

V 系列 Camera WDR 模式配置指南

【主题】

V 系列 Tina SDK 环境 Camera WDR 模式配置指南

【说明】

本文描述的配置方法仅适用 V 系列 (V53x、V83x、V85x 等) SDK 开发环境, 配置步骤涉及硬件支持、Tina SDK 内核选项配置、libisp 效果文件配置、多媒体 MPP VI 组件配置

【Camera 硬件支持】

- 1) 阅读 Camera 规格书, 或咨询原厂, 检查 Camera 型号是否支持 WDR/HDR 模式, 如图 1 所示, SC530AI 规格书会有 WDR 模式细节说明
- 2) 如图 2 所示, 参考 SC530AI 驱动为例, 实现 Camera 驱动与 WDR 模式相关的曝光/增益函数, 诸如曝光比, 长短帧曝光寄存器的使用细节均可在 Camera 规格书中获取
- 3) 如图 3 所示, 在规格书或原厂支持获取当前 WDR 模式类型及寄存器配置, 填写 Camera 驱动中 sensor_win_sizes 结构体

2.2. 宽动态

宽动态 (HDR) 是指通过把两帧相同场景、不同曝光时间的图片合成一帧, 从而提高图像的动态范围。SC530AI 支持行交叠 HDR。

2.2.1. 行交叠 HDR

SC530AI 行交叠 HDR 是指两种不同长短曝光时间的图像在帧内逐行交替输出。SC530AI 行交叠 HDR 的优势是同一像素的长短曝光时间间隔短, 这样进行 HDR 合成时, 可以一定程度上避免合成带来的拖尾现象。SC530AI 行交叠 HDR 是通过不同曝光实现的, 具有噪声小的优势。

SC530AI 可以通过 MIPI 接口的 virtual channel 来区分长短曝光数据, 默认长曝光的 virtual channel 为 2'b00, 默认短曝光的 virtual channel 为 2'b01。

SC530AI 行交叠 HDR 使用 virtual channel 数据读出时序如下图所示。

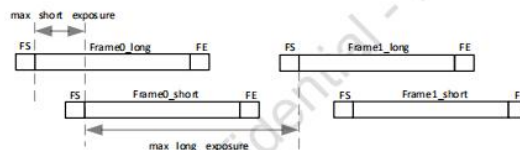


图 2-3 行交叠 HDR 使用 virtual channel 数据读出时序

SC530AI 也可不通过 virtual channel 区分长短曝光数据, 通过长短曝光数据读出行偏差来区分。这其中, 又分为两种模式, 模式 a 与模式 b。模式 a 时, 长短曝光数据只输出有效行; 模式 b 时, 长短曝光数据插入无效 (dummy) 行数据。

SC530AI 行交叠 HDR 不使用 virtual channel 时, 数据模式 a 的读出时序如下图所示。

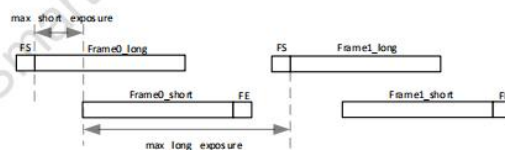


图 2-4 行交叠 HDR 不使用 virtual channel 时, 数据模式 a 读出时序

图 1

```
static int sensor_s_exp(struct v4l2_subdev *sd, unsigned int exp_val)
{
    data_type explow, expmid, exphigh = 0;
    struct sensor_info *info = to_state(sd);

    if (info->isp_wdr_mode == ISP_DOL_WDR_MODE) {
        if (exp_val < 16 * HDR_RATIO)
            exp_val = 16 * HDR_RATIO;

        exphigh = (unsigned char) (0x0f & (exp_val>>15));
        expmid = (unsigned char) (0xff & (exp_val>>7));
        explow = (unsigned char) (0xf0 & (exp_val<<1));

        sensor_write(sd, 0x3e02, explow);
        sensor_write(sd, 0x3e01, expmid);
        sensor_write(sd, 0x3e00, exphigh);

        sensor_dbg("sensor_set_long_exp = %d line Done!\n", exp_val);

        exp_val /= HDR_RATIO;

        expmid = (unsigned char) (0xff & (exp_val>>7));
        explow = (unsigned char) (0xf0 & (exp_val<<1));

        sensor_write(sd, 0x3e05, explow);
        sensor_write(sd, 0x3e04, expmid);

        sensor_dbg("sensor_set_short_exp = %d line Done!\n", exp_val);
    } else {
        if (exp_val < 16)
            exp_val = 16;

        exphigh = (unsigned char) (0xf & (exp_val>>15));
        expmid = (unsigned char) (0xff & (exp_val>>7));
        explow = (unsigned char) (0xf0 & (exp_val<<1));

        sensor_write(sd, 0x3e02, explow);
        sensor_write(sd, 0x3e01, expmid);
        sensor_write(sd, 0x3e00, exphigh);
        sensor_dbg("sensor_set_exp = %d line Done!\n", exp_val);
    }
    info->exp = exp_val;
    return 0;
}
```

