RZ/A2M Group
DRP Library User's Manual and Functional Design Specifications
First Edition (Rev. 1.00)

ResizeBilinearCrop

September 24, 2019		
Renesas Electronics		
Approved by	Examined by	Author

User's Manual

RZ/A2M Group

DRP Custom Library User's Manual

ResizeBilinearCrop

Renesas Electronics
www.renesas.com

Rev.1.00 Sept. 2019

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. 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 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. Unused pins should be handled as described under Handling of Unused Pins in the manual.

2. Processing at Power-on

   The state of the product is undefined at the moment 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 moment 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 moment 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 moment when power is supplied until the power reaches the level at which resetting has been specified.

3. Prohibition of Access to Reserved Addresses

   Access to reserved addresses is prohibited.

   — The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.

4. Clock Signals

   After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has 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. Moreover, 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.

5. Differences between Products

   Before changing from one product to another, i.e. to a product with a different part number, confirm that the change will not lead to problems.

   — The characteristics of Microprocessing unit or Microcontroller unit products in the same group but having a different part number may differ in terms of the 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 manual is intended to provide the user with an understanding of the functions of the DRP library and how to utilize them. It is aimed at users designing application systems making use of the DRP library. In order to use this manual, you will need a basic knowledge of programming languages and microprocessors.

> 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.

> 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.

Contents

1. Introduction

1.1 Summary

This manual describes the functions and usage of the DRP library, which run on the dynamically reconfigurable processor (DRP) of RZ/A2M Group Microprocessors.

The DRP can perform various functions according to user's setting. In this document, the function performed by DRP is called "circuit", and the data representing circuit information is called "configuration data". Writing of the circuit to DRP can be performed by loading the configuration data using DRP Driver[*1]. DRP Library is a collection of configuration data with various functions, mainly image processing.

Note 1. For details of DRP Driver, refer to "RZ/A2M Group DRP Driver User's Manual (R01US0355)".

1.2 Functions

The functions of the configuration data contained in the DRP library are listed below.

Table 1.1 DRP Library Functions

Category	Function Name	Outline	Page
Image conversion	ResizeBilinearCrop	Resizes the image (bilinear interpolation, scale factor: any), specified by a source crop window	11

2. Operation Conditions

The DRP library operates under the conditions listed below.

Table 2.1 Operation Conditions

Item	Description
Microprocessor	RZ/A2M Group Microprocessors*1 • R7S921051VCBG • R7S921052VCBG • R7S921053VCBG

Note 1. The DRP library operates on RZ/A2M Group Microprocessors equipped with a DRP function module. It will not operate on RZ/A2M Group Microprocessors without a DRP function module.

This library was confirmed to operate in the following development environment:

Renesas e^2 studio 7.5.0

The following toolchain is compatible:

GCC ARM Embedded Toolchain 6-2017-q2-update

3. File Structure

Figure 3.1. shows the file structure of configuration data and header files in the DRP library.

```
r_drp_resize_bilinear_crop                    ResizeBilinearCrop
   +asm
   |   +  r_drp_resize_bilinear_crop_t6.asm
   |
   +dat
   |  +  r_drp_resize_bilinear_crop_t6.dat
   |
   +inc
   |  +  r_drp_resize_bilinear_crop_x_t6.h
   |  +  r_drp_resize_bilinear_crop.h
   + doc
      + <this document>
```

Figure 3.1 File Structure

4. DRP Library Reference

4.1 How to Read the DRP Library Reference

In this section the specifications of the configuration data contained in the DRP library are presented in the format shown below.

Function name*1

Function outline

Configuration data file	The name of the configuration data file. Use the DRP Driver's R_DK2_Load() function to load the data in the DRP.
Supported version	Lists the version of the configuration data that operates under present specification. Use the DRP Driver's R_DK2_GetInfo() function to get the version.
Configuration data size (byte)	Lists the size of the configuration data. Lists all versions, if there are different versions.
Header file	The name of the header file for using the configuration data. Use #include "header file" to include the file.
Parameter	Lists the parameters required by the circuit. Parameters are passed from the CPU to the DRP by means of the DRP driver's R_DK2_Start() function. Parameters are defined as a structure within the header file. Before running the circuit, set the parameters on the CPU side. The data type defined in stdint.h is used. Also, the area where parameters are stored and the area indicated by parameters representing addresses such as 'src' and 'dst' must be located in physical memory. *2
I/O details	Lists the details of the data specified by the parameters. Unless otherwise indicated, the same address may be specified for the input buffer address and output buffer address.
Number of tiles	The number of tiles used by the circuit. The DRP has 6 tiles. The DRP Driver's R_DK2_Load() function is used to assign circuits to tiles.
Segmented processing	Indicates that the function can be processed in parallel by multiple circuits. In parallel processing, the input image is divided up in the vertical direction and processed accordingly. The segmented processing can be executed by utilizing the 6 tiles of DRP and loading multiple configuration data of 3 tiles or less. For details on loading multiple configuration data of 3 tiles or less into DRP, see the explanation of R_DK2_Load () function in "RZ/A2M Group DRP Driver User's Manual".

