

HIGH PERFORMANCE IMAGE DEMOSAICING HARDWARE DESIGNS

by

SERKAN YALIMAN

Submitted to the Graduate School of Engineering and Natural Sciences

in partial fulfillment of

the requirements for the degree of

Master of Science

Sabancı University

Fall 2013

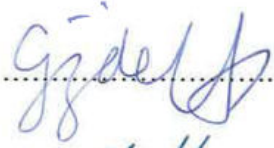
HIGH PERFORMANCE IMAGE DEMOSAICING HARDWARE DESIGNS

APPROVED BY

Assoc. Prof. Dr. İlker HAMZAOĞLU
(Thesis Supervisor)



Assoc. Prof. Dr. Gözde ÜNAL



Assist. Prof. Dr. Hüsnü YENİGÜN



DATE OF APPROVAL:

13.01.2014

© Serkan Yalman 2014

All Rights Reserved

HIGH PERFORMANCE IMAGE DEMOSAICING HARDWARE DESIGNS

Serkan YALIMAN

EE, MS Thesis, 2014

Thesis Supervisor: Assoc. Prof. Dr. İlker HAMZAOĞLU

Keywords: Demosaicing, effective color interpolation, enhanced effective color interpolation, alternating projections, hardware implementation, FPGA.

Abstract

Most digital cameras capture only one color channel (red, green, or blue) per pixel, because capturing three color channels per pixel would require three image sensors which increases the cost of digital cameras. Therefore, only one image sensor is used, and images pass through a color filter array (CFA) before being captured by the image sensor. Bayer pattern is the most commonly used CFA pattern in digital cameras. Demosaicing is the process of reconstructing the missing color channels of the pixels in the color filtered image using their available neighboring pixels. There are many image demosaicing algorithms with varying reconstructed image quality and computational complexity. In this thesis, high performance hardware architectures are designed for three high quality image demosaicing algorithms, and the proposed hardware architectures are implemented on FPGA.

A high performance hardware architecture for Effective Color Interpolation (ECI) demosaicing algorithm is proposed. A modified version of Enhanced ECI demosaicing algorithm and a high performance hardware architecture for this image demosaicing algorithm are proposed. A hybrid ECI and Alternating Projections demosaicing algorithm and a high performance hardware architecture for this image demosaicing algorithm are proposed. The proposed hardware architectures are implemented using Verilog HDL. The Verilog RTL codes are mapped to Xilinx Virtex 6 FPGA. The proposed FPGA implementations are verified with post place & route simulations. They are capable of processing 160, 118, and 119 full HD images per second.

YÜKSEK PERFORMANSLI RESİM DEMOZAIKLEME DONANIM TASARIMLARI

Serkan YALIMAN

EE, Yüksek Lisans Tezi, 2014

Tez Danışmanı: Doç. Dr. İlker HAMZAOĞLU

Anahtar Kelimeler: Demozaikleme, etkin renk interpolasyonu, pekiştirilmiş etkin renk interpolasyonu, değişken projeksiyon, donanım tasarımı, FPGA.

Özet

Çoğu dijital fotoğraf makinesi piksel başına sadece bir renk kanalı (kırmızı, yeşil veya mavi) yakalar, çünkü piksel başına üç renk kanalını yakalamak için üç resim sensörü kullanılması gerekir ve bu dijital fotoğraf makinesinin maliyetini artırır. Bu nedenle sadece bir resim sensörü kullanılır ve resimler sensörde yakalanmadan önce renk filtresi düzeneğinden geçer. Dijital fotoğraf makinelerinde en sık kullanılan renk filtresi düzeneği Bayer'dir. Demozaikleme, renk filtresinden geçmiş resmin piksellerindeki eksik renk kanallarının mevcut komşu piksellerden elde edilmesi işlemidir. Farklı yeniden oluşturulan resim kalitesi ve işlemsel karmaşıklığa sahip birçok demozaikleme algoritması vardır. Bu tezde, üç yüksek kaliteli resim demozaikleme algoritması için yüksek performanslı donanım mimarileri tasarlandı ve önerilen donanımlar FPGA üzerinde gerçekleştirildi.

Efektif Renk İnterpolasyonu (ERİ) demozaikleme algoritması için yüksek performanslı donanım mimarisi önerildi. Zenginleştirilmiş ERİ demozaikleme algoritmasının modifiye edilmiş versiyonu ve bu resim demozaikleme algoritması için yüksek performanslı donanım mimarisi önerildi. Hibrit ERİ ve değişken projeksiyonlar demozaikleme algoritması ve bu resim demozaikleme algoritması için yüksek performanslı donanım mimarisi önerildi. Önerilen donanım mimarileri Verilog donanım tanımlama dili kullanılarak gerçekleştirildi. Verilog kodları Xilinx Virtex 6 FPGA'ya yerleştirildi. FPGA gerçeklemeleri yerleştirme ve yönlendirme sonrası simülasyonlar ile doğrulandı. Bu FPGA gerçeklemeleri saniyede 160, 118 ve 119 tam yüksek çözünürlüklü (YÇ) resim işleyebilmektedir.

Acknowledgements

I would like to thank my advisor Dr. İlker Hamzaoğlu for his guidance, support and teachings during my masters' education. His suggestions, corrections and reviews were immensely helpful throughout my thesis.

I am grateful to my thesis committee members, Dr. Gözde Ünal and Dr. Hüsnü Yenigün, for their precious feedback and time.

I would like to thank to members of System-on-Chip Design and Test Lab, Yusuf Adıbelli, Zafer Özcan, Erdem Özcan, Ercan Kalalı, Kamil Erdayandı, Yusuf Akşehir, Hasan Azgın for their support. I also would like to thank Fatih Melemez, Çağatay Yılmaz, Tolga Dinç and Mehmet Ahat for their friendship throughout my studies.

I am also thankful to my family for their support during my educational path.

Lastly, I would like to appreciate Sabancı University and TÜBİTAK for supporting me during my masters studies.

TABLE OF CONTENTS

HIGH PERFORMANCE IMAGE DEMOSAICING HARDWARE DESIGNS	I
HIGH PERFORMANCE IMAGE DEMOSAICING HARDWARE DESIGNS	IV
Abstract.....	IV
YÜKSEK PERFORMANSLI RESİM DEMOZAİKLEME DONANIM TASARIMLARI ...	V
Özet	V
Acknowledgements	VI
TABLE OF CONTENTS	VII
LIST OF FIGURES	VIII
LIST OF TABLES	IX
LIST OF ABBREVIATIONS	X
Chapter 1	1
INTRODUCTION	1
1.1. Thesis Contribution	3
Chapter 2	7
EFFECTIVE COLOR INTERPOLATION HARDWARE	7
2.1 Hardware Implementation.....	10
Chapter 3	16
MODIFIED ENHANCED EFFECTIVE COLOR INTERPOLATION HARDWARE	16
3.1 Modified Enhanced ECI Algorithm	18
3.2 Hardware Implementation	25
Chapter 4	31
HYBRID EFFECTIVE COLOR INTERPOLATION AND ALTERNATING PROJECTIONS HARDWARE.....	31
4.1 Hybrid ECI and AP Algorithm.....	34
4.2 Hardware Implementation.....	38
Chapter 5	43
CONCLUSION AND FUTURE WORK	43
Bibliography	44

LIST OF FIGURES

Figure 1.1 RGB color filter arrays (a) Bayer [2] (b)Yamanaka [10] (c) Lukac and Plataniotis [11] (d) Vertical-stripe [12] (e) Diagonal-stripe [12] (f) Modified Bayer [12] (g) HVS-based [13]	1
Figure 1.2 Image demosaicing.....	2
Figure 1.3 Aliasing problem in bilinearly interpolated image.....	3
Figure 1.4 Kodak Image Suite	5
Figure 2.1 ECI flow.....	7
Figure 2.2 Reference CFA pattern for ECI calculations	8
Figure 2.3 An example reconstructed image by ECI	8
Figure 2.4 Proposed ECI hardware	12
Figure 2.5 (a) Image sensor data scan order for ECI hardware (b) Raster scan order	13
Figure 2.6 Rotating block RAM organization (a) when processing rows n through n+7 (b) when processing rows n+2 through n+9 (8x8 current window is shown in grey)	13
Figure 2.7 (a) 8x8 mosaicked window for processing the central 2x2 (b) K_R' and K_B' are calculated by bilinearly interpolating red and blue values (c) K_R'' , K_B'' are calculated by averaging surrounding 4 K_R' , K_B' values (d) K_R''' , K_B''' , K_R'''' , K_B'''' values are calculated for obtaining missing red, blue values	15
Figure 3.1 (a) ECI flow (b) Enhanced ECI flow (c) Modified Enhanced ECI flow	19
Figure 3.2 An example reconstructed image by EECI algorithms.....	20
Figure 3.3 Datapath stage1 inputs and outputs	26
Figure 3.4 Datapath stage2 inputs and outputs	27
Figure 3.5 Datapath stage3 inputs and outputs	27
Figure 3.6 Datapath of the proposed modified EECI hardware.....	28
Figure 3.7 (a) K value calculation (b) weight calculation	29
Figure 3.8 K calculator used in the proposed modified EECI hardware.....	30
Figure 4.1 Alternating Projections algorithm	31
Figure 4.2 MSE performance of different AP iterations for 24 Kodak Suite images	33
Figure 4.3 Proposed hybrid ECI and AP algorithm	34
Figure 4.4 An example reconstructed image by demosaicing algorithms.....	35
Figure 4.5 Proposed hybrid ECI and AP demosaicing hardware	39
Figure 4.6 (a) Low High filter for 3x3 window (b) Inverse Low High filter for 5x5 window	41

LIST OF TABLES

TABLE 2.1 PSNR (dB) COMPARISON OF IMAGE DEMOSAICING ALGORITHMS	9
TABLE 3.1 AVERAGE PSNR VALUES in dB OVER 30 IMAGES [4].....	18
TABLE 3.2 SCALED RANGE FOR α VALUES	19
TABLE 3.3 PSNR (dB) COMPARISON OF ENHANCED ECI ALGORITHMS	21
TABLE 3.4 PSNR (dB) COMPARISON OF IMAGE DEMOSAICING ALGORITHMS	23
TABLE 3.5 COMPLEXITY COMPARISON	25
TABLE 4.1 PSNR (dB) PERFORMANCE OF DIFFERENT AP ITERATIONS FOR LIGHTHOUSE IMAGE.....	33
TABLE 4.2 PSNR (dB) COMPARISON OF DEMOSAICING ALGORITHMS FOR LIGHTHOUSE IMAGE.....	35
TABLE 4.3 PSNR (dB) COMPARISON OF IMAGE DEMOSAICING ALGORITHMS	36
TABLE 4.4 COMPLEXITY COMPARISON	38

LIST OF ABBREVIATIONS

CFA	:	Color Filter Array
RGB	:	Red, Green, Blue
PSNR	:	Peak Signal-to-Noise Ratio
MSE	:	Mean Squared Error
BRAM:		Block RAM
HD	:	High Definition
cPSNR:		Composite Peak Signal-to-Noise Ratio
fps	:	Frames per second
POCS	:	Projection onto Convex Sets
AP	:	Alternating Projections
ECI	:	Effective Color Interpolation
LUT	:	Lookup Table
HVS	:	Human Visual System
DSC	:	Digital Still Camera

Chapter 1

INTRODUCTION

Most digital cameras capture only one color channel (red(R), green(G), or blue(B)) per pixel. Because, capturing three color channels per pixel would require three image sensors which increases the cost of digital cameras. Therefore, only one image sensor is used, and images pass through a color filter array (CFA) before being captured by the image sensor. As shown in Figure 1.1, there are several CFA patterns. Bayer pattern, shown in Figure 1.1 (a), is the most commonly used CFA pattern in digital cameras [1]. Bayer pattern takes the human vision systems relatively higher sensitivity to green into account by sampling green channel at twice the rate of red and blue channels [2].

As shown in Figure 1.2, CFA interpolation, commonly known as demosaicing (or demosaicking), is the process of reconstructing the missing color channels of the pixels in the color filtered image using their available neighboring pixels. There are many demosaicing algorithms with varying reconstructed image quality and computational complexity.

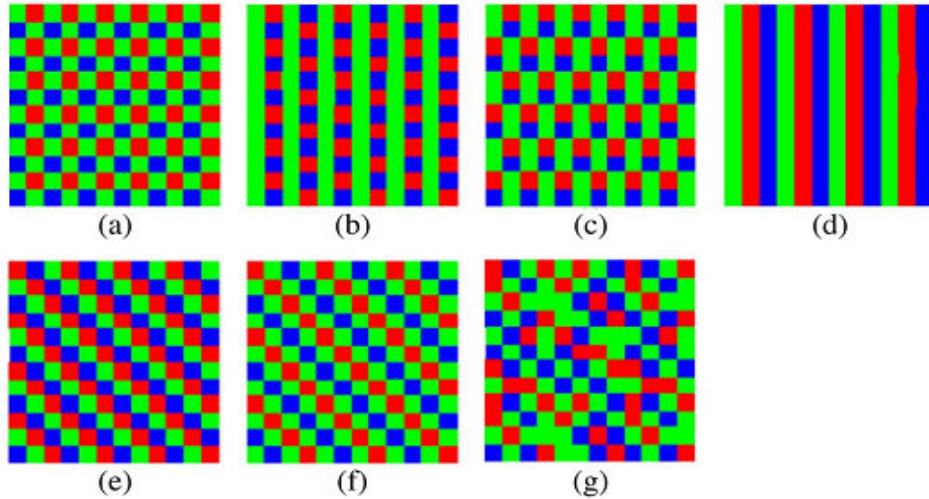
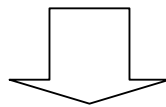


Figure 1.1 RGB color filter arrays (a) Bayer [2] (b)Yamanaka [10] (c) Lukac and Plataniotis [11] (d) Vertical-stripe [12] (e) Diagonal-stripe [12] (f) Modified Bayer [12] (g) HVS-based [13]

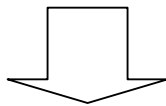
Original Image

R	R	R	R	G	G	G	G	B	B	B	B
R	R	R	R	G	G	G	G	B	B	B	B
R	R	R	R	G	G	G	G	B	B	B	B
R	R	R	R	G	G	G	G	B	B	B	B



CFA

G	R	G	R
B	G	B	G
G	R	G	R
B	G	B	G



Demosaicing

Demosaiced Image

R recr.	R orig.	R recr.	R orig.	G orig.	G recr.	G orig.	G recr.	B recr.	B recr.	B recr.	B recr.
R recr.	R recr.	R recr.	R recr.	G recr.	G orig.	G recr.	G orig.	B orig.	B recr.	B orig.	B recr.
R recr.	R orig.	R recr.	R orig.	G orig.	G recr.	G orig.	G recr.	B recr.	B recr.	B recr.	B recr.
R recr.	R recr.	R recr.	R recr.	G recr.	G orig.	G recr.	G orig.	B orig.	B recr.	B orig.	B recr.

