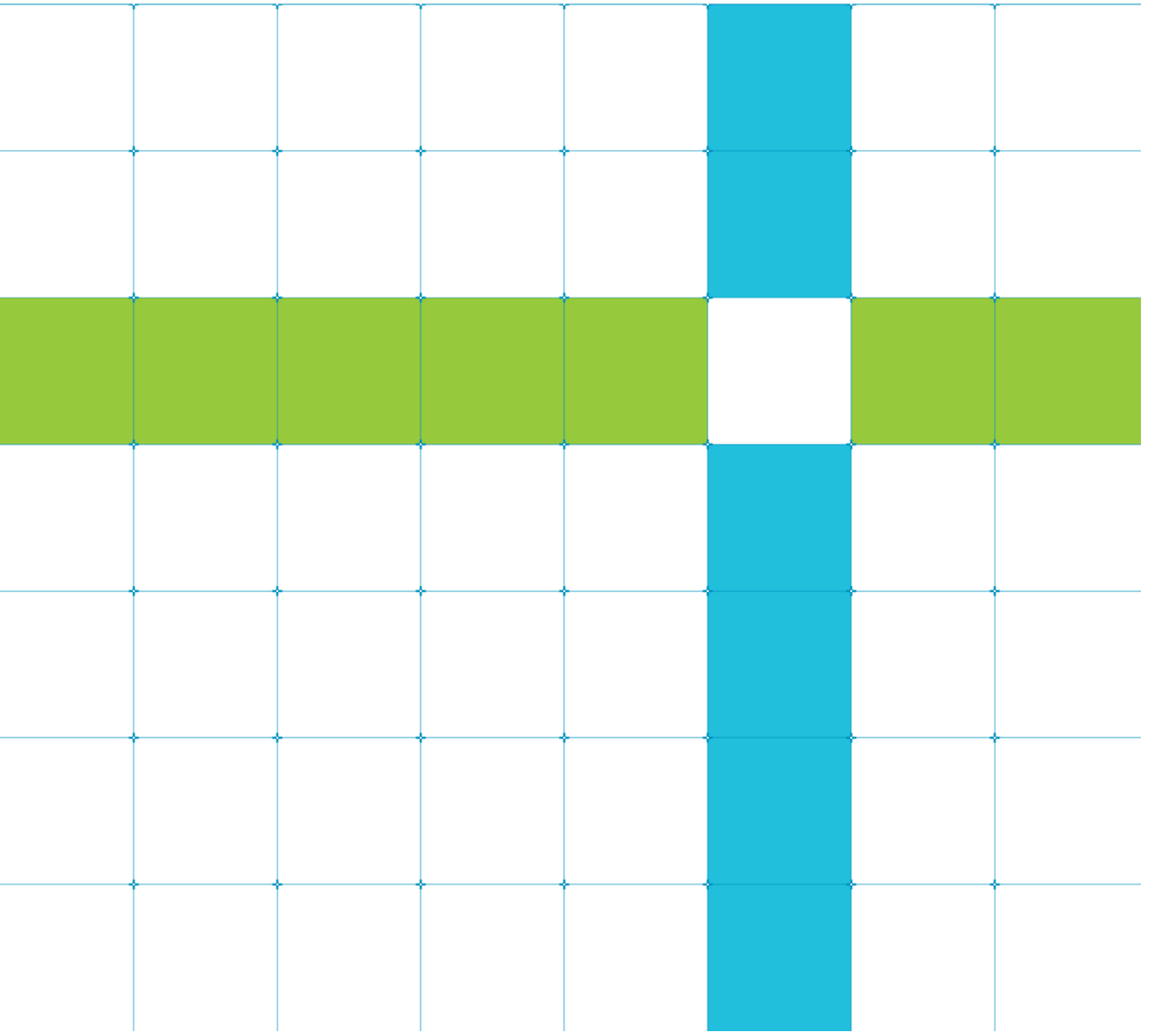
arm CHINA

Application Note

Zhouyi Compass Operators Specification

Version 7.3
Document ID: ACN-61010017-010
Confidential



Confidential Proprietary Notice

This document is **CONFIDENTIAL** and any use by you is subject to the terms of the agreement between you and Arm Technology (China) Co., Ltd ("Arm China") or the terms of the agreement between you and the party authorized by Arm China to disclose this document to you.

This document is protected by copyright and other related rights and the practice or implementation of the information contained in this document may be protected by one or more patents or pending patent applications. No part of this document may be reproduced in any form by any means without the express prior written permission of Arm China. **No license, express or implied, by estoppel or otherwise to any intellectual property rights is granted by this document unless specifically stated.**

Your access to the information in this document is conditional upon your acceptance that you will not use or permit others to use the information: **(i)** for the purposes of determining whether implementations infringe any third party patents; **(ii)** for developing technology or products which avoid any of Arm China's intellectual property; or **(iii)** as a reference for modifying existing patents or patent applications or creating any continuation, continuation in part, or extension of existing patents or patent applications; or **(iv)** for generating data for publication or disclosure to third parties, which compares the performance or functionality of the Arm China technology described in this document with any other products created by you or a third party, without obtaining Arm China's prior written consent.

THIS DOCUMENT IS PROVIDED "AS IS". ARM CHINA PROVIDES NO REPRESENTATIONS AND NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY, SATISFACTORY QUALITY, NON-INFRINGEMENT OR FITNESS FOR A PARTICULAR PURPOSE WITH RESPECT TO THE DOCUMENT. For the avoidance of doubt, Arm China makes no representation with respect to, and has undertaken no analysis to identify or understand the scope and content of, third party patents, copyrights, trade secrets, or other rights.

This document may include technical inaccuracies or typographical errors.

TO THE EXTENT NOT PROHIBITED BY LAW, IN NO EVENT WILL ARM CHINA BE LIABLE FOR ANY DAMAGES, INCLUDING WITHOUT LIMITATION ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, PUNITIVE, OR CONSEQUENTIAL DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF ANY USE OF THIS DOCUMENT, EVEN IF ARM CHINA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

This document consists solely of commercial items. You shall be responsible for ensuring that any use, duplication or disclosure of this document complies fully with any relevant export laws and regulations to assure that this document or any portion thereof is not exported, directly or indirectly, in violation of such export laws. Use of the word "partner" in reference to Arm China's customers is not intended to create or refer to any partnership relationship with any other company. Arm China may make changes to this document at any time and without notice.

If any of the provisions contained in these terms conflict with any of the provisions of any click through or signed written agreement covering this document with Arm China, then the click through or signed written agreement prevails over and supersedes the conflicting provisions of these terms. This document may be translated into other languages for convenience, and you agree that if there is any conflict between the English version of this document and any translation, the terms of the English version of the Agreement shall prevail.

Release Information

Document History

Issue	Date	Confidentiality	Change
A	18/06/2021	Confidential	First draft
B	30/09/2021	Confidential	Added more operators
C	21/12/2021	Confidential	Updated some operator descriptions
D	31/03/2022	Confidential	Deleted a few operators and added the main execution unit information
E	30/06/2022	Confidential	Added more operators and quantization method support information
F	30/09/2022	Confidential	Added more operators and the operator map
G	31/12/2022	Confidential	Added more operators and some comments
H	31/03/2023	Confidential	Added more operators
I	30/06/2023	Confidential	Add more operators and update some operator descriptions
J	30/09/2023	Confidential	Add more operators and update some operator descriptions

Contents

1 About this document 4
1.1. References 4
1.2. Terms and abbreviations 4
1.3. Conventions and feedback 5
1.3.1 Feedback on this product 5
1.3.2 Feedback on documentation 5

2 Introduction 6

3 Operators 6

4 Operator map 147

1 About this document

This Application Note is intended for developers/programmers/users who use the Arm China Zhouyi Compass. This Application Note gives you a basic understanding of Zhouyi Compass *Neural Processing Unit* (NPU) operators and describes how to use them with the Zhouyi Compass.

In this document, the *Artificial Intelligence Processing Unit* (AIPU) has the same meaning as the NPU.

1.1. References

Reference	Document number	Title
1	61010011_0303_00	Arm China Zhouyi Compass Software Technical Overview
2	ACN-61010013-010	Arm China Zhouyi Compass IR Definition Application Note

1.2. Terms and abbreviations

This document uses the following terms and abbreviations.

Term	Meaning
N	Batch number
H	Height
W	Width
C	Channel
NHWC	A layout for tensor with the storing order in batch, height, width, channel.
NCHW	A layout for tensor with the storing order in batch channel, height, width.
NCHWC’	A layout for tensor with the storing order in batch, channel/C', height, width, C', where C' can be 16 or 32. This format is aligned in C, which is more friendly for hardware. Normally, this format has better performance in hardware.
NCHWC32	A layout for tensor which is a case of NCHWC' where C'=32.
NCHWC16	A layout for tensor which is a case of NCHWC' where C'=16.
NTC	A layout for tensor with the storing order in batch, timestep, channel.
Shape_dims	The total dimension of the tensor.
Shape_size	Cumulative multiplication of all dimensions for the tensor. If it is a 4-dimensional tensor, the shape_size is dim[0] * dim[1] * dim[2] * dim[3].
Ranges	Using the mathematically common representation of the range of values. For example: $x \in [a,b] \Longleftrightarrow x \text{ in } \{a,a+1,\ldots,b-1,b\}$, $x \in (a,b) \Longleftrightarrow x \text{ in } \{a+1,\ldots,b-1\}$, $a < b$

1.3. Conventions and feedback

The following describes the typographical conventions and how to give feedback:

Convention	Meaning
monospace	denotes text that can be entered at the keyboard, such as commands, file and program names, and source code.
monospace	denotes a permitted abbreviation for a command or option. The underlined text can be entered instead of the full command or option name.
monospace italic	denotes arguments to commands and functions where the argument is to be replaced by a specific value.
monospace bold	denotes language keywords when used outside example code.
italic	highlights important notes, introduces special terminology, denotes internal cross-references, and citations.
bold	highlights interface elements, such as menu names. Also used for emphasis in descriptive lists, where appropriate, and for Arm China processor signal names.

1.3.1 Feedback on this product

If you have any comments and suggestions about this product, contact your supplier and give:

- Your name and company.
- The serial number of the product.
- Details of the release you are using.
- Details of the platform you are using, such as the hardware platform, operating system type and version.
- A small standalone sample of code that reproduces the problem.
- A clear explanation of what you expected to happen, and what actually happened.
- The commands you used, including any command-line options.
- Sample output illustrating the problem.
- The version string of the tools, including the version number and build numbers.

1.3.2 Feedback on documentation

If you have comments on the documentation, e-mail errata@armchina.com. Give:

- The title.
- The number, [Document ID Value], [Issue].
- If viewing online, the topic names to which your comments apply.
- If viewing a PDF version of a document, the page numbers to which your comments apply.
- A concise explanation of your comments.

Arm China also welcomes general suggestions for additions and improvements.

2 Introduction

The AIPUBuilder in Zhouyi Compass supports operators of many neural networks. This document lists all supported operators and describes their specification.

For more information about the names and definitions of the operators, see the *Arm China Zhouyi Compass IR Definition Application Note*.

3 Operators

This section describes all officially supported built-in operators.

Except in special cases, all input and output tensors must meet the following conditions:

- shape_size <= 64MB
- dim[0] <= 32

Note

When using the asymmetric per-tensor quantization method, the operator will include two attributes `layer_top_scale` and `layer_top_zp` with values in the ranges [0, ∞] and [-255, 255], respectively.

Abs

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape. Converted to leakyrelu in GBuilder.