Example: A case where the input image is divided into three portions in the vertical direction

Description	Describes the specifications of the configuration data.
Note	Additional notes appear here.

Note 1. The function name of configuration data is a character string that can be obtained from the configuration data by using the DRP Driver's R_DK2_GetInfo() function.

Note 2. If the values of physical memory in the area of parameters and input/output data of the circuit are incorrect because the values are in the Cortex-A9 cache, etc., the circuit does not work properly. It must be necessary to clean the cache before calling the DRP driver's R_DK2_Start() function or to allocate the parameters and input/output data of circuit to a non-cached area.

For information on using the API functions of the DRP Driver, refer to "RZ/A2M Group DRP Driver User's Manual (R01US0355)".

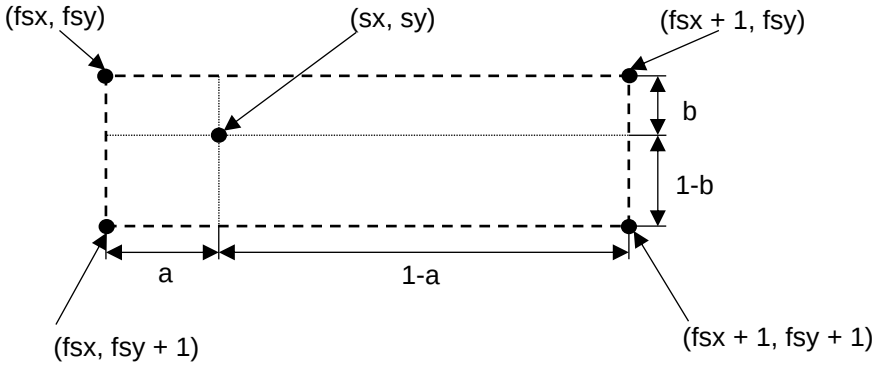
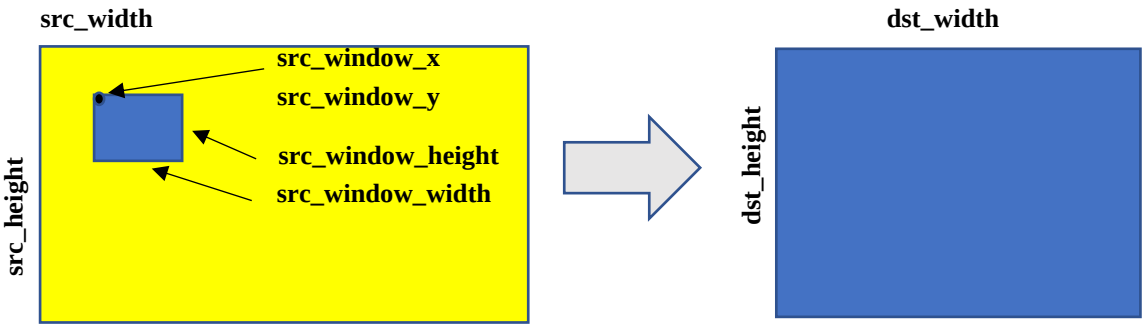
4.2 Image Conversion

4.2.1 ResizeBilinearCrop

ResizeBilinearCrop

Resizes the image in the cropping window
(bilinear interpolation, scale factor: any)

