Technical Note

MT9V023

Defective Pixel Correction – Description and Usage

Introduction

Defective pixel correction is intended to compensate for defective pixels by replacing their value with a calculated value based on the surrounding pixels, making the defect less noticeable to the human eye. The locations of up to 32 defective pixels as defined by Micron can be stored in an on-chip Defect Map during the manufacturing process. Defective pixel correction is not generally recommended for machine-vision applications.

This technical note provides technical details on function and use of defective pixel correction. Sections covered include:

Definition of Defective Pixel Correction

The response performance of MT9V023 pixels are qualified under different test condition categories. The number of pixels allowed to vary from the mean for each of these categories is defined in detail in the MT9V023 Outgoing Defect Specification (ODS), or the MT9V023 Color Outgoing Defect Specification, for RGB parts.

Defect pixel correction is designed to operate on pixels defined as hot or very hot pixels in the ODS:

- A hot pixel is defined as any single pixel that is greater than 120 LSBs above the mean value of the array when the sensor is operated under no illumination, with an analog gain of 4x and exposure time = 20 msec.
- A very hot pixel is defined as any single pixel that is greater than 500 LSBs above the mean value of the array when the sensor is operated under no illumination, with an analog gain of 4x and exposure time = 20 msec.

Note: The total allowable defects of all types, including hot and very hot pixels, is 30.

The locations of these defective pixels are stored in an internal table. When enabled, defective pixel correction uses this table to perform pixel averaging. A pixel addressed in the table will be substituted with either an average of its neighboring pixels, or its nearest-neighbor pixel, depending on the condition of the neighboring pixels.

The detailed definition of averaging algorithm is provided in the following section.

Defect Correction Algorithm

The correction logic uses information from five of a pixel's same-color neighbors in the same row. For monochrome sensors, same-color neighbors are all adjacent; if a monochrome pixel is in column C and readout is not mirrored, its closest left neighbor (L1) is in column (C - 1). L2 is in column (C - 2), and the nearest right neighbor, R1, is in column (C + 1).

The offset is different for RGB sensors: if a pixel is in column C and readout is not mirrored, its closest left neighbor (L1) is in column (C - 2), L2 is in column (C - 4). Likewise, the nearest right neighbor, R1, is in column (C + 2), R2 is in column (C + 4), and R3 is in column (C + 6). Each of these neighbor pixels can be either good or bad.

To distinguish between monochrome and RGB sensors, it is required to set the Color/Mono bit, R0x0F[1], appropriately. If a pixel is defined as bad in the sensor Defect Map, its value will be replaced with a value P using one of the methods below, depending on the condition of its neighboring pixels.

Table 1: Defective Pixel Replacement Logic

L2	L1	P	R1	R2	R3	Notes
	Good	(L1 + R1)/2	Good			
	Good	(3*L1 + R2)/4	Bad	Good		
	Good	(3*L1 + R3)/4	Bad	Bad	Good	
	Good	L1	Bad	Bad	Bad	
Good	Bad	(L2 + 3*R1)/4	Good			
Good	Bad	(L2 + R2)/2	Bad	Good		
Good	Bad	(3*L2 + R3)/4	Bad	Bad	Good	
Good	Bad	L2	Bad	Bad	Bad	
Bad	Bad	(L3 + 3*R1)/4	Good			
Bad	Bad	(L2 + R2)/2	Bad	Good		Uses the corrected value of L2.
Bad	Bad	(3*L2 + R3)/4	Bad	Bad	Good	Uses the corrected value of L2.
Bad	Bad	L2	Bad	Bad	Bad	Uses the corrected value of L2.
n/a	n/a	R1	Good			On the left edge.
n/a	n/a	R2	Bad	Good		On the left edge.
n/a	n/a	R3	Bad	Bad	Good	On the left edge.

