



AX RISCV 使用说明

文档版本：V0.2

发布日期：2024/05/16

目 录

前 言	3
修订历史.....	4
1 RAW 检测简介.....	5
1.1 Raw 检测架构.....	5
2 RAW 检测配置.....	6
2.1 Raw 检测应用.....	6
2.2 Raw 检测配置.....	6
3 RAW 检测测试.....	9

权利声明

爱芯元智半导体股份有限公司或其许可人保留一切权利。

非经权利人书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

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

AX620Q 系列产品

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。
☞ 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，我们需要您注意的事项。

修订历史

文档版本	发布时间	修订说明
V0.1	2024/04/10	文档初版
V0.2	2024/05/16	新增动态配置 raw 检测帧数

1 RAW 检测简介

在 AX620E 快速启动中，增加了 riscv raw 检测功能，主要目的是为了节省功耗。在快速启动过程中，riscv 接收到 sensor 数据后，将 raw 数据交给 npu 做检测，如果检测人，车目标则继续启动 linux 检测；否则关闭 soc power 节省电源消耗，等待外部 MCU 下次唤醒检测。

1.1 Raw 检测架构

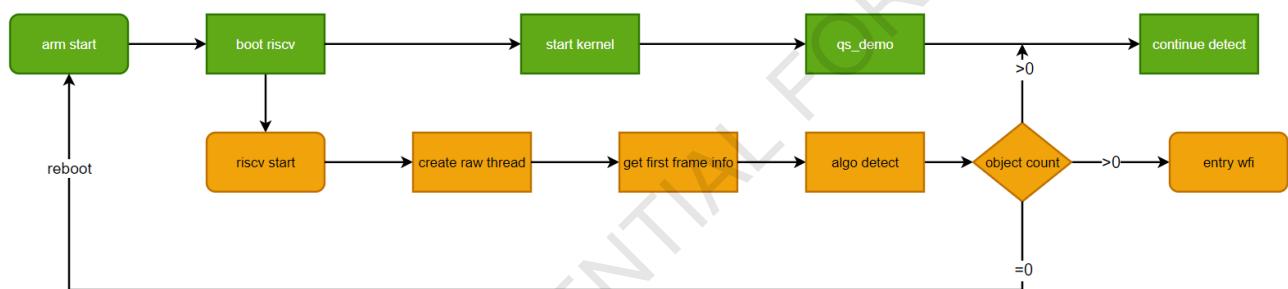


图1-1 RAW 检测架构

Arm 侧启动，spl 启动 riscv，arm 继续启 kernel 等流程。arm 启动的同时，Riscv 启动，然后 riscv 创建 raw/qs_sensor 等线程，raw 检测线程从 qs_sensor 线程中获取 frame raw 数据信息，将 raw 数据信息交给 npu 检测，npu 将检测结果返回后，判断是否检测到人，车目标。检测到则交给 linux 继续检测，riscv 进入休眠；没检测到，riscv 重启 soc。

2 RAW 检测配置

2.1 Raw 检测应用

raw 检测应用代码及检测模型位置：

code: riscv/applications/detect

在 AX620E SDK 中， raw 检测默认检测一帧 raw 图，最大可配置检测 4 帧 raw 图， raw 检测流程是检测到人，交由 linux 继续执行检测；没检测人， riscv 重启 soc。用户可以根据自己的策略修改此部分流程。

```
if (!info.objects.inserted_count)
| do_reboot(); //No object detected, do reboot
```

model: riscv/applications/test_engine/model.h

此部分是 raw 检测模型存放位置，模型数据以数组形式存储，如果用户需要替换自己的检测模型，可以替换此数组数据。

2.2 Raw 检测配置

目前 SDK 中仅 fastnor 工程支持 riscv raw 检测功能， raw 检测功能配置如下：

- 方式一：默认配置 RAW 检测模式(SDK 中默认配置为 YUV 检测)

