Gowin RiscV AE350 SOC NN 软件编程

用户手册

MUG1181-1.1, 2025-03-21

免责声明

本文档并未授予任何知识产权的许可，并未以明示或暗示，或以禁止反言或其它方式授予任何知识产权许可。除高云半导体在其产品的销售条款和条件中声明的责任之外，高云半导体概不承担任何法律或非法律责任。高云半导体对高云半导体产品的销售和／或使用不作任何明示或暗示的担保，包括对产品的特定用途适用性、适销性或对任何专利权、版权或其它知识产权的侵权责任等，均不作担保。高云半导体对文档中包含的文字、图片及其它内容的准确性和完整性不承担任何法律或非法律责任，高云半导体保留修改文档中任何内容的权利，恕不另行通知。高云半导体不承诺对这些文档进行适时的更新。

版本信息

日期	版本	描述
2023/12/29	1.0	初始版本。
2025/03/21	1.1	支持 TinyYOLO v3 示例参考设计。

目录

图目录

表目录

1 关于本手册

1.1 手册内容

本手册主要描述 Gowin RiscV AE350 SOC NN 软件编程函数库的功能规范，如何使用 NN 函数接口软件编程，应用程序演示等。

1.2 术语、缩略语

本手册中的相关术语、缩略语及相关释义如表 1-1 所示。

表 1-1 术语、缩略语

术语、缩略语	全称	含义
DSP	Digital Signal Processor	数字信号处理器
FPU	Floating Point Unit	浮点单元
ISA	Instruction Set Architecture	指令集架构
NN	Neural Network	神经网络
ReLU	Rectified Linear Unit	线性整流函数
RISC- V	Reduced Instruction Set Computer V	第五代精简指令集计算机
SOC	System on Chip	片上系统
Zfh	Half Precision Floating Point	半精度浮点

1.3 技术支持与反馈

高云半导体提供全方位技术支持，在使用过程中如有疑问或建议，可直接与公司联系：

网址：www.gowinsemi.com.cn

E-mail：support@gowinsemi.com

Tel：+86 755 8262 0391

2 概述

Gowin RiscV AE350 SOC NN 软件编程函数库，包含了用于 RDS 软件工具链的神经网络函数功能说明信息，以及描述了如何使用这些函数接口。

NN 软件编程库提供纯 C 语言实现或针对 DSP ISA 扩展进行优化的全部函数功能，这些函数功能包括激活、卷积、全连接、池化和其他工具类辅助函数。使用 RDS 软件工具链进行设计与开发，NN 软件编程库封装了神经网络计算中的计算复杂度，提供了易于使用的应用函数接口，有利于用户节省开发资源，使用户更多得专注于系统设计，降低开发时间。

NN 软件编程函数库的函数接口可以分为以下几类：激活函数、基本数学函数、连接函数、卷积函数、全连接函数、池化函数、归一化指数函数和工具类辅助函数，第 3 章将详细描述每个函数接口。

当调用 NN 函数接口时，只需包含 NN 软件编程函数库的头文件，这些头文件定义了函数接口的原型和实例化结构体，以功能类别作为命名方式，如下所示：

- riscv_nn_activation.h
- riscv_nn_basic.h
- riscv_nn_concatenation.h
- riscv_nn_convolution.h
- riscv_nn_fully_connected.h
- riscv_nn_pooling.h
- riscv_nn_softmax.h
- riscv_nn_util.h

例如，如果需要使用激活函数来激活神经网络数据，则可以在 C 文件中包含头文件 riscv_nn_activation.h。

NN 软件编程函数库定义了几种数据类型来保存数据，以便于后续计算处理。这些数据类型，定义于 riscv_math_types.h，分别如下所示，q7_t 表示有符号 8 位整数，u8_t 表示无符号 8 位整数，q15_t 表示有符号 16 位整数，u16_t 表示无符号 16 位整数，q31_t 表示有符号 32 位整数，u32_t 表

示无符号 32 位整数，q63_t 表示有符号 64 位整数，u64_t 表示无符号 64 位整数，float16_t 表示 16 位半精度浮点，float32_t 表示 32 位单精度浮点，float64_t 表示 64 位双精度浮点，如下所示：

- typedef int8_t q7_t; /* for signed 8-bit integer */
- typedef uint8_t u8_t; /* for unsigned 8-bit integer */
- typedef int16_t q15_t; /* for signed 16-bit integer */
- typedef uint16_t u16_t; /* for unsigned 16-bit integer */
- typedef int32_t q31_t; /* for signed 32-bit integer */
- typedef uint32_t u32_t; /* for unsigned 32-bit integer */
- typedef int64_t q63_t; /* for signed 64-bit integer */
- typedef uint64_t u64_t; /* for unsigned 64-bit integer */
- typedef _Float16 float16_t; /* for half-precision (16-bit) floating-point */
- typedef float float32_t; /* for single-precision (32-bit) floating-point */
- typedef double float64_t; /* for double-precision (64-bit) floating-point */

另外，NN 软件编程库还定义了 riscv_nn_activation_fun 类型，定义于 riscv_nn_activation.h，用于选择激活函数：

```
typedef enum
{
NN_SIGMOID = 0, /* Use sigmoid activation function */
NN_TANH = 1, /* Use tanh activation function */
} riscv_nn_activation_fun;
```

2.1 Newlib 和 MCUlib 工具链的链接选项

除上述的头文件外，不同的工具链关联不同的链接选项，本节详细介绍用于 Newlib 和 MCUlib 的选项，包括：

- -lnn 链接 NN 软件编程函数库（libnn.a）
- -mext-dsp 开启 DSP ISA 扩展
- -mzfh 开启半精度浮点数据类型和 Zfh 扩展

如果目标系统不支持 DSP ISA 扩展来加速 NN 软件编程函数库，则使用-lnn 构建一个链接 NN 软件编程函数库（libnn.a）的应用程序。如果目标系统支持 DSP ISA 扩展，则使用-lnn，-mext-dsp 选项开启 DSP ISA 扩展以及链接 NN 软件编程库 libnn.a，以达到更高的性能。请注意使用-lnn，-mext-dsp 选项构建的应用程序，不能在不支持 DSP ISA 扩展的系统上运

行，否则加载时会出现问题。

如果应用程序需要使用包含半精度浮点数据类型（即以“f16”命名的函数），请务必使用支持 FPU 的工具链（即 v5f 或 v5d 工具链），并且使用选项-mzfh 开启半精度浮点数据类型和 Zfh 扩展。

2.2 Glibc 工具链的链接选项

对于 Glibc 工具链，可用的链接选项，包括：

- -static 使用静态链接
- -lnn 链接纯 C 语言实现的 NN 软件编程函数库
- -lnn_p 链接具有 DSP ISA 扩展的 NN 软件编程函数库

静态链接和动态链接需要不同的选项，表 2-1 总结了不同链接类型以及是否需要 DSP ISA 扩展支持的链接选项。

表 2-1 链接选项

链接类型	DSP ISA 扩展	链接选项
Static Linking	Without	-static -lnn
	With	-static -lnn_p
Dynamic Linking (default)	Without	-lnn
	With	-lnn_p

所需选项之中，具有 DSP ISA 扩展（即-static -lnn_p 和-lnn_p）的选项，用于支持扩展的目标系统构建应用程序，而其他选项（即-static -lnn 和 -lnn），仅用于不支持 DSP ISA 扩展的目标系统。请用户确保指定的选项对应于所用的目标系统。

3 函数接口描述

以下各节按功能分类，分别详细描述 NN 软件编程函数库的函数接口。

3.1 激活函数

激活函数，用于过滤掉一些输入数据，包括 sigmoid 函数、tanh 函数和 ReLU 函数。

3.1.1 riscv_nn_activate_s8

原型：

void riscv_nn_activate_s8 (q7_t *in_out, uint32_t size, uint16_t int_bits, riscv_nn_activation_fun act_fun)

描述：

对有符号 8 位整数输入向量，该函数使用 sigmoid 或 tanh 函数执行激活。

参数：

- [in, out] *in_out 指向输入或输出向量的指针
- [in] size 输入或输出向量的元素数量
- [in] int_bits 整数部分的位数，取值应小于 4
- [in] act_fun 激活函数的选择，NN_SIGMOID 用于选择 sigmoid 激活函数，NN_TANH 用于选择 tanh 激活函数

返回值：

无

3.1.2 riscv_nn_activate_s16

原型：

void riscv_nn_activate_s16 (q15_t *in_out, uint32_t size, uint16_t

int_bits, riscv_nn_activation_fun act_fun)

描述：

对有符号 16 位整数输入向量，该函数使用 sigmoid 或 tanh 函数执行激活。

参数：

- [in, out] *in_out 指向输入或输出向量的指针
- [in] size 输入或输出向量的元素数量
- [in] int_bits 整数部分的位数，取值应小于 4
- [in] act_fun 激活函数的选择，NN_SIGMOID 用于选择 sigmoid 激活函数，NN_TANH 用于选择 tanh 激活函数

返回值：

无

3.1.3 riscv_nn_leaky_relu_s8

原型：

void riscv_nn_leaky_relu_s8 (q7_t *in_out, uint32_t size, q15_t slope)

描述：

对有符号 8 位整数输入向量，该函数使用 leaky ReLU 函数执行激活。

参数：

- [in, out] *in_out 指向输入或输出向量的指针
- [in] size 输入或输出向量的元素数量
- [in] slope 与负输入相乘的斜率值，结果将右移 15 位回缩为有符号 8 位整数

返回值：

无

示例：

```
#define SIZE 1024
q15_t slope = 16384;
q7_t in_out[SIZE] = {...};
riscv_nn_leaky_relu_s8 (in_out, SIZE, slope);
```

3.1.4 riscv_nn_relu_any_s8

原型：

void riscv_nn_relu_any_s8 (q7_t *data, uint16_t size, q7_t max_val)

描述：

对有符号 8 位整数输入向量，该函数使用 ReLU 函数执行激活。

参数：

- [in, out] *data 指向输入或输出向量的指针
- [in] size 输入或输出向量的元素数量
- [in] max_val 用于限制输出向量的最大值

返回值：

无

3.1.5 riscv_nn_relu_s8

原型：

```
void riscv_nn_relu_s8 (q7_t *in_out, uint32_t size)
```

描述：

对有符号 8 位整数输入向量，该函数使用 ReLU 函数执行激活。

参数：

- [in, out] *in_out 指向输入或输出向量的指针
- [in] size 输入或输出向量的元素数量

返回值：

无

示例：

```
#define H 16
#define W 16
#define CH 5
#define NUM (H * W * CH)
q7_t in_out[NUM] = {...};
riscv_nn_relu_s8 (in_out, NUM);
```

3.1.6 riscv_nn_relu_s16

原型：

```
void riscv_nn_relu_s16 (q15_t *in_out, uint32_t size)
```

描述：

对有符号 16 位整数输入向量，该函数使用 ReLU 函数执行激活。

参数：

- [in, out] *in_out 指向输入或输出向量的指针
- [in] size 输入或输出向量的元素数量

返回值：

无

3.1.7 riscv_nn_sigmoid_f16

原型：

int32_t riscv_nn_sigmoid_f16 (const float16_t * in_vec, uint32_t size, float16_t * out_vec)

描述：

对半精度浮点输入向量，该函数使用 sigmoid 函数执行激活。

参数：

- [in] *in_vec 指向输入向量的指针
- [in] size 输入或输出向量的元素数量
- [out] *out_vec 指向输出向量的指针

返回值：

0

注!

输入向量的限制范围[-10, 10]。

3.1.8 riscv_nn_tanh_f16

原型：

int32_t riscv_nn_tanh_f16 (const float16_t * in_vec, uint32_t size, float16_t * out_vec)

描述：

对半精度浮点输入向量，该函数使用 tanh 函数执行激活。

参数：

- [in] *in_vec 指向输入向量的指针
- [in] size 输入或输出向量的元素数量
- [out] *out_vec 指向输出向量的指针

返回值：

0

注!

输入向量的限制范围[-10, 10]。

3.2 基本函数

基本函数用于执行逐元素的基本算术运算。

3.2.1 riscv_nn_add_s8_sym

原型：

void riscv_nn_add_s8_sym (const q7_t *in_tensor1, const q7_t * in_tensor2, const int16_t *scale1, const int16_t *scale2, const uint32_t size, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q7_t * out)

描述：

该函数对有符号 8 位整数输入向量执行逐元素加法，以及对输出执行对称量化。

参数：

- [in] *in_tensor1 指向第一个输入向量的指针
- [in] *in_tensor2 指向第二个输入向量的指针
- [in] *scale1 指向第一个缩放向量的指针
- [in] *scale2 指向第二个缩放向量的指针
- [in] size 输入向量的元素数量
- [in] pre_rshift 缩放前累加器的右移量
- [in] out_scale 累加器的缩放值
- [in] post_rshift 缩放后累加器的右移量
- [out] *out 指向逐元素加法结果的指针

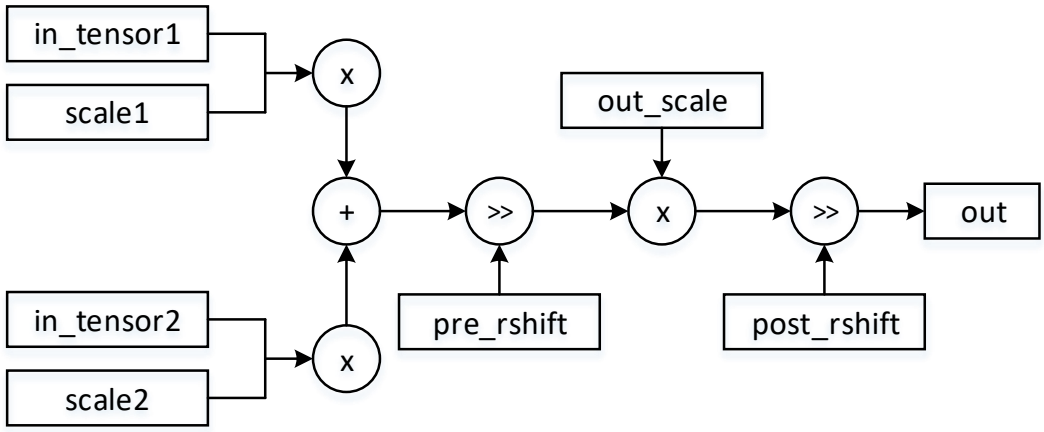
返回值：

无

注!

每个元素的计算流程如图 3-1 所示，其中矩阵表示输入或输出，圆圈表示算术运算，实线表示必需的步骤，虚线（如果有的话）表示是可选的。

图 3-1 riscv_nn_add_s8_sym 算法流程



示例：

```
#define SIZE 1024
uint16_t pre_rshift = 8; // The addition results of both scaled input
// tensors are in the range of 24-bit; thus, the
// pre_rshift should be in the range of [0, 24].
// Here we scale down the results into 16-bit
// range.
uint16_t out_scale = 3; // Scale up the result into 18-bit range.
uint16_t post_rshift = 11; // Scale down the result into 7-bit range.
q7_t in_tensor1[SIZE] = {...};
q7_t in_tensor2[SIZE] = {...};
q15_t scale1[SIZE] = {...};
q15_t scale2[SIZE] = {...};
q7_t out[SIZE];
riscv_nn_add_s8_sym (in_tensor1, in_tensor2, scale1, scale2, SIZE,
pre_rshift, out_scale, post_rshift, out);
```

3.2.2 riscv_nn_add_s8_sym_round

原型：

```
void riscv_nn_add_s8_sym_round (const q7_t *in_tensor1, const q7_t
* in_tensor2, const uint32_t scale1, const uint32_t scale2, const uint32_t
size, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t
post_rshift, q7_t * out)
```

描述：

该函数对有符号 8 位整数输入向量执行逐元素加法，以及对输出执行对称量化和舍入。

参数：

- [in] *in_tensor1 指向第一个输入向量的指针
- [in] *in_tensor2 指向第二个输入向量的指针
- [in] scale1 第一个输入向量的缩放值，取值范围应为 0~2^{23}
- [in] scale2 第二个输入向量的缩放值，取值范围应为 0~2^{23}
- [in] size 输入向量的元素数量
- [in] pre_rshift 缩放前累加器的右移量
- [in] out_scale 累加器的缩放值
- [in] post_rshift 缩放后累加器的右移量
- [out] *out 指向逐元素加法结果的指针

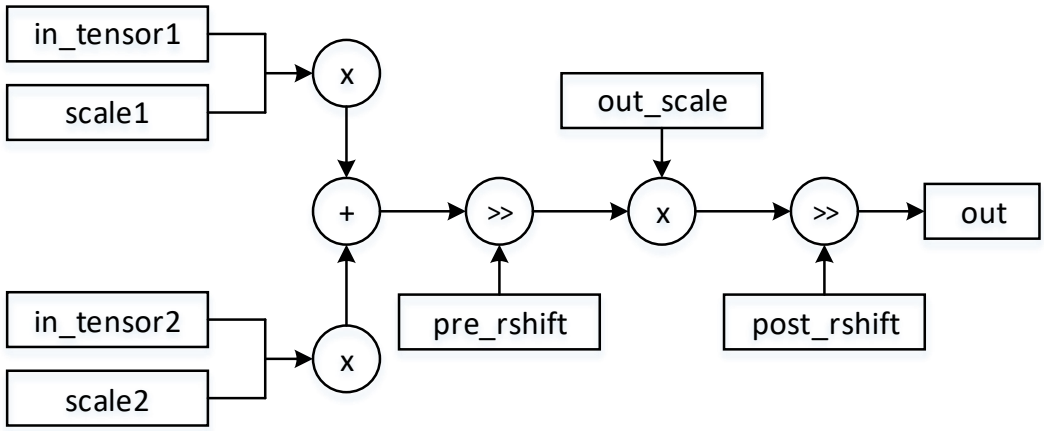
返回值：

无

注!

- 每个元素的计算流程如图 3-2 所示。
- 该函数的右移操作包括舍入。

图 3-2 riscv_nn_add_s8_sym_round 算法流程



3.2.3 riscv_nn_ew_add_s8_asym

原型：

int riscv_nn_ew_add_s8_asym (const int8_t *in_tensor1, const int8_t * in_tensor2, const int32_t in_offset1, const int32_t in_scale1, const int32_t in_rshift1, const int32_t in_offset2, const int32_t in_scale2, const int32_t in_rshift2, const int32_t lshift, int8_t * out, const int32_t out_offset,

const int32_t out_scale, const int32_t out_rshift, const int32_t act_min, const int32_t act_max, const uint32_t size)

描述：

该函数对有符号 8 位整数输入向量执行逐元素加法。

参数：

- [in] *in_tensor1 指向第一个输入向量的指针
- [in] *in_tensor2 指向第二个输入向量的指针
- [in] in_offset1 第一个输入向量的偏移值，取值范围应为-127~128
- [in] scale1 第一个输入向量的缩放值
- [in] in_rshift1 第一个输入向量的右移量
- [in] in_offset2 第二个输入向量的偏移值，取值范围应为-127~128
- [in] scale2 第二个输入向量的缩放值
- [in] in_rshift2 第二个输入向量的右移量
- [in] lshift 第一个和第二个输入向量的左移量
- [out] *out 指向逐元素加法结果的指针
- [in] out_offset 输出向量的偏移量
- [in] out_scale 输出向量的缩放值
- [in] out_rshift 输出向量的右移量
- [in] act_min 限制输出向量的最小值
- [in] act_max 限制输出向量的最大值
- [in] size 输入向量的元素数量

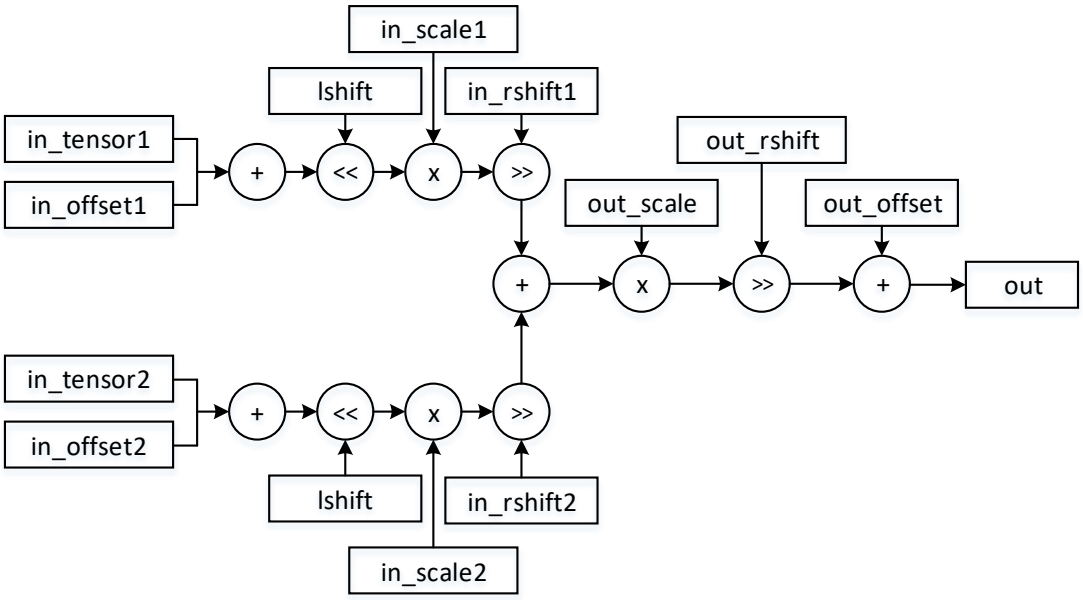
返回值：

0

注!