AccidentalHits

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	2	-
input shape(s)	Input0: Supports int16 data type. Supports 2-dimensional input. Dim[0] ∈ [1, 1920] Dim[1] ∈ [1, 16384] Input1: Supports int16 data type. Supports 1-dimensional input. Dim[0] ∈ [1, 16384]	-

Acos

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

Acosh

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

Add

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	2	-
Number of output tensors	1	-
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims:	Input format can be: [[C],[1]], [[N,C],[1]], [[N,C],[C]], [[N,C],[N,1]], [[N,H,C],[1]], [[N,H,C],[C]], [[N,H,C],[H,1]], [[N,H,C],[N,1,1]], [[N,H,W,C],[1]], [[N,H,W,C],[C]], [[N,H,W,C],[W,1]], [[N,H,W,C],[H,1,1]], [[N,H,W,C],[N,1,1,1]], and the order of the two inputs can be swapped.

Parameter	Valid value or range	Comment
	Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096] Input1: Supports int8, uint8, int16, and uint16 data types. Supports 2-dimensional input.	
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

ArgMax

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports 1–5 dimensional input, with no size limit for every dimension.
method	{MAX}	-
axis	[0, 4]	-
select_last_index	{true, false}	-

ArgMin

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports 1–5 dimensional input, with no size limit for every dimension.
method	{MIN}	-
axis	[0, 4]	-
select_last_index	{true, false}	-

Asin

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

Asinh

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

AveragePooling2D

The main execution unit: TPC or AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384]	-

Parameter	Valid value or range	Comment
	Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 16384]	
kernel_x	[1, 65]	The kernel can exceed 65 when the following conditions are met: (kernel_x==input_shape[2](w) and kernel_y==input_shape[1](h), and (kernel_x*kernel_y < 28*1024 or kernel_x*kernel_y*bit(input_data) < max(int32)))
kernel_y	[1, 65]	The kernel can exceed 65 when the following conditions are met: (kernel_x==input_shape[2](w) and kernel_y==input_shape[1](h), and (kernel_x*kernel_y < 28*1024 or kernel_x*kernel_y*bit(input_data) < max(int32)))
stride_x	[1, 6]	The following must be true: • stride_x <= kernel_x • stride_y <= kernel_y Or: • kernel_x == kernel_y == 1 • stride_x == stride_y == 2 when the pooling is global avg pooling and the stride has no limit.
stride_y	[1, 6]	The following must be true: • stride_x <= kernel_x • stride_y <= kernel_y Or: • kernel_x == kernel_y == 1 • stride_x == stride_y == 2 when the pooling is global avg pooling and the stride has no limit.
pad_top	[0, 6]	The following must be true: • pad_top/bottom < kernel_y • pad_left/right < kernel_x
pad_bottom	[0, 6]	The following must be true: • pad_top/bottom < kernel_y • pad_left/right < kernel_x
pad_left	[0, 6]	The following must be true: • pad_top/bottom < kernel_y • pad_left/right < kernel_x
pad_right	[0, 6]	The following must be true: • pad_top/bottom < kernel_y • pad_left/right < kernel_x
dilation_x	{1}	-
dilation_y	{1}	-

Parameter	Valid value or range	Comment
ceil_mode	{true, false}	-
count_include_pad	{true, false}	-
method	{AVG}	-

AveragePooling3D

The main execution unit: TPC or AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8 and uint8 data types. Supports 5-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 100] Dim[2] ∈ [1, 1080] Dim[3] ∈ [1, 1920] Dim[4] ∈ [1, 2048]	-
kernel_x	[1, 17]	kernel_x * kernel_y * kernel_z should be <= 256.
kernel_y	[1, 17]	kernel_x * kernel_y * kernel_z should be <= 256.
kernel_z	[1, 17]	kernel_x * kernel_y * kernel_z should be <= 256.
stride_x	[1, 8]	The following must be true: • stride_x <= kernel_x • stride_y <= kernel_y • stride_z <= kernel_z Or: • kernel_x == kernel_y == kernel_z == 1 • stride_x == stride_y == stride_z == 2
stride_y	[1, 8]	The following must be true: • stride_x <= kernel_x • stride_y <= kernel_y • stride_z <= kernel_z

Parameter	Valid value or range	Comment
		Or: • kernel_x == kernel_y == kernel_z == 1 • stride_x == stride_y == stride_z == 2
stride_z	[1, 8]	The following must be true: • stride_x <= kernel_x • stride_y <= kernel_y • stride_z <= kernel_z Or: • kernel_x == kernel_y == kernel_z == 1 • stride_x == stride_y == stride_z == 2
pad_x_begin	[0, 6]	The following must be true: • pad_x_begin/end < kernel_x • pad_y_begin/end < kernel_y • pad_z_begin/end < kernel_z
pad_x_end	[0, 6]	The following must be true: • pad_x_begin/end < kernel_x • pad_y_begin/end < kernel_y • pad_z_begin/end < kernel_z
pad_y_begin	[0, 6]	The following must be true: • pad_x_begin/end < kernel_x • pad_y_begin/end < kernel_y • pad_z_begin/end < kernel_z
pad_y_end	[0, 6]	The following must be true: • pad_x_begin/end < kernel_x • pad_y_begin/end < kernel_y • pad_z_begin/end < kernel_z
pad_z_begin	[0, 6]	The following must be true: • pad_x_begin/end < kernel_x • pad_y_begin/end < kernel_y • pad_z_begin/end < kernel_z
pad_z_end	[0, 6]	The following must be true: • pad_x_begin/end < kernel_x • pad_y_begin/end < kernel_y • pad_z_begin/end < kernel_z
dilation_x	{1}	-
dilation_y	{1}	-
dilation_z	{1}	-
ceil_mode	{true, false}	-
count_include_pad	{true, false}	-
method	{AVG}	-

BNLL

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384]	-

BasicLSTM

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	N

Parameter	Valid value or range	Comment
Number of input tensors	3	-
Number of output tensors	1-3	-
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Supports 3-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 3071] Dim[2] ∈ [1, 3072] Input1: Supports int8 and int16 data types. Supports 2-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 3072] Input2: Supports int8 and int16 data types. Supports 2-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 3072]	• Input0: {X} • Input1: {HO} • Input2: {CO}
output shape(s)	Output0_2: Supports int8 and int16 data types.	Output: {Y}, {H}, {C}, {Y, H}, {Y, C}, {H, C} or {Y, H, C}, which depends on out_sequence.
out_sequence	{{Y}, {H}, {C}, {Y, H}, {Y, C}, {H, C}, {Y, H, C}}	-
activations	{Tanh, Sigmoid}	-
direction	{Forward, Reverse}	-

BatchNormalization

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	• For 2-dims, axis == 1 • For 3-dims, axis == 2 • For 4-dims, axis == 3. Supports any shape since Zhouyi Z2.
output shape(s)	Supports int8 and int16 data types.	-
axis	-	axis: ∈ [1, input_dims-1]

BatchToSpace

The main execution unit: DMA

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	-
block_size_x	[1, 16]	-
block_size_y	[1, 16]	-
crop_left	[0, 16]	-
crop_right	[0, 16]	-
crop_top	[0, 16]	-

Parameter	Valid value or range	Comment
crop_bottom	[0, 16]	-

BiasAdd

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	-

BitwiseAnd

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1-2	-
Number of output tensors	1	-

Parameter	Valid value or range	Comment
input shape(s)	Input0: Supports int8, uint8, int16, uint16, int32, and uint32 data types. Supports 2, 3, 4, 5-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] For 5-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] Dim[4] ∈ [1, 16384] Input1: Supports int8, uint8, int16, uint16, int32, and uint32 data types. Supports 2, 3, 4, 5-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384]	The shape of the second input is equal to the shape of the first input. Input1 may not exist. Supports any shape.

Parameter	Valid value or range	Comment
	For 5-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] Dim[4] ∈ [1, 16384]	
output shape(s)	Supports int8, uint8, int16, uint16, int32, and uint32 data types.	-
method	{AND}	-

BitwiseNot

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1-2	-
Number of output tensors	1	-
input shape(s)	Input0: Supports int8, uint8, int16, uint16, int32, and uint32 data types. Supports 2, 3, 4, 5-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32]	The shape of the second input is equal to the shape of the first input. Input1 may not exist. Supports any shape.

Parameter	Valid value or range	Comment
	Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] For 5-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] Dim[4] ∈ [1, 16384] Input1: Supports int8, uint8, int16, uint16, int32, and uint32 data types. Supports 2, 3, 4, 5-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] For 5-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] Dim[4] ∈ [1, 16384]	
output shape(s)	Supports int8, uint8, int16, uint16, int32, and uint32 data types.	-
method	{NOT}	-

BitwiseOr

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1-2	-
Number of output tensors	1	-
input shape(s)	Input0: Supports int8, uint8, int16, uint16, int32, and uint32 data types. Supports 2, 3, 4, 5-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] For 5-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] Dim[4] ∈ [1, 16384] Input1: Supports int8, uint8, int16, uint16,	The shape of the second input is equal to the shape of the first input. Input1 may not exist. Supports any shape.

Parameter	Valid value or range	Comment
	int32, and uint32 data types. Supports 2, 3, 4, 5-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] For 5-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] Dim[4] ∈ [1, 16384]	
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
method	{OR}	-

BitwiseXor

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1-2	-

Parameter	Valid value or range	Comment
Number of output tensors	1	-
input shape(s)	Input0: Supports int8, uint8, int16, uint16, int32, and uint32 data types. Supports 2, 3, 4, 5-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] For 5-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] Dim[4] ∈ [1, 16384] Input1: Supports int8, uint8, int16, uint16, int32, and uint32 data types. Supports 2, 3, 4, 5-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384]	The shape of the second input is equal to the shape of the first input. Input1 may not exist. Supports any shape.

Parameter	Valid value or range	Comment
	Dim[3] ∈ [1, 16384] For 5-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] Dim[4] ∈ [1, 16384]	
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
method	{XOR}	If method=='XOR', len(bottoms) ==1 & len(scale_value == 1).

BoundingBox

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	N
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	N

Parameter	Valid value or range	Comment
Number of input tensors	2	-
Number of output tensors	1	-
input shape(s)	Input0: Supports int16 data type. Supports 3-dimensional input. Dim[0] ∈ [1, 16] Dim[1] ∈ [1, 10000] Dim[2] ∈ [4, 4] Input1: Supports int8 data type. Supports 3-dimensional input. Dim[0] ∈ [1, 16]	-

Parameter	Valid value or range	Comment
	Dim[1] ∈ [1, 10000] Dim[2] ∈ [4, 4]	
output shape(s)	Supports int16 data type.	-

CRelu

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384]	-
method	{CRELU}	-
axis	-	axis = input_dims-1 or axis = -1

CTCGreedyDecoder

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	2	-
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Supports 3-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Input1: Supports uint16 data type. Supports 1-dimensional input. Dim[0] ∈ [1, 32]	-
merge_repeated	{true}	-

Cast

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.

Parameter	Valid value or range	Comment
ignore_scale_zp	{true, false}	`false` is the default mode.
clip_mode	{SATURETION, TRUNCATION}	`SATURETION` is the default mode.
to_dtype	{int8, uint8, int16, uint16, int32}	-

Ceil

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

Celu

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-

Parameter	Valid value or range	Comment
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
method	{CELU}	-

ChannelShuffle

The main execution unit: DMA

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384]	-
group	[1, 16384]	-
splits	[1, 16]	1 or input_shape[-1]/group

Clip

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.

Compress

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	N
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	2	-
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] Input1:	The shape of input1 should be <= input0.shape[axis].

Parameter	Valid value or range	Comment
	Supports int8 data type. Supports 1-dimensional input. Dim[0] ∈ [1, 16384]	
axis	[0, 3]	-

Concat

The main execution unit: DMA

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	2-20	-
input shape(s)	Input0_19: Supports int8, uint8, int16, and uint16 data types.	Supports any shape and any number of inputs.
axis	-	axis: ∈ [-1, input_dims-1]

Constant

The main execution unit: DMA

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	0	-
input shape(s)	-	Empty, that is, there is no input initial data tensor X.

