Error-Tolerant Image Processing Application Based On Stochastic Logic
Checkpoint 3 Report
Team Silicon

- **Problem to be addressed**

  Aggressive scaling of semiconductor devices exacerbates issues of reliability. High density of devices leads to more probable occurrences of hard errors during manufacturing. In addition, soft errors including particles striking and ray radiation will also have a larger impact on the high-density chips compared to the chips less progressive scaling and lower densities. A particle striking on the chip, for example, can cause a bit flip. Therefore, it is important to develop a method to enhance the reliability of the system.

- **Why does this problem matter?**

  The modern techniques that we adapt now are limited in ensuring absolute reliability. An error-tolerant system is in urgent need to eliminate various external factors that can cause errors. A single transient fault may cause the whole system to perform erroneously. Unreliability in systems can lead to huge loss in all fields, including economic loss of company as well as potential hazards in real-world applications.

- **Idea/solution to be investigated by the project**

  Our idea is to use stochastic logics to implement image processing hardware such as sharpening/smoothing filters and make it more error-tolerant for transient faults.

  Stochastic logic is error-tolerant due to its nature and can create analog signal states in digital-based systems. In stochastic logic, a normal input will take advantage of randomness and be transformed to a bit stream. The bit streams or wire bundles are digital, carrying zeros and ones, and the signal is conveyed through the statistical distribution of the logical values.

  The advantages of the stochastic logic are:

  1. Error tolerance. A single binary value is presented with bit stream which reduces the impact when transient fault happens. In normal operations, if the significant bit of one value is corrupted, it is likely to have a large impact on the result. However, in the stochastic logics, the impact is minimized by equally distributing value weight to all bits in the bit stream.
  2. Fast computation in specific applications. While the stochastic creates more bits than traditional logics, in some cases it will be faster by taking advantage of statistic concept. For example in Figure 1, the multiplication $y = x_1 \cdot x_2$ in directly computed by using a simple AND gate. Another example is scaled addition shown in Figure 2. Using a single MUX, the stochastic logic can perform a weighted average easily.

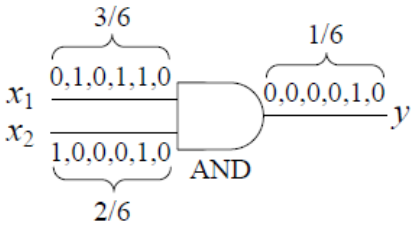


Figure 1. Multiplication in Stochastic Logic

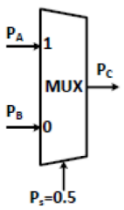


Figure 2. Scaled Addition

➢ Progress so far

1. Error-Tolerant
    a. Inject Transient Errors

    To test the error-tolerant capability of stochastic logic, we randomly flipped the input of the filter at some error rate to see the behavior. The results are shown below. The stochastic filter suffers less from the increasing of the errors visually.