- 每个元素的计算流程如图 3-3 所示。
- in_scale1、in_scale2 和 out_scale 的乘法，大致可以表示为：32b = ((int64_t)32b * 32b) >> 31。

图 3-3 riscv_nn_ew_add_s8_asym_算法流程



示例：

```
#define SIZE 1024
int32_t in_offset1 = 16; // Offset for in_tensor1
int32_t in_scale1 = (1<<28); // Scale down in_tensor1 by 1/2^3
int32_t in_rshift1 = 3; // Scale down in_tensor1 by 1/2^3
int32_t in_offset2 = 17; // Offset for in_tensor2
int32_t in_scale2 = (1<<28); // Scale down in_tensor2 by 1/2^3
int32_t in_rshift2 = 3; // Scale down in_tensor2 by 1/2^3
int32_t lshift = 10; // Scale up the input tensor by 2^10 times
int32_t out_offset = 18; // Offset for the output tensor
int32_t out_scale = (1<<30); // Scale down in_tensor2 by 1/2
int32_t out_rshift = 4; // Scale down in_tensor2 by 1/2^4
int32_t act_min = 0xffffffa3; // Limit the outputs in the range of
// [0xffffffa3, 0x0000005d]
int32_t act_max = 0x0000005d; // Limit the outputs in the range of
// [0xffffffa3, 0x0000005d]
int8_t in_tensor1[SIZE] = {...};
int8_t in_tensor2[SIZE] = {...};
int8_t out[SIZE];
```

riscv_nn_ew_add_s8_asym (in_tensor1, in_tensor2, in_offset1, in_scale1, in_rshift1, in_offset2, in_scale2, in_rshift2, lshift, out, out_offset, out_scale, out_rshift, act_min, act_max, SIZE);

3.2.4 riscv_nn_ew_mul_s8_asym

原型：

int riscv_nn_ew_mul_s8_asym (const int8_t * in_tensor1, const int8_t * in_tensor2, const int32_t in_offset1, const int32_t in_offset2, int8_t * out, const int32_t out_offset, const int32_t out_scale, const int32_t out_shift, const int32_t act_min, const int32_t act_max, const uint32_t size)

描述：

该函数对有符号 8 位整数输入向量执行逐元素乘法。

参数：

- [in] *in_tensor1 指向第一个输入向量的指针
- [in] *in_tensor2 指向第二个输入向量的指针
- [in] in_offset1 第一个输入向量的偏移值，取值范围应为-127~128
- [in] in_offset2 第二个输入向量的偏移值，取值范围应为-127~128
- [out] *out 指向逐元素乘法结果的指针
- [in] out_offset 输出向量的偏移量
- [in] out_scale 输出向量的缩放值
- [in] out_shift 输出向量的移位量
- [in] act_min 限制输出向量的最小值
- [in] act_max 限制输出向量的最大值
- [in] size 输入向量的元素数量

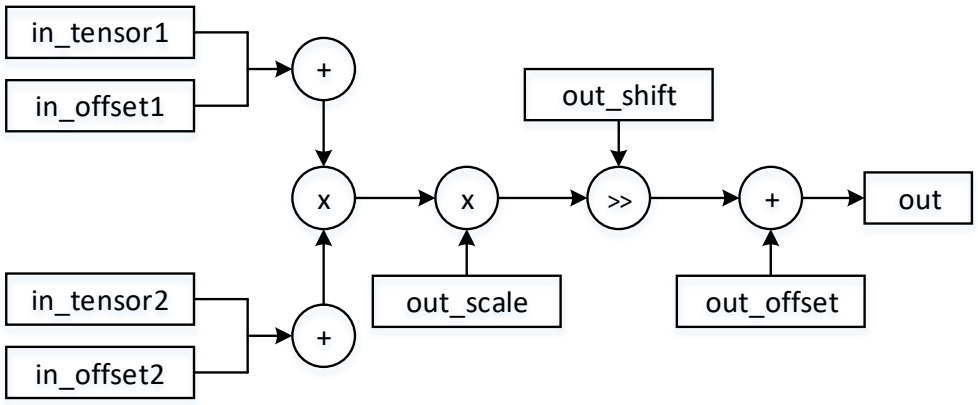
返回值：

0

注！

- 每个元素的计算流程如图 3-4 所示。
- out_scale 的乘法，大致可以表示为：32b = ((int64_t)32b * 32b) >> 31

图 3-4 riscv_nn_ew_mul_s8_asym 算法流程



示例：

```
#define SIZE 1024
int32_t in_offset1 = 16; // Offset for in_tensor1
int32_t in_offset2 = 17; // Offset for in_tensor2
int32_t out_offset = 18; // Offset for the output tensor
int32_t out_scale = (1<<30); // Scale down the output tensor by 1/2
int32_t out_shift = -4; // Scale down the output tensor by 1/2^4
int32_t act_min = 0xffffffa3; // Limit the outputs in the range of
// [0xffffffa3, 0x0000005d]
int32_t act_max = 0x0000005d; // Limit the outputs in the range of
// [0xffffffa3, 0x0000005d]
in_tensor1[SIZE] = {...};
in_tensor2[SIZE] = {...};
out[SIZE];
riscv_nn_ew_mul_s8_asym (in_tensor1, in_tensor2, in_offset1,
in_offset2, out, out_offset, out_scale, out_shift, act_min, act_max, SIZE);
```

3.3 连接函数

连接函数用于沿指定轴连接张量。

3.3.1 riscv_nn_concate_s8_w

原型：

```
void riscv_nn_concate_s8_w (const int8_t *in_tensor, const uint16_t
in_tensor_x, const uint16_t in_tensor_y, const uint16_t in_tensor_z, const
```

uint16_t in_tensor_w, int8_t *out_tensor, const uint32_t out_offset_w)

描述：

该函数沿 w 轴连接 int8_t/uint8_t 输入张量和输出张量。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_x 输入张量的 x 维度
- [in] in_tensor_y 输入张量的 y 维度
- [in] in_tensor_z 输入张量的 z 维度
- [in] in_tensor_w 输入张量的 w 维度
- [out] *out_tensor 指向输出张量的指针
- [in] out_offset_w 在连接之前要加到输出张量的 w 轴上的偏移值

返回值：

无

注！

输出张量的 x、y 和 z 维度与输入张量的相同。

3.3.2 riscv_nn_concate_s8_x

原型：

void riscv_nn_concate_s8_x (const int8_t *in_tensor, const uint16_t in_tensor_x, const uint16_t in_tensor_y, const uint16_t in_tensor_z, const uint16_t in_tensor_w, int8_t *out_tensor, const uint16_t out_tensor_x, const uint32_t out_offset_x)

描述：

该函数沿 x 轴连接 int8_t/uint8_t 输入张量和输出张量。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_x 输入张量的 x 维度
- [in] in_tensor_y 输入张量的 y 维度
- [in] in_tensor_z 输入张量的 z 维度
- [in] in_tensor_w 输入张量的 w 维度
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_x 输出张量的 x 维度
- [in] out_offset_x 在连接之前要加到输出张量的 x 轴上的偏移值

返回值：

无

注！

输出张量的 y、z 和 w 维度与输入张量的相同。

3.3.3 riscv_nn_concate_s8_y

原型：

void riscv_nn_concate_s8_y (const int8_t *in_tensor, const uint16_t in_tensor_x, const uint16_t in_tensor_y, const uint16_t in_tensor_z, const uint16_t in_tensor_w, int8_t *out_tensor, const uint16_t out_tensor_y, const uint32_t out_offset_y)

描述：

该函数沿 y 轴连接 int8_t/uint8_t 输入张量和输出张量。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_x 输入张量的 x 维度
- [in] in_tensor_y 输入张量的 y 维度
- [in] in_tensor_z 输入张量的 z 维度
- [in] in_tensor_w 输入张量的 w 维度
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_y 输出张量的 y 维度
- [in] out_offset_y 在连接之前要加到输出张量的 y 轴上的偏移值

返回值：

无

注！

输出张量的 x、z 和 w 维度与输入张量的相同。

3.3.4 riscv_nn_concate_s8_z

原型：

void riscv_nn_concate_s8_z (const int8_t *in_tensor, const uint16_t in_tensor_x, const uint16_t in_tensor_y, const uint16_t in_tensor_z, const uint16_t in_tensor_w, int8_t * out_tensor, const uint16_t out_tensor_z, const uint32_t out_offset_z)

描述：

该函数沿 z 轴连接 int8_t/uint8_t 输入张量和输出张量。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_x 输入张量的 x 维度
- [in] in_tensor_y 输入张量的 y 维度
- [in] in_tensor_z 输入张量的 z 维度
- [in] in_tensor_w 输入张量的 w 维度
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_z 输出张量的 z 维度
- [in] out_offset_z 在连接之前要加到输出张量的 z 轴上的偏移值

返回值：

无

注！

输出张量的 x、y 和 w 维度与输入张量的相同。

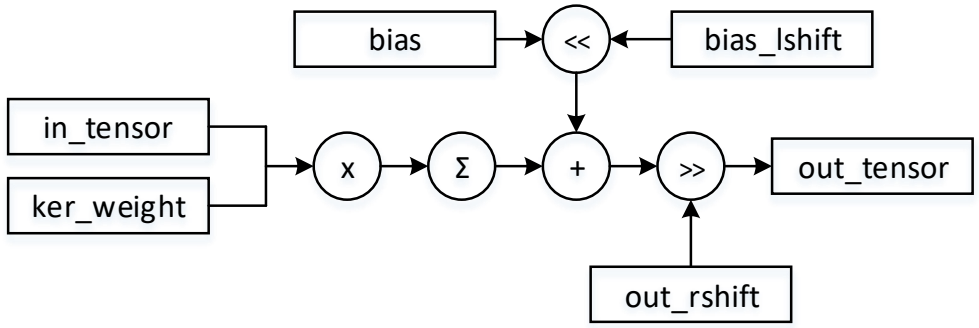
3.4 卷积函数

卷积函数使用 im2col 将输入矩阵转换为列向量，然后使用矩阵-矩阵乘法得到卷积结果。卷积输出有三种类型的量化：基于移位的量化（以“sft”命名）、对称量化（以“sym”命名）和非对称量化（以“asym”命名）。

下面说明了各个量化类型的算法流程，其中矩形表示输入或输出，圆圈表示算术运算，实线表示必需的步骤，虚线表示是可选的。

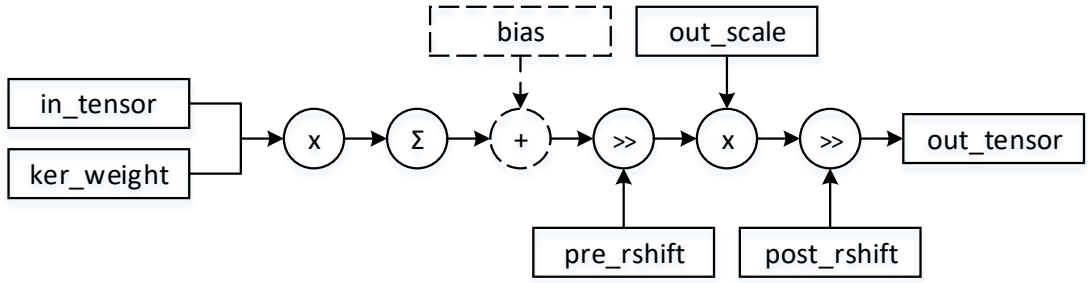
- 基于移位量化的卷积函数的算法流程如图 3-5 所示。

图 3-5 基于移位量化的卷积函数的算法流程



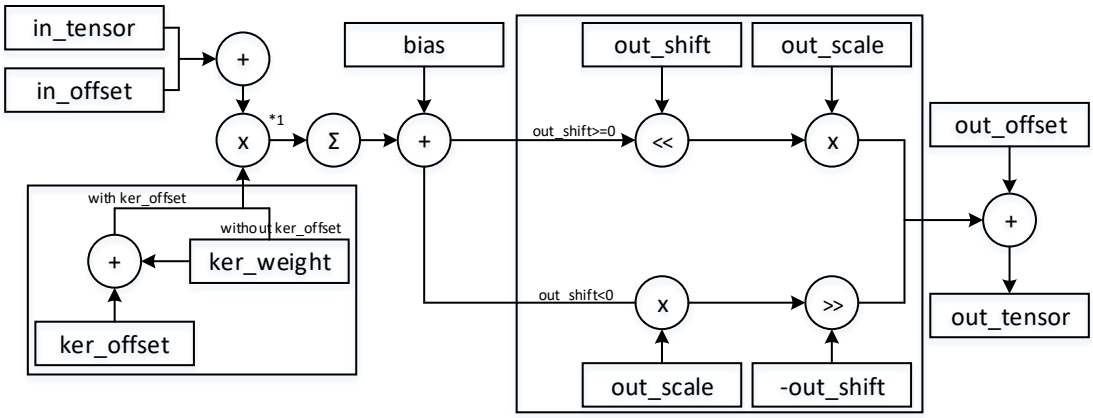
- 对称量化的卷积函数的算法流程如图 3-6 所示。

图 3-6 对称量化的卷积函数的算法流程



- 非对称量化的卷积函数的算法流程如图 3-7 所示。

图 3-7 非对称量化的卷积函数的算法流程



注意在非对称量化的卷积函数中，乘法运算（*1）的输入可以是“ker_weight + ker_offset”或“ker_weight”，这取决于参数 ker_offset。此外，只有当 out_shift 为正时才执行左移操作，只有当 out_shift 为负时才执行右移操作。out_scale 的乘法运算大致可以表示为：

32b = ((int64_t)32b * 32b) >> 31

3.4.1 riscv_nn_conv_1x1_HWC_s8_s8_s8_sft_bias_fast_any

原型：

int32_t riscv_nn_conv_1x1_HWC_s8_s8_s8_sft_bias_fast_any (const q7_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const uint16_t stride_y, const q7_t * bias, const uint16_t bias_lshift, const uint16_t out_rshift, q7_t * out_tensor, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, q15_t * in_tmp_buf, q7_t * tmp_buf)

描述：

该函数在任意 x 和 y 维度对有符号 8 位整数输入/输出执行 1x1 核卷积，以及对输出执行基于移位的量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] *bias 指向偏置向量的指针
- [in] bias_lshift 偏置的左移量
- [in] out_rshift 输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针
- [in] *tmp_buf 空指针

返回值：

该函数成功则返回 0；否则，如果输入不满足约束条件，则返回-1，详细信息请参照下面的注释。

注！

该函数的输入约束条件：

- in_tensor_ch 4 的倍数
- out_tensor_ch 2 的倍数
- ker_dim_x 1
- ker_dim_y 1

- pad_x 0
- pad_y 0
- stride_x 1
- stride_y 1

示例：

```
//Convolve a 160x120x20 input tensor with a 1x1 kernel and generate a 160x120x8
//output tensor. Let both dimensions' padding be 0 and their stride be 1.
#define IN_X 160
#define IN_Y 120
#define IN_CH 20
#define OUT_CH 8
#define KER_DIM_X 1
#define KER_DIM_Y 1
#define PAD_X 0
#define PAD_Y 0
#define STRIDE_X 1
#define STRIDE_Y 1
#define BIAS_LSHIFT 6 //Scale up the bias by 2^6
#define OUT_RSHIFT 9 //Scale down the output tensor by 1/2^9
#define OUT_X 160
#define OUT_Y 120
q7_t in_data[IN_CH * IN_X * IN_Y] = {...};
q7_t weight[IN_CH * KER_DIM_X * KER_DIM_Y * OUT_CH] = {...};
q7_t bias[OUT_CH] = {...};
q15_t in_tmp_buf[2 * IN_CH * KER_DIM_X * KER_DIM_Y] = {0};
q7_t out_data[OUT_CH * OUT_X * OUT_Y];
riscv_nn_conv_1x1_HWC_s8_s8_s8_sft_bias_fast_any (in_data, IN_X, IN_Y, IN_CH, weight, OUT_CH, KER_DIM_X, KER_DIM_Y, PAD_X, PAD_Y, STRIDE_X, STRIDE_Y, bias, BIAS_LSHIFT, OUT_RSHIFT, out_data, OUT_X, OUT_Y, in_tmp_buf, NULL);
```

3.4.2 riscv_nn_conv_HWC_s8_s8_s8_RGB_sft_bias

原型：

int32_t riscv_nn_conv_HWC_s8_s8_s8_RGB_sft_bias (const q7_t * in_tensor, const uint16_t in_tensor_dim, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const q7_t * bias, const uint16_t bias_lshift, const uint16_t out_rshift, q7_t * out_tensor, const uint16_t out_tensor_dim, q15_t * in_tmp_buf, q7_t * tmp_buf)

描述：

该函数对 RGB 图像执行有符号 8 位整数卷积，以及对输出执行基于移位的量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小
- [in] stride 卷积步距
- [in] *bias 指向偏置向量的指针
- [in] bias_lshift 偏置的左移量
- [in] out_rshift 输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针
- [in] *tmp_buf 指向卷积核权重临时缓存区的指针

返回值：

0

示例：

```
//Convolve a 28x28x3 input tensor with a 5x5 kernel and generate a 24x24x20 output

//tensor. Let both dimensions' padding be 0 and their stride be 1.

#define IN_DIM 28
```

```
#define KER_DIM 5
#define PAD 0
#define STRIDE 1
#define BIAS_LSHIFT 6 //Scale up the bias by 2^6
#define OUT_RSHIFT 10 //Scale down the output tensor by 1/2^10
#define OUT_CH 20
#define OUT_DIM 24
q7_t in_data[3 * IN_DIM * IN_DIM] = {...};
q7_t weight[3 * KER_DIM * KER_DIM * OUT_CH] = {...};
q7_t bias[OUT_CH] = {...};
q15_t in_tmp_buf[2 * 3 * KER_DIM * KER_DIM] = {0};
q7_t out_data[OUT_CH * OUT_DIM * OUT_DIM];
riscv_nn_conv_HWC_s8_s8_s8_RGB_sft_bias (in_data, IN_DIM, weight, OUT_CH, KER_DIM, PAD, STRIDE, bias, BIAS_LSHIFT, OUT_RSHIFT, out_data, OUT_DIM, in_tmp_buf, NULL);
```

3.4.3 riscv_nn_conv_HWC_s8_s8_s8_RGB_sft_bias_fast

原型：

```
int32_t riscv_nn_conv_HWC_s8_s8_s8_RGB_sft_bias_fast (const q7_t * in_tensor, const uint16_t in_tensor_dim, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const q7_t * bias, const uint16_t bias_lshift, const uint16_t out_rshift, q7_t * out_tensor, const uint16_t out_tensor_dim, q15_t * in_tmp_buf, q15_t * wt_tmp_buf)
```

描述：

该函数对 RGB 图像执行快速的有符号 8 位整数卷积，以及对输出执行基于移位的量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小
- [in] stride 卷积步距

- [in] *bias 指向偏置向量的指针
- [in] bias_lshift 偏置的左移量
- [in] out_rshift 输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针
- [in] *wt_tmp_buf 指向卷积核权重临时缓存区的指针

返回值：

0

示例：

```
//Convolve a 28x28x3 input tensor with a 5x5 kernel and generate a 24x24x20 output
//tensor. Let both dimensions' padding be 0 and their stride be 1.
#define IN_DIM 28
#define KER_DIM 5
#define PAD 0
#define STRIDE 1
#define BIAS_LSHIFT 6 //Scale up the bias by 2^6
#define OUT_RSHIFT 10 //Scale down the output tensor by 1/2^10
#define OUT_CH 20
#define OUT_DIM 24
q7_t in_data[3 * IN_DIM * IN_DIM] = {...};
q7_t weight[3 * KER_DIM * KER_DIM * OUT_CH] = {...};
q7_t bias[OUT_CH] = {...};
q15_t in_tmp_buf[2 * (3 * KER_DIM * KER_DIM + 1)] = {0};
q15_t wt_tmp_buf[OUT_CH * (3 * KER_DIM * KER_DIM + 1)];
q7_t out_data[OUT_CH * OUT_DIM * OUT_DIM];
riscv_nn_conv_HWC_s8_s8_s8_RGB_sft_bias_fast (in_data, IN_DIM, weight, OUT_CH, KER_DIM, PAD, STRIDE, bias, BIAS_LSHIFT, OUT_RSHIFT, out_data, OUT_DIM, in_tmp_buf, wt_tmp_buf);
```

3.4.4 riscv_nn_conv_HWC_s8_s8_s8_sft_bias

原型：

int32_t riscv_nn_conv_HWC_s8_s8_s8_sft_bias (const q7_t *in_tensor, const uint16_t in_tensor_dim, const uint16_t in_tensor_ch, const q7_t *ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const q7_t *bias, const uint16_t bias_lshift, const uint16_t out_rshift, q7_t * out_tensor, const uint16_t out_tensor_dim, q15_t *in_tmp_buf, q7_t *tmp_buf)

描述：

该函数执行有符号 8 位整数卷积，以及对输出执行基于移位的量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小
- [in] stride 卷积步距
- [in] *bias 指向偏置向量的指针
- [in] bias_lshift 偏置的左移量
- [in] out_rshift 输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针
- [in] *tmp_buf 空指针

返回值：

0

示例：

//Convolve a 28x28x1 input tensor with a 5x5 kernel and generate a 24x24x20

//output tensor. Let both dimensions' padding be 0 and their stride be 1.

```
#define IN_DIM 28
#define IN_CH 1
#define KER_DIM 5
#define PAD 0
#define STRIDE 1
#define BIAS_LSHIFT 6 //Scale up the bias by 2^6
#define OUT_RSHIFT 10 //Scale down the output tensor by 1/2^10
#define OUT_CH 20
#define OUT_DIM 24
q7_t in_data[IN_CH * IN_DIM * IN_DIM] = {...};
q7_t weight[IN_CH * KER_DIM * KER_DIM * OUT_CH] = {...};
q7_t bias[OUT_CH] = {...};
q15_t in_tmp_buf[2 * IN_CH * KER_DIM * KER_DIM] = {0};
q7_t out_data[OUT_CH * OUT_DIM * OUT_DIM];
riscv_nn_conv_HWC_s8_s8_s8_sft_bias (in_data, IN_DIM, IN_CH, weight, OUT_CH, KER_DIM, PAD, STRIDE, bias, BIAS_LSHIFT, OUT_RSHIFT, out_data, OUT_DIM, in_tmp_buf, NULL);
```

3.4.5 riscv_nn_conv_HWC_s8_s8_s8_sft_bias_any

原型：

```
void riscv_nn_conv_HWC_s8_s8_s8_sft_bias_any (const q7_t *in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const q7_t *ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const uint16_t stride_y, const q7_t *bias, const uint16_t bias_lshift, const uint16_t out_rshift, q7_t *out_tensor, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, q15_t *in_tmp_buf, q7_t *tmp_buf)
```

描述：

该函数在任意 x 和 y 维度执行有符号 8 位整数卷积，以及对输出执行基于移位的量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度

- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] *bias 指向偏置向量的指针
- [in] bias_lshift 偏置的左移量
- [in] out_rshift 输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针
- [in] *tmp_buf 空指针

