Available online www.jsaer.com

***Journal of Scientific and Engineering Research*, 2024, 11(7):136-139**

Research Article

ISSN: 2394-2630
CODEN(USA): JSERBR

Glossary of Image Capture Artifacts

Karthik Poduval, Jason Xiong

Abstract On a mordern digital image capturing system on an SoC, we often times find certain atrifacts in the captured image. This paper describes common such artifacts and their possible causes.

Keywords Image Capture Artifacts

1. Introduction

On a modern day SoC that contains a camera capture interface such as a MIPI CSI2 interface and an inbuilt ISP, we often see various image capture artifacts during the development of the software. Often times such artifacts or issues only have a visual component and no errors are seen in the logs that indicate on how the issue may have occurred. In such situations the only way to make out the root cause of the issue is by studying the capture artifact and fixing the issue based on the type of artifact.

2. Understanding the Capture System

An introduction of the capture system is necessary for the sections that will follow. Refer to Figure 1 where we have an ISP where we have a raw Bayer sensor [1]. Such Bayer image sensors will interface with and SoC using MIPI CSI2 type bus interface [2]. Once inside the SoC there would typically be a CSI2 Receiver that may have a DMA to write out the raw Bayer format and Bayer data also gets sent into an ISP that may have various stages like black level correction, defective pixel correction, demosaic, color correction, scaling and color space conversion. The processed image from the ISP is then written out to memory using a DMA. The ISP may also write out statistics that typically gets used by 3A algorithms [3] to dynamically adjust the image sensor and ISP in real-time to produce a good quality image. In the next section we will look at the common image capture artifacts and what are the possible causes for them.

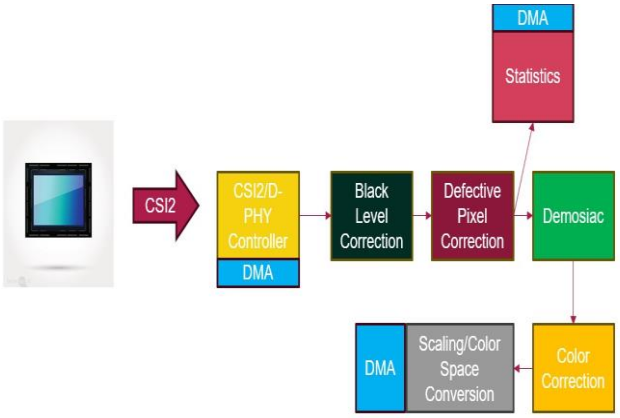

Figure 1: Image Capture System

3. Image Capture Artifacts

A. Incorrect CFA

This happens when we try to view a raw Bayer image using a tool if the bayer color pattern isn't set correctly the colors seem messed up. As seen in Figure 2 the captured image bayer pattern in BGGR but when mistakenly viewed as GBRG the colors all looked up but when viewed as BGGR as shown in Figure 3 it looks correct. The raw Bayer is from project xkISP from OpenASIC [4] and the viewer used is PixelViewer [5]. To fix this issue one must look up the data sheet of the image sensor to determine what kind of bayer pattern it outputs and select it accordingly. Note that cropping and/or rotation of the image can also affect the bayer pattern, hence all those settings need to be analyzed to view the raw Bayer correctly.