Parameter	Valid value or range	Comment
weights_shape	-	Any length and dimension, as long as the total size meets the requirements

ConvTranspose2D

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384]	The input must be an NHWC format tensor.
kernel_x	[1, 64]	-
kernel_y	[1, 64]	-
stride_x	[1, 16]	-
stride_y	[1, 16]	-
pad_top	[0, 16]	-
pad_bottom	[0, 16]	-
pad_left	[0, 16]	-
pad_right	[0, 16]	-
dilation_x	[1, 16]	If dilation_x != 1, the following must be true: output_shape_size * sizeof (output_type) * stride_x * stride_y <= 1G
dilation_y	[1, 16]	If dilation_y != 1, the following must be true: output_shape_size * sizeof (output_type) * stride_x * stride_y <= 1G

Parameter	Valid value or range	Comment
group	{1}	-
with_activation	{NONE, RELU, CLIP, PRELU, LEAKYRELU}	-

ConvTranspose3D

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 5-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 100] Dim[2] ∈ [1, 1080] Dim[3] ∈ [1, 1920] Dim[4] ∈ [1, 4096]	The input must be an NDHWC format tensor. Dims[4] * Dims[2]: [1, 4096]
kernel_x	[1, 11]	-
kernel_y	[1, 11]	-
kernel_z	[1, 11]	The following must be true: (Input_c * kernel_z % 32 == 0 and 32<=input_c * kernel_z <=4096) Or: Input_c * kernel_z ∈ {1, 3, 4}
stride_x	[1, 6]	The following must be true: • stride_x <= kernel_x • stride_y <= kernel_y • stride_z <= kernel_z Or: • kernel_x == kernel_y == kernel_z == 1

Parameter	Valid value or range	Comment
		• stride_x == stride_y == stride_z == 2
stride_y	[1, 6]	The following must be true: • stride_x <= kernel_x • stride_y <= kernel_y • stride_z <= kernel_z Or: • kernel_x == kernel_y == kernel_z == 1 • stride_x == stride_y == stride_z == 2
stride_z	[1, 6]	The following must be true: • stride_x <= kernel_x • stride_y <= kernel_y • stride_z <= kernel_z Or: • kernel_x == kernel_y == kernel_z == 1 • stride_x == stride_y == stride_z == 2
pad_x_begin	[0, 6]	The following must be true: • pad_x_begin/end < kernel_x • pad_y_begin/end < kernel_y • pad_z_begin/end < kernel_z
pad_x_end	[0, 6]	The following must be true: • pad_x_begin/end < kernel_x • pad_y_begin/end < kernel_y • pad_z_begin/end < kernel_z
pad_y_begin	[0, 6]	The following must be true: • pad_x_begin/end < kernel_x • pad_y_begin/end < kernel_y • pad_z_begin/end < kernel_z
pad_y_end	[0, 6]	The following must be true: • pad_x_begin/end < kernel_x • pad_y_begin/end < kernel_y • pad_z_begin/end < kernel_z
pad_z_begin	[0, 6]	The following must be true: • pad_x_begin/end < kernel_x • pad_y_begin/end < kernel_y • pad_z_begin/end < kernel_z
pad_z_end	[0, 6]	The following must be true: • pad_x_begin/end < kernel_x • pad_y_begin/end < kernel_y • pad_z_begin/end < kernel_z
dilation_x	{1}	-
dilation_y	{1}	-
dilation_z	{1}	-

Parameter	Valid value or range	Comment
group	{1}	-
with_activation	{NONE, RELU, CLIP, LEAKYRELU}	-

Convolution2D

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384]	The input must be an NHWC format tensor.
output shape(s)	Supports int8 and int16 data types.	The output must be an NHWC format tensor.
kernel_x	[1, 64]	-
kernel_y	[1, 64]	-
stride_x	[1, 16]	-
stride_y	[1, 16]	-
pad_top	[0, 16]	-
pad_bottom	[0, 16]	-
pad_left	[0, 16]	-
pad_right	[0, 16]	-
dilation_x	[1, 40]	If dilation_x != 1, the following must be true: output_shape_size * sizeof (output_type) * stride_x * stride_y <= 1G

Parameter	Valid value or range	Comment
dilation_y	[1, 40]	If dilation_y != 1, the following must be true: output_shape_size * sizeof (output_type) * stride_x * stride_y <= 1G
group	{1}	-
with_activation	{NONE, RELU, CLIP, PRELU, LEAKYRELU}	-

Convolution3D

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	The input must be an NDHWC format tensor. Supports any shape but the following condition should be met: Dims[4] * kernel_z <= 4096
kernel_x	[1, 11]	-
kernel_y	[1, 11]	-
kernel_z	[1, 11]	The following must be true: (Input_c * kernel_z % 32 == 0 and 32<=input_c * kernel_z <=4096) Or: Input_c * kernel_z ∈ {1, 3, 4}
stride_x	[1, 6]	The following must be true: • stride_x <= kernel_x • stride_y <= kernel_y • stride_z <= kernel_z Or: • kernel_x == kernel_y == kernel_z == 1 • stride_x == stride_y == stride_z == 2
stride_y	[1, 6]	The following must be true:

Parameter	Valid value or range	Comment
		• stride_x <= kernel_x • stride_y <= kernel_y • stride_z <= kernel_z Or: • kernel_x == kernel_y == kernel_z == 1 • stride_x == stride_y == stride_z == 2
stride_z	[1, 6]	The following must be true: • stride_x <= kernel_x • stride_y <= kernel_y • stride_z <= kernel_z Or: • kernel_x == kernel_y == kernel_z == 1 • stride_x == stride_y == stride_z == 2
pad_x_begin	[0, 6]	The following must be true: • pad_x_begin/end < kernel_x • pad_y_begin/end < kernel_y • pad_z_begin/end < kernel_z
pad_x_end	[0, 6]	The following must be true: • pad_x_begin/end < kernel_x • pad_y_begin/end < kernel_y • pad_z_begin/end < kernel_z
pad_y_begin	[0, 6]	The following must be true: • pad_x_begin/end < kernel_x • pad_y_begin/end < kernel_y • pad_z_begin/end < kernel_z
pad_y_end	[0, 6]	The following must be true: • pad_x_begin/end < kernel_x • pad_y_begin/end < kernel_y • pad_z_begin/end < kernel_z
pad_z_begin	[0, 6]	The following must be true: • pad_x_begin/end < kernel_x • pad_y_begin/end < kernel_y • pad_z_begin/end < kernel_z
pad_z_end	[0, 6]	The following must be true: • pad_x_begin/end < kernel_x • pad_y_begin/end < kernel_y • pad_z_begin/end < kernel_z
dilation_x	{1}	-
dilation_y	{1}	-
dilation_z	{1}	-
group	{1}	-

Parameter	Valid value or range	Comment
with_activation	{NONE, RELU, CLIP, LEAKYRELU}	-

Cosh

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

Cosine

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.

Parameter	Valid value or range	Comment
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

Count

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports uint8 and uint16 data types. Supports 2-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384]	-
min	[0, 65534]	Any value in the range of input data type. (max > min)
max	[1, 65535]	Any value in the range of input data type. (max > min)
nbins	[1, 4096]	-
discrete	{true}	-

Crop

The main execution unit: DMA

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4, 5-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] For 5-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] Dim[4] ∈ [1, 16384]	-
crops	crops[0][0] ∈ [0, 30] crops[0][1] ∈ [1, 31] crops[1][0] ∈ [0, 98] crops[1][1] ∈ [1, 99] crops[2][0] ∈ [0, 1078] crops[2][1] ∈ [1, 1079] crops[3][0] ∈ [0, 1918] crops[3][1] ∈ [1, 1919] crops[4][0] ∈ [0, 4094] crops[4][1] ∈ [1, 4095]	crops[i][0] should be less than crops[i][1].

CropAndResize

The main execution unit: TPC or AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	3	-
input shape(s)	Input0: Supports uint8 data type. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096] Input1: Supports uint16 data type. Supports 2-dimensional input. Dim[0] ∈ [1, 16384] Dim[1] ∈ [4, 4] Input2: Supports uint8 data type. Supports 1-dimensional input. Dim[0] ∈ [1, 16384]	Number of ROIs
crop_size	crop_size[0] ∈ [0, 1080] crop_size[1] ∈ [1, 1920]	-
method	{BILINEAR, NEAREST}	-
extrapolation_value	{0, 1}	-

CumProd

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y

Quantization methods	Supported (Y or N)
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	N

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8 and uint8 data types. Supports 2, 3, 4, 5-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] For 5-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] Dim[4] ∈ [1, 16384]	shape[axis] should be in the range [1, 65535].
output shape(s)	Supports int8 and uint8 data types.	-
axis	-	The axis only can be a number. All the values are ∈ [-1, input_dims - 1].
method	{PROD}	-
exclusive	{false, true}	-
reverse	{false, true}	-

CumSum

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	N

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8 and uint8 data types. Supports 2, 3, 4, 5-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] For 5-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] Dim[4] ∈ [1, 16384]	The shape[axis] should be in the range [1, 65535].
output shape(s)	Supports int8 and uint8 data types.	-
axis	-	The axis only can be a number. All the values are ∈ [-1, input_dims - 1].
method	{SUM}	-
exclusive	{false, true}	-
reverse	{false, true}	-

DataStride

The main execution unit: DMA

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384]	-
kernel_x	[1, 16]	kernel_x must be smaller than or equal to stride_x.
kernel_y	[1, 16]	kernel_y must be smaller than or equal to stride_y.
stride_x	[1, 16]	stride_x should be smaller than or equal to input_w. Only supports kernel_x == kernel_y, stride_x == stride_y.
stride_y	[1, 16]	stride_y should be smaller than or equal to input_h. Only supports kernel_x == kernel_y, stride_x == stride_y.

DecodeBox

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	N
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	N

Parameter	Valid value or range	Comment
Number of input tensors	2	-
Number of output tensors	5	-
input shape(s)	Input0: Supports uint8 data type. Supports 3-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 5000] Dim[2] ∈ [1, 100] Input1: Supports int8 data type. Supports 3-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 5000] Dim[2] ∈ [4, 4]	-
output shape(s)	Output0: Supports int16 data type. Output1: Supports uint16 data type. Output2: Supports uint16 data type. Output3: Supports uint8 data type. Output4: Supports uint16 data type.	-

DepthToSpace

The main execution unit: DMA

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	-
block_size_x	[1, 16]	Only supports block_size_x == block_size_y.
block_size_y	[1, 16]	Only supports block_size_x == block_size_y.

DepthwiseConvolution

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384]	The input must be an NHWC format tensor.
kernel_x	[1, 64]	-
kernel_y	[1, 64]	-
stride_x	[1, 16]	-
stride_y	[1, 16]	-
pad_top	[0, 16]	-

Parameter	Valid value or range	Comment
pad_bottom	[0, 16]	-
pad_left	[0, 16]	-
pad_right	[0, 16]	-
dilation_x	[1, 16]	-
dilation_y	[1, 16]	-
group	[1, 16384]	The following must be true: Input_dim[3] == group
multiplier	[1, 4096]	The following must be true: Output_dim[3] == Input_dim[3] * multiplier
with_activation	{NONE, RELU, CLIP, PRELU, LEAKYRELU}	-

Dilation2D

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	N

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384]	The input must be an NHWC format tensor. mixed_prec is not supported.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	The output must be an NHWC format tensor. Input type -> output type should be combinations as follows: • int8->int8

Parameter	Valid value or range	Comment
		• uint8->int8 (when weights are all positive) • uint8->uint8 • int16->int16 • uint16->int16 (when weights are all positive) • uint16->uint16
kernel_x	[1, 64]	-
kernel_y	[1, 64]	-
stride_x	[1, 16]	-
stride_y	[1, 16]	-
pad_top	[0, 16]	-
pad_bottom	[0, 16]	-
pad_left	[0, 16]	-
pad_right	[0, 16]	-
dilation_x	[1, 16]	If dilation_x != 1, the following must be true: output_shape_size * sizeof (output_type) * stride_x * stride_y <= 1G
dilation_y	[1, 16]	If dilation_y != 1, the following must be true: output_shape_size * sizeof (output_type) * stride_x * stride_y <= 1G

Div

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	2	-
input shape(s)	Input0: Supports uint8 data type. Input1: Supports uint8 data type.	Supports any shape.