Figure 1.2 Image demosaicing

Simple demosaicing algorithms such as bilinear or bicubic interpolation produce low quality reconstructed images, often accompanied with aliasing problems (zipper effect) around edges, as shown in Figure 1.3(b). Therefore, complex demosaicing algorithms such as Effective Color Interpolation (ECI) [3], Enhanced ECI [4] and Alternating Projections (AP) [5] are proposed. These algorithms use larger number of neighboring pixels for color interpolation and the correlation between different color channels of the neighboring pixels. They produce higher quality reconstructed images at the expense of increased computational complexity. Therefore, efficient hardware architectures should be designed for real-time implementation of these complex image demosaicing algorithms. In addition, these algorithms can be modified to make them more suitable for hardware implementation.

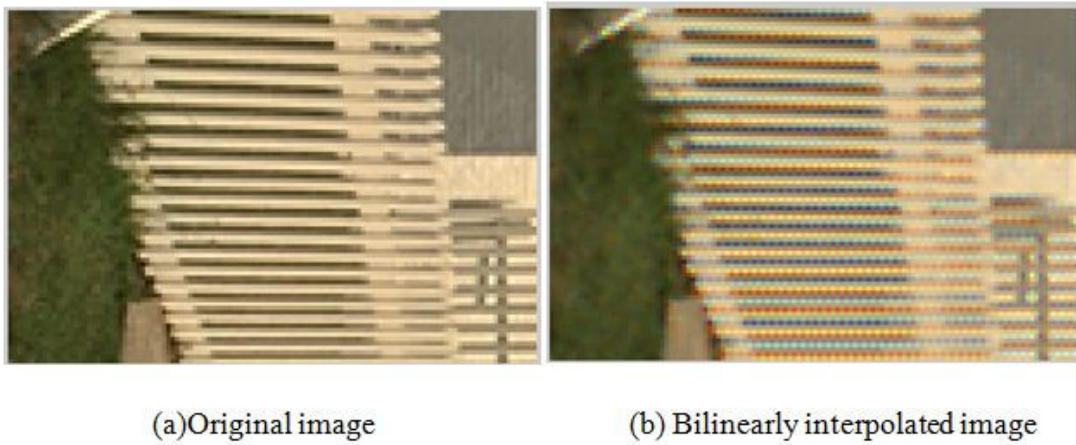


Figure 1.3 Aliasing problem in bilinearly interpolated image

1.1. Thesis Contribution

Since there are real-time hardware implementations of only low quality demosaicing algorithms in the literature, in this thesis, high performance hardware architectures are designed for three high quality image demosaicing algorithms, and the proposed hardware architectures are implemented on FPGAs. First, we propose a high performance hardware architecture for Effective Color Interpolation (ECI) demosaicing algorithm [3]. The proposed hardware architecture is implemented using Verilog HDL. The Verilog RTL code is mapped

to a Xilinx Virtex 6 FPGA. The proposed FPGA implementation is verified with post place & route simulations. It is capable of processing 160 full High Definition (HD) images per second or 40 quad HD images per second.

Then, we propose a modified version of Enhanced ECI (EECI) algorithm [4]. EECI algorithm produces high quality reconstructed images, but it uses a large number of division operations. Therefore, in order to reduce hardware complexity, we propose some modifications to EECI algorithm. The modified EECI algorithm produces 1.48 dB, 0.64 dB, and 0.26 dB better PSNR results than ECI algorithm for red, green, and blue channels, respectively. We also propose a high performance hardware architecture for this demosaicing algorithm. The proposed hardware architecture is implemented using Verilog HDL. The Verilog RTL code is mapped to a Xilinx Virtex 6 FPGA. The proposed FPGA implementation is verified with post place & route simulations. It is capable of processing 118 full HD images per second or 29 quad HD images per second.

Finally, we propose a hybrid ECI and Alternating Projections (AP) demosaicing algorithm [5]. In order to reduce hardware complexity, we propose avoiding the iterations in the AP algorithm by adding ECI algorithm in the intermediate step. The hybrid ECI and AP algorithm produces 1.88 dB better PSNR results than ECI algorithm for green channel, and same PSNR results for red and blue channels. We also propose a high performance hardware architecture for this demosaicing algorithm. The proposed hardware architecture is implemented using Verilog HDL. The Verilog RTL code is mapped to Xilinx Virtex 6 FPGA. The proposed FPGA implementation is verified with post place & route simulations. It is capable of processing 119 full HD images per second or 29 quad HD images per second.

In this thesis, demosaicing algorithms are evaluated as follows. Images in the Kodak Image Suite shown in Figure 1.4 [14], which include all three color channels, are filtered through Bayer pattern. The color filtered images are reconstructed by the demosaicing algorithm. Each color channel (R, G, B) of each reconstructed image is then compared to the corresponding color channel of the original image using Peak-Signal-to-Noise (PSNR) metric. Higher PSNR value indicates that the reconstructed image more closely matches the original image. PSNR is calculated using Mean Squared Error (MSE), as shown in Equations 1.1 and 1.2. N and M denote the image height and width respectively. R is the reconstructed image and O is the original image. MAX is the maximum value a pixel can take. In this

thesis, this value is set to 255. Composite PSNR (sometimes referred as color PSNR) is calculated as shown in Equations 1.3 and 1.4.

$$MSE = \frac{1}{NM} \sum_{i=0}^{N-1} \sum_{j=0}^{M-1} (R(i,j) - O(i,j))^2 \quad (1.1)$$

$$PSNR = 10 \cdot \log_{10} \left(\frac{MAX^2}{MSE} \right) = 20 \cdot \log_{10} \left(\frac{MAX}{\sqrt{MSE}} \right) \quad (1.2)$$

$$cPSNR = 10 \cdot \log_{10} \left(\frac{MAX^2}{MSE_c} \right) = 20 \cdot \log_{10} \left(\frac{MAX}{\sqrt{MSE_c}} \right) \quad (1.3)$$

$$MSE_c = \frac{1}{3NM} \sum_{i=0}^{N-1} \sum_{j=0}^{M-1} \sum_{k=0}^2 (R(i,j,k) - O(i,j,k))^2 \quad (1.4)$$

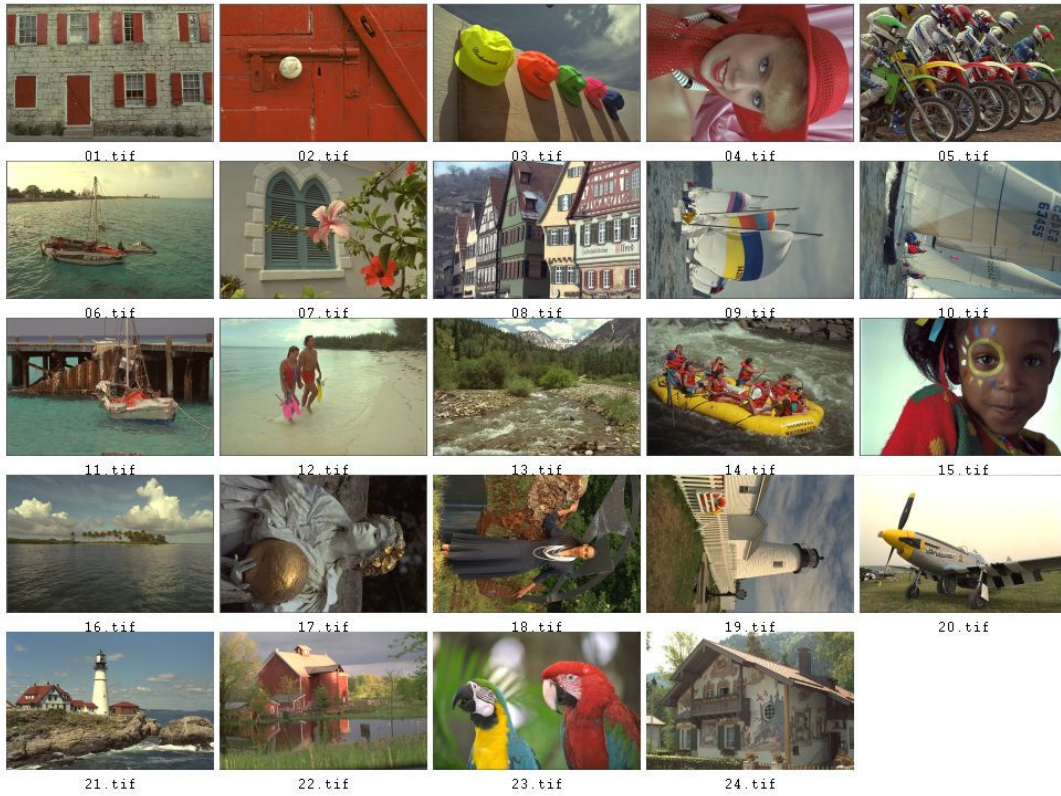


Figure 1.4 Kodak Image Suite

The rest of the thesis is organized as follows. In Chapter 2, the proposed high performance ECI hardware is presented, and its implementation results are given. In Chapter 3, the proposed modified EECI demosaicing algorithm is presented, and its performance results are given. In addition, the proposed high performance hardware for this algorithm is presented, and its implementation results are given. In Chapter 4, the proposed hybrid ECI and AP demosaicing algorithm is presented, and its performance results are given. In addition, the proposed high performance hardware for this algorithm is presented, and its implementation results are given. Chapter 5 concludes the thesis.

Chapter 2

EFFECTIVE COLOR INTERPOLATION HARDWARE

Effective color interpolation demosaicing algorithm takes advantage of the inter-channel correlation between green and blue channels, as well as between green and red channels [3]. ECI flow is shown in Figure 2.1. The reference CFA pattern used for the following ECI calculations is shown in Figure 2.2. ECI calculates K_R and K_B values for green pixels as shown in Equations 2.1 and 2.2. It calculates missing R and B values for green pixels as shown in Equations 2.3 and 2.4.

$$K_R = G - R \quad (2.1)$$

$$K_B = G - B \quad (2.2)$$

$$K'_R 3 = G3 - R'3 = G3 - \frac{1}{2}(R1 + R7) \quad (2.3)$$

$$K'_R 6 = G6 - R'6 = G6 - \frac{1}{2}(R5 + R7) \quad (2.4)$$

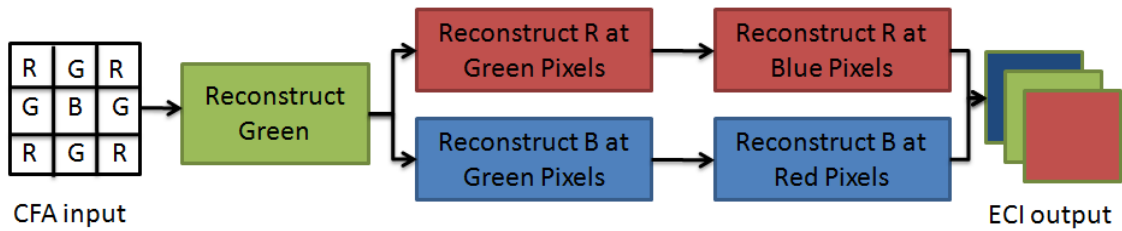


Figure 2.1 ECI flow

R	G	R1	G	R
G	B2	G3	B4	G
R5	G6	R7	G8	R9
G	B10	G11	B12	G
R	G	R13	G	R

Figure 2.2 Reference CFA pattern for ECI calculations

ECI algorithm first calculates missing green values as shown in Equation 2.5. Missing green values at blue pixels are calculated similarly using K'_B values. After calculating all missing green values, it calculates missing red and blue values as shown in Equations 2.6 and 2.7.

$$G'7 = R7 + \frac{1}{4}(K'_R3 + K'_R6 + K'_R8 + K'_R11) \quad (2.5)$$

$$R'3 = G3 - \frac{1}{2}(K'_R1 + K'_R7) \quad (2.6)$$

$$B'7 = G'7 - \frac{1}{4}(K'_B2 + K'_B4 + K'_B10 + K'_B12) \quad (2.7)$$



(a) Reconstructed by ECI



(b) Original

Figure 2.3 An example reconstructed image by ECI

The objective qualities of the reconstructed images by ECI demosaicing algorithm are shown in Table 2.1. The subjective quality of an example reconstructed image by ECI demosaicing algorithm is shown in Figure 2.3.