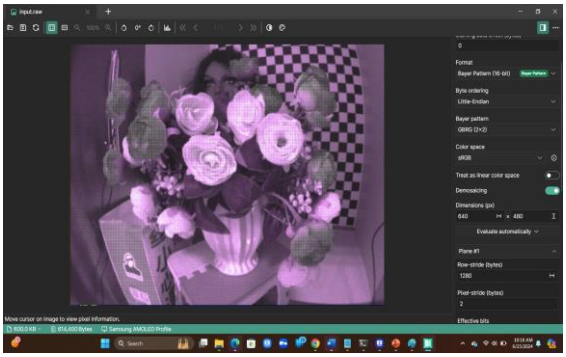

Figure 2: RAW BGGR viewed as GBRG

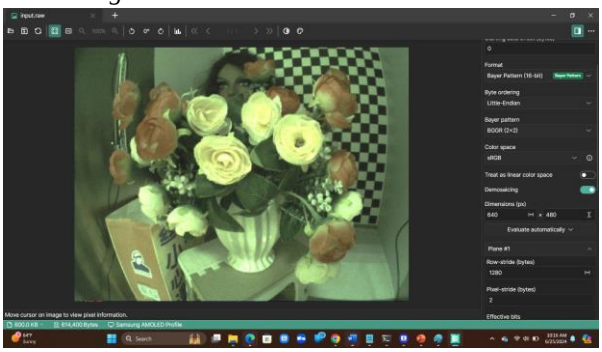

Figure 3: RAW BGGR viewed as BGGR

B. Cache Line Artifacts

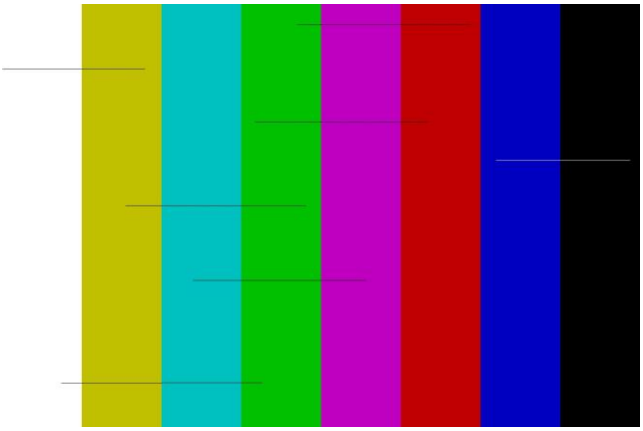

Figure 4: Incoherent Cache Artifact

These kinds of artifacts are seen on the captured image in the from of random colored lines about the size of the CPU's cache line length. They arise due to cache incoherency. When the DMA operation of the ISP writes captured image data into the memory, it does so directly to the memory via a DMA controller. However CPU's read this image from the memory via its local cache. The local cache may not be coherent after the image is written by DMA so when CPU reads the image it sees random lines appearing in the image which are of cache

line length due to incoherent caches as can be seen from Figure 4. To fix such issues, the driver software needs to invalidate caches before it writes out the image through DMA.

C. Frame Tear

Figure 5: Frame Tear

Frame tear is and issue much like screen tearing [6] as shown in Figure 5. This happens typically due to synchronization issue between when hardware writes into frame and when software reads from it. The issue can also occur if 2 of ISP's DMA address registers are misconfigured to write into same address. If such an issue is observed its good to check the software stack for such issues.

D. Split Image Issue

Figure 6: Normal

Figure 7: Split

When this issue the image splits in horizontal axis as shown in 6 and Figure 7. This issue can happen due to hardware not synchronizing properly to start and end frame synchronization signals in the image stream. This can cause the hardware to start capturing mid point during image line transmission and capture part of one line of image data and part of another line of image data that that can lead to such an artifact. The split image can also happen to vertical lines. Some ISP hardware implementation counts pixels to determine frame end. And soft reset can cause the pixel counter to reset. Asynchronized soft reset in the middle of frame transmission can lead to both horizontal and vertical split image.

E. Partial Image

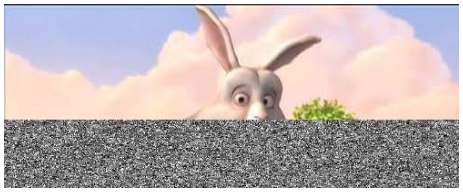

Figure 8: Partial Capture

Partial image capture as shown in Figure 8 is when the captured image is smaller in height. In some cases there also maybe a corrupting in the last few lines of data captured. This can happen due to height misconfiguration in

the CSI2 or ISP configuration. It may also occur due to FIFO overflows of some blocks in the hardware and the diagnosis would be to check for height related and FIFO depth configurations.

F. Stale Frame

Stale frame issue occurs when we see an old frame occurring in image video stream. The best way to test is to hold a clock in front of the camera and run a preview or record a video or image stream. If the clock runs backwards then we have a stale frame. This issue occurs when the ISP's DMA fails to update a frame and the old content of the frame gets shipped to the application. If this occurs, ideally the frame should be dropped as its not been updated. If such an issue occurs one should check the DMA to see if there have been any frame update failures or not.

References

[1]. B. E. Bayer, "Color imaging array," Jul 1976.
[2]. "Mipi csi2." [Online]. Available: https://www.mipi.org/specifications/ csi-2
[3]. "3a algorithms." [Online]. Available: https://gimoonnam.github.io/ imageprocessing/3A-initiation/
[4]. "xkisp." [Online]. Available: https://github.com/openasic-org/xkISP
[5]. "Pixel viewer." [Online]. Available: https://github.com/carina-studio/ PixelViewer/
[6]. "Screen tearing." [Online]. Available: https://medium.com/@chiawei/ reducing-screen-tearing-and-input-lag-cc9fcea37c7a