Freescale Semiconductor
Application Note

AN2886/D
Rev. 0, 11/2004

Operating Principle of Resize in the eMMA Pre-Processor

MC9328MX21

by: Cliff Wong

1 Abstract

This document describes the operating principle of the Resize unit in the MC9328MX21 processor's enhanced Multimedia Accelerator (eMMA) pre-processor (PRP) module. The averaging and bilinear algorithms are discussed and example register setting are included.

The eMMA PRP is able to do downscaling only. Upscaling is not supported.

Table 1 on page 2 provides the definition of terms used throughout this document.

Contents

Table 1. Definition of Terms

Registers	Fields	Descriptions
PRP_CH1_RZ_HORI_COEF1 PRP_CH1_RZ_HORI_COEF2 PRP_CH1_RZ_VERT_COEF1 PRP_CH1_RZ_VERT_COEF2 PRP_CH2_RZ_HORI_COEF1 PRP_CH2_RZ_HORI_COEF2 PRP_CH2_RZ_VERT_COEF1 PRP_CH2_RZ_VERT_COEF2	Coefficients (HC, VC)	Filter coefficients. There can be up to 20 coefficients for each filter setting. The length of the filter is variable and set by the table length. The sum of all coefficients must be 8.
PRP_CH1_RZ_HORI_VALID PRP_CH1_RZ_VERT_VALID PRP_CH2_RZ_HORI_VALID PRP_CH2_RZ_VERT_VALID	Table Length (HORI_TBL_LEN, VERT_TBL_LEN)	Length of the coefficient table. This also equals to the window size of the filter. For resize ratios of *m*:*n*, the table length should be set to *m*, which means for every *m* pixel input, *n* output pixels are generated. *n* is determined by output valid.
	Output Valid (HOV, VOV)	Valid output pixel position. Indicates which pixel of the filtering window is used as output. For Example: Given: Table length = 5, Output valid = set bit 1, bit 4: and filter window size = 5, 1. Accumulator starts with pixel 0 2. Then at pixel 1 3. Generates output (since valid = 1) 4. Accumulator resets 5. Filter re-starts at pixel 2 6. Then at pixel 3, and pixel 4 7. Generates output (since valid = 1) 8. Accumulator resets, filter window finished

2 Algorithms

Depending on the use case, averaging or bilinear algorithms and their derivatives are employed. The same algorithms apply to both channel 1 and channel 2—that is, horizontal and vertical. Channel 1 and channel 2 can operate in either cascade mode or in separate mode. Horizontal and vertical operations are independent of each other and they can operate in different modes. The examples shown Table 2 are only for horizontal resize.

Table 2. Horizontal Resize Sample Ratios

Ratio *m*:*n*	Ratio type	Algorithm
ratio = 1	Integer	Averaging
1 < ratio < 2	Fractional	Bilinear
2 ≤ ratio ≤ 8	Integer	Averaging
2 < ratio < 8	Fractional	Averaging mix mode
9 ≤ ratio ≤ 20	Integer	Averaging skip mode

2.1 Averaging Algorithms

This is the simplest mode of operation, an averaging filter is moving along the input vector and the convolution sum is taken as the output vector. The filter window is moving without overlapping. That means each of the input pixels is used only once. One output pixel is generated per every window. The filter length is variable and depends on the resize ratio.

2.1.1 Ratio = 1:1

For a ratio of 1:1, an averaging algorithm is recommended. The Resize unit simply copies data from input to output directly. This operation is illustrated in Figure 1.

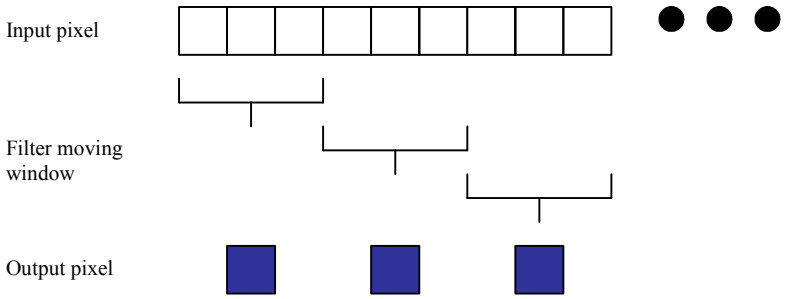

Figure 1. Moving Window Without Overlapping Pixels

2.1.2 Ratio Between 2:1 and 8:1

For any integer ratio (m:n, where n=1) between 2:1 and 8:1, an averaging algorithm is recommended. The output pixel is the weighted sum of a group of input pixels. The group (window) size is equal to the resize ratio m.