返回值

无

示例：

```
//Convolve a 160x120x3 input tensor with a 3x5 kernel and generate an 80x59x5
//output tensor. Let both dimensions' padding be 1 and their stride be 2.
#define IN_X 160
#define IN_Y 120
#define IN_CH 3
#define OUT_CH 5
#define KER_DIM_X 3
#define KER_DIM_Y 5
#define PAD_X 1
#define PAD_Y 1
```

```
#define STRIDE_X 2
#define STRIDE_Y 2
#define BIAS_LSHIFT 6 //Scale up the bias by 2^6
#define OUT_RSHIFT 9 //Scale down the output tensor by 1/2^9
#define OUT_X 80
#define OUT_Y 59
q7_t in_data[IN_CH * IN_X * IN_Y] = {...};
q7_t weight[IN_CH * KER_DIM_X * KER_DIM_Y * OUT_CH] = {...};
q7_t bias[OUT_CH] = {...};
q15_t in_tmp_buf[2 * IN_CH * KER_DIM_X * KER_DIM_Y] = {0};
q7_t out_data[OUT_CH * OUT_X * OUT_Y];
riscv_nn_conv_HWC_s8_s8_s8_sft_bias_any (in_data, IN_X, IN_Y , IN_CH, weight, OUT_CH, KER_DIM_X, KER_DIM_Y, PAD_X, PAD_Y, STRIDE_X, STRIDE_Y, bias, BIAS_LSHIFT, OUT_RSHIFT, out_data, OUT_X, OUT_Y, in_tmp_buf, NULL);
```

3.4.6 riscv_nn_conv_HWC_s8_s8_s8_sft_bias_fast

原型：

```
int32_t riscv_nn_conv_HWC_s8_s8_s8_sft_bias_fast (const q7_t *in_tensor, const uint16_t in_tensor_dim, const uint16_t in_tensor_ch, const q7_t *ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const q7_t *bias, const uint16_t bias_lshift, const uint16_t out_rshift, q7_t *out_tensor, const uint16_t out_tensor_dim, q15_t *in_tmp_buf, q7_t *tmp_buf)
```

描述：

该函数执行快速的有符号 8 位整数卷积，以及对输出执行基于移位的量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小

- [in] stride 卷积步距
- [in] *bias 指向偏置向量的指针
- [in] bias_lshift 偏置的左移量
- [in] out_rshift 输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针
- [in] *tmp_buf 空指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 是 4 的倍数和 out_tensor_ch 是 2 的倍数的约束条件，则返回-1。

示例：

```
//Convolve a 12x12x20 input tensor with a 5x5 kernel and generate an 8x8x50
//output tensor. Let both dimensions' padding be 0 and their stride be 1.
#define IN_DIM 12
#define IN_CH 20
#define KER_DIM 5
#define PAD 0
#define STRIDE 1
#define BIAS_LSHIFT 6 //Scale up the bias by 2^6
#define OUT_RSHIFT 10 //Scale down the output tensor by 1/2^10
#define OUT_CH 50
#define OUT_DIM 8
q7_t in_data[IN_CH * IN_DIM * IN_DIM] = {...};
q7_t weight[IN_CH * KER_DIM * KER_DIM * OUT_CH] = {...};
q7_t bias[OUT_CH] = {...};
q15_t in_tmp_buf[2 * IN_CH * KER_DIM * KER_DIM] = {0};
q7_t out_data[OUT_CH * OUT_DIM * OUT_DIM];
riscv_nn_conv_HWC_s8_s8_s8_sft_bias_fast (in_data, IN_DIM, IN_CH, weight, OUT_CH, KER_DIM, PAD, STRIDE, bias, BIAS_LSHIFT, OUT_RSHIFT, out_data, OUT_DIM, in_tmp_buf, NULL);
```

3.4.7 riscv_nn_conv_HWC_s8_s8_s8_sft_bias_fast_any

原型：

int32_t riscv_nn_conv_HWC_s8_s8_s8_sft_bias_fast_any (const q7_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const uint16_t stride_y, const q7_t * bias, const uint16_t bias_lshift, const uint16_t out_rshift, q7_t * out_tensor, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, q15_t * in_tmp_buf, q7_t * tmp_buf)

描述：

该函数在任意 x 和 y 维度执行快速的有符号 8 位整数卷积，以及对输出执行基于移位的量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] *bias 指向偏置向量的指针
- [in] bias_lshift 偏置的左移量
- [in] out_rshift 输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

- [in] *tmp_buf 空指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 是 4 的倍数和 out_tensor_ch 是 2 的倍数的约束条件，则返回-1。

示例：

```
//Convolve a 160x120x20 input tensor with a 3x5 kernel and generate an 80x59x8
//output tensor. Let both dimensions' padding be 1 and their stride be 2.
#define IN_X 160
#define IN_Y 120
#define IN_CH 20
#define OUT_CH 8
#define KER_DIM_X 3
#define KER_DIM_Y 5
#define PAD_X 1
#define PAD_Y 1
#define STRIDE_X 2
#define STRIDE_Y 2
#define BIAS_LSHIFT 6 //Scale up the bias by 2^6
#define OUT_RSHIFT 9 //Scale down the output tensor by 1/2^9
#define OUT_X 80
#define OUT_Y 59
q7_t in_data[IN_CH * IN_X * IN_Y] = {...};
q7_t weight[IN_CH * KER_DIM_X * KER_DIM_Y * OUT_CH] = {...};
q7_t bias[OUT_CH] = {...};
q15_t in_tmp_buf[2 * IN_CH * KER_DIM_X * KER_DIM_Y] = {0};
q7_t out_data[OUT_CH * OUT_Y * OUT_X];
riscv_nn_conv_HWC_s8_s8_s8_sft_bias_fast_any (in_data, IN_W, IN_Y , IN_CH, weight, OUT_CH, KER_DIM_X, KER_DIM_Y, PAD_X, PAD_Y, STRIDE_X, STRIDE_Y, bias, BIAS_LSHIFT, OUT_RSHIFT, out_data, OUT_X, OUT_Y, in_tmp_buf, NULL);
```

3.4.8 riscv_nn_conv_HWC_s16_s16_s16_sft_bias

原型：

int32_t riscv_nn_conv_HWC_s16_s16_s16_sft_bias (const q15_t * in_tensor, const uint16_t in_tensor_dim, const uint16_t in_tensor_ch, const q15_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const q15_t * bias, const uint16_t bias_lshift, const uint16_t out_rshift, q15_t * out_tensor, const uint16_t out_tensor_dim, q15_t * in_tmp_buf, q7_t * tmp_buf)

描述：

该函数执行有符号 16 位整数卷积，以及对输出执行基于移位的量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小
- [in] stride 卷积步距
- [in] *bias 指向偏置向量的指针
- [in] bias_lshift 偏置的左移量
- [in] out_rshift 输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针
- [in] *tmp_buf 空指针

返回值：

0

示例：

//Convolve a 28x28x1 input tensor with a 5x5 kernel and generate a 24x24x20

//output tensor. Let both dimensions' padding be 0 and their stride be 1.

```
#define IN_DIM 28
#define IN_CH 1
#define KER_DIM 5
#define PAD 0
#define STRIDE 1
#define BIAS_LSHIFT 6 //Scale up the bias by 2^6
#define OUT_RSHIFT 10 //Scale down the output tensor by 1/2^10
#define OUT_CH 20
#define OUT_DIM 24
q15_t input_data[IN_CH * IN_DIM * IN_DIM] = {...};
q15_t weight[IN_CH * KER_DIM * KER_DIM * OUT_CH] = {...};
q15_t bias[OUT_CH] = {...};
q15_t in_tmp_buf[IN_CH * KER_DIM * KER_DIM] = {0};
q15_t out_data[OUT_CH * OUT_DIM * OUT_DIM];
riscv_nn_conv_HWC_s16_s16_s16_sft_bias (input_data, IN_DIM, IN_CH, weight, OUT_CH, KER_DIM, PAD, STRIDE, bias, BIAS_LSHIFT, OUT_RSHIFT, out_data, OUT_DIM, in_tmp_buf, NULL);
```

3.4.9 riscv_nn_conv_HWC_s16_s16_s16_sft_bias_fast

原型：

```
int32_t riscv_nn_conv_HWC_s16_s16_s16_sft_bias_fast (const q15_t * in_tensor, const uint16_t in_tensor_dim, const uint16_t in_tensor_ch, const q15_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const q15_t * bias, const uint16_t bias_lshift, const uint16_t out_rshift, q15_t * out_tensor, const uint16_t out_tensor_dim, q15_t * in_tmp_buf, q7_t * tmp_buf)
```

描述：

该函数执行快速的有符号 16 位整数卷积，以及对输出执行基于移位的量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针

- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小
- [in] stride 卷积步距
- [in] *bias 指向偏置向量的指针
- [in] bias_lshift 偏置的左移量
- [in] out_rshift 输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针
- [in] *tmp_buf 空指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 和 out_tensor_ch 都是 2 的倍数的约束条件，则返回-1。

示例：

```
//Convolve a 28x28x4 input tensor with a 5x5 kernel and generate a 24x24x8 output
//tensor. Let both dimensions' padding be 0 and their stride be 1.
#define IN_DIM 28
#define IN_CH 4
#define KER_DIM 5
#define PAD 0
#define STRIDE 1
#define BIAS_LSHIFT 6
#define OUT_RSHIFT 10
#define OUT_CH 8
#define OUT_DIM 24
q15_t in_data[IN_CH * IN_DIM * IN_DIM] = {...};
q15_t weight[IN_CH * KER_DIM * KER_DIM * OUT_CH] = {...};
q15_t bias[OUT_CH] = {...};
q15_t in_tmp_buf[IN_CH * KER_DIM * KER_DIM] = {0};
q15_t out_data[OUT_CH * OUT_DIM * OUT_DIM];
riscv_nn_conv_HWC_s16_s16_s16_sft_bias_fast (in_data, IN_DIM,
```

IN_CH, weight, OUT_CH, KER_DIM, PAD, STRIDE, bias, BIAS_LSHIFT, OUT_RSHIFT, out_data, OUT_DIM, in_tmp_buf, NULL);

3.4.10 riscv_nn_conv_HWC_s16_s16_s16_sft_bias_fast_any

原型：

int32_t riscv_nn_conv_HWC_s16_s16_s16_sft_bias_fast_any (const q15_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const q15_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const uint16_t stride_y, const q15_t * bias, const uint16_t bias_lshift, const uint16_t out_rshift, q15_t * out_tensor, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, q15_t * in_tmp_buf, q7_t * tmp_buf)

描述：

该函数在任意 x 和 y 维度执行快速的有符号 16 位整数卷积，以及对输出执行基于移位的量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] *bias 指向偏置向量的指针
- [in] bias_lshift 偏置的左移量
- [in] out_rshift 输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim_x 输出张量的 x 维度

- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针
- [in] *tmp_buf 空指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 和 out_tensor_ch 都是 2 的倍数的约束条件，则返回-1。

示例：

```
//Convolve a 160x120x20 input tensor with a 3x5 kernel and generate an 80x59x8
//output tensor. Let both dimensions' padding be 1 and their stride be 2.
#define IN_X 160
#define IN_Y 120
#define IN_CH 20
#define OUT_CH 8
#define KER_DIM_X 3
#define KER_DIM_Y 5
#define PAD_X 1
#define PAD_Y 1
#define STRIDE_X 2
#define STRIDE_Y 2
#define BIAS_LSHIFT 6 //Scale up the bias by 2^6
#define OUT_RSHIFT 9 //Scale down the output tensor by 1/2^9
#define OUT_X 80
#define OUT_Y 59
q15_t in_data[IN_CH * IN_X * IN_Y] = {...};
q15_t weight[IN_CH * KER_DIM_X * KER_DIM_Y * OUT_CH] = {...};
q15_t bias[OUT_CH] = {...};
q15_t in_tmp_buf[2 * IN_CH * KER_DIM_X * KER_DIM_Y] = {0};
q15_t out_data[OUT_CH * OUT_X * OUT_Y];
riscv_nn_conv_HWC_s16_s16_s16_sft_bias_fast_any (in_data, IN_X, IN_Y , IN_CH, weight, OUT_CH, KER_DIM_X, KER_DIM_Y, PAD_X, PAD_Y, STRIDE_X, STRIDE_Y, bias, BIAS_LSHIFT, OUT_RSHIFT, out_data, OUT_X, OUT_Y, in_tmp_buf, NULL);
```

3.4.11 riscv_nn_conv_dw_HWC_s8_s8_s8_sft_bias

原型：

int32_t riscv_nn_conv_dw_HWC_s8_s8_s8_sft_bias (const q7_t * in_tensor, const uint16_t in_tensor_dim, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const q7_t * bias, const uint16_t bias_lshift, const uint16_t out_rshift, q7_t * out_tensor, const uint16_t out_tensor_dim, q15_t * in_tmp_buf, q7_t * tmp_buf)

描述：

该函数执行有符号 8 位整数深度卷积，以及对输出执行基于移位的量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小
- [in] stride 卷积步距
- [in] *bias 指向偏置向量的指针
- [in] bias_lshift 偏置的左移量
- [in] out_rshift 输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针
- [in] *tmp_buf 空指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 等于 out_tensor_ch 的约束条件，则返回-1。

示例：

```
//Convolve a 11x11x28 input tensor with a 3x3 kernel and generate a 9x9x48 output
//tensor. Let both dimensions' padding be 0 and their stride be 1.
#define IN_DIM 11
#define IN_CH 28
#define OUT_CH 48
#define KER_DIM 3
#define PAD 0
#define STRIDE 1
#define OUT_RSHIFT 7
#define OUT_DIM 9
q7_t in_data[IN_CH * IN_DIM * IN_DIM] = {...};
q7_t weight[IN_CH * KER_DIM * KER_DIM * IN_CH] = {...};
q7_t bias[IN_CH] = {...};
q15_t in_tmp_buf[2 * OUT_CH * KER_DIM * KER_DIM] = {0};
q7_t out_data[OUT_CH * OUT_DIM * OUT_DIM];
riscv_nn_conv_dw_HWC_s8_s8_s8_sft_bias (in_data, IN_DIM, IN_CH, weight, OUT_CH, KER_DIM, PAD, STRIDE, bias, 0, OUT_RSHIFT, out_data, OUT_DIM, in_tmp_buf, NULL);
```

3.4.12 riscv_nn_conv_dw_HWC_s8_s8_s8_sft_bias_any

原型：

```
int32_t riscv_nn_conv_dw_HWC_s8_s8_s8_sft_bias_any (const q7_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const uint16_t stride_y, const q7_t * bias, const uint16_t bias_lshift, const uint16_t out_rshift, q7_t * out_tensor, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, q15_t * in_tmp_buf, q7_t * tmp_buf)
```

描述：

该函数在任意 x 和 y 维度执行有符号 8 位整数深度卷积，以及对输出执行基于移位的量化。

参数：

- [in] *in_tensor 指向输入张量的指针

- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] *bias 指向偏置向量的指针
- [in] bias_lshift 偏置的左移量
- [in] out_rshift 输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针
- [in] *tmp_buf 空指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 等于 out_tensor_ch 的约束条件，则返回-1。

示例：

```
//Perform a depth-wise convolution for a 79x59x12 input tensor with a 3x3 kernel
//and generate a 77x57x12 output tensor. Let both dimensions' padding be 0 and
//their stride be 1.
#define IN_DIM_X 79
#define IN_DIM_Y 59
#define IN_CH 12
#define OUT_CH 12
#define KER_DIM 3
```

```
#define PAD 0
#define STRIDE 1
#define BIAS_SHIFT 0
#define OUT_RSHIFT 7
#define OUT_DIM_X 77
#define OUT_DIM_Y 57
q7_t in_data[IN_CH * IN_DIM_X * IN_DIM_Y] = {...};
q7_t weight[IN_CH * KER_DIM * KER_DIM] = {...};
q7_t bias[IN_CH] = {...};
q15_t in_tmp_buf[2 * OUT_CH * KER_DIM * KER_DIM] = {0};
q7_t out_data[OUT_CH * OUT_DIM_X * OUT_DIM_Y];
riscv_nn_conv_dw_HWC_s8_s8_s8_sft_bias_any (in_data,
IN_DIM_X, IN_DIM_Y, IN_CH, weight, OUT_CH, KER_DIM, KER_DIM,
PAD, PAD, STRIDE, STRIDE, bias, BIAS_SHIFT, OUT_RSHIFT, out_data,
OUT_DIM_X, OUT_DIM_Y, in_tmp_buf, NULL);
```

3.4.13 riscv_nn_conv_1x1_HWC_s8_s8_s8_sym_bias_fast_any

原型：

```
int32_t riscv_nn_conv_1x1_HWC_s8_s8_s8_sym_bias_fast_any
(const q7_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t
in_tensor_dim_y, const uint16_t in_tensor_ch, const q7_t * ker_weight,
const uint16_t out_tensor_ch, const uint16_t ker_dim_x, const uint16_t
ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t
stride_x, const uint16_t stride_y, const q31_t * bias, const uint16_t
pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q7_t *
out_tensor, const uint16_t out_tensor_dim_x, const uint16_t
out_tensor_dim_y, q15_t * in_tmp_buf)
```

描述：

该函数在任意 x 和 y 维度对有符号 8 位整数输入/输出执行 1x1 核卷积，以及对输出执行偏置输入和对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针

- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] *bias 指向偏置向量的指针
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足约束条件，则返回-1，详细信息参照下面的注释。

注！

1. 该函数的输入约束条件：
   - in_tensor_ch 4 的倍数
   - out_tensor_ch 2 的倍数
   - ker_dim_x 1
   - ker_dim_y 1
   - pad_x 0
   - pad_y 0
   - stride_x 1
   - stride_y 1
2. 输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.14 riscv_nn_conv_1x1_HWC_s8_s16_s8_sym_bias_fast_any

原型：

int32_t riscv_nn_conv_1x1_HWC_s8_s16_s8_sym_bias_fast_any (const q7_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const uint16_t stride_y, const q31_t * bias, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q15_t * out_tensor, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, q15_t * in_tmp_buf)

描述：

该函数在任意 x 和 y 维度对有符号 8 位整数输入和有符号 16 位整数输出执行 1x1 核卷积，以及对输出执行偏置输入和对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] *bias 指向偏置向量的指针
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度

- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足约束条件，则返回-1（详细信息请参照下面的注释）。

注！

1. 该函数的输入约束条件：
   - in_tensor_ch 4 的倍数
   - out_tensor_ch 2 的倍数
   - ker_dim_x 1
   - ker_dim_y 1
   - pad_x 0
   - pad_y 0
   - stride_x 1
   - stride_y 1
2. 输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.15 riscv_nn_conv_1x1_HWC_u8_u8_s8_sym_bias_fast_any

原型：

int32_t riscv_nn_conv_1x1_HWC_u8_u8_s8_sym_bias_fast_any (const u8_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const uint16_t stride_y, const q31_t * bias, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, u8_t * out_tensor, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, q15_t * in_tmp_buf)

描述：

该函数在任意 x 和 y 维度对无符号 8 位整数输入/输出执行 1x1 核卷积，以及对输出执行偏置输入和对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度

- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] *bias 指向偏置向量的指针
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足约束条件，则返回-1（详细信息请参照下面的注释）。

注！

1. 该函数的输入约束条件：
   - in_tensor_ch 4 的倍数
   - out_tensor_ch 2 的倍数
   - ker_dim_x 1
   - ker_dim_y 1
   - pad_x 0
   - pad_y 0
   - stride_x 1
   - stride_y 1
2. 输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.16 riscv_nn_conv_1x1_HWC_u8_s8_s8_sym_bias_fast_any

原型：

int32_t riscv_nn_conv_1x1_HWC_u8_s8_s8_sym_bias_fast_any (const u8_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const uint16_t stride_y, const q31_t * bias, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q7_t * out_tensor, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, q15_t * in_tmp_buf)

描述：

该函数在任意 x 和 y 维度对无符号 8 位整数输入和有符号 8 位整数输出执行 1x1 核卷积，以及对输出执行偏置输入和对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] *bias 指向偏置向量的指针
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度

- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足约束条件，则返回-1，详细信息请参照下面的注释。

注！

1. 该函数的输入约束条件：
   - in_tensor_ch 4 的倍数
   - out_tensor_ch 2 的倍数
   - ker_dim_x 1
   - ker_dim_y 1
   - pad_x 0
   - pad_y 0
   - stride_x 1
   - stride_y 1
2. 输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.17 riscv_nn_conv_1x1_HWC_u8_s16_s8_sym_bias_fast_any

原型：

int32_t riscv_nn_conv_1x1_HWC_u8_s16_s8_sym_bias_fast_any (const u8_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const uint16_t stride_y, const q31_t * bias, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q15_t * out_tensor, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, q15_t * in_tmp_buf)

描述：

该函数在任意 x 和 y 维度对无符号 8 位整数输入和有符号 16 位整数输出执行 1x1 核卷积，以及对输出执行偏置输入和对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度

- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] *bias 指向偏置向量的指针
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足约束条件，则返回-1，详细信息请参照下面的注释。

注！

1. 该函数的输入约束条件：
   - in_tensor_ch 4 的倍数
   - out_tensor_ch 2 的倍数
   - ker_dim_x 1
   - ker_dim_y 1
   - pad_x 0
   - pad_y 0
   - stride_x 1
   - stride_y 1
2. 输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.18 riscv_nn_conv_1x1_HWC_s8_s8_s8_sym_fast_any

原型：

int32_t riscv_nn_conv_1x1_HWC_s8_s8_s8_sym_fast_any (const q7_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const uint16_t stride_y, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q7_t * out_tensor, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, q15_t * in_tmp_buf)

描述：

该函数在任意 x 和 y 维度对有符号 8 位整数输入/输出执行 1x1 核卷积，以及对输出执行对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足约束条件，则返回-1，详细信息请参照下面的注释。

注！

1. 该函数的输入约束条件：
   - in_tensor_ch 4 的倍数
   - out_tensor_ch 2 的倍数
   - ker_dim_x 1
   - ker_dim_y 1
   - pad_x 0
   - pad_y 0
   - stride_x 1
   - stride_y 1
2. 输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.19 riscv_nn_conv_1x1_HWC_s8_s16_s8_sym_fast_any

