User's Manual

OpenCV Accelerator

User's Manual

RZ/V2L

Renesas Electronics
www.renesas.com

Rev.1.10 Feb. 2024

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving patents, copyrights, or other intellectual property rights of third parties, by or arising from the use of Renesas Electronics products or technical information described in this document, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You shall be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
5. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
6. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.
   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (“Vulnerability Issues”). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
8. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, such as safety design for hardware and software, including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. Renesas Electronics products and technologies shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
12. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
14. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.
(Note2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.5.0-1 October 2020)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan

www.renesas.com

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.

Contact information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit: www.renesas.com/contact/.

General Precautions in the Handling of Microprocessing Unit and Microcontroller Unit Products

The following usage notes are applicable to all Microprocessing unit and Microcontroller unit products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Precaution against Electrostatic Discharge (ESD)
   A strong electrical field, when exposed to a CMOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop the generation of static electricity as much as possible, and quickly dissipate it when it occurs. Environmental control must be adequate. When it is dry, a humidifier should be used. This is recommended to avoid using insulators that can easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors must be grounded. The operator must also be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions must be taken for printed circuit boards with mounted semiconductor devices.
2. Processing at power-on
   The state of the product is undefined at the time when power is supplied. The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the time when power is supplied. In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the time when power is supplied until the reset process is completed. In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the time when power is supplied until the power reaches the level at which resetting is specified.
3. Input of signal during power-off state
   Do not input signals or an I/O pull-up power supply while the device is powered off. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Follow the guideline for input signal during power-off state as described in your product documentation.
4. Handling of unused pins
   Handle unused pins in accordance with the directions given under handling of unused pins in the manual. The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of the LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible.
5. Clock signals
   After applying a reset, only release the reset line after the operating clock signal becomes stable. When switching the clock signal during program execution, wait until the target clock signal is stabilized. When the clock signal is generated with an external resonator or from an external oscillator during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Additionally, when switching to a clock signal produced with an external resonator or by an external oscillator while program execution is in progress, wait until the target clock signal is stable.
6. Voltage application waveform at input pin
   Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between V_{IL} (Max.) and V_{IH} (Min.) due to noise, for example, the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between V_{IL} (Max.) and V_{IH} (Min.).
7. Prohibition of access to reserved addresses
   Access to reserved addresses is prohibited. The reserved addresses are provided for possible future expansion of functions. Do not access these addresses as the correct operation of the LSI is not guaranteed.
8. Differences between products
   Before changing from one product to another, for example to a product with a different part number, confirm that the change will not lead to problems. The characteristics of a microprocessing unit or microcontroller unit products in the same group but having a different part number might differ in terms of internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

How to Use This Manual

1. Purpose and Target Readers

This document is designed to provide the user with an understanding of the software development environment for RZ/V2L Group processors. It is intended for users developing software incorporating the processors. A basic knowledge of software development and Linux systems is necessary to use this document.

Particular attention should be paid to the precautionary notes when using the manual. These notes occur within the body of the text, at the end of each section, and in the Usage Notes section.

The revision history summarizes the locations of revisions and additions. It does not list all revisions. Refer to the text of the manual for details.

2. List of Abbreviations and Acronyms

Abbreviation	Full form
BSP	Board support package
DRP	Dynamically Reconfigurable Processor
OpenCV	Open-source Computer Vision library
OpenCVA	OpenCV Accelerator

Table of Contents

1. Overview

1.1 overview

This manual explains the OpenCV Accelerator in RZ/V2L Linux. OpenCV Accelerator performs OpenCV function in high performance without changing OpenCV API.

1.2 About OpenCV Accelerator

OpenCV Accelerator (referred to below as OpenCVA) can perform OpenCV functions using Dynamically Reconfigurable Processor (=DRP).

If DRP is enabled (see chapter 5.1) and the parameters of OpenCV meet the conditions (see chapter 4), OpenCV is executed using DRP. If DRP is disabled or the parameters do not match, then OpenCV is executed using CPU. If the DRP is used, OpenCV will be executed at high performance. The performance depends on the parameters.

