RKNN3 C API 参考手册

文件标识：RK-YH-YF-425

发布版本：V1.0.0

日期：2025-12-22

文件密级：□绝密 □秘密 □内部资料 ■公开

免责声明

本文档按"现状"提供，瑞芯微电子股份有限公司（"本公司"，下同）不对本文档的任何陈述、信息和内容的准确性、可靠性、完整性、适销性、特定目的性和非侵权性提供任何明示或暗示的声明或保证。本文档仅作为使用指导的参考。

由于产品版本升级或其他原因，本文档将可能在未经任何通知的情况下，不定期进行更新或修改。

商标声明

"Rockchip"、"瑞芯微"、"瑞芯"均为本公司的注册商标，归本公司所有。

本文档可能提及的其他所有注册商标或商标，由其各自拥有者所有。

瑞芯微电子股份有限公司

Rockchip Electronics Co., Ltd.

地址： 福建省福州市铜盘路软件园A区18号

网址： www.rock-chips.com

客户服务电话： +86-4007-700-590

客户服务传真： +86-591-83951833

客户服务邮箱： fae@rock-chips.com

前言

概述

本文是Rockchip RKNN3 C API 参考手册。

读者对象

本文档（本指南）主要适用于以下工程师：

技术支持工程师

软件开发工程师

修订记录

版本	修改人	修改日期	修改说明	核定人
V0.1.0	HPC团队	2025-05-12	初始版本	熊伟
V0.2.0	HPC团队	2025-08-04	更新API接口说明	熊伟
V0.3.0b0	HPC团队	2025-09-10	新增rknn3_set_internal_mem接口描述；增加KV缓存存储方式与数据类型定义；完善LLM配置结构体内容	熊伟
V0.4.0b0	HPC团队	2025-11-03	更新rknn3_vocab_info和rknn3_llm_multimodal_tensor结构体和相关接口描述	熊伟
V0.5.0	HPC团队	2025-11-29	增加后处理相关查询命令与自定义算子插件机制;调整输出回调函数接口;增加YUV输入相关结构体和接口描述	熊伟
V1.0.0	HPC团队	2025-12-22	增加性能和内存分析接口	熊伟

目录

1. 概述

RKNN3 C API是RKNN3 Runtime（运行时库）的C语言接口。开发者使用C/C++开发应用程序，通过RKNN3 C API部署模型推理。本文对RKNN3 C API的各个函数、数据结构以及状态码等定义进行说明。

2. 硬件平台

本文档适用如下硬件平台：

- RK1820
- RK1828

3. RKNN3 API编译说明

在RKNN3 SDK中，应用程序的编译和运行控制在主控SoC完成，协处理器仅作为NPU加速单元。主控SoC通过PCIe/USB高速接口连接协处理器，例如RK3588连接到RK1820/RK1828。

开发者在编译RKNN3应用时需要包含rknn3_api.h头文件并链接librknn3_api.so运行时库。rknn3_api.h定义了用于通用深度学习模型部署的基础推理接口（如rknn3_init、rknn3_run等）以及专门为大型语言模型优化设计的session系列接口（如rknn3_session_init、rknn3_session_run等），这些接口提供了KV缓存管理、LoRA支持、采样策略配置等功能。开发者需要根据目标主控的硬件平台和系统类型选择相应的交叉编译工具链，并确保正确链接对应平台的RKNN3 API库。

主控SoC的交叉编译器的下载和安装方法可参考<`Rockchip_RK182X_Quick_Start_RKNN3_SDK`>文档。

4. RKNN3 C API调用流程

4.1 非大语言模型推理流程

CNN/ViT等非大语言模型使用基础API的推理流程主要包括以下几个步骤。首先，用户需调用rknn3_init接口初始化运行时上下文，并通过rknn3_load_model_from_path或rknn3_load_model_from_data加载模型。模型加载完成后，需调用rknn3_model_init进行模型初始化配置。随后，用户可通过rknn3_query接口获取输入输出tensor的属性信息，并为输入输出分配内存。推理时，调用rknn3_run接口完成同步推理。最后，使用rknn3_destroy释放上下文资源，基础API的推理流程如图4-1所示。

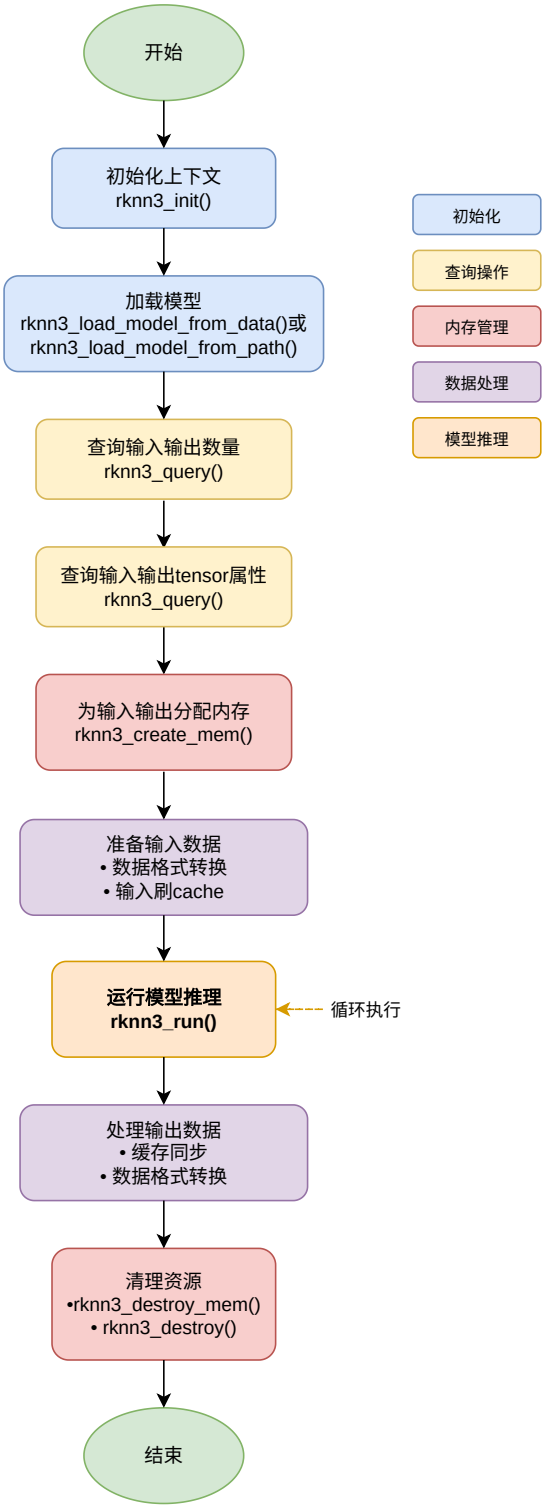

图4-1 基础API的推理流程

4.2 大语言模型推理流程

RKNN3大语言模型推理流程通常包括以下主要步骤。首先，用户需通过rknn3_find_devices获取可用设备列表，并调用rknn3_init初始化上下文，随后使用rknn3_load_model_from_path加载模型文件，并通过rknn3_model_init完成模型初始化配置。接下来，用户需准备分词器和embedding等相关资源，并设置rknn3_llm_param结构体以配置会话参数。之后，调用rknn3_session_init初始化会话，并通过rknn3_session_set_callback设置推理过程中的回调函数。推理前可根据需要设置聊天模板或清理KV缓存。

在推理阶段，用户构造rknn3_llm_input输入结构体，调用rknn3_session_run接口进行推理，推理结果通过回调函数异步获取。推理完成后，可根据需要统计性能数据或进行多轮对话。最后，需调用rknn3_session_destroy和rknn3_destroy接口释放会话和上下文资源，完成整个推理流程。大语言模型推理流程如图4-2所示。