TABLE 2.1 PSNR (dB) COMPARISON OF IMAGE DEMOSAICING ALGORITHMS

Image	Bilinear	Laplacian	ECI	Image	Bilinear	Laplacian	ECI
1	25.31	30.86	33.02	13	23.09	27.67	31.13
	29.58	34.71	35.66		26.49	30.68	32.52
	35.35	30.80	33.50		23.00	27.43	30.82
2	31.89	35.92	36.05	14	28.15	33.12	33.16
	36.26	40.82	41.66		32.01	37.31	37.91
	32.36	37.27	39.99		28.60	33.49	36.65
3	33.45	38.27	39.69	15	28.15	34.81	36.49
	37.17	42.19	43.06		32.01	39.41	41.60
	33.83	38.32	41.25		28.60	35.60	40.08
4	32.61	35.72	37.89	16	30.30	35.82	37.08
	36.53	40.76	41.92		34.73	39.71	39.97
	32.91	37.49	40.56		30.39	35.88	37.44
5	25.75	31.76	34.41	17	31.63	36.48	39.33
	29.32	35.99	36.77		34.54	39.33	40.51
	25.92	31.77	35.49		30.86	35.58	38.18
6	26.70	32.27	34.27	18	27.32	31.79	35.38
	31.05	36.12	36.94		30.54	34.77	36.88
	27.00	32.32	34.42		26.82	31.29	34.62
7	32.57	38.11	39.30	19	26.78	33.26	34.59
	36.47	42.27	42.03		31.74	38.31	37.21
	32.58	37.94	41.13		26.95	33.48	34.52
8	22.53	28.74	29.76	20	30.81	36.24	38.52
	27.40	33.49	33.02		34.58	39.75	40.54
	22.48	28.70	29.90		30.59	35.45	38.11
9	31.56	37.21	38.88	21	27.63	32.92	35.22
	35.72	41.34	41.45		31.54	36.40	37.48
	31.38	37.13	38.45		27.53	32.66	35.21
10	31.84	37.13	39.59	22	29.84	34.53	35.92
	35.37	41.24	42.21		33.34	37.83	38.37
	31.23	36.56	39.17		29.22	34.04	36.22
11	28.17	33.43	34.93	23	34.43	39.44	39.71
	32.22	37.21	37.99		37.98	43.46	43.52
	38.34	33.76	36.18		34.12	39.48	42.46
12	32.67	37.65	38.89	24	26.40	31.06	34.10
	36.82	42.34	42.40		29.38	33.50	35.59
	32.35	37.77	39.82		25.29	29.16	32.25
Average				Red	29.15	34.34	36.14
				Green	33.03	38.29	39.05
				Blue	29.90	34.31	36.93

2.1 Hardware Implementation

The proposed ECI hardware architecture is shown in Figure 2.4. Since all missing green channel values are reconstructed before reconstructing missing red and blue channel values, these cannot be executed in parallel. In order to improve its throughput, the proposed datapath has 3 pipeline stages with registers between them. Datapath stage 1 calculates 6 Kr' and 6 Kb' values. Datapath stage 2 calculates 2 Kr'' and 2 Kb'' values. Datapath stage 3 calculates 2 Kr''' , 2 Kb''' , 1 Kr'''' and 1 Kb'''' values.

For ECI operation, the minimum operation window is 7x7. Extending this window to 8x8 enables the hardware to process the 2x2 window in the middle simultaneously. Also, since the Bayer pattern is a repetition of 2x2 window, the proposed hardware avoids reconfiguration for different types of pixels.

Figure 2.5 indicates the data extraction process for 8x8 window. 2x2 window in the middle in Figure 2.5(b) indicates which missing color channels could be reconstructed with the available data. At each time step, hardware reads 4 color filtered pixels through a 32-bit SRAM data bus. To minimize the memory access times, a rotating Block RAM organization is implemented as shown in Figure 2.6. When the image is being demosaicked, 8 out of 10 block RAMs are read, while the remaining 2 block RAMs are filled with next line data. When end of the line is reached, newly filled block RAMs are taken into the read group and 2 block RAMs containing oldest line data are filled with next line data.

The proposed hardware works as follows. After the start signal, the proposed hardware first stores the pixels in the first 8 lines of Bayer filtered image into BRAMs A through H by generating the proper read addresses for off-chip memory, and write addresses and write enables for corresponding BRAMs. Then, an 8x8 pixel register file is filled from BRAMs. Right half of the 8x8 window, shown in Figure 2.7(a), is then passed through Datapath stage 1 in order to calculate Kr' , Kb' values shown in Figure 2.7(b). The Kr' , Kb' values are then passed through Datapath stage 2 in order to obtain Kr'' , Kb'' values and the corresponding missing green values shown in Figure 2.7(c). Datapath stage 3 uses Kr'' , Kb'' values and green values obtained from Datapath stage 2 and Bayer filtered input image to

calculate missing R, B values of the 2x2 window in the middle as shown in Figure 2.7 (d). The results are written into off-chip memory.

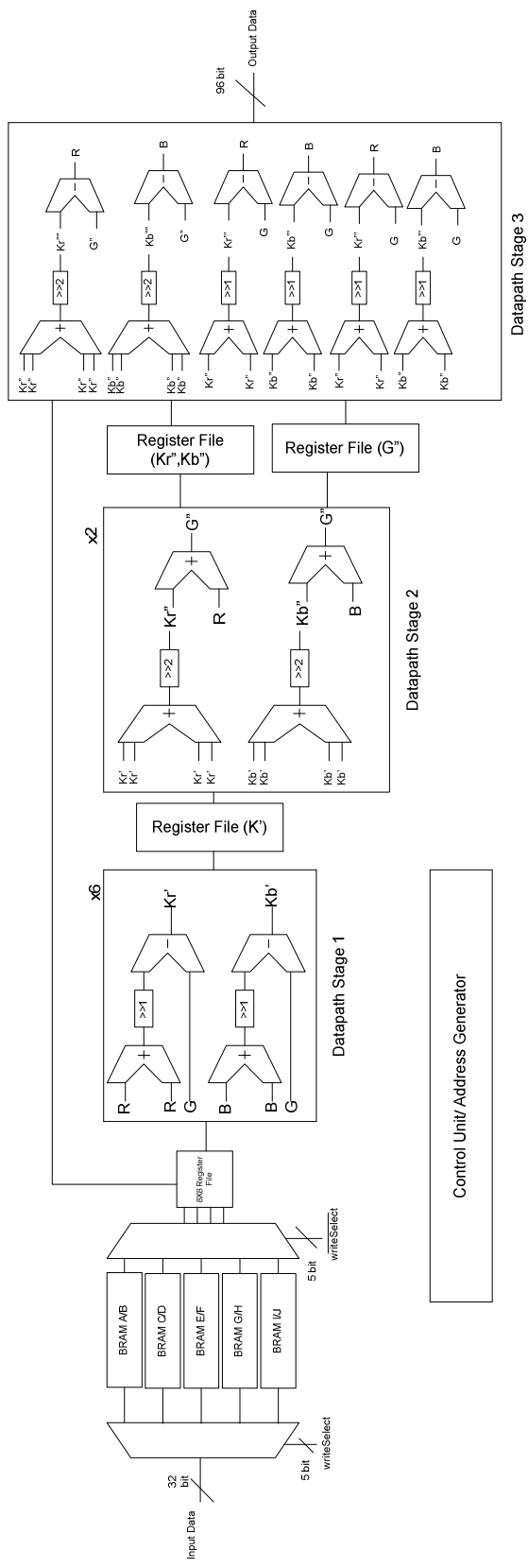


Figure 2.4 Proposed ECI hardware

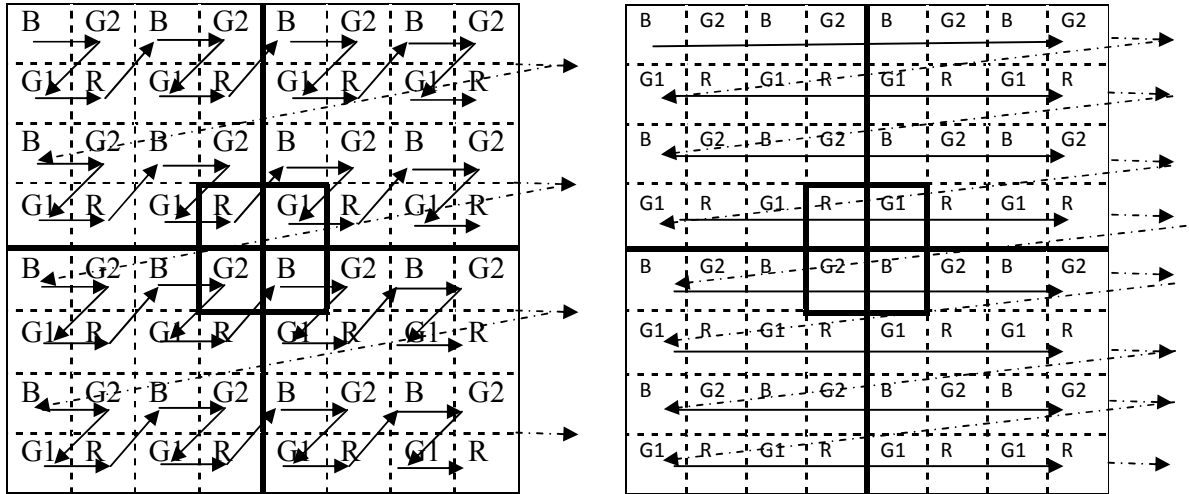
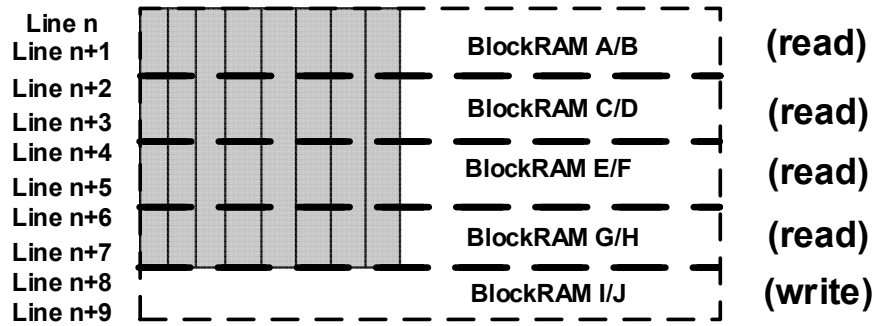
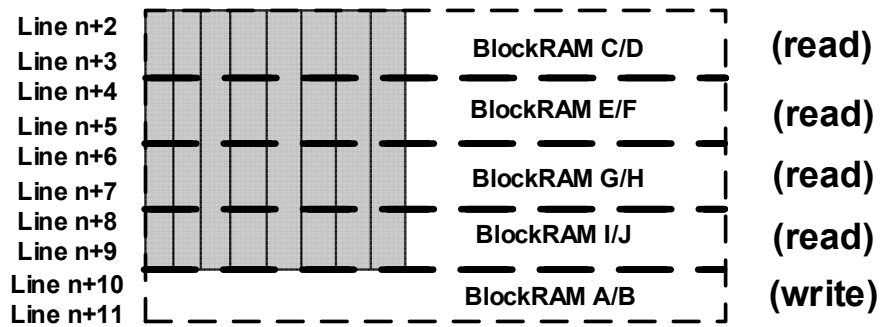


Figure 2.5 (a) Image sensor data scan order for ECI hardware (b) Raster scan order



(a)



(b)

Figure 2.6 Rotating block RAM organization (a) when processing rows n through $n+7$ (b) when processing rows $n+2$ through $n+9$ (8x8 current window is shown in grey)

B	G2	B	G2	B	G2	B	G2
G1	R	G1	R	G1	R	G1	R
B	G2	B	G2	B	G2	B	G2
G1	R	G1	R	G1	R	G1	R
B	G2	B	G2	B	G2	B	G2
G1	R	G1	R	G1	R	G1	R
B	G2	B	G2	B	G2	B	G2
G1	R	G1	R	G1	R	G1	R

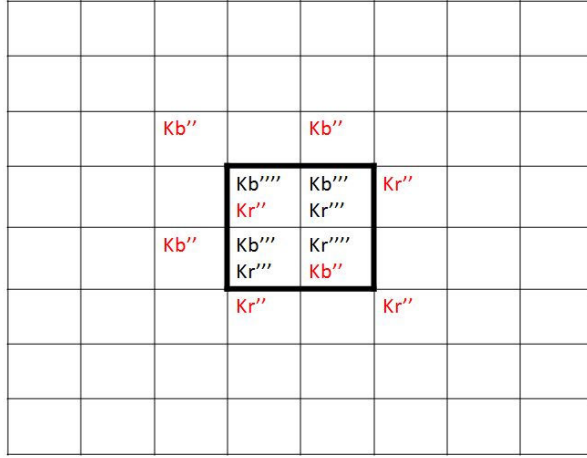
(a)

		Kr' Kb'		Kr' Kb'		Kr' Kb'	
	Kr' Kb'		Kr' Kb'		Kr' Kb'		
		Kr' Kb'		Kr' Kb'		Kr' Kb'	
	Kr' Kb'		Kr' Kb'		Kr' Kb'		
		Kr' Kb'		Kr' Kb'		Kr' Kb'	
	Kr' Kb'		Kr' Kb'		Kr' Kb'		

(b)

		Kr' Kb'		Kr' Kb'		Kr' Kb'	
	Kr' Kb'	Kb''	Kr' Kb'	Kb''	Kr' Kb'		
		Kr' Kb'	Kr'' Kb'	Kr' Kb'	Kr'' Kb'	Kr' Kb'	
	Kr' Kb'	Kb''	Kr' Kb'	Kb''	Kr' Kb'		
		Kr' Kb'	Kr'' Kb'	Kr' Kb'	Kr'' Kb'		
			Kr' Kb'		Kr' Kb'		

(c)



(d)

Figure 2.7 (a) 8x8 mosaicked window for processing the central 2x2 (b) K_R' and K_B' are calculated by bilinearly interpolating red and blue values (c) K_R'' , K_B'' are calculated by averaging surrounding 4 K_R' , K_B' values (d) K_R''' , K_B''' , K_R'' , K_B'' values are calculated for obtaining missing red, blue values

The proposed ECI demosaicing hardware architecture including datapath, control unit and on-chip memory is implemented in Verilog HDL. The resulting Verilog RTL codes are synthesized and placed & routed to a Xilinx Virtex 6 LX240T-3FF1759 FPGA using Xilinx ISE 13.4. The resulting netlist is verified with post place & route simulations using Questa 10.0d. The FPGA implementation occupies 257 slices, and it uses 800 LUTs, 655 registers, and 10 RAMB18E1. The FPGA implementation can work at 76.9 MHz, and it can process 160 1920x1080 full HD images per second.

