To the Graduate Council:

I am submitting herewith a thesis written by D. Eric Harrah entitled "Hardware Implementation of the PET Backprojection Algorithm using FPGA Technology." I have examined the final electronic copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Electrical Engineering.

Danny Newport, Major Professor

We have read this thesis
and recommend its acceptance:

Don Bouldin

Gregory Peterson

Accepted for the Council:

Anne Mayhew
Vice Provost and Dean of
Graduate Studies

(Original signatures are on with official student records.)

Hardware Design of the PET Backprojection Algorithm using FPGA Technology

A Thesis

Presented for a

Master of Science

Degree in Electrical Engineering

The University of Tennessee, Knoxville

D. Eric Harrah

August 2002

ACKNOWLEDGEMENTS

I was given much encouragement and patience in completing this thesis. I must especially thank my major professor, Dr. Danny Newport, for his help, patience and encouragement over the past 2 years. Thanks are also given to Dr. Greg Peterson and Dr. Don Bouldin, the members of my thesis committee, for their continual prompting as to when they were going to see something. Also to Bryan Adams of Annapolis Microsystems for his constant patience and assistance through many, many email questions about the Wildcard.

I would especially like to thank my family for all the love and encouragement they have given me over the years to be my own man and follow my own path. To my Dad for keeping me straight when I tended to waiver in my strength to keep going and finish things up. To my Mom for all the hugs and kisses over the years and reminding me that God was on my side through it all. To my Grandpa Steve for blessing me with his Wisdom and guidance. And to my brothers and sister for always just being around to fight with, talk with and learn some of the hard lessons of life together. And especially to God for giving me a chance to live and explore this great world that He has given to us.

ABSTRACT

Backprojection is used in the recovery of 2D and 3D images in positron emission tomography (PET). PET is used by medical personnel in the detection and location of growths or tumors that lie within the human body. Current image reconstruction using the Backprojection algorithm requires a great deal of processing time to complete. The general method used to decrease processing time is a multi-processor system with each processor working on a portion of the final image to be reconstructed.

This thesis will focus on implementing the Backprojection algorithm utilizing a hardware platform. The Wildcard PCMCIA card, which contains a Virtex 300 FPGA, will be the focus of this effort. The goals of this thesis are to test the validity of using a FPGA to perform lengthy, numerical algorithms, in this case the PET backprojection algorithm, a task that has primarily been left to processors, and to compare the relative speed and accuracy of using the FPGA versus using a processor. The algorithm will be written for and tested by reconstructing a 128x128 2D PET image from sinogram data provided by CTI.

TABLE OF CONTENTS

LIST OF FIGURES

FIGURE **PAGE**

CHAPTER 1

Introduction

1.1 **Overview**

In the past 15 years FPGA technology has vastly improved, increasing the number and variety of applications FPGA's can perform. The processing performance and capability of the FPGA market has closely followed that of the microprocessor industry. The general trend in the use of FPGA[1]s is as dedicated hardware for increased data transfer with some logic manipulation.

With the increase in the size and processing capability of FPGA's, it becomes possible to utilize their adaptive nature in applications that perform complex numerical algorithms as part of their process. One of the primary difficulties with this use of FPGA[1]s is the limited experience many designers of both FPGA[1]s and synthesis tools have with developing their systems to meet the needs of an engineer. However, due to the increased capabilities of synthesis and design tools, developing an application that achieves improved numerical computation performance is viable.

The advantage to using FPGA's as the processing platform for computational algorithms is an increase in performance as well as a decrease in cost of systems to perform the algorithm. FPGA's are considered "Designated Hardware". Since they do not have an operating system and very few external variables, all of their processing time is spent on the applications at hand. They are capable of quickly passing data from one operation to the next. Also FPGA's can easily perform operations using multiple gates at

once, making parallel processing much easier. These factors allow a FPGA to outperform a general purpose processor for a specific application.

As for cost reduction, FPGA's cost less than a typical computer system, especially a multi-processor platform. When one considers the performance improvement and cost reduction of using an FPGA, the advantage to designing algorithms to perform on FPGA's is clear.

1.2 Intent of Thesis

The intent of this thesis is to implement a complex numerical algorithm using VHDL for a FPGA. This will be done by designing and implementing the backprojection algorithm on the Wildcard V300 chip for the reconstruction of a 128x128 2D-image. This design uses sinogram data that has been provided by CTI.

First, an introduction to FPGA's and computer aided design software is given in chapter 2. Brief histories of each are given to introduce the reader to the tools used in the development of this thesis. The primary background from which this thesis is derived is positron emission tomography (PET) used in medical imaging. Therefore, a brief history and a background of PET system is also presented.

In chapter 3, a description of the Wildcard V300 FPGA platform along with the methods used in designing the algorithm for the Wildcard are detailed. The problem of creating an algorithm with numerical computation for VHDL synthesis is described in this chapter as well as the solution to implementing the algorithm as synthesizable code.

