

FPGA IMPLEMENTATION OF A RECONFIGURABLE ADDRESS  
GENERATION UNIT FOR IMAGE PROCESSING APPLICATIONS

KAM KOK HORNG

UNIVERSITI TEKNOLOGI MALAYSIA

FPGA IMPLEMENTATION OF A RECONFIGURABLE ADDRESS  
GENERATION UNIT FOR IMAGE PROCESSING APPLICATIONS

KAM KOK HORNG

A project report submitted in partial fulfilment of the  
requirements for the award of the degree of  
Master of Engineering (Electrical – Computer and Microelectronics System)

Faculty of Electrical Engineering  
Universiti Teknologi Malaysia

JUNE 2013

*Specially dedicated to my family, lecturers, fellow friends and those who have guided  
and inspired me throughout my journey of education*

## **ACKNOWLEDGEMENT**

First and foremost, I would like to take this opportunity to express my deepest gratitude to my project supervisor, Prof. Dr. Mohamed Khalil bin Hj Mohd Hani for his guidance, sharing of knowledge and encouragement throughout the project period.

Special thanks to my company, Intel for funding my part-time Master study at Universiti Teknologi Malaysia (UTM). I would like to thanks to my manager and my colleagues for their support and understanding throughout the duration of my studies. Thanks to my friends and course mates who helped me throughout the studies.

Last, I would like to thanks to my family for their support, understand and encouragement for me to complete the project.

## ABSTRACT

Nowadays, the DSP algorithms are being widely used in the world of digital image processing. Example of the DSP algorithms that used in image processing is 2D correlation, 2D convolution, fast Fourier transforms, FIR filter and etc. The performance of the DSP algorithm is highly depends on its processing speed and memory bandwidth. Those algorithms require intensive data manipulation and calculation happens in parallel. The DSP algorithms also require complex address pattern calculation. The DSP processor needs to handle the data processing and also complex address calculation in the same time. The complex address pattern calculation using RISC processor is not efficient and therefore slower down the overall memory access speed. Hence, a dedicated hardware blocks to perform the address generation is essential. Such hardware known as Address Generation Unit(AGU). The prior arts of AGU have limitations as some of the AGU do not able to handle image edge condition and data reuse. Besides that, the prior art of the AGU have not been verified in the actual SOC environment. In this project, a reconfigurable AGU that targeted for 2D correlation, sum of absolute difference and Finite Impulse Response (FIR) is proposed. The proposed AGU able to take care of the image edge conditions by padding it with edge pixels. The proposed AGU also being integrated into the Altera Avalon fabric and fully verified in Altera DE2-70 FPGA. It also shows 30% to 40% improvements in the performance at certain area.

## ABSTRAK

Pada masa kini, algoritma DSP digunakan secara luas dalam dunia pemrosesan imej digital. Algoritma DSP yang digunakan dalam pemrosesan imej digital termasuk korelasi 2D, convolution 2D, Fast Fourier Transform, penapis FIR dan lain lain. Prestasi algoritma DSP bergantung kepada kelajuan pemrosesan dan juga jalur lebar memori. Selain daripada itu, algoritma DSP juga memerlukan manipulasi data dan pengiraan alamat memori bercorak kompleks. Pemproses DSP perlu mengendalikan pemrosesan maklumat tersebut secara selari. Pemproses RISC diketahui kurang cekap dalam mengendalikan manipulasi data yang kompleks. Oleh kerana ini, blok perkakasan khusus untuk mengira alamat diperlukan. Perkakasan untuk mengira alamat dikenali sebagai Unit Generasi Alamat (AGU). Generasi AGU terlebih dahulu tidak mampu mengendalikan keadaan tepi imej dan menggunakan semula data. Selain itu, generasi AGU terlebih dahulu tidak pernah disahkan dalam persekitaran SOC sebenar. Dalam projek ini, AGU yang boleh dikonfigur semula untuk korelasi 2D, SAD dan penapis FIR dicadangkan. AGU tersebut berupaya mengendalikan keadaan tepi imej dan juga menggunakan data semula. AGU tersebut akan dimasukkan ke dalam Altera Avalon Fabric dan dilaksanakan dalam papan Altera FPGA bermodel DE2-70. AGU yang dicadangkan juga mempunyai lebih kurang 30% ke 40% lebih laju daripada AGU terlebih dahulu dalam keadaan tertentu.