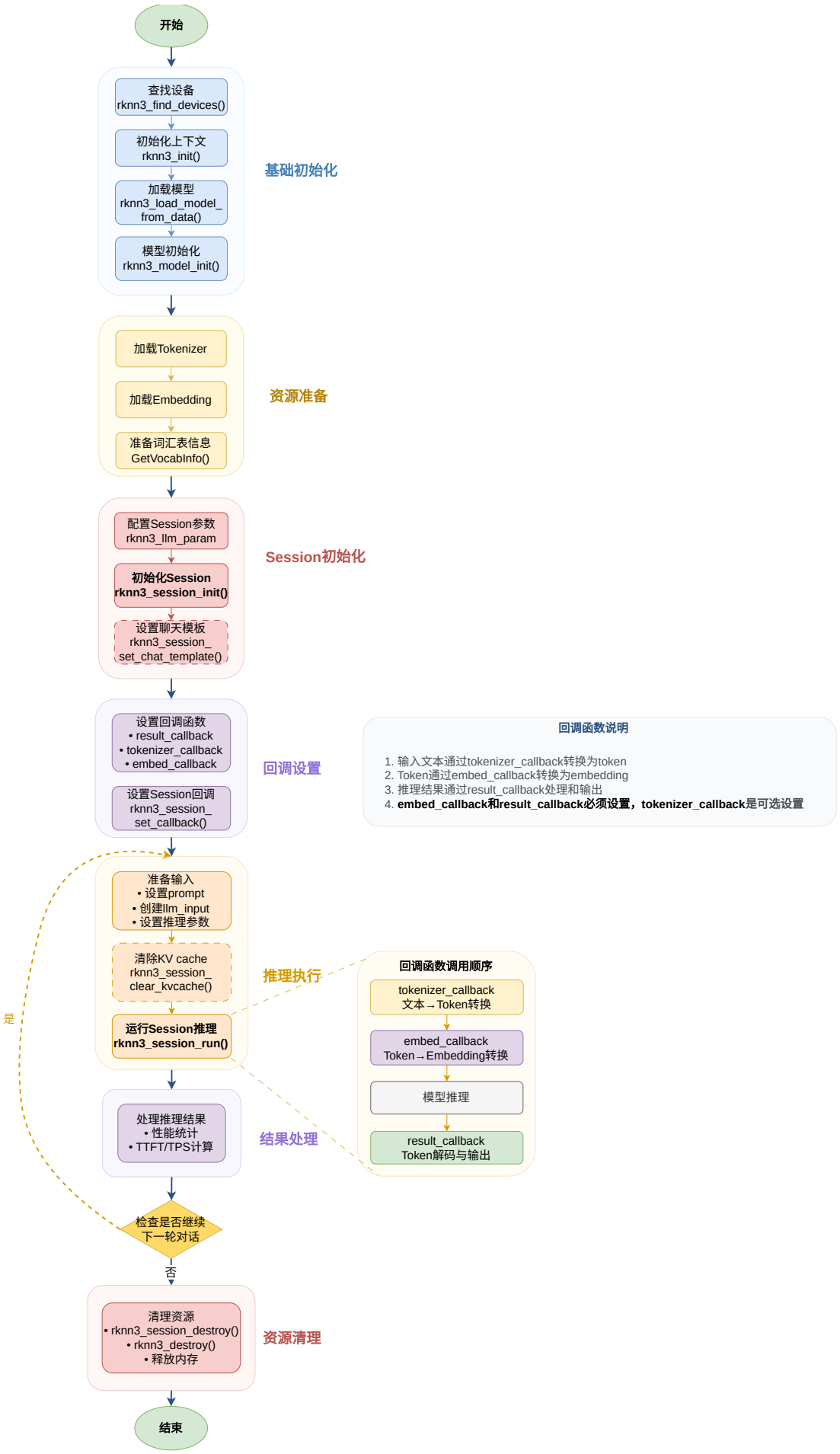

图4-2 大语言模型推理流程

5. RKNN3 C API说明

5.1. API 函数

5.1.1. rknn3_get_type_string

获取 tensor 类型的字符串表示。

API	rknn3_get_type_string
功能	获取 tensor 类型的字符串表示。
参数	rknn3_tensor_type type： tensor 的类型。
返回值	const char* 类型字符串

5.1.2. rknn3_get_qnt_type_string

获取量化类型的字符串表示。

API	rknn3_get_qnt_type_string
功能	获取量化类型的字符串表示。
参数	rknn3_tensor_qnt_type type： tensor 的量化类型。
返回值	const char* 量化类型字符串

5.1.3. rknn3_get_layout_string

获取布局类型的字符串表示。

API	rknn3_get_layout_string
功能	获取布局类型的字符串表示。
参数	rknn3_tensor_layout layout： tensor 的布局类型。
返回值	const char* 布局类型字符串

5.1.4. rknn3_init

该接口用于初始化RKNN3运行时上下文，必须在加载和运行模型前调用。context为输出参数，init_extend可用于指定设备ID等扩展信息。初始化完成后，需在不再使用时调用rknn3_destroy释放资源。多次初始化需确保资源正确释放，避免内存泄漏。

API	rknn3_init
功能	初始化一个新的RKNN3运行时上下文，该上下文用于在Rockchip NPU硬件上运行RKNN3模型。
参数	rknn3_context* context：指向将被初始化的RKNN3上下文句柄的指针
	rknn3_init_extend* init_extend：指向设备特定初始化信息的指针
返回值	int 状态码：见 5.2. 状态码
注意	使用完成后，请调用 rknn3_destroy(context) 释放上下文资源

5.1.5. rknn3_load_model_from_path

该接口用于从指定的模型文件路径和权重文件路径加载RKNN3模型到已初始化的上下文中。model_path和weight_path分别为模型和权重文件的路径，weight_path可为NULL。加载前需确保context已初始化，路径有效。加载失败时返回错误码，成功后可进行模型初始化和推理。

API	rknn3_load_model_from_path
功能	从文件路径加载RKNN3模型到指定的上下文中。
参数	rknn3_context context：RKNN3上下文句柄
	const char* model_path：RKNN3模型文件的路径
	const char* weight_path：RKNN3权重文件的路径
返回值	int 状态码：见 5.2. 状态码

5.1.6. rknn3_load_model_from_data

该接口用于从内存数据加载RKNN3模型，适用于模型和权重已在内存中的场景。model_data和weight_data分别为模型和权重的内存指针，model_size和weight_size为对应数据大小。调用前需确保context已初始化，数据指针和大小有效。加载失败返回错误码，成功后可进行模型初始化和推理。

API	rknn3_load_model_from_data
功能	从内存数据加载RKNN3模型。
参数	rknn3_context context：RKNN3上下文句柄
	const void* model_data：指向内存中模型数据的指针
	uint64_t model_size：模型数据的大小（字节）
	const void* weight_data：指向内存中权重数据的指针
	uint64_t weight_size：权重数据的大小（字节）
返回值	int 状态码：见 5.2. 状态码

5.1.7. rknn3_model_init

该接口用于初始化已加载模型的运行配置，包括优先级、超时时间、核心掩码等。config为配置参数指针。调用前需确保模型已加载，config参数有效。初始化失败返回错误码，成功后模型可用于推理。

API	rknn3_model_init
功能	初始化RKNN3模型。
参数	rknn3_context context：RKNN3上下文句柄
	rknn3_config* config：模型配置参数
返回值	int 状态码：见 5.2. 状态码

5.1.8. rknn3_dup_context

该接口用于复制已存在的RKNN3上下文，便于多线程或多实例场景下的上下文复用。context_in为源上下文，context_out为目标上下文指针。调用前需确保源上下文有效，目标指针非空。复制后需分别管理和释放各自的上下文资源。

API	rknn3_dup_context
功能	复制现有的RKNN3上下文。
参数	rknn3_context context_in：要复制的源RKNN3上下文
	rknn3_context* context_out：接收复制的RKNN3上下文的指针
返回值	int 状态码：见 5.2. 状态码

5.1.9. rknn3_destroy

该接口用于销毁RKNN3运行时上下文并释放相关资源。调用后context句柄失效，不可再用于其他操作。多次销毁同一context会导致未定义行为，需确保每个context只销毁一次。

API	rknn3_destroy
功能	销毁RKNN3运行时上下文并释放资源。
参数	rknn3_context context：要销毁的RKNN3上下文句柄
返回值	int 状态码：见 5.2. 状态码

5.1.10. rknn3_query

该接口用于查询模型和运行时的各种信息，如输入输出属性、SDK版本等。cmd指定查询类型，详见 5.4.1. rknn3_query_cmd，info为结果缓冲区指针，size为缓冲区大小。调用前需确保context和info有效，size与查询类型对应结构体大小一致。

API	rknn3_query
功能	查询RKNN3信息或状态。
参数	rknn3_context context：RKNN3模型的上下文
	rknn3_query_cmd cmd：查询命令类型
	void* info：用于存储查询结果的缓冲区指针
	uint64_t size：info缓冲区的大小（字节）
返回值	int 状态码：见 5.2. 状态码
注意	此函数用于查询有关RKNN3模型和运行时的各种信息，如SDK版本、设备信息、模型信息等。返回的具体信息取决于指定的查询命令

5.1.11. rknn3_run

该接口用于执行同步推理，inputs和outputs分别为输入输出tensor数组，n_inputs和n_outputs为数量。调用前需确保context、inputs、outputs均已正确分配和配置。推理过程中会阻塞直到完成，返回值为状态码。输入输出tensor的内存需由用户提前分配。

API	rknn3_run
功能	执行RKNN3模型推理。
参数	rknn3_context context：从rknn3_init获得的RKNN3上下文句柄
	const rknn3_tensor inputs[]：包含输入数据的输入 tensor 数组
	uint32_t n_inputs：输入 tensor 的数量
	rknn3_tensor outputs[]：用于存储推理结果的输出 tensor 数组
	uint32_t n_outputs：输出 tensor 的数量
返回值	int 状态码：见 5.2. 状态码
注意	此函数使用指定的RKNN3模型执行推理。它通过inputs数组接收输入数据，并将结果写入outputs数组。在调用此函数之前，必须正确分配和配置输入和输出 tensor

5.1.12. rknn3_run_async

该接口用于异步执行推理，参数与rknn3_run一致。调用后立即返回，推理过程在后台进行。需结合rknn3_wait等待推理完成。异步推理适合需要并发或流水线处理的场景。

API	rknn3_run_async
功能	异步执行RKNN3模型推理。
参数	rknn3_context context：从rknn3_init获得的RKNN3上下文句柄
	const rknn3_tensor inputs[]：包含输入数据的输入 tensor 数组
	uint32_t n_inputs：输入 tensor 的数量
	rknn3_tensor outputs[]：用于存储推理结果的输出 tensor 数组
	uint32_t n_outputs：输出 tensor 的数量
返回值	int 状态码：见 5.2. 状态码
注意	此函数使用指定的RKNN3模型执行异步推理。它通过inputs数组接收输入数据，并将结果写入outputs数组。在调用此函数之前，必须正确分配和配置输入和输出 tensor

5.1.13. rknn3_wait

该接口用于阻塞等待异步推理完成。context为模型实例的上下文句柄。仅在调用了rknn3_run_async后需要调用。同步推理无需调用此接口。

API	rknn3_wait
功能	等待推理/执行完成。
参数	rknn3_context context：RKNN3模型实例的上下文句柄
返回值	int 状态码：见 5.2. 状态码
注意	此函数会阻塞，直到RKNN3设备上的推理或执行完成

5.1.14. rknn3_create_mem_from_phys

从物理地址创建 tensor 内存对象。调用前需确保context有效，相关参数（如地址、大小等）正确。分配的内存需在不再使用时通过rknn3_destroy_mem释放，避免内存泄漏。