图 2

```
{
    .width = 2880,
    .height = 1620,
    .hoffset = 0,
    .voffset = 0,
    .hts = 3200,
    .vts = 3300,
    .pclk = 316800000,
    .mipi_bps = 792 * 1000 * 1000,
    .fps_fixed = 30,
    .bin_factor = 1,
    .if_mode = MIPI_VC_WDR_MODE,
    .wdr_mode = ISP_DOL_WDR_MODE,
    .intg_min = 1 << 4,
    .intg_max = (2 * 3300 - 8) << 4,
    .gain_min = 1 << 4,
    .gain_max = 326 << 4,
    .regs = sensor_2880x1620_30fps_shdr_regs,
    .regs_size = ARRAY_SIZE(sensor_2880x1620_30fps_shdr_regs),
    .set_size = NULL,
},
```

图 3

【内核配置】

1) 如图 4 示，在 SDK 根目录执行 `make kernel_menuconfig`，打开 WDR 模式的内核选项

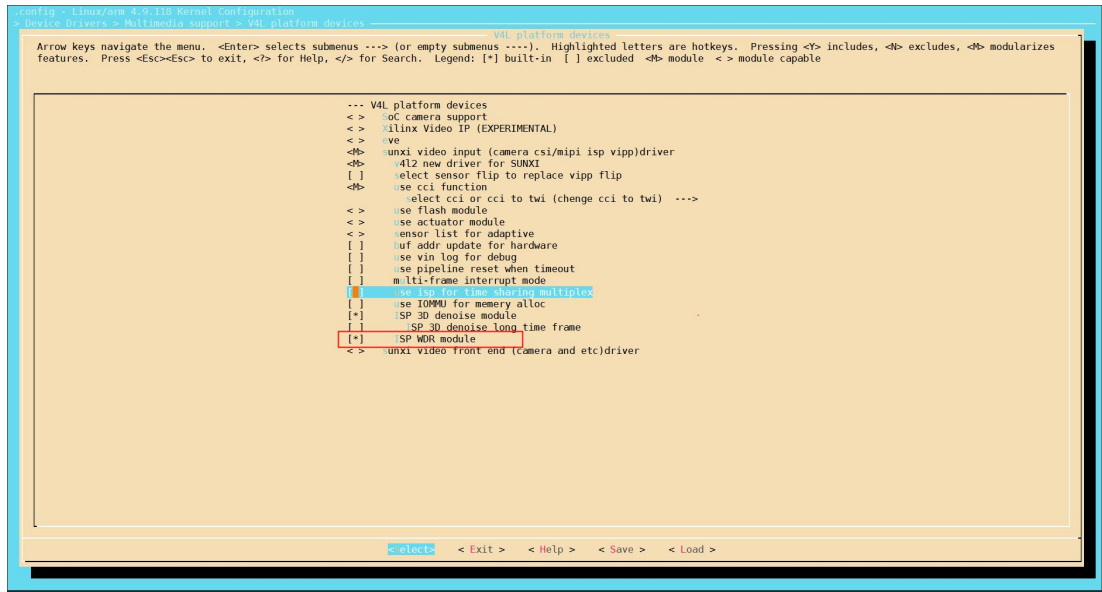
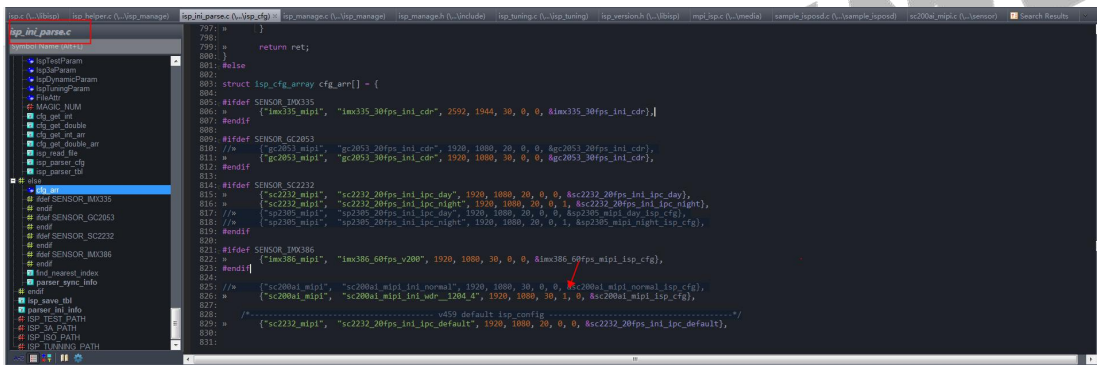