## TABLE OF CONTENTS

CHAPTER	TITLE	PAGE
	<b>DECLARATION</b>	ii
	<b>DEDICATION</b>	iii
	<b>ACKNOWLEDGEMENT</b>	iv
	<b>ABSTRACT</b>	v
	<b>ABSTRAK</b>	vi
	<b>TABLE OF CONTENTS</b>	vii
	<b>LIST OF TABLES</b>	x
	<b>LIST OF FIGURES</b>	xi
	<b>LIST OF ABBREVIATIONS</b>	xvi
	<b>LIST OF APPENDICES</b>	xvii
<b>1</b>	<b>INTRODUCTION</b>	1
	1.1 Problem Background	1
	1.2 Problem Statement	2
	1.3 Objective	2
	1.4 Scope of Work	3
<b>2</b>	<b>LITERATURE REVIEW</b>	4
<b>3</b>	<b>THEORY AND METHODOLOGY</b>	13
	3.1 2D Correlation	13
	3.2 Finite Impulse Response	15
	3.3 Sum of Absolute Differences	17

3.4	Project Methodology	18
3.5	Tool used in the project	21
<b>4</b>	<b>DESIGN AND IMPLEMENTATION</b>	<b>23</b>
4.1	2D Correlation	24
4.1.1	Arch_2D_Corr	25
4.1.2	AGU_2D_Corr	32
4.2	Finite Impulse Response (FIR) Filter	35
4.2.1	Arch_FIR	35
4.2.2	AGU_FIR	39
4.3	Sum of Absolute Differences	43
4.3.1	Arch_SAD	43
4.3.2	AGU_SAD	49
4.4	Reconfigurable AGU	52
4.5	Design Integration	52
4.5.1	Integration to Avalon Bus	53
<b>5</b>	<b>RESULT AND DISCUSSION</b>	<b>57</b>
5.1	Verification Methodology	57
5.2	2D Correlation	59
5.2.1	Timing simulation using simple dataset	59
5.2.2	Algorithm modeling in MATLAB	63
5.2.3	Image pixels extraction	63
5.2.4	Timing simulation using real image	64
5.2.5	Verify design in the FPGA	66
5.2.6	Output pixels extraction	68
5.2.7	Output image comparison	69
5.3	SAD	71
5.3.1	Timing simulation using simple dataset	71
5.3.2	Algorithm modeling in MATLAB	74
5.3.3	Image pixels extraction	75
5.3.4	Timing simulation using real image	75
5.3.5	Verify design in FPGA	79



5.3.6	Output result comparison	81
5.4	FIR	78
5.4.1	Timing simulation using simple dataset	78
5.4.2	Verify design in FPGA	79
5.4.3	Output result comparison	81
5.5	Performance Analysis	81
<b>6</b>	<b>CONCLUSION AND FUTURE WORK</b>	<b>83</b>
6.1	Conclusion	83
6.2	Future Improvements	85
	<b>REFERENCES</b>	<b>86</b>
	Appendices A-B	85-115

**LIST OF TABLES**

<b>TABLE NO.</b>	<b>TITLE</b>	<b>PAGE</b>
4.1	RTL-CS table for Arch_2D_Corr	29
4.2	RTL-CS table for Arch_FIR	38
4.3	RTL-CS table for SAD	47
4.4	AGU Top Module Register Information	56
5.1	Comparing the output of Arch_2D_Corr with Matlab Output for 10x5 image and 3x3 kernel	62
5.2	Performance comparison between AGU for 2D Correlation, SAD and FIR	82

## LIST OF FIGURES

FIGURE NO.	TITLE	PAGE
2.1	Reconfigurable address generation unit based on matrix Address sequence	4
2.2	Block diagram of reconfigurable address generation unit based on a matrix address sequence	5
2.3	Address sequence for X coordinate and Y coordinate for Arch1	6
2.4	Functional block diagram of AGU1	7
2.5	Address sequence for X coordinate and Y coordinate for Arch2	8
2.6	Functional block diagram of AGU2	9
2.7	Hardware schematic of AGU for data fetch of convolution kernel	11
2.8	Hardware schematic of AGU for accessing data for a Linear Phase FIR Kernel	12
2.9	Datapath for SAD computation	12
3.1	Simple averaging Filter (w)	14
3.2	Sample 4x4 image(f)	14
3.3	Sample 4x4 image padded with the edge pixel (f)	14
3.4	Results of the correlation (g)	15
3.5	Kernel movements in correlation using 3x3 kernel	15
3.6	FIR Block Diagram	16
3.7	Boxcar Filter	16