A real-time bilinear interpolation demosaicing hardware for HD video cameras is proposed in [15]. This hardware is implemented on a Xilinx Virtex 4 XC4VLX25 FPGA. It works at 150 MHz, and it processes 72 full HD images per second. The proposed ECI demosaicing hardware processes 88 more full HD images per second with 6.99 dB, 6.02 dB, and 7.03 dB better reconstructed image quality in red, green, and blue channels, respectively.

Chapter 3

MODIFIED ENHANCED EFFECTIVE COLOR INTERPOLATION HARDWARE

ECI demosaicing algorithm achieves very good performance with relatively low computational cost. One of the shortcomings of ECI is that it does not take into account the locational non-uniformity in the color difference planes. Bilinear interpolation on the color difference channel results in zipper effect artifacts [4]. Enhanced ECI demosaicing algorithm proposed in [4] uses an adaptive weighted interpolation scheme in order to reduce the zipper artifacts around the non-uniform parts of the color difference planes. Enhanced Effective Color Interpolation algorithm has four steps.

1) Step 1: Populate Green Channels

Similar to the original ECI algorithm, initial step is to populate the missing green channels at red and blue pixels using the neighboring color differences (left, right, top, bottom) obtained through bilinear demosaicing. Obtained K_R or K_B is added to the original red or blue value respectively to obtain the missing green value. Assignment of the weights is shown below.

$$\begin{aligned}\alpha_{i-1,j} &= |A_{i-2,j} - A_{i,j}| + |C_{i-1,j} - C_{i+1,j}| \\ \alpha_{i+1,j} &= |A_{i+2,j} - A_{i,j}| + |C_{i-1,j} - C_{i+1,j}| \\ \alpha_{i,j-1} &= |A_{i,j-2} - A_{i,j}| + |C_{i,j-1} - C_{i,j+1}| \\ \alpha_{i,j+1} &= |A_{i,j+2} - A_{i,j}| + |C_{i,j-1} - C_{i,j+1}| \end{aligned} \tag{3.1}$$

where A represents green and C represents red or blue, depending on the pixel being processed.

$$\tilde{K}_{X(i,j)} = \frac{\frac{\tilde{K}_{X(i-1,j)}}{1+\alpha_{i-1,j}} + \frac{\tilde{K}_{X(i+1,j)}}{1+\alpha_{i+1,j}} + \frac{\tilde{K}_{X(i,j-1)}}{1+\alpha_{i,j-1}} + \frac{\tilde{K}_{X(i,j+1)}}{1+\alpha_{i,j+1}}}{\frac{1}{1+\alpha_{i-1,j}} + \frac{1}{1+\alpha_{i+1,j}} + \frac{1}{1+\alpha_{i,j-1}} + \frac{1}{1+\alpha_{i,j+1}}} \quad (3.2)$$

where \tilde{K}_X is K_R or K_B .

2) Step 2: Populate Red Channels at Blue Pixels, Blue Channels at Red Pixels

Missing red channels at blue pixels and blue channels at red pixels are populated using a similar adaptive weighted scheme, with the directions of weights and K values taken from cornering neighbors as shown below.

$$\begin{aligned} \alpha_{i-1,j-1} &= |B_{i-2,j-2} - B_{i,j}| + |R_{i-1,j-1} - R_{i+1,j+1}| \\ \alpha_{i+1,j+1} &= |B_{i+2,j+2} - B_{i,j}| + |R_{i+1,j+1} - R_{i-1,j-1}| \\ \alpha_{i+1,j-1} &= |B_{i+2,j-2} - B_{i,j}| + |R_{i+1,j-1} - R_{i-1,j+1}| \\ \alpha_{i-1,j+1} &= |B_{i-2,j+2} - B_{i,j}| + |R_{i-1,j+1} - R_{i+1,j-1}| \end{aligned} \quad (3.3)$$

$$\tilde{K}_{X(i,j)} = \frac{\frac{\tilde{K}_{X(i-1,j-1)}}{1+\alpha_{i-1,j-1}} + \frac{\tilde{K}_{X(i+1,j+1)}}{1+\alpha_{i+1,j+1}} + \frac{\tilde{K}_{X(i+1,j-1)}}{1+\alpha_{i+1,j-1}} + \frac{\tilde{K}_{X(i-1,j+1)}}{1+\alpha_{i-1,j+1}}}{\frac{1}{1+\alpha_{i-1,j-1}} + \frac{1}{1+\alpha_{i+1,j+1}} + \frac{1}{1+\alpha_{i+1,j-1}} + \frac{1}{1+\alpha_{i-1,j+1}}} \quad (3.4)$$

3) Step 3: Populate Red and Blue Channels at Green Pixels

After Step 2 is completed, K_R and K_B values of the left, right, top and bottom neighbors of green pixels are available. Similarly, all necessary values for the weight calculations are available. Missing blue and red values at green pixels are estimated with calculations similar to Step 1 (A as green and C as red or blue).

4) Step 4: Refinement

After all the interpolations are done, results can be refined by using the weight and adaptive averaging equations in Step 1. This reduces the demosaicing artifacts. As reported in [4], enhanced ECI outperforms most demosaicing algorithms.

TABLE 3.1 AVERAGE PSNR VALUES in dB OVER 30 IMAGES [4]

	Bilinear[2]	Freeman[8]	NEDI[9]	ECI[3]	AP[5]	Enhanced ECI[4]
Red	28.54	34.04	34.53	35.66	37.96	37.99
Green	32.41	39.75	37.22	38.23	40.56	41.64
Blue	28.53	34.19	34.80	35.75	37.22	38.24

3.1 Modified Enhanced ECI Algorithm

Although Enhanced ECI algorithm provides very good results, some modifications to the original algorithm are needed to make the algorithm more suitable for hardware implementation. The most important modification is the elimination of division. Division hardware is costly both in terms of hardware area and execution time. Therefore, we proposed reducing the number of available alpha values to 10 {0, 1, 3, 7, 15, 31, 63, 127, 255, 511}. This enables implementing corresponding $\tilde{K} * \frac{1}{1+\alpha}$ expressions using shift operations instead of costly division operations. Since the alpha values are reduced to a relatively small subset, it is feasible to precalculate the inverses of all possible sums of $\frac{1}{1+\alpha_1} + \frac{1}{1+\alpha_2} + \frac{1}{1+\alpha_3} + \frac{1}{1+\alpha_4}$ and store them as 18 bit fixed point values with 9 bit used for decimal part in a LUT with 440 values.

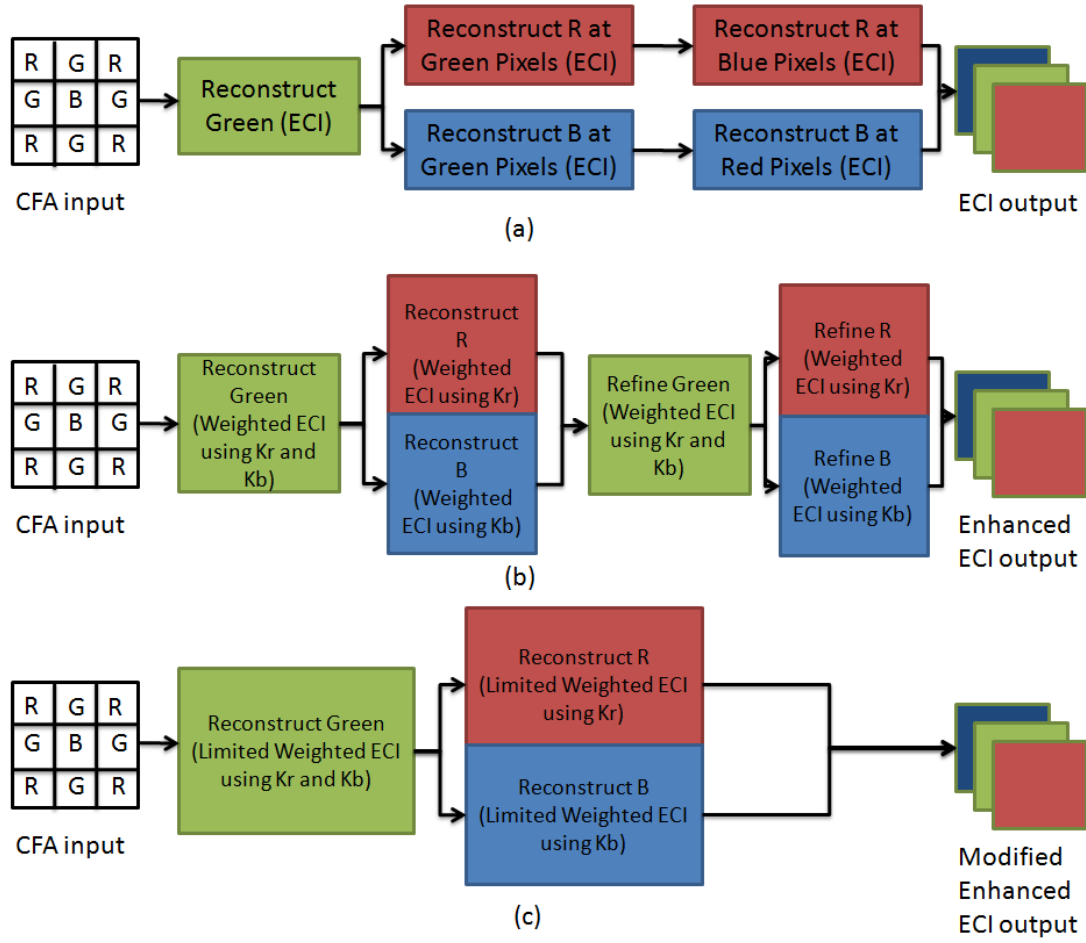


Figure 3.1 (a) ECI flow (b) Enhanced ECI flow (c) Modified Enhanced ECI flow

TABLE 3.2 SCALED RANGE FOR α VALUES

Actual α values	Reduced α values	Reduced $\frac{1}{1+\alpha}$ values	Corresponding shifts
0	0	1	>>>>0
1...2	1	1/2	>>>>1
3...4	3	1/4	>>>>2
5...10	7	1/8	>>>>3
11...22	15	1/16	>>>>4
23...47	31	1/32	>>>>5
48...95	63	1/64	>>>>6
96...193	127	1/128	>>>>7
194...383	255	1/256	>>>>8
384...511	511	1/512	>>>>9

Therefore, K estimator is calculated as shown below

$$\tilde{K}_X = (\tilde{K}_1 * \frac{1}{1+\alpha_1} + \tilde{K}_2 * \frac{1}{1+\alpha_2} + \tilde{K}_3 * \frac{1}{1+\alpha_3} + \tilde{K}_4 * \frac{1}{1+\alpha_4}) * \frac{1}{\frac{1}{1+\alpha_1} + \frac{1}{1+\alpha_2} + \frac{1}{1+\alpha_3} + \frac{1}{1+\alpha_4}} \quad (3.6)$$

where boxed term is calculated using LUT. Also, the refinement step is not performed for increasing the processing speed. Quality losses in image reconstruction because of these simplifications are shown in Figure 3.2, Table 3.3 and Table 3.4.