Chapter 4 details the history and development of the backprojection algorithm. The methods used in the algorithm to recreate an image from the raw sinogram data are given and an analysis of the overall algorithm is presented. Serious time delay problems in the execution of the algorithm are also described to indicate the bottlenecks in the overall perform of the algorithm. The breadth of that chapter is on 2D-image reconstruction. Chapter 5 continues the discussion of Chapter 4 by reporting the final results of the thesis and comparison to results created by existing reconstruction systems. An analysis of the findings is given to conclude the design and implementation section of the thesis.

The primary effort of the thesis is described in Chapters 4 and 5. Future work and possible uses of the thesis are found in Chapter 6 and are based on the findings of Chapter 5.

CHAPTER 2

Background

2.1 FPGA Technology

Field Programmable Gate Arrays (FPGA's) were first developed in 1985 as a means of alleviating some of the design load placed on microprocessors. FPGA's have the capability of being programmed on the fly to perform a specified algorithm. They can be programmed and reprogrammed as many times as the user wishes in order to achieve the desired result. The major design benefit in this lies in the ability to test designs that "might" work. Prior to the development of the FPGA, designers were forced to test and retest a system many, many times in software to ensure that the design would work before it was fabricated. Since the fabrication process can be quite expensive and very time consuming, a designer does not know if the design is functional until the final chip arrives several weeks to a month later. If there were design flaws in the chip, the designer would have to redesign, retest and resubmit for fabrication. The use of FPGA's in the design process allows the designer flexibility, while reducing the cost should the design not work the first time. If the design fails after being tested on a FPGA, the designer can simply rework the design and download it again to the FPGA. Use of an FPGA would thus eliminate the loss in development time caused by a faulty initial design, as well as giving the designer knowledge of whether or not the design works.

Until recently the use of FPGA's has not been very widespread. They have only been used as a method of hardware testing and as a replacement for random logic. In the

past 5 to 10 years the performance and capability of the software tools has improved tremendously allowing FPGA's to be applied to a greater variety of applications. Since FPGA's had a rather narrow field of use in the past, there was little money and time being spent on developing FPGA technology. The technology fell behind processors in terms of size and performance. As the number of applications using FPGA¹s increased, the number of innovations made in the field increased. Today many FPGA's have performance and capabilities rivaling advanced processor technology.

2.2 Computer Aided Design

A software development platform is required for the development and implementation of FPGA designs. There is a wide variety of software tools available for FPGA's. In general there are two different types of tools that are used in the creation of applications, synthesis tools and place and route (P&R) tools. Synthesis tools are designed to accept files containing the design implemented in a hardware description language (HDL). The user designates a particular FPGA for the design to be mapped into by the synthesizer. The synthesizer will check the users HDL file(s) for syntax errors as well as erroneous data handling errors, such as illegal operations performed on a certain type of data or problems that the synthesizer might encounter in creating a netlist description for a particular section of code. If the HDL passes the syntax check, the synthesis tool will synthesize the design. During synthesis, the tool creates a netlist file, which contains a listing of all of the connections and components used in the design. The

netlist file is used by the P&R tools to define the programming for the FPGA to perform the desired application.

Every FPGA vendor has their own software tools specifically for their FPGA families Note that, in general, a designer must use the P&R software provided by the vendor for a particular FPGA. P&R tools perform one of the most important aspects of the design process. They define the programming of the FPGA to match the intended structure of the given application. Most of today's P&R software tools are optimized to give the most ideal layout for each design. Users have some control over this optimization, generally optimizing for speed or area.

2.3 Introduction to PET

Since the primary focus for this thesis is the development of the backprojection algorithm, which is a key component in the use of a PET system for medical imaging, an introduction to PET has been included to familiarize the reader with any terminology that may be used. Diagrams of a PET system as well as an array of detectors are show in Fig. 2.1 and Fig 2.2.

A PET imaging system is used by the medical field in the detection and prevention of cancerous tumors in patients. The patient is injected with a radioactive substance, which the body is known to normally metabolize. The radioactive isotope is positively charged and emits positrons into the patient's body. These positrons will eventually dissipate enough energy and collide with electrons found in the patient's body. In general, the body treats these injected isotopes as substances the body normally uses.

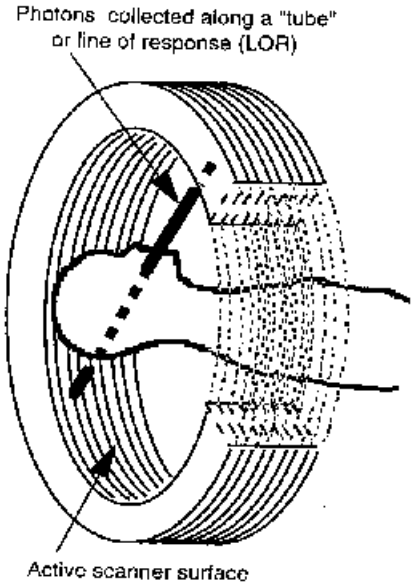

Fig. 2.1 Ring of detectors used by PET to collect sinogram data

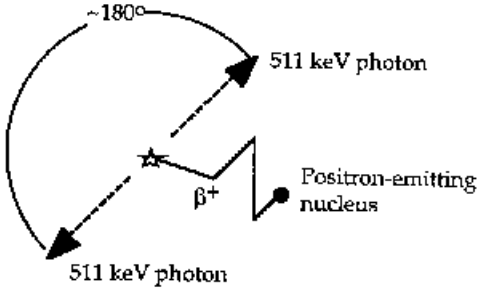

Fig. 2.2 Positron and electron matter/anti-matter collision based on LOR

For instance Fluro-deoxyglucose (FDG), is treated as glucose, causing the substance to be metabolized at a much higher level in areas such as the brain, muscle tissue, and tumors. When the positrons collide with electrons, a matter-antimatter annihilation occurs. This annihilation will cause the creation of two gamma rays that will move directly away from each other at 180° separation. Each gamma ray has an energy level of 511keV. The gamma rays are detected in a ring of scintillator crystals which surround the patient [1].