The output OpenCVA using DRP is almost identical to output OpenCV using CPU, but not exactly the same. Much of this difference is due to the accuracy of in the DRP's arithmetic unit, and some is due to differences in algorithms.

DRP is also used for AI processing, however since DRP is a common HW resource, OpenCVA and AI cannot be used at the same time. For the same reason, OpenCVA must be used in single process and single thread. In case of conflicting use of DRP, it causes an exception errors or performance degradation.

For details, please see Chapter 5.2 and 6.

1.3 Related Documents

1. RZ/V2L OpenCV Accelerator Support Package Release Note (R11AN0845)
2. RZ/V2L OpenCV Accelerator Sample Application Note(R11AN0846)

1.4 Restrictions

None

2.Usage of OpenCV Accelerator

2.1 Requirements

Before using OpenCV Accelerator, you need to install RZ/V2L OpenCV Accelerator Support package.

For the details, refer to the RZ/V2L OpenCV Accelerator Support Package Release Note (R11AN0845).

2.2 OpenCV version

OpenCVA is based on as follows version of OpenCV.

OpenCV 4.1.0-r0

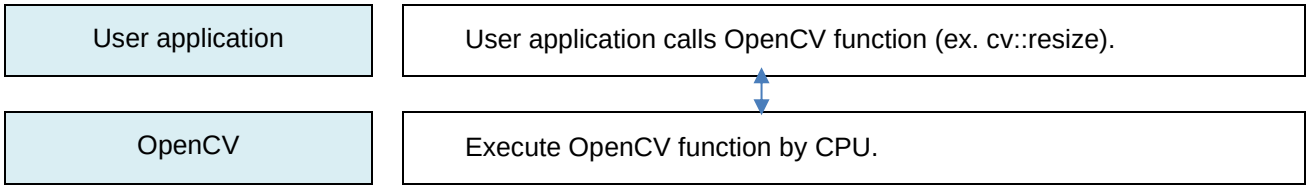
OpenCV Document URL: https://docs.opencv.org/4.1.0/index.html

2.3 How to use

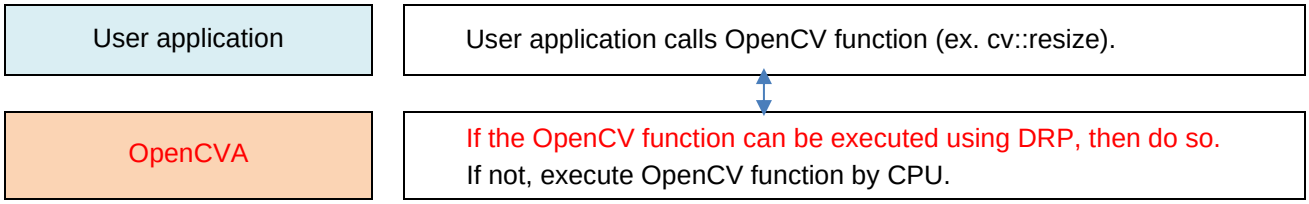
You can use OpenCVA same as OpenCV as usual and you do not need to consider of OpenCVA architecture. OpenCVA is automatically executed by DRP as follows if it matches the conditions under which DRP can be used. For the DRP using conditions, see Chapter 4.

OpenCVA can disable DRP, for each function. See Chapter 5.1 for details.

In case of normal OpenCV



In case of OpenCVA



Note: One OpenCV function may use another OpenCV functions. For example, "morphologyEX" function uses "dilate" and “erode” functions internally. In this case, if "dilate" and “erode” can use DRP in OpenCVA, then these functions included in “morphologyEX” would also use DRP in OpenCVA. There are other similar cases.

3. OpenCVA Function List

The following table lists the OpenCV functions that can be executed using DRP in RZ/V2L OpenCV Accelerator Support Package Ver 1.10.

OpenCV function name	Function
resize	Image Resize.
cvtColor	Change color space.
cvtColorTwoPlane	Change color space.
GaussianBlur	Gaussian filter process.
dilate	Areas of bright regions grow.
erode	Areas of dark regions grow.
morphologyEX	Combination of dilate and erode.
filter2D	Image convolving.
Sobel	Extracting image edges.
adaptiveThreshold	Transforms a grayscale image to a binary image according to the formulae.
matchTemplate	Compares a template against overlapped image regions.
warpAffine	Transforms the source image using the 2x3 matrix.
warpPerspective	Transforms the source image using the 3x3 matrix.
pyrDown	Downsampling step of the Gaussian pyramid construction.
pyrUp	Upsampling step of the Gaussian pyramid construction.