ElementwiseAdd

The main execution unit: TPC or AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	2	-
Number of output tensors	1	-
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Input1: Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
method	{ADD}	-
with_activation	{NONE, RELU, LEAKYRELU, PRELU, CLIP}	-

ElementwiseMax

The main execution unit: TPC or AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	2	-
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Input1: Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
method	{MAX}	-
with_activation	{NONE, RELU, LEAKYRELU, PRELU, CLIP}	-

ElementwiseMin

The main execution unit: TPC or AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	2	-
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Input1: Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
method	{MIN}	-
with_activation	{NONE, RELU, LEAKYRELU, PRELU, CLIP}	-

ElementwiseMul

The main execution unit: TPC or AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	2	-
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Input1: Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
method	{MUL}	-
with_activation	{NONE, RELU, LEAKYRELU, PRELU, CLIP}	-

ElementwiseSub

The main execution unit: TPC or AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	2	-
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Input1: Supports int8, uint8, int16, and uint16 data types.	Supports any shape.

Parameter	Valid value or range	Comment
method	{SUB}	-
with_activation	{NONE, RELU, LEAKYRELU, PRELU, CLIP}	-

Elu

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
method	{ELU}	-

EmbeddingLookupSparse

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	4	-
Number of output tensors	1	-
input shape(s)	Input0: Supports int8 and uint8 data types. Supports 2, 3, 4, 5-dimensional input. For 2-dims: Dim[0] ∈ [1, 4096] Dim[1] ∈ [1, 3968] For 3-dims: Dim[0] ∈ [1, 4096] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 3968] For 4-dims: Dim[0] ∈ [1, 4096] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 3968] For 5-dims: Dim[0] ∈ [1, 4096] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 1080] Dim[3] ∈ [1, 1920] Dim[4] ∈ [1, 4096] Input1: Supports int16 data type. Supports 2-dimensional input. Dim[0] ∈ [1, 4096] Dim[1] ∈ [2, 2] Input2: Supports int16 and int8 data types. Supports 1-dimensional input. Dim[0] ∈ [1, 4096] Input3: Supports int8 data type. Supports 1-dimensional input. Dim[0] ∈ [1, 4096]	-
output shape(s)	Supports int8 data type.	-

Parameter	Valid value or range	Comment
cominer	{MEAN, SUM, SQRTN}	-
max_norm	{NONE}	-

Erf

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

Erosion2D

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	N

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-

Parameter	Valid value or range	Comment
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384]	The input must be an NHWC format tensor. mixed_prec is not supported.
output shape(s)	Supports int8 and int16 data types.	The output must be an NHWC format tensor.
kernel_x	[1, 64]	-
kernel_y	[1, 64]	-
stride_x	[1, 16]	-
stride_y	[1, 16]	-
pad_top	[0, 16]	-
pad_bottom	[0, 16]	-
pad_left	[0, 16]	-
pad_right	[0, 16]	-
dilation_x	[1, 16]	If dilation_x != 1, the following must be true: output_shape_size * sizeof (output_type) * stride_x * stride_y <= 1G
dilation_y	[1, 16]	If dilation_y != 1, the following must be true: output_shape_size * sizeof (output_type) * stride_x * stride_y <= 1G

Exp

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-

Parameter	Valid value or range	Comment
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

Filter

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	2-10	-
Number of output tensors	2	-
input shape(s)	Input0_8: Supports int8 and int16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096] Input9:	Input9: A vector with the same length as the specified axis in input 0

Parameter	Valid value or range	Comment
	Supports int8 and int16 data types. Supports 1-dimensional input.	
output shape(s)	Output0: Supports int8 and int16 data types. Output1: Supports int8 and int16 data types.	Output0: The same shape as input 0
axis	{0}	-
num	[1, 8]	-

Floor

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

FractionalPool

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y

Quantization methods	Supported (Y or N)
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	3	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	• k = (input_shape[1](h) + outout_shape[1](h) - 1) / output_shape[1](h) • k * input_shape[2](w) * bits(input_data) <= 892 * 32 Only 16-bit data is supported when `method` is `MAX`.
output shape(s)	Output0: Supports int8, uint8, int16, and uint16 data types. Output1: Supports int16 data type. Output2: Supports int16 data type.	Output1 and output2 are randomly generated sequences.
method	{MAX, AVG}	-
overlap	{true, false}	-
pseudo	{true, false}	-
seed	{int32}	-

FullyConnected

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	The input must be an NC format tensor. The first dim has not limit and the maximum value of the second dim size is 65535 for 8-bit data and 32768 for 16-bit data.
output shape(s)	Supports int8, int16, uint8, and uint16 data types.	The output must be an NC format tensor.

GRUv1

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	N

Parameter	Valid value or range	Comment
Number of input tensors	2	-
Number of output tensors	2	-
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Supports 3-dimensional input. Dim[0] ∈ [1, 16384] Dim[1] ∈ [1, 4096] Dim[2] ∈ [1, 16384] Input1: Supports int8, uint8, int16, and uint16 data types. Supports 2-dimensional input. Dim[0] ∈ [1, 16384] Dim[1] ∈ [1, 8192]	-
output shape(s)	Output0: Supports int8, uint8, int16, and uint16	DType of output is equal to dtype of input1.

Parameter	Valid value or range	Comment
	data types. Output1: Supports int8, uint8, int16, and uint16 data types.	
out_sequence	{{H}, {Hn}, {H, Hn}}	-
activations	{Relu, Tanh, Sigmoid, Affine, LeakyRelu, ThresholdedRelu, HardSighmoid, Elu, Softsign, Softplus}	-
direction	{forward}	-

GRUv3

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	N

Parameter	Valid value or range	Comment
Number of input tensors	2	-
Number of output tensors	2	-
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Supports 3-dimensional input. Dim[0] ∈ [1, 16384] Dim[1] ∈ [1, 4096] Dim[2] ∈ [1, 16384] Input1: Supports int8, uint8, int16, and uint16 data types. Supports 2-dimensional input. Dim[0] ∈ [1, 16384] Dim[1] ∈ [1, 8192]	-

Parameter	Valid value or range	Comment
output shape(s)	Output0: Supports int8, uint8, int16, and uint16 data types. Output1: Supports int8, uint8, int16, and uint16 data types.	DType of output is equal to dtype of input1.
out_sequence	{{H}, {Hn}, {H, Hn}}	-
activations	{Relu, Tanh, Sigmoid, Affine, LeakyRelu, ThresholdedRelu, HardSighmoid, Elu, Softsign, Softplus}	-
direction	{forward}	-

Gather

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4, 5-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32]	-

Parameter	Valid value or range	Comment
	Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] For 5-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 1080] Dim[3] ∈ [1, 1920] Dim[4] ∈ [1, 4096]	
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
axis	-	axis ∈ [0, input0_dims-1]
Batch_dims	-	Batch_dims ∈ [0, axis]

GatherElements

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	2	-
Number of output tensors	1	-
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Input1: Supports uint16 data type.	Supports up to 5-dimensional input, with no size limit for every dimension.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

Parameter	Valid value or range	Comment
axis	-	axis ∈ [0, input0_dims-1]

GatherND

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	2	-
Number of output tensors	1	-
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4, 5-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] Dim[2] ∈ [1, 4096] Dim[3] ∈ [1, 4096] For 5-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 1080] Dim[3] ∈ [1, 1920]	The maximum value of every dim has no limit but the following conditions should be met: • Batch_dims <= min(rank(input0), rank(input1)) • Input1.shape[-1] <= rank(input0) - batch_dims • Input0.shape[:batch_dims] = input1.shape[:batch_dims]

Parameter	Valid value or range	Comment
	Dim[4] ∈ [1, 4096] Input1: Supports uint16 data type. Supports 2, 3, 4, 5-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096] For 5-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 1080] Dim[3] ∈ [1, 1920] Dim[4] ∈ [1, 4096]	
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

Gelu

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
method	{GELU}	-

Gemm

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	3	-
Number of output tensors	1	-
input shape(s)	Input0: Supports int8 and uint8 data types. Supports 2-dimensional input. Dim[0] ∈ [1, 1920] Dim[1] ∈ [1, 1920] Input1: Supports int8 and uint8 data types. Supports 2-dimensional input. Dim[0] ∈ [1, 1920] Dim[1] ∈ [1, 1920] Input2: Supports int8 and uint8 data types. Supports 2-dimensional input. Dim[0] ∈ [1, 1920] Dim[1] ∈ [1, 1920]	The matrix size must satisfy the matrix multiplication rule.

Parameter	Valid value or range	Comment
output shape(s)	Supports int8 and uint8 data types.	The matrix size must satisfy the matrix multiplication rule.
trans_a	{True, False}	-
trans_b	{True, False}	-
alpha	[1, 19]	-
beta	[1, 19]	-

GetValidCount

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	3	-
input shape(s)	Supports int16 data type. Supports 2, 3, 4, 5-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [5, 6] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [5, 6] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [5, 6] For 5-dims: Dim[0] ∈ [1, 32]	-

Parameter	Valid value or range	Comment
	Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] Dim[4] ∈ [5, 6]	
output shape(s)	Output0: Supports int16 data type. Output1: Supports int16 data type. Output2: Supports int16 data type.	-
score_index	{0, 1, 2, 3, 4, 5}	-
id_index	{0, 1, 2, 3, 4, 5}	-

GridSample

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	2	-
Number of output tensors	1	-
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384]	-

Parameter	Valid value or range	Comment
	Input1: Supports int16 data type. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [2, 2]	
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
method	{NEAREST, BILINEAR}	-
padding_mode	{ZEROS, BORDER}	-
align_corners	{True, False}	-

GroupConvolution

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8 and int16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384]	The input must be an NHWC format tensor.
output shape(s)	Supports int8 and int16 data types.	The output must be an NHWC format tensor.
kernel_x	[1, 64]	-
kernel_y	[1, 64]	-

Parameter	Valid value or range	Comment
stride_x	[1, 16]	-
stride_y	[1, 16]	-
pad_top	[0, 16]	-
pad_bottom	[0, 16]	-
pad_left	[0, 16]	-
pad_right	[0, 16]	-
dilation_x	[1, 16]	If dilation_x != 1, the following must be true: output_shape_size * sizeof (output_type) * stride_x * stride_y <= 1G
dilation_y	[1, 16]	If dilation_y != 1, the following must be true: output_shape_size * sizeof (output_type) * stride_x * stride_y <= 1G
group	{1, 32}	The following must be true: • Output_dim[3] % group == 0 • Input_dim[3] % group == 0 • group > 1

GroupNormalization

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8 and uint8 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384]	-
axis	{3}	-

Parameter	Valid value or range	Comment
group	[1, 16384]	The following conditions must be true: • input.shape[axis] % group == 0 • input.shape[axis] % 4 == 0

HardSigmoid

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
method	{HARDSIGMOID}	-

HardSwish

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-

Parameter	Valid value or range	Comment
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
method	{HARDSWISH}	-

InTopK

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	2	-
Number of output tensors	1	-
input shape(s)	Input0: Supports int8 and uint8 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] Input1:	-

Parameter	Valid value or range	Comment
	Supports uint8 data type. Supports 2-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384]	
output shape(s)	Supports int8 and uint8 data types.	-
k	[1, 16384]	-
axis	{-1}	-

InstanceNormalization

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384]	The input must be an NHWC format tensor.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	The output must be an NHWC format tensor.