A histogram of the incident gamma rays on the detectors is taken based on the line of response (LOR) between two detectors and is stored in an array of data called a sinogram. A LOR is generally thought of as a line or tube upon which coincident events are detected by two individual detectors. Sinogram data is passed into the backprojection algorithm in order to reconstruct the image. The details of the backprojection algorithm will be explored in greater detail in chapter 4.

CHAPTER 3

Hardware Platform

3.1 Description of Wildcard V300 Platform

The Wildcard Virtex 300 FPGA PCMCIA card was used as the hardware design platform for this thesis. The Wildcard platform was developed by Annapolis Microsystems Inc. The Wildcard contains a Virtex 300 FPGA and two memory arrays. Each memory array can contain 512k DWORDs of data that are accessed by a 20 bit address. The Wildcard uses a 32-bit CardBus I/O to access the chip. The CardBus controller is the interface between the card and the computer containing the PCMCIA card. This computer is known as the host computer. This interface is used to move data onto and off of the chip. The LAD bus routes into the processing element (PE), i.e. the FPGA, and from there the data can be routed to either the registers or the on board memory as seen in Fig 3.1 [2].

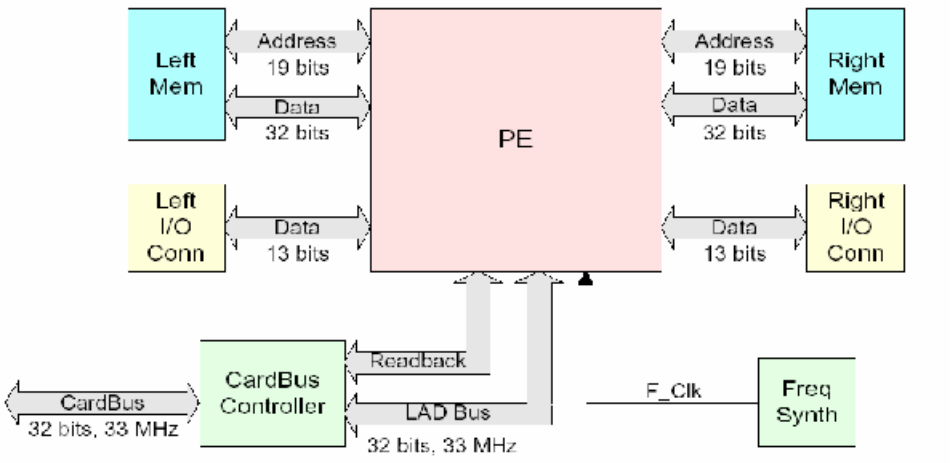

Fig. 3.1 Layout for the Annapolis Pro Wildcard PCMCIA V300 FPGA

In order for a designer to use the Wildcard platform, a host program must be used to control the card. The host program is written in C and is executed on the host computer. The host program controls the initialization of the clock, memory, and interrupts. An API library is included with the Wildcard Windows software and is used in the development of host programs. The library contains a number of function calls specifically related to the Wildcard. Each function call interacts with the Wildcard's driver to access the LAD bus and perform the desired function [2].

The Wildcard application software contains a set of simulation and application templates that the user can use to test the logic of and develop a synthesizable design for the Wildcard. The simulation software is designed to use Microsim and the application software was developed for Synplicity 6.20. The user adds the application VHDL code to the templates and performs either simulation or synthesis. Subtle changes to the VHDL code and makefiles are required depending on the operating system (OS) and revision of the Wildcard that is being used to implement the design. In the VHDL, the user will have to change the version declaration to the revision of the card being used where the user port maps the Clock_std_if. Similar changes will also have to be made in the several of the source files provided by Annapolis. This will ensure that when the make software provided by Annapolis performs the P&R on the design, it will map the design to the correct platform.

Changes to the makefile include editing the make_shell.bat file so that the correct OS is being used and that the correct path to the Xilinx Tools is provided. Several executable files located within the Xilinx tools directory will be used by the makefile to complete P&R of the design, so it is imperative that the designer have a copy of the tools

installed and that the makefile be edited to search in the correct directory. Once changes to make_shell.bat are complete, the designer should edit the project_m1.mak file. The project_m1.mak file is a script file used by the make tool. This file identifies which programs to use in the P&R, as well as the settings and revision of the Wildcard being used. The designer can declare any particular design options that might be required during the Mapping or Place and Route. The user should make sure that the proper revision of the card is being used as well as the correct card and speed. The particular card used in this thesis was a RevB Wildcard which contained a V300bg352, speed grade -4.