The general equation is shown in Equation 1.

$$out[j] = \frac{1}{8}\sum_{i=0}^{m} in[i] \times w[i] \qquad \textit{Eqn. 1}$$

where m is the resize ratio, w[i] is the weighting coefficient.

2.1.2.1 Example: Ratio = 3:1

Figure 2 illustrates an averaging algorithm when using a ratio of 3:1. This section discusses the weighting coefficients and demonstrates the equation to determine the register settings for this ratio.

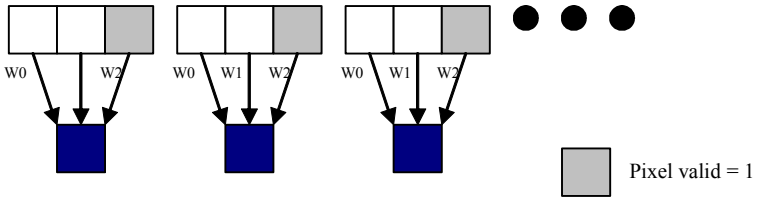

Figure 2. Using an Averaging Algorithm with a 3:1 Ratio

Weighting coefficients *w*[i] are stored in the following registers:

PRP_CH1_RZ_HORI_COEF1
PRP_CH1_RZ_HORI_COEF2
PRP_CH1_RZ_VERT_COEF1
PRP_CH1_RZ_VERT_COEF2

The weighting coefficients *w*[i] are referred to as HC[i] and VC[i] for horizontal resize and vertical resize respectively.

In our scenario of a ratio of 3:1, HC[i] (or VC[i]) must be from 1 to 8, and not greater than 8. Coefficients are 3 bits long, so number 7 is not used and a 3b'111 is employed to represent number 8. The sum of all coefficients must be equal to 8, such that the output pixel is in the correct range. Equation 2 demonstrates this scenario.

$$\sum_{i=0}^{m} w[i] = 8$$

Eqn. 2

where *w*[i] corresponds to HC[i] (or VC[i]).

The Table Length field is set to the group (window) size and the Pixel Valid field is set to the last pixel of each group.

Therefore, for this example the register settings will be as follows:

HORI_TBL_LEN = 3
HOV = set bit 2
PRP_CH1_RZ_HORI_VALID = 0x03000004

2.2 Averaging Mix Mode

In mix mode operation, each window can be further divided into several component windows. The component window's sizes can be different from each other. For example, a component window size of 3 and 2 together forms a window size of 5.

2.2.1 Fractional Ratios Between 2:1 and 8:1

For fractional ratio between 2:1 and 8:1, an averaging algorithm is used in mix mode.

The fractional ratio *m*:*n* is broken down into several component ratios *m*[*k*]:*n*[*k*], each of which is an integer ratio. The resultant ratio is the combined value of the component ratios as shown in Equation 3.

where *k* is the number of component ratios.

$$m : n = \sum_{k} m[k] : \sum_{k} n[k]$$

Eqn. 3

The window size is still equal to *m*, as shown in Equation 4.

$$m = \sum_{k} m[k]$$

Eqn. 4

To preserve the correct range of the output pixels, for each component group the sum of weighting coefficients must be equal to 8 as shown in Equation 5.

$$\sum_{j} w_k[j] = 8$$

Eqn. 5

2.2.2 Example: Ratio = 5:2

In this section an example using an averaging algorithm in mix mode with a ratio of 5:2 is given and the equations to determine the register settings are provided.

Using a ratio of 5:2, the ratio can be broken down into 2 component ratios, 3:1 and 2:1. This breakdown is illustrated in Figure 3.

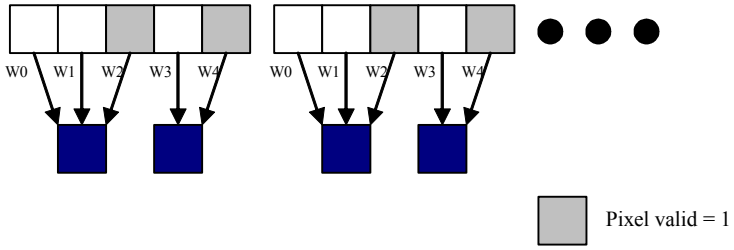

Table 3. Averaging a Ratio of 5:2

The resultant ratio = (3+2):(1+1) = 5:2 and is demonstrated in Equation 6.

$$\sum_{j=0}^{2} w[j] = 8 \quad , \quad \sum_{j=3}^{4} w[j] = 8$$

Eqn. 6

Therefore, for this example the register settings will be as follows:

HORI_TBL_LEN = 5

HOV = set bit 2 and 4

PRP_CH1_RZ_HORI_VALID = 0x05000014

2.2.3 Example: Ratio = 8:3

In this section an example using an averaging algorithm in mix mode with a ratio of 8:3 is given and the equations to determine the register settings are provided.