4. OpenCVA API specification and condition for using DRP

This chapter describes the OpenCV API that can be executed by DRP, and their conditions for using DRP.

4.1 resize

4.1.1 outline

Resize an image, set desired scale of each axis or set desired size for the output image.

```
void cv::resize (InputArray src, OutputArray dst, Size dsize,
                 double fx = 0, double fy = 0, int interpolation = INTER_LINEAR)
```

parameter	required/optional	description
src	required	Source image
dst	required	Destination image
dsize	required	Desired size for the destination image
fx (=0)	optional	Horizontal axis scale
fy (=0)	optional	Vertical axis scale
interpolation (=INTER_LINEAR)	optional	One of the interpolation algorithm methods (see InterpolationFlags in OpenCV document)

Note: For more information, refer to the OpenCV Document.

4.1.2 Conditions for using DRP

If the following conditions apply, OpenCVA execute resizing process using DRP.

parameter	range/values	note
src	width: 2 - 3840 height: 2 - 2160 channels: 1 - 4 bit-depth: 8bit	
dsize	width: 2 - 3840 height: 2 - 2160	'0' can also be set. In that case, "fx" and "fy" parameter are valid.
fx	(src_img width * fx): 2 - 3840	Valid if dsize is 0.
fy	(src_img height * fy): 2 - 2160	Valid if dsize is 0.
interpolation	INTER_NEAREST or INTER_LINEAR	

4.2 cvtColor

4.2.1 outline

Converts an input image from one color space to another.

```
Void cv::cvtColor (InputArray src, OutputArray dst, int code, int dstCn = 0 )
```

parameter	required/optional	description
src	required	Source image
dst	required	Destination image
code	required	Color space conversion code (see ColorConversionCode in OpenCV document)
dstCn (=0)	optional	Number of channels in the destination image. If the default parameter (=0) specified, the number of the channels is derived automatically from src and code

Note: For more information, refer to the OpenCV Document.

4.2.2 Conditions for using DRP

If the following conditions apply, OpenCVA execute cvtColor process using DRP.

parameter	range/values	note
src	width: 4 - 3840, even height: 6 - 2160, even channels: 2 bit-depth: 8bit	
code	COLOR_YUV2RGB_UYVY COLOR_YUV2BGR_UYVY COLOR_YUV2RGB_YUY2 COLOR_YUV2BGR_YUY2 COLOR_YUV2RGB_YVYU COLOR_YUV2BGR_YVYU COLOR_YUV2RGB_VYUY COLOR_YUV2BGR_VYUY	
dstCn	0	Default value

4.3 cvtColorTwoPlane

4.3.1 outline

Converts an image from one color space to another where the source image is stored in two planes.

```
Void cv::cvtColorTwoPlane (InputArray src1, InputArray src2,
                           OutputArray dst, int code)
```

parameter	required/optional	description
src1	required	Source image of the Y plane.
src2	required	Source image of U/V plane.
dst	required	Destination image
code	required	Specifies the type of conversion

Note: For more information, refer to the OpenCV Document.

4.3.2 Conditions for using DRP

If the following conditions apply, OpenCVA execute cvtColorTwoPlane process using DRP.

parameter	range/values	note
src1	width: 4 - 3840, even height: 6 - 2160, even channels: 1 bit-depth: 8bit	
src2	width: Half size of src_img1 height: Half size of src_img1 channels: 2 bit-depth: 8bit	
code	COLOR_YUV2BGR_NV12 COLOR_YUV2RGB_NV12 COLOR_YUV2BGR_NV21 COLOR_YUV2RGB_NV21	

4.4 GaussianBlur

4.4.1 outline

GaussianBlur is one of image blurring (image smoothing) function. In this function a Gaussian kernel is used to produce the smoothed image.