编译：**make p=AX620Q_fastnor_arm32_k419 clean all install -j128**

- 方式二：外部编译参数选择检测模式

编译：**make p=AX620Q_fastnor_arm32_k419 detect=raw clean all install -j128**

```

77 # The detect type can be modified by compiling parameters
78
79 ifeq ($(strip $(detect)), raw)
80 DETECT_TYPE := raw
81 AX_FAST_RISCV_RAW_DET_SUPPORT := TRUE
82 else ifeq ($(strip $(detect)), md)
83 DETECT_TYPE := md
84 AX_FAST_RISCV_MD_DET_SUPPORT := TRUE
85 else ifeq ($(strip $(detect)), yuv)
86 DETECT_TYPE := yuv
87 AX_FAST_RISCV_YUV_DET_SUPPORT := TRUE
88 else
89 AX_FAST_RISCV_YUV_DET_SUPPORT := TRUE #default config
90 endif

```

- 动态配置 raw 检测帧数

Linux: 执行 ax_env.sh set raw_det_count 4 (默认一帧, 最大不超过 4 帧)

Riscv: 通过 fw_getenv 获取 raw detect count

```

char *detect_num_str = fw_getenv(FW_ENV_RAW_DET_COUNT);
if (detect_num_str != NULL)
    detect_count = (unsigned int)atoi(detect_num_str);

if ((detect_count == 0) || (detect_count > DETECT_MAX_COUNT))
    detect_count = DETECT_MAX_COUNT;

```

Riscv partition size 配置

由于 raw 检测模型数据是以数组形式存放在 riscv 代码中, 编译出的镜像会很大, 所以 RISCV 分区需要足够大的 flash 空间来存储 riscv 镜像。在 fastnor 工程中, sdk 中 raw 检测模式 riscv partition size 默认设置为 2M。

```

RISCV_PARTITION_SIZE := 2M
CUSTOMER_PARTITION_SIZE := 1M
AUTO_FIT_PARTITION := OPT

```

！ 注意：

- 用户替换自己的模型时，需要注意替换模型的大小，需要注意编译出来压缩后的 riscv 镜像是否超过配置的 riscv partition size，用户可以通过增大 riscv partition size 来兼容更大的检测模型，但需要注意的是，ax620e nor 共有 16MB，sdk 中目前非 recovery 模式只有 1M flash 空间剩余。
- 由于 nor flash size 不足原因，用户在开启 raw 检测功能时，不能同时支持开启 recovery 模式

