



The **OpenVX™** S16 Extension

Version 1.1 (Provisional)

Document Revision: b420242

Generated on Mon Dec 19 2016 15:07:43

Khronos Vision Working Group

Editor: Frank Brill
Editor: Erik Rainey
Editor: Jesse Villarreal

Copyright ©2016 The Khronos Group Inc.



Copyright ©2016 The Khronos Group Inc. All Rights Reserved.

This specification is protected by copyright laws and contains material proprietary to the Khronos Group, Inc. It or any components may not be reproduced, republished, distributed, transmitted, displayed, broadcast or otherwise exploited in any manner without the express prior written permission of Khronos Group. You may use this specification for implementing the functionality therein, without altering or removing any trademark, copyright or other notice from the specification, but the receipt or possession of this specification does not convey any rights to reproduce, disclose, or distribute its contents, or to manufacture, use, or sell anything that it may describe, in whole or in part.

Khronos Group grants express permission to any current Promoter, Contributor or Adopter member of Khronos to copy and redistribute UNMODIFIED versions of this specification in any fashion, provided that NO CHARGE is made for the specification and the latest available update of the specification for any version of the API is used whenever possible. Such distributed specification may be reformatted AS LONG AS the contents of the specification are not changed in any way. The specification may be incorporated into a product that is sold as long as such product includes significant independent work developed by the seller. A link to the current version of this specification on the Khronos Group website should be included whenever possible with specification distributions.

Khronos Group makes no, and expressly disclaims any, representations or warranties, express or implied, regarding this specification, including, without limitation, any implied warranties of merchantability or fitness for a particular purpose or non-infringement of any intellectual property. Khronos Group makes no, and expressly disclaims any, warranties, express or implied, regarding the correctness, accuracy, completeness, timeliness, and reliability of the specification. Under no circumstances will the Khronos Group, or any of its Promoters, Contributors or Members or their respective partners, officers, directors, employees, agents or representatives be liable for any damages, whether direct, indirect, special or consequential damages for lost revenues, lost profits, or otherwise, arising from or in connection with these materials.

SAMPLE CODE and EXAMPLES, as identified herein, are expressly depicted herein with a "grey" watermark and are included for illustrative purposes only and are expressly outside of the Scope as defined in Attachment A - Khronos Group Intellectual Property (IP) Rights Policy of the Khronos Group Membership Agreement. A Member or Promoter Member shall have no obligation to grant any licenses under any Necessary Patent Claims covering SAMPLE CODE and EXAMPLES.

Contents

- 1 Introduction** **2**
- 1.1 Overview of Extension 2
- 1.2 Changes to the OpenVX 1.1 Specification 2
 - 1.2.1 Inputs 2
 - 1.2.2 Outputs 4
 - 1.2.3 Vision Functions 5
 - Bitwise Operations 5
 - Custom Convolution 5
 - Fast Corners 6

Chapter 1

Introduction

1.1 Overview of Extension

This extension is intended to define the subset of behaviors and data types of the signed 16-bit support for OpenVX.

1.2 Changes to the OpenVX 1.1 Specification

The S16 extension enhances the input and output types per each kernel defined in the OpenVX 1.1 standard. The table below indicates the changes to each kernel for input and output.

Input and output argument types should be the same (e.g. input S16 and output S16) unless stated otherwise in the function description. In cases where having S16 inputs could lead to the overflow of S16 outputs, the behavior is analogous to what is currently in the standard for when the inputs are U8.

In the main standard, where the input is U8 and the output is also U8, then the output is converted according to the overflow policy in the function definition. Analogously, for this extension, where the inputs and outputs are both S16, the output is converted as necessary according to the overflow policy in the function definition.

In the main standard, where the input can be U8 and the output S16, the zero-extended answer is just written into the output. Analogously, for this extension, where the input can be S16 and the output S32, the sign-bit-extended result is written to the output.

