FAST FOURIER TRANSFORM MODULE FOR IMPLEMENTATION IN
NIOS II EMBEDDED PROCESSOR

HISHAM AZHARI AHMED OSMAN

UNIVERSITI TEKNOLOGI MALAYSIA
FAST FOURIER TRANSFORM MODULE FOR IMPLEMENTATION IN NIOS II EMBEDDED PROCESSOR

HISHAM AZHARI AHMED

A project report submitted in partial fulfillment of the requirements for the award of the degree of Master of Electrical-Electronics & Telecommunication Engineering

Faculty of Electrical Engineering
Universiti Teknologi Malaysia

MAY 2008
Specially dedicated to
my beloved mother and father

“Only those who dare to fail greatly, can ever achieve success greatly”
First and foremost, I am greatly indebted to Almighty Allah for giving me endurance and strength to finish this project.

I would like to express my deepest gratitude and thanks to my supervisor, Professor Dr. Mohamed Khalil bin Haji Mohd Hani for giving me this chance to explore new grounds in the computer-aided design of electronics system. Really, his encouragement, guidance, critic and friendship were the reason to carry this project to a profitable completion. Honestly, in one year I have learnt a lot from him, not only in my study but also in the lessons of life.

My genuine appreciation goes out to all those who contributed with me to the completion of this project. In particular, I mention Mr. Illiasaak, Mohd Nazrin, Alamin Ali and Mrs. Danaletchumi for their sincere guidance and members of research group in VLSI-eCAD lab.

Special thanks also to my colleagues at the faculty who gave valuable advice and opinion on the project.

Finally, I would like to express my love and appreciation to my family members for their support and help even though they were not around. I am also very grateful for encouragement provided by my friends who have provided assistance and support at various occasions.
ABSTRACT

The Fast Fourier Transform is an indispensable algorithm in many digital signal processing applications but yet is deemed computationally expensive cost when designed it on hardware. This thesis proposes a design and implementation of Fast Fourier Transform algorithm in embedded system by utilize it in Nios II embedded processor and integrate with Nios II floating point custom instruction. The design is based on Decimation-In-Time and Decimation-In-Frequency radix-2 for the better performance and speed. Hardware implementation, the ALTERA CYCLONE II EP2C35F672C6 (DE2 board) is used. Hardware interfacing, Graphical User Interface (GUI) has been developed using MATLAB software; it’s an original method for interfacing between ALTERA Field Programmable Gate Array (FPGA) and software in host PC. Input values are sent from MATLAB to ALTERA development board via serial port and the calculation data return back to MATLAB. The purpose of this technique is take advantages of the MATLAB in analysis and plot the result.
ABSTRAK

Penukaran Pantas Fourier merupakan suatu algoritma yang mustahak dalam kebanyakan aplikasi pemprosesan isyarat digital dan hanya dianggap sebagai suatu kos pengiraan yang tinggi bila ia direka bentuk dalam sesuatu “hardware”. Tesis ini mencadangkan suatu corak dan implementasi algoritma Fast Fourier Transform dalam system yang terikat dan diaplikasikan di dalam prosessor Nios II dan dintegrasikan dengan arahan khas titik terapung Nios II. Corak tersebut adalah berasakan Decimation-In-Time dan Decimation-In-Frequency radix-2 untuk mendapatkan hasil yang lebih bermutu dan cepat. Dalam implementasi “hardware”, ALTERA CYCLONE II EP2C35F672C6 (DE2 board) digunakan. “Hardware interfacing”, Antaramuka Grafik Pengguna telah dikembangkan dengan menggunakan “software” MATLAB; ia adalah kaedah tulen untuk mewujudkan sesuatu ruang hubung kait antara ALTERA Field Programmable Gate Array (FPGA) dan “hos software” dalam PC. Nilai input dihantar dari MATLAB ke ALTERA development board melalui port serial dan data pengiraan pula dihantar balik ke MATLAB. Tujuan teknik ini digunakan adalah untuk meggunakan faedah MATLAB dalam analisis dan untuk mengeplot hasil / keputusan yang diperoleh.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>CHAPTER</th>
<th>TITLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>DECLARATION</td>
<td></td>
<td>ii</td>
</tr>
<tr>
<td>DEDICATION</td>
<td></td>
<td>iii</td>
</tr>
<tr>
<td>ACKNOWLEDGEMENT</td>
<td></td>
<td>iv</td>
</tr>
<tr>
<td>ABSTRACT</td>
<td></td>
<td>v</td>
</tr>
<tr>
<td>ABSTRAK</td>
<td></td>
<td>vi</td>
</tr>
<tr>
<td>TABLE OF CONTENTS</td>
<td></td>
<td>vii</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td></td>
<td>xi</td>
</tr>
<tr>
<td>LIST OF APPENDIX</td>
<td></td>
<td>xiv</td>
</tr>
</tbody>
</table>