GaussianBlur process with DRP supports following Gaussian filters with kernel sizes.

Kernel size: 3, 5, or 7

Filters used are as follows.

[Kernel size:3x3]

1/16	2/16	1/16
2/16	4/16	2/16
1/16	2/16	1/16

[Kernel size:5x5]

1/256	4/256	6/256	4/256	1/256
4/256	16/256	24/256	16/256	4/256
6/256	24/256	36/256	24/256	6/256
4/256	16/256	24/256	16/256	4/256
1/256	4/256	6/256	4/256	1/256

[Kernel size:7x7]

1/4096	6/4096	15/4096	20/4096	15/4096	6/4096	1/4096
6/4096	36/4096	90/4096	120/4096	90/4096	36/4096	6/4096
15/4096	90/4096	225/4096	300/4096	225/4096	90/4096	15/4096
20/4096	120/4096	300/4096	400/4096	300/4096	120/4096	20/4096
15/4096	90/4096	225/4096	300/4096	225/4096	90/4096	15/4096
6/4096	36/4096	90/4096	120/4096	90/4096	36/4096	6/4096
1/4096	6/4096	15/4096	20/4096	15/4096	6/4096	1/4096

```
Void GaussianBlur (InputArray src, OutputArray dst, Size ksize,
                   double sigmaX, double sigmaY = 0, int borderType = BORDER_DEFAULT)
```

parameter	required/optional	description
src	required	Input image
dst	required	Destination image
ksize	required	Gaussian kernel size
sigmaX	required	Gaussian kernel standard deviation in X direction
sigmaY (=0)	optional	Gaussian kernel standard deviation in Y direction
borderType (=BORDER_DEFAULT)	optional	Pixel extrapolation method (see BorderTypes in OpenCV document)

Note: For more information, refer to the OpenCV Document.

4.4.2 Conditions for using DRP

If the following conditions and apply, OpenCVA execute GaussianBlur process using DRP.

parameter	range/values	note
src	width: 16 - 3840, even height: 16 - 2160, even channels: 1 - 4 bit-depth: 8bit size limit: 4K(3840x2160)	
ksize	{3,3}, {5,5}, or {7,7}	Gaussian kernel size = 3, 5, or 7
sigmaX	0	
sigmaY	0	Default value
borderType	BORDER_DEFAULT	Default value

4.5 dilate

4.5.1 outline

Expands the area of bright areas in the image.

```
void cv::dilate (InputArray src, OutputArray dst, InputArray kernel,
                 Point anchor = Point (-1, -1), int iterations = 1,
                 int borderType = BORDER_CONSTANT,
                 const Scalar & borderValue = morphologyDefaultBorderValue() )
```

parameter	required/optional	description
src	required	Input image
dst	required	Destination image
kernel	required	Dilate kernel size
anchor (=Point (-1, -1))	optional	Position of the anchor within the element. Default value (-1, -1) means that the anchor is at the element center.
iterations (=1)	optional	Number of times dilation
borderType (=BORDER_CONSTANT)	optional	Pixel extrapolation method
borderValue (=morphologyDefaultBorderValue())	optional	Border value (see morphologyDefaultBorderValue() in OpenCV document)

Note: For more information, refer to the OpenCV Document.

4.5.2 Conditions for using DRP

If the following conditions apply, OpenCVA execute dilate process using DRP.

Item	range/values	note
src	width: 16 - 3840, multiple of 4 height: 16 - 2160, multiple of 4 channels: 1 - 4 bit-depth: 8bit size limit: 4K(3840x2160)	
kernel	Mat()	3 x 3 rectangular structuring element
anchor	(-1, -1)	Default value
iterations	1 - 3840	
borderType	BORDER_CONSTANT or BORDER_REPLICATE	
borderValue	morphologyDefaultBorderValue()	Default value

4.6 erode

4.6.1 outline

Reduce the area of bright areas in the image.

```
void cv::erode (InputArray src, OutputArray dst, InputArray kernel,
                Point anchor = Point (-1, -1), int iterations = 1,
                int borderType = BORDER_CONSTANT,
                const Scalar & borderValue = morphologyDefaultBorderValue() )
```

Parameter	required/optional	description
src	required	Input image
dst	required	Destination image
kernel	required	Erode kernel size
anchor (=Point (-1, -1))	optional	Position of the anchor within the element. Default value (-1, -1) means that the anchor is at the element center.
iterations (=1)	optional	Number of times erosion
borderType (=BORDER_CONSTANT)	optional	Pixel extrapolation method
borderValue (=morphologyDefaultBorderValue())	optional	Border value

Note: For more information, refer to the OpenCV Document.