For a ratio of 8:3, the ratio can be broken down into 3 component ratios, 3:1, 2:1, and 3:1. This breakdown is illustrated in Figure 3 on page 6.

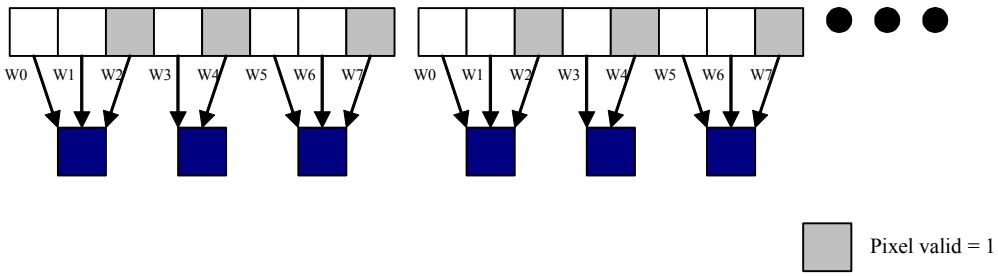

Figure 3. Averaging a Ratio of 8:3

The resultant ratio = (3+2+3):(1+1+1) = 8:3 and is demonstrated in Equation 7.

$$\sum_{i=0}^{2} w[i] = 8 \quad , \quad \sum_{i=3}^{4} w[i] = 8 \quad , \quad \sum_{i=5}^{7} w[i] = 8 \qquad \textit{Eqn. 7}$$

Therefore, for this example the register settings will be as follows:

HORI_TBL_LEN = 8
HOV = set bit 2, 4, 7
PRP_CH1_RZ_HORI_VALID = 0x08000094

2.3 Averaging Skip Mode

In skip mode operation, the window size is larger than 8. Not all of the pixels within a window are taken into account to generate the output. To preserve a sum of 8, some of the coefficients are set to 0 so that the corresponding pixel is skipped by the filter.

2.3.1 Ratios Between 9:1 and 20:1

For any integer resize ratios between 9:1 and 20:1, averaging skip mode is recommended.

The equation used for this operation is the same as the general one and repeated here in Equation 8.

$$out[j] = \frac{1}{8}\sum_{i=0}^{m} in[i] \times w[i] \qquad \textit{Eqn. 8}$$

It is important to note that when averaging ratios between 9:1 and 20:1 that some of the $w[i]$ coefficients are set to zero, meaning that the corresponding input pixel will be skipped.

To preserve the data in correct range, the sum of all coefficients must be equal to 8 no matter how large m is. The equation for this scenario is demonstrated in Equation 9.

$$\sum_{i=0}^{m} w[i] = 8 \qquad \textit{Eqn. 9}$$

2.3.2 Example: Ratio = 10:1

In this section an example using an averaging algorithm in skip mode with a ratio of 10:1 is given and the equations to determine the register settings are provided. Figure 4 illustrates this example.

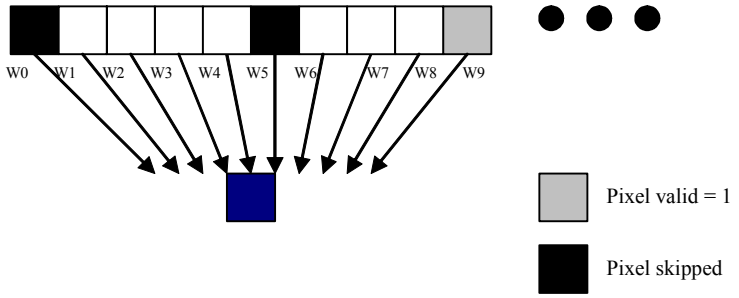

Figure 4. Averaging Skip Mode with Ratios Between 9:1 and 20:1

The equation complete with numeric values is shown in Equation 10.

$$\sum_{i=0}^{9} w[i] = 8 \qquad \textit{Eqn. 10}$$

Because more than 8 coefficients are used, one (or some) of the weighting coefficient are set to 0. In this case,

$w[0] = 0$, $w[5] = 0$

In other words, pixel 0 and pixel 5 are skipped.