3.8	Original image block and block to be compared	17
3.9	SAD Result	17
3.10	Project Flow	18
3.11	Algorithm Modeling and Architecture Development	19
3.12	The High Level Structure of the AGU Design integrated to Avalon Bus	20
4.1	Arch_2D_Corr	24
4.2	SFG and schedule of Arch_2D_Corr for a 3x3 Kernel	26
4.3	ASM chart of Arch_2D_Corr	27
4.4	Functional block diagram of datapath unit for Arch_2D_Corr for a 3x3 kernel	28
4.5	Functional block diagram of CU for Arch_2D_Corr	29
4.6	Functional block diagram for Arch_2D_Corr	31
4.7	Address Sequences for AGU_2D_Corr	32
4.8	ASM chart of AGU_2D_Corr	33
4.9	Functional Block Diagram of AGU_2D_Corr	34
4.10	SFG and schedule of Arch_FIR for a 3x3 kernel	36
4.11	ASM Chart of Arch_FIR	36
4.12	Functional block diagram of DU for Arch_FIR	37
4.13	Functional block diagram of CU for Arch_FIR	38
4.14	Functional block diagram of Arch_FIR	39
4.15	ASM Chart of AGU for linear phase FIR filter	41
4.16	Functional block diagram of DU for AGU_FIR	42
4.17	Signal Flow Graph (SFG) of SAD_Arch	44
4.18	ASM chart of SAD_Arch	44

4.19	Scheduling and RTL codes of SAD_Arch	45
4.20	Functional block diagram of datapath Unit	46
4.21	Functional block diagram of the CU	47
4.22	Functional block diagram of SAD Arch	48
4.23	Address Sequences for AGU_SAD	49
4.24	ASM Chart of AGU_SAD	50
4.25	Functional Block Diagram of Datapath Unit for AGU_SAD	51
4.26	Functional block diagram of reconfigurable AGU	52
4.27	Functional block diagram of the top module	53
4.28	Avalon Bus Read Waveform	54
4.29	Avalon Bus Write Waveform	54
4.30	Functional block diagram of the top module with Avalon Bus Decoding Logic	54
4.31	Structure of the Altera Nios II SOPC with AGU Top Module integrated	55
5.1	Simulation results of AGU_2D_Corr with 8x4 image and 5x5 kernel	60
5.2	Expected Result (Left) versus Simulation Result (Right)	60
5.3	Simulation results of AGU2 with 10x5 image and 3x3 kernel	60
5.4	Expected Result (Left) versus Simulation Result (Right)	61
5.5	The input RAM memory setup to verify Arch_2D_Corr	61
5.6	Simulation results of Arch_2D_Corr with 10x5 image and 3x3 kernel	61
5.7	Simulation results of Arch_2D_Corr with 10x5	62

	image and 3x3 kernel – at the end of the image	
5.8	MATLAB Code for 2D Correlation Algorithm	63
5.9	MATLAB Code for image pixel extraction for 2D Correlation	63
5.10	Simulation results of AGU_TOP_MODULE in 2D Correlation Mode	64
5.11	2D Correlation Simulation Report	65
5.12	2D Correlation Output Pixel File from ModelSim	66
5.13	C Code to configure the AGU_TOP_MODULE	67
5.14	C Code to configure the AGU_TOP_MODULE for 2D Correlation Mode	67
5.15	Snapshot of the Nios II output console for 2D Correlation Mode	68
5.16	Snapshot of the Altera In-System Content editor for 2D Correlation	69
5.17	MATLAB code to read the ModelSim Result	69
5.18	MATLAB code to process the hex data from FPGA memory	69
5.19	Comparing the output image of 2D correlation mode for Matlab, Modelsim and FPGA	70
5.20	Comparing the output image of 2D correlation mode for Matlab, Modelsim and FPGA using another image	70
5.21	Simulation results of AGU_SAD using 8x4 image	71
5.22	Expected Result (Left) versus Simulation Result (Right)	72
5.23	Simulation results of AGU_SAD using 8x4 image	72
5.24	Expected Result (Left) versus Simulation Result (Right)	73
5.25	The input RAM 1 memory setup to verify Arch_SAD	73

5.26	The input RAM 2 memory setup to verify Arch_SAD	73
5.27	Simulation results of Arch_SAD using 8x4 image	74
5.28	MATLAB Code for SAD Algorithm	74
5.29	MATLAB Code for image pixel extraction for SAD	75
5.30	Simulation results of AGU_TOP_MODULE in SAD Mode	75
5.31	C Code to configure the AGU_TOP_MODULE for SAD Mode	76
5.32	Snapshot of the Nios II output console for SAD Mode	76
5.33	Image 1 is being loaded to input RAM1 while image 2 is being loaded to input RAM2	77
5.34	Simulation results of AGU for FIR filter	78
5.35	Memory setup for FIR Mode	78
5.36	Simulation results of AGU_TOP_MODULE in FIR Mode	79
5.37	C Code to configure the AGU_TOP_MODULE for FIR Mode	79
5.38	Snapshot of the Nios II output console for SAD Mode	80

**LIST OF ABBREVIATIONS**

AGU	-	Address Generation Unit
ASM	-	Algorithmic State Machine
DSP	-	Digital Signal Processing
FPGA	-	Field-Programmable Gate Array
RTL-CS	-	Register Transfer Level – Control Signal
FIR	-	Finite Impulse Response
SAD	-	Sum of Absolute difference
2D	-	Two Dimension



**LIST OF APPENDICES**

<b>APPENDIX</b>	<b>TITLE</b>	<b>PAGE</b>
A	Verilog Codes	86
B	C Codes	113

## **CHAPTER 1**

### **INTRODUCTION**

#### **1.1 Problem Background**

Nowadays, digital signal processing (DSP) hardware required a large amount of processing elements running in parallel. Performance gain in a hardware system are achieved by using techniques such as pipelining, customized functional units, optimized memory hierarchies and running processes in parallel. Therefore large amount of the data access is needed.