原型：

int32_t riscv_nn_conv_1x1_HWC_s8_s16_s8_sym_fast_any (const q7_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const uint16_t stride_y, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q15_t * out_tensor, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, q15_t * in_tmp_buf)

描述：

该函数在任意 x 和 y 维度对有符号 8 位整数输入和有符号 16 位整数输出执行 1x1 核卷积，以及对输出执行对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针

- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足约束条件，则返回-1，详细信息请参照下面的注释。

注！

1. 该函数的输入约束条件：
   - in_tensor_ch 4 的倍数
   - out_tensor_ch 2 的倍数
   - ker_dim_x 1
   - ker_dim_y 1
   - pad_x 0
   - pad_y 0
   - stride_x 1
   - stride_y 1
2. 输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.20 riscv_nn_conv_1x1_HWC_u8_u8_s8_sym_fast_any

原型：

int32_t riscv_nn_conv_1x1_HWC_u8_u8_s8_sym_fast_any (const u8_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const uint16_t stride_y, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, u8_t * out_tensor, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, q15_t * in_tmp_buf)

描述：

该函数在任意 x 和 y 维度对无符号 8 位整数输入/输出执行 1x1 核卷积，以及对输出执行对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足约束条件，则返回-1（详细信息请参照下面的注释）。

注！

1. 该函数的输入约束条件：
   - in_tensor_ch 4 的倍数
   - out_tensor_ch 2 的倍数
   - ker_dim_x 1
   - ker_dim_y 1
   - pad_x 0
   - pad_y 0
   - stride_x 1
   - stride_y 1
2. 输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.21 riscv_nn_conv_1x1_HWC_u8_s8_s8_sym_fast_any

原型：

int32_t riscv_nn_conv_1x1_HWC_u8_s8_s8_sym_fast_any (const u8_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const uint16_t stride_y, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q7_t * out_tensor, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, q15_t * in_tmp_buf)

描述：

该函数在任意 x 和 y 维度对无符号 8 位整数输入和有符号 8 位整数输出执行 1x1 核卷积，以及对输出执行对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针

- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足约束条件，则返回-1，详细信息请参照下面的注释。

注！

1. 该函数的输入约束条件：
   - in_tensor_ch 4 的倍数
   - out_tensor_ch 2 的倍数
   - ker_dim_x 1
   - ker_dim_y 1
   - pad_x 0
   - pad_y 0
   - stride_x 1
   - stride_y 1
2. 输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.22 riscv_nn_conv_1x1_HWC_u8_s16_s8_sym_fast_any

原型：

int32_t riscv_nn_conv_1x1_HWC_u8_s16_s8_sym_fast_any (const u8_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const uint16_t stride_y, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q15_t * out_tensor, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, q15_t * in_tmp_buf)

描述：

该函数在任意 x 和 y 维度对无符号 8 位整数输入和有符号 16 位整数输出执行 1x1 核卷积，以及对输出执行对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足约束条件，则返回-1，详细信息请参照下面的注释。

注！

1. 该函数的输入约束条件：
   - in_tensor_ch 4 的倍数
   - out_tensor_ch 2 的倍数
   - ker_dim_x 1
   - ker_dim_y 1
   - pad_x 0
   - pad_y 0
   - stride_x 1
   - stride_y 1
2. 输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.23 riscv_nn_conv_HWC_s8_s8_s8_RGB_sym_bias_fast

原型：

int32_t riscv_nn_conv_HWC_s8_s8_s8_RGB_sym_bias_fast (const q7_t * in_tensor, const uint16_t in_tensor_dim, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const q31_t * bias, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q7_t * out_tensor, const uint16_t out_tensor_dim, q15_t * in_tmp_buf, q15_t * wt_tmp_buf)

描述：

关于 RGB 图像，该函数对有符号 8 位整数输入/输出执行快速卷积，以及对输出执行偏置输入和对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小

- [in] stride 卷积步距
- [in] *bias 指向偏置向量的指针
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针
- [in] *wt_tmp_buf 指向卷积核权重临时缓存区的指针

返回值：

0

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.24 riscv_nn_conv_HWC_s8_s16_s8_RGB_sym_bias_fast

原型：

int32_t riscv_nn_conv_HWC_s8_s16_s8_RGB_sym_bias_fast (const q7_t * in_tensor, const uint16_t in_tensor_dim, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const q31_t * bias, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q15_t * out_tensor, const uint16_t out_tensor_dim, q15_t * in_tmp_buf, q15_t * wt_tmp_buf)

描述：

关于 RGB 图像，该函数对有符号 8 位整数输入和有符号 16 位整数输出执行快速卷积，以及对输出执行偏置输入和对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小
- [in] stride 卷积步距

- [in] *bias 指向偏置向量的指针
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针
- [in] *wt_tmp_buf 指向卷积核权重临时缓存区的指针

返回值：

0

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.25 riscv_nn_conv_HWC_u8_u8_s8_RGB_sym_bias_fast

原型：

int32_t riscv_nn_conv_HWC_u8_u8_s8_RGB_sym_bias_fast (const u8_t * in_tensor, const uint16_t in_tensor_dim, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const q31_t * bias, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, u8_t * out_tensor, const uint16_t out_tensor_dim, q15_t * in_tmp_buf, q15_t * wt_tmp_buf)

描述：

关于 RGB 图像，该函数对无符号 8 位整数输入/输出执行快速卷积，以及对输出执行偏置输入和对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小
- [in] stride 卷积步距
- [in] *bias 指向偏置向量的指针

- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针
- [in] *wt_tmp_buf 指向卷积核权重临时缓存区的指针

返回值：

0

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.26 riscv_nn_conv_HWC_u8_s8_s8_RGB_sym_bias_fast

原型：

int32_t riscv_nn_conv_HWC_u8_s8_s8_RGB_sym_bias_fast (const u8_t * in_tensor, const uint16_t in_tensor_dim, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const q31_t * bias, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q7_t * out_tensor, const uint16_t out_tensor_dim, q15_t * in_tmp_buf, q15_t * wt_tmp_buf)

描述：

关于 RGB 图像，该函数对无符号 8 位整数输入和有符号 8 位整数输出执行快速卷积，以及对输出执行偏置输入和对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小
- [in] stride 卷积步距
- [in] *bias 指向偏置向量的指针
- [in] pre_rshift 缩放前输出的右移量

- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针
- [in] *wt_tmp_buf 指向卷积核权重临时缓存区的指针

返回值：

0

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.27 riscv_nn_conv_HWC_u8_s16_s8_RGB_sym_bias_fast

原型：

int32_t riscv_nn_conv_HWC_u8_s16_s8_RGB_sym_bias_fast (const u8_t * in_tensor, const uint16_t in_tensor_dim, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const q31_t * bias, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q15_t * out_tensor, const uint16_t out_tensor_dim, q15_t * in_tmp_buf, q15_t * wt_tmp_buf)

描述：

关于 RGB 图像，该函数对无符号 8 位整数输入和有符号 16 位整数输出执行快速卷积，以及对输出执行偏置输入和对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小
- [in] stride 卷积步距
- [in] *bias 指向偏置向量的指针
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值

- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针
- [in] *wt_tmp_buf 指向卷积核权重临时缓存区的指针

返回值：

0

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.28 riscv_nn_conv_HWC_s8_s8_s8_RGB_sym_fast

原型：

int32_t riscv_nn_conv_HWC_s8_s8_s8_RGB_sym_fast (const q7_t * in_tensor, const uint16_t in_tensor_dim, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q7_t * out_tensor, const uint16_t out_tensor_dim, q15_t * in_tmp_buf, q15_t * wt_tmp_buf)

描述：

关于 RGB 图像，该函数对有符号 8 位整数输入/输出执行快速卷积，以及对输出执行对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小
- [in] stride 卷积步距
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针

- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针
- [in] *wt_tmp_buf 指向卷积核权重临时缓存区的指针

返回值：

0

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.29 riscv_nn_conv_HWC_s8_s16_s8_RGB_sym_fast

原型：

int32_t riscv_nn_conv_HWC_s8_s16_s8_RGB_sym_fast (const q7_t * in_tensor, const uint16_t in_tensor_dim, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q15_t * out_tensor, const uint16_t out_tensor_dim, q15_t * in_tmp_buf, q15_t * wt_tmp_buf)

描述：

关于 RGB 图像，该函数对有符号 8 位整数输入和有符号 16 位整数输出执行快速卷积，以及对输出执行对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小
- [in] stride 卷积步距
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

- [in] *wt_tmp_buf 指向卷积核权重临时缓存区的指针

返回值：

0

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.30 riscv_nn_conv_HWC_u8_u8_s8_RGB_sym_fast

原型：

int32_t riscv_nn_conv_HWC_u8_u8_s8_RGB_sym_fast (const u8_t * in_tensor, const uint16_t in_tensor_dim, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, u8_t * out_tensor, const uint16_t out_tensor_dim, q15_t * in_tmp_buf, q15_t * wt_tmp_buf)

描述：

关于 RGB 图像，该函数对无符号 8 位整数输入/输出执行快速卷积，以及对输出执行对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小
- [in] stride 卷积步距
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针
- [in] *wt_tmp_buf 指向卷积核权重临时缓存区的指针

返回值：

0

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.31 riscv_nn_conv_HWC_u8_s8_s8_RGB_sym_fast

原型：

int32_t riscv_nn_conv_HWC_u8_s8_s8_RGB_sym_fast (const u8_t * in_tensor, const uint16_t in_tensor_dim, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q7_t * out_tensor, const uint16_t out_tensor_dim, q15_t * in_tmp_buf, q15_t * wt_tmp_buf)

描述：

关于 RGB 图像，该函数对无符号 8 位整数输入和有符号 8 位整数输出执行快速卷积，以及对输出执行对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小
- [in] stride 卷积步距
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针
- [in] *wt_tmp_buf 指向卷积核权重临时缓存区的指针

返回值：

0

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.32 riscv_nn_conv_HWC_u8_s16_s8_RGB_sym_fast

原型：

int32_t riscv_nn_conv_HWC_u8_s16_s8_RGB_sym_fast (const u8_t * in_tensor, const uint16_t in_tensor_dim, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q15_t * out_tensor, const uint16_t out_tensor_dim, q15_t * in_tmp_buf, q15_t * wt_tmp_buf)

描述：

关于 RGB 图像，该函数对无符号 8 位整数输入和有符号 16 位整数输出执行快速卷积，以及对输出执行对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小
- [in] stride 卷积步距
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针
- [in] *wt_tmp_buf 指向卷积核权重临时缓存区的指针

返回值：

0

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.33 riscv_nn_conv_HWC_s8_s8_s8_sym_bias_fast

原型：

int32_t riscv_nn_conv_HWC_s8_s8_s8_sym_bias_fast (const q7_t *in_tensor, const uint16_t in_tensor_dim, const uint16_t in_tensor_ch, const q7_t *ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const q31_t *bias, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q7_t * out_tensor, const uint16_t out_tensor_dim, q15_t *in_tmp_buf)

描述：

该函数对有符号 8 位整数输入/输出执行快速卷积，以及对输出执行偏置输入和对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小
- [in] stride 卷积步距
- [in] *bias 指向偏置向量的指针
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 是 4 的倍数和 out_tensor_ch 是 2 的倍数的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.34 riscv_nn_conv_HWC_s8_s16_s8_sym_bias_fast

原型：

int32_t riscv_nn_conv_HWC_s8_s16_s8_sym_bias_fast (const q7_t *in_tensor, const uint16_t in_tensor_dim, const uint16_t in_tensor_ch, const q7_t *ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const q31_t *bias, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q15_t *out_tensor, const uint16_t out_tensor_dim, q15_t *in_tmp_buf)

描述：

该函数对有符号 8 位整数输入和有符号 16 位整数输出执行快速卷积，以及对输出执行偏置输入和对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小
- [in] stride 卷积步距
- [in] *bias 指向偏置向量的指针
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 是 4 的倍数和 out_tensor_ch 是 2 的倍数的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.35 riscv_nn_conv_HWC_u8_u8_s8_sym_bias_fast

原型：

int32_t riscv_nn_conv_HWC_u8_u8_s8_sym_bias_fast (const u8_t * in_tensor, const uint16_t in_tensor_dim, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const q31_t * bias, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, u8_t * out_tensor, const uint16_t out_tensor_dim, q15_t * in_tmp_buf)

描述：

该函数对无符号 8 位整数输入/输出执行快速卷积，以及对输出执行偏置输入和对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小
- [in] stride 卷积步距
- [in] *bias 指向偏置向量的指针
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 是 4 的倍数和 out_tensor_ch 是 2 的倍数的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.36 riscv_nn_conv_HWC_u8_s8_s8_sym_bias_fast

原型：

int32_t riscv_nn_conv_HWC_u8_s8_s8_sym_bias_fast (const u8_t * in_tensor, const uint16_t in_tensor_dim, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const q31_t * bias, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q7_t * out_tensor, const uint16_t out_tensor_dim, q15_t * in_tmp_buf)

描述：

该函数对无符号 8 位整数输入和有符号 8 位整数输出执行快速卷积，以及对输出执行偏置输入和对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小
- [in] stride 卷积步距
- [in] *bias 指向偏置向量的指针
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 是 4 的倍数和 out_tensor_ch 是 2 的倍数的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.37 riscv_nn_conv_HWC_u8_s16_s8_sym_bias_fast

原型：

int32_t riscv_nn_conv_HWC_u8_s16_s8_sym_bias_fast (const u8_t * in_tensor, const uint16_t in_tensor_dim, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const q31_t * bias, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q15_t * out_tensor, const uint16_t out_tensor_dim, q15_t * in_tmp_buf)

描述：

该函数对无符号 8 位整数输入和有符号 16 位整数输出执行快速卷积，以及对输出执行偏置输入和对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小
- [in] stride 卷积步距
- [in] *bias 指向偏置向量的指针
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 是 4 的倍数和 out_tensor_ch 是 2 的倍数的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.38 riscv_nn_conv_HWC_s8_s8_s8_sym_fast

原型：

int32_t riscv_nn_conv_HWC_s8_s8_s8_sym_fast (const q7_t * in_tensor, const uint16_t in_tensor_dim, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q7_t * out_tensor, const uint16_t out_tensor_dim, q15_t * in_tmp_buf)

描述：

该函数对有符号 8 位整数输入/输出执行快速卷积，以及对输出执行对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小
- [in] stride 卷积步距
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 是 4 的倍数和 out_tensor_ch 是 2 的倍数的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.39 riscv_nn_conv_HWC_s8_s16_s8_sym_fast

原型：

int32_t riscv_nn_conv_HWC_s8_s16_s8_sym_fast (const q7_t * in_tensor, const uint16_t in_tensor_dim, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q15_t * out_tensor, const uint16_t out_tensor_dim, q15_t * in_tmp_buf)

描述：

该函数对有符号 8 位整数输入和有符号 16 位整数输出执行快速卷积，以及对输出执行对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小
- [in] stride 卷积步距
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 是 4 的倍数和 out_tensor_ch 是 2 的倍数的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.40 riscv_nn_conv_HWC_u8_u8_s8_sym_fast

原型：

int32_t riscv_nn_conv_HWC_u8_u8_s8_sym_fast (const u8_t * in_tensor, const uint16_t in_tensor_dim, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, u8_t * out_tensor, const uint16_t out_tensor_dim, q15_t * in_tmp_buf)

描述：

该函数对无符号 8 位整数输入/输出执行快速卷积，以及对输出执行对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小
- [in] stride 卷积步距
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 是 4 的倍数和 out_tensor_ch 是 2 的倍数的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.41 riscv_nn_conv_HWC_u8_s8_s8_sym_fast

原型：

int32_t riscv_nn_conv_HWC_u8_s8_s8_sym_fast (const u8_t * in_tensor, const uint16_t in_tensor_dim, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q7_t * out_tensor, const uint16_t out_tensor_dim, q15_t * in_tmp_buf)

描述：

该函数对无符号 8 位整数输入和有符号 8 位整数输出执行快速卷积，以及对输出执行对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小
- [in] stride 卷积步距
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 是 4 的倍数和 out_tensor_ch 是 2 的倍数的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.42 riscv_nn_conv_HWC_u8_s16_s8_sym_fast

原型：

int32_t riscv_nn_conv_HWC_u8_s16_s8_sym_fast (const u8_t * in_tensor, const uint16_t in_tensor_dim, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q15_t * out_tensor, const uint16_t out_tensor_dim, q15_t * in_tmp_buf)

描述：

该函数对无符号 8 位整数输入和有符号 16 位整数输出执行快速卷积，以及对输出执行对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小
- [in] stride 卷积步距
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 是 4 的倍数和 out_tensor_ch 是 2 的倍数的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.43 riscv_nn_conv_HWC_s8_s8_s8_sym_bias_fast_any

原型：

int32_t riscv_nn_conv_HWC_s8_s8_s8_sym_bias_fast_any (const q7_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const uint16_t stride_y, const q31_t * bias, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q7_t * out_tensor, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, q15_t * in_tmp_buf)

描述：

该函数在任意 x 和 y 维度对有符号 8 位整数输入/输出执行快速卷积，以及对输出执行偏置输入和对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] *bias 指向偏置向量的指针
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度

- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 是 4 的倍数和 out_tensor_ch 是 2 的倍数的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.44 riscv_nn_conv_HWC_s8_s16_s8_sym_bias_fast_any

原型：

int32_t riscv_nn_conv_HWC_s8_s16_s8_sym_bias_fast_any (const q7_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const uint16_t stride_y, const q31_t * bias, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q15_t * out_tensor, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, q15_t * in_tmp_buf)

描述：

该函数在任意 x 和 y 维度对有符号 8 位整数输入和有符号 16 位整数输出执行快速卷积，以及对输出执行偏置输入和对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] *bias 指向偏置向量的指针

- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 是 4 的倍数和 out_tensor_ch 是 2 的倍数的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.45 riscv_nn_conv_HWC_u8_u8_s8_sym_bias_fast_any

原型：

int32_t riscv_nn_conv_HWC_u8_u8_s8_sym_bias_fast_any (const u8_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const uint16_t stride_y, const q31_t * bias, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, u8_t * out_tensor, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, q15_t * in_tmp_buf)

描述：

该函数在任意 x 和 y 维度对无符号 8 位整数输入/输出执行快速卷积，以及对输出执行偏置输入和对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度

- [in] ker_dim_y 卷积核的 y 维度
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] *bias 指向偏置向量的指针
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 是 4 的倍数和 out_tensor_ch 是 2 的倍数的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.46 riscv_nn_conv_HWC_u8_s8_s8_sym_bias_fast_any

原型：

int32_t riscv_nn_conv_HWC_u8_s8_s8_sym_bias_fast_any (const u8_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const uint16_t stride_y, const q31_t * bias, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q7_t * out_tensor, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, q15_t * in_tmp_buf)

描述：

该函数在任意 x 和 y 维度对无符号 8 位整数输入和有符号 8 位整数输出执行快速卷积，以及对输出执行偏置输入和对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针

- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] *bias 指向偏置向量的指针
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 是 4 的倍数和 out_tensor_ch 是 2 的倍数的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.47 riscv_nn_conv_HWC_u8_s16_s8_sym_bias_fast_any

原型：

int32_t riscv_nn_conv_HWC_u8_s16_s8_sym_bias_fast_any (const u8_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const uint16_t stride_y, const q31_t * bias, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q15_t *

out_tensor, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, q15_t * in_tmp_buf)

描述：

该函数在任意 x 和 y 维度对无符号 8 位整数输入和有符号 16 位整数输出执行快速卷积，以及对输出执行偏置输入和对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] *bias 指向偏置向量的指针
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 是 4 的倍数和 out_tensor_ch 是 2 的倍数的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.48 riscv_nn_conv_HWC_s8_s8_s8_sym_fast_any

原型：

int32_t riscv_nn_conv_HWC_s8_s8_s8_sym_fast_any (const q7_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const uint16_t stride_y, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q7_t * out_tensor, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, q15_t * in_tmp_buf)

描述：

该函数在任意 x 和 y 维度对有符号 8 位整数输入/输出执行快速卷积，以及对输出执行对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 是 4 的倍数和 out_tensor_ch 是 2 的倍数的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.49 riscv_nn_conv_HWC_s8_s16_s8_sym_fast_any

原型：

int32_t riscv_nn_conv_HWC_s8_s16_s8_sym_fast_any (const q7_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const uint16_t stride_y, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q15_t * out_tensor, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, q15_t * in_tmp_buf)

描述：

该函数在任意 x 和 y 维度对有符号 8 位整数输入和有符号 16 位整数输出执行快速卷积，以及对输出执行对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值

- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 是 4 的倍数和 out_tensor_ch 是 2 的倍数的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.50 riscv_nn_conv_HWC_u8_u8_s8_sym_fast_any

原型：

int32_t riscv_nn_conv_HWC_u8_u8_s8_sym_fast_any (const u8_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const uint16_t stride_y, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, u8_t * out_tensor, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, q15_t * in_tmp_buf)

描述：

该函数在任意 x 和 y 维度对无符号 8 位整数输入/输出执行快速卷积，以及对输出执行对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度
- [in] pad_x x 维度的填充大小

- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 是 4 的倍数和 out_tensor_ch 是 2 的倍数的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.51 riscv_nn_conv_HWC_u8_s8_s8_sym_fast_any

原型：

int32_t riscv_nn_conv_HWC_u8_s8_s8_sym_fast_any (const u8_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const uint16_t stride_y, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q7_t * out_tensor, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, q15_t * in_tmp_buf)

描述：

该函数在任意 x 和 y 维度对无符号 8 位整数输入和有符号 8 位整数输出执行快速卷积，以及对输出执行对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_ch 输入张量的通道数量

- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 是 4 的倍数和 out_tensor_ch 是 2 的倍数的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.52 riscv_nn_conv_HWC_u8_s16_s8_sym_fast_any

原型：

int32_t riscv_nn_conv_HWC_u8_s16_s8_sym_fast_any (const u8_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const uint16_t stride_y, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q15_t * out_tensor, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, q15_t * in_tmp_buf)

描述：

该函数在任意 x 和 y 维度对无符号 8 位整数输入和有符号 16 位整数输出执行快速卷积，以及对输出执行对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 是 4 的倍数和 out_tensor_ch 是 2 的倍数的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.53 riscv_nn_conv_dw_HWC_s8_s8_s8_sym_bias

