
ISP_EXPOSURE_ATTR_S stExpAttr;
CVI_ISP_GetExposureAttr(ViPipe, &stExpAttr);
stExpAttr.bByPass = CVI_FALSE;
stExpAttr.enOpType = OP_TYPE_AUTO;
stExpAttr.stAuto.stExpTimeRange.u32Min = 33333;
stExpAttr.stAuto.stExpTimeRange.u32Max = 33333;
stExpAttr.stAuto.enGainType = AE_TYPE_ISO;
stExpAttr.stAuto.stISONumRange.u32Max = 3200;
CVI_ISP_SetExposureAttr(ViPipe, &stExpAttr);

//手动模式设固定增益(32x, 使用 ISO Num)
VI_PIPE ViPipe = 0;
ISP_EXPOSURE_ATTR_S stExpAttr;
CVI_ISP_GetExposureAttr(ViPipe, &stExpAttr);
stExpAttr.bByPass = CVI_FALSE;
stExpAttr.enOpType = OP_TYPE_MANUAL;
stExpAttr.stManual.enGainType = AE_TYPE_ISO;
stExpAttr.stManual.enISONumOpType = OP_TYPE_MANUAL;
stExpAttr.stManual.u32ISONum = 3200;
CVI_ISP_SetExposureAttr(ViPipe, &stExpAttr);

//手动模式设固定增益(32x, 使用 Gain)
VI_PIPE ViPipe = 0;
ISP_EXPOSURE_ATTR_S stExpAttr;
CVI_ISP_GetExposureAttr(ViPipe, &stExpAttr);
stExpAttr.bByPass = CVI_FALSE;
stExpAttr.enOpType = OP_TYPE_MANUAL;
stExpAttr.stManual.enGainType = AE_TYPE_ISO;
stExpAttr.stManual.enAGainOpType = OP_TYPE_MANUAL;
stExpAttr.stManual.enDGainOpType = OP_TYPE_MANUAL;
stExpAttr.stManual.enISPDGainOpType = OP_TYPE_MANUAL;
```

(下页继续)

(续上页)

```

stExpAttr.stManual.u32AGain = 0x8000;
stExpAttr.stManual.u32DGain = 0x400;
stExpAttr.stManual.u32ISPDGain = 0x400;
CVI_ISP_SetExposureAttr(ViPipe, &stExpAttr);

//自动快门
VI_PIPE ViPipe = 0;
ISP_EXPOSURE_ATTR_S stExpAttr;
CVI_ISP_GetExposureAttr(ViPipe, &stExpAttr);
stExpAttr.bByPass = CVI_FALSE;
stExpAttr.enOpType = OP_TYPE_AUTO;
CVI_ISP_SetExposureAttr(ViPipe, &stExpAttr);

//手动设定快门 (16384 us)
VI_PIPE ViPipe = 0;
ISP_EXPOSURE_ATTR_S stExpAttr;
CVI_ISP_GetExposureAttr(ViPipe, &stExpAttr);
stExpAttr.bByPass = CVI_FALSE;
stExpAttr.stManual.enExpTimeOpType = OP_TYPE_MANUAL;
stExpAttr.stManual.u32ExpTime = 16384;
CVI_ISP_SetExposureAttr(ViPipe, &stExpAttr);

//快门模式 (16384 us)
VI_PIPE ViPipe = 0;
ISP_EXPOSURE_ATTR_S stExpAttr;
CVI_ISP_GetExposureAttr(ViPipe, &stExpAttr);
stExpAttr.bByPass = CVI_FALSE;
stExpAttr.enOpType = OP_TYPE_AUTO;
stExpAttr.stAuto.stExpTimeRange.u32Min = 16384;
stExpAttr.stAuto.stExpTimeRange.u32Max = 16384;
CVI_ISP_SetExposureAttr(ViPipe, &stExpAttr);

//设定 antiflicker 50Hz
VI_PIPE ViPipe = 0;
ISP_EXPOSURE_ATTR_S stExpAttr;
CVI_ISP_GetExposureAttr(ViPipe, &stExpAttr);
stExpAttr.stAuto.stAntiflicker.bEnable = 1;
stExpAttr.stAuto.stAntiflicker.enMode = ISP_ANTIFLICKER_NORMAL_MODE;
stExpAttr.stAuto.stAntiflicker.enFrequency = AE_FREQUENCY_50HZ;
CVI_ISP_SetExposureAttr(ViPipe, &stExpAttr);

//设定各 LV 的目标亮度
#define TABLE_SIZE 21
CVI_U8 target_max[TABLE_SIZE] = {15, 15, 15, 15,
    15, 15, 15, 20, 20, 25, 30, 35, 40, 40, 50, 50, 55, 60, 60,
    60, 60};
CVI_U8 target_min[TABLE_SIZE] = {40, 40, 40, 40, 40, 40, 40, 40, 40, 40, 40, 40, 45,
    50, 50, 50, 50, 50, 50, 60, 60, 60};

CVI_U8 i;
VI_PIPE ViPipe = 0;
ISP_EXPOSURE_ATTR_S stExpAttr;
CVI_ISP_GetExposureAttr(ViPipe, &stExpAttr);

```

(下页继续)

(续上页)

```

for (i = 0; i < TABLE_SIZE; i++) {
    stExpAttr.stAuto.au8AdjustTargetMin[i] = target_min[i];
    stExpAttr.stAuto.au8AdjustTargetMax[i] = target_max[i];
}
CVI_ISP_SetExposureAttr(ViPipe, &stExpAttr);

```

【相关主题】

- [CVI_ISP_GetExposureAttr](#)

4.4.2.2 CVI_ISP_GetExposureAttr**【描述】**

获取 AE 曝光属性。

【语法】

```
CVI_S32 CVI_ISP_GetExposureAttr(VI_PIPE ViPipe, ISP_EXPOSURE_ATTR_S *pstExpAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstExpAttr	AE 曝光属性结构体指针	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 错误码 。

【需求】

- 头文件：cvi_ae.h
- 库文件：libae.a

【注意】

无。

【举例】

无。

【相关主题】

- [CVI_ISP_SetExposureAttr](#)

4.4.2.3 CVI_ISP_SetWDRExposureAttr

【描述】

设置 WDR 模式下的 AE 曝光属性。

【语法】

```
CVI_S32 CVI_ISP_SetWDRExposureAttr(VI_PIPE ViPipe, const ISP_WDR_EXPOSURE_
→ATTR_S *pstWDRExpAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstWDRExpAttr	WDR 模式下的 AE 曝光属性结构体指针	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_ae.h
- 库文件：libae.a

【注意】

无。

【举例】

```
//手动设定曝光比 8x
VI_PIPE ViPipe = 0;

ISP_WDR_EXPOSURE_ATTR_S stWdrExpAttr;
CVI_ISP_GetWDRExposureAttr(ViPipe, &stWdrExpAttr);
stWdrExpAttr.enExpRatioType = OP_TYPE_MANUAL;
stWdrExpAttr.au32ExpRatio[0] = 512;
CVI_ISP_SetWDRExposureAttr(ViPipe, &stWdrExpAttr);

//设定自动模式曝光比限制在 8x ~ 16x
ISP_WDR_EXPOSURE_ATTR_S stWdrExpAttr;
CVI_ISP_GetWDRExposureAttr(ViPipe, &stWdrExpAttr);
stWdrExpAttr.enExpRatioType = OP_TYPE_AUTO;
stWdrExpAttr.u32ExpRatioMin = 512;
stWdrExpAttr.u32ExpRatioMax = 1024;
CVI_ISP_SetWDRExposureAttr(ViPipe, &stWdrExpAttr);

//设定长/短帧各 LV 的目标亮度
#define TABLE_SIZE 21
CVI_U8 LeTarget[TABLE_SIZE] = {15, 15, 15, 15,
```

(下页继续)

(续上页)

```

15, 15, 15, 20, 20, 25, 30, 35, 40, 40, 50, 50, 55, 60, 60,
60, 60};

CVI_U8 SeTarget[TABLE_SIZE] = {5, 5, 5, 5,
5, 5, 5, 10, 10, 15, 15, 15, 15, 20, 20, 20, 20, 20,
20, 20};

CVI_U8 i;
ISP_WDR_EXPOSURE_ATTR_S stWdrExpAttr;
CVI_ISP_GetWDRExposureAttr(ViPipe, &stWdrExpAttr);

for (i = 0; i < TABLE_SIZE; i++) {
    stWdrExpAttr.au8LEAdjustTargetMin[i] =
    stWdrExpAttr.au8LEAdjustTargetMax[i] = LeTarget[i];

    stWdrExpAttr.au8SEAdjustTargetMin[i] =
    stWdrExpAttr.au8SEAdjustTargetMax[i] = SeTarget[i];
}
CVI_ISP_SetWDRExposureAttr(ViPipe, &stWdrExpAttr);

```

【相关主题】

- CVI_ISP_GetWDRExposureAttr

4.4.2.4 CVI_ISP_GetWDRExposureAttr**【描述】**

获取 WDR 模式下的 AE 曝光属性。

【语法】

```

CVI_S32 CVI_ISP_GetWDRExposureAttr(VI_PIPE ViPipe, ISP_WDR_EXPOSURE_ATTR_S_
↪ *pstWDRExpAttr);

```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstWDRExpAttr	WDR 模式下的 AE 曝光属性结构体指针	输出

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件: cvi_ae.h

- 库文件: libae.a

【注意】

无。

【举例】

无。

【相关主题】

- CVI_ISP_SetWDRExposureAttr

4.4.2.5 CVI_ISP_SetAERouteAttr

【描述】

设置 AE 曝光分配策略。

【语法】

```
CVI_S32 CVI_ISP_SetAERouteAttr(VI_PIPE ViPipe, const ISP_AE_ROUTE_S_  
→*pstAERouteAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstAERouteAttr	AE 曝光分配策略结构体指针	输入

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见错误码。

【需求】

- 头文件: cvi_ae.h
- 库文件: libae.a

【注意】

此接口用于设定 AE 曝光分配路线, AE 计算得到的曝光量将按照设定的路线进行分配, 用户可以根据自己的需求设定曝光优先、增益优先、光圈优先。

AE 分配路线遵循以下规则:

- 最大支持 16 个节点, 每个节点有曝光时间、增益、光圈三个分量, 增益包含模拟增益、数字增益、ISP 数字增益。
- 节点中曝光时间的单位为 us, 不能设置为 0, 也不能设置太小导致实际对应的曝光行数为 0, 否则可能产生异常。

- 光圈分量仅支持 P-Iris，不支持 DC-Iris，因为 DC-Iris 无法精确控制，所以 DC-Iris 和手动光圈镜头光圈分量是无效的。即光圈类型为 DC-Iris 时，节点光圈分量不会对曝光量分配产生任何影响。
- 节点的曝光量是曝光时间、增益和光圈的乘积，节点曝光量需为单调递增，后一个节点的曝光量应大于或等于前一个节点的曝光量，第一个节点的曝光量最小，最后一个节点的曝光量最大。
- 如果相邻节点的曝光量增加，那么应该有一个分量增加，其他分量固定，增加的分量决定该段路线的分配策略。例如曝光时间分量增加，那么该段路线的分配策略是曝光时间优先。
- 不支持设置等曝光量节点。

用户可以根据不同的场景设置不同的路线，分配路线支持动态切换。

- 针对 DC-Iris 和手动光圈镜头，默认 AE 分配策略是首先分配曝光时间，其次分配增益。针对 P-Iris 镜头，默认 AE 分配策略是首先调节光圈，将光圈调至最大后调节曝光时间，最后再分配增益。如果当前曝光量不在用户设定的路线范围当中，按默认策略分配。
- 在线进行 DC-Iris 和 P-Iris 切换，AE route 会重置为与光圈类型相匹配的默认分配策略，用户可以根据需要在切换光圈类型时自行设置 AE route。
- 自动降帧时，最大曝光时间的改变会更新到分配路线中。
- 帧率切换时，若用户设置的最大曝光目标时间大于切换后 1 帧所允许的最大曝光时间，那么分配路线的最大曝光时间会更新为切换后 1 帧所允许的最大曝光时间。
- 发生自动降帧、线性与 WDR 模式切换、帧率切换、限制曝光时间或增益的最大最小值等情况时，实际生效的 AE route 可能与 MPI 设置的不一致，此时可以通过 CVI_ISP_QueryExposureInfo 获取实际生效的 AE route。

【举例】

```
//设定 gain 128x 的 route
VI_PIPE ViPipe = 0;

ISP_AE_ROUTE_S stRoute;
CVI_ISP_SetAERouteAttr(ViPipe, &stRoute);

stRoute.u32TotalNum = 3;

stRoute.astRouteNode[0].u32IntTime = 30;
stRoute.astRouteNode[0].u32SysGain = 1024;
stRoute.astRouteNode[0].enIrisFNO = 10;
stRoute.astRouteNode[0].u32IrisFNOLin = 1024;

stRoute.astRouteNode[1].u32IntTime = 33333;
stRoute.astRouteNode[1].u32SysGain = 1024;
stRoute.astRouteNode[1].enIrisFNO = 10;
stRoute.astRouteNode[1].u32IrisFNOLin = 1024;

stRoute.astRouteNode[2].u32IntTime = 33333;
stRoute.astRouteNode[2].u32SysGain = 131072;
stRoute.astRouteNode[2].enIrisFNO = 10;
stRoute.astRouteNode[2].u32IrisFNOLin = 1024;

CVI_ISP_SetAERouteAttr(ViPipe, &stRoute);
```

【相关主题】

- CVI_ISP_GetAERouteAttr
- ISP_AE_ROUTE_S

4.4.2.6 CVI_ISP_GetAERouteAttr**【描述】**

获取 AE 曝光分配策略。

【语法】

```
CVI_S32 CVI_ISP_GetAERouteAttr(VI_PIPE ViPipe, ISP_AE_ROUTE_S *pstAERouteAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstAERouteAttr	AE 曝光分配策略结构体指针	输出

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_ae.h
- 库文件：libae.a

【注意】

无。

【举例】

无。

【相关主题】

- CVI_ISP_SetAERouteAttr

4.4.2.7 CVI_ISP_SetAERouteAttrEx

【描述】

设置 AE 曝光分配扩展属性，支持分别设置 AE 分配策略中的 sensor 模拟增益，sensor 数字增益和 ISP 数字增益。

【语法】

```
CVI_S32 CVI_ISP_SetAERouteAttrEx(VI_PIPE ViPipe, const ISP_AE_ROUTE_EX_S_
→*pstAERouteAttrEx);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstAERouteAttrEx	AE 曝光分配策略扩展属性结构体指针	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_ae.h
- 库文件：libae.a

【注意】

- 此接口用于设定 AE 曝光分配扩展属性，AE 计算得到的曝光量将按照设定的路线进行分配，用户可以根据自己的需求设定曝光时间优先、sensor 模拟增益优先、sensor 数字增益优先、ISP 数字增益优先和光圈优先。该接口可用于设置 WDR 模式下的曝光分配路线，减轻正常室内照度多帧合成 WDR 产生的工频闪现象，优化 WDR 模式图像效果。
- AE 曝光分配扩展属性是否生效可通过配置 CVI_ISP_SetExposureAttr 接口中的 bAERouteExValid 来实现。bAERouteExValid 为 CVI_TRUE 时使用扩展 AE route，否则使用正常 AE route。
- 其它注意事项与 CVI_ISP_SetAERouteAttr 的规则相同

【举例】

无。

【相关主题】

- CVI_ISP_GetAERouteAttrEx
- ISP_AE_ROUTE_EX_S

4.4.2.8 CVI_ISP_GetAERouteAttrEx

【描述】

获取 AE 曝光分配策略扩展属性。

【语法】

```
CVI_S32 CVI_ISP_GetAERouteAttrEx(VI_PIPE ViPipe, ISP_AE_ROUTE_EX_S_
↪ *pstAERouteAttrEx);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstAERouteAttrEx	AE 曝光分配策略扩展属性结构体指针	输出

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_ae.h
- 库文件：libae.a

【注意】

无。

【举例】

```
VI_PIPE ViPipe = 0;

ISP_EXPOSURE_ATTR_S stExpAttr;
ISP_AE_ROUTE_EX_S stRouteEx;
CVI_U32 au32RouteExNode[6][5]
= {{ 30, 1024, 1024, 1024, 0},
{ 30, 1024, 1024, 1024, 10},
{ 30, 16384, 1024, 1024, 10},
{400000, 16384, 1024, 1024, 10},
{400000, 16384, 4096, 1024, 10},
{400000, 16384, 4096, 4096, 10}};
CVI_ISP_GetAERouteAttrEx(ViPipe, &stRouteEx);
CVI_ISP_GetExposureAttr(ViPipe, &stExpAttr);
stExpAttr.bAERouteExValid = CVI_TRUE;
stRouteEx.u32TotalNum = 6;
memcpy(stRouteEx.astRouteExNode, au32RouteExNode, sizeof(au32RouteExNode));
```

【相关主题】

- CVI_ISP_SetAERouteAttrEx

4.4.2.9 CVI_ISP_QueryExposureInfo

【描述】

获取 AE 内部状态信息，包括 256 段直方图和平均亮度等统计信息，同时还可获取 AE 运行状态中的曝光时间、增益、曝光量、当帧的亮度、目前的环境亮度和实际生效的 AE route 等信息。

【语法】

```
CVI_S32 CVI_ISP_QueryExposureInfo(VI_PIPE ViPipe, ISP_EXP_INFO_S *pstExpInfo);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstExpInfo	曝光内部状态信息结构体指针	输出

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_ae.h
- 库文件：libae.a

【注意】

获取的曝光时间以微秒 (us) 为单位，获取的 sensor 模拟增益、sensor 数字增益和 ISP 数字增益以倍数为单位，精度是 10bit。

【举例】

```
// get sensor again, sensor dgain, isp dgain and exposure ratio
VI_PIPE ViPipe = 0;
ISP_EXP_INFO_S stExpInfo;

CVI_ISP_QueryExposureInfo(ViPipe, &stExpInfo);

printf("sensor used time = %d\n", stExpInfo. u32ExpTime);
printf("sensor used again = %d\n", stExpInfo. u32AGain);
printf("sensor used dgain = %d\n", stExpInfo. u32DGain);
printf("isp used dgain = %d\n", stExpInfo. u32ISPDGain);
printf("WDR Exposure ratio = %d\n", stExpInfo. u32WDRExpRatio);
printf("Light Value = %f\n", stExpInfo. fLightValue);
```

【相关主题】

无。

4.4.2.10 CVI_ISP_SetAntiFlicker

【描述】

设置 AE anti flicker 功能。

【语法】

```
CVI_S32 CVI_ISP_SetAntiFlicker(VI_PIPE ViPipe, CVI_BOOL enable, CVI_U8 frequency);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
enable	AE anti flicker 使能	输入
frequency	抗闪频率	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_ae.h
- 库文件：libae.a

【注意】

无。

【举例】

```
//设定 AE anti flicker 50Hz
VI_PIPE ViPipe = 0;
CVI_U8 enable, frequency;
enable = 1;
frequency = 50;

CVI_ISP_SetAntiFlicker(ViPipe, enable, frequency);
```

【相关主题】

- CVI_ISP_GetAntiFlicker

4.4.2.11 CVI_ISP_GetAntiFlicker

【描述】

获取 AE anti flicker 设定。

【语法】

```
CVI_S32 CVI_ISP_GetAntiFlicker(VI_PIPE ViPipe, CVI_BOOL *pEnable, CVI_U8 *pFrequency);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pEnable	AE anti flicker 使能指针	输入
pFrequency	抗闪频率指针	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_ae.h
- 库文件：libae.a

【注意】

无。

【举例】

无。

【相关主题】

无。

4.4.2.12 CVI_ISP_QueryFps

【描述】

获取 AE 当前 fps。

【语法】

```
CVI_S32 CVI_ISP_QueryFps(VI_PIPE ViPipe, CVI_FLOAT *pFps);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pFps	AE fps 指针	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 错误码 。

【需求】

- 头文件：cvi_ae.h
- 库文件：libae.a

【注意】

无。

【举例】

无。

【相关主题】

- [CVI_ISP_QueryExposureInfo](#)

4.4.2.13 CVI_ISP_GetCurrentLvX100**【描述】**

用于获取当前 AE 的测光结果，即环境光亮度值 LV。可用于判断黑夜和白天，达到类似光敏电阻的效果。

【语法】

```
CVI_S32 CVI_ISP_GetCurrentLvX100(VI_PIPE ViPipe, CVI_S16 *ps16Lv);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
ps16Lv	返回当前 LV 的值	输出

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见 错误码 。

【需求】

- 头文件: cvi_ae.h
- 库文件: libae.a

【注意】

无。

【举例】

```
#include "cvi_ae.h"

#define ENTER_NIGHT_LV_LEVEL 0
#define ENTER_DAY_LV_LEVEL 700
#define CHECK_COUNT 5

bool checkDayOrNight(void)
{
    static bool dayOrNight = true;
    static CVI_U8 checkDayCount = 0;
    static CVI_U8 checkNightCount = 0;
    CVI_S16 lv = 0;

    CVI_ISP_GetCurrentLvX100(0, &lv);
    if (lv > ENTER_DAY_LV_LEVEL) {
        if (checkDayCount < CHECK_COUNT) {
            checkDayCount++;
        } else {
            dayOrNight = true;
        }
        checkNightCount = 0;
    } else if (lv < ENTER_NIGHT_LV_LEVEL) {
        if (checkNightCount < CHECK_COUNT) {
            checkNightCount++;
        } else {
            dayOrNight = false;
        }
        checkDayCount = 0;
    } else {
        checkDayCount = 0;
        checkNightCount = 0;
    }
    return dayOrNight;
}

int main(void)
{
    while (1) {
        sleep(1);
        printf("checkDayOrNight: %s\n", checkDayOrNight() ? "day" : "night");
    }
    return 0;
}
```

【相关主题】

无。

4.4.2.14 CVI_ISP_SetSmartExposureAttr

【描述】

设置智能模式下的 AE 曝光属性。仅在有智能信息时生效。

【语法】

```
CVI_S32 CVI_ISP_SetSmartExposureAttr(VI_PIPE ViPipe, const ISP_SMART_EXPOSURE_
→ATTR_S *pstSmartExpAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstSmartExpAttr	智能模式下的 AE 曝光属性结构体指针	输出

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_ae.h
- 库文件：libae.a

【注意】

无。

【举例】

```
#include "cvi_ae.h"

VI_PIPE ViPipe = 0;
ISP_SMART_EXPOSURE_ATTR_S stSmartExpAttr;

CVI_ISP_GetSmartExposureAttr(ViPipe, &stSmartExpAttr);

stSmartExpAttr.bEnable = 1;
stSmartExpAttr.u8LumaTarget = 56;
stSmartExpAttr.u16ExpCoef= 1024;
stSmartExpAttr.u16ExpCoefMax= 4096;
stSmartExpAttr.u16ExpCoefMin = 256;
stSmartExpAttr.u8SmartInterval = 1;
stSmartExpAttr.u8SmartSpeed = 32;
stSmartExpAttr.u16SmartDelayNum = 5;
stSmartExpAttr.u8Weight = 80;
stSmartExpAttr.u8NarrowRatio = 75;

CVI_ISP_SetSmartExposureAttr(ViPipe, &stSmartExpAttr);
```

【相关主题】

无。

4.4.2.15 CVI_ISP_GetSmartExposureAttr**【描述】**

获取智能模式下的 AE 曝光属性。仅在有智能信息时生效。

【语法】

```
CVI_S32 CVI_ISP_GetSmartExposureAttr(VI_PIPE ViPipe, ISP_SMART_EXPOSURE_ATTR_S_
→*pstSmartExpAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstSmartExpAttr	智能模式下的 AE 曝光属性结构体指针	输出

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_ae.h
- 库文件：libae.a

【注意】

无。

【举例】

无。

【相关主题】

- CVI_ISP_SetSmartExposureAttr

4.4.2.16 CVI_ISP_SetAERouteSFAttr

【描述】

设置 AE WDR 短侦曝光分配策略。

【语法】

```
CVI_S32 CVI_ISP_SetAERouteSFAttr(VI_PIPE ViPipe, const ISP_AE_ROUTE_S_  
→*pstAERouteSFAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstAERouteSFAttr	AE WDR 短侦曝光分配策略结构体指针	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_ae.h
- 库文件：libae.a

【注意】

无。

【举例】

无。

【相关主题】

- CVI_ISP_GetAERouteSFAttr
- ISP_AE_ROUTE_S

4.4.2.17 CVI_ISP_GetAERouteSFAttr

【描述】

获取 AE WDR 短侦曝光分配策略。

【语法】

```
CVI_S32 CVI_ISP_GetAERouteSFAttr(VI_PIPE ViPipe, ISP_AE_ROUTE_S_  
→*pstAERouteSFAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstAERouteSFAttr	AE WDR 短侦曝光分配策略结构体指针	输出

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_ae.h
- 库文件：libae.a

【注意】

无。

【举例】

无。

【相关主题】

- CVI_ISP_SetAERouteSFAttr

4.4.2.18 CVI_ISP_SetAERouteSFAttrEx

【描述】

设置 AE WDR 短侦曝光分配扩展属性，支持分别设置 AE 分配策略中的 sensor 模拟增益，sensor 数字增益和 ISP 数字增益。

【语法】

```
CVI_S32 CVI_ISP_SetAERouteSFAttrEx(VI_PIPE ViPipe, const ISP_AE_ROUTE_EX_S_
↪ *pstAERouteSFAttrEx);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstAERouteSFAttrEx	AE WDR 短侦曝光分配策略扩展属性结构体指针	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件: `cvi_ae.h`
- 库文件: `libae.a`

【注意】

无。

【举例】

无。

【相关主题】

- `CVI_ISP_GetAERouteSFAttrEx`
- `ISP_AE_ROUTE_EX_S`

4.4.2.19 CVI_ISP_GetAERouteSFAttrEx

【描述】

获取 AE WDR 短侦曝光分配策略扩展属性。

【语法】

```
CVI_S32 CVI_ISP_GetAERouteSFAttrEx(VI_PIPE ViPipe, ISP_AE_ROUTE_EX_S_
→ *pstAERouteSFAttrEx);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstAERouteSFAttrEx	AE WDR 短侦曝光分配策略扩展属性结构体指针	输出

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见错误码。

【需求】

- 头文件: `cvi_ae.h`
- 库文件: `libae.a`

【注意】

无。

【举例】

无。

【相关主题】

- CVI_ISP_SetAERouteSFAttrEx

4.4.3 AI 控制模块

4.4.3.1 CVI_ISP_SetIrisAttr

【描述】

设定光圈的控制属性，该函数可实现手动光圈属性和光圈类型等参数的设置。

【语法】

```
CVI_S32 CVI_ISP_SetIrisAttr(VI_PIPE ViPipe, const ISP_IRIS_ATTR_S *pstIrisAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstIrisAttr	光圈控制属性结构体指针	输出

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_ae.h
- 库文件：libae.a

【注意】

- 根据实际对接镜头光圈类型，设置正确的光圈类型属性，由此再去设置相关的 DC-Iris/P-Iris 控制属性。若对接的是手动光圈镜头，可将光圈类型设置为 ISP_IRIS_DC_TYPE，建议此时关闭 AI 使能。
- 手动光圈属性主要用于调试，可通过该 MPI 进行设置。对于 P-Iris 镜头，手动 enIrisFNO 值会受到最大、最小光圈目标值的影响。自动光圈属性的更多参数需要调用 CVI_ISP_SetDcirisAttr 和 CVI_ISP_SetPirisAttr 进行设置。

【举例】

无。

【相关主题】

- ISP_IRIS_ATTR_S
- CVI_ISP_SetDcirisAttr
- CVI_ISP_SetPirisAttr

4.4.3.2 CVI_ISP_GetIrisAttr

【描述】

获取光圈的控制属性。

【语法】

```
CVI_S32 CVI_ISP_GetIrisAttr(VI_PIPE ViPipe, ISP_IRIS_ATTR_S *pstIrisAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstIrisAttr	光圈控制属性结构体指针	输出

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_ae.h
- 库文件：libae.a

【注意】

无。

【举例】

无。

【相关主题】

无。

4.4.3.3 CVI_ISP_SetDcirisAttr

【描述】

设定 DC-Iris AI 算法的控制属性，该函数可实现 DC-Iris 自动光圈的参数设置。

【语法】

```
CVI_S32 CVI_ISP_SetDcirisAttr(VI_PIPE ViPipe, const ISP_DCIRIS_ATTR_S *pstDcirisAttr);;
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstIrisAttr	DC-Iris 自动光圈控制属性结构体指针	输入/输出

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_ae.h
- 库文件：libae.a

【注意】

DC-Iris 光圈控制采用 PID 算法，算法根据画面亮度，调节 PWM 占空比对光圈大小进行控制。当曝光时间和增益达到最小目标值之后，会进入光圈控制区域。当光圈控制能满足目标亮度的要求时，AE 直接返回，保持曝光时间和增益不变。当画面亮度稳定且 PWM 占空比维持在打开值一段时间后，AI 算法会认为光圈已经打开至最大，退出光圈控制区，将控制权交还给 AE

【举例】

无。

【相关主题】

- [ISP_IRIS_ATTR_S](#)
- [ISP_DCIRIS_ATTR_S](#)

4.4.3.4 CVI_ISP_GetDcirisAttr**【描述】**

获取 DC-Iris 自动光圈的控制属性。

【语法】

```
CVI_S32 CVI_ISP_GetDcirisAttr(VI_PIPE ViPipe, ISP_DCIRIS_ATTR_S *pstDcirisAttr);;
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstIrisAttr	DC-Iris 自动光圈控制属性结构体指针	输出

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_ae.h
- 库文件：libae.a

【注意】

无。

【举例】

无。

【相关主题】

无。

4.4.3.5 CVI_ISP_SetPirisAttr

【描述】

设定 P-Iris 自动光圈的控制属性

【语法】

```
CVI_S32 CVI_ISP_SetPirisAttr(VI_PIPE ViPipe, const ISP_PIRIS_ATTR_S *pstPirisAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstPirisAttr	P-Iris 自动光圈控制属性结构体指针	输出

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_ae.h
- 库文件：libae.a

【注意】

目前尚未支持 P-iris

【举例】

无。

【相关主题】

- [ISP_PIRIS_ATTR_S](#)

4.4.3.6 CVI_ISP_GetPirisAttr

【描述】

获取 P-Iris 自动光圈的控制属性。

【语法】

```
CVI_S32 CVI_ISP_GetPirisAttr(VI_PIPE ViPipe, ISP_PIRIS_ATTR_S *pstPirisAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstIrisAttr	P-Iris 自动光圈控制属性结构体指针	输出

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_ae.h
- 库文件：libae.a

【注意】

目前尚未支持 P-iris

【举例】

无。

【相关主题】

- [ISP_PIRIS_ATTR_S](#)

4.5 数据类型

4.5.1 Register

- `AE_SENSOR_REGISTER_S`：定义 sensor 注册结构体。
- `AE_SENSOR_EXP_FUNC_S`：定义 sensor 回调函数结构体。
- `AE_SENSOR_DEFAULT_S`：定义 AE 算法库初始化的参数结构体。
- `AE_ACCURACY_E`：定义曝光时间、增益精度类型的枚举。
- `AE_ACCURACY_S`：定义曝光时间、增益精度的结构体。

4.5.1.1 AE_SENSOR_REGISTER_S

【说明】

定义 sensor 注册结构体。

【定义】

```
typedef struct _AE_SENSOR_REGISTER_S
{
    AE_SENSOR_EXP_FUNC_S stAeExp;
} AE_SENSOR_REGISTER_S;
```

【成员】

成员名称	描述
stAeExp	Sensor 注册的回调函数结构体

【注意事项】

无。

【相关数据类型及接口】

- `AE_SENSOR_EXP_FUNC_S`

4.5.1.2 AE_SENSOR_EXP_FUNC_S

【说明】

定义 sensor 回调函数结构体。

【定义】

```
typedef struct _AE_SENSOR_EXP_FUNC_S
{
    CVI_S32(*pfn_cmos_get_ae_default)(VI_PIPE ViPipe, AE_SENSOR_DEFAULT_S_
    →*pstAeSnsDft);
    CVI_VOID(*pfn_cmos_fps_set)(VI_PIPE ViPipe, CVI_FLOAT f32Fps,
```

(下页继续)

(续上页)

```

AE_SENSOR_DEFAULT_S *pstAeSnsDft);
CVI_VOID(*pfn_cmos_slow_framerate_set)(VI_PIPE ViPipe, CVI_U32 u32FullLines, AE_
↪SENSOR_DEFAULT_S *pstAeSnsDft);
CVI_VOID(*pfn_cmos_inttime_update)(VI_PIPE ViPipe, CVI_U32 *u32IntTime);
CVI_VOID(*pfn_cmos_gains_update)(VI_PIPE ViPipe, CVI_U32 *u32Again, CVI_U32_
↪*u32Dgain);
CVI_VOID(*pfn_cmos_again_calc_table)(VI_PIPE ViPipe, CVI_U32 *pu32AgainLin, CVI_U32_
↪*pu32AgainDb);
CVI_VOID(*pfn_cmos_dgain_calc_table)(VI_PIPE ViPipe, CVI_U32 *pu32DgainLin, CVI_U32_
↪*pu32DgainDb);
CVI_VOID(*pfn_cmos_get_inttime_max)(VI_PIPE ViPipe, CVI_U16 u16ManRatioEnable, CVI_
↪U32 *au32Ratio, CVI_U32 *au32IntTimeMax, CVI_U32 *au32IntTimeMin, CVI_U32_
↪*pu32LFMaxIntTime);
CVI_VOID(*pfn_cmos_ae_fswdr_attr_set)(VI_PIPE ViPipe, AE_FSWDR_ATTR_S_
↪*pstAeFSWDRAttr);
} AE_SENSOR_EXP_FUNC_S;

```

【成员】

成员名称	描述
pfn_cmos_get_ae_default	获取 AE 算法库初始值的回调函数指针。
pfn_cmos_fps_set	设置 sensor 的帧率。
pfn_cmos_slow_framerate_set	设置 sensor 的降帧。
pfn_cmos_inttime_update	设置 sensor 的曝光时间。
pfn_cmos_gains_update	设置 sensor 的模拟增益和数字增益。
pfn_cmos_again_calc_table	计算 TABLE 类型 sensor 模拟增益。
pfn_cmos_dgain_calc_table	计算 TABLE 类型 sensor 数字增益。
pfn_cmos_get_inttime_max	WDR 模式下，计算短帧最大曝光时间的回调函数指针，与 sensor 强相关。
pfn_cmos_ae_fswdr_attr_set	2to1LineWDR 模式下，设置长帧模式

【注意事项】

- 如果回调函数指针不需要赋值，需要置为 NULL。
- pfn_cmos_inttime_update 和 pfn_cmos_gains_update 中设置的曝光时间和增益如何转换成 sensor 的配置值与 sensor 强相关，请参阅 sensor 手册。

【相关数据类型及接口】

- [AE_SENSOR_DEFAULT_S](#)
- [ISP_SENSOR_EXP_FUNC_S](#)

4.5.1.3 AE_SENSOR_DEFAULT_S

【说明】

定义 AE 算法库的初始化参数结构体。

【定义】

```
typedef struct _AE_SENSOR_DEFAULT_S
{
    CVI_U8 au8HistThresh[HIST_THRESH_NUM];
    CVI_U8 u8AeCompensation;
    CVI_U32 u32LinesPer500ms;
    CVI_U32 u32FlickerFreq;
    CVI_U32 u32HmaxTimes;
    CVI_U32 u32InitExposure;
    CVI_U32 u32InitAESpeed;
    CVI_U32 u32InitAETolerance;
    CVI_U32 u32FullLinesStd;
    CVI_U32 u32FullLinesMax;
    CVI_U32 u32FullLines;
    CVI_U32 u32MaxIntTime;
    CVI_U32 u32MinIntTime;
    CVI_U32 u32MaxIntTimeTarget;
    CVI_U32 u32MinIntTimeTarget;
    AE_ACCURACY_S stIntTimeAccu;
    CVI_U32 u32MaxAgain;
    CVI_U32 u32MinAgain;
    CVI_U32 u32MaxAgainTarget;
    CVI_U32 u32MinAgainTarget;
    AE_ACCURACY_S stAgainAccu;
    CVI_U32 u32MaxDgain;
    CVI_U32 u32MinDgain;
    CVI_U32 u32MaxDgainTarget;
    CVI_U32 u32MinDgainTarget;
    AE_ACCURACY_S stDgainAccu;
    CVI_U32 u32MaxISPDgainTarget;
    CVI_U32 u32MinISPDgainTarget;
    CVI_U32 u32ISPDgainShift;
    CVI_U32 u32MaxIntTimeStep;
    CVI_U32 u32LFMaxShortTime;
    CVI_U32 u32LFMinExposure;
    ISP_AE_STRATEGY_E enAeExpMode;
    CVI_U16 u16ISOCalCoef;
    CVI_U8 u8AERunInterval;
    CVI_FLOAT f32Fps;
    CVI_FLOAT f32MinFps;
    CVI_U32 denom;
    CVI_U32 u32AEResponseFrame;
    CVI_U32 u32SnsResponseFrame;
    CVI_U32 u32SnsStableFrame;
    AE_BLC_TYPE_E enBlcType;
    ISP_SNS_GAIN_MODE_E enWDRGainMode;
} AE_SENSOR_DEFAULT_S;
```

【成员】

成员名称	描述
u8AeCompensation	AE 亮度目标值，取值范围为 [0,255]，建议用 0x38~0x40。
u32LinesPer500ms	每 500ms 的总行数。
u32FlickerFreq	抗闪频率，数值为当前电源频率的 256 倍。
u32HmaxTimes	Sensor 读出一行的时间，单位：ns。
u32InitExposure	默认初始曝光量，等于曝光时间 (行数)* 系统增益 (6bit 小数精度)。AE 算法采用该值作为初始 5 帧的曝光量
u32InitAESpeed	默认初始 AE 调节速度，AE 算法采用该值作为初始 100 帧的调节速度
u32InitAETolerance	默认初始 AE 曝光容忍偏差，AE 算法采用该值作为初始 100 帧的曝光容忍偏差值，可用于设置得到首次 AE 收敛稳定标志的亮度范围
u32FullLinesStd	基准帧率时 1 帧的有效行数。
u32FullLinesMax	sensor 降帧后 1 帧所能达到的最大行数，一般设为 sensor 所支持的最大行数。
u32FullLines	sensor 每帧实际生效的总行数。使用 CVI AE 算法时，回调 cmos_fps_set 和 cmos_slow_framerate_set 时必须给该值赋值，用于返回每帧实际生效的总行数。
u32MaxIntTime	最大曝光时间，以行为单位。
u32MinIntTime	最小曝光时间，以行为单位。
u32MaxIntTimeTarget	最大曝光时间目标值，以行为单位。
u32MinIntTimeTarget	最小曝光时间目标值，以行为单位。
stIntTimeAccu	曝光时间精度。
u32MaxAgain	最大模拟增益，以倍为单位。
u32MinAgain	最小模拟增益，以倍为单位。
u32MaxAgainTarget	最大模拟增益目标值，以倍为单位。
u32MinAgainTarget	最小模拟增益目标值，以倍为单位。
stAgainAccu	模拟增益精度。
u32MaxDgain	最大数字增益，以倍为单位。
u32MinDgain	最小数字增益，以倍为单位。
u32MaxDgainTarget	最大数字增益目标值，以倍为单位。
u32MinDgainTarget	最小数字增益目标值，以倍为单位。
stDgainAccu	数字增益精度。
u32MaxISPDgainTarget	最大 ISP 数字增益目标值，以倍为单位。
u32MinISPDgainTarget	最小 ISP 数字增益目标值，以倍为单位。
u32ISPDgainShift	ISP 数字增益精度。
u32MaxIntTimeStep	短帧曝光时间减小的最大调整步长，以行数为单位。仅在两帧合成模式下有效。
u32LFMaxShortTime	自动长帧模式下短帧最大曝光时间。
u32LFMinExposure	自动长帧模式下强制输出长帧的最小曝光量
enAeExpMode	默认曝光策略，高光优先或低光优先。建议 FSWDR 模式设置为低光优先，线性模式及 BuiltInWDR 设置为高光优先。若不设置该值，默认为高光优先。
u16ISOCalCoef	ISO 标定系数，用于保证拍照所需 DCF 信息中显示的 ISO 是标准的，8bit 精度。 取值范围：[0x0, 0xFFFF]， 默认值：为 0x100。

下页继续

表 4.61 – 续上页

成员名称	描述
u8AERunInterval	默认 AE 算法运行间隔，以帧为单位。若不设置，则默认 AE 每帧执行一次。 取值范围：(0x0, 0xFF]
f32Fps	基准帧率
u32AEResponseFrame	Sensor 曝光时间/增益同步生效的所需的帧数
u32SnsResponseFrame	Sensor 反应帧数

【注意事项】

- 线性/WDR 模式切换时，会回调 pfn_cmos_get_ae_default 函数更新 AE 相关默认参数。若 WDR 模式要使用 AE 扩展分配路线而线性模式不需要，建议在 cmos_get_ae_default 函数里面先对 AE 路线清零：baERouteExValid = CVI_FALSE, stAERouteAttr.u32TotalNum = 0, stAERouteAttrEx.u32TotalNum = 0，然后视需要在 WDR 分支赋值。
- u32LFMaxShortTime 为自动长帧模式下短帧曝光时间的最大值，如果此参数设置过小会导致自动长帧模式下亮区噪声表现变差。

【相关数据类型及接口】

- ISP_SENSOR_EXP_FUNC_S

4.5.1.4 AE_ACCURACY_E**【说明】**

定义曝光时间、增益的精度类型的枚举。

【定义】

```
typedef enum _AE_ACCURACY_E
{
    AE_ACCURACY_DB = 0,
    AE_ACCURACY_LINEAR,
    AE_ACCURACY_TABLE,
    AE_ACCURACY_BUTT,
} AE_ACCURACY_E;
```

【成员】

成员名称	描述
AE_ACCURACY_DB DB	精度类型。
AE_ACCURACY_LINEAR	线性精度类型。
AE_ACCURACY_TABLE	表格类型。

【注意事项】

无。

【相关数据类型及接口】

- AE_ACCURACY_S

4.5.1.5 AE_ACCURACY_S

【说明】

定义曝光时间、增益的精度结构体

【定义】

```
typedef struct _AE_ACCURACY_S
{
    AE_ACCURACY_E enAccuType;
    float f32Accuracy;
    float f32Offset;
} AE_ACCURACY_S;
```

【成员】

成员名称	描述
enAccuType	精度类型，包括线性类型、DB 类型和 TABLE 类型。。
f32Accuracy	精度值。
f32Offset	曝光时间的偏移量，支持正负偏移量设置，以行为单位，该值配置与 sensor 强相关。

【注意事项】

无。

【相关数据类型及接口】

无。

4.5.2 AE

- ISP_AE_MODE_E：定义自动曝光的模式。
- ISP_AE_STRATEGY_E：定义 AE 曝光策略模式。
- ISP_AE_DELAY_S：定义 AE 延时属性。
- ISP_AE_RANGE_S：定义曝光时间或增益的最大值和最小值。
- ISP_ANTIFLICKER_MODE_E：定义抗闪模式。
- ISP_ANTIFLICKER_S：定义抗闪属性。
- ISP_SUBFLICKER_S：定义 ISP 图像亚抗闪属性。
- ISP_FSWDR_MODE_E：定义 ISP FSWDR 运行模式。
- ISP_AE_ATTR_S：定义自动曝光属性。
- ISP_ME_ATTR_S：定义手动曝光属性。
- ISP_EXPOSURE_ATTR_S：定义 ISP 曝光属性。
- ISP_WDR_EXPOSURE_ATTR_S：定义 WDR 模式下的曝光属性。

- `ISP_AE_ROUTE_NODE_S`：定义 AE 分配路线节点属性。
- `ISP_AE_ROUTE_S`：定义 AE 曝光分配策略属性。
- `ISP_AE_ROUTE_EX_NODE_S`：定义 AE 扩展分配路线节点属性。
- `ISP_AE_ROUTE_EX_S`：定义 AE 曝光分配策略扩展属性。
- `ISP_EXP_INFO_S`：定义 ISP 曝光内部状态信息。

4.5.2.1 ISP_AE_MODE_E

【说明】

定义自动曝光的模式。

【定义】

```
typedef enum _ISP_AE_MODE_E
{
    AE_MODE_SLOW_SHUTTER = 0,
    AE_MODE_FIX_FRAME_RATE = 1,
    AE_MODE_BUTT
} ISP_AE_MODE_E;
```

【成员】

成员名称	描述
<code>AE_MODE_SLOW_SHUTTER</code>	自动降帧模式，即 <code>SLOW_SHUTTER</code> 模式。
<code>AE_MODE_FIX_FRAME_RATE</code>	固定帧率模式。

【注意事项】

- 自动降帧模式是指自动曝光调节时会优先增大曝光时间来尽量减小增益。当 sensor 增益达到用户设置的最大值时，自动曝光调节会逐渐降帧率并延长曝光时间，持续到曝光时间等于自动曝光的最大时间为止。低照度环境下噪声较小但帧率会降低。
- 固定帧率模式是指自动曝光调节时保持帧率不变，低照度环境下噪声会较大。

【相关数据类型及接口】

无。

4.5.2.2 ISP_AE_STRATEGY_E

【说明】

定义 AE 曝光策略模式。

【定义】

```
typedef enum _ISP_AE_STRATEGY_E
{
    AE_EXP_HIGHLIGHT_PRIOR = 0,
    AE_EXP_LOWLIGHT_PRIOR = 1,
```

(下页继续)

(续上页)

```

    AE_STRATEGY_MODE_BUTT
} ISP_AE_STRATEGY_E;

```

【成员】

成员名称	描述
AE_EXP_HIGHLIGHT_PRIORITY	高光优先曝光模式。
AE_EXP_LOWLIGHT_PRIORITY	低光优先曝光模式。

【注意事项】

- 线性模式下，AE 算法默认曝光策略为高光优先，尽量避免画面过曝，但在逆光场景时，暗处亮度较低。此时若要关注暗处区域，可采用低光优先模式，但亮处容易过曝。
- WDR 模式下，AE 算法默认使用低光优先模式，控制长帧曝光，此时手动曝光时间、最大最小曝光时间、AE_Route、AE_Route_Ex 中曝光时间参数影响长帧曝光时间。如果切换为高光优先模式，则控制短帧曝光，此时手动曝光时间、最大最小曝光时间、AE_Route、AE_Route_Ex 中曝光时间参数影响短帧曝光时间。

【相关数据类型及接口】

无。

4.5.2.3 ISP_AE_DELAY_S**【说明】**

定义 AE 延时属性。

【定义】

```

typedef struct _ISP_AE_DELAY_S
{
    CVI_U16 u16BlackDelayFrame;
    CVI_U16 u16WhiteDelayFrame;
} ISP_AE_DELAY_S;

```

【成员】

成员名称	描述
u16BlackDelayFrame	图像亮度小于目标亮度时间超过 u16BlackDelayFrame 帧时，AE 开始调节。
u16WhiteDelayFrame	图像亮度大于目标亮度时间超过 u16WhiteDelayFrame 帧时，AE 开始调节。

【注意事项】

- u16BlackDelayFrame/u16WhiteDelayFrame 设置越大，在 AE 调节步长相同的情况下，调节至目标亮度需要越长时间。人眼对于过曝比较敏感，建议 u16WhiteDelayFrame 要设置得比 u16BlackDelayFrame 小一些。
- 若帧率较低时，建议 u16BlackDelayFrame/u16WhiteDelayFrame 不要设置过大，否则 AE

收敛时间会较长。

【相关数据类型及接口】

无。

4.5.2.4 ISP_AE_RANGE_S

【说明】

定义曝光时间或增益的最大值和最小值。

【定义】

```
typedef struct _ISP_AE_RANGE_S
{
    CVI_U32 u32Max;
    CVI_U32 u32Min;
} ISP_AE_RANGE_S;
```

【成员】

成员名称	描述
u32Max	最大值。
u32Min	最小值。

【注意事项】

最小值不能大于最大值。

【相关数据类型及接口】

无。

4.5.2.5 ISP_ANTIFLICKER_MODE_E

【说明】

定义抗闪模式。

【定义】

```
typedef enum _ISP_ANTIFLICKER_MODE_E
{
    ISP_ANTIFLICKER_NORMAL_MODE = 0x0,
    ISP_ANTIFLICKER_AUTO_MODE = 0x1,
    ISP_ANTIFLICKER_MODE_BUTT
} ISP_ANTIFLICKER_MODE_E;
```

【成员】

成员名称	描述
ISP_ANTIFLICKER_NORMAL_MODE	普通抗闪模式。

下页继续

表 4.68 – 续上页

成员名称	描述
ISP_ANTIFLICKER_AUTO	自动抗闪模式。

【注意事项】

- ISP_ANTIFLICKER_NORMAL_MODE 为普通抗闪模式，曝光时间可以根据亮度进行调节，最小曝光时间固定为 1/120 sec (60Hz) 或 1/100 sec(50Hz)，不受曝光时间最小值的限制。
 - 有灯光的环境：曝光时间可与光源频率相匹配，能够防止图像闪烁。
 - 高亮度的环境：亮度越高，要求曝光时间就越短。而普通抗闪模式的最小曝光时间并不会根据高亮度进行调节，因此会产生过曝情况。
- ISP_ANTIFLICKER_AUTO_MODE 为自动抗闪模式，曝光时间可以根据亮度进行调节，最小曝光时间可达到 sensor 的最小曝光时间。与普通抗闪模式的差别主要在高亮度的环境体现。
 - 高亮度的环境：最小曝光时间可以达到 sensor 的最小曝光时间，能够有效抑制过曝，但此时抗闪失效。

【相关数据类型及接口】

无。

4.5.2.6 ISP_ANTIFLICKER_S

【说明】

定义抗闪属性。

【定义】

```
typedef struct _ISP_ANTIFLICKER_S
{
    CVI_BOOL bEnable;
    ISP_AE_ANTIFLICKER_FREQUENCY_E enFrequency;
    ISP_ANTIFLICKER_MODE_E enMode;
} ISP_ANTIFLICKER_S;
```

【成员】

成员名称	描述
bEnable	bEnable 为 CVI_TRUE 时使能图像抗闪， 为 CVI_FALSE 时不使能图像抗闪。
enFrequency	抗闪频率值。 取值范围：[0, 1]， 默认值：为 0。0: 60Hz 1: 50Hz
enMode	抗闪模式，普通抗闪或自动抗闪。

【注意事项】

抗闪开启后曝光时间会受到最大/最小曝光时间的限制，若最小抗闪时间大于最大曝光时间，那

么开启抗闪后曝光时间将被限制为最大曝光时间。

【相关数据类型及接口】

- ISP_ANTIFLICKER_MODE_E

4.5.2.7 ISP_SUBFLICKER_S

【说明】

定义 ISP 图像亚抗闪属性。

【定义】

```
typedef struct _ISP_SUBFLICKER_S
{
    CVI_BOOL bEnable;
    CVI_U8  u8LumaDiff;
} ISP_SUBFLICKER_S;
```

【成员】

成员名称	描述
bEnable	bEnable 为 CVI_TRUE 时使能图像亚抗闪功能，为 CVI_FALSE 时不使能图像亚抗闪。
u8LumaDiff	抗闪程度设置，范围 [0x0, 0x64]。亚抗闪功能生效时，该值越大越接近于抗闪。

【注意事项】

- 强制抗闪模式时，最小曝光时间固定为 1/120 sec (60Hz) 或 1/100 sec(50Hz)，在某些场景（如室内对准室外窗户的背光场景）画面可能会过曝严重，但不抗闪画面工频闪又比较厉害。在这种情况下引入亚抗闪模式是为了取得过曝与闪烁之间的平衡。在强制抗闪模式下，当亚抗闪功能生效时，若画面亮度小于 (AeCompensation + u8LumaDiff)，那么曝光时间仍会固定为最小抗闪时间 1/120 sec (60Hz) 或 1/100 sec(50Hz) 以防止图像闪烁。若画面亮度大于 (AeCompensation + u8LumaDiff)，则取消抗闪，调整画面目标亮度为 (AeCompensation + u8LumaDiff)，通过引入一定程度的画面闪烁来避免画面过曝严重。
- 只有在满足打开抗闪、强制抗闪模式且抗闪频率值不等于 0 的前提下，亚抗闪功能才能生效。

【相关数据类型及接口】

无。

4.5.2.8 ISP_FSWDR_MODE_E

【说明】

定义 ISP FSWDR 运行模式。

【定义】

```
typedef enum _ISP_FSWDR_MODE_E
{
    ISP_FSWDR_NORMAL_MODE = 0x0,
    ISP_FSWDR_LONG_FRAME_MODE = 0x1,
    ISP_FSWDR_AUTO_LONG_FRAME_MODE = 0x2,
    ISP_FSWDR_MODE_BUTT
}ISP_FSWDR_MODE_E;
```

【成员】

成员名称	描述
ISP_FSWDR_NORMAL_MODE	正常 FSWDR 模式，此时 AE 和合成模块按照自动/手动曝光比进行工作。
ISP_FSWDR_LONG_FRAME_MODE	长帧模式，此时 AE 将短帧曝光时间设置为最小值，长帧曝光时间接近 1 帧所允许的最大值，合成模块只输出长帧数据，可用于优化弱宽动态场景或低照度时的图像质量。
ISP_FSWDR_AUTO_LONG_FRAME_MODE	自动长帧模式，此时 AE 短帧曝光时间不会超过 sensor 限制的长帧模式下短帧最大曝光时间，在曝光量超过 sensor 设置的阈值时，自动切换为长帧模式，合成模块只输出长帧数据。

【注意事项】

- 长帧模式仅在行模式 WDR 下有效，自动长帧模式仅在 2to1 行模式 WDR 下有效，并且为了保证图像质量，WDR 模块的运动检测及长短帧融合阈值均为自动配置，不支持手动配置。在线将模式切换到 WDR 模式，默认为正常 FSWDR 模式。
- 自动长帧模式下，手动曝光比生效，如果手动曝光比大于 1，即使曝光量大于设置的阈值也不会进入长帧模式；如果手动曝光比为 1，则会自动进入长帧模式。

【相关数据类型及接口】

无。

4.5.2.9 ISP_AE_GAIN_TYPE_E

【说明】

定义增益使用的方式。

【定义】

```
typedef enum _ISP_AE_GAIN_TYPE_E {
    AE_TYPE_GAIN = 0,
    AE_TYPE_ISO = 1,
    AE_TYPE_BUTT
}ISP_AE_GAIN_TYPE_E;
```

【成员】

成员名称	描述
enGainType	0：使用 gain 的方式 1：使用 ISO num 的方式

【注意事项】

无。

4.5.2.10 ISP_AE_ATTR_S

【说明】

定义自动曝光属性。

【定义】

```
typedef struct _ISP_AE_ATTR_S
{
    ISP_AE_RANGE_S stExpTimeRange;
    ISP_AE_RANGE_S stAGainRange;
    ISP_AE_RANGE_S stDGainRange;
    ISP_AE_RANGE_S stISPDGainRange;
    ISP_AE_RANGE_S stSysGainRange;
    CVI_U32 u32GainThreshold;
    CVI_U8 u8Speed;
    CVI_U16 u16BlackSpeedBias;
    CVI_U8 u8Tolerance;
    CVI_U8 u8Compensation;
    CVI_U16 u16EVBias;
    ISP_AE_STRATEGY_E enAESTrategyMode;
    CVI_U16 u16HistRatioSlope;
    CVI_U8 u8MaxHistOffset;
    ISP_AE_MODE_E enAEMode;
    ISP_ANTIFLICKER_S stAntiflicker;
    ISP_SUBFLICKER_S stSubflicker;
    ISP_AE_DELAY_S stAEDelayAttr;
    CVI_BOOL bManualExpValue;
    CVI_U32 u32ExpValue;
    ISP_FSWDR_MODE_E enFSWDRMode;
    CVI_BOOL bWDRQuick;
    CVI_U16 u16ISOCalCoef;
    ISP_AE_GAIN_TYPE_E enGainType;
    ISP_AE_RANGE_S stISONumRange;
    CVI_U8 au8AdjustTargetMin[LV_TOTAL_NUM];
    CVI_U8 au8AdjustTargetMax[LV_TOTAL_NUM];
    CVI_BOOL bEnableFaceAE;
    CVI_U8 u8FaceTargetLuma;
    CVI_U8 u8FaceWeight;
    CVI_U8 u8HighLightLumaThr;
    CVI_U8 u8HighLightBufLumaThr;
    CVI_U8 u8LowLightLumaThr;
    CVI_U8 u8LowLightBufLumaThr;
```

(下页继续)

(续上页)

```

CVI_BOOL bHistogramAssist;
CVI_U16 u16AEStrategyThr;
CVI_U32 au32Reserve[RESERVE_SIZE];
} ISP_AE_ATTR_S;

```

【成员】

成员名称	描述
stExpTimeRange	曝光时间范围, 设置最大值和最小值, 以微秒 (us) 为单位。 取值范围: [0x0, 0xFFFFFFFF], 具体范围与: sensor 相关。
stISONumRange	ISO num 范围, 设置最大值和最小值, 此设定只在 enGainType = AE_TYPE_ISO 时有效。 取值范围: [0x64, 0xFFFFFFFF], 具体范围与: sensor 相关。
stAGainRange	Sensor 模拟增益范围, 设置最大值和最小值, 10bit 小数精度。 取值范围: [0x400, 0xFFFFFFFF], 具体范围与: sensor 相关。
stDGainRange	Sensor 数字增益范围, 设置最大值和最小值, 10bit 小数精度。 取值范围: [0x400, 0xFFFFFFFF], 具体范围与: sensor 相关。
stISPDGainRange	ISP 数字增益范围, 设置最大值和最小值, 10bit 小数精度。 取值范围: [0x400, 0x7FFF]。
stSysGainRange	系统增益范围, 设置最大值和最小值, 10bit 小数精度。 取值范围: [0x400, 0xFFFFFFFF], 具体范围与: sensor 相关。
u32GainThreshold	自动降帧时的系统增益门限值, 10bit 小数精度。 取值范围: [0x400, 0xFFFFFFFF], 默认值: 为 0x400000
u8Speed	自动曝光调整时的速度。 取值范围: [0x0, 0xFF], 默认值: 为 0x40。
u16BlackSpeedBias	画面由暗到亮 AE 调节速度的偏差值, 该值越大, 画面从暗到亮的速度越快。 取值范围: [0x0, 0xFFFF], 默认值: 为 0x90。
u8Tolerance	自动曝光调整时对画面亮度的容忍偏差。 取值范围: [0x0, 0xFF], 默认值: 为 0x2。
u8Compensation	自动曝光调整时的目标亮度。 取值范围: [0x0, 0xFF], 默认值: 为 0x38。
u16EVBias	自动曝光调整时的曝光量偏差值, 10bit 小数精度。 取值范围: [0x0, 0xFFFF], 默认值: 为 0x400。
enAEStrategyMode	自动曝光策略, 高光优先或低光优先, 默认值: 为 AE_EXP_HIGHLIGHT_PRIOR

下页继续

表 4.73 – 续上页

成员名称	描述
u16AESTrategyThr	高/低光优先模式时, 设置进入高低光优先状态的阈值, normal 模式下时, 此数值为满足条件的 AE grid 数目, bHistogramAssist enable 时此数值为满足条件的像素点数目 * 1/1000 * u16AESTrategyThr。 取值范围: [0x1, 0x3E8], 默认值: 为 0x2。
u16HistRatioSlope	高/低光优先时, 调整目标亮度的步幅。 取值范围: [0x0, 0xFFFF], 默认值: 为 0x8。
u8MaxHistOffset	感兴趣区域对统计平均值影响的最大程度。 取值范围: [0x0, 0xFF], 默认值: 为 0x10。
enAEMode	自动曝光模式, 自动降帧模式或固定帧率模式。 默认值: 为 AE_MODE_FIX_FRAME_RATE
stAntiflicker	抗闪属性设置。默认抗闪不使能。
stSubflicker	亚抗闪属性设置。默认亚抗闪不使能。
stAEDelayAttr	延时属性设置。默认 u16BlackDelayFrame=0u16WhiteDelayFrame=0。
bManualExpValue	手动曝光量使能, 该值为 CVI_TRUE 时, AE 算法采用 u32ExpValue 作为当前曝光量进行曝光时间和增益等的分配, 为 CVI_FALSE 时采用自动计算的曝光量进行分配。 默认值: 为 CVI_FALSE。
u32ExpValue	手动曝光量值, 等于曝光时间 (行数)* 系统增益 (6bit 小数精度)。 默认值: 为 0 取值范围: (0x0, 0xFFFFFFFF)。
enFSWDRMode	FSWDR 运行模式。默认为 ISP_FSWDR_NORMAL_MODE。
bWDRQuick	WDR 模式下, AE 算法从稳定状态 (亮度误差小于等于容忍偏差值 u8Tolerance) 重新调整时, 默认前 50 帧调整会进行时域滤波, 以调整更加平滑。该值为 CVI_TRUE 时, 取消 50 帧时域滤波, 使 AE 收敛速度更快。默认值为 CVI_FALSE。
u16ISOCalCoef	ISO 标定系数, 用于保证拍照所需 DCF 信息中显示的 ISO 是标准的, 8bit 精度。 取值范围: [0x0, 0xFFFF], 默认值: 为 0x100。
enGainType	选择以 ISO Num 或者是 Gain 的方式来控制增益
au8AdjustTargetMin	设定环境亮度各 LV AE 收敛的目标亮度的最小值 取值范围:: (0x0, 0x100) LV -5 ~ 15 的 默认值: 为 { 40, 40, 40, 40, 40, 40, 40, 40, 40, 40, 40, 45, 50, 50, 50, 50, 50, 60, 60, 60}
au8AdjustTargetMax	设定环境亮度各 LV AE 收敛的目标亮度的最大值 取值范围:: (0x0, 0x100) LV -5 ~ 15 的 默认值: 为 { 50, 50, 50, 50, 50, 50, 50, 50, 50, 50, 50, 60, 60, 60, 60, 60, 60, 70, 70, 70}
bEnableFaceAE	人脸识别连动 AE 测光 (Face AE) 使能
u8FaceTargetLuma	人脸测光的目标亮度 (8 bits)
u8FaceWeight	人脸测光与整体环境测光的权重比例
u8HighLightLumaThr	高亮区的亮度阈值

下页继续

表 4.73 – 续上页

成员名称	描述
u8HighLightBufLumaThr	高亮缓冲区的亮度阈值
u8LowLightLumaThr	低亮区的亮度阈值
u8LowLightBufLumaThr	低亮缓冲区的亮度阈值
bHistogramAssist	使用直方图统计值来辅助高光策略，应对高光区域小于一个 block size 的情况
au32Reserve[RESERVE_SIZE]	保留字段，无意义

【注意事项】

- 自动模式下，更改手动曝光属性的值不会生效。
- 自动曝光的最大最小时间及增益。

可根据不同的场景对曝光时间及增益进行限定，如有高速运动物体场景可限定最大曝光时间值为较小值，这样可减轻运动物体拖影现象。

- 自动曝光的系统增益

若 (sensor 模拟增益最小值 * sensor 数字增益最小值 * ISP 数字增益最小值) 小于系统增益最小值，则 AE 算法内部计算时最小增益会被限制到系统增益的最小值。若 (sensor 模拟增益最大值 * sensor 数字增益最大值 * ISP 数字增益最大值) 大于系统增益最大值，则 AE 算法内部计算时最大增益会被限制到系统增益的最大值。推荐通过设置系统增益的最大、最小值进行增益限制，分别限制 sensor 模拟增益、sensor 数字增益和 ISP 数字增益时，若把较高精度的 ISP 数字增益限制到 1 倍，容易导致闪烁。实际上，AE 算法内部利用系统增益来计算最大/最小曝光量，而并不是直接限制某项增益的值。比如说系统增益最大值限制为 1 倍，但 sensor 模拟增益的最小值限制为 2 倍，则实际生效的结果以 sensor 模拟增益的限制为准，sensor 数字增益及 ISP 数字增益的限制也有同理。

- 自动降帧时的系统增益门限值

在 SLOW_SHUTTER 模式下，当系统增益达到所设置的门限值时，系统将自动进入 SLOW_SHUTTER 模式。

- u8Speed 用于设定自动曝光时的收敛速度，该值越大曝光的收敛速度越快，但也会导致收敛过程中出现反复震荡。
- u16BlackSpeedBias 用于设定画面由暗到亮 AE 调节速度的偏差值。默认 u8Speed 下，画面从亮到暗的速度会快于画面从暗到亮的速度，若想双方向的 AE 调节速度差不多，可以调高 u16BlackSpeedBias。
- 亮度补偿属性 u8Compensation 用于调节曝光的目标亮度。曝光亮度补偿值越大则图像亮度越高。
- 曝光偏差属性 u16EVBias 用于在特殊场景下对画面的目标亮度进行微调，也可认为是更高精度的亮度补偿值，通过调节该值来改变画面目标亮度，真实生效的 AE 目标亮度为 $u8Compensation * u16EVBias / 1024$ 。u8Compensation 不变时，该值越大则图像亮度越高。
- 曝光容忍偏差属性 u8Tolerance 用于调节曝光对环境的灵敏度，曝光容忍偏差值越大则曝光越不敏感，且可能导致同一目标亮度值多次调节得到的亮度差异越大，所以该属性推荐不能设定过大。
- 曝光策略属性 enAESTrategyMode 用于选择对高光优先或低光优先的曝光策略。高光优先意味着对高光敏感，尽量避免画面过曝。低光优先意味着对低光敏感，尽量看清楚暗处区域，不管画面是否过曝。默认的曝光策略是高光优先，用户可根据场景需要进行调整，高

光优先时, 用户可设定 `u8HighLightLumaThr` 来决定高光区域的亮度阈值, 当画面中有超过此亮度阈值的 windows (该 windows 的权重不为 0) 时, 则会降低目标亮度来抑制此高光 windows 的亮度, 当所有的 windows 的亮度都小于 `u8HighLightBufLumaThr` 时, 则恢复到原本设定的目标亮度, 低光优先时, 当画面中有低于此 `u8LowLightLumaThr` 亮度阈值的 windows (该 windows 的权重不为 0) 时, 则会提高目标亮度来拉亮此低光 windows 的亮度, 当所有的 windows 的亮度都大于 `u8LowLightBufLumaThr` 时, 则恢复到原本设定的目标亮度。

- `u16HistRatioSlope` 用于设定感兴趣区域的权重。若是高光优先曝光模式, 则该值设置的是高光区域的权重, 该值越大, 则意味着对高光区域越敏感。反之, 若是低光优先模式, 则该值设置的是低光区域的权重, 该值越大, 则意味着对暗处区域越敏感, 建议 `u16HistRatioSlope` 的值设置不要超过 `0x100`。
- 自动曝光时, 可设定感兴趣区域对统计平均值影响的最大程度 `u8MaxHistOffset`。该值相当于对提高 `u16HistRatioSlope` 时增加的权重做限制, 若该值为 0, 无论 `u16HistRatioSlope` 多大, 也不会对高光或低光区域做特殊处理, 此时的统计平均值就是原始值。通过合理设置该值, 可以保证任何场景 AE 稳定后画面平均亮度都在一定范围内, 在高光优先曝光模式下, 如果该值设置较大, 在对比度稍高一些的场景, 如晴天室外场景, 有天空有树木, 则可能导致整体画面亮度偏低, 因为此时优先保证了亮区天空的效果, 通过限制该值从而限制对亮区的倾斜程度可以解决该问题。
- 曝光策略切换时, 最好同时更新 `u16HistRatioSlope` 和 `u8MaxHistOffset` 的值, 否则这 2 个值会采用上一种策略下的配置, 效果可能与预期不符。
- 在做强光抑制方案时, 建议在高光优先曝光模式下, 通过降低 AE 目标亮度, 同时合理设置 `u16HistRatioSlope` 和 `u8MaxHistOffset` 来抑制强光, 暗区则可以通过使能 DRC 来看清楚。低光优先则可以用于实现非指定区域的背光补偿。
- AE 曝光控制类型为自动时, 可设定曝光模式 `enAEMode`。该值可设定为慢快门模式 (SLOW_SHUTTER) 或固定帧率模式。慢快门模式通常用于低照度场景下进行自动降帧, 以减少画面噪声。
- 抗闪属性结构体 `stAntiflicker` 可用于设定抗闪使能, 抗闪频率和抗闪模式等属性。FSWDR 模式下, 只对长帧抗闪有效。由于随着曝光比的变化, 短帧曝光时间的最大值会受到限制, 建议 FSWDR 模式使用自动抗闪模式。
- 亚抗闪属性结构体 `stSubflicker` 可用于设定亚抗闪使能及抗闪程度属性。若有自动光圈, 建议关闭亚抗闪功能。
- AE 曝光控制类型为自动时, 可设定 AE 延时生效属性结构体 `stAEDelayAttr`。该值的合理设置可提高画面亮度的稳定性, 防止快速运动物体经过导致画面亮度发生变化。低码率设置时应该适当增加该值, 以避免 AE 调节时出现块效应。
- 用户可以通过将 `bManualExpValue` 置为 `CVI_TRUE`, 手动设置曝光量 `u32ExpValue` 来屏蔽 AE 算法的曝光量调整部分, 只用到曝光量分配部分。`u32ExpValue` 的值会受到最大、最小曝光量的限制。最大、最小曝光量分别对应最大、最小曝光时间与增益的乘积。
- FSWDR 运行模式改变时, 需要相应修改 `cmos.c` 中的以下回调函数:

`cmos_get_ae_default`, `cmos_fps_set`, `cmos_get_inttime_max`, 以保证长、短帧曝光时间在合理范围内。正常 WDR 和长帧模式切换时, 会回调 `cmos_get_ae_default`, 更新曝光时间最大/最小值, 增益最大/最小值, 光圈最大/最小值和 AE compensation 等参数, 同时, AE 内部会将 `u16HistRatioSlope` 和 `u8MaxHistOffset` 更新为算法默认值。需要注意的是, 此时曝光比相关设置不会更新, 仍会保持上一次的值。

- au8AdjustTargetMin/ au8AdjustTargetMax 可针对不同的环境亮度设定不同的目标亮度最小/大值, AE 会根据当前的环境亮度收敛目标亮度到该区间内

【相关数据类型及接口】

- CVI_ISP_SetExposureAttr

4.5.2.11 ISP_ME_ATTR_S**【说明】**

定义手动曝光属性。

【定义】

```
typedef struct _ISP_ME_ATTR_S
{
    ISP_OP_TYPE_E enExpTimeOpType;
    ISP_OP_TYPE_E enAGainOpType;
    ISP_OP_TYPE_E enDGainOpType;
    ISP_OP_TYPE_E enISPDGainOpType;
    CVI_U32 u32ExpTime;
    CVI_U32 u32AGain;
    CVI_U32 u32DGain;
    CVI_U32 u32ISPDGain;
    ISP_OP_TYPE_E enISONumOpType;
    ISP_AE_GAIN_TYPE_E enGainType;
    CVI_U32 u32ISONum;
} ISP_ME_ATTR_S;
```

【成员】

成员名称	描述
enExpTimeOpType	手动曝光时间使能, 默认值: 为 OP_TYPE_AUTO
enAGainOpType	手动 sensor 模拟增益使能, 默认值: 为 OP_TYPE_AUTO
enDGainOpType	手动 sensor 数字增益使能, 默认值: 为 OP_TYPE_AUTO
enISPDGainOpType	手动 ISP 数字增益使能, 默认值: 为 OP_TYPE_AUTO
u32ExpTime	手动曝光时间, 以微秒 (us) 为单位, 默认值: 为 0x4000。 取值范围: [0x0, 0xFFF FFFF], 具体范围与: sensor 相关。
u32AGain	手动 sensor 模拟增益, 10bit 小数精度, 默认值: 为 0x400。 取值范围: [0x400, 0xFFF FFFF], 具体范围与: sensor 相关。

下页继续

表 4.74 – 续上页

成员名称	描述
u32DGain	手动 sensor 数字增益, 10bit 小数精度, 默认值: 为 0x400。 取值范围: [0x400, 0xFFF FFFF], 具体范围与: sensor 相关。
u32ISPDGain	手动 ISP 数字增益, 10bit 小数精度, 默认值: 为 0x400。 取值范围: [0x400, 0x 40000], 具体范围与: sensor 相关。
enISONumOpType	选择以 ISO Num 或 gain 的方式控制增益, 默认值: 为 0
enGainType	增益使用的方式
u32ISONum	手动 ISO Num 增益, 默认值: 为 100, 只在 enGainType = AE_TYPE_ISO 有作用。

【注意事项】

- 手动模式下, 手动曝光时间和增益的值会受到自动模式下最大/最小曝光时间和增益的限制。
- 手动曝光使能参数有效时, 必须设置相应的手动曝光参数, 若不设置, 则采用系统默认值:。
- 增益单位为 10bit 小数精度的倍数, 即 1024 代表 1 倍。
- 若曝光参数设置超出最大(小)值, 将使用 sensor 支持的最大(小)值代替。

【相关数据类型及接口】

- CVI_ISP_SetExposureAttr

4.5.2.12 ISP_EXPOSURE_ATTR_S**【说明】**

定义 ISP 曝光属性。

【定义】

```
typedef struct _ISP_EXPOSURE_ATTR_S
{
    CVI_BOOL bByPass;
    ISP_OP_TYPE_E enOpType;
    CVI_U8 u8AERunInterval;
    CVI_BOOL bHistStatAdjust;
    CVI_BOOL bAERouteExValid;
    ISP_ME_ATTR_S stManual;
    ISP_AE_ATTR_S stAuto;
    CVI_U8 u8DebugMode;
    CVI_BOOL bAEGainSepCfg;
    CVI_S8 s8SensorSensitivity;
} ISP_EXPOSURE_ATTR_S;
```