As the primary communication method used by the Wildcard to communicate with the host computer, the LAD Bus plays a vital role in any design using the Wildcard and therefore requires attention. The LAD Bus is used to connect the PE on the Wildcard to the Card Bus Controller on the host computer. When a design needs to transfer data from the host computer to either the PE or the RAM on the Wildcard, the data must go through the LAD Bus. All data transfers to and from the Wildcard are initiated by the host computer. When a data transfer is initiated, a VHDL design must be currently running on the PE that is setup to handle the passing of the data. The method of delivering data is dependent on where the data is to be transferred from, in the case of a read from the card, or where the data is going to be stored, in the case of a write to the card. The user can either setup the data transfer to be handled in a case statement, where the data to be transferred is dependent on the address being called at the time. In this example the data is stored as signals in the VHDL design. The designers at Annapolis

also developed a RAM Block which can be used to quickly pass data through the PE to the on board RAM.

Direct Memory Accesses (DMA) is used by the Wildcard system as a means of transferring data to and from the memory on the card to the memory on the host computer. Using DMA to transfer data frees up the host computer's processor from having to work on the memory transfer, giving the Wildcard user the freedom of performing any number of other calculations while data is being transferred. However the DMA system requires that a block of contiguous memory be allocated to the DMA function prior to performing the data transfer. This allocation of memory is required whether the data is coming into or leaving the host computer. Unfortunately the DMA system for the Wildcard was still under construction by Annapolis Microsystems at the time of this thesis, so DMA was not used in the implementation of the algorithm. Mention of the DMA is done here as a reference for future work, as utilizing such a feature with this application would be highly desired. Further discussion on the uses of DMA with this thesis is discussed in Chapter 6.

3.2 Design Methodology using Wildcard

The development of the host program is the first step in creating a Wildcard based application. Development of the host program begins by first collecting any data required by the hardware application. Any preprocessing of the data that is required can be performed by the host program before being transferred to the Wildcard's memory. The

function calls to the Wildcard API Library are made as the desired function is requested by the C code.

Before any application can be run on the Wildcard, the card must be initialized. The first step in initializing the Wildcard is designating the Wildcard's PCMCIA slot. This function will check to see if the Wildcard is actually in the designated slot and will initialize the Wildcard's Lad Bus so that interaction between the host computer and the Wildcard can take place. Next the power to the card is turned on and the global reset line in enabled. Enabling the global reset line will clear all of the flip-flops on the card and hold them cleared until the reset is de-asserted. This disables any application on the PE from running. This function can be utilized in a design, should the user wish at any point in his design to clear out all the registers and set all of the signals in an application back to zero. After asserting the reset line, the PE is programmed with the .x86 file of the desired HDL design. The design will not begin processing until after the host computer de-asserts the global reset, so the user has time to perform any data manipulation that is required before the application begins processing. The interrupt is then reset and the global reset is de-asserted. This will allow the design to begin processing. The user can then either enable or disable the interrupt line, depending on the needs of the design. It is recommended by Annapolis Microsystems to designate the interrupt one way or another, as leaving it floating might cause the interrupt to fall into an unintended state.

Once initialization is complete on the Wildcard, the PE process is running and the host computer is either, processing more data, sending data to the card, or waiting for the design to complete its process. The host computer can be made to wait by either utilizing a wait function in C, or the interrupt line provided by the Wildcard. The interrupt is a

simple and convenient method of indicating to the host computer that the process is complete. However any time the interrupt is used by the design, it must then be re-asserted by the host computer before any other interrupts on that card can be performed again.

CHAPTER 4

2D Backprojection

4.1 Description of Algorithm

The backprojection algorithm is the most time consuming aspect of 2D PET image reconstruction. Due to the large amount of data used as well as the numerous iterations performed, backprojection is the bottleneck in image reconstruction. A description of the algorithm and reasons for the delay will be shown below in order for the user to fully understand the driving force behind the pursuit of this thesis.

The backprojection is thought of as the adjoint to the forward projection of the data done during data collection by the PET system. We can mathematically describe the 2D backprojection as:

$$b(x, y) = \int_0^{\pi} d\phi p(x_r, \phi)$$

where $b(x, y)$ is the image reconstruction array and $p(x_r, \phi)$ is the filtered sinogram data that is to be backprojected. We have to repeat the integration for each point in $b(x, y)$ in order to complete backprojection of the data. The large number of iterations involved is the primary reason the backprojection is the most time consuming part of image reconstruction [1].