L1Normalization

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4, 5-dimensional input. For 2-dims: Dim[0] ∈ [1, 16] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 16] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 16] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] For 5-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] Dim[4] ∈ [1, 16384]	Supports 2-5 dimensional input.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
axis	-	[0, input_dims - 1] or -1
method	{L1}	-

L1Pooling2D

The main execution unit: TPC or AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
kernel_x	[1, 65]	-
kernel_y	[1, 65]	-
stride_x	[1, 6]	The following must be true: • stride_x <= kernel_x • stride_y <= kernel_y Or: • kernel_x == kernel_y == 1 • stride_x == stride_y == 2
stride_y	[1, 6]	The following must be true: • stride_x <= kernel_x • stride_y <= kernel_y Or: • kernel_x == kernel_y == 1 • stride_x == stride_y == 2
pad_top	[0, 6]	The following must be true: • pad_top/bottom < kernel_y • pad_left/right < kernel_x

Parameter	Valid value or range	Comment
pad_bottom	[0, 6]	The following must be true: • pad_top/bottom < kernel_y • pad_left/right < kernel_x
pad_left	[0, 6]	The following must be true: • pad_top/bottom < kernel_y • pad_left/right < kernel_x
pad_right	[0, 6]	The following must be true: • pad_top/bottom < kernel_y • pad_left/right < kernel_x
method	{L1}	-

L2Normalization

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4, 5-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384]	-

Parameter	Valid value or range	Comment
	Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] For 5-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] Dim[4] ∈ [1, 16384]	
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
method	-	"L2"
axis	-	[0, len(input0_shape)-1] or -1

L2Pooling2D

The main execution unit: TPC or AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

Parameter	Valid value or range	Comment
kernel_x	[1, 65]	-
kernel_y	[1, 65]	-
stride_x	[1, 6]	The following must be true: • stride_x <= kernel_x • stride_y <= kernel_y Or: • kernel_x == kernel_y == 1 • stride_x == stride_y == 2
stride_y	[1, 6]	The following must be true: • stride_x <= kernel_x • stride_y <= kernel_y Or: • kernel_x == kernel_y == 1 • stride_x == stride_y == 2
pad_top	[0, 6]	The following must be true: • pad_top/bottom < kernel_y • pad_left/right < kernel_x
pad_bottom	[0, 6]	The following must be true: • pad_top/bottom < kernel_y • pad_left/right < kernel_x
pad_left	[0, 6]	The following must be true: • pad_top/bottom < kernel_y • pad_left/right < kernel_x
pad_right	[0, 6]	The following must be true: • pad_top/bottom < kernel_y • pad_left/right < kernel_x
method	{L2}	-

LRN

The main execution unit: TPC or AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
method	{ACROSS_CHANNELS, WITHIN_CHANNEL}	-
size	[1, 64]	-
bias	[1, 64]	-
alpha	[1, 64]	-
beta	[0.0, 1.0]	Left-open right-open interval

LayerNormalization

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	N

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	• Supports 2–6 dimensional input. • If it is not per-channel quant, the size of every dim has no limit, otherwise the last dim cannot be greater than 1024.

Parameter	Valid value or range	Comment
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	The output must be an NHWC format tensor.
axis	{1, 2, 3}	-

LeakyRelu

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
method	{LEAKYRELU}	-

LeftShift

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	2	-

Parameter	Valid value or range	Comment
Number of output tensors	1	-
input shape(s)	Input0: Supports int8, uint8, int16, uint16, int32, and uint32 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096] Input1: Supports uint8 data type. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	Broadcast is not supported, and both inputs should share the same shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
direction	{LEFT}	-

Log

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

LogSoftmax

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8 and uint8 data types.	Supports any shape.
output shape(s)	Supports int8 data type.	-
axis	-	axis ∈ [-1, input_dims - 1]

Logical

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1-2	-
Number of output tensors	1	-
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Input1: Supports int8, uint8, int16, and uint16 data types.	Supports any shape. The shape of the second input is equal to the shape of the first input. Input1 may not exist.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
method	{EQUAL, NOT_EQUAL, GREATER, GREATER_EQUAL, LESS, LESS_EQUAL, OR, AND, XOR, NOT}	If method=='NOT', len(bottoms) ==1 & len(scale_value == 1).

MatMul

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	2	-
Number of output tensors	1	-
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Input1: Supports int8, uint8, int16, and uint16 data types.	Supports 3-dimensional input and the size of the first dim has no limit.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	The range of Dim[0] * Dim[1] ∈ [1,16384], and must be equal to Dim[0] * Dim[1] in the input shape. The matrix size must satisfy the matrix multiplication rule.
trans_a	{True, False}	-
trans_b	{True, False}	-

MaxPooling2D

The main execution unit: TPC or AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 16384]	-

Parameter	Valid value or range	Comment
output shape(s)	Supports int8 and int16 data types.	-
kernel_x	[1, 65]	-
kernel_y	[1, 65]	-
stride_x	[1, 16]	The following must be true: • stride_x <= kernel_x • stride_y <= kernel_y Or: • kernel_x == kernel_y == 1 • stride_x == stride_y == 2
stride_y	[1, 16]	The following must be true: • stride_x <= kernel_x • stride_y <= kernel_y Or: • kernel_x == kernel_y == 1 • stride_x == stride_y == 2
pad_top	[0, 16]	The following must be true: • pad_top/bottom < kernel_y • pad_left/right < kernel_x
pad_bottom	[0, 16]	The following must be true: • pad_top/bottom < kernel_y • pad_left/right < kernel_x
pad_left	[0, 16]	The following must be true: • pad_top/bottom < kernel_y • pad_left/right < kernel_x
pad_right	[0, 16]	The following must be true: • pad_top/bottom < kernel_y • pad_left/right < kernel_x
method	{MAX}	-

MaxPooling3D

The main execution unit: TPC or AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8 and uint8 data types. Supports 5-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 100] Dim[2] ∈ [1, 1080] Dim[3] ∈ [1, 1920] Dim[4] ∈ [1, 4096]	-
output shape(s)	Supports int8 and uint8 data types.	-
kernel_x	[1, 1080]	-
kernel_y	[1, 1920]	-
kernel_z	[1, 100]	-
stride_x	[1, 1080]	-
stride_y	[1, 1920]	-
pad_x_begin	[0, 16]	The following must be true: • pad_x_begin /end < kernel_x • pad_y_begin /end < kernel_y • pad_z_begin /end < kernel_z
pad_x_end	[0, 16]	The following must be true: • pad_x_begin /end < kernel_x • pad_y_begin /end < kernel_y • pad_z_begin /end < kernel_z
pad_y_begin	[0, 16]	The following must be true: • pad_x_begin /end < kernel_x • pad_y_begin /end < kernel_y • pad_z_begin /end < kernel_z
pad_y_end	[0, 16]	The following must be true: • pad_x_begin /end < kernel_x • pad_y_begin /end < kernel_y • pad_z_begin /end < kernel_z
pad_z_begin	[0, 16]	The following must be true: • pad_x_begin /end < kernel_x • pad_y_begin /end < kernel_y • pad_z_begin /end < kernel_z
pad_z_end	[0, 16]	The following must be true: • pad_x_begin /end < kernel_x • pad_y_begin /end < kernel_y • pad_z_begin /end < kernel_z

Parameter	Valid value or range	Comment
count_include_pad	{False}	-
ceil_mod	{False}	-
method	{MAX}	-

MaxPoolingWithArgMax

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	2	-
input shape(s)	Supports int8 data type. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384]	-
output shape(s)	Output0: Supports int8 data type. Output1: Supports int32 data type.	-
kernel_x	[1, 17]	-
kernel_y	[1, 17]	-
stride_x	[1, 17]	The following must be true: • stride_x <= kernel_x • stride_y <= kernel_y Or: • kernel_x == kernel_y == 1 • stride_x == stride_y == 2
stride_y	[1, 17]	The following must be true:

Parameter	Valid value or range	Comment
		• stride_x <= kernel_x • stride_y <= kernel_y Or: • kernel_x == kernel_y == 1 • stride_x == stride_y == 2
pad_top	[0, 6]	The following must be true: • pad_top/bottom < kernel_y • pad_left/right < kernel_x
pad_bottom	[0, 6]	The following must be true: • pad_top/bottom < kernel_y • pad_left/right < kernel_x
pad_left	[0, 6]	The following must be true: • pad_top/bottom < kernel_y • pad_left/right < kernel_x
pad_right	[0, 6]	The following must be true: • pad_top/bottom < kernel_y • pad_left/right < kernel_x
dilation_x	{1}	-
dilation_y	{1}	-
storage_order	{0, 1}	-

MaxRoiPool

The main execution unit: TPC or AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	2	-
Number of output tensors	1	-
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input.	-

Parameter	Valid value or range	Comment
	Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096] Input1: Supports uint16 data type. Supports 2-dimensional input. Dim[0] ∈ [1, 16384] Dim[1] ∈ [5, 5]	
output shape(s)	Supports int8 data type.	-
pooled_shape	pooled_shape[0] ∈ [1, 1080] pooled_shape[1] ∈ [1, 1920]	-
spatial	spatial[0] ∈ [1, 65535] spatial[1] ∈ [1, 65535]	-

MaxUnpool

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	N
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	2	-
input shape(s)	Input0: Supports int8 and uint8 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 16] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384]	-

Parameter	Valid value or range	Comment
	Input1: Supports int8 and uint8 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 16] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384]	
flattem_dim	{HW, HWC, NHWC, NCHW}	-
storage_order	{0, 1}	-
output_shape	output_shape[0] ∈ [1, 16] output_shape[1] ∈ [1, 16384] output_shape[2] ∈ [1, 16384] output_shape[3] ∈ [1, 16384]	-

MeanVarianceNormalization

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports up to 6-dimensional input, with no size limit for every dimension.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
axis	{{1, 2, 3}, {1, 2}}	-
epsilon	[0.0, 64.0]	Optional, left-open right-closed interval

Meshgrid

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1-2	-
Number of output tensors	1	-
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Supports 1-dimensional input. Dim[0] ∈ [1, 16384] Input1: Supports int8, uint8, int16, and uint16 data types. Supports 1-dimensional input. Dim[0] ∈ [1, 16384]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
indexing	{xy, ij}	-
sparse	{false, true}	-
copy	{false, true}	-

Mish

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y

Quantization methods	Supported (Y or N)
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
method	{MISH}	-

Mod

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8 and uint8 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32]	-

Parameter	Valid value or range	Comment
	Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384]	
output shape(s)	Supports int8 and uint8 data types.	-
fmod	{0, 1}	Optional

Moments

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4, 5-dimensional input. For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] For 5-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 1080] Dim[3] ∈ [1, 1920] Dim[4] ∈ [1, 4096]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

Parameter	Valid value or range	Comment
axis	{{0}, {1}, {2}, {3}, {0, 1}, {0, 2}, {0, 3}, {1, 1}, {1, 2}, {1, 3}, {0, 1, 2}, {0, 1, 3}, {0, 2, 3}, {1, 2, 3}, {0, 1, 2, 3}}	-
keepdims	{True, False}	Optional

Mul

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	2	-
Number of output tensors	1	-
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096] Input1: Supports int8, uint8, int16, and uint16	Input format can be: [[N,C],[C]], [[N,C],[1]], [[N,C],[N,1]], [[N,H,C],[C]], [[N,H,C],[1]], [[N,H,C],[H,1]], [[N,H,C],[N,1,1]], [[N,H,W,C],[C]], [[N,H,W,C],[1]], [[N,H,W,C],[H,1,1]], [[N,H,W,C],[N,1,1,1]], and the order of the two inputs can be swapped.