1 INTRODUCTION 1

1.1 Motivation 1
1.2 Problem Statement 2
1.3 Project Objectives 3
1.4 Scope of Work 4
1.5 Project Contribution 4
1.6 Thesis Organization 5
2 FAST FOURIER TRANSFORM

2.1 Introduction 6
2.2 FFT Algorithms 7
  2.2.1 Radix-2 FFT Algorithms 8
    2.2.1.1 Decimation-In-Time FFT 9
    2.2.1.2 Decimation-In-Frequency FFT 15
2.3 Algorithms Implementation 17

3 EMBEDDED SYSTEM DESIGN 18

3.1 Project Procedure 18
3.2 System Architecture 20
3.3 Features Embedded Systems 21
  3.3.1 History and Future of Embedded System 23
  3.3.2 Real Time System 24
3.4 Embedded Software of FFT Algorithm 24
  3.4.1 The Complex Class 25
  3.4.2 The Main Function 27
    3.4.2.1 The System Inputs 28
    3.4.2.2 The Earlier Stages 29
    3.4.2.3 The Final Stage 32
3.5 Nios II based System on Chip Development Platform 33
  3.5.1 Nios II Custom Instruction 34
  3.5.2 Nios II Floating Point Custom Instruction 35
3.6 Application Software 36
3.7 ALTERA DE2 Development Kit 37
4 MATLAB GRAPHICAL USER INTERFACE 40

4.1 Introduction 40
4.2 Communication Interface 41
4.3 Serial Port Overview 42
  4.3.1 Serial Communication 43
  4.3.2 The Serial Port Interface Standard 43
  4.3.3 Connecting Two Devices with a Serial Cable 44
4.4 MATLAB Software 45
4.5 Integration MATLAB with DE2 Board 45

5 RESULTS AND PERFORMANCE EVALUATION 48

5.1 Introduction 48
5.2 System Results 49
5.3 Performance Evaluation 53
5.4 Spectral Analysis 55

6 CONCLUSION 57

6.1 Concluding Remarks 57
6.2 Recommendation for Future Work 58
  6.2.1 Higher N-Point FFT Computation 58
  6.2.2 The Algorithm Architecture In The Decimation-In-Frequency 59
  6.2.3 High Radix Used 61
  6.2.4 Use of the System in Other Application 62

REFERENCES 63
Appendix A - C 66-81
# LIST OF TABLES

<table>
<thead>
<tr>
<th>TABLE NO.</th>
<th>TITLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Main design steps in the project</td>
<td>19</td>
</tr>
<tr>
<td>3.2</td>
<td>Math library floating-point usage</td>
<td>36</td>
</tr>
<tr>
<td>5.1</td>
<td>Comparison between the systems that includes floating point custom instruction and the system not including floating point custom instruction for several function</td>
<td>55</td>
</tr>
</tbody>
</table>
# Lists of Figures