Configuration data file			r_drp_resize_bilinear_crop_t6.dat	
Supported version			0.91	
Configuration data size (byte)			553440	
Header file			r_drp_resize_bilinear_crop_x_t6.h (r_drp_resize_bilinear_crop.h)	
Parameter	Structure name			
	r_drp_resize_bilinear_crop_t			
	Member name	Type	Description	
	src	uint32_t	Input image address	
	dst	uint32_t	Output image address	
	src_width	uint16_t	Horizontal width of input image (pixels)	
	src_height	uint16_t	Vertical width of input image (pixels)	
	dst_width	uint16_t	Horizontal width of output image (pixels)	
	dst_height	uint16_t	Vertical width of output image (pixels)	
	src_window_x	uint16_t	X position o crop window	
	src_window_y	uint16_t	Y position of crop window	
	src_window_widh	uint16_t	Horizontal width of crop window	
	src_window_height	uint16_t	Vertical width of crop window	
I/O details	Input image	Address:	Specified by src. (Specify an address that differs from dst.)	
		Width (pixels):	Specified by src_width. (2to 1280)	
		Height (pixels):	Specified by src_height. (2 to 960)	
		WindowWidth(pixels):	Specified by src_window_width. (2to 1280)	
		WindowHeight(pixels):	Specified by src_window_height. (2 to 960)	
		WindowX(pixels):	Specified by src_window_x. (2 to 1280)	
		WindowY(pixels):	Specified by src_window_Y. (2 to 960)	
		Format:	8-bit grayscale (1 byte per pixel)	
		Data size:	(src_window_width) × (src_window_height) × 1 byte	
	Output image	Address:	Specified by dst. (Specify an address that differs from src.)	
		Width (pixels):	Specified by dst_width. (2 to 1280)	
		Height (pixels):	Specified by dst_height. (2 to 960)	
		Format:	8-bit grayscale (1 byte per pixel)	
		Data size:	(dst_width) × (dst_height) × 1 byte	
Number of tiles	6			
Segmented processing	Not supported			
Range check	User has to take care			

Description	This function enlarges or reduces the image at the address specified by src and outputs the result to the address specified by dst. This is done for the picture part specified by the cropping window. It is necessary to add or remove pixels when the image is enlarged or reduced, and this function uses bilinear method for this purpose. In the bilinear method, a grid of 2 × 2 pixels peripheral to the input image in the position corresponding to the target pixel of the output image is used and linear interpolation is applied. This function uses the following calculations for the bilinear method. Assuming that the coordinate (sx,sy) in the input image corresponds to the coordinate (dx,dy) of the output image, sx and sy are expressed by the following equations (explanation without cropping). $$sx = (dx + 0.5) \times src\_window\_width \div dst\_width - 0.5$$ $$sy = (dy + 0.5) \times src\_window\_height \div dst\_height - 0.5$$ Assuming that fsx=Floor(sx) and fsy=Floor(sy), the coordinates of the grid of 2 × 2 pixels peripheral to (sx,sy) are (fsx,fsy), (fsx+1,fsy), (fsx,fsy+1) and (fsx+1,fsy+1). Assuming that the brightness value at the coordinate (x,y) of the input image is src(x,y) and the brightness value at the coordinate (x,y) of the output image is dst(x,y), dst(dx,dy) is expressed by the following equation. $$dst(dx, dy) = (1 - b) \times (1 - a) \times src(fsx, fsy) + (1 - b) \times a \times src(fsx + 1, fsy)$$ $$+b \times (1 - a) \times src(fsx, fsy + 1) + b \times a \times src(fsx + 1, fsy + 1)$$ $$\text{However}, a = sx - fsx, b = sy - fsy$$  The processing performed by this function is equivalent to that of the OpenCV cv::resize function with specifying dst_width to the argument dsize.width, dst_height to dsize.height, and INTER_LINEAR to interpolation. The crop function is an add on. Reference URL: https://opencv.org/ 
Note	None

5. Using the DRP Library

To use this library, it is necessary to initialize the DRP, load configuration data, etc. Also, since the parameters are different for each configuration data, set the parameters based on the specification of the configuration data to be used. For application example of DRP library, refer to "RZ/A2M Group 2D Barcode Application Note (R01AN4503)".

Usage example:

#include “r_drp_resize_bilinear_crop_x.h”

…..