【成员】

成员名称	描述
bByPass	AE 模块 byp ass 功能使能，默认为 CVI_FALSE。
enOpType	自动曝光或手动曝光开关，默认为 OP_TYPE_AUTO。
u8AERunInterval	AE 算法运行的间隔，取值范围为 [1,255]。 取值为 1 时表示每帧都运行 AE 算法；取值为 2 时表每 2 帧运行 1 次 AE 算法，依此类推。 建议该值设置不要大于 2， 否则 AE 调节速度会受到影响。 WDR 模式时，该值建议设置为 1，这样 AE 收敛会更加平滑。 该值默认为 1。
bHistStatAdjust	此参数会压低 AE 对于亮区的测光亮度，使亮区更亮，对于大面积天空场景，若测光偏暗，可设定此参数默认为 0
bAERouteExValid	AE 扩展分配路线是否生效开关 CVI_TRUE 时使用 AE 扩展分配路线，否则使用普通 AE 分配路线。 默认为 CVI_FALSE。
s8SensorSensitivity	AE 设置 sensor 感光度数值，用于校正因 sensor 感光度不一致导致 LV 值计算差异，默认值为 0。 取值范围：[-0xA, 0xA]
stManual	手动曝光属性结构体
stAuto	自动曝光属性结构体
u8DebugMode	设置 AE 的 debug mode，用于 dumplog 等 debug 功能，正常为 0
bAEGainSepCfg	长短帧增益是否分开分配

【注意事项】

- AE ByPass 为 CVI_TRUE 时，AE 模块被 bypass，任何 AE 配置都不会对图像亮度产生影响。ISP_AE_RESULT_S ‘_’ 保持为 AE bypass 前一帧的值。
- WDR/线性模式切换时，u8AERunInterval 会重置为 1，用户可以在切换完成后，根据需要修改该值。
- bAEGainSepCfg 仅在 2to1WDR 模式下配置且 sensor 支持此功能才能生效。
- 手动模式不支持长短帧增益分开分配

【相关数据类型及接口】

- CVI_ISP_SetExposureAttr
- CVI_ISP_GetExposureAttr

4.5.2.13 ISP_WDR_EXPOSURE_ATTR_S

【说明】

定义 WDR 模式下的曝光属性。

【定义】

```
#define EXP_RATIO_NUM      (3)
typedef struct _ISP_WDR_EXPOSURE_ATTR_S
{
    ISP_OP_TYPE_E enExpRatioType;
    CVI_U32 au32ExpRatio[EXP_RATIO_NUM];
    CVI_U32 u32ExpRatioMax;
    CVI_U32 u32ExpRatioMin;
    CVI_U16 u16Tolerance;
    CVI_U16 u16Speed;
    CVI_U16 u16RatioBias;
    CVI_U8 u8SECompensation;
    ISP_CHANNEL_LIST_E enExpMainChn;
    CVI_U8 au8LEAdjustTargetMin[LV_TOTAL_NUM];
    CVI_U8 au8LEAdjustTargetMax[LV_TOTAL_NUM];
    CVI_U8 au8SEAdjustTargetMin[LV_TOTAL_NUM];
    CVI_U8 au8SEAdjustTargetMax[LV_TOTAL_NUM];
} ISP_WDR_EXPOSURE_ATTR_S;
```

【成员】

成员名称	描述
enExpRatioType	<p>仅在多帧合成 WDR 模式下有效。</p> <p>OP_TYPE_AUTO: 根据场景自动计算长短帧曝光比;</p> <p>OP_TYPE_MANUAL: 手动配置长短帧曝光比。</p>
au32ExpRatio	<p>仅在多帧合成 WDR 模式下有效。</p> <p>当 enExpRatioType 为 OP_TYPE_AUTO 时, au32ExpRatio 无效。</p> <p>当 enExpRatioType 为 OP_TYPE_MANUAL 时, au32ExpRatio 表示二帧合成 WDR 相邻 2 帧曝光比期望值。</p> <p>其中 au32ExpRatio[0] 作为长帧/短帧曝光比,。6bit 小数精度, 0x40 表示曝光比为 1 倍。</p> <p>取值范围: [0x40, 0xFFF]</p>
u32ExpRatioMax	<p>仅在多帧合成 WDR 模式下有效。</p> <p>当 enExpRatioType 为 OP_TYPE_AUTO 时, u32ExpRatioMax 表示最长帧与最短帧曝光时间比值的最大值。</p> <p>即 2 帧合成时表示长帧/短帧曝光比的最大值。</p> <p>当 enExpRatioType 为 OP_TYPE_MANUAL 时, u32ExpRatioMax 无效。6bit 小数精度, 0x40 表示曝光比为 1 倍。</p> <p>取值范围: [0x40, 0x4000]</p>

下页继续

表 4.76 – 续上页

成员名称	描述
u32ExpRatioMin	<p>仅在多帧合成 WDR 模式下有效。</p> <p>当 enExpRatioType 为 OP_TYPE_AUTO 时, u32ExpRatioMin 表示长帧曝光时间与短帧曝光时间比值的最小值。</p> <p>当 enExpRatioType 为 OP_TYPE_MANUAL 时, u32ExpRatioMin 无效。</p> <p>格式为无符号 6.6bit 定点, 0x40 表示长帧曝光时间与短帧曝光时间的比值为 1 倍。</p> <p>默认值: 为 0x40。</p> <p>取值范围: [0x40,u32ExpRatioMax]</p>
u16Tolerance	<p>曝光调整时对画面亮度的容忍偏差, 仅在两帧合成 WDR 模式下有效。</p> <p>默认值: 为 5 取值范围: [0x0, 0xFF]</p>
u16Speed	<p>自动曝光比调节速度, 仅在两帧合成 WDR 模式下有效。</p> <p>当 enExpRatioType 为 OP_TYPE_AUTO 时, 该值越大, 自动曝光比调节速度越快。</p> <p>默认值: 为 0x20。</p> <p>取值范围: [0x0, 0xFF]</p>
u16RatioBias	<p>曝光比偏差值, 仅在多帧合成 WDR 模式下有效。</p> <p>当 enExpRatioType 为 OP_TYPE_AUTO 时, 该值越大, 自动曝光比越大。</p> <p>默认值: 为 0x400, 表示不对自动曝光比算法的计算结果进行调整。</p> <p>经过该值调整的曝光比会受到曝光比最大/最小值的限制。</p> <p>取值范围: [0x0, 0xFFFF]</p>
u8SECompensation	<p>短帧的目标亮度</p> <p>默认值: 为 56</p>
enExpMainChn	<p>AE 选择的主要通道</p> <p>默认值: 为 ISP_CHANNEL_LE</p>
au8LEAdjustTargetMin	<p>设定长帧环境亮度各 LV AE 收敛的目标亮度的最小值</p> <p>取值范围: (0x0, 0x100), LV -5 ~ 15 的</p> <p>默认值: 为 { 15, 15, 15, 15, 15, 15, 15, 20, 20, 25, 30, 35, 40, 40, 50, 50, 55, 60, 60, 60, 60 }</p>
au8LEAdjustTargetMax	<p>设定长帧环境亮度各 LV AE 收敛的目标亮度的最大值</p> <p>取值范围: (0x0, 0x100), LV -5 ~ 15 的</p> <p>默认值: 为 { 25, 25, 25, 25, 25, 25, 25, 30, 30, 35, 40, 45, 50, 50, 60, 60, 70, 75, 75, 75, 75 }</p>
au8SEAdjustTargetMin	<p>设定短帧环境亮度各 LV AE 收敛的目标亮度的最小值</p> <p>取值范围: (0x0, 0x100), LV -5 ~ 15 的</p> <p>默认值: 为 { 5, 5, 5, 5, 5, 5, 5, 10, 10, 15, 15, 15, 15, 20, 20, 20, 20, 20, 20, 20 }</p>
au8SEAdjustTargetMax	<p>设定长帧环境亮度各 LV AE 收敛的目标亮度的最大值</p> <p>取值范围: (0x0, 0x100), LV -5 ~ 15 的</p> <p>默认值: 为 { 15, 15, 15, 15, 15, 15, 15, 20, 20, 25, 25, 25, 25, 30, 40, 50, 60, 60, 60, 60, 60 }</p>

【注意事项】

- 针对某些对短帧最大曝光时间有限制的 sensor，曝光比较小时，长帧最大曝光时间较短，图像动态范围及噪声表现较差，导致自动曝光比计算不准确，此时建议限制最小曝光比，保证长帧最大曝光时间至少达到 3ms。
- 建议 u16Speed 不要设置小于 0x8，避免某些场景由于计算精度不足导致曝光比调节太慢甚至不调。u16Speed 过大可能导致曝光比变化太快，造成画面亮度出现振荡。
- 建议手动模式下 u32ExpRatio 不要设置大于 0x400。如果曝光比大于 0x400，在较亮的超宽动态场景，适当调高曝光比会优化长帧图像噪声表现。但是在较暗或低宽动态场景，如果曝光比过大，由于短帧最大曝光时间会被压缩，导致图像噪声表现会变差，而且会出现明显的噪声不连续，运动表现也会变差。

【相关数据类型及接口】

- CVI_ISP_SetWDRExposureAttr
- CVI_ISP_GetWDRExposureAttr

4.5.2.14 ISP_AE_ROUTE_NODE_S

【说明】

定义 AE 分配路线节点属性。

【定义】

```
typedef struct _ISP_AE_ROUTE_NODE_S
{
    CVI_U32 u32IntTime;
    CVI_U32 u32SysGain;
    ISP_IRIS_F_NO_E enIrisFNO;
    CVI_U32 u32IrisFNOLin;
} ISP_AE_ROUTE_NODE_S;
```

【成员】

成员名称	描述
u32IntTime	节点曝光时间，单位为微秒 (us)。 取值范围：(0x0, 0xFFFFFFFF]
u32SysGain	节点增益，包括 sensor 模拟增益，sensor 数字增益和 ISP 数字增益，10bit 精度。 取值范围：[0x400, 0xFFFFFFFF]
enIrisFNO	节点光圈 F 值大小，仅支持 P-Iris，不支持 DC-Iris。 取值范围：[ISP_IRIS_F_NO_32_0, ISP_IRIS_F_NO_1_0]。
u32IrisFNOLin	节点光圈 F 值等效增益大小，仅支持 P-Iris，不支持 DC-Iris。 取值范围：[1, 1024]

【注意事项】

无。

【相关数据类型及接口】

- CVI_ISP_SetAERouteAttr

4.5.2.15 ISP_AE_ROUTE_S

【说明】

定义 AE 曝光分配策略属性。

【定义】

```
#define ISP_AE_ROUTE_MAX_NODES (16)
typedef struct _ISP_AE_ROUTE_S
{
    CVI_U32 u32TotalNum;
    ISP_AE_ROUTE_NODE_S astRouteNode[ISP_AE_ROUTE_MAX_NODES];
} ISP_AE_ROUTE_S;
```

【成员】

成员名称	描述
u32TotalNum	曝光分配路线节点数目，目前最大为 16。
astRouteNode[ISP_AE_ROUTE_MAX_NODES]	曝光分配路线节点属性。

【注意事项】

无。

【相关数据类型及接口】

- CVI_ISP_SetAERouteAttr

4.5.2.16 ISP_AE_ROUTE_EX_NODE_S

【说明】

定义 AE 扩展分配路线节点属性。

【定义】

```
typedef struct _ISP_AE_ROUTE_EX_NODE_S
{
    CVI_U32 u32IntTime;
    CVI_U32 u32Again;
    CVI_U32 u32Dgain;
    CVI_U32 u32IspDgain;
    ISP_IRIS_F_NO_E enIrisFNO;
    CVI_U32 u32IrisFNOLin;
} ISP_AE_ROUTE_EX_NODE_S;
```

【成员】

成员名称	描述
u32IntTime	节点曝光时间，单位为微秒 (us)。 取值范围：(0x0, 0xFFFFFFFF]

下页继续

表 4.79 – 续上页

成员名称	描述
u32Again	sensor 模拟增益, 10bit 精度。 取值范围: [0x400, 0x3FFFFFF]
u32Dgain	sensor 数字增益, 10bit 精度。 取值范围: [0x400, 0x3FFFFFF]
u32IspDgain	ISP 数字增益, 10bit 精度。 取值范围: [0x400, 0x40000]
enIrisFNO	节点光圈 F 值大小, 仅支持 P-Iris, 不支持 DC-Iris。 取值范围: [ISP_IRIS_F_NO_32_0, ISP_IRIS_F_NO_1_0]
u32IrisFNOLin	节点光圈 F 值等效增益大小, 仅支持 P-Iris, 不支持 DC-Iris。 取值范围: [1, 1024]

【注意事项】

无。

【相关数据类型及接口】

· CVI_ISP_SetAERouteAttrEx

4.5.2.17 ISP_AE_ROUTE_EX_S

【说明】

定义 AE 曝光分配策略扩展属性。

【定义】

```
#define ISP_AE_ROUTE_EX_MAX_NODES (16)
typedef struct _ISP_AE_ROUTE_EX_S
{
    CVI_U32 u32TotalNum;
    ISP_AE_ROUTE_EX_NODE_S astRouteExNode[ISP_AE_ROUTE_EX_MAX_NODES];
} ISP_AE_ROUTE_EX_S;
```

【成员】

成员名称	描述
u32TotalNum	曝光扩展分配路线节点数目, 目前最大为 16。
astRouteExNode[ISP_AE_ROUTE_EX_MAX_NODES]	曝光扩展分配路线节点属性。

【注意事项】

无。

【相关数据类型及接口】

· CVI_ISP_SetAERouteAttrEx

4.5.2.18 ISP_EXP_INFO_S

【说明】

定义 ISP 曝光内部状态信息。

【定义】

```
#define HIST_NUM (256)
typedef struct _ISP_EXP_INFO_S
{
    CVI_U32 u32ExpTime;
    CVI_U32 u32ShortExpTime;
    CVI_U32 u32MedianExpTime;
    CVI_U32 u32LongExpTime;
    CVI_U32 u32AGain;
    CVI_U32 u32DGain;
    CVI_U32 u32ISPDGain;
    CVI_U32 u32Exposure;
    CVI_BOOL bExposureIsMAX;
    CVI_S16 s16HistError;
    CVI_U32 au32AE_Hist256Value[HIST_NUM];
    CVI_U8 u8AveLum;
    CVI_U32 u32LinesPer500ms;
    CVI_U32 u32PirisFNO;
    CVI_U32 u32Fps;
    CVI_U32 u32ISO;
    CVI_U32 u32ISOCalibrate;
    CVI_U32 u32RefExpRatio;
    CVI_U32 u32FirstStableTime;
    ISP_AE_ROUTE_S stAERoute;
    ISP_AE_ROUTE_EX_S stAERouteEx;
    CVI_U8 u8WDRShortAveLuma;
    CVI_U32 u32WDRExpRatio;
    CVI_U8 u8LEFrameAvgLuma;
    CVI_U8 u8SEFrameAvgLuma;
    CVI_FLOAT fLightValue;
    CVI_U32 u32AGainSF;
    CVI_U32 u32DGainSF;
    CVI_U32 u32ISPDGainSF;
    CVI_U32 u32ISOSF;
    ISP_AE_ROUTE_S stAERouteSF;
    ISP_AE_ROUTE_EX_S stAERouteSFEx;
    CVI_BOOL bGainSepStatus;
}ISP_EXP_INFO_S;
```

【成员】

成员名称	描述
u32ExpTime	当前曝光时间，单位为微秒 (us)。 取值范围：[0x0, 0xFFFFFFFF]
u32ShortExpTime	FSW DR 模式下，表示当前短帧 (S) 曝光时间，单位为微秒 (us)。线性模式不用关注该值。 取值范围：[0x0, 0xFFFFFFFF]

下页继续

表 4.81 – 续上页

成员名称	描述
u32LongExpTime	FSWDR 模式下，表示当前长帧曝光时间，单位为微秒 (us)。取值范围：[0x0, 0xFFFFFFFF]
u32AGain	当前 sensor 模拟增益，10bit 小数精度。取值范围：[0x400, 0xFFFFFFFF]
u32DGain	当前 sensor 数字增益，10bit 小数精度。取值范围：[0x400, 0xFFFFFFFF]
u32ISPDGain	当前 ISP 数字增益，10bit 小数精度。取值范围：[0x400, 0xFFFFFFFF]
u32Exposure	当前曝光量，等于曝光时间与曝光增益的乘积，其中曝光时间的单位为曝光行数，曝光增益为 6bit 小数精度。取值范围：[0x40, 0xFFFFFFFF]
bExposureIsMAX	0: ISP 未达到最大曝光水平；1: ISP 达到最大曝光水平。
s16HistError	统计信息，AE 的目标亮度值与实际值的差，该值为正表示当前期望的亮度信息大于实际的亮度信息，该值为负表示期望的亮度信息小于实际的亮度信息。
au32AE_Hist256Value	全局 256 段直方图统计信息取值范围：[0x0, 0xFFFFFFFF]
u8AveLum	平均亮度信息取值范围：[0x0, 0xFF]
u32LinesPer500ms	当前每 500ms 对应的曝光行数，可用于将曝光时间的单位由 us 转换成行数。取值范围：[0x0, 0xFFFFFFFF]
u32PirisFNO	当前 P-Iris 光圈 F 值对应的等效增益。取值范围：[0x0, 0x400]
u32Fps	实际图像帧率 * 100 取值范围：[0x0, 0xFFFFFFFF]
u32ISO	当前 sensor 模拟增益 * sensor 数字增益 * ISP 数字增益 * 100，其中增益的精度都为 10bit。 取值范围：[0x64, 0xFFFFFFFF]
u32ISOCalibrate	标准 ISO，用于拍照 DCF 信息显示。u32ISOCalibrate = u32ISO * 256 / u16ISOCalCoef。
u32RefExpRatio	参考曝光比，用于估计当前场景的动态范围，会受到 ISP_WDR_EXPOSURE_ATTR_S 中 Tolerance 和 Speed 等值的影响。 取值范围：[0x40, 0x4000]
u32FirstStableTime	首次 AE 收敛稳定的时间，单位为微秒 (us)。取值范围：[0x0, 0xFFFFFFFF]
stAERoute	实际生效的 AE route，各个节点中的曝光时间以 us 为单位，增益为 10bit 精度，光圈取值范围为 [ISP_IRIS_F_NO_32_0, ISP_IRIS_F_NO_1_0]。光圈类型为 DC-Iris 时，节点光圈值不会对曝光量分配产生影响。
stAERouteEx	实际生效的扩展 AE route，各个节点中的曝光时间以 us 为单位，增益为 10bit 精度，光圈取值范围为 [ISP_IRIS_F_NO_32_0, ISP_IRIS_F_NO_1_0]。光圈类型为 DC-Iris 时，节点光圈值不会对曝光量分配产生影响。
u32WDRExpRatio	WDR 长/短帧的曝光比，6 bits 精度
u8WDRShortAveLuma	短帧的画面亮度
u8LEFrameAvgLuma	长帧的画面平均亮度
u8SEFrameAvgLuma	短帧的画面平均亮度

下页继续

表 4.81 – 续上页

成员名称	描述
fLightValue	环境亮度值

【注意事项】

无。

【相关数据类型及接口】

- CVI_ISP_QueryExposureInfo

4.5.2.19 ISP_SMART_EXPOSURE_ATTR_S**【说明】**

定义智能模式下的 AE 曝光属性。

【定义】

```
typedef struct _ISP_SMART_EXPOSURE_ATTR_S
{
    CVI_BOOL bEnable;
    CVI_BOOL bIRMode;
    ISP_OP_TYPE_E enSmartExpType;
    CVI_U8 u8LumaTarget;
    CVI_U16 u16ExpCoef;
    CVI_U16 u16ExpCoefMax;
    CVI_U16 u16ExpCoefMin;
    CVI_U8 u8SmartInterval;
    CVI_U8 u8SmartSpeed;
    CVI_U16 u16SmartDelayNum;
    CVI_U8 u8Weight;
    CVI_U8 u8NarrowRatio;
} ISP_SMART_EXPOSURE_ATTR_S;
```

【成员】

成员名称	描述
bEnable	智能 AE 是否使能，默认 CVI_FALSE。
bIRMode	是否为红外模式，默认 CVI_FALSE。CVI_TRUE 为红外模式；。CVI_FALSE 为普通模式。
enSmartExpType	OP_TYPE_AUTO：根据目标检测结果自动调整曝光系数； OP_TYPE_MANUAL：手动配置曝光系数。
u8LumaTarget	识别区域 (如人脸) 目标亮度取值范围：[0x0, 0xFF]

下页继续

表 4.82 – 续上页

成员名称	描述
u16ExpCoef	<p>基于原始曝光调整的曝光系数，值越大，图像整体亮度越高；值越小，图像整体亮度越低。</p> <p>当 enSmartExpType 为 OP_TYPE_AUTO 时，u16ExpCoef 为 AE 算法根据目标检测结果计算的曝光系数。</p> <p>当 enExpRatioType 为 OP_TYPE_MANUAL 时，u16ExpCoef 为手动配置的曝光系数。</p> <p>10bit 小数精度，0x400 表示曝光系数为 1 倍。</p> <p>取值范围：[0x0, 0xFFFF]</p>
u16ExpCoefMax	<p>智能曝光系数最大值。</p> <p>当 enSmartExpType 为 OP_TYPE_AUTO 时，u16ExpCoefMax 表示智能曝光系数最大值。</p> <p>当 enSmartExpType 为 OP_TYPE_MANUAL 时，u16ExpCoefMax 无效。</p> <p>10bit 小数精度，0x400 表示曝光系数最大值为 1 倍。</p> <p>取值范围：[0x0, 0xFFFF]</p>
u16ExpCoefMin	<p>智能曝光系数最小值。</p> <p>当 enSmartExpType 为 OP_TYPE_AUTO 时，u16ExpCoefMin 表示。智能曝光系数最大值。</p> <p>当 enSmartExpType 为 OP_TYPE_MANUAL 时，u16ExpCoefMin 无效。</p> <p>10bit 小数精度，0x400 表示曝光系数最小值为 1 倍。</p> <p>取值范围：[0x0, 0xFFFF]</p>
u8SmartInterval	<p>智能 AE 运行间隔。</p> <p>取值为 1 时表示每帧运行，取值为 n 时，表示每 n 帧运行一次。</p> <p>取值范围：[0x1, 0xFF]</p>
u8SmartSpeed	<p>智能 AE 调整速度。</p> <p>默认值：为 0x40，值越小，调整速度越慢；值越大，调整速度越快，也越容易出现震荡。</p> <p>取值范围：[0x0, 0xFF]</p>
u16SmartDelayNum	<p>智能 AE 延时帧数，取值为 4 时表示 4 帧未设置识别区域信息，清空检测区域信息 ISP_SMART_INFO_S。</p>
u8Weight	<p>识别区域运算权重，仅智能模式生效</p> <p>取值范围：[0x0, 0x64]</p>
u8NarrowRatio	<p>识别区域有效面积缩小比例</p> <p>取值范围：[0x0, 0x64]</p>

【注意事项】

无。

【相关数据类型及接口】

- CVI_ISP_SetSmartExposureAttr
- CVI_ISP_GetSmartExposureAttr

4.5.3 IRIS

- `ISP_IRIS_STATUS_E`：定义 ISP 光圈状态。
- `ISP_IRIS_TYPE_E`：定义 ISP 光圈类型。
- `ISP_IRIS_F_NO_E`：定义 ISP 光圈 F 值。
- `ISP_MI_ATTR_S`：定义手动光圈属性。
- `ISP_DCIRIS_ATTR_S`：定义 DC-Iris AI 算法属性。
- `ISP_PIRIS_ATTR_S`：定义 P-Iris 属性。
- `ISP_IRIS_ATTR_S`：定义 ISP 光圈属性。

4.5.3.1 ISP_IRIS_STATUS_E

【说明】

定义 ISP 光圈状态。

【定义】

```
typedef enum _ISP_IRIS_STATUS_E
{
    ISP_IRIS_KEEP = 0,
    ISP_IRIS_OPEN = 1,
    ISP_IRIS_CLOSE = 2,
    ISP_IRIS_BUTT
} ISP_IRIS_STATUS_E;
```

【成员】

成员名称	描述
<code>ISP_IRIS_KEEP</code>	光圈保持当前状态。
<code>ISP_IRIS_OPEN</code>	光圈全开。
<code>ISP_IRIS_CLOSE</code>	光圈全关。

【注意事项】

该值设置为 `ISP_IRIS_OPEN` 或 `ISP_IRIS_CLOSE` 时，光圈处于全开或全关的状态，可用于测试 AI 电路和驱动是否正确。`OPEN` 和 `CLOSE` 的优先级高于 AI 使能和手动/自动模式，AI 算法运行时，为保证其能够正常工作，需将该值设置为 `ISP_IRIS_KEEP`。

【相关数据类型及接口】

无。

4.5.3.2 ISP_IRIS_TYPE_E

【说明】

定义 ISP 光圈类型。

【定义】

```
typedef enum _ISP_IRIS_TYPE_E
{
    ISP_IRIS_DC_TYPE = 0,
    ISP_IRIS_P_TYPE,
    ISP_IRIS_TYPE_BUTT,
} ISP_IRIS_TYPE_E;
```

【成员】

成员名称	描述
ISP_IRIS_DC_TYPE	DC-Iris 光圈
ISP_IRIS_P_TYPE	P-Iris 光圈

【注意事项】

- 必须设置正确的光圈类型，AI 算法才能正常工作。
- 若对接的是手动光圈镜头，可将该值设置为 ISP_IRIS_DC_TYPE，建议此时关闭 AI 使能。

【相关数据类型及接口】

无。

4.5.3.3 ISP_IRIS_F_NO_E

【说明】

定义 ISP 光圈 F 值。

【定义】

```
typedef enum _ISP_IRIS_F_NO_E
{
    ISP_IRIS_F_NO_32_0 = 0,ISP_IRIS_F_NO_22_0,ISP_IRIS_F_NO_16_0,ISP_IRIS_F_NO_
    ↪11_0,ISP_IRIS_F_NO_8_0,ISP_IRIS_F_NO_5_6,ISP_IRIS_F_NO_4_0,ISP_IRIS_F_NO_
    ↪2_8,ISP_IRIS_F_NO_2_0,ISP_IRIS_F_NO_1_4,ISP_IRIS_F_NO_1_0,ISP_IRIS_F_NO_
    ↪BUTT,
} ISP_IRIS_F_NO_E;
```

【成员】

成员名称	描述	等效增益
ISP_IRIS_F_NO_32_0	光圈 F32.0	1
ISP_IRIS_F_NO_22_0	光圈 F22.0	2
ISP_IRIS_F_NO_16_0	光圈 F16.0	4

下页继续

表 4.85 – 续上页

成员名称	描述	等效增益
ISP_IRIS_F_NO_11	光圈 F11.0	8
ISP_IRIS_F_NO_8	0光圈 F8.0	16
ISP_IRIS_F_NO_5	6光圈 F5.6	32
ISP_IRIS_F_NO_4	0光圈 F4.0	64
ISP_IRIS_F_NO_2	8光圈 F2.8	128
ISP_IRIS_F_NO_2	0光圈 F2.0	256
ISP_IRIS_F_NO_1	4光圈 F1.4	512
ISP_IRIS_F_NO_1	0光圈 F1.0	1024

【注意事项】

- 针对 P-Iris，AE 算法根据分配路线计算曝光量时，光圈 F 值要等效成一个增益，公式如下：
等效增益 $FNO = 1 \llcorner ISP_IRIS_F_NO_XX_XX$ 。由此可知 F32.0 对应增益 1，F22.0 对应增益 2，F16.0 对应增益 4，以此类推，F1.0 对应增益 1024。

【相关数据类型及接口】

无。

4.5.3.4 ISP_MI_ATTR_S**【说明】**

定义手动光圈属性。

【定义】

```
typedef struct _ISP_MI_ATTR_S
{
    CVI_U32 u32HoldValue;
    ISP_IRIS_F_NO_E enIrisFNO;
} ISP_MI_ATTR_S;
```

【成员】

成员名称	描述
u32HoldValue	AI 校正值，用于 DC-Iris 的调试。取值范围为 [0x0, 0x3E8]。
enIrisFNO	手动光圈大小，根据光圈 F 值进行区分，仅支持 P-Iris，不支持 DC-Iris。

【注意事项】

- 对接 DC-Iris 镜头时，若 ISP_IRIS_STATUS_E 设置为 ISP_IRIS_KEEP，手动光圈使能，u32HoldValue 可用于 DC-Iris 的调试，此时 PWM 的占空比即为 u32HoldValue。
- 对接 P-Iris 镜头时，若 ISP_IRIS_STATUS_E 设置为 ISP_IRIS_KEEP，手动光圈使能，enIrisFNO 可用于 P-Iris 的调试，此时会控制 P-Iris 步进电机走到光圈 F 值与 enIrisFNO 最接近的位置。自动曝光模式下，P-Iris 手动光圈不生效，此时若要固定光圈为某个 F 值，可以将 enMaxIrisFNOTarget/ enMinIrisFNOTarget 设置为相同值来实现。

【相关数据类型及接口】

无。

4.5.3.5 ISP_DCIRIS_ATTR_S

【说明】

定义 DC-Iris AI 算法属性。

【定义】

```
typedef struct _ISP_DCIRIS_ATTR_S
{
    CVI_S32 s32Kp;
    CVI_S32 s32Ki;
    CVI_S32 s32Kd;
    CVI_U32 u32MinPwmDuty;
    CVI_U32 u32MaxPwmDuty;
    CVI_U32 u32OpenPwmDuty;
} ISP_DCIRIS_ATTR_S;
```

【成员】

成员名称	描述
s32Kp	比例增益，用于调节光圈的开关速度，该值越大光圈打开和关闭的速度越快。 该值过小光线剧烈变化收敛过程中容易出现振荡，该值过大容易出现超调，也会导致振荡。 该值的合理设置与电路特性和镜头相关。建议值为 7000。 取值范围为 [0, 100000]。
s32Ki	积分增益，用于调节光圈的开关速度，该值越大光圈打开和关闭的速度越快。 该值较小时，碰到强光收敛后画面会稳定在一个比较低的亮度上；该值较大时可能会导致强光场景光圈无法关闭。 该值的合理设置与电路特性和镜头相关。建议值为 100，该值一般不需要修改。 取值范围为 [0, 1000]。
s32Kd	微分增益，用于限制光线剧烈变化时光圈的开关速度，该值越大光线剧烈变化时光圈打开和关闭的速度越慢。 该值过大对于瞬间变化的亮度过于敏感，会导致场景亮度快速变化时画面出现振荡。 该值的合理设置与电路特性和镜头相关。建议值为 3000。 取值范围为 [0, 100000]。
u32MinPwmDuty	最小 PWM 占空比。该值越小过曝时光圈关闭速度越快，但容易导致光圈来回震荡。 该值的合理设置与电路特性和镜头相关。建议值为 250。 取值范围为 [0, 1000]。

下页继续

表 4.87 – 续上页

成员名称	描述
u32MaxPwmDuty	最大 PWM 占空比。该值越大画面全黑时光圈打开速度越快，该值过小则可能导致退出光圈控制区域时光圈仍未达到最大，造成画面噪声严重。 该值的合理设置与电路特性和镜头相关。建议值为 950。 取值范围为 [0, 1000]。
u32OpenPwmDuty	光圈打开时的 PWM 占空比。 当画面亮度稳定并且 PWM 占空比大于该值一段时间后，退出光圈控制区域。 所以该值不能太小，否则容易导致光圈未达到最大就退出了光圈控制区域，造成画面噪声严重。 该值的合理设置与电路特性和镜头相关。建议值为 800。 取值范围为 [0, 1000]。

【注意事项】

- 当光圈关闭出现震荡时，一般意味着光圈关闭速度太快了，可以通过适当减小 s32Kp 和增大 u32MinPwmDuty 来解决。
- u32OpenPwmDuty 的取值要求在 u32MinPwmDuty 和 u32MaxPwmDuty 之间，必须确保该值能将光圈较快的打开。

【相关数据类型及接口】

- CVI_ISP_SetDcirisAttr
- CVI_ISP_GetDcirisAttr

4.5.3.6 ISP_PIRIS_ATTR_S**【说明】**

定义 P-Iris 属性。

【定义】

无。

【成员】

成员名称	描述
无	无

【注意事项】

目前尚未实作支持 P- Iris

【相关数据类型及接口】

无。

4.5.3.7 ISP_IRIS_ATTR_S

【说明】

定义 ISP 光圈属性。

【定义】

```
typedef struct _ISP_IRIS_ATTR_S {  
    CVI_BOOL bEnable;  
    ISP_OP_TYPE_E enOpType;  
    ISP_IRIS_TYPE_E enIrisType;  
    ISP_IRIS_STATUS_E enIrisStatus;  
    ISP_MI_ATTR_S stMIAttr;  
} ISP_IRIS_ATTR_S;
```

【成员】

成员名称	描述
bEnable	自动光圈使能。
enOpType	自动光圈或手动光圈模式选择。
enIrisType	光圈类型，DC-Iris 或 P-Iris。
enIrisStatus	光圈状态。
stMIAttr	手动光圈属性设置结构体。

【注意事项】

该值设置为 ISP_IRIS_OPEN 或 ISP_IRIS_CLOSE 时，光圈处于全开或全关的状态，可用于测试 AI 电路和驱动是否正确。OPEN 和 CLOSE 的优先级高于 AI 使能和手动/自动模式，AI 算法运行时，为保证其能够正常工作，需将该值设置为 ISP_IRIS_KEEP。

【相关数据类型及接口】

- CVI_ISP_SetIrisAttr
- CVI_ISP_GetIrisAttr

5 AWB

5.1 概述

色温随可见光的光谱成分变化而变化，在低色温光源下，白色物体偏红，在高色温光源下，白色物体偏蓝。人眼可根据大脑的记忆判断，识别物体的真实颜色，AWB 算法功能是降低外界光源对物体真实颜色的影响，使得我们采集的颜色信息转变为在理想日光光源下的无偏色信息

5.2 重要概念

- 色温：色温是按绝对黑体来定义的，光源的辐射在可见区和绝对黑体的辐射完全相同时，此时黑体的温度就称此光源的色温。
- 白平衡：在不同色温的光源下，白色在传感器中的响应会偏蓝或偏红。白平衡算法通过调整 R, G, B 三个颜色通道的强度，使白色真实呈现。

5.3 功能描述

AWB 模块有硬件的 WB 信息统计模块及 AWB 策略控制算法 firmware 两部分组成。

ISP 的 WB 信息统计模块判断 sensor 输出的每个像素是否满足用户设定的白点条件，计算所有满足条件的像素的 R、G、B 三个颜色通道平均值。

其支持将图像分成 M*N (M 行 N 列) 区域，统计每个区域的 R、G、B 均值以及参与统计的白点个数，也支持输出整幅图像的 R、G、B 均值以及参与统计的白点个数。

AWB 工作原理如 图 5.1 所示

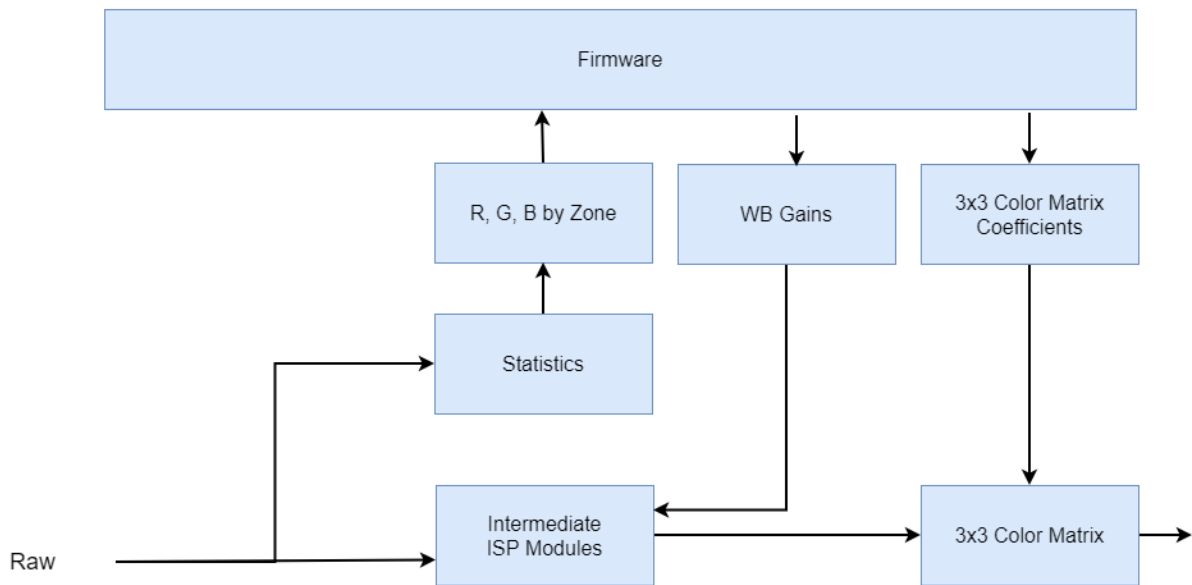


图 5.1: AWB 工作原理图

5.4 API 参考

5.4.1 AWB 库接口

所有 AWB 库接口都只是针对 CVI AWB 库，如果客户自己实现 AWB 库，不需要管这些接口，同时也无法使用这些接口。

- `CVI_AWB_Register`：向 ISP 注册 AWB 库。
- `CVI_AWB_UnRegister`：向 ISP 注销 AWB 库。
- `CVI_AWB_SensorRegCallBack`：AWB 库提供的 sensor 注册的回调接口
- `CVI_AWB_SensorUnRegCallBack`：AWB 库提供的 sensor 注销的回调接口

5.4.1.1 CVI_AWB_Register

【描述】

向 ISP 注册 AWB 库。

【语法】

```
CVI_S32 CVI_AWB_Register(VI_PIPE ViPipe, ALG_LIB_S *pstAwbLib);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstAwbLib	AWB 算法库结构体指针	输入

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见错误码。

【需求】

- 头文件: cvi_awb.h
- 库文件: libawb.a

【注意】

- 该接口调用了 ISP 库提供的 AWB 注册回调接口 CVI_ISP_AWBLibRegCallBack, 以实现 AWB 向 ISP 库注册的功能。
- 此接口不支持多进程操作。

【举例】

无。

【相关主题】

无。

5.4.1.2 CVI_AWB_UnRegister**【描述】**

向 ISP 反注册 AWB 库。

【语法】

```
CVI_S32 CVI_AWB_UnRegister(VI_PIPE ViPipe, ALG_LIB_S *pstAwbLib);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstAwbLib	AWB 算法库结构体指针	输入

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见错误码。

【需求】

- 头文件: cvi_awb.h
- 库文件: libawb.a

【注意】

- 该接口调用了 ISP 库提供的 AWB 反注册回调接口 CVI_ISP_AWBLibUnRegCallBack , 以实现 AWB 向 ISP 库反注册的功能。
- 此接口不支持多进程操作。
- 此接口在双系统 SDK 的 linux 侧暂不支持。

【举例】

无。

【相关主题】

无。

5.4.1.3 CVI_AWB_SensorRegCallBack

【描述】

AWB 库提供的 sensor 注册的回调接口。

【语法】

```
CVI_S32 CVI_AWB_SensorRegCallBack(VI_PIPE ViPipe, ALG_LIB_S *pstAwbLib, ISP_SNS_
→ATTR_INFO_S *pstSnsAttrInfo, AWB_SENSOR_REGISTER_S *pstRegister);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstAeLib	AWB 算法库结构体指针	输入
pstSnsAttrInfo	向 AWB 注册的 Sensor 的属性	输入
pstRegister	Sensor 注册结构体指针	输入

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见错误码。

【需求】

- 头文件: cvi_awb.h
- 库文件: libawb.a

【注意】

- SensorId 是 sensor 库中自定义的值, 主要用于校对向 ISP 注册的 sensor 和向 3A 注册的

- sensor 是否为同一个 sensor。
- AWB 通过 sensor 注册的一系列回调接口，获取差异化的初始化参数，并控制 sensor。
 - 此接口不支持多进程操作。

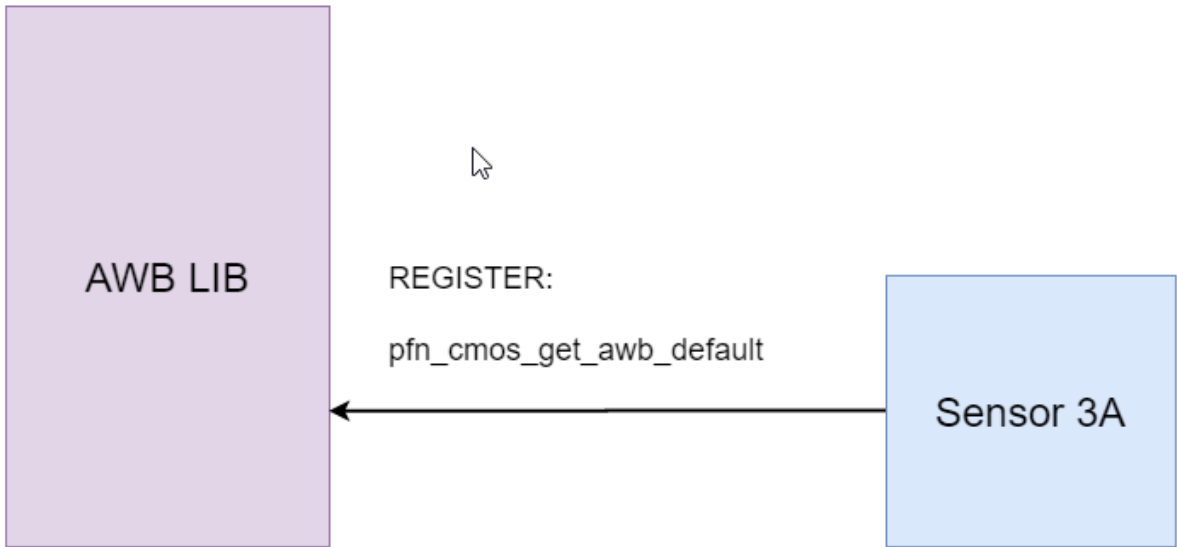


图 5.2: AWB 库与 sensor 库间的接口

【举例】

无。

【相关主题】

无。

5.4.1.4 CVI_AWB_SensorUnRegCallBack

【描述】

AWB 库提供的 sensor 反注册的回调接口。

【语法】

```
CVI_S32 CVI_AWB_SensorUnRegCallBack(VI_PIPE ViPipe, ALG_LIB_S *pstAwbLib, SENSOR_
↪ID SensorId);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstAwbLib	AWB 算法库结构体指针	输入
SensorId	向 AWB 反注册的 Sensor 的 Id	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_awb.h
- 库文件：libawb.a

【注意】

- SensorId 是 sensor 库中自定义的值，主要用于校对向 ISP 注册的 sensor 和向 3A 注册的 sensor 是否为同一个 sensor。
- 此接口不支持多进程操作。

【举例】

无。

【相关主题】

无。

5.4.2 AWB 控制模块

- CVI_ISP_SetWBAttr：设置白平衡属性。
- CVI_ISP_GetWBAttr：获取白平衡属性。
- CVI_ISP_SetAWBAttrEx：设置白平衡扩展属性。
- CVI_ISP_GetAWBAttrEx：获取白平衡扩展属性。
- CVI_ISP_QueryWBInfo：获取当前白平衡增益系数，检测色温，饱和度值，颜色校正矩阵系数。
- CVI_ISP_SetAWBLogPath：使用 CVI Awb lib 时，存放 AWB 调试日志的路径。
- CVI_ISP_SetAWBLogName：使用 CVI Awb lib 时，存放 AWB 调试日志的名称。
- CVI_ISP_GetGrayWorldAwbInfo：获取灰世界 WB 信息。

5.4.2.1 CVI_ISP_SetWBAttr

【描述】

设置白平衡属性。

【语法】

```
CVI_S32 CVI_ISP_SetWBAttr(VI_PIPE ViPipe, const ISP_WB_ATTR_S *pstWBAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstWBAttr	白平衡属性结构体指针	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_awb.h
- 库文件：libawb.a

【注意】

- 白平衡控制类型为自动时，AWB 算法自动调节白平衡系数。
- 白平衡控制类型为手动时，AWB 算法失效，需自行设定 Rgain、Ggain、Bgain。
- 环境色温、照度会影响白点的分布范围，CVI AWB 算法在运行过程中，会根据环境参数自动刷新 AWB Byaer 域统计信息的白点参数。用户希望在 CVI AWB 算法运行时修改 AWB 统计参数，需要调用 CVI_ISP_SetWBAttr 接口，关闭统计参数自动刷新功能。用户关闭统计参数自动刷新功能，到 ISP 收到配置并响应，有 2 帧的延时。因此，用户设置后需要等待 2 帧时间再修改 AWB 统计参数。

【举例】

无。

【相关主题】

- [CVI_ISP_GetWBAttr](#)

5.4.2.2 CVI_ISP_GetWBAttr**【描述】**

获取白平衡属性。

【语法】

```
CVI_S32 CVI_ISP_GetWBAttr(VI_PIPE ViPipe, ISP_WB_ATTR_S *pstWBAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstWBAttr	白平衡属性结构体指针	输出

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_awb.h
- 库文件：libawb.a

【注意】

无。

【举例】

无。

【相关主题】

- CVI_ISP_SetWBAttr

5.4.2.3 CVI_ISP_SetAWBAttrEx**【描述】**

设置白平衡扩展属性。

【语法】

```
CVI_S32 CVI_ISP_SetAWBAttrEx(VI_PIPE ViPipe, const ISP_AWB_ATTR_EX_S_
↪ *pstAWBAttrEx);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
PstAWBAttrEx	扩展白平衡属性结构体指针	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_awb.h
- 库文件：libawb.a

【注意】

- 当 CVI_ISP_SetWBAttr 接口成员 pstWBAttr->enAlgType 为 AWB_ALG_ADVANCE 时，此接口才有效。

【举例】

无。

【相关主题】

- CVI_ISP_GetAWBAttrEx

5.4.2.4 CVI_ISP_GetAWBAttrEx**【描述】**

获取白平衡扩展属性。

【语法】

```
CVI_S32 CVI_ISP_GetAWBAttrEx(VI_PIPE ViPipe, ISP_AWB_ATTR_EX_S *pstAWBAttrEx);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstAWBAttrEx	扩返回展白平衡属性结构体指针	输出

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_awb.h
- 库文件：libawb.a

【注意】

无。

【举例】

无。

【相关主题】

- CVI_ISP_SetAWBAttrEx

5.4.2.5 CVI_ISP_QueryWBInfo

【描述】

获取当前白平衡增益系数、检测色温、饱和度值、颜色校正矩阵系数。

【语法】

```
CVI_S32 CVI_ISP_QueryWBInfo(VI_PIPE ViPipe, ISP_WB_INFO_S *pstWBInfo);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
PstWBInfo	颜色相关状态参数	输出

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_awb.h
- 库文件：libawb.a

【注意】

无。

【举例】

无。

【相关主题】

无。

5.4.2.6 CVI_ISP_SetAWBLogPath

【描述】

使用 CVI Awb lib 时，存放 AWB 调试日志的路径。

【语法】

```
CVI_S32 CVI_ISP_SetAWBLogPath(const char *szPath)
```

【参数】

参数名称	描述	输入/输出
szPath	调试日志路径	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_awb.h
- 库文件：libawb.a

【注意】

- 默认路径为 /var/log

【举例】

无。

【相关主题】

无。

5.4.2.7 CVI_ISP_SetAWBLogName

【描述】

使用 CVI Awb lib 时, 存放 AWB 调试日志的档名。

【语法】

```
CVI_S32 CVI_ISP_SetAWBLogName(const char *szName)
```

【参数】

参数名称	描述	输入/输出
szName	文件名	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_awb.h
- 库文件：libawb.a

【注意】

- 预设文件名为 AwbLog0.txt

【举例】

无。

【相关主题】

无。

5.4.2.8 CVI_ISP_GetGrayWorldAwbInfo

【描述】

获取灰度世界 WB 信息。

【语法】

```
CVI_S32 CVI_ISP_GetGrayWorldAwbInfo(VI_PIPE ViPipe, CVI_U16 *pRgain, CVI_U16 *pBgain);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pRgain	R 通道增益	输出
pBgain	B 通道增益	输出

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_awb.h
- 库文件：libawb.a

【注意】

无。

【举例】

无。

【相关主题】

无。

5.5 数据类型

5.5.1 Register

- AWB_SENSOR_REGISTER_S：定义 sensor 注册结构体。
- AWB_SENSOR_EXP_FUNC_S：定义 sensor 回调函数结构体。
- AWB_SENSOR_DEFAULT_S：定义 AWB 算法库的初始化参数结构体。
- AWB_SPEC_SENSOR_DEFAULT_S：定义 SPEC AWB 算法的初始化参数结构体。
- AWB_AGC_TABLE_S：定义饱和度初始化参数结构体。
- AWB_CCM_TAB_S：定义不同色温下自动颜色校正矩阵系数。
- AWB_CCM_S：定义 CCM 颜色校正矩阵属性。

5.5.2 WB

- ISP_AWB_ATTR_S：定义 ISP 自动白平衡属性。
- ISP_AWB_CBCR_TRACK_ATTR_S：定义 Bayer 域统计信息的联动参数。
- ISP_AWB_LUM_HISTGRAM_ATTR_S：定义白平衡的亮度直方图统计参数。
- ISP_AWB_ALG_E：定义白平衡的计算方式属性。
- ISP_AWB_CT_LIMIT_ATTR_S：定义白平衡的增益范围限制属性。
- ISP_MWB_ATTR_S：定义 ISP 手动白平衡属性。
- ISP_WB_ATTR_S：定义白平衡属性。
- ISP_AWB_ALG_TYPE_E：定义 AWB 算法属性。
- ISP_AWB_ATTR_EX_S：定义自动白平衡扩展属性。
- ISP_AWB_EXTRA_LIGHTSOURCE_INFO_S：定义独立光源点的信息。
- ISP_AWB_IN_OUT_ATTR_S：对场景作室内外的判定。
- ISP_AWB_MULTI_LS_TYPE_E：定义混合光源下的 AWB 策略类型。
- ISP_AWB_INDOOR_OUTDOOR_STATUS_E：定义 AWB 室内室外状态。
- ISP_WB_INFO_S：定义白平衡、饱和度、颜色校正信息。

5.5.2.1 ISP_MWB_ATTR_S

【说明】

定义手动 WB

【定义】

```
typedef struct _ISP_MWB_ATTR_S {  
    CVI_U16 u16Rgain;  
    CVI_U16 u16Grgain;  
    CVI_U16 u16Gbgain;  
    CVI_U16 u16Bgain;  
} ISP_MWB_ATTR_S;
```

【成员】

成员名称	描述
u16Rgain	MWB 时的 R 通道增益
u16Grgain	MWB 时的 Gr 通道增益
u16Gbgain	MWB 时的 Gb 通道增益
u16Bgain	MWB 时的 B 通道增益

【注意事项】

- RGB 通道增益一倍为 0x400

【相关数据类型及接口】

无。

5.5.2.2 ISP_AWB_CT_LIMIT_ATTR_S

【说明】

定义白平衡的增益范围限制属性

【定义】

```
typedef struct _ISP_AWB_CT_LIMIT_ATTR_S {  
    CVI_BOOL bEnable;  
    ISP_OP_TYPE_E enOpType;  
    CVI_U16 u16HighRgLimit;  
    CVI_U16 u16HighBgLimit;  
    CVI_U16 u16LowRgLimit;  
    CVI_U16 u16LowBgLimit;  
} ISP_AWB_CT_LIMIT_ATTR_S;
```

【成员】

成员名称	描述
bEnable	白平衡的增益范围限制开关
enOpType	自动或手动设定自平衡的增益范围

下页继续

表 5.26 – 续上页

成员名称	描述
u16HighRgLimit	手动模式下高色温下的最大 R 增益
u16HighBgLimit	手动模式下高色温下的最小 B 增益
u16LowRgLimit	手动模式下低色温下的最小 R 增益
u16LowBgLimit	手动模式下低色温下的最大 B 增益

【注意事项】

- 自动模式下 AWB 算法会自己计算上下限增益值, 当切换为手动模式时, 会使用上述四个 R.B 增益的限制值
- 一倍增益为 0x400

【相关数据类型及接口】

无。

5.5.2.3 ISP_AWB_ALG_E

【说明】

定义 AWB 算法类别

【定义】

```
typedef enum _ISP_AWB_ALG_E {
    ALG_AWB,
    ALG_AWB_SPEC,
    ALG_BUTT
} ISP_AWB_ALG_E;
```

【成员】

成员名称	描述
AWB	一般 AWB 算法返回
ALG_AWB_SPEC	基于 machine learning 的 AWB 算法, 目前不支持。

【注意事项】

无。

【相关数据类型及接口】

无。

5.5.2.4 ISP_AWB_ATTR_S

【说明】

定义自动 WB 属性

【定义】

```
typedef struct _ISP_AWB_ATTR_S {
    CVI_BOOL bEnable;
    CVI_U16 u16RefColorTemp;
    CVI_U16 au16StaticWB[ISP_BAYER_CHN_NUM];
    CVI_S32 as32CurvePara[AWB_CURVE_PARA_NUM];
    ISP_AWB_ALG_TYPE_E enAlgType;
    CVI_U8 u8RGStrength;
    CVI_U8 u8BGStrength;
    CVI_U16 u16Speed;
    CVI_U16 u16ZoneSel;
    CVI_U16 u16HighColorTemp;
    CVI_U16 u16LowColorTemp;
    ISP_AWB_CT_LIMIT_ATTR_S stCTLimit;
    CVI_BOOL bShiftLimitEn;
    CVI_U16 u16ShiftLimit[AWB_CURVE_BOUND_NUM];
    CVI_BOOL bGainNormEn;
    CVI_BOOL bNaturalCastEn;
    ISP_AWB_CBCR_TRACK_ATTR_S stCbCrTrack;
    ISP_AWB_LUM_HISTGRAM_ATTR_S stLumaHist;
    CVI_BOOL bAWBZoneWtEn;
    CVI_U8 au8ZoneWt[AWB_ZONE_ORIG_ROW * AWB_ZONE_ORIG_COLUMN];
} ISP_AWB_ATTR_S;
```

【成员】

成员名称	描述
bEnable	自动白平衡算法开关
u16RefColorTemp	静态白平衡的光源色温值, 建议为 5000K 灯源
au16StaticWB	静态白平衡系数, 由 AWB 标定工具给出。返回取值范围: [0, 0x3FFF]。
as32CurvePara	CurvePara[0-2] 普朗克曲线系数, 由 AWB 标定工具给出。普朗克曲线描绘白色块返回在不同色温的标准光源下的颜色表现。CurvePara[3-5] 色温曲线系数, 由 AWB 标定工具给出。色温曲线描绘白色块的颜色表现与色温的对应关系。
enAlgType	AWB 算法类别选择, AWB_ALG_LOW COST, AWB_ALG_ADVANCE 两种 ISP_AWB_ATTR_EX_S 扩展属性只有在 AWB_ALG_ADVANCE 才有作用
u8RGStrength	自动白平衡 R 通道校准强度。 取值范围: [0, 255]
u8BGStrength	自动白平衡 B 通道校准强度。 取值范围: [0, 255]
u16Speed	自动白平衡算法收敛速度。 取值范围: [0, 4095]

下页继续

表 5.28 – 续上页

成员名称	描述
u16ZoneSel	参数为 0 或 255 时, 采用近似灰世界的白平衡算法, 其他值则为进行分类筛选, 提升精度
u16HighColorTemp	自动白平衡算法的色温上限。 取值范围: [8000, 10000]
u16LowColorTemp	自动白平衡算法的色温下限。 取值范围: [0x0, u8HighColorTemp)
stCTLimit	手动或自动限制白平衡增益值
bShiftLimitEn	AWB 超过类白点范围的增益映像回白点范围的开关
u16ShiftLimit	AWB 计算白点范围参数
bGainNormEn	对 RGB 通道增益进行返回限制, 可以改善低色温、低照度场景的信噪比, 预设开启
bNaturalCastEn	低色温下 AWB 风格喜好开关。预设关闭
stCbCrTrack	AWB 统计范围与 ISO 的连动参数
stLumaHist	AWB 亮度与权重参数
bAWBZoneWtEn	画面分区权重开关, 预设关闭
au8ZoneWt	32x32 画面权重, 取值范围 [0,0xFF] (尚未实作) 返回

【注意事项】

无。

【相关数据类型及接口】

无。

5.5.2.5 ISP_AWB_ALG_TYPE_E

【说明】

定义 AWB 算法属性

【定义】

```
typedef enum ISP_AWB_ALG_TYPE_E {
    AWB_ALG_LOWCOST,
    AWB_ALG_ADVANCE,
    AWB_ALG_BUTT
} ISP_AWB_ALG_TYPE_E;
```

【成员】

成员名称	描述
AWB_ALG_LOWCOST	简易低运算量的 AWB 算法
AWB_ALG_ADVANCE	进阶扩展 AWB 算法, 与 ISP_AWB_ATTR_EX_S 相关
AWB_ALG_BUTT	无效设定值返回

【注意事项】

无。

【相关数据类型及接口】

ISP_AWB_ATTR_EX_S

5.5.2.6 ISP_AWB_CBCR_TRACK_ATTR_S

【说明】

定义 AWB 统计范围与 ISO 的连动参数

【定义】

```
typedef struct _ISP_AWB_CBCR_TRACK_ATTR_S {  
    CVI_BOOL bEnable;  
    CVI_U16 au16CrMax[ISP_AUTO_ISO_STRENGTH_NUM];  
    CVI_U16 au16CrMin[ISP_AUTO_ISO_STRENGTH_NUM];  
    CVI_U16 au16CbMax[ISP_AUTO_ISO_STRENGTH_NUM];  
    CVI_U16 au16CbMin[ISP_AUTO_ISO_STRENGTH_NUM];  
} ISP_AWB_CBCR_TRACK_ATTR_S;
```

【成员】

成员名称	描述
bEnable	AWB 统计范围与 ISO 的连动开关
au16CrMax	不同 ISO 下 R/G 的最大值
au16CrMin	不同 ISO 下 R/G 的最小值
au16CbMax	不同 ISO 下 B/G 的最大值
au16CbMin	不同 ISO 下 B/G 的最小值

【注意事项】

- 建议在低色温下标定 CrMax(R/G),CbMin(B/G)

【相关数据类型及接口】

无。

5.5.2.7 ISP_AWB_LUM_HISTGRAM_ATTR_S

【说明】

定义 AWB 亮度与权重参数

【定义】

```
typedef struct _ISP_AWB_LUM_HISTGRAM_ATTR_S {  
    CVI_BOOL bEnable;  
    ISP_OP_TYPE_E enOpType;  
    CVI_U8 au8HistThresh[AWB_LUM_HIST_NUM];  
    CVI_U16 au16HistWt[AWB_LUM_HIST_NUM];  
} ISP_AWB_LUM_HISTGRAM_ATTR_S;
```

宏定义如下

```
#define AWB_LUM_HIST_NUM (6)
```

【成员】

成员名称	描述
bEnable	不同亮度是否开启权重, 预设开启
enOpType	自动模式:AWB 自动分配权重 手动模式: 用户可自行设定亮度分类与权重
au8HistThresh	亮度分类的阈值 (手动模式下有效)
au16HistWt	亮度分类的权重 (手动模式下有效)

【注意事项】

- au8HistThresh[0] 固定为 0, au8HistThresh[5] 固定为 255.
- au8HistThresh[i+1] 必须大于 au8HistThresh[i];
- au16HistWt 权重范围为 32~512

【相关数据类型及接口】

无。

5.5.2.8 ISP_WB_ATTR_S

【说明】

定义白平衡属性

【定义】

```
typedef struct _ISP_WB_ATTR_S {
    CVI_BOOL bByPass;
    CVI_U8 u8AWBRunInterval;
    ISP_OP_TYPE_E enOpType;
    ISP_MWB_ATTR_S stManual;
    ISP_AWB_ATTR_S stAuto;
    ISP_AWB_ALG_E enAlgType;
    CVI_U8 u8DebugMode;
} ISP_WB_ATTR_S;
```

【成员】

成员名称	描述
bByPass	白平衡模块 Bypass 使能, 默认值 CVI_FALSE。返回
u8AWBRunInterval	白平衡模块工作频率。 取值范围: [0x1, 0xFF]
enOpType	自动白平衡和手动白平衡切换。
stManual	手动参数
stAuto	自动参数
enAlgType	AWB 算法类别返回
u8DebugMode	Debug 时使用, 一般不需设定

【注意事项】

- bByPass 为 TRUE 时, WB 其他参数设定不生效, RGB 通道增益固定为一倍 0x400
- u8AWBRunInterval 默认值为 6, 表示每 6 帧执行一次 AWB, 可根据需求增加此数值, 让 AWB 运行频率降低

【相关数据类型及接口】

无。

5.5.2.9 ISP_AWB_ATTR_EX_S

【说明】

定义自动白平衡扩展属性

【定义】

```
typedef struct _ISP_AWB_ATTR_EX_S {
    CVI_U8 u8Tolerance;
    CVI_U8 u8ZoneRadius;
    CVI_U16 u16CurveLLimit;
    CVI_U16 u16CurveRLimit;
    CVI_BOOL bExtraLightEn;
    ISP_AWB_EXTRA_LIGHTSOURCE_INFO_S stLightInfo[AWB_LS_NUM];
    ISP_AWB_IN_OUT_ATTR_S stInOrOut;
    CVI_BOOL bMultiLightSourceEn;
    ISP_AWB_MULTI_LS_TYPE_E enMultiLSType;
    CVI_U16 u16MultiLSScaler;
    CVI_U16 au16MultiCTBin[AWB_CT_BIN_NUM];
    CVI_U16 au16MultiCTWt[AWB_CT_BIN_NUM];
    CVI_BOOL bFineTunEn;
    CVI_U8 u8FineTunStrength;
    //AWB Algo 6
    struct ST_ISP_AWB_INTERFERENCE_S stInterference;
    struct ST_ISP_AWB_SKIN_S stSkin;
    struct ST_ISP_AWB_SKY_S stSky;
    struct ST_ISP_AWB_GRASS_S stGrass;
    struct ST_ISP_AWB_CT_WGT_S stCtLv;
    struct ST_ISP_AWB_SHIFT_LV_S stShiftLv;
    struct ST_ISP_AWB_REGION_S stRegion;
    CVI_U8 adjBgainMode;
} ISP_AWB_ATTR_EX_S;

// 宏定义如下:
#define AWB_CT_BIN_NUM    (8)
#define AWB_LS_NUM    (4)
```

【成员】

成员名称	描述
u8Tolerance	AWB 调整的偏差返回范围, 误差在此范围内时, AWB 不做调整
u8ZoneRadius	AWB 统计值分区的大小。该值越小, AWB 精度越高, 但会降低 AWB 算法稳定性

下页继续

表 5.33 – 续上页

成员名称	描述
u16CurveLLimit	AWB 色温曲线返回的左边界限 (R/G,B/G), 取值范围: [0x0, 0x200]
u16CurveRLimit	AWB 色温曲线的右边界限 (R/G,B/G), 范围: [0x200, 0x3FF]
bExtraLightEn	是否开启独立光源
stLightInfo	AWB 计算时是否考虑色温曲线外的独立光源点, 最多四个独立点
stInOrOut	AWB 对场景做室内外判断的参数
bMultiLightSourceEn	AWB 检测当前场返回景是否为混合光源, 来调整饱和度或 CCM
enMultiLSType	调整饱和度或是 CCM
u16MultiLSScaler	当混合光源下调整饱和度或 CCM 的强度
au16MultiCTBin	色温分段参数, 须为递增序列
au16MultiCTWt	色温分段权重, 范围: [0x0, 0x400]
bFineTunEn	AWB 特殊色检测开关, 例如肤色
u8FineTunStrength	肤色、蓝色等特殊色检测的强度

【注意事项】

- ISP_AWB_ATTR_EX_S 扩展属性只有在 pstWBAttr->enAlgType 为 AWB_ALG_ADVANCE 才有作用
- u8Tolerance 为 AWB 的灵敏度参数, 值为 0 时,AWB 参数会实时更新, 值比较大时, 环境色温微微变化时并不会更新 AWB 参数, 画面会轻微的偏色, 建议户外为 0, 室内设置为 2
- u16CurveLLimit 取值 <512, 主要为排除绿色物体, u16CurveRLimit 取值 >512, 为排除偏紫色区域
- bMultiLightSourceEn 开启后,AWB 会判断场景是否为混光源, 在这样的场景下会降低饱和度或是 CCM, 以减少偏色
- WDR 模式下会自动关闭混光源侦测功能

【相关数据类型及接口】

无。

5.5.2.10 ISP_AWB_EXTRA_LIGHTSOURCE_INFO_S**【说明】**

定义独立光源点的参数

【定义】

```
typedef struct _ISP_AWB_EXTRA_LIGHTSOURCE_INFO_S {
    CVI_U16 u16WhiteRgain;
    CVI_U16 u16WhiteBgain;
    CVI_U16 u16ExpQuant;
    CVI_U8 u8LightStatus;
    CVI_U8 u8Radius;
} ISP_AWB_EXTRA_LIGHTSOURCE_INFO_S;
```


【成员】

成员名称	描述
u16WhiteRgain	特殊光源点的 R 通道增益
u16WhiteBgain	特殊光源点的 B 通道增益
u16ExpQuant	根据外在亮度做判断。 ExpQuant 为开启的亮度限制值例如 ExpQuant = 6, 表示 LV6 以下返回开启此 WB 光源点 (一般夜景为 LV6 以下) ExpQuant = 106 表示 LV6 以上开启 ExpQuant = 112 表示 LV12 以上开启 (LV12 一般为户外)
u8LightStatus	特殊光源点的种类, 0: 不作动 1: 加入光源点返回 2: 删除光源点附近的计算
u8Radius	特殊光源点的区域大小, 取值范围: [0x0, 0xFF]

【注意事项】

无。

【相关数据类型及接口】

无。

5.5.2.11 ISP_AWB_IN_OUT_ATTR_S

【说明】

定义 AWB 对场景做户外室内判断的参数

【定义】

```
typedef struct _ISP_AWB_IN_OUT_ATTR_S {
    CVI_BOOL bEnable;
    ISP_OP_TYPE_E enOpType;
    ISP_AWB_INDOOR_OUTDOOR_STATUS_E enOutdoorStatus;
    CVI_U32 u32OutThresh;
    CVI_U16 u16LowStart;
    CVI_U16 u16LowStop;
    CVI_U16 u16HighStart;
    CVI_U16 u16HighStop;
    CVI_BOOL bGreenEnhanceEn;
    CVI_U u8OutShiftLimit;
} ISP_AWB_IN_OUT_ATTR_S;
```

【成员】

成员名称	描述
bEnable	判断场景为室内室外的开关
enOpType	判断室内室外 (自动或手动)

下页继续

表 5.35 – 续上页

成员名称	描述
enOutdoorStatus	室内或室外模式 (手动模式下)
u32OutThresh	判定室内室外的阈值, 亮度小于时, 则判定为室内, 户外 LV 大多超过 15
u16LowStart	将低色温的权重返回重拉低, 低色温区的起始点, 建议为 5000K
u16LowStop	将低色温的权重返回拉低, 低色温区的终止点, 建议为 4500K, 取值范围: (0,u16LowStart)
u16HighStart	将高色温的权重拉低, 高色温区的起始点, 建议为 6500K, 取值范围: (u16LowStart, 0xFFFF]
u16HighStop	将高色温的权重返回拉低, 高色温区的终止点, 建议为 8000K, 取值范围: (u16HighStart, 0xFFFF]
bGreenEnhanceEn	在绿色植物场景下, 对绿色通道增加的开关
u8OutShiftLimit	当判定为户外场景时,AWB 算法的白点范围限制

【注意事项】

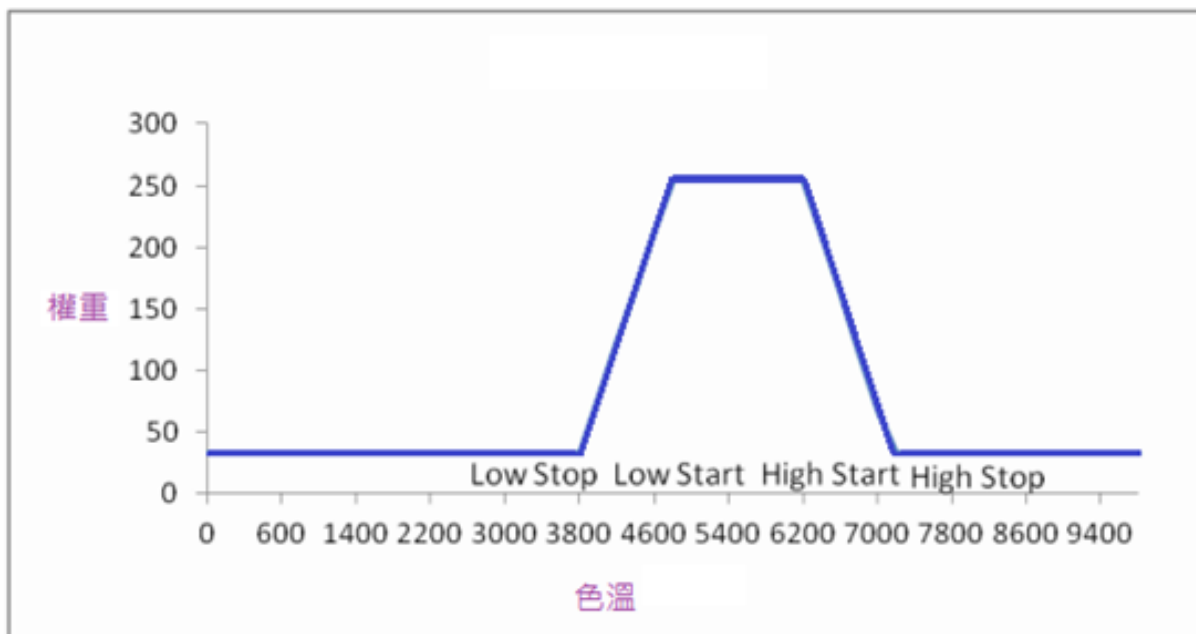
- 户外亮度大多 LV 大于 15 所以 u32OutThresh 建议为大于 15
- 四个参数的取值范围要求: $u16LowStop < u16LowStart < u16HighStart < u16HighStop$

【相关数据类型及接口】

如下图, u16LowStop 为 3800K, u16LowStart 为 5000K

u16HighStart 为 6200K, u16HighStop 为 7200K

一般权重为 32, 室外色温最高权重为 256



5.5.2.12 ISP_AWB_MULTI_LS_TYPE_E

【说明】

定义混光源下 AWB 的策略

【定义】

```
typedef enum _ISP_AWB_MULTI_LS_TYPE_E {  
    AWB_MULTI_LS_SAT,  
    AWB_MULTI_LS_CCM,  
    AWB_MULTI_LS_BUTT  
} ISP_AWB_MULTI_LS_TYPE_E;
```

【成员】

成员名称	描述
AWB_MULTI_LS_SAT	混合光源下调整饱和度
AWB_MULTI_LS_CCM	混合光源下调整 CCM
AWB_MULTI_LS_BUTT	无效

【注意事项】

无。

【相关数据类型及接口】

无。

5.5.2.13 ISP_AWB_INDOOR_OUTDOOR_STATUS_E

【说明】

定义 AWB 室内室外状态

【定义】

```
typedef enum _ISP_AWB_INDOOR_OUTDOOR_STATUS_E {  
    AWB_INDOOR_MODE,  
    AWB_OUTDOOR_MODE,  
    AWB_INDOOR_OUTDOOR_BUTT  
} ISP_AWB_INDOOR_OUTDOOR_STATUS_E;
```

【成员】

成员名称	描述
AWB_INDOOR_MODE	室内模式
AWB_OUTDOOR_MODE	室外模式
AWB_INDOOR_OUTDOOR_BUTT	无效

【注意事项】

无。

【相关数据类型及接口】

无。

5.5.2.14 ISP_WB_INFO_S

【说明】

定义白平衡，饱和度，颜色校正信息

【定义】

```
typedef struct _ISP_WB_INFO_S {
    CVI_U16 u16Rgain;
    CVI_U16 u16Grgain;
    CVI_U16 u16Gbgain;
    CVI_U16 u16Bgain;
    CVI_U16 u16Saturation;
    CVI_U16 u16ColorTemp;
    CVI_U16 au16CCM[CCM_MATRIX_SIZE];
    CVI_U16 u16LS0CT;
    CVI_U16 u16LS1CT;
    CVI_U16 u16LS0Area;
    CVI_U16 u16LS1Area;
    CVI_U8 u8MultiDegree;
    CVI_U16 u16ActiveShift;
    CVI_U32 u32FirstStableTime;
    ISP_AWB_INDOOR_OUTDOOR_STATUS_E enInOutStatus;
    CVI_S16 s16Bv;
} ISP_WB_INFO_S;
```

【成员】

成员名称	描述
u16Rgain	当前 R 通道增益值
u16Grgain	当前 Gr 通道增益值
u16Gbgain	当前 Gb 通道增益值
u16Bgain	当前 B 通道增益值
u16Saturation	目前饱和度返回
u16ColorTemp	目前推算的色温值
au16CCM	当前的颜色校正矩阵，10bit 小数精度。bit 15 是符号位，0 表示正数，1 表示负数，例如 0x8010 表示-16
u16LS0CT	混光源场景的主光源色温
u16LS1CT	混光源场景的次光源色温
u16LS0Area	混光源场景的主光源面积
u16LS1Area	混光源场景的次光源面积
u8MultiDegree	目前场景是混光源的机率
u16ActiveShift	目前白平衡范围的限制值
u32FirstStableTime	首次 AWB 收敛稳定的时间，以帧数为单位
ISP_AWB_INDOOR_OUTDOOR_STATUS_E enInOutStatus	室内室外检测结果
s16Bv	当前环境的 bv 值返回

【注意事项】

无。

【相关数据类型及接口】

无。

6 AF

6.1 概述

CVI AF 模块实现的功能是：根据硬件统计信息模块输出的图像清晰度值 FV (Focus Value)，通过自动对焦算法找到最佳对焦位置，并控制对焦马达移动到该位置，以获得最清晰的图像。

AF 算法采用三段式对焦搜索策略：首先确定对焦方向，然后以较大步长进行粗搜索找到 FV 峰值附近区域，最后以小步长进行精确定位。同时支持追焦（变焦后自动重新对焦）、实时追焦（检测画面变化自动重新对焦）、齿隙补偿等功能。

