AX Sensor Bring-up Quick Start

文档版本：V1.5

发布日期：2024/07/18

目 录

权利声明

注意

您购买的产品、服务或特性等应受商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非商业合同另有约定，本公司对本文档内容不做任何明示或默示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

前 言

适用产品

- AX650A/N
- AX620E 系列产品（AX630C、AX620Q）

适读人群

- 软件开发工程师
- 技术支持工程师

符号与格式定义

符号/格式	说明
`xxx`	表示您可以执行的命令行。
斜体	表示变量。如，“*安装目录*/AX620E_SDK_Vx.x.x/build 目录”中的“安装目录”是一个变量，由您的实际环境决定。
说明/备注：	表示您在使用产品的过程中，我们向您说明的事项。
注意：	表示您在使用产品的过程中，需要您特别注意的事项。

修订历史

文档版本	发布时间	修订说明
V1.0	2024/04/15	文档初版
V1.1	2024/04/18	更新线性度测试说明，添加必要指令，如开 LOG，处理 LOG 数据等
V1.2	2024/06/11	步骤流程优化，并添加图片指导说明
V1.3	2024/07/01	添加 i2c 读取写入操作指令等
V1.4	2024/07/11	文档分离出演示视频到附件中
V1.5	2024/07/18	添加附加文档相关说明

1 介绍

1.1 概述

该文档提供 Sensor Bring-up 快速开始方法和调试的步骤，可快速熟悉 Bring-up 过程中需要注意的事项。

参考资料如下：

1. **线性度测试数据样例：**包含线性度测试数据，以及线性度演示视频

2. **AXERA sensor settings 需求文档-sensor-name.xlsx ：** 向 Sensor FAE 申请对应的 Setting 表格参考

3. **Sensor Checklist.xlsx ：** sensor bring up check list 样例表格其他相关文档：

   1. **12 - AX Sensor 调试指南.docx** (board 目录下)

   2. **03 - AX 图像在线调试指南.docx**(pc 目录下)

1.2 Sensor bringup 步骤

1) 确认需求（分辨率、帧率、接口类型、位宽等信息）；

2) 申请 setting（根据需求，找 sensor 厂家 fae 申请初始化配置序列与数据手册等）；

3) 准备硬件环境（获取 sensor 和确认连接原理图）；

4) 参考现有的驱动程序框架，并按照数据手册描述，编写驱动；

5) 驱动出图调试：

1） 基于 tuningserver/tuningtool, 调用 sensor 驱动；

2） 调试 I2C/SPI，获取 Chip ID，并完成 Sensor Setting 初始化；

3） 尝试用 tuningtool 进行预览实时画面。

6) 进行 Sensor checklist 功能测试（包含线性度测试）；

7) 输出测试报告。

2 具体步骤

2.1 需求确认

☞ 目标：明确项目所需的 sensor 规格，对齐需求

根据项目的实际场景需求，明确使用的 Sensor 类型以及不同参数需求，方便向 Sensor FAE 申请对应的 Setting。

例如可以按照如下方式(参考)，填写相关信息，并提供给 fae:

sensor基本信息		
sensor name	sensor-name	
interface type	MIPI	
MIPI line number	4 lane	
master/slave mode	master	
mipi data rate	<2.5Gbps	
HDR output mode	VC mode	
window croping	无	
setting	Settings-1	Settings-2
HDR mode	SDR	HDR
mode	stagger	stagger
input clock	0: 27M	0: 27M
resolution	1920*1080	1920*1080
fps	**30fps**	**30fps**
interface	MIPI	MIPI
MIPI Lane number	4lane	4lane
bit	10	10
bayer pattern	RGGB	RGGB

2.2 搭建硬件环境

☞ 目标：确认硬件管脚信息、硬件接线情况

硬件环境一般包括如下三个部分：

1）Sensor 端

2）板端（数据接收）

3）原理图（硬件连接）

通过原理图我们可以确认如下等相关信息：

例如：

- I2C Num
- Reset Gpio
- Mipi DEV （LANE 的硬件连接情况）
- Mclk Num

2.3 申请 Setting

> ☞ 目标：从 sensor 原厂获取该 sensor 的初始化序列

根据当前的项目实际需求，与 FAE 申请对应 Sensor Setting，申请可使用模板：AXERA sensor settings 需求文档-sensor_name.xlsx