(a) EECI with refinement



(b) EECI without refinement



(c) EECI without refinement, reduced alpha



(d) EECI without refinement, reduced alpha, fixed point

Figure 3.2 An example reconstructed image by EECI algorithms

TABLE 3.3 PSNR (dB) COMPARISON OF ENHANCED ECI ALGORITHMS

Image	EECI with refinement	EECI without refinement	EECI without refinement, reduced alpha	EECI without refinement, reduced alpha, fixed point
1	36.96	34.92	35.11	34.65
	40.62	36.51	36.39	36.39
	37.41	35.18	35.45	34.89
2	38.27	38.29	37.76	37.72
	43.67	42.43	42.28	42.29
	41.41	41.02	41.18	40.21
3	41.98	41.46	41.17	40.71
	45.60	43.82	43.67	43.65
	41.07	40.57	40.59	39.83
4	39.58	39.48	38.92	39.32
	42.06	41.47	41.37	41.28
	41.12	40.75	41.04	40.31
5	37.43	36.33	36.29	35.88
	40.94	37.75	37.49	37.51
	36.73	35.71	35.63	35.33
6	37.21	35.63	35.84	35.36
	41.12	37.60	37.48	37.48
	37.10	35.39	35.52	35.12
7	41.94	41.75	41.48	40.94
	45.40	43.93	43.72	43.74
	41.82	41.63	41.25	40.65
8	34.48	32.93	32.94	32.68
	38.44	35.08	34.92	34.92
	34.29	32.70	32.79	32.46
9	42.75	41.57	41.39	41.27
	45.41	42.99	42.83	42.69
	40.94	40.11	40.11	39.70
10	42.63	41.83	41.48	41.56
	45.16	43.20	42.95	42.80
	41.09	40.38	40.32	39.92
11	37.97	36.75	36.77	36.40
	42.24	38.84	38.69	38.70
	39.26	37.53	37.72	37.06
12	41.79	40.82	40.67	40.06
	45.78	43.59	43.41	43.40
	41.81	40.79	40.77	39.99
13	34.13	31.65	32.04	31.48
	36.71	32.40	32.30	32.30
	33.14	30.96	31.22	30.78

TABLE 3.3 (cont) PSNR (dB) COMPARISON OF ENHANCED ECI ALGORITHMS

Image	EECI with refinement	EECI without refinement	EECI without refinement, reduced alpha	EECI without refinement, reduced alpha, fixed point
14	35.20	35.34	35.00	35.04
	40.48	38.85	38.68	38.69
	36.93	36.54	36.31	36.17
15	40.19	37.77	37.39	37.36
	43.98	41.07	40.94	40.93
	40.16	39.39	39.59	38.82
16	42.28	38.70	38.87	38.21
	44.17	40.88	40.75	40.75
	39.62	38.61	38.72	38.09
17	38.05	40.66	40.79	40.41
	39.78	40.87	40.74	40.66
	35.89	38.58	38.75	38.28
18	39.96	36.45	36.55	36.36
	42.80	36.62	36.51	36.48
	39.00	34.51	34.67	34.34
19	41.61	38.45	38.51	38.30
	44.38	39.84	39.71	39.64
	39.63	37.71	37.85	37.45
20	41.61	40.58	40.67	40.21
	44.38	41.74	41.60	41.58
	39.63	38.79	38.77	38.27
21	38.61	36.85	37.13	36.48
	41.78	38.14	38.02	38.03
	37.67	36.05	36.18	35.70
22	37.99	37.37	37.15	37.07
	41.09	39.39	39.30	39.32
	37.55	36.89	36.77	36.50
23	42.21	42.17	41.72	41.28
	45.56	44.40	44.22	44.24
	41.91	41.92	41.78	40.87
24	35.15	34.25	34.28	34.11
	37.85	35.31	35.20	35.20
	33.07	32.05	32.16	31.87
Ave. R	39.17	38.00	37.91	37.62
Ave. G	42.48	39.86	39.72	39.69
Ave. B	38.68	37.66	37.71	37.19

TABLE 3.4 PSNR (dB) COMPARISON OF IMAGE DEMOSAICING ALGORITHMS

Image	Laplacian	ECI	AP single iteration	Enhanced ECI	Modified Enhanced ECI
1	30.86	33.02	30.70	36.96	34.65
	34.71	35.66	40.42	40.62	36.39
	30.80	33.50	30.82	37.41	34.89
2	35.92	36.05	35.93	38.27	37.72
	40.82	41.66	42.43	43.67	42.29
	37.27	39.99	37.28	41.41	40.21
3	38.27	39.69	38.13	41.98	40.71
	42.19	43.06	43.51	45.60	43.65
	38.32	41.25	38.85	41.07	39.83
4	35.72	37.89	36.49	39.58	39.32
	40.76	41.92	43.78	42.06	41.28
	37.49	40.56	38.05	41.12	40.31
5	31.76	34.41	31.49	37.43	35.88
	35.99	36.77	39.70	40.94	37.51
	31.77	35.49	31.84	36.73	35.33
6	32.27	34.27	31.91	37.21	35.36
	36.12	36.94	41.48	41.12	37.48
	32.32	34.42	32.42	37.10	35.12
7	38.11	39.30	37.55	41.94	40.94
	42.27	42.03	43.90	45.40	43.74
	37.94	41.13	37.57	41.82	40.65
8	28.74	29.76	27.50	34.48	32.68
	33.49	33.02	38.55	38.44	34.92
	28.70	29.90	27.48	34.29	32.46
9	37.21	38.88	36.30	42.75	41.27
	41.34	41.45	43.40	45.41	42.69
	37.13	38.45	36.58	40.94	39.70
10	37.13	39.59	37.30	42.63	41.56
	41.24	42.21	44.31	45.16	42.80
	36.56	39.17	36.56	41.09	39.92
11	33.43	34.93	33.26	37.97	36.40
	37.21	37.99	41.63	42.24	38.70
	33.76	36.18	33.79	39.26	37.06
12	37.65	38.89	37.57	41.79	40.06
	42.34	42.40	45.16	45.78	43.40
	37.77	39.82	37.24	41.81	39.99
13	27.67	31.13	28.79	34.13	31.48
	30.68	32.52	36.84	36.71	32.30
	27.43	30.82	28.64	33.14	30.78

TABLE 3.4 (cont) PSNR (dB) COMPARISON OF IMAGE DEMOSAICING ALGORITHMS

Image	Laplacian	ECI	AP single iteration	Enhanced ECI	Modified Enhanced ECI
14	33.12	33.16	32.54	35.20	35.04
	37.31	37.91	38.09	40.48	38.69
	33.49	36.65	33.51	36.93	36.17
15	34.81	36.49	36.24	40.19	37.36
	39.41	41.60	42.26	43.98	40.93
	35.60	40.08	37.36	40.16	38.82
16	35.82	37.08	35.29	42.28	38.21
	39.71	39.97	44.76	44.17	40.75
	35.88	37.44	35.58	39.62	38.09
17	36.48	39.33	36.84	38.05	40.41
	39.33	40.51	43.03	39.78	40.66
	35.58	38.18	36.14	35.89	38.28
18	31.79	35.38	32.71	39.96	36.36
	34.77	36.88	39.32	42.80	36.48
	31.29	34.62	32.37	39.00	34.34
19	33.26	34.59	31.60	41.61	38.30
	38.31	37.21	42.08	44.38	39.64
	33.48	34.52	32.17	39.63	37.45
20	36.24	38.52	36.07	41.61	40.21
	39.75	40.54	43.45	44.38	41.58
	35.45	38.11	35.67	39.63	38.27
21	32.92	35.22	32.87	38.61	36.48
	36.40	37.48	41.56	41.78	38.03
	32.66	35.21	32.86	37.67	35.70
22	34.53	35.92	34.37	37.99	37.07
	37.83	38.37	39.73	41.09	39.32
	34.04	36.22	33.66	37.55	36.50
23	39.44	39.71	39.01	42.21	41.28
	43.46	43.52	43.98	45.56	44.24
	39.48	42.46	38.56	41.91	40.87
24	31.06	34.10	32.13	35.15	34.11
	33.50	35.59	37.46	37.85	35.20
	29.16	32.25	29.89	33.07	31.87
Ave. R	34.34	36.14	34.27	39.17	37.62
Ave. G	38.29	39.05	41.70	42.48	39.69
Ave. B	34.31	36.93	34.37	38.68	37.19

For enhanced ECI, estimating the missing green channel values requires 31 additions, 8 absolute value operations, 9 multiplications, and 4 shift operations. Estimating missing red or blue channel values require 23 additions, 8 absolute value operations, 9 multiplications. Refinement step for a single color channel requires 4 additions and 5 multiplications. Therefore, for an $M \times N$ image, enhanced ECI algorithm requires $58MN$ additions, $16MN$ absolute value operations, $2MN$ shift operations, and $28MN$ multiplications.

For the modified Enhanced ECI, estimating the missing green channel values requires 27 additions, 8 absolute value operations, 1 multiplication, 8 shift operations, and 16 LUT operations. Estimating the missing red or blue channel values requires 19 additions, 8 absolute value operations, 1 multiplication, 4 shift operations, and 16 LUT operations.

TABLE 3.5 COMPLEXITY COMPARISON

Method	Addition/ Subtraction	Absolute	Shift	Multiplication
Original ECI	10.0MN		4.0MN	
Original AP (3 iterations)	391.5MN	2.0MN	3.5MN	384.0MN
Original AP (5 iterations)	583.5MN	2.0MN	3.5MN	576.0MN
Enhanced ECI	58.0MN	16.0MN	2.0MN	28.0MN
Modified Enhanced ECI	42.0MN	16.0MN	10.0MN	2.0MN

3.2 Hardware Implementation

The proposed modified EECI hardware is shown in Figure 3.6. The proposed hardware includes control unit, 3 stage pipelined datapath, and several on-chip memories. Control unit determines address signals for read/write operations for on-chip Block RAMs and off-chip memory. The proposed hardware reads the input color filtered image from off-

chip memory through a 32-bit bus. The outputs are written to off-chip memory through three 32-bit busses, one for each color channel.

Datapath stage 1 produces 6 green values by using the proposed modified EECl algorithm. Its inputs and outputs are shown in Figure 3.3. Datapath stage 2 produces 2 red and 2 blue values. Its inputs and outputs are shown in Figure 3.4. Datapath stage 3 produces 2 red and 2 blue values, completing the reconstructed 2x2 window in the middle. Its inputs and outputs are shown in Figure 3.5. K values and weights are calculated using the hardware shown in Figure 3.7. For obtaining the K value specified in Equation 3.6, the proposed hardware uses K calculator shown in Figure 3.8. α values are passed through a Look up Table (LUT). Reduced alpha values are passed through 2 different LUTs. LUT inverter, shown in light blue in Figure 3.8, determines the inverse of the reduced $\frac{1}{1+\alpha}$ value. LUT Shift_size, shown in light brown in Figure 3.8, determines the number of bits corresponding k values to be shifted.

G	B	G	B	G	B
R	G	R	G	R	G
G	B	G	B	G	B
R	G	R	G	R	G
G	B	G	B	G	B
R	G	R	G	R	G
G	B	G	B	G	B
R	G	R	G	R	G
G	B	G	B	G	B
R	G	R	G	R	G

			G		
		G			
			G		
		G			
			G		
		G			

Figure 3.3 Datapath stage1 inputs and outputs

G	B	G	B	G	B
R	G	R	G	R	G
G	B,G	G	B,G	G	B,G
R,G	G	R,G	G	R,G	G
G	B,G	G	B,G	G	B,G
R,G	G	R,G	G	R,G	G
G	B,G	G	B,G	G	B,G
R,G	G	R,G	G	R,G	G
G	B	G	B	G	B
R	G	R	G	R	G

		B			
			R		
		B			
			R		

Figure 3.4 Datapath stage2 inputs and outputs

G	B	G	B	G	B
R	G	R	G	R	G
G	B	G	B	G	B
R,G,B	G	R,G,B	G	R,G,B	G
G	B,G,R	G	B,G,R	G	B,G,R
R,G,B	G	R,G,B	G	R,G,B	G
G	B,G,R	G	B,G,R	G	B,G,R
R	G	R	G	R	G
G	B	G	B	G	B
R	G	R	G	R	G

		R,B			
			R,B		

Figure 3.5 Datapath stage3 inputs and outputs

The proposed hardware architecture is implemented using Verilog HDL. The Verilog RTL code is synthesized and mapped to Xilinx Virtex 6 LX240T-3FF1759 FPGA with a speed rating of -3 using Xilinx ISE 13.4. The resulting netlist is verified with post place &

route simulations using Questa 10.0d. The proposed FPGA implementation occupies 6649 slices, and it uses 3046 registers, 17446 LUTs and 18569 LUT flip-flop pairs. The FPGA implementation can operate at 62.50 MHz, and it can process 118 full HD or 29 quad full HD images per second.