The manner in which backprojection was performed for this thesis was done by taking a given ϕ in $p(x_r, \phi)$ and 'back-propagating' for each value of x_r (Fig 4.1). This

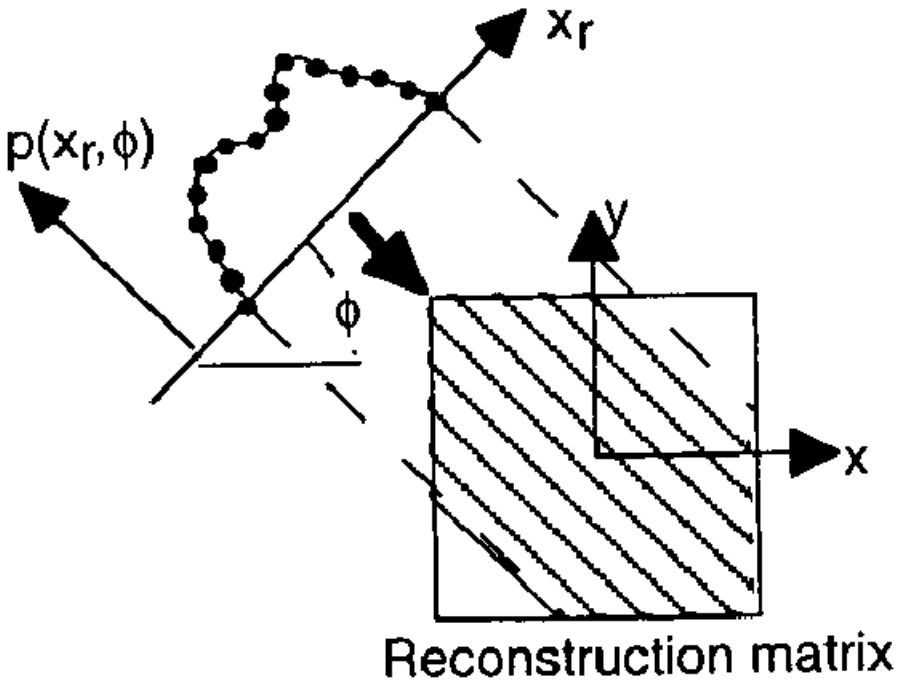

Fig 4.1 Illustration of Backprojection in 2D

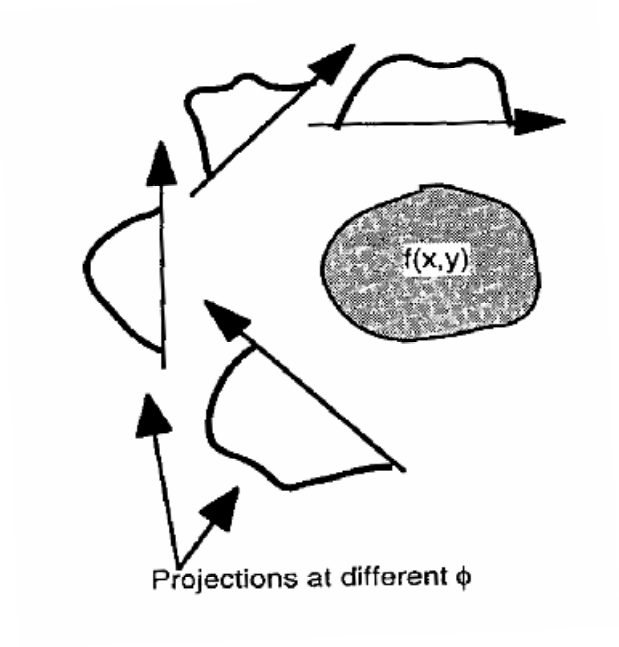

Fig. 4.2 Demonstration of Backprojection through changing ϕ

process is repeated for each x_r (Fig 4.2) by incrementing through the reconstruction array as appropriate to each value of $p(x_r,\phi)$. After all the values for $b(x, y)$ are computed for the given value ofϕ, the process is repeated for the next value of ϕ until backprojection has been performed on each value of ϕ given.

Before backprojection can be performed, however, the sinogram data must be filtered in order to prevent a Gaussian distribution of the data across the 2D image plane. However the filtering algorithm is beyond the scope of this thesis, although a brief description of it has been included for the personal edification of the reader. If filtering is not performed on the data, there would be a large collection of data at the center of the image that would tend to be distributed out towards the edges of the image in a Gaussian distribution. A method of filtering the sinogram data has thus been developed to "smooth out" the image so that the results from backprojection provide a clear image. This method of image reconstruction is known as FBP (filtered backprojection) and is done as follows [1]:

1. Take the Fourier Transform of the projection $P(v_{xr},\phi) = F_1\{p(x_r,\phi)\}$ for a givenϕ
2. Filter the projection in frequency space $P^F(v_{xr},\phi) = |v_{xr}|P(v_{xr},\phi)$
3. inverse Fourier Transform the filtered projection $p^F(x_r,\phi) = F_1^{-1}\{P^F(v_{xr},\phi)\}$
4. backproject the filtered projection $f(x, y) = f(x, y) + \Delta\phi \cdot p^F(x_r,\phi)$ for all x_r
5. Repeat steps 1-4 for each $\phi : 0 < \phi < \pi$

For this thesis, filtering of the sinogram data has been performed prior to performing backprojection.

4.2 Implementation of Algorithm

The backprojection algorithm requires numerous iterations to be completed. This made it necessary to develop a method that would limit the design in order for it to be implemented into hardware. The design steps taken as well as the HDL implementation of the backprojection algorithm are described below. Also the process of designing the system for the Wildcard V300 platform is detailed. The core focus for this thesis is the design and implementation of the backprojection algorithm in a hardware system.