除此之外，需要确认如下信息：

序号	item	支持情况	含义
1	支持的接口类型	*填写改sensor输出支持哪些类型（全集）*	MIPI/LVDS/DVP/BT
2	支持的HDR模式	*填写该sensor支持的HDR技术*	DCG-HDR、build-in HDR、DOL HDR、staggered hdr
3	支持的曝光类型	*填写：rolling shutter / global shutter*	rolling shutter / global shutter
4	是否支持 within / beyond	*填写支持哪种，或者两种都支持*	
5	Gain/Shutter寄存器生效时机	*填写：* *gain 第N帧配置，第N+x 帧生效* *shutter 第N帧配置，第N+x 帧生效*	多数sensor是：曝光时间及增益若在第 N 帧写入，第 N+2 帧生效
6	是否支持HCG/LCG	*填写：支持 / 不支持*	两者相互转换的转换比，ratio参数需要标定得到
7	HCG寄存器的生效时机	*填写：* *HCG寄存器在 第N帧配置，第N+x 帧生效*	sony sensor：HCG寄存器很多是N+1帧生效（如sony），跟gain的N+2生效不同步，需要驱动侧延后一帧配置HCG寄存器 ov sensor ： HCG 在帧头配置时会存在闪烁问题(一帧内的上下亮度不同)，需要借助group hold 机制配置HCG寄存器，再结合gain的生效时机，整体调试验证
8	快门调节步长	*填写： 1H / 2H / 0.5H*	多数是以1H为单位，也有些sensor以半行为单位（eg: SC530AI）
9	mirror / flip 特性	*填写：sensor端 支持 / 不支持 mirror & flip*	
10	是否支持读取温度	*填写：支持 / 不支持 读取sensor温度*	
11	开关流特性		开流后的第一帧，时序或者图像是否有异常？ HDR 关流后的最后一帧， 能否输出完整的时序？
12	曝光固定偏移时间	*填写：* *SDR 下， exp offset 为 xxx行* *HDR 下， exp offset 为 xxx行*	在N行曝光的基础上，sensor实际曝光的行数是 N+offset， offset一般小于1H
13	是否支持master/slave 模式	*填写：* *支持 / 不支持 master模式* *支持 / 不支持 slave模式*	是否支持主从模式，以及如果实现两个sensor的曝光同步

申请的邮件可以按照如下方式与 FAE 确认信息，方便后续 Bring-up 调试

1. Sensor 的 mipi clk 是否为非连续模式（请帮忙默认配置为非连续模式）。
2. Sensor 的 again、dgain 的表和相关的计算方法。
3. gain(again dgain) 、shutter 线性度相关的数据（测试环境，线性度情况[线性增长过程数据是否有偏移情况，是否按比例增加等）。
4. sensor 的 行长 需要大于（分辨率宽度 * （1 / 416））us（sdr & hdr）。

5. 是否支持 hcglcg,如果支持，对应的 ratio 是多少？
6. Gain/Shutter 寄存器生效时机、HCG 寄存器的生效时机。
7. 是否支持读取温度。
8. 曝光是否有固定偏移时间。
9. 是否支持 master/slave 模式。
10. 是否支持软复位功能。

2.4 编写 Sensor 驱动

> ☞ **目标：编写 sensor drv 库源码，能够编译通过（warning 也需要解决），动态库成果物为 libsns_xxx.so**

1) 准备驱动程序框架。

这里可以参考 SDK 中原有的 Sensor 驱动框架为基础，进行修改适配。推荐找到对应的 sensor 厂家的相关驱动，这样驱动的适配逻辑，更加的符合当前需求。

2) 按照 Sensor 数据手册相关描述，修改 Sensor 各种参数得限制数值。

这里数值，可以参考驱动中相关的宏定义，一般是，shutter gain vts ratio 最大小值信息，与 shutter offset 等。

具体可参考文档：**《12 - AX Sensor 调试指南.docx》**

2.5 调试 sensor 出图

> ☞ 目标：VIN DEV/PIPE/CHN 都能够正常出图，帧计数正常增加；AE 曝光控制逻辑正常运行

SDK 提供 tuning-server 可执行程序，可在不修改代码情况下，通过添加对应 Sensor ini 配置文件，即可调用 Sensor 驱动进行调试、预览以及相关功能测试，方便快速集成调试不同 Sensor 驱动。