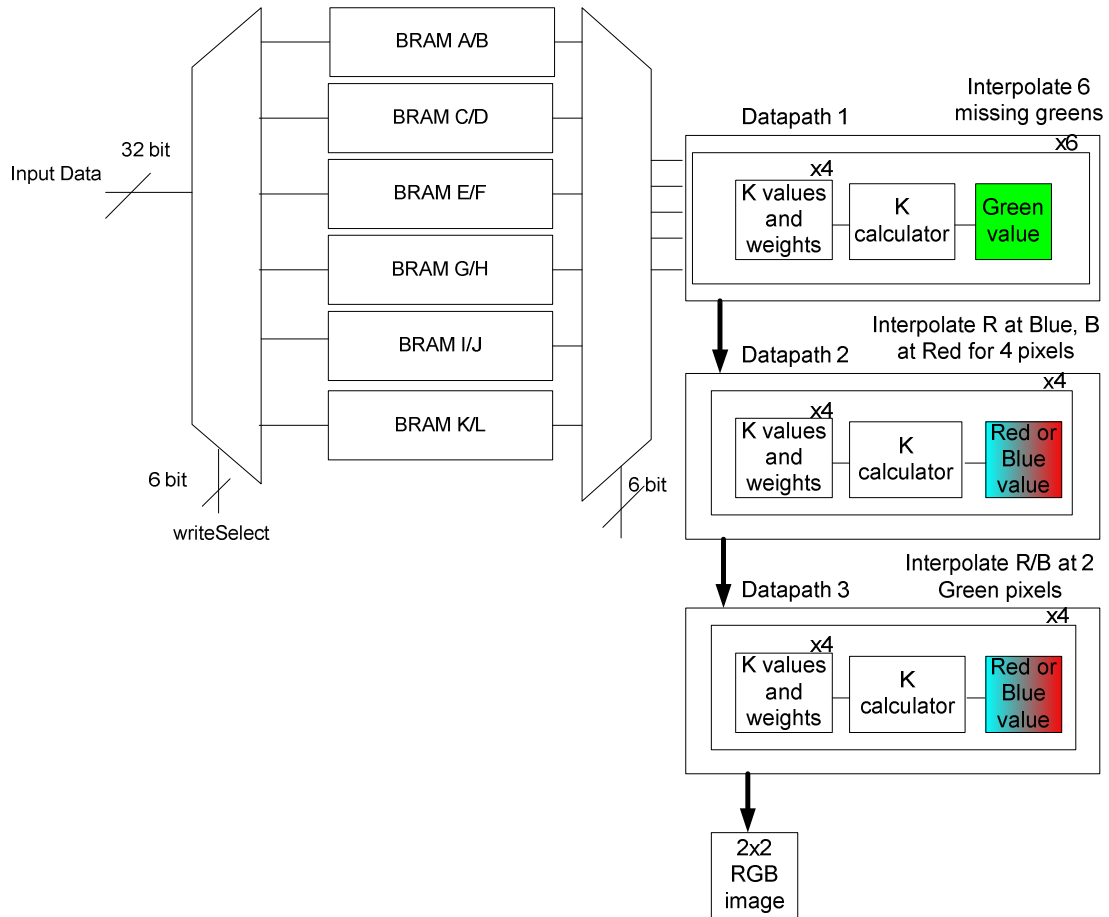


Figure 3.6 Datapath of the proposed modified EECI hardware

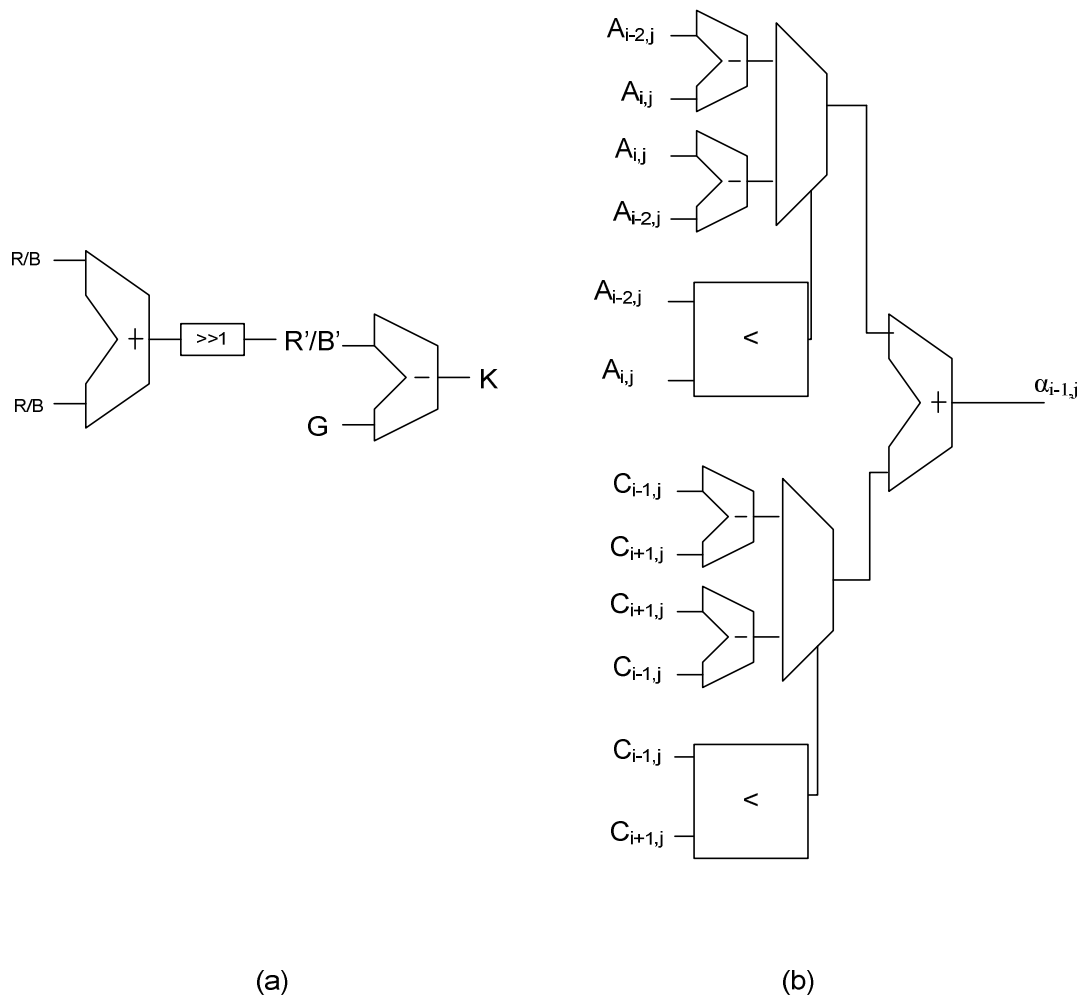


Figure 3.7 (a) K value calculation (b) weight calculation

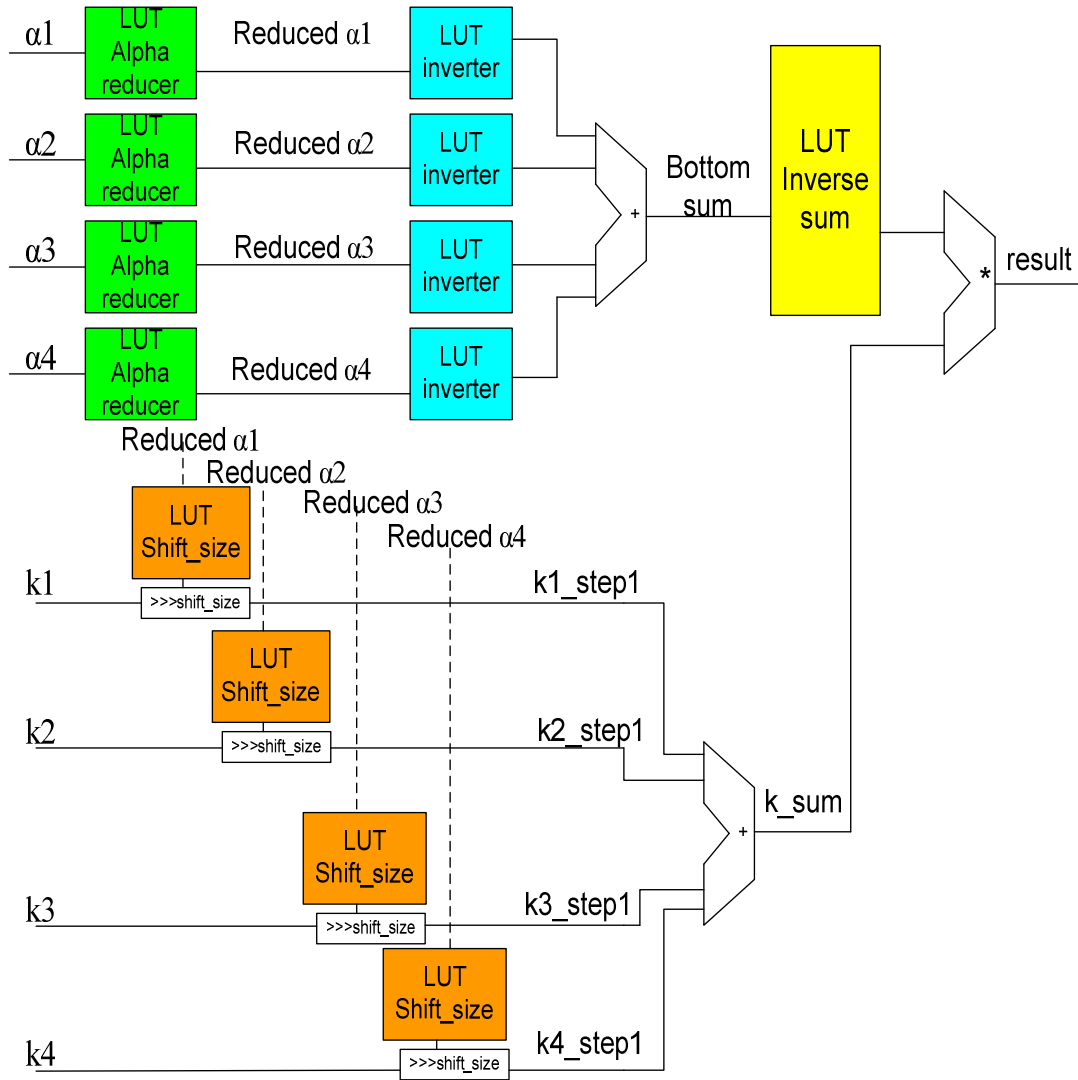


Figure 3.8 K calculator used in the proposed modified EECI hardware

Chapter 4

HYBRID EFFECTIVE COLOR INTERPOLATION AND ALTERNATING PROJECTIONS HARDWARE

Alternating Projections (AP), which is also known as Projection onto Convex Sets (POCS), is a very complex and high quality demosaicing algorithm proposed in [5]. AP algorithm is shown in Figure 4.1. It uses the correlation between R, G, B channels. It uses the higher correlation available in the high frequency components when reconstructing the high frequency parts of the missing values. Therefore, it performs better than the demosaicing algorithms which do not perform effective interpolation around the edges present in the image that correspond to the high frequency parts of the image. AP algorithm usually converges after five iterations. Its convergence property is shown in [5].

AP algorithm defines two constraint sets on the reconstructed image.

- A. Observation Projection: Interpolated image must be consistent with the captured data
- B. Detail Projection: High-frequency components of the red and blue channels must be similar to that of green channel

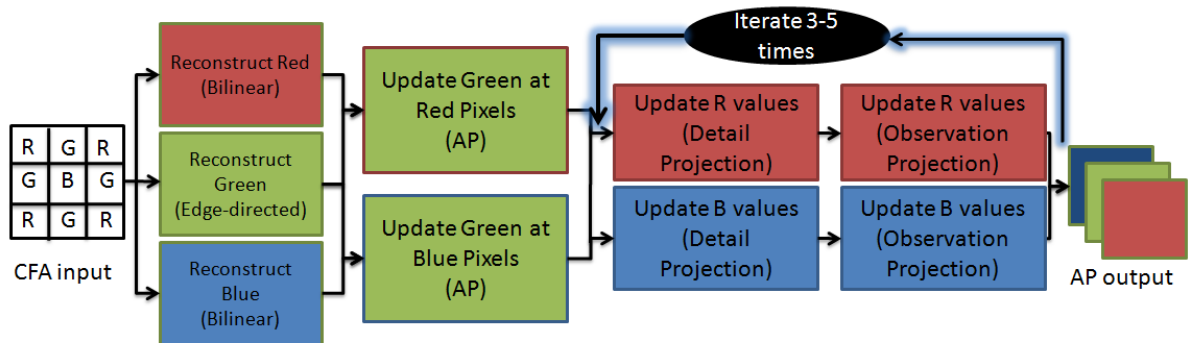


Figure 4.1 Alternating Projections algorithm

Detail projection is obtained by using combinations of low-pass ($\begin{bmatrix} 1 & 2 & 1 \end{bmatrix}/4$) and high-pass ($\begin{bmatrix} 1 & -2 & 1 \end{bmatrix}/4$) filters on red, blue and green channels. Resulting subbands are named as

- a. LL, where both rows and columns are low-pass filtered
- b. LH, where rows are low-pass filtered and columns are high-pass filtered
- c. HL, where rows are high-pass filtered and columns are low-pass filtered
- d. HH, where both rows and columns are high-pass filtered

The missing pixels are reconstructed by inverse filtering using the appropriate subbands.

AP algorithm works as follows.

- 1) Green channel is reconstructed by edge-directed demosaicing. Red and blue channels are reconstructed by bilinear demosaicing.
- 2) Down-sampled red and down-sampled blue channels are formed.
- 3) Green channel is then reconstructed using LL of green and LH, HL and HH of down-sampled red or down-sampled blue respectively. The reconstructed green values are replaced with the original green values available in the CFA input image.
- 4) After green channel is updated using detail projection and observation projection, similar process is performed for red and blue channels.
 - a. Red channel is reconstructed by inverse filtering LL subband of red and LH, HL and HH subbands of green.
 - b. Blue channel is reconstructed by inverse filtering LL subband of blue and LH, HL and HH subbands of green.
 - c. The reconstructed red and blue values are replaced with the original ones available in the CFA input image.
- 5) Step 4 is repeated 4-5 times.

AP algorithm is an iterative algorithm. The PSNR performance of AP algorithm for an example image for different number of iterations is given in Table 4.1. As seen in Figure 4.2, AP algorithm tends to converge at around 4-5 iterations for 24 images in Kodak Suite.