<table>
<thead>
<tr>
<th>Figure No.</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Radix-2 for an N point FFT.</td>
<td>8</td>
</tr>
<tr>
<td>2.2</td>
<td>First step in the decimation-in-time algorithm.</td>
<td>10</td>
</tr>
<tr>
<td>2.3</td>
<td>Five stages in the computation of an n = 32-point DFT.</td>
<td>12</td>
</tr>
<tr>
<td>2.4</td>
<td>Thirty two-point decimation-in-time FFT algorithm.</td>
<td>13</td>
</tr>
<tr>
<td>2.5</td>
<td>Basic butterfly computation in the decimation-in-time FFT algorithm.</td>
<td>13</td>
</tr>
<tr>
<td>2.6</td>
<td>Shuffling of the data and bit reversal.</td>
<td>14</td>
</tr>
<tr>
<td>2.7</td>
<td>Thirty two-point decimation-in-frequency FFT algorithm</td>
<td>16</td>
</tr>
<tr>
<td>3.1</td>
<td>Project workflow</td>
<td>19</td>
</tr>
<tr>
<td>3.2</td>
<td>System architecture of the design</td>
<td>20</td>
</tr>
<tr>
<td>3.3</td>
<td>A Generic embedded system</td>
<td>22</td>
</tr>
<tr>
<td>3.4</td>
<td>Nios II processor system</td>
<td>34</td>
</tr>
<tr>
<td>3.5</td>
<td>Custom instruction logic connects to the Nios II ALU</td>
<td>35</td>
</tr>
<tr>
<td>3.6</td>
<td>ALTERA DE2 board</td>
<td>38</td>
</tr>
<tr>
<td>3.7</td>
<td>Block diagram of the DE2 board.</td>
<td>39</td>
</tr>
<tr>
<td>4.1</td>
<td>Communication interface between FFT module and MATLAB software</td>
<td>40</td>
</tr>
<tr>
<td>4.2</td>
<td>A male DE-9 connector used for a serial port on a PC style computer.</td>
<td>42</td>
</tr>
<tr>
<td>4.3</td>
<td>Connecting two devices with a serial cable</td>
<td>43</td>
</tr>
</tbody>
</table>
4.4 Integration MATLAB with DE2 board 45
5.1 Ramp function 49
5.2 Output in MATLAB command window 50
5.3 The final output of ramp function 51
5.4 Step function 52
5.5 Final output of step function 52
5.6 Console view displaying Nios II hardware output using floating point custom instruction for step function 53
5.7 Console view displaying Nios II hardware output without floating point custom instruction for step function 54
5.8 Sinusoidal Signal 56
5.8 Result of Sinusoidal signal 56
6.1a) FFT for ramp discrete time signal N=32 59
6.1b) FFT for ramp discrete time signal N=1024 59
6.2a) Butterfly algorithms for DIF 59
6.2b) Butterfly algorithms for DIT 60
6.3 Thirty two-point Decimation-In-Frequency FFT algorithms 60
6.4 Basic butterfly computations in a radix-4 FFT algorithm 61
6.5 Radix-4 for a 16 point FFT 61
B.1 CPU window 69
B.2 JTAG UART window 70
B.3 Timer window 71
B.4 The system components 72
B.5 Floating point custom instruction location in Nios II embedded processor 72
B.6 Floating point hardware 73
B.7 Complete configuration of the floating point custom instruction 73
<table>
<thead>
<tr>
<th></th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>B.8</td>
<td>The entire components system in Quartus II</td>
<td>75</td>
</tr>
<tr>
<td>B.9</td>
<td>Interface Protocols</td>
<td>76</td>
</tr>
<tr>
<td>B.10</td>
<td>System library properties</td>
<td>77</td>
</tr>
<tr>
<td>B.11</td>
<td>Run the setting</td>
<td>78</td>
</tr>
<tr>
<td>B.12</td>
<td>Complete configuration of the serial connection</td>
<td>79</td>
</tr>
</tbody>
</table>
# LIST OF APPENDIX

<table>
<thead>
<tr>
<th>APPENDIX</th>
<th>TITLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>FFT code in C++</td>
<td>66</td>
</tr>
<tr>
<td>B</td>
<td>Building and configuration of the embedded system into Nios II</td>
<td>69</td>
</tr>
<tr>
<td></td>
<td>embedded processor</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>MATLAB code for connection with DE2 board</td>
<td>80</td>
</tr>
</tbody>
</table>
CHAPTER 1

INTRODUCTION

This thesis proposes a design of Fast Fourier Transform and applies it into Nios II Embedded Processor. This chapter covers the motivation, problem statement, project objectives, scope of work, project contributions and finally thesis organization.