Most of the DSP hardware is in cooperate with RISC processor. Manipulation with structure data-types are not efficiently supported by RISC processor. RISC processor is designed to have minimum number of instruction set and a complex compiler is needed. Hence, a considerable amount of codes need to be generated to cater the complex arithmetic manipulation that required by the DSP algorithm. This creates large amount of overhead on the performance of DSP hardware and many clock cycles are needed to fetch the data and instruction, decode, execute and process the data.

DSP algorithm can be used for image processing purposes. Images are stored in the memory, DSP processor are required to read the image from memory, process the image and store it back to the memory. Some of the DSP algorithm required complex address offset calculation and relying on software to calculate the address offset creates more overhead than a dedicated hardware for address calculation.

## 1.2 Problem Statement

There are several numbers of DSP algorithms which are being widely used for digital image processing. They are 2D convolution, 2D correlation, fast Fourier transforms (FFT), FIR filter and etc. Those algorithms require intensive index manipulations, resulting in complex address pattern. Complex address pattern calculation using RISC processor is in-efficient and it will slow down the overall system performance and resulting larger latency for accessing data.

Therefore, hardware accelerator is needed to speed up the address calculation and reduce the overall latency for accessing data rather than using software to do the address calculation. Such hardware accelerator for address calculation is known as Address Generation Unit (AGU).

Prior arts of reconfigurable AGU [1] only focus on the implementation for 2D correlation. It does not support the address generation for DSP algorithms other than 2D correlation. Furthermore, the prior arts of reconfiguration AGU [1] has only shows the Modelsim simulation results and never been implemented in FGPA.

Enhancement is needed to the design and more address sequences for different DSP algorithm need to be added. The AGU needs to be capable of generating one address per clock in required sequence [6].

## 1.3 Objective

The main objective of this project is to enhance the prior AGU and implement it to FPGA. The objectives that need to be achieved in this project are listed below:

1. Design a reconfigurable AGU that:
  - Capable to generate an address per clock.
  - Support windows based operation.

- Able to be used for various DSP applications.
  - The RTL design needs to be able to configure easily through parameter.
  - Support different image size and kernel size.
  - Take care of image border and considering the data reuse.
2. Prototype and implement the design into the FPGA.

#### **1.4 Scope of work**

The project is targeted for the hardware architecture of Address Generation Unit (AGU) in Altera DE2-70 FPGA. The targeted DSP modules for the AGU in the project include 2D Correlation, Finite Impulse Response (FIR) and Sum of absolute difference (SAD). The DSP modules for targeted application are designed and work together with the AGU. The full design is then integrated to Altera Avalon Bus.

The AGU design is capable to generate an address per clock. Besides that, it also supports windows based operation with different kernel sizes and image dimension. The RTL is being parameterized and easy to be configured. The data reuse and image border consideration is being take care in the design as well.

MATLAB is being used to model the DSP algorithm and Modelsim is used as a simulator in the project. Altera Eclipse IDE is used as the environment to develop the C program to communicate with the design after the full integration of the design. Image pixels are extracted from simulation result in Modelsim and memory in the FPGA to compare with the output result from MATLAB.

## REFERENCES

1. Heng Ai Hoon. *Reconfigurable address generation unit for 2D Correlation in FPGA*. 2011.
2. Tetsuo Kawano. *Reconfigurable address generation circuit for image processing, and reconfigurable LSI comprising the same*. U.S. Patent 7, 515, 159. 2009.
3. P. Hulina, L. Coraor, L. Kurian, and E. John. Design and VLSI implementation of an address generation coprocessor. *Proc. IEE Computers and Digital Techniques*, 1995. 142(2): 145-151.
4. Ramesh M. Kini, and S. David. Comprehensive address generator for digital signal processing. *International Conference on Industrial and Information Systems (ICIIS)*. December 28-31, 2009. Sri Lanka: IEEE. 2009. 325-330.
5. Ramesh M. Kini, and S. David. ASIC implementation of address generation unit for digital signal processing kernel-processor. *ICGST-PDCS*, 2011. 11(1): 1-9.
6. Ramesh Kini M and Summam David S. Address Generation for DSP Kernel. *Communications and Signal Processing (ICCSP), 2011 International Conference*. February 10-12, 2011: 112-116