4.6.2 Conditions for using DRP

If the following conditions apply, OpenCVA execute erode process using DRP.

Item	range/values	note
src	width: 16 - 3840, multiple of 4 height: 16 - 2160, multiple of 4 channels: 1 - 4 bit-depth: 8bit size limit: 4K(3840x2160)	
kernel	Mat()	3 x 3 rectangular structuring element
anchor	(-1, -1)	Default value
iterations	1 - 3840	
borderType	BORDER_CONSTANT or BORDER_REPLICATE	
borderValue	morphologyDefaultBorderValue()	Default value

4.7 morphologyEX

4.7.1 outline

Performs advanced morphological transformations.

This function uses both “dilate” and “erode” internally.

```
void cv::morphologyEX (InputArray src, OutputArray dst, int op,
                       InputArray kernel, Point anchor = Point (-1, -1),
                       int iterations = 1, int borderType = BORDER_CONSTANT,
                    const Scalar & borderValue = morphologyDefaultBorderValue() )
```

Parameter	required/optional	description
src	required	Input image
dst	required	Destination image
op	required	Type of a morphological operation (see MorphTypes in OpenCV document)
kernel	required	Kernel size
anchor (=Point (-1, -1))	optional	Position of the anchor within the element. Default value (-1, -1) means that the anchor is at the element center.
iterations (=1)	optional	Number of times transformation
borderType (=BORDER_CONSTANT)	optional	Pixel extrapolation method
borderValue (=morphologyDefaultBorderValue())	optional	Border value

Note: For more information, refer to the OpenCV Document.

4.7.2 Conditions for using DRP

If the following conditions apply, OpenCVA execute morphologyEX process using DRP.

Item	range/values	note
src	width: 16 - 3840, multiple of 4 height: 16 - 2160, multiple of 4 channels: 1 - 4 bit-depth: 8bit size limit: 4K(3840x2160)	
kernel	Mat()	3 x 3 rectangular structuring element
anchor	(-1, -1)	Default value
iterations	1 - 3840	
borderType	BORDER_CONSTANT or BORDER_REPLICATE	
borderValue	morphologyDefaultBorderValue()	Default value

4.8 filter2D

4.8.1 outline

Convolves an image with the kernel.

```
void cv::filter2D (InputArray src, OutputArray dst, int ddepth, InputArray kernel,
                   Point anchor = Point(-1,-1) , double delta = 0,
                   int borderType = BORDER_DEFAULT )
```

Parameter	required/optional	description
src	required	Input image
dst	required	Destination image
ddepth	required	Destination image depth
kernel	required	Convolution kernel
anchor (=Point(-1,-1))	optional	Position of the anchor within the element. Default value (-1, -1) means that the anchor is at the element center.
delta (=0)	optional	Value to add to destination image
borderType (=BORDER_DEFAULT)	optional	Pixel extrapolation method

Note: For more information, refer to the OpenCV Document.

4.8.2 Conditions for using DRP

If the following conditions apply, OpenCVA execute filter2D process using DRP.

Item	range/values	note
src	width: 16 - 3840, even height: 16 - 2160, even channels: 1 - 4 bit-depth: 8bit size limit: 4K(3840x2160)	
ddepth	-1	The same depth as the input image.
kernel	size: {3,3} depth: CV_32F value: all element values are between -128 and 127	
anchor	(-1, -1)	Default value
delta	0	Default value
borderType	BORDER_DEFAULT	Default value

4.9 Sobel

4.9.1 outline

Calculates the first image derivatives.

```
void cv::Sobel (InputArray src, OutputArray dst, int ddepth, int dx, int dy,
                int ksize = 3, double scale = 1, double delta = 0,
                int borderType = BORDER_DEFAULT)
```

Parameter	required/optional	description
src	required	Input image
dst	required	Destination image
ddepth	required	Destination image depth
dx	required	Order of the derivative x
dy	required	Order of the derivative y
ksize (=3)	optional	Size of the extended Sobel kernel
scale (=1)	optional	Scale factor
delta (=0)	optional	Value to add to destination image
intborderType (=BORDER_DEFAULT)	optional	Pixel extrapolation method

Note: For more information, refer to the OpenCV Document.