TABLE 4.1 PSNR (dB) PERFORMANCE OF DIFFERENT AP ITERATIONS FOR LIGHTHOUSE IMAGE

	Green edge-d. R, B bilinear	Original AP Iteration (0)	Original AP Iteration (1)	Original AP Iteration (2)	Original AP Iteration (3)	Original AP Iteration (4)	Original AP Iteration (5)	Original AP Iteration (6)
R	26.8048	26.8048	31.6216	34.3668	36.1292	37.1546	37.7079	37.9942
G	38.5888	41.6238	41.6238	41.6238	41.6238	41.6238	41.6238	41.6238
B	27.2719	27.2719	32.8222	35.9908	37.6613	38.2807	38.4171	38.3905
cPSNR	30.8885	31.9002	35.3559	37.3271	38.4714	39.01987	39.2496	39.3362
Diff. (dB)	--	1.0117	4.4674	6.4386	7.5829	8.1312	8.3611	8.4477

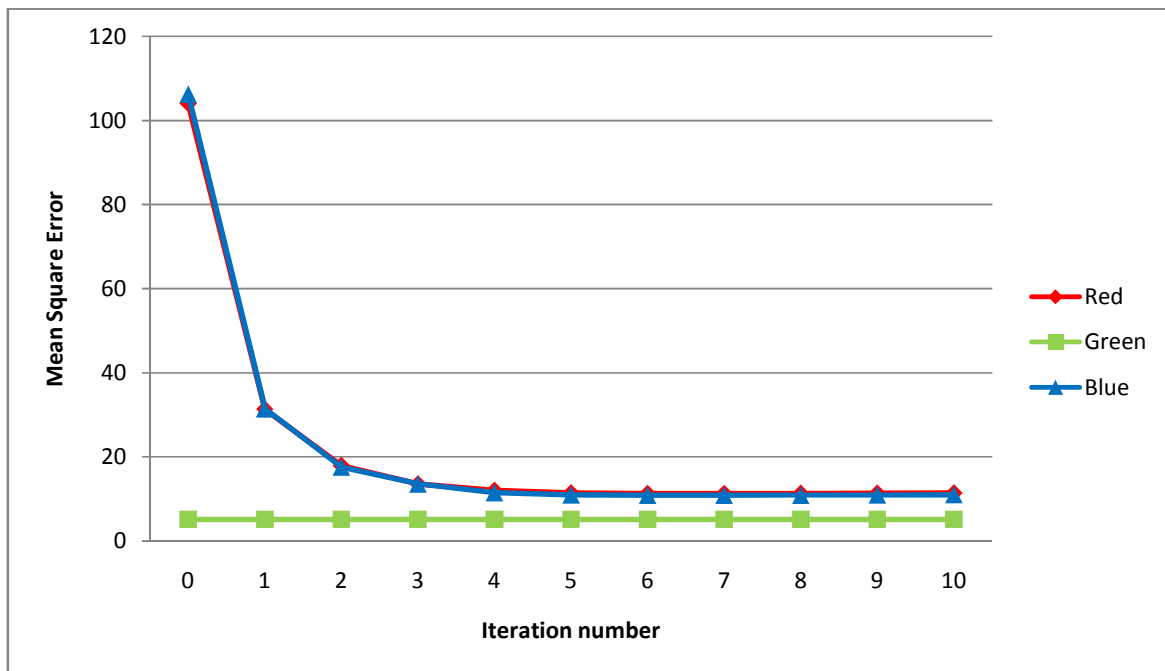


Figure 4.2 MSE performance of different AP iterations for 24 Kodak Suite images

Low-pass and high-pass filters in AP algorithm are separable. In addition, in both low-pass and high-pass filters, there are divisions by powers of 2 which can be implemented with shifter hardware. However, because of the relatively large number of iterations and the data dependencies between them, AP hardware implementation will be quite complex.

4.1 Hybrid ECI and AP Algorithm

One of the major bottlenecks in AP algorithm is the number of iterations. Although green channel is iterated only once, AP algorithm achieves good performance for green channel as edge-directed interpolation before POCS part has relatively high accuracy. Since red and blue channels are initially interpolated using bilinear interpolation, iterations for these channels provide large improvements. As it can be seen in Table 4.1, at least 3 iterations are needed in order to obtain good performance.

In order to make AP algorithm more suitable for hardware implementation, several modifications are proposed. In order to reduce the number of iterations to zero, the accuracies of red and blue channels are increased by using ECI algorithm instead of bilinear interpolation, and the accuracy of green channel is increased by using ECI algorithm instead of edge-directed interpolation before POCS part. Since there are no dependencies between them, ECI algorithm for red and blue values can be executed in parallel with the POCS part for green values. The proposed hybrid ECI and AP algorithm is shown in Figure 4.3.

The PSNR performances of the AP algorithm with multiple iterations and the proposed hybrid ECI and AP algorithm for an example image are given in Table 4.2.

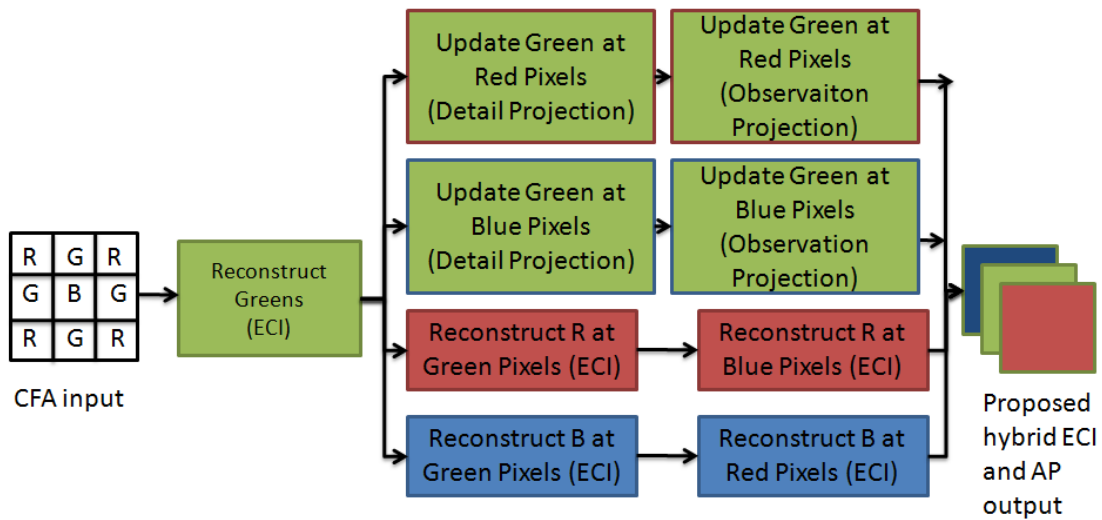


Figure 4.3 Proposed hybrid ECI and AP algorithm

TABLE 4.2 PSNR (dB) COMPARISON OF DEMOSAICING ALGORITHMS FOR LIGHTHOUSE IMAGE

	Green edge-d. R, B bilinear	Green edge-d. R, B ECI	Original AP Iteration (0)	Original AP Iteration (1)	Original AP Iteration (2)	Proposed hybrid ECI and AP
R	26.8048	36.0716	26.8048	31.6216	34.3668	36.14
G	38.5888	38.5336	41.6238	41.6238	41.6238	40.93
B	27.2719	35.9383	27.2719	32.8222	35.9908	36.93
Average	28.6438	36.6957	28.7182	33.7014	36.4055	37.48
Diff.(dB)	--	--	0.0000	4.9832	7.6873	8.76



(a) Original

(b) ECI

(c) AP (5 iterations)

(d) Hybrid ECI and
AP

Figure 4.4 An example reconstructed image by demosaicing algorithms

TABLE 4.3 PSNR (dB) COMPARISON OF IMAGE DEMOSAICING ALGORITHMS

Image	Laplacian	ECI	AP single iteration	Proposed hybrid ECI and AP
1	30.86	33.02	30.70	33.02
	34.71	35.66	40.42	39.61
	30.80	33.50	30.82	33.50
2	35.92	36.05	35.93	36.05
	40.82	41.66	42.43	42.88
	37.27	39.99	37.28	39.99
3	38.27	39.69	38.13	39.69
	42.19	43.06	43.51	42.36
	38.32	41.25	38.85	41.25
4	35.72	37.89	36.49	37.89
	40.76	41.92	43.78	42.08
	37.49	40.56	38.05	40.56
5	31.76	34.41	31.49	34.41
	35.99	36.77	39.70	38.90
	31.77	35.49	31.84	35.49
6	32.27	34.27	31.91	34.27
	36.12	36.94	41.48	40.29
	32.32	34.42	32.42	34.41
7	38.11	39.30	37.55	39.30
	42.27	42.03	43.90	42.62
	37.94	41.13	37.57	41.13
8	28.74	29.76	27.50	29.76
	33.49	33.02	38.55	36.04
	28.70	29.90	27.48	29.90
9	37.21	38.88	36.30	38.88
	41.34	41.45	43.40	43.07
	37.13	38.45	36.58	38.45
10	37.13	39.59	37.30	39.59
	41.24	42.21	44.31	44.00
	36.56	39.17	36.56	39.17
11	33.43	34.93	33.26	34.92
	37.21	37.99	41.63	41.30
	33.76	36.18	33.79	36.18
12	37.65	38.89	37.57	38.89
	42.34	42.40	45.16	43.60
	37.77	39.82	37.24	39.82
13	27.67	31.13	28.79	31.13
	30.68	32.52	36.84	36.98
	27.43	30.82	28.64	30.82

TABLE 4.3 (cont). PSNR (dB) COMPARISON OF IMAGE DEMOSAICING ALGORITHMS

Image	Laplacian	ECI	AP single iteration	Proposed hybrid ECI and AP
14	33.12	33.16	32.54	33.16
	37.31	37.91	38.09	37.69
	33.49	36.65	33.51	36.65
15	34.81	36.49	36.24	36.49
	39.41	41.60	42.26	42.16
	35.60	40.08	37.36	40.08
16	35.82	37.08	35.29	37.08
	39.71	39.97	44.76	42.70
	35.88	37.44	35.58	37.44
17	36.48	39.33	36.84	39.33
	39.33	40.51	43.03	43.77
	35.58	38.18	36.14	38.18
18	31.79	35.38	32.71	35.38
	34.77	36.88	39.32	39.93
	31.29	34.62	32.37	34.62
19	33.26	34.59	31.60	34.59
	38.31	37.21	42.08	40.31
	33.48	34.52	32.17	34.52
20	36.24	38.52	36.07	38.51
	39.75	40.54	43.45	42.15
	35.45	38.11	35.67	38.11
21	32.92	35.22	32.87	35.22
	36.40	37.48	41.56	40.53
	32.66	35.21	32.86	35.21
22	34.53	35.92	34.37	35.92
	37.83	38.37	39.73	38.84
	34.04	36.22	33.66	36.22
23	39.44	39.71	39.01	39.71
	43.46	43.52	43.98	43.18
	39.48	42.46	38.56	42.46
24	31.06	34.10	32.13	34.10
	33.50	35.59	37.46	37.36
	29.16	32.25	29.89	32.25
Average R	34.34	36.14	34.27	36.14
Average G	38.29	39.05	41.70	40.93
Average B	34.31	36.93	34.37	36.93

For an $M \times N$ image, the proposed hybrid ECI and AP algorithm first calculates the missing green values using ECI algorithm. This requires 2 additions and 1 shift operations for $MN/2$ of the pixels. It then performs POCS for green channel. This requires 96.0 MN additions and 96.0 MN multiplications. It calculates missing red and blue values using ECI algorithm. This requires 4.0 MN additions and 1.5 MN shift operations. Therefore, the proposed hybrid ECI and AP algorithm requires 106.0 MN additions, 4.0 MN shift operations and 96.0 MN multiplications.

TABLE 4.4 COMPLEXITY COMPARISON

Method	Addition/ Subtraction	Absolute	Shift	Multiplication
Original ECI	10.0MN		4.0MN	
Original AP (3 iterations)	391.5MN	2.0MN	3.5MN	384.0MN
Original AP (5 iterations)	583.5MN	2.0MN	3.5MN	576.0MN
Proposed hybrid ECI and AP	106.0MN		4.0MN	96.0MN

4.2 Hardware Implementation

The proposed hybrid ECI and AP demosaicing hardware is shown in Figure 4.5. It includes 40 filter hardware (2 sets of 5 HH, 5 HL, 5 LH, 5 LL) and 2 inverse filter hardware (2 sets of 1 inverse HH, 1 inverse HL, 1 inverse LH, 1 inverse LL). The hardware implementation of a LH and an inverse LH filter are shown in Figure 4.6. The multiplications with constant coefficients are implemented with shift and addition operations in the hardware. For example, multiplication with 6 ($4+2$) is implemented with 2 shift and 1 addition operations.