The project has been designed to handle a sinogram of size 128x128. That means that there are 128 arrays of data, represented by ϕ in the above equations, each containing 128 points of data, x_r from above. For 3D PET there would be a 3rd number associated with the sinogram that would indicate the number of image planes that are to be created. For this project, we are only working with 1 plane of data since it is designed for 2D image reconstruction. The design will generate a 2D reconstructed image that is black and white and has a size of 128x128 pixels.

As a means of minimizing the number of iterations and being limited by the Wildcard chip, it was necessary to break the algorithm down so that it performed the calculation for only one value in $b(x, y)$ at one value of x_r for any given ϕ. The

backprojection is done as a separate hardware process. When the appropriate data flags are set the process will start. The process will use the offsets to determine which values in the sinogram array will be used in the calculation of that particular data point. Linear interpolation is used since a point in the sinogram may not, and usually does not, fall on one particular pixel, in fact the value is found somewhere between two sinogram values.

Once the process determines which portion of sinogram data it will need and calculates the fractional values, the two sinogram values are multiplied by one of the fractions and the results are added together. The resulting value is the image pixel at that point for that particular sinogram. The image value is then stored as a signal and the interrupt is triggered to indicate to the host computer that the pixel is ready and that the data transfer can now take place. The process then proceeds to the next x_r and calculates the value of $b(x, y)$ for that data point. The process will perform backprojection on each value x_r for that particularϕ, and then move onto the nextϕ. This process will continue for every ϕ angle. At the end of the ϕ angles, backprojection is complete for that sinogram and the process moves onto the next sinogram slice (See Fig 4.3).

The Wildcard V300 platform called for a "Host Program" to be written in C and run on the host computer that contained the Wildcard in one of its PCMCIA card slots. This host program initializes the Wildcard, sets the frequency of the physical element's (PE) clock, stores the sinogram data and offsets in the memory banks located on the card, downloads the x86 data file and triggers the reset flags allowing the algorithm to begin processing. The x86 file contains the gate level layout the V300 FPGA on the Wildcard