Riscv mem size 配置

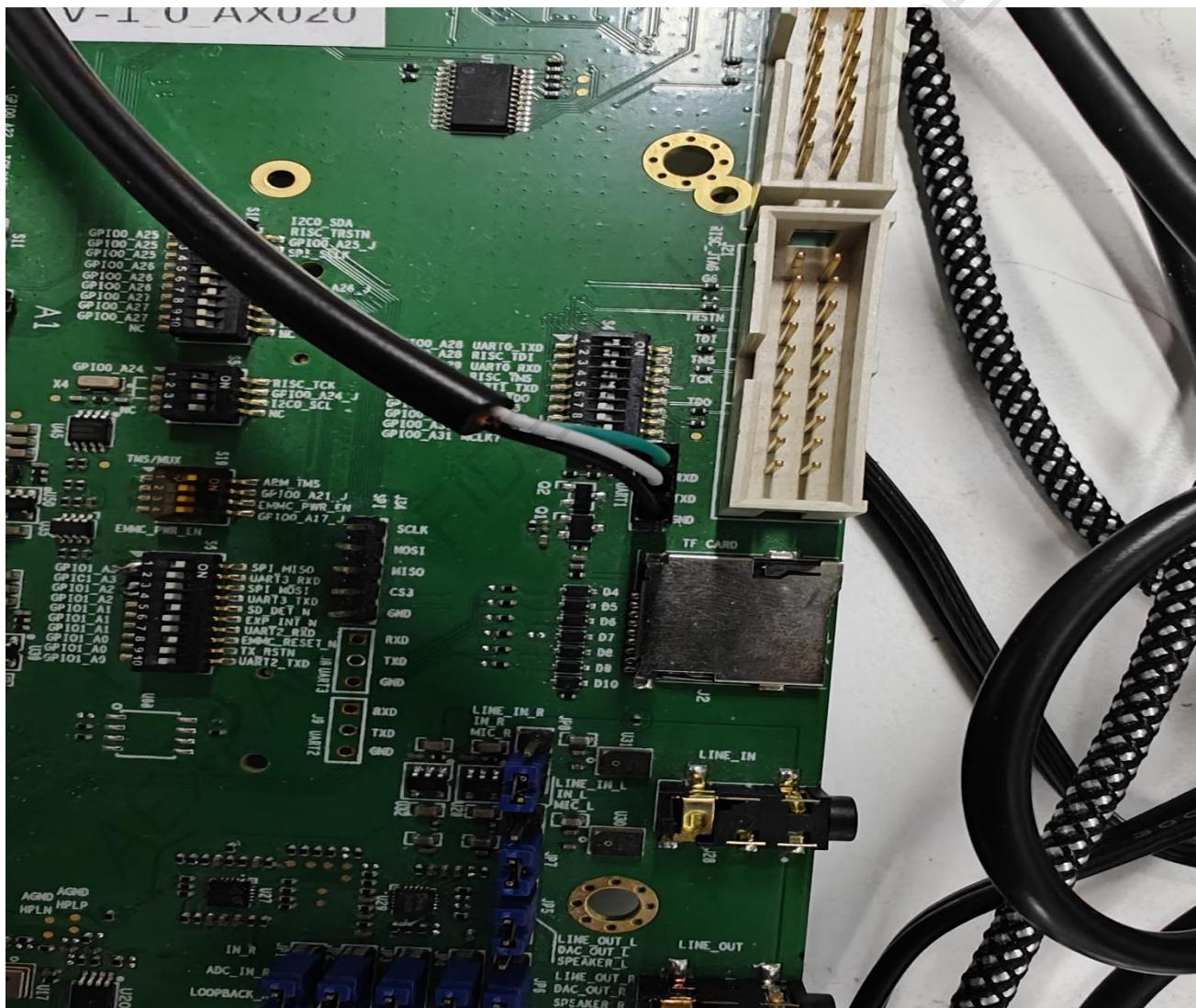
Riscv mem 在 ddr 上 reserved 一片内存，主要用于加载 riscv 镜像，和 riscv os mem，riscv 程序需要的内存都将从这片内存中分配。raw 检测功能开启时，由于检测流程中需要申请大量内存，所以 riscv mem size 配置不能小于 10M，否则可能导致内存不足等问题。

```
# Allocate running memory space for riscv.
ifeq ($(strip $(AX_FAST_RISCV_RAW_DET_SUPPORT)),TRUE)
RISCV_MEM_SIZE_MB := 10 #MB
else
RISCV_MEM_SIZE_MB := 3 #MB
endif
RISCV_MEM_SIZE_B := $(shell printf "0x%X" $$((($RISCV_MEM_SIZE_MB)*1048576)))
RISCV_DDR_END_ADDR := $(ISP_IMAGE_DDR_START)
RISCV_BIN_DDR_START := $(call SubAddressMB, $(RISCV_DDR_END_ADDR), $(RISCV_MEM_SIZE_MB))
RISCV_HEADER_DDR_START := $(call SubAddressB, $(RISCV_BIN_DDR_START), $(SEC_SIGN_HEADER_SIZE))
```