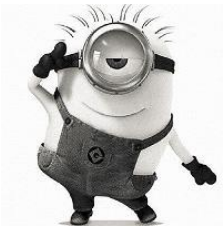


Figure 3. Raw Example Picture

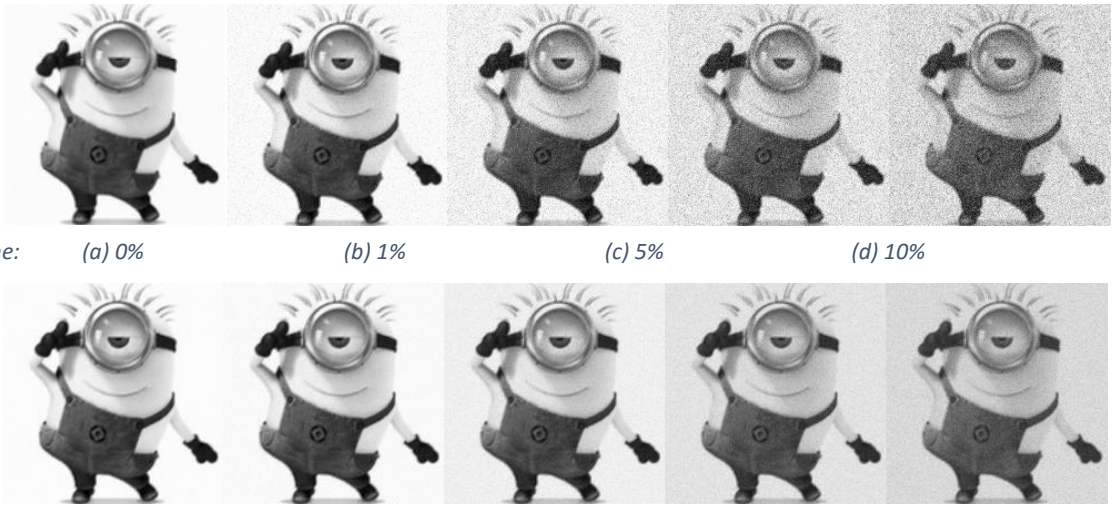
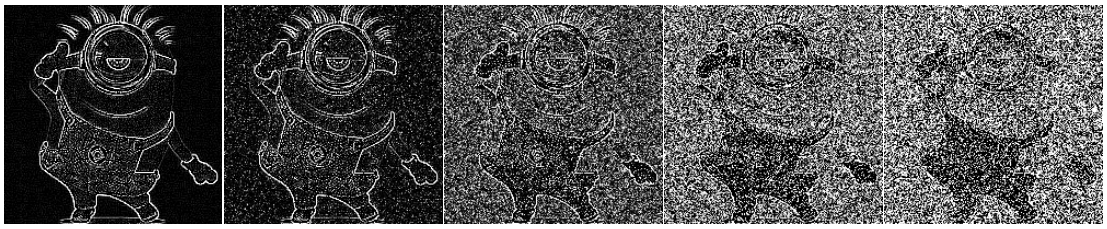
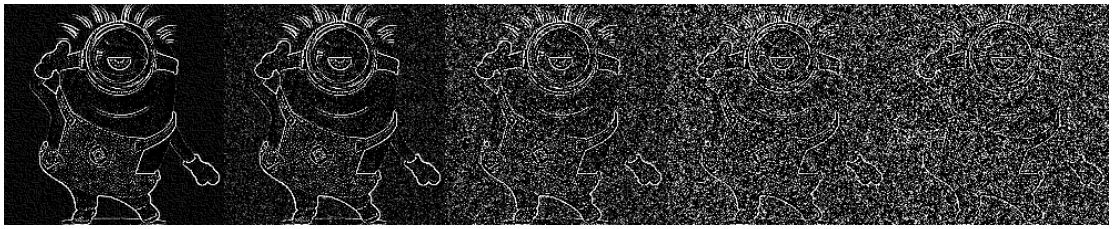


Figure 4. Error-Injection Results with Low-Pass filter



Baseline: (a) 0% (b) 1% (c) 5% (d) 10%



Stochastic: (a) 0% (b) 1% (c) 5% (d) 10% (e)

Figure 5. Error-Injection Results with High-Pass filter

b. Evaluation methodology

To evaluate the results quantitatively, we use SSIM and PSNR metrics to check the similarity between reference picture and the one with errors injected.

- The Structural SIMilarity (SSIM) index is a method for measuring the similarity between two images. The SSIM index can be viewed as a quality measure for one of the images being compared, provided the other image is regarded as of perfect quality.
- The PSNR block computes the peak signal-to-noise ratio, in decibels, between two images. The higher the PSNR, the better the quality of the compressed, or reconstructed image.

A few observations:

- The stochastic filter outperforms the baseline filter in every error injection rate in terms of the error-tolerance.
- The high-pass filter is more prone to faults. This is because the low-pass filter will smooth the picture and tolerant errors to some extents by its nature while the high-pass filter will actually amplify the faults.