Parameter	Valid value or range	Comment
	data types. Supports 2-dimensional input.	
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

MultiboxTransformLoc

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	N

Parameter	Valid value or range	Comment
Number of input tensors	3	-
Number of output tensors	2	-
input shape(s)	Input0: Supports uint8 data type. Supports 3-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 100] Dim[2] ∈ [1, 5000] Input1: Supports int8 data type. Supports 2-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 20000] Input2: Supports int16 data type. Supports 3-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 5000] Dim[2] ∈ [4, 4]	-

Parameter	Valid value or range	Comment
output shape(s)	Output0: Supports int16 data type. Output1: Supports uint16 data type.	-

NMS

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	4	-
Number of output tensors	4	-
input shape(s)	Input0: Supports int16 data type. Supports 3-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [4, 4] Input1: Supports uint16 data type. Supports 2-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Input2: Supports uint16 data type. Supports 2-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1] Input3:	-

Parameter	Valid value or range	Comment
	Supports uint8 data type. Supports 3-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [4, 4]	
output shape(s)	Output0: Supports int16 data type. Output1: Supports uint16 data type. Output2: Supports uint8 data type. Output3: Supports uint16 data type.	-
iou_threshold	{0, 16384}	-
center_point_box	{0, 1}	Optional

Negative

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
output shape(s)	Supports int8 and int16 data types.	-

NormalizedMoments

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	N

Parameter	Valid value or range	Comment
Number of input tensors	3	-
Number of output tensors	2	-
input shape(s)	Input0: Supports int8 and uint8 data types. Input1: Supports int8 and uint8 data types. Input2: Supports int8 and uint8 data types.	Supports up to 5-dimensional input. Input0 and input2 should support the same types.
output shape(s)	Output0: Supports int8 and uint8 data types. Output2: Supports int8 and uint8 data types.	Only supported for Zhouyi Z2/X1.

OneHot

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 1, 2, 3, 4-dimensional input. For 1-dims: Dim[0] ∈ [1, 16384] For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384]	-
axis	{-1, 3}	-
depth	[1, 16384]	-
on_value	[0, 65535]	-
off_value	[0, 65535]	-

PRelu

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-

Parameter	Valid value or range	Comment
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
method	{PRELU}	-
negative_slope_type	{uint16}	-
negative_slope_shape	-	Equal to the value of input0_shape[-1]

Pad

The main execution unit: DMA

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4, 5-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] For 5-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384]	-

Parameter	Valid value or range	Comment
	Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] Dim[4] ∈ [1, 16384]	
pads	pads[0][0] ∈ [0, 128] pads[0][1] ∈ [0, 128] pads[1][0] ∈ [0, 128] pads[1][1] ∈ [0, 128] pads[2][0] ∈ [0, 128] pads[2][1] ∈ [0, 128] pads[3][0] ∈ [0, 128] pads[3][1] ∈ [0, 128] pads[4][0] ∈ [0, 128] pads[4][1] ∈ [0, 128]	-
mode	{CONSTANT, REFLECT, SYMMETRIC}	-

Pow

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	2	-
input shape(s)	Input0: Supports int8 and uint8 data types. Input1: Supports int8 and uint8 data types.	Supports any shape.
exponent	[1, 9]	-

Reciprocal

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

ReduceAll

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8 and uint8 data types.	Supports up to 6-dimensional input, with no size limit for every dimension.
output shape(s)	Supports uint8 data type.	-

Parameter	Valid value or range	Comment
axis	-	The axis can be a number or a set of number. All the values are ∈ [-1, input_dims - 1].
method	{ALL}	-

ReduceAny

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8 and uint8 data types.	Supports up to 6-dimensional input, with no size limit for every dimension.
output shape(s)	Supports uint8 data type.	-
axis	-	The axis can be a number or a set of number. All the values are ∈ [-1, input_dims - 1].
method	{ANY}	-

ReduceL1

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8 and uint8 data types.	Supports up to 6-dimensional input, with no size limit for every dimension.
output shape(s)	Supports uint8 data type.	-
axis	-	The axis can be a number or a set of number. All the values are ∈ [-1, input_dims - 1].
method	{L1}	-

ReduceL2

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8 and uint8 data types.	Supports up to 6-dimensional input, with no size limit for every dimension.
output shape(s)	Supports uint8 data type.	-
axis	-	The axis can be a number or a set of number. All the values are ∈ [-1, input_dims - 1].
method	{L2}	-

ReduceMax

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y

Quantization methods	Supported (Y or N)
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8 and uint8 data types.	Supports up to 6-dimensional input, with no size limit for every dimension.
output shape(s)	Supports int8 and uint8 data types.	-
axis	-	The axis can be a number or a set of number. All the values are ∈ [-1, input_dims - 1].
method	{MAX}	-

ReduceMean

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8 an uint8 data types.	Supports up to 6-dimensional input, with no size limit for every dimension.
output shape(s)	Supports int8 and uint8 data types.	-
axis	-	The axis can be a number or a set of number. All the values are ∈ [-1, input_dims - 1].
method	{MEAN}	-

ReduceMin

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8 and uint8 data types.	Supports up to 6-dimensional input, with no size limit for every dimension.
output shape(s)	Supports int8 and uint8 data types.	-
axis	-	The axis can be a number or a set of number. All the values are $\in$ [-1, input_dims - 1].
method	{MIN}	-

ReduceProd

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8 and uint8 data types.	Supports up to 6-dimensional input, with no size limit for every dimension.

Parameter	Valid value or range	Comment
output shape(s)	Supports int8 and uint8 data types.	-
axis	-	The axis can be a number or a set of number. All the values are ∈ [-1, input_dims - 1].
method	{PROD}	-

ReduceSum

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8 and uint8 data types.	Supports up to 6-dimensional input, with no size limit for every dimension.
output shape(s)	Supports int8 and uint8 data types.	-
axis	-	The axis can be a number or a set of number. All the values are ∈ [-1, input_dims - 1].
method	{SUM}	-

ReduceUnbiasedVariance

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8 and uint8 data types.	Supports up to 6-dimensional input, with no size limit for every dimension.
output shape(s)	Supports uint8 data type.	-
axis	-	The axis can be a number or a set of number. All the values are ∈ [-1, input_dims - 1].
method	{UNBIASED_VARIANCE}	-

ReduceVariance

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8 and uint8 data types.	Supports any shape.
output shape(s)	Supports uint8 data type.	-
axis	-	The axis can be a number or a set of number. All the values are ∈ [-1, input_dims - 1].
method	{VARIANCE}	-

Region

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	N
16-bit feature map symmetric per-tensor quantization	Y

Quantization methods	Supported (Y or N)
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	N

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	5	-
input shape(s)	Supports int8 data type. Supports 5-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 13] Dim[2] ∈ [1, 13] Dim[3] ∈ [1, 6] Dim[4] ∈ [6, 105]	-
output shape(s)	Output0: Supports uint8 data type. Output1: Supports int16 data type. Output2: Supports int16 data type. Output3: Supports int16 data type. Output4: Supports int16 data type.	-

RegionFuse

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	N
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	N

Parameter	Valid value or range	Comment
Number of input tensors	10	-
Number of output tensors	5	-
input shape(s)	Input0: Supports uint8 data type. Supports 2-dimensional input. Dim[0] ∈ [1, 16] Dim[1] ∈ [1, 5000] Input1: Supports uint8 data type. Supports 2-dimensional input. Dim[0] ∈ [1, 16] Dim[1] ∈ [1, 5000] Input2: Supports int16 data type. Supports 3-dimensional input. Dim[0] ∈ [1, 16] Dim[1] ∈ [1, 5000] Dim[2] ∈ [4, 4] Input3: Supports int16 data type. Supports 3-dimensional input. Dim[0] ∈ [1, 16] Dim[1] ∈ [1, 5000] Dim[2] ∈ [4, 4] Input4: Supports int16 data type. Supports 2-dimensional input. Dim[0] ∈ [1, 16] Dim[1] ∈ [1, 100] Input5: Supports int16 data type. Supports 2-dimensional input. Dim[0] ∈ [1, 16] Dim[1] ∈ [1, 100] Input6: Supports int16 data type.	-

Parameter	Valid value or range	Comment
	Supports 2-dimensional input. Dim[0] ∈ [1, 16] Dim[1] ∈ [1, 100] Input7: Supports int16 data type. Supports 2-dimensional input. Dim[0] ∈ [1, 16] Dim[1] ∈ [1, 100] Input8: Supports int16 data type. Supports 2-dimensional input. Dim[0] ∈ [1, 16] Dim[1] ∈ [1, 1] Input9: Supports int16 data type. Supports 2-dimensional input. Dim[0] ∈ [1, 16] Dim[1] ∈ [1, 1]	
output shape(s)	Output0: Supports uint8 data type. Output1: Supports int16 data type. Output2: Supports int16 data type. Output3: Supports int16 data type. Output4: Supports int16 data type.	-

Relu

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y

Quantization methods	Supported (Y or N)
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4, 5-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] For 5-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] Dim[4] ∈ [1, 16384]	-
method	{RELU}	-

Relu6

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y

Quantization methods	Supported (Y or N)
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4, 5-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] For 5-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] Dim[4] ∈ [1, 16384]	-
method	{CRELU6}	-

Repeat

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N

Quantization methods	Supported (Y or N)
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	2	-
input shape(s)	Input0: Supports int8 and uint8 data types. Supports 2, 3, 4, 5-dimensional input. For 2-dims: Dim[0] ∈ [1, 16] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 16] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 16] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] For 5-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 1080] Dim[3] ∈ [1, 1920] Dim[4] ∈ [1, 4096] Input1: Supports uint16 data type. Supports 1-dimensional input.	[input1_shape = input0_dims[axis]] if axis else [data_size of input0]
axis	-	axis ∈ [-1, input0_dims - 1] or None

Reshape

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N

Quantization methods	Supported (Y or N)
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384]	-

Resize

The main execution unit: TPC or AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any size of 4d input but the following condition should be met:

Parameter	Valid value or range	Comment
		"inw * 2 * sizeof(dtype) < min(13.84, (32768 *inh * inw / (out * outw)))"
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
ratio_y	[1, 65]	No limit, but the following condition should be met: "inw * 2 * sizeof(dtype) < min(13.84, (32768 *inh * inw / (out * outw)))"
ratio_x	[1, 65]	No limit, but the following condition should be met: "inw * 2 * sizeof(dtype) < min(13.84, (32768 *inh * inw / (out * outw)))"
method	{NEAREST, BILINEAR}	-
mode	{ALIGN_CORNERS, HALF_PIXEL, ASYMMETRIC, PYTORCH_HALF_PIXEL, TF_HALF_PIXEL_FOR_NN}	-
nearest_mode	{FLOOR, CEIL, ROUND_PREFER_CEIL, SIMPLE}	-
interp_shift_value	[0, 13]	This parameter is required for TPC, but not for AIFF.

ReverseSequence

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1-2	-
Number of output tensors	1	-
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Supports 2, 3-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384]	The input1 cannot be provided.

Parameter	Valid value or range	Comment
	For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Input1: Supports uint16 data type. Supports 1-dimensional input. Dim[0] ∈ [1, 1]	
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
batch_axis	{0, 1}	-
time_axis	{0, 1, 2}	The value 2 only can be used for 3-dims input0, and time_axis cannot be equal to batch_axis.

RgbToYuv

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384]	-

Parameter	Valid value or range	Comment
	For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384]	
format	{I420}	-
bits	{8}	-
conversion	{BT709}	-
coefficient	{[0, 0, 0, 0, 128, 128, 218, 732, 74, -118, -395, 512, 512, -465, -47]}	-
coefficient_dtype	{int16}	-
coefficient_shift	{10}	-

RightShift

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	2	-
Number of output tensors	1	-
input shape(s)	Input0: Supports int8, uint8, int16, uint16, int32, and uint32 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920]	Broadcast is not supported, and both inputs should share the same shape.

Parameter	Valid value or range	Comment
	Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096] Input1: Supports uint8 data type. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096]	
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
direction	{RIGHT}	-

RoiAlign

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	2	-
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] Input1: Supports uint16 data type. Supports 2-dimensional input. Dim[0] ∈ [1, 16384] Dim[1] ∈ [5, 5]	-
method	{AVG, MAX}	-
sample	{[0, 8], [0, 8]}	-
coordinate_transformation_mode	{OUTPUT_HALF_PIXEL, HARF_PIXEL}	-

Round

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

Rsqrt

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

ScatterElements

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	3	-
Number of output tensors	1	-
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4, 5-dimensional input. For 2-dims:	• input1_dim = input0_dim • input1_shape <= input0_shape • input2_shape = input1_shape • When reduction = None, the shape can be unlimited.