API	rknn3_create_mem_from_phys
功能	从物理地址创建 tensor 内存对象。
参数	rknn3_context context：RKNN3上下文句柄
	uint64_t phys_addr：内存的物理地址
	void* virt_addr：内存的虚拟地址
	uint64_t size：内存的大小（字节）
返回值	rknn3_tensor_mem* 指向创建的 tensor 内存对象的指针，如果创建失败则为NULL
注意	此函数从提供的物理和虚拟地址创建 tensor 内存对象。内存必须预先分配，且物理/虚拟地址必须有效

5.1.15. rknn3_create_mem_from_fd

从文件描述符创建 tensor 内存对象。调用前需确保context有效，相关参数（如fd、大小等）正确。分配的内存需在不再使用时通过rknn3_destroy_mem释放，避免内存泄漏。

API	**rknn3_create_mem_from_fd**
功能	从文件描述符创建 tensor 内存对象。
参数	rknn3_context context：RKNN3上下文句柄
	int32_t fd：内存的文件描述符
	void* virt_addr：内存的虚拟地址
	uint64_t size：内存的大小（字节）
	uint64_t offset：从fd引用的内存开始的偏移量
返回值	rknn3_tensor_mem* 指向创建的 tensor 内存对象的指针，如果创建失败则为NULL

5.1.16. rknn3_create_mem

该接口用于为tensor分配或注册内存。调用前需确保context有效，输入和输出tensor的core_id设置成rknn3_query接口获取的rknn3_tensor_attr结构体的core_id成员。分配的内存需在不再使用时通过rknn3_destroy_mem释放，避免内存泄漏。

API	**rknn3_create_mem**
功能	为RKNN3 tensor 创建内存缓冲区。
参数	rknn3_context context：RKNN3上下文句柄
	uint64_t size：要分配的内存大小（字节）
	int32_t core_id：内存分配的目标NPU核心ID
	rknn3_mem_alloc_flags flags：控制分配行为的内存分配标志
返回值	rknn3_tensor_mem* 指向分配的 tensor 内存的指针，如果分配失败则为NULL
注意	此函数分配可用于RKNN3 tensor 操作的内存。内存在指定的核心上分配，具有给定的标志

5.1.17. rknn3_destroy_mem

该接口用于释放通过RKNN3 API分配或注册的tensor内存。调用前需确保context和mem有效。重复释放同一内存会导致未定义行为。

API	rknn3_destroy_mem
功能	销毁为RKNN3 tensor 分配的内存。
参数	rknn3_context context：RKNN3上下文句柄
	rknn3_tensor_mem* mem：指向要销毁的 tensor 内存结构的指针
返回值	int 状态码：见 5.2. 状态码

5.1.18. rknn3_mem_sync

该接口用于在CPU和设备间同步内存数据。mode指定同步方向，常用于推理前后确保数据一致性。调用前需确保context和mem有效，mode设置正确。未同步可能导致推理结果异常。

API	rknn3_mem_sync
功能	在CPU和设备之间同步内存数据。
参数	rknn3_context context：RKNN3模型的上下文
	rknn3_tensor_mem* mem： tensor 的内存句柄
	rknn3_mem_sync_mode mode：表示刷新CPU cache和DDR数据的模式。 **RKNN3_MEMORY_SYNC_TO_DEVICE**：表示CPU cache数据同步到DDR中，通常用于CPU写入内存后，NPU访问相同内存前使用该模式将cache中的数据写回DDR。 **RKNN3_MEMORY_SYNC_FROM_DEVICE**：表示DDR数据同步到CPU cache，通常用于NPU写入内存后，使用该模式让下次CPU访问相同内存时，cache数据无效，CPU从DDR重新读取数据。 **RKNN3_MEMORY_SYNC_BIDIRECTIONAL**：表示CPU cache数据同步到DDR同时令CPU重新从DDR读取数据。
返回值	int 状态码：见 5.2. 状态码

5.1.19. rknn3_set_shape

该接口用于为动态输入模型设置当前形状。shape_id为模型中预定义的形状ID。调用前需确保context有效，shape_id合法。仅动态输入模型需调用，静态模型无需设置。

API	rknn3_set_shape
功能	为动态输入设置模型形状。
参数	rknn3_context context：RKNN3模型的上下文句柄
	int32_t shape_id：要设置的形状的ID。
返回值	int 状态码：见 5.2. 状态码

5.1.20. rknn3_set_kvcache_mem

该接口用于为指定NPU核心设置KV cache内存。mem为KV cache内存数组，npu_core_indices为核心索引数组，n_core为核心数量。调用前需确保context、mem、npu_core_indices有效，n_core大于0。mem和npu_core_indices需一一对应。

API	rknn3_set_kvcache_mem
功能	设置KV cache内存。
参数	rknn3_context context：RKNN3模型的上下文句柄
	rknn3_tensor_mem* mem[]：KV cache内存相关信息
	int* npu_core_indices：NPU核心索引
	int n_core：要设置kvcache的NPU核心数
返回值	int 状态码：见 5.2. 状态码

5.1.21. rknn3_set_internal_mem

为多个核心设置用户分配的内部Tensor内存。每个mem的core_id字段指定目标核心。

API	rknn3_set_internal_mem
功能	设置多个核心的用户分配内部Tensor内存
参数	rknn3_context context：RKNN3上下文
	rknn3_tensor_mem* mem[]：用户分配的内存对象数组
	uint32_t n_core：核心数量
返回值	int 状态码：见 5.2. 状态码

5.1.22. rknn3_dump_features

按层执行模型并将每层输出的Tensor以npy格式dump到指定目录。若未提供输出Tensor会自动分配。

API	rknn3_dump_features
功能	按层运行模型并保存中间Tensor数据
参数	rknn3_context context：RKNN3上下文
	const rknn3_tensor inputs[]：输入Tensor数组
	uint32_t n_inputs：输入Tensor数量
	rknn3_tensor outputs[]：可选的输出Tensor数组，为NULL或数量为0时由内部自动分配
	uint32_t n_outputs：输出Tensor数量，可设为0以启用内部输出分配
	const char* dump_dir：保存npy文件的目录路径
返回值	int 状态码：见 5.2. 状态码
注意	dump目录需可写；按层dump使用的核心来自 `rknn3_model_init` 设置的 run_core_mask

5.1.23. rknn3_find_devices

该接口用于获取当前系统可用的RKNN3设备列表。pdevs为设备信息结构体指针。调用前需确保pdevs有效。返回0表示无设备或未实现。

API	rknn3_find_devices
功能	获取可用的RKNN3设备列表。
参数	rknn3_devices* pdevs：将接收设备列表的结构的指针
返回值	int 状态码：见 5.2. 状态码
注意	此函数用系统上所有可用的RKNN3设备的信息填充提供的rknn3_devices结构

5.1.24. rknn3_session_init

该接口用于初始化新的RKNN3会话，支持LLM等高级功能。context为上下文，params为会话参数，n_params为参数数量。调用前需确保context和params有效。初始化成功返回会话句柄，失败返回NULL。会话需通过rknn3_session_destroy释放。

API	rknn3_session_init
功能	使用指定参数初始化新的RKNN3会话。
参数	rknn3_context context：用于会话的RKNN3上下文指针
	rknn3_llm_param* params：包含会话配置参数的rknn3_llm_param结构指针
	int n_params：参数数量
返回值	rknn3_session* 如果成功返回会话句柄，如果初始化失败则返回NULL

5.1.25. rknn3_session_enable_lora

为RKNN3会话启用LoRA。调用前需确保session和lora有效。

API	rknn3_session_enable_lora
功能	为RKNN3会话启用LoRA。
参数	rknn3_session* session：RKNN3会话指针
	rknn3_lora* lora：要启用的LoRA指针
返回值	int 状态码：见 5.2. 状态码

5.1.26. rknn3_session_disable_lora

为RKNN3会话禁用LoRA。调用前需确保session和lora有效。

API	rknn3_session_disable_lora
功能	为RKNN3会话禁用LoRA。
参数	rknn3_session* session：RKNN3会话指针
	rknn3_lora* lora：要禁用的LoRA指针
返回值	int 状态码：见 5.2. 状态码

5.1.27. rknn3_session_query_lora

从RKNN3会话查询LoRA信息。

API	rknn3_session_query_lora
功能	从RKNN3会话查询LoRA信息。
参数	rknn3_session* session：RKNN3会话指针
	rknn3_lora** lora：用于存储LoRA信息数组的二级指针
	int* n_lora：用于存储LoRA条目数量的指针
返回值	int 状态码：见 5.2. 状态码

5.1.28. rknn3_session_set_kvcache_policy

设置RKNN3会话的KV cache策略。适用于Transformer等需要缓存机制的模型。调用前需确保session有效，相关参数正确。KV cache操作需与推理流程配合使用，避免缓存不一致。

API	rknn3_session_set_kvcache_policy
功能	设置RKNN3会话的KV cache策略。
参数	rknn3_session* session：RKNN3会话句柄
	rknn3_kvcache_policy policy：要设置的KV cache策略
	rknn3_kvcache_policy_param* param：指定KV cache策略的参数
返回值	int 状态码：见 5.2. 状态码

5.1.29. rknn3_session_clear_kvcache

根据指定策略清除RKNN3会话的KV cache。

API	rknn3_session_clear_kvcache
功能	根据指定策略清除RKNN3会话的KV cache。
参数	rknn3_session* session：RKNN3会话句柄指针
	rknn3_kvcache_clear_policy clear：清除KV cache的策略，定义如何清除缓存
返回值	int 状态码：见 5.2. 状态码

5.1.30. rknn3_session_load_kvcache

从指定路径加载KV cache。

API	rknn3_session_load_kvcache
功能	从指定路径加载KV cache。
参数	rknn3_session* session：RKNN3会话指针
	const char* kvcache_path：KV cache文件的路径
	int64_t size：KV cache文件路径的长度，即 strlen(kvcache_path)
返回值	int 状态码：见 5.2. 状态码