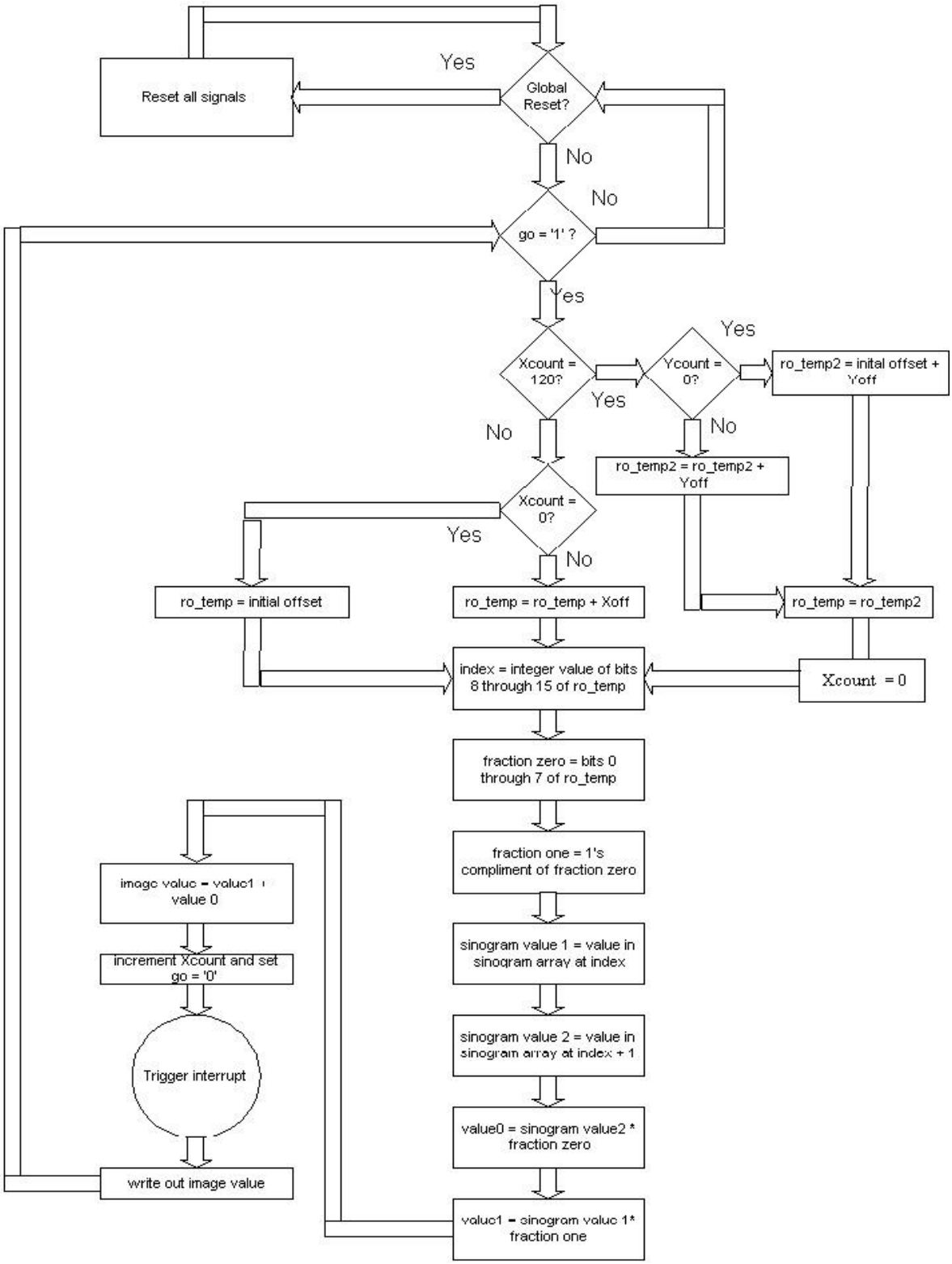

Fig. 4.3 Block Diagram of VHDL Program

will use to perform the backprojection algorithm. The host program is primarily in charge of powering up the card before it is used, controlling the input and output from the card and shutting the card down after the application has run its course. The host program will be used to initialize the sinogram data and offsets before delivering them to the PE on the Wildcard via the LAD Bus (See Fig 4.4).

One major difficulty in performing an algorithm in fixed point hardware that is commonly done in floating point notation, is determining how to maintain accuracy in the data when performing calculations. Since the sinogram data used to create the test data for this thesis was done as floating point data, it was necessary to convert the data to a fixed point notation before performing any calculations on the data in hardware. 8 point decimal precision was decided upon as an accurate estimate of the original input data. In order to be able to perform the algorithm and maintain decimal precision in fixed point notation, without the need for extra coding that is designed to keep up with the decimal place, a bit of data manipulation was done prior to the data being sent to the chip.

The data from the sinogram is read into an array and stored as floats. The array is scanned by the host program to determine the minimum value in the sinogram. Once the minimum value has been found, each value will have the magnitude of the minimum added to it. This process is called normalizing the data. In this case the data is being normalized to zero. After the data is normalized, each point is multiplied by 256. Multiplying it by 256 will perform a basic arithmetic shift left on each value by 8 points. The data is then stored as 32 bit unsigned longs and sent to the Wildcard via the LAD Bus where it is collected and stored in an array of signals. The offsets are simply multiplied by 256 in order for them to maintain the same level of precision as the

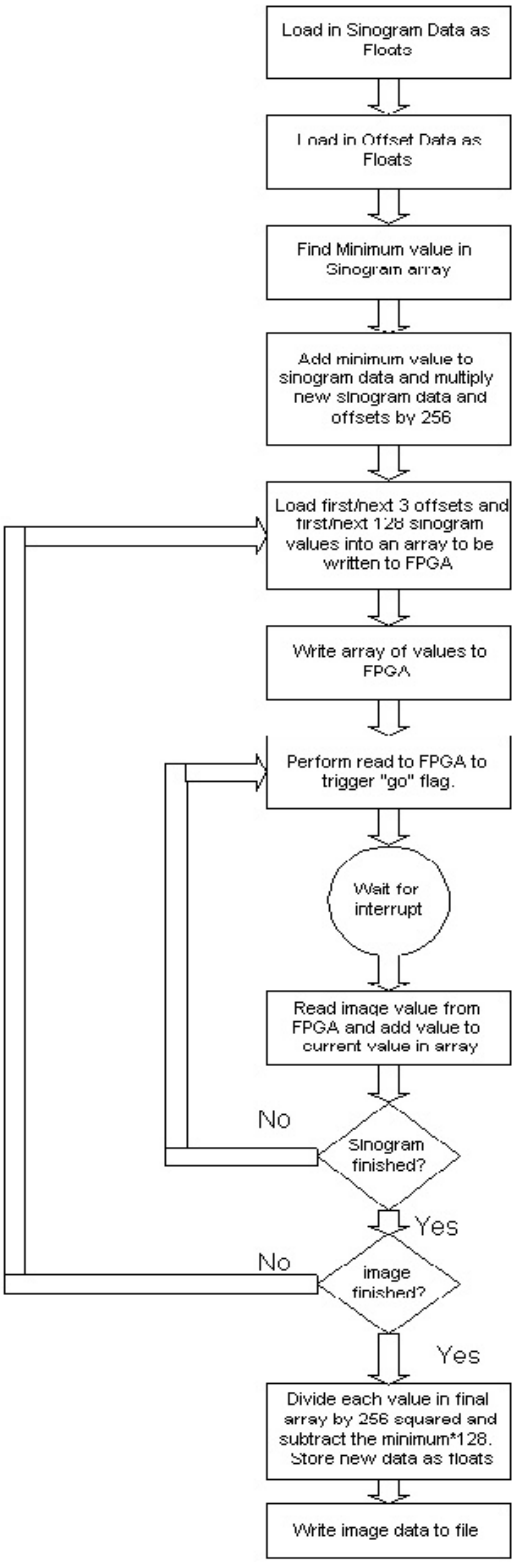

Fig. 4.4 Block Diagram of Host Program

sinogram data. However the VHDL is set up to handle negative offsets, in the cases where negative values are present. In these situations, the image value is set to zero as the offset is referencing a value outside the range of the sinogram for that particular slice. When the VHDL encounters a negative offset that is added to a positive offset to determine the offset for that value, the values are done as 2's complement addition. When the data is converted from float to unsigned long, any values that are negative are converted to the 2's complement of that value in 32 bit format. For example, when the 32 bit 2's complement of a number is added to a value that causes the value to become larger than what a 32 bit number can handle, i.e. what should be a 33rd bit to go high, the value in 32 bit notation will be equal to the value of the positive number plus the negative number.