该方式可动态链接 Sensor 驱动 so 动态库，从而获取 Sensor 的回调接口函数，实现对 Sensor 控制操作。具体文件示例在代码 app\camera\etc\external\文件夹 ini 中，一般配置需要改动的参数即可，其他保持默认即可

1) 需要修改的参数

Enter ini

需要修改 Dev num、Reset gpio、Sensor 通信接口类型 i2c/spi、I2c num 等，如下图：

os04a10_single_sdr_online_entry.ini M

camera > etc > external > os04a10_single_sdr_online_entry.ini

You, 1 second ago | 3 authors (yu juan and others)

```
[path]
tuning_bin_path = /opt/etc/
sns_ini_path = /opt/etc/
ext_ipt_ini_path = /opt/etc/
sns_drv_path = /opt/lib/
raw_data_path = /opt/data/

[port]
nNtStreamPort = 6000        ; The net stream port
nNtCtrlPort = 8082          ; The net ctrl port

[isp0]
sensorIniFileName = os04a10_sdr_online.ini
extIptIniFileName =
sensorTuningBinName = null.bin

nRxDev = 0
nDevId = 0
nPipeId = 0
nBusType = 0        ;  0: i2c (default setting), 1: spi, 2: invalid value
nDevNode = 0        ; -1: Use this value only on AX demo&evb  board, 0: i2c-0, 1: i2c-1
ePhySel = 0
nSnsClkIdx = 0    ;sensor clock index, support 0 to 5, 0-->mclk0,  1-->mclk1
eSnsClkRate = 24000000
```

Sensor ini

需要修改 分辨率、帧率、位宽、Raw type、Mipi lane num、Mipi vc dt，如下图：

os04a10_sdr_online.ini

camera > etc > external > os04a10_sdr_online.ini

linerhuan, 4 months ago | 7 authors (yu juan and others)

```
##
# os04a10 sdr
##

[sensor]
nSensorObjName = gSnsos04a10Obj
nSensorLibName = libsns_os04a10.so
nWidth = 2688
nHeight = 1520
fFrameRate = 30
eSnsMode = 1          ; linear: 1, hdr_2x: 2, hdr_3x: 3
eRawType = 10         ; raw8: 8, raw10: 10, raw12: 12, raw14: 14, raw16: 16
eBayerPattern = 0         ; 0:RGGB, 1:GRBG, 2:GBRG, 3:BGGR
bTestPatternEnable = 0
eMasterSlaveSel = 0    ; 0:master 1:slave
nSettingIndex = 0      ; 0 useless

[mipi]
eLaneComboMode = 4  ; 0: 8lane + 8lane
nMipiRxID = 0
eInputMode = 0      ; MIPI: 0, SUBLVDS: 1, LVDS: 2, HISPI: 3, SLVS: 4
# MIPI #
ePhyMode = 0        ; DPHY: 0, CPHY: 1
eLaneNum = 4
nDataRate = 80     ; Mbps
nDataLaneMap0 = 0       ; FIXME
nDataLaneMap1 = 1
nDataLaneMap2 = 3
nDataLaneMap3 = 4
nClkLane0 = 2       ; FIXME
nClkLane1 = 5
```

2) 配置好以上等信息 即可在板端准备运行

tuning-server -p /opt/etc/sensor-name_single_sdr_4lane_entry.ini

3) Tuning-Tool 的连接方式和使用方法可以参考文档**《03 - AX 图像在线调试指南.docx》**

2.6 调试 I2C,获取 sensor id

☞ 目标：调通硬件，完成 sensor 初始化

运行 tuningserver 后，会调用 sensor 的 get chip id 回调接口

成功获取到 chip id 需要满足以下几个条件：

- 硬件连接正确（和原理图对应）；
- reset gpio 配置正确并成功复位；
- i2c num 配置正确；
- i2c slave addr 配置正确；
- chip id 寄存器地址配置正确；
- mclk num 配置正确，并成功开启；