4.9.2 Conditions for using DRP

If the following conditions apply, OpenCVA execute sobel process using DRP.

Item	range/values	note
src	width: 16 - 3840, even height: 16 - 2160, even channels: 1 - 4 bit-depth: 8bit size limit: 4K(3840x2160)	
ddepth	CV_8U	
dx, dy	“dx = 0, dy = 1” or “dx = 1, dy = 0”	
ksize	3	default value
scale	1	default value
delta	0	default value
borderType	BORDER_DEFAULT	default value

4.10 adaptiveThreshold

4.10.1 outline

Transforms a grayscale image to a binary image according to the formulae.

```
Void cv::adaptiveThreshold (InputArray src, OutputArray dst, double maxValue,
                            int adaptiveMethod, int thresholdType, int blockSize,
                            double C)
```

Parameter	required/optional	description
src	required	Input image
dst	required	Destination image
maxValue	required	Non-zero value assigned to the pixels for which the condition is satisfied
adaptiveMethod	required	Adaptive thresholding algorithm to use (see adaptiveThresholdTypes in OpenCV document)
thresholdType	required	Threshold type (see ThresholdTypes in OpenCV document)
blocksize	required	Size of a pixel neighborhood that is used to calculate a threshold value
C	required	Constant subtracted from the mean

Note: For more information, refer to the OpenCV Document.

4.10.2 Conditions for using DRP

If the following conditions apply, OpenCVA execute adaptiveThreshold process using DRP.

Item	range/values	note
src	width: 16 - 3840, even height: 16 - 2160, even channels: 1 bit-depth: 8bit size limit: 4K(3840x2160)	
maxValue	0 - 255	
adaptiveMethod	ADAPTIVE_THRES_MEAN_C	
thresholdType	THRES_BINARY THRES_BINARY_INV	
blocksize	3 - 255, odd	
C	0 - 255	

4.11 matchTemplate

4.11.1 outline

Compares a template against overlapped image regions.

Caution: "matchTemplate" function is very slow in some cases using DRP, depending on the parameters. Please test if the matchTemplate process with DRP performance is enough, using the assumed parameter.

void cv::matchTemplate(InputArray image, InputArray templ, OutputArray result, int method,

InputArray mask = noArray())

Parameter	required/optional	description
image	required	Input image
templ	required	Template image
result	required	Map of comparison results Depth:32FC1 Width: image width - templ width +1 Height: image height - templ height +1
method	required	Specifying the comparison method (see TemplateMatchModes in OpenCV document)
mask (= noArray())	optional	Mask of searched template

Note: For more information, refer to the OpenCV Document.

4.11.2 Conditions for using DRP

If the following conditions apply, OpenCVA execute matchTemplate process using DRP.

Item	range/values	note
image	width: 16 - 1920, even height: 16 - 1080, even channels: 1 - 4 bit-depth: 8bit size limit: FHD(1920x1080)	
templ	width: 8 - 1920, even height: 8 - 1080, even channels: same value as image bit-depth: same value as image size limit: image size	
method	CV_TM_SQDIFF CV_TM_SQDIFF_NORMED CV_TM_CCORR CV_TM_CCRR_NORMED	
mask	noArray()	Default value

4.12 warpAffine

4.12.1 outline

An affine transformation to an image.

```
void cv::warpAffine (InputArray src, OutputArray dst, InputArray M, Size dsize,
                     int flags = INTER_LINEAR, int borderMode = BORDER_CONSTARNT
                     const Scalar & borderValue = Scalar() )
```

Parameter	required/optional	description
src	required	Input image
dst	required	Destination image
M	required	Transformation matrix (2x3)
dsize	required	Size of destination image
flags (=INTER_LINEAR)	optional	One of the interpolation algorithm methods (see InterpolationFlags in OpenCV document)
borderMode (=BORDER_CONSTARNT)	optional	Pixel extrapolation method (see BorderTypes in OpenCV document)
borderValue (=Scalar())	optional	Border value

Note: For more information, refer to the OpenCV Document.