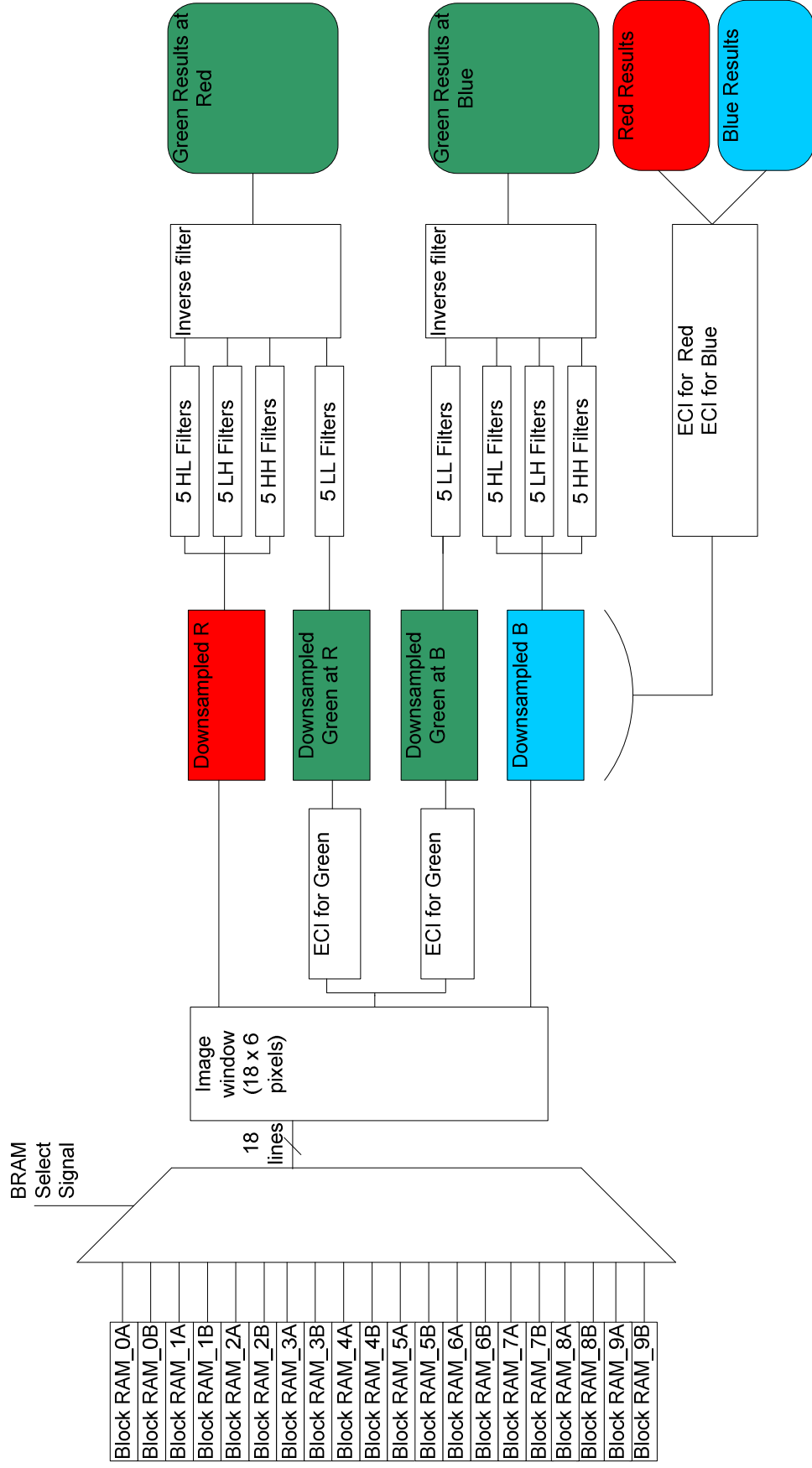
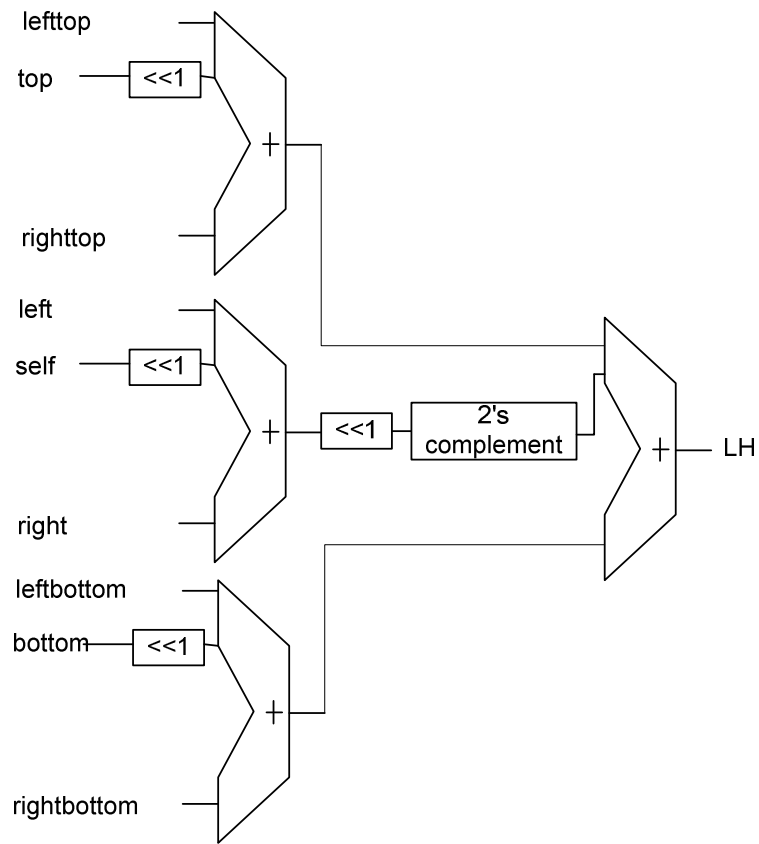
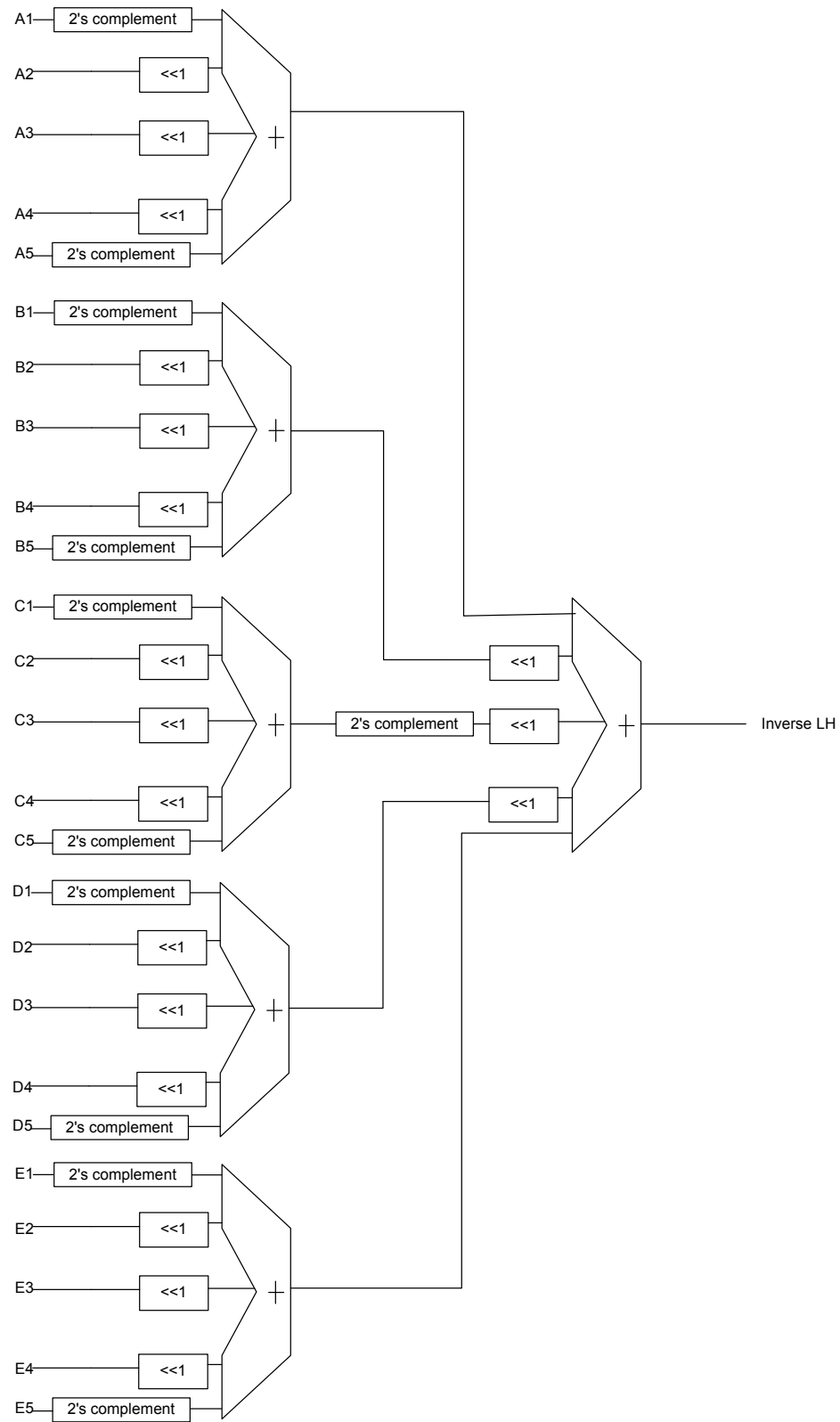


Figure 4.5 Proposed hybrid ECI and AP demosaicing hardware



(a)



(b)

Figure 4.6 (a) Low High filter for 3x3 window (b) Inverse Low High filter for 5x5 window

The proposed hardware reads the CFA input image pixels from off-chip memory through a 32-bit bus and stores them to on-chip BRAMs. After 18 lines are stored to 18 BRAMs, the hardware starts working while the remaining 2 BRAMs are being filled from the off-chip memory using rotating BRAM organization shown in Figure 2.6. The proposed hardware writes the reconstructed pixels to off-chip memory through four 24-bit buses.

The proposed hardware architecture is implemented using Verilog HDL. The Verilog RTL code is synthesized and mapped to Xilinx Virtex 6 LX240T-3FF1759 FPGA with a speed rating of -3 using Xilinx ISE 13.4. The resulting netlist is verified with post place & route simulations using Questa 10.0d. The proposed hardware occupies 2562 slices, and it uses 6388 registers, 8142 LUTs and 9333 LUT flip flop pairs. The proposed hardware uses 20 RAMB18E1 block RAMs. The proposed FPGA implementation works at 62.50 MHz. It is capable of processing one full HD image in 522,129 clock cycles including the initial loading of on-chip memory. Therefore, it is capable of processing 119 full HD or 29 quad full HD images per second.

Chapter 5

CONCLUSION AND FUTURE WORK

In this thesis, first, we proposed a high performance hardware architecture for ECI demosaicing algorithm. The proposed hardware architecture is implemented using Verilog HDL on a Xilinx Virtex 6 FPGA. The FPGA implementation is capable of processing 160 full HD images per second or 40 quad HD images per second.

Then, we proposed a modified version of EECI algorithm. We also proposed a high performance hardware architecture for this demosaicing algorithm. The proposed hardware architecture is implemented using Verilog HDL on a Xilinx Virtex 6 FPGA. The FPGA implementation is capable of processing 118 full HD images per second or 29 quad HD images per second.

Finally, we proposed a hybrid ECI and AP demosaicing algorithm. We also proposed a high performance hardware architecture for this demosaicing algorithm. The proposed hardware architecture is implemented using Verilog HDL on a Xilinx Virtex 6 FPGA. The FPGA implementation is capable of processing 119 full HD images per second or 29 quad HD images per second.

As future work, a high performance EECI demosaicing hardware and a high performance AP demosaicing hardware can be designed and implemented. These demosaicing hardware can be compared with the proposed modified EECI demosaicing hardware and the proposed hybrid ECI and AP demosaicing hardware. The power and energy consumptions of these demosaicing hardware can be estimated.

BIBLIOGRAPHY

- [1] B. K. Gunturk, J. Glotzbach, Y. Altunbasak, "Demosaicking: Color Filter Array Interpolation," *IEEE Signal Processing Magazine*, Vol. 22, No.1, pp. 44-54, Jan. 2005.
- [2] B. E. Bayer, "Color Imaging Array," *U.S. Patent No. 3 971 065*, Jul. 1976.
- [3] S. Pei, I. Tam, "Effective Color Interpolation in CCD Color Filter Arrays Using Signal Correlation," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 13, No. 6, pp. 503-513, Jun. 2003.
- [4] L. Chang, Y. Tan, "Effective Use of Spatial and Spectral Correlations for Color Filter Array Demosaicking," *IEEE Transactions on Consumer Electronics*, Vol. 50, No. 1, pp. 355-365, Feb. 2004.
- [5] B. K. Gunturk, Y. Altunbasak, R. M. Mersereau, "Color Plane Interpolation Using Alternating Projections," *IEEE Transactions on Image Processing*, Vol. 11, No. 9, pp. 997-1013, Sep. 2002.
- [6] J. F. Hamilton, J. E. Adams, "Adaptive Color Plan Interpolation in Single Sensor Color Electronic Camera," *U.S. Patent No. 5 629 734*, May 1997.
- [7] C. A. Laroche, M. A. Prescott, "Apparatus and Method for Adaptively Interpolating A Full Color Image Utilizing Chrominance Gradients," *U.S. Patent No. 5 373 322*, Dec. 1994.
- [8] W. T. Freeman, "Median Filter For Reconstructing Missing Color Samples," *U.S. Patent No. 4 724 395*, Feb. 1988.
- [9] X. Li, M. T. Orchard, "New Edge-directed Interpolation," *IEEE Transactions on Image Processing*, Vol. 10 No. 10, pp. 1521-1527, Oct. 2001.
- [10] S. Yamanaka, "Solid State Camera," *U.S. Patent 4 054 906*, Oct. 1977.
- [11] R. Lukac and K. N. Plataniotis, "Color Filter Arrays: Design and Performance Analysis," *IEEE Transactions on Consumer Electronics*, Vol. 51, No. 4, pp. 1260-1267, Nov. 2005.
- [12] M. Parmar and S. J. Reeves, "A Perceptually Based Design Methodology For Color Filter Arrays," *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Vol. 3, pp.473-476, May 2004.
- [13] K. Hirakawa and P. J. Wolfe, "Spatio-spectral Color Filter Array Design For Optimal Image Recovery," *IEEE Transactions on Image Processing*, Vol. 17, No.10, pp. 1876-1890, Oct. 2008.
- [14] Eastman Kodak Company, *Kodak Lossless True Color Image Suite*, <<http://r0k.us/graphics/kodak/>>, Retrieved Aug. 2011.

[15] J. Garcia-Lamont, M. Aleman-Arce, J. Waissman-Vilanova, "A Digital Real Time Image Demosaicking Implementation for High Definition Video Cameras", *Electronics, Robotics and Automotive Mechanics Conference (CERMA)*, pp. 565-569, Sep. 2008.