1.2.1 Inputs

Vision Function	U8	U16	S16	U32	S32	F32	color
AbsDiff	1.0		1.0.1				
Accumulate	1.0		ext				
Accumulate↔ Squared	1.0		ext				
Accumulate↔ Weighted	1.0		ext				
Add	1.0		1.0				
And	1.0		ext				
Box3x3	1.0		ext				
Canny↔ Edge↔ Detector	1.0		ext				

Channel↔ Combine	1.0						
Channel↔ Extract							1.0
Color↔ Convert							1.0
Convert↔ Depth	1.0	ext	1.0	ext	ext		
Convolve	1.0		ext				
Dilate3x3	1.0						
Equalize↔ Histogram	1.0		ext				
Erode3x3	1.0						
Fast↔ Corners	1.0		ext				
Gaus- sian3x3	1.0		ext				
Harris↔ Corners	1.0		ext				
Half↔ Scale↔ Gaussian	1.0		ext				
Histogram	1.0		ext				
Integral↔ Image	1.0						
Table↔ Lookup	1.0		1.1				
Laplacian↔ Pyramid	1.1						
Laplacian↔ Reconstruct			1.1				
Magnitude			1.0				
MeanStd↔ Dev	1.0		ext				
Median3x3	1.0		ext				
MinMax↔ Loc	1.0		1.0				
Multiply	1.0		1.0				
Non↔ Linear↔ Filter	1.1						
Not	1.0		ext				
Optical↔ FlowPyr↔ LK	1.0		ext				
Or	1.0		ext				
Phase			1.0				
Gaussian↔ Pyramid	1.0		ext				

Remap	1.0		ext				
Scale↔ Image	1.0		ext				
Sobel3x3	1.0		ext				
Subtract	1.0		1.0				
Threshold	1.0		ext				
WarpAffine	1.0		ext				
Warp↔ Perspective	1.0		ext				
Xor	1.0		ext				

1.2.2 Outputs

Vision Function	U8	U16	S16	U32	S32	F32	color
AbsDiff	1.0	ext	1.0.1				
Accumulate			1.0		ext		
Accumulate↔ Squared			1.0		ext		
Accumulate↔ Weighted	1.0				ext		
Add	1.0		1.0		ext		
And	1.0		ext				
Box3x3	1.0		ext				
Canny↔ Edge↔ Detector	1.0		ext				
Channel↔ Combine							1.0
Channel↔ Extract	1.0						
Color↔ Convert							1.0
Convert↔ Depth	1.0	ext	1.0	ext	ext		
Convolve	1.0		1.0		ext		
Dilate3x3	1.0						
Equalize↔ Histogram	1.0		ext				
Erode3x3	1.0						
Fast↔ Corners	1.0						
Gaus- sian3x3	1.0		ext				
Harris↔ Corners	1.0						
Half↔ Scale↔ Gaussian	1.0		ext				

Histogram				1.0			
Integral↔ Image				1.0			
Table↔ Lookup	1.0		1.1				
Laplacian↔ Pyramid			1.1				
Laplacian↔ Reconstruct	1.1						
Magnitude			1.0				
MeanStd↔ Dev						1.0	
Median3x3	1.0		ext				
MinMax↔ Loc	1.0		1.0	1.0			
Multiply	1.0		1.0		ext		
Non↔ Linear↔ Filter	1.1						
Not	1.0		ext				
Optical↔ FlowPyr↔ LK							
Or	1.0		ext				
Phase	1.0						
Gaussian↔ Pyramid	1.0		ext				
Remap	1.0		ext				
Scale↔ Image	1.0		ext				
Sobel3x3			1.0		ext		
Subtract	1.0		1.0		ext		
Threshold	1.0		ext				
WarpAffine	1.0		ext				
Warp↔ Perspective	1.0		ext				
Xor	1.0		ext				

1.2.3 Vision Functions

The following sections describe additional changes and clarifications to existing kernel definitions beyond those already described in sections [Inputs](#) and [Outputs](#).

Bitwise Operations

Referring to: AND, EXCLUSIVE OR, INCLUSIVE OR, and NOT.

All bit-wise operations on signed operands are executed in twos-complement representation of the values.

Custom Convolution

The current spec says if the input type is U8 and the output type is S16, then the output is simply the sum/scale. However, if the output type is U8, then the output saturates on both ends: 0 if sum/scale < 0, and 255 if sum/scale > 255. Analogously, S16 outputs should saturate to -32768 if sum/scale < -32768, and 32767 if sum/scale > 32767, and just sum/scale otherwise.

For `VX_DF_IMAGE_S16` output, an additional step is taken:

$$out\ put(x,y) = \begin{cases} -32768 & \text{if } sum/scale < -32768 \\ 32767 & \text{if } sum/scale > 32767 \\ sum/scale & \text{otherwise} \end{cases}$$

For `VX_DF_IMAGE_S32` output, the summation is simply set to the output

$$out\ put(x,y) = sum/scale$$

Fast Corners

When the input image is of type `VX_DF_IMAGE_S16`, the value of the intensity difference threshold *strength_thresh* of type `VX_TYPE_FLOAT32` must be within:

$$UINT16_{MIN} < t < UINT16_{MAX}$$