Therefore, for this example the register settings will be as follows:

HORI_TBL_LEN = 10

HOV = set bit 9

PRP_CH1_RZ_HORI_VALID = 0x0A000200

2.3.3 Example: Ratio = 20:1

The maximum length of an averaging filter is 20, which means the maximum downscaling ratio is 20:1, where some pixels must be skipped. The user can select any pixel to be skipped. There are many solutions for any particular resize ratio. To minimize the image artifacts, the skipping position should be evenly distributed along the filter as shown in Figure 5.

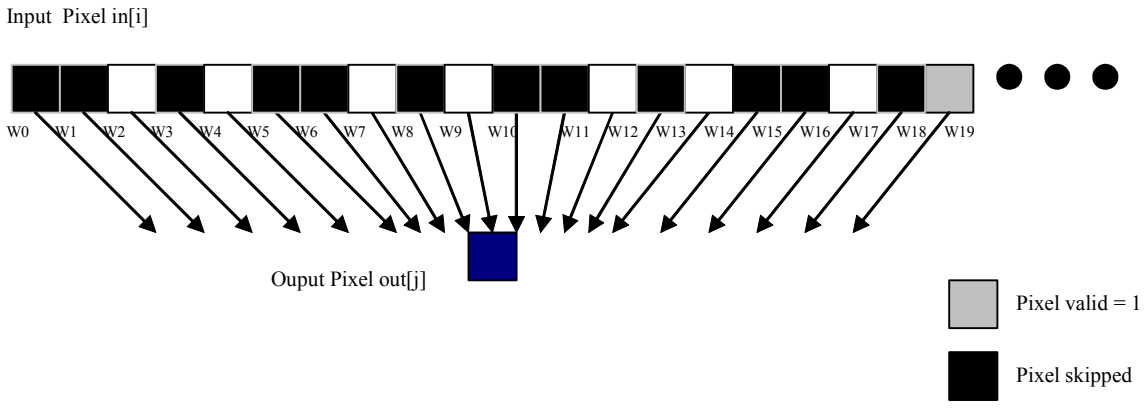

Figure 5. Ration of 20:1 Showing Even Distribution Along the Filter

Equation 11demonstrates the ratio of 20:1 using an averaging algorithm in skip mode.

$$out[j] = \frac{1}{8}\sum_{i=0}^{19} in[i] \times w[i]$$

Eqn. 11

where $w[0, 1, 3, 5, 6, 8, 10, 11, 13, 15, 16, 18] = 0$.

In this scenario, the maximum number of coefficients used is 8, so any “excess” coefficients must be set to zero. A zero coefficient has the same effect as skipping the corresponding pixel.

Therefore, for this example the register settings will be as follows:

HORI_TBL_LEN = 20

HOV = set bit 19

PRP_CH1_RZ_HORI_VALID = 0x14080000

2.4 Bilinear Algorithms

A bilinear algorithm is recommended for fractional resize ratios between 1:1 and 2:1. For resize ratios between 2:1 and 8:1, a bilinear algorithm can also be used by specifying the proper coefficients and pixel valid fields. However, the quality will not be as good as when using an averaging algorithm.

In a bilinear algorithm, a filter window is moving along the input vector and the convolution sum is used to generate the output vector. The window size is always equal to 2 which means every 2 adjacent input pixels are processed to generate 1 resultant pixel. The resultant pixel is written to output if the corresponding VALID bit is on, otherwise the resultant pixel is ignored.

The window is moving in pixel steps—that is, after each completed operation the window is shifted by only 1 pixel. As a result, the window is said to be overlapping. This is the primary difference between the bilinear and averaging algorithms.

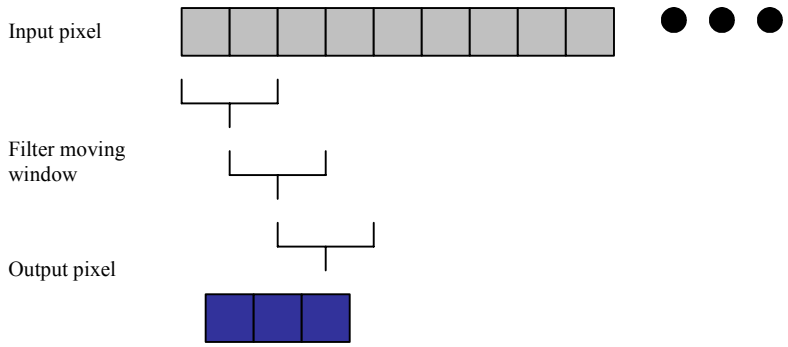

Figure 6. Moving Window with Overlapping Pixels

The general equation of the bilinear algorithm filter is demonstrated in Equation 12.

$$out[i] = \frac{1}{8} \times (w[i] \times in[i] + (8 - w[i]) \times in[i+1]) \,\&\, valid[i]$$

Eqn. 12

In this scenario, $w[i]$ is called the left coefficient, while (8-$w[i]$) is called the right coefficient. The right coefficient is automatically generated by hardware.