1.1 Motivation

The Fast Fourier transform is a critical tool in digital signal processing where its value in analyzing the spectral content of signals has found application in a wide variety of applications. The most prevalent of these applications is being in the field of communications where the ever increasing demand on signal processing capabilities have given rise to the importance of the Fourier transform to the field. However, the Fourier transform is a part of many systems in a wide variety of industrial and research fields. Its uses range from signal processing for the analysis of physical phenomena to analysis of data in mathematical and financial systems.
The majority of systems requiring Fourier transforms are real time systems which necessitate high speed processing of data. Given the complexity in performing the Discrete Fourier, the implementation of high speed Fast Fourier transform has required the use of dedicated hardware processors. The majority of high performance Fourier transforms has required the use of full custom integrated circuits and has typically been in the form of an application, specifically integrated circuit. Although much work has been put into raising performance while reducing hardware requirements, and also cost, the cost of full custom hardware still limits the availability of Fourier transform hardware to low volume production.

Nevertheless the development of programmable logic hardware has produced devices that are increasingly capable of handling large scale hardware. High density field programmable gate arrays (FPGA) that are already available in the market can boast upwards of 180,000 logic elements, nine megabits of memory, and on board processors.

The use of FPGA in implementing hardware eliminates the need for the long and costly process of creating a full custom integrated circuit and the time and cost of testing and verification. Saving cost in designing, testing, and time from design to a functional device.

These features of the FPGA make it especially attractive for the purpose of creating embedded processors for research and development purposes.

However the design of any of embedded processors must consider two important factors efficiency and flexibility for reaching an ideal design.

### 1.2 Problem Statement

Efficiency and flexibility are two of the most important driving factors in embedded system design. Efficient implementations are required to meet the tight cost, timing, and power constraints present in embedded systems. Flexibility, albeit
tough to quantify, is equally important; it allows system designs to be easily modified or enhanced in response to bugs, evolution of standards, market shifts, or user requirements, during the design cycle and even after production.

Various implementation alternatives for a given function, ranging from custom-designed hardware to software running on embedded processors, provide a system designer with differing degrees of efficiency and flexibility. Unfortunately, it is often the case that these are conflicting design goals. While efficiency is obtained through custom hardwired implementations, flexibility is best provided through programmable implementations.

Hardware/software partitioning separating a system’s functionality into embedded software (running on programmable processors) and custom hardware (implemented as coprocessors or peripheral units) is one approach to achieve a good balance between flexibility and efficiency.

1.3 Project Objectives

The aims of this project are as follow:

1. Design and implementation of Fast Fourier Transform (FFT) algorithm into embedded system by:
   a) Utilizing Nios II embedded processor.
   b) Integrating it with Nios II Floating Point Custom Instruction.

2. Developing MATLAB user interface to verify the proposed FFT system.
1.4  Scope of Work

Taking into account the resources and time available, this project is narrowed down to the following scope of work.

1. This project only considers 32 point FFT floating point. The Decimation-In-Time (DIT) algorithm is chosen.
2. The algorithm is implemented in C++ language.
3. Floating Point Custom Instruction is targeted for Nios II platform and implemented in ALTERA Cyclone II DE2 board.
4. MATLAB Graphical User Interface (GUI) has been used for the purpose of interfacing with FPGA hardware to provide inputs and display outputs.
5. Serial port (RS232) is used for transmitting and receiving data between FPGA board and MATLAB.
6. This Embedded system is applied in Spectral Analysis as an application.

1.5  Project Contributions

The most important contributions of this project are:

1. Integration framework of MATLAB and ALTERA development kit platform.
2. Utilizing Nios II Floating Point Custom Instruction in the design to increase performance and accelerate speed.
3. Created a simple protocol that is used for interaction with and communication between hardware and software via computer serial port.
REFERENCES


19. Altera Corporation, “Nios II Hardware Development Tutorial”


23. MathWorks Corporation. “Getting Started with MATLAB 7”