Parameter	Valid value or range	Comment
	Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] For 5-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] Dim[4] ∈ [1, 16384] Input1: Supports int8, uint8, int16, and uint16 data types. Supports 2-dimensional input. Input2: Supports int8, uint8, int16, and uint16 data types. Supports 2-dimensional input.	• When reduction = Add, the shape should be less than 16384. • When reduction = Mul, the shape should be less than 4096.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

ScatterND

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	3	-
Number of output tensors	1	-
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4, 5-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] For 5-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] Dim[4] ∈ [1, 16384] Input1: Supports int8, uint8, int16, and uint16 data types. Supports 2-dimensional input. Input2: Supports int8, uint8, int16, and uint16 data types. Supports 2-dimensional input.	input1_shape[-1] <= dim(input0) input2_shape=input1_shape[:-1] + input0_shape[input1_shape[-1]:]
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

SegmentSumReduce

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	2	-
Number of output tensors	1	-
input shape(s)	Input0: Supports int8 and uint8 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] Input1: Supports uint16 data type. Supports 1-dimensional input. Dim[0] ∈ [1, 32]	The input1 shape must be equal to the input0_shape[0].
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
method	{SUM}	-

Selu

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
method	{SELU}	-

Shrink

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

Parameter	Valid value or range	Comment
method	{SHRINK}	-

Sigmoid

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
method	{SIGMOID}	-

Sign

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-

Parameter	Valid value or range	Comment
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

Silu

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
method	{SILU}	-

Sine

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

Sinh

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

Slice

The main execution unit: DMA

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4, 5-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] For 5-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] Dim[4] ∈ [1, 16384]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

Parameter	Valid value or range	Comment
begin	-	The value of `begin` must be less than the total shape of input and must be greater than 0.
end	-	The value of `end` must be less than the total shape of input and must be greater than 0.

Softmax

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports up to 6-dimensional input, with no size limit for every dimension.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
axis	-	[0,len(input0_shape)-1] or -1

Softplus

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
method	{SOFTPLUS}	-

Softsign

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
method	{SOFTSIGN}	-

SpaceToBatch

The main execution unit: DMA

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N

Quantization methods	Supported (Y or N)
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
block_size_x	[1, 16]	-
block_size_y	[1, 16]	-
pad_top	[0, 16]	-
pad_bottom	[0, 16]	-
pad_left	[0, 16]	-
pad_right	[0, 16]	-

SpaceToDepth

The main execution unit: DMA

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-

Parameter	Valid value or range	Comment
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384]	-
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
block_size_x	[1, 16]	-
block_size_y	[1, 16]	-

Split

The main execution unit: DMA

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	2-100	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
output shape(s)	Output0_99: Supports int8, uint8, int16, and uint16 data types.	If the split is average to n part, the output number has no limit, otherwise the maximum output number is 48.
splits	[1, 100]	-
axis	[0, 3]	-

Sqrt

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

Square

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

SquaredDifference

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	2	-
Number of output tensors	1	-
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4, 5-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] For 5-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] Dim[4] ∈ [1, 16384] Input1: Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4, 5-dimensional input. For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384]	-

Parameter	Valid value or range	Comment
	For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] For 5-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] Dim[4] ∈ [1, 16384]	
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

Sub

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	2	-
Number of output tensors	1	-
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4-dimensional input. For 2-dims:	Input format can be: [[N,C],[C]], [[N,C],[1]], [[N,C],[N,1]], [[N,H,C],[C]], [[N,H,C],[1]], [[N,H,C],[H,1]], [[N,H,C],[N,1,1]], [[N,H,W,C],[C]], [[N,H,W,C],[1]], [[N,H,W,C],[H,1,1]], [[N,H,W,C],[N,1,1,1]], and the order of the two inputs can be swapped.

Parameter	Valid value or range	Comment
	Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096] Input1: Supports int8, uint8, int16, and uint16 data types. Supports 2-dimensional input.	
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

SufficientStatistics

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	2	-
Number of output tensors	2	-
input shape(s)	Input0: Supports int8, uint8, int16, and uint16 data types. Input1:	Supports up to 5-dimensional input. Input0 and input1 should support the same types.

Parameter	Valid value or range	Comment
	Supports int8, uint8, int16, and uint16 data types.	
output shape(s)	Output0: Supports int8 and int16 data types. Output2: Supports uint8 and uint16 data types.	• When the input type is int/uint8, output0 must be int8, and output1 must be uint8. • When the input type is int/uint16, output0 must be int16, and output1 must be uint16.
axis	-	The axis can be a number or a set of number. All the values are ∈ [-1, input_dims - 1]
scale	[0, 16384]	scale number must less tran int15 range

Swish

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
method	{SWISH}	-

Tan

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y

Quantization methods	Supported (Y or N)
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

Tanh

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
method	{TANH}	-

ThresholdedRelu

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
method	{THRESHOLDEDRELU}	-

Tile

The main execution unit: DMA

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any input shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
repeats	-	The shape of `repeats` is less than or equal to the output shape.

TopK

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1-2	-
Number of output tensors	2	-
input shape(s)	Input0: Supports int8 and uint8 data types. Supports 1, 2, 3, 4, 5-dimensional input. For 1-dims: Dim[0] ∈ [1, 16384] For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] For 3-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] For 5-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] Dim[4] ∈ [1, 16384]	Input1 is optional.

Parameter	Valid value or range	Comment
	Input1: Supports int8 and uint8 data types. Supports 1-dimensional input.	
output shape(s)	Output0: Supports int8 and uint8 data types. Output1: Supports int8 and uint8 data types.	For dim in [0, shape_dims-1] and dim != axis, out_shape[dim] == input0_shape[dim]
k	[1, 16384]	-
axis	-	[0, len(input0_shape)-1], or -1
largest	{true, false}	-
sorted	{true, false}	-
select_index	{last}	-

Transpose

The main execution unit: DMA

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 2, 3, 4, 5, 6-dimensional input. For 2-dims: Dim[0] ∈ [1, 16] Dim[1] ∈ [1, 4096] For 3-dims: Dim[0] ∈ [1, 16] Dim[1] ∈ [1, 16384]	-

Parameter	Valid value or range	Comment
	Dim[2] ∈ [1, 16384] For 4-dims: Dim[0] ∈ [1, 16] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] For 5-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] Dim[4] ∈ [1, 16384] For 6-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] Dim[4] ∈ [1, 16384] Dim[5] ∈ [1, 16384]	
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-
perm	-	perm ∈ [0, input_dims - 1]

Trunc

The main execution unit: AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	Y
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-

Parameter	Valid value or range	Comment
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

UpsampleByIndex

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	2	-
Number of output tensors	1	-
input shape(s)	Input0: Supports int8 and uint8 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 16] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384] Input1: Supports int8 and uint8 data types. Supports 4-dimensional input. Dim[0] ∈ [1, 16] Dim[1] ∈ [1, 16384] Dim[2] ∈ [1, 16384] Dim[3] ∈ [1, 16384]	-
output shape(s)	Supports int8 and uint8 data types.	-
flattem_dim	{HW, HWC, NHWC, NCHW}	-

Parameter	Valid value or range	Comment
storage_order	{0, 1}	-
output_shape	output_shape[0] ∈ [1, 16] output_shape[1] ∈ [1, 16384] output_shape[2] ∈ [1, 16384] output_shape[3] ∈ [1, 16384]	-

Where

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	Y
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1-3	-
Number of output tensors	1	-
input shape(s)	Input0_2: Supports int8, uint8, int16, and uint16 data types.	Supports any shape.
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

YuvToRgb

The main execution unit: TPC and AIFF

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports uint8 data type. Supports 2-dimensional input. Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 6220800]	-
output shape(s)	Supports uint8 data type.	-
format	{I420}	-
bits	{8}	-
conversion	{BT601, BT709, BT2020, SELF}	-
coefficient	-	1-d array. The length is 15.
coefficient_dtype	{int16}	-

ZeroFraction

The main execution unit: TPC

Supported quantization methods:

Quantization methods	Supported (Y or N)
8-bit feature map symmetric per-tensor quantization	Y
16-bit feature map symmetric per-tensor quantization	N
8-bit feature map asymmetric per-tensor quantization	N
8-bit weight per-channel quantization	Y

Parameter	Valid value or range	Comment
Number of input tensors	1	-
Number of output tensors	1	-
input shape(s)	Supports int8, uint8, int16, and uint16 data types. Supports 1, 2, 3, 4, 5-dimensional input. For 1-dims: Dim[0] ∈ [1, 4096] For 2-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 4096] For 3-dims:	-

Parameter	Valid value or range	Comment
	Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 4096] For 4-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1080] Dim[2] ∈ [1, 1920] Dim[3] ∈ [1, 4096] For 5-dims: Dim[0] ∈ [1, 32] Dim[1] ∈ [1, 1920] Dim[2] ∈ [1, 1080] Dim[3] ∈ [1, 1920] Dim[4] ∈ [1, 4096]	
output shape(s)	Supports int8, uint8, int16, and uint16 data types.	-

4 Operator map

The same operator in different AI frameworks may have slight difference in calculation. The Zhouyi Compass operators are developed to realize the calculation of TensorFlow, TensorFlow-Lite, Caffe, and ONNX.

The following table shows a map between the Zhouyi Compass operators and the operators (API) of TensorFlow, TensorFlow-Lite, Caffe, and ONNX. For each operator in different AI frameworks, the reference links are attached if available.

Released operators	TensorFlow	TensorFlow Lite	ONNX	Caffe
Abs	tf.math.abs	tfl.abs	Abs	AbsVal
AccidentalHits	tf.nn.compute_accidental_hits	NA	NA	NA
Acos	tf.math.acos	NA	Acos	NA
Acosh	tf.math.acosh	NA	Acosh	NA
Add	tf.math.add_n	tfl.add_n	Sum	Eltwise
ArgMax	tf.math.argmax	tfl.arg_max	ArgMax	ArgMax
ArgMin	tf.math.argmin	tfl.arg_min	ArgMin	NA
Asin	tf.math.asin	NA	Asin	NA
Asinh	tf.math.asinh	NA	Asinh	NA
AveragePooling2D	tf.nn.avg_pool2d	tfl.average_pool_2d	AveragePool	Pooling
			GlobalAveragePool	
AveragePooling3D	tf.nn.avg_pool3d	NA	AveragePool	NA
BasicLSTM	tf.keras.layers.LSTM	tfl.unidirectional_sequence_lstm	LSTM	LSTM
BatchNormalization	tf.nn.batch_normalization	NA	BatchNormalization	BatchNorm
BatchToSpace	tf.batch_to_space	tfl.batch_to_space_nd	NA	NA
BiasAdd	tf.nn.bias_add	NA	NA	Bias
BitwiseAnd	tf.bitwise.bitwise_and	NA	NA	NA
BitwiseNot	NA	NA	NA	NA
BitwiseOr	tf.bitwise.bitwise_or	NA	NA	NA
BitwiseXor	tf.bitwise.bitwise_xor	NA	NA	NA
BoundingBox	NA	NA	NA	NA
BNLL	NA	NA	NA	BNLL
Cast	tf.cast	tfl.cast	Cast	NA
Ceil	tf.math.ceil	tfl.ceil	Ceil	NA
Celu	NA	NA	Celu	NA