原型：

int32_t riscv_nn_conv_dw_HWC_s8_s8_s8_sym_bias (const q7_t * in_tensor, const uint16_t in_tensor_dim, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const q31_t * bias, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t

post_rshift, q7_t * out_tensor, const uint16_t out_tensor_dim, q15_t * in_tmp_buf)

描述：

该函数对有符号 8 位整数输入/输出执行深度卷积，以及对输出执行偏置输入和对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小
- [in] stride 卷积步距
- [in] *bias 指向偏置向量的指针
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 等于 out_tensor_ch 的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.54 riscv_nn_conv_dw_HWC_s8_s16_s8_sym_bias

原型：

int32_t riscv_nn_conv_dw_HWC_s8_s16_s8_sym_bias (const q7_t * in_tensor, const uint16_t in_tensor_dim, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const q31_t * bias, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t

post_rshift, q15_t * out_tensor, const uint16_t out_tensor_dim, q15_t * in_tmp_buf)

描述：

该函数对有符号 8 位整数输入和有符号 16 位整数输出执行深度卷积，以及对输出执行偏置输入和对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小
- [in] stride 卷积步距
- [in] *bias 指向偏置向量的指针
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 等于 out_tensor_ch 的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.55 riscv_nn_conv_dw_HWC_u8_u8_s8_sym_bias

原型：

int32_t riscv_nn_conv_dw_HWC_u8_u8_s8_sym_bias (const u8_t * in_tensor, const uint16_t in_tensor_dim, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const q31_t * bias, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t

post_rshift, u8_t * out_tensor, const uint16_t out_tensor_dim, q15_t * in_tmp_buf)

描述：

该函数对无符号 8 位整数输入/输出执行深度卷积，以及对输出执行偏置输入和对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小
- [in] stride 卷积步距
- [in] *bias 指向偏置向量的指针
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 等于 out_tensor_ch 的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.56 riscv_nn_conv_dw_HWC_u8_s8_s8_sym_bias

原型：

int32_t riscv_nn_conv_dw_HWC_u8_s8_s8_sym_bias (const u8_t * in_tensor, const uint16_t in_tensor_dim, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const q31_t * bias,

const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q7_t * out_tensor, const uint16_t out_tensor_dim, q15_t * in_tmp_buf)

描述：

该函数对无符号 8 位整数输入和有符号 8 位整数输出执行深度卷积，以及对输出执行偏置输入和对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小
- [in] stride 卷积步距
- [in] *bias 指向偏置向量的指针
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 等于 out_tensor_ch 的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.57 riscv_nn_conv_dw_HWC_u8_s16_s8_sym_bias

原型：

int32_t riscv_nn_conv_dw_HWC_u8_s16_s8_sym_bias (const u8_t * in_tensor, const uint16_t in_tensor_dim, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const q31_t * bias,

const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q15_t * out_tensor, const uint16_t out_tensor_dim, q15_t * in_tmp_buf)

描述：

该函数对无符号 8 位整数输入和有符号 16 位整数输出执行深度卷积，以及对输出执行偏置输入和对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小
- [in] stride 卷积步距
- [in] *bias 指向偏置向量的指针
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 等于 out_tensor_ch 的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.58 riscv_nn_conv_dw_HWC_s8_s8_s8_sym

原型：

int32_t riscv_nn_conv_dw_HWC_s8_s8_s8_sym (const q7_t * in_tensor, const uint16_t in_tensor_dim, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t

ker_dim, const uint16_t pad, const uint16_t stride, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q7_t * out_tensor, const uint16_t out_tensor_dim, q15_t * in_tmp_buf)

描述：

该函数对有符号 8 位整数输入/输出执行深度卷积，以及对输出执行对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小
- [in] stride 卷积步距
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 等于 out_tensor_ch 的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.59 riscv_nn_conv_dw_HWC_s8_s16_s8_sym

原型：

int32_t riscv_nn_conv_dw_HWC_s8_s16_s8_sym (const q7_t * in_tensor, const uint16_t in_tensor_dim, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q15_t *

out_tensor, const uint16_t out_tensor_dim, q15_t * in_tmp_buf)

描述：

该函数对有符号 8 位整数输入和有符号 16 位整数输出执行深度卷积，以及对输出执行对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小
- [in] stride 卷积步距
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 等于 out_tensor_ch 的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.60 riscv_nn_conv_dw_HWC_u8_u8_s8_sym

原型：

int32_t riscv_nn_conv_dw_HWC_u8_u8_s8_sym (const u8_t * in_tensor, const uint16_t in_tensor_dim, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, u8_t * out_tensor, const uint16_t out_tensor_dim, q15_t * in_tmp_buf)

描述：

该函数对无符号 8 位整数输入/输出执行深度卷积，以及对输出执行对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小
- [in] stride 卷积步距
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 等于 out_tensor_ch 的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.61 riscv_nn_conv_dw_HWC_u8_s8_s8_sym

原型：

int32_t riscv_nn_conv_dw_HWC_u8_s8_s8_sym (const u8_t * in_tensor, const uint16_t in_tensor_dim, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q7_t * out_tensor, const uint16_t out_tensor_dim, q15_t * in_tmp_buf)

描述：

该函数对无符号 8 位整数输入和有符号 8 位整数输出执行深度卷积，

以及对输出执行对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小
- [in] stride 卷积步距
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 等于 out_tensor_ch 的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.62 riscv_nn_conv_dw_HWC_u8_s16_s8_sym

原型：

int32_t riscv_nn_conv_dw_HWC_u8_s16_s8_sym (const u8_t * in_tensor, const uint16_t in_tensor_dim, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q15_t * out_tensor, const uint16_t out_tensor_dim, q15_t * in_tmp_buf)

描述：

该函数对无符号 8 位整数输入和有符号 16 位整数输出执行深度卷积，以及对输出执行对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小
- [in] stride 卷积步距
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 等于 out_tensor_ch 的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.63 riscv_nn_conv_dw_HWC_s8_s8_s8_sym_bias_any

原型：

int32_t riscv_nn_conv_dw_HWC_s8_s8_s8_sym_bias_any (const q7_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const uint16_t stride_y, const q31_t * bias, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q7_t * out_tensor, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, q15_t * in_tmp_buf)

描述：

该函数在任意 x 和 y 维度对有符号 8 位整数输入/输出执行深度卷积，以及对输出执行偏置输入和对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] *bias 指向偏置向量的指针
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 等于 out_tensor_ch 的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.64 riscv_nn_conv_dw_HWC_s8_s16_s8_sym_bias_any

原型：

int32_t riscv_nn_conv_dw_HWC_s8_s16_s8_sym_bias_any (const q7_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim_x, const uint16_t

ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const uint16_t stride_y, const q31_t * bias, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q15_t * out_tensor, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, q15_t * in_tmp_buf)

描述：

该函数在任意 x 和 y 维度对有符号 8 位整数输入和有符号 16 位整数输出执行深度卷积，以及对输出执行偏置输入和对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] *bias 指向偏置向量的指针
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 等于 out_tensor_ch 的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.65 riscv_nn_conv_dw_HWC_u8_u8_s8_sym_bias_any

原型：

int32_t riscv_nn_conv_dw_HWC_u8_u8_s8_sym_bias_any (const u8_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const uint16_t stride_y, const q31_t * bias, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, u8_t * out_tensor, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, q15_t * in_tmp_buf)

描述：

该函数在任意 x 和 y 维度对无符号 8 位整数输入/输出执行深度卷积，以及对输出执行偏置输入和对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] *bias 指向偏置向量的指针
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针

- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 等于 out_tensor_ch 的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.66 riscv_nn_conv_dw_HWC_u8_s8_s8_sym_bias_any

原型：

int32_t riscv_nn_conv_dw_HWC_u8_s8_s8_sym_bias_any (const u8_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const uint16_t stride_y, const q31_t * bias, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q7_t * out_tensor, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, q15_t * in_tmp_buf)

描述：

该函数在任意 x 和 y 维度对无符号 8 位整数输入和有符号 8 位整数输出执行深度卷积，以及对输出执行偏置输入和对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距

- [in] stride_y y 维度的卷积步距
- [in] *bias 指向偏置向量的指针
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 等于 out_tensor_ch 的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.67 riscv_nn_conv_dw_HWC_u8_s16_s8_sym_bias_any

原型：

int32_t riscv_nn_conv_dw_HWC_u8_s16_s8_sym_bias_any (const u8_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const uint16_t stride_y, const q31_t * bias, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q15_t * out_tensor, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, q15_t * in_tmp_buf)

描述：

该函数在任意 x 和 y 维度对无符号 8 位整数输入和有符号 16 位整数输出执行深度卷积，以及对输出执行偏置输入和对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针

- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] *bias 指向偏置向量的指针
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 等于 out_tensor_ch 的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.68 riscv_nn_conv_dw_HWC_s8_s8_s8_sym_any

原型：

int32_t riscv_nn_conv_dw_HWC_s8_s8_s8_sym_any (const q7_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const uint16_t stride_y, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q7_t * out_tensor, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, q15_t * in_tmp_buf)

描述：

该函数在任意 x 和 y 维度对有符号 8 位整数输入/输出执行深度卷积，以及对输出执行对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 等于 out_tensor_ch 的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.69 riscv_nn_conv_dw_HWC_s8_s16_s8_sym_any

原型：

int32_t riscv_nn_conv_dw_HWC_s8_s16_s8_sym_any (const q7_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const

uint16_t stride_y, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q15_t * out_tensor, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, q15_t * in_tmp_buf)

描述：

该函数在任意 x 和 y 维度对有符号 8 位整数输入和有符号 16 位整数输出执行深度卷积，以及对输出执行对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 等于 out_tensor_ch 的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.70 riscv_nn_conv_dw_HWC_u8_u8_s8_sym_any

原型：

int32_t riscv_nn_conv_dw_HWC_u8_u8_s8_sym_any (const u8_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const uint16_t stride_y, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, u8_t * out_tensor, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, q15_t * in_tmp_buf)

描述：

该函数在任意 x 和 y 维度对无符号 8 位整数输入/输出执行深度卷积，以及对输出执行对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 等于 out_tensor_ch 的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.71 riscv_nn_conv_dw_HWC_u8_s8_s8_sym_any

原型：

int32_t riscv_nn_conv_dw_HWC_u8_s8_s8_sym_any (const u8_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const uint16_t stride_y, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q7_t * out_tensor, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, q15_t * in_tmp_buf)

描述：

该函数在任意 x 和 y 维度对无符号 8 位整数输入和有符号 8 位整数输出执行深度卷积，以及对输出执行对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] pre_rshift 缩放前输出的右移量

- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 等于 out_tensor_ch 的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.72 riscv_nn_conv_dw_HWC_u8_s16_s8_sym_any

原型：

int32_t riscv_nn_conv_dw_HWC_u8_s16_s8_sym_any (const u8_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const uint16_t stride_y, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q15_t * out_tensor, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, q15_t * in_tmp_buf)

描述：

该函数在任意 x 和 y 维度对无符号 8 位整数输入和有符号 16 位整数输出执行深度卷积，以及对输出执行对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度

- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 等于 out_tensor_ch 的约束条件，则返回-1。

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.4.73 riscv_nn_conv_1x1_HWC_s8_s8_s8_asym_bias_fast_any

原型：

int32_t riscv_nn_conv_1x1_HWC_s8_s8_s8_asym_bias_fast_any (const q7_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const uint16_t in_tensor_group, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const uint16_t stride_y, const int32_t * bias, q7_t * out_tensor, const int32_t * out_shift, const int32_t * out_scale, const int32_t out_offset, const int32_t in_offset, const int32_t act_min, const int32_t act_max, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, q15_t * tmp_buf)

描述：

该函数在任意 x 和 y 维度对有符号 8 位整数输入/输出执行快速 1x1 核卷积，以及对输出执行偏置输入和非对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度

- [in] in_tensor_ch 输入张量的通道数量
- [in] in_tensor_group 输入张量的组数
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] *bias 指向偏置向量的指针
- [out] *out_tensor 指向输出张量的指针
- [in] *out_shift 指向移位向量的指针，用于输出张量
- [in] *out_scale 指向缩放向量的指针，用于输出张量
- [in] out_offset 输出张量的偏移值，取值范围应为-128~127
- [in] in_offset 输入张量的偏移值，取值范围应为-127~128
- [in] act_min 限制输出张量的最小值，取值范围应为-128~127
- [in] act_max 限制输出张量的最大值，取值范围应为-128~127
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] *tmp_buf 空指针

返回值：

该函数成功则返回 0；否则，如果输入不满足约束条件，则返回-1（详细信息请参照下面的注释）。

注！

该函数的输入约束条件：

- in_tensor_ch 4 的倍数
- pad_x 0
- pad_y 0
- stride_x 1
- stride_y 1

3.4.74 riscv_nn_conv_1x1_HWC_s8_s8_s8_asym_bias_fast_any_get_buffer_size

原型：

int32_t riscv_nn_conv_1x1_HWC_s8_s8_s8_asym_bias_fast_any_get_buffer_size (const uint16_t in_tensor_ch)

描述：

该函数以字节为单元，返回 riscv_nn_conv_1x1_HWC_s8_s8_s8_asym_bias_fast_any 函数所需的输入临时缓存区大小。

参数：

- [in] in_tensor_ch 输入张量的通道数量

返回值：

该函数返回所需的临时缓存区大小。

3.4.75 riscv_nn_conv_1xn_HWC_s8_s8_s8_asym_bias_any

原型：

int riscv_nn_conv_1xn_HWC_s8_s8_s8_asym_bias_any (const q7_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_ch, const uint16_t in_tensor_group, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim_x, const uint16_t pad_x, const uint16_t stride_x, const int32_t * bias, q7_t * out_tensor, const int32_t * out_shift, const int32_t * out_scale, const int32_t out_offset, const int32_t in_offset, const int32_t act_min, const int32_t act_max, const uint16_t out_tensor_dim_x, q15_t * in_tmp_buf)

描述：

该函数在任意 x 和 y 维度对有符号 8 位整数输入/输出执行 1xn 核卷积，以及对输出执行偏置输入和非对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_ch 输入张量的通道数量

- [in] in_tensor_group 空
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] pad_x x 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] *bias 指向偏置向量的指针
- [out] *out_tensor 指向输出张量的指针
- [in] *out_shift 指向移位向量的指针，用于输出张量
- [in] *out_scale 指向缩放向量的指针，用于输出张量
- [in] out_offset 输出张量的偏移值，取值范围应为-128~127
- [in] in_offset 输入张量的偏移值，取值范围应为-127~128
- [in] act_min 限制输出张量的最小值，取值范围应为-128~127
- [in] act_max 限制输出张量的最大值，取值范围应为-128~127
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] *tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_dim_x 是 4 的倍数的约束条件，则返回-1。

3.4.76

riscv_nn_conv_1xn_HWC_s8_s8_s8_asym_bias_any_get_buffer_size

原型：

int32_t riscv_nn_conv_1xn_HWC_s8_s8_s8_asym_bias_any_get_buffer_size (const uint16_t in_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y)

描述：

该函数以字节为单元，返回 riscv_nn_conv_1xn_HWC_s8_s8_s8_asym_bias_any 函数所需的输入临时缓存区大小。

参数：

- [in] in_tensor_ch 输入张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度

- [in] ker_dim_y 卷积核的 y 维度，恒为 1

返回值：

该函数返回所需的临时缓存区大小。

3.4.77 riscv_nn_conv_HWC_s8_s8_s8_asym_bias_any

原型：

int32_t riscv_nn_conv_HWC_s8_s8_s8_asym_bias_any (const q7_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const uint16_t in_tensor_group, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const uint16_t stride_y, const int32_t * bias, q7_t * out_tensor, const int32_t * out_shift, const int32_t * out_scale, const int32_t out_offset, const int32_t in_offset, const int32_t act_min, const int32_t act_max, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, q15_t * in_tmp_buf)

描述：

该函数在任意 x 和 y 维度对有符号 8 位整数输入/输出执行卷积，以及对输出执行偏置输入和非对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] in_tensor_group 输入张量的组数
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] *bias 指向偏置向量的指针
- [out] *out_tensor 指向输出张量的指针

- [in] *out_shift 指向移位向量的指针，用于输出张量
- [in] *out_scale 指向缩放向量的指针，用于输出张量
- [in] out_offset 输出张量的偏移值，取值范围应为-128~127
- [in] in_offset 输入张量的偏移值，取值范围应为-127~128
- [in] act_min 限制输出张量的最小值，取值范围应为-128~127
- [in] act_max 限制输出张量的最大值，取值范围应为-128~127
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

0

3.4.78 riscv_nn_conv_HWC_s8_s8_s8_asym_bias_any_get_buffer_size

原型：

int32_t riscv_nn_conv_HWC_s8_s8_s8_asym_bias_any_get_buffer_size (const uint16_t in_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y)

描述：

该函数以字节为单元，返回 riscv_nn_conv_HWC_s8_s8_s8_asym_bias_any 函数所需的输入临时缓存区大小。

参数：

- [in] in_tensor_ch 输入张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度

返回值：

该函数返回所需的临时缓存区大小。

3.4.79 riscv_nn_conv_dw_HWC_3x3_s8_s8_s8_asym_bias_any

原型：

int32_t riscv_nn_conv_dw_HWC_3x3_s8_s8_s8_asym_bias_any (const int8_t * in_tensor, const int32_t in_tensor_dim_x, const int32_t in_tensor_dim_y, const int32_t in_tensor_ch, const int8_t * ker_weight, const int32_t out_tensor_ch, const int32_t pad_x, const int32_t pad_y,

const int32_t stride_x, const int32_t stride_y, const int32_t * bias, int8_t * out_tensor, const int32_t * out_shift, const int32_t * out_scale, const int32_t out_tensor_dim_x, const int32_t out_tensor_dim_y, const int32_t out_offset, const int32_t in_offset, const int32_t act_min, const int32_t act_max, const int32_t dilation_x, const int32_t dilation_y, int16_t * tmp_buf)

描述：

该函数在任意 x 和 y 维度对有符号 8 位整数输入/输出执行深度 3x3 核卷积，以及对输出执行偏置输入和非对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] *bias 指向偏置向量的指针
- [out] *out_tensor 指向输出张量的指针
- [in] *out_shift 指向移位向量的指针，用于输出张量
- [in] *out_scale 指向缩放向量的指针，用于输出张量
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] out_offset 输出张量的偏移值，取值范围应为-128~127
- [in] in_offset 输入张量的偏移值，取值范围应为-127~128
- [in] act_min 限制输出张量的最小值，取值范围应为-128~127
- [in] act_max 限制输出张量的最大值，取值范围应为-128~127
- [in] dilation_x 空
- [in] dilation_y 空
- [in] *tmp_buf 空指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 等于 out_tensor_ch 和 pad_x 小于 1 的约束条件，则返回-1。

3.4.80 riscv_nn_conv_dw_HWC_s8_s8_s8_asym_bias_any

原型：

int32_t riscv_nn_conv_dw_HWC_s8_s8_s8_asym_bias_any (const q7_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ch_mult, const uint16_t ker_dim_x, const uint16_t ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const uint16_t stride_y, const int32_t * bias, q7_t * out_tensor, const int32_t * out_shift, const int32_t * out_scale, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, const int32_t out_offset, const int32_t in_offset, const int32_t act_min, const int32_t act_max, const uint16_t dilation_x, const uint16_t dilation_y, q15_t * tmp_buf)

描述：

该函数在任意 x 和 y 维度对有符号 8 位整数输入/输出执行深度卷积，以及对输出执行偏置输入和非对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量，等于“ch_mult * in_tensor_ch”
- [in] ch_mult 输入张量通道的乘数
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] *bias 指向偏置向量的指针

- [out] *out_tensor 指向输出张量的指针
- [in] *out_shift 指向移位向量的指针，用于输出张量
- [in] *out_scale 指向缩放向量的指针，用于输出张量
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] out_offset 输出张量的偏移值，取值范围应为-128~127
- [in] in_offset 输入张量的偏移值，取值范围应为-127~128
- [in] act_min 限制输出张量的最小值，取值范围应为-128~127
- [in] act_max 限制输出张量的最大值，取值范围应为-128~127
- [in] dilation_x 空
- [in] dilation_y 空
- [in] *tmp_buf 空指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 等于 out_tensor_ch 的约束条件，则返回-1。

3.4.81 riscv_nn_conv_dw_HWC_s8_s8_s8_asym_bias_fast_any

原型：

int32_t riscv_nn_conv_dw_HWC_s8_s8_s8_asym_bias_fast_any (const q7_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const q7_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const uint16_t stride_y, const int32_t * bias, q7_t * out_tensor, const int32_t * out_shift, const int32_t * out_scale, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, const int32_t out_offset, const int32_t in_offset, const int32_t act_min, const int32_t act_max, const uint16_t dilation_x, const uint16_t dilation_y, q15_t * in_tmp_buf)

描述：

该函数在任意 x 和 y 维度对有符号 8 位整数输入/输出执行快速深度卷积，以及对输出执行偏置输入和非对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度

- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] *bias 指向偏置向量的指针
- [out] *out_tensor 指向输出张量的指针
- [in] *out_shift 指向移位向量的指针，用于输出张量
- [in] *out_scale 指向缩放向量的指针，用于输出张量
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] out_offset 输出张量的偏移值，取值范围应为-128~127
- [in] in_offset 输入张量的偏移值，取值范围应为-127~128
- [in] act_min 限制输出张量的最小值，取值范围应为-128~127
- [in] act_max 限制输出张量的最大值，取值范围应为-128~127
- [in] dilation_x 空
- [in] dilation_y 空
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 等于 out_tensor_ch 的约束条件，则返回-1。

3.4.82 riscv_nn_conv_dw_HWC_s8_s8_s8_asym_bias_fast_any_get_buffer_size

原型：

int32_t riscv_nn_conv_dw_HWC_s8_s8_s8_asym_bias_fast_any_get_buffer_size

(const uint16_t in_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y)