The resultant pixel is logically ANDed with the pixel valid—that is, if *valid*[*i*] = 1, the pixel is written to output, otherwise the pixel is ignored.

2.4.1 Example 1: Ratio = 5:3

In this section an example using a bilinear algorithm with a ratio of 5:3 is given and the equations to determine the register settings are provided.

The recommended settings are:

(HC[*i*], HOV[*i*]) = (5,1), (0,1), (*X*,0), (3,1), (*X*,0), where *X* means not in use.

The details of the filter operation are illustrated in Figure 7on page 9.

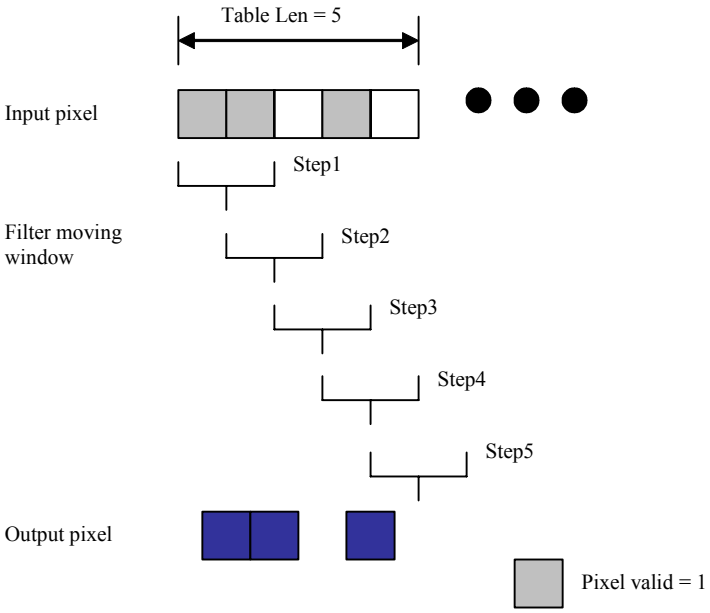

Figure 7. Bilinear Algorithm with a Ratio of 5:3

The steps shown in Figure 7 are demonstrated in the following equations:

Step1:

$$out[0] = \frac{1}{8} \times (5 \times in[0] + (8-5) \times in[1]) \,\&\, 1 \qquad \textit{Eqn. 13}$$

Step 2:

$$out[1] = \frac{1}{8} \times (0 \times in[1] + (8-0) \times in[2]) \,\&\, 1 \qquad \textit{Eqn. 14}$$

Step 3:

$$X = \frac{1}{8} \times (X \times in[2] + (8-X) \times in[3]) \,\&\, 0 \qquad \textit{Eqn. 15}$$

Step 4:

$$out[2] = \frac{1}{8} \times (3 \times in[3] + (8-3) \times in[4]) \,\&\, 1$$

Eqn. 16

Step 5:

$$X = \frac{1}{8} \times (X \times in[4] + (8-X) \times in[5]) \,\&\, 0$$

Eqn. 17

For steps 3 and 5, because the corresponding valid[i] is 0, the output pixel is not taken. Thus the coefficients are also not important and can be set to any X. The index of output pixel array is also not advanced.

2.4.2 Example: Ratio = 4:3

In this section an example using a bilinear algorithm with a ratio of 4:3 is given and the equations to determine the register settings are provided.

The recommended settings are:

(HC[i], HOV[i]) = (6,1), (4,1), (1,1), (X,0), where X means not in use.

The details of the filter operation are illustrated in Figure 8.

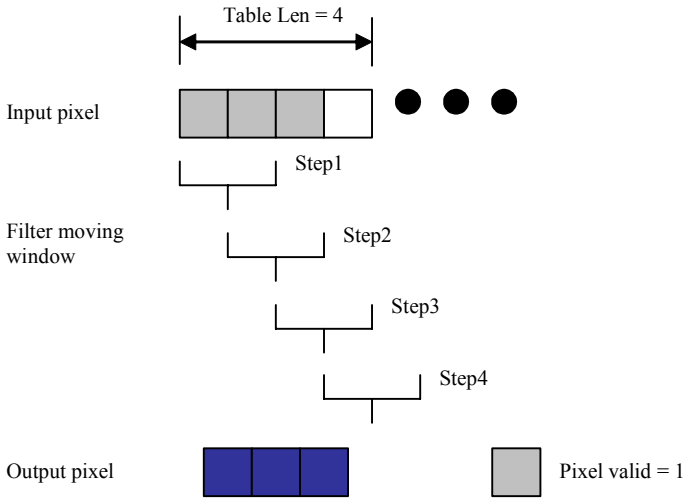

Figure 8. Bilinear Algorithm with a Ratio = 4:3

The steps shown in Figure 8 are demonstrated in the following equations.