图 4

【配置 WDR 效果文件】

- 1) 在 libisp/isp_cfg/isp_ini_parse.c 中，包含 WDR 效果头文件
- 2) 完成 `isp_cfg_array` 结构体的配置，并将 `wdr` 标志位置 1



【MPP 配置 WDR 模式】

- 1) 在使用 MPP 初始化 `MPI_VI` 组件时，`VI_ATTR_S` 结构体的 `wdr_mode` 成员变量需要配置成 1，如图 5 所示，以 `sample_virvi` 为例

```

260:  AW_MPI_SYS_SetConf(&mSysConf);
261:  ret = AW_MPI_SYS_Init();
262:  if (ret < 0)
263:  {
264:      aloge("sys Init failed!");
265:      return -1;
266:  }
267:
268:  VI_ATTR_S stAttr;
269:
270:  /* dev:0, chn:0,1,2,3 */
271:  /* dev:1, chn:0,1,2,3 */
272:  /* dev:2, chn:0,1,2,3 */
273:  /* dev:3, chn:0,1,2,3 */
274:  /*Set VI Channel Attribute*/
275:  memset(&stAttr, 0, sizeof(VI_ATTR_S));
276:  stAttr.type = V4L2_BUF_TYPE_VIDEO_CAPTURE_MPLANE;
277:  stAttr.memtype = V4L2_MEMORY_MMAP;
278:  stAttr.format.pixelformat = stContext.mConfigPara.PicFormat; // V4L2_PIX_FMT_SBGGR12;
279:  stAttr.format.field = V4L2_FIELD_NONE;
280:  stAttr.format.width = stContext.mConfigPara.PicWidth;
281:  stAttr.format.height = stContext.mConfigPara.PicHeight;
282:  stAttr.nbufs = 5;
283:  stAttr.nplanes = 2;
284:  stAttr.fps = stContext.mConfigPara.FrameRate;
285:  /* update configuration anyway, do not use current configuration */
286:  stAttr.use_current_wdn = 0;
287:  stAttr.wdr_mode = 1;
288:  stAttr.capturemode = V4L2_MODE_VIDEO; /* V4L2_MODE_VIDEO; V4L2_MODE_IMAGE; V4L2_MODE_PREVIEW */
289:  stAttr.drop_frame_num = 0; // drop 2 second video data, default=0
290:  AW_MPI_VI_CreateVipp(vipp_dev);
291:  AW_MPI_VI_SetVippAttr(vipp_dev, &stAttr);
292:  AW_MPI_VI_EnableVipp(vipp_dev);
293:  #define ISP_RUN 1
294:  #if ISP_RUN
295:      int iIspDev = 0;
296:      /* open isp */
    
```

图 5

【检查通路模式】

- 1) 如图 6 所示，执行 `cat /sys/kernel/debug/mpp/vi`，检查当前的 `isp_mode` 是否已配置为需要的 WDR 模式

```

root@TinaLinux:/# cat /sys/kernel/debug/mpp/vi
*****
VIN hardware feature list:
mcsi 2, ncsi 1, parser 3, isp 1, vipp 4, dma 4
CSI_VERSION: CSI300_200, ISP_VERSION: ISP600_100
CSI_CLK: 339428571, ISP_CLK: 24000000
*****
vi0:
gc4663_mipi => mipi0 => csi0 => tdm_rx0 => isp0 => vipp0
input => hoff: 0, voff: 0, w: 2560, h: 1440, fmt: GRBG10
output => width: 2560, height: 1440, fmt: NV12M
interface: MIPI, isp_mode: DOL_WDR, hflip: 0, vflip: 0
prs_in => x: 2560, y: 1440, hb: 3864, hs: 3252
bkbuff => cnt: 3 size: 5529600 rest: 3, work_mode: online
frame => cnt: 125, lost_cnt: 0, error_cnt: 0
internal => avg: 50(ms), max: 50(ms), min: 49(ms)
*****
    
```

图 6