Released operators	TensorFlow	TensorFlow Lite	ONNX	Caffe
ChannelShuffle	Reshape+Transpose+Reshape+Split	Reshape+Transpose+Reshape+Split	Reshape+Transpose+Reshape+Split	NA
Clip	tf.clip_by_value	NA	Clip	NA
Compress	NA	NA	Compress	NA
Concat	tf.concat	tfl.concatenation	Concat	Concat
Constant	tf.ones	NA	NA	NA
	tf.ones_like	NA	NA	NA
	tf.zeros	NA	NA	NA
	tf.zeros_like	NA	NA	NA
	tf.range	tfl.range	Range	NA
	tf.constant	NA	Constant	NA
Convolution3D	tf.nn.conv3d	NA	Conv	Convolution
Convolution2D	tf.nn.conv2d	tfl.conv_2d	Conv	Convolution
ConvTranspose2D	tf.nn.conv2d_transpose	tfl.transpose_conv	ConvTranspose	Deconvolution
ConvTranspose3D	tf.nn.conv3d_transpose	NA	ConvTranspose	Deconvolution
Cosh	tf.math.cosh	NA	Cosh	NA
Cosine	tf.math.cos	tfl.cos	Cos	NA
Count	NA	NA	NA	NA
CRelu	tf.nn.crelu	NA	NA	NA
Crop	NA	NA	NA	Crop
CropAndResize	tf.image.crop_and_resize	NA	Resize	NA
CTCGreedyDecoder	tf.nn.ctc_greedy_decoder	NA	NA	NA
CumProd	tf.math.cumprod	NA	NA	NA
CumSum	tf.math.cumsum	tfl.cumsum	CumSum	NA
DataStride	NA	NA	NA	NA
DecodeBox	NA	NA	NA	NA
DepthToSpace	tf.nn.depth_to_space	tfl.depth_to_space	DepthToSpace	NA
DepthwiseConvolution	tf.nn.depthwise_conv2d	tfl.depthwise_conv_2d	NA	NA
Dilation2D	tf.nn.dilation2d	NA	NA	NA
Div	tf.math.divide	tfl.div	Div	NA
ElementwiseAdd	tf.math.add	tfl.add	Add	Eltwise

Released operators	TensorFlow	TensorFlow Lite	ONNX	Caffe
ElementwiseMax	tf.math.maximum	tfl.maximum	Max	Eltwise
ElementwiseMin	tf.math.minimum	tfl.minimum	Min	NA
ElementwiseMul	tf.math.multiply	tfl.mul	Mul	Eltwise
ElementwiseSub	tf.math.subtract	tfl.sub	Sub	NA
Elu	tf.nn.elu	tfl.elu	Elu	ELU
EmbeddingLookupSparse	tf.nn.embedding_lookup_sparse	NA	NA	NA
Erf	tf.math.erf	NA	Erf	NA
Erosion2D	tf.nn.erosion2d	NA	NA	NA
Exp	tf.math.exp	tfl.exp	Exp	Exp
Filter	NA	NA	NA	Filter
Floor	tf.math.floor	tfl.floor	Floor	NA
FractionalPool	tf.nn.fractional_avg_pool tf.nn.fractional_max_pool	NA	NA	NA
FullyConnected	tf.linalg.matmul	tfl.fully_connected	NA	InnerProduct
Gather	tf.gather	tfl.gather	Gather	NA
GatherElements	NA	NA	GatherElements	NA
GatherND	tf.gather_nd	tfl.gather_nd	GatherND	NA
Gelu	tf.nn.gelu	tfl.gelu	NA	NA
Gemm	NA	NA	Gemm	NA
GetValidCount	NA	NA	NA	NA
GridSample	NA	NA	GridSample	NA
GroupConvolution	split+conv2d+concat	split+conv2d+concat	Conv	Convolution
GroupNormalization	tfa.layers.GroupNormalization	NA	NA	NA
GRUv1	tf.keras.layers.GRU	NA	GRU	NA
GRUv3	tf.keras.layers.GRU	NA	GRU	NA
HardSigmoid	tf.keras.activations.hard_sigmoid	NA	HardSigmoid	NA
HardSwish	NA	tfl.hard_swish	HardSwish	NA
InstanceNormalization	tfa.layers.InstanceNormalization	NA	InstanceNormalization	NA
InTopK	tf.math.in_top_k	NA	NA	NA
LayerNormalization	tf.keras.layers.LayerNormalization	NA	LayerNormalizatin	NA
LeakyRelu	tf.nn.leaky_relu	tfl.leaky_relu	LeakyRelu	ReLU
LeftShift	tf.bitwise.left_shift	NA	BitShift	NA

Released operators	TensorFlow	TensorFlow Lite	ONNX	Caffe
L1Normalization	tf.linalg.normalize	NA	LpNormalization	NA
L1Pooling2D	NA	NA	LpPool	NA
L2Normalization	tf.math.l2_normalize	tfl.l2_normalization	LpNormalization	NA
L2Pooling2D	NA	NA	LpPool	NA
Log	tf.math.log	tfl.log	Log	Log
Logical	tf.math.logical_and	tfl.logical_and	And	NA
	tf.math.logical_not	tfl.logical_not	Not	NA
	tf.math.logical_or	tfl.logical_or	Or	NA
	tf.math.logical_xor	NA	Xor	NA
	tf.math.equal	tfl.equal	Equal	NA
	tf.math.not_equal	tfl.not_equal	NA	NA
	tf.math.greater	tfl.greater	Greater	NA
	tf.math.greater_equal	tfl.greater_equal	GreaterOrEqual	NA
	tf.math.less	tfl.less	Less	NA
	tf.math.less_equal	tfl.less_equal	LessOrEqual	NA
LogSoftmax	tf.nn.log_softmax	tfl.log_softmax	LogSoftmax	NA
LRN	tf.nn.local_response_normalization	tfl.local_response_normalization	LRN	LRN
MatMul	tf.linalg.matmul	tfl.batch_matmul	MatMul	InnerProduct
MaxPooling2D	tf.nn.max_pool2d	tfl.max_pool_2d	MaxPool	Pooling
			GlobalMaxPool	
MaxPooling3D	tf.nn.max_pool3d	NA	MaxPool	NA
MaxPoolingWithArgMax	tf.nn.max_pool_with_argmax	NA	MaxPool	Pooling
MaxRoiPool	NA	NA	MaxRoiPool	NA
MaxUnpool	NA	NA	MaxUnpool	NA
MeanVarianceNormalization	NA	NA	MeanVarianceNormalization	MVN
Meshgrid	NA	NA	NA	NA
Mish	tfa.activations.mish	NA	NA	NA
Mod	tf.math.floormod	tfl.floor_mod	Mod	NA
Moments	tf.nn.moments	NA	NA	NA
Mul	tf.math.multiply	tfl.mul	Mul	Eltwise

Released operators	TensorFlow	TensorFlow Lite	ONNX	Caffe
MultiboxTransformLoc	NA	NA	NA	NA
Negative	tf.math.negative	tfl.neg	Neg	NA
NMS	tf.image.non_max_suppression	NA	NonMaxSuppression	NA
	tf.raw_ops.NonMaxSuppressionV4	tfl.non_max_suppression_v4		
	tf.image.non_max_suppression_with_scores	tfl.non_max_suppression_v4		
NormalizedMoments	tf.nn.normalize_moments	NA	NA	NA
OneHot	tf.one_hot	tfl.one_hot	OneHot	NA
Pad	tf.pad	tfl.pad	Pad	NA
		tfl.padv2		
		tfl.mirror_pad		
Pow	tf.math.pow	tfl.pow	Pow	Power
PRelu	tf.keras.layers.PReLU	tfl.prelu	Prelu	PReLU
Reciprocal	tf.math.reciprocal	NA	Reciprocal	NA
ReduceAll	tf.math.reduce_all	tfl.reduce_all	NA	NA
ReduceAny	tf.math.reduce_any	tfl.reduce_any	NA	NA
ReduceL1	NA	NA	ReduceL1	NA
ReduceL2	NA	NA	ReduceL2	NA
ReduceMax	tf.math.reduce_max	tfl.reduce_max	ReduceMax	NA
ReduceMean	tf.math.reduce_mean	tfl.mean	ReduceMean	Reduction
ReduceMin	tf.math.reduce_min	tfl.reduce_min	ReduceMin	NA
ReduceProd	tf.math.reduce_prod	tfl.reduce_prod	ReduceProd	NA
ReduceSum	tf.math.reduce_sum	tfl.sum	ReduceSum	Reduction
ReduceUnbiasedVariance	NA	NA	NA	NA
ReduceVariance	tf.math.reduce_variance	NA	NA	NA
Region	NA	NA	NA	NA
RegionFuse	NA	NA	NA	NA
Relu	tf.nn.relu	tfl.relu	Relu	ReLU
Relu6	tf.nn.relu6	tfl.relu6	NA	NA
Repeat	NA	NA	NA	NA
Reshape	tf.reshape	tfl.reshape	Reshape	Reshape
	tf.expand_dims	tfl.expand_dims	Expand	NA

Released operators	TensorFlow	TensorFlow Lite	ONNX	Caffe
	tf.keras.layers.Flatten	NA	Flatten	Flatten
	tf.squeeze	tfl.squeeze	Squeeze	NA
Resize	tf.compat.v1.image.resize_bilinear	tfl.resize_bilinear	Resize	NA
	tf.compat.v1.image.resize_nearest_neighbor	tfl.resize_nearest_neighbor		
ReverseSequence	tf.reverse	tfl.reverse_v2	ReverseSequence	NA
	tf.reverse_sequence	tfl.reverse_sequence		
RgbToYuv	NA	NA	NA	NA
RightShift	tf.bitwise.right_shift	NA	BitShift	NA
RoiAlign	NA	NA	RoiAlign	NA
Round	tf.math.round	tfl.round	Round	NA
Rsqrt	tf.math.rsqrt	tfl.rsqrt	NA	NA
ScatterElements	NA	NA	ScatterElements	NA
ScatterND	tf.scatter_nd	tfl.scatter_nd	ScatterND	NA
	tf.tensor_scatter_nd_add			
	tf.tensor_scatter_nd_update			
SegmentSumReduce	tf.math.segment_sum	tfl.segment_sum	NA	NA
Selu	tf.nn.selu	NA	Selu	NA
Shrink	NA	NA	Shrink	NA
Sigmoid	tf.math.sigmoid	tfl.logistic	Sigmoid	Sigmoid
Sign	tf.math.sign	NA	Sign	NA
Silu	NA	NA	Sigmoid+Mul	NA
Sine	tf.math.sin	tfl.sin	Sin	NA
Sinh	tf.math.sinh	NA	Sinh	NA
Slice	tf.slice	tfl.slice	Slice	Slice
	tf.strided_slice	tfl.strided_slice		
Softmax	tf.nn.softmax	tfl.softmax	Softmax	Softmax
Softplus	tf.math.softplus	NA	Softplus	NA
Softsign	tf.nn.softsign	NA	Softsign	NA
SpaceToBatch	tf.space_to_batch	tfl.space_to_batch_nd	NA	NA
SpaceToDepth	tf.nn.space_to_depth	tfl.space_to_depth	SpaceToDepth	NA
Split	tf.split	tfl.split	Split	NA

Released operators	TensorFlow	TensorFlow Lite	ONNX	Caffe
Sqrt	tf.math.sqrt	tfl.sqrt	Sqrt	NA
Square	tf.math.square	tfl.square	NA	NA
SquaredDifference	tf.math.squared_difference	tfl.squared_difference	NA	NA
Sub	tf.math.subtract	tfl.sub	Sub	NA
SufficientStatistics	tf.nn.sufficient_statistics	NA	NA	NA
Swish	tf.keras.activations.swish	NA	NA	NA
Tan	tf.math.tan	NA	Tan	NA
Tanh	tf.math.tanh	tfl.tanh	Tanh	TanH
ThresholdedRelu	tf.keras.layers.ThresholdedReLU	NA	ThresholdedRelu	NA
Tile	tf.tile	tfl.tile	Tile	NA
TopK	tf.math.top_k	tfl.topk_v2	TopK	ArgMax
Transpose	tf.transpose	tfl.transpose	Transpose	NA
Trunc	NA	NA	NA	NA
UpsampleByIndex	NA	NA	NA	NA
Where	tf.where	tfl.select	NA	NA
YuvToRgb	NA	NA	NA	NA
ZeroFraction	tf.math.zero_fraction	NA	NA	NA