Step 1:

$$out[0] = \frac{1}{8} \times (6 \times in[0] + (8-6) \times in[1]) \,\&\, 1 \qquad \textit{Eqn. 18}$$

Step 2:

$$out[1] = \frac{1}{8} \times (4 \times in[1] + (8-4) \times in[2]) \,\&\, 1 \qquad \textit{Eqn. 19}$$

Step 3:

$$out[2] = \frac{1}{8} \times (1 \times in[2] + (8-1) \times in[3]) \,\&\, 1 \qquad \textit{Eqn. 20}$$

Step 4:

$$X = \frac{1}{8} \times (X \times in[3] + (8-X) \times in[4] \,\&\, 0 \qquad \textit{Eqn. 21}$$

3 Coefficient Tables

There are hundreds of resize ratios supported by eMMA's Pre-Processor. The resize ratio coefficients are not limited to the examples given in this application note. Table 4 provides typical example settings for some of the many solutions.

Table 4. Sample Resize Ratios

Ratio *m*:*n*	Ratio type	Algorithm
ratio = 1	Integer	Averaging
1 < ratio < 2	Fractional	Bilinear
2 ≤ ratio ≤ 8	Integer	Averaging
2 < ratio < 8	Fractional	Averaging mix mode
9 ≤ ratio ≤ 20	Integer	Averaging skip mode

3.1 Integer Ratio Between 1:1 and 8:1

There are only 8 integer ratio cases between 1:1 and 8:1, all of these are listed Table 5. The settings apply to both Channel 1 and Channel 2, the horizontal and vertical Resize units. Examples listed are for Channel 1 horizontal resize.

Table 5. Averaging Algorithm

Ratio *m*:*n*	PRP_CH1_RZ_HORI_COEF1	PRP_CH1_RZ_HORI_COEF2	PRP_CH1_RZ_HORI_VALID
1	0x00000008	0x00000000	0x01000001
2	0x00000024	0x00000000	0x02000002
3	0x0000009A	0x00000000	0x03000004
4	0x00000492	0x00000000	0x04000008
5	0x00001491	0x00000000	0x05000010
6	0x00012251	0x00000000	0x06000020
7	0x0009124A	0x00000000	0x07000040
8	0x00491239	0x00000000	0x08000080

3.2 Fractional Ratio Between 1 and 2

There are over 100 cases where fractional ratios between 1 and 2 are possible, only some of the most typical cases are listed Table 6.

Table 6. Bilinear Algorithm

Ratio *m*:*n*	PRP_CH1_RZ_HORI_COEF1	PRP_CH1_RZ_HORI_COEF2	PRP_CH1_RZ_HORI_VALID
3:2	0x00000016	0x00000000	0x83000003
4:3	0x00000066	0x00000000	0x84000007
5:3	0x00000605	0x00000000	0x8500000B
5:4	0x000002EE	0x00000000	0x8500000F
6:5	0x00001536	0x00000000	0x8600001F
7:4	0x00030385	0x00000000	0x8700002D
7:5	0x0002501E	0x00000000	0x87000037
7:6	0x00012776	0x00000000	0x8700003F
8:5	0x0016080E	0x00000000	0x8800006B
8:7	0x000A3976	0x00000000	0x8800007F
9:2	0x00004004	0x00000000	0x89000011
9:4	0x00204104	0x00000000	0x89000055
9:5	0x00C20185	0x00000000	0x890000AD
9:7	0x005E00AE	0x00000000	0x890000EF
9:8	0x00534BB7	0x00000000	0x890000FF

3.3 Fractional Ratio Between 2 and 8

There are only 4 cases when a fractional ratio between 2 and 8 occur, all are listed Table 7.

Table 7. Averaging Mix Mode Algorithm

Ratio *m*:*n*	PRP_CH1_RZ_HORI_COEF1	PRP_CH1_RZ_HORI_COEF2	PRP_CH1_RZ_HORI_VALID
5 :2	0x000048A2	0x00000000	0x05000014
7 :2	0x00142492	0x00000000	0x07000048
7 :3	0x002428A4	0x00000000	0x07000052
8 :3	0x012244A2	0x00000000	0x080000A4

3.4 Integer Ratio Between 9 and 20

There are 12 cases when integer ratios occur between 9 and 20, all are listed Table 8.