AF 模块的工作需要马达驱动的支持，客户需实现马达控制回调函数并注册到 AF 模块中。

6.2 重要概念

- 清晰度值 (FV, Focus Value)：通过分析图像的高频分量来表征图像清晰度。图像越清晰，FV 值越大。
- 对焦搜索三阶段：AF 算法将对焦过程分为三个阶段——初始步长搜索 (InitStep)、粗搜索 (FindStep)、精定位 (LocateStep)。
- 追焦 (Chasing Focus)：当镜头变焦导致对焦位置偏移时，根据 Zoom-Focus 查找表自动调整对焦位置以恢复清晰图像。
- 实时追焦 (Real-Time Focus)：通过持续监测 FV 值和亮度变化，在检测到画面变化时自动触发重新对焦。
- 齿隙补偿 (Backlash)：马达正反转切换时存在机械间隙，AF 算法通过反向预走一定步长来消除该间隙的影响。
- 变焦对焦追踪表 (Zoom-Focus Table)：记录不同变焦位置对应的最佳对焦位置范围，共 9 个采样点，用于变焦时快速估算对焦位置。
- VCM (Voice Coil Motor)：音圈马达，通过电流驱动镜头移动，无机械齿隙。
- 步进马达 (Stepper Motor)：通过步进驱动的马达，存在机械齿隙需补偿。

6.3 功能描述

AF 模块主要由 ISP 硬件统计信息模块和 AF 控制算法两部分组成。

ISP 硬件统计信息模块负责对输入图像进行滤波处理并统计每个区块的高频分量值，提供 H0、H1、V0 三种滤波器的统计结果以及高亮点计数（HlCnt）。

AF 控制算法采用有限状态机实现，主要流程包括：马达归位初始化、触发对焦、方向检测、粗搜索、精定位、对焦完成。此外还支持追焦（Chasing Focus）、实时对焦（Real-Time Focus）以及马达方向切换时的齿隙补偿。

6.4 API 参考

6.4.1 AF 库接口

所有 AF 库接口都只是针对 CVI AF 库，如果客户自己实现 AF 库，不需要关注这些接口，且无法使用这些接口。

- `CVI_AF_Register`：向 ISP 注册 AF 库。
- `CVI_AF_UnRegister`：向 ISP 反注册 AF 库。
- `CVI_AF_MOTOR_Register`：注册马达控制回调函数。
- `CVI_AF_MOTOR_UnRegister`：反注册马达控制回调函数。

6.4.1.1 CVI_AF_Register

【描述】

向 ISP 注册 AF 库。

【语法】

```
CVI_S32 CVI_AF_Register(VI_PIPE ViPipe, ALG_LIB_S *pstAfLib);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstAfLib	AF 算法库结构体指针	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件: `cvi_af.h`

【注意】

无。

【相关主题】

- [CVI_AF_UnRegister](#)

6.4.1.2 CVI_AF_UnRegister

【描述】

向 ISP 反注册 AF 库。

【语法】

```
CVI_S32 CVI_AF_UnRegister(VI_PIPE ViPipe, ALG_LIB_S *pstAfLib);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstAfLib	AF 算法库结构体指针	输入

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见错误码。

【需求】

- 头文件: `cvi_af.h`

【注意】

无。

【相关主题】

- [CVI_AF_Register](#)

6.4.1.3 CVI_AF_MOTOR_Register

【描述】

注册马达控制回调函数。客户需要实现马达控制相关的回调函数, 包括马达初始化、正转、反转、速度设置、镜头信息获取等。

【语法】

```
CVI_S32 CVI_AF_MOTOR_Register(VI_PIPE ViPipe, ISP_AF_MOTOR_FUNC_S_
↪ *pstAfMotorCb);
```


【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstAfMotorCb	马达控制回调函数结构体指针	输入

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见错误码。

【需求】

- 头文件: cvi_af.h

【注意】

- 该函数必须在调用 AF 算法之前注册。
- 客户需实现 ISP_AF_MOTOR_FUNC_S 中的所有回调函数, 包括 pfn_af_motor_init、pfn_af_motor_deinit、pfn_af_set_focus_in、pfn_af_set_focus_out、pfn_af_set_focus_speed、pfn_af_set_zoom_in、pfn_af_set_zoom_out、pfn_af_set_zoom_speed、pfn_af_set_zoom_focus、pfn_af_get_len_info。
- pfn_af_get_len_info 回调需正确返回镜头的硬件特性信息, 包括对焦范围、变焦范围、齿隙值、最大速度等。

【相关主题】

- [CVI_AF_MOTOR_UnRegister](#)
- [ISP_AF_MOTOR_FUNC_S](#)

6.4.1.4 CVI_AF_MOTOR_UnRegister

【描述】

反注册马达控制回调函数。

【语法】

```
CVI_S32 CVI_AF_MOTOR_UnRegister(VI_PIPE ViPipe, ISP_AF_MOTOR_FUNC_S_
↪ *pstAfMotorCb);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstAfMotorCb	马达控制回调函数结构体指针	输入

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见错误码。

【需求】

- 头文件: `cvi_af.h`

【注意】

无。

【相关主题】

- `CVI_AF_MOTOR_Register`

6.4.2 AF 控制模块

- `CVI_ISP_AFAutoFocus` : 触发自动对焦。
- `CVI_ISP_AFGetFv` : 获取当前帧的 FV 值。
- `CVI_ISP_SetAFAttr` / `CVI_ISP_GetAFAttr` : 设置/获取 AF 属性。
- `CVI_ISP_SetAFVcmAttr` / `CVI_ISP_GetAFVcmAttr` : 设置/获取 VCM 镜头属性。
- `CVI_ISP_SetAFStatisticsConfig` / `CVI_ISP_GetAFStatisticsConfig` : 设置/获取 AF 统计信息配置。
- `CVI_ISP_SetAFZoneFocusAttr` / `CVI_ISP_GetAFZoneFocusAttr` : 设置/获取 ROI 对焦区域属性。
- `CVI_ISP_AFQueryFocusInfo` : 查询对焦状态信息。
- `CVI_ISP_AFSetFocus` / `CVI_ISP_AFSetFocusSpeed` : 手动对焦控制。
- `CVI_ISP_AFSetZoom` / `CVI_ISP_AFSetZoomSpeed` : 手动变焦控制。
- `CVI_AF_GetMotorCB` : 获取马达控制回调函数指针。

6.4.2.1 CVI_ISP_AFAutoFocus

【描述】

触发自动对焦。调用此函数后 AF 算法开始执行三段式对焦搜索（方向检测 → 粗搜索 → 精定位）。

【语法】

```
CVI_S32 CVI_ISP_AFAutoFocus(VI_PIPE ViPipe);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见错误码。

【需求】

- 头文件: `cvi_af.h`

【注意】

- 调用前需确保 AF 已正确初始化并注册了马达回调。
- 对焦过程中 AE 会被暂时挂起（锁定曝光），对焦完成后自动恢复。

【相关主题】

- [CVI_ISP_AFGetFv](#)
- [CVI_ISP_AFQueryFocusInfo](#)

6.4.2.2 CVI_ISP_AFGetFv

【描述】

获取当前帧的 AF 清晰度值（Focus Value）。FV 值越大表示图像越清晰。

【语法】

```
CVI_S32 CVI_ISP_AFGetFv(VI_PIPE ViPipe, CVI_U32 *pFv);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pFv	指向 FV 值的指针	输出

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见错误码。

【需求】

- 头文件: `cvi_af.h`

【注意】

- FV 值需要在垂直消隐期间获取，建议配合 `CVI_ISP_GetVDTimeOut` 使用以获取同步的

统计值。

【相关主题】

- [CVI_ISP_AFAutoFocus](#)

6.4.2.3 CVI_ISP_SetAFAttr

【描述】

设置 AF 算法控制参数。包括使能开关、工作模式、搜索步长、追焦配置、比率阈值等。

【语法】

```
CVI_S32 CVI_ISP_SetAFAttr(VI_PIPE ViPipe, const ISP_FOCUS_ATTR_S *pstFocusAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstFocusAttr	AF 属性结构体指针	输入

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见错误码。

【需求】

- 头文件: `cvi_af.h`

【注意】

- 修改 AF 属性时若 AF 正在运行, 新参数将在当前对焦周期结束后生效。
- `u8RunInterval` 取值范围为 `[0x1, 0x10]`。
- 三个步长参数 (`u8InitStep`/`u8FindStep`/`u8LocateStep`) 取值范围均为 `[0x1, 0xFF]`。
- 比率参数 (`DiffRatio` 系列) 基于 `AF_RATIO_BASE` (1024) 计算。

【相关主题】

- [CVI_ISP_GetAFAttr](#)
- [ISP_FOCUS_ATTR_S](#)

6.4.2.4 CVI_ISP_GetAFAttr

【描述】

获取当前 AF 算法控制参数。

【语法】

```
CVI_S32 CVI_ISP_GetAFAttr(VI_PIPE ViPipe, ISP_FOCUS_ATTR_S *pstFocusAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstFocusAttr	AF 属性结构体指针	输出

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见错误码。

【需求】

- 头文件: cvi_af.h

【注意】

无。

【相关主题】

- CVI_ISP_SetAFAttr
- ISP_FOCUS_ATTR_S

6.4.2.5 CVI_ISP_SetAFVcmAttr

【描述】

设置 VCM 镜头标定属性, 包括无限远位置、最近对焦位置和中点位置。

【语法】

```
CVI_S32 CVI_ISP_SetAFVcmAttr(VI_PIPE ViPipe, const ISP_AF_VCM_ATTR_S  
→ *pstVCMAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstVCMAttr	VCM 属性结构体指针	输入

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见错误码。

【需求】

- 头文件: `cvi_af.h`

【注意】

- 这些值需要通过实际标定获得。建议使用 `AF_ENTER_CALIB_FLOW` 标定工具中的 `AF_RANGE_OFFSET_CALIB` 模式获取。
- `u32InfinitePos` 表示无限远（远处物体清晰对焦）时的镜头位置。
- `u32MacroPos` 表示最近对焦（微距）时的镜头位置。
- `u32MediumPos` 表示中间位置的镜头位置。

【相关主题】

- [CVI_ISP_GetAFVcmAttr](#)
- [ISP_AF_VCM_ATTR_S](#)

6.4.2.6 CVI_ISP_GetAFVcmAttr

【描述】

获取 VCM 镜头标定属性。

【语法】

```
CVI_S32 CVI_ISP_GetAFVcmAttr(VI_PIPE ViPipe, ISP_AF_VCM_ATTR_S *pstVCMAAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstVCMAAttr	VCM 属性结构体指针	输出

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见错误码。

【需求】

- 头文件: `cvi_af.h`

【注意】

无。

【相关主题】

- [CVI_ISP_SetAFVcmAttr](#)

6.4.2.7 CVI_ISP_SetAFStatisticsConfig**【描述】**

设置 AF 硬件统计信息配置，包括滤波器系数、窗口大小、裁剪区域、权重表等。

【语法】

```
CVI_S32 CVI_ISP_SetAFStatisticsConfig(VI_PIPE ViPipe, const ISP_FOCUS_STATISTICS_
→CFG_S *pstAfStatCfg);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstAfStatCfg	AF 统计信息配置结构体指针	输入

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见错误码。

【需求】

- 头文件: `cvi_af.h`

【注意】

- 统计区域大小建议设置为 17（水平）x 15（垂直）。
- 裁剪区域的 X 坐标范围为 [0x8, 图像宽度-8], Y 坐标范围为 [0x2, 图像高度-2]。
- 权重表大小为 15 x 17, 每个权重值取值范围为 [0x0, 0xF]。
- FIR 滤波器系数需要根据镜头光学特性进行调整。参考配置可参见 3A 开发用户指南中的 FV 计算参考代码。

【相关主题】

- [CVI_ISP_GetAFStatisticsConfig](#)
- [ISP_FOCUS_STATISTICS_CFG_S](#)

6.4.2.8 CVI_ISP_GetAFStatisticsConfig

【描述】

获取 AF 硬件统计信息配置。

【语法】

```
CVI_S32 CVI_ISP_GetAFStatisticsConfig(VI_PIPE ViPipe, ISP_FOCUS_STATISTICS_CFG_S_
↪ *pstAfStatCfg);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstAfStatCfg	AF 统计信息配置结构体指针	输出

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见错误码。

【需求】

- 头文件: cvi_af.h

【注意】

无。

【相关主题】

- CVI_ISP_SetAFStatisticsConfig

6.4.2.9 CVI_ISP_SetAFZoneFocusAttr

【描述】

设置 ROI 对焦区域属性。可指定对焦区域的坐标、大小和权重, 用于局部区域对焦。

【语法】

```
CVI_S32 CVI_ISP_SetAFZoneFocusAttr(VI_PIPE ViPipe, const ISP_AF_ZONE_FOCUS_ATTR_
↪ S *pstAfZoneFocusAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstAfZoneFocusAttr	ROI 对焦区域属性结构体指针	输入

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见错误码。

【需求】

- 头文件: `cvi_af.h`

【注意】

- `enable` 为 `CVI_FALSE` 时使用全局权重表进行对焦。
- `u8ZoneWeight` 取值范围为 `[0, 255]`。

【相关主题】

- [CVI_ISP_GetAFZoneFocusAttr](#)
- [ISP_AF_ZONE_FOCUS_ATTR_S](#)

6.4.2.10 CVI_ISP_GetAFZoneFocusAttr

【描述】

获取 ROI 对焦区域属性。

【语法】

```
CVI_S32 CVI_ISP_GetAFZoneFocusAttr(VI_PIPE ViPipe, ISP_AF_ZONE_FOCUS_ATTR_S_
↪ *pstAfZoneFocusAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstAfZoneFocusAttr	ROI 对焦区域属性结构体指针	输出

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见错误码。

【需求】

- 头文件: `cvi_af.h`

【注意】

无。

【相关主题】

- [CVI_ISP_SetAFZoneFocusAttr](#)

6.4.2.11 CVI_ISP_AFQueryFocusInfo

【描述】

查询 AF 算法运行状态信息，包括当前对焦状态、FV 值、马达位置和速度等。

【语法】

```
CVI_S32 CVI_ISP_AFQueryFocusInfo(VI_PIPE ViPipe, ISP_FOCUS_Q_INFO_S_
→*pstFocusQInfo);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
pstFocusQInfo	对焦状态信息结构体指针	输出

【返回值】

返回值	描述
0	成功。
非 0	失败，请参见错误码。

【需求】

- 头文件：cvi_af.h

【注意】

- 可用于调试和监控 AF 算法运行状态。
- eStatus 字段反映当前 AF 状态机的状态。

【相关主题】

- ISP_FOCUS_Q_INFO_S

6.4.2.12 CVI_ISP_AFSetFocus

【描述】

设置对焦马达移动方向和步长，用于手动对焦模式。

【语法】

```
CVI_S32 CVI_ISP_AFSetFocus(VI_PIPE ViPipe, CVI_BOOL direct, CVI_U16 step);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
direct	移动方向，CVI_TRUE 为 FAR（远）， CVI_FALSE 为 NEAR（近）	输入

下页继续

表 6.31 – 续上页

参数名称	描述	输入/输出
step	移动步长	输入

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见错误码。

【需求】

- 头文件: `cvi_af.h`

【注意】

- 该函数在手动对焦模式下使用, 自动对焦过程中不建议调用。

【相关主题】

- [CVI_ISP_AFSetFocusSpeed](#)
- [CVI_ISP_AFSetZoom](#)

6.4.2.13 CVI_ISP_AFSetFocusSpeed

【描述】

设置对焦马达移动速度。

【语法】

```
CVI_S32 CVI_ISP_AFSetFocusSpeed(VI_PIPE ViPipe, ISP_AF_MOTOR_SPEED_E eSpeed);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
eSpeed	马达速度: 4X / 2X / 1X / 半速	输入

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见错误码。

【需求】

- 头文件: `cvi_af.h`

【注意】

无。

【相关主题】

- CVI_ISP_AFSetFocus
- ISP_AF_MOTOR_SPEED_E

6.4.2.14 CVI_ISP_AFSetZoom**【描述】**

设置变焦马达移动方向和步长。

【语法】

```
CVI_S32 CVI_ISP_AFSetZoom(VI_PIPE ViPipe, CVI_BOOL direct, CVI_U16 step);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
direct	移动方向	输入
step	移动步长	输入

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见错误码。

【需求】

- 头文件: cvi_af.h

【注意】

- 变焦操作可能触发追焦（若 bChasingFocus 已使能）。

【相关主题】

- CVI_ISP_AFSetZoomSpeed
- CVI_ISP_AFSetFocus

6.4.2.15 CVI_ISP_AFSetZoomSpeed**【描述】**

设置变焦马达移动速度，一般不需要设置。

【语法】

```
CVI_S32 CVI_ISP_AFSetZoomSpeed(VI_PIPE ViPipe, ISP_AF_MOTOR_SPEED_E eSpeed);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入
eSpeed	马达速度: 4X / 2X / 1X / 半速	输入

【返回值】

返回值	描述
0	成功。
非 0	失败, 请参见错误码。

【需求】

- 头文件: `cvi_af.h`

【注意】

- 该函数一般不需要调用, 使用默认速度即可。

【相关主题】

- [CVI_ISP_AFSetZoom](#)
- [ISP_AF_MOTOR_SPEED_E](#)

6.4.2.16 CVI_AF_GetMotorCB**【描述】**

获取马达控制回调函数指针。

【语法】

```
ISP_AF_MOTOR_FUNC_S *CVI_AF_GetMotorCB(VI_PIPE ViPipe);
```

【参数】

参数名称	描述	输入/输出
ViPipe	ViPipe 号	输入

【返回值】

返回值	描述
非 NULL	马达控制回调函数指针。
NULL	失败, 未注册或 ViPipe 无效。

【需求】

- 头文件: `cvi_af.h`

【注意】

无。

【相关主题】

- CVI_AF_MOTOR_Register

6.5 数据类型

6.5.1 枚举类型

6.5.1.1 AF_STATUS

【说明】

定义 AF 算法状态机状态。

【定义】

```
typedef enum _AF_STATUS {
    AF_NOT_INIT,
    AF_INIT_POSITION,
    AF_INIT,
    AF_TRIGGER_FOCUS,
    AF_DETECT_DIRECTION,
    AF_SERACH_BEST_POS,
    AF_LOCATE_BEST_POS,
    AF_FOCUSED,
    AF_REVERSE_DIRECTION,
    AF_CHASING_FOCUS
} AF_STATUS;
```

【成员】

成员名称	描述
AF_NOT_INIT	未初始化状态。
AF_INIT_POSITION	初始位置，马达归位阶段。将对焦/变焦马达移动到物理边界进行复位。
AF_INIT	空闲状态。等待对焦触发，处理外部变焦/对焦命令。
AF_TRIGGER_FOCUS	触发对焦。挂起 AE，开始对焦搜索。
AF_DETECT_DIRECTION	方向检测。移动对焦镜头并监测 FV 值趋势以确定最佳搜索方向。
AF_SERACH_BEST_POS	粗搜索。以较大步长进行对焦搜索，跟踪最大 FV 值。
AF_LOCATE_BEST_POS	精定位。以较小步长进行精细搜索，确认峰值位置。
AF_FOCUSED	对焦完成。搜索结束，恢复 AE，更新稳定 FV 值。
AF_REVERSE_DIRECTION	方向反转。处理马达方向切换时的齿隙补偿。
AF_CHASING_FOCUS	追焦。变焦后根据 Zoom-Focus 查找表快速重新寻找焦点。

【注意事项】

- AF_STATUS 主要用于 ISP_FOCUS_Q_INFO_S 的 eStatus 字段，反映 AF 当前运行状态。

【相关数据类型及接口】

- ISP_FOCUS_Q_INFO_S

6.5.1.2 AF_DIRECTION

【说明】

定义对焦马达移动方向。

【定义】

```
typedef enum _AF_DIRECTION {
    AF_DIR_NEAR,
    AF_DIR_FAR,
} AF_DIRECTION;
```

【成员】

成员名称	描述
AF_DIR_NEAR	近处方向。马达向近距离方向移动镜头。
AF_DIR_FAR	远处方向。马达向远距离方向移动镜头。

【注意事项】

无。

【相关数据类型及接口】

- ISP_FOCUS_ATTR_S
- CVI_ISP_AFSetFocus
- CVI_ISP_AFSetZoom

6.5.1.3 AF_CHASINGFOCUS_MODE

【说明】

定义追焦模式类型。

【定义】

```
typedef enum _AF_CHASINGFOCUS_MODE {
    AF_CHASINGFOCUS_FAST_MODE,
    AF_CHASINGFOCUS_NORMAL_MODE,
} AF_CHASINGFOCUS_MODE;
```

【成员】

成员名称	描述
AF_CHASINGFOCUS_FAST_MODE	快速追焦模式。以更快的速度重新对焦，适合动态场景。
AF_CHASINGFOCUS_NORMAL_MODE	普通追焦模式。标准速度重新对焦。

【注意事项】

无。

【相关数据类型及接口】

· ISP_FOCUS_ATTR_S

6.5.1.4 AF_MANUAL_TYPE

【说明】

定义手动操作类型。

【定义】

```
typedef enum _AF_MANUAL_TYPE {  
    OP_TYPE_ZOOM,  
    OP_TYPE_FOCUS,  
    OP_TYPE_AUTO_FOCUS,  
    OP_TYPE_ZOOM_POS,  
    OP_TYPE_FOCUS_POS,  
    OP_TYPE_AF_BUTT  
} AF_MANUAL_TYPE;
```

【成员】

成员名称	描述
OP_TYPE_ZOOM	手动变焦操作。
OP_TYPE_FOCUS	手动对焦操作。
OP_TYPE_AUTO_FOCUS	自动对焦操作。
OP_TYPE_ZOOM_POS	设置变焦位置。
OP_TYPE_FOCUS_POS	设置对焦位置。
OP_TYPE_AF_BUTT	边界值，表示有效类型的数量。

【注意事项】

无。

【相关数据类型及接口】

· ISP_FOCUS_MANUAL_ATTR_S

6.5.1.5 ISP_AF_MOTOR_SPEED_E

【说明】

定义 AF 马达速度等级。

【定义】

```
typedef enum _ISP_AF_MOTOR_SPEED_E {  
    AF_MOTOR_SPEED_4X,  
    AF_MOTOR_SPEED_2X,  
    AF_MOTOR_SPEED_1X,  
}
```

(下页继续)

(续上页)

```
AF_MOTOR_SPEED_HALF,  
} ISP_AF_MOTOR_SPEED_E;
```

【成员】

成员名称	描述
AF_MOTOR_SPEED_4X	4 倍速度。最高速度等级。
AF_MOTOR_SPEED_2X	2 倍速度。
AF_MOTOR_SPEED_1X	1 倍速度。标准速度。
AF_MOTOR_SPEED_HALF	半速。最低速度等级。

【注意事项】

无。

【相关数据类型及接口】

- CVI_ISP_AFSetFocusSpeed
- CVI_ISP_AFSetZoomSpeed
- ISP_FOCUS_Q_INFO_S

6.5.1.6 AF_MOTOR_TYPE

【说明】

定义马达类型。

【定义】

```
typedef enum _AF_MOTOR_TYPE {  
    AF_MOTOR_STEPPER,  
    AF_MOTOR_VCM,  
} AF_MOTOR_TYPE;
```

【成员】

成员名称	描述
AF_MOTOR_STEPPER	步进马达。通过步进驱动，存在机械齿隙需补偿。
AF_MOTOR_VCM	音圈马达 (Voice Coil Motor)。通过电流驱动，无机械齿隙。

【注意事项】

- 马达类型通过 pfn_af_get_len_info 回调中的 ISP_AF_LEN_INFO_S 返回。
- VCM 镜头齿隙补偿为 1 帧震动补偿，步进马达需根据实际测量设置 focus_backlash。

【相关数据类型及接口】

- ISP_AF_LEN_INFO_S

6.5.2 结构体类型

6.5.2.1 ISP_FOCUS_ATTR_S

【说明】

定义 AF 主配置属性。包含 AF 使能、工作模式、搜索步长、追焦配置、比率阈值、手动对焦等参数。

【定义】

```
typedef struct _ISP_FOCUS_ATTR_S {  
    CVI_BOOL bEnable;  
    CVI_U8 u8DebugMode;  
    ISP_OP_TYPE_E enOpType;  
    CVI_U8 u8RunInterval;  
    CVI_BOOL bRealTimeFocus;  
    CVI_BOOL bChasingFocus;  
    CVI_BOOL bReFocus;  
    CVI_BOOL bMixHlc;  
    CVI_U8 u8RtFocusStableFrm;  
    CVI_U16 u16RtMaxDiffFvRatio;  
    CVI_U16 u16MaxDiffFvRatio;  
    CVI_U16 u16MaxDiffLumaRatio;  
    CVI_U16 u16DetectDiffRatio;  
    CVI_U16 u16SearchDiffRatio;  
    CVI_U16 u16LocalDiffRatio;  
    CVI_U8 u8InitStep;  
    CVI_U8 u8FindStep;  
    CVI_U8 u8LocateStep;  
    AF_DIRECTION enInitDir;  
    CVI_U16 u16MaxRotateCnt;  
    AF_CHASINGFOCUS_MODE enChasingFocusMode;  
    ISP_FOCUS_MANUAL_ATTR_S stManual;  
} ISP_FOCUS_ATTR_S;
```

【成员】

成员名称	描述
bEnable	AF 使能。为 CVI_TRUE 时使能 AF，为 CVI_FALSE 时不使能。
u8DebugMode	调试模式。用于设置 AF 调试等级。
enOpType	操作类型。自动或手动对焦模式切换。
u8RunInterval	AF 算法运行帧间隔。取值范围为 [0x1, 0x10]。
bRealTimeFocus	实时对焦使能。为 CVI_TRUE 时开启实时追焦，检测到画面变化自动重新对焦。
bChasingFocus	追焦使能。为 CVI_TRUE 时开启变焦追焦，变焦后自动根据 Zoom-Focus 查找表调整对焦位置。
bReFocus	重新对焦使能。为 CVI_TRUE 时画面变化后自动重新对焦。
bMixHlc	混合高亮点掩码使能。使用 HlCnt 信息抑制光源对 FV 值的影响。

下页继续

表 6.47 – 续上页

成员名称	描述
u8RtFocusStableFrm	实时对焦稳定帧数。FV 变化持续超过此帧数后触发追焦。取值范围为 [0x1, 0xFF]，默认值为 5。
u16RtMaxDiffFvRatio	实时对焦最大 FV 差异比例。基于 AF_RATIO_BASE (1024)。取值范围为 [0x400, 0x1000]。
u16MaxDiffFvRatio	最大 FV 差异比例。用于检测画面是否发生显著变化。取值范围为 [0x400, 0x4000]。
u16MaxDiffLumaRatio	最大亮度差异比例。用于亮度变化检测。取值范围为 [0x400, 0x1000]。
u16DetectDiffRatio	方向检测差异比例。用于判断方向检测阶段 FV 变化是否显著。取值范围为 [0x400, 0x1000]。
u16SearchDiffRatio	搜索差异比例。用于粗搜索阶段判断是否已越过 FV 峰值。取值范围为 [0x400, 0x1000]。
u16LocalDiffRatio	定位差异比例。用于精定位阶段确认峰值。取值范围为 [0x400, 0x1000]。
u8InitStep	初始搜索步长。对焦搜索第一阶段使用的马达移动步长。取值范围为 [0x1, 0xFF]。
u8FindStep	精确搜索步长。对焦搜索第二阶段（粗搜索）使用的马达移动步长。取值范围为 [0x1, 0xFF]。
u8LocateStep	定位步长。对焦搜索第三阶段（精定位）使用的马达移动步长。取值范围为 [0x1, 0xFF]。
enInitDir	初始搜索方向。对焦搜索开始时的马达移动方向（NEAR 或 FAR）。
u16MaxRotateCnt	最大复位次数。马达复位（归位到物理边界）的最大重试次数。取值范围为 [0x400, 0x4000]。
enChasingFocusMode	追焦模式。快速模式或普通模式。
stManual	手动对焦属性。包含手动操作类型、方向、步长和位置。

【注意事项】

- 所有比率参数均以 AF_RATIO_BASE (1024) 为基准。例如 u16DetectDiffRatio = 1024 表示 100% 的差异阈值。
- 三个步长参数（InitStep/FindStep/LocateStep）应根据镜头特性调整。步进马达建议 InitStep=16, FindStep=8, LocateStep=2；VCM 镜头可适当减小。

【相关数据类型及接口】

- CVI_ISP_SetAFAttr
- CVI_ISP_GetAFAttr
- ISP_FOCUS_MANUAL_ATTR_S

6.5.2.2 ISP_FOCUS_Q_INFO_S

【说明】

定义 AF 查询信息/运行时状态。用于获取 AF 当前运行状态、FV 值、马达位置等信息。

【定义】

```
typedef struct _ISP_AF_Q_INFO_S {  
    AF_STATUS eStatus;  
    CVI_U32 u32PreFv;  
    CVI_U32 u32CurFv;  
    CVI_U32 u32MaxFv;  
    CVI_U8 u8PreLuma;  
    CVI_U8 u8CurLuma;  
    AF_DIRECTION eZoomDir;  
    CVI_U16 u16ZoomStep;  
    CVI_U16 u16ZoomPos;  
    AF_DIRECTION eFocusDir;  
    CVI_U16 u16FocusStep;  
    CVI_U16 u16FocusPos;  
    CVI_U16 u16MaxFvPos;  
    CVI_U16 u16ZoomMinPos;  
    CVI_U16 u16ZoomMaxPos;  
    CVI_U16 u16FocusMinPos;  
    CVI_U16 u16FocusMaxPos;  
    ISP_AF_MOTOR_SPEED_E eZoomSpeed;  
    ISP_AF_MOTOR_SPEED_E eFocusSpeed;  
} ISP_FOCUS_Q_INFO_S;
```

【成员】

成员名称	描述
eStatus	当前 AF 状态。反映 AF_STATUS 枚举值。
u32PreFv	上一帧 FV 值。
u32CurFv	当前帧 FV 值。
u32MaxFv	搜索过程中记录的最大 FV 值。
u8PreLuma	上一帧平均亮度。
u8CurLuma	当前帧平均亮度。
eZoomDir	当前变焦移动方向。
u16ZoomStep	当前变焦步长。
u16ZoomPos	当前变焦位置。
eFocusDir	当前对焦移动方向。
u16FocusStep	当前对焦步长。
u16FocusPos	当前对焦位置。
u16MaxFvPos	最大 FV 值对应的对焦位置（最佳对焦点）。
u16ZoomMinPos	变焦最小位置。
u16ZoomMaxPos	变焦最大位置。
u16FocusMinPos	对焦最小位置。
u16FocusMaxPos	对焦最大位置。
eZoomSpeed	当前变焦马达速度。
eFocusSpeed	当前对焦马达速度。

【注意事项】

无。

【相关数据类型及接口】

- CVI_ISP_AFQueryFocusInfo
- AF_STATUS
- ISP_AF_MOTOR_SPEED_E

6.5.2.3 ISP_FOCUS_MANUAL_ATTR_S**【说明】**

定义手动对焦属性。用于手动对焦模式下的参数配置。

【定义】

```
typedef struct _ISP_FOCUS_MANUAL_ATTR_S {  
    AF_MANUAL_TYPE enManualOpType;  
    AF_DIRECTION enManualDir;  
    CVI_U16 u16ManualStep;  
    CVI_U16 u16ManualPos;  
} ISP_FOCUS_MANUAL_ATTR_S;
```

【成员】

成员名称	描述
enManualOpType	手动操作类型。变焦、对焦、自动对焦等。取值范围参见AF_MANUAL_TYPE。
enManualDir	手动操作方向。NEAR 或 FAR。取值范围参见AF_DIRECTION。
u16ManualStep	手动操作步长。取值范围为 [0x0, 0x400]。
u16ManualPos	手动操作目标位置。取值范围为 [0x0, 0x8000]。

【注意事项】

无。

【相关数据类型及接口】

- ISP_FOCUS_ATTR_S
- AF_MANUAL_TYPE

6.5.2.4 ISP_AF_VCM_ATTR_S

【说明】

定义 VCM 镜头标定属性。用于记录镜头的无限远位置、最近对焦位置和中点位置。

【定义】

```
typedef struct _ISP_AF_VCM_ATTR_S {  
    CVI_U32 u32InfinitePos;  
    CVI_U32 u32MacroPos;  
    CVI_U32 u32MediumPos;  
} ISP_AF_VCM_ATTR_S;
```

【成员】

成员名称	描述
u32InfinitePos	无限远位置。远处物体清晰对焦时的镜头位置，单位为步 (step)。
u32MacroPos	最近对焦位置。微距拍摄时的镜头位置，单位为步 (step)。
u32MediumPos	中点位置。无限远和最近对焦之间的中间位置，单位为步 (step)。

【注意事项】

- 这些值需要通过实际标定获得。建议使用标定工具 AF_ENTER_CALIB_FLOW 中的 AF_RANGE_OFFSET_CALIB 模式。
- 标定完成后，AF 算法利用这些值确定对焦搜索的合理范围。

【相关数据类型及接口】

- CVI_ISP_SetAFVcmAttr
- CVI_ISP_GetAFVcmAttr

6.5.2.5 ISP_AF_ZONE_FOCUS_ATTR_S

【说明】

定义 ROI 对焦区域属性。用于指定局部对焦区域的坐标、大小和权重。

【定义】

```
typedef struct _ISP_AF_ZONE_FOCUS_ATTR_S {  
    CVI_BOOL enable;  
    CVI_U8 u8ZoneWeight;  
    CVI_U32 u32ZoneX;  
    CVI_U32 u32ZoneY;  
    CVI_U32 u32ZoneWidth;  
    CVI_U32 u32ZoneHeight;  
} ISP_AF_ZONE_FOCUS_ATTR_S;
```

【成员】

成员名称	描述
enable	使能开关。为 CVI_TRUE 时启用 ROI 对焦区域，为 CVI_FALSE 时使用全局权重表。
u8ZoneWeight	区域权重。取值范围为 [0, 255]。
u32ZoneX	区域 X 坐标（左上角）。
u32ZoneY	区域 Y 坐标（左上角）。
u32ZoneWidth	区域宽度。
u32ZoneHeight	区域高度。

【注意事项】

- ROI 对焦区域可用于实现点击对焦等功能，用户指定画面中某区域后 AF 将优先对该区域进行对焦。

【相关数据类型及接口】

- CVI_ISP_SetAFZoneFocusAttr
- CVI_ISP_GetAFZoneFocusAttr

6.5.2.6 ISP_AF_ZOOM_FOCUS_TAB**【说明】**

定义 Zoom-Focus 查找表单个表项。记录特定变焦位置对应的对焦位置范围。

【定义】

```
typedef struct _ISP_AF_ZOOM_FOCUS_TAB {
    CVI_U32 zoom_pos;
    CVI_U32 focus_pos_min;
    CVI_U32 focus_pos_max;
} ISP_AF_ZOOM_FOCUS_TAB;
```

【成员】

成员名称	描述
zoom_pos	变焦位置。
focus_pos_min	该变焦位置下对焦位置的最小值。
focus_pos_max	该变焦位置下对焦位置的最大值。

【注意事项】

- Zoom-Focus 表共 ZOOM_FOCUS_TAB_SIZE (9) 个表项，覆盖整个变焦范围。
- 该表通过标定工具 AF_ZOOM_FOCUS_TABLE_CALIB 模式生成，在不同变焦位置记录最佳对焦位置范围。
- 追焦时使用线性插值在表项之间估算对焦位置。

【相关数据类型及接口】

- ISP_AF_LEN_INFO_S

6.5.2.7 ISP_AF_LEN_INFO_S

【说明】

定义镜头硬件特性信息。通过马达驱动回调 pfn_af_get_len_info 提供，包含镜头的机械和电气特性参数。

【定义】

```
typedef struct ISP_AF_LEN_INFO_S {
    ISP_AF_ZOOM_FOCUS_TAB zoom_focus_table[ZOOM_FOCUS_TAB_SIZE];
    CVI_U32 focus_range;
    CVI_U32 zoom_range;
    CVI_U32 focus_offset;
    CVI_U32 zoom_offset;
    CVI_U32 focus_backlash;
    CVI_U32 zoom_backlash;
    CVI_U32 focus_max_speed;
    CVI_U32 zoom_max_speed;
    CVI_U32 focus_time_cost_one_step;
    CVI_U32 zoom_time_cost_one_step;
    CVI_U32 focus_max_step;
    CVI_U32 zoom_max_step;
    AF_MOTOR_TYPE motor_type;
} ISP_AF_LEN_INFO_S;
```

【成员】

成员名称	描述
zoom_focus_table	Zoom-Focus 查找表。共 9 个表项，记录变焦位置与对焦位置范围的映射关系。
focus_range	对焦移动范围，单位为步（step）。
zoom_range	变焦移动范围，单位为步（step）。
focus_offset	对焦偏移量，单位为步（step）。表示镜头到达边界后仍有模糊的距离。
zoom_offset	变焦偏移量，单位为步（step）。
focus_backlash	对焦齿隙值，硬件限制，单位为步（step）。马达反向运动时需预走的补偿步长。
zoom_backlash	变焦齿隙值，硬件限制，单位为步（step）。
focus_max_speed	对焦最大速度。该值作为 MOTOR_SPEED_4X 的实际速度。
zoom_max_speed	变焦最大速度。该值作为 MOTOR_SPEED_4X 的实际速度。
focus_time_cost_one_step	对焦单步时间成本，单位为微秒（us）。
zoom_time_cost_one_step	变焦单步时间成本，单位为微秒（us）。
focus_max_step	对焦最大步数。
zoom_max_step	变焦最大步数。
motor_type	马达类型。步进马达或 VCM。参见 AF_MOTOR_TYPE。

【注意事项】

- 该结构体由马达驱动回调函数 pfn_af_get_len_info 填充，客户需正确实现该回调。
- focus_backlash 和 zoom_backlash 可通过标定工具 AF_BACK_LASH_CALIB 模式测量。

- `focus_time_cost_one_step` 和 `zoom_time_cost_one_step` 可通过标定工具 `AF_ONE_STEP_TIME_COST_CALIB` 模式测量。

【相关数据类型及接口】

- `ISP_AF_MOTOR_FUNC_S`
- `ISP_AF_ZOOM_FOCUS_TAB`
- `AF_MOTOR_TYPE`

6.5.2.8 ISP_AF_MOTOR_FUNC_S

【说明】

定义马达控制回调函数集合。客户需实现所有回调函数以驱动对焦/变焦马达。

【定义】

```
typedef struct _ISP_AF_MOTOR_CTL_FUNC_S {
    CVI_S32 (*pfn_af_motor_init)(VI_PIPE ViPipe);
    CVI_S32 (*pfn_af_motor_deinit)(VI_PIPE ViPipe);
    CVI_S32 (*pfn_af_set_zoom_in)(VI_PIPE ViPipe, CVI_U8 step);
    CVI_S32 (*pfn_af_set_zoom_out)(VI_PIPE ViPipe, CVI_U8 step);
    CVI_S32 (*pfn_af_set_zoom_speed)(VI_PIPE ViPipe, CVI_U8 speed);
    CVI_S32 (*pfn_af_set_focus_in)(VI_PIPE ViPipe, CVI_U8 step);
    CVI_S32 (*pfn_af_set_focus_out)(VI_PIPE ViPipe, CVI_U8 step);
    CVI_S32 (*pfn_af_set_focus_speed)(VI_PIPE ViPipe, CVI_U8 speed);
    CVI_S32 (*pfn_af_set_zoom_focus)(VI_PIPE ViPipe, AF_DIRECTION eDirz, AF_
→ DIRECTION eDirf, CVI_U8 zoomStep, CVI_U8 focusStep);
    CVI_S32 (*pfn_af_get_len_info)(VI_PIPE ViPipe, ISP_AF_LEN_INFO_S *info);
} ISP_AF_MOTOR_FUNC_S;
```

【成员】

成员名称	描述
<code>pfn_af_motor_init</code>	马达初始化回调。用于初始化马达硬件和状态。
<code>pfn_af_motor_deinit</code>	马达反初始化回调。用于释放马达资源。
<code>pfn_af_set_zoom_in</code>	设置变焦马达向内移动指定步长。
<code>pfn_af_set_zoom_out</code>	设置变焦马达向外移动指定步长。
<code>pfn_af_set_zoom_speed</code>	设置变焦马达速度等级。
<code>pfn_af_set_focus_in</code>	设置对焦马达向内移动指定步长。
<code>pfn_af_set_focus_out</code>	设置对焦马达向外移动指定步长。
<code>pfn_af_set_focus_speed</code>	设置对焦马达速度等级。
<code>pfn_af_set_zoom_focus</code>	同时设置变焦和对焦马达的移动方向和步长。用于实现变焦和对焦同时运动。
<code>pfn_af_get_len_info</code>	获取镜头硬件特性信息。需填充 <code>ISP_AF_LEN_INFO_S</code> 结构体。

【注意事项】

- 所有回调函数都必须正确实现，否则 AF 算法无法正常工作。
- 通过 `CVI_AF_MOTOR_Register` 注册到 AF 模块。

- `pfn_af_set_zoom_focus` 回调用于实现变焦和对焦的联动，可在变焦过程中同时调整对焦位置。

【相关数据类型及接口】

- `CVI_AF_MOTOR_Register`
- `ISP_AF_LEN_INFO_S`

6.5.2.9 ISP_FOCUS_STATISTICS_CFG_S

【说明】

定义 AF 硬件统计信息配置。包含滤波器配置、窗口大小、权重表等参数。

【定义】

```
#define AF_WEIGHT_ZONE_ROW 15
#define AF_WEIGHT_ZONE_COLUMN 17
#define FIR_H_GAIN_NUM (5)
#define FIR_V_GAIN_NUM (3)

typedef struct ISP_FOCUS_STATISTICS_CFG_S {
    ISP_AF_CFG_S stConfig;
    ISP_AF_H_PARAM_S stHParam_FIR0;
    ISP_AF_H_PARAM_S stHParam_FIR1;
    ISP_AF_V_PARAM_S stVParam_FIR;
    CVI_U8 au8Weight[AF_WEIGHT_ZONE_ROW][AF_WEIGHT_ZONE_COLUMN];
} ISP_FOCUS_STATISTICS_CFG_S;
```

【成员】

成员名称	描述
<code>stConfig</code>	AF 基础配置。包含使能、窗口大小、低通滤波系数、裁剪区域、高通滤波 <code>coting</code> 值等。参见 <code>ISP_AF_CFG_S</code> 。
<code>stHParam_FIR0</code>	水平方向 FIR 滤波器 0 参数（高通滤波器）。5 抽头系数。参见 <code>ISP_AF_H_PARAM_S</code> 。
<code>stHParam_FIR1</code>	水平方向 FIR 滤波器 1 参数（高通滤波器）。5 抽头系数。参见 <code>ISP_AF_H_PARAM_S</code> 。
<code>stVParam_FIR</code>	垂直方向 FIR 滤波器参数（高通滤波器）。3 抽头系数。参见 <code>ISP_AF_V_PARAM_S</code> 。
<code>au8Weight</code>	区块权重表。大小为 15 x 17，每个权重取值范围为 [0x0, 0xF]。用于对不同区域的 FV 值进行加权。

【注意事项】

- 统计区域大小建议设置为 17（水平）x 15（垂直），与权重表大小一致。
- FIR 滤波器系数需要根据镜头光学特性调整。
- 参考配置可参见 3A 开发用户指南中的 FV 计算参考代码。

【相关数据类型及接口】

- `CVI_ISP_SetAFStatisticsConfig`

- CVI_ISP_GetAFStatisticsConfig
- ISP_AF_CFG_S

6.5.2.10 ISP_AF_CFG_S

【说明】

定义 AF 统计基础配置。包含使能、窗口大小、滤波配置、裁剪区域、高亮点阈值等。

【定义】

```
typedef struct _ISP_AF_CFG_S {  
    CVI_BOOL bEnable;  
    CVI_U16 u16Hwnd;  
    CVI_U16 u16Vwnd;  
    CVI_U8 u8HFltShift;  
    CVI_S8 s8HVFltLpCoeff[FIR_H_GAIN_NUM];  
    ISP_AF_RAW_CFG_S stRawCfg;  
    ISP_AF_PRE_FILTER_CFG_S stPreFltCfg;  
    ISP_AF_CROP_S stCrop;  
    CVI_U8 H0FltCoring;  
    CVI_U8 H1FltCoring;  
    CVI_U8 V0FltCoring;  
    CVI_U16 u16HighLumaTh;  
    CVI_U8 u8ThLow;  
    CVI_U8 u8ThHigh;  
    CVI_U8 u8GainLow;  
    CVI_U8 u8GainHigh;  
    CVI_U8 u8SlopLow;  
    CVI_U8 u8SlopHigh;  
} ISP_AF_CFG_S;
```

【成员】

成员名称	描述
bEnable	使能开关。为 CVI_TRUE 时使能 AF 统计。
u16Hwnd	水平方向区块窗口数量。取值范围为 [0x2, 0x11] (即 2~17)。
u16Vwnd	垂直方向区块窗口数量。取值范围为 [0x2, 0xF] (即 2~15)。
u8HFltShift	水平滤波移位量。取值范围为 [0x0, 0xF]。
s8HVFltLpCoeff	水平/垂直低通 FIR 滤波系数。5 抽头, 取值范围为 [0x0, 0x1F]。
stRawCfg	RAW 域前处理配置。包含 PreGamma 使能和 256 项 Gamma 查找表。
stPreFltCfg	预滤波配置。用于消除椒盐噪声对统计值的影响。
stCrop	图像裁剪配置。用于限定 AF 统计的区域范围。参见ISP_AF_CROP_S。
H0FltCoring	H0 滤波器 Coring 值。用于去除小幅度噪声。取值范围为 [0x0, 0xFF]。
H1FltCoring	H1 滤波器 Coring 值。取值范围为 [0x0, 0xFF]。
V0FltCoring	V0 滤波器 Coring 值。取值范围为 [0x0, 0xFF]。

下页继续

表 6.56 – 续上页

成员名称	描述
u16HighLumaTh	高亮点亮度阈值。用于抑制光源对 FV 值的影响。取值范围为 [0x0, 0xFF]。
u8ThLow	低阈值。用于增益和斜率计算。
u8ThHigh	高阈值。用于增益和斜率计算。
u8GainLow	低增益值。取值范围为 [0x0, 0xFE]。
u8GainHigh	高增益值。取值范围为 [0x0, 0xFE]。
u8SlopLow	低斜率值。取值范围为 [0x0, 0xF]。
u8SlopHigh	高斜率值。取值范围为 [0x0, 0xF]。

【注意事项】

- AF 窗口 X 区域必须在 [0x8 ~ 图像 xsize - 8] 之间，Y 区域必须在 [0x2 ~ 图像 ysize - 2] 之间。

【相关数据类型及接口】

- ISP_FOCUS_STATISTICS_CFG_S
- ISP_AF_CROP_S

6.5.2.11 ISP_AF_CROP_S**【说明】**

定义 AF 统计信息输入图像裁剪配置。用户可通过 X、Y、W、H 来限定 AF 统计区域。

【定义】

```
typedef struct _ISP_AF_CROP_S {
    CVI_BOOL bEnable;
    CVI_U16 u16X;
    CVI_U16 u16Y;
    CVI_U16 u16W;
    CVI_U16 u16H;
} ISP_AF_CROP_S;
```

【成员】

成员名称	描述
bEnable	使能开关。为 CVI_TRUE 时启用裁剪。
u16X	裁剪区域 X 起始坐标。取值范围为 [0x8, 0xFFF]。
u16Y	裁剪区域 Y 起始坐标。取值范围为 [0x2, 0xFFF]。
u16W	裁剪区域宽度。取值范围为 [0x10, 0xFFF]。
u16H	裁剪区域高度。取值范围为 [0xF0, 0xFFF]。

【注意事项】

- X 坐标最小值为 0x8，Y 坐标最小值为 0x2，这是硬件限制。
- 裁剪区域不能超出图像有效区域。

【相关数据类型及接口】

- ISP_AF_CFG_S

6.5.2.12 ISP_FOCUS_ZONE_S

【说明】

定义单个区块的 AF 统计信息输出。包含高通滤波值和高亮点计数。

【定义】

```
typedef struct _ISP_FOCUS_ZONE_S {  
    CVI_U16 u16HlCnt;  
    CVI_U64 u64h0;  
    CVI_U64 u64h1;  
    CVI_U32 u32v0;  
} ISP_FOCUS_ZONE_S;
```

【成员】

成员名称	描述
u16HlCnt	高亮点像素计数。用于检测窗口中的过曝亮点数量，辅助判断光源对焦的影响。
u64h0	水平方向 FIR 滤波器 0 的输出累积值。
u64h1	水平方向 FIR 滤波器 1 的输出累积值。
u32v0	垂直方向 FIR 滤波器的输出累积值。

【注意事项】

无。

【相关数据类型及接口】

- ISP_FE_FOCUS_STATISTICS_S
- ISP_AF_STATISTICS_S

6.5.2.13 ISP_AF_STATISTICS_S

【说明】

定义 AF 统计信息数据结构。封装前端（FE）AF 统计输出。

【定义】

```
typedef struct _ISP_AF_STATISTICS_S {  
    ISP_FE_FOCUS_STATISTICS_S stFEAFStat;  
} ISP_AF_STATISTICS_S;
```

【成员】

成员名称	描述
stFEAFStat	前端 AF 统计信息。包含 15 x 17 区块的度量数据。参见 ISP_FE_FOCUS_STATISTICS_S。

【注意事项】

- 该数据结构在 ISP 运行时由硬件自动更新，用户通过 CVI_ISP_GetFocusStatistics 获取。
- 目前 AF 只支持 Bayer 域的统计信息。

【相关数据类型及接口】

- CVI_ISP_GetFocusStatistics
- ISP_FOCUS_ZONE_S
- ISP_FE_FOCUS_STATISTICS_S

7 IMP

8 BLC

8.1 功能描述

BLC 为当外界没有任何光线照入时，Sensor 仍然输出的亮度值。ISP 需要扣除，以使亮度、颜色正常。

8.2 API 参考

- `CVI_ISP_SetBlackLevelAttr`：设置黑电平属性参数
- `CVI_ISP_GetBlackLevelAttr`：获取黑电平属性参数

8.2.1 `CVI_ISP_SetBlackLevelAttr`

【描述】

设置黑电平属性参数

【语法】

```
CVI_S32 CVI_ISP_SetBlackLevelAttr(VI_PIPE ViPipe, const ISP_BLACK_LEVEL_ATTR_S_
↪ *pstBlackLevelAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstBlackLevelAttr	黑电平属性参数	输入

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

```
VI_PIPE ViPipe = 0;
ISP_BLACK_LEVEL_ATTR_S stAttr;
CVI_ISP_GetBlackLevelAttr(ViPipe, &stAttr);
stAttr.enOpType = OP_TYPE_AUTO;
CVI_ISP_SetBlackLevelAttr(ViPipe, &stAttr);
```

【相关主题】

- [CVI_ISP_GetBlackLevelAttr](#)

8.2.2 CVI_ISP_GetBlackLevelAttr

【描述】

获取黑电平属性参数

【语法】

```
CVI_S32 CVI_ISP_GetBlackLevelAttr(VI_PIPE ViPipe, ISP_BLACK_LEVEL_ATTR_S_
↪ *pstBlackLevelAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstBlackLevelAttr	黑电平属性参数	输出

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

无

【相关主题】

- CVI_ISP_SetBlackLevelAttr

8.3 数据类型

- ISP_BLACK_LEVEL_MANUAL_ATTR_S：定义手动黑电平属性参数
- ISP_BLACK_LEVEL_AUTO_ATTR_S：定义自动黑电平属性参数
- ISP_BLACK_LEVEL_ATTR_S：定义黑电平属性参数

8.3.1 ISP_BLACK_LEVEL_MANUAL_ATTR_S

【说明】

定义手动黑电平属性参数

【定义】

```
typedef struct _ISP_BLACK_LEVEL_MANUAL_ATTR_S {
    CVI_U16 OffsetR; /*Rw; Range:[0, 4095]*/
    CVI_U16 OffsetGr; /*Rw; Range:[0, 4095]*/
    CVI_U16 OffsetGb; /*Rw; Range:[0, 4095]*/
    CVI_U16 OffsetB; /*Rw; Range:[0, 4095]*/
    CVI_U16 OffsetR2; /*Rw; Range:[0, 4095]*/
    CVI_U16 OffsetGr2; /*Rw; Range:[0, 4095]*/
    CVI_U16 OffsetGb2; /*Rw; Range:[0, 4095]*/
    CVI_U16 OffsetB2; /*Rw; Range:[0, 4095]*/
} ISP_BLACK_LEVEL_MANUAL_ATTR_S;
```

【成员】

成员名称	描述
OffsetR	R 像素暗电流值 取值范围：[0, 4095] 数据类型：CVI_U16
OffsetGr	Gr 像素暗电流值 取值范围：[0, 4095] 数据类型：CVI_U16
OffsetGb	Gb 像素暗电流值 取值范围：[0, 4095] 数据类型：CVI_U16
OffsetB	B 像素暗电流值 取值范围：[0, 4095] 数据类型：CVI_U16

下页继续

表 8.3 – 续上页

成员名称	描述
OffsetR2	第二级 R 像素暗电流值 取值范围：[0, 4095] 数据类型：CVI_U16
OffsetGr2	第二级 Gr 像素暗电流值 取值范围：[0, 4095] 数据类型：CVI_U16
OffsetGb2	第二级 Gb 像素暗电流值 取值范围：[0, 4095] 数据类型：CVI_U16
OffsetB2	第二级 B 像素暗电流值 取值范围：[0, 4095] 数据类型：CVI_U16

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetBlackLevelAttr
- CVI_ISP_GetBlackLevelAttr

8.3.2 ISP_BLACK_LEVEL_AUTO_ATTR_S

【说明】

定义自动黑电平属性参数

【定义】

```
typedef struct _ISP_BLACK_LEVEL_AUTO_ATTR_S {
    CVI_U16 OffsetR[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 4095]*/
    CVI_U16 OffsetGr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 4095]*/
    CVI_U16 OffsetGb[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 4095]*/
    CVI_U16 OffsetB[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 4095]*/
    CVI_U16 OffsetR2[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 4095]*/
    CVI_U16 OffsetGr2[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 4095]*/
    CVI_U16 OffsetGb2[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 4095]*/
    CVI_U16 OffsetB2[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 4095]*/
} ISP_BLACK_LEVEL_AUTO_ATTR_S;
```

【成员】

成员名称	描述
OffsetR	R 像素暗电流值 取值范围：[0, 4095] 数据类型：CVI_U16

下页继续

表 8.4 – 续上页

成员名称	描述
OffsetGr	Gr 像素暗电流值 取值范围：[0, 4095] 数据类型：CVI_U16
OffsetGb	Gb 像素暗电流值 取值范围：[0, 4095] 数据类型：CVI_U16
OffsetB	B 像素暗电流值 取值范围：[0, 4095] 数据类型：CVI_U16
OffsetR2	第二级 R 像素暗电流值 取值范围：[0, 4095] 数据类型：CVI_U16
OffsetGr2	第二级 Gr 像素暗电流值 取值范围：[0, 4095] 数据类型：CVI_U16
OffsetGb2	第二级 Gb 像素暗电流值 取值范围：[0, 4095] 数据类型：CVI_U16
OffsetB2	第二级 B 像素暗电流值 取值范围：[0, 4095] 数据类型：CVI_U16

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetBlackLevelAttr
- CVI_ISP_GetBlackLevelAttr

8.3.3 ISP_BLACK_LEVEL_ATTR_S

【说明】

定义黑电平属性参数

【定义】

```
typedef struct _ISP_BLACK_LEVEL_ATTR_S {  
    CVI_BOOL Enable; /*Rw; Range:[0, 1]*/  
    ISP_OP_TYPE_E enOpType;  
    CVI_U8 UpdateInterval; /*Rw; Range:[0, 255]*/  
    ISP_BLACK_LEVEL_MANUAL_ATTR_S stManual;  
    ISP_BLACK_LEVEL_AUTO_ATTR_S stAuto;  
} ISP_BLACK_LEVEL_ATTR_S;
```

【成员】

成员名称	描述
Enable	模块使能开关 取值范围：[0, 1] 数据类型：CVI_BOOL
enOpType	工作类型 OP_TYPE_AUTO：自动模式 OP_TYPE_MANUAL：手动模式 数据类型：ISP_OP_TYPE_E
UpdateInterval	影响参数更新间隔，值越大画面变化越慢，效能越好。 取值范围：[0, 255] 数据类型：CVI_U8
stManual	手动参数
stAuto	自动参数

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetBlackLevelAttr
- CVI_ISP_GetBlackLevelAttr

9 DPC

9.1 功能描述

DPC 全称为 Defect Pixel Correction，可以修正 Sensor 上有问题的坏点。

修正方法有两种，分别为静态校正与动态校正。

静态校正: 校正时分为暗点校正与亮点校正。

亮点校正时，把镜头遮黑，启动坏点标定程序。

暗点校正时，不遮黑在平坦背景时如灰度箱前，调整曝光让图像整体亮度约为 50%。

静态坏点最多允许 4095 个坏点。

动态校正: 使用此方法时，不使用校正数据而是直接动态判断坏点，并且加以修正。在低噪时对画面偏色有帮助，但若强度太强可能会致画面细节降低。

【特别提醒】

这里 cv184x 系列只用到动态校正

9.2 API 参考

- `CVI_ISP_SetDPDynamicAttr`：设置动态坏点校正属性
- `CVI_ISP_GetDPDynamicAttr`：获取动态坏点校正属性

9.2.1 CVI_ISP_SetDPDynamicAttr

【描述】

设置动态坏点校正属性

【语法】

```
CVI_S32 CVI_ISP_SetDPDynamicAttr(VI_PIPE ViPipe, const ISP_DP_DYNAMIC_ATTR_S_
↪ *pstDPDynamicAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstDPDynamicAttr	动态坏点校正属性	输入

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: cvi_isp.h, cvi_comm_isp.h
- 库文件: libisp.so

【注意】

无

【举例】

```
VI_PIPE ViPipe = 0;
ISP_DP_DYNAMIC_ATTR_S stAttr;
CVI_ISP_GetDPDynamicAttr(ViPipe, &stAttr);
stAttr.enOpType = OP_TYPE_AUTO;
CVI_ISP_SetDPDynamicAttr(ViPipe, &stAttr);
```

【相关主题】

- [CVI_ISP_GetDPDynamicAttr](#)

9.2.2 CVI_ISP_GetDPDynamicAttr

【描述】

获取动态坏点校正属性

【语法】

```
CVI_S32 CVI_ISP_GetDPDynamicAttr(VI_PIPE ViPipe, ISP_DP_DYNAMIC_ATTR_S_
↪ *pstDPDynamicAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstDPDynamicAttr	动态坏点校正属性	输出

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: cvl_isp.h, cvl_comm_isp.h
- 库文件: libisp.so

【注意】

无

【举例】

无

【相关主题】

- CVI_ISP_SetDPDynamicAttr

9.3 数据类型

- ISP_DP_DYNAMIC_MANUAL_ATTR_S：定义手动动态坏点校正属性
- ISP_DP_DYNAMIC_AUTO_ATTR_S：定义自动动态坏点校正属性
- ISP_DP_DYNAMIC_ATTR_S：定义动态坏点校正属性

9.3.1 ISP_DP_DYNAMIC_MANUAL_ATTR_S

【说明】

定义手动动态坏点校正属性

【定义】

```
typedef struct _ISP_DP_DYNAMIC_MANUAL_ATTR_S {  
    CVI_U8 DefectCnt1; /*Rw; Range:[0, 16]*/  
    CVI_U8 DefectCnt2; /*Rw; Range:[0, 16]*/  
    CVI_U8 AdvMode; /*Rw; Range:[0, 1]*/  
    CVI_U8 AvgMode; /*Rw; Range:[0, 1]*/  
} ISP_DP_DYNAMIC_MANUAL_ATTR_S;
```

【成员】

成员名称	描述
DefectCnt1	坏点判断逻辑的阈值 1 取值范围：[0, 16] 数据类型：CVI_U8

下页继续

表 9.3 – 续上页

成员名称	描述
DefectCnt2	坏点判断逻辑的阈值 2 一般 DefectCnt2 需要小于 DefectCnt1 取值范围: [0, 16] 数据类型: CVI_U8
AdvMode	第二方向性补偿使能。 0: 关闭, 补偿方式取决于 AvgMode。 1: 使能。 取值范围: [0, 1] 数据类型: CVI_U8
AvgMode	坏点补偿模式。 0: 方向性补偿。 1: 无方向性补偿 (平均补偿)。 取值范围: [0, 1] 数据类型: CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetDPDynamicAttr
- CVI_ISP_GetDPDynamicAttr

9.3.2 ISP_DP_DYNAMIC_AUTO_ATTR_S

【说明】

定义自动动态坏点校正属性

【定义】

```
typedef struct _ISP_DP_DYNAMIC_AUTO_ATTR_S {
    CVI_U8 DefectCnt1[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 16]*/
    CVI_U8 DefectCnt2[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 16]*/
    CVI_U8 AdvMode[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 1]*/
    CVI_U8 AvgMode[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 1]*/
} ISP_DP_DYNAMIC_AUTO_ATTR_S;
```

【成员】

成员名称	描述
DefectCnt1	坏点判断逻辑的阈值 1 取值范围: [0, 16] 数据类型: CVI_U8

下页继续

表 9.4 – 续上页

成员名称	描述
DefectCnt2	坏点判断逻辑的阈值 2 一般 DefectCnt2 需要小于 DefectCnt1 取值范围: [0, 16] 数据类型: CVI_U8
AdvMode	第二方向性补偿使能。 0: 关闭, 补偿方式取决于 AvgMode。 1: 使能。 取值范围: [0, 1] 数据类型: CVI_U8
AvgMode	坏点补偿模式。 0: 方向性补偿。 1: 无方向性补偿 (平均补偿)。 取值范围: [0, 1] 数据类型: CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetDPDynamicAttr
- CVI_ISP_GetDPDynamicAttr

9.3.3 ISP_DP_DYNAMIC_ATTR_S

【说明】

定义动态坏点校正属性

【定义】

```
typedef struct _ISP_DP_DYNAMIC_ATTR_S {
    CVI_BOOL Enable; /*Rw; Range:[0, 1]*/
    ISP_OP_TYPE_E enOpType;
    CVI_U8 UpdateInterval; /*Rw; Range:[0, 255]*/
    CVI_U16 DarkDefectThresh[DP_THR_LUT_NUM]; /*Rw; Range:[0, 4095]*/
    CVI_U16 BrightDefectThresh[DP_THR_LUT_NUM]; /*Rw; Range:[0, 4095]*/
    CVI_U16 DarkDefectThreshOffset; /*Rw; Range:[0, 4095]*/
    CVI_U16 BrightDefectThreshOffset; /*Rw; Range:[0, 4095]*/
    CVI_U8 TransitionWeight; /*Rw; Range:[0, 255]*/
    ISP_DP_DYNAMIC_MANUAL_ATTR_S stManual;
    ISP_DP_DYNAMIC_AUTO_ATTR_S stAuto;
} ISP_DP_DYNAMIC_ATTR_S;
```

【成员】

成员名称	描述
Enable	模块使能开关 取值范围：[0, 1] 数据类型：CVI_BOOL
enOpType	工作类型 OP_TYPE_AUTO：自动模式 OP_TYPE_MANUAL：手动模式 数据类型：ISP_OP_TYPE_E
UpdateInterval	影响参数更新间隔，值越大画面变化越慢，效能越好。 取值范围：[0, 255] 数据类型：CVI_U8
DarkDefectThresh	根据 luma 查阈值 1 的 Lut，针对暗坏点。 取值范围：[0, 4095] 数据类型：CVI_U16
BrightDefectThresh	根据 luma 查阈值 1 的 Lut，针对亮坏点。 取值范围：[0, 4095] 数据类型：CVI_U16
DarkDefectThreshOffset	根据偏移量求得阈值 2，针对暗坏点。 取值范围：[0, 4095] 数据类型：CVI_U16
BrightDefectThreshOffset	根据偏移量求得阈值 2，针对亮坏点。 取值范围：[0, 4095] 数据类型：CVI_U16
TransitionWeight	当坏点判断条件处于过渡范围内时（不确定是不是坏点），这个权重决定是做 dpc 还是不做的比例。 取值范围：[0, 255] 数据类型：CVI_U8
stManual	手动参数
stAuto	自动参数

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetDPDynamicAttr
- CVI_ISP_GetDPDynamicAttr

10 LSC

10.1 功能描述

LSC, 即 Lens shading correction, 镜头阴影校正。

在 cv184x 系列中, LSC 会分成两个部分: 一个是 MLSC, 即 MLSC (Mesh Lens shading correction), 主要用来校正 Color Shading。在处理器中 LSC 算法使用网格方法对图像先做标定, 然后矫正, 算法会将 Bayer 上的域画面划分成 37×37 个子区块。Bayer 域四个通道由三组不同的 RGB 增益数组来做计算。由于只做颜色上的校正, 所以一般 G 通道的增益保持一倍的设定, 即 512。当标定 MLSC 组数为多组时, 则会根据当前的色温选出邻近的两色温之 MLSC 标定表进行内插, 产生对应当前色温的 MLSC 增益。标定的组数则由 LscGainLutSize 参数定义之。

另一个是 RLSC (Radius Lens shading correction), 主要用来校正 Luma Shading。在 LSC 算法中, 会以画面的亮度中心为圆心, 各像素点和圆心之间的距离为半径, 做不同程度亮度补偿。一般离亮度中心越远, 补偿强度越大。

这里主要介绍 MLSC 的相关 API 和数据结构, RLSC 将在下一章节介绍。

10.2 API 参考

- CVI_ISP_SetMeshShadingAttr : 设置 Mesh Shading 算法参数
- CVI_ISP_GetMeshShadingAttr : 获取 Mesh Shading 算法参数
- CVI_ISP_SetMeshShadingGainLutAttr : 设置 LSC 网格形式补偿增益表
- CVI_ISP_GetMeshShadingGainLutAttr : 获取 LSC 网格形式补偿增益表

10.2.1 CVI_ISP_SetMeshShadingAttr

【描述】

设置 Mesh Shading 算法参数

【语法】

```
CVI_S32 CVI_ISP_SetMeshShadingAttr(VI_PIPE ViPipe, const ISP_MESH_SHADING_ATTR_S_
↪ *pstMeshShadingAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstMeshShadingAttr	Mesh Shading 算法参数	输入

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: cvi_isp.h, cvi_comm_isp.h
- 库文件: libisp.so

【注意】

无

【举例】

```
VI_PIPE ViPipe = 0;
ISP_MESH_SHADING_ATTR_S stAttr;
CVI_ISP_GetMeshShadingAttr(ViPipe, &stAttr);
stAttr.enOpType = OP_TYPE_AUTO;
CVI_ISP_SetMeshShadingAttr(ViPipe, &stAttr);
```

【相关主题】

- [CVI_ISP_GetMeshShadingAttr](#)

10.2.2 CVI_ISP_GetMeshShadingAttr

【描述】

获取 Mesh Shading 算法参数

【语法】

```
CVI_S32 CVI_ISP_GetMeshShadingAttr(VI_PIPE ViPipe, ISP_MESH_SHADING_ATTR_S_
↪ *pstMeshShadingAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstMeshShadingAttr	Mesh Shading 算法参数	输出

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: cvi_isp.h, cvi_comm_isp.h
- 库文件: libisp.so

【注意】

无

【举例】

无

【相关主题】

- [CVI_ISP_SetMeshShadingAttr](#)

10.2.3 CVI_ISP_SetMeshShadingGainLutAttr

【描述】

设置 LSC 网格形式补偿增益表

【语法】

```
CVI_S32 CVI_ISP_SetMeshShadingGainLutAttr(VI_PIPE ViPipe, const ISP_MESH_SHADING_
→GAIN_LUT_ATTR_S *pstMeshShadingGainLutAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstMeshShading-GainLutAttr	LSC 网格形式补偿增益表	输入

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: cvi_isp.h, cvi_comm_isp.h
- 库文件: libisp.so

【注意】

无

【举例】

```
VI_PIPE ViPipe = 0;
ISP_MESH_SHADING_GAIN_LUT_ATTR_S stAttr;
CVI_ISP_GetMeshShadingGainLutAttr(ViPipe, &stAttr);
stAttr.enOpType = OP_TYPE_AUTO;
CVI_ISP_SetMeshShadingGainLutAttr(ViPipe, &stAttr);
```

【相关主题】

- [CVI_ISP_GetMeshShadingGainLutAttr](#)

10.2.4 CVI_ISP_GetMeshShadingGainLutAttr

【描述】

获取 LSC 网格形式补偿增益表

【语法】

```
CVI_S32 CVI_ISP_GetMeshShadingGainLutAttr(VI_PIPE ViPipe, ISP_MESH_SHADING_GAIN_
↪LUT_ATTR_S *pstMeshShadingGainLutAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstMeshShading-GainLutAttr	LSC 网格形式补偿增益表	输出

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

无

【相关主题】

- [CVI_ISP_SetMeshShadingGainLutAttr](#)

10.3 数据类型

- `ISP_MESH_SHADING_ATTR_S`：定义 Mesh Shading 算法参数
- `ISP_MESH_SHADING_GAIN_LUT_ATTR_S`：定义 LSC 网格形式补偿增益表
- `ISP_MESH_SHADING_GAIN_LUT_S`：定义 LSC 网格线形式补偿增益表细项

10.3.1 `ISP_MESH_SHADING_ATTR_S`

【说明】

定义 Mesh Shading 算法参数

【定义】

```
typedef struct _ISP_MESH_SHADING_ATTR_S {  
    CVI_BOOL Enable; /*Rw; Range:[0, 1]*/  
    ISP_OP_TYPE_E enOpType;  
    CVI_U8 UpdateInterval; /*Rw; Range:[0, 255]*/  
} ISP_MESH_SHADING_ATTR_S;
```

【成员】

成员名称	描述
Enable	模块使能开关 取值范围：[0, 1] 数据类型：CVI_BOOL
enOpType	工作类型 OP_TYPE_AUTO：自动模式 OP_TYPE_MANUAL：手动模式 数据类型：ISP_OP_TYPE_E
UpdateInterval	影响参数更新间隔，值越大画面变化越慢，效能越好。 取值范围：[0, 255] 数据类型：CVI_U8

【注意事项】

无

【相关数据类型及接口】

- `CVI_ISP_SetMeshShadingAttr`
- `CVI_ISP_GetMeshShadingAttr`

10.3.2 ISP_MESH_SHADING_GAIN_LUT_ATTR_S

【说明】

定义 LSC 网格形式补偿增益表

【定义】

```
typedef struct _ISP_MESH_SHADING_GAIN_LUT_ATTR_S {
    CVI_U8 Size; /*Rw; Range:[1, 7]*/
    ISP_MESH_SHADING_GAIN_LUT_S LscGainLut[ISP_MLSC_COLOR_TEMPERATURE_
    SIZE];
} ISP_MESH_SHADING_GAIN_LUT_ATTR_S;
```

【成员】

成员名称	描述
Size	色温自适应 LSC 补偿增益表数量 取值范围：[1, 7] 数据类型：CVI_U8
LscGainLut	LSC 网格线形式补偿增益表 数据类型：ISP_MESH_SHADING_GAIN_LUT_S

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetMeshShadingGainLutAttr
- CVI_ISP_GetMeshShadingGainLutAttr

10.3.3 ISP_MESH_SHADING_GAIN_LUT_S

【说明】

定义 LSC 网格线形式补偿增益表细项

【定义】

```
typedef struct _ISP_MESH_SHADING_GAIN_LUT_S {
    CVI_U16 ColorTemperature; /*Rw; Range:[0, 30000]*/
    CVI_U16 RGain[CVI_ISP_LSC_GRID_POINTS]; /*Rw; Range:[0, 4095]*/
    CVI_U16 GGain[CVI_ISP_LSC_GRID_POINTS]; /*Rw; Range:[0, 4095]*/
    CVI_U16 BGain[CVI_ISP_LSC_GRID_POINTS]; /*Rw; Range:[0, 4095]*/
} ISP_MESH_SHADING_GAIN_LUT_S;
```

【成员】

成员名称	描述
ColorTemperature	色温自适应 LSC 补偿增益表所对应之色温，单位为 K 取值范围：[0, 30000] 数据类型：CVI_U16
RGain	红色通道增益 取值范围：[0, 4095] 数据类型：CVI_U16
GGain	绿色通道增益 取值范围：[0, 4095] 数据类型：CVI_U16
BGain	蓝色通道增益 取值范围：[0, 4095] 数据类型：CVI_U16

【注意事项】

无

【相关数据类型及接口】

11 RLSC

11.1 功能描述

LSC, 即 Lens shading correction, 镜头阴影校正。

在 cv184x 系列中, LSC 会分成两个部分: 一个是 LSC, 即 MLSC (Mesh Lens shading correction), 主要用来校正 Color Shading。在处理器中 LSC 算法使用网格方法对图像先做标定, 然后矫正, 算法会将 Bayer 上的域画面划分成 37×37 个子区块。Bayer 域四个通道由三组不同的 RGB 增益数组来做计算。由于只做颜色上的校正, 所以一般 G 通道的增益保持一倍的设定, 即 512。当标定 MLSC 组数为复数组时, 则会根据当前的色温选出邻近的两色温之 MLSC 标定表进行内插, 产生对应当前色温的 MLSC 增益。标定的组数则由 LscGainLutSize 参数定义之。

另一个是 RLSC (Radius Lens shading correction), 主要用来校正 Luma Shading。在 LSC 算法中, 会以画面的亮度中心为圆心, 各像素点和圆心之间的距离为半径, 做不同程度亮度补偿。一般离亮度中心越远, 补偿强度越大。

这里主要介绍 RLSC 的相关 API 和数据结构, MLSC 已经在上一章节介绍。

11.2 API 参考

- CVI_ISP_SetRadialShadingAttr : 设置 Radius Shading 算法参数
- CVI_ISP_GetRadialShadingAttr : 获取 Radius Shading 算法参数
- CVI_ISP_SetRadialShadingGainLutAttr : 设置 LSC 半径形式补偿增益表
- CVI_ISP_GetRadialShadingGainLutAttr : 获取 LSC 半径形式补偿增益表

11.2.1 CVI_ISP_SetRadialShadingAttr

【描述】

设置 Radius Shading 算法参数

【语法】

```
CVI_S32 CVI_ISP_SetRadialShadingAttr(VI_PIPE ViPipe, const ISP_RADIAL_SHADING_
→ATTR_S *pstRadialShadingAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstRadialShadingAttr	Radius Shading 算法参数	输入

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