4.12.2 Conditions for using DRP

If the following conditions apply, OpenCVA execute warpAffine process using DRP.

Item	range/values	note
src	width: 16 - 3840, even height: 16 - 2160, even channels: 1 - 4 bit-depth: 8bit size limit: 4K(3840x2160)	
M	CV_32F	
dsize	width: 16 - 3840, even height: 16 - 2160, even size limit: 4K(3840x2160)	
flags	INTER_LINEAR	default value
borderMode	BORDER_CONSTARNT	default value

4.13 warpPerspective

4.13.1 outline

A perspective transformation to an image.

```
void cv::warpPerspective (InputArray src, OutputArray dst, InputArray M, Size dsize,
                          int flags = INTER_LINEAR, int borderMode = BORDER_CONSTARNT
                           const Scalar & borderValue = Scalar() )
```

Parameter	required/optional	description
src	required	Input image
dst	required	Destination image
M	required	Transformation matrix (3x3)
dsize	required	Size of destination image
flags (=INTER_LINEAR)	optional	One of the interpolation algorithm methods (see InterpolationFlags in OpenCV document)
borderMode (=BORDER_CONSTARNT)	optional	Pixel extrapolation method (see BorderTypes in OpenCV document)
borderValue (=Scalar())	optional	Border value

Note: For more information, refer to the OpenCV Document.

4.13.2 Conditions for using DRP

If the following conditions apply, OpenCVA execute warpPerspective process using DRP.

Item	range/values	note
src	width: 16 - 3840, even height: 16 - 2160, even channels: 1 - 4 bit-depth: 8bit size limit: 4K(3840x2160)	
M	CV_32F	
dsize	width: 16 - 3840, even height: 16 - 2160, even size limit: 4K(3840x2160)	
flags	INTER_LINEAR	default value
borderMode	BORDER_CONSTARNT	default value

4.14 pyrDown

4.14.1 outline

Blurs an image and downsamples it.

```
void cv::pyrDown (InputArray src, OutputArray dst, const Size & dstsize = Size(),
                  int borderType = BORDER_DEFAULT)
```

Parameter	required/optional	description
src	required	Input image
dst	required	Destination image
dstsize(=Size())	optional	Destination image size
intborderType (=BORDER_DEFAULT)	optional	Pixel extrapolation method

Note: For more information, refer to the OpenCV Document.

4.14.2 Conditions for using DRP

If the following conditions apply, OpenCVA execute pyrDown process using DRP.

Item	range/values	note
src	width: 16 - 3840, even height: 16 - 2160, even channels: 1 - 4 bit-depth: 8bit size limit: 4K(3840x2160)	
dstsize	Size() or Size(width/2, height/2)	
borderType	BORDER_DEFAULT	default value

4.15 pyrUp

4.15.1 outline

Blurs an image and upsamples it.

```
void cv::pyrUp (InputArray src, OutputArray dst, const Size & dstsize = Size(),
                int borderType = BORDER_DEFAULT)
```

Parameter	required/optional	description
src	required	Input image
dst	required	Destination image
dstsize(=Size())	optional	Destination image size
intborderType (=BORDER_DEFAULT)	optional	Pixel extrapolation method

Note: For more information, refer to the OpenCV Document.

4.15.2 Conditions for using DRP

If the following conditions apply, OpenCVA execute pyrUp process using DRP.

Item	range/values	note
src	width: 8 - 1920, even height: 8 - 1080, even channels: 1 - 4 bit-depth: 8bit size limit: FHD(1920x1080)	
dstsize	Size() or Size(width*2, height*2)	
borderType	BORDER_DEFAULT	default value

5.API functions to control OpenCVA

This chapter describes API functions to control OpenCVA.

5.1 OCA_Activate

[Summary] Disable or enable DRP used for OpenCV process.