如果这里的信息确认完成，调用 get chip id 接口就可以获取到 chip id,

可以通过修改 log 打印等级，查看获取到的 chip id 是否和预期的一致（数据手册可以找到）

如果获取 id 出现异常，请参考 FAQ 获取不到 chip id 怎么办？

成功获取 id 后，会完成 sensor init，和开流操作

接下来就可以使用 tuning tool 尝试预览实时画面了！

2.7 Checklist 功能测试

Checklist 主要分为两个部分：

1.check sensor 各个 setting 的基本信息和功能;

2.check gain 与 shutter 的线性度。

2.7.1 Checklist 表格

> ☞ 目标：检查 sensor 每一项功能是否运行正常

Checklist 表格主要分为两个部分：

1.确认 sensor 各个 setting 的基本信息

如位宽、分辨率、mipi lane、行长、blc、曝光最大小值、mipi clk mode 等信息。

2.Check sensor 各个部分的功能是否正常和满足预期，具体分如下四个部分：

1） 基本 case

check sensor 运行参数是否和需求中的需求点一致

同时看出图是否正常

2） 曝光相关

check 曝光相关的范围信息，线性度是否满足要求等

3） gain 相关

check 增益相关的范围信息，线性度是否满足要求等

4） 其他

check，软硬件复位是否支持且功能是否正常，硬件模组是否正常（是否有偏色等），

抓图预览是否正常等

具体可以参考文档 ： **Sensor Checklist.xlsx**

Check 过程部分需要查看 log 信息来确认相关参数，打开 log 方式如下：

开 AE log: **echo ulog 42 7 > /proc/ax_proc/logctl**

开 sensor log: **echo ulog 34 7 > /proc/ax_proc/logctl**

2.7.2 线性度测试

☞ 目标：确定 shutter、again、dgain 等曝光量是否线性变化

此处参考文档：MC20E_sensor_name_线性度测试数据_data.xlsx

理论上来说，对不同帧设置不同的 gain/shutter，帧与帧之间的画面亮度与其 gain/shutter 的倍数关系相同，但在实际中，驱动配置 gain/shutter 与 AE 曝光不同步，那么随着 gain/shutter 的变化，容易造成画面亮度变化不呈线性关系。如果人工去测量不同 gain/shutter 下画面亮度的线性关系，工作量非常大。

测试方法：

执行 *echo ulog 25 7 > /proc/ax_proc/logctl* [配置 log 输出]

支持在 Manual 模式下进行 again/dgain/shutter 扫描，支持 sdr/hdr_2x，lcg/hcg，相应信息输出在 log 中，log 等级是 axae_dbg，可通过 log 标识 [linearity test] 进行筛选。

Hdr 模式通过 long、short 区分长短帧 log 数据。

处理 log 数据，方式示例如下：

Sdr:

cat /opt/data/AXSyslog/syslog/.log | grep -nri "linearity test" > sensor-name_sdr_again.txt*

Hdr :

Long frame:

cat /opt/data/AXSyslog/syslog/.log | grep -nri "linearity test" | gre*

p "long" > sensor-name _hdr_again_long.txt

Short frame:

cat /opt/data/AXSyslog/syslog/.log | grep -nri "linearity test" | gre*

p "short" > sensor-name _hdr_again_short.txt

过滤短帧 meanluma 信息方法：

tail -f /opt/data/AXSyslog/syslog/.log | grep meanLumaShortFrame*

注意：每次新的测试，需要清除上一次得 log : *rm -rf /opt/data/AXSyslog/**

对应的 log 输出信息示例：

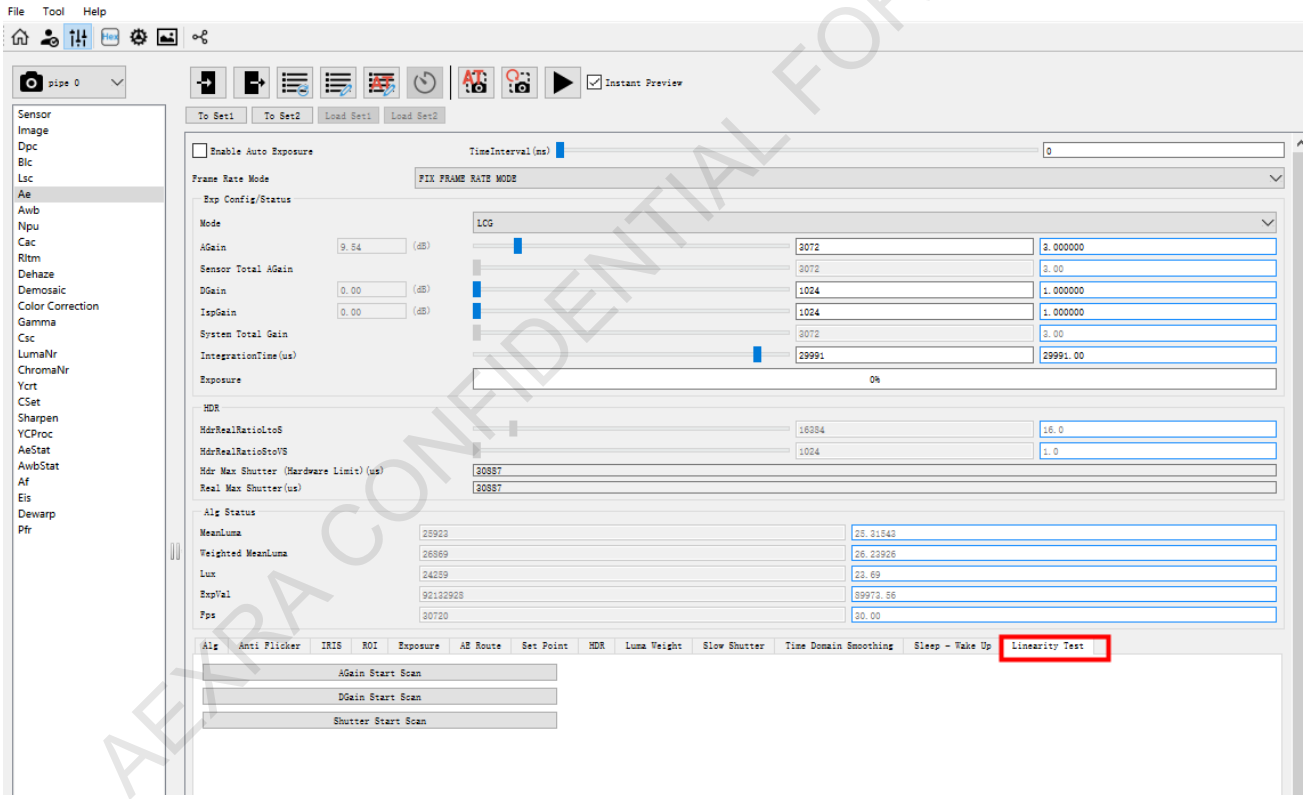
sdr：

[linearity test]: pipe, snsRegAgain , hcgstatus, snsTotalAgain, snsDgain , snsShutter , meanLuma , meanLumaShortFrame

hdr：

[linearity test]: pipe, snsRegAgain, hcgstatus, snsTotalAgain, snsDgain, snsShutter, hdrRealRatioLtoS, meanLuma, meanLumaShortFrame

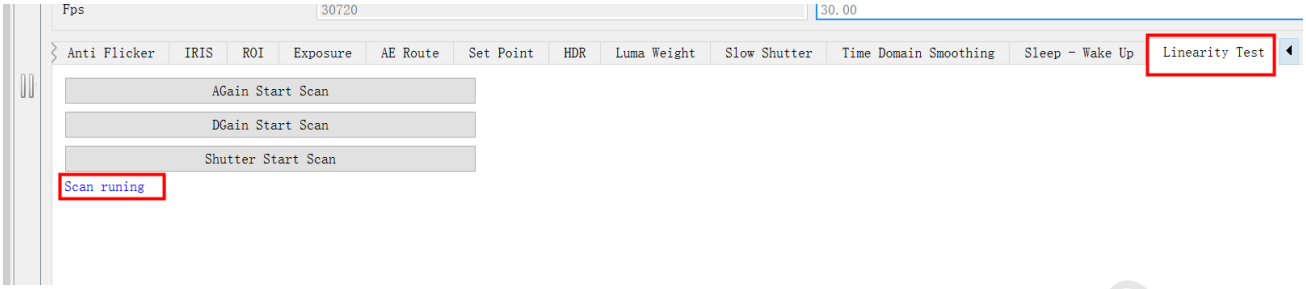
Linearity Test UI 界面:



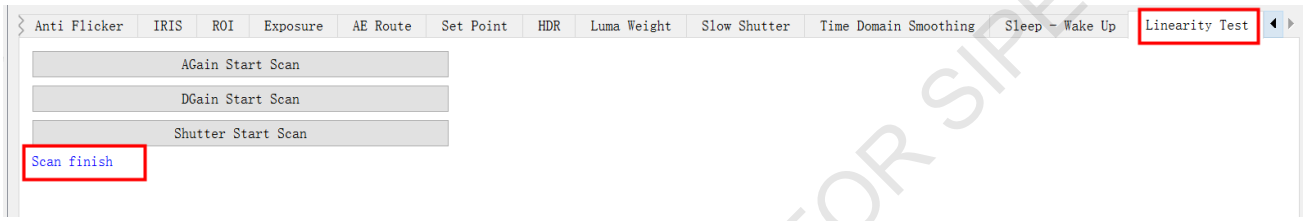
共有三个扫描按钮：

- *AGain Start Scan*
- *DGain Start Scan*
- *Shutter Start Scan*

点击按钮即开始扫描，界面提示 *Scan runing*

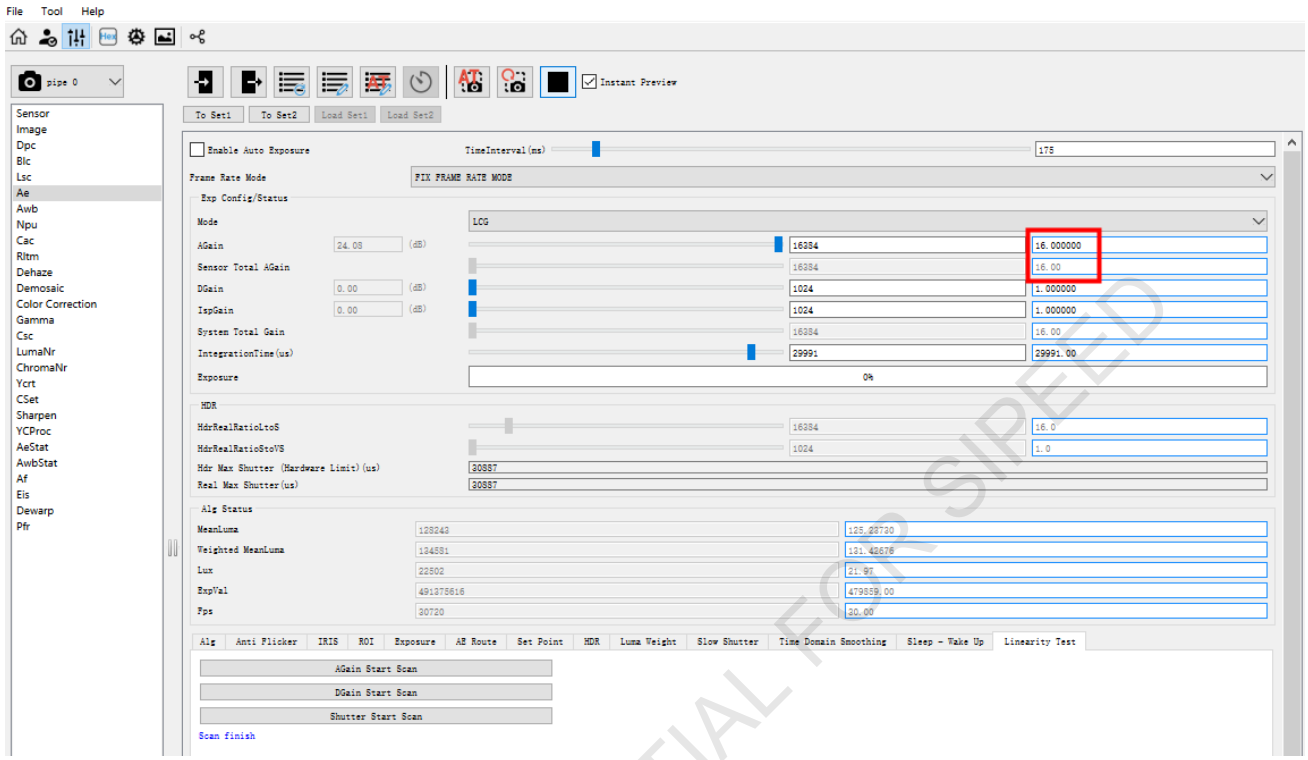


扫描结束之后，界面提示 *Scan finishing*



1. 在扫描某个参数之前，需通过调节其他参数，使最小 meanluma 在 2 以上（防止极暗场景下噪声过大），且最大 meanluma 不达到饱和状态，保证测试结果可以看到完整的线性趋势。
2. 在每次扫描结束之后，需重新将不进行扫描的其他参数手动调整到合适的值，再进行下次扫描。

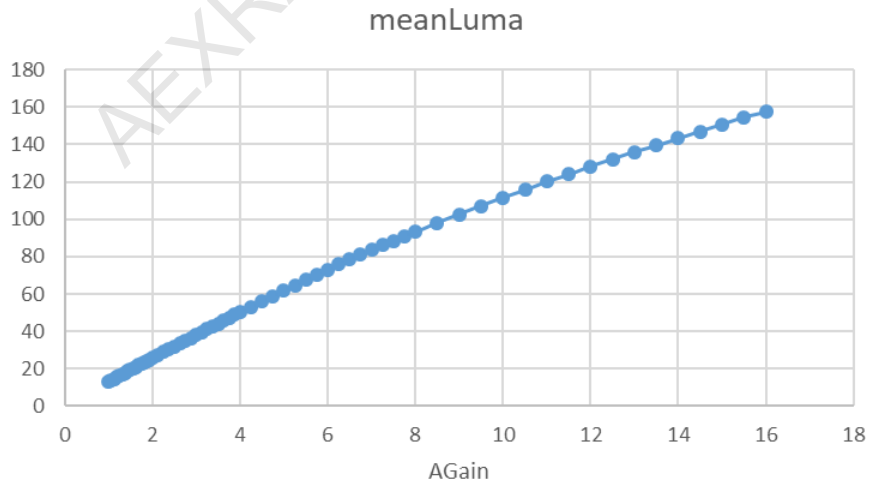
例如：在 again 扫描结束之后，工具界面上 again 的值会停留在 again table 中的最大值，此时若直接开启另一个参数的扫描，会在 max again 的状态下进行扫描，画面亮度很快就达到饱和状态，测试结果不能看到完整的线性趋势，此时需要手动将 again 调整到一个合适的值，再进行扫描。



shutter 线性度扫描测试需要在灯箱无 flicker 下进行，在测试之前，可将曝光时间设置为 1ms，检查灯箱是否无 flicker 现象。

验收标准

绘图：当 gain/shutter 增大时，画面亮度应该随着 gain/shutter 的值线性增大，如下图是在 sdr-lcg 模式下，当 dgain=1，shutter=29999 时，对 again 进行扫描的测试结果，随着 again 的增大，画面的 meanluma 呈现出线性增大的趋势。



计算不同 gain/shutter 值之间 meanluma 的斜率，当差异在容忍度之内，认为不同 gain/shutter 值之间的 meanluma 是呈线性关系的。

3. FAQ

获取不到 chip id 怎么办？

首先运行 tuningserver,确保 mclk 已经打开（sensor 需要有 mclk，i2c 才能正常响应）

尝试使用 i2cdetect debug,sensor 是否可以正常识别

首先确认 sensor 挂在哪个 i2c bus 上，

几个常用的 i2cdetect 指令如下：

1)查看当前 i2c bus 注册情况：

i2cdetect –l

这个指令可以查看当前已经注册的 i2c bus

```
/ # i2cdetect -l
i2c-3   i2c             Synopsys DesignWare I2C adapter         I2C adapter
i2c-8   i2c             Synopsys DesignWare I2C Slave adapter   I2C adapter
i2c-4   i2c             Synopsys DesignWare I2C adapter         I2C adapter
i2c-2   i2c             Synopsys DesignWare I2C adapter         I2C adapter
i2c-0   i2c             Synopsys DesignWare I2C adapter         I2C adapter
```