描述：

该函数以字节为单元，返回 riscv_nn_conv_dw_HWC_s8_fast_any 函数所需的输入临时缓存区大小。

参数：

- [in] in_tensor_ch 输入张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度

返回值：

该函数返回所需的临时缓存区大小。

3.4.83 riscv_nn_conv_dw_HWC_u8_u8_u8_asym_bias_any

原型：

int32_t riscv_nn_conv_dw_HWC_u8_u8_u8_asym_bias_any (const uint8_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const uint8_t * ker_weight, const uint16_t ker_dim_x, const uint16_t ker_dim_y, const int16_t ch_mult, const int16_t pad_x, const int16_t pad_y, const int16_t stride_x, const int16_t stride_y, const int16_t dilation_x, const int16_t dilation_y, const int32_t * bias, const int32_t in_offset, const int32_t ker_offset, const int32_t out_offset, uint8_t * out_tensor, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, const int32_t act_min, const int32_t act_max, const int32_t out_shift, const int32_t out_scale)

描述：

该函数在任意 x 和 y 维度对无符号 8 位整数输入/输出执行深度卷积，以及对输出执行偏置输入和非对称量化。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度

- [in] ker_dim_y 卷积核的 y 维度
- [in] ch_mult 输入张量通道的乘数
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] dilation_x 空
- [in] dilation_y 空
- [in] *bias 指向偏置向量的指针
- [in] in_offset 输入张量的偏移值，取值范围应为-255~0
- [in] ker_offset 卷积核的偏移值，取值范围应为-255~0
- [in] out_offset 输出张量的偏移值，取值范围应为 0~255
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] act_min 限制输出张量的最小值，取值范围应为 0~255
- [in] act_max 限制输出张量的最大值，取值范围应为 0~255
- [in] out_shift 输出张量的移位量
- [in] out_scale 输出张量的缩放值

返回值：

该函数成功则返回 0；否则，如果输入不满足 ch_mult 和 ker_dim_x 都是 2 的倍数的约束条件，则返回-1。

3.4.84 riscv_nn_conv_1x1_HWC_f16_f16_f16_bias_any

原型：

int32_t riscv_nn_conv_1x1_HWC_f16_f16_f16_bias_any (const float16_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const float16_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const uint16_t stride_y, const float16_t * bias, float16_t * out_tensor, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, float16_t * in_tmp_buf, float16_t * tmp_buf)

描述：

该函数在任意 x 和 y 维度对半精度浮点输入/输出执行 1x1 核卷积。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] *bias 指向偏置向量的指针
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] *in_tmp_buf 空指针
- [in] *tmp_buf 空指针

返回值：

0

注！

该函数的输入约束条件：

- in_tensor_ch 4 的倍数
- out_tensor_ch 2 的倍数
- ker_dim_x 1
- ker_dim_y 1
- pad_x 0
- pad_y 0
- stride_x 1
- stride_y 1

3.4.85 riscv_nn_conv_HWC_f16_f16_f16_bias

原型：

int32_t riscv_nn_conv_HWC_f16_f16_f16_bias (const float16_t * in_tensor, const uint16_t in_tensor_dim, const uint16_t in_tensor_ch, const float16_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const float16_t * bias, float16_t * out_tensor, const uint16_t out_tensor_dim, float16_t * in_tmp_buf, float16_t * tmp_buf)

描述：

该函数对半精度浮点输入/输出执行卷积。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小
- [in] stride 卷积步距
- [in] *bias 指向偏置向量的指针
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针
- [in] *tmp_buf 空指针

返回值：

0

3.4.86 riscv_nn_conv_dw_HWC_f16_f16_f16_bias

原型：

int32_t riscv_nn_conv_dw_HWC_f16_f16_f16_bias (const float16_t * in_tensor, const uint16_t in_tensor_dim, const uint16_t in_tensor_ch, const float16_t * ker_weight, const uint16_t out_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const float16_t * bias, float16_t * out_tensor, const uint16_t out_tensor_dim, float16_t *

in_tmp_buf, float16_t * tmp_buf)

描述：

该函数对半精度浮点输入/输出执行深度卷积。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] *ker_weight 指向卷积核权重的指针
- [in] out_tensor_ch 输出张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小
- [in] stride 卷积步距
- [in] *bias 指向偏置向量的指针
- [out] *out_tensor 指向输出张量的指针
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针
- [in] *tmp_buf 空指针

返回值：

该函数成功则返回 0；否则，如果输入不满足 in_tensor_ch 等于 out_tensor_ch 的约束条件，则返回-1。

3.5 全连接函数

全连接函数实现输入向量乘以一个权重矩阵，在结果中在加上一个偏置（如果有的话）。输入向量和权重矩阵所支持的组合，包括（有符号 8 位整数，有符号 8 位整数）、（无符号 8 位整数，有符号 8 位整数）、（有符号 16 位整数，有符号 8 位整数）、（有符号 16 位整数，有符号 16 位整数）和（半精度浮点，半精度浮点）。

3.5.1 riscv_nn_fc_s8_s8_s8_sft_bias

原型：

int32_t riscv_nn_fc_s8_s8_s8_sft_bias (const q7_t * in_vec, const q7_t * wt_mat, const uint16_t size, const uint16_t wt_row_num, const uint16_t bias_lshift, const uint16_t out_rshift, const q7_t * bias, q7_t * out_vec, q15_t * in_tmp_buf)

描述：

该函数是一个全连接层函数，用于有符号 8 位整数输入，以及对输出执行基于移位的量化。

参数：

- [in] *in_vec 指向输入向量的指针
- [in] *wt_mat 指向权重矩阵的指针
- [in] size 输入向量的元素数量
- [in] wt_row_num 权重矩阵的行数
- [in] bias_lshift 偏置的左移量
- [in] out_rshift 输出的右移量
- [in] *bias 指向偏置向量的指针
- [out] *out_vec 指向输出向量的指针
- [in] *in_tmp_buf 空指针

返回值：

0

示例：

```
#define IN_SIZE 2048
#define OUT_SIZE 256
#define BIAS_LSHIFT 9 //Scale up the bias by 2^9
#define OUT_RSHIFT 9 //Scale down the outputs by 1/2^9
q7_t in_vec[IN_SIZE] = {...};
q7_t wt_mat[IN_SIZE * OUT_SIZE] {...};
q7_t bias[OUT_SIZE] = {...};
q7_t out_vec[OUT_SIZE];
riscv_nn_fc_s8_s8_s8_sft_bias (in_vec, wt_mat, IN_SIZE, OUT_SIZE, BIAS_LSHIFT, OUT_RSHIFT, bias, out_vec, NULL);
```

3.5.2 riscv_nn_fc_s8_s8_s8_sft_bias_fast

原型：

```
int32_t riscv_nn_fc_s8_s8_s8_sft_bias_fast (const q7_t * in_vec, const q7_t * wt_mat, const uint16_t size, const uint16_t wt_row_num, const uint16_t bias_lshift, const uint16_t out_rshift, const q7_t * bias, q7_t * out_vec, q15_t * in_tmp_buf)
```

描述：

该函数是一个全连接层函数，用于有符号 8 位整数输入，以及对输出执行交叉相乘和基于移位的量化。

参数：

- [in] *in_vec 指向输入向量的指针
- [in] *wt_mat 指向权重矩阵的指针
- [in] size 输入向量的元素数量
- [in] wt_row_num 权重矩阵的行数
- [in] bias_lshift 偏置的左移量
- [in] out_rshift 输出的右移量
- [in] *bias 指向偏置向量的指针
- [out] *out_vec 指向输出向量的指针
- [in] *in_tmp_buf 指向输入向量临时缓存区的指针

返回值：

0

注！

该函数中，输入向量乘以一个由 riscv_nn_fc_s8_wt_converter 函数返回的交替存储格式的权重矩阵。

3.5.3 riscv_nn_fc_s16_s16_s16_sft_bias

原型：

int32_t riscv_nn_fc_s16_s16_s16_sft_bias (const q15_t * in_vec, const q15_t * wt_mat, const uint16_t size, const uint16_t wt_row_num, const uint16_t bias_lshift, const uint16_t out_rshift, const q15_t * bias, q15_t * out_vec, q15_t * tmp_buf)

描述：

该函数是一个全连接层函数，用于有符号 16 位整数输入，以及对输出执行基于移位的量化。

参数：

- [in] *in_vec 指向输入向量的指针
- [in] *wt_mat 指向权重矩阵的指针
- [in] size 输入向量的元素数量
- [in] wt_row_num 权重矩阵的行数

- [in] bias_lshift 偏置的左移量
- [in] out_rshift 输出的右移量
- [in] *bias 指向偏置向量的指针
- [out] *out_vec 指向输出向量的指针
- [in] *tmp_buf 空指针

返回值：

0

3.5.4 riscv_nn_fc_s16_s16_s16_sft_bias_fast

原型：

int32_t riscv_nn_fc_s16_s16_s16_sft_bias_fast (const q15_t * in_vec, const q15_t * wt_mat, const uint16_t size, const uint16_t wt_row_num, const uint16_t bias_lshift, const uint16_t out_rshift, const q15_t * bias, q15_t * out_vec, q15_t * in_tmp_buf)

描述：

该函数是一个全连接层函数，用于有符号 16 位整数输入，以及对输出执行交叉相乘和基于移位的量化。

参数：

- [in] *in_vec 指向输入向量的指针
- [in] *wt_mat 指向权重矩阵的指针
- [in] size 输入向量的元素数量
- [in] wt_row_num 权重矩阵的行数
- [in] bias_lshift 偏置的左移量
- [in] out_rshift 输出的右移量
- [in] *bias 指向偏置向量的指针
- [out] *out_vec 指向输出向量的指针
- [in] *in_tmp_buf 指向输入向量临时缓存区的指针

返回值：

0

注！

该函数中，输入向量乘以一个由 riscv_nn_fc_s16_wt_converter 函数返回的交替存储格式的权重矩阵。

3.5.5 riscv_nn_fc_mat_vec_s16_s16_s8_sft_bias

原型：

int32_t riscv_nn_fc_mat_vec_s16_s16_s8_sft_bias (const q15_t * in_vec, const q7_t * wt_mat, const uint16_t size, const uint16_t wt_row_num, const uint16_t bias_lshift, const uint16_t out_rshift, const q7_t * bias, q15_t * out_vec, q15_t * tmp_buf)

描述：

该函数实现一个有符号 16 位整数输入向量乘以一个有符号 8 位整数权重矩阵，以及对输出执行基于移位的量化。

参数：

- [in] *in_vec 指向输入向量的指针
- [in] *wt_mat 指向权重矩阵的指针
- [in] size 输入向量的元素数量
- [in] wt_row_num 权重矩阵的行数
- [in] bias_lshift 偏置的左移量
- [in] out_rshift 输出的右移量
- [in] *bias 指向偏置向量的指针
- [out] *out_vec 指向输出向量的指针
- [in] *tmp_buf 空指针

返回值：

0

3.5.6 riscv_nn_fc_mat_vec_s16_s16_s8_sft_bias_fast

原型：

int32_t riscv_nn_fc_mat_vec_s16_s16_s8_sft_bias_fast (const q15_t * in_vec, const q7_t * wt_mat, const uint16_t size, const uint16_t wt_row_num, const uint16_t bias_lshift, const uint16_t out_rshift, const q7_t * bias, q15_t * out_vec, q15_t * tmp_buf)

描述：

该函数实现一个有符号 16 位整数输入向量乘以一个交替存储格式的有符号 8 位整数权重矩阵，以及对输出执行基于移位的量化。

参数：

- [in] *in_vec 指向输入向量的指针
- [in] *wt_mat 指向权重矩阵的指针

- [in] size 输入向量的元素数量
- [in] wt_row_num 权重矩阵的行数
- [in] bias_lshift 偏置的左移量
- [in] out_rshift 输出的右移量
- [in] *bias 指向偏置向量的指针
- [out] *out_vec 指向输出向量的指针
- [in] *tmp_buf 空指针

返回值：

0

注！

该函数中，输入向量乘以一个由 riscv_nn_fc_mat_vec_s8_wt_converter 函数返回的交替存储格式的权重矩阵。

3.5.7 riscv_nn_fc_s8_s8_s8_sym_bias

原型：

int32_t riscv_nn_fc_s8_s8_s8_sym_bias (const q7_t * in_vec, const q7_t * wt_mat, const uint16_t size, const uint16_t wt_row_num, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, const q31_t * bias, q7_t * out_vec, q15_t * in_tmp_buf)

描述：

该函数是一个全连接层函数，用于有符号 8 位整数输入/输出，以及对输出执行偏置输入和对称量化。

参数：

- [in] *in_vec 指向输入向量的指针
- [in] *wt_mat 指向权重矩阵的指针
- [in] size 输入向量的元素数量
- [in] wt_row_num 权重矩阵的行数
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [in] *bias 指向偏置向量的指针
- [out] *out_vec 指向输出向量的指针
- [in] *in_tmp_buf 指向输入向量临时缓存区的指针

返回值：

0

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.5.8 riscv_nn_fc_s8_s16_s8_sym_bias

原型：

int32_t riscv_nn_fc_s8_s16_s8_sym_bias (const q7_t * in_vec, const q7_t * wt_mat, const uint16_t size, const uint16_t wt_row_num, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, const q31_t * bias, q15_t * out_vec, q15_t * in_tmp_buf)

描述：

该函数是一个全连接层函数，用于有符号 8 位整数输入和有符号 16 位整数输出，以及对输出执行偏置输入和对称量化。

参数：

- [in] *in_vec 指向输入向量的指针
- [in] *wt_mat 指向权重矩阵的指针
- [in] size 输入向量的元素数量
- [in] wt_row_num 权重矩阵的行数
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [in] *bias 指向偏置向量的指针
- [out] *out_vec 指向输出向量的指针
- [in] *in_tmp_buf 指向输入向量临时缓存区的指针

返回值：

0

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.5.9 riscv_nn_fc_u8_u8_s8_sym_bias

原型：

int32_t riscv_nn_fc_u8_u8_s8_sym_bias (const u8_t * in_vec, const

q7_t * wt_mat, const uint16_t size, const uint16_t wt_row_num, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, const q31_t * bias, u8_t * out_vec, q15_t * in_tmp_buf)

描述：

该函数是一个全连接层函数，用于无符号 8 位整数输入/输出，以及对输出执行偏置输入和对称量化。

参数：

- [in] *in_vec 指向输入向量的指针
- [in] *wt_mat 指向权重矩阵的指针
- [in] size 输入向量的元素数量
- [in] wt_row_num 权重矩阵的行数
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [in] *bias 指向偏置向量的指针
- [out] *out_vec 指向输出向量的指针
- [in] *in_tmp_buf 指向输入向量临时缓存区的指针

返回值：

0

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.5.10 riscv_nn_fc_u8_s8_s8_sym_bias

原型：

int32_t riscv_nn_fc_u8_s8_s8_sym_bias (const u8_t * in_vec, const q7_t * wt_mat, const uint16_t size, const uint16_t wt_row_num, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, const q31_t * bias, q7_t * out_vec, q15_t * in_tmp_buf)

描述：

该函数是一个全连接层函数，用于无符号 8 位整数输入和有符号 8 位整数输出，以及对输出执行偏置输入和对称量化。

参数：

- [in] *in_vec 指向输入向量的指针
- [in] *wt_mat 指向权重矩阵的指针

- [in] size 输入向量的元素数量
- [in] wt_row_num 权重矩阵的行数
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [in] *bias 指向偏置向量的指针
- [out] *out_vec 指向输出向量的指针
- [in] *in_tmp_buf 指向输入向量临时缓存区的指针

返回值：

0

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.5.11 riscv_nn_fc_u8_s16_s8_sym_bias

原型：

int32_t riscv_nn_fc_u8_s16_s8_sym_bias (const u8_t * in_vec, const q7_t * wt_mat, const uint16_t size, const uint16_t wt_row_num, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, const q31_t * bias, q15_t * out_vec, q15_t * in_tmp_buf)

描述：

该函数是一个全连接层函数，用于无符号 8 位整数输入和有符号 16 位整数输出，以及对输出执行偏置输入和对称量化。

参数：

- [in] *in_vec 指向输入向量的指针
- [in] *wt_mat 指向权重矩阵的指针
- [in] size 输入向量的元素数量
- [in] wt_row_num 权重矩阵的行数
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [in] *bias 指向偏置向量的指针
- [out] *out_vec 指向输出向量的指针
- [in] *in_tmp_buf 指向输入向量临时缓存区的指针

返回值：

0

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.5.12 riscv_nn_fc_s8_s8_s8_sym

原型：

int32_t riscv_nn_fc_s8_s8_s8_sym (const q7_t * in_vec, const q7_t * wt_mat, const uint16_t size, const uint16_t wt_row_num, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q7_t * out_vec, q15_t * in_tmp_buf)

描述：

该函数是一个全连接层函数，用于有符号 8 位整数输入/输出，以及对输出执行对称量化。

参数：

- [in] *in_vec 指向输入向量的指针
- [in] *wt_mat 指向权重矩阵的指针
- [in] size 输入向量的元素数量
- [in] wt_row_num 权重矩阵的行数
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_vec 指向输出向量的指针
- [in] *in_tmp_buf 指向输入向量临时缓存区的指针

返回值：

0

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.5.13 riscv_nn_fc_s8_s16_s8_sym

原型：

int32_t riscv_nn_fc_s8_s16_s8_sym (const q7_t * in_vec, const q7_t * wt_mat, const uint16_t size, const uint16_t wt_row_num, const uint16_t

pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q15_t * out_vec, q15_t * in_tmp_buf)

描述：

该函数是一个全连接层函数，用于有符号 8 位整数输入和有符号 16 位整数输出，以及对输出执行对称量化。

参数：

- [in] *in_vec 指向输入向量的指针
- [in] *wt_mat 指向权重矩阵的指针
- [in] size 输入向量的元素数量
- [in] wt_row_num 权重矩阵的行数
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_vec 指向输出向量的指针
- [in] *in_tmp_buf 指向输入向量临时缓存区的指针

返回值：

0

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.5.14 riscv_nn_fc_u8_u8_s8_sym

原型：

int32_t riscv_nn_fc_u8_u8_s8_sym (const u8_t * in_vec, const q7_t * wt_mat, const uint16_t size, const uint16_t wt_row_num, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, u8_t * out_vec, q15_t * in_tmp_buf)

描述：

该函数是一个全连接层函数，用于无符号 8 位整数输入/输出，以及对输出执行对称量化。

参数：

- [in] *in_vec 指向输入向量的指针
- [in] *wt_mat 指向权重矩阵的指针
- [in] size 输入向量的元素数量

- [in] wt_row_num 权重矩阵的行数
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_vec 指向输出向量的指针
- [in] *in_tmp_buf 指向输入向量临时缓存区的指针

返回值：

0

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.5.15 riscv_nn_fc_u8_s8_s8_sym

原型：

int32_t riscv_nn_fc_u8_s8_s8_sym (const u8_t * in_vec, const q7_t * wt_mat, const uint16_t size, const uint16_t wt_row_num, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q7_t * out_vec, q15_t * in_tmp_buf)

描述：

该函数是一个全连接层函数，用于无符号 8 位整数输入和有符号 8 位整数输出，以及对输出执行对称量化。

参数：

- [in] *in_vec 指向输入向量的指针
- [in] *wt_mat 指向权重矩阵的指针
- [in] size 输入向量的元素数量
- [in] wt_row_num 权重矩阵的行数
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_vec 指向输出向量的指针
- [in] *in_tmp_buf 指向输入向量临时缓存区的指针

返回值：

0

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.5.16 riscv_nn_fc_u8_s16_s8_sym

原型：

int32_t riscv_nn_fc_u8_s16_s8_sym (const u8_t * in_vec, const q7_t * wt_mat, const uint16_t size, const uint16_t wt_row_num, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q15_t * out_vec, q15_t * in_tmp_buf);

描述：

该函数是一个全连接层函数，用于无符号 8 位整数输入和有符号 16 位整数输出，以及对输出执行对称量化。

参数：

- [in] *in_vec 指向输入向量的指针
- [in] *wt_mat 指向权重矩阵的指针
- [in] size 输入向量的元素数量
- [in] wt_row_num 权重矩阵的行数
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_vec 指向输出向量的指针
- [in] *in_tmp_buf 指向输入向量临时缓存区的指针

返回值：

0

注！

输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.5.17 riscv_nn_fc_s8_s8_s8_sym_bias_fast

原型：

int32_t riscv_nn_fc_s8_s8_s8_sym_bias_fast (const q7_t * in_vec, const q7_t * wt_mat, const uint16_t size, const uint16_t wt_row_num, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, const q31_t * bias, q7_t * out_vec, q15_t * in_tmp_buf)

描述：

该函数是一个全连接层函数，用于有符号 8 位整数输入/输出，以及对输出执行偏置输入、交叉相乘和对称量化。

参数：

- [in] *in_vec 指向输入向量的指针
- [in] *wt_mat 指向权重矩阵的指针
- [in] size 输入向量的元素数量
- [in] wt_row_num 权重矩阵的行数
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [in] *bias 指向偏置向量的指针
- [out] *out_vec 指向输出向量的指针
- [in] *in_tmp_buf 指向输入向量临时缓存区的指针

返回值：

0

注！

1. 该函数中，输入向量乘以一个由 riscv_nn_fc_s8_wt_converter 函数返回的交替存储格式的权重矩阵。
2. 输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.5.18 riscv_nn_fc_s8_s16_s8_sym_bias_fast

原型：

int32_t riscv_nn_fc_s8_s16_s8_sym_bias_fast (const q7_t * in_vec, const q7_t * wt_mat, const uint16_t size, const uint16_t wt_row_num, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, const q31_t * bias, q15_t * out_vec, q15_t * in_tmp_buf)

描述：

该函数是一个全连接层函数，用于有符号 8 位整数输入和有符号 16 位整数输出，以及对输出执行偏置输入、交叉相乘和对称量化。

参数：

- [in] *in_vec 指向输入向量的指针
- [in] *wt_mat 指向权重矩阵的指针
- [in] size 输入向量的元素数量

- [in] wt_row_num 权重矩阵的行数
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [in] *bias 指向偏置向量的指针
- [out] *out_vec 指向输出向量的指针
- [in] *in_tmp_buf 指向输入向量临时缓存区的指针

返回值：

0

注！

1. 该函数中，输入向量乘以一个由 riscv_nn_fc_s8_wt_converter 函数返回的交替存储格式的权重矩阵。
2. 输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.5.19 riscv_nn_fc_u8_u8_s8_sym_bias_fast

原型：

int32_t riscv_nn_fc_u8_u8_s8_sym_bias_fast (const u8_t * in_vec, const q7_t * wt_mat, const uint16_t size, const uint16_t wt_row_num, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, const q31_t * bias, u8_t * out_vec, q15_t * in_tmp_buf)