3 RAW 检测测试

■ 环境搭建

连接 riscv 串口线，串口线连接到 uart1 上，接好 sc850sl sensor，8n1，波特率 921600



■ 版本下载选择带 raw 字符 axp 包

- 版本下载好后, riscv 会去自动进行 raw 检测, 如果没有检测到人, 两侧系统会不断重启, 会不断打印重复 log

```
[VNOCNODC0RS] 435654[[qdemo]](qdemo) Build at Mar 26 2024 21:24:36
Cannot open /dev/stdin: No such file or directory
[ ] 44924[[qdemo]] aisp-node-1
[ ] 44945[[qdemo]] sys and pool init done
[ ] 44952[[qdemo]] set Itp online vpp done
Loading usb ko ...
Loading net ko ...
[VNOCNODC0RS] 437315[[qdemo]](qdemo) Build at Mar 26 2024 21:24:36
Loading usb ko ◊VNOCNODC0RS| 436764[[qdemo]](qdemo) Build at Mar 26 2024 21:24:36
Loading net ko ◊VNOCNODC0RS| 436765[[qdemo]](qdemo) Build at Mar 26 2024 21:24:36
Loading usb ko ◊VNOCNODC0RS| 436811[[qdemo]](qdemo) Build at Mar 26 2024 21:24:36
Warning: Bad CRC
## Error: "qs_aisppnode" not defined
[ ] 45072[[qdemo]] aisp-node-1
[ ] 45094[[qdemo]] sys and pool init done
[ ] 46041[[qdemo]] set Itp online vpp done
Loading usb ko ...
[VNOCNODC0RS] 435678[[qdemo]](qdemo) Build at Mar 26 2024 21:24:36
Loading net ko ...
[VNOCNODC0RS] 436961[[qdemo]](qdemo) Build at Mar 26 2024 21:24:36
Warning: Bad CRC
## Error: "qs_aisppnode" not defined
[ ] 45260[[qdemo]] aisp-node-1
[ ] 45191[[qdemo]] sys and pool init done
[ ] 45295[[qdemo]] set Itp online vpp done
Loading usb ko ...
[VNOCNODC0RS] 435760[[qdemo]](qdemo) Build at Mar 26 2024 21:24:36
Loading net ko ...
[VNOCNODC0RS] 436951[[qdemo]](qdemo) Build at Mar 26 2024 21:24:36
Cannot open /dev/stdin: No such file or directory
[ ] 44926[[qdemo]] aisp-node-1
[ ] 44945[[qdemo]] sys and pool init done
[ ] 44952[[qdemo]] set Itp online vpp done
Loading usb ko ...
[VNOCNODC0RS] 435922[[qdemo]](qdemo) Build at Mar 26 2024 21:24:36
Loading net ko ...
[VNOCNODC0RS] 435798[[qdemo]](qdemo) Build at Mar 26 2024 21:24:36
Loading usb ko ◊VNOCNODC0RS| 437487[[qdemo]](qdemo) Build at Mar 26 2024 21:24:36
Loading net ko ◊VNOCNODC0RS| 436805[[qdemo]](qdemo) Build at Mar 26 2024 21:24:36
Warning: Bad CRC
## Error: "qs_aisppnode" not defined
[ ] 45072[[qdemo]] aisp-node-1
[ ] 45170[[qdemo]] sys and pool init done
[ ] 45295[[qdemo]] set Itp online vpp done
Loading usb ko ...
Loading net ko ...
[VNOCNODC0RS] 437792[[qdemo]](qdemo) Build at Mar 26 2024 21:24:36
Warning: Bad CRC
## Error: "qs_aisppnode" not defined
[ ] 43953[[qdemo]](qdemo) Build at Mar 26 2024 21:24:36
[VNOCNODC0RS] 439401[[qdemo]](qdemo) Build at Mar 26 2024 21:24:36
Loading usb ko ◊VNOCNODC0RS| 437430[[qdemo]](qdemo) Build at Mar 26 2024 21:24:36
Loading usb ko ◊VNOCNODC0RS|
```