Table 8. Averaging Skip Mode Algorithm

Ratio *m*:*n*	PRP_CH1_RZ_HORI_COEF1	PRP_CH1_RZ_HORI_COEF2	PRP_CH1_RZ_HORI_VALID
9	0x00491249	0x00000000	0x09000100
10	0x02491248	0x00000000	0x0A000200
11	0x12481248	0x00000000	0x0B000400
12	0x10491048	0x00000001	0x0C000800
13	0x02411048	0x00000009	0x0D001000
14	0x02411048	0x00000041	0x0E002000
15	0x12090208	0x00000208	0x0F004000
16	0x02081041	0x00001041	0x10008000
17	0x10410208	0x00010208	0x11010000
18	0x00410208	0x00081041	0x12020000
19	0x00410208	0x00401041	0x13040000
20	0x02081008	0x02010208	0x14080000

4 Resize Table

All supported resize ratios are listed Table 9.

Table 9. Resize Ratios

Ratio *m*:*n*																				
	n																			
m	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1																				
2																				
3																				
4																				
5																				
6																				
7																				
8																				
9																				
10																				
11																				
12																				
13																				
14																				
15																				
16																				
17																				
18																				
19																				
20																				

Note: Legend for Table 9

Color Legend	Algorithm	Ratio *m*:*n*
	Averaging	ratio = 1
	Bilinear	1 < ratio < 2, fractional
	Averaging	2 ≤ ratio ≤ 8, integer
	Averaging mix mode	2 < ratio < 8, fractional
	Averaging skip mode	9 ≤ ratio ≤ 20, integer
	Redundant	
	Not supported	

NOTE

For an averaging algorithm, the maximum number of output per window—that is, *n* is 8.Therefore, in the case of 19:9 and 20:9, although the ratio falls into the range of 1<ratio<2, an averaging algorithm cannot be used and a bilinear algorithm must be used instead.

5 Limitations

This section describes the limitations on the resize algorithm and resize ratio support. The maximum size supported by the PRP is 2044 × 2044, however, when the input size is large, the hardware will modify the algorithm, therefore it will be slightly different from what is expected.

For fractional resize ratios, the accuracy is limited by the filter length. Hence not all of the fractional ratios are supported. In those cases, the user must approximate the best value.

5.1 Vertical Resize Limitation

The eMMA Pre-Processor should be able to support an image size up to 2044 × 2044 pixels for both an input and an output port.

When memory-PRP mode is used, the eMMA Pre-Processor is in active position and reads data from the memory. The Resize unit will support both input and output of up to 2044 × 2044 pixels.

When the CSI-PRP link is enabled, the eMMA Pre-Processor is fed by the CSI. The PRP is in a passive position. If the Resize unit is not in use—that is, the resize ratio = 1:1, then the Resize unit will support both input and output of up to 2044 × 2044 pixels.

However, when the Resize unit is in use with the CSI, the vertical Resize unit does not work after 800 pixels. The PRP will not hang, however, the image quality will be impacted. The output image is segmented into 2 parts, the left part with 800 pixels, and right part of (output width – 800) pixels. The left part is generated from the input with correct resize ratios in both horizontal and vertical directions. However for the right part, only the horizontal resize works, the vertical resize algorithm does not. The hardware will try to fulfil the resizing requirement by skipping lines. As a result, vertically the data is downscaled at the correct ratio, 5:4, but not by the specified algorithm (bilinear or averaging). Therefore, the image quality for this part will be slightly different from the other part.

5.1.1 Example: H-Resize = 5:4, V-Resize = 5:4

In this section an example is given using a reduction from 1600 × 1200 to 1280 × 960 pixels—that is a ratio where H-resize = 5:4, and V-resize = 5:4.

Logically, the resize operation can be thought of 2 steps, horizontal resize 1600 to 1280 and vertical resize 1200 to 960 as illustrated in Figure 9.