描述：

该函数是一个全连接层函数，用于无符号 8 位整数输入/输出，以及对输出执行偏置输入、交叉相乘和对称量化。

参数：

- [in] *in_vec 指向输入向量的指针
- [in] *wt_mat 指向权重矩阵的指针
- [in] size 输入向量的元素数量
- [in] wt_row_num 权重矩阵的行数
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [in] *bias 指向偏置向量的指针
- [out] *out_vec 指向输出向量的指针

- [in] *in_tmp_buf 指向输入向量临时缓存区的指针

返回值：

0

注！

1. 该函数中，输入向量乘以一个由 riscv_nn_fc_s8_wt_converter 函数返回的交替存储格式的权重矩阵。
2. 输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.5.20 riscv_nn_fc_u8_s8_s8_sym_bias_fast

原型：

int32_t riscv_nn_fc_u8_s8_s8_sym_bias_fast (const u8_t * in_vec, const q7_t * wt_mat, const uint16_t size, const uint16_t wt_row_num, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, const q31_t * bias, q7_t * out_vec, q15_t * in_tmp_buf)

描述：

该函数是一个全连接层函数，用于无符号 8 位整数输入和有符号 8 位整数输出，以及对输出执行偏置输入、交叉相乘和对称量化。

参数：

- [in] *in_vec 指向输入向量的指针
- [in] *wt_mat 指向权重矩阵的指针
- [in] size 输入向量的元素数量
- [in] wt_row_num 权重矩阵的行数
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [in] *bias 指向偏置向量的指针
- [out] *out_vec 指向输出向量的指针
- [in] *in_tmp_buf 指向输入向量临时缓存区的指针

返回值：

0

注！

1. 该函数中，输入向量乘以一个由 riscv_nn_fc_s8_wt_converter 函数返回的交替存储格式的权重矩阵。

2. 输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.5.21 riscv_nn_fc_u8_s16_s8_sym_bias_fast

原型：

int32_t riscv_nn_fc_u8_s16_s8_sym_bias_fast (const u8_t * in_vec, const q7_t * wt_mat, const uint16_t size, const uint16_t wt_row_num, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, const q31_t * bias, q15_t * out_vec, q15_t * in_tmp_buf)

描述：

该函数是一个全连接层函数，用于无符号 8 位整数输入和有符号 16 位整数输出，以及对输出执行偏置输入、交叉相乘和对称量化。

参数：

- [in] *in_vec 指向输入向量的指针
- [in] *wt_mat 指向权重矩阵的指针
- [in] size 输入向量的元素数量
- [in] wt_row_num 权重矩阵的行数
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [in] *bias 指向偏置向量的指针
- [out] *out_vec 指向输出向量的指针
- [in] *in_tmp_buf 指向输入向量临时缓存区的指针

返回值：

0

注！

1. 该函数中，输入向量乘以一个由 riscv_nn_fc_s8_wt_converter 函数返回的交替存储格式的权重矩阵。
2. 输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.5.22 riscv_nn_fc_s8_s8_s8_sym_fast

原型：

int32_t riscv_nn_fc_s8_s8_s8_sym_fast (const q7_t * in_vec, const q7_t * wt_mat, const uint16_t size, const uint16_t wt_row_num, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift,

q7_t * out_vec, q15_t * in_tmp_buf)

描述：

该函数是一个全连接层函数，用于有符号 8 位整数输入/输出，以及对输出执行交叉相乘和对称量化。

参数：

- [in] *in_vec 指向输入向量的指针
- [in] *wt_mat 指向权重矩阵的指针
- [in] size 输入向量的元素数量
- [in] wt_row_num 权重矩阵的行数
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_vec 指向输出向量的指针
- [in] *in_tmp_buf 指向输入向量临时缓存区的指针

返回值：

0

注！

1. 该函数中，输入向量乘以一个由 riscv_nn_fc_s8_wt_converter 函数返回的交替存储格式的权重矩阵。
2. 输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.5.23 riscv_nn_fc_s8_s16_s8_sym_fast

原型：

int32_t riscv_nn_fc_s8_s16_s8_sym_fast (const q7_t * in_vec, const q7_t * wt_mat, const uint16_t size, const uint16_t wt_row_num, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q15_t * out_vec, q15_t * in_tmp_buf)

描述：

该函数是一个全连接层函数，用于有符号 8 位整数输入和有符号 16 位整数输出，以及对输出执行交叉相乘和对称量化。

参数：

- [in] *in_vec 指向输入向量的指针
- [in] *wt_mat 指向权重矩阵的指针
- [in] size 输入向量的元素数量

- [in] wt_row_num 权重矩阵的行数
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_vec 指向输出向量的指针
- [in] *in_tmp_buf 指向输入向量临时缓存区的指针

返回值：

0

注！

1. 该函数中，输入向量乘以一个由 riscv_nn_fc_s8_wt_converter 函数返回的交替存储格式的权重矩阵。
2. 输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.5.24 riscv_nn_fc_u8_u8_s8_sym_fast

原型：

int32_t riscv_nn_fc_u8_u8_s8_sym_fast (const u8_t * in_vec, const q7_t * wt_mat, const uint16_t size, const uint16_t wt_row_num, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, u8_t * out_vec, q15_t * in_tmp_buf)

描述：

该函数是一个全连接层函数，用于无符号 8 位整数输入/输出，以及对输出执行交叉相乘和对称量化。

参数：

- [in] *in_vec 指向输入向量的指针
- [in] *wt_mat 指向权重矩阵的指针
- [in] size 输入向量的元素数量
- [in] wt_row_num 权重矩阵的行数
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_vec 指向输出向量的指针
- [in] *in_tmp_buf 指向输入向量临时缓存区的指针

返回值：

0

注！

1. 该函数中，输入向量乘以一个由 riscv_nn_fc_s8_wt_converter 函数返回的交替存储格式的权重矩阵。
2. 输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.5.25 riscv_nn_fc_u8_s8_s8_sym_fast

原型：

int32_t riscv_nn_fc_u8_s8_s8_sym_fast (const u8_t * in_vec, const q7_t * wt_mat, const uint16_t size, const uint16_t wt_row_num, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q7_t * out_vec, q15_t * in_tmp_buf)

描述：

该函数是一个全连接层函数，用于无符号 8 位整数输入和有符号 8 位整数输出，以及对输出执行交叉相乘和对称量化。

参数：

- [in] *in_vec 指向输入向量的指针
- [in] *wt_mat 指向权重矩阵的指针
- [in] size 输入向量的元素数量
- [in] wt_row_num 权重矩阵的行数
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_vec 指向输出向量的指针
- [in] *in_tmp_buf 指向输入向量临时缓存区的指针

返回值：

0

注！

1. 该函数中，输入向量乘以一个由 riscv_nn_fc_s8_wt_converter 函数返回的交替存储格式的权重矩阵。
2. 输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.5.26 riscv_nn_fc_u8_s16_s8_sym_fast

原型：

int32_t riscv_nn_fc_u8_s16_s8_sym_fast (const u8_t * in_vec, const q7_t * wt_mat, const uint16_t size, const uint16_t wt_row_num, const uint16_t pre_rshift, const uint16_t out_scale, const uint16_t post_rshift, q15_t * out_vec, q15_t * in_tmp_buf)

描述：

该函数是一个全连接层函数，用于无符号 8 位整数输入和有符号 16 位整数输出，以及对输出执行交叉相乘和对称量化。

参数：

- [in] *in_vec 指向输入向量的指针
- [in] *wt_mat 指向权重矩阵的指针
- [in] size 输入向量的元素数量
- [in] wt_row_num 权重矩阵的行数
- [in] pre_rshift 缩放前输出的右移量
- [in] out_scale 输出的缩放值
- [in] post_rshift 缩放后输出的右移量
- [out] *out_vec 指向输出向量的指针
- [in] *in_tmp_buf 指向输入向量临时缓存区的指针

返回值：

0

注！

1. 该函数中，输入向量乘以一个由函数 riscv_nn_fc_s8_wt_converter 返回的交替存储格式的权重矩阵。
2. 输出将在被保存前进行两级移位，即 out = ((out >> pre_rshift) * out_scale) >> post_rshift。

3.5.27 riscv_nn_fc_s8_wt_converter

原型：

void riscv_nn_fc_s8_wt_converter (const q7_t * wt_mat, const uint32_t size, const uint32_t wt_row_num, q7_t * wt_mat_out)

描述：

这是一个权重转换器，用于有符号 8 位权重数据以及以“fast”命名的全连接函数。

参数：

- [in] *wt_mat 指向权重矩阵的指针
- [in] size 输入向量的元素数量
- [in] wt_row_num 权重矩阵的行数
- [out] *wt_mat_out 指向按特定顺序存储的权重矩阵的指针

返回值：

无

3.5.28 riscv_nn_fc_s16_wt_converter

原型：

void riscv_nn_fc_s16_wt_converter (const q15_t * wt_mat, const uint32_t size, const uint32_t wt_row_num, q15_t * wt_mat_out)

描述：

这是一个权重转换器，用于有符号 16 位权重数据以及以“fast”命名的全连接函数。

参数：

- [in] *wt_mat 指向权重矩阵的指针
- [in] size 输入向量的元素数量
- [in] wt_row_num 权重矩阵的行数
- [out] *wt_mat_out 指向按特定顺序存储的权重矩阵的指针

返回值：

无

3.5.29 riscv_nn_fc_mat_vec_s8_wt_converter

原型：

void riscv_nn_fc_mat_vec_s8_wt_converter (const q7_t * wt_mat, const uint32_t size, const uint32_t wt_row_num, q7_t * wt_mat_out)

描述：

这是一个权重转换器，用于 riscv_nn_fc_mat_vec_s16_s16_s8_sft_bias_fast 函数。

参数：

- [in] *wt_mat 指向权重矩阵的指针
- [in] size 输入向量的元素数量
- [in] wt_row_num 权重矩阵的行数

- [out] *wt_mat_out 指向按特定顺序存储的权重矩阵的指针

返回值：

无

3.5.30 riscv_nn_fc_s8_s8_s8_asym_bias

原型：

int32_t riscv_nn_fc_s8_s8_s8_asym_bias (const int8_t * in_vec, const int8_t * wt_mat, const uint16_t in_vec_col, const uint16_t wt_mat_row, const uint16_t in_vec_group, const int32_t in_offset, const int32_t wt_offset, const int32_t out_scale, const int32_t out_shift, const int32_t out_offset, const int32_t * bias, int8_t * out_vec, const int32_t act_min, const int32_t act_max, q15_t * tmp_buf)

描述：

该函数是一个全连接层函数，用于有符号 8 位整数输入/输出，以及对输出执行偏置输入和非对称量化。

参数：

- [in] *in_vec 指向输入向量的指针
- [in] *wt_mat 指向权重矩阵的指针
- [in] in_vec_col 输入向量（或转置权重矩阵）的列数
- [in] wt_mat_row 转置权重矩阵的行数
- [in] in_vec_group 输入向量的组数
- [in] in_offset 要加到输入向量的偏移值，取值范围应为-127~128
- [in] wt_offset 要加到权重的偏移值，取值范围应为-127~128
- [in] out_scale 输出向量的缩放值
- [in] out_shift 输出向量的移位量
- [in] out_offset 要加到输出向量的偏移值，取值范围应为-128~127
- [in] *bias 指向偏置向量的指针
- [out] *out_vec 指向输出向量的指针
- [in] act_min 限制输出向量的最小值，取值范围应为-128~127
- [in] act_max 限制输出向量的最大值，取值范围应为-128~127
- [in] *tmp_buf 空指针

返回值：

0

3.5.31 riscv_nn_fc_ s8_s8_s8_asym_bias_get_buffer_size

原型：

int32_t riscv_nn_fc_s8_s8_s8_asym_bias_get_buffer_size (const uint16_t in_vec_col)

描述：

该函数以字节为单元，返回 riscv_nn_fc_s8_s8_s8_asym_bias 函数所需的输入临时缓存区大小。

参数：

- [in] in_vec_col 输入向量（或转置权重矩阵）的列数

返回值：

该函数返回所需的临时缓存区大小。

3.5.32 riscv_nn_fc_f16_f16_f16_bias

原型：

int32_t riscv_nn_fc_f16_f16_f16_bias (const float16_t * in_vec, const float16_t * wt_mat, const uint16_t size, const uint16_t wt_row_num, const float16_t * bias, float16_t * out_vec, float16_t * tmp_buf)

描述：

该函数是一个全连接层函数，用于半精度浮点输入/输出。

参数：

- [in] *in_vec 指向输入向量的指针
- [in] *wt_mat 指向权重矩阵的指针
- [in] size 输入向量的元素数量
- [in] wt_row_num 权重矩阵的行数
- [in] *bias 指向偏置向量的指针
- [out] *out_vec 指向输出向量的指针
- [in] *tmp_buf 空指针

返回值：

0

3.6 池化函数

池化函数用于下采样输入数据，包括最大池和平均池函数。

3.6.1 riscv_nn_avepool_HWC_s8

原型：

void riscv_nn_avepool_HWC_s8 (q7_t * in_tensor, const uint16_t in_tensor_dim, const uint16_t in_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const uint16_t out_tensor_dim, q7_t * in_tmp_buf, q7_t * out_tensor)

描述：

该函数是一个平均池函数，用于有符号 8 位整数输入。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小
- [in] stride 卷积步距
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针
- [out] *out_tensor 指向输出张量的指针

返回值：

无

示例：

```
#define IN_DIM 32
#define IN_CH 32
#define KER_DIM 3
#define PAD 0
#define STRIDE 2
#define OUT_DIM 15
q7_t in_data[IN_CH * IN_DIM * IN_DIM] = {...};
q7_t out_data[IN_CH * OUT_DIM * OUT_DIM] = {...};
q7_t in_tmp_buf[2 * OUT_DIM * IN_CH];
```

riscv_nn_avepool_HWC_s8 (in_data, IN_DIM, IN_CH, KER_DIM, PAD, STRIDE, OUT_DIM, in_tmp_buf, out_data);

3.6.2 riscv_nn_avepool_HWC_s8_any

原型：

void riscv_nn_avepool_HWC_s8_any (q7_t * in_tensor, const uint16_t in_tensor_dim_x, const uint16_t in_tensor_dim_y, const uint16_t in_tensor_ch, const uint16_t ker_dim_x, const uint16_t ker_dim_y, const uint16_t pad_x, const uint16_t pad_y, const uint16_t stride_x, const uint16_t stride_y, const uint16_t out_tensor_dim_x, const uint16_t out_tensor_dim_y, q7_t * in_tmp_buf, q7_t * out_tensor, const uint16_t out_lshift)

描述：

该函数是一个平均池函数，用于在任意 x 和 y 维度的有符号 8 位整数输入。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_ch 输入张量通道的数量
- [in] ker_dim_x 卷积核的 x 维度
- [in] ker_dim_y 卷积核的 y 维度
- [in] pad_x x 维度的填充大小
- [in] pad_y y 维度的填充大小
- [in] stride_x x 维度的卷积步距
- [in] stride_y y 维度的卷积步距
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针
- [out] *out_tensor 指向输出张量的指针
- [in] out_lshift 输出的左移量

返回值：

无

示例：

```
#define IN_X 160
```

```
#define IN_Y 120
#define IN_CH 3
#define KER_DIM_X 3
#define KER_DIM_Y 5
#define PAD_X 1
#define PAD_Y 1
#define STRIDE_X 2
#define STRIDE_Y 2
#define OUT_LSHIFT 3
#define OUT_X 80
#define OUT_Y 59
q7_t in_data[IN_CH * IN_X * IN_Y] = {...};
q7_t out_data[IN_CH * OUT_X * OUT_Y] = {...};
q7_t in_tmp_buf[2 * IN_CH * OUT_X * OUT_Y];
riscv_nn_avepool_HWC_s8_any (in_data, IN_X, IN_Y, IN_CH,
KER_DIM_X, KER_DIM_Y, PAD_X, PAD_Y, STRIDE_X, STRIDE_Y,
OUT_X, OUT_Y, in_tmp_buf, out_data, OUT_LSHIFT);
```

3.6.3 riscv_nn_avepool_HWC_s8_any_act

原型：

```
int32_t riscv_nn_avepool_HWC_s8_any_act (const int
in_tensor_dim_y, const int in_tensor_dim_x, const int out_tensor_dim_y,
const int out_tensor_dim_x, const int stride_y, const int stride_x, const int
ker_dim_y, const int ker_dim_x, const int pad_y, const int pad_x, const int
act_min, const int act_max, const int in_tensor_ch, int8_t * in_tensor,
int16_t * in_tmp_buf, int8_t * out_tensor)
```

描述：

该函数是一个平均池函数，用于在任意 x 和 y 维度的有符号 8 位整数输入，以及使用激活参数限制输出。

参数：

- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] stride_y y 维度的卷积步距

- [in] stride_x x 维度的卷积步距
- [in] ker_dim_y 卷积核的 y 维度
- [in] ker_dim_x 卷积核的 x 维度
- [in] pad_y y 维度的填充大小
- [in] pad_x x 维度的填充大小
- [in] act_min 限制输出张量的最小值，取值范围应为-128~127
- [in] act_max 限制输出张量的最大值，取值范围应为-128~127
- [in] in_tensor_ch 输入张量的通道数量
- [in] *in_tensor 指向输入张量的指针
- [in] *in_tmp_buf 指向输入张量临时缓存区的指针
- [out] *out_tensor 指向输出张量的指针

返回值：

0

3.6.4 riscv_nn_avepool_HWC_ s8_any_act_get_buffer_size

原型：

int32_t riscv_nn_avepool_HWC_s8_any_act_get_buffer_size (const int out_tensor_dim_x, const int in_tensor_ch)

描述：

该函数以字节为单元，返回 riscv_nn_avepool_HWC_s8_any_act 函数所需的输入临时缓存区大小。

参数：

- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] in_tensor_ch 输入张量的通道数量

返回值：

该函数返回所需的临时缓存区大小。

3.6.5 riscv_nn_maxpool_HWC_s8

原型：

void riscv_nn_maxpool_HWC_s8 (q7_t * in_tensor, const uint16_t in_tensor_dim, const uint16_t in_tensor_ch, const uint16_t ker_dim, const uint16_t pad, const uint16_t stride, const uint16_t out_tensor_dim, q7_t * in_tmp_buf, q7_t * out_tensor)

描述：

该函数是一个最大池函数，用于有符号 8 位整数输入。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_dim 输入张量的维度
- [in] in_tensor_ch 输入张量的通道数量
- [in] ker_dim 卷积核的维度
- [in] pad 填充大小
- [in] stride 卷积步距
- [in] out_tensor_dim 输出张量的维度
- [in] *in_tmp_buf 空指针
- [out] *out_tensor 指向输出张量的指针

返回值：

无

示例：

```
#define IN_DIM 32
#define IN_CH 32
#define KER_DIM 3
#define PAD 0
#define STRIDE 2
#define OUT_DIM 15
q7_t in_data[IN_CH * IN_DIM * IN_DIM] = {...};
q7_t out_data[IN_CH * OUT_DIM * OUT_DIM] = {...};
riscv_nn_maxpool_HWC_s8 (in_data, IN_DIM, IN_CH, KER_DIM, PAD, STRIDE, OUT_DIM, NULL, out_data);
```

3.6.6 riscv_nn_maxpool_HWC_s8_any_act

原型：

```
int32_t riscv_nn_maxpool_HWC_s8_any_act (const uint16_t in_tensor_dim_y, const uint16_t in_tensor_dim_x, const uint16_t out_tensor_dim_y, const uint16_t out_tensor_dim_x, const uint16_t stride_y, const uint16_t stride_x, const uint16_t ker_dim_y, const uint16_t ker_dim_x, const uint16_t pad_y, const uint16_t pad_x, const int8_t act_min, const int8_t act_max, const uint16_t in_tensor_ch, int8_t *
```

in_tensor, int16_t * tmp_buffer, int8_t * out_tensor)

描述：

该函数是一个最大池函数，用于在任意 x 和 y 维度的有符号 8 位整数输入，以及使用激活参数限制输出。

参数：

- [in] in_tensor_dim_y 输入张量的 y 维度
- [in] in_tensor_dim_x 输入张量的 x 维度
- [in] out_tensor_dim_y 输出张量的 y 维度
- [in] out_tensor_dim_x 输出张量的 x 维度
- [in] stride_y y 维度的卷积步距
- [in] stride_x x 维度的卷积步距
- [in] ker_dim_y 卷积核的 y 维度
- [in] ker_dim_x 卷积核的 x 维度
- [in] pad_y y 维度的填充大小
- [in] pad_x x 维度的填充大小
- [in] act_min 限制输出张量的最小值，取值范围应为-128~127
- [in] act_max 限制输出张量的最大值，取值范围应为-128~127
- [in] in_tensor_ch 输入张量的通道数量
- [in] *in_tensor 指向输入张量的指针
- [in] *tmp_buf 空指针
- [out] *out_tensor 指向输出张量的指针

返回值：

0

3.7 归一化指数函数

归一化指数函数是以 2 为底的指数函数。

3.7.1 riscv_nn_softmax_s8_fast

原型：

void riscv_nn_softmax_s8_fast (const q7_t * in_vec, const uint16_t size, q7_t * out_vec)

描述：

该函数是一个 softmax 函数，用于有符号 8 位整数输入向量。

参数：

- [in] *in_vec 指向输入向量的指针
- [in] size 输入向量的元素数量
- [out] *out_vec 指向输出向量的指针

返回值：

无

示例：

```
#define LENGTH 10
q7_t in_data[LENGTH] = {...};
q7_t out_data[LENGTH];
riscv_nn_softmax_s8_fast (in_data, LENGTH, out_data);
```

3.7.2 riscv_nn_softmax_s16_fast

原型：

void riscv_nn_softmax_s16_fast (const q15_t * in_vec, const uint16_t size, q15_t * out_vec)

描述：

该函数是一个 softmax 函数，用于有符号 16 位整数输入向量。

参数：

- [in] *in_vec 指向输入向量的指针
- [in] size 输入向量的元素数量
- [out] *out_vec 指向输出向量的指针

返回值：

无

3.7.3 riscv_nn_softmax_s8_hp

原型：

void riscv_nn_softmax_s8_hp (const int8_t * in_tensor, const int32_t in_tensor_row, const int32_t in_tensor_col, const int32_t scale, const int32_t lshift, const int32_t diff_min, int8_t * out_tensor)