5.1.31. rknn3_session_save_kvcache

将KV cache保存到指定路径。

API	rknn3_session_save_kvcache
功能	将KV cache保存到指定路径。
参数	rknn3_session* session：RKNN3会话指针
	char* kvcache_path：保存KV cache的路径
返回值	int 状态码：见 5.2. 状态码

5.1.32. rknn3_session_query_state

该接口用于查询当前会话的运行状态，包括已生成token数、KV策略等。session为会话指针，state为状态结构体指针。调用前需确保session和state有效。可用于监控推理进度和会话状态。

API	rknn3_session_query_state
功能	查询RKNN3会话的当前状态。
参数	rknn3_session* session：要查询的RKNN3会话指针
	RKLLMRunState* state：用于存储查询运行状态的指针
返回值	int 状态码：见 5.2. 状态码

5.1.33. rknn3_session_set_chat_template

该接口用于设置LLM的聊天模板，包括系统提示、前缀和后缀。session为会话句柄，system_prompt、prompt_prefix、prompt_postfix为模板内容。调用前需确保session有效，字符串内容符合模型要求。自定义模板有助于调整模型行为和对话风格。

API	rknn3_session_set_chat_template
功能	设置LLM的聊天模板，包括系统提示、前缀和后缀。
参数	rknn3_session* session：RKNN3会话句柄
	const char* system_prompt：定义语言模型上下文或行为的系统提示
	const char* prompt_prefix：聊天中用户输入前添加的前缀
	const char* prompt_postfix：聊天中用户输入后添加的后缀
返回值	int 状态码：见 5.2. 状态码

5.1.34. rknn3_session_set_callback

该接口用于为会话设置回调函数，支持自定义结果处理、采样、分词、嵌入、输出tensor回调等。session为会话指针，callback为回调结构体指针。调用前需确保session和callback有效。回调函数在推理过程中被触发，便于用户自定义推理流程。

API	rknn3_session_set_callback
功能	为RKNN3会话设置回调函数。
参数	rknn3_session* session：RKNN3会话实例指针
	RKLLMCallback* callback：包含回调函数的RKLLMCallback结构指针
返回值	int 状态码：见 5.2. 状态码

5.1.35. rknn3_session_run

使用提供的输入和参数运行推理。调用前需确保session、inputs、param有效。

API	rknn3_session_run
功能	使用提供的输入和参数运行推理。
参数	rknn3_session* session：RKNN3会话句柄指针
	rknn3_llm_input inputs[]：包含输入数据的输入 tensor 数组
	uint32_t n_inputs：提供的输入 tensor 数量
	rknn3_llm_infer_param* param：推理参数配置指针

API	rknn3_session_run
返回值	int 状态码：见 5.2. 状态码

5.1.36. rknn3_session_run_async

为大型语言模型会话异步运行推理。调用前需确保session、inputs、param有效。异步推理需结合回调或等待机制获取结果。

API	rknn3_session_run_async
功能	为大型语言模型会话异步运行推理。
参数	rknn3_session* session：RKNN3会话句柄指针
	rknn3_llm_input inputs[]：模型的输入 tensor 数组
	uint32_t n_inputs：输入 tensor 数量
	rknn3_llm_infer_param* param：推理参数配置指针
返回值	int 状态码：见 5.2. 状态码
注意	此函数对给定的LLM会话执行异步推理。它允许使用提供的输入和参数对模型进行非阻塞执行

5.1.37. rknn3_session_stop

终止当前会话推理。调用后仅终止当前推理，后续可继续发起新推理。适用于需要中断长时间推理的场景。

API	rknn3_session_stop
功能	终止当前会话推理。
参数	rknn3_session* session：RKNN3会话指针
返回值	int 状态码：见 5.2. 状态码

5.1.38. rknn3_session_destroy

该接口用于销毁RKNN3会话并释放相关资源。session为会话指针。调用后session指针失效，不可再用于其他操作。需确保每个会话只销毁一次，避免资源泄漏或重复释放。

API	rknn3_session_destroy
功能	销毁RKNN3会话并释放相关资源。
参数	rknn3_session* session：要销毁的RKNN3会话指针
返回值	int 状态码：见 5.2. 状态码
注意	调用此函数后，会话指针变为无效，不应再使用

5.1.39. rknn3_session_set_function_tools

该接口用于为会话设置函数工具描述（function tools），便于模型在对话中调用工具。调用前需确保session和tools字符串有效。

API	rknn3_session_set_function_tools
功能	设置LLM会话的函数工具描述。
参数	rknn3_session* session：RKNN3会话句柄
	const char* tools：函数工具描述字符串
返回值	int 状态码：见 5.2. 状态码

5.1.40. rknn3_profile_ops

按层执行模型并打印算子性能信息，包括时间、Cycles和带宽。若未提供输出Tensor会自动分配。

API	rknn3_profile_ops
功能	按层运行模型并打印算子性能信息。
参数	rknn3_context context：RKNN3上下文句柄
	const rknn3_tensor inputs[]：输入Tensor数组
	uint32_t n_inputs：输入Tensor数量
	rknn3_tensor outputs[]：输出Tensor数组，可为NULL
	uint32_t n_outputs：输出Tensor数量，可为0以启用内部输出分配
	uint32_t log_level：日志级别（0：仅汇总；1：算子明细+汇总；2：算子明细(含时间/周期/带宽)+汇总）
返回值	int 状态码：见 5.2. 状态码
注意	按层profiling使用的核心来自 `rknn3_model_init` 设置的 run_core_mask

5.1.41. rknn3_profile_mem

该接口用于打印各NPU核心的内存使用信息，需在 rknn3_model_init 之后调用。

API	rknn3_profile_mem
功能	打印各NPU核心的内存使用信息。
参数	rknn3_context context：RKNN3上下文句柄
返回值	int 状态码：见 5.2. 状态码

API	rknn3_profile_mem
注意	建议在 rknn3_model_init 之后调用以确保内存已分配

5.1.42. rknn3_register_custom_ops_plugins

注册自定义算子插件（目前支持后处理插件）。该接口加载插件并将其注册到指定上下文。

API	rknn3_register_custom_ops_plugins
功能	注册自定义算子插件。
参数	rknn3_context ctx：RKNN3上下文句柄
	const char* plugin_path：插件共享库路径
	int64_t size：保留字段，可置0
返回值	int 状态码：见 5.2. 状态码

5.1.43. rknn3_register_custom_op_func

自定义算子注册函数指针类型，用于在插件库内注册自定义算子。

函数指针类型	rknn3_register_custom_op_func
功能	根据算子索引返回自定义算子描述结构体指针
参数	int op_index：算子索引（从0开始）
返回值	rknn3_custom_op*：对应的自定义算子描述结构体指针；若无该索引则返回NULL

5.1.44. LLMResultCallback

LLM结果回调函数类型。

函数指针类型	LLMResultCallback
功能	处理LLM生成结果的回调函数
参数	void* userdata：用户自定义数据指针 RKLLMResult* result：指向LLM结果的指针 LLMCallState state：LLM调用状态（如完成、错误等）
返回值	int：成功返回0，错误返回非零值

5.1.45. LLMSamplingCallback

LLM采样回调函数类型。

函数指针类型	LLMSamplingCallback
功能	处理采样logits的回调函数
参数	void* userdata：用户自定义数据指针 float16* logits：指向logits数组的指针 char* logits_name：logits的名称
返回值	int：返回所选token ID (>=0)表示成功，负值表示错误码，见 5.2. 状态码

5.1.46. LLMGetEmbedCallback

获取LLM嵌入向量的回调函数类型。

函数指针类型	LLMGetEmbedCallback
功能	获取LLM嵌入向量的回调函数
参数	void* userdata：用户自定义数据指针 int32_t* tokens：token ID数组 uint64_t num_tokens：tokens数组中的token数量 void* embed：存储嵌入输出的缓冲区指针 uint64_t len：嵌入缓冲区的长度（字节）
返回值	int 状态码：见 5.2. 状态码

5.1.47. LLMTokenizerCallback

LLM分词器回调函数类型。

函数指针类型	LLMTokenizerCallback
功能	处理文本分词的回调函数
参数	void* userdata：用户自定义数据指针 const char* text：输入文本字符串指针 int32_t text_len：文本长度 int32_t* tokens：token ID数组指针 int32_t n_tokens_max：最大生成token数量
返回值	int：返回生成的token数量 (>=0)表示成功，负值表示错误码，见 5.2. 状态码

5.1.48. LLMOutputCallback

获取输出tensor的回调函数类型。

函数指针类型	LLMOutputCallback
功能	获取输出tensor的回调函数
参数	void* userdata：用户自定义数据指针 rknn3_tensor* output_tensors：输出tensor数组指针 uint32_t n_output_tensors：输出tensor数量 LLMOutputCallbackState state：输出回调的状态
返回值	int：成功返回0，错误返回非零值，见 5.2. 状态码

5.1.49. rknn3_im_mem_create

该接口用于创建 RKNN3 图像内存对象 `rknn3_im_mem`，通常用于模型推理前后的图像预处理/后处理流程。

API	rknn3_im_mem_create
功能	在设备上分配一个 RKNN3 图像内存对象。
参数	rknn3_context context：RKNN3 上下文句柄
	int width：图像宽度（像素）
	int height：图像高度（像素）
	rknn3_im_fmt format：图像格式，见 5.4.15. rknn3_im_fmt
	int size：图像内存大小（字节），为0时自动按宽高与格式计算
	int core_id：分配目标NPU核心ID
	rknn3_mem_alloc_flags flags：内存分配标志