- riscv 检测到人，系统继续运行，串口不会再重复打 log，系统运行完成，riscv 进入休眠

```
639828][qspidn] rgn init done
640155][qspidn] pipe[0] AX_ISP_Create --
640328][qspidn] pipe[0] AX_ISP_Open ++
/etc/init.d/network: line 27: ethtool: not found
/etc/init.d/network: line 28: ethtool: not found
load net to done!!
# [ 723029][qspidn] pipe[0] AX_ISP_Open --
# [ 723030][qspidn] pipe[0] AX_ISP_Start ++
739430][qspidn] pipe[0] AX_ISP ++
739575][qspidn] pipe[0] AX_ISP_StreamOn done
782291][qspidn] Init latency: total:288192, sps:31416, npu:75755, isp:29329, cam open:161692, detect:92854
767653][qspidn] Detect[0] 1st frame sepo: 6.
811538][qspidn] Detect[0]: 1st frame got detected result
818897][qspidn] main launch timestamp: 440323
818949][qspidn] round[1] latency result: 1st Raw Image: 246558, 1st Detect Result: 813457, 1st Venc Frame: 818899, main laun
ch: 440323,
[ 818944][qspidn] round[1] latency from boot: 796888,
[ 819004][qspidn] round[1] vin latency sm[0]: started: 107516, opened: 165428, 1st raw ready: 246558, out next: 769907
[ 819055][qspidn] Venc[0]: 1064 1st sepo: 6.
[ 820545][qspidn] Jenc[2]: 1st raw is saved: /tmp/jenc.jpg, sepo: 6, pts: 339885, size: 40444
[ 4398187][qspidn] Venc[0]: 1064 is saved: /tmp/venc0.264, size: 446297 bytes

# #
```

- 串口不在重复打印 log 后，查看检测结果是否检测到人，在 arm 侧使用 cat /proc/ax_proc/riscv/log_dump 查看 riscv log, Detected {x} objects 为检测到的人象个数。

```
/ # cat /proc/ax_proc/riscv/log_dump
riscv log dump version 1.0.0
riscv log addr 0x448fe000 size 0x1000 offset 0 len 1535 cnt 1535

\ | /
- RT -      Thread Operating System
/ | \  4.1.0 build Mar 26 2024 21:24:58
2006 - 2022 Copyright by RT-Thread team
mbox_register_callback test2 !
[AX][ERR][ax_riscv_isp_camera_open][1839]: nSensorCnt 1 img_start_addr 0x46100000 ae_start_addr 0x486b6f00 share_buf_addr 0x486ff000
0 img_end_addr 0x48eff000 ae_total_size 147584

[AX][ERR][ax_riscv_isp_camera_open][1859]: pipe 0 img size over 4M, 1/2 binning init

__success__
master thread exit
[ALGO][Debug][do_raw_detct 113]: Raw_det thread tid: 45a8ae70.

[ALGO][Debug][entry_raw_detect 24]: Raw detect working...

[AX][INFO][main][22]: hello axera riscv
[AX][INFO][load_rootfs_thread][41]: loadrootfs start
[AX][ERR][process_3a_stat0][716]: pipe_id 0 seq_num ae:4 fsof:5 nAeStable 1

[ALGO][Debug][entry_raw_detect 35]: [Dylan] : height = 0x3c0
[ALGO][Debug][entry_raw_detect 36]: [Dylan] : width = 0x21c
[ALGO][Debug][entry_raw_detect 37]: [Dylan] : phyaddr = 0x46100000
[ALGO][Debug][entry_raw_detect 38]: [Dylan] : size = 0x7f800

[AX][INFO][load_rootfs_thread][51]: loadrootfs finished
[ALGO][Debug][entry_raw_detect 66]: Detected {1} objects:
[ALGO][Debug][entry raw detect 75]: [DET][Object] {55.3%} -> { 0, 179, 63, 515}.
mbox_receive_message2 stop ae0 2
mbox_receive_message3 get stat attr0, addr = 0x486b6ea8, wb gain 479 256 256 489
mbox_receive_message0 rtt_switch 0 0 addr0 486ff000 addr1 486ff650
[AX][ERR][fs0f_feof_irq_handler][594]: fs0f mbox_send_message share_info 486ff000 pipe_id 0 index 13 cnt 7

[AX][INFO][lowpower_thread][70]: entry WFI
/ #
```