2)查看当前 i2c bus 硬件有哪些可以识别到的 i2c 设备的 slave addr

i2cdetect -r -y 0 [0 为 i2c bus num]

```
/ # i2cdetect -r -y 0
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- 21 -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- 36 -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
```

如上图，可以看到 i2c-0 有 slave addr 为 0x36 的设备

如果这个地址和你的 sensor 的一致，那么就代表硬件上可以识别到 sensor

否则就要排查，硬件连接是否 ok,sensor 是否有异常等。

在识别到 i2c 设备的情况下，可以通过如下指令，读取 sensor 具体的寄存器数值

i2c 读取 0x0100 数值:

i2ctransfer -y -f 0 w2@0x36 0x01 0X00 r1

i2c 写入 0x0100 数值 0x01:

i2ctransfer -y -f 0 w3@0x36 0x01 0x00 0x01

相关命令参数解释如下:

-f 0 是 i2c num

@0x36 是 i2c 设备地址

r1 是 读取一个寄存器数值

w2 是读取 [寄存器地址是两个字节情况下]

w3 是写入 [寄存器地址是两个字节情况下]

HDR Ratio 实际亮度比值不符合预期

在 hdr 模式下，长短帧的曝光比值预期和实际图像亮度比值一致

如果出现不匹配的异常，可能是如下原因导致：

背景知识：

有的 sensor 曝光最小值可以设置为 0，但是实际的预览，还是可以看到图像

这是为什么呢？（理论上，快门时间为 0，整个画面应该是完全黑的）

原因是，sensor 在设计的时候，曝光就不会真实的设置为 0（虽然表面看寄存器配置为 0）

这时就会有一个 offset 的数值，一般 sdr 和 hdr 的数值不同，并且数值大小一般不大于一行

解决方法：

在驱动中是有相关的 offset 变量传递给算法的，算法拿到这个数值，会计算出真实的 ratio 比

值，给后级模块

这个 offset 的数值，需要在 setting 申请过程和 fae 确认好即可

配置示例如下：

```
#define OS04A10_EXP_OFFSET_SDR          (0.3f) //unit:line
#define OS04A10_EXP_OFFSET_HDR_2STAGGER (0.6f)
#define OS04A10_EXP_OFFSET_HDR_3STAGGER (0.75f)
```

```
if (sns_obj->sns_mode_obj.eHDRMode == AX_SNS_LINEAR_MODE) {
    sns_obj->ae_ctrl_param.sns_ae_param.fIntegrationTimeOffset[HDR_LONG_FRAME_IDX] = OS04A10_EXP_OFFSET_SDR;
} else if (sns_obj->sns_mode_obj.eHDRMode == AX_SNS_HDR_2X_MODE) {
    sns_obj->ae_ctrl_param.sns_ae_param.fIntegrationTimeOffset[HDR_LONG_FRAME_IDX] = OS04A10_EXP_OFFSET_HDR_2STAGGER;
    sns_obj->ae_ctrl_param.sns_ae_param.fIntegrationTimeOffset[HDR_MEDIUM_FRAME_IDX] = OS04A10_EXP_OFFSET_HDR_2STAGGER;
} else if (sns_obj->sns_mode_obj.eHDRMode == AX_SNS_HDR_3X_MODE) {
    sns_obj->ae_ctrl_param.sns_ae_param.fIntegrationTimeOffset[HDR_LONG_FRAME_IDX] = OS04A10_EXP_OFFSET_HDR_3STAGGER;
    sns_obj->ae_ctrl_param.sns_ae_param.fIntegrationTimeOffset[HDR_MEDIUM_FRAME_IDX] = OS04A10_EXP_OFFSET_HDR_3STAGGER;
    sns_obj->ae_ctrl_param.sns_ae_param.fIntegrationTimeOffset[HDR_SHORT_FRAME_IDX] = OS04A10_EXP_OFFSET_HDR_3STAGGER;
} else {
    sns_obj->ae_ctrl_param.sns_ae_param.fIntegrationTimeOffset[HDR_LONG_FRAME_IDX] = OS04A10_EXP_OFFSET_SDR;
}
```