API	rknn3_im_mem_create
	rknn3_im_mem* im_mem：输出参数，接收创建好的 `rknn3_im_mem` 对象指针
返回值	int 状态码：见 5.2. 状态码
注意	创建成功后，需配合 `rknn3_im_mem_destroy` 接口释放该对象

5.1.50. rknn3_im_mem_destroy

该接口用于销毁通过 `rknn3_im_mem_create` 创建的 RKNN3 图像内存对象，并释放其占用的设备内存。

API	rknn3_im_mem_destroy
功能	销毁 RKNN3 图像内存对象并释放设备内存。
参数	rknn3_context context：RKNN3 上下文句柄
	rknn3_im_mem* im_mem：待销毁的 `rknn3_im_mem` 对象指针
返回值	int 状态码：见 5.2. 状态码
注意	调用后 `im_mem` 不可再被使用

5.1.51. rknn3_im_cvt_color

该接口用于在 RKNN3 图像内存中进行图像颜色格式转换，可在模型推理前将输入图像从一种 RGB/YUV 格式转换为另一种格式。

API	rknn3_im_cvt_color
功能	在 RKNN3 图像内存对象之间进行颜色空间转换。
参数	rknn3_context context：RKNN3 上下文句柄
	rknn3_im_mem* src：源图像内存对象
	rknn3_im_mem* dst：目标图像内存对象
返回值	int 状态码：见 5.2. 状态码
注意	`src` 和 `dst` 均应由 `rknn3_im_mem_create` 创建

5.2. 状态码

状态码	值	描述
RKNN3_SUCCESS	0	执行成功
RKNN3_ERR_FAIL	-1	执行失败
RKNN3_ERR_ARGUMENT_INVALID	-2	参数无效
RKNN3_ERR_MODEL_INVALID	-3	模型无效
RKNN3_ERR_CTX_INVALID	-4	上下文无效
RKNN3_ERR_RUN_TASK_FAILED	-5	任务运行失败
RKNN3_ERR_OUT_OF_MEMORY	-6	内存不足
RKNN3_ERR_TIMEOUT	-7	执行超时
RKNN3_ERR_INPUT_INVALID	-8	输入无效
RKNN3_ERR_OUTPUT_INVALID	-9	输出无效
RKNN3_ERR_DEVICE_UNAVAILABLE	-10	设备不可用
RKNN3_ERR_DEVICE_UNMATCH	-11	设备不匹配
RKNN3_ERR_TARGET_PLATFORM_UNMATCH	-12	目标平台不匹配
RKNN3_ERR_COMMUNICATION	-13	通信错误
RKNN3_ERR_MEM_SYNC_FAILED	-14	内存同步失败
RKNN3_WARN_NPU_CORE_UNUSED	-100	NPU核心未使用，仅作为警告，不影响模型执行

5.3. 常量

5.3.1. tensor 相关常量

常量名	值	描述
RKNN3_MAX_DIMS	16	tensor 的最大维度
RKNN3_MAX_STRIDE_DIMS	17	tensor 的最大步长维度
RKNN3_MAX_NAME_LEN	256	tensor 名称的最大长度
RKNN3_MAX_DYNAMIC_SHAPE_NUM	512	每个输入的最大动态形状数量
RKNN3_MAX_LORA_NUM	128	最大LoRA数量
RKNN3_MAX_SPECIAL_BOS_ID_NUM	64	特殊序列开始(BOS)Token的ID 的最大数量

常量名	值	描述
RKNN3_MAX_SPECIAL_EOS_ID_NUM	64	特殊序列结束(EOS)Token的ID 的最大数量

5.3.2. 设备相关常量

常量名	值	描述
RKNN3_MAX_DEVS	64	最大设备数量
RKNN3_MAX_DEV_LEN	64	设备ID/类型的最大长度
RKNN3_MAX_NPU_NODE_NUM	128	最大NPU节点数量

5.4. 枚举

5.4.1. rknn3_query_cmd

查询命令枚举，用于rknn3_query函数。

枚举值	描述
RKNN3_QUERY_IN_OUT_NUM	查询输入和输出 tensor 的数量
RKNN3_QUERY_INPUT_ATTR	查询输入 tensor 的属性
RKNN3_QUERY_OUTPUT_ATTR	查询输出 tensor 的属性
RKNN3_QUERY_SDK_VERSION	查询SDK和驱动版本
RKNN3_QUERY_CORE_MEM_SIZE	查询每个核心的权重和内部内存大小
RKNN3_QUERY_NATIVE_INPUT_ATTR	查询原生输入 tensor 的属性
RKNN3_QUERY_NATIVE_OUTPUT_ATTR	查询原生输出 tensor 的属性
RKNN3_QUERY_DEVICE_MEM_INFO	查询RKNN3内存信息的属性
RKNN3_QUERY_CORE_NUMBER	查询核心数量
RKNN3_QUERY_ALLOCATION_INFO	查询分配信息
RKNN3_QUERY_DYNAMIC_SHAPE_CONFIG	查询完整的动态形状配置
RKNN3_QUERY_DYNAMIC_SHAPE_INFO	查询所有支持的形状组合
RKNN3_QUERY_LLM_CONFIG	查询LLM特有的配置，比如 chat template
RKNN3_QUERY_POSTPROCESS_IN_OUT_NUM	查询后处理的输入输出数量，仅在启用后处理时有效
RKNN3_QUERY_POSTPROCESS_OUTPUT_ATTR	查询后处理输出tensor属性，仅在启用后处理时有效

枚举值	描述
RKNN3_QUERY_POSTPROCESS_DYNAMIC_SHAPE_INFO	查询后处理的动态形状信息，仅在启用后处理时有效
RKNN3_QUERY_CMD_MAX	查询命令枚举的最大值

5.4.2. rknn3_tensor_type

tensor 数据类型枚举。

枚举值	描述
RKNN3_TENSOR_FLOAT32	float32类型
RKNN3_TENSOR_FLOAT16	float16类型
RKNN3_TENSOR_INT8	int8类型
RKNN3_TENSOR_UINT8	uint8类型
RKNN3_TENSOR_INT16	int16类型
RKNN3_TENSOR_UINT16	uint16类型
RKNN3_TENSOR_INT32	int32类型
RKNN3_TENSOR_UINT32	uint32类型
RKNN3_TENSOR_INT64	int64类型
RKNN3_TENSOR_UINT64	uint64类型
RKNN3_TENSOR_BOOL	bool类型
RKNN3_TENSOR_INT4	int4类型
RKNN3_TENSOR_TYPE_MAX	tensor 数据类型枚举的最大值

5.4.3. rknn3_tensor_qnt_type

量化类型枚举。

枚举值	描述
RKNN3_TENSOR_QNT_NONE	无量化
RKNN3_TENSOR_PER_LAYER_SYMMETRIC	按层对称量化
RKNN3_TENSOR_PER_LAYER_ASYMMETRIC	按层非对称量化
RKNN3_TENSOR_PER_CHANNEL_SYMMETRIC	按通道对称量化
RKNN3_TENSOR_PER_CHANNEL_ASYMMETRIC	按通道非对称量化

枚举值	描述
RKNN3_TENSOR_PER_GROUP_SYMMETRIC	按组对称量化
RKNN3_TENSOR_PER_GROUP_ASYMMETRIC	按组非对称量化
RKNN3_TENSOR_QNT_MAX	量化类型枚举的最大值

5.4.4. rknn3_tensor_layout

tensor 数据布局枚举。

枚举值	描述
RKNN3_TENSOR_UNDEFINED	未定义
RKNN3_TENSOR_NCHW	数据布局为NCHW
RKNN3_TENSOR_NHWC	数据布局为NHWC
RKNN3_TENSOR_NC1HWC2	数据布局为NC1HWC2
RKNN3_TENSOR_CHWN	保留值，暂未使用
RKNN3_TENSOR_HWIO	保留值，暂未使用
RKNN3_TENSOR_OIHW	保留值，暂未使用
RKNN3_TENSOR_O1I1HWI2O2	保留值，暂未使用
RKNN3_TENSOR_LAYOUT_MAX	tensor 数据布局枚举的最大值

5.4.5. rknn3_mem_alloc_flags

创建RKNN3 tensor 内存的内存分配标志。

枚举值	描述
RKNN3_FLAG_MEMORY_FLAGS_DEFAULT	与RKNN3_FLAG_MEMORY_CACHEABLE相同
RKNN3_FLAG_MEMORY_CACHEABLE	创建cacheable内存
RKNN3_FLAG_MEMORY_NON_CACHEABLE	创建non-cacheable内存

5.4.6. rknn3_mem_sync_mode

rknn3_mem_sync函数的内存同步模式。

枚举值	描述
RKNN3_MEMORY_SYNC_TO_DEVICE	表示CPU cache数据同步到DDR中，通常用于CPU写入内存后，NPU访问相同内存前使用该模式将cache中的数据写回DDR
RKNN3_MEMORY_SYNC_FROM_DEVICE	表示DDR数据同步到CPU cache，通常用于NPU写入内存后，使用该模式让下次CPU访问相同内存时，cache数据无效，CPU从DDR重新读取数据
RKNN3_MEMORY_SYNC_BIDIRECTIONAL	表示CPU cache数据同步到DDR同时令CPU重新从DDR读取数据

5.4.7. rknn3_core_mask

在目标NPU核心上运行的模式。

枚举值	描述
RKNN3_NPU_CORE_AUTO	默认，表示自动调度模型，自动运行在当前空闲的NPU核上
RKNN3_NPU_CORE_0	在NPU核心0上运行
RKNN3_NPU_CORE_1	在NPU核心1上运行
RKNN3_NPU_CORE_2	在NPU核心2上运行
RKNN3_NPU_CORE_3	在NPU核心3上运行
RKNN3_NPU_CORE_4	在NPU核心4上运行
RKNN3_NPU_CORE_5	在NPU核心5上运行
RKNN3_NPU_CORE_6	在NPU核心6上运行
RKNN3_NPU_CORE_7	在NPU核心7上运行
RKNN3_NPU_CORE_ALL	表示工作在所有的NPU核心上