// variable declaration

```
int32_t ret_val;

static uint8_t drp_lib_id[R_DK2_TILE_NUM] = {0};
static volatile uint8_t drp_lib_status[R_DK2_TILE_NUM] = {DRP_NOT_FINISH};

static r_drp_resize_bilinear_crop_t param_resize_bilinear_crop __attribute__ ((section("Uncache_IRAM")));
```

…

```
static void cb_drp_finish(uint8_t id)
{
    uint32_t tile_no;
    /* Change the operation state of the DRP library notified by the argument to finish
*/
    for (tile_no = 0; tile_no < R_DK2_TILE_NUM; tile_no++)
    {
        if (drp_lib_id[tile_no] == id)
        {
            drp_lib_status[tile_no] = DRP_FINISH;
            break;
        }
    }
    return;
}

/***********************************************************************************
* End of function cb_drp_finish
***********************************************************************************/
```

…

// usage example

```
/*******************************/
/* Load DRP Library            */
/*        +------------------+ */
/* tile 0 |                  | */
/*        +                  + */
/* tile 1 |                  | */
/*        +                  + */
/* tile 2 |                  | */
/*        + Resize Bilinear  + */
/* tile 3 |      Crop        | */
/*        +                  + */
/* tile 4 |                  | */
/*        +                  + */
/* tile 5 |                  | */
/*        +------------------+ */
/*******************************/

ret_val = R_DK2_Load(&g_drp_lib_resize_bilinear_crop_t6[0],R_DK2_TILE_0,
                     R_DK2_TILE_PATTERN_6, NULL, &cb_drp_finish, &drp_lib_id[0]);
DRP_DRV_ASSERT(ret_val);

/************************/
/* Activate DRP Library */
/************************/
ret_val = R_DK2_Activate(drp_lib_id[TILE_0], 0);
DRP_DRV_ASSERT(ret_val);

/*************************************/
/* Set R_DK2_Start function parameters */
/*************************************/

param_resize_bilinear_crop.src              = (uint32_t)&…; /* Addr of input image.*/
param_resize_bilinear_crop.src_width        = 640; /* width of input image.        */
param_resize_bilinear_crop.src_height       = 480; /* height of input image.       */
param_resize_bilinear_crop.src_window_x     = 10 ; /* x position of crop window.   */
param_resize_bilinear_crop.src_window_y     = 40 ; /* y position of crop window.   */
param_resize_bilinear_crop.src_window_width  = 50 ; /* width of crop area.         */
param_resize_bilinear_crop.src_window_height = 60 ; /* height of crop area         */

param_resize_bilinear_crop.dst              = (uint32_t)&…; /* Addr of output image.*/
param_resize_bilinear_crop.dst_width        = 100; /* target width of output image. */
param_resize_bilinear_crop.dst_height       = 120; /* target height of output image.*/


/* Initialize variables to be used in termination judgment of the DRP library */
drp_lib_status[TILE_0] = DRP_NOT_FINISH;

/*********************/
* Start DRP Library */
/*********************/
ret_val = R_DK2_Start(drp_lib_id[TILE_0], (void *)&param_resize_bilinear_crop,
                      sizeof(r_drp_resize_bilinear_crop_t));
DRP_DRV_ASSERT(ret_val);
```

```
/*************************************/
/* Wait until DRP processing is finish */
/*************************************/
while (drp_lib_status[TILE_0] == DRP_NOT_FINISH);
```

…..

// clean up

```
/**********************/
/* Unload DRP Library */
/**********************/
ret_val = R_DK2_Unload(drp_lib_id[TILE_0], &drp_lib_id[TILE_0]);
DRP_DRV_ASSERT(ret_val);
```

6. Reference Documents

User's Manual: Hardware

RZ/A2M Group User's Manual: Hardware (R01UH0746)

(Download the latest version of the update or news from the Renesas Electronics website.)

User's Manual: Software

RZ/A2M Group DRP Driver User's Manual (R01US0355)

(Download the latest version of the update or news from the Renesas Electronics website.)

RZ/A2M Group 2D Barcode Sample Program Application Note (R01AN4503)

(Download the latest version of the update or news from the Renesas Electronics website.)

User's Manual: Development environment

For the Renesas Electronics integrated development environment (e^2 studio), visit the Renesas Electronics website to download the latest version.

Technical Update/Technical News

(Download the latest version of the update or news from the Renesas Electronics website.)

Revision History	RZ/A2M Group DRP Library User's Manual

Rev.	Date	Description	
		Page	Summary
1.00	Sept. 24, 2019	—	First Edition issued

RZ/A2M Group DRP Custom Library User’s Manual

Publication Date: Rev.1.00 Sept. 24, 2019

Published by: Renesas Electronics Corporation

RZ/A2M Group

Renesas Electronics Corporation