Table 1: Comparison of SSIM and PSNR for Low-Pass Filter

Low-Pass Filter	**SSIM**		**PSNR (dB)**	
	Baseline	Stochastic	Baseline	Stochastic
1%	0.826	0.9614	31.558	36.4829
5%	0.552	0.8860	22.819	25.7218
10%	0.421	0.8088	18.255	20.0282
15%	0.358	0.7520	15.744	17.0239

Table 2: Comparison of SSIM and PSNR for High-Pass Filter

High-Pass Filter	SSIM		PSNR (dB)	
	Baseline	Stochastic	Baseline	Stochastic
1%	0.470	0.5473	4.334	6.3432
5%	0.231	0.3042	-3.160	-0.3760
10%	0.142	0.2184	-5.904	-2.7470
15%	0.099	0.1788	-6.834	-3.5915

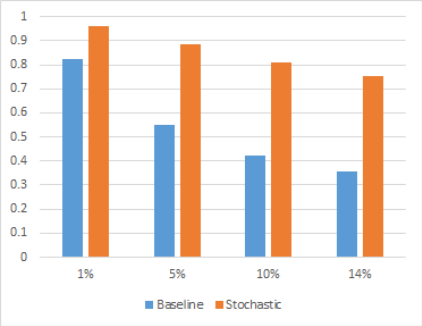


Figure 6. Low-Pass SSIM Comparison

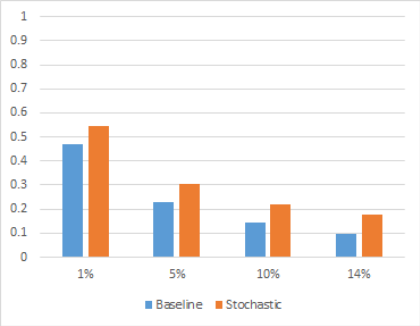


Figure 7. High-Pass SSIM Comparison

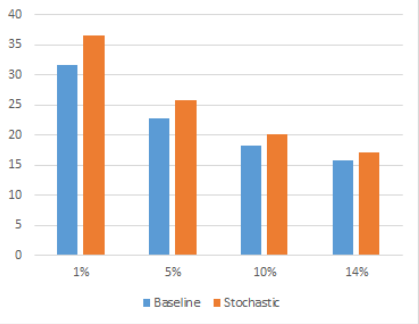


Figure 8. Low-Pass PSNR Comparison

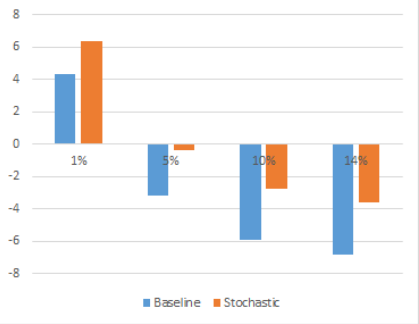


Figure 9. High-Pass PSNR Comparison

2. SSIM and PSNR Comparison between Baseline and Stochastic

We want to compare the similarity between baseline and stochastic filter implementations. We can see the baseline and stochastic low-pass filter is almost the same while there exists some difference in high-pass filter (though not visually identifiable). This is because the high-pass filter amplifies the quantization errors between these two algorithms.

Table 3: Comparison of SSIM and PSNR for Baseline and Stochastic Filters

	SSIM	PSNR
Low-Pass	0.9925	45.6314
High-Pass	0.6886	10.3558

3. Clock Period/Synthesized Area

   We evaluated speed/area of two designs. We can see from Table 1, the clock period and area of stochastic design are both smaller than baseline design. However, the stochastic design will need more cycles to execute (2^8 in our case). After some optimization, we managed to keep the Period∗Area∗#Cycle roughly the same for baseline design and stochastic design (if only count the core module).

Table 4. Comparison of Synthesis Results

	Baseline	Stochastic	Stochastic (core only)
Clock Period (ns)	7.8	1.8	1.8
Synthesis Area (μm^2)	49,687	26,796	1,063
Total Cycle	N	$N \cdot 2^8$	$N \cdot 2^8$
Period∗Area∗#Cycle	387,559N	12,347,596N	489, 830N

4. Conclusion

   The stochastic logic filter can operate at a faster clock frequency and consume less silicon area. While it will take more cycle numbers than baseline filter to execute, stochastic logic takes advantage of randomness to make the filter errors tolerant and more reliable.

➢ Future works

1. Poster
2. Report

➢ Issues

1. Pictures
   a. Do we need to show the results of different pictures? Will that be too redundant and unnecessary?
   b. What is the best style to present our SSIM/PSNR results? (figure 6,7,8,9)
   c. Do we need to keep the data range of the PSNR results the same? The negative value makes the graph a little bit weird. (figure 8,9)
2. What should we include in the poster?