[Function name] OCA_Activate

[Calling format] int OCA_Activate (unsigned long* OCA_list);

[Arguments] OCA_list: OpenCVA function activation table.

[Return] 0 : OK
-1: Error

[Feature] Disable or enable DRP used for OpenCV, for each of OpenCV function. By default, all are enabled. If disabled, the OpenCV function execute by CPU.

The argument of the API function is the array variable OCA_list[]. See the following table for OCA_list[] index and OpenCV functions.

index of OCA_list[]	**OpenCV function**
0	resize
2	cvtColor, cvtColorTwoPlane
4	GaussianBlur
5	dilate, morphologyEX
6	erode, morphologyEX
7	Filter2D
8	Sobel
9	adaptiveThreshold
10	matchTemplate
11	warpAffine
12	pyrDown
13	perUp
14	warpPerspective
Others	(unused)

note: OCA_list[] table type and size is “unsigned long OCA_list[16]”.

Setting the OCA_list[index] to 1 and then executing *OCA_Activate()*, then the corresponding DRP is enabled. If 0 is set, it is desabled.
Values other than 0 and 1 are ignored.

[Sample] The following is a sample of how to disable DRP in the OpenCV’s Sobel function

```
unsigned long OCA_list[16];

for(int i=0; i<16; i++)OCA_list[i]=2;

/* Disable DRP(Sobel) */

OCA_list[8] = 0; //Disable

OCA_Activate( &OCA_List[0] );
```

5.2 OCA_ConflictNotification

[Summary] Sets the behavior when DRP conflicts occur (exception error occurs or not).

[Function name] OCA_ConflictNotification

[Calling format] void OCA_ConflictNotification (int oca_conflict);

[Arguments] oca_conflict 0: When DRP conflicts, exception error occurs.
Not 0: When DRP conflicts, exception error does not occur, and the OpenCV function is executed by CPU.

[Feature] Sets the behavior when DRP conflicts occur (exception error occurs or not). By default, it causes an exception error do to DRP conflict.
e.g., execute OpenCV using DRP while the AI process is using the DRP.
The error code for exception errors is -501.

[Sample] The following is a sample to handle exception error.

```
/* Exception error enable */

OCA_ConflictNotification( 0 );

try{

    cv::GaussianBlur(src, dst, {7,7},0,0);

} catch(cv::Exception& e) {

    if( e == -501 ){/* exception error handling */}

}
```

6.DRP conflict

This chapter describes DRP conflict.

6.1 About DRP conflict.

OpenCVA use “Dynamically Reconfigurable Processor” (=DRP). And the AI processing functions use the same DRP. There is only one DRP used by these functions on a device. And the function occupies the DRP while it is executing.

If the OpenCV use the DRP but is unable to do so because the DRP is already being used by another function, this is called “DRP conflict.”

OpenCVA cause DRP conflicts in the following 2cases:
case1. Case of start OpenCV using DRP, while the AI process function uses DRP in parallel.
case2. Case of start OpenCV using DRP, while other OpenCV is using DRP in another process (or thread).

If neither of the 2cases matches, then the DRP does not conflict. i.e.,
1. AI processing and OpenCV using DRP are executed sequentially.
2. OpenCV using DRP is executed by single process/thread.

6.2 What happened if there was a conflict, and how to handle it

If the OpenCVA occurred DRP conflict, it raises an exception error or execute OpenCV function by CPU instead of the DRP. (selected by the *OCA_ConflictNotification()*).
If the DRP conflict occur exception error, the user application must be able to handle the exception error, as in Chapter 5.2 sample.
If the OpenCV function executed by CPU, the user applications do not need to support anything, but performance of the OpenCV should be confirmed.

Version History

Revision History	RZ/V2L OpenCV Accelerator User's Manual

Rev.	Date	Description	
		Page	Summary
1.00	Jan.18.24	-	First Edition issued
1.10	Feb.5.24	7	Deleted the text of RZ/V2L ISP support package.

RZ/V2L OpenCV Accelerator User’s Manual

Publication Date: Rev.1.10 5. Feb. 2024

Published by: Renesas Electronics Corporation

RZ/V2L Group