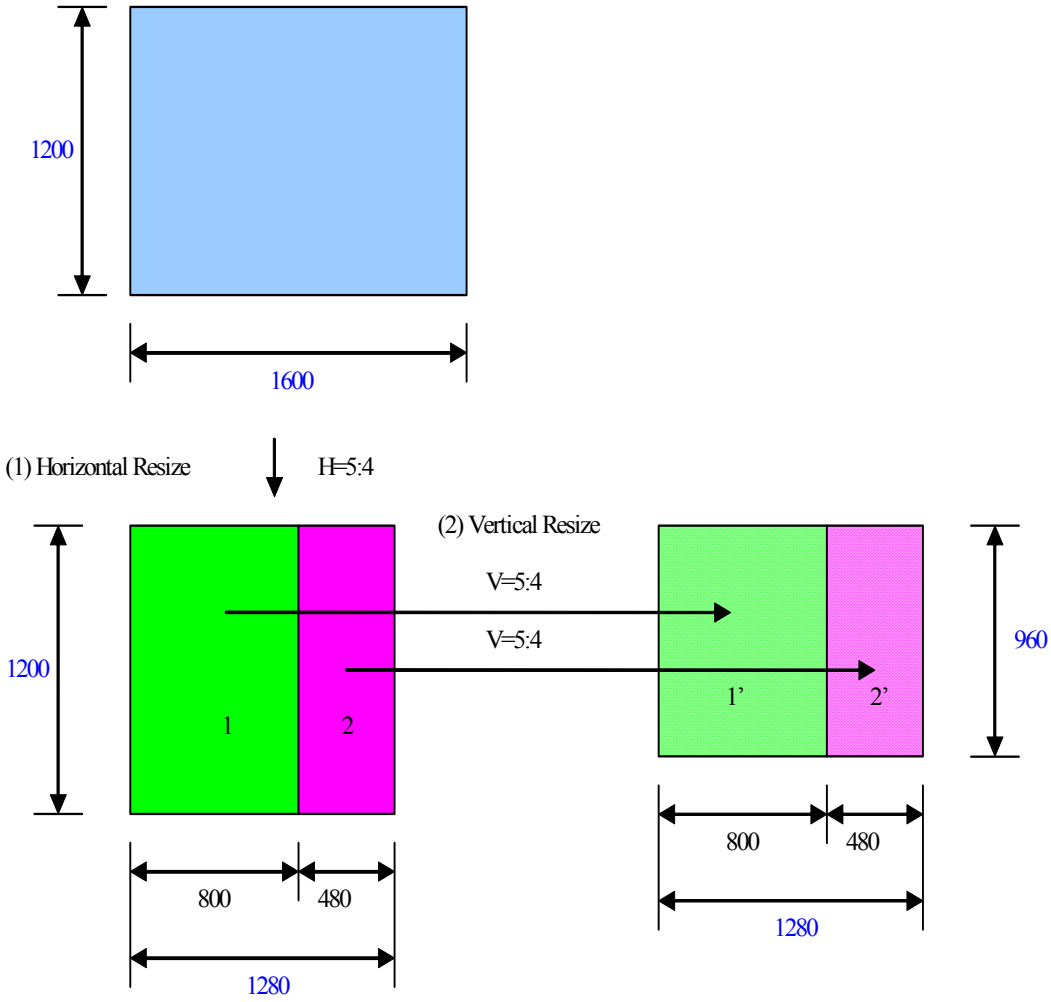

Figure 9. H-Resize = 5:4, V-Resize = 5:4

As shown in Figure 9 in the left part (green color) of the output image, for every line the first 800 output pixels are generated by the Resize unit (ratio 5:4) from 1000 input pixels. Vertically the 960 output lines are generated from 1200 input lines. Both horizontal and vertical resize are doing well by the specified algorithm.

In the right part (purple color) of the output image, for every line the 480 output pixels are generated by the Resize unit (ratio 5:4) from 600 input pixels. Vertically the downscaling is done using the line skipping approach at a ratio = 5:4. Even though the aspect ratio remains, some of the vertical data is discarded.

5.2 Supported Resize Ratios

Theoretically the Resize unit is able to handle maximum resize ratio of 20:1. However, as illustrated, such a high downscaling ratio is achieved by dropping a certain number of pixels and as a result, the image quality is degraded. For smaller ratios using bilinear and averaging, typically all pixels are taken into account so that the image quality will be higher.

The algorithm is designed to support ratio of *m*:*n*, where *m* is at most equal to 20. Such a limitation is due to the filter length. *m* is equal to the filter length, while maximum filter length is 20. This means the Resize unit is able to support ratios of 20:*n*. For other ratios, for example 28:10, when the ratio itself falls into the valid range—that is, between 1:1 to 20:1, the value of *m* is out of range, and therefore the Resize unit will not support this ratio.

How to Reach Us:

USA/Europe/Locations Not Listed:
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-521-6274 or 480-768-2130

Japan:
Freescale Semiconductor Japan Ltd.
Technical Information Center
3-20-1, Minami-Azabu, Minato-ku
Tokyo 106-8573, Japan
81-3-3440-3569

Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
852-26668334

Home Page:
www.freescale.com

AN2886/D
Rev. 0
11/2004

freescale™
semiconductor