描述：

该函数是一个 softmax 函数，用于有符号 8 位整数输入张量以及高精度算法。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_row 输入张量的行数
- [in] in_tensor_col 输入张量的列数
- [in] scale 输入量化的缩放值
- [in] lshift 输入量化的左移量
- [in] diff_min 执行量化指数运算的最小阈值，差值可以通过行中最大值减去输入值而得到
- [out] *out_tensor 指向输出张量的指针

返回值：

无

3.7.4 riscv_nn_softmax_u8_hp

原型：

void riscv_nn_softmax_u8_hp (const uint8_t * in_tensor, const int32_t in_tensor_row, const int32_t in_tensor_col, const int32_t scale, const int32_t lshift, const int32_t diff_min, uint8_t * out_tensor)

描述：

该函数是一个 softmax 函数，用于无符号 8 位整数输入张量以及高精度算法。

参数：

- [in] *in_tensor 指向输入张量的指针
- [in] in_tensor_row 输入张量的行数
- [in] in_tensor_col 输入张量的列数
- [in] scale 输入量化的缩放值
- [in] lshift 输入量化的左移量
- [in] diff_min 执行量化指数运算的最小阈值，差值可以通过行中最大值减去输入值而得到
- [out] *out_tensor 指向输出张量的指针

返回值：

无

3.7.5 riscv_nn_softmax_f16

原型：

int32_t riscv_nn_softmax_f16 (const float16_t * in_vec, const uint32_t size, float16_t * out_vec)

描述：

该函数是一个 softmax 函数，用于半精度浮点输入向量。

参数：

- [in] *in_vec 指向输入向量的指针
- [in] size 输入向量的元素数量
- [out] *out_vec 指向输出向量的指针

返回值：

0

3.8 工具类辅助函数

工具类辅助函数是各种辅助工具。

3.8.1 riscv_nn_exp_f16

原型：

int32_t riscv_nn_exp_f16 (const float16_t * in_vec, const uint32_t size, float16_t * out_vec)

描述：

该函数用于计算半精度浮点输入的底数 e 指数。

参数：

- [in] *in_vec 指向输入向量的指针
- [in] size 输入向量的元素数量
- [out] *out_vec 指向输出向量的指针

返回值：

0

3.8.2 riscv_nn_reshape_s8

原型：

void riscv_nn_reshape_s8 (const int8_t * in_tensor, int8_t * out_tensor, const uint32_t size)

描述：

该函数用于将输入张量转换为另一个张量，这两个张量具有相同的数据但形状不同。

参数：

- [in] *in_tensor 指向输入张量的指针
- [out] *out_tensor 指向输出张量的指针
- [in] size 以字节为单元，总输入张量的大小

返回值：

无

示例：

```
#define SIZE 1024
int8_t in_tensor[SIZE] = {...};
int8_t out_tensor[SIZE];
riscv_nn_reshape_s8 (in_tensor, out_tensor, SIZE);
```

3.8.3 riscv_nn_top_k_s8

原型：

int32_t riscv_nn_top_k_s8 (q7_t * in_vec, uint32_t size, uint32_t k, q7_t * val, uint32_t * idx)

描述：

该函数用于从有符号 8 位整数输入向量中找出 k 个最大的值及其下标。

参数：

- [in] *in_vec 指向输入向量的指针
- [in] size 输入向量的元素数量
- [in] k 要搜索的最大值的个数
- [out] *val 指向输入向量中 k 个最大值的指针
- [out] *idx 指向输入向量中 k 个最大值下标的指针

返回值：

0

注！

k 个最大值从大到小排序，存储在“val”输出向量中。如果多个元素具有相同的值，则索引较小的元素具有更高的优先级。

3.8.4 riscv_nn_top_k_f16

原型：

int32_t riscv_nn_top_k_f16 (float16_t * in_vec, uint32_t size, uint32_t k, float16_t * val, uint32_t * idx)

描述：

该函数用于从半精度浮点输入向量中找出 k 个最大的值及其下标。

参数：

- [in] *in_vec 指向输入向量的指针
- [in] size 输入向量的元素数量
- [in] k 要搜索的最大值的个数
- [out] *val 指向输入向量中 k 个最大值的指针
- [out] *idx 指向输入向量中 k 个最大值下标的指针

返回值：

0

注！

k 个最大值从大到小排序，存储在“val”输出向量中。如果多个元素具有相同的值，则索引较小的元素具有更高的优先级。

4 应用程序

4.1 NN CIFAR-10

4.1.1 程序描述

该程序包含卷积、ReLU、最大池化、全连接等函数，演示了 RISC-V 处理器的加速运算，通过训练的 CIFAR-10 NN 模型，推理和识别 10 个目标 0~9，分别为飞机、汽车、鸟、猫、鹿、狗、青蛙、马、船和卡车。

该程序的推理结果显示目标“8”为最高得分 127，其他目标得分为 0，所以测试所用的输入图像最有可能是船。

该程序的输出信息包括指令计数、周期计数、测试图像的推理分数和结果精确度验证。

4.1.2 应用程序

RiscV_AE350_SOC 提供 NN CIFAR-10 应用程序设计：

…\ref_design\MCU_RefDesign\ae350_nn_cifar10。

4.1.3 程序运行

NN CIFAR-10 应用程序运行结果如下所示。

```
It is a Neural Network Model with the CIFAR-10 dataset demo.
Start execution
Start...
after init the checksum of img_buffer2 = 00000000093b43f6
after riscv_nn_conv_HWC_s8_s8_s8_RGB_sft_bias_fast the checksum of img_buffer1 = ffffffffc8e2c90
```

```
after riscv_nn_relu_s8 the checksum of img_buffer1 = 000000000023b2f0

after riscv_nn_maxpool_HWC_s8 the checksum of img_buffer2 = 00000000001c94ee

after riscv_nn_conv_HWC_s8_s8_s8_sft_bias_fast the checksum of img_buffer1 = ffffffffffb1ed7e

after riscv_nn_relu_s8 the checksum of img_buffer1 = 000000000007f614

after riscv_nn_maxpool_HWC_s8 the checksum of img_buffer2 = 0000000000076bca

after riscv_nn_conv_HWC_s8_s8_s8_sft_bias_fast the checksum of img_buffer1 = fffffffffe8a57b4

after riscv_nn_relu_s8 the checksum of img_buffer1 = 0000000000012fc8

after riscv_nn_maxpool_HWC_s8 the checksum of img_buffer2 = 0000000000018662

after riscv_nn_fc_s8_s8_s8_sft_bias_fast the checksum of output_data = 000000000000464c

after riscv_nn_softmax_s8_fast the checksum of output_data = 00000000001f4208

Inst: 52064421, Cycle: 374343076

0:airplane       0

1:automobile          0

2:bird       0

3:cat         0

4:deer        0

5:dog         0

6:frog       0

7:horse       0

8:ship       127

9:truck       0

>>>>> Accuracy checking 'ship' PASS. <<<<<
```

4.2 NN MobileNet-V1 INT8

4.2.1 程序描述

该程序演示了 NN MobileNet 模型。MobileNet 是使用深度可分离卷积来构建轻量级和高效深度神经网络的模型，主要用于移动和嵌入式视觉领域来处理目标检测和图像分类等应用。

该程序是一个量化为 INT8 的 MobileNet 模型，输出信息包括指令计数、周期计数、测试图像的推理分数和结果精确度验证。

4.2.2 应用程序

RiscV_AE350_SOC 提供 NN MobileNet-V1 INT8 应用程序设计：

…\ref_design\MCU_RefDesign\ae350_nn_mobilenet_v1_int8。

4.2.3 程序运行

NN MobileNet-V1 INT8 应用程序运行结果如下所示。

```
It's a Neural Network with MobileNet-V1 INT8 model demo.
Start executing...
Start...
after init the checksum of image_data = 00000001233bdcf2
after riscv_nn_conv_HWC_s8_s8_s8_RGB_sym_bias_fast the checksum of buffer1 = ffffffc9e93e9ee
after riscv_nn_relu_s8 the checksum of buffer1 = 0000001d0d2d385a
after riscv_nn_conv_dw_HWC_s8_s8_s8_sym_bias the checksum of buffer2 = ffffffe68949866
after riscv_nn_relu_s8 the checksum of buffer2 = 00000009f1729db6
after riscv_nn_conv_1x1_HWC_s8_s8_s8_sym_bias_fast_any the checksum of buffer1 = 00000002a28c0594
after riscv_nn_relu_s8 the checksum of buffer1 = 000000064e9bde6c
after riscv_nn_conv_dw_HWC_s8_s8_s8_sym_bias the checksum of buffer2 = 0000000143325e50
after riscv_nn_relu_s8 the checksum of buffer2 = 000000039f7f55c0
after riscv_nn_conv_1x1_HWC_s8_s8_s8_sym_bias_fast_any the checksum of buffer1 = 00000003d23f3a6a
after riscv_nn_relu_s8 the checksum of buffer1 = 000000085ec8093a
```

```
after riscv_nn_conv_dw_HWC_s8_s8_s8_sym_bias the checksum of buffer2 = 000000005f0a4bf0

after riscv_nn_relu_s8 the checksum of buffer2 = 0000000316896b76

after riscv_nn_conv_1x1_HWC_s8_s8_s8_sym_bias_fast_any the checksum of buffer1 = ffffffff66efeafc

after riscv_nn_relu_s8 the checksum of buffer1 = 000000012361dbc4

after riscv_nn_conv_dw_HWC_s8_s8_s8_sym_bias the checksum of buffer2 = 00000000a16ee37e

after riscv_nn_relu_s8 the checksum of buffer2 = 00000000b4775278

after riscv_nn_conv_1x1_HWC_s8_s8_s8_sym_bias_fast_any the checksum of buffer1 = 000000018a0613ca

after riscv_nn_relu_s8 the checksum of buffer1 = 000000019a0099e6

after riscv_nn_conv_dw_HWC_s8_s8_s8_sym_bias the checksum of buffer2 = 00000000114b864a

after riscv_nn_relu_s8 the checksum of buffer2 = 000000009287b20c

after riscv_nn_conv_1x1_HWC_s8_s8_s8_sym_bias_fast_any the checksum of buffer1 = fffffffffe6e7220

after riscv_nn_relu_s8 the checksum of buffer1 = 000000004624aba0

after riscv_nn_conv_dw_HWC_s8_s8_s8_sym_bias the checksum of buffer2 = 000000005b0da5fa

after riscv_nn_relu_s8 the checksum of buffer2 = 0000000068700136

after riscv_nn_conv_1x1_HWC_s8_s8_s8_sym_bias_fast_any the checksum of buffer1 = 00000000871ff516

after riscv_nn_relu_s8 the checksum of buffer1 = 000000009229946c

after riscv_nn_conv_dw_HWC_s8_s8_s8_sym_bias the checksum of buffer2 = ffffffffcd3e2ca2

after riscv_nn_relu_s8 the checksum of buffer2 = 000000003fd73c5c

after riscv_nn_conv_1x1_HWC_s8_s8_s8_sym_bias_fast_any the checksum of buffer1 = 000000000b876592

after riscv_nn_relu_s8 the checksum of buffer1 = 000000001e2aa242

after riscv_nn_conv_dw_HWC_s8_s8_s8_sym_bias the checksum of buffer2 = fffffffe797f224

after riscv_nn_relu_s8 the checksum of buffer2 = 000000001358ebb4

after riscv_nn_conv_1x1_HWC_s8_s8_s8_sym_bias_fast_any the checksum of buffer1 = 0000000001f8766a
```

```
after riscv_nn_relu_s8 the checksum of buffer1 = 00000000189c9a12

after riscv_nn_conv_dw_HWC_s8_s8_s8_sym_bias the checksum of buffer2 = 0000000007f8becc

after riscv_nn_relu_s8 the checksum of buffer2 = 0000000050d2bdd6

after riscv_nn_conv_1x1_HWC_s8_s8_s8_sym_bias_fast_any the checksum of buffer1 = ffffffffffd3b226

after riscv_nn_relu_s8 the checksum of buffer1 = 000000000a8bd970

after riscv_nn_conv_dw_HWC_s8_s8_s8_sym_bias the checksum of buffer2 = 00000000114adb1e

after riscv_nn_relu_s8 the checksum of buffer2 = 000000003179d8fc

after riscv_nn_conv_1x1_HWC_s8_s8_s8_sym_bias_fast_any the checksum of buffer1 = fffffffffd846a78

after riscv_nn_relu_s8 the checksum of buffer1 = 000000000b6d1b50

after riscv_nn_conv_dw_HWC_s8_s8_s8_sym_bias the checksum of buffer2 = 000000000b208e9a

after riscv_nn_relu_s8 the checksum of buffer2 = 00000000202d3a44

after riscv_nn_conv_1x1_HWC_s8_s8_s8_sym_bias_fast_any the checksum of buffer1 = 00000000056b9676

after riscv_nn_relu_s8 the checksum of buffer1 = 0000000014d57b56

after riscv_nn_conv_dw_HWC_s8_s8_s8_sym_bias the checksum of buffer2 = 000000000a4c525c

after riscv_nn_relu_s8 the checksum of buffer2 = 000000000e55a490

after riscv_nn_conv_1x1_HWC_s8_s8_s8_sym_bias_fast_any the checksum of buffer1 = ffffffff4d4486a

after riscv_nn_relu_s8 the checksum of buffer1 = 0000000002f7646e

after riscv_nn_conv_dw_HWC_s8_s8_s8_sym_bias the checksum of buffer2 = 000000000cfd3304

after riscv_nn_relu_s8 the checksum of buffer2 = 00000000266a863c

after riscv_nn_conv_1x1_HWC_s8_s8_s8_sym_bias_fast_any the checksum of buffer1 = fffffffed98ba3a

after riscv_nn_relu_s8 the checksum of buffer1 = 0000000000bb5982

after riscv_nn_avepool_HWC_s8 the checksum of buffer2 = 0000000000000001

after riscv_nn_fc_s8_s8_s8_sym_bias the checksum of out_data = 00000000000baa8e
```

```
after riscv_nn_softmax_s8_fast the checksum of out_data = 00000000001f45e6
Inst: 90956449, Cycle: 119727866
>>>>> Accuracy checking PASS. <<<<<
```

4.3 NN TinyYOLO-V3

4.3.1 程序描述

该程序演示了 NN TinyYOLO-V3 模型。TinyYOLO 是实时目标检测系统“You Only Look Once”（YOLO）的一个变体，对于受约束的环境具有较小的模型尺寸。

该程序的 TinyYOLO-V3 是 YOLO 的一种轻量化版本，具有简化的骨干网络和两级特征图检测，主要为资源受限的设备（例如嵌入式系统、移动设备或物联网设备）设计。

该程序的输出信息包括指令计数、周期计数和结果精确度验证。

4.3.2 应用程序

RiscV_AE350_SOC 提供 NN TinyYOLO-V3 应用程序设计：

…\ref_design\MCU_RefDesign\ae350_nn_tinyyolo_v3。

4.3.3 程序运行

NN TinyYOLO-V3 应用程序运行结果如下所示。

```
It's a Neural Network with TinyYOLO v3 model demo.
Start executing model...
Start...
after init the checksum of image_data = 00000036bdd17bbc
after riscv_nn_conv_HWC_s8_s8_s8_RGB_sym_bias_fast the checksum of buffer1 = ffffff2efaf75b4
after riscv_nn_leaky_relu_s8 the checksum of buffer1 = 00000097c470d19c
after riscv_nn_maxpool_HWC_s8 the checksum of buffer2 = 000000034f10415c
after riscv_nn_conv_HWC_s8_s8_s8_sym_bias_fast the checksum of buffer1 = 0000000593c74a7c
```

```
after riscv_nn_leaky_relu_s8 the checksum of buffer1 =
00000113b0f644ba

after riscv_nn_maxpool_HWC_s8 the checksum of buffer2 =
0000000362cb292c

after riscv_nn_conv_HWC_s8_s8_s8_sym_bias_fast the checksum of
buffer1 = ffffffddb234434

after riscv_nn_leaky_relu_s8 the checksum of buffer1 =
000000679282c680

after riscv_nn_maxpool_HWC_s8 the checksum of buffer2 =
00000000fca7413c

after riscv_nn_conv_HWC_s8_s8_s8_sym_bias_fast the checksum of
buffer1 = ffffffeea8c97a0

after riscv_nn_leaky_relu_s8 the checksum of buffer1 =
00000033b51118da

after riscv_nn_maxpool_HWC_s8 the checksum of buffer2 =
000000003fcde4b0

after riscv_nn_conv_HWC_s8_s8_s8_sym_bias_fast the checksum of
buffer1 = ffffffffb1c72628

after riscv_nn_leaky_relu_s8 the checksum of buffer1 =
0000000a6634634c

after riscv_nn_maxpool_HWC_s8 the checksum of buffer2 =
0000000007f79719

after riscv_nn_conv_HWC_s8_s8_s8_sym_bias_fast the checksum of
buffer1 = ffffffffac79e31a

after riscv_nn_leaky_relu_s8 the checksum of buffer1 =
0000000cf280f62a

after riscv_nn_maxpool_HWC_s8 the checksum of buffer2 =
000000000f8e086f

after riscv_nn_conv_HWC_s8_s8_s8_sym_bias_fast the checksum of
buffer1 = 00000000469cc836

after riscv_nn_leaky_relu_s8 the checksum of buffer1 =
0000001a1a8ff9aa

after riscv_nn_conv_HWC_s8_s8_s8_sym_bias_fast the checksum of
buffer2 = ffffffffa7c14f22

after riscv_nn_leaky_relu_s8 the checksum of buffer2 =
000000065365f146

======run route...
```

```
after riscv_nn_conv_HWC_s8_s8_s8_sym_bias_fast the checksum of buffer3 = ffffffffd7010c7a

after riscv_nn_leaky_relu_s8 the checksum of buffer3 = 00000003308f3bfe

after riscv_nn_conv_HWC_s8_s8_s8_sym_bias_fast the checksum of buffer4 = ffffffff2d3a0b4a

after riscv_nn_leaky_relu_s8 the checksum of buffer4 = 00000021c13e7396

after riscv_nn_conv_HWC_s8_s8_s8_sym_bias_fast the checksum of buffer3 = 000000151f27449e

======run route end...

after riscv_nn_conv_HWC_s8_s8_s8_sym_bias_fast the checksum of buffer1 = ffffffffff220b478

after riscv_nn_leaky_relu_s8 the checksum of buffer1 = 0000000d1339f80c

after riscv_nn_conv_HWC_s8_s8_s8_sym_bias_fast the checksum of buffer2 = 000000064cfe76f5

Inst: -174936110, Cycle: 1589518486

Done
```

4.4 NN TinyYOLO-V1

4.4.1 程序描述

该程序演示了 NN TinyYOLO-V1 模型。TinyYOLO 是实时目标检测系统“You Only Look Once”（YOLO）的一个变体，对于受约束的环境具有较小的模型尺寸。

该程序的 TinyYOLO-V1 是一个通过使用对称激活数据（即强制零点等于 0）的方案量化为 INT8 的 TinyYOLO 模型。

该程序的输出信息包括指令计数、周期计数和结果精确度验证。

4.4.2 应用程序

RiscV_AE350_SOC 提供 NN TinyYOLO-V1 应用程序设计：

…\ref_design\MCU_RefDesign\ae350_nn_tinyyolo_v1。

4.4.3 程序运行

NN TinyYOLO-V1 应用程序运行结果如下所示。

```
It's a Neural Network with TinyYOLO v1 model demo.
Start executing model...
Start...
after init the checksum of image_data = 00000036bdd17bbc
after riscv_nn_conv_HWC_s8_s8_s8_RGB_sym_bias_fast the checksum of buffer1 = fffffffe476b89c4
after riscv_nn_leaky_relu_s8 the checksum of buffer1 = 00000001473069d6
after riscv_nn_maxpool_HWC_s8 the checksum of buffer2 = 000000000b3008e4
after riscv_nn_conv_HWC_s8_s8_s8_sym_bias_fast the checksum of buffer1 = fffffff21105a370
after riscv_nn_leaky_relu_s8 the checksum of buffer1 = 0000000075d18612
after riscv_nn_maxpool_HWC_s8 the checksum of buffer2 = 00000000017914b2
after riscv_nn_conv_HWC_s8_s8_s8_sym_bias_fast the checksum of buffer1 = fffffffbc5f86c94
after riscv_nn_leaky_relu_s8 the checksum of buffer1 = 000000005b321f12
after riscv_nn_maxpool_HWC_s8 the checksum of buffer2 = 0000000000d8461e
after riscv_nn_conv_HWC_s8_s8_s8_sym_bias_fast the checksum of buffer1 = fffffffdf2c7fe70
after riscv_nn_leaky_relu_s8 the checksum of buffer1 = 000000002003068c
after riscv_nn_maxpool_HWC_s8 the checksum of buffer2 = 00000000001a0532
after riscv_nn_conv_HWC_s8_s8_s8_sym_bias_fast the checksum of buffer1 = fffffffebc0181aa
after riscv_nn_leaky_relu_s8 the checksum of buffer1 = 0000000007d995d0
```

```
after riscv_nn_maxpool_HWC_s8 the checksum of buffer2 = 00000000000320fa

after riscv_nn_conv_HWC_s8_s8_s8_sym_bias_fast the checksum of buffer1 = ffffffff1af40ea2

after riscv_nn_leaky_relu_s8 the checksum of buffer1 = 0000000002090daa

after riscv_nn_maxpool_HWC_s8 the checksum of buffer2 = 0000000000005da3

after riscv_nn_conv_HWC_s8_s8_s8_sym_bias_fast the checksum of buffer1 = ffffffffb491053a

after riscv_nn_leaky_relu_s8 the checksum of buffer1 = 000000000268bece

after riscv_nn_conv_HWC_s8_s8_s8_sym_bias_fast the checksum of buffer2 = ffffffffecc0bd14

after riscv_nn_leaky_relu_s8 the checksum of buffer2 = 0000000000af9b74

after riscv_nn_fc_s8_s8_s8_sym_bias the checksum of out_data = 0000000001a5f0a0

Inst: -1345112762, Cycle: 1042775620

Done
```


GOWIN高云
智慧逻辑 定制未来