5.4.8. rknn3_kvcache_policy

KV缓存的策略。

枚举值	描述
RKNN3_KVCACHE_POLICY_DEFAULT	默认策略，等同于RKNN3_KVCACHE_POLICY_RECURRENT
RKNN3_KVCACHE_POLICY_RECURRENT	使用递归缓存策略
RKNN3_KVCACHE_POLICY_NORMAL	使用普通缓存策略，仅在max_context_len范围内使用KV缓存

5.4.9. rknn3_kvcache_clear_policy

清除KV缓存的策略。

枚举值	描述
RKNN3_KVCACHE_CLEAR_ALL	完全清除所有KV缓存条目
RKNN3_KVCACHE_KEEP_SYSTEM_PROMPT	清除KV缓存但保留系统提示

5.4.10. rknn3_llm_input_type

定义可以输入到LLM的输入类型。

枚举值	描述
RKNN3_LLM_INPUT_PROMPT	输入是文本提示
RKNN3_LLM_INPUT_TOKEN	输入是一系列token
RKNN3_LLM_INPUT_EMBED	输入是嵌入向量
RKNN3_LLM_INPUT_MULTIMODAL	多模态输入
RKNN3_LLM_INPUT_AUX	其他类型输入
RKNN3_LLM_INPUT_MAX	输入类型枚举的最大值

5.4.11. LLMCallState

描述LLM调用的可能状态。

枚举值	描述
RKLLM_RUN_NORMAL	LLM调用处于正常运行状态
RKLLM_RUN_WAITING	LLM调用正在等待完整的UTF-8编码字符
RKLLM_RUN_FINISH	LLM调用已完成执行
RKLLM_RUN_STOP	用户停止了LLM调用
RKLLM_RUN_MAX_NEW_TOKEN_REACHED	已达到最大新Token数
RKLLM_RUN_ERROR	LLM调用期间发生错误

5.4.12. rknn3_mem_type

内存类型枚举。

枚举值	描述
RKNN3_MEMORY_TYPE_NPU_DRAM	NPU DRAM内存
RKNN3_MEMORY_TYPE_EXT_DDR	外部DDR内存

5.4.13. rknn3_kvcache_dtype

KV缓存的数据类型。这些类型为内部定义数据类型，影响KV缓存大小和管理，用户无须解析。

枚举值	描述
RKNN3_KVCACHE_DTYPE_UNDEFINED	未定义
RKNN3_KVCACHE_DTYPE_INT4_TO_F16	INT4转FLOAT16
RKNN3_KVCACHE_DTYPE_INT4_TO_F8	INT4转FLOAT8
RKNN3_KVCACHE_DTYPE_INT8_TO_F16	INT8转FLOAT16
RKNN3_KVCACHE_DTYPE_FLOAT4_TO_F16	FLOAT4转FLOAT16
RKNN3_KVCACHE_DTYPE_FLOAT4_TO_F8	FLOAT4转FLOAT8
RKNN3_KVCACHE_DTYPE_FLOAT8_TO_F16	FLOAT8转FLOAT16
RKNN3_KVCACHE_DTYPE_FLOAT8_TO_F8	FLOAT8转FLOAT8
RKNN3_KVCACHE_DTYPE_FLOAT16	FLOAT16

5.4.14. rknn3_kvcache_store_method

KV缓存的存储方式。这些类型为内部定义数据类型，影响KV缓存大小和管理，用户无须解析。

枚举值	描述
RKNN3_KVCACHE_STORE_METHOD_UNDEFINED	未定义
RKNN3_KVCACHE_STORE_METHOD_NORMAL	普通存储
RKNN3_KVCACHE_STORE_METHOD_GROUP_QUANT	分组量化存储

5.4.15. rknn3_im_fmt

图像格式枚举，用于 RKNN3 图像相关接口。

枚举值	描述
RKNN3_IM_FMT_RGB888	RGB888 格式，常用于通用图像预处理
RKNN3_IM_FMT_BGR888	BGR888 格式，常用于 OpenCV 图像处理
RKNN3_IM_FMT_GRAY8	8 位灰度图格式
RKNN3_IM_FMT_YCbCr_420_SP	YCbCr 4:2:0 半平面格式，多用于视频编解码
RKNN3_IM_FMT_YCrCb_420_SP	YCrCb 4:2:0 半平面格式，多用于视频编解码
RKNN3_IM_FMT_YCbCr_422_SP	YCbCr 4:2:2 半平面格式，多用于视频编解码

枚举值	描述
RKNN3_IM_FMT_YCrCb_422_SP	YCrCb 4:2:2 半平面格式，多用于视频编解码
RKNN3_IM_FMT_UNKNOWN	未知或暂不支持的图像格式

5.4.16. rknn3_llm_task_type

LLM任务类型枚举。

枚举值	描述
RKNN3_LLM_TASK_GENERATE	生成任务
RKNN3_LLM_TASK_EMBEDDING	向量化/嵌入任务

5.4.17. LLMOutputCallbackState

输出回调阶段枚举。

枚举值	描述
RKLLM_OUTPUT_CALLBACK_PREFILL_PROCESSING	预填充阶段回调处理中
RKLLM_OUTPUT_CALLBACK_PREFILL_FINISHED	预填充阶段回调结束
RKLLM_OUTPUT_CALLBACK_DECODE_PROCESSING	解码阶段回调处理中
RKLLM_OUTPUT_CALLBACK_DECODE_FINISHED	解码阶段回调结束

5.4.18. rknn3_im_proc_flag

图像处理标志位枚举。

枚举值	描述
RKNN3_IM_PROC_FLAG_NONE	无处理
RKNN3_IM_PROC_FLAG_CROP	裁剪
RKNN3_IM_PROC_FLAG_RESIZE	缩放
RKNN3_IM_PROC_FLAG_FILL	填充
RKNN3_IM_PROC_FLAG_COLOR_SPACE_CONVERT	色彩空间转换
RKNN3_IM_PROC_FLAG_DECODE	解码
RKNN3_IM_PROC_FLAG_ENCODE	编码

5.4.19. rknn3_op_target_type

自定义算子后端目标枚举。

枚举值	描述
RKNN3_OP_TARGET_TYPE_CPU	运行在CPU

枚举值	描述
RKNN3_OP_TARGET_TYPE_MAX	枚举最大值

5.4.20. rknn3_op_plugin_type

自定义算子插件类型枚举。

枚举值	描述
RKNN3_OP_PLUGIN_TYPE_POSTPROCESS	后处理插件
RKNN3_OP_PLUGIN_TYPE_CUSTOM_OP	自定义算子插件（当前暂不支持）
RKNN3_OP_PLUGIN_TYPE_MAX	枚举最大值

5.5. 结构体

5.5.1. rknn3_input_output_num

RKNN3_QUERY_IN_OUT_NUM的信息结构体。

成员变量	数据类型	含义
n_input	uint32_t	输入的数量
n_output	uint32_t	输出的数量

5.5.2. rknn3_quant_info

量化信息结构体。

成员变量	数据类型	含义
scale	float	缩放数据
zero_point	int32_t	零点数据

5.5.3. rknn3_tensor_attr

存储RKNN3_QUERY_INPUT_ATTR/RKNN3_QUERY_OUTPUT_ATTR查询结果的结构体。

成员变量	数据类型	含义
index	uint32_t	输入参数：输入/输出 tensor 的索引，在调用rknn3_query前需要设置index的值
name	char[RKNN3_MAX_NAME_LEN]	tensor 的名称
n_dims	uint32_t	维度的数量

成员变量	数据类型	含义
shape	uint32_t[RKNN3_MAX_DIMS]	有效维度数组
aligned_size	uint64_t	按字节计算的对齐形状的 tensor 大小
n_stride	uint32_t	步长的数量
stride	uint64_t[RKNN3_MAX_STRIDE_DIMS]	tensor 的步长，例如，16x16 tensor 的步长为 [16*16, 16, 1]
n_elems	uint32_t	tensor 的元素数量
dtype	rknn3_tensor_type	tensor 的数据类型
layout	rknn3_tensor_layout	tensor 的数据布局
qnt_type	rknn3_tensor_qnt_type	tensor 的量化类型
qnt_info	rknn3_quant_info	tensor 的量化信息
core_id	int32_t	tensor 缓冲区的核心ID

5.5.4. rknn3_sdk_version

SDK版本信息结构体。

成员变量	数据类型	含义
api_version	char[64]	RKNN3 API的版本
drv_version	char[64]	RKNN3驱动的版本

5.5.5. rknn3_core_mem_size

每个核心的权重和内部内存大小信息结构体。

成员变量	数据类型	含义
core_id	int32_t	内存归属的物理NPU核心ID
weight_size	uint64_t	权重内存大小（字节）
internal_size	uint64_t	内部Tensor内存大小（字节）
reserved	uint8_t	保留字段

5.5.6. rknn3_custom_string

自定义字符串信息结构体。

成员变量	数据类型	含义
string	char[1024]	自定义字符串，最大长度为1024字节

5.5.7. rknn3_tensor_mem

tensor 内存信息结构体。

成员变量	数据类型	含义
virt_addr	void*	tensor 缓冲区的虚拟地址
phys_addr	uint64_t	tensor 缓冲区的物理地址
fd	int32_t	tensor 缓冲区的文件描述符
buffer_id	uint64_t	tensor 缓冲区的缓冲区ID，用于内存管理
offset	uint64_t	内存的偏移量
size	uint64_t	tensor 缓冲区的大小
flags	uint64_t	tensor 缓冲区的标志，保留字段
core_id	int32_t	NPU核心ID
priv_data	void*	tensor 缓冲区的私有数据
mem_type	rknn3_mem_type	tensor 缓冲区的内存类型