```
VI_PIPE ViPipe = 0;
ISP_RADIAL_SHADING_ATTR_S stAttr;
CVI_ISP_GetRadialShadingAttr(ViPipe, &stAttr);
stAttr.enOpType = OP_TYPE_AUTO;
CVI_ISP_SetRadialShadingAttr(ViPipe, &stAttr);
```

【相关主题】

- [CVI_ISP_GetRadialShadingAttr](#)

11.2.2 CVI_ISP_GetRadialShadingAttr

【描述】

获取 Radius Shading 算法参数

【语法】

```
CVI_S32 CVI_ISP_GetRadialShadingAttr(VI_PIPE ViPipe, ISP_RADIAL_SHADING_ATTR_S_
↪ *pstRadialShadingAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstRadialShadingAttr	Radius Shading 算法参数	输出

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

无

【相关主题】

- [CVI_ISP_SetRadialShadingAttr](#)

11.2.3 CVI_ISP_SetRadialShadingGainLutAttr

【描述】

设置 LSC 半径形式补偿增益表

【语法】

```
CVI_S32 CVI_ISP_SetRadialShadingGainLutAttr(VI_PIPE ViPipe, const ISP_RADIAL_
↪SHADING_GAIN_LUT_ATTR_S *pstRadialShadingGainLutAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstRadialShading-GainLutAttr	LSC 半径形式补偿增益表	输入

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

```
VI_PIPE ViPipe = 0;
ISP_RADIAL_SHADING_GAIN_LUT_ATTR_S stAttr;
CVI_ISP_GetRadialShadingGainLutAttr(ViPipe, &stAttr);
stAttr.enOpType = OP_TYPE_AUTO;
CVI_ISP_SetRadialShadingGainLutAttr(ViPipe, &stAttr);
```

【相关主题】

- [CVI_ISP_GetRadialShadingGainLutAttr](#)

11.2.4 CVI_ISP_GetRadialShadingGainLutAttr

【描述】

获取 LSC 半径形式补偿增益表

【语法】

```
CVI_S32 CVI_ISP_GetRadialShadingGainLutAttr(VI_PIPE ViPipe, ISP_RADIAL_SHADING_
↪GAIN_LUT_ATTR_S *pstRadialShadingGainLutAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstRadialShading-GainLutAttr	LSC 半径形式补偿增益表	输出

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: cvi_isp.h, cvi_comm_isp.h
- 库文件: libisp.so

【注意】

无

【举例】

无

【相关主题】

- CVI_ISP_SetRadialShadingGainLutAttr

11.3 数据类型

- ISP_RADIAL_SHADING_MANUAL_ATTR_S：定义手动 Radius Shading 算法参数
- ISP_RADIAL_SHADING_AUTO_ATTR_S：定义自动 Radius Shading 算法参数
- ISP_RADIAL_SHADING_ATTR_S：定义 Radius Shading 算法参数
- ISP_RADIAL_SHADING_GAIN_LUT_ATTR_S：定义 LSC 半径形式补偿增益表

11.3.1 ISP_RADIAL_SHADING_MANUAL_ATTR_S

【说明】

定义手动 Radius Shading 算法参数

【定义】

```
typedef struct _ISP_RADIAL_SHADING_MANUAL_ATTR_S {
    CVI_U16 RadiusStr; /*Rw; Range:[0, 4095]*/
    CVI_U8 NDStr; /*Rw; Range:[0, 255]*/
    CVI_U8 NDThr; /*Rw; Range:[0, 255]*/
} ISP_RADIAL_SHADING_MANUAL_ATTR_S;
```

【成员】

成员名称	描述
RadiusStr	RLSC 的全局强度。 取值范围：[0, 4095] 数据类型：CVI_U16
NDStr	调节噪声保留的强度，值越大，噪声保留越多。 取值范围：[0, 255] 数据类型：CVI_U8
NDThr	允许保留噪声的最大值。 取值范围：[0, 255] 数据类型：CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetRadialShadingAttr
- CVI_ISP_GetRadialShadingAttr

11.3.2 ISP_RADIAL_SHADING_AUTO_ATTR_S

【说明】

定义自动 Radius Shading 算法参数

【定义】

```
typedef struct _ISP_RADIAL_SHADING_AUTO_ATTR_S {
    CVI_U16 RadiusStr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 4095]*/
    CVI_U8 NDStr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U8 NDThr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
} ISP_RADIAL_SHADING_AUTO_ATTR_S;
```

【成员】

成员名称	描述
RadiusStr	RLSC 的全局强度。 取值范围：[0, 4095] 数据类型：CVI_U16
NDStr	调节噪声保留的强度，值越大，噪声保留越多。 取值范围：[0, 255] 数据类型：CVI_U8
NDThr	允许保留噪声的最大值。 取值范围：[0, 255] 数据类型：CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetRadialShadingAttr
- CVI_ISP_GetRadialShadingAttr

11.3.3 ISP_RADIAL_SHADING_ATTR_S

【说明】

定义 Radius Shading 算法参数

【定义】

```
typedef struct _ISP_RADIAL_SHADING_ATTR_S {
    CVI_BOOL Enable; /*Rw; Range:[0, 1]*/
    ISP_OP_TYPE_E enOpType;
    CVI_U8 UpdateInterval; /*Rw; Range:[0, 255]*/
    CVI_U16 CenterX; /*Rw; Range:[0, 8191]*/
    CVI_U16 CenterY; /*Rw; Range:[0, 8191]*/
    CVI_U16 RadiusScaleRGB; /*Rw; Range:[0, 32767]*/
    ISP_RADIAL_SHADING_MANUAL_ATTR_S stManual;
```

(下页继续)

(续上页)

```

    ISP_RADIAL_SHADING_AUTO_ATTR_S stAuto;
} ISP_RADIAL_SHADING_ATTR_S;

```

【成员】

成员名称	描述
Enable	模块使能开关 取值范围：[0, 1] 数据类型：CVI_BOOL
enOpType	工作类型 OP_TYPE_AUTO：自动模式 OP_TYPE_MANUAL：手动模式 数据类型：ISP_OP_TYPE_E
UpdateInterval	影响参数更新间隔，值越大画面变化越慢，效能越好。 取值范围：[0, 255] 数据类型：CVI_U8
CenterX	径向阴影校正的中心点 X 坐标。 取值范围：[0, 8191] 数据类型：CVI_U16
CenterY	径向阴影校正的中心点 Y 坐标。 取值范围：[0, 8191] 数据类型：CVI_U16
RadiusScaleRGB	控制径向阴影校正的半径缩放比例，影响 RGB 通道的校正范围。 取值范围：[0, 32767] 数据类型：CVI_U16
stManual	手动参数
stAuto	自动参数

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetRadialShadingAttr
- CVI_ISP_GetRadialShadingAttr

11.3.4 ISP_RADIAL_SHADING_GAIN_LUT_ATTR_S

【说明】

定义 LSC 半径形式补偿增益表

【定义】

```

typedef struct _ISP_RADIAL_SHADING_GAIN_LUT_ATTR_S {
    CVI_U16 GGain[ISP_RLSC_WINDOW_SIZE]; /*Rw; Range:[0, 4095]*/
} ISP_RADIAL_SHADING_GAIN_LUT_ATTR_S;

```

【成员】

成员名称	描述
GGain	亮度补偿增益表，由标定得来。 取值范围：[0, 4095] 数据类型：CVI_U16

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetRadialShadingGainLutAttr
- CVI_ISP_GetRadialShadingGainLutAttr

12 CCM

12.1 功能描述

CCM 为 Color Correction Matrix 的简称。因为 Sensor 在 RGB 的响应与人眼不同，所以需要使用时使用此矩阵做转换，让前端捕获的图片亮度与人眼一致。

12.2 API 参考

- `CVI_ISP_SetCCMAAttr`：设置色彩校正矩阵属性参数
- `CVI_ISP_GetCCMAAttr`：获取色彩校正矩阵属性参数
- `CVI_ISP_SetCCMSaturationAttr`：设置色彩饱和度属性参数
- `CVI_ISP_GetCCMSaturationAttr`：获取色彩饱和度属性参数
- `CVI_ISP_SetSaturationAttr`：设置色彩饱和度属性参数
- `CVI_ISP_GetSaturationAttr`：获取色彩饱和度属性参数

12.2.1 CVI_ISP_SetCCMAAttr

【描述】

设置色彩校正矩阵属性参数

【语法】

```
CVI_S32 CVI_ISP_SetCCMAAttr(VI_PIPE ViPipe, const ISP_CCM_ATTR_S *pstCCMAAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstCCMAAttr	色彩校正矩阵属性参数	输入

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

```
VI_PIPE ViPipe = 0;
ISP_CCM_ATTR_S stAttr;
CVI_ISP_GetCCMAAttr(ViPipe, &stAttr);
stAttr.enOpType = OP_TYPE_AUTO;
CVI_ISP_SetCCMAAttr(ViPipe, &stAttr);
```

【相关主题】

- [CVI_ISP_GetCCMAAttr](#)

12.2.2 CVI_ISP_GetCCMAAttr

【描述】

获取色彩校正矩阵属性参数

【语法】

```
CVI_S32 CVI_ISP_GetCCMAAttr(VI_PIPE ViPipe, ISP_CCM_ATTR_S *pstCCMAAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstCCMAAttr	色彩校正矩阵属性参数	输出

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

无

【相关主题】

· [CVI_ISP_SetCCMAAttr](#)

12.2.3 CVI_ISP_SetCCMSaturationAttr

【描述】

设置色彩饱和度属性参数

【语法】

```
CVI_S32 CVI_ISP_SetCCMSaturationAttr(VI_PIPE ViPipe, const ISP_CCM_SATURATION_
↪ATTR_S *pstCCMSaturationAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstCCMSaturationAttr	色彩饱和度属性参数	输入

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

```
VI_PIPE ViPipe = 0;
ISP_CCM_SATURATION_ATTR_S stAttr;
CVI_ISP_GetCCMSaturationAttr(ViPipe, &stAttr);
stAttr.enOpType = OP_TYPE_AUTO;
CVI_ISP_SetCCMSaturationAttr(ViPipe, &stAttr);
```

【相关主题】

- [CVI_ISP_GetCCMSaturationAttr](#)

12.2.4 CVI_ISP_GetCCMSaturationAttr

【描述】

获取色彩饱和度属性参数

【语法】

```
CVI_S32 CVI_ISP_GetCCMSaturationAttr(VI_PIPE ViPipe, ISP_CCM_SATURATION_ATTR_
↪ S *pstCCMSaturationAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstCCMSaturationAttr	色彩饱和度属性参数	输出

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

无

【相关主题】

- [CVI_ISP_SetCCMSaturationAttr](#)

12.2.5 CVI_ISP_SetSaturationAttr

【描述】

设置色彩饱和度属性参数

【语法】

```
CVI_S32 CVI_ISP_SetSaturationAttr(VI_PIPE ViPipe, const ISP_SATURATION_ATTR_S_
→*pstSaturationAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstSaturationAttr	色彩饱和度属性参数	输入

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: cvi_isp.h, cvi_comm_isp.h
- 库文件: libisp.so

【注意】

无

【举例】

```
VI_PIPE ViPipe = 0;
ISP_SATURATION_ATTR_S stAttr;
CVI_ISP_GetSaturationAttr(ViPipe, &stAttr);
stAttr.enOpType = OP_TYPE_AUTO;
CVI_ISP_SetSaturationAttr(ViPipe, &stAttr);
```

【相关主题】

- [CVI_ISP_GetSaturationAttr](#)

12.2.6 CVI_ISP_GetSaturationAttr

【描述】

获取色彩饱和度属性参数

【语法】

```
CVI_S32 CVI_ISP_GetSaturationAttr(VI_PIPE ViPipe, ISP_SATURATION_ATTR_S_
→*pstSaturationAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstSaturationAttr	色彩饱和度属性参数	输出

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: cvi_isp.h, cvi_comm_isp.h
- 库文件: libisp.so

【注意】

无

【举例】

无

【相关主题】

- CVI_ISP_SetSaturationAttr

12.3 数据类型

- ISP_CCM_ATTR_S：定义色彩校正矩阵属性参数
- ISP_CCM_MANUAL_ATTR_S：定义色彩校正矩阵属性手动参数
- ISP_CCM_AUTO_ATTR_S：定义色彩校正矩阵属性自动参数
- ISP_COLORMATRIX_ATTR_S：定义色彩校正矩阵
- ISP_CCM_SATURATION_MANUAL_ATTR_S：定义手动色彩饱和度属性参数
- ISP_CCM_SATURATION_AUTO_ATTR_S：定义自动色彩饱和度属性参数

- `ISP_CCM_SATURATION_ATTR_S`：定义色彩饱和度属性参数
- `ISP_SATURATION_MANUAL_ATTR_S`：定义手动色彩饱和度属性参数
- `ISP_SATURATION_AUTO_ATTR_S`：定义自动色彩饱和度属性参数
- `ISP_SATURATION_ATTR_S`：定义色彩饱和度属性参数

12.3.1 ISP_CCM_ATTR_S

【说明】

定义色彩校正矩阵属性参数

【定义】

```
typedef struct _ISP_CCM_ATTR_S {
    CVI_BOOL Enable; /*Rw; Range:[0, 1]*/
    ISP_OP_TYPE_E enOpType;
    CVI_U8 UpdateInterval; /*Rw; Range:[0, 255]*/
    CVI_U8 CCMOverStr; /*Rw; Range:[0, 255]*/
    CVI_U8 CCMOverThr; /*Rw; Range:[0, 255]*/
    ISP_CCM_MANUAL_ATTR_S stManual;
    ISP_CCM_AUTO_ATTR_S stAuto;
} ISP_CCM_ATTR_S;
```

【成员】

成员名称	描述
Enable	模块使能开关 取值范围：[0, 1] 数据类型：CVI_BOOL
enOpType	工作类型 OP_TYPE_AUTO：自动模式 OP_TYPE_MANUAL：手动模式 数据类型：ISP_OP_TYPE_E
UpdateInterval	影响参数更新间隔，值越大画面变化越慢，效能越好。 取值范围：[0, 255] 数据类型：CVI_U8
CCMOverStr	过饱和保护强度。 取值范围：[0, 255] 数据类型：CVI_U8
CCMOverThr	过饱和保护。过饱和判断阈值，值越大，越容易进入到过饱和保护状态。 取值范围：[0, 255] 数据类型：CVI_U8
stManual	手动参数 数据类型：ISP_CCM_MANUAL_ATTR_S
stAuto	自动参数 数据类型：ISP_CCM_AUTO_ATTR_S

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetCCMAAttr
- CVI_ISP_GetCCMAAttr

12.3.2 ISP_CCM_MANUAL_ATTR_S

【说明】

定义色彩校正矩阵属性手动参数

【定义】

```
typedef struct _ISP_CCM_MANUAL_ATTR_S {  
    CVI_BOOL SatEnable; /*Rw; Range:[0, 1]*/  
    CVI_S16 CCM[9]; /*Rw; Range:[-8192, 8191]*/  
} ISP_CCM_MANUAL_ATTR_S;
```

【成员】

成员名称	描述
SatEnable	手动模式下，饱和度是否生效。 取值范围：[0, 1] 数据类型：CVI_BOOL
CCM	3x3 色彩校正矩阵实际内容 取值范围：[-8192, 8191] 数据类型：CVI_S16

【注意事项】

无

【相关数据类型及接口】

12.3.3 ISP_CCM_AUTO_ATTR_S

【说明】

定义色彩校正矩阵属性自动参数

【定义】

```
typedef struct _ISP_CCM_AUTO_ATTR_S {  
    CVI_BOOL ISOActEnable; /*Rw; Range:[0, 1]*/  
    CVI_BOOL TempActEnable; /*Rw; Range:[0, 1]*/  
    CVI_U8 CCMTabNum; /*Rw; Range:[3, 7]*/  
    ISP_COLORMATRIX_ATTR_S CCMTab[7];  
} ISP_CCM_AUTO_ATTR_S;
```

【成员】

成员名称	描述
ISOActEnable	低照度下 CCM Bypass 功能使能。 0: 关闭。 1: 使能。 取值范围: [0, 1] 数据类型: CVI_BOOL
TempActEnable	高低色温下 CCM Bypass 功能使能。 0: 关闭。 1: 使能。 取值范围: [0, 1] 数据类型: CVI_BOOL
CCMTabNum	当前配置的 CCM 矩阵个数。 取值范围: [3, 7] 数据类型: CVI_U8
CCMTab	不同色温下的颜色校正矩阵 数据类型: ISP_COLORMATRIX_ATTR_S

【注意事项】

无

【相关数据类型及接口】

12.3.4 ISP_COLORMATRIX_ATTR_S

【说明】

定义色彩校正矩阵

【定义】

```
typedef struct _ISP_COLORMATRIX_ATTR_S {
    CVI_U16 ColorTemp; /*Rw; Range:[500, 30000]*/
    CVI_S16 CCM[9]; /*Rw; Range:[-8192, 8191]*/
} ISP_COLORMATRIX_ATTR_S;
```

【成员】

成员名称	描述
ColorTemp	此色彩校正矩阵适用的色温 取值范围: [500, 30000] 数据类型: CVI_U16
CCM	3x3 色彩校正矩阵实际内容 取值范围: [-8192, 8191] 数据类型: CVI_S16

【注意事项】

无

【相关数据类型及接口】

12.3.5 ISP_CCM_SATURATION_MANUAL_ATTR_S

【说明】

定义手动色彩饱和度属性参数

【定义】

```
typedef struct _ISP_CCM_SATURATION_MANUAL_ATTR_S {  
    CVI_U8 SaturationLE; /*Rw; Range:[0, 255]*/  
    CVI_U8 SaturationSE; /*Rw; Range:[0, 255]*/  
} ISP_CCM_SATURATION_MANUAL_ATTR_S;
```

【成员】

成员名称	描述
SaturationLE	长曝饱和度 取值范围：[0, 255] 数据类型：CVI_U8
SaturationSE	短曝饱和度 取值范围：[0, 255] 数据类型：CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetCCMSaturationAttr
- CVI_ISP_GetCCMSaturationAttr

12.3.6 ISP_CCM_SATURATION_AUTO_ATTR_S

【说明】

定义自动色彩饱和度属性参数

【定义】

```
typedef struct _ISP_CCM_SATURATION_AUTO_ATTR_S {  
    CVI_U8 SaturationLE[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/  
    CVI_U8 SaturationSE[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/  
} ISP_CCM_SATURATION_AUTO_ATTR_S;
```

【成员】

成员名称	描述
SaturationLE	长曝饱和度 取值范围：[0, 255] 数据类型：CVI_U8
SaturationSE	短曝饱和度 取值范围：[0, 255] 数据类型：CVI_U8

【注意事项】

无

【相关数据类型及接口】

- [CVI_ISP_SetCCMSaturationAttr](#)
- [CVI_ISP_GetCCMSaturationAttr](#)

12.3.7 ISP_CCM_SATURATION_ATTR_S

【说明】

定义色彩饱和度属性参数

【定义】

```
typedef struct _ISP_CCM_SATURATION_ATTR_S {  
    ISP_CCM_SATURATION_MANUAL_ATTR_S stManual;  
    ISP_CCM_SATURATION_AUTO_ATTR_S stAuto;  
} ISP_CCM_SATURATION_ATTR_S;
```

【成员】

成员名称	描述
stManual	手动参数
stAuto	自动参数

【注意事项】

无

【相关数据类型及接口】

- [CVI_ISP_SetCCMSaturationAttr](#)
- [CVI_ISP_GetCCMSaturationAttr](#)

12.3.8 ISP_SATURATION_MANUAL_ATTR_S

【说明】

定义手动色彩饱和度属性参数

【定义】

```
typedef struct _ISP_SATURATION_MANUAL_ATTR_S {  
    CVI_U8 Saturation; /*Rw; Range:[0, 255]*/  
} ISP_SATURATION_MANUAL_ATTR_S;
```

【成员】

成员名称	描述
Saturation	画面的整体饱和度 取值范围：[0, 255] 数据类型：CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetSaturationAttr
- CVI_ISP_GetSaturationAttr

12.3.9 ISP_SATURATION_AUTO_ATTR_S

【说明】

定义自动色彩饱和度属性参数

【定义】

```
typedef struct _ISP_SATURATION_AUTO_ATTR_S {  
    CVI_U8 Saturation[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/  
} ISP_SATURATION_AUTO_ATTR_S;
```

【成员】

成员名称	描述
Saturation	画面的整体饱和度 取值范围：[0, 255] 数据类型：CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetSaturationAttr
- CVI_ISP_GetSaturationAttr

12.3.10 ISP_SATURATION_ATTR_S

【说明】

定义色彩饱和度属性参数

【定义】

```
typedef struct _ISP_SATURATION_ATTR_S {  
    ISP_OP_TYPE_E enOpType;  
    ISP_SATURATION_MANUAL_ATTR_S stManual;  
    ISP_SATURATION_AUTO_ATTR_S stAuto;  
} ISP_SATURATION_ATTR_S;
```

【成员】

成员名称	描述
enOpType	工作类型 OP_TYPE_AUTO: 自动模式 OP_TYPE_MANUAL: 手动模式 数据类型: ISP_OP_TYPE_E
stManual	手动参数
stAuto	自动参数

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetSaturationAttr
- CVI_ISP_GetSaturationAttr

13 Noise profile

13.1 功能描述

记录各种 Sensor Gain 的噪声情形。一般由校正程序获得，不需特别调整。

【特别提醒】

在 cv184x 系列中，我们有提供 Noise Profile 相关接口，但实际上暂未使用 Noise profile 功能。

13.2 API 参考

- CVI_ISP_SetNoiseProfileAttr: 设置 Noise Profile 属性参数
- CVI_ISP_GetNoiseProfileAttr: 获取 Noise Profile 属性参数

13.2.1 CVI_ISP_SetNoiseProfileAttr

【描述】

设置 Noise Profile 属性参数

【语法】

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstNoiseProfileAttr	Noise Profile 属性参数	输入

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

无

【相关主题】

`CVI_ISP_GetNoiseProfileAttr`

13.2.2 CVI_ISP_GetNoiseProfileAttr

【描述】

获取 Noise Profile 属性参数

【语法】

```
CVI_S32 CVI_ISP_GetNoiseProfileAttr(VI_PIPE ViPipe, const ISP_CMOS_NOISE_
→CALIBRATION_S *pstNoiseProfileAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstNoiseProfileAttr	Noise Profile 属性参数	输出

【返回值】

返回值	描述
0	成功
非 0	失败, 其值为错误码

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

无

【相关主题】

`CVI_ISP_SetNoiseProfileAttr`

13.3 数据类型

· ISP_CMOS_NOISE_CALIBRATION_S: Noise Profile 属性参数

13.3.1 ISP_CMOS_NOISE_CALIBRATION_S

【说明】

Noise Profile 属性参数

【定义】

```
typedef struct cviISP_CMOS_NOISE_CALIBRATION_S {
    CVI_FLOAT CalibrationCoef[NOISE_PROFILE_ISO_NUM][NOISE_PROFILE_CHANNEL_
    ↪NUM][NOISE_PROFILE_LEVEL_NUM];
} ISP_CMOS_NOISE_CALIBRATION_S;
```

【成员】

成员名称	描述
CalibrationCoef [16][4][2]	Raw 数据噪声模型，分 R/Gr/Gb/B 信道，浮点数据形式 取值范围：[-2147483648,2147483647] 数据类型：CVI_FLOAT
NOISE_PROFILE_ISO_NUM	噪声配置文件的 ISO 设置数量
NOISE_PROFILE_CHANNEL_NUM	噪声配置文件通道数
NOISE_PROFILE_LEVEL_NUM	噪声分布级别的数量

【注意事项】

无

【相关数据类型及接口】

CVI_ISP_SetNoiseProfileAttr

CVI_ISP_GetNoiseProfileAttr

14 BNR

14.1 功能描述

在 Bayer Domain 进行的空域去噪算法。

14.2 API 参考

- `CVI_ISP_SetBNRAAttr`：设置 Bayer 降噪参数属性
- `CVI_ISP_GetBNRAAttr`：获取 Bayer 降噪参数属性
- `CVI_ISP_SetBNRFilterAttr`：设置 Bayer 降噪滤波器属性
- `CVI_ISP_GetBNRFilterAttr`：获取 Bayer 降噪滤波器属性

14.2.1 CVI_ISP_SetBNRAAttr

【描述】

设置 Bayer 降噪参数属性

【语法】

```
CVI_S32 CVI_ISP_SetBNRAAttr(VI_PIPE ViPipe, const ISP_BNR_ATTR_S *pstBNRAAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstBNRAAttr	Bayer 降噪参数属性	输入

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

```
VI_PIPE ViPipe = 0;
ISP_BNR_ATTR_S stAttr;
CVI_ISP_GetBNRAttr(ViPipe, &stAttr);
stAttr.enOpType = OP_TYPE_AUTO;
CVI_ISP_SetBNRAttr(ViPipe, &stAttr);
```

【相关主题】

- [CVI_ISP_GetBNRAttr](#)

14.2.2 CVI_ISP_GetBNRAttr

【描述】

获取 Bayer 降噪参数属性

【语法】

```
CVI_S32 CVI_ISP_GetBNRAttr(VI_PIPE ViPipe, ISP_BNR_ATTR_S *pstBNRAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstBNRAttr	Bayer 降噪参数属性	输出

【返回值】

返回值	描述
0	成功
非 0	失败, 其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

无

【相关主题】

- [CVI_ISP_SetBNRAAttr](#)

14.2.3 CVI_ISP_SetBNRFilterAttr

【描述】

设置 Bayer 降噪滤波器属性

【语法】

```
CVI_S32 CVI_ISP_SetBNRFilterAttr(VI_PIPE ViPipe, const ISP_BNR_FILTER_ATTR_S_
↪ *pstBNRFilterAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstBNRFilterAttr	Bayer 降噪滤波器属性	输入

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

```
VI_PIPE ViPipe = 0;
ISP_BNR_FILTER_ATTR_S stAttr;
CVI_ISP_GetBNRFilterAttr(ViPipe, &stAttr);
stAttr.enOpType = OP_TYPE_AUTO;
CVI_ISP_SetBNRFilterAttr(ViPipe, &stAttr);
```

【相关主题】

- [CVI_ISP_GetBNRFilterAttr](#)

14.2.4 CVI_ISP_GetBNRFilterAttr

【描述】

获取 Bayer 降噪滤波器属性

【语法】

```
CVI_S32 CVI_ISP_GetBNRFilterAttr(VI_PIPE ViPipe, ISP_BNR_FILTER_ATTR_S_
→*pstBNRFilterAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstBNRFilterAttr	Bayer 降噪滤波器属性	输出

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: cvi_isp.h, cvi_comm_isp.h
- 库文件: libisp.so

【注意】

无

【举例】

无

【相关主题】

- CVI_ISP_SetBNRFilterAttr

14.3 数据类型

- ISP_BNR_MANUAL_ATTR_S：定义手动 Bayer 降噪参数属性
- ISP_BNR_AUTO_ATTR_S：定义自动 Bayer 降噪参数属性
- ISP_BNR_ATTR_S：定义 Bayer 降噪参数属性
- ISP_BNR_FILTER_MANUAL_ATTR_S：定义手动 Bayer 降噪滤波器属性
- ISP_BNR_FILTER_AUTO_ATTR_S：定义自动 Bayer 降噪滤波器属性
- ISP_BNR_FILTER_ATTR_S：定义 Bayer 降噪滤波器属性

14.3.1 ISP_BNR_MANUAL_ATTR_S

【说明】

定义手动 Bayer 降噪参数属性

【定义】

```
typedef struct _ISP_BNR_MANUAL_ATTR_S {  
    CVI_U8 FilterKsize; /*Rw; Range:[0, 1]*/  
    CVI_U16 PreFilterStdThr1; /*Rw; Range:[0, 1023]*/  
    CVI_U16 PreFilterStdThr2; /*Rw; Range:[0, 1023]*/  
    CVI_U8 GussianKernelWt[BNR_WEIGHT_LUT_NUM]; /*Rw; Range:[0, 31]*/  
    CVI_U8 NrBlendWt; /*Rw; Range:[0, 16]*/  
} ISP_BNR_MANUAL_ATTR_S;
```

【成员】

成员名称	描述
FilterKsize	去噪滤波核大小。 取值范围：[0, 1] 数据类型：CVI_U8
PreFilterStdThr1	NR Weight TH1 in Pre filter BF。 取值范围：[0, 1023] 数据类型：CVI_U16
PreFilterStdThr2	NR Weight TH2 in Pre filter BF。 取值范围：[0, 1023] 数据类型：CVI_U16
GussianKernelWt	Spatial NR Weight LUT。 取值范围：[0, 31] 数据类型：CVI_U8
NrBlendWt	Blend Weight between NLMS&BF。 取值范围：[0, 16] 数据类型：CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetBNRAAttr
- CVI_ISP_GetBNRAAttr

14.3.2 ISP_BNR_AUTO_ATTR_S

【说明】

定义自动 Bayer 降噪参数属性

【定义】

```
typedef struct _ISP_BNR_AUTO_ATTR_S {  
    CVI_U8 FilterKsize[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 1]*/  
    CVI_U16 PreFilterStdThr1[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 1023]*/  
    CVI_U16 PreFilterStdThr2[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 1023]*/  
    CVI_U8 GussianKernelWt[BNR_WEIGHT_LUT_NUM][ISP_AUTO_ISO_STRENGTH_↪  
    NUM]; /*Rw; Range:[0, 31]*/  
    CVI_U8 NrBlendWt[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 16]*/  
} ISP_BNR_AUTO_ATTR_S;
```

【成员】

成员名称	描述
FilterKsize	去噪滤波核大小。 取值范围：[0, 1] 数据类型：CVI_U8
PreFilterStdThr1	NR Weight TH1 in Pre filter BF。 取值范围：[0, 1023] 数据类型：CVI_U16
PreFilterStdThr2	NR Weight TH2 in Pre filter BF。 取值范围：[0, 1023] 数据类型：CVI_U16
GussianKernelWt	Spatial NR Weight LUT。 取值范围：[0, 31] 数据类型：CVI_U8
NrBlendWt	Blend Weight between NLMS&BF。 取值范围：[0, 16] 数据类型：CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetBNRAAttr
- CVI_ISP_GetBNRAAttr

14.3.3 ISP_BNR_ATTR_S

【说明】

定义 Bayer 降噪参数属性

【定义】

```
typedef struct _ISP_BNR_ATTR_S {  
    CVI_BOOL Enable; /*Rw; Range:[0, 1]*/  
    ISP_OP_TYPE_E enOpType;  
    CVI_U8 UpdateInterval; /*Rw; Range:[0, 255]*/  
    CVI_BOOL PreFilterEnable; /*Rw; Range:[0, 1]*/  
    CVI_BOOL PreFilterModeSel; /*Rw; Range:[0, 1]*/  
    ISP_BNR_MANUAL_ATTR_S stManual;  
    ISP_BNR_AUTO_ATTR_S stAuto;  
} ISP_BNR_ATTR_S;
```

【成员】

成员名称	描述
Enable	模块使能开关 取值范围：[0, 1] 数据类型：CVI_BOOL
enOpType	工作类型 OP_TYPE_AUTO：自动模式 OP_TYPE_MANUAL：手动模式 数据类型：ISP_OP_TYPE_E
UpdateInterval	影响参数更新间隔，值越大画面变化越慢，效能越好。 取值范围：[0, 255] 数据类型：CVI_U8
PreFilterEnable	Pre filter BF 使能 取值范围：[0, 1] 数据类型：CVI_BOOL
PreFilterModeSel	Pre filter BF center mode sel when similar pixel cnt <= 2 取值范围：[0, 1] 数据类型：CVI_BOOL
stManual	手动参数
stAuto	自动参数

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetBNRAAttr
- CVI_ISP_GetBNRAAttr

14.3.4 ISP_BNR_FILTER_MANUAL_ATTR_S

【说明】

定义手动 Bayer 降噪滤波器属性

【定义】

```
typedef struct _ISP_BNR_FILTER_MANUAL_ATTR_S {
    CVI_U16 Filter1GaussianCurve[BNR_BASE_LUT_NUM]; /*Rw; Range:[0, 1023]*/
    CVI_U16 Filter1LumaOffset[BNR_LUMA_LUT_NUM]; /*Rw; Range:[0, 1023]*/
    CVI_U16 Filter2GaussianCurve[BNR_BASE_LUT_NUM]; /*Rw; Range:[0, 1023]*/
    CVI_U16 Filter2LumaOffset[BNR_LUMA_LUT_NUM]; /*Rw; Range:[0, 1023]*/
} ISP_BNR_FILTER_MANUAL_ATTR_S;
```

【成员】

成员名称	描述
Filter1GaussianCurve	NLMS base LUT, Gaussian。 取值范围：[0, 1023] 数据类型：CVI_U16
Filter1LumaOffset	NLMS Luma LUT, Signal independen。 取值范围：[0, 1023] 数据类型：CVI_U16
Filter2GaussianCurve	BF base LUT, Gaussian。 取值范围：[0, 1023] 数据类型：CVI_U16
Filter2LumaOffset	BF Luma LUT, Signal independen。 取值范围：[0, 1023] 数据类型：CVI_U16

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetBNRFilterAttr
- CVI_ISP_GetBNRFilterAttr

14.3.5 ISP_BNR_FILTER_AUTO_ATTR_S

【说明】

定义自动 Bayer 降噪滤波器属性

【定义】

```
typedef struct _ISP_BNR_FILTER_AUTO_ATTR_S {
    CVI_U16 Filter1GaussianCurve[BNR_BASE_LUT_NUM][ISP_AUTO_ISO_STRENGTH_
↪ NUM]; /*Rw; Range:[0, 1023]*/
```

(下页继续)

(续上页)

```

    CVI_U16 Filter1LumaOffset[BNR_LUMA_LUT_NUM][ISP_AUTO_ISO_STRENGTH_
↪NUM]; /*Rw; Range:[0, 1023]*/
    CVI_U16 Filter2GaussianCurve[BNR_BASE_LUT_NUM][ISP_AUTO_ISO_STRENGTH_
↪NUM]; /*Rw; Range:[0, 1023]*/
    CVI_U16 Filter2LumaOffset[BNR_LUMA_LUT_NUM][ISP_AUTO_ISO_STRENGTH_
↪NUM]; /*Rw; Range:[0, 1023]*/
} ISP_BNR_FILTER_AUTO_ATTR_S;

```

【成员】

成员名称	描述
Filter1GaussianCurve	NLMS base LUT, Gaussian。 取值范围：[0, 1023] 数据类型：CVI_U16
Filter1LumaOffset	NLMS Luma LUT, Signal independen。 取值范围：[0, 1023] 数据类型：CVI_U16
Filter2GaussianCurve	BF base LUT, Gaussian。 取值范围：[0, 1023] 数据类型：CVI_U16
Filter2LumaOffset	BF Luma LUT, Signal independen。 取值范围：[0, 1023] 数据类型：CVI_U16

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetBNRFilterAttr
- CVI_ISP_GetBNRFilterAttr

14.3.6 ISP_BNR_FILTER_ATTR_S

【说明】

定义 Bayer 降噪滤波器属性

【定义】

```

typedef struct _ISP_BNR_FILTER_ATTR_S {
    CVI_U8 DebugMode; /*Rw; Range:[0, 3]*/
    ISP_BNR_FILTER_MANUAL_ATTR_S stManual;
    ISP_BNR_FILTER_AUTO_ATTR_S stAuto;
} ISP_BNR_FILTER_ATTR_S;

```

【成员】

成员名称	描述
DebugMode	设置调试模式 取值范围：[0, 3] 数据类型：CVI__U8
stManual	手动参数
stAuto	自动参数

【注意事项】

无

【相关数据类型及接口】

- CVI__ISP_SetBNRFilterAttr
- CVI__ISP_GetBNRFilterAttr

15 CNR

15.1 功能描述

在 YUV Domain 针对色度噪声进行的空域去噪算法。cv184x 系列的 CNR 模块通过双边滤波处理对 UV 通道做整体的降噪处理。

15.2 API 参考

- `CVI_ISP_SetCNRAttr`：设置 CNR 参数属性
- `CVI_ISP_GetCNRAttr`：获取 CNR 参数属性
- `CVI_ISP_SetCNRFilterAttr`：设置 CNR FILTER 参数属性
- `CVI_ISP_GetCNRFilterAttr`：获取 CNR FILTER 参数属性

15.2.1 CVI_ISP_SetCNRAttr

【描述】

设置 CNR 参数属性

【语法】

```
CVI_S32 CVI_ISP_SetCNRAttr(VI_PIPE ViPipe, const ISP_CNR_ATTR_S *pstCNRAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstCNRAttr	CNR 参数属性	输入

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

```
VI_PIPE ViPipe = 0;
ISP_CNR_ATTR_S stAttr;
CVI_ISP_GetCNRAttr(ViPipe, &stAttr);
stAttr.enOpType = OP_TYPE_AUTO;
CVI_ISP_SetCNRAttr(ViPipe, &stAttr);
```

【相关主题】

- [CVI_ISP_GetCNRAttr](#)

15.2.2 CVI_ISP_GetCNRAttr

【描述】

获取 CNR 参数属性

【语法】

```
CVI_S32 CVI_ISP_GetCNRAttr(VI_PIPE ViPipe, ISP_CNR_ATTR_S *pstCNRAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstCNRAttr	CNR 参数属性	输出

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

无

【相关主题】

- [CVI_ISP_SetCNRAttr](#)

15.2.3 CVI_ISP_SetCNRFilterAttr

【描述】

设置 CNR FILTER 参数属性

【语法】

```
CVI_S32 CVI_ISP_SetCNRFilterAttr(VI_PIPE ViPipe, const ISP_CNR_FILTER_ATTR_S_
↪*pstCNRFilterAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstCNRFilterAttr	CNR FILTER 参数属性	输入

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

```
VI_PIPE ViPipe = 0;
ISP_CNR_FILTER_ATTR_S stAttr;
CVI_ISP_GetCNRFilterAttr(ViPipe, &stAttr);
stAttr.enOpType = OP_TYPE_AUTO;
CVI_ISP_SetCNRFilterAttr(ViPipe, &stAttr);
```

【相关主题】

- [CVI_ISP_GetCNRFilterAttr](#)

15.2.4 CVI_ISP_GetCNRFilterAttr

【描述】

获取 CNR FILTER 参数属性

【语法】

```
CVI_S32 CVI_ISP_GetCNRFilterAttr(VI_PIPE ViPipe, ISP_CNR_FILTER_ATTR_S_
→*pstCNRFilterAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstCNRFilterAttr	CNR FILTER 参数属性	输出

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: cvi_isp.h, cvi_comm_isp.h
- 库文件: libisp.so

【注意】

无

【举例】

无

【相关主题】

- CVI_ISP_SetCNRFilterAttr

15.3 数据类型

- ISP_CNR_MANUAL_ATTR_S：定义手动 CNR 参数属性
- ISP_CNR_AUTO_ATTR_S：定义自动 CNR 参数属性
- ISP_CNR_ATTR_S：定义 CNR 参数属性
- ISP_CNR_FILTER_MANUAL_ATTR_S：定义手动 CNR FILTER 参数属性
- ISP_CNR_FILTER_AUTO_ATTR_S：定义自动 CNR FILTER 参数属性
- ISP_CNR_FILTER_ATTR_S：定义 CNR FILTER 参数属性

15.3.1 ISP_CNR_MANUAL_ATTR_S

【说明】

定义手动 CNR 参数属性

【定义】

```
typedef struct _ISP_CNR_MANUAL_ATTR_S {
    CVI_U8 MedianFltKsize; /*Rw; Range:[0, 2]*/
    CVI_U8 CnrByMotion[3]; /*Rw; Range:[0, 31]*/
    CVI_U8 CnrRefYStr; /*Rw; Range:[0, 31]*/
    CVI_U8 CnrRefUVStr; /*Rw; Range:[0, 31]*/
    CVI_U8 CnrYRange; /*Rw; Range:[0, 3]*/
    CVI_U8 CnrYThr; /*Rw; Range:[0, 31]*/
    CVI_U8 CnrYPrcStr; /*Rw; Range:[0, 2]*/
    CVI_U8 CnrYStrH; /*Rw; Range:[0, 32]*/
    CVI_U8 CnrYStrT; /*Rw; Range:[0, 32]*/
    CVI_U8 CnrUVRange; /*Rw; Range:[0, 4]*/
    CVI_U8 CnrUVThr; /*Rw; Range:[0, 32]*/
    CVI_U8 CnrUVPrcStr; /*Rw; Range:[0, 3]*/
    CVI_U8 CnrUVStrH; /*Rw; Range:[0, 64]*/
    CVI_U8 CnrUVStrT; /*Rw; Range:[0, 64]*/
    CVI_U8 CnrSatRange; /*Rw; Range:[0, 3]*/
    CVI_U8 CnrSatPrtThr; /*Rw; Range:[0, 31]*/
    CVI_U8 CnrSatPrcStr; /*Rw; Range:[0, 2]*/
    CVI_U8 CnrSatStrH; /*Rw; Range:[0, 32]*/
    CVI_U8 CnrSatStrT; /*Rw; Range:[0, 32]*/
    CVI_U8 CnrSatBldCoring; /*Rw; Range:[0, 32]*/
} ISP_CNR_MANUAL_ATTR_S;
```

【成员】

成员名称	描述
MedianFltKsize	Filter size selection for CMF。 取值范围：[0, 2] 数据类型：CVI_U8
CnrByMotion	控制不同运动状态下的 CNR 强度。CnrByMotion[0]~CnrByMotion[2] 分别对应静止区、转换区、运动区。 取值范围：[0, 31] 数据类型：CVI_U8
CnrRefYStr	在 Y 通道上的色度参考权重。 取值范围：[0, 31] 数据类型：CVI_U8
CnrRefUVStr	在 UV 通道上的色度参考权重。 取值范围：[0, 31] 数据类型：CVI_U8
CnrYRange	在 Y 通道上的亮度差阈值。值越大，高亮度差阈值越大，CnrYStrH 对高亮度差区域的影响范围越大。 取值范围：[0, 3] 数据类型：CVI_U8

下页继续

表 15.5 – 续上页

成员名称	描述
CnrYThr	在 Y 通道上的亮度差阈值。亮度差低于 CnrYThr 的部分，CNR 强度由 CnrYStrH 控制。 取值范围：[0, 31] 数据类型：CVI_U8
CnrYPrcStr	Chroma adaptation weighting precision for Y channel。 0: 8; 1: 16; 2: 32; 取值范围：[0, 2] 数据类型：CVI_U8
CnrYStrH	在 Y 通道上控制亮度差异较小的区域的 CNR 强度。值越大，去噪强度越大。 取值范围：[0, 32] 数据类型：CVI_U8
CnrYStrT	在 Y 通道上控制亮度差异较大的区域的 CNR 强度。值越大，去噪强度越大。 取值范围：[0, 32] 数据类型：CVI_U8
CnrUVRange	在 UV 通道上控制高色差阈值。值越大，高色差阈值越大，CnrUVStrH 对高色差区域的影响范围越大。 0: 8; 1: 16; 2: 32; 3: 64; 4: 128; 7-5: Reserved; 取值范围：[0, 4] 数据类型：CVI_U8
CnrUVThr	在 UV 通道上控制高饱和阈值。值越大，高饱和阈值越大，CnrSatStrH 对高饱和区域的影响范围越大。 取值范围：[0, 32] 数据类型：CVI_U8
CnrUVPrsStr	Chroma adaptation weighting precision for UV channels。 0: 8; 1: 16; 2: 32; 3: 64; 取值范围：[0, 3] 数据类型：CVI_U8
CnrUVStrH	在 UV 通道上控制低色差区域的 CNR 强度。值越大，去噪强度越大。 取值范围：[0, 64] 数据类型：CVI_U8
CnrUVStrT	在 UV 通道上控制高色差区域的 CNR 强度。值越大，去噪强度越大。 取值范围：[0, 64] 数据类型：CVI_U8
CnrSatRange	控制高饱和阈值。值越大，高饱和阈值越大，CnrSatStrH 对高饱和区域的影响范围越大。 0: 8; 1: 16; 2: 32; 3: 64; 取值范围：[0, 3] 数据类型：CVI_U8
CnrSatPrtThr	低饱和度阈值。饱和度低于 CnrSatPrtThr 的部分，CNR 强度由 CnrSatStrH 控制。 取值范围：[0, 31] 数据类型：CVI_U8

下页继续

表 15.5 – 续上页

成员名称	描述
CnrSatPrcStr	Chroma adaptation weighting precision for Saturation。 0: 8; 1: 16; 2: 32; 取值范围: [0, 2] 数据类型: CVI_U8
CnrSatStrH	控制低饱和度区域的 CNR 强度。值越大, 去噪强度越大。 取值范围: [0, 32] 数据类型: CVI_U8
CnrSatStrT	控制高饱和度区域的 CNR 强度。值越大, 去噪强度越大。 取值范围: [0, 32] 数据类型: CVI_U8
CnrSatBldCoring	Chroma adaptation weighting for Saturation blending。 取值范围: [0, 32] 数据类型: CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetCNRAAttr
- CVI_ISP_GetCNRAAttr

15.3.2 ISP_CNR_AUTO_ATTR_S

【说明】

定义自动 CNR 参数属性

【定义】

```
typedef struct _ISP_CNR_AUTO_ATTR_S {
    CVI_U8 MedianFltKsize[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 2]*/
    CVI_U8 CnrByMotion[3][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 31]*/
    CVI_U8 CnrRefYStr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 31]*/
    CVI_U8 CnrRefUVStr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 31]*/
    CVI_U8 CnrYRange[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 3]*/
    CVI_U8 CnrYThr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 31]*/
    CVI_U8 CnrYPrcStr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 2]*/
    CVI_U8 CnrYStrH[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 32]*/
    CVI_U8 CnrYStrT[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 32]*/
    CVI_U8 CnrUVRange[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 4]*/
    CVI_U8 CnrUVThr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 32]*/
    CVI_U8 CnrUVPrcStr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 3]*/
    CVI_U8 CnrUVStrH[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 64]*/
    CVI_U8 CnrUVStrT[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 64]*/
    CVI_U8 CnrSatRange[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 3]*/
    CVI_U8 CnrSatPrtThr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 31]*/
    CVI_U8 CnrSatPrcStr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 2]*/
    CVI_U8 CnrSatStrH[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 32]*/
}
```

(下页继续)

(续上页)

```

    CVI_U8 CnrSatStrT[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 32]*/
    CVI_U8 CnrSatBldCoring[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 32]*/
} ISP_CNR_AUTO_ATTR_S;

```

【成员】

成员名称	描述
MedianFltKsize	Filter size selection for CMF。 取值范围：[0, 2] 数据类型：CVI_U8
CnrByMotion	控制不同运动状态下的 CNR 强度。CnrByMotion[0]~CnrByMotion[2] 分别对应静止区、转换区、运动区。 取值范围：[0, 31] 数据类型：CVI_U8
CnrRefYStr	在 Y 通道上的色度参考权重。 取值范围：[0, 31] 数据类型：CVI_U8
CnrRefUVStr	在 UV 通道上的色度参考权重。 取值范围：[0, 31] 数据类型：CVI_U8
CnrYRange	在 Y 通道上的亮度差阈值。值越大，高亮度差阈值越大，CnrYStrH 对高亮度差区域的影响范围越大。 取值范围：[0, 3] 数据类型：CVI_U8
CnrYThr	在 Y 通道上的亮度差阈值。亮度差低于 CnrYThr 的部分，CNR 强度由 CnrYStrH 控制。 取值范围：[0, 31] 数据类型：CVI_U8
CnrYPrcStr	Chroma adaptation weighting precision for Y channel。 0: 8; 1: 16; 2: 32; 取值范围：[0, 2] 数据类型：CVI_U8
CnrYStrH	在 Y 通道上控制亮度差异较小的区域的 CNR 强度。值越大，去噪强度越大。 取值范围：[0, 32] 数据类型：CVI_U8
CnrYStrT	在 Y 通道上控制亮度差异较大的区域的 CNR 强度。值越大，去噪强度越大。 取值范围：[0, 32] 数据类型：CVI_U8
CnrUVRange	在 UV 通道上控制高色差阈值。值越大，高色差阈值越大，CnrUVStrH 对高色差区域的影响范围越大。 0: 8; 1: 16; 2: 32; 3: 64; 4: 128; 7-5: Reserved; 取值范围：[0, 4] 数据类型：CVI_U8

下页继续

表 15.6 – 续上页

成员名称	描述
CnrUVThr	在 UV 通道上控制高饱和阈值。值越大，高饱和阈值越大，CnrSatStrH 对高饱和区域的影响范围越大。 取值范围：[0, 32] 数据类型：CVI_U8
CnrUVPrsStr	Chroma adaptation weighting precision for UV channels。 0: 8; 1: 16; 2: 32; 3: 64; 取值范围：[0, 3] 数据类型：CVI_U8
CnrUVStrH	在 UV 通道上控制低色差区域的 CNR 强度。值越大，去噪强度越大。 取值范围：[0, 64] 数据类型：CVI_U8
CnrUVStrT	在 UV 通道上控制高色差区域的 CNR 强度。值越大，去噪强度越大。 取值范围：[0, 64] 数据类型：CVI_U8
CnrSatRange	控制高饱和阈值。值越大，高饱和阈值越大，CnrSatStrH 对高饱和区域的影响范围越大。 0: 8; 1: 16; 2: 32; 3: 64; 取值范围：[0, 3] 数据类型：CVI_U8
CnrSatPrtThr	低饱和度阈值。饱和度低于 CnrSatPrtThr 的部分，CNR 强度由 CnrSatStrH 控制。 取值范围：[0, 31] 数据类型：CVI_U8
CnrSatPrsStr	Chroma adaptation weighting precision for Saturation。 0: 8; 1: 16; 2: 32; 取值范围：[0, 2] 数据类型：CVI_U8
CnrSatStrH	控制低饱和度区域的 CNR 强度。值越大，去噪强度越大。 取值范围：[0, 32] 数据类型：CVI_U8
CnrSatStrT	控制高饱和度区域的 CNR 强度。值越大，去噪强度越大。 取值范围：[0, 32] 数据类型：CVI_U8
CnrSatBldCoring	Chroma adaptation weighting for Saturation blending。 取值范围：[0, 32] 数据类型：CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetCNRAAttr
- CVI_ISP_GetCNRAAttr

15.3.3 ISP_CNR_ATTR_S

【说明】

定义 CNR 参数属性

【定义】

```
typedef struct _ISP_CNR_ATTR_S {  
    CVI_BOOL Enable; /*Rw; Range:[0, 1]*/  
    ISP_OP_TYPE_E enOpType;  
    CVI_U8 UpdateInterval; /*Rw; Range:[1, 255]*/  
    CVI_BOOL MedianFltEnable; /*Rw; Range:[0, 1]*/  
    CVI_BOOL CnrEnable; /*Rw; Range:[0, 1]*/  
    CVI_BOOL CnrSatBldEnable; /*Rw; Range:[0, 1]*/  
    CVI_U8 SubScI Sel; /*Rw; Range:[0, 1]*/  
    CVI_U8 CnrDebugMode; /*Rw; Range:[0, 6]*/  
    CVI_BOOL CnrBypass1; /*Rw; Range:[0, 1]*/  
    CVI_BOOL CnrBypass2; /*Rw; Range:[0, 1]*/  
    ISP_CNR_SCALE_FACTOR_E SubImgScI Factor;  
    ISP_CNR_MANUAL_ATTR_S stManual;  
    ISP_CNR_AUTO_ATTR_S stAuto;  
} ISP_CNR_ATTR_S;
```

【成员】

成员名称	描述
Enable	模块使能开关 取值范围：[0, 1] 数据类型：CVI_BOOL
enOpType	工作类型 OP_TYPE_AUTO：自动模式 OP_TYPE_MANUAL：手动模式 数据类型：ISP_OP_TYPE_E
UpdateInterval	影响参数更新间隔，值越大画面变化越慢，效能越好。 取值范围：[1, 255] 数据类型：CVI_U8
MedianFltEnable	UV 通道预滤波处理使能。Enable = 1 时生效。 取值范围：[0, 1] 数据类型：CVI_BOOL
CnrEnable	CNR 模块使能 取值范围：[0, 1] 数据类型：CVI_BOOL
CnrSatBldEnable	enable sat blend 取值范围：[0, 1] 数据类型：CVI_BOOL
SubScI Sel	Selection sub-image output flow: 0: output sub image after LCA; 1: output sub image before CMF。 取值范围：[0, 1] 数据类型：CVI_U8

下页继续

表 15.7 – 续上页

成员名称	描述
CnrDebugMode	Register debug mode 0: no debug, normal output; 1: debug output wY, wUV, w; 2: debug output abs distU, distV, distUV; 3: debug output sat weight; 4: debug output SAT blend out; 5: debug output cnr motion; 6: debug output subimg sclupimg; 取值范围: [0, 6] 数据类型: CVI_U8
CnrBypass1	Set 1 to bypass chra(LCA) 取值范围: [0, 1] 数据类型: CVI_BOOL
CnrBypass2	根据亮度差、饱和度、色差、运动状态调整 CNR 强度的功能使能。 取值范围: [0, 1] 数据类型: CVI_BOOL
SubImgSclFactor	Sub image scaling factor 数据类型: ISP_CNR_SCALE_FACTOR_E
stManual	手动参数
stAuto	自动参数

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetCNRAAttr
- CVI_ISP_GetCNRAAttr

15.3.4 ISP_CNR_FILTER_MANUAL_ATTR_S

【说明】

定义手动 CNR FILTER 参数属性

【定义】

```
typedef struct _ISP_CNR_FILTER_MANUAL_ATTR_S {
    CVI_U8 SubImgNrFltKsize; /*Rw; Range:[0, 3]*/
    CVI_U8 SubImgNrEdgeDirPN; /*Rw; Range:[0, 255]*/
    CVI_U8 SubImgNrEdgeDirHV; /*Rw; Range:[0, 255]*/
    CVI_U8 SubImgNrEdgeKsize; /*Rw; Range:[0, 1]*/
    CVI_U8 SubImgNrYCCRef; /*Rw; Range:[0, 31]*/
    CVI_U8 SubImgNrUVCCRef; /*Rw; Range:[0, 31]*/
    CVI_U8 OutierThY; /*Rw; Range:[0, 7]*/
    CVI_U8 OutierThUV; /*Rw; Range:[0, 7]*/
    CVI_U8 YOutierDifTh; /*Rw; Range:[0, 255]*/
    CVI_U8 UOutierDifTh; /*Rw; Range:[0, 255]*/
    CVI_U8 VOutierDifTh; /*Rw; Range:[0, 255]*/
}
```

(下页继续)

(续上页)

```

CVI_U8 SubImgNrRefCThrYDif[3]; /*Rw; Range:[0, 255]*/
CVI_U8 SubImgNrRefCStrY[4]; /*Rw; Range:[0, 8]*/
CVI_U8 SubImgNrRefCThrUVDif[3]; /*Rw; Range:[0, 255]*/
CVI_U8 SubImgNrRefCStrUV[4]; /*Rw; Range:[0, 8]*/
CVI_U8 SubImgNrDirFltThrY[5]; /*Rw; Range:[0, 255]*/
CVI_U8 SubImgNrDirFltStrY[6]; /*Rw; Range:[0, 16]*/
CVI_U8 SubImgNrDirFltThrU[5]; /*Rw; Range:[0, 255]*/
CVI_U8 SubImgNrDirFltStrU[6]; /*Rw; Range:[0, 16]*/
CVI_U8 SubImgNrDirFltThrV[5]; /*Rw; Range:[0, 255]*/
CVI_U8 SubImgNrDirFltStrV[6]; /*Rw; Range:[0, 16]*/
} ISP_CNR_FILTER_MANUAL_ATTR_S;

```

【成员】

成员名称	描述
SubImgNrFltKsize	Filter size selection for Y/UV channels 0: 3x3; 1: 5x5; 2: 7x7; 3: 9x9; 取值范围: [0, 3] 数据类型: CVI_U8
SubImgNrEdgeDirPN	正 (负)45 度方向判断阈值。值越大, 越容易判断为无方向。值过大, 容易造成边缘渗色问题。 取值范围: [0, 255] 数据类型: CVI_U8
SubImgNrEdgeDirHV	水平/垂直方向判断阈值。值越大, 越容易判断为无方向。值过大, 容易造成边缘渗色问题。 取值范围: [0, 255] 数据类型: CVI_U8
SubImgNrEdgeKsize	Edge kernel size selection。 0: 3x3; 1: 5x5; 取值范围: [0, 1] 数据类型: CVI_U8
SubImgNrYCRef	Reference center weighting for Y channel 取值范围: [0, 31] 数据类型: CVI_U8
SubImgNrUVCTRef	Reference center weighting for UV channel 取值范围: [0, 31] 数据类型: CVI_U8
OutierThY	Reference center outlier threshold of Y channel 取值范围: [0, 7] 数据类型: CVI_U8
OutierThUV	Reference center outlier threshold of UV channel 取值范围: [0, 7] 数据类型: CVI_U8
YOutierDifTh	Reference center outlier difference threshold for Y channel 取值范围: [0, 255] 数据类型: CVI_U8
UOutierDifTh	Reference center outlier difference threshold for U channel 取值范围: [0, 255] 数据类型: CVI_U8

下页继续

表 15.8 – 续上页

成员名称	描述
VOutierDifTh	Reference center outlier difference threshold for V channel 取值范围: [0, 255] 数据类型: CVI_U8
SubImgNrRefCThrYDif	Range filter threshold for reference center of Y channel 取值范围: [0, 255] 数据类型: CVI_U8
SubImgNrRefCStrY	Range filter weighting for 0x0] reference center of Y channel 取值范围: [0, 8] 数据类型: CVI_U8
SubImgNrRefCThrUVDif	Range filter threshold for reference center of UV channel 取值范围: [0, 255] 数据类型: CVI_U8
SubImgNrRefCStrUV	Range filter weighting for 0x0] reference center of UV channel 取值范围: [0, 8] 数据类型: CVI_U8
SubImgNrDirFltThrY	Filter threshold for Y channel 取值范围: [0, 255] 数据类型: CVI_U8
SubImgNrDirFltStrY	Filter weighting for Y channel 取值范围: [0, 16] 数据类型: CVI_U8
SubImgNrDirFltThrU	Filter threshold for U channel, 和 SubImgNrDirFltStrU 共同控制 U 通道的降噪强度 取值范围: [0, 255] 数据类型: CVI_U8
SubImgNrDirFltStrU	Filter threshold for U channel, 和 SubImgNrDirFltThrU 共同控制 U 通道的降噪强度 取值范围: [0, 16] 数据类型: CVI_U8
SubImgNrDirFltThrV	Filter threshold for V channel, 和 SubImgNrDirFltStrV 共同控制 V 通道的降噪强度 取值范围: [0, 255] 数据类型: CVI_U8
SubImgNrDirFltStrV	Filter threshold for V channel, 和 SubImgNrDirFltThrV 共同控制 V 通道的降噪强度 取值范围: [0, 16] 数据类型: CVI_U8

【注意事项】

无

【相关数据类型及接口】

- [CVI_ISP_SetCNRFilterAttr](#)
- [CVI_ISP_GetCNRFilterAttr](#)

15.3.5 ISP_CNR_FILTER_AUTO_ATTR_S

【说明】

定义自动 CNR FILTER 参数属性

【定义】

```
typedef struct _ISP_CNR_FILTER_AUTO_ATTR_S {
    CVI_U8 SubImgNrFltKsize[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 3]*/
    CVI_U8 SubImgNrEdgeDirPN[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U8 SubImgNrEdgeDirHV[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U8 SubImgNrEdgeKsize[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 1]*/
    CVI_U8 SubImgNrYCCRef[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 31]*/
    CVI_U8 SubImgNrUVCCRef[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 31]*/
    CVI_U8 OutierThY[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 7]*/
    CVI_U8 OutierThUV[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 7]*/
    CVI_U8 YOutierDifTh[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U8 UOutierDifTh[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U8 VOutierDifTh[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U8 SubImgNrRefCThrYDif[3][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U8 SubImgNrRefCStrY[4][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 8]*/
    CVI_U8 SubImgNrRefCThrUVDif[3][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U8 SubImgNrRefCStrUV[4][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 8]*/
    CVI_U8 SubImgNrDirFltThrY[5][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U8 SubImgNrDirFltStrY[6][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 16]*/
    CVI_U8 SubImgNrDirFltThrU[5][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U8 SubImgNrDirFltStrU[6][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 16]*/
    CVI_U8 SubImgNrDirFltThrV[5][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U8 SubImgNrDirFltStrV[6][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 16]*/
} ISP_CNR_FILTER_AUTO_ATTR_S;
```

【成员】

成员名称	描述
SubImgNrFltKsize	Filter size selection for Y/UV channels 0: 3x3; 1: 5x5; 2: 7x7; 3: 9x9; 取值范围: [0, 3] 数据类型: CVI_U8
SubImgNrEdgeDirPN	正 (负)45 度方向判断阈值。值越大, 越容易判断为无方向。值过大, 容易造成边缘渗色问题。 取值范围: [0, 255] 数据类型: CVI_U8
SubImgNrEdgeDirHV	水平/垂直方向判断阈值。值越大, 越容易判断为无方向。值过大, 容易造成边缘渗色问题。 取值范围: [0, 255] 数据类型: CVI_U8

下页继续

表 15.9 – 续上页

成员名称	描述
SubImgNrEdgeKsize	Edge kernel size selection。 0: 3x3; 1: 5x5; 取值范围: [0, 1] 数据类型: CVI_U8
SubImgNrYCRef	Reference center weighting for Y channel 取值范围: [0, 31] 数据类型: CVI_U8
SubImgNrUVCRef	Reference center weighting for UV channel 取值范围: [0, 31] 数据类型: CVI_U8
OutierThY	Reference center outlier threshold of Y channel 取值范围: [0, 7] 数据类型: CVI_U8
OutierThUV	Reference center outlier threshold of UV channel 取值范围: [0, 7] 数据类型: CVI_U8
YOutierDifTh	Reference center outlier difference threshold for Y channel 取值范围: [0, 255] 数据类型: CVI_U8
UOutierDifTh	Reference center outlier difference threshold for U channel 取值范围: [0, 255] 数据类型: CVI_U8
VOutierDifTh	Reference center outlier difference threshold for V channel 取值范围: [0, 255] 数据类型: CVI_U8
SubImgNrRefCThrYDif	Range filter threshold for reference center of Y channel 取值范围: [0, 255] 数据类型: CVI_U8
SubImgNrRefCStrY	Range filter weighting for 0x0] reference center of Y channel 取值范围: [0, 8] 数据类型: CVI_U8
SubImgNrRefCThrUVDif	Range filter threshold for reference center of UV channel 取值范围: [0, 255] 数据类型: CVI_U8
SubImgNrRefCStrUV	Range filter weighting for 0x0] reference center of UV channel 取值范围: [0, 8] 数据类型: CVI_U8
SubImgNrDirFltThrY	Filter threshold for Y channel 取值范围: [0, 255] 数据类型: CVI_U8
SubImgNrDirFltStrY	Filter weighting for Y channel 取值范围: [0, 16] 数据类型: CVI_U8
SubImgNrDirFltThrU	Filter threshold for U channel, 和 SubImgNrDirFltStrU 共同控制 U 通道的降噪强度 取值范围: [0, 255] 数据类型: CVI_U8

下页继续

表 15.9 – 续上页

成员名称	描述
SubImgNrDirFltStrU	Filter threshold for U channel, 和 SubImgNrDirFltThrU 共同控制 U 通道的降噪强度 取值范围: [0, 16] 数据类型: CVI_U8
SubImgNrDirFltThrV	Filter threshold for V channel, 和 SubImgNrDirFltStrV 共同控制 V 通道的降噪强度 取值范围: [0, 255] 数据类型: CVI_U8
SubImgNrDirFltStrV	Filter threshold for V channel, 和 SubImgNrDirFltThrV 共同控制 V 通道的降噪强度 取值范围: [0, 16] 数据类型: CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetCNRFilterAttr
- CVI_ISP_GetCNRFilterAttr

15.3.6 ISP_CNR_FILTER_ATTR_S

【说明】

定义 CNR FILTER 参数属性

【定义】

```
typedef struct _ISP_CNR_FILTER_ATTR_S {
    CVI_BOOL SubImgNrEnable; /*Rw; Range:[0, 1]*/
    CVI_BOOL YNrEnable; /*Rw; Range:[0, 1]*/
    CVI_BOOL EdgeDebugEnable; /*Rw; Range:[0, 1]*/
    CVI_BOOL RefCenterDebugEnable; /*Rw; Range:[0, 1]*/
    ISP_CNR_FILTER_MANUAL_ATTR_S stManual;
    ISP_CNR_FILTER_AUTO_ATTR_S stAuto;
} ISP_CNR_FILTER_ATTR_S;
```

【成员】

成员名称	描述
SubImgNrEnable	Set 1 to enable filter for ife2 sub image。 取值范围: [0, 1] 数据类型: CVI_BOOL
YNrEnable	IFE2 enable key for Y。 取值范围: [0, 1] 数据类型: CVI_BOOL

下页继续

表 15.10 – 续上页

成员名称	描述
EdgeDebugEnabled	Debug key for Edge map in IFE2。 取值范围：[0, 1] 数据类型：CVI_BOOL
RefCenterDebugEnabled	Debug key for Ref center filter in IFE2。 取值范围：[0, 1] 数据类型：CVI_BOOL
stManual	手动参数
stAuto	自动参数

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetCNRFilterAttr
- CVI_ISP_GetCNRFilterAttr

16 TNR

16.1 功能描述

在 YUV Domain 进行的时域去噪算法，又称为 3DNR。

16.2 API 参考

- `CVI_ISP_SetTNRAttr`：设置 TNR 属性参数
- `CVI_ISP_GetTNRAttr`：获取 TNR 属性参数
- `CVI_ISP_SetTNRMVAttr`：设置 TNR MV 属性参数
- `CVI_ISP_GetTNRMVAttr`：获取 TNR MV 属性参数
- `CVI_ISP_SetTNRPSAttr`：设置 TNR PS 属性参数
- `CVI_ISP_GetTNRPSAttr`：获取 TNR PS 属性参数
- `CVI_ISP_SetTNRNRAttr`：设置 TNR NR 属性参数
- `CVI_ISP_GetTNRNRAttr`：获取 TNR NR 属性参数

16.2.1 CVI_ISP_SetTNRAttr

【描述】

设置 TNR 属性参数

【语法】

```
CVI_S32 CVI_ISP_SetTNRAttr(VI_PIPE ViPipe, const ISP_TNR_ATTR_S *pstTNRAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstTNRAttr	TNR 属性参数	输入

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

```
VI_PIPE ViPipe = 0;
ISP_TNR_ATTR_S stAttr;
CVI_ISP_GetTNRAttr(ViPipe, &stAttr);
stAttr.enOpType = OP_TYPE_AUTO;
CVI_ISP_SetTNRAttr(ViPipe, &stAttr);
```

【相关主题】

- [CVI_ISP_GetTNRAttr](#)

16.2.2 CVI_ISP_GetTNRAttr

【描述】

获取 TNR 属性参数

【语法】

```
CVI_S32 CVI_ISP_GetTNRAttr(VI_PIPE ViPipe, ISP_TNR_ATTR_S *pstTNRAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstTNRAttr	TNR 属性参数	输出

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`

· 库文件: libisp.so

【注意】

无

【举例】

无

【相关主题】

· [CVI_ISP_SetTNRAttr](#)

16.2.3 CVI_ISP_SetTNRMvAttr

【描述】

设置 TNR MV 属性参数

【语法】

```
CVI_S32 CVI_ISP_SetTNRMvAttr(VI_PIPE ViPipe, const ISP_TNR_MV_ATTR_S_
↪ *pstTNRMvAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstTNRMvAttr	TNR MV 属性参数	输入

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

· 头文件: cvi_isp.h, cvi_comm_isp.h
· 库文件: libisp.so

【注意】

无

【举例】

```
VI_PIPE ViPipe = 0;
ISP_TNR_MV_ATTR_S stAttr;
CVI_ISP_GetTNRMvAttr(ViPipe, &stAttr);
stAttr.enOpType = OP_TYPE_AUTO;
CVI_ISP_SetTNRMvAttr(ViPipe, &stAttr);
```

【相关主题】

- [CVI_ISP_GetTNRMvAttr](#)

16.2.4 CVI_ISP_GetTNRMvAttr

【描述】

获取 TNR MV 属性参数

【语法】

```
CVI_S32 CVI_ISP_GetTNRMvAttr(VI_PIPE ViPipe, ISP_TNR_MV_ATTR_S *pstTNRMvAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstTNRMvAttr	TNR MV 属性参数	输出

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

无

【相关主题】

- [CVI_ISP_SetTNRMvAttr](#)

16.2.5 CVI_ISP_SetTNRPsAttr

【描述】

设置 TNR PS 属性参数

【语法】

```
CVI_S32 CVI_ISP_SetTNRPsAttr(VI_PIPE ViPipe, const ISP_TNR_PS_ATTR_S  
↪ *pstTNRPsAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstTNRPsAttr	TNR PS 属性参数	输入

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

```
VI_PIPE ViPipe = 0;
ISP_TNR_PS_ATTR_S stAttr;
CVI_ISP_GetTNRPsAttr(ViPipe, &stAttr);
stAttr.enOpType = OP_TYPE_AUTO;
CVI_ISP_SetTNRPsAttr(ViPipe, &stAttr);
```

【相关主题】

- [CVI_ISP_GetTNRPsAttr](#)

16.2.6 CVI_ISP_GetTNRPsAttr

【描述】

获取 TNR PS 属性参数

【语法】

```
CVI_S32 CVI_ISP_GetTNRPsAttr(VI_PIPE ViPipe, ISP_TNR_PS_ATTR_S *pstTNRPsAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstTNRPsAttr	TNR PS 属性参数	输出

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

无

【相关主题】

- [CVI_ISP_SetTNRPsAttr](#)

16.2.7 CVI_ISP_SetTNRNrAttr

【描述】

设置 TNR NR 属性参数

【语法】

```
CVI_S32 CVI_ISP_SetTNRNrAttr(VI_PIPE ViPipe, const ISP_TNR_NR_ATTR_S_
↪ *pstTNRNrAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstTNRNrAttr	TNR NR 属性参数	输入

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

```

VI_PIPE ViPipe = 0;
ISP_TNR_NR_ATTR_S stAttr;
CVI_ISP_GetTNRNrAttr(ViPipe, &stAttr);
stAttr.enOpType = OP_TYPE_AUTO;
CVI_ISP_SetTNRNrAttr(ViPipe, &stAttr);

```

【相关主题】

- [CVI_ISP_GetTNRNrAttr](#)

16.2.8 CVI_ISP_GetTNRNrAttr

【描述】

获取 TNR NR 属性参数

【语法】

```

CVI_S32 CVI_ISP_GetTNRNrAttr(VI_PIPE ViPipe, ISP_TNR_NR_ATTR_S *pstTNRNrAttr);

```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstTNRNrAttr	TNR NR 属性参数	输出

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

无

【相关主题】

- [CVI_ISP_SetTNRNrAttr](#)

16.3 数据类型

- `ISP_TNR_MANUAL_ATTR_S`：定义手动 TNR 属性参数
- `ISP_TNR_AUTO_ATTR_S`：定义自动 TNR 属性参数
- `ISP_TNR_ATTR_S`：定义 TNR 属性参数
- `ISP_TNR_MV_MANUAL_ATTR_S`：定义手动 TNR MV 属性参数
- `ISP_TNR_MV_AUTO_ATTR_S`：定义自动 TNR MV 属性参数
- `ISP_TNR_MV_ATTR_S`：定义 TNR MV 属性参数
- `ISP_TNR_PS_MANUAL_ATTR_S`：定义手动 TNR PS 属性参数
- `ISP_TNR_PS_AUTO_ATTR_S`：定义自动 TNR PS 属性参数
- `ISP_TNR_PS_ATTR_S`：定义 TNR PS 属性参数
- `ISP_TNR_NR_MANUAL_ATTR_S`：定义手动 TNR NR 属性参数
- `ISP_TNR_NR_AUTO_ATTR_S`：定义自动 TNR NR 属性参数
- `ISP_TNR_NR_ATTR_S`：定义 TNR NR 属性参数

16.3.1 `ISP_TNR_MANUAL_ATTR_S`

【说明】

定义手动 TNR 属性参数

【定义】

```
typedef struct _ISP_TNR_MANUAL_ATTR_S {  
    CVI_U8 DyBlurStr; /*Rw; Range:[0, 3]*/  
    CVI_U8 DyBlurYWt[TNR_STATUS_NUM]; /*Rw; Range:[0, 15]*/  
} ISP_TNR_MANUAL_ATTR_S;
```

【成员】

成员名称	描述
DyBlurStr	None 取值范围：[0, 3] 数据类型：CVI_U8
DyBlurYWt	调节 Y 通道时域滤波强度 取值范围：[0, 15] 数据类型：CVI_U8

【注意事项】

无

【相关数据类型及接口】

- `CVI_ISP_SetTNRAttr`

- CVI_ISP_GetTNRAttr

16.3.2 ISP_TNR_AUTO_ATTR_S

【说明】

定义自动 TNR 属性参数

【定义】

```
typedef struct _ISP_TNR_AUTO_ATTR_S {  
    CVI_U8 DyBlurStr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 3]*/  
    CVI_U8 DyBlurYWt[TNR_STATUS_NUM][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 15]*/  
} ISP_TNR_AUTO_ATTR_S;
```

【成员】

成员名称	描述
DyBlurStr	None 取值范围：[0, 3] 数据类型：CVI_U8
DyBlurYWt	调节 Y 通道时域滤波强度 取值范围：[0, 15] 数据类型：CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetTNRAttr
- CVI_ISP_GetTNRAttr

16.3.3 ISP_TNR_ATTR_S

【说明】

定义 TNR 属性参数

【定义】

```
typedef struct _ISP_TNR_ATTR_S {  
    CVI_BOOL Enable; /*Rw; Range:[0, 1]*/  
    ISP_OP_TYPE_E enOpType;  
    CVI_U8 updateInterval; /*Rw; Range:[0, 255]*/  
    CVI_U8 DbgMode; /*Rw; Range:[0, 8]*/  
    CVI_U8 DyBlurUVWt; /*Rw; Range:[0, 15]*/  
    CVI_U8 DyBlurJndTh; /*Rw; Range:[0, 31]*/  
    ISP_TNR_MANUAL_ATTR_S stManual;
```