After backprojection is performed on the sinogram data and the entire image has been collected in the host computer in an array, the previous data manipulation is reversed with a slight alteration. Since the backprojection requires that two of the values be multiplied together, the conversion to return the data to its original format will be different. The data will thus be divided by the square of 256 then the minimum value is subtracted from the result. At this point the numbers are also converted to a float format, to match with the standard of the given test images. Once this final calculation is performed the final image is stored in a file on the host computer and can be viewed.

Using the Wildcard platform added a level of complexity to the VHDL code used in the design. Pre-generated VHDL code was available from the vendor that provided the platform's user with the necessary code to access the chip's clocks, memory, registers and the LAD bus, which connected the card to the host computer. In order to properly use

the Wildcard however, additional code was added to the backprojection code for LAD bus operations. While this added a level of logic to the code, it is easily replaced by logic required for a different platform, should future work on this design be desired.

CHAPTER 5

Analysis and Overview of System

5.1 Description of Final Results

The final implementation of the work allowed for 128 sinogram data points along with the initial offset (R0), the X offset (?x) and the Y offset (?y) to be loaded to the PE of the Wildcard via the LAD Bus at one time. The values of the sinogram data are stored in an array of signals, and the offsets are stored as their own signals. Backprojection is then performed using that sinogram slice before more sinogram data is loaded to the card. The process will perform backprojection on one pixel at a time, trigger the interrupt flag then wait until a read is performed by the host computer which will trigger the "go" flag to start the process to calculate the value for the next pixel in the image. The previous steps are repeated until the value of each pixel in the image has been calculated for that sinogram, then the next set of sinogram data is loaded to the card along with the new set of offsets. This creates a delay in the overall speed of the application. Several clock cycles will expire between the time the design triggers the interrupt flag, indicating that processing on a pixel is complete, and the time the host computer finishes its processing and performs a "read" to the LAD Bus, triggering the "go" flag for the PE application. A timing diagram detailing the loss of time due to the additional operations, time lost due to data transfers and in reconstructing the data to a float format will be discussed in Chapter 5.3. This will give the reader a qualitative sense of the speed of the operation on the FPGA, as well as the necessity of developing a more effective method of storing and recovering data.

The approach of producing only one image pixel at a time was decided upon due to the limitations on size of the Wildcard. However the size constraint was not on the number of logic gates required, as the design only took 71, 000 of the 300,000 available logic gates on the Wildcard. The major size constraint was on the available routing space for the design. Due to the use of a large amount of 32 bit numbers in the design, the design used 96% of the available routing on the Wildcard. The lack of area required manipulation on the part of the designer in order to create a "true" backprojection on the Wildcard. Although the design does lack the ability to sum up the final image on the Wildcard itself, such a function would be straightforward to implement on a larger platform. The size constraint made caused an additional limitation in the validity of the algorithm. The multiplication between the two fraction values and the sinogram data is done as 32 bit multiplication in the test C code method of backprojection. It was necessary in the VHDL to limit the multiplication to 16 bit multiplication in order to decrease the routing enough that the design would be routable on the Wildcard. This limitation however did not cause a loss of data in either of the two test cases as neither set had values that were too large to be described by 16 bit numbers. But nonetheless the truncation of the values has the possibility of causing data loss.

The image output from the backprojection produced results that indicate that the design backprojects the data correctly. The images have a slightly different visual appearance to them than the test data generated by Dr. Newport as will be seen in the comparison of the test data given in Chapter 5.2.

5.2 Comparison of Design with Existing System

As a method of proving the validity of the backprojection produced by this thesis, two test cases were provided by Dr. Danny Newport of Concorde Microsystems. Images from the test data were created using an optimized floating point backprojection algorithm. Since the speed of the hardware system relative to the software system was a key factor in the motivation of this thesis, the analysis of this data is detailed separate from the data comparison. The timing analysis can be found in Chapter 5.3.

The final test results from the hardware backprojection of the data, in both cases, were within 2% of the results generated by the C backprojection. A waveform of the response of the data across the image plane was taken as a comparison of the response of the data in both cases. The images produced from the hardware backprojection however contained a sharp drop off in the value of the image along the edges of the image outside of the sphere of interest for the backprojection. The rotational manner in which images are recreated using backprojection creates a “sphere” in the center of the image. This sphere is where all of the data that a user is interested in seeing, so the loss of data along the edges of the image is of little importance; although the writer of this thesis has little reason to explain this rather strange phenomenon (See Figures 5.1 through 5.8).

From the final images the reader can see the areas around the edges of the image where the hardware and software implementations of the backprojection differ. However the “sphere” that contains the data which is the focus of backprojection can easily be seen in the images created from hardware. The waveforms serve to show that the two images