5.5.8. rknn3_config

模型加载的控制参数结构体。

成员变量	数据类型	含义
priority	int32_t	运行优先级
run_timeout	uint32_t	运行超时时间（毫秒），0表示无超时
run_core_mask	uint32_t	模型执行的核心掩码
user_mem_weight	uint8_t	权重内存是否由用户分配
user_mem_internal	uint8_t	内部内存是否由用户分配
user_sram	uint8_t	SRAM内存是否由用户分配
reserved	uint8_t	保留字段

5.5.9. rknn3_tensor

表示包含内存和属性信息的RKNN3 tensor 结构体。

该结构体包含了RKNN3操作中使用的 tensor 的内存信息和属性，是RKNN3运行时处理 tensor 的基本数据结构。

成员变量	数据类型	含义
mem	rknn3_tensor_mem*	tensor 的内存信息
attr	rknn3_tensor_attr*	tensor 的属性

5.5.10. rknn3_allocation_info

RKNN3模型的内存分配信息结构体。

该结构体提供了不同内存类型（命令、权重、内部、KV cache）及其在NPU核心上分布的详细内存分配信息。

成员变量	数据类型	含义
core_id	int32_t	NPU核心ID
command_mem	rknn3_tensor_mem	命令内存的信息
weight_mem	rknn3_tensor_mem	权重内存的信息
internal_mem	rknn3_tensor_mem	内部内存的信息
kvcache_mem	rknn3_tensor_mem	KV cache内存的信息
reserved	uint8_t	保留字段

5.5.11. rknn3_shape_info

RKNN3模型 tensor 形状信息结构体。

该结构体包含了RKNN3模型输入和输出 tensor 的形状的信息，包括 tensor 属性和形状配置详情。

成员变量	数据类型	含义
shape_id	int32_t	此形状组合的唯一ID
n_inputs	uint32_t	输入 tensor 的数量
input_attrs	rknn3_tensor_attr*	输入 tensor 属性数组
n_outputs	uint32_t	输出 tensor 的数量
output_attrs	rknn3_tensor_attr*	输出 tensor 属性数组
is_default	uint8_t	是否为默认形状
reserved	uint8_t	保留字段

5.5.12. rknn3_shape_config

动态形状设置的配置结构体。

该结构体包含了RKNN3模型中动态形状推理的形状组合和当前活动形状的信息。

成员变量	数据类型	含义
n_shapes	uint32_t	可用的形状组合数量
current_shape_id	int32_t	当前活动形状配置的ID，值为-1表示没有活动形状

5.5.13. rknn3_llm_config

LLM配置结构体。

该结构体包含了初始化和运行LLM所需的基础配置。

成员变量	数据类型	含义
chat_template	char*	聊天模板字符串
vocab_size	uint32_t	词表大小
embedding_dim	uint32_t	嵌入维度（通常等于hidden size）
max_ctx_len	uint32_t	最大上下文长度
max_position_embeddings	uint32_t	最大位置编码长度
kvcache_store_method	rknn3_kvcache_store_method	KV缓存存储方式
kvcache_dtype	rknn3_kvcache_dtype	KV缓存数据类型
kvcache_group_size	uint32_t	KV缓存分组量化的组大小
kvcache_residual_depth	uint32_t	KV缓存剩余深度
model_type	char*	模型类型字符串
task_type	rknn3_llm_task_type	LLM任务类型
reserved	uint8_t	保留字段

5.5.14. rknn3_init_extend

设备特定初始化信息结构体。

该结构体用于在RKNN3运行时上下文初始化期间指定设备特定参数，包括设备ID和为将来使用保留的空间。

成员变量	数据类型	含义
device_id	char*	输入参数，指示选择哪个设备。如果只连接了一个设备，可以设置为NULL

成员变量	数据类型	含义
reserved	uint8_t	保留字段

5.5.15. rknn3_node_mem_info

RKNN3设备节点内存信息结构体。

该结构体提供了RKNN3设备中每个节点的详细内存信息，包括可用于分配的总内存和可用内存。

成员变量	数据类型	含义
total	uint64_t	此节点可用的总内存，单位为字节
free	uint64_t	此节点可用的空闲内存，单位为字节

5.5.16. rknn3_dev_mem_info

RKNN3设备节点内存信息结构体。

该结构体提供了RKNN3设备中每个节点的详细内存信息，包括可用于分配的总内存和可用内存。

成员变量	数据类型	含义
node_num	uint32_t	设备中的节点数量
sys_total	uint64_t	设备的总系统内存，单位为字节
sys_free	uint64_t	设备的可用系统内存，单位为字节
node_mem_info	rknn3_node_mem_info	每个节点的内存信息

5.5.17. rknn3_device

表示RKNN3设备的结构体。

该结构体包含了特定RKNN3设备的信息，包括其ID、类型和内存信息。

成员变量	数据类型	含义
id	char[RKNN3_MAX_DEV_LEN]	设备ID
type	char[RKNN3_MAX_DEV_LEN]	设备类型
mem_info	rknn3_dev_mem_info	设备的内存信息

5.5.18. rknn3_devices

包含RKNN3设备信息的结构体。

该结构体包含了可用RKNN3设备的数量。

成员变量	数据类型	含义
n_devices	uint32_t	设备数量
devices	rknn3_device	设备信息

5.5.19. rknn3_vocab_info

RKNN3模型词汇信息结构体。

该结构体包含了RKNN3模型中使用的词汇信息，包括大小和特殊Token ID。

成员变量	数据类型	含义
vocab_size	int	词汇表大小
special_bos_id	int[RKNN3_MAX_SPECIAL_BOS_ID_NUM]	特殊序列开始(BOS)Token的ID
special_eos_id	int[RKNN3_MAX_SPECIAL_EOS_ID_NUM]	特殊序列结束(EOS)Token的ID
n_special_bos_id	int	特殊序列开始(BOS)Token ID 的数量
n_special_eos_id	int	特殊序列结束(EOS)Token ID 的数量
linefeed_id	int	换行Token的ID
skip_special_token	bool	生成时是否跳过特殊Token
ignore_eos_token	bool	生成时是否忽略EOS Token
reserved	uint8_t	保留字段，用于未来扩展

5.5.20. rknn3_llm_extend_param

LLM扩展参数结构体。

成员变量	数据类型	含义
reserved	uint8_t	保留字段

5.5.21. rknn3_sampling_params

定义LLM实例的采样参数结构体。

成员变量	数据类型	含义
top_k	int32_t	用于Token生成的Top-K采样参数
top_p	float	Top-P（核心）采样参数
temperature	float	采样温度，影响Token选择的随机性
repeat_penalty	float	生成中重复Token的惩罚
frequency_penalty	float	生成过程中对频繁Token的惩罚
presence_penalty	float	基于Token在输入中的存在对其进行惩罚

5.5.22. rknn3_llm_param

定义配置LLM实例的参数结构体。

成员变量	数据类型	含义
logits_name	char*	输出logits的名称。仅当模型有多个输出时需要显式设置该字段，否则可以为NULL
max_context_len	int32_t	上下文中的最大Token数量
sampling_param	rknn3_sampling_params	Token生成的采样参数
vocab_info	rknn3_vocab_info	词汇信息
extend_param	rknn3_llm_extend_param	扩展参数

5.5.23. rknn3_lora

定义模型微调中使用的LoRA(Low-Rank Adaptation)参数结构体。

成员变量	数据类型	含义
lora_name	char[RKNN3_MAX_NAME_LEN]	LoRA的名称
scale	float	应用LoRA的缩放因子

5.5.24. rknn3_kvcache_policy_param

定义KV缓存策略的参数结构体。
若模型包含system prompt，recurrent参数中的 n_keep 与 n_keep_aligned 会被忽略。

成员变量	数据类型	含义
recurrent.n_keep	int64_t	递归策略下保留的缓存条目数量；若模型包含system prompt则忽略
recurrent.n_keep_aligned	int64_t	递归策略下对齐到kvcache_group_size的保留条目数量
reserved	uint8_t[64]	保留字段

5.5.25. rknn3_llm_multimodal_tensor

表示多模态输入（例如文本、图像、音频和视频）的结构体。
顶层成员

成员变量	数据类型	含义
name	const char*	此 tensor 的名称
prompt	const char*	文本提示输入
tokens	int32_t*	token 输入。与 prompt 输入是二选一关系，用户可提供任意一种输入。注意：1.如果两种输入都提供，则默认按照 prompt 输入推理。2.选择 token 输入时，用户需要自行将文本和多模态标签按照规则解析成 token。
n_tokens	uint64_t	token 输入中 token 的数量。选择 token 输入时需要设置。
enable_thinking	bool	控制是否启用“思考模式”
image	struct	图像子结构
audio	struct	音频子结构
video	struct	视频子结构

image子结构相关成员

成员变量	数据类型	含义
image_embed	float16*	图像的嵌入（大小为 n_image × n_image_tokens × embedding_dim × sizeof(float16)）
n_image_tokens	uint64_t	每张图像的 token 数量
n_image	uint64_t	图像数量
image_start	const char*	多模态输入中图像的起始标签
image_end	const char*	多模态输入中图像的结束标签
image_content	const char*	多模态输入中图像内容占位符标签
image_width	uint64_t	图像宽度
image_height	uint64_t	图像高度

audio子结构相关成员

成员变量	数据类型	含义
audio_embed	float16*	音频的嵌入（大小为 n_audio × n_audio_tokens × embedding_dim × sizeof(float16)）
n_audio_tokens	uint64_t	每段音频的 token 数量
n_audio	uint64_t	音频数量
audio_start	const char*	多模态输入中音频起始标签
audio_end	const char*	多模态输入中音频结束标签
audio_content	const char*	多模态输入中音频内容占位符标签

video子结构相关成员

成员变量	数据类型	含义
video_embed	float16*	视频的嵌入（大小为 n_video × n_frame_per_video × n_frame_tokens × embedding_dim × sizeof(float16)）
n_frame_tokens	uint64_t	每帧的 token 数量
n_frame_per_video	uint64_t	每个视频包含的帧数
n_video	uint64_t	视频数量
video_start	const char*	多模态输入中视频起始标签
video_end	const char*	多模态输入中视频结束标签
video_content	const char*	多模态输入中视频内容占位符标签
frame_width	uint64_t	视频帧的宽度
frame_height	uint64_t	视频帧的高度