(下页继续)

(续上页)

```

    ISP_TNR_AUTO_ATTR_S stAuto;
} ISP_TNR_ATTR_S;

```

【成员】

成员名称	描述
Enable	模块使能开关 取值范围：[0, 1] 数据类型：CVI_BOOL
enOpType	工作类型 OP_TYPE_AUTO：自动模式 OP_TYPE_MANUAL：手动模式 数据类型：ISP_OP_TYPE_E
updateInterval	影响参数更新间隔，值越大画面变化越慢，效能越好。 取值范围：[0, 255] 数据类型：CVI_U8
DbgMode	设置 debug 模式，用于输出可视化辅助信息。 取值范围：[0, 8] 数据类型：CVI_U8
DyBlurUVWt	调节 UV 通道时域滤波强度 取值范围：[0, 15] 数据类型：CVI_U8
DyBlurJndTh	None 取值范围：[0, 31] 数据类型：CVI_U8
stManual	手动参数
stAuto	自动参数

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetTNRAttr
- CVI_ISP_GetTNRAttr

16.3.4 ISP_TNR_MV_MANUAL_ATTR_S

【说明】

定义手动 TNR MV 属性参数

【定义】

```

typedef struct _ISP_TNR_MV_MANUAL_ATTR_S {
    CVI_U8 StableIntensity; /*Rw; Range:[0, 8]*/
    CVI_U8 L0YPSSBlur; /*Rw; Range:[0, 255]*/
    CVI_U16 AcbdSadAdj[TNR_ACBD_RANK]; /*Rw; Range:[0, 16383]*/

```

(下页继续)

(续上页)

```

CVI_U8 AcbdEdgeAdj[TNR_A CBD_RANK]; /*Rw; Range:[0, 16]*/
CVI_U8 LumaJndTh[TNR_LUMA_RANK]; /*Rw; Range:[0, 255]*/
CVI_U8 LumaJndRatio; /*Rw; Range:[0, 255]*/
CVI_U8 JndSadWt; /*Rw; Range:[0, 15]*/
CVI_U16 WarpB0[TNR_LUMA_RANK]; /*Rw; Range:[0, 16383]*/
CVI_U8 WarpEdge[TNR_LUMA_RANK]; /*Rw; Range:[0, 63]*/
CVI_U16 WarpBase[TNR_LUMA_RANK]; /*Rw; Range:[0, 16383]*/
CVI_U16 WarpStd[TNR_LUMA_RANK]; /*Rw; Range:[0, 16383]*/
} ISP_TNR_MV_MANUAL_ATTR_S;

```

【成员】

成员名称	描述
StableIntensity	稳定性系数，控制运动后画面变稳定的速度，值越大，画面越稳定 取值范围：[0, 8] 数据类型：CVI_U8
L0YPSSBlur	None 取值范围：[0, 255] 数据类型：CVI_U8
AcbdSadAdj	细调节不同光流值的置信度惩罚参数，对应的值越大，越不信任该光流值。对应关系同 AcbdEdgeAdj 取值范围：[0, 16383] 数据类型：CVI_U16
AcbdEdgeAdj	粗调节不同光流值的惩罚系数，对应的值越大，越不信任该光流值。AcbdEdgeAdj[0]~AcbdEdgeAdj[3] 分别控制当前帧光流值、前一帧当前块光流值、前一帧非当前块光流值、随机光流值的惩罚值 取值范围：[0, 16] 数据类型：CVI_U8
LumaJndTh	自动调整光流惩罚值的亮度阈值 取值范围：[0, 255] 数据类型：CVI_U8
LumaJndRatio	自动调整光流惩罚值的比例，255 为 1 倍不调整，小于 255 则等比例缩小 取值范围：[0, 255] 数据类型：CVI_U8
JndSadWt	None 取值范围：[0, 15] 数据类型：CVI_U8
WarpB0	当前后帧差异小于 WarpB0 时，认为该区域为静止区。值越大，越容易判定为静止 取值范围：[0, 16383] 数据类型：CVI_U16

下页继续

表 16.12 – 续上页

成员名称	描述
WarpEdge	细节区域的噪声增加权重参数。值越大对噪声容忍程度越大，此时受噪声扰动的静止区域较容易被判定为静止状态。值过大会使微小移动区域被误判为静止。可以分亮度设置不同的值 取值范围：[0, 63] 数据类型：CVI_U8
WarpBase	平坦区域的噪声偏移量参数 取值范围：[0, 16383] 数据类型：CVI_U16
WarpStd	控制运动状态判断的基数，值越大，对噪声的容忍程度越大，越不容易被判断为运动区。可以分亮度设置不同的值 取值范围：[0, 16383] 数据类型：CVI_U16

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetTNRMvAttr
- CVI_ISP_GetTNRMvAttr

16.3.5 ISP_TNR_MV_AUTO_ATTR_S

【说明】

定义自动 TNR MV 属性参数

【定义】

```
typedef struct _ISP_TNR_MV_AUTO_ATTR_S {
    CVI_U8 StableIntensity[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 8]*/
    CVI_U8 LoYPSSBlur[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U16 AcbdSadAdj[TNR_ACBD_RANK][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; ↵
↵Range:[0, 16383]*/
    CVI_U8 AcbdEdgeAdj[TNR_ACBD_RANK][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; ↵
↵Range:[0, 16]*/
    CVI_U8 LumaJndTh[TNR_LUMA_RANK][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; ↵
↵Range:[0, 255]*/
    CVI_U8 LumaJndRatio[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U8 JndSadWt[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 15]*/
    CVI_U16 WarpB0[TNR_LUMA_RANK][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; ↵
↵Range:[0, 16383]*/
    CVI_U8 WarpEdge[TNR_LUMA_RANK][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; ↵
↵Range:[0, 63]*/
    CVI_U16 WarpBase[TNR_LUMA_RANK][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; ↵
↵Range:[0, 16383]*/
    CVI_U16 WarpStd[TNR_LUMA_RANK][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; ↵
↵Range:[0, 16383]*/
} ISP_TNR_MV_AUTO_ATTR_S;
```

【成员】

成员名称	描述
StableIntensity	稳定性系数，控制运动后画面变稳定的速度，值越大，画面越稳定 取值范围：[0, 8] 数据类型：CVI_U8
L0YPSSBlur	None 取值范围：[0, 255] 数据类型：CVI_U8
AcbdSadAdj	细调节不同光流值的置信度惩罚参数，对应的值越大，越不信任该光流值。对应关系同 AcbdEdgeAdj 取值范围：[0, 16383] 数据类型：CVI_U16
AcbdEdgeAdj	粗调节不同光流值的惩罚系数，对应的值越大，越不信任该光流值。AcbdEdgeAdj[0]~AcbdEdgeAdj[3] 分别控制当前帧光流值、前一帧当前块光流值、前一帧非当前块光流值、随机光流值的惩罚值 取值范围：[0, 16] 数据类型：CVI_U8
LumaJndTh	自动调整光流惩罚值的亮度阈值 取值范围：[0, 255] 数据类型：CVI_U8
LumaJndRatio	自动调整光流惩罚值的比例，255 为 1 倍不调整，小于 255 则等比例缩小 取值范围：[0, 255] 数据类型：CVI_U8
JndSadWt	None 取值范围：[0, 15] 数据类型：CVI_U8
WarpB0	当前后帧差异小于 WarpB0 时，认为该区域为静止区。值越大，越容易判定为静止 取值范围：[0, 16383] 数据类型：CVI_U16
WarpEdge	细节区域的噪声增加权重参数。值越大对噪声容忍程度越大，此时受噪声扰动的静止区域较容易被判定为静止状态。值过大会使微小移动区域被误判为静止。可以分亮度设置不同的值 取值范围：[0, 63] 数据类型：CVI_U8
WarpBase	平坦区域的噪声偏移量参数 取值范围：[0, 16383] 数据类型：CVI_U16
WarpStd	控制运动状态判断的基数，值越大，对噪声的容忍程度越大，越不容易被判断为运动区。可以分亮度设置不同的值 取值范围：[0, 16383] 数据类型：CVI_U16

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetTNRMvAttr
- CVI_ISP_GetTNRMvAttr

16.3.6 ISP_TNR_MV_ATTR_S

【说明】

定义 TNR MV 属性参数

【定义】

```
typedef struct _ISP_TNR_MV_ATTR_S {
    CVI_BOOL NullFlowEn; /*Rw; Range:[0, 1]*/
    ISP_TNR_MV_MANUAL_ATTR_S stManual;
    ISP_TNR_MV_AUTO_ATTR_S stAuto;
} ISP_TNR_MV_ATTR_S;
```

【成员】

成员名称	描述
NullFlowEn	None 取值范围: [0, 1] 数据类型: CVI_BOOL
stManual	手动参数
stAuto	自动参数

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetTNRMvAttr
- CVI_ISP_GetTNRMvAttr

16.3.7 ISP_TNR_PS_MANUAL_ATTR_S

【说明】

定义手动 TNR PS 属性参数

【定义】

```
typedef struct _ISP_TNR_PS_MANUAL_ATTR_S {
    CVI_U8 JndStdRange[TNR_STATUS_CHANGE]; /*Rw; Range:[0, 32]*/
    CVI_U8 WarpStdRange[TNR_STATUS_CHANGE]; /*Rw; Range:[0, 32]*/
    CVI_U8 NbrBldRatio[TNR_STATUS_CHANGE]; /*Rw; Range:[0, 255]*/
    CVI_U16 StillRegionTh[TNR_STATUS_CHANGE]; /*Rw; Range:[0, 65535]*/
}
```

(下页继续)

(续上页)

```
CVI_U8 FlowJndTh; /*Rw; Range:[0, 63]*/
} ISP_TNR_PS_MANUAL_ATTR_S;
```

【成员】

成员名称	描述
JndStdRange	None 取值范围：[0, 32] 数据类型：CVI_U8
WarpStdRange	根据运动补偿建立噪声模型时，控制运动状态判断的系数。 WarpStdRange[0]：控制静止区与转换区的系数，值越大，越多的区域被判断为静止区；WarpStdRange[1]：控制转换区与运动区的系数，值越大，越多的区域被判断为转换区。WarpStdRange[0] 需小于 WarpStdRange[1] 取值范围：[0, 32] 数据类型：CVI_U8
NbrBldRatio	None 取值范围：[0, 255] 数据类型：CVI_U8
StillRegionTh	平坦区、大边缘区的划分阈值。特征值小于 StillRegionTh[0] 的区域认为是平坦区；特征值大于 StillRegionTh[1] 的区域认为是大边缘区；其余部分认为是纹理区。这部分的设定会影响静止区中滤波的处理 取值范围：[0, 65535] 数据类型：CVI_U16
FlowJndTh	将周围光流值长度 >FlowJndTh 的静止区，重定义为非静止区 取值范围：[0, 63] 数据类型：CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetTNRPsAttr
- CVI_ISP_GetTNRPsAttr

16.3.8 ISP_TNR_PS_AUTO_ATTR_S

【说明】

定义自动 TNR PS 属性参数

【定义】

```
typedef struct _ISP_TNR_PS_AUTO_ATTR_S {
    CVI_U8 JndStdRange[TNR_STATUS_CHANGE][ISP_AUTO_ISO_STRENGTH_NUM]; /
    ↪ *Rw; Range:[0, 32]*/
```

(下页继续)

(续上页)

```

    CVI_U8 WarpStdRange[TNR_STATUS_CHANGE][ISP_AUTO_ISO_STRENGTH_NUM]; /
    ↪ *Rw; Range:[0, 32]*/
    CVI_U8 NbrBldRatio[TNR_STATUS_CHANGE][ISP_AUTO_ISO_STRENGTH_NUM]; /
    ↪ *Rw; Range:[0, 255]*/
    CVI_U16 StillRegionTh[TNR_STATUS_CHANGE][ISP_AUTO_ISO_STRENGTH_NUM]; /
    ↪ *Rw; Range:[0, 65535]*/
    CVI_U8 FlowJndTh[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 63]*/
} ISP_TNR_PS_AUTO_ATTR_S;

```

【成员】

成员名称	描述
JndStdRange	None 取值范围：[0, 32] 数据类型：CVI_U8
WarpStdRange	根据运动补偿建立噪声模型时，控制运动状态判断的系数。 WarpStdRange[0]：控制静止区与转换区的系数，值越大，越多的区域被判断为静止区；WarpStdRange[1]：控制转换区与运动区的系数，值越大，越多的区域被判断为转换区。WarpStdRange[0] 需小于 WarpStdRange[1] 取值范围：[0, 32] 数据类型：CVI_U8
NbrBldRatio	None 取值范围：[0, 255] 数据类型：CVI_U8
StillRegionTh	平坦区、大边缘区的划分阈值。特征值小于 StillRegionTh[0] 的区域认为是平坦区；特征值大于 StillRegionTh[1] 的区域认为是大边缘区；其余部分认为是纹理区。这部分的设定会影响静止区中滤波的处理 取值范围：[0, 65535] 数据类型：CVI_U16
FlowJndTh	将周围光流值长度 > FlowJndTh 的静止区，重定义为非静止区 取值范围：[0, 63] 数据类型：CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetTNRPsAttr
- CVI_ISP_GetTNRPsAttr

16.3.9 ISP_TNR_PS_ATTR_S

【说明】

定义 TNR PS 属性参数

【定义】

```
typedef struct _ISP_TNR_PS_ATTR_S {  
    CVI_BOOL SadDebiasEN; /*Rw; Range:[0, 1]*/  
    CVI_U8 SadLpfMode; /*Rw; Range:[0, 3]*/  
    ISP_TNR_PS_MANUAL_ATTR_S stManual;  
    ISP_TNR_PS_AUTO_ATTR_S stAuto;  
} ISP_TNR_PS_ATTR_S;
```

【成员】

成员名称	描述
SadDebiasEN	None 取值范围：[0, 1] 数据类型：CVI_BOOL
SadLpfMode	None 取值范围：[0, 3] 数据类型：CVI_U8
stManual	手动参数
stAuto	自动参数

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetTNRPsAttr
- CVI_ISP_GetTNRPsAttr

16.3.10 ISP_TNR_NR_MANUAL_ATTR_S

【说明】

定义手动 TNR NR 属性参数

【定义】

```
typedef struct _ISP_TNR_NR_MANUAL_ATTR_S {  
    CVI_U8 L0YBlurStr[TNR_STATUS_NUM]; /*Rw; Range:[0, 255]*/  
    CVI_U8 L0YSigmaStr[TNR_STATUS_NUM]; /*Rw; Range:[0, 255]*/  
    CVI_U8 L0UVSigmaStr[TNR_STATUS_NUM]; /*Rw; Range:[0, 255]*/  
    CVI_U16 L0YGardBlurTh; /*Rw; Range:[0, 65535]*/  
    CVI_U8 L0YGardBlurStr; /*Rw; Range:[0, 255]*/  
    CVI_U8 L1PFTStr[TNR_FREQ_RANK]; /*Rw; Range:[0, 255]*/  
    CVI_U16 L1SpaGlobalStr; /*Rw; Range:[0, 65535]*/  
}
```

(下页继续)

(续上页)

```

CVI_U16 L1TmpGlobalStr; /*Rw; Range:[0, 65535]*/
CVI_U8 L1LumaAdj[TNR_LUMA_RANK]; /*Rw; Range:[0, 255]*/
CVI_U8 L1FSJndAdj[TNR_STATUS_NUM]; /*Rw; Range:[0, 255]*/
CVI_U8 L1FTJndAdj[TNR_STATUS_NUM]; /*Rw; Range:[0, 255]*/
CVI_U8 L1FSFreAdj[TNR_FREQ_RANK]; /*Rw; Range:[0, 255]*/
CVI_U8 L1FTFreAdj[TNR_FREQ_RANK]; /*Rw; Range:[0, 255]*/
CVI_U8 L1FTLumaStr; /*Rw; Range:[0, 255]*/
CVI_U8 L1PFTDamp[TNR_STATUS_CHANGE]; /*Rw; Range:[0, 255]*/
CVI_U8 L2YDampStr[TNR_STATUS_CHANGE]; /*Rw; Range:[0, 255]*/
CVI_U8 L2YLut[TNR_DETAIL_RANK]; /*Rw; Range:[0, 127]*/
CVI_U8 L2UVDampStr[TNR_STATUS_CHANGE]; /*Rw; Range:[0, 255]*/
CVI_U8 L2UVLut[TNR_DETAIL_RANK]; /*Rw; Range:[0, 127]*/
} ISP_TNR_NR_MANUAL_ATTR_S;

```

【成员】

成员名称	描述
L0YBlurStr	L0 滤波，Y 通道第一级空域滤波强度。值越大，滤波强度越强。L0YBlurStr[0]~L0YBlurStr[2] 分别对应静止区、转换区、运动区的滤波强度 取值范围：[0, 255] 数据类型：CVI_U8
L0YSigmaStr	L0 滤波，Y 通道第一级时域滤波强度。值越大，滤波强度越强。L0YSigmaStr[0]~L0YSigmaStr[2] 分别对应静止区、转换区、运动区的滤波强度 取值范围：[0, 255] 数据类型：CVI_U8
L0UVSigmaStr	L0 滤波，UV 通道第一级时域滤波强度。参数趋势同 L0YSigmaStr 取值范围：[0, 255] 数据类型：CVI_U8
L0YGardBlurTh	None 取值范围：[0, 65535] 数据类型：CVI_U16
L0YGardBlurStr	None 取值范围：[0, 255] 数据类型：CVI_U8
L1PFTStr	L1P 滤波，静止区的滤波强度。可以分不同的频率设置不同的强度，值越大，滤波强度越强。L1PFTStr[0]~L1PFTStr[3] 表示从低频率到高频率的滤波强度 取值范围：[0, 255] 数据类型：CVI_U8
L1SpaGlobalStr	L1 滤波，空域滤波强度基值。值越大，滤波强度越强 取值范围：[0, 65535] 数据类型：CVI_U16
L1TmpGlobalStr	L1 滤波，时域滤波强度基值。值越大，滤波强度越强 取值范围：[0, 65535] 数据类型：CVI_U16

下页继续

表 16.18 – 续上页

成员名称	描述
L1LumaAdj	L1 滤波，分亮度设置不同的滤波强度，时域滤波和空域滤波共用同一组设定。值越大，滤波强度越强。L1LumaAdj[0]~L1LumaAdj[7] 表示从暗区到亮区的滤波强度 取值范围：[0, 255] 数据类型：CVI_U8
L1FSJndAdj	L1 滤波，空域滤波分运动状态设置不同的滤波强度。值越大，滤波强度越强。L1FSJndAdj[0]~L1FSJndAdj[2] 分别表示静止区、转换区、运动区的滤波强度 取值范围：[0, 255] 数据类型：CVI_U8
L1FTJndAdj	L1 滤波，时域滤波分运动状态设置不同的滤波强度。参数趋势同 L1FSJndAdj 取值范围：[0, 255] 数据类型：CVI_U8
L1FSFreAdj	L1 滤波，空域滤波分频率设置不同的滤波强度。值越大，滤波强度越强。L1FSFreAdj[0]~L1FSFreAdj[3] 表示从低频到高频的滤波强度 取值范围：[0, 255] 数据类型：CVI_U8
L1FTFreAdj	L1 滤波，时域滤波分频率设置不同的滤波强度。参数趋势同 L1FSFreAdj 取值范围：[0, 255] 数据类型：CVI_U8
L1FTLumaStr	None 取值范围：[0, 255] 数据类型：CVI_U8
L1PFTDamp	L1P 滤波，转换区和运动区的滤波强度系数。值越大，滤波强度越强，即越接近静止区的滤波强度，也越容易造成拖影 取值范围：[0, 255] 数据类型：CVI_U8
L2YDampStr	L2 滤波，Y 通道降噪分运动状态设置不同的滤波强度。L2YDampStr[0]~L2YDampStr[1] 分别表示转换区、运动区的滤波强度 取值范围：[0, 255] 数据类型：CVI_U8
L2YLut	L2 滤波，Y 通道降噪强度控制。值越小，残留的时域噪声越小。值过小，容易造成拖影以及噪声收敛慢 取值范围：[0, 127] 数据类型：CVI_U8
L2UVDampStr	L2 后滤波，UV 通道降噪分运动状态设置不同的滤波强度。参数趋势同 L2YDampStr 取值范围：[0, 255] 数据类型：CVI_U8
L2UVLut	L2 后滤波，UV 通道降噪强度控制。值越小，残留的时域噪声越小 取值范围：[0, 127] 数据类型：CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetTNRNrAttr
- CVI_ISP_GetTNRNrAttr

16.3.11 ISP_TNR_NR_AUTO_ATTR_S**【说明】**

定义自动 TNR NR 属性参数

【定义】

```
typedef struct _ISP_TNR_NR_AUTO_ATTR_S {
    CVI_U8 L0YBlurStr[TNR_STATUS_NUM][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw;↵
↵Range:[0, 255]*/
    CVI_U8 L0YSigmaStr[TNR_STATUS_NUM][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw;↵
↵Range:[0, 255]*/
    CVI_U8 L0UVSigmaStr[TNR_STATUS_NUM][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw;↵
↵Range:[0, 255]*/
    CVI_U16 L0YGardBlurTh[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 65535]*/
    CVI_U8 L0YGardBlurStr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U8 L1PFTStr[TNR_FREQ_RANK][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw;↵
↵Range:[0, 255]*/
    CVI_U16 L1SpaGlobalStr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 65535]*/
    CVI_U16 L1TmpGlobalStr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 65535]*/
    CVI_U8 L1LumaAdj[TNR_LUMA_RANK][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw;↵
↵Range:[0, 255]*/
    CVI_U8 L1FSJndAdj[TNR_STATUS_NUM][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw;↵
↵Range:[0, 255]*/
    CVI_U8 L1FTJndAdj[TNR_STATUS_NUM][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw;↵
↵Range:[0, 255]*/
    CVI_U8 L1FSFreAdj[TNR_FREQ_RANK][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw;↵
↵Range:[0, 255]*/
    CVI_U8 L1FTFreAdj[TNR_FREQ_RANK][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw;↵
↵Range:[0, 255]*/
    CVI_U8 L1FTLumaStr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U8 L1PFTDamp[TNR_STATUS_CHANGE][ISP_AUTO_ISO_STRENGTH_NUM]; /
↵*Rw; Range:[0, 255]*/
    CVI_U8 L2YDampStr[TNR_STATUS_CHANGE][ISP_AUTO_ISO_STRENGTH_NUM]; /
↵*Rw; Range:[0, 255]*/
    CVI_U8 L2YLut[TNR_DETAIL_RANK][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw;↵
↵Range:[0, 127]*/
    CVI_U8 L2UVDampStr[TNR_STATUS_CHANGE][ISP_AUTO_ISO_STRENGTH_NUM]; /
↵*Rw; Range:[0, 255]*/
    CVI_U8 L2UVLut[TNR_DETAIL_RANK][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw;↵
↵Range:[0, 127]*/
} ISP_TNR_NR_AUTO_ATTR_S;
```

【成员】

成员名称	描述
L0YBlurStr	L0 滤波，Y 通道第一级空域滤波强度。值越大，滤波强度越强。L0YBlurStr[0]~L0YBlurStr[2] 分别对应静止区、转换区、运动区的滤波强度 取值范围：[0, 255] 数据类型：CVI_U8
L0YSigmaStr	L0 滤波，Y 通道第一级时域滤波强度。值越大，滤波强度越强。L0YSigmaStr[0]~L0YSigmaStr[2] 分别对应静止区、转换区、运动区的滤波强度 取值范围：[0, 255] 数据类型：CVI_U8
L0UVSigmaStr	L0 滤波，UV 通道第一级时域滤波强度。参数趋势同 L0YSigmaStr 取值范围：[0, 255] 数据类型：CVI_U8
L0YGardBlurTh	None 取值范围：[0, 65535] 数据类型：CVI_U16
L0YGardBlurStr	None 取值范围：[0, 255] 数据类型：CVI_U8
L1PFTStr	L1P 滤波，静止区的滤波强度。可以分不同的频率设置不同的强度，值越大，滤波强度越强。L1PFTStr[0]~L1PFTStr[3] 表示从低频率到高频率的滤波强度 取值范围：[0, 255] 数据类型：CVI_U8
L1SpaGlobalStr	L1 滤波，空域滤波强度基值。值越大，滤波强度越强 取值范围：[0, 65535] 数据类型：CVI_U16
L1TmpGlobalStr	L1 滤波，时域滤波强度基值。值越大，滤波强度越强 取值范围：[0, 65535] 数据类型：CVI_U16
L1LumaAdj	L1 滤波，分亮度设置不同的滤波强度，时域滤波和空域滤波共用同一组设定。值越大，滤波强度越强。L1LumaAdj[0]~L1LumaAdj[7] 表示从暗区到亮区的滤波强度 取值范围：[0, 255] 数据类型：CVI_U8
L1FSJndAdj	L1 滤波，空域滤波分运动状态设置不同的滤波强度。值越大，滤波强度越强。L1FSJndAdj[0]~L1FSJndAdj[2] 分别表示静止区、转换区、运动区的滤波强度 取值范围：[0, 255] 数据类型：CVI_U8
L1FTJndAdj	L1 滤波，时域滤波分运动状态设置不同的滤波强度。参数趋势同 L1FSJndAdj 取值范围：[0, 255] 数据类型：CVI_U8

下页继续

表 16.19 – 续上页

成员名称	描述
L1FSFreAdj	L1 滤波，空域滤波分频率设置不同的滤波强度。值越大，滤波强度越强。L1FSFreAdj[0]~L1FSFreAdj[3] 表示从低频到高频的滤波强度 取值范围：[0, 255] 数据类型：CVI_U8
L1FTFreAdj	L1 滤波，时域滤波分频率设置不同的滤波强度。参数趋势同 L1FSFreAdj 取值范围：[0, 255] 数据类型：CVI_U8
L1FTLumaStr	None 取值范围：[0, 255] 数据类型：CVI_U8
L1PFTDamp	L1P 滤波，转换区和运动区的滤波强度系数。值越大，滤波强度越强，即越接近静止区的滤波强度，也越容易造成拖影 取值范围：[0, 255] 数据类型：CVI_U8
L2YDampStr	L2 滤波，Y 通道降噪分运动状态设置不同的滤波强度。L2YDampStr[0]~L2YDampStr[1] 分别表示转换区、运动区的滤波强度 取值范围：[0, 255] 数据类型：CVI_U8
L2YLut	L2 滤波，Y 通道降噪强度控制。值越小，残留的时域噪声越小。值过小，容易造成拖影以及噪声收敛慢 取值范围：[0, 127] 数据类型：CVI_U8
L2UVDampStr	L2 后滤波，UV 通道降噪分运动状态设置不同的滤波强度。参数趋势同 L2YDampStr 取值范围：[0, 255] 数据类型：CVI_U8
L2UVLut	L2 后滤波，UV 通道降噪强度控制。值越小，残留的时域噪声越小 取值范围：[0, 127] 数据类型：CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetTNRNrAttr
- CVI_ISP_GetTNRNrAttr

16.3.12 ISP_TNR_NR_ATTR_S

【说明】

定义 TNR NR 属性参数

【定义】

```
typedef struct _ISP_TNR_NR_ATTR_S {  
    CVI_BOOL L12NrYEN; /*Rw; Range:[0, 1]*/  
    CVI_BOOL L2UVNrEN; /*Rw; Range:[0, 1]*/  
    CVI_BOOL L2UVMonoEN; /*Rw; Range:[0, 1]*/  
    CVI_BOOL DcEn; /*Rw; Range:[0, 1]*/  
    ISP_TNR_NR_MANUAL_ATTR_S stManual;  
    ISP_TNR_NR_AUTO_ATTR_S stAuto;  
} ISP_TNR_NR_ATTR_S;
```

【成员】

成员名称	描述
L12NrYEN	前中后滤波的 Y 通道处理使能 0: 关闭 1: 使能 取值范围: [0, 1] 数据类型: CVI_BOOL
L2UVNrEN	后滤波的 UV 通道处理使能 0: 关闭 1: 使能 取值范围: [0, 1] 数据类型: CVI_BOOL
L2UVMonoEN	None 取值范围: [0, 1] 数据类型: CVI_BOOL
DcEn	None 取值范围: [0, 1] 数据类型: CVI_BOOL
stManual	手动参数
stAuto	自动参数

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetTNRNrAttr
- CVI_ISP_GetTNRNrAttr

17 Crosstalk

17.1 功能描述

可以平衡 raw 之间临近像素 Gr 和 Gb 之间的差异，能而有效防止 demosaic 插值算法产生的方格或其他类似 pattern。

17.2 API 参考

- `CVI_ISP_SetCrosstalkAttr`：设置 Crosstalk 属性参数
- `CVI_ISP_GetCrosstalkAttr`：获取 Crosstalk 属性参数

17.2.1 CVI_ISP_SetCrosstalkAttr

【描述】

设置 Crosstalk 属性参数

【语法】

```
CVI_S32 CVI_ISP_SetCrosstalkAttr(VI_PIPE ViPipe, const ISP_CROSSTALK_ATTR_S_
↪ *pstCrosstalkAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstCrosstalkAttr	Crosstalk 属性参数	输入

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

```
VI_PIPE ViPipe = 0;
ISP_CROSSTALK_ATTR_S stAttr;
CVI_ISP_GetCrosstalkAttr(ViPipe, &stAttr);
stAttr.enOpType = OP_TYPE_AUTO;
CVI_ISP_SetCrosstalkAttr(ViPipe, &stAttr);
```

【相关主题】

- [CVI_ISP_GetCrosstalkAttr](#)

17.2.2 CVI_ISP_GetCrosstalkAttr

【描述】

获取 Crosstalk 属性参数

【语法】

```
CVI_S32 CVI_ISP_GetCrosstalkAttr(VI_PIPE ViPipe, ISP_CROSSTALK_ATTR_S_
↪ *pstCrosstalkAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstCrosstalkAttr	Crosstalk 属性参数	输出

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

无

【相关主题】

- CVI_ISP_SetCrosstalkAttr

17.3 数据类型

- ISP_CROSSTALK_MANUAL_ATTR_S：定义手动 Crosstalk 属性参数
- ISP_CROSSTALK_AUTO_ATTR_S：定义自动 Crosstalk 属性参数
- ISP_CROSSTALK_ATTR_S：定义 Crosstalk 属性参数

17.3.1 ISP_CROSSTALK_MANUAL_ATTR_S

【说明】

定义手动 Crosstalk 属性参数

【定义】

```
typedef struct _ISP_CROSSTALK_MANUAL_ATTR_S {  
    CVI_U16 Strength; /*Rw; Range:[0, 256]*/  
} ISP_CROSSTALK_MANUAL_ATTR_S;
```

【成员】

成员名称	描述
Strength	G 通道平衡全局强度 取值范围：[0, 256] 数据类型：CVI_U16

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetCrosstalkAttr
- CVI_ISP_GetCrosstalkAttr

17.3.2 ISP_CROSSTALK_AUTO_ATTR_S

【说明】

定义自动 Crosstalk 属性参数

【定义】

```
typedef struct _ISP_CROSSTALK_AUTO_ATTR_S {  
    CVI_U16 Strength[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 256]*/  
} ISP_CROSSTALK_AUTO_ATTR_S;
```

【成员】

成员名称	描述
Strength	G 通道平衡全局强度 取值范围：[0, 256] 数据类型：CVI_U16

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetCrosstalkAttr
- CVI_ISP_GetCrosstalkAttr

17.3.3 ISP_CROSSTALK_ATTR_S

【说明】

定义 Crosstalk 属性参数

【定义】

```
typedef struct _ISP_CROSSTALK_ATTR_S {  
    CVI_BOOL Enable; /*Rw; Range:[0, 1]*/  
    ISP_OP_TYPE_E enOpType;  
    CVI_U8 UpdateInterval; /*Rw; Range:[0, 255]*/  
    CVI_U16 GrGbDiffThreSec[4]; /*Rw; Range:[0, 4095]*/  
    CVI_U16 FlatThre[4]; /*Rw; Range:[0, 4095]*/  
    ISP_CROSSTALK_MANUAL_ATTR_S stManual;  
    ISP_CROSSTALK_AUTO_ATTR_S stAuto;  
} ISP_CROSSTALK_ATTR_S;
```

【成员】

成员名称	描述
Enable	模块使能开关 取值范围：[0, 1] 数据类型：CVI_BOOL
enOpType	工作类型 OP_TYPE_AUTO：自动模式 OP_TYPE_MANUAL：手动模式 数据类型：ISP_OP_TYPE_E
UpdateInterval	影响参数更新间隔, 值越大画面变化越慢, 效能越好。 取值范围：[0, 255] 数据类型：CVI_U8
GrGbDiffThreSec	G 通道平衡节点 1-4 阈值 取值范围：[0, 4095] 数据类型：CVI_U16
FlatThre	平坦区侦测节点 1-4 阈值 取值范围：[0, 4095] 数据类型：CVI_U16
stManual	手动参数
stAuto	自动参数

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetCrosstalkAttr
- CVI_ISP_GetCrosstalkAttr

18 DEMOSAIC

18.1 功能描述

把图像由 Bayer Pattern 内插出完整的 RGB 而提升细节与清晰度。

18.2 API 参考

- `CVI_ISP_SetDemosaicAttr` : 设置 Demosaic 去马赛克属性参数
- `CVI_ISP_GetDemosaicAttr` : 获取 Demosaic 去马赛克属性参数
- `CVI_ISP_SetDemosaicDemoireAttr` : 设置 Demosaic 去马赛克属性参数
- `CVI_ISP_GetDemosaicDemoireAttr` : 获取 Demosaic 去马赛克属性参数

18.2.1 CVI_ISP_SetDemosaicAttr

【描述】

设置 Demosaic 去马赛克属性参数

【语法】

```
CVI_S32 CVI_ISP_SetDemosaicAttr(VI_PIPE ViPipe, const ISP_DEMOSAIC_ATTR_S_
↪ *pstDemosaicAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstDemosaicAttr	Demosaic 去马赛克属性参数	输入

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

```
VI_PIPE ViPipe = 0;
ISP_DEMOSAIC_ATTR_S stAttr;
CVI_ISP_GetDemosaicAttr(ViPipe, &stAttr);
stAttr.enOpType = OP_TYPE_AUTO;
CVI_ISP_SetDemosaicAttr(ViPipe, &stAttr);
```

【相关主题】

- [CVI_ISP_GetDemosaicAttr](#)

18.2.2 CVI_ISP_GetDemosaicAttr

【描述】

获取 Demosaic 去马赛克属性参数

【语法】

```
CVI_S32 CVI_ISP_GetDemosaicAttr(VI_PIPE ViPipe, ISP_DEMOSAIC_ATTR_S_
↪ *pstDemosaicAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstDemosaicAttr	Demosaic 去马赛克属性参数	输出

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`

- 库文件: libisp.so

【注意】

无

【举例】

无

【相关主题】

- [CVI_ISP_SetDemosaicAttr](#)

18.2.3 CVI_ISP_SetDemosaicDemoireAttr

【描述】

设置 Demosaic 去马赛克属性参数

【语法】

```
CVI_S32 CVI_ISP_SetDemosaicDemoireAttr(VI_PIPE ViPipe, const ISP_DEMOSAIC_DEMOIRE_
→ATTR_S *pstDemosaicDemoireAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstDemosaicDemoireAttr	Demosaic 去马赛克属性参数	输入

【返回值】

返回值	描述
0	成功
非 0	失败, 其值为错误码。

【需求】

- 头文件: cvi_isp.h, cvi_comm_isp.h
- 库文件: libisp.so

【注意】

无

【举例】

```
VI_PIPE ViPipe = 0;
ISP_DEMOSAIC_DEMOIRE_ATTR_S stAttr;
CVI_ISP_GetDemosaicDemoireAttr(ViPipe, &stAttr);
stAttr.enOpType = OP_TYPE_AUTO;
CVI_ISP_SetDemosaicDemoireAttr(ViPipe, &stAttr);
```

【相关主题】

- [CVI_ISP_GetDemosaicDemoireAttr](#)

18.2.4 CVI_ISP_GetDemosaicDemoireAttr

【描述】

获取 Demosaic 去马赛克属性参数

【语法】

```
CVI_S32 CVI_ISP_GetDemosaicDemoireAttr(VI_PIPE ViPipe, ISP_DEMOSAIC_DEMOIRE_
↪ATTR_S *pstDemosaicDemoireAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstDemosaicDe- moireAttr	Demosaic 去马赛克属性参数	输出

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

无

【相关主题】

- [CVI_ISP_SetDemosaicDemoireAttr](#)

18.3 数据类型

- `ISP_DEMOSAIC_MANUAL_ATTR_S`：定义手动 Demosaic 去马赛克属性参数
- `ISP_DEMOSAIC_AUTO_ATTR_S`：定义自动 Demosaic 去马赛克属性参数
- `ISP_DEMOSAIC_ATTR_S`：定义 Demosaic 去马赛克属性参数
- `ISP_DEMOSAIC_DEMOIRE_MANUAL_ATTR_S`：定义手动 Demosaic 去马赛克属性参数
- `ISP_DEMOSAIC_DEMOIRE_AUTO_ATTR_S`：定义自动 Demosaic 去马赛克属性参数
- `ISP_DEMOSAIC_DEMOIRE_ATTR_S`：定义 Demosaic 去马赛克属性参数

18.3.1 `ISP_DEMOSAIC_MANUAL_ATTR_S`

【说明】

定义手动 Demosaic 去马赛克属性参数

【定义】

```
typedef struct _ISP_DEMOSAIC_MANUAL_ATTR_S {  
    CVI_U16 CoarseEdgeThr; /*Rw; Range:[0, 4095]*/  
    CVI_U16 CoarseStr; /*Rw; Range:[0, 4095]*/  
    CVI_U16 FineEdgeThr; /*Rw; Range:[0, 4095]*/  
    CVI_U16 FineStr; /*Rw; Range:[0, 4095]*/  
    CVI_U16 RbSigLumaThd; /*Rw; Range:[0, 4095]*/  
    CVI_U8 FilterMode; /*Rw; Range:[0, 1]*/  
} ISP_DEMOSAIC_MANUAL_ATTR_S;
```

【成员】

成员名称	描述
CoarseEdgeThr	边缘粗调侦测阈值。值越小，侦测为边缘的数量越多。建议搭配参数 CoarseStr 调试。 取值范围：[0, 4095] 数据类型：CVI_U16
CoarseStr	边缘粗调强度值。值越小，越偏方向性的处理。反之，越偏无方向性的处理。 取值范围：[0, 4095] 数据类型：CVI_U16
FineEdgeThr	边缘细调侦测阈值。值越小，侦测为边缘的数量越多。建议搭配参数 FineStr 调试。 取值范围：[0, 4095] 数据类型：CVI_U16

下页继续

表 18.5 – 续上页

成员名称	描述
FineStr	边缘细调强度值。值越小，越偏方向性的处理。反之，越偏无方向性的处理。 取值范围：[0, 4095] 数据类型：CVI_U16
RbSigLumaThd	None 取值范围：[0, 4095] 数据类型：CVI_U16
FilterMode	DC 影像的锐化宽度，数值越大，边缘被锐利化的宽度越宽，视觉上会越醒目。 取值范围：[0, 1] 数据类型：CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetDemosaicAttr
- CVI_ISP_GetDemosaicAttr

18.3.2 ISP_DEMOSAIC_AUTO_ATTR_S

【说明】

定义自动 Demosaic 去马赛克属性参数

【定义】

```
typedef struct _ISP_DEMOSAIC_AUTO_ATTR_S {  
    CVI_U16 CoarseEdgeThr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 4095]*/  
    CVI_U16 CoarseStr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 4095]*/  
    CVI_U16 FineEdgeThr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 4095]*/  
    CVI_U16 FineStr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 4095]*/  
    CVI_U16 RbSigLumaThd[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 4095]*/  
    CVI_U8 FilterMode[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 1]*/  
} ISP_DEMOSAIC_AUTO_ATTR_S;
```

【成员】

成员名称	描述
CoarseEdgeThr	边缘粗调侦测阈值。值越小，侦测为边缘的数量越多。建议搭配参数 CoarseStr 调试。 取值范围：[0, 4095] 数据类型：CVI_U16

下页继续

表 18.6 – 续上页

成员名称	描述
CoarseStr	边缘粗调强度值。值越小，越偏方向性的处理。反之，越偏无方向性的处理。 取值范围：[0, 4095] 数据类型：CVI_U16
FineEdgeThr	边缘细调侦测阈值。值越小，侦测为边缘的数量越多。建议搭配参数 FineStr 调试。 取值范围：[0, 4095] 数据类型：CVI_U16
FineStr	边缘细调强度值。值越小，越偏方向性的处理。反之，越偏无方向性的处理。 取值范围：[0, 4095] 数据类型：CVI_U16
RbSigLumaThd	None 取值范围：[0, 4095] 数据类型：CVI_U16
FilterMode	DC 影像的锐化宽度，数值越大，边缘被锐利化的宽度越宽，视觉上会越醒目。 取值范围：[0, 1] 数据类型：CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetDemosaicAttr
- CVI_ISP_GetDemosaicAttr

18.3.3 ISP_DEMOSAIC_ATTR_S

【说明】

定义 Demosaic 去马赛克属性参数

【定义】

```
typedef struct _ISP_DEMOSAIC_ATTR_S {  
    CVI_BOOL Enable; /*Rw; Range:[0, 1]*/  
    ISP_OP_TYPE_E enOpType;  
    CVI_U8 UpdateInterval; /*Rw; Range:[0, 255]*/  
    CVI_BOOL TuningMode; /*Rw; Range:[0, 1]*/  
    CVI_BOOL RbVtEnable; /*Rw; Range:[0, 1]*/  
    ISP_DEMOSAIC_MANUAL_ATTR_S stManual;  
    ISP_DEMOSAIC_AUTO_ATTR_S stAuto;  
} ISP_DEMOSAIC_ATTR_S;
```

【成员】

成员名称	描述
Enable	模块使能开关 取值范围：[0, 1] 数据类型：CVI_BOOL
enOpType	工作类型 OP_TYPE_AUTO：自动模式 OP_TYPE_MANUAL：手动模式 数据类型：ISP_OP_TYPE_E
UpdateInterval	影响参数更新间隔，值越大画面变化越慢，效能越好。 取值范围：[0, 255] 数据类型：CVI_U8
TuningMode	输出调试方案，辅助调节参数。0: Demosaic 图像结果。1: 平坦/垂直/水平边缘侦测图像结果。 取值范围：[0, 1] 数据类型：CVI_BOOL
RbVtEnable	None 取值范围：[0, 1] 数据类型：CVI_BOOL
stManual	手动参数
stAuto	自动参数

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetDemosaicAttr
- CVI_ISP_GetDemosaicAttr

18.3.4 ISP_DEMOSAIC_DEMOIRE_MANUAL_ATTR_S

【说明】

定义手动 Demosaic 去马赛克属性参数

【定义】

```
typedef struct _ISP_DEMOSAIC_DEMOIRE_MANUAL_ATTR_S {
    CVI_U8 AntiFalseColorStr; /*Rw; Range:[0, 255]*/
    CVI_U16 SatGainIn[2]; /*Rw; Range:[0, 4095]*/
    CVI_U16 SatGainOut[2]; /*Rw; Range:[0, 4095]*/
    CVI_U16 ProtectColorGainIn[2]; /*Rw; Range:[0, 4095]*/
    CVI_U16 ProtectColorGainOut[2]; /*Rw; Range:[0, 4095]*/
    CVI_U16 UserDefineProtectColor1; /*Rw; Range:[0, 4095]*/
    CVI_U16 UserDefineProtectColor2; /*Rw; Range:[0, 4095]*/
    CVI_U16 UserDefineProtectColor3; /*Rw; Range:[0, 4095]*/
    CVI_U16 EdgeGainIn[2]; /*Rw; Range:[0, 4095]*/
    CVI_U16 EdgeGainOut[2]; /*Rw; Range:[0, 4095]*/
    CVI_U16 DetailGainIn[2]; /*Rw; Range:[0, 4095]*/
    CVI_U16 DetailGainOut[2]; /*Rw; Range:[0, 4095]*/
}
```

(下页继续)

(续上页)

```

CVI_U16 DetailDetectLumaStr; /*Rw; Range:[0, 4095]*/
CVI_U8 DetailSmoothStr; /*Rw; Range:[0, 255]*/
CVI_U8 DetailWgtThr; /*Rw; Range:[0, 255]*/
CVI_U16 DetailWgtMin; /*Rw; Range:[0, 256]*/
CVI_U16 DetailWgtMax; /*Rw; Range:[0, 256]*/
CVI_U16 DetailWgtSlope; /*Rw; Range:[0, 1024]*/
CVI_U8 EdgeWgtNp; /*Rw; Range:[0, 255]*/
CVI_U8 EdgeWgtThr; /*Rw; Range:[0, 255]*/
CVI_U16 EdgeWgtMin; /*Rw; Range:[0, 256]*/
CVI_U16 EdgeWgtMax; /*Rw; Range:[0, 256]*/
CVI_U16 EdgeWgtSlope; /*Rw; Range:[0, 1024]*/
CVI_U8 DetailSmoothMapTh; /*Rw; Range:[0, 255]*/
CVI_U16 DetailSmoothMapMin; /*Rw; Range:[0, 256]*/
CVI_U16 DetailSmoothMapMax; /*Rw; Range:[0, 256]*/
CVI_U16 DetailSmoothMapSlope; /*Rw; Range:[0, 1024]*/
CVI_U8 LumaWgt; /*Rw; Range:[0, 255]*/
CVI_U8 SharpenGain; /*Rw; Range:[0, 255]*/
} ISP_DEMOSAIC_DEMOIRE_MANUAL_ATTR_S;

```

【成员】

成员名称	描述
AntiFalseColorStr	None 取值范围：[0, 255] 数据类型：CVI_U8
SatGainIn	None 取值范围：[0, 4095] 数据类型：CVI_U16
SatGainOut	None 取值范围：[0, 4095] 数据类型：CVI_U16
ProtectColorGainIn	None 取值范围：[0, 4095] 数据类型：CVI_U16
ProtectColorGainOut	None 取值范围：[0, 4095] 数据类型：CVI_U16
UserDefineProtectColor1	None 取值范围：[0, 4095] 数据类型：CVI_U16
UserDefineProtectColor2	None 取值范围：[0, 4095] 数据类型：CVI_U16
UserDefineProtectColor3	None 取值范围：[0, 4095] 数据类型：CVI_U16
EdgeGainIn	None 取值范围：[0, 4095] 数据类型：CVI_U16

下页继续

表 18.8 – 续上页

成员名称	描述
EdgeGainOut	None 取值范围：[0, 4095] 数据类型：CVI_U16
DetailGainIn	None 取值范围：[0, 4095] 数据类型：CVI_U16
DetailGaintOut	None 取值范围：[0, 4095] 数据类型：CVI_U16
DetailDetectLumaStr	None 取值范围：[0, 4095] 数据类型：CVI_U16
DetailSmoothStr	细节平滑强度。值越大，平滑强度越强，对伪细节的抑制强度越大。 取值范围：[0, 255] 数据类型：CVI_U8
DetailWgtThr	细节保留范围阈值。值越小，细节保留作用的范围越大。 取值范围：[0, 255] 数据类型：CVI_U8
DetailWgtMin	边缘细节平滑保留允许之最小增益。 取值范围：[0, 256] 数据类型：CVI_U16
DetailWgtMax	边缘细节平滑保留允许之最大增益。 取值范围：[0, 256] 数据类型：CVI_U16
DetailWgtSlope	细节保留强度。值越大，细节保留越多。 取值范围：[0, 1024] 数据类型：CVI_U16
EdgeWgtNp	None 取值范围：[0, 255] 数据类型：CVI_U8
EdgeWgtThr	细节保留范围阈值。值越小，细节保留作用的范围越大。 取值范围：[0, 255] 数据类型：CVI_U8
EdgeWgtMin	边缘细节平滑保留允许之最小增益。 取值范围：[0, 256] 数据类型：CVI_U16
EdgeWgtMax	边缘细节平滑保留允许之最大增益。 取值范围：[0, 256] 数据类型：CVI_U16
EdgeWgtSlope	细节保留强度。值越大，细节保留越多。 取值范围：[0, 1024] 数据类型：CVI_U16
DetailSmoothMapTh	None 取值范围：[0, 255] 数据类型：CVI_U8

下页继续

表 18.8 – 续上页

成员名称	描述
DetailSmoothMapMin	None 取值范围：[0, 256] 数据类型：CVI_U16
DetailSmoothMapMax	None 取值范围：[0, 256] 数据类型：CVI_U16
DetailSmoothMapSlope	None 取值范围：[0, 1024] 数据类型：CVI_U16
LumaWgt	None 取值范围：[0, 255] 数据类型：CVI_U8
SharpenGain	对 CFA 输出的 RGB 图施加的锐化强度，值越大，锐化越强，32 为 1x gain。 取值范围：[0, 255] 数据类型：CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetDemosaicDemoireAttr
- CVI_ISP_GetDemosaicDemoireAttr

18.3.5 ISP_DEMOSAIC_DEMOIRE_AUTO_ATTR_S

【说明】

定义自动 Demosaic 去马赛克属性参数

【定义】

```
typedef struct _ISP_DEMOSAIC_DEMOIRE_AUTO_ATTR_S {
    CVI_U8 AntiFalseColorStr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U16 SatGainIn[2][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 4095]*/
    CVI_U16 SatGainOut[2][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 4095]*/
    CVI_U16 ProtectColorGainIn[2][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 4095]*/
    CVI_U16 ProtectColorGainOut[2][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 4095]*/
    CVI_U16 UserDefineProtectColor1[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 4095]*/
    CVI_U16 UserDefineProtectColor2[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 4095]*/
    CVI_U16 UserDefineProtectColor3[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 4095]*/
    CVI_U16 EdgeGainIn[2][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 4095]*/
    CVI_U16 EdgeGainOut[2][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 4095]*/
}
```

(下页继续)

(续上页)

```

CVI_U16 DetailGainIn[2][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 4095]*/
CVI_U16 DetailGainOut[2][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 4095]*/
CVI_U16 DetailDetectLumaStr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 4095]*/
CVI_U8 DetailSmoothStr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
CVI_U8 DetailWgtThr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
CVI_U16 DetailWgtMin[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 256]*/
CVI_U16 DetailWgtMax[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 256]*/
CVI_U16 DetailWgtSlope[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 1024]*/
CVI_U8 EdgeWgtNp[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
CVI_U8 EdgeWgtThr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
CVI_U16 EdgeWgtMin[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 256]*/
CVI_U16 EdgeWgtMax[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 256]*/
CVI_U16 EdgeWgtSlope[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 1024]*/
CVI_U8 DetailSmoothMapTh[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
CVI_U16 DetailSmoothMapMin[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 256]*/
CVI_U16 DetailSmoothMapMax[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 256]*/
CVI_U16 DetailSmoothMapSlope[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0,
↪1024]*/
CVI_U8 LumaWgt[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
CVI_U8 SharpenGain[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
} ISP_DEMOSAIC_DEMOIRE_AUTO_ATTR_S;

```

【成员】

成员名称	描述
AntiFalseColorStr	None 取值范围：[0, 255] 数据类型：CVI_U8
SatGainIn	None 取值范围：[0, 4095] 数据类型：CVI_U16
SatGainOut	None 取值范围：[0, 4095] 数据类型：CVI_U16
ProtectColorGainIn	None 取值范围：[0, 4095] 数据类型：CVI_U16
ProtectColorGainOut	None 取值范围：[0, 4095] 数据类型：CVI_U16
UserDefineProtectColor1	None 取值范围：[0, 4095] 数据类型：CVI_U16
UserDefineProtectColor2	None 取值范围：[0, 4095] 数据类型：CVI_U16
UserDefineProtectColor3	None 取值范围：[0, 4095] 数据类型：CVI_U16

下页继续

表 18.9 – 续上页

成员名称	描述
EdgeGainIn	None 取值范围：[0, 4095] 数据类型：CVI_U16
EdgeGainOut	None 取值范围：[0, 4095] 数据类型：CVI_U16
DetailGainIn	None 取值范围：[0, 4095] 数据类型：CVI_U16
DetailGaintOut	None 取值范围：[0, 4095] 数据类型：CVI_U16
DetailDetectLumaStr	None 取值范围：[0, 4095] 数据类型：CVI_U16
DetailSmoothStr	细节平滑强度。值越大，平滑强度越强，对伪细节的抑制强度越大。 取值范围：[0, 255] 数据类型：CVI_U8
DetailWgtThr	细节保留范围阈值。值越小，细节保留作用的范围越大。 取值范围：[0, 255] 数据类型：CVI_U8
DetailWgtMin	边缘细节平滑保留允许之最小增益。 取值范围：[0, 256] 数据类型：CVI_U16
DetailWgtMax	边缘细节平滑保留允许之最大增益。 取值范围：[0, 256] 数据类型：CVI_U16
DetailWgtSlope	细节保留强度。值越大，细节保留越多。 取值范围：[0, 1024] 数据类型：CVI_U16
EdgeWgtNp	None 取值范围：[0, 255] 数据类型：CVI_U8
EdgeWgtThr	细节保留范围阈值。值越小，细节保留作用的范围越大。 取值范围：[0, 255] 数据类型：CVI_U8
EdgeWgtMin	边缘细节平滑保留允许之最小增益。 取值范围：[0, 256] 数据类型：CVI_U16
EdgeWgtMax	边缘细节平滑保留允许之最大增益。 取值范围：[0, 256] 数据类型：CVI_U16
EdgeWgtSlope	细节保留强度。值越大，细节保留越多。 取值范围：[0, 1024] 数据类型：CVI_U16

下页继续

表 18.9 – 续上页

成员名称	描述
DetailSmoothMapTh	None 取值范围：[0, 255] 数据类型：CVI_U8
DetailSmoothMapMin	None 取值范围：[0, 256] 数据类型：CVI_U16
DetailSmoothMapMax	None 取值范围：[0, 256] 数据类型：CVI_U16
DetailSmoothMapSlope	None 取值范围：[0, 1024] 数据类型：CVI_U16
LumaWgt	None 取值范围：[0, 255] 数据类型：CVI_U8
SharpenGain	对 CFA 输出的 RGB 图施加的锐化强度，值越大，锐化越强，32 为 1x gain。 取值范围：[0, 255] 数据类型：CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetDemosaicDemoireAttr
- CVI_ISP_GetDemosaicDemoireAttr

18.3.6 ISP_DEMOSAIC_DEMOIRE_ATTR_S

【说明】

定义 Demosaic 去马赛克属性参数

【定义】

```
typedef struct _ISP_DEMOSAIC_DEMOIRE_ATTR_S {  
    CVI_BOOL AntiFalseColorEnable; /*Rw; Range:[0, 1]*/  
    CVI_BOOL ProtectColorEnable; /*Rw; Range:[0, 1]*/  
    CVI_BOOL DetailDetectLumaEnable; /*Rw; Range:[0, 1]*/  
    CVI_BOOL DetailSmoothEnable; /*Rw; Range:[0, 1]*/  
    CVI_BOOL DetailMode; /*Rw; Range:[0, 1]*/  
    ISP_DEMOSAIC_DEMOIRE_MANUAL_ATTR_S stManual;  
    ISP_DEMOSAIC_DEMOIRE_AUTO_ATTR_S stAuto;  
} ISP_DEMOSAIC_DEMOIRE_ATTR_S;
```

【成员】

成员名称	描述
AntiFalseColorEnable	None 取值范围：[0, 1] 数据类型：CVI_BOOL
ProtectColorEnable	None 取值范围：[0, 1] 数据类型：CVI_BOOL
DetailDetectLumaEnable	None 取值范围：[0, 1] 数据类型：CVI_BOOL
DetailSmoothEnable	None 取值范围：[0, 1] 数据类型：CVI_BOOL
DetailMode	None 取值范围：[0, 1] 数据类型：CVI_BOOL
stManual	手动参数
stAuto	自动参数

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetDemosaicDemoireAttr
- CVI_ISP_GetDemosaicDemoireAttr

19 SHARPEN

19.1 功能描述

此模块用于增强图像清晰度，位于 3DNR 之后，主要锐化图像中的大边缘。

19.2 API 参考

- `CVI_ISP_SetSharpenAttr`：设置 Sharpen 参数属性
- `CVI_ISP_GetSharpenAttr`：获取 Sharpen 参数属性

19.2.1 CVI_ISP_SetSharpenAttr

【描述】

设置 Sharpen 参数属性

【语法】

```
CVI_S32 CVI_ISP_SetSharpenAttr(VI_PIPE ViPipe, const ISP_SHARPEN_ATTR_S_
→*pstSharpenAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstSharpenAttr	Sharpen 参数属性	输入

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

```
VI_PIPE ViPipe = 0;
ISP_SHARPEN_ATTR_S stAttr;
CVI_ISP_GetSharpenAttr(ViPipe, &stAttr);
stAttr.enOpType = OP_TYPE_AUTO;
CVI_ISP_SetSharpenAttr(ViPipe, &stAttr);
```

【相关主题】

- [CVI_ISP_GetSharpenAttr](#)

19.2.2 CVI_ISP_GetSharpenAttr

【描述】

获取 Sharpen 参数属性

【语法】

```
CVI_S32 CVI_ISP_GetSharpenAttr(VI_PIPE ViPipe, ISP_SHARPEN_ATTR_S *pstSharpenAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstSharpenAttr	Sharpen 参数属性	输出

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

无

【相关主题】

- CVI_ISP_SetSharpenAttr

19.3 数据类型

- ISP_SHARPEN_MANUAL_ATTR_S：定义手动 Sharpen 参数属性
- ISP_SHARPEN_AUTO_ATTR_S：定义自动 Sharpen 参数属性
- ISP_SHARPEN_ATTR_S：定义 Sharpen 参数属性

19.3.1 ISP_SHARPEN_MANUAL_ATTR_S

【说明】

定义手动 Sharpen 参数属性

【定义】

```
typedef struct _ISP_SHARPEN_MANUAL_ATTR_S {  
    CVI_U8 EE1CoringThr; /*Rw; Range:[0, 255]*/  
    CVI_U8 NoiseBase; /*Rw; Range:[0, 255]*/  
    CVI_U8 EE2BalanceRatio; /*Rw; Range:[0, 255]*/  
    CVI_U8 EE2SharpStr; /*Rw; Range:[0, 255]*/  
    CVI_U8 EE2CoringThr; /*Rw; Range:[0, 255]*/  
    CVI_U8 OverShootRatio; /*Rw; Range:[0, 255]*/  
    CVI_U8 UnderShootRatio; /*Rw; Range:[0, 255]*/  
    CVI_U8 EE1BlendWgt; /*Rw; Range:[0, 8]*/  
    CVI_U16 EE1FlatThr; /*Rw; Range:[0, 2047]*/  
    CVI_U16 EE1EdgeThr; /*Rw; Range:[0, 2047]*/  
    CVI_U8 EE1FlatRegionStr; /*Rw; Range:[0, 255]*/  
    CVI_U8 EE1EdgeRegionStr; /*Rw; Range:[0, 255]*/  
    CVI_U8 EE1MotionStr; /*Rw; Range:[0, 255]*/  
    CVI_U8 EE1StaticStr; /*Rw; Range:[0, 255]*/  
    CVI_U8 EE1TransStr; /*Rw; Range:[0, 255]*/  
    CVI_U8 NoiseLut[17]; /*Rw; Range:[0, 255]*/  
    CVI_U8 HueShtCtrl[33]; /*Rw; Range:[0, 63]*/  
    CVI_U8 SatShtGainIn[4]; /*Rw; Range:[0, 255]*/  
    CVI_U8 SatShtGainOut[4]; /*Rw; Range:[0, 128]*/  
} ISP_SHARPEN_MANUAL_ATTR_S;
```

【成员】

成员名称	描述
EE1CoringThr	待完善 取值范围：[0, 255] 数据类型：CVI_U8
NoiseBase	待完善 取值范围：[0, 255] 数据类型：CVI_U8

下页继续

表 19.3 – 续上页

成员名称	描述
EE2BalanceRatio	待完善 取值范围：[0, 255] 数据类型：CVI_U8
EE2SharpStr	待完善 取值范围：[0, 255] 数据类型：CVI_U8
EE2CoringThr	待完善 取值范围：[0, 255] 数据类型：CVI_U8
OverShootRatio	白边抑制。值越小，白边抑制程度越大。 取值范围：[0, 255] 数据类型：CVI_U8
UnderShootRatio	黑边抑制。值越小，黑边抑制程度越大。 取值范围：[0, 255] 数据类型：CVI_U8
EE1BlendWgt	待完善 取值范围：[0, 8] 数据类型：CVI_U8
EE1FlatThr	待完善 取值范围：[0, 2047] 数据类型：CVI_U16
EE1EdgeThr	待完善 取值范围：[0, 2047] 数据类型：CVI_U16
EE1FlatRegionStr	待完善 取值范围：[0, 255] 数据类型：CVI_U8
EE1EdgeRegionStr	待完善 取值范围：[0, 255] 数据类型：CVI_U8
EE1MotionStr	待完善 取值范围：[0, 255] 数据类型：CVI_U8
EE1StaticStr	待完善 取值范围：[0, 255] 数据类型：CVI_U8
EE1TransStr	待完善 取值范围：[0, 255] 数据类型：CVI_U8
NoiseLut	噪声强度设定。根据亮度不同，设置不同的噪声强度。噪声强度越大，锐化强度越小。 取值范围：[0, 255] 数据类型：CVI_U8
HueShtCtrl	基于指定的色彩做边缘增强 取值范围：[0, 63] 数据类型：CVI_U8

下页继续

表 19.3 – 续上页

成员名称	描述
SatShtGainIn	基于指定的饱和度做边缘增强，此为输入节点，输入饱和度。 取值范围：[0, 255] 数据类型：CVI_U8
SatShtGainOut	基于指定的饱和度做边缘增强，此为输出节点，输出对应饱和度的边缘强度。 取值范围：[0, 128] 数据类型：CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetSharpenAttr
- CVI_ISP_GetSharpenAttr

19.3.2 ISP_SHARPEN_AUTO_ATTR_S

【说明】

定义自动 Sharpen 参数属性

【定义】

```
typedef struct ISP_SHARPEN_AUTO_ATTR_S {
    CVI_U8 EE1CoringThr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U8 NoiseBase[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U8 EE2BalanceRatio[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U8 EE2SharpStr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U8 EE2CoringThr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U8 OverShootRatio[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U8 UnderShootRatio[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U8 EE1BlendWgt[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 8]*/
    CVI_U16 EE1FlatThr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 2047]*/
    CVI_U16 EE1EdgeThr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 2047]*/
    CVI_U8 EE1FlatRegionStr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U8 EE1EdgeRegionStr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U8 EE1MotionStr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U8 EE1StaticStr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U8 EE1TransStr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U8 NoiseLut[17][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U8 HueShtCtrl[33][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 63]*/
    CVI_U8 SatShtGainIn[4][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U8 SatShtGainOut[4][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 128]*/
} ISP_SHARPEN_AUTO_ATTR_S;
```

【成员】

成员名称	描述
EE1CoringThr	待完善 取值范围：[0, 255] 数据类型：CVI_U8
NoiseBase	待完善 取值范围：[0, 255] 数据类型：CVI_U8
EE2BalanceRatio	待完善 取值范围：[0, 255] 数据类型：CVI_U8
EE2SharpStr	待完善 取值范围：[0, 255] 数据类型：CVI_U8
EE2CoringThr	待完善 取值范围：[0, 255] 数据类型：CVI_U8
OverShootRatio	白边抑制。值越小，白边抑制程度越大。 取值范围：[0, 255] 数据类型：CVI_U8
UnderShootRatio	黑边抑制。值越小，黑边抑制程度越大。 取值范围：[0, 255] 数据类型：CVI_U8
EE1BlendWgt	待完善 取值范围：[0, 8] 数据类型：CVI_U8
EE1FlatThr	待完善 取值范围：[0, 2047] 数据类型：CVI_U16
EE1EdgeThr	待完善 取值范围：[0, 2047] 数据类型：CVI_U16
EE1FlatRegionStr	待完善 取值范围：[0, 255] 数据类型：CVI_U8
EE1EdgeRegionStr	待完善 取值范围：[0, 255] 数据类型：CVI_U8
EE1MotionStr	待完善 取值范围：[0, 255] 数据类型：CVI_U8
EE1StaticStr	待完善 取值范围：[0, 255] 数据类型：CVI_U8
EE1TransStr	待完善 取值范围：[0, 255] 数据类型：CVI_U8

下页继续

表 19.4 – 续上页

成员名称	描述
NoiseLut	噪声强度设定。根据亮度不同，设置不同的噪声强度。噪声强度越大，锐化强度越小。 取值范围：[0, 255] 数据类型：CVI_U8
HueShtCtrl	基于指定的色彩做边缘增强 取值范围：[0, 63] 数据类型：CVI_U8
SatShtGainIn	基于指定的饱和度做边缘增强，此为输入节点，输入饱和度。 取值范围：[0, 255] 数据类型：CVI_U8
SatShtGainOut	基于指定的饱和度做边缘增强，此为输出节点，输出对应饱和度的边缘强度。 取值范围：[0, 128] 数据类型：CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetSharpenAttr
- CVI_ISP_GetSharpenAttr

19.3.3 ISP_SHARPEN_ATTR_S

【说明】

定义 Sharpen 参数属性

【定义】

```
typedef struct ISP_SHARPEN_ATTR_S {  
    CVI_BOOL Enable; /*Rw; Range:[0, 1]*/  
    ISP_OP_TYPE_E enOpType;  
    CVI_U8 UpdateInterval; /*Rw; Range:[1, 255]*/  
    CVI_U8 TuningMode; /*Rw; Range:[0, 7]*/  
    CVI_BOOL MotionEn; /*Rw; Range:[0, 1]*/  
    CVI_BOOL SatShtCtrlEn; /*Rw; Range:[0, 1]*/  
    ISP_EE_FILTER_SIZE_TYPE_E enFilterType;  
    ISP_EE_WEIGHT_SRC_TYPE_E EE1SrcType;  
    ISP_SHARPEN_MANUAL_ATTR_S stManual;  
    ISP_SHARPEN_AUTO_ATTR_S stAuto;  
} ISP_SHARPEN_ATTR_S;
```

【成员】

成员名称	描述
Enable	模块使能开关 取值范围：[0, 1] 数据类型：CVI_BOOL
enOpType	工作类型 OP_TYPE_AUTO：自动模式 OP_TYPE_MANUAL：手动模式 数据类型：ISP_OP_TYPE_E
UpdateInterval	影响参数更新间隔，值越大画面变化越慢，效能越好。 取值范围：[1, 255] 数据类型：CVI_U8
TuningMode	输出调试方案，辅助调节参数 0=normal; 1=EdgeEnergy; 2=EdgeWeight; 3=Enhance; 4=ReguinClass; 5=Edgevalue, 6=EdgeNoRefine 取值范围：[0, 7] 数据类型：CVI_U8
MotionEn	根据运动状态不同，调整锐化强度的使能开关 取值范围：[0, 1] 数据类型：CVI_BOOL
SatShtCtrlEn	由饱和度调整边缘增强的使能 取值范围：[0, 1] 数据类型：CVI_BOOL
enFilterType	Filter SIZE 类型 OP_TYPE_3x3: 3x3 核; OP_TYPE_5x5: 5x5 核; 数据类型：ISP_EE_FILTER_SIZE_TYPE_E
EE1SrcType	WEIGHT SRC 类型 数据类型：ISP_EE_WEIGHT_SRC_TYPE_E
stManual	手动参数
stAuto	自动参数

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetSharpenAttr
- CVI_ISP_GetSharpenAttr

20 PRESARPEN

20.1 功能描述

此模块用于增强图像清晰度，位于 3DNR 之前，主要锐化图像中具方向性的边缘和细节纹理。

20.2 API 参考

- `CVI_ISP_SetPreSharpenAttr`：设置 PreSharpen 参数属性
- `CVI_ISP_GetPreSharpenAttr`：获取 PreSharpen 参数属性
- `CVI_ISP_SetPreSharpenRefineAttr`：设置 PreSharpen Refine 参数属性
- `CVI_ISP_GetPreSharpenRefineAttr`：获取 PreSharpen Refine 参数属性
- `CVI_ISP_SetPreSharpenEdgeExtAttr`：设置 PreSharpen Edge 参数属性
- `CVI_ISP_GetPreSharpenEdgeExtAttr`：获取 PreSharpen Edge 参数属性

20.2.1 CVI_ISP_SetPreSharpenAttr

【描述】

设置 PreSharpen 参数属性

【语法】

```
CVI_S32 CVI_ISP_SetPreSharpenAttr(VI_PIPE ViPipe, const ISP_PRESHARPEN_ATTR_S_
→*pstPreSharpenAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstPreSharpenAttr	PreSharpen 参数属性	输入

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

```
VI_PIPE ViPipe = 0;
ISP_PRESHARPEN_ATTR_S stAttr;
CVI_ISP_GetPreSharpenAttr(ViPipe, &stAttr);
stAttr.enOpType = OP_TYPE_AUTO;
CVI_ISP_SetPreSharpenAttr(ViPipe, &stAttr);
```

【相关主题】

- [CVI_ISP_GetPreSharpenAttr](#)

20.2.2 CVI_ISP_GetPreSharpenAttr

【描述】

获取 PreSharpen 参数属性

【语法】

```
CVI_S32 CVI_ISP_GetPreSharpenAttr(VI_PIPE ViPipe, ISP_PRESHARPEN_ATTR_S_
↪ *pstPreSharpenAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstPreSharpenAttr	PreSharpen 参数属性	输出

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`

- 库文件: libisp.so

【注意】

无

【举例】

无

【相关主题】

- [CVI_ISP_SetPreSharpenAttr](#)

20.2.3 CVI_ISP_SetPreSharpenRefineAttr

【描述】

设置 PreSharpen Refine 参数属性

【语法】

```
CVI_S32 CVI_ISP_SetPreSharpenRefineAttr(VI_PIPE ViPipe, const ISP_PRESHARPEN_
→REFINE_ATTR_S *pstPreSharpenRefineAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstPreSharpenRe- fineAttr	PreSharpen Refine 参数属性	输入

【返回值】

返回值	描述
0	成功
非 0	失败, 其值为错误码。

【需求】

- 头文件: cvi_isp.h, cvi_comm_isp.h
- 库文件: libisp.so

【注意】

无

【举例】

```
VI_PIPE ViPipe = 0;
ISP_PRESHARPEN_REFINE_ATTR_S stAttr;
CVI_ISP_GetPreSharpenRefineAttr(ViPipe, &stAttr);
stAttr.enOpType = OP_TYPE_AUTO;
CVI_ISP_SetPreSharpenRefineAttr(ViPipe, &stAttr);
```

【相关主题】

- [CVI_ISP_GetPreSharpenRefineAttr](#)

20.2.4 CVI_ISP_GetPreSharpenRefineAttr

【描述】

获取 PreSharpen Refine 参数属性

【语法】

```
CVI_S32 CVI_ISP_GetPreSharpenRefineAttr(VI_PIPE ViPipe, ISP_PRESHARPEN_REFINE_
↪ATTR_S *pstPreSharpenRefineAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstPreSharpenRe- fineAttr	PreSharpen Refine 参数属性	输出

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

无

【相关主题】

- [CVI_ISP_SetPreSharpenRefineAttr](#)

20.2.5 CVI_ISP_SetPreSharpenEdgeExtAttr

【描述】

设置 PreSharpen Edge 参数属性

【语法】

```
CVI_S32 CVI_ISP_SetPreSharpenEdgeExtAttr(VI_PIPE ViPipe, const ISP_PRESHARPEN_
↪EDGE_EXT_ATTR_S *pstPreSharpenEdgeExtAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstPreSharpenEdgeExtAttr	PreSharpen Edge 参数属性	输入

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: cvi_isp.h, cvi_comm_isp.h
- 库文件: libisp.so

【注意】

无

【举例】

```
VI_PIPE ViPipe = 0;
ISP_PRESHARPEN_EDGE_EXT_ATTR_S stAttr;
CVI_ISP_GetPreSharpenEdgeExtAttr(ViPipe, &stAttr);
stAttr.enOpType = OP_TYPE_AUTO;
CVI_ISP_SetPreSharpenEdgeExtAttr(ViPipe, &stAttr);
```

【相关主题】

- [CVI_ISP_GetPreSharpenEdgeExtAttr](#)

20.2.6 CVI_ISP_GetPreSharpenEdgeExtAttr

【描述】

获取 PreSharpen Edge 参数属性

【语法】

```
CVI_S32 CVI_ISP_GetPreSharpenEdgeExtAttr(VI_PIPE ViPipe, ISP_PRESHARPEN_EDGE_
→EXT_ATTR_S *pstPreSharpenEdgeExtAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstPreSharpenEdgeExtAttr	PreSharpen Edge 参数属性	输出

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: cvi_isp.h, cvi_comm_isp.h
- 库文件: libisp.so

【注意】

无

【举例】

无

【相关主题】

- CVI_ISP_SetPreSharpenEdgeExtAttr

20.3 数据类型

- ISP_PRESHARPEN_MANUAL_ATTR_S：定义手动 PreSharpen 参数属性
- ISP_PRESHARPEN_AUTO_ATTR_S：定义自动 PreSharpen 参数属性
- ISP_PRESHARPEN_ATTR_S：定义 PreSharpen 参数属性
- ISP_PRESHARPEN_REFINE_MANUAL_ATTR_S：定义手动 PreSharpen Refine 参数属性

- `ISP_PRESHARPEN_REFINE_AUTO_ATTR_S`：定义自动 PreSharpen Refine 参数属性
- `ISP_PRESHARPEN_REFINE_ATTR_S`：定义 PreSharpen Refine 参数属性
- `ISP_PRESHARPEN_EDGE_EXT_MANUAL_ATTR_S`：定义手动 PreSharpen Edge 参数属性
- `ISP_PRESHARPEN_EDGE_EXT_AUTO_ATTR_S`：定义自动 PreSharpen Edge 参数属性
- `ISP_PRESHARPEN_EDGE_EXT_ATTR_S`：定义 PreSharpen Edge 参数属性