Notes:
1. In column mirror mode, the L and R pixel columns are swapped, such that the pixels are still read in the order L2, L1, P, R1, R2, and R3.
2. In any case not covered in the table above, P = 0. There should be no such cases.
3. Defective pixels on the left (right in column mirror mode) side of the active window are treated as a special case above. The correction equations assume that for defective pixels on the right side of the active window, any pixels beyond the edge of the window are defective.
4. In binned modes, each bin is considered to be one pixel for the purposes of the calculations. A bin is defective if any pixel in the bin is defective. Thus, for 2x2 binning, if any of the four pixels in the bin are defective, the bin is defective. Correction is performed only on the binned result.
5. Row binning is supported for defect correction, in which case defect correction is done after the rows are binned. Column binning is also supported, but in that mode binning occurs after defect correction. Thus defect correction calculations perform as normal for column binning.
6. The correction logic will handle up to 12 defective pixels in any given row, in any configuration, after taking binning into account. If a row has more than 12 defects in the Defect

Map, or more than 12 defects result from binning multiple rows together, the datapath will correct the first 12 defects starting from the left.

7. Defective Pixel Correction is supported in the dark rows, which are treated like normal rows for the purposes of defect correction.
8. Defective Pixel Correction is supported in the dark columns. The dark columns are treated as a separate active region from the real active region. That is, the last dark pixel cannot "see" the first active pixel when calculating its corrected value.

Reading Pixel Defect Addresses

Users may require the pixel locations of the 32-pixel Defect Map. To do so, Defect Map pixel row/column offsets are read out through an indexed register, then an offset value added to obtain the active image row/column values. The full procedure is:

1. Disable defective pixel correction: write a value of 0 to R0x07[9].
2. Write the column offset index for the first defective pixel table entry (value is 32 for first pixel, see Table 2) to R0x60.
3. Read R0x60 to obtain the map column address of the first pixel.
4. Calculate active image column value: image column value = map column address - 1.
5. Write the row offset index for the first defective pixel table entry (= 33) to R0x60.
6. Read R0x60 to obtain the row address of the first defective pixel.
7. Calculate active image row value: image row value = map column address - 4.
8. Repeat column and row accesses for all 32 entries, or until a read returns a value of zero, which means there are no more defective pixel entries.

Table 2: Defective Pixel Table

Defective Pixel Number	Column Offset Index	Row Offset Index
1	32	33
2	34	35
3	36	37
4	38	39
5	40	41
...	…	…
29	88	89
30	90	91
31	92	93
32	94	95

Notes:
1. Register R0x60 may be read only once after writing an index value. Subsequent reads return indeterminate values.
2. The offsets calculated above are relative to a default window, with default binning, window start, window height and width, and non-mirroring. Thus, the results apply for these registers at default: Read Mode (R0xD, R0x0E), Window Height (R0x03, R0xCB), Window Width (R0x04, R0xCC), Column Start (R0x01, R0xC9), and Row Start (R0x02, R0xCA). Additional calculations would be required to determine pixel offsets in a non-default window.

Limitations

Pixel defect test conditions for the MT9V023 are defined in the MT9V023 Outgoing Defect Specification (ODS), or the MT9V023 Color ODS, for RGB color parts. The pixel locations captured during production represent only the defective pixels found under these conditions. Other operating conditions (such as different register settings, or different temperatures) can create a different set of defect pixels. For example, a pixel that may be identified as "hot" under one condition may not be "hot" under another condition and vice versa. Micron makes no claim to the accuracy of the stored defective pixel data when the device is evaluated under different conditions. There is no way to change or augment the defective pixel address table at a later time.

Defective pixel correction works well in monochrome sensors. To use defective pixel correction on color (RGB) parts, the user needs to enable the color bit (R0x0F[1] = 1) in order for the defect correction algorithm to process nearest same-color neighbors, rather than nearest adjacent neighbors. However, enabling the color bit also results in unequal offsets being added to the three color planes. These unequal offsets in turn are generally not useful to machine vision applications where accurate response is required, and may also distort color high dynamic range mode response.

Conclusion

Defective Pixel Correction is a useful feature for improving image quality. This technical note provides information on how defective pixels, as defined in the Defect Pixel Map, are improved using Defective Pixel Correction. For more information on this or other features, refer to the MT9V023 data sheet on Micron's Web site at www.micron.com/imaging.

8000 S. Federal Way, P.O. Box 6, Boise, ID 83707-0006, Tel: 208-368-3900
prodmktg@micron.com www.micron.com Customer Comment Line: 800-932-4992

Revision History

Rev. A 1/7/2008

- Initial release