5.5.26. rknn3_llm_tensor

表示大型语言模型操作的 tensor 结构体。

该结构体包含了处理语言模型嵌入所需的基本组件，包括 tensor 名称、嵌入向量、Token ID和Token数量。

成员变量	数据类型	含义
name	const char*	此 tensor 的名称
prompt	const char*	如果input_type为RKNN3_LLM_INPUT_PROMPT，则为文本提示输入
embed	float16*	如果input_type为RKNN3_LLM_INPUT_EMBED，则为指向嵌入向量的指针（大小为n_tokens * hidden_size）
tokens	int32_t*	如果input_type为RKNN3_LLM_INPUT_TOKEN，则为Token ID数组
n_tokens	uint64_t	嵌入中表示的Token数量
enable_thinking	bool	控制是否启用"思考模式"

5.5.27. rknn3_llm_input

通过联合体表示LLM不同类型输入的结构体。

成员变量	数据类型	含义
role	const char*	消息角色："user"（用户输入）、"tool"（函数结果）
input_type	rknn3_llm_input_type	指定提供的输入类型（例如token、embed、aux）
llm_input	rknn3_llm_tensor	当input_type为RKNN3_LLM_INPUT_EMBED时为嵌入向量；当input_type为RKNN3_LLM_INPUT_TOKEN时为Token数组；当input_type为RKNN3_LLM_INPUT_PROMPT时使用prompt字段
multimodal_input	rknn3_llm_multimodal_tensor	当input_type为RKNN3_LLM_INPUT_MULTIMODAL时为多模态输入
aux_input	rknn3_aux_tensor	如果input_type为RKNN3_LLM_INPUT_AUX，则为AUX输入,目前rknn3_aux_tensor和rknn3_tensor同类型

5.5.28. rknn3_llm_infer_param

LLM推理参数结构体，定义了推理过程中的参数。

成员变量	数据类型	含义
keep_history	int	确定历史记录保留标志（1：保留历史记录，0：丢弃历史记录）
max_new_tokens	int32_t	生成的新Token最大数量
reserved	uint8_t	保留字段，用于未来扩展

5.5.29. RKLLMResult

表示LLM推理结果的结构体。

成员变量	数据类型	含义
token_ids	int*	指向LLM生成的token的指针
num_tokens	int	生成的token数量

5.5.30. RKLLMCallback

结构体RKLLMCallback包含LLM操作的回调函数。

成员变量	数据类型	含义
result_callback	LLMResultCallback	LLM返回结果的回调函数
result_userdata	void*	LLMResultCallback的用户数据
sampling_callback	LLMSamplingCallback	可选：仅在需要自定义采样时使用
sampling_userdata	void*	LLMSamplingCallback的用户数据
tokenizer_callback	LLMTokenizerCallback	可选：仅在需要自定义分词器时使用
tokenizer_userdata	void*	LLMTokenizerCallback的用户数据
embed_callback	LLMGetEmbedCallback	可选：仅在需要自定义嵌入检索时使用
embed_userdata	void*	LLMGetEmbedCallback的用户数据
output_callback	LLMOutputCallback	可选：仅在需要获取输出tensor时使用
output_userdata	void*	LLMOutputCallback的用户数据
output_tensors	rknn3_tensor*	输出tensor数组指针
n_output_tensors	uint32_t	输出tensor数量

5.5.31. RKLLMRunState

结构体RKLLMRunState包含LLM运行的状态信息。

成员变量	数据类型	含义
n_total_tokens	uint64_t	当前已处理的总Token数量
n_max_tokens	uint64_t	可处理的最大Token数量
n_decode_tokens	uint64_t	解码阶段已生成的Token数量
n_prefill_tokens	uint64_t	预填充阶段处理的Token数量
kvcache_policy	rknn3_kvcache_policy	KV缓存策略
n_loras_enabled	int32_t	启用的LoRA数量
loras_enabled	rknn3_lora*	启用的LoRA列表

5.5.32. rknn3_custom_op_context

自定义算子上下文结构体，由框架管理，用户可通过user_data存放自定义信息。

成员变量	数据类型	含义
rknn_ctx	rknn3_context	RKNN3上下文句柄
priv_data	void*	框架私有数据
user_data	void*	用户自定义数据

5.5.33. rknn3_custom_op

自定义算子结构体，包含算子元信息与回调函数集合。

成员变量	数据类型	含义
op_type	const char*	自定义算子类型名称
plugin_type	rknn3_op_plugin_type	插件类型
target	rknn3_op_target_type	后端执行目标
version	uint32_t	算子版本号
author	const char*	作者信息
description	const char*	算子描述
init	int ()*(rknn3_custom_op_context*)	初始化回调（可选）
prepare	int ()*(rknn3_custom_op_context*, rknn3_tensor, *uint32_t, rknn3_tensor*, uint32_t)	预处理回调（可选）

成员变量	数据类型	含义
compute	int ()(*rknn3_custom_op_context*, rknn3_tensor, *uint32_t*, *rknn3_tensor*, uint32_t)	计算回调（必选）
deinit	int ()(*rknn3_custom_op_context*)	反初始化回调（可选）
get_output_num	int ()(*rknn3_custom_op_context*)	获取输出数量回调（后处理插件，可选）
get_attrs	int ()(*rknn3_custom_op_context*, rknn3_tensor_attr, *uint32_t*, *rknn3_tensor_attr*, uint32_t)	获取输入输出tensor属性回调（后处理插件，可选）

5.5.34. rknn3_im_rect

图像矩形区域结构体，用于描述图像处理中的兴趣区域（ROI）。

成员变量	数据类型	含义
x	int	左上角 X 坐标
y	int	左上角 Y 坐标
width	int	矩形宽度（像素）
height	int	矩形高度（像素）

5.5.35. rknn3_im_metadata

与 RKNN3 图像内存关联的元数据结构体，用于主机与设备之间的额外信息记录。

成员变量	数据类型	含义
peer_im_mem_addr	uint64_t	对端（远端）图像内存对象的地址

5.5.36. rknn3_im_mem

图像内存信息结构体，描述存储在设备内存中的图像缓冲区，供 `rknn3_im_mem_create`、`rknn3_im_cvt_color` 等图像接口使用。

成员变量	数据类型	含义
width	int	图像宽度（像素）
height	int	图像高度（像素）
stride	int	图像缓冲区步长（字节）
format	rknn3_im_fmt	图像格式，见 5.4.15. rknn3_im_fmt
sync_to_host	bool	是否需要将图像数据回写到主机，默认 false

成员变量	数据类型	含义
data_mem	rknn3_tensor_mem*	底层 tensor 内存信息指针
metadata	rknn3_im_metadata	主机与设备共享的额外元数据

6. RKNN3 C API 变动说明

v0.3.0 -> v0.4.0 Session API 主要改动 (2025-11-03)

> 参考 demo1: rknn3-runtime/examples/rknn3_session_test_demo
> 参考 demo2: rknn3_model_zoo/examples/Qwen2_5_VL

1. 新增多模态 `音频/视频` 输入类型，并支持多种模态数据以任意顺序和组合方式输入。
    - 在输入 prompt 时，应明确插入各模态的标记标签，例如：`你需要做3件事：1.<audio>将这段语音转为文本； 2.<image>描述一下这张图片的内容；3.<video>描述一下这段视频的内容`。
    - rknn3_llm_multimodal_tensor 结构体优化：例如，将原有的 input_tensor.n_image 等成员调整为 input_tensor.image.n_image 等子结构体表达。
2. 新增 `思考（thinking）模式` 切换功能。
    - 通过设置 enable_thinking = true 开启思考模式 (默认是 false)。
3. 新增支持配置多个 `eos token` 来控制推理结束。
    - params.vocab_info.special_eos_id 的初始化方式更新为以列表形式进行赋值。
4. 新增 `rknn3_dump_features` 接口，用于按层运行模型并将中间Tensor以 `.npy` 形式导出。支持未提供输出时自动分配输出Tensor。

v0.4.0b0 -> v0.5.0 主要改动 (2025-11-29)

1. 新增后处理相关查询命令与自定义算子插件机制。
    - 新增 `RKNN3_QUERY_POSTPROCESS_*` 查询项。
    - 新增 `rknn3_register_custom_ops_plugins` 以及相关结构体与枚举。
2. LLM回调与配置增强。
    - 新增 `rknn3_session_set_function_tools` 接口。
    - 以 `LLMOutputCallback` + `LLMOutputCallbackState` 取代“最后隐藏层回调”。
    - `rknn3_llm_config` 新增 `model_type` 和 `task_type` 字段。
    - `rknn3_kvcache_policy_param` 新增 recurrent 参数。
3. 图像/YUV相关接口完善。
    - `rknn3_im_mem_create` 参数扩展，支持 size/core_id/flags。
    - 新增 `rknn3_im_proc_flag` 枚举，用于描述图像处理流程。

v0.5.0 -> v1.0.0 主要改动 (2025-12-22)

1. 新增性能与内存分析接口。
    - 新增 `rknn3_profile_ops`、`rknn3_profile_mem`。