20.3.1 ISP_PRESHARPEN_MANUAL_ATTR_S

【说明】

定义手动 PreSharpen 参数属性

【定义】

```
typedef struct ISP_PRESHARPEN_MANUAL_ATTR_S {
    CVI_U16 FlatThr; /*Rw; Range:[0, 1023]*/
    CVI_U16 EdgeThr; /*Rw; Range:[0, 1023]*/
    CVI_U16 FlatThrHLD; /*Rw; Range:[0, 1023]*/
    CVI_U16 EdgeThrHLD; /*Rw; Range:[0, 1023]*/
    CVI_U16 LumaThrHLD; /*Rw; Range:[0, 1023]*/
    CVI_U8 ThinStr; /*Rw; Range:[0, 255]*/
    CVI_U8 RobustStr; /*Rw; Range:[0, 255]*/
    CVI_U8 FlatStr; /*Rw; Range:[0, 255]*/
    CVI_U8 EdgeStr; /*Rw; Range:[0, 255]*/
    CVI_S16 StrFlatSlope; /*Rw; Range:[-32768, 32767]*/
    CVI_S16 StrEdgeSlope; /*Rw; Range:[-32768, 32767]*/
    CVI_U8 ConEngBlendWt; /*Rw; Range:[0, 15]*/
    CVI_U16 EdgeLowEndThr; /*Rw; Range:[0, 1023]*/
    CVI_U16 EdgeMidEndThr; /*Rw; Range:[0, 1023]*/
    CVI_U16 EdgeHighEndThr; /*Rw; Range:[0, 1023]*/
    CVI_U8 HueShtCtrl[33]; /*Rw; Range:[0, 63]*/
    CVI_U8 SatShtGainIn[4]; /*Rw; Range:[0, 255]*/
    CVI_U8 SatShtGainOut[4]; /*Rw; Range:[0, 128]*/
} ISP_PRESHARPEN_MANUAL_ATTR_S;
```

【成员】

成员名称	描述
FlatThr	平坦区判定阈值。 取值范围：[0, 1023] 数据类型：CVI_U16
EdgeThr	边缘区判定阈值。 取值范围：[0, 1023] 数据类型：CVI_U16

下页继续

表 20.7 – 续上页

成员名称	描述
FlatThrHLD	高亮区域的平坦区判定阈值。 取值范围：[0, 1023] 数据类型：CVI_U16
EdgeThrHLD	高亮区域的边缘区判定阈值。 取值范围：[0, 1023] 数据类型：CVI_U16
LumaThrHLD	高亮区判定阈值，亮度大于 LumaThrHLD 的部分，认为是高亮区。 取值范围：[0, 1023] 数据类型：CVI_U16
ThinStr	Thin filter 强度调整。RegionStrEn = 1 时，生效。值越大，强度越强；值越小，强度越小。 取值范围：[0, 255] 数据类型：CVI_U8
RobustStr	Robust filter 强度调整。RegionStrEn = 1 时，生效。值越大，强度越强；值越小，强度越小。 取值范围：[0, 255] 数据类型：CVI_U8
FlatStr	限制平坦区锐化强度范围。当 StrFlatSlope < 0 时，决定平坦区锐化强度的最小值；当 StrFlatSlope > 0 时，决定平坦区锐化强度的最大值。 取值范围：[0, 255] 数据类型：CVI_U8
EdgeStr	限制边缘区锐化强度范围。当 StrEdgeSlope > 0 时，决定边缘区锐化强度的最大值；当 StrFlatSlope < 0 时，决定边缘区锐化强度的最小值。 取值范围：[0, 255] 数据类型：CVI_U8
StrFlatSlope	控制平坦区锐化强度。RegionStrEn = 1 时，生效。值越小，平坦区锐化越弱。 取值范围：[-32768, 32767] 数据类型：CVI_S16
StrEdgeSlope	控制边缘区锐化强度。RegionStrEn = 1 时，生效。值越大，边缘区锐化越强。 取值范围：[-32768, 32767] 数据类型：CVI_S16
ConEngBlendWt	weighting between edge energy and contrast. 0: all energy; 8: all contrast 取值范围：[0, 15] 数据类型：CVI_U8
EdgeLowEndThr	edge intensity thres at low-end 取值范围：[0, 1023] 数据类型：CVI_U16
EdgeMidEndThr	edge intensity thres at mid-end 取值范围：[0, 1023] 数据类型：CVI_U16

下页继续

表 20.7 – 续上页

成员名称	描述
EdgeHighEndThr	edge intensity thres at high-end 取值范围：[0, 1023] 数据类型：CVI_U16
HueShtCtrl	基于指定的色彩做边缘增强 取值范围：[0, 63] 数据类型：CVI_U8
SatShtGainIn	基于指定的饱和度做边缘增强，此为输入节点，输入饱和度 取值范围：[0, 255] 数据类型：CVI_U8
SatShtGainOut	基于指定的饱和度做边缘增强，此为输出节点，输出对应饱和度的边缘强度。(max = 128) 取值范围：[0, 128] 数据类型：CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetPreSharpenAttr
- CVI_ISP_GetPreSharpenAttr

20.3.2 ISP_PRESHARPEN_AUTO_ATTR_S

【说明】

定义自动 PreSharpen 参数属性

【定义】

```
typedef struct _ISP_PRESHARPEN_AUTO_ATTR_S {
    CVI_U16 FlatThr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 1023]*/
    CVI_U16 EdgeThr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 1023]*/
    CVI_U16 FlatThrHLD[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 1023]*/
    CVI_U16 EdgeThrHLD[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 1023]*/
    CVI_U16 LumaThrHLD[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 1023]*/
    CVI_U8 ThinStr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U8 RobustStr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U8 FlatStr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U8 EdgeStr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_S16 StrFlatSlope[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[-32768, 32767]*/
    CVI_S16 StrEdgeSlope[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[-32768, 32767]*/
    CVI_U8 ConEngBlendWt[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 15]*/
    CVI_U16 EdgeLowEndThr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 1023]*/
    CVI_U16 EdgeMidEndThr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 1023]*/
    CVI_U16 EdgeHighEndThr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 1023]*/
    CVI_U8 HueShtCtrl[33][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 63]*/
    CVI_U8 SatShtGainIn[4][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U8 SatShtGainOut[4][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 128]*/
} ISP_PRESHARPEN_AUTO_ATTR_S;
```

【成员】

成员名称	描述
FlatThr	平坦区判定阈值。 取值范围：[0, 1023] 数据类型：CVI_U16
EdgeThr	边缘区判定阈值。 取值范围：[0, 1023] 数据类型：CVI_U16
FlatThrHLD	高亮区域的平坦区判定阈值。 取值范围：[0, 1023] 数据类型：CVI_U16
EdgeThrHLD	高亮区域的边缘区判定阈值。 取值范围：[0, 1023] 数据类型：CVI_U16
LumaThrHLD	高亮区判定阈值，亮度大于 LumaThrHLD 的部分，认为是高亮区。 取值范围：[0, 1023] 数据类型：CVI_U16
ThinStr	Thin filter 强度调整。RegionStrEn = 1 时，生效。值越大，强度越强；值越小，强度越小。 取值范围：[0, 255] 数据类型：CVI_U8
RobustStr	Robust filter 强度调整。RegionStrEn = 1 时，生效。值越大，强度越强；值越小，强度越小。 取值范围：[0, 255] 数据类型：CVI_U8
FlatStr	限制平坦区锐化强度范围。当 StrFlatSlope < 0 时，决定平坦区锐化强度的最小值；当 StrFlatSlope > 0 时，决定平坦区锐化强度的最大值。 取值范围：[0, 255] 数据类型：CVI_U8
EdgeStr	限制边缘区锐化强度范围。当 StrEdgeSlope > 0 时，决定边缘区锐化强度的最大值；当 StrFlatSlope < 0 时，决定边缘区锐化强度的最小值。 取值范围：[0, 255] 数据类型：CVI_U8
StrFlatSlope	控制平坦区锐化强度。RegionStrEn = 1 时，生效。值越小，平坦区锐化越弱。 取值范围：[-32768, 32767] 数据类型：CVI_S16
StrEdgeSlope	控制边缘区锐化强度。RegionStrEn = 1 时，生效。值越大，边缘区锐化越强。 取值范围：[-32768, 32767] 数据类型：CVI_S16

下页继续

表 20.8 – 续上页

成员名称	描述
ConEngBlendWt	weighting between edge energy and contrast. 0: all energy; 8: all contrast 取值范围: [0, 15] 数据类型: CVI_U8
EdgeLowEndThr	edge intensity thres at low-end 取值范围: [0, 1023] 数据类型: CVI_U16
EdgeMidEndThr	edge intensity thres at mid-end 取值范围: [0, 1023] 数据类型: CVI_U16
EdgeHighEndThr	edge intensity thres at high-end 取值范围: [0, 1023] 数据类型: CVI_U16
HueShtCtrl	基于指定的色彩做边缘增强 取值范围: [0, 63] 数据类型: CVI_U8
SatShtGainIn	基于指定的饱和度做边缘增强, 此为输入节点, 输入饱和度 取值范围: [0, 255] 数据类型: CVI_U8
SatShtGainOut	基于指定的饱和度做边缘增强, 此为输出节点, 输出对应饱和度的边缘强度。(max = 128) 取值范围: [0, 128] 数据类型: CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetPreSharpenAttr
- CVI_ISP_GetPreSharpenAttr

20.3.3 ISP_PRESHARPEN_ATTR_S**【说明】**

定义 PreSharpen 参数属性

【定义】

```
typedef struct _ISP_PRESHARPEN_ATTR_S {
    CVI_BOOL Enable; /*Rw; Range:[0, 1]*/
    ISP_OP_TYPE_E enOpType;
    CVI_U8 UpdateInterval; /*Rw; Range:[1, 255]*/
    CVI_U8 TuningMode; /*Rw; Range:[0, 15]*/
    CVI_BOOL GammaSelection; /*Rw; Range:[0, 1]*/
    CVI_BOOL ChnSelection; /*Rw; Range:[0, 1]*/
    CVI_BOOL ParamAutoEn; /*Rw; Range:[0, 1]*/
}
```

(下页继续)

(续上页)

```

CVI_BOOL SatShtCtrlEn; /*Rw; Range:[0, 1]*/
CVI_BOOL EdgeOutFormat; /*Rw; Range:[0, 1]*/
CVI_U8 ReduceByEvRatioThr; /*Rw; Range:[64, 255]*/
CVI_U8 ReduceByEvRatioStr; /*Rw; Range:[0, 255]*/
ISP_PRESHARPEN_MANUAL_ATTR_S stManual;
ISP_PRESHARPEN_AUTO_ATTR_S stAuto;
} ISP_PRESHARPEN_ATTR_S;

```

【成员】

成员名称	描述
Enable	模块使能开关 取值范围：[0, 1] 数据类型：CVI_BOOL
enOpType	工作类型 OP_TYPE_AUTO：自动模式 OP_TYPE_MANUAL：手动模式 数据类型：ISP_OP_TYPE_E
UpdateInterval	影响参数更新间隔，值越大画面变化越慢，效能越好。 取值范围：[1, 255] 数据类型：CVI_U8
TuningMode	调整模式，输出可视化辅助信息。 0 以及 9~15：不输出可视化辅助信息。 1: Ultra HF 滤波结果。 2: HF 滤波结果。 3: Subtle HF 滤波结果。 4: MF 滤波结果。 5: HF 滤波结果。 6: Subtle HF 滤波结果。 取值范围：[0, 15] 数据类型：CVI_U8
GammaSelection	0: ori lum; 1: gamma lum 取值范围：[0, 1] 数据类型：CVI_BOOL
ChnSelection	Select use RGB to Y or G channel as the input. 0: G channel edge extraction; 1: Y channel edge extraction 取值范围：[0, 1] 数据类型：CVI_BOOL
ParamAutoEn	auto mode: auto determine thTexture, thEdge, thTexture_HLD, thEdge_HLD, thLum_HLD, th_overshoot, th_undershoot_lum, th_undershoot_eng 取值范围：[0, 1] 数据类型：CVI_BOOL
SatShtCtrlEn	由饱和度调整边缘增强的使能。 取值范围：[0, 1] 数据类型：CVI_BOOL

下页继续

表 20.9 – 续上页

成员名称	描述
EdgeOutFormat	edge threshold format. 0: flag; 1, real 8bit 取值范围: [0, 1] 数据类型: CVI_BOOL
ReduceByEvRatioThr	待完善 取值范围: [64, 255] 数据类型: CVI_U8
ReduceByEvRatioStr	待完善 取值范围: [0, 255] 数据类型: CVI_U8
stManual	手动参数
stAuto	自动参数

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetPreSharpenAttr
- CVI_ISP_GetPreSharpenAttr

20.3.4 ISP_PRESHARPEN_REFINE_MANUAL_ATTR_S

【说明】

定义手动 PreSharpen Refine 参数属性

【定义】

```
typedef struct _ISP_PRESHARPEN_REFINE_MANUAL_ATTR_S {
    CVI_U16 LumaLowThr; /*Rw; Range:[0, 1023]*/
    CVI_U16 LumaHighThr; /*Rw; Range:[0, 1023]*/
    CVI_U8 LumaLowStep; /*Rw; Range:[0, 7]*/
    CVI_U8 LumaHighStep; /*Rw; Range:[0, 7]*/
    CVI_U8 LumaLowLut[8]; /*Rw; Range:[0, 255]*/
    CVI_U8 LumaHighLut[8]; /*Rw; Range:[0, 255]*/
    CVI_U16 EdgeLowThr; /*Rw; Range:[0, 1023]*/
    CVI_U16 EdgeHighThr; /*Rw; Range:[0, 1023]*/
    CVI_U8 EdgeLowStep; /*Rw; Range:[0, 7]*/
    CVI_U8 EdgeHighStep; /*Rw; Range:[0, 7]*/
    CVI_U8 EdgeLowLut[8]; /*Rw; Range:[0, 255]*/
    CVI_U8 EdgeHighLut[8]; /*Rw; Range:[0, 255]*/
    CVI_U8 OvershootThr; /*Rw; Range:[0, 255]*/
    CVI_U16 OshootSlope; /*Rw; Range:[0, 32767]*/
    CVI_U16 OshootWgtHigh; /*Rw; Range:[0, 255]*/
    CVI_U8 UndershootThr; /*Rw; Range:[0, 255]*/
    CVI_U8 UshootSlope; /*Rw; Range:[0, 255]*/
    CVI_U8 UshootWgtHigh; /*Rw; Range:[0, 255]*/
    CVI_U8 UshootLumThrLow; /*Rw; Range:[0, 255]*/
    CVI_U8 UshootLumThrHigh; /*Rw; Range:[0, 255]*/
}
```

(下页继续)

(续上页)

```

    CVI_U8 UshootLumClampWgt; /*Rw; Range:[0, 255]*/
    CVI_U8 UshootEngThrLow; /*Rw; Range:[0, 255]*/
    CVI_U8 UshootEngThrHigh; /*Rw; Range:[0, 255]*/
    CVI_U8 UshootEngClampWgt; /*Rw; Range:[0, 255]*/
    CVI_U8 UshootLumEngStr; /*Rw; Range:[0, 255]*/
    CVI_U8 UshootLumEngNorm; /*Rw; Range:[0, 255]*/
} ISP_PRESHARPEN_REFINE_MANUAL_ATTR_S;

```

【成员】

成员名称	描述
LumaLowThr	亮度阈值。亮度小于 LumLowThr 的部分，锐化强度为 LumLowLut[0]。 取值范围：[0, 1023] 数据类型：CVI_U16
LumaHighThr	亮度阈值。亮度大于 LumHighThr 的部分，锐化强度按照 LumHighLut[0]~LumHighLut[7] 的设定逐渐变化。 取值范围：[0, 1023] 数据类型：CVI_U16
LumaLowStep	控制锐度随亮度变化时，亮度上的步幅。亮度大于 LumLowThr 且小于 LumHighThr 的部分，步幅由 LumLowStep 控制。 取值范围：[0, 7] 数据类型：CVI_U8
LumaHighStep	控制锐度随亮度变化时，亮度上的步幅。亮度大于 LumHighThr 的部分，步幅由 LumHighStep 控制。 取值范围：[0, 7] 数据类型：CVI_U8
LumaLowLut	控制锐度强度。亮度大于 LumLowThr 且小于 LumHighThr 的部分，随亮度增加，锐度强度按照 LumLowLut[0]~LumLowLut[7] 的设定逐渐变化。 取值范围：[0, 255] 数据类型：CVI_U8
LumaHighLut	控制锐度强度。亮度大于 LumHighThr 的部分，锐化强度按照 LumHighLut[0]~LumHighLut[7] 的设定逐渐变化。 取值范围：[0, 255] 数据类型：CVI_U8
EdgeLowThr	锐化幅度阈值。锐化幅度小于 EdgeLowThr 的部分，锐化强度为 EdgeLowLut[0]。 取值范围：[0, 1023] 数据类型：CVI_U16
EdgeHighThr	锐化幅度阈值。锐化幅度大于 EdgeHighThr 的部分，锐化强度按照 EdgeHighLut[0]~EdgeHighLut[7] 的设定逐渐变化。 取值范围：[0, 1023] 数据类型：CVI_U16

下页继续

表 20.10 – 续上页

成员名称	描述
EdgeLowStep	控制锐化强度随锐化幅度变化时，锐化幅度上的步幅。锐化幅度大于 EdgeLowThr 且小于 EdgeHighThr 的部分，步幅由 EdgeLowStep 控制。 取值范围：[0, 7] 数据类型：CVI_U8
EdgeHighStep	控制锐化强度随锐化幅度变化时，锐化幅度上的步幅。锐化幅度大于 EdgeHighThr 的部分，步幅由 EdgeHighStep 控制。 取值范围：[0, 7] 数据类型：CVI_U8
EdgeLowLut	控制锐化强度。锐化幅度大于 EdgeLowThr 且小于 EdgeHighThr 的部分，随锐化幅度增加，锐度强度按照 EdgeLowLut[0]~EdgeLowLut[7] 的设定逐渐变化。 取值范围：[0, 255] 数据类型：CVI_U8
EdgeHighLut	控制锐化强度。锐化幅度大于 EdgeHighThr 的部分，锐化强度按照 EdgeHighLut[0]~EdgeHighLut[7] 的设定逐渐变化。 取值范围：[0, 255] 数据类型：CVI_U8
OvershootThr	亮度阈值。亮度小于 OvershootThr 的部分，白边的强度由 OshootWgtHigh 决定。 取值范围：[0, 255] 数据类型：CVI_U8
OshootSlope	决定亮度大于 OvershootThr 部分的白边强度。值越大，随着亮度增加，白边强度降低越快。 取值范围：[0, 32767] 数据类型：CVI_U16
OshootWgtHigh	决定亮度小于 OvershootThr 部分的白边强度。值越大，白边强度越强。 取值范围：[0, 255] 数据类型：CVI_U16
UndershootThr	亮度阈值。亮度大于 UndershootThr 的部分，黑边的强度由 UshootWgtHigh 决定。 取值范围：[0, 255] 数据类型：CVI_U8
UshootSlope	决定亮度小于 UndershootThr 部分的黑边强度。值越大，随着亮度降低，黑边强度降低越快。 取值范围：[0, 255] 数据类型：CVI_U8
UshootWgtHigh	决定亮度大于 UndershootThr 部分的黑边强度。值越大，黑边强度越大。 取值范围：[0, 255] 数据类型：CVI_U8
UshootLumThrLow	亮度阈值。亮度小于 UshootLumThrLow 的部分，黑边对应的强度为 0。 取值范围：[0, 255] 数据类型：CVI_U8

下页继续

表 20.10 – 续上页

成员名称	描述
UshootLumThrHigh	亮度阈值。亮度大于 UshootLumThrHigh 的部分，黑边对应的强度由 UshootLumClampWgt 决定。 取值范围：[0, 255] 数据类型：CVI_U8
UshootLumClampWgt	决定亮度大于 UshootLumThrHigh 部分的黑边强度。值越大，黑边强度越强。 取值范围：[0, 255] 数据类型：CVI_U8
UshootEngThrLow	特征值阈值。特征值小于 UshootEngThrLow 的部分，黑边的强度由 UshootEngClampWgt 决定。 取值范围：[0, 255] 数据类型：CVI_U8
UshootEngThrHigh	特征值阈值。特征值大于 UshootEngThrHigh 的部分，黑边的强度由 UshootEngClampWgt 决定。 取值范围：[0, 255] 数据类型：CVI_U8
UshootEngClampWgt	决定特征值大于 UshootEngThrHigh 部分的黑边强度。值越大，黑边的强度于越强。 取值范围：[0, 255] 数据类型：CVI_U8
UshootLumEngStr	决定黑边强度偏移量的放大倍数。值越大，放大倍数越大。 取值范围：[0, 255] 数据类型：CVI_U8
UshootLumEngNorm	决定黑边强度偏移量的缩小倍数。值越大，缩小倍数越大。 取值范围：[0, 255] 数据类型：CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetPreSharpenRefineAttr
- CVI_ISP_GetPreSharpenRefineAttr

20.3.5 ISP_PRESHARPEN_REFINE_AUTO_ATTR_S

【说明】

定义自动 PreSharpen Refine 参数属性

【定义】

```
typedef struct _ISP_PRESHARPEN_REFINE_AUTO_ATTR_S {
    CVI_U16 LumaLowThr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 1023]*/
    CVI_U16 LumaHighThr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 1023]*/
    CVI_U8 LumaLowStep[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 7]*/
}
```

(下页继续)

(续上页)

```

CVI_U8 LumaHighStep[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 7]*/
CVI_U8 LumaLowLut[8][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
CVI_U8 LumaHighLut[8][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
CVI_U16 EdgeLowThr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 1023]*/
CVI_U16 EdgeHighThr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 1023]*/
CVI_U8 EdgeLowStep[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 7]*/
CVI_U8 EdgeHighStep[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 7]*/
CVI_U8 EdgeLowLut[8][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
CVI_U8 EdgeHighLut[8][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
CVI_U8 OvershootThr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
CVI_U16 OshootSlope[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 32767]*/
CVI_U16 OshootWgtHigh[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
CVI_U8 UndershootThr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
CVI_U8 UshootSlope[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
CVI_U8 UshootWgtHigh[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
CVI_U8 UshootLumThrLow[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
CVI_U8 UshootLumThrHigh[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
CVI_U8 UshootLumClampWgt[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
CVI_U8 UshootEngThrLow[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
CVI_U8 UshootEngThrHigh[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
CVI_U8 UshootEngClampWgt[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
CVI_U8 UshootLumEngStr[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
CVI_U8 UshootLumEngNorm[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
} ISP_PRESHARPEN_REFINE_AUTO_ATTR_S;

```

【成员】

成员名称	描述
LumaLowThr	亮度阈值。亮度小于 LumLowThr 的部分，锐化强度为 LumLowLut[0]。 取值范围：[0, 1023] 数据类型：CVI_U16
LumaHighThr	亮度阈值。亮度大于 LumHighThr 的部分，锐化强度按照 LumHighLut[0]~LumHighLut[7] 的设定逐渐变化。 取值范围：[0, 1023] 数据类型：CVI_U16
LumaLowStep	控制锐度随亮度变化时，亮度上的步幅。亮度大于 LumLowThr 且小于 LumHighThr 的部分，步幅由 LumLowStep 控制。 取值范围：[0, 7] 数据类型：CVI_U8
LumaHighStep	控制锐度随亮度变化时，亮度上的步幅。亮度大于 LumHighThr 的部分，步幅由 LumHighStep 控制。 取值范围：[0, 7] 数据类型：CVI_U8
LumaLowLut	控制锐度强度。亮度大于 LumLowThr 且小于 LumHighThr 的部分，随亮度增加，锐度强度按照 LumLowLut[0]~LumLowLut[7] 的设定逐渐变化。 取值范围：[0, 255] 数据类型：CVI_U8

下页继续

表 20.11 – 续上页

成员名称	描述
LumaHighLut	控制锐度强度。亮度大于 LumHighThr 的部分，锐化强度按照 LumHighLut[0]~LumHighLut[7] 的设定逐渐变化。 取值范围：[0, 255] 数据类型：CVI_U8
EdgeLowThr	锐化幅度阈值。锐化幅度小于 EdgeLowThr 的部分，锐化强度为 EdgeLowLut[0]。 取值范围：[0, 1023] 数据类型：CVI_U16
EdgeHighThr	锐化幅度阈值。锐化幅度大于 EdgeHighThr 的部分，锐化强度按照 EdgeHighLut[0]~EdgeHighLut[7] 的设定逐渐变化。 取值范围：[0, 1023] 数据类型：CVI_U16
EdgeLowStep	控制锐化强度随锐化幅度变化时，锐化幅度上的步幅。锐化幅度大于 EdgeLowThr 且小于 EdgeHighThr 的部分，步幅由 EdgeLowStep 控制。 取值范围：[0, 7] 数据类型：CVI_U8
EdgeHighStep	控制锐化强度随锐化幅度变化时，锐化幅度上的步幅。锐化幅度大于 EdgeHighThr 的部分，步幅由 EdgeHighStep 控制。 取值范围：[0, 7] 数据类型：CVI_U8
EdgeLowLut	控制锐化强度。锐化幅度大于 EdgeLowThr 且小于 EdgeHighThr 的部分，随锐化幅度增加，锐度强度按照 EdgeLowLut[0]~EdgeLowLut[7] 的设定逐渐变化。 取值范围：[0, 255] 数据类型：CVI_U8
EdgeHighLut	控制锐化强度。锐化幅度大于 EdgeHighThr 的部分，锐化强度按照 EdgeHighLut[0]~EdgeHighLut[7] 的设定逐渐变化。 取值范围：[0, 255] 数据类型：CVI_U8
OvershootThr	亮度阈值。亮度小于 OvershootThr 的部分，白边的强度由 OshootWgtHigh 决定。 取值范围：[0, 255] 数据类型：CVI_U8
OshootSlope	决定亮度大于 OvershootThr 部分的白边强度。值越大，随着亮度增加，白边强度降低越快。 取值范围：[0, 32767] 数据类型：CVI_U16
OshootWgtHigh	决定亮度小于 OvershootThr 部分的白边强度。值越大，白边强度越强。 取值范围：[0, 255] 数据类型：CVI_U16
UndershootThr	亮度阈值。亮度大于 UndershootThr 的部分，黑边的强度由 UshootWgtHigh 决定。 取值范围：[0, 255] 数据类型：CVI_U8

下页继续

表 20.11 – 续上页

成员名称	描述
UshootSlope	决定亮度小于 UndershootThr 部分的黑边强度。值越大，随着亮度降低，黑边强度降低越快。 取值范围：[0, 255] 数据类型：CVI_U8
UshootWgtHigh	决定亮度大于 UndershootThr 部分的黑边强度。值越大，黑边强度越大。 取值范围：[0, 255] 数据类型：CVI_U8
UshootLumThrLow	亮度阈值。亮度小于 UshootLumThrLow 的部分，黑边对应的强度为 0。 取值范围：[0, 255] 数据类型：CVI_U8
UshootLumThrHigh	亮度阈值。亮度大于 UshootLumThrHigh 的部分，黑边对应的强度由 UshootLumClampWgt 决定。 取值范围：[0, 255] 数据类型：CVI_U8
UshootLumClampWgt	决定亮度大于 UshootLumThrHigh 部分的黑边强度。值越大，黑边强度越强。 取值范围：[0, 255] 数据类型：CVI_U8
UshootEngThrLow	特征值阈值。特征值小于 UshootEngThrLow 的部分，黑边的强度由 UshootEngClampWgt 决定。 取值范围：[0, 255] 数据类型：CVI_U8
UshootEngThrHigh	特征值阈值。特征值大于 UshootEngThrHigh 的部分，黑边的强度由 UshootEngClampWgt 决定。 取值范围：[0, 255] 数据类型：CVI_U8
UshootEngClampWgt	决定特征值大于 UshootEngThrHigh 部分的黑边强度。值越大，黑边的强度于越强。 取值范围：[0, 255] 数据类型：CVI_U8
UshootLumEngStr	决定黑边强度偏移量的放大倍数。值越大，放大倍数越大。 取值范围：[0, 255] 数据类型：CVI_U8
UshootLumEngNorm	决定黑边强度偏移量的缩小倍数。值越大，缩小倍数越大。 取值范围：[0, 255] 数据类型：CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetPreSharpenRefineAttr
- CVI_ISP_GetPreSharpenRefineAttr

20.3.6 ISP_PRESHARPEN_REFINE_ATTR_S

【说明】

定义 PreSharpen Refine 参数属性

【定义】

```
typedef struct _ISP_PRESHARPEN_REFINE_ATTR_S {
    CVI_BOOL OverUnderShootEn; /*Rw; Range:[0, 1]*/
    ISP_REFINE_TYPE_E RefineEdgeType;
    ISP_PRESHARPEN_REFINE_MANUAL_ATTR_S stManual;
    ISP_PRESHARPEN_REFINE_AUTO_ATTR_S stAuto;
} ISP_PRESHARPEN_REFINE_ATTR_S;
```

【成员】

成员名称	描述
OverUnderShootEn	OverShoot、UnderShoot 调整的使能。 取值范围：[0, 1] 数据类型：CVI_BOOL
RefineEdgeType	Refine 类型 数据类型：ISP_REFINE_TYPE_E
stManual	手动参数
stAuto	自动参数

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetPreSharpenRefineAttr
- CVI_ISP_GetPreSharpenRefineAttr

20.3.7 ISP_PRESHARPEN_EDGE_EXT_MANUAL_ATTR_S

【说明】

定义手动 PreSharpen Edge 参数属性

【定义】

```
typedef struct _ISP_PRESHARPEN_EDGE_EXT_MANUAL_ATTR_S {
    CVI_U16 E5aEnhance; /*Rw; Range:[0, 1023]*/
    CVI_U16 E5bEnhance; /*Rw; Range:[0, 1023]*/
    CVI_U16 E5cEnhance; /*Rw; Range:[0, 1023]*/
    CVI_U16 E7Enhance; /*Rw; Range:[0, 1023]*/
    CVI_U8 BlendThrLow; /*Rw; Range:[0, 255]*/
    CVI_U8 BlendThrHigh; /*Rw; Range:[0, 255]*/
    CVI_U8 BlendWgtLow; /*Rw; Range:[0, 255]*/
    CVI_U8 BlendWgtHigh; /*Rw; Range:[0, 255]*/
}
```

(下页继续)

(续上页)

```

CVI_U8 ThinWgtFlat; /*Rw; Range:[0, 16]*/
CVI_U8 ThinWgtEdge; /*Rw; Range:[0, 16]*/
CVI_U8 ThinWgtUHF_HF; /*Rw; Range:[0, 8]*/
CVI_U8 RobustWgtHF_SHF; /*Rw; Range:[0, 8]*/
CVI_U8 ThinWgtAgainstSHF; /*Rw; Range:[0, 8]*/
CVI_U8 RobustWgtAgainstMF; /*Rw; Range:[0, 8]*/
CVI_U8 ThinWgtFlatHLD; /*Rw; Range:[0, 16]*/
CVI_U8 ThinWgtEdgeHLD; /*Rw; Range:[0, 16]*/
CVI_U8 ThinWgtUHF_HF_HLD; /*Rw; Range:[0, 8]*/
CVI_U8 RobustWgtHF_SHF_HLD; /*Rw; Range:[0, 8]*/
CVI_U8 ThinWgtAgainstSHF_HLD; /*Rw; Range:[0, 8]*/
CVI_U8 RobustWgtAgainstMF_HLD; /*Rw; Range:[0, 8]*/
} ISP_PRESHARPEN_EDGE_EXT_MANUAL_ATTR_S;

```

【成员】

成员名称	描述
E5aEnhance	the enhanced item of the 5x5a edge kernel, 16=1x 取值范围：[0, 1023] 数据类型：CVI_U16
E5bEnhance	the enhanced item of the 5x5b edge kernel, 16=1x 取值范围：[0, 1023] 数据类型：CVI_U16
E5cEnhance	the enhanced item of the 5x5c edge kernel, 16=1x 取值范围：[0, 1023] 数据类型：CVI_U16
E7Enhance	the enhanced item of the 7x7 edge kernel, 16=1x 取值范围：[0, 1023] 数据类型：CVI_U16
BlendThrLow	过 Gamma 的滤波结果和不过 Gamma 的滤波结果，根据亮度选择不同的组合比例。暗区划分阈值，亮度在 BlendThrLow 以内的部分，BlendingWeighting 参考 BlendWgtLow。 取值范围：[0, 255] 数据类型：CVI_U8
BlendThrHigh	亮区划分阈值，亮度大于 BlendThrHigh 的部分，BlendingWeighting 参考 BlendWgtHigh。 取值范围：[0, 255] 数据类型：CVI_U8
BlendWgtLow	暗区参考的 BlendingWeighting。 取值范围：[0, 255] 数据类型：CVI_U8
BlendWgtHigh	亮区参考的 BlendingWeighting。 取值范围：[0, 255] 数据类型：CVI_U8
ThinWgtFlat	控制平坦区 Thin filter 和 Robust filter 的融合比例。值越小，越偏向 Robust filter，即平坦区的滤波结果越偏向 Robust filter 的结果。 取值范围：[0, 16] 数据类型：CVI_U8

下页继续

表 20.13 – 续上页

成员名称	描述
ThinWgtEdge	控制边缘区 Thin filter 和 Robust filter 的融合比例。值越大，越偏向 Thin filter，即边缘区的滤波结果越偏向 Thin filter 的结果。 取值范围：[0, 16] 数据类型：CVI_U8
ThinWgtUHF_HF	Thin filter 频率的第一轮融合调整。值越大，越偏向 Ultra HF；值越小，越偏向 HF。 取值范围：[0, 8] 数据类型：CVI_U8
RobustWgtHF_SHF	Robust filter 频率的第一轮融合调整。值越大，越偏向 HF；值越小，越偏向 Subtle HF。 取值范围：[0, 8] 数据类型：CVI_U8
ThinWgtAgainstSHF	Thin filter 频率的第二轮融合调整。值越大，越偏向第一轮融合调整的结果；值越小，越偏向 Subtle HF。 取值范围：[0, 8] 数据类型：CVI_U8
RobustWgtAgainstMF	Robust filter 频率的第二轮融合调整。值越大，越偏向第一轮融合调整的结果；值越小，越偏向 MF。 取值范围：[0, 8] 数据类型：CVI_U8
ThinWgtFlatHLD	控制高亮区域平坦区 Thin filter 和 Robust filter 的融合比例。值越小，越偏向 Robust filter，即平坦区的滤波结果越偏向 Robust filter 的结果。 取值范围：[0, 16] 数据类型：CVI_U8
ThinWgtEdgeHLD	控制高亮区域边缘区 Thin filter 和 Robust filter 的融合比例。值越大，越偏向 Thin filter，即边缘区的滤波结果越偏向 Thin filter 的结果。 取值范围：[0, 16] 数据类型：CVI_U8
ThinWgtUHF_HF_HLD	高亮纹理区 Thin filter 频率的第一轮融合调整。值越大，越偏向 Ultra HF；值越小，越偏向 HF。 取值范围：[0, 8] 数据类型：CVI_U8
RobustWgtHF_SHF_HLD	高亮纹理区 Robust filter 频率的第一轮融合调整。值越大，越偏向 HF；值越小，越偏向 Subtle HF。 取值范围：[0, 8] 数据类型：CVI_U8
ThinWgtAgainstSHF_HLD	高亮纹理区 Thin filter 频率的第二轮融合调整。值越大，越偏向第一轮融合调整的结果；值越小，越偏向 Subtle HF。 取值范围：[0, 8] 数据类型：CVI_U8
RobustWgtAgainstMF_HLD	高亮纹理区 Robust filter 频率的第二轮融合调整。值越大，越偏向第一轮融合调整的结果；值越小，越偏向 MF。 取值范围：[0, 8] 数据类型：CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetPreSharpenEdgeExtAttr
- CVI_ISP_GetPreSharpenEdgeExtAttr

20.3.8 ISP_PRESHARPEN_EDGE_EXT_AUTO_ATTR_S

【说明】

定义自动 PreSharpen Edge 参数属性

【定义】

```
typedef struct ISP_PRESHARPEN_EDGE_EXT_AUTO_ATTR_S {
    CVI_U16 E5aEnhance[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 1023]*/
    CVI_U16 E5bEnhance[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 1023]*/
    CVI_U16 E5cEnhance[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 1023]*/
    CVI_U16 E7Enhance[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 1023]*/
    CVI_U8 BlendThrLow[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U8 BlendThrHigh[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U8 BlendWgtLow[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U8 BlendWgtHigh[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U8 ThinWgtFlat[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 16]*/
    CVI_U8 ThinWgtEdge[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 16]*/
    CVI_U8 ThinWgtUHF_HF[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 8]*/
    CVI_U8 RobustWgtHF_SHF[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 8]*/
    CVI_U8 ThinWgtAgainstSHF[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 8]*/
    CVI_U8 RobustWgtAgainstMF[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 8]*/
    CVI_U8 ThinWgtFlatHLD[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 16]*/
    CVI_U8 ThinWgtEdgeHLD[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 16]*/
    CVI_U8 ThinWgtUHF_HF_HLD[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 8]*/
    CVI_U8 RobustWgtHF_SHF_HLD[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 8]*/
    CVI_U8 ThinWgtAgainstSHF_HLD[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 8]*/
    CVI_U8 RobustWgtAgainstMF_HLD[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 8]*/
} ISP_PRESHARPEN_EDGE_EXT_AUTO_ATTR_S;
```

【成员】

成员名称	描述
E5aEnhance	the enhanced item of the 5x5a edge kernel, 16=1x 取值范围：[0, 1023] 数据类型：CVI_U16
E5bEnhance	the enhanced item of the 5x5b edge kernel, 16=1x 取值范围：[0, 1023] 数据类型：CVI_U16

下页继续

表 20.14 – 续上页

成员名称	描述
E5cEnhance	the enhanced item of the 5x5c edge kernel, 16=1x 取值范围: [0, 1023] 数据类型: CVI_U16
E7Enhance	the enhanced item of the 7x7 edge kernel, 16=1x 取值范围: [0, 1023] 数据类型: CVI_U16
BlendThrLow	过 Gamma 的滤波结果和不过 Gamma 的滤波结果, 根据亮度选择不同的组合比例。暗区划分阈值, 亮度在 BlendThrLow 以内的部分, BlendingWeighting 参考 BlendWgtLow。 取值范围: [0, 255] 数据类型: CVI_U8
BlendThrHigh	亮区划分阈值, 亮度大于 BlendThrHigh 的部分, BlendingWeighting 参考 BlendWgtHigh。 取值范围: [0, 255] 数据类型: CVI_U8
BlendWgtLow	暗区参考的 BlendingWeighting。 取值范围: [0, 255] 数据类型: CVI_U8
BlendWgtHigh	亮区参考的 BlendingWeighting。 取值范围: [0, 255] 数据类型: CVI_U8
ThinWgtFlat	控制平坦区 Thin filter 和 Robust filter 的融合比例。值越小, 越偏向 Robust filter, 即平坦区的滤波结果越偏向 Robust filter 的结果。 取值范围: [0, 16] 数据类型: CVI_U8
ThinWgtEdge	控制边缘区 Thin filter 和 Robust filter 的融合比例。值越大, 越偏向 Thin filter, 即边缘区的滤波结果越偏向 Thin filter 的结果。 取值范围: [0, 16] 数据类型: CVI_U8
ThinWgtUHF_HF	Thin filter 频率的第一轮融合调整。值越大, 越偏向 Ultra HF; 值越小, 越偏向 HF。 取值范围: [0, 8] 数据类型: CVI_U8
RobustWgtHF_SHF	Robust filter 频率的第一轮融合调整。值越大, 越偏向 HF; 值越小, 越偏向 Subtle HF。 取值范围: [0, 8] 数据类型: CVI_U8
ThinWgtAgainstSHF	Thin filter 频率的第二轮融合调整。值越大, 越偏向第一轮融合调整的结果; 值越小, 越偏向 Subtle HF。 取值范围: [0, 8] 数据类型: CVI_U8
RobustWgtAgainstMF	Robust filter 频率的第二轮融合调整。值越大, 越偏向第一轮融合调整的结果; 值越小, 越偏向 MF。 取值范围: [0, 8] 数据类型: CVI_U8

下页继续

表 20.14 – 续上页

成员名称	描述
ThinWgtFlatHLD	控制高亮区域平坦区 Thin filter 和 Robust filter 的融合比例。值越小，越偏向 Robust filter，即平坦区的滤波结果越偏向 Robust filter 的结果。 取值范围：[0, 16] 数据类型：CVI_U8
ThinWgtEdgeHLD	控制高亮区域边缘区 Thin filter 和 Robust filter 的融合比例。值越大，越偏向 Thin filter，即边缘区的滤波结果越偏向 Thin filter 的结果。 取值范围：[0, 16] 数据类型：CVI_U8
ThinWgtUHF_HF_HLD	高亮纹理区 Thin filter 频率的第一轮融合调整。值越大，越偏向 Ultra HF；值越小，越偏向 HF。 取值范围：[0, 8] 数据类型：CVI_U8
RobustWgtHF_SHF_HLD	高亮纹理区 Robust filter 频率的第一轮融合调整。值越大，越偏向 HF；值越小，越偏向 Subtle HF。 取值范围：[0, 8] 数据类型：CVI_U8
ThinWgtAgainstSHF_HLD	高亮纹理区 Thin filter 频率的第二轮融合调整。值越大，越偏向第一轮融合调整的结果；值越小，越偏向 Subtle HF。 取值范围：[0, 8] 数据类型：CVI_U8
RobustWgtAgainstMF_HLD	高亮纹理区 Robust filter 频率的第二轮融合调整。值越大，越偏向第一轮融合调整的结果；值越小，越偏向 MF。 取值范围：[0, 8] 数据类型：CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetPreSharpenEdgeExtAttr
- CVI_ISP_GetPreSharpenEdgeExtAttr

20.3.9 ISP_PRESHARPEN_EDGE_EXT_ATTR_S**【说明】**

定义 PreSharpen Edge 参数属性

【定义】

```
typedef struct _ISP_PRESHARPEN_EDGE_EXT_ATTR_S {
    CVI_BOOL RegionStrEn; /*Rw; Range:[0, 1]*/
    ISP_PRESHARPEN_EDGE_EXT_MANUAL_ATTR_S stManual;
    ISP_PRESHARPEN_EDGE_EXT_AUTO_ATTR_S stAuto;
} ISP_PRESHARPEN_EDGE_EXT_ATTR_S;
```

【成员】

成员名称	描述
RegionStrEn	根据不同区域类型（平坦区、边缘区、纹理区）调整锐化强度功能使能。 取值范围：[0, 1] 数据类型：CVI_BOOL
stManual	手动参数
stAuto	自动参数

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetPreSharpenEdgeExtAttr
- CVI_ISP_GetPreSharpenEdgeExtAttr

21 RGBGAMMA

21.1 功能描述

因人眼对低亮度较敏感, 故对图像的亮度做非线性转换, 压缩亮区、拉伸暗区使整体亮度感受更好。

21.2 API 参考

- `CVI_ISP_SetGammaAttr` : 设置 gamma 属性参数
- `CVI_ISP_GetGammaAttr` : 获取 gamma 属性参数

21.2.1 CVI_ISP_SetGammaAttr

【描述】

设置 gamma 属性参数

【语法】

```
CVI_S32 CVI_ISP_SetGammaAttr(VI_PIPE ViPipe, const ISP_GAMMA_ATTR_S_
↪ *pstGammaAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstGammaAttr	gamma 属性参数	输入

【返回值】

返回值	描述
0	成功
非 0	失败, 其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

```
VI_PIPE ViPipe = 0;
ISP_GAMMA_ATTR_S stAttr;
CVI_ISP_GetGammaAttr(ViPipe, &stAttr);
stAttr.enOpType = OP_TYPE_AUTO;
CVI_ISP_SetGammaAttr(ViPipe, &stAttr);
```

【相关主题】

- [CVI_ISP_GetGammaAttr](#)

21.2.2 CVI_ISP_GetGammaAttr

【描述】

获取 gamma 属性参数

【语法】

```
CVI_S32 CVI_ISP_GetGammaAttr(VI_PIPE ViPipe, ISP_GAMMA_ATTR_S *pstGammaAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstGammaAttr	gamma 属性参数	输出

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

无

【相关主题】

- CVI_ISP_SetGammaAttr

21.3 数据类型

- ISP_GAMMA_ATTR_S：定义 gamma 属性参数

21.3.1 ISP_GAMMA_ATTR_S

【说明】

定义 gamma 属性参数

【定义】

```
typedef struct ISP_GAMMA_ATTR_S {  
    CVI_BOOL Enable; /*Rw; Range:[0, 1]*/  
    CVI_U8 UpdateInterval; /*Rw; Range:[0, 255]*/  
    CVI_U16 Table[GAMMA_NODE_NUM]; /*Rw; Range:[0, 4095]*/  
    ISP_GAMMA_CURVE_TYPE_E enCurveType;  
} ISP_GAMMA_ATTR_S;
```

【成员】

成员名称	描述
Enable	模块使能开关 取值范围：[0, 1] 数据类型：CVI_BOOL
UpdateInterval	影响参数更新间隔，值越大画面变化越慢，效能越好。 取值范围：[0, 255] 数据类型：CVI_U8
Table	Gamma 曲线节点数值 取值范围：[0, 4095] 数据类型：CVI_U16
enCurveType	Gamma 曲线类型 数据类型：ISP_GAMMA_CURVE_TYPE_E

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetGammaAttr
- CVI_ISP_GetGammaAttr

22 DCI

22.1 功能描述

Dynamic Contrast Improvement 的简称，藉由调整直方图来加强图像对比度，提升暗部细节。

22.2 API 参考

- `CVI_ISP_SetDCIAttr`：设置 DCI 参数属性
- `CVI_ISP_GetDCIAttr`：获取 DCI 参数属性
- `CVI_ISP_SetDciAutoGammaAttr`：设置 Dci Gamma 参数属性
- `CVI_ISP_GetDciAutoGammaAttr`：获取 Dci Gamma 参数属性

22.2.1 CVI_ISP_SetDCIAttr

【描述】

设置 DCI 参数属性

【语法】

```
CVI_S32 CVI_ISP_SetDCIAttr(VI_PIPE ViPipe, const ISP_DCI_ATTR_S *pstDCIAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstDCIAttr	DCI 参数属性	输入

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

```
VI_PIPE ViPipe = 0;
ISP_DCI_ATTR_S stAttr;
CVI_ISP_GetDCIAttr(ViPipe, &stAttr);
stAttr.enOpType = OP_TYPE_AUTO;
CVI_ISP_SetDCIAttr(ViPipe, &stAttr);
```

【相关主题】

- [CVI_ISP_GetDCIAttr](#)

22.2.2 CVI_ISP_GetDCIAttr

【描述】

获取 DCI 参数属性

【语法】

```
CVI_S32 CVI_ISP_GetDCIAttr(VI_PIPE ViPipe, ISP_DCI_ATTR_S *pstDCIAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstDCIAttr	DCI 参数属性	输出

【返回值】

返回值	描述
0	成功
非 0	失败, 其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

无

【相关主题】

- [CVI_ISP_SetDCIAttr](#)

22.2.3 CVI_ISP_SetDciAutoGammaAttr

【描述】

设置 Dci Gamma 参数属性

【语法】

```
CVI_S32 CVI_ISP_SetDciAutoGammaAttr(VI_PIPE ViPipe, const ISP_DCI_AUTO_GAMMA_
↪ATTR_S *pstDciAutoGammaAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstDciAutoGammaAttr	Dci Gamma 参数属性	输入

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

```
VI_PIPE ViPipe = 0;
ISP_DCI_AUTO_GAMMA_ATTR_S stAttr;
CVI_ISP_GetDciAutoGammaAttr(ViPipe, &stAttr);
stAttr.enOpType = OP_TYPE_AUTO;
CVI_ISP_SetDciAutoGammaAttr(ViPipe, &stAttr);
```

【相关主题】

- [CVI_ISP_GetDciAutoGammaAttr](#)

22.2.4 CVI_ISP_GetDciAutoGammaAttr

【描述】

获取 Dci Gamma 参数属性

【语法】

```
CVI_S32 CVI_ISP_GetDciAutoGammaAttr(VI_PIPE ViPipe, ISP_DCI_AUTO_GAMMA_ATTR_
→S *pstDciAutoGammaAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstDciAutoGammaAttr	Dci Gamma 参数属性	输出

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: cvi_isp.h, cvi_comm_isp.h
- 库文件: libisp.so

【注意】

无

【举例】

无

【相关主题】

- CVI_ISP_SetDciAutoGammaAttr

22.3 数据类型

- ISP_DCI_ATTR_S：定义 DCI 参数属性
- ISP_DCI_AUTO_GAMMA_ATTR_S：定义 Dci Gamma 参数属性
- ISP_DCI_GAMMA_CURVE_ATTR_S：定义 Dci Gamma Curve 参数属性

22.3.1 ISP_DCI_ATTR_S

【说明】

定义 DCI 参数属性

【定义】

```
typedef struct _ISP_DCI_ATTR_S {
    CVI_BOOL Enable; /*Rw; Range:[0, 1]*/
    ISP_DCI_CURVE_MODE_E CurveMode;
    CVI_U8 UpdateInterval; /*Rw; Range:[1, 255]*/
    CVI_U16 DciStrength; /*Rw; Range:[0, 8192]*/
    CVI_U8 DciGamma[5]; /*Rw; Range:[0, 31]*/
    CVI_U8 DciContrast[5]; /*Rw; Range:[0, 3]*/
    CVI_U8 DciOffset[5]; /*Rw; Range:[0, 15]*/
    CVI_U8 Sensitivity; /*Rw; Range:[0, 255]*/
    CVI_U16 Speed; /*Rw; Range:[0, 500]*/
    CVI_U8 DehazeLight; /*Rw; Range:[0, 5]*/
    CVI_U8 DehazeStrength; /*Rw; Range:[0, 100]*/
    CVI_U8 DehazeLut[6]; /*Rw; Range:[0, 100]*/
} ISP_DCI_ATTR_S;
```

【成员】

成员名称	描述
Enable	模块使能开关 取值范围：[0, 1] 数据类型：CVI_BOOL
CurveMode	DciGammaCurve 生成模式 DCI_CURVE_AUTO: 自动模式 DCI_CURVE_AUTO_GAMMA: 通过 gamma 曲线生成 DCI 曲线 DCI_CURVE_DEHAZE: 去雾曲线 数据类型：ISP_DCI_CURVE_MODE_E
UpdateInterval	影响参数更新间隔，值越大画面变化越慢，效能越好。 取值范围：[1, 255] 数据类型：CVI_U8
DciStrength	自动模式下生效，用来控制 DCI 的强度。值越大，对比度越大。 取值范围：[0, 8192] 数据类型：CVI_U16
DciGamma	Auto gamma 模式，辅助 pqtool 端手动生成 DciGammaCurve。 用来控制亮度抬升程度。值越大，亮度抬升程度越大。 取值范围：[0, 31] 数据类型：CVI_U8
DciContrast	Auto gamma 模式，辅助 pqtool 端手动生成 DciGammaCurve。 用来控制对比度强度，值越大，对比度越大。 取值范围：[0, 3] 数据类型：CVI_U8

下页继续

表 22.5 – 续上页

成员名称	描述
DciOffset	Auto gamma 模式, 辅助 pqtool 端手动生成 DciGammaCurve。 DciOffset 以内保持线性。 取值范围: [0, 15] 数据类型: CVI_U8
Sensitivity	灵敏度: 灵敏度越高, dci 重运算的 threshold 越低 取值范围: [0, 255] 数据类型: CVI_U8
Speed	DCI 曲线时间域上变化的平顺度。数值越大, 时间域变化越平顺, 反之, 则变化越快。 取值范围: [0, 500] 数据类型: CVI_U16
DehazeLight	亮度直方图百分比 取值范围: [0, 5] 数据类型: CVI_U8
DehazeStrength	去雾强度 取值范围: [0, 100] 数据类型: CVI_U8
DehazeLut	去雾表 取值范围: [0, 100] 数据类型: CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetDCIAttr
- CVI_ISP_GetDCIAttr

22.3.2 ISP_DCI_AUTO_GAMMA_ATTR_S

【说明】

定义 Dci Gamma 参数属性

【定义】

```
typedef struct _ISP_DCI_AUTO_GAMMA_ATTR_S {
    CVI_U8 GammaTabNum; /*Rw; Range:[1, 5]*/
    ISP_DCI_GAMMA_CURVE_ATTR_S GammaTab[5];
} ISP_DCI_AUTO_GAMMA_ATTR_S;
```

【成员】

成员名称	描述
GammaTabNum	自适应 Gamma 曲线数量。 取值范围：[1, 5] 数据类型：CVI_U8
GammaTab	自适应 Gamma 曲线信息。 数据类型：ISP_DCI_GAMMA_CURVE_ATTR_S

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetDciAutoGammaAttr
- CVI_ISP_GetDciAutoGammaAttr

22.3.3 ISP_DCI_GAMMA_CURVE_ATTR_S

【说明】

定义 Dci Gamma Curve 参数属性

【定义】

```
typedef struct _ISP_DCI_GAMMA_CURVE_ATTR_S {  
    CVI_S16 Lv; /*Rw; Range:[-500, 1500]*/  
    CVI_U16 Tbl[256]; /*Rw; Range:[0, 255]*/  
} ISP_DCI_GAMMA_CURVE_ATTR_S;
```

【成员】

成员名称	描述
Lv	Gamma 曲线对应 Lv 范围。 取值范围：[-500, 1500] 数据类型：CVI_S16
Tbl	Gamma 曲线节点数值。 取值范围：[0, 255] 数据类型：CVI_U16

【注意事项】

无

【相关数据类型及接口】

23 LDCI

23.1 功能描述

Local Dynamic Contrast Improvement 的简称，藉由调整直方图来加强图像对比度，提升暗部细节。

23.2 API 参考

- `CVI_ISP_SetLDCIAttr`：设置 LDCI 参数属性
- `CVI_ISP_GetLDCIAttr`：获取 LDCI 参数属性

23.2.1 CVI_ISP_SetLDCIAttr

【描述】

设置 LDCI 参数属性

【语法】

```
CVI_S32 CVI_ISP_SetLDCIAttr(VI_PIPE ViPipe, const ISP_LDCI_ATTR_S *pstLDCIAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstLDCIAttr	LDCI 参数属性	输入

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

```
VI_PIPE ViPipe = 0;
ISP_LDCI_ATTR_S stAttr;
CVI_ISP_GetLDCIAttr(ViPipe, &stAttr);
stAttr.enOpType = OP_TYPE_AUTO;
CVI_ISP_SetLDCIAttr(ViPipe, &stAttr);
```

【相关主题】

- [CVI_ISP_GetLDCIAttr](#)

23.2.2 CVI_ISP_GetLDCIAttr

【描述】

获取 LDCI 参数属性

【语法】

```
CVI_S32 CVI_ISP_GetLDCIAttr(VI_PIPE ViPipe, ISP_LDCI_ATTR_S *pstLDCIAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstLDCIAttr	LDCI 参数属性	输出

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

无

【相关主题】

- CVI_ISP_SetLDCIAttr

23.3 数据类型

- ISP_LDCI_MANUAL_ATTR_S：定义手动 LDCI 参数属性
- ISP_LDCI_AUTO_ATTR_S：定义自动 LDCI 参数属性
- ISP_LDCI_ATTR_S：定义 LDCI 参数属性

23.3.1 ISP_LDCI_MANUAL_ATTR_S

【说明】

定义手动 LDCI 参数属性

【定义】

```
typedef struct _ISP_LDCI_MANUAL_ATTR_S {  
    CVI_U8 LdciStrength[9]; /*Rw; Range:[0, 255]*/  
    CVI_U8 AdaptiveGainMax; /*Rw; Range:[0, 255]*/  
    CVI_U8 AdaptiveGainMin; /*Rw; Range:[0, 255]*/  
    CVI_U8 FBCStrength; /*Rw; Range:[0, 31]*/  
    CVI_U8 LdciThres1; /*Rw; Range:[0, 255]*/  
} ISP_LDCI_MANUAL_ATTR_S;
```

【成员】

成员名称	描述
LdciStrength	根据不同的亮度，做不同强度的对比度拉升。 取值范围：[0, 255] 数据类型：CVI_U8
AdaptiveGainMax	根据当前点亮度与局部平均亮度的差异，做不同强度的对比度拉升。值越大，亮度差异大的点，对比度拉升程度越大。 取值范围：[0, 255] 数据类型：CVI_U8
AdaptiveGainMin	根据当前点亮度与局部平均亮度的差异，做不同强度的对比度拉升。值越大，亮度差异小的点，对比度拉升程度越大。 取值范围：[0, 255] 数据类型：CVI_U8
FBCStrength	逆光人脸保护功能。0 表示功能不开启；非 0 表示功能开启，并且值越小，人脸亮度保护程度越大。 取值范围：[0, 31] 数据类型：CVI_U8
LdciThres1	对像素点亮度与 block 平均亮度的差异值做收缩。 取值范围：[0, 255] 数据类型：CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetLDCIAttr
- CVI_ISP_GetLDCIAttr

23.3.2 ISP_LDCI_AUTO_ATTR_S

【说明】

定义自动 LDCI 参数属性

【定义】

```
typedef struct _ISP_LDCI_AUTO_ATTR_S {  
    CVI_U8 LdciStrength[9][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/  
    CVI_U8 AdaptiveGainMax[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/  
    CVI_U8 AdaptiveGainMin[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/  
    CVI_U8 FBCStrength[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 31]*/  
    CVI_U8 LdciThres1[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/  
} ISP_LDCI_AUTO_ATTR_S;
```

【成员】

成员名称	描述
LdciStrength	根据不同的亮度，做不同强度的对比度拉升。 取值范围：[0, 255] 数据类型：CVI_U8
AdaptiveGainMax	根据当前点亮度与局部平均亮度的差异，做不同强度的对比度拉升。值越大，亮度差异大的点，对比度拉升程度越大。 取值范围：[0, 255] 数据类型：CVI_U8
AdaptiveGainMin	根据当前点亮度与局部平均亮度的差异，做不同强度的对比度拉升。值越大，亮度差异小的点，对比度拉升程度越大。 取值范围：[0, 255] 数据类型：CVI_U8
FBCStrength	逆光人脸保护功能。0 表示功能不开启；非 0 表示功能开启，并且值越小，人脸亮度保护程度越大。 取值范围：[0, 31] 数据类型：CVI_U8
LdciThres1	对像素点亮度与 block 平均亮度的差异值做收缩。 取值范围：[0, 255] 数据类型：CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetLDCIAttr
- CVI_ISP_GetLDCIAttr

23.3.3 ISP_LDCI_ATTR_S

【说明】

定义 LDCI 参数属性

【定义】

```
typedef struct _ISP_LDCI_ATTR_S {  
    CVI_BOOL Enable; /*Rw; Range:[0, 1]*/  
    ISP_OP_TYPE_E enOpType;  
    CVI_U8 UpdateInterval; /*Rw; Range:[0, 255]*/  
    CVI_U8 BlockNum; /*Rw; Range:[4, 62]*/  
    ISP_LDCI_MANUAL_ATTR_S stManual;  
    ISP_LDCI_AUTO_ATTR_S stAuto;  
} ISP_LDCI_ATTR_S;
```

【成员】

成员名称	描述
Enable	模块使能开关 取值范围：[0, 1] 数据类型：CVI_BOOL
enOpType	工作类型 OP_TYPE_AUTO：自动模式 OP_TYPE_MANUAL：手动模式 数据类型：ISP_OP_TYPE_E
UpdateInterval	影响参数更新间隔，值越大画面变化越慢，效能越好。 取值范围：[0, 255] 数据类型：CVI_U8
BlockNum	统计 luma 均值的分块数量。 取值范围：[4, 62] 数据类型：CVI_U8
stManual	手动参数
stAuto	自动参数

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetLDCIAttr
- CVI_ISP_GetLDCIAttr

24 ColorTone

24.1 功能描述

在 WB 的基础之下，再额外调整图像颜色偏红或偏蓝等偏好。

24.2 API 参考

- CVI_ISP_SetColorToneAttr: 设置色调属性参数
- CVI_ISP_GetColorToneAttr: 获取色调属性参数

24.2.1 CVI_ISP_SetColorToneAttr

【描述】

设置色调属性参数

【语法】

```
CVI_S32 CVI_ISP_SetColorToneAttr(VI_PIPE ViPipe, const ISP_COLOR_TONE_ATTR_S_
→ *pstColorToneAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstColorToneAttr	色调属性参数	输入

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

无

【相关主题】

`CVI_ISP_GetColorToneAttr`

24.2.2 CVI_ISP_GetColorToneAttr

【描述】

获取色调属性参数

【语法】

```
CVI_S32 CVI_ISP_GetColorToneAttr(VI_PIPE ViPipe, ISP_COLOR_TONE_ATTR_S_
→*pstWBGAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstColorToneAttr	色调属性参数	输出

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

无

【相关主题】

`CVI_ISP_SetColorToneAttr`

24.3 数据类型

· `ISP_COLOR_TONE_ATTR_S`: 色调属性参数

24.3.1 `ISP_COLOR_TONE_ATTR_S`

【说明】

色调属性参数

【定义】

```
typedef struct _ISP_COLOR_TONE_ATTR_S {  
    CVI_U16 u16RedCastGain;  
    CVI_U16 u16GreenCastGain;  
    CVI_U16 u16BlueCastGain;  
} ISP_COLOR_TONE_ATTR_S;
```

【成员】

成员名称	描述
u16RedCastGain	R 通道增益，8bit 小数精度。 取值范围：[0x0, 0x1000] 数据类型：CVI_U16
u16GreenCastGain	G 通道增益，8bit 小数精度。 取值范围：[0x0, 0x1000] 数据类型：CVI_U16
u16BlueCastGain	B 通道增益，8bit 小数精度。 取值范围：[0x0, 0x1000] 数据类型：CVI_U16

【注意事项】

无

【相关数据类型及接口】

`CVI_ISP_SetColorToneAttr`

`CVI_ISP_GetColorToneAttr`

25 Saturation

25.1 功能描述

调整颜色饱和度。

25.2 API 参考

- `CVI_ISP_SetSaturationAttr`：设置饱和度属性参数
- `CVI_ISP_GetSaturationAttr`：获取饱和度属性参数

25.2.1 CVI_ISP_SetSaturationAttr

【描述】

设置饱和度属性参数

【语法】

```
CVI_S32 CVI_ISP_SetSaturationAttr(VI_PIPE ViPipe, const ISP_SATURATION_ATTR_S_
↪ *pstSaturationAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstSaturationAttr	饱和度属性参数	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无。

【举例】

无。

【相关主题】

- [CVI_ISP_GetSaturationAttr](#)

25.2.2 CVI_ISP_GetSaturationAttr

【描述】

获取饱和度属性参数

【语法】

```
CVI_S32 CVI_ISP_GetSaturationAttr(VI_PIPE ViPipe, ISP_SATURATION_ATTR_S_
↪ *pstSaturationAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstSaturationAttr	饱和度属性参数	输出

【返回值】

返回值	描述
0	成功。
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无。

【举例】

无。

【相关主题】

- [CVI_ISP_SetSaturationAttr](#)

25.3 数据类型

- `ISP_SATURATION_MANUAL_ATTR_S`：饱和度属性手动参数
- `ISP_SATURATION_AUTO_ATTR_S`：饱和度属性自动参数
- `ISP_SATURATION_ATTR_S`：饱和度属性参数

25.3.1 `ISP_SATURATION_MANUAL_ATTR_S`

【说明】

饱和度属性手动参数

【定义】

```
typedef struct _ISP_SATURATION_MANUAL_ATTR_S {  
    CVI_U8 Saturation;  
} ISP_SATURATION_MANUAL_ATTR_S;
```

【成员】

成员名称	描述
Saturation	由三个数值组成的数组，定义输出的 UV 增益。 根据输入饱和度查找 UV 的增益，值越大，饱和度越高；反之，则越小。 取值范围：[0x0, 0xFF] 数据类型：CVI_U8

【注意事项】

无。

【相关数据类型及接口】

- `CVI_ISP_SetSaturationAttr`
- `CVI_ISP_GetSaturationAttr`

25.3.2 `ISP_SATURATION_AUTO_ATTR_S`

【说明】

饱和度属性自动参数

【定义】

```
typedef struct _ISP_SATURATION_AUTO_ATTR_S {  
    CVI_U8 Saturation[ISP_AUTO_ISO_STRENGTH_NUM];  
} ISP_SATURATION_AUTO_ATTR_S;
```

【成员】

成员名称	描述
Saturation	由三个数值组成的数组，定义输出的 UV 增益。 根据输入饱和度查找 UV 的增益，值越大，饱和度越高；反之，则越小。 取值范围：[0x0, 0xFF] 数据类型：CVI_U8

【注意事项】

无。

【相关数据类型及接口】

- CVI_ISP_SetSaturationAttr
- CVI_ISP_GetSaturationAttr

25.3.3 ISP_SATURATION_ATTR_S

【说明】

饱和度属性参数

【定义】

```
typedef struct _ISP_SATURATION_ATTR_S {  
    ISP_OP_TYPE_E enOpType;  
    ISP_SATURATION_MANUAL_ATTR_S stManual;  
    ISP_SATURATION_AUTO_ATTR_S stAuto;  
} ISP_SATURATION_ATTR_S;
```

【成员】

成员名称	描述
enOpType	工作类型 OP_TYPE_AUTO: 自动模式 OP_TYPE_MANUAL: 手动模式
stAuto	自动模式参数属性
stManual	手动模式参数属性

【注意事项】

无。

【相关数据类型及接口】

- CVI_ISP_SetSaturationAttr
- CVI_ISP_GetSaturationAttr

26 PFR

26.1 功能描述

色彩畸变校正, 因不同颜色的光折射率不同, 在高亮区与低亮区交界的物体周围容易形成紫边, 以此模块实现图像去紫边功能, 改善图像边缘的紫边现象

26.2 API 参考

- `CVI_ISP_SetPFRAttr` : 设置 PFR 属性参数
- `CVI_ISP_GetPFRAttr` : 获取 PFR 属性参数

26.2.1 CVI_ISP_SetPFRAttr

【描述】

设置 PFR 属性参数

【语法】

```
CVI_S32 CVI_ISP_SetPFRAttr(VI_PIPE ViPipe, const ISP_PFR_ATTR_S *pstPFRAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstPFRAttr	PFR 属性参数	输入

【返回值】

返回值	描述
0	成功
非 0	失败, 其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

```
VI_PIPE ViPipe = 0;
ISP_PFR_ATTR_S stAttr;
CVI_ISP_GetPFRAttr(ViPipe, &stAttr);
stAttr.enOpType = OP_TYPE_AUTO;
CVI_ISP_SetPFRAttr(ViPipe, &stAttr);
```

【相关主题】

- [CVI_ISP_GetPFRAttr](#)

26.2.2 CVI_ISP_GetPFRAttr

【描述】

获取 PFR 属性参数

【语法】

```
CVI_S32 CVI_ISP_GetPFRAttr(VI_PIPE ViPipe, ISP_PFR_ATTR_S *pstPFRAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstPFRAttr	PFR 属性参数	输出

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

无

【相关主题】

- CVI_ISP_SetPFRAttr

26.3 数据类型

- ISP_PFR_MANUAL_ATTR_S：定义手动 PFR 属性参数
- ISP_PFR_AUTO_ATTR_S：定义自动 PFR 属性参数
- ISP_PFR_ATTR_S：定义 PFR 属性参数

26.3.1 ISP_PFR_MANUAL_ATTR_S

【说明】

定义手动 PFR 属性参数

【定义】

```
typedef struct _ISP_PFR_MANUAL_ATTR_S {  
    CVI_U8 LumaIniVal; /*Rw; Range:[0, 255]*/  
    CVI_U8 LumLevelTh; /*Rw; Range:[0, 255]*/  
    CVI_U8 LumLevelLut[PFR_LUMA_LEVEL]; /*Rw; Range:[0, 255]*/  
} ISP_PFR_MANUAL_ATTR_S;
```

【成员】

成员名称	描述
LumaIniVal	Luma 的初始值。建议不要调。 取值范围：[0, 255] 数据类型：CVI_U8
LumLevelTh	Luma 的阈值。大于这个阈值的地方消紫边参考 Luma 信息。 取值范围：[0, 255] 数据类型：CVI_U8
LumLevelLut	Luma level 的插值 LUT 表，一般不需要更改。 取值范围：[0, 255] 数据类型：CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetPFRAttr
- CVI_ISP_GetPFRAttr

26.3.2 ISP_PFR_AUTO_ATTR_S

【说明】

定义自动 PFR 属性参数

【定义】

```
typedef struct _ISP_PFR_AUTO_ATTR_S {
    CVI_U8 LumaIniVal[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U8 LumLevelTh[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
    CVI_U8 LumLevelLut[PFR_LUMA_LEVEL][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/
} ISP_PFR_AUTO_ATTR_S;
```

【成员】

成员名称	描述
LumaIniVal	Luma 的初始值。建议不要调。 取值范围：[0, 255] 数据类型：CVI_U8
LumLevelTh	Luma 的阈值。大于这个阈值的地方消紫边参考 Luma 信息。 取值范围：[0, 255] 数据类型：CVI_U8
LumLevelLut	Luma level 的插值 LUT 表，一般不需要更改。 取值范围：[0, 255] 数据类型：CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetPFRAttr
- CVI_ISP_GetPFRAttr

26.3.3 ISP_PFR_ATTR_S

【说明】

定义 PFR 属性参数

【定义】

```
typedef struct _ISP_PFR_ATTR_S {
    CVI_BOOL Enable; /*Rw; Range:[0, 1]*/
    ISP_OP_TYPE_E enOpType;
    CVI_U8 UpdateInterval; /*Rw; Range:[0, 255]*/
    CVI_BOOL TuningMode; /*Rw; Range:[0, 1]*/
    CVI_BOOL LumaEN; /*Rw; Range:[0, 1]*/
    CVI_BOOL UVENLut[PFR_COLOR_NUM]; /*Rw; Range:[0, 1]*/
}
```

(下页继续)

(续上页)

```

CVI_BOOL HueENLut[PFR_COLOR_NUM]; /*Rw; Range:[0, 1]*/
CVI_U8 ULut[PFR_COLOR_NUM]; /*Rw; Range:[0, 255]*/
CVI_U8 VLut[PFR_COLOR_NUM]; /*Rw; Range:[0, 255]*/
CVI_U8 UVDiffThLut[PFR_COLOR_NUM]; /*Rw; Range:[0, 255]*/
CVI_U8 UVDiffLut0[PFR_COLOR_LUT_NUM]; /*Rw; Range:[0, 255]*/
CVI_U8 UVDiffLut1[PFR_COLOR_LUT_NUM]; /*Rw; Range:[0, 255]*/
CVI_U8 RWetLut[PFR_COLOR_NUM]; /*Rw; Range:[0, 255]*/
CVI_U8 BWetLut[PFR_COLOR_NUM]; /*Rw; Range:[0, 255]*/
CVI_U16 HueLut[PFR_COLOR_NUM]; /*Rw; Range:[0, 359]*/
CVI_U8 HueRngLut[PFR_COLOR_NUM]; /*Rw; Range:[0, 255]*/
CVI_U8 HueTh; /*Rw; Range:[0, 255]*/
CVI_U8 EdgeThLut[PFR_EDGE_LUT_NUM]; /*Rw; Range:[0, 255]*/
CVI_U8 GDiffLut[PFR_COLOR_LUT_NUM]; /*Rw; Range:[0, 127]*/
ISP_PFR_MANUAL_ATTR_S stManual;
ISP_PFR_AUTO_ATTR_S stAuto;
} ISP_PFR_ATTR_S;

```

【成员】

成员名称	描述
Enable	模块使能开关 取值范围：[0, 1] 数据类型：CVI_BOOL
enOpType	工作类型 OP_TYPE_AUTO：自动模式 OP_TYPE_MANUAL：手动模式 数据类型：ISP_OP_TYPE_E
UpdateInterval	影响参数更新间隔，值越大画面变化越慢，效能越好。 取值范围：[0, 255] 数据类型：CVI_U8
TuningMode	None 取值范围：[0, 1] 数据类型：CVI_BOOL
LumaEN	去紫边处理时，参考 luma 信息功能使能。紫边通常与 Luma 信息相关，建议开启。 0：关闭。 1：使能。 取值范围：[0, 1] 数据类型：CVI_BOOL
UVENLut	参考自定义紫边 UV 值使能。可以支持自定义两组 UV 值，由 ULut、VLut 定义。 0：关闭。 1：使能。 取值范围：[0, 1] 数据类型：CVI_BOOL

下页继续

表 26.5 – 续上页

成员名称	描述
HueENLut	参考自定义紫边 Hue 值使能。可以支持自定义两组 Hue 值，由 HueLut 定义。 0: 关闭。 1: 使能。 取值范围: [0, 1] 数据类型: CVI_BOOL
ULut	标定的紫边的两组 U 值。一版情况下不建议修改 default 值。不同 ISO 或不同场景下，当紫边颜色与 default 值差异较大，可以根据情况修改。 取值范围: [0, 255] 数据类型: CVI_U8
VLut	标定的紫边的两组 V 值。一版情况下不建议修改 default 值。不同 ISO 或不同场景下，当紫边颜色与 default 值差异较大，可以根据情况修改。 取值范围: [0, 255] 数据类型: CVI_U8
UVDiffThLut	None 取值范围: [0, 255] 数据类型: CVI_U8
UVDiffLut0	None 取值范围: [0, 255] 数据类型: CVI_U8
UVDiffLut1	None 取值范围: [0, 255] 数据类型: CVI_U8
RWetLut	None 取值范围: [0, 255] 数据类型: CVI_U8
BWetLut	None 取值范围: [0, 255] 数据类型: CVI_U8
HueLut	标定的紫边的两组 Hue 值。一版情况下不建议修改 default 值。不同 ISO 或不同场景下，当紫边 Hue 值与 default 值差异较大，可以根据情况修改。 取值范围: [0, 359] 数据类型: CVI_U16
HueRngLut	与标定的 Hue 值可被允许的差异范围。超出这个阈值认为完全偏离标定值；在范围内与 HUE 标定值的距离越近是紫色的概率越大。range 调太大会消除更多的颜色，调太小可能会消不全。 取值范围: [0, 255] 数据类型: CVI_U8
HueTh	是否使用 Hue 修正后像素的阈值。小于等于这个阈值认为使用 Hue 修正后的像素；大于这个阈值使用原像素与修正后像素 blend 之后的结果 取值范围: [0, 255] 数据类型: CVI_U8

下页继续

表 26.5 – 续上页

成员名称	描述
EdgeThLut	根据纹理大小做分段，以此查找对应的 Edge Weigting。 取值范围：[0, 255] 数据类型：CVI_U8
GDiffLut	Gradient match 模块里算 G 通道差异的 LUT，一版不需要更改。整个 LUT 分段开始和结束的值越小，判断越激进，紫边消除效果越强 取值范围：[0, 127] 数据类型：CVI_U8
stManual	手动参数
stAuto	自动参数

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetPFRAAttr
- CVI_ISP_GetPFRAAttr

27 FSHDR

27.1 功能描述

宽动态范围合成相关参数。

27.2 API 参考

- `CVI_ISP_SetFSHDRAttr`：设置帧合成属性参数
- `CVI_ISP_GetFSHDRAttr`：获取帧合成属性参数

27.2.1 CVI_ISP_SetFSHDRAttr

【描述】

设置帧合成属性参数

【语法】

```
CVI_S32 CVI_ISP_SetFSHDRAttr(VI_PIPE ViPipe, const ISP_FSHDR_ATTR_SET *pstFSHDRAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstFSHDRAttr	帧合成属性参数	输入

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

```
VI_PIPE ViPipe = 0;
ISP_FSHDR_ATTR_S stAttr;
CVI_ISP_GetFSHDRAttr(ViPipe, &stAttr);
stAttr.enOpType = OP_TYPE_AUTO;
CVI_ISP_SetFSHDRAttr(ViPipe, &stAttr);
```

【相关主题】

- [CVI_ISP_GetFSHDRAttr](#)

27.2.2 CVI_ISP_GetFSHDRAttr

【描述】

获取帧合成属性参数

【语法】

```
CVI_S32 CVI_ISP_GetFSHDRAttr(VI_PIPE ViPipe, ISP_FSHDR_ATTR_S *pstFSHDRAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstFSHDRAttr	帧合成属性参数	输出

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

无

【相关主题】

- CVI_ISP_SetFSHDRAttr

27.3 数据类型

- ISP_FSHDR_ATTR_S：定义帧合成属性参数

27.3.1 ISP_FSHDR_ATTR_S

【说明】

定义帧合成属性参数

【定义】

```
typedef struct _ISP_FSHDR_ATTR_S {
    ISP_OP_TYPE_E enOpType;
    CVI_U8 UpdateInterval; /*Rw; Range:[0, 255]*/
    CVI_BOOL FusionEnable; /*Rw; Range:[0, 1]*/
    CVI_U8 FusionMode; /*Rw; Range:[0, 2]*/
    CVI_U8 FusionDbgMode; /*Rw; Range:[0, 4]*/
    CVI_U8 FusionYsel; /*Rw; Range:[0, 2]*/
    CVI_U8 FusionBldNsel; /*Rw; Range:[0, 1]*/
    CVI_U8 FusionBldDsel; /*Rw; Range:[0, 1]*/
    CVI_U16 FusionDS1Th; /*Rw; Range:[0, 4095]*/
    CVI_U8 FusionDS1Step; /*Rw; Range:[0, 255]*/
    CVI_U8 FusionDS1Lb; /*Rw; Range:[0, 255]*/
    CVI_U16 FusionDS2Th; /*Rw; Range:[0, 4095]*/
    CVI_U8 FusionDS2Step; /*Rw; Range:[0, 255]*/
    CVI_U8 FusionDS2Lb; /*Rw; Range:[0, 255]*/
    CVI_U16 LENormBldRange; /*Rw; Range:[0, 4095]*/
    CVI_U16 LENormBldP0; /*Rw; Range:[0, 4095]*/
    CVI_U16 SENormBldRange; /*Rw; Range:[0, 4095]*/
    CVI_U16 SENormBldP0; /*Rw; Range:[0, 4095]*/
    CVI_U16 LEDynmBldRange; /*Rw; Range:[0, 4095]*/
    CVI_U16 LEDynmBldP0; /*Rw; Range:[0, 4095]*/
    CVI_U16 SEDynmBldRange; /*Rw; Range:[0, 4095]*/
    CVI_U16 SEDynmBldP0; /*Rw; Range:[0, 4095]*/
    CVI_U16 NDBldLumTh; /*Rw; Range:[0, 256]*/
    CVI_U8 NDBldDarkWt; /*Rw; Range:[0, 16]*/
    CVI_U8 NDBldDiffLut[FS_DIFF_DW_LENGTH]; /*Rw; Range:[0, 16]*/
    CVI_BOOL MCurveEnable; /*Rw; Range:[0, 1]*/
    ISP_MCURVE_MODE_E MCurveMode;
    CVI_U16 MCurveAutoSEMin; /*Rw; Range:[0, 4095]*/
    CVI_U16 MCurveManualSEMax; /*Rw; Range:[0, 4095]*/
    CVI_U8 MCurveSmooth; /*Rw; Range:[0, 255]*/
    CVI_U16 MCurveDelta; /*Rw; Range:[0, 8560]*/
    CVI_U16 MCurveX1; /*Rw; Range:[0, 65535]*/
    CVI_U8 MCurveBldRatio; /*Rw; Range:[0, 10]*/
    CVI_U16 MCurveXMaxRatio; /*Rw; Range:[0, 256]*/
    CVI_U8 MCurveYsel; /*Rw; Range:[0, 2]*/
}
```

(下页继续)

(续上页)

```

    CVI_U8 MCurveYvWet; /*Rw; Range:[0, 8]*/
    CVI_U8 MCurveFlumWet[FS_FLUMW_LUT_LENGTH]; /*Rw; Range:[0, 255]*/
} ISP_FSHDR_ATTR_S;

```

【成员】

成员名称	描述
enOpType	工作类型 OP_TYPE_AUTO: 自动模式 OP_TYPE_MANUAL: 手动模式 数据类型: ISP_OP_TYPE_E
UpdateInterval	影响参数更新间隔, 值越大画面变化越慢, 效能越好。 取值范围: [0, 255] 数据类型: CVI_U8
FusionEnable	HDR Fusion 使能。HDR mode 设置 1; linear mode 设置 0。 取值范围: [0, 1] 数据类型: CVI_BOOL
FusionMode	选择输出结果。 0: 输出 HDR Fusion 的结果; 1: 输出 SE 结果; 2: 输出 LE 结果; 取值范围: [0, 2] 数据类型: CVI_U8
FusionDbgMode	设置调试模式 取值范围: [0, 4] 数据类型: CVI_U8
FusionYsel	用 Bayer 计算 Luma 的方式, 0: RGB 的平均值, 1: G 的平均值, 2: 原始 Bayer 值 取值范围: [0, 2] 数据类型: CVI_U8
FusionBldNsel	计算静态融合权重的标准, 0: 以 LE 的亮度为标准计算静态融合权重, 1: 以 SE 的亮度为标准计算静态融合权重 取值范围: [0, 1] 数据类型: CVI_U8
FusionBldDsel	计算动态融合权重的标准, 0: 以 LE 的亮度为标准计算动态融合权重, 1: 以 SE 的亮度为标准计算动态融合权重 取值范围: [0, 1] 数据类型: CVI_U8
FusionDS1Th	控制 SE 的 Y 和 raw 开始融合的范围, 值越小, 开始融合的亮度越低, 越少的暗区被降低饱和度 取值范围: [0, 4095] 数据类型: CVI_U16
FusionDS1Step	控制 SE 的 Y 和 raw 融合的过渡区间, 值越小, 过渡越平缓 取值范围: [0, 255] 数据类型: CVI_U8

下页继续

表 27.3 – 续上页

成员名称	描述
FusionDS1Lb	控制 SE 的 Y 和 raw 的融合权重，值越小，暗区使用更多 Y，暗区饱和度越低 取值范围：[0, 255] 数据类型：CVI_U8
FusionDS2Th	控制 LE 的 Y 和 raw 开始融合的范围，值越小，开始融合的亮度越低，越少的暗区被降低饱和度 取值范围：[0, 4095] 数据类型：CVI_U16
FusionDS2Step	控制 LE 的 Y 和 raw 融合的过渡区间，值越小，过渡越平缓 取值范围：[0, 255] 数据类型：CVI_U8
FusionDS2Lb	控制 LE 的 Y 和 raw 的融合权重，值越小，暗区使用更多 Y，暗区饱和度越低 取值范围：[0, 255] 数据类型：CVI_U8
LENormBldRange	LE Normal blending curve range，静止区长短曝融合时，以长曝为基准的过渡区域范围。高于 LENormBldP0+ LENormBldRange 的图像数据将只选择短曝光数据合成 HDR 图像。 取值范围：[0, 4095] 数据类型：CVI_U16
LENormBldP0	LE Normal blending curve P0，静止区长短曝融合时，以长曝为基准的临界值，低于该临界值的图像数据将只选择长曝光数据合成 HDR 图像。 取值范围：[0, 4095] 数据类型：CVI_U16
SENormBldRange	SE Normal blending curve range，静止区长短曝融合时，以短曝为基准的过渡区域范围。高于 SENormBldP0+SENormBldRange 的图像数据将只选择短曝光数据合成 HDR 图像。 取值范围：[0, 4095] 数据类型：CVI_U16
SENormBldP0	SE Normal blending curve P0，静止区长短曝融合时，以短曝为基准的临界值，低于该临界值的图像数据将只选择长曝光数据合成 HDR 图像。 取值范围：[0, 4095] 数据类型：CVI_U16
LEDynmBldRange	LE Dynamic blending curve range，运动区长短曝融合时，以长曝为基准的过渡区域范围。高于 LENormBldP0+ LENormBldRange 的图像数据将只选择短曝光数据合成 HDR 图像。 取值范围：[0, 4095] 数据类型：CVI_U16
LEDynmBldP0	LE Dynamic blending curve P0，运动区长短曝融合时，以长曝为基准的临界值，低于该临界值的图像数据将只选择长曝光数据合成 HDR 图像。 取值范围：[0, 4095] 数据类型：CVI_U16

下页继续

表 27.3 – 续上页

成员名称	描述
SEDynmBldRange	SE Dynamic blending curve range, 运动区长短曝融合时, 以短曝为基准的过渡区域范围。高于 SENormBldP0+ SENormBldRange 的图像数据将只选择短曝光数据合成 HDR 图像。 取值范围: [0, 4095] 数据类型: CVI_U16
SEDynmBldP0	SE Dynamic blending curve P0, 运动区长短曝融合时, 以短曝为基准的临界值, 低于该临界值的图像数据将只选择长曝光数据合成 HDR 图像。 取值范围: [0, 4095] 数据类型: CVI_U16
NDBldLumTh	动静区权重融合的亮度阈值, 值越小, 对越暗区的运动越敏感 取值范围: [0, 256] 数据类型: CVI_U16
NDBldDarkWt	动静区权重融合的最小值, 值越大, 最暗区使用的动态权重越大 取值范围: [0, 16] 数据类型: CVI_U8
NDBldDiffLut	动静区权重融合的查找表, 值越大, 对运动区越敏感, 动态权重越大 取值范围: [0, 16] 数据类型: CVI_U8
MCurveEnable	Map Curve 的开关。HDR mode 设置 1; linear mode 设置 0。 取值范围: [0, 1] 数据类型: CVI_BOOL
MCurveMode	Auto Map Curve 的控制开关。0: Auto 模式, 自动生成 Curve, 1: Manual 模式, 设置 MCurveManualSEMax 调整 Curve 最大值。 数据类型: ISP_MCURVE_MODE_E
MCurveAutoSEMin	限制自动模式下参与计算的短帧最大值的下限, 可以改善短帧过暗时对画面亮度抬升过大的问题 取值范围: [0, 4095] 数据类型: CVI_U16
MCurveManualSEMax	手动设置 Curve 横坐标最大值, 在 Manual 模式下使用。值越小, 全局提亮程度越大。 取值范围: [0, 4095] 数据类型: CVI_U16
MCurveSmooth	调整 mcurve 变化的快慢, 值越大越慢 取值范围: [0, 255] 数据类型: CVI_U8
MCurveDelta	决定 Map Curve 暗部的斜率。值越小, 暗部提亮越明显。 取值范围: [0, 8560] 数据类型: CVI_U16
MCurveX1	对于亮度大于 X1 的亮区做对比度保护。值越小, 对比度保护区域越大。 取值范围: [0, 65535] 数据类型: CVI_U16

下页继续

表 27.3 – 续上页

成员名称	描述
MCurveBldRatio	亮区对比度保护程度。值越小，保护程度越大。0: linear, 10: curve 取值范围: [0, 10] 数据类型: CVI_U8
MCurveXMaxRatio	调整暗区提亮程度。在 Auto 模式下使用。128 表示 Map Curve 完全自动生成。小于 128 时，值越小，暗区提亮程度越大。大于 128 时，值越大，画面整体压暗程度越大。 取值范围: [0, 256] 数据类型: CVI_U16
MCurveYsel	用 Bayer 计算 Luma 的方式, 0: RGB 的平均值 (4G+2B+2R), 1: G 的平均值 (4G), 2: 原始 Bayer 值 取值范围: [0, 2] 数据类型: CVI_U8
MCurveYvWet	Y 和 LocalMax 融合的权重，值越大越偏向 LocalMax, 0: 全取 y, 8: 全取 v 取值范围: [0, 8] 数据类型: CVI_U8
MCurveFlumWet	Y 和 Raw 融合的权重表，值越大越偏向 Raw, 0: 全用 ymean 取值范围: [0, 255] 数据类型: CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetFSHDRAAttr
- CVI_ISP_GetFSHDRAAttr

28 DRC

28.1 功能描述

整幅图像的动态范围，使之能在显示设备上的显示效果与人眼视觉感受一致。

28.2 API 参考

- `CVI_ISP_SetDRCAttr`：设置 DRC 属性参数
- `CVI_ISP_GetDRCAttr`：获取 DRC 属性参数

28.2.1 CVI_ISP_SetDRCAttr

【描述】

设置 DRC 属性参数

【语法】

```
CVI_S32 CVI_ISP_SetDRCAttr(VI_PIPE ViPipe, const ISP_DRC_ATTR_S *pstDRCAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstDRCAttr	DRC 属性参数	输入

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`

- 库文件: libisp.so

【注意】

无

【举例】

```
VI_PIPE ViPipe = 0;
ISP_DRC_ATTR_S stAttr;
CVI_ISP_GetDRCAttr(ViPipe, &stAttr);
stAttr.enOpType = OP_TYPE_AUTO;
CVI_ISP_SetDRCAttr(ViPipe, &stAttr);
```

【相关主题】

- [CVI_ISP_GetDRCAttr](#)

28.2.2 CVI_ISP_GetDRCAttr

【描述】

获取 DRC 属性参数

【语法】

```
CVI_S32 CVI_ISP_GetDRCAttr(VI_PIPE ViPipe, ISP_DRC_ATTR_S *pstDRCAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstDRCAttr	DRC 属性参数	输出

【返回值】

返回值	描述
0	成功
非 0	失败, 其值为错误码。

【需求】

- 头文件: cvi_isp.h, cvi_comm_isp.h
- 库文件: libisp.so

【注意】

无

【举例】

无

【相关主题】

- CVI_ISP_SetDRCAttr

28.3 数据类型

- ISP_DRC_MANUAL_ATTR_S：定义手动 DRC 属性参数
- ISP_DRC_AUTO_ATTR_S：定义自动 DRC 属性参数
- ISP_DRC_ATTR_S：定义 DRC 属性参数

28.3.1 ISP_DRC_MANUAL_ATTR_S

【说明】

定义手动 DRC 属性参数

【定义】

```
typedef struct ISP_DRC_MANUAL_ATTR_S {  
    CVI_U8 Strength; /*Rw; Range:[0, 255]*/  
    CVI_U8 StrengthMin; /*Rw; Range:[0, 255]*/  
    CVI_U8 StrengthMax; /*Rw; Range:[0, 255]*/  
} ISP_DRC_MANUAL_ATTR_S;
```

【成员】

成员名称	描述
Strength	控制 DRC 的整体强度。值越大，强度越强。 取值范围：[0, 255] 数据类型：CVI_U8
StrengthMin	autoMode=1 时生效，控制自动算出来的 drc 强度 *gain 的范围。 取值范围：[0, 255] 数据类型：CVI_U8
StrengthMax	autoMode=1 时生效，控制自动算出来的 drc 强度 *gain 的范围。 取值范围：[0, 255] 数据类型：CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetDRCAttr
- CVI_ISP_GetDRCAttr

28.3.2 ISP_DRC_AUTO_ATTR_S

【说明】

定义自动 DRC 属性参数

【定义】

```
typedef struct _ISP_DRC_AUTO_ATTR_S {  
    CVI_U8 Strength[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/  
    CVI_U8 StrengthMin[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/  
    CVI_U8 StrengthMax[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 255]*/  
} ISP_DRC_AUTO_ATTR_S;
```

【成员】

成员名称	描述
Strength	控制 DRC 的整体强度。值越大，强度越强。 取值范围：[0, 255] 数据类型：CVI_U8
StrengthMin	autoMode=1 时生效，控制自动算出来的 drc 强度 *gain 的范围。 取值范围：[0, 255] 数据类型：CVI_U8
StrengthMax	autoMode=1 时生效，控制自动算出来的 drc 强度 *gain 的范围。 取值范围：[0, 255] 数据类型：CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetDRCAttr
- CVI_ISP_GetDRCAttr

28.3.3 ISP_DRC_ATTR_S

【说明】

定义 DRC 属性参数

【定义】

```
typedef struct _ISP_DRC_ATTR_S {  
    CVI_BOOL Enable; /*Rw; Range:[0, 1]*/  
    ISP_OP_TYPE_E enOpType;  
    CVI_U8 UpdateInterval; /*Rw; Range:[0, 255]*/  
    CVI_BOOL AutoMode; /*Rw; Range:[0, 1]*/  
}
```

(下页继续)

(续上页)

```

CVI_U16 SaturationLow; /*Rw; Range:[0, 4095]*/
CVI_U16 SaturationHigh; /*Rw; Range:[0, 4095]*/
CVI_U16 ContrastLow; /*Rw; Range:[0, 4095]*/
CVI_U16 ContrastHigh; /*Rw; Range:[0, 4095]*/
CVI_U8 BlockNumW; /*Rw; Range:[7, 47]*/
CVI_U8 GainUB; /*Rw; Range:[0, 255]*/
CVI_U8 GainLB; /*Rw; Range:[0, 255]*/
CVI_U8 GainProtStr; /*Rw; Range:[0, 255]*/
CVI_U8 FSBPstr; /*Rw; Range:[0, 255]*/
CVI_U8 DeFlareMode; /*Rw; Range:[0, 2]*/
CVI_U8 DeFlareRatio; /*Rw; Range:[0, 255]*/
CVI_U8 DeFlareSlope; /*Rw; Range:[0, 255]*/
CVI_U16 AdaptiveLowStr; /*Rw; Range:[0, 4095]*/
CVI_U16 AdaptiveHighStr; /*Rw; Range:[0, 4095]*/
CVI_U16 AdaptiveRatio; /*Rw; Range:[0, 1000]*/
CVI_U8 ColorProtStr; /*Rw; Range:[0, 255]*/
CVI_U16 ColorProtThr; /*Rw; Range:[0, 4095]*/
ISP_DRC_MANUAL_ATTR_S stManual;
ISP_DRC_AUTO_ATTR_S stAuto;
} ISP_DRC_ATTR_S;

```

【成员】

成员名称	描述
Enable	模块使能开关 取值范围：[0, 1] 数据类型：CVI_BOOL
enOpType	工作类型 OP_TYPE_AUTO：自动模式 OP_TYPE_MANUAL：手动模式 数据类型：ISP_OP_TYPE_E
UpdateInterval	影响参数更新间隔，值越大画面变化越慢，效能越好。 取值范围：[0, 255] 数据类型：CVI_U8
AutoMode	控制 auto drc 取值范围：[0, 1] 数据类型：CVI_BOOL
SaturationLow	控制 Y 和 localMax 的融合比例 取值范围：[0, 4095] 数据类型：CVI_U16
SaturationHigh	控制 Y 和 local Max 的融合比例 取值范围：[0, 4095] 数据类型：CVI_U16
ContrastLow	控制第一轮融合的输出和 raw 的融合比例 取值范围：[0, 4095] 数据类型：CVI_U16
ContrastHigh	控制第一轮融合的输出和 raw 的融合比例 取值范围：[0, 4095] 数据类型：CVI_U16

下页继续

表 28.5 – 续上页

成员名称	描述
BlockNumW	控制局部亮度控制的范围 取值范围：[7, 47] 数据类型：CVI_U8
GainUB	1 为 1 倍，限制算出来的 out_luma 为 in_luma 的多少倍以内 取值范围：[0, 255] 数据类型：CVI_U8
GainLB	256 为 1 倍，限制算出来的 out_luma 为 in_luma 的多少倍以上 取值范围：[0, 255] 数据类型：CVI_U8
GainProtStr	限制产生色彩偏移的 DRC gain，值越大，限制强度越大，色彩保护越强 取值范围：[0, 255] 数据类型：CVI_U8
FSBPstr	对高反差区域做特别处理 取值范围：[0, 255] 数据类型：CVI_U8
DeFlareMode	光晕抑制的模式。0：关闭，1：手动抑制模式，2：自适应抑制模式 取值范围：[0, 2] 数据类型：CVI_U8
DeFlareRatio	控制光晕抑制的强度，Ratio 越小，抑制程度越大，光晕越弱，对比度越弱 取值范围：[0, 255] 数据类型：CVI_U8
DeFlareSlope	控制光晕抑制的强度，Slope 越小，抑制程度越大，光晕越弱，对比度越弱 取值范围：[0, 255] 数据类型：CVI_U8
AdaptiveLowStr	根据亮度自适应控制 DRC 的强度的下限 取值范围：[0, 4095] 数据类型：CVI_U16
AdaptiveHighStr	根据亮度自适应控制 DRC 的强度的上限 取值范围：[0, 4095] 数据类型：CVI_U16
AdaptiveRatio	根据亮度自适应控制 DRC 的强度的过渡斜率 取值范围：[0, 1000] 数据类型：CVI_U16
ColorProtStr	限制产生色彩偏移的 DRC gain，值越大，限制强度越大，色彩保护越强 取值范围：[0, 255] 数据类型：CVI_U8
ColorProtThr	控制色彩保护的亮度范围，值越大，开启保护的亮度越偏向高亮区 取值范围：[0, 4095] 数据类型：CVI_U16
stManual	手动参数

下页继续

表 28.5 – 续上页

成员名称	描述
stAuto	自动参数

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetDRCAAttr
- CVI_ISP_GetDRCAAttr

29 MONO

29.1 功能描述

设置 Mono 属性参数

29.2 API 参考

- `CVI_ISP_SetMonoAttr`：设置 Mono 参数属性
- `CVI_ISP_GetMonoAttr`：获取 Mono 参数属性

29.2.1 CVI_ISP_SetMonoAttr

【描述】

设置 Mono 参数属性

【语法】

```
CVI_S32 CVI_ISP_SetMonoAttr(VI_PIPE ViPipe, const ISP_MONO_ATTR_S *pstMonoAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstMonoAttr	Mono 参数属性	输入

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`

- 库文件: libisp.so

【注意】

无

【举例】

```
VI_PIPE ViPipe = 0;
ISP_MONO_ATTR_S stAttr;
CVI_ISP_GetMonoAttr(ViPipe, &stAttr);
stAttr.enOpType = OP_TYPE_AUTO;
CVI_ISP_SetMonoAttr(ViPipe, &stAttr);
```

【相关主题】

- [CVI_ISP_GetMonoAttr](#)

29.2.2 CVI_ISP_GetMonoAttr

【描述】

获取 Mono 参数属性

【语法】

```
CVI_S32 CVI_ISP_GetMonoAttr(VI_PIPE ViPipe, ISP_MONO_ATTR_S *pstMonoAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstMonoAttr	Mono 参数属性	输出

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: cvi_isp.h, cvi_comm_isp.h
- 库文件: libisp.so

【注意】

无

【举例】

无

【相关主题】

- CVI_ISP_SetMonoAttr

29.3 数据类型

- ISP_MONO_ATTR_S：定义 Mono 参数属性

29.3.1 ISP_MONO_ATTR_S

【说明】

定义 Mono 参数属性

【定义】

```
typedef struct _ISP_MONO_ATTR_S {  
    CVI_BOOL Enable; /*Rw; Range:[0, 1]*/  
    CVI_U8 UpdateInterval; /*Rw; Range:[1, 255]*/  
} ISP_MONO_ATTR_S;
```

【成员】

成员名称	描述
Enable	模块使能开关 取值范围：[0, 1] 数据类型：CVI_BOOL
UpdateInterval	影响参数更新间隔，值越大画面变化越慢，效能越好。 取值范围：[1, 255] 数据类型：CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetMonoAttr
- CVI_ISP_GetMonoAttr

30 YCONTRAST

30.1 功能描述

Y 值域线性对比度。

30.2 API 参考

- `CVI_ISP_SetYContrastAttr`：设置 Y Contrast 参数属性
- `CVI_ISP_GetYContrastAttr`：获取 Y Contrast 参数属性

30.2.1 CVI_ISP_SetYContrastAttr

【描述】

设置 Y Contrast 参数属性

【语法】

```
CVI_S32 CVI_ISP_SetYContrastAttr(VI_PIPE ViPipe, const ISP_YCONTRAST_ATTR_S_
↪ *pstYContrastAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstYContrastAttr	Y Contrast 参数属性	输入

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

```
VI_PIPE ViPipe = 0;
ISP_YCONTRAST_ATTR_S stAttr;
CVI_ISP_GetYContrastAttr(ViPipe, &stAttr);
stAttr.enOpType = OP_TYPE_AUTO;
CVI_ISP_SetYContrastAttr(ViPipe, &stAttr);
```

【相关主题】

- [CVI_ISP_GetYContrastAttr](#)

30.2.2 CVI_ISP_GetYContrastAttr

【描述】

获取 Y Contrast 参数属性

【语法】

```
CVI_S32 CVI_ISP_GetYContrastAttr(VI_PIPE ViPipe, ISP_YCONTRAST_ATTR_S_
→*pstYContrastAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstYContrastAttr	Y Contrast 参数属性	输出

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

无

【相关主题】

- CVI_ISP_SetYContrastAttr

30.3 数据类型

- ISP_YCONTRAST_MANUAL_ATTR_S：定义手动 Y Contrast 参数属性
- ISP_YCONTRAST_AUTO_ATTR_S：定义自动 Y Contrast 参数属性
- ISP_YCONTRAST_ATTR_S：定义 Y Contrast 参数属性

30.3.1 ISP_YCONTRAST_MANUAL_ATTR_S

【说明】

定义手动 Y Contrast 参数属性

【定义】

```
typedef struct _ISP_YCONTRAST_MANUAL_ATTR_S {  
    CVI_U8 ContrastLow; /*Rw; Range:[0, 100]*/  
    CVI_U8 ContrastHigh; /*Rw; Range:[0, 100]*/  
    CVI_U8 CenterLuma; /*Rw; Range:[0, 64]*/  
} ISP_YCONTRAST_MANUAL_ATTR_S;
```

【成员】

成员名称	描述
ContrastLow	小于中心点位置的区域对比度强度。值越大，对比越强。 取值范围：[0, 100] 数据类型：CVI_U8
ContrastHigh	大于中心点位置的区域对比度强度。值越大，对比越强。 取值范围：[0, 100] 数据类型：CVI_U8
CenterLuma	中心点位置，会以中心点往两边加强对比。 取值范围：[0, 64] 数据类型：CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetYContrastAttr
- CVI_ISP_GetYContrastAttr

30.3.2 ISP_YCONTRAST_AUTO_ATTR_S

【说明】

定义自动 Y Contrast 参数属性

【定义】

```
typedef struct _ISP_YCONTRAST_AUTO_ATTR_S {
    CVI_U8 ContrastLow[ISP_AUTO_LV_NUM]; /*Rw; Range:[0, 100]*/
    CVI_U8 ContrastHigh[ISP_AUTO_LV_NUM]; /*Rw; Range:[0, 100]*/
    CVI_U8 CenterLuma[ISP_AUTO_LV_NUM]; /*Rw; Range:[0, 64]*/
} ISP_YCONTRAST_AUTO_ATTR_S;
```

【成员】

成员名称	描述
ContrastLow	小于中心点位置的区域对比度强度。值越大，对比越强。 取值范围：[0, 100] 数据类型：CVI_U8
ContrastHigh	大于中心点位置的区域对比度强度。值越大，对比越强。 取值范围：[0, 100] 数据类型：CVI_U8
CenterLuma	中心点位置，会以中心点往两边加强对比。 取值范围：[0, 64] 数据类型：CVI_U8

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetYContrastAttr
- CVI_ISP_GetYContrastAttr

30.3.3 ISP_YCONTRAST_ATTR_S

【说明】

定义 Y Contrast 参数属性

【定义】

```
typedef struct _ISP_YCONTRAST_ATTR_S {
    CVI_BOOL Enable; /*Rw; Range:[0, 1]*/
    ISP_OP_TYPE_E enOpType;
    CVI_U8 UpdateInterval; /*Rw; Range:[1, 255]*/
    ISP_YCONTRAST_MANUAL_ATTR_S stManual;
    ISP_YCONTRAST_AUTO_ATTR_S stAuto;
} ISP_YCONTRAST_ATTR_S;
```

【成员】

成员名称	描述
Enable	模块使能开关 取值范围：[0, 1] 数据类型：CVI_BOOL
enOpType	工作类型 OP_TYPE_AUTO：自动模式 OP_TYPE_MANUAL：手动模式 数据类型：ISP_OP_TYPE_E
UpdateInterval	影响参数更新间隔, 值越大画面变化越慢, 效能越好。 取值范围：[1, 255] 数据类型：CVI_U8
stManual	手动参数
stAuto	自动参数

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetYContrastAttr
- CVI_ISP_GetYContrastAttr

31 CA

31.1 功能描述

调整 UV domain 饱和度, 此模组分为 CA 模式与 CP (热成像) 模式, 同时间只能选一个开启。CA 模式: 根据 input Y 来调整饱和度, 因此可以局部调整饱和度。CP 模式: 因为热成像模式只有 Y 值, 会根据 Y 值查找预先调试的颜色模板, 查找对应的 YUV 值, 使其上色。

31.2 API 参考

- `CVI_ISP_SetCAAttr` : 设置 CA 参数属性
- `CVI_ISP_GetCAAttr` : 获取 CA 参数属性

31.2.1 CVI_ISP_SetCAAttr

【描述】

设置 CA 参数属性

【语法】

```
CVI_S32 CVI_ISP_SetCAAttr(VI_PIPE ViPipe, const ISP_CA_ATTR_S *pstCAAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstCAAttr	CA 参数属性	输入

【返回值】

返回值	描述
0	成功
非 0	失败, 其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

```
VI_PIPE ViPipe = 0;
ISP_CA_ATTR_S stAttr;
CVI_ISP_GetCAAttr(ViPipe, &stAttr);
stAttr.enOpType = OP_TYPE_AUTO;
CVI_ISP_SetCAAttr(ViPipe, &stAttr);
```

【相关主题】

- [CVI_ISP_GetCAAttr](#)

31.2.2 CVI_ISP_GetCAAttr

【描述】

获取 CA 参数属性

【语法】

```
CVI_S32 CVI_ISP_GetCAAttr(VI_PIPE ViPipe, ISP_CA_ATTR_S *pstCAAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstCAAttr	CA 参数属性	输出

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

无

【相关主题】

- [CVI_ISP_SetCAAttr](#)

31.3 数据类型

- [ISP_CA_MANUAL_ATTR_S](#)：定义手动 CA 参数属性
- [ISP_CA_AUTO_ATTR_S](#)：定义自动 CA 参数属性
- [ISP_CA_ATTR_S](#)：定义 CA 参数属性

31.3.1 ISP_CA_MANUAL_ATTR_S

【说明】

定义手动 CA 参数属性

【定义】

```
typedef struct _ISP_CA_MANUAL_ATTR_S {  
    CVI_U16 ISORatio; /*Rw; Range:[0, 2047]*/  
    CVI_U16 YRatioLut[256]; /*Rw; Range:[0, 2047]*/  
} ISP_CA_MANUAL_ATTR_S;
```

【成员】

成员名称	描述
ISORatio	CA 模式, 根据 ISO 值查找 UV 的增益。所有像素点的 UV 调整增益都是相同的, 建议在低 ISO 的时候此增益可以设置大一些, 而高 ISO 时此增益值可以设定小一些, 来抑制暗区的色噪。 取值范围: [0, 2047] 数据类型: CVI_U16
YRatioLut	CA 模式, 根据亮度 Y 查找 UV 的增益。此值根据不同亮度等级可设置不同的 UV 增益, 建议在亮区的增益可以设置大一些, 颜色会较为鲜艳, 而暗区的增益可以设置小一些, 来抑制暗区色噪。 取值范围: [0, 2047] 数据类型: CVI_U16

【注意事项】

无

【相关数据类型及接口】

- [CVI_ISP_SetCAAttr](#)
- [CVI_ISP_GetCAAttr](#)

31.3.2 ISP_CA_AUTO_ATTR_S

【说明】

定义自动 CA 参数属性

【定义】

```
typedef struct _ISP_CA_AUTO_ATTR_S {
    CVI_U16 ISORatio[ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 2047]*/
    CVI_U16 YRatioLut[256][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 2047]*/
} ISP_CA_AUTO_ATTR_S;
```

【成员】

成员名称	描述
ISORatio	CA 模式, 根据 ISO 值查找 UV 的增益。所有像素点的 UV 调整增益都是相同的, 建议在低 ISO 的时候此增益可以设置大一些, 而高 ISO 时此增益值可以设定小一些, 来抑制暗区的色噪。 取值范围: [0, 2047] 数据类型: CVI_U16
YRatioLut	CA 模式, 根据亮度 Y 查找 UV 的增益。此值根据不同亮度等级可设置不同的 UV 增益, 建议在亮区的增益可以设置大一些, 颜色会较为鲜艳, 而暗区的增益可以设置小一些, 来抑制暗区色噪。 取值范围: [0, 2047] 数据类型: CVI_U16

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetCAAttr
- CVI_ISP_GetCAAttr

31.3.3 ISP_CA_ATTR_S

【说明】

定义 CA 参数属性

【定义】

```
typedef struct _ISP_CA_ATTR_S {
    CVI_BOOL Enable; /*Rw; Range:[0, 1]*/
    ISP_OP_TYPE_E enOpType;
    CVI_U8 UpdateInterval; /*Rw; Range:[1, 255]*/
    CVI_BOOL CaCpMode; /*Rw; Range:[0, 1]*/
    CVI_U8 CPLutY[256]; /*Rw; Range:[0, 255]*/
    CVI_U8 CPLutU[256]; /*Rw; Range:[0, 255]*/
}
```

(下页继续)

(续上页)

```

    CVI_U8 CPLutV[256]; /*Rw; Range:[0, 255]*/
    ISP_CA_MANUAL_ATTR_S stManual;
    ISP_CA_AUTO_ATTR_S stAuto;
} ISP_CA_ATTR_S;

```

【成员】

成员名称	描述
Enable	模块使能开关 取值范围：[0, 1] 数据类型：CVI_BOOL
enOpType	工作类型 OP_TYPE_AUTO：自动模式 OP_TYPE_MANUAL：手动模式 数据类型：ISP_OP_TYPE_E
UpdateInterval	影响参数更新间隔，值越大画面变化越慢，效能越好。 取值范围：[1, 255] 数据类型：CVI_U8
CaCpMode	CA 模式选择： 0: CA 模式。 1: CP 模式。 取值范围：[0, 1] 数据类型：CVI_BOOL
CPLutY	CP 模式，根据亮度 Y 查找 LUT 对应的 Y 值。 取值范围：[0, 255] 数据类型：CVI_U8
CPLutU	CP 模式，根据亮度 Y 查找 LUT 对应的 U 值。 取值范围：[0, 255] 数据类型：CVI_U8
CPLutV	CP 模式，根据亮度 Y 查找 LUT 对应的 V 值。 取值范围：[0, 255] 数据类型：CVI_U8
stManual	手动参数
stAuto	自动参数

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetCAAttr
- CVI_ISP_GetCAAttr

32 CA2

32.1 功能描述

调整 UV domain 饱和度, 根据输入像素的饱和度, 重新调整饱和度等级

32.2 API 参考

- `CVI_ISP_SetCA2Attr` : 设置 CA2 参数属性
- `CVI_ISP_GetCA2Attr` : 获取 CA2 参数属性

32.2.1 CVI_ISP_SetCA2Attr

【描述】

设置 CA2 参数属性

【语法】

```
CVI_S32 CVI_ISP_SetCA2Attr(VI_PIPE ViPipe, const ISP_CA2_ATTR_S *pstCA2Attr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstCA2Attr	CA2 参数属性	输入

【返回值】

返回值	描述
0	成功
非 0	失败, 其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`

· 库文件: libisp.so

【注意】

无

【举例】

```
VI_PIPE ViPipe = 0;
ISP_CA2_ATTR_S stAttr;
CVI_ISP_GetCA2Attr(ViPipe, &stAttr);
stAttr.enOpType = OP_TYPE_AUTO;
CVI_ISP_SetCA2Attr(ViPipe, &stAttr);
```

【相关主题】

· [CVI_ISP_GetCA2Attr](#)

32.2.2 CVI_ISP_GetCA2Attr

【描述】

获取 CA2 参数属性

【语法】

```
CVI_S32 CVI_ISP_GetCA2Attr(VI_PIPE ViPipe, ISP_CA2_ATTR_S *pstCA2Attr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstCA2Attr	CA2 参数属性	输出

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: cvi_isp.h, cvi_comm_isp.h
- 库文件: libisp.so

【注意】

无

【举例】

无

【相关主题】

- CVI_ISP_SetCA2Attr

32.3 数据类型

- ISP_CA2_MANUAL_ATTR_S：定义手动 CA2 参数属性
- ISP_CA2_AUTO_ATTR_S：定义自动 CA2 参数属性
- ISP_CA2_ATTR_S：定义 CA2 参数属性

32.3.1 ISP_CA2_MANUAL_ATTR_S

【说明】

定义手动 CA2 参数属性

【定义】

```
typedef struct ISP_CA2_MANUAL_ATTR_S {  
    CVI_U8 Ca2In[6]; /*Rw; Range:[0, 192]*/  
    CVI_U16 Ca2Out[6]; /*Rw; Range:[0, 2047]*/  
} ISP_CA2_MANUAL_ATTR_S;
```

【成员】

成员名称	描述
Ca2In	由六个数值组成的数组，决定输入饱和度等级 取值范围：[0, 192] 数据类型：CVI_U8
Ca2Out	由六个数值组成的数组，定义输出的 UV 增益。根据输入饱和度查找 UV 的增益，值越大，饱和度越高；反之，则越小 取值范围：[0, 2047] 数据类型：CVI_U16

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetCA2Attr
- CVI_ISP_GetCA2Attr

32.3.2 ISP_CA2_AUTO_ATTR_S

【说明】

定义自动 CA2 参数属性

【定义】

```
typedef struct _ISP_CA2_AUTO_ATTR_S {  
    CVI_U8 Ca2In[6][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 192]*/  
    CVI_U16 Ca2Out[6][ISP_AUTO_ISO_STRENGTH_NUM]; /*Rw; Range:[0, 2047]*/  
} ISP_CA2_AUTO_ATTR_S;
```

【成员】

成员名称	描述
Ca2In	由六个数值组成的数组，决定输入饱和度等级 取值范围：[0, 192] 数据类型：CVI_U8
Ca2Out	由六个数值组成的数组，定义输出的 UV 增益。根据输入饱和度和查找 UV 的增益，值越大，饱和度越高；反之，则越小 取值范围：[0, 2047] 数据类型：CVI_U16

【注意事项】

无

【相关数据类型及接口】

- [CVI_ISP_SetCA2Attr](#)
- [CVI_ISP_GetCA2Attr](#)

32.3.3 ISP_CA2_ATTR_S

【说明】

定义 CA2 参数属性

【定义】

```
typedef struct _ISP_CA2_ATTR_S {  
    CVI_BOOL Enable; /*Rw; Range:[0, 1]*/  
    ISP_OP_TYPE_E enOpType;  
    CVI_U8 UpdateInterval; /*Rw; Range:[1, 255]*/  
    ISP_CA2_MANUAL_ATTR_S stManual;  
    ISP_CA2_AUTO_ATTR_S stAuto;  
} ISP_CA2_ATTR_S;
```

【成员】

成员名称	描述
Enable	模块使能开关 取值范围：[0, 1] 数据类型：CVI_BOOL
enOpType	工作类型 OP_TYPE_AUTO：自动模式 OP_TYPE_MANUAL：手动模式 数据类型：ISP_OP_TYPE_E
UpdateInterval	影响参数更新间隔，值越大画面变化越慢，效能越好。 取值范围：[1, 255] 数据类型：CVI_U8
stManual	手动参数
stAuto	自动参数

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetCA2Attr
- CVI_ISP_GetCA2Attr

33 CLUT

33.1 功能描述

通过一个 17x17x17 的 3D LUT, 在 RGB domain 上做线性转换。将 RGB 像素值查表内插得到新的 RGB 像素值, 可以借此调试颜色、亮度。

33.2 API 参考

- `CVI_ISP_SetClutAttr` : 设置 CLUT 属性参数
- `CVI_ISP_GetClutAttr` : 获取 CLUT 属性参数
- `CVI_ISP_SetClutHslAttr` : 设置 CLUT hsl 属性参数
- `CVI_ISP_GetClutHslAttr` : 获取 CLUT hsl 属性参数

33.2.1 CVI_ISP_SetClutAttr

【描述】

设置 CLUT 属性参数

【语法】

```
CVI_S32 CVI_ISP_SetClutAttr(VI_PIPE ViPipe, const ISP_CLUT_ATTR_S *pstClutAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstClutAttr	CLUT 属性参数	输入

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

```
VI_PIPE ViPipe = 0;
ISP_CLUT_ATTR_S stAttr;
CVI_ISP_GetClutAttr(ViPipe, &stAttr);
stAttr.enOpType = OP_TYPE_AUTO;
CVI_ISP_SetClutAttr(ViPipe, &stAttr);
```

【相关主题】

- [CVI_ISP_GetClutAttr](#)

33.2.2 CVI_ISP_GetClutAttr

【描述】

获取 CLUT 属性参数

【语法】

```
CVI_S32 CVI_ISP_GetClutAttr(VI_PIPE ViPipe, ISP_CLUT_ATTR_S *pstClutAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstClutAttr	CLUT 属性参数	输出

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

无

【相关主题】

- [CVI_ISP_SetClutAttr](#)

33.2.3 CVI_ISP_SetClutHslAttr

【描述】

设置 CLUT hsl 属性参数

【语法】

```
CVI_S32 CVI_ISP_SetClutHslAttr(VI_PIPE ViPipe, const ISP_CLUT_HSL_ATTR_S_
↪*pstClutHslAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstClutHslAttr	CLUT hsl 属性参数	输入

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

```
VI_PIPE ViPipe = 0;
ISP_CLUT_HSL_ATTR_S stAttr;
CVI_ISP_GetClutHslAttr(ViPipe, &stAttr);
stAttr.enOpType = OP_TYPE_AUTO;
CVI_ISP_SetClutHslAttr(ViPipe, &stAttr);
```

【相关主题】

- [CVI_ISP_GetClutHslAttr](#)

33.2.4 CVI_ISP_GetClutHslAttr

【描述】

获取 CLUT hsl 属性参数

【语法】

```
CVI_S32 CVI_ISP_GetClutHslAttr(VI_PIPE ViPipe, ISP_CLUT_HSL_ATTR_S *pstClutHslAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstClutHslAttr	CLUT hsl 属性参数	输出

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: cvi_isp.h, cvi_comm_isp.h
- 库文件: libisp.so

【注意】

无

【举例】

无

【相关主题】

- CVI_ISP_SetClutHslAttr

33.3 数据类型

- ISP_CLUT_ATTR_S：定义 CLUT 属性参数
- ISP_CLUT_HSL_ATTR_S：定义 CLUT hsl 属性参数

33.3.1 ISP_CLUT_ATTR_S

【说明】

定义 CLUT 属性参数

【定义】

```
typedef struct _ISP_CLUT_ATTR_S {  
    CVI_BOOL Enable; /*Rw; Range:[0, 1]*/  
    CVI_U8 UpdateInterval; /*Rw; Range:[0, 255]*/  
    CVI_U16 ClutR[ISP_CLUT_LUT_LENGTH]; /*Rw; Range:[0, 1023]*/  
    CVI_U16 ClutG[ISP_CLUT_LUT_LENGTH]; /*Rw; Range:[0, 1023]*/  
    CVI_U16 ClutB[ISP_CLUT_LUT_LENGTH]; /*Rw; Range:[0, 1023]*/  
} ISP_CLUT_ATTR_S;
```

【成员】

成员名称	描述
Enable	模块使能开关 取值范围：[0, 1] 数据类型：CVI_BOOL
UpdateInterval	保留，未使用参数。 取值范围：[0, 255] 数据类型：CVI_U8
ClutR	R 通道的 LUT。 取值范围：[0, 1023] 数据类型：CVI_U16
ClutG	G 通道的 LUT。 取值范围：[0, 1023] 数据类型：CVI_U16
ClutB	B 通道的 LUT。 取值范围：[0, 1023] 数据类型：CVI_U16

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetClutAttr
- CVI_ISP_GetClutAttr

33.3.2 ISP_CLUT_HSL_ATTR_S

【说明】

定义 CLUT hsl 属性参数

【定义】

```
typedef struct _ISP_CLUT_HSL_ATTR_S {  
    CVI_BOOL Enable; /*Rw; Range:[0, 1]*/  
    CVI_FLOAT Sigma; /*Rw; Range:[0, 64]*/  
    CVI_S16 HByH[ISP_CLUT_HUE_LENGTH]; /*Rw; Range:[-30, 30]*/  
    CVI_U16 SByH[ISP_CLUT_HUE_LENGTH]; /*Rw; Range:[0, 100]*/  
    CVI_U16 LByH[ISP_CLUT_HUE_LENGTH]; /*Rw; Range:[0, 100]*/  
    CVI_U16 SByS[ISP_CLUT_SAT_LENGTH]; /*Rw; Range:[0, 100]*/  
} ISP_CLUT_HSL_ATTR_S;
```

【成员】

成员名称	描述
Enable	hsl 模块使能。 取值范围：[0, 1] 数据类型：CVI_BOOL
Sigma	None 取值范围：[0, 64] 数据类型：CVI_FLOAT
HByH	长度 37 的数组，将 hue 36 等分，可设置每个 hue 范围旋转的角度。 取值范围：[-30, 30] 数据类型：CVI_S16
SByH	长度 37 的数组，将 hue 36 等分，可设置每个 hue 范围饱和度增益，50 为一倍。 取值范围：[0, 100] 数据类型：CVI_U16
LByH	长度 37 的数组，将 hue 36 等分，可设置每个 hue 范围亮度增益，50 为一倍。 取值范围：[0, 100] 数据类型：CVI_U16
SByS	长度 21 的数组，将饱和度 20 等分，可设置每个饱和度范围饱和度的增益，50 为一倍。 取值范围：[0, 100] 数据类型：CVI_U16

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetClutHslAttr
- CVI_ISP_GetClutHslAttr

34 csc

34.1 功能描述

该功能提供了色域转换时的一些相关画质设定，通过设定 hue、luma、saturation、contrast、色域转换标准规范类型，进而影响画面的效果，而无需知晓转换矩阵的计算过程，当然也可以让使用者直接通过设定转换矩阵的 coeff 和 offset，进而影响画质效果

34.2 API 参考

- CVI_ISP_SetCSCAttr：设置 CSC 属性参数
- CVI_ISP_GetCSCAttr：获取 CSC 属性参数

34.2.1 CVI_ISP_SetCSCAttr

【描述】

设置 CSC 属性参数

【语法】

```
CVI_S32 CVI_ISP_SetCSCAttr(VI_PIPE ViPipe, const ISP_CSC_ATTR_S *pstCSCAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstCSCAttr	CSC 属性参数	输入

【返回值】

返回值	描述
0	成功
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

```
VI_PIPE ViPipe = 0;
ISP_CSC_ATTR_S stAttr;
CVI_ISP_GetCSCAttr(ViPipe, &stAttr);
stAttr.enOpType = OP_TYPE_AUTO;
CVI_ISP_SetCSCAttr(ViPipe, &stAttr);
```

【相关主题】

- [CVI_ISP_GetCSCAttr](#)

34.2.2 CVI_ISP_GetCSCAttr

【描述】

获取 CSC 属性参数

【语法】

```
CVI_S32 CVI_ISP_GetCSCAttr(VI_PIPE ViPipe, ISP_CSC_ATTR_S *pstCSCAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstCSCAttr	CSC 属性参数	输出

【返回值】

返回值	描述
0	成功
非 0	失败, 其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无

【举例】

无

【相关主题】

- CVI_ISP_SetCSCAttr

34.3 数据类型

- ISP_CSC_ATTR_S：定义 CSC 属性参数
- ISP_CSC_MATRX_S：定义自定义转换矩阵

34.3.1 ISP_CSC_ATTR_S

【说明】

定义 CSC 属性参数

【定义】

```
typedef struct _ISP_CSC_ATTR_S {  
    CVI_BOOL Enable; /*Rw; Range:[0, 1]*/  
    ISP_CSC_COLORGAMUT enColorGamut;  
    CVI_U8 UpdateInterval; /*Rw; Range:[0, 255]*/  
    CVI_U8 Hue; /*Rw; Range:[0, 100]*/  
    CVI_U8 Luma; /*Rw; Range:[0, 100]*/  
    CVI_U8 Contrast; /*Rw; Range:[0, 100]*/  
    CVI_U8 Saturation; /*Rw; Range:[0, 100]*/  
    ISP_CSC_MATRX_S stUserMatrx;  
} ISP_CSC_ATTR_S;
```

【成员】

成员名称	描述
Enable	模块使能开关 取值范围：[0, 1] 数据类型：CVI_BOOL
enColorGamut	色域转换类型选择 0: ISP_CSC_COLORGAMUT_BT601 1: ISP_CSC_COLORGAMUT_BT709 2: ISP_CSC_COLORGAMUT_BT2020 3: ISP_CSC_COLORGAMUT_USER 数据类型：ISP_CSC_COLORGAMUT
UpdateInterval	影响参数更新间隔，值越大画面变化越慢，效能越好。 取值范围：[0, 255] 数据类型：CVI_U8
Hue	调节色度 取值范围：[0, 100] 数据类型：CVI_U8

下页继续

表 34.3 – 续上页

成员名称	描述
Luma	调节亮度 取值范围：[0, 100] 数据类型：CVI_U8
Contrast	调节对比度 取值范围：[0, 100] 数据类型：CVI_U8
Saturation	调节饱和度 取值范围：[0, 100] 数据类型：CVI_U8
stUserMatrx	自定义色域转换矩阵 数据类型：ISP_CSC_MATRX_S

【注意事项】

无

【相关数据类型及接口】

- CVI_ISP_SetCSCAttr
- CVI_ISP_GetCSCAttr

34.3.2 ISP_CSC_MATRX_S

【说明】

定义自定义转换矩阵

【定义】

```
typedef struct _ISP_CSC_MATRX_S {  
    CVI_S16 userCscCoef[CSC_MATRIX_SIZE]; /*Rw; Range:[-8192, 8191]*/  
    CVI_S16 userCscOffset[CSC_OFFSET_SIZE]; /*Rw; Range:[-256, 255]*/  
} ISP_CSC_MATRX_S;
```

【成员】

成员名称	描述
userCscCoef	3*3 色域转换矩阵的系数 取值范围：[-8192, 8191] 数据类型：CVI_S16
userCscOffset	输出 offset 取值范围：[-256, 255] 数据类型：CVI_S16

【注意事项】

无

【相关数据类型及接口】

35 VC

35.1 功能描述

调整 Video Codec 端 Motion Map 设定

35.2 API 参考

- CVI_ISP_SetVCAAttr：设置 VC 属性参数
- CVI_ISP_GetVCAAttr：获取 VC 属性参数

35.2.1 CVI_ISP_SetVCAAttr

【描述】

设置 VC 属性参数

【语法】

```
CVI_S32 CVI_ISP_SetVCAAttr(VI_PIPE ViPipe, const ISP_VC_ATTR_S *pstVCAAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstVCAAttr	VC 属性参数	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，其值为错误码。

【需求】

- 头文件: cvi_isp.h, cvi_comm_isp.h

- 库文件: libisp.so

【注意】

无。

【举例】

无。

【相关主题】

- [CVI_ISP_GetVCAAttr](#)

35.2.2 CVI_ISP_GetVCAAttr

【描述】

获取 VC 属性参数

【语法】

```
CVI_S32 CVI_ISP_GetVCAAttr(VI_PIPE ViPipe, ISP_VC_ATTR_S *pstVCAAttr);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstVCAAttr	VC 属性参数	输出

【返回值】

返回值	描述
0	成功。
非 0	失败，其值为错误码。

【需求】

- 头文件: cvi_isp.h, cvi_comm_isp.h
- 库文件: libisp.so

【注意】

无。

【举例】

无。

【相关主题】

- [CVI_ISP_SetVCAAttr](#)

35.3 数据类型

- `ISP_VC_ATTR_S` : VC 属性参数

35.3.1 `ISP_VC_ATTR_S`

【说明】

VC 属性参数

【定义】

```
typedef struct _ISP_VC_ATTR_S {  
    CVI_U8 UpdateInterval;  
    CVI_U8 MotionThreshold[ISP_AUTO_ISO_STRENGTH_NUM];  
} ISP_VC_ATTR_S;
```

【成员】

成员名称	输入/输出
UpdateInterval	影响参数更新间隔，值越大画面变化越慢，效能越好 取值范围：[0x0, 0xff] 数据类型：CVI_U8
MotionThreshold	Motion Map 阈值 取值范围：[0x0, 0xff] 数据类型：CVI_U8

【注意事项】

无。

【相关数据类型及接口】

- `CVI_ISP_SetVCAttr`
- `CVI_ISP_GetVCAttr`

36 统计讯息

36.1 概述

本章节说明 ISP 提供的 3A 统计信息，及配置方式。

36.2 API 参考

统计讯息的接口必需在调用 CVI_ISP_Init 接口之后才能调用。

- CVI_ISP_SetStatisticsConfig：设置 ISP 统计信息配置
- CVI_ISP_GetStatisticsConfig：获取 ISP 统计信息配置
- CVI_ISP_GetAESTatistics：获取 ISP AE 统计信息
- CVI_ISP_GetWBStatistics：获取 ISP AWB 统计信息
- CVI_ISP_GetFocusStatistics：获取 ISP AF 统计信息

36.2.1 CVI_ISP_SetStatisticsConfig

【描述】

设置 ISP 统计讯息配置。

【语法】

```
CVI_S32 CVI_ISP_SetStatisticsConfig(VI_PIPE ViPipe, const ISP_STATISTICS_CFG_S_
→*pstStatCfg);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstStatCfg	ISP 统计信息配置	输入

【返回值】

返回值	描述
0	成功。
非 0	失败，其值为错误码。

【需求】

- 头文件: cvi_isp.h, cvi_comm_isp.h
- 库文件: libisp.so

【注意】

无。

【举例】

```
// Set AE / AWB / AF related statistic window setting
VI_PIPE ViPipe = 0;
ISP_STATISTICS_CFG_S stsCfg;

// 设定AE0 windows ROI.
stsCfg.stAECfg.stCrop[0].bEnable = 1;
stsCfg.stAECfg.stCrop[0].u16X = stsCfg.stAECfg.stCrop[0].u16Y = 0;
stsCfg.stAECfg.stCrop[0].u16W = 1920;
stsCfg.stAECfg.stCrop[0].u16H = 1080;

// 设定AWB统计值在x & y方向的window数.
stsCfg.stWBCfg.u16ZoneRow = AWB_ZONE_ORIG_ROW;
stsCfg.stWBCfg.u16ZoneCol = AWB_ZONE_ORIG_COLUMN;

// 设定AWB window ROI.
stsCfg.stWBCfg.stCrop.u16X = stsCfg.stWBCfg.stCrop.u16Y = 0;
stsCfg.stWBCfg.stCrop.u16W = 1920;
stsCfg.stWBCfg.stCrop.u16H = 1080;

// 设定AWB统计亮度的阈值.
stsCfg.stWBCfg.u16BlackLevel = 0;
stsCfg.stWBCfg.u16WhiteLevel = 4095;

// 设定AF统计值开关.
stsCfg.stFocusCfg.stConfig.bEnable = 1;

// 设定AF统计的前处理开关
stsCfg.stFocusCfg.stConfig.stRawCfg.PreGammaEn = 0;
stsCfg.stFocusCfg.stConfig.stPreFltCfg.PreFltEn = 1;

// 设定AF统计值在 x & y方向的window数
stsCfg.stFocusCfg.stConfig.u16Hwnd = 17;
stsCfg.stFocusCfg.stConfig.u16Vwnd = 15;

// 设定AF window ROI.
stsCfg.stFocusCfg.stConfig.stCrop.bEnable = 1;
stsCfg.stFocusCfg.stConfig.stCrop.u16X = AF_XOFFSET_MIN;
stsCfg.stFocusCfg.stConfig.stCrop.u16Y = AF_YOFFSET_MIN;
stsCfg.stFocusCfg.stConfig.stCrop.u16W = 1920 - AF_XOFFSET_MIN * 2;
stsCfg.stFocusCfg.stConfig.stCrop.u16H = 1080 - AF_YOFFSET_MIN * 2;
```

(下页继续)

(续上页)

```

// 设定AF水平方向低通滤波器系数.
stsCfg.stFocusCfg.stConfig.u8HFfltShift = 1;
stsCfg.stFocusCfg.stConfig.s8HFfltLpCoeff[0] = 1;
stsCfg.stFocusCfg.stConfig.s8HFfltLpCoeff[1] = 2;
stsCfg.stFocusCfg.stConfig.s8HFfltLpCoeff[2] = 3;
stsCfg.stFocusCfg.stConfig.s8HFfltLpCoeff[3] = 5;
stsCfg.stFocusCfg.stConfig.s8HFfltLpCoeff[4] = 10;

// 设定AF水平方向高通滤波器系数.
stsCfg.stFocusCfg.stHParam_FIR0.s8HFfltHpCoeff[0] = 0;
stsCfg.stFocusCfg.stHParam_FIR0.s8HFfltHpCoeff[1] = 0;
stsCfg.stFocusCfg.stHParam_FIR0.s8HFfltHpCoeff[2] = 13;
stsCfg.stFocusCfg.stHParam_FIR0.s8HFfltHpCoeff[3] = 24;
stsCfg.stFocusCfg.stHParam_FIR0.s8HFfltHpCoeff[4] = 0;

// 设定AF垂直方向高通滤波器系数.
stsCfg.stFocusCfg.stVParam_FIR.s8VFfltHpCoeff[0] = 13;
stsCfg.stFocusCfg.stVParam_FIR.s8VFfltHpCoeff[1] = 24;
stsCfg.stFocusCfg.stVParam_FIR.s8VFfltHpCoeff[2] = 0;

// 设定各统计值取值开关.
stsCfg.unKey.bit1FEAeGloStat = stsCfg.unKey.bit1FEAeLocStat =
stsCfg.unKey.bit1AwbStat1 = stsCfg.unKey.bit1AwbStat2 = stsCfg.unKey.bit1FEAfStat = 1;

CVI_ISP_SetStatisticsConfig(ViPipe, &stsCfg);

// 设定AF低通滤波器系数规则.
AF低通滤波器拥有九个参数 {x0, x1, x2, x3, x4, x5, x6, x7, x8} ,
最后四个参数与前四个参数对称(x0 = x8, x1 = x7, x2 = x6, x3 = x5) ,
使用者只须设置前五个参数(x0 ~ x5)即可
// 设定AF水平高通滤波器系数规则.
AF水平高通滤波器拥有九个参数 {x0, x1, x2, x3, x4, x5, x6, x7, x8} ,
最后四个参数与前四个参数 ,
数字相同但正负符号相反 (x0 = -x8, x1 = -x7, x2 = -x6, x3 = -x5) ,
使用者只须设置前五个参数(x0 ~ x5)即可
// 设定AF垂直高通滤波器系数规则.
AF垂直高通滤波器拥有五个参数 {x0, x1, x2, x3, x4} ,
最后两个参数与前两个参数 ,
数字相同但正负符号相反 (x0 = -x4, x1 = -x3) ,
使用者只须设置前三个参数(x0 ~ x2)即可

```

filter 参数设定范例

滤波器可通过的频带	滤波器参数 x0	滤波器参数 x1	滤波器参数 x2
0.1 ~ 0.2	20	16	0
0.1 ~ 0.5	17	20	0
0.1 ~ 0.6	13	24	0
0.1 ~ 0.7	12	25	0
0.1 ~ 0.8	-10	-27	0
0.2 ~ 0.9	-10	27	0

【相关主题】

- [CVI_ISP_GetStatisticsConfig](#)

36.2.2 CVI_ISP_GetStatisticsConfig

【描述】

获取 ISP 统计讯息配置。

【语法】

```
CVI_S32 CVI_ISP_GetStatisticsConfig(VI_PIPE ViPipe, ISP_STATISTICS_CFG_S *pstStatCfg);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstStatCfg	ISP 统计信息配置	输出

【返回值】

返回值	描述
0	成功。
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无。

【举例】

无。

【相关主题】

- [CVI_ISP_SetStatisticsConfig](#)

36.2.3 CVI_ISP_GetAESTatistics

【描述】

获取 ISP AE 统计信息

【语法】

```
CVI_S32 CVI_ISP_GetAESTatistics(VI_PIPE ViPipe, ISP_AE_STATISTICS_S *pstAeStat);
```


【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstAeStat	AE 统计信息输出	输出

【返回值】

返回值	描述
0	成功。
非 0	失败，其值为错误码。

【需求】

- 头文件: cvl_isp.h, cvl_comm_isp.h
- 库文件: libisp.so

【注意】

无。

【举例】

```
//设定 AE window weight
#define AE_ZONE_ROW 15
#define AE_ZONE_COLUMN 17

ISP_AE_WIN_STATISTICS_CFG_S aeWinCfg;
CVI_U8 i,j;
VI_PIPE ViPipe = 0;
CVI_U8 u8Weighttable[AE_ZONE_ROW][AE_ZONE_COLUMN]={
{ 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1 },
{ 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 1 },
{ 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 1 },
{ 1, 2, 2, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 2, 1 },
{ 1, 2, 2, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 2, 1 },
{ 1, 2, 2, 4, 4, 4, 8, 8, 8, 8, 8, 8, 4, 4, 4, 2, 1 },
{ 1, 2, 2, 4, 4, 4, 8, 8, 8, 8, 8, 8, 4, 4, 4, 2, 1 },
{ 1, 2, 2, 4, 4, 4, 8, 8, 8, 8, 8, 8, 4, 4, 4, 2, 1 },
{ 1, 2, 2, 4, 4, 4, 8, 8, 8, 8, 8, 8, 4, 4, 4, 2, 1 },
{ 1, 2, 2, 4, 4, 4, 8, 8, 8, 8, 8, 8, 4, 4, 4, 2, 1 },
{ 1, 2, 2, 4, 4, 4, 8, 8, 8, 8, 8, 8, 4, 4, 4, 2, 1 },
{ 1, 2, 2, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 2, 1 },
{ 1, 2, 2, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 2, 1 },
{ 1, 2, 2, 2, 2, 4, 4, 4, 4, 4, 4, 4, 2, 2, 2, 2, 1 },
{ 1, 2, 2, 2, 2, 4, 4, 4, 4, 4, 4, 4, 2, 2, 2, 2, 1 },
{ 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 1, 1, 1, 1, 1 }},};

CVI_ISP_GetAEWinStatistics(ViPipe, &aeWinCfg);
for (i = 0; i < AE_ZONE_ROW; i++)
{
    for (j = 0; j < AE_ZONE_COLUMN; j++)
    {
        aeWinCfg.au8Weight[i][j] = u8Weighttable[i][j];
    }
}
```

(下页继续)

(续上页)

```
}  
CVI_ISP_SetAEWinStatistics(ViPipe, &aeWinCfg);
```

【相关主题】

无。

36.2.4 CVI_ISP_GetWBStatistics

【描述】

获取 ISP AWB 统计信息

【语法】

```
CVI_S32 CVI_ISP_GetWBStatistics(VI_PIPE ViPipe, ISP_WB_STATISTICS_S *pstAwbStat);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstAwbStat	AWB 统计信息	输出

【返回值】

返回值	描述
0	成功。
非 0	失败，其值为错误码。

【需求】

- 头文件: cvi_isp.h, cvi_comm_isp.h
- 库文件: libisp.so

【注意】

无。

【举例】

无。

【相关主题】

无。

36.2.5 CVI_ISP_GetFocusStatistics

【描述】

获取 ISP AF 统计信息

【语法】

```
CVI_S32 CVI_ISP_GetFocusStatistics(VI_PIPE ViPipe, ISP_AF_STATISTICS_S *pstAfStat);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstAfStat	AF 统计信息	输出

【返回值】

返回值	描述
0	成功。
非 0	失败，其值为错误码。

【需求】

- 头文件: cvi_isp.h, cvi_comm_isp.h
- 库文件: libisp.so

【注意】

无。

【举例】

无。

【相关主题】

无。

36.3 数据类型

36.3.1 ISP_STATISTICS_CTRL_U

【说明】

定义 ISP 统计值信息使能

【定义】

```
typedef union _ISP_STATISTICS_CTRL_U {
    CVI_U64 u64Key;
    struct {
        CVI_U64 bit1FEAeGloStat : 1; /* [0] */
        CVI_U64 bit1FEAeLocStat : 1; /* [1] */
        CVI_U64 bit1AwbStat1 : 1; /* [2] */
        CVI_U64 bit1AwbStat2 : 1; /* [3] */
        CVI_U64 bit1FEAfStat : 1; /* [4] */
        CVI_U64 bit14Rsv : 59; /* [5:63] */
    }
} ISP_STATISTICS_CTRL_U;
```

【成员】

成员名称	描述
bit1FEAeGloStat	AE 全局统计值使能
bit1FEAeLocStat	AE 分块统计值使能
bit1AwbStat1	AWB 全局统计值使能
bit1AwbStat2	AWB 分块统计值使能
bit1FEAfStat	AF 分块统计值使能
bit14Rsv	保留位

【注意事项】

无。

【相关数据类型及接口】

无。

36.3.2 ISP_AE_STATISTICS_CFG_S

【说明】

AE 统计讯息配置

【定义】

```
typedef struct _ISP_AE_STATISTICS_CFG_S {
    CVI_BOOL bHisStatisticsEnable; /*RW; Range:[0x0,0x1]*/
    ISP_AE_CROP_S stCrop[AE_MAX_NUM];
    ISP_AE_FACE_CROP_S stFaceCrop[FACE_WIN_NUM];
    CVI_BOOL fast2A_ena; /*RW; Range:[0x0,0x1]*/
    CVI_U8 fast2A_ae_low; /*RW; Range:[0x0,0xFF]*/
    CVI_U8 fast2A_ae_high; /*RW; Range:[0x0,0xFF]*/
    CVI_U16 fast2A_awb_top; /*RW; Range:[0x0,0xFFFF]*/
    CVI_U16 fast2A_awb_bot; /*RW; Range:[0x0,0xFFFF]*/
    CVI_U16 over_exp_thr; /*RW; Range:[0x0,0x3FF]*/
    CVI_U8 au8Weight[AE_WEIGHT_ZONE_ROW][AE_WEIGHT_ZONE_COLUMN]; /*RW; Range:[0x0, 0xF]*/
} ISP_AE_STATISTICS_CFG_S;
```

【成员】

成员名称	描述
enAESwitch	海思兼容参数, 目前不使用
stHistConfig	海思兼容参数, 目前不使用
enFourPlaneMode	海思兼容参数, 目前不使用
enHistMode	海思兼容参数, 目前不使用
enAverMode	海思兼容参数, 目前不使用
enMaxGainMode	海思兼容参数, 目前不使用
stCrop[AE_MAX_NUM];	AE 统计值输入图像裁减设定
au8Weight[AE_ZONE_ROW][AE_ZONE_COL];	AE 统计值输入图像裁减权重设定
bHisStatisticsEnable	AE/Hist sts 使能
u8StatisticsShiftBits	AE 统计值要缩小几个 bit
u16RGain	R 通道是否经过 RGain
u16GGain	G 通道是否经过 GGain
u16BGain	B 通道是否经过 BGain

【相关数据类型及接口】

· CVI_ISP_SetAEWinStatistics

36.3.3 ISP_AE_CROP_S

【说明】

AE 统计值输入图像裁减设定

【定义】

```
typedef struct _ISP_AE_CROP_S {
    CVI_BOOL bEnable;
    CVI_U16 u16X;
    CVI_U16 u16Y;
    CVI_U16 u16W;
    CVI_U16 u16H;
} ISP_AE_CROP_S;
```

【成员】

成员名称	描述
bEnable	Crop 使能, 0 为不使能, 将设置整张图片作为裁减设定, 1 为使能
u16X	CropX 起始位置, 取值需大于 0
u16Y	CropY 起始位置, 取值需大于 0
u16W	Crop 宽度
u16H	Crop 高度

【注意事项】

无

【相关数据类型及接口】

无。

36.3.4 ISP_AE_FACE_CROP_S

【说明】

AE face 统计值输入图像裁减设定

【定义】

```
typedef struct _ISP_AE_FACE_CROP_S {
    CVI_BOOL bEnable; /*RW; Range:[0x0,0x1]*/
    CVI_U16 u16X; /*RW; Range:[0x00,0x1FFF]*/
    CVI_U16 u16Y; /*RW; Range:[0x00,0x1FFF]*/
    CVI_U8 u16W; /*RW; Range:[0x00,0xFF]*/
    CVI_U8 u16H; /*RW; Range:[0x00,0xFF]*/
} ISP_AE_FACE_CROP_S;
```

【成员】

成员名称	描述
bEnable	Crop 使能，0 为不使能，将设置整张图片作为裁减设定，1 为使能
u16X	CropX 起始位置，取值需大于 0
u16Y	CropY 起始位置，取值需大于 0
u16W	Crop 宽度
u16H	Crop 高度

【注意事项】

无

【相关数据类型及接口】

无。

36.3.5 ISP_WB_STATISTICS_CFG_S

【说明】

WB 统计讯息配置

【定义】

```
typedef struct _ISP_WB_STATISTICS_CFG_S {
    ISP_AWB_SWITCH_E enAWBSwitch;
    CVI_U16 u16ZoneRow;
    CVI_U16 u16ZoneCol;
    CVI_U16 u16ZoneBin;
    CVI_U16 au16HistBinThresh[4];
    CVI_U16 u16WhiteLevel;
```

(下页继续)

(续上页)

```

CVI_U16 u16BlackLevel;
CVI_U16 u16CbMax;
CVI_U16 u16CbMin;
CVI_U16 u16CrMax;
CVI_U16 u16CrMin;
ISP_AWB_CROP_S stCrop;
} ISP_WB_STATISTICS_CFG_S;

```

【成员】

成员名称	描述
enAWBSwitch	ISP_AWB_AFTER_DG = 0 ISP_AWB_AFTER_Expander, ISP_AWB_AFTER_DRC, ISP_AWB_SWITCH_BUTT 设定 AWB block location
u16ZoneRow	AWB 统计窗口行数
u16ZoneCol	AWB 统计窗口列数
u16ZoneBin	海思兼容参数, 目前不使用
au16HistBinThresh	海思兼容参数, 目前不使用
u16WhiteLevel	AWB 亮点排除的 Threshold 设置
u16BlackLevel	AWB 暗点排除的 Threshold 设置
u16CbMax	不同 ISO 下 R/G 的最大值
u16CbMin	不同 ISO 下 R/G 的最小值
u16CrMax	不同 ISO 下 B/G 的最大值
u16CrMin	不同 ISO 下 B/G 的最小值

【注意事项】

无。

【相关数据类型及接口】

无。

36.3.6 ISP_AWB_CROP_S

【说明】

AWB 统计值输入图像裁减设定

【定义】

```

typedef struct _ISP_AWB_CROP_S {
    CVI_BOOL bEnable;
    CVI_U16 u16X;
    CVI_U16 u16Y;

```

(下页继续)

(续上页)

```

    CVI_U16 u16W;
    CVI_U16 u16H;
} ISP_AWB_CROP_S;

```

【成员】

成员名称	描述
bEnable	C rop 使能，假如 0 为不使能，将设置整张图片作为裁减设定，假如 1 为使能
u16X	CropX 起始位置，取值需大于 0 取值范围：[0, 图像宽度 - 16 * u16ZoneCol]
u16Y	CropY 起始位置，取值需大于 0 取值范围：[0, 图像高度 - 16 * u16ZoneRow]
u16W	Crop 宽度 取值范围：[16 * u16ZoneCol, 图像宽度]
u16H	Crop 高度 取值范围：[0, 图像高度 - 16 * u16ZoneRow]

【注意事项】

无。

【相关数据类型及接口】

无。

36.3.7 ISP_WB_STATISTICS_S

【说明】

定义 AWB 统计信息

【定义】

```

typedef struct _ISP_WB_STATISTICS_S {
    CVI_U16 u16GlobalR;
    CVI_U16 u16GlobalG;
    CVI_U16 u16GlobalB;
    CVI_U16 u16CountAll;
    CVI_U16 au16ZoneAvgR[AWB_ZONE_NUM];
    CVI_U16 au16ZoneAvgG[AWB_ZONE_NUM];
    CVI_U16 au16ZoneAvgB[AWB_ZONE_NUM];
    CVI_U16 au16ZoneCountAll[AWB_ZONE_NUM];
    ISP_AWB_GRID_INFO_S stGridInfo;
} ISP_WB_STATISTICS_S;

```

宏定义

```
#define AWB_ZONE_ORIG_ROW (32)
```

```
#define AWB_ZONE_ORIG_COLUMN (64)
```

```
#define AWB_ZONE_NUM (AWB_ZONE_ORIG_ROW * AWB_ZONE_ORIG_COLUMN)
```

【成员】

成员名称	描述
u16GlobalR	Bayer 域全局 R 分量的平均值 取值范围：[0, 0x3ff] 数据类型：CVI_BOOL
u16GlobalG	Bayer 域全局 G 分量的平均值 取值范围：[0, 0x3ff] 数据类型：CVI_BOOL
u16GlobalB	Bayer 域全局 B 分量的平均值 取值范围：[0, 0x3ff] 数据类型：CVI_BOOL
u16CountAll	全局统计区域的像素个数 取值范围：[0, 0xffff] 数据类型：CVI_BOOL
au16ZoneAvgR	Bayer 域分区间 R 分量的平均值 取值范围：[0, 0x3ff] 数据类型：CVI_BOOL
au16ZoneAvgG	Bayer 域分区间 G 分量的平均值 取值范围：[0, 0x3ff] 数据类型：CVI_BOOL
au16ZoneAvgB	Bayer 域分区间 B 分量的平均值 取值范围：[0, 0x3ff] 数据类型：CVI_BOOL
au16ZoneCountAll	Bayer 域分区间的像素个数 取值范围：[0, 0xffff] 数据类型：CVI_BOOL
ISP_AWB_GRID_INFO_S	AWB 统计信息坐标信息

【注意事项】

无。

【相关数据类型及接口】

无。

36.3.8 ISP_AWB_GRID_INFO_S

【说明】

AWB 统计信息坐标信息

【定义】

```
typedef struct ISP_AWB_GRID_INFO_S {
    CVI_U16 au16GridYPos[AWB_ZONE_ORIG_ROW + 1];
    CVI_U16 au16GridXPos[AWB_ZONE_ORIG_COLUMN + 1];
    CVI_U8 u8Status;
} ISP_AWB_GRID_INFO_S;
```

宏定义

(下页继续)

(续上页)

```
#define AWB_ZONE_ORIG_ROW (32)
#define AWB_ZONE_ORIG_COLUMN (32)
```

【成员】

成员名称	描述
au16GridYPos	暂未使用
au16GridXPos	暂未使用
u8Status	暂未使用

【注意事项】

无。

【相关数据类型及接口】

无。

36.3.9 ISP_FOCUS_STATISTICS_CFG_S

【说明】

AF 统计讯息配置

【定义】

```
typedef struct _ISP_FOCUS_STATISTICS_CFG_S {
    ISP_AF_CFG_S stConfig;
    ISP_AF_H_PARAM_S stHParam_FIR0;
    ISP_AF_H_PARAM_S stHParam_FIR1;
    ISP_AF_V_PARAM_S stVParam_FIR;
} ISP_FOCUS_STATISTICS_CFG_S;
```

【成员】

成员名称	描述
stConfig	AF 全局配置参数
stHParam_FIR0	水平滤波器第一组 FIR 参数设置
stHParam_FIR1	水平滤波器第二组 FIR 参数设置
stVParam_FIR	垂直滤波器 FIR 参数设置

【注意事项】

无。

【相关数据类型及接口】

无。

36.3.10 ISP_AF_CFG_S

【说明】

定义 AF 统计值参数配置

【定义】

```
typedef struct _ISP_AF_CFG_S {
    CVI_BOOL bEnable;
    CVI_U16 u16Hwnd; /*RW; Range:[0x2, 0x11]*/
    CVI_U16 u16Vwnd; /*RW; Range:[0x2, 0xF]*/
    CVI_U8 u8HFltShift; /*RW; Range:[0x0, 0xF]*/
    CVI_S8 s8HFVltLpCoeff[FIR_H_GAIN_NUM]; /*RW; Range:[0x0, 0x1F]*/
    ISP_AF_RAW_CFG_S stRawCfg;
    ISP_AF_PRE_FILTER_CFG_S stPreFltCfg;
    ISP_AF_CROP_S stCrop;
    CVI_U8 u8HFltCoring; /*RW; Range:[0x0, 0xFF]*/
    CVI_U8 u8HFltCoring; /*RW; Range:[0x0, 0xFF]*/
    CVI_U8 u8VFltCoring; /*RW; Range:[0x0, 0xFF]*/
    CVI_U16 u16HighLumaTh; /*RW; Range:[0x0, 0xFF]*/
    CVI_U8 u8ThLow;
    CVI_U8 u8ThHigh;
    CVI_U8 u8GainLow; /*RW; Range:[0x0, 0xFE]*/
    CVI_U8 u8GainHigh; /*RW; Range:[0x0, 0xFE]*/
    CVI_U8 u8SlopLow; /*RW; Range:[0x0, 0xF]*/
    CVI_U8 u8SlopHigh; /*RW; Range:[0x0, 0xF]*/
} ISP_AF_CFG_S;
```

【成员】

成员名称	描述
bEnable	AF 使能, 0 为不使能, 1 为使能
u16Hwnd	AF 水平方向窗口数, 最大可设置为 17 取值范围: [0x2, 0x11] 数据类型: CVI_U16
u16Vwnd	AF 垂直方向窗口数, 最大可设置为 15 取值范围: [0x2, 0xf] 数据类型: CVI_U16
u8HFltShift	AF 低通滤波器统计值移位寄存器值 取值范围: [0, 16] 数据类型: CVI_U8
s8HFVltLpCoeff[FIR_H_GAIN_NUM]	AF 低通滤波器系数, 用于控制 FIR 滤波器的频率响应 取值范围: [-32, 31] 数据类型: CVI_S8
stRawCfg	AF Bayer 域相关配置
stPreFltCfg	AF 预滤波处理使能设定
stCrop	AF 输入图像的裁剪配置
u16HighLumaTh	AF 高亮点统计值 Threshold 设置 取值范围: [0, 4095]

【注意事项】

无。

【相关数据类型及接口】

无。

36.3.11 ISP_AF_RAW_CFG_S

【说明】

AF Bayer 域影像预处理配置

【定义】

```
typedef struct _ISP_AF_RAW_CFG_S {  
    CVI_U8 PreGammaEn;  
    CVI_U8 PreGammaTable[AF_GAMMA_NUM];  
} ISP_AF_RAW_CFG_S;
```

【成员】

成员名称	描述
PreGammaEn	AF 模块 Gamma 使能, 0 为不使能, 1 为使能
PreGammaTable[AF_GAMMA_NUM]	AF 模块 Gamma table 设置

【注意事项】

无。

【相关数据类型及接口】

无。

36.3.12 ISP_AF_PRE_FILTER_CFG_S

【说明】

AF Bayer 域影像滤波预处理配置

【定义】

```
typedef struct _ISP_AF_PRE_FILTER_CFG_S {  
    CVI_BOOL PreFltEn;  
} ISP_AF_PRE_FILTER_CFG_S;
```

【成员】

成员名称	描述
PreFltEn	AF 模块滤波预处理使能, 0 为不使能, 1 为使能

【注意事项】

无。

【相关数据类型及接口】

无。

36.3.13 ISP_AF_CROP_S

【说明】

AF 统计值输入图像裁减设定

【定义】

```
typedef struct _ISP_AF_CROP_S {  
    CVI_BOOL bEnable;  
    CVI_U16 u16X;  
    CVI_U16 u16Y;  
    CVI_U16 u16W;  
    CVI_U16 u16H;  
} ISP_AF_CROP_S;
```

【成员】

成员名称	描述
bEnable	Crop 使能, 0 为不使能, 将设置整张图片作为裁减设定, 1 为使能
u16X	CropX 起始位置 取值范围: [8, 图像宽度 - 8]
u16Y	CropY 起始位置 取值范围: [2, 图像高度 - 2]
u16W	Crop 宽度 取值范围: [16 * u16ZoneCol, 图像宽度 - 16]
u16H	Crop 高度 取值范围: [16 * u16ZoneRow, 图像高度 - 4]

【注意事项】

无。

【相关数据类型及接口】

无。

36.3.14 ISP_AF_H_PARAM_S

【说明】

定义 AF 水平滤波器参数设置

【定义】

```
typedef struct ISP_AF_H_PARAM_S {  
    CVI_S8 s8HFltHpCoeff[FIR_H_GAIN_NUM];  
} ISP_AF_H_PARAM_S;
```

【成员】

成员名称	描述
s8HFltHpCoeff[FIR_H_GAIN_NUM]	水平滤波器系数，用于控制 FIR 滤波器的频率响应 取值范围: [-32, 31]

【注意事项】

无。

【相关数据类型及接口】

无。

36.3.15 ISP_AF_V_PARAM_S

【说明】

定义 AF 垂直滤波器参数设置

【定义】

```
typedef struct ISP_AF_V_PARAM_S {  
    CVI_S8 s8VFltHpCoeff[FIR_V_GAIN_NUM];  
} ISP_AF_V_PARAM_S;
```

【成员】

成员名称	描述
s8VFltHpCoeff[FIR_V_GAIN_NUM]	垂直滤波器系数，用于控制 FIR 滤波器的频率响应 取值范围: [-32, 31]

【注意事项】

无。

【相关数据类型及接口】

无。

36.3.16 ISP_STATISTICS_CFG_S

【说明】

ISP 统计讯息配置

【定义】

```
typedef struct _ISP_STATISTICS_CFG_S {  
    ISP_STATISTICS_CTRL_U unKey;  
    ISP_AE_STATISTICS_CFG_S stAECfg;  
    ISP_WB_STATISTICS_CFG_S stWBCfg;  
    ISP_FOCUS_STATISTICS_CFG_S stFocusCfg;  
} ISP_STATISTICS_CFG_S;
```

【成员】

成员名称	描述
unKey	统计信息使能
stAECfg	AE 统计信息配置
stWBCfg	WB 统计信息配置
stFocusCfg	AF 统计信息配置

【注意事项】

无。

【相关数据类型及接口】

无。

36.3.17 ISP_FOCUS_ZONE_S

【说明】

定义 AF 计算出的统计结果

【定义】

```
typedef struct _ISP_FOCUS_ZONE_S {  
    CVI_U16 u16HlCnt;  
    CVI_U64 u64h0;  
    CVI_U64 u64h1;  
    CVI_U32 u32v0;  
} ISP_FOCUS_ZONE_S;
```

【成员】

成员名称	描述
u16HlCnt	AF 分区间内统计超过高亮点阈值的点数值，将设置整张图片作为裁减设定，1 为使能
u64h0	AF 分区间内统计水平方向第一组 FIR 滤波器的结果

下页继续

表 36.22 – 续上页

成员名称	描述
u64h1	AF 分区间内统计水平方向第二组 FIR 滤波器的结果
u32v0	AF 分区间内统计垂直方向 FIR 滤波器的结果

【注意事项】

无。

【相关数据类型及接口】

无。

36.3.18 ISP_FE_FOCUS_STATISTICS_S

【说明】

定义 AF FE 提供的统计讯息

【定义】

```
typedef struct _ISP_FE_FOCUS_STATISTICS_S {  
    ISP_FOCUS_ZONE_S stZoneMetrics[AF_ZONE_ROW][AF_ZONE_COLUMN];  
} ISP_FE_FOCUS_STATISTICS_S;
```

【成员】

成员名称	描述
stZoneMetrics	ISP AF 的分块统计信息

【注意事项】

无。

【相关数据类型及接口】

无。

36.3.19 ISP_AF_STATISTICS_S

【说明】

定义 AF 提供的所有统计信息

【定义】

```
typedef struct _ISP_AF_STATISTICS_S {  
    ISP_FE_FOCUS_STATISTICS_S stFEAFStat;  
} ISP_AF_STATISTICS_S;
```

【成员】

成员名称	描述
stFEAFStat	AF 在 Bayer 域的统计信息

【注意事项】

无。

【相关数据类型及接口】

无。

37 查询内部状态消息

37.1 概述

本章节说明 Inner State information 的相关接口。此接口的作用为提供用户查询系统内部状态以及数个与 ISO 相关的参数目前所设置的真实数值。

使用者可在调试过程中透过此接口获取 ISO 相关参数目前填入的真实数值以确认参数是否正确配置。透过此接口只能获得数值，无法改变相关参数。

37.2 API 参考

- `CVI_ISP_QueryInnerStateInfo`: 获取系统内部信息以及 ISO 相关参数目前设定的真实数值。

37.2.1 CVI_ISP_QueryInnerStateInfo

【描述】

获取系统内部信息以及 ISO 相关参数目前设定的真实数值

【语法】

```
CVI_S32 CVI_ISP_QueryInnerStateInfo(VI_PIPE ViPipe, ISP_INNER_STATE_INFO_S_
→*pstInnerStateInfo);
```

【参数】

参数名称	描述	输入/输出
ViPipe	VI_PIPE 号	输入
pstInnerStateInfo	内部信息与目前实际参数设定值	输出

【返回值】

返回值	描述
0	成功。
非 0	失败，其值为错误码。

【需求】

- 头文件: `cvi_isp.h`, `cvi_comm_isp.h`
- 库文件: `libisp.so`

【注意】

无。

【举例】

无。

【相关主题】

无。

37.3 数据类型

- `ISP_INNER_STATE_INFO_S` : 定义内部信息与目前实际参数设定值

37.3.1 `ISP_INNER_STATE_INFO_S`

【说明】

定义内部信息与目前实际参数设定值

【定义】

```
#define MAX_HIST_BINS 256
#define MAX_EXPOSURE_RATIO 256
#define LTM_DARK_CURVE_NODE_NUM 257
#define LTM_BRIGHT_CURVE_NODE_NUM 513
typedef struct _ISP_INNER_STATE_INFO_S {
    CVI_U32 blcOffsetR;
    CVI_U32 blcOffsetGr;
    CVI_U32 blcOffsetGb;
    CVI_U32 blcOffsetB;
    CVI_U32 blcGainR;
    CVI_U32 blcGainGr;
    CVI_U32 blcGainGb;
    CVI_U32 blcGainB;
    CVI_S32 ccm[9];
    CVI_U16 drcGlobalToneBinNum;
    CVI_U16 drcGlobalToneBinSEStep;
    CVI_U32 drcGlobalTone[LTM_GLOBAL_CURVE_NODE_NUM];
    CVI_U32 drcDarkTone[LTM_DARK_CURVE_NODE_NUM];
}
```

(下页继续)

(续上页)

```

CVI_U32 drcBrightTone[LTM_BRIGHT_CURVE_NODE_NUM];
CVI_BOOL bWDRSwitchFinish;
// For 2TO1/3TO1/4TO1 use.
CVI_U32 u32WDRExpRatioActual[ISP_WDR_FRAME_IDX_SIZE];
ISP_MESH_SHADING_GAIN_LUT_S mlscGainTable;
} ISP_INNER_STATE_INFO_S;

```

【成员】

成员名称	描述
wdrHistBinNum	目前画面 wdr 分布直方图的数组数量
blcOffsetR	目前黑电平在 R 分量扣除的值
blcOffsetGr	目前黑电平在 Gr 分量扣除的值
blcOffsetGb	目前黑电平在 Gb 分量扣除的值
blcOffsetB	目前黑电平在 B 分量扣除的值
blcGainR	目前 ISP 在 R 分量的数字增益
blcGainGr	目前 ISP 在 Gr 分量的数字增益
blcGainGb	目前 ISP 在 Gb 分量的数字增益
blcGainB	目前 ISP 在 B 分量的数字增益
Ccm[9]	目前 ISP 使用的色彩还原矩阵真实数值
wdrHistogramBefore[MAX_HIST_BINS * MAX_EXPOSURE_RATIO]	目前 wdr 的分布直方图
wdrHistogramAfter[MAX_HIST_BINS * MAX_EXPOSURE_RATIO]	目前 wdr 分布直方图所产生的色调映像曲线
drcDarkTone[LTM_DARK_CURVE_NODE_NUM]	目前 ISP 使用的暗区色调映像曲线
drcBrightTone[LTM_BRIGHT_CURVE_NODE_NUM]	目前 ISP 使用的亮区色调映像曲线
bWDRSwitchFinish	标示 sensor mode 切换是否完成

【注意事项】

无。

【相关数据类型及接口】

无。

38 Debug

39 错误码

39.1 Proc 调试信息说明

39.2 概述

调试信息采用 procfs，可以根据上层设定的 proc param 和 proc level，实时反映 ISP 模组内的设定参数，3A 统计值，3A 运算结果等信息，供开发人员定位问题和分析问题使用。

39.3 使用方法

【文件目录】

/proc/soph/isp

【开启方法】

在呼叫 CVI_ISP_MemInit 后，呼叫 CVI_ISP_SetCtrlParam 设定 ISP 控制参数 u32ProcParam=n，其中 n 不能为 0，n 表示收集 ISP 信息的频率，每隔 n 帧收集一次 ISP 信息，默认值为 30

呼叫 CVI_ISP_SetCtrlParam 设定 ISP 控制参数 u32ProcLevel=m，其中 m 的范围为【0, 3】m=0 时表示 proc 功能关闭，m=1 时表示打印 level1 级别的资讯，m=2 时表示打印 level2 级别的资讯，m=3 时表示打印 level3 级别的资讯，level 级别越高，表示打印的资讯越详细，而且高级别的打印内容一定包含低级别的打印内容，默认值为 0

在使用 ISP_TOOL_DAEMON 时，若要开启 proc 功能，可以在跑 ISP_TOOL_DAEMON 前 export PROC_LEVEL=n(n 为 1-3,n=0 不开启)

【查看信息方法】

ISP module 正常运行后

在控制台通过调用 cat /proc/soph/isp，相关 log 随后会打印于控制台，cp /proc/soph/isp 到 pc 端查看。

39.4 ISP

39.4.1 LEVEL1 级别调试信息分析

ISP 的各模块参数打印

【参数说明】

参数		描述
MOD- ULE/CTRLPARAM	ProcParam	表示收集 ISP 信息的收集频率
	ProcLevel	表示 ISP proc 的 level 级别
	AESatIntvl	表示 ISP AE 统计信息更新频率
	AWBStatIntvl	表示 ISP AWB 统计信息更新频率
	AFStatIntvl	表示 ISP AF 统计信息更新频率
	UpdatePos	表示中断更新位置是用帧起始中断还是在帧结束中断，0 是帧起始
	IntTimeOut	表示获取 ISP 中断超时的最大时间
	PwmNumber	表示 pwm 使用情况
	PortIntDelay	表示 Port 中断延时时间
FSWDR	Enable	WDR 模块使能
	MotionCom- pEnable	WDR 运动侦测使能开关
	CombineS- NRAwareEn	具 SNR 感知的宽动态融合模式使能，能根据短帧噪声自适应取决长短帧融合比例
	CombineS- NRAwareLowThr	短帧噪声 SNR 自适应低临界值。当短帧估测噪声低于低临界值则进行一般长短帧融合，噪声介于低临界值与高临界值间则按比例进行 SNR 自适应融合。
	CombineS- NRAwareHigh- Thr	当短帧估测噪声高于高临界值则根据 SNRAwareToleranceLevel 强度进行 SNR 自适应融合，噪声介于低临界值与高临界值间则按比例进行 SNR 自适应融合。
	CombineS- NRAwareSm- Level	短帧噪声 SNR 自适应之时间域变化平滑程度
	ExposureRatio	WDR 长短曝的曝光比
	ShortMaxVal	WDR 宽动态的最大值
	isManualMode	手动模式或自动模式
	Combine- LongThr	长曝光临界值，低于该临界值的图像数据将只选择长曝光数据合成 WDR 影像
	CombineShort- Thr	短曝光临界值，超过该临界值的图像数据将只选择短曝光数据合成 WDR 影像
	CombineM- inWeight	长短曝图像数据融合最低权重值。权重值越大，融合时长曝光占的比重越多，反之则短曝光占的比重越多
	Combine- MaxWeight	长短曝图像数据融合最高权重值。权重值越大，融合时长曝光占的比重越多，反之则短曝光占的比重越多

下页继续

表 39.1 – 续上页

参数		描述
	MergeMode	WDR 运动侦测模式 0: 运动侦测资讯取长短帧运动资讯最大值 1: 运动侦测资讯为长短帧运动资讯之等比例融合, 比例由 MergeModeAlpha 定义
	MergeModeAlpha	MergeMode 模式为 1 的移动资讯融合比例
	CombineS-NRAware-TolLevel	短帧噪声 SNR 自适应之噪声容忍强度
Shading	MeshEnable	MeshLSC 功能使能
	MeshisManualMode	手动模式或自动模式
	MeshStr	LSC 补偿强度
	MeshLscGain-Lut Size	色温自适应 LSC 补偿增益表数量
	MeshLscGain-LutColorTemp	色温自适应 LSC 补偿增益表所对应之色温
	RadialEnable	RadialLSC 功能使能
	RadialCenterX	图像传感器镜心 X 方向坐标
	RadialCenterY	图像传感器镜心 Y 方向坐标
	RadialStr	LSC 补偿强度
	Low-RadialGainLut	LSC R adius 形式补偿增益表
	High-RadialGainLut	LSC R adius 形式补偿增益表
DCI	Enable	DCI 模块使能
	Speed	Smooth 强度, 值越高, 则变化越慢
	DciStrength	用来控制 DCI 的强度, 值越大, 对比度越大。
	isManualMode	手动模式或自动模式
	ContrastGain	保留每个 BIN 最低斜率, 值越大斜率越小
	BlcThr	用来决定暗区范围的阈值。值越大, 包含的暗区范围越大。
	WhtThr	用来决定亮区范围的阈值。值越小, 包含的亮区范围越大。
	BlcCtrl	用来决定暗区的对比度。数值为 256 时, 暗区对比度不变。比 256 大时, 值越大, 暗区对比度越大; 反之, 比 256 小时, 值越小, 暗区对比度越小。
	WhtCtrl	用来决定亮区的对比度。数值为 256 时, 亮区对比度不变。比 256 大时, 值越大, 亮区对比度越大; 反之, 比 256 小时, 值越小, 亮区对比度越小。
Dehaze	Enable	Dehaze 功能使能
	isManualMode	手动模式或自动模式
	Strength	用来控制 Dehaze 的强度。值越大, 去雾强度越强
BlackLevel	Enable	BLC 模块使能
	isManualMode	手动模式或自动模式
	PreOffsetR	BLC R 像素暗电流值
	PreOffsetGr	BLC GR 像素暗电流值

下页继续

表 39.1 – 续上页

参数		描述
	PreOffsetGb	BLC Gb 像素暗电流值
	PreOffsetB	BLC B 像素暗电流值
	PreGainR	BLC R 像素暗电流补偿增益
	PreGainGr	BLC Gr 像素暗电流补偿增益
	PreGainGb	BLC Gb 像素暗电流补偿增益
	PreGainB	BLC B 像素暗电流补偿增益
	PostOffsetR	BLC R 像素暗电流值
	PostOffsetGr	BLC Gr 像素暗电流值
	PostOffsetGb	BLC Gb 像素暗电流值
	PostOffsetB	BLC B 像素暗电流值
	PostGainR	BLC R 像素暗电流补偿增益
	PostGainGr	BLC Gr 像素暗电流补偿增益
	PostGainGb	BLC Gb 像素暗电流补偿增益
	PostGainB	BLC B 像素暗电流补偿增益
DPC	Enable	DPC 模块使能
	isManualMode	手动模式或自动模式
	ClusterSize	群聚坏点面积上限，值越高越能修正群聚坏点，但可能会造成高频区域解像力的衰减
	Bright-DefToNor-PixRatio	可视亮坏点值与周围像素的倍率
	DarkDefToNor-PixRatio	可视暗坏点值与周围像素的倍率
	FlatThreR	R 通道判别平坦区临界值，值越小越能保留边缘信息
	FlatThreG	B 通道判别平坦区临界值，值越小越能保留边缘信息
	FlatThreB	B 通道判别平坦区临界值，值越小越能保留边缘信息
	FlatThreMinG	G 通道判别平坦区最小临界值
	FlatThreMinRB	RB 通道判别平坦区最小临界值
TNR	Enable	TNR 模块使能
	DeflickerMode	抗闪烁模式
	DeflickerToleranceLevel	抗闪烁模式 mode4 抗闪容忍值，数值越大越能抗闪烁，但亮区的移动残影也会增加。
	LowMtPrtEn	空域降噪微小运动保护使能
	isManualMode	手动模式或自动模式
	TnrStrength0	长曝光 TNR 强度增益
	MapThdLow0	长曝光 TNR 强度上限
	MapThdHigh0	长曝光 TNR 强度下限
	Brightness-NoiseLevelLE	长曝光亮度噪声容忍值
	Brightness-NoiseLevelSE	短曝光亮度噪声容忍值
	RNoiseLevel0	长曝光红色通道噪声容忍值
	RNoiseHiLevel0	长曝光红色通道亮部噪声容忍值
	GNoiseLevel0	长曝光绿色通道噪声容忍值
	GNoiseHiLevel0	长曝光绿色通道亮部噪声容忍值
	BNoiseLevel0	长曝光蓝色通道噪声容忍值

下页继续

表 39.1 – 续上页

参数		描述
	BNoiseHiLevel0	长曝光蓝色通道亮部噪声容忍值
	RNoiseLevel1	短曝光红色通道噪声容忍值
	RNoiseHiLevel1	短曝光红色通道亮部噪声容忍值
	GNoiseLevel1	短曝光绿色通道噪声容忍值
	GNoiseHiLevel1	短曝光绿色通道亮部噪声容忍值
	BNoiseLevel1	短曝光蓝色通道噪声容忍值
	BNoiseHiLevel1	短曝光蓝色通道亮部噪声容忍值
	LowMtPrtLevel	保护限值
	L2mIn0	长曝光 TNR 亮度对强度增益特性表。由四组数值组成的数组。定义灰度等级，值越大灰度越高。
	L2mOut0	长曝光 TNR 亮度对强度增益特性表。由四组数值组成的数组。定义强度增益，值越大强度越强。
	L2mIn1	短曝光 TNR 亮度对强度增益特性表。由四组数值组成的数组。定义灰度等级，值越大灰度越高。
	L2mOut1	短曝光 TNR 亮度对强度增益特性表。由四组数值组成的数组。定义强度增益，值越大强度越强。
	PrtctIn0	长曝光 TNR 动量对拖尾消除程度特性表，由四组数值组成的数组。定义运动等级对运动物体拖尾的消除程度，值越大运动幅度越强。
	PrtctOut0	长曝光 TNR 动量对拖尾消除程度特性表，由四组数值组成的数组。定义运动等级对运动物体拖尾的消除程度，值越小拖尾消除程度越强。
	PrtctIn1	短曝光 TNR 动量对拖尾消除程度特性表，由四组数值组成的数组。定义运动等级对运动物体拖尾的消除程度，值越大运动幅度越强。
	PrtctOut1	短曝光 TNR 动量对拖尾消除程度特性表，由四组数值组成的数组。定义运动等级对运动物体拖尾的消除程度，值越小拖尾消除程度越强。
	LowMtPrtIn	定义运动等级
	LowMtPrtOut	值越大，拖影越不明显，噪声越明显
CAC	Enable	CAC 模块使能
	VarThr	边缘侦测的阈值。值越小，越多区域被判断为边缘。
	PurpleDetRange	紫边侦测的阈值。值越大，越多区域被判断为紫边。
	PurpleCb	紫色在 Cb domain 的坐标。
	PurpleCr	紫色在 Cr domain 的坐标。
	GreenCb	绿色在 Cb domain 的坐标。
	GreenCr	绿色在 Cr domain 的坐标。
	isManualMode	手动模式或自动模式
	DePurpleStr	紫边侦测的阈值。值越大，越多区域被判断为紫边。
CNR	Enable	CNR 模块使能。
	isManualMode	手动模式或自动模式
	CnrStr	色噪去噪强度。值越大，色噪去噪强度越大。
	NoiseSuppressStr	色噪抑制强度。值越大，色噪去除强度越大。
	NoiseSuppressGain	色噪抑制强度增益。值越小，色噪去噪强度越大。

下页继续

表 39.1 – 续上页

参数		描述
	FilterType	色噪去噪滤波器强度。值越大，色噪去除强度越大。
	MotionNrStr	调节物体运动区域的色噪去噪强度。值越大，运动区域的色噪越少。
	DetailSmooth-Mode	去噪细节平滑功能使能
Sharpen	Enable	Y Sharpen 模块使能
	EdgeGain	边缘的增强参数值越大时，边缘锐化强度越大
	TextureGain	细节纹理的增强参数值越大时，细节纹理锐化强度越大
	EdgeThr	边缘和噪声的分界阈值，大于该值被视为是边缘，小于该值则被视为噪声该值越大，越少的边缘被增加；该值越小，越多的边缘被增强。曝光增益越大时，该值建议设置越大
	TextureThr	细节纹理和噪声的分界阈值，大于该值被视为是细节纹理，小于该值则被视为噪声
	LumaAdpCoringEn	自动亮度噪声抑止阈值开关
	LumaAdpGainEn	亮度锐化权重使能
	DeltaAdpGainEn	锐化权重使能
	WdrCoringCompensationEn	亮度锐化噪声值在 wdr 模式的补偿使能。
	WdrCoringCompensation-Mode	亮度锐化噪声值在 wdr 模式的补偿模式。0: 根据 WdrCoringHighThrd 与 WdrCoringLowthd 补偿亮度锐化噪声。1: 根据 DRC tone mapping curve 自动补偿亮度锐化噪声。
	WdrCoringToleranceLevel	亮度锐化噪声值在 wdr 模式补偿的容忍值。数值越小锐化的效果越明显但也越容易将噪声锐化。数值越大则不易将噪声锐化，但锐化的效果则越不明显。
	WdrCoringHighThr	亮度锐化噪声 wdr 模式补偿的高临界值。亮度低于此临界值则不进行噪声补偿。亮度高于此临界值则根据 WdrCoringToleranceLevel 数值进行噪声补偿。亮度介于 WdrCoringLowThrd 与 WdrCoringHighThrd 之间则按比例进行补偿。
	WdrCoringLowThr	亮度锐化噪声 wdr 模式补偿的低临界值。亮度低于此临界值则不进行噪声补偿。亮度高于此临界值则根据 WdrCoringToleranceLevel 数值进行噪声补偿。亮度介于 WdrCoringLowThrd 与 WdrCoringHighThrd 之间则按比例进行补偿。
	isManualMode	手动模式或自动模式
	EdgeFreq	图像具方向性的边缘频段控制
	TextureFreq	图像无方向性的细节纹理频段控制
	GlobalGain	全局锐化权重
	OverShootThr	白边锐化上限幅度
	UnderShootThr	黑边锐化下限幅度

下页继续

表 39.1 – 续上页

参数		描述
	YNoiseLevel	亮度锐化噪声值放大倍率，一倍为 64。数值越大亮度锐化噪声越被放大，反之则缩小。
	DeltaAdpGain	锐化权重
	LumaAdpCor-ing	亮度锐化噪声值，细节纹理或边缘的增强会排除该容忍值所贡献的增强。数值越小锐化的效果越明显但也越容易将噪声锐化。数值越大则不易将噪声锐化，但锐化的效果则越不明显。
	LumaAdpGain	亮度锐化权重
Saturation	isManualMode	手动模式或自动模式
	Saturation	饱和度
Gamma	Enable	Gamma 功能使能
	enCurveType	Gamma 曲线类型
	isManualMode	手动模式或自动模式
HSV	Enable	Saturation Tuning 使能
	isManualMode	手动模式或自动模式
	SatCoringLin-earTh	饱和度的 coring 值
	SatCoringLin-earLmt	最大输出饱和度
CCM	Enable	CCM 模块使能
	isManualMode	手动模式或自动模式
	ISOActEnable	低照度下 CCM Bypass 功能使能
	TempActEnable	高低色温下 CCMBypass 功能使能。
	CCMTabNum	当前配置的 CCM 矩阵个数
	CCM	颜色校正矩阵
	CCMTab	不同色温下的 CCM 矩阵系数
YNR	Enable	YNR 模块使能
	CoringPara-mEnable	控制是否使用手动 coring
	isManualMode	手动模式或自动模式
	WindowType	去噪滤波局域程度。其值越小，作用越局域
	DetailSmooth-Mode	去噪细节平滑功能使能
	NoiseSup-pressStr	噪声抑制强度。值越大，亮噪去除强度越大
	FilterType	去噪滤波器强度。值越大，亮噪去除强度越大
	MotionThr	物体移动量阈值。值越小，侦测为运动的区域范围越大。
	MotionNrPos-Gain	调节在大于移动量阈值的区域之去噪强度。值越大，噪声保留越少。
	MotionNrNeg-Gain	调节在小于移动量阈值的区域之去噪强度。值越小，噪声保留越少。
	VarThr	侦测边缘的阈值。值越大，判断为边缘的数量越少。
	CoringWgtLF	调节在低频区域的随机噪声强度。值越大，在低频区域保留的噪声越多。
	CoringWgtHF	调节在高频区域的随机噪声强度。值越大，在高频区域的保留的噪声越多。

下页继续

表 39.1 – 续上页

参数		描述
	NonDirFiltStr	调节在低频区的去噪强度。值越大，在低频区域去除的噪声越多。
	VhDirFiltStr	调节在水平和垂直区的去噪强度。值越大，在水平和垂直边缘去除的噪声越多。
	AaDirFiltStr	调节在对角线边缘的去噪强度。值越大，在对角线边缘去除的噪声越多。
	NoiseCoring-BaseLuma	运动区亮度值。
	NoiseCoring-BaseOffset	运动区亮度噪声容忍值，运动区的判断与 TNr 运动区侦测连动。值越大，对运动区去噪强度越大。
	NoiseCoringAdvLuma	静止区亮度值
	NoiseCoringAdvOffset	静止区亮度噪声容忍值，静止区的判断与 TNr 运动区侦测连动。值越大，对静止区去噪强度越大。
WDRExposure-Attr	ExpRatioType	仅在多帧合成 WDR 模式下有效。OP_TYPE_AUTO: 根据场景自动计算长短帧曝光比; OP_TYPE_MANUAL: 手动配置长短帧曝光比。
	ExpRatio	仅在多帧合成 WDR 模式下有效。当 enExpRatioType 为 OP_TYPE_AUTO 时, au32ExpRatio 无效。当 enExpRatioType 为 OP_TYPE_MANUAL 时, au32ExpRatio 为可擦写, 表示多帧合成 WDR 相邻 2 帧曝光比期望值。取值范围: 【0x40,0xFFF】
	ExpRatioMax	仅在多帧合成 WDR 模式下有效。当 enExpRatioType 为 OP_TYPE_AUTO 时, u32ExpRatioMax 表示最长帧与最短帧曝光时间比值的最大值。当 enExpRatioType 为 OP_TYPE_MANUAL 时, u32ExpRatioMax 无效。6bit 小数精度, 0x40 表示曝光比为 1 倍。
	ExpRatioMin	仅在多帧合成 WDR 模式下有效。当 enExpRatioType 为 OP_TYPE_AUTO 时, u32ExpRatioMin 表示长帧曝光时间与短帧曝光时间比值的最小值。当 enExpRatioType 为 OP_TYPE_MANUAL 时, u32ExpRatioMin 无效。格式为无符号 6.6bit 定点, 0x40 表示长帧曝光时间与短帧曝光时间的比值为 1 倍。默认值为 0x40。取值范围: 【0x40, u32ExpRatioMax】
	Tolerance	曝光比容忍值, 仅在两帧合成 WDR 模式下有效。当 enExpRatioType 为 OP_TYPE_AUTO 时, 该值越大, 表示场景动态范围变化在一定范围内时, 曝光比保持不变。默认值为 0xC。取值范围: 【0x0, 0xFF】
	Speed	自动曝光比调节速度, 仅在两帧合成 WDR 模式下有效。当 enExpRatioType 为 OP_TYPE_AUTO 时, 该值越大, 自动曝光比调节速度越快。默认值为 0x20。取值范围: 【0x0, 0xFF】

下页继续

表 39.1 – 续上页

参数		描述
	RatioBias	曝光比偏差值，仅在多帧合成 WDR 模式下有效。当 enExpRatioType 为 OP_TYPE_AUTO 时，该值越大，自动曝光比越大。默认值为 0x400，表示不对自动曝光比算法的计算结果进行调整。经过该值调整的曝光比会受到曝光比最大/最小值的限制。取值范围：【0x0, 0xFFFF】
	SECompensation	调整短帧画面的目标亮度值取值范围：【0x0, 0xFF】
	SEHisThr	计算短帧 frame 超过长帧 frame 的曝光比阈值 (1x =64)
	SEHis255CntTargetDownRatio	短帧 histogram bin 255 数量大于此阈值，则调降短帧的目标亮度
	SEHis255CntTargetUpRatio	短帧 histogram bin 255 数量小于此阈值，则将调降的短帧的目标亮度恢复原来的目标亮度
	LEHisCntTargetDownRatio	长帧 histogram bin 255 数量大于此阈值，则调降的长帧的目标亮度
	LEHisCntTargetUpRatio	长帧 histogram bin 255 数量小于此阈值，则将调降的长帧的目标亮度恢复原来的目标亮度
	LEAdjustTargetMin	长帧各 LV 的 target min 40,40, 40, 40, 40, 40, 45, 50, 60, 60, 60, 60, 60, 60, 60
	LEAdjustTargetMax	长帧各 LV 的 target max 50,50, 50, 50, 50, 50, 60, 70, 85, 100, 110, 110, 120, 120, 120, 120
	SEAdjustTargetMin	短帧各 LV 的 target min 20,20, 20, 20, 20, 20, 20, 20, 20, 20, 20, 20, 20, 20
	SEAdjustTargetMax	短帧各 LV 的 target max 60, 60, 60, 60, 60, 60, 60, 60, 60, 60, 60, 60, 60, 60
	DiffPixelNum	长帧 histogram bin 255 的 pixel 数阈值，超过此值，则降低长曝的目标亮度
	LELowBinThr	长帧亮度大于此阈值的 window 才能加入 AE 长帧的测光
	LEHighBinThr	长帧亮度小于此阈值的 window 才能加入 AE 长帧的测光
	SELowBinThr	短帧亮度大于此阈值的 window 才能加入 AE 短帧的测光
	SEHighBinTh	短帧亮度小于此阈值的 window 才能加入 AE 短帧的测光
ExposureAttr	ByPass	AE 模块 bypass 功能使能，默认为 CVI_FALSE FSWDR 模式下，表示当前最短帧 (VS) 曝光时间。
	OpType	自动曝光或手动曝光开关，默认为 OP_TYPE_AUTO。
	AERunInterval	AE 算法运行的间隔取值范围为【1,255】。取值为 1 时表示每帧都运行 AE 算法；取值为 2 时表示每 2 帧运行 1 次 AE 算法，依此类推。建议该值设置不要大于 2，否则 AE 调节速度会受到影响。WDR 模式时，该值建议设置为 1，这样 AE 收敛会更加平滑。该值默认为 1。线性模式不用关注该值。
	ExpTimeOp-Type	手动曝光时间使能，默认值为 OP_TYPE_AUTO

下页继续

表 39.1 – 续上页

参数		描述
	AGainOpType	手动 sensor 模拟增益使能, 默认值为 OP_TYPE_AUTO
	DGainOpType	手动 sensor 数字增益使能, 默认值为 OP_TYPE_AUTO
	ISPDGainOp-Type	手动 ISP 数字增益, 10bit 小数精度, 默认值为 0x400。
	ExpTime	手动曝光时间, 以微秒 (us) 为单位, 默认值为 0x4000
	AGain	手动 sensor 模拟增益, 10bit 小数精度, 默认值为 0x400。
	DGain	手动 sensor 数字增益, 10bit 小数精度, 默认值为 0x400。
	ISPDGain	手动 ISP 数字增益, 10bit 小数精度, 默认值为 0x400。
	ISONumOp-Type	ISO num 手动/自动模式切换
	UseISONum	手动模式时, 使用 ISO num 的型式来控制增益
	ISONum	手动模式时, 设定的 ISO num
	ExpTimeRange-Max	曝光时间范围, 设置最大值, 以微秒 (us) 为单位。
	ExpTimeRangeMin	曝光时间范围, 设置最小值, 以微秒 (us) 为单位。
	AGainRange-Max	Sensor 模拟增益范围, 设置最大值, 10bit 小数精度。
	AGainRangeMin	Sensor 模拟增益范围, 设置最小值, 10bit 小数精度。
	DGainRange-Max	Sensor 数字增益范围, 设置最大值, 10bit 小数精度。
	DGainRangeMin	Sensor 数字增益范围, 设置最小值, 10bit 小数精度。
	ISPDGainRangeMax	ISP 数字增益范围, 设置最大值, 10bit 小数精度。
	ISPDGainRangeMin	ISP 数字增益范围, 设置最小值, 10bit 小数精度。
	SysGainRangeMin	系统增益范围, 设置最小值, 10bit 小数精度
	SysGainRangeMax	系统增益范围, 设置最大值, 10bit 小数精度
	Speed	自动曝光调整时的速度。
	BlackSpeedBias	画面由暗到亮 AE 调节速度的偏差值, 该值越大, 画面从暗到亮的速度越快。
	Tolerance	自动曝光调整时对画面亮度的容忍偏差。
	Compensation	自动曝光调整时的目标亮度。
	EVBias	自动曝光调整时的曝光量偏差值, 10bit 小数精度
	AEStrategy-Mode	自动曝光策略, 高光优先或低光优先。
	AntiflickerEnable	抗闪属性设置。默认抗闪不使能。
	AntiflickerFrequency	抗闪频率值

下页继续

表 39.1 – 续上页

参数		描述
	DebugMode	AE 的 debug mode 参数
	IRCutOnLv	不使用 IR Cut 的环境 Lv 值, 精度为 100
	IRCutOffLv	使用 IR Cut 的环境 Lv 值, 精度为 100
	HistRatioSlope	目前 frame 的亮度与高光区亮度的混合权重
	MaxHistOffset	目前 frame 的亮度与高光区亮度的混合后, 提升的亮度范围
	ISONum-RangeMin	ISO num 范围的最小值
	ISONumRange-Max	ISO num 范围的最大值
	AdjustTarget-Max	各 LV 的 target max50,50, 50, 50, 50, 50, 55, 60, 60, 60, 60, 60, 60, 65, 70, 70
	AdjustTarget-Min	各 LV 的 target min40,40, 40, 40, 40, 40, 45, 50, 50, 50, 50, 50, 50, 55, 60, 60
ExposureInfo	ExposureTime	当前曝光时间, 单位为微秒 (us)。FSWDR 模式下, 表示当前最短帧 (VS) 曝光时间。
	ShortExpTime	FSWDR 模式下, 表示当前短帧 (S) 曝光时间, 单位为微秒 (us)。线性模式不用关注该值。
	LongExpTime	FSWDR 模式下, 表示当前长帧 (L) 曝光时间, 单位为微秒 (us)。线性模式不用关注该值。
	AGain	当前 sensor 模拟增益, 10bit 小数精度。
	DGain	当前 sensor 数字增益, 10bit 小数精度。
	ISPDGain	当前 ISP 数字增益, 10bit 小数精度
	AveLum	平均亮度信息。
	ISO	当前 sensor 模拟增益 *sensor 数字增益 *ISP 数字增益 *100, 其中增益的精度都为 10bit。
	WDRShort-AveLuma	WDR 短帧画面平均亮度信息。
	WDRExpRatio	WDR 长, 短帧的曝光比
	LEFrameAvgLuma	WDR 长帧画面的平均亮度
	SE-FrameAvgLuma	WDR 短帧画面的平均亮度
	LightValue	估计的环境亮度值
AE Route	TotalNum	曝光分配路线节点数目, 目前最大为 16
	AERouteNodeX.IntTime	曝光分配路线节点属性的节点曝光时间, 单位为微秒 (us)
	AERouteNodeX.SysGain	曝光分配路线节点属性的节点增益, 包括 sensor 模拟增益, sensor 数字增益和 ISP 数字增益, 10bit 精度。
	AERouteNodeX.IrisFNO	曝光分配路线节点属性的节点光圈 F 值大小, 仅支持 P-Iris, 不支持 DC-Iris
	AERouteNodeX.IrisFNOLin	曝光分配路线节点属性的节点光圈 F 值等效增益大小, 仅支持 P-Iris, 不支持 DC-Iris
WBAttr	ByPass	白平衡模块 Bypass 使能, 默认值 CVI_FALSE
	AWBRunInterval	白平衡模块工作频率
	OpType	自动白平衡和手动白平衡切换

下页继续

表 39.1 – 续上页

参数	描述
AlgType	ALG_AWB ALG_AWB_SPEC
DebugMode	debug mode
Rgain	手动白平衡红色通道增益
Grgain	手动白平衡 Gr 通道增益
Gbgain	手动白平衡 Gb 通道增益
Bgain	手动白平衡 Gb 通道增益
Enable	自动白平衡使能
RefColorTemp	静态白平衡系数标定的环境色温, 单位 Kelvin。推荐在 Macbeth D50 标准光源环境或室外晴天环境取得 24 色卡 Raw 数据进行标定
RGStrength	自动白平衡 R 通道校准强度
BGStrength	自动白平衡 B 通道校准强度
Speed	自动白平衡算法收敛速度
ZoneSel	参数为 0 或 255 时, 采用近似灰世界的白平衡算法, 其他值则为进行分类筛选, 提升精度
HighColorTemp	自动白平衡算法的色温上限
LowColorTemp	自动白平衡算法的色温下限
ShiftLimitEn	AWB 超过类白点范围的增益映像回白点范围的开关
GainNormEn	对 RGB 通道增益进行限制, 可以改善低色温、低照度场景的信噪比
NaturalCastEn	低色温下 AWB 风格喜好开关
AWBZoneWtEn	画面分区权重开关, 预设为关闭
stCTLimit.bEnable	定义自平衡的增益范围限制属性
stCTLimit.enOpType	Auto Mode
stCTLimit.HighRgain	手动模式下高色温下的最大 R 增益
stCTLimit.HighBgain	手动模式下高色温下的最小 B 增益
stCTLimit.LowRgain	手动模式下低色温下的最小 R 增益
stCTLimit.LowBgain	手动模式下低色温下的最大 B 增益
stLumaHist.bEnable	AW B 亮度与权重参数开关
stLumaHist.enOpType	Auto Mode
stCbCr-Track.bEnable	AWB 统计范围与 ISO 的连动参数
Tolerance	自动白平衡调整的偏差范围, 检测误差在门限范围内时,AWB 不作动作
ZoneRadius	自动白平衡统计中对像素分类时用的距离范围。该值越小,AWB 精度越高, 但会降低 AWB 算法稳定性
CurveLLimit	自动白平衡色温曲线的左边界限
CurveRLimit	自动白平衡色温曲线的右边界限
ExtraLightEn	自动白平衡计算时是否考虑色温曲线外的独立光源点, 最多四个独立点
WhiteBgain	特殊光源点的 B 通道增益
ExpQuant	根据外在亮度做判断。ExpQuant 为开启的亮度限制值 例如 ExpQuant = 6, 表示 LV6 以下开启此 WB 光源点 (一般夜景为 LV6 以下) ExpQuant =106 表示 LV6 以上开启 ExpQuant =112 表示 LV12 以上开启 (LV12 一般为户外)

下页继续

表 39.1 – 续上页

参数		描述
	LightStatus	特殊光源点的种类, 0: 不作动作 1: 加入光源点 2: 删除光源点附近的计算
	Radius	特殊光源点的区域大小,
	OutdoorStatus	室内或室外模式 (手动模式下)
	OutThresh	判定室内室外的阈值, 亮度小于时, 则判定为室内, 户外 LV 大多超过 15
	LowStart	将高色温的权重拉低, 高色温区的起始点, 建议为 6500K
	LowStop	将低色温的权重拉低, 低色温区的终止点, 建议为 4500K
	HighStart	将高色温的权重拉低, 高色温区的起始点, 建议为 6500K
	HighStop	将高色温的权重拉低, 高色温区的终止点, 建议为 8000K
	GreenEnhanceEn	在绿色植物场景下, 对绿色通道增加的开关
	OutShiftLimit	当判定为户外场景时,AWB 算法的白点范围限制
	MultiLightSourceEn	AWB 检测当前场景是否为混合光源, 来调整饱和度或 CCM
	MultiLSType	调整饱和度或是 CCM
	MultiLSScaler	当混合光源下调整饱和度或 CCM 的强度
	FineTunEn	AWB 特殊色检测开关, 例如肤色
	FineTunStrength	肤色、蓝色等特殊色检测的强度
	ShiftLimit	当判定为户外场景时,AWB 算法的白点范围限制
	CurvePara	CurvePara 【0-2】普朗克曲线系数, 由 AWB 标定工具给出。普朗克曲线描绘白色块在不同色温的标准光源下的颜色表现。CurvePara 【3-5】色温曲线系数, 由 AWB 标定工具给出。色温曲线描绘白色块的颜色表现与色温的对应关系。
	StaticWB	静态白平衡系数, 由 AWB 标定工具给出。
	AttrZoneWt	32x32 画面权重
	stLumaHist.HistTh	亮度分类的阈值 (手动模式下有效)
	stLumaHist.HistWt	亮度分类的权重 (手动模式下有效)
	stCbCrTrack.CrMax	不同 ISO 下 R/G 的最大值
	stCbCrTrack.CrMin	不同 ISO 下 R/G 的最小值
	stCbCrTrack.CbMax	不同 ISO 下 B/G 的最大值
	stCbCrTrack.CbMin	不同 ISO 下 B/G 的最小值
	MultiCTBin	色温分段参数
	MultiCTWt	色温分段权重
WBInfo	Rgain	当前 R 通道增益值
	Ggain	当前 G 通道增益值
	Bgain	当前 B 通道增益值
	Saturation	当前饱和度值
	ColorTemp	当前色温值
	InOutStatus	室内外检测结果

下页继续

表 39.1 – 续上页

参数		描述
	Bv	当前环境 bv 值
	CCM	当前颜色校正矩阵值，8bit 小数精度。bit15 是符号位，0 表示正数，1 表示负数，例如 0x8010 表示-16。
BNR	Enable	BNR 模块使能
	isManualMode	手动模式或自动模式
	WindowType	去噪滤波局域程度。其值越小，作用越局域。
	DetailSmooth-Mode	去噪细节平滑功能使能。
	NoiseSup-pressStr	噪声抑制强度。值越大，亮噪去除强度越大。
	FilterType	去噪滤波器强度。值越大，亮噪去除强度越大。
	NrLscRatio	参考 LSC 增益调节去噪强度。值越大，参考 LSC 增益的比例越多。
	VarThr	侦测边缘的阈值。值越大，判断为边缘的数量越少。
	CoringWgtLF	调节在低频区域的随机噪声强度。值越大，在低频区域保留的噪声越多。
	CoringWgtHF	调节在高频区域的随机噪声强度。值越大，在高频区域的保留的噪声越多。
	NonDirFiltStr	调节在低频区的去噪强度。值越大，在低频区域去除的噪声越多。
	VhDirFiltStr	调节在水平和垂直区的去噪强度。值越大，在水平和垂直边缘去除的噪声越多。
	AaDirFiltStr	调节在对角线边缘的去噪强度。值越大，在对角线边缘去除的噪声越多。
Crosstalk	Enable	GE 模块使能
	isManualMode	手动模式或自动模式
	Strength	G 通道平衡全局强度
	FlatThre	平坦区侦测节点 1-4 阈值
	GrGbD-iffThreSec	G 通道平衡节点 1-4 阈值
Demosaic	Enable	Demosaic 模块使能
	LumaTuned-CoringEn	(OpType, CoringEn) (x,0): by noise profile (1,1): by ISO table (0,1): by manual
	isManualMode	手动模式或自动模式
	CoarseEdgeThr	边缘粗调侦测阈值。值越小，侦测为边缘的数量越多。建议搭配参数 CoarseStr 调试。
	CoarseStr	边缘粗调强度值。值越小，越偏方向性的处理。反之，越偏无方向性的处理。
	FineEdgeThr	边缘细调侦测阈值。值越小，侦测为边缘的数量越多。建议搭配参数 FineStr 调试。
	FineStr	边缘细调强度值。值越小，越偏方向性的处理。反之，越偏无方向性的处理。
	De-tailSmoothEn-able	细节平滑功能使能

下页继续

表 39.1 – 续上页

参数		描述
	DetailSmooth-Str	细节平滑强度。值越大，平滑强度越强，对伪细节的抑制强度越大。
	DetailWgtThr	细节保留范围阈值。值越小，细节保留作用的范围越大。
	DetailWgtSlope	细节保留强度。值越大，细节保留越多。
	DetailWgtMin	边缘细节平滑保留允许之最小增益。
	DetailWgtMax	边缘细节平滑保留允许之最大增益。
	LumaWgtThr	亮度细节平滑作用范围阈值。值越小，细节平滑作用的范围越大。
	LumaWgtSlope	亮度细节平滑边缘保留强度。值越大，细节平滑强度越强。
	LumaWgtMin	亮度细节平滑强度允许之最小增益。
	LumaWgtMax	亮度细节平滑强度允许之最大增益。
	EdgeEnhanceEnable	边缘强化功能使能
	OverShtGain	调节 overshoot 程度。值越大，边缘强化越大。OverShtGain 为一倍时，其值为 256。
	UnderShtGain	调节 undershoot 程度。值越大，边缘强化越大。UnderShtGain 为一倍时，其值为 256。
	NoiseSuppressStr	噪声抑制强度。值越大，越能抑制噪声被强化。
	GainTable	无方向中频纹理增强强度。值越大，无方向中频纹理的锐度越高。该参数是一个 33 的数组，表现为一个连续 33 段的强度曲线。数值为 128 时，增益为一倍。
	LumaTuned-Coring	噪声抑制强度。值越大，越能抑制噪声被强化。
AE config	WinWeight	AE 测光权重

39.4.2 LEVEL2 级别调试信息分析

在 level1 的基础上，补充了 meshLscGainLut，exposureInfo 的额外资讯，以及 GammaTable

【参数说明】

参数		描述
Module/GammaEx	GammaTable	Gamma 曲线节点数值
ShadingEx	MeshLscGain-Lut	LSC 网格线形式红色通道补偿增益表, 包含 R G B 三个通道
ExposureInfoEx	Exposure	当前曝光量, 等于曝光时间与曝光增益的乘积, 其中曝光时间的单位为曝光行数, 曝光增益为 6bit 小数精度。
	ExposureIsMax	0: ISP 未达到最大曝光水平; 1: ISP 达到最大曝光水平。
	HistError	统计信息, AE 的目标亮度值与实际值的差该值为正表示当前期望的亮度信息大于实际的亮度信息, 该值为负表示期望的亮度信息小于实际的亮度信息。
	PirisFno	当前 P-Iris 光圈 F 值对应的等效增益。
	Fps	实际图像帧率 * 100。
	RefExpRatio	参考曝光比, 用于估计当前场景的动态范围, 会受到 ISP_WDR_EXPOSURE_ATTR_S 中 Tolerance 和 Speed 等值的影响。
	AE_RouteTotalNum	曝光分配路线节点数目, 目前最大为 16
	AERouteNodeX.IntTime	曝光分配路线节点属性的节点曝光时间, 单位为微秒(us)
	AERouteNodeX.SysGain	曝光分配路线节点属性的节点增益, 包括 sensor 模拟增益, sensor 数字增益和 ISP 数字增益, 10bit 精度。
	AERouteNodeX.IrisFNO	曝光分配路线节点属性的节点光圈 F 值大小, 仅支持 P-Iris, 不支持 DC-Iris
	AE_RouteNodeX.IrisFNOLin	曝光分配路线节点属性的节点光圈 F 值等效增益大小, 仅支持 P-Iris, 不支持 DC-Iris
	AE_Hist256Value	全局 256 段直方图统计信息

39.4.3 LEVEL3 级别调试信息分析

在 level2 的基础上, 补充了 3a 的统计值

【参数说明】

参数		描述
AE statistics	LEGlobalAvgR	长曝光帧全局统计的 R 分量统计平均值
	LEGlobalAvgGr	长曝光帧全局统计的 GR 分量统计平均值
	LEGlobal-AvgGb	长曝光帧全局统计的 GB 分量统计平均值
	LEGlobalAvgB	长曝光帧全局统计的 B 分量统计平均值
	SEGlobalAvgR	短曝光帧全局统计的 R 分量统计平均值
	SEGlobalAvgGr	短曝光帧全局统计的 GR 分量统计平均值
	SEGlobalAvgGb	短曝光帧全局统计的 GB 分量统计平均值
	SEGlobalAvgB	短曝光帧全局统计的 B 分量统计平均值
	LEHistogram-Mem	长曝光帧全局统计的像素 histogram 统计值

下页继续

表 39.2 – 续上页

参数		描述
	SEHistogram-Mem	短曝光帧全局统计的像素 histogram 统计值
	LEZoneRAvg	长曝光帧各分区间 R 通道的分量统计平均值
	LEZoneGrAvg	长曝光帧各分区间 Gr 通道的分量统计平均值
	LEZoneGbAvg	长曝光帧各分区间 Gb 通道的分量统计平均值
	LEZoneBAvg	长曝光帧各分区间 B 通道的分量统计平均值
	SEZoneRAvg	短曝光帧各分区间 R 通道的分量统计平均值
	SEZoneGrAvg	短曝光帧各分区间 Gr 通道的分量统计平均值
	SEZoneGbAvg	短曝光帧各分区间 Gb 通道的分量统计平均值
	SEZoneBAvg	短曝光帧各分区间 B 通道的分量统计平均值
AWB statistics	LEGlobalR	长曝光帧全局统计的 R 分量统计平均值
	LEGlobalG	长曝光帧全局统计的 G 分量统计平均值
	LEGlobalB	长曝光帧全局统计的 B 分量统计平均值
	LECountAll	长曝光帧全局统计的像素统计平均值
	SEGlobalR	短曝光帧全局统计的 R 分量统计平均值
	SEGlobalG	短曝光帧全局统计的 G 分量统计平均值
	SEGlobalB	短曝光帧全局统计的 B 分量统计平均值
	SECountAll	短曝光帧全局统计的像素统计平均值
	LEZoneAvgR	长曝光帧各分区间的 R 分量统计平均值
	LEZoneAvgG	长曝光帧各分区间的 G 分量统计平均值
	LEZoneAvgB	长曝光帧各分区间的 B 分量统计平均值
	LEZoneCountAll	长曝光帧各分区间的像素统计值
	SEZoneAvgR	短曝光帧各分区间的 R 分量统计平均值
	SEZoneAvgG	短曝光帧各分区间的 G 分量统计平均值
	SEZoneAvgB	短曝光帧各分区间的 B 分量统计平均值
	SEZoneCountAll	曝光帧各分区间的像素统计值

40 3A 开发指南概述

41 3A 开发用户指南

41.1 AF 统计信息使用说明

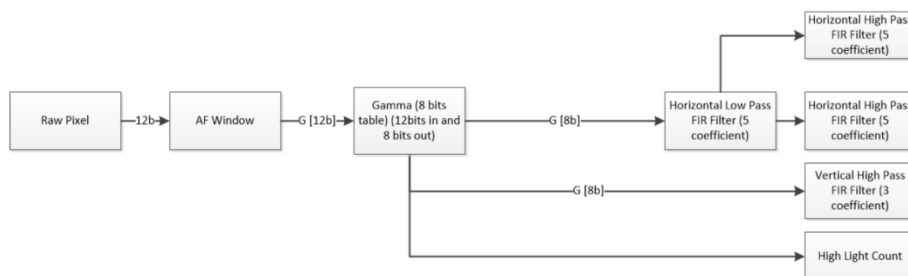
41.1.1 概述

自动对焦是通过分析图像的特征得到目前图像清晰值 FV(Focus Value), 由于图像越清晰 FV 越大, 因此经过比对每个位置的清晰值可以找出 FV 值曲线的最高点, 此位置即为对焦点, 找出对焦点后使用者即可控制对焦马达达到最佳位置完成自动对焦。

目前 AF 共提供四个滤波器与亮度讯息, 分别为水平方向 H1, H2, 垂直方向 V1, 其中水平方向会先经过一个低通滤波器后再分别经过两个高通滤波器以得到 H1,H2 的值, 垂直方向则只经过高通滤波器,

目前 AF 只支持 bayer 域的统计信息

Sequence



41.1.2 输入图像的裁剪

AF 支持对输入图像的裁剪, 用户可通过 stCrop 中的 X, Y, W, H 来决定目前 AF 的统计区域, 详情可以参考 ISP_STATISTICS_CFG_S 以获得相关结构信息

41.1.3 Bayer 域的配置

数据在进 AF 模块前有两样前处理可以使用

1. 使用者可选择是否需要经过 gamma, 假如需要的话, 需使能 gamma 并填入适当的 gamma 曲线
2. 使用者可选择是否开启预滤波处理以消除椒盐噪声对于统计值的影响

41.1.4 抑制光源对于 FV 值的影响

在图像中具有点光源的情况, 在对焦模糊时会因为光晕扩散影响到 FV 值, 而发生图像模糊 FV 值却增大的状况, 为了抑制此现象, 增加 u16HlCnt 来统计窗口中高亮度点的数值, 使用者可调整 u16HighLumaTh 来决定高亮点的阈值, 当模糊时窗口中的亮点数量增加, 清晰时窗口中的亮点个数最少, 用户可使用此信息来判断最适合的对焦点

41.1.5 统计信息配置注意事项

类型	描述
分块大小	最大 7 * 15
统计模块工作域	RAW
Bayer 统计参数是否减去黑电平	已减去

41.1.6 FV 值的获取

当图像的最后一个 pixel 通过后即可取得统计信息, 使用者可通过 CVI_ISP_GetVDTimeOut 以同步获得统计值, 可参考 33.1.8 的流程

41.1.7 FV 值的计算

一个 block 可取得的统计值有三种, 分别为 Horizontal 第一组 filter 得到的 H0, Horizontal 第二组 filter 得到的 H1, Vertical filter 得到的 V0, 我们将每个 block 的 FV 值命名为 FV_n, 为每个统计值设置自己的权重, 分别为 W0 / W1 / W2, 则 FV_n 的值为

$$FV_n = \frac{W_0 * H0_n + W_1 * H1_n + W_2 * V0_n}{W_0 + W_1 + W_2}$$

而最终的 FV 值还需要为每个 block 的 FV 加上权重, 假设第 n 个 block 的 weight 为 W_n, 最终 FV 的值则为:

$$FV = \frac{\sum_{n=0}^{blocks} FV_n * W_n}{\sum_{n=0}^{blocks} W_n}$$

41.1.8 FV 计算参考代码

```

ISP_AF_STATISTICS_S afStat;
CVI_U32 row, col;
CVI_S32 s32Ret = CVI_SUCCESS;
CVI_U64 stsValue = 0;
VI_PIPE ViPipe = 0;
ISP_VD_TYPE_E enIspVDType = ISP_VD_FE_START;
CVI_CHAR input[10];
ISP_STATISTICS_CFG_S stsCfg;
ISP_PUB_ATTR_S stPubAttr;
struct timeval t1, t2;
// AF weighting table
static int AFWeight[15][17] = {{1,1,1,1,1,1,1,1,1,1,1,1,1,1,1},
                                {1,2,2,2,2,2,2,2,2,2,2,2,2,2,1},
                                {1,2,2,2,2,2,2,2,2,2,2,2,2,2,1},
                                {1,2,2,2,2,2,2,2,2,2,2,2,2,2,1},
                                {1,2,2,2,2,2,2,2,2,2,2,2,2,2,1},
                                {1,2,2,2,2,2,2,2,2,2,2,2,2,2,1},
                                {1,2,2,2,2,2,2,2,2,2,2,2,2,2,1},
                                {1,2,2,2,2,2,2,2,2,2,2,2,2,2,1},
                                {1,2,2,2,2,2,2,2,2,2,2,2,2,2,1},
                                {1,2,2,2,2,2,2,2,2,2,2,2,2,2,1},
                                {1,2,2,2,2,2,2,2,2,2,2,2,2,2,1},
                                {1,2,2,2,2,2,2,2,2,2,2,2,2,2,1},
                                {1,2,2,2,2,2,2,2,2,2,2,2,2,2,1},
                                {1,2,2,2,2,2,2,2,2,2,2,2,2,2,1},
                                {1,1,1,1,1,1,1,1,1,1,1,1,1,1,1}};

// Get current statistic and related size setting.
s32Ret = CVI_ISP_GetStatisticsConfig(ViPipe, &stsCfg);
s32Ret |= CVI_ISP_GetPubAttr(ViPipe, &stPubAttr);
if (s32Ret != CVI_SUCCESS) {
    CVI_TRACE_LOG(CVI_DBG_ERR, "Get Statistic info fail with %s!\n", s32Ret);
    return s32Ret;
}
// Config AF Enable.
stsCfg.stFocusCfg.stConfig.bEnable = 1;
// Config low pass filter.
stsCfg.stFocusCfg.stConfig.u8HFiltShift = 0;
stsCfg.stFocusCfg.stConfig.s8HVFltLpCoeff[0] = 0;
stsCfg.stFocusCfg.stConfig.s8HVFltLpCoeff[1] = 1;
stsCfg.stFocusCfg.stConfig.s8HVFltLpCoeff[2] = 2;
stsCfg.stFocusCfg.stConfig.s8HVFltLpCoeff[3] = 3;
stsCfg.stFocusCfg.stConfig.s8HVFltLpCoeff[4] = 4;
// Config gamma enable.

```

(下页继续)

(续上页)

```

stsCfg.stFocusCfg.stConfig.stRawCfg.PreGammaEn = 0;
// Config pre NR enable.
stsCfg.stFocusCfg.stConfig.stPreFltCfg.PreFltEn = 1;
// Config H & V window.
stsCfg.stFocusCfg.stConfig.u16Hwnd = 17;
stsCfg.stFocusCfg.stConfig.u16Vwnd = 15;
// Config crop related setting. Has some limitation
stsCfg.stFocusCfg.stConfig.stCrop.bEnable = 1;
stsCfg.stFocusCfg.stConfig.stCrop.u16X = 8;
stsCfg.stFocusCfg.stConfig.stCrop.u16Y = 2;
stsCfg.stFocusCfg.stConfig.stCrop.u16W = stPubAttr.stWndRect.u32Width - 8 * 2;
stsCfg.stFocusCfg.stConfig.stCrop.u16H = stPubAttr.stWndRect.u32Height - 2 * 2;
// Config first horizontal high pass filter.
stsCfg.stFocusCfg.stHParam_FIR0.s8HFltHpCoeff[0] = 0;
stsCfg.stFocusCfg.stHParam_FIR0.s8HFltHpCoeff[1] = -3;
stsCfg.stFocusCfg.stHParam_FIR0.s8HFltHpCoeff[2] = 0;
stsCfg.stFocusCfg.stHParam_FIR0.s8HFltHpCoeff[3] = -10;
stsCfg.stFocusCfg.stHParam_FIR0.s8HFltHpCoeff[4] = 0;
// Config 2nd horizontal high pass filter.
stsCfg.stFocusCfg.stHParam_FIR1.s8HFltHpCoeff[0] = 0;
stsCfg.stFocusCfg.stHParam_FIR1.s8HFltHpCoeff[1] = -3;
stsCfg.stFocusCfg.stHParam_FIR1.s8HFltHpCoeff[2] = 0;
stsCfg.stFocusCfg.stHParam_FIR1.s8HFltHpCoeff[3] = -10;
stsCfg.stFocusCfg.stHParam_FIR1.s8HFltHpCoeff[4] = 0;
// Config vertical high pass filter.
stsCfg.stFocusCfg.stVParam_FIR.s8VFltHpCoeff[0] = 8;
stsCfg.stFocusCfg.stVParam_FIR.s8VFltHpCoeff[1] = -15;
stsCfg.stFocusCfg.stVParam_FIR.s8VFltHpCoeff[2] = 0;
stsCfg.unKey.bit1FEAfStat = 1;
s32Ret = CVI_ISP_SetStatisticsConfig(ViPipe, &stsCfg);
if (s32Ret != CVI_SUCCESS) {
    CVI_TRACE_LOG(CVI_DBG_ERR, "ISP Set Statistic failed with %#x!\n", s32Ret);
    return s32Ret;
}

printf("select. c -> Fv curve\n");
printf("..... h0 -> print h0 blocks statistic\n");
printf("..... h1 -> print h1 blocks statistic\n");
printf("..... v0 -> print v0 blocks statistic\n");
printf("..... hlc -> print hlcnt blocks statistic\n");
scanf("%s", input);

while(1) {
    // Wait VD start for get focus statistic data.
    s32Ret = CVI_ISP_GetVDTimeOut(ViPipe, enIsPVDType, 5000);
    s32Ret |= CVI_ISP_GetFocusStatistics(ViPipe, &afStat);
    if (s32Ret != CVI_SUCCESS) {
        CVI_TRACE_LOG(CVI_DBG_ERR, "Get Statistic failed with %#x!\n", s32Ret);
        return CVI_FAILURE;
    }
    // print each focus statistic.
    if (strcmp(input, "c", 1) != 0) {
        for (row = 0; row < AF_ZONE_ROW; row++) {
            for (col = 0; col < AF_ZONE_COLUMN; col++) {

```

(下页继续)

(续上页)

```

        if (strncmp(input, "h0", 2) == 0) {
            stsValue = afStat.stFEAFStat.stZoneMetrics[row][col].u64h0;
        } else if (strncmp(input, "h1", 2) == 0) {
            stsValue = afStat.stFEAFStat.stZoneMetrics[row][col].u64h1;
        } else if (strncmp(input, "v0", 2) == 0) {
            stsValue = afStat.stFEAFStat.stZoneMetrics[row][col].u32v0;
        } else {
            stsValue = afStat.stFEAFStat.stZoneMetrics[row][col].u16HlCnt;
        }
        printf("%d ", stsValue);
    }
    printf("\n");
}
continue;
}

CVI_U64 FVn = 0, FV = 0;
CVI_U32 totalWeightSum = 0;
// weight for each statistic
const CVI_U32 weight1 = 1, weight2 = 1, weight3 = 1;
const CVI_U32 blockWeightSum = weight1 + weight2 + weight3;
// calculate AF statistics
for (row = 0; row < AF_ZONE_ROW; row++) {
    for (col = 0; col < AF_ZONE_COLUMN; col++) {
        CVI_U64 h0 = afStat.stFEAFStat.stZoneMetrics[row][col].u64h0;
        CVI_U64 h1 = afStat.stFEAFStat.stZoneMetrics[row][col].u64h1;
        CVI_U32 v0 = afStat.stFEAFStat.stZoneMetrics[row][col].u32v0;

        FVn = (weight1 * h0 + weight2 * h1 + weight3 * v0) / blockWeightSum;
        FV += FVn * AFWeight[row][col];
        totalWeightSum += AFWeight[row][col];
    }
}
FV = FV / totalWeightSum;
}
return CVI_SUCCESS;

```

42 开发者指南

42.1 概述

当图像的最后一个 pixel 通过后即可取得统计信息，使用者可通过 CVI_ISP_GetVDTimeOut 以同步获得统计值，可参考 33.1.8 的流程。

因为 Linux 系统 user space 任务调度不能保证一致的实时性，需要将驱动配置放在 kernel space 完成。

ISP 提供同步回调接口的注册，可以实现与 VD 同步。

在本章有相应的接口使用描述，用户可以将实时性要求较高的任务放在同步回调里面，底层提供 HwIRQ，Workqueue 两种方式实现，可以选择相应的实现方式以确定实时级别。

HwIRQ 是指任务放在中断服务中实现，实时性最高，Workqueue 的实时性取决于 linux 系统调用。

42.2 API 参考

- `isp_sync_task_register`：向 ISP 注册同步回调接口。
- `isp_sync_task_unregister`：向 ISP 反注册同步回调接口。

42.2.1 `isp_sync_task_register`

【描述】

向 ISP 注册同步回调接口

【语法】

```
int isp_sync_task_register(int vi_pipe, struct isp_sync_task_node *new_node);
```

【参数】

参数名称	描述	输入/输出
vi_pipe	VI_PIPE 号	输入
new_node	新插入的同步回调节点	输入

【返回值】

返回值	描述
0	成功。
非 0	失败。

【需求】

- 头文件: `cvi_vip_isp_ext.h`
- 库文件:

【注意】

- 使用前需要确保 ISP 驱动已加载。
- 因为 ISP 同步回调内部实现不保存用户传入的 `new_node` 指向的实体，所以要求使用 `isp_sync_task_node` 定义实体时不能为局部变量。

【举例】

无。

【相关主题】

- `isp_sync_task_unregister`

42.2.2 isp_sync_task_unregister

【描述】

向 ISP 反注册同步回调接口

【语法】

```
int isp_sync_task_unregister(int vi_pipe, struct isp_sync_task_node *del_node);
```

【参数】

参数名称	描述	输入/输出
vi_pipe	VI_PIPE 号	输入
del_node	需要删除的同步回调节点	输入

【返回值】

返回值	描述
0	成功。
非 0	失败。

【需求】

- 头文件: `cvi_vip_isp_ext.h`
- 库文件:

【注意】

- 使用前需要确保 ISP 驱动已加载。

【举例】

无。

【相关主题】

- `isp_sync_task_register`

42.3 数据类型

- `isp_sync_tsk_method`：定义同步回调方法，决定实时性。
- `isp_sync_task_node`：定义同步回调节点信息。

42.3.1 `isp_sync_tsk_method`

【说明】

定义同步回调方法，决定实时性

【定义】

```
enum isp_sync_tsk_method {  
    ISP_SYNC_TSK_METHOD_HW_IRQ = 0,  
    ISP_SYNC_TSK_METHOD_WORKQUEUE,  
    ISP_SYNC_TSK_METHOD_BUTT  
};
```

【成员】

成员名称	描述
<code>ISP_SYNC_TSK_METHOD_HW_IRQ</code>	使用硬件中断方式回调。
<code>ISP_SYNC_TSK_METHOD_WORKQUEUE</code>	使用工作队列方式回调。

【注意事项】

无。

【相关数据类型及接口】

无。

42.3.2 isp_sync_task_node

【说明】

定义同步回调节点信息

【定义】

```
struct isp_sync_task_node {  
    enum isp_sync_tsk_method method;  
    __s32 (*isp_sync_tsk_call_back)(__u64 data);  
    __u64 data;  
    const char *sz_id;  
    struct list_head list;  
};
```

【成员】

成员名称	描述
method	回调方式。
isp_sync_tsk_call_back	回调函数，用户注册时传入。
data	回调函数参数，用户注册时传入。
sz_id	节点 ID。
list	list 节点，用于管理多个回调节点，无需关注。

【举例】

```
isp_sync_task_node sync_node = {  
    .method = ISP_SYNC_TSK_METHOD_HW_IRQ,  
    .isp_sync_tsk_call_back = sync_af_calc,  
    .data = 0,  
    .sz_id = "hw_0"  
};
```

【注意事项】

无。

【相关数据类型及接口】

- `isp_sync_tsk_method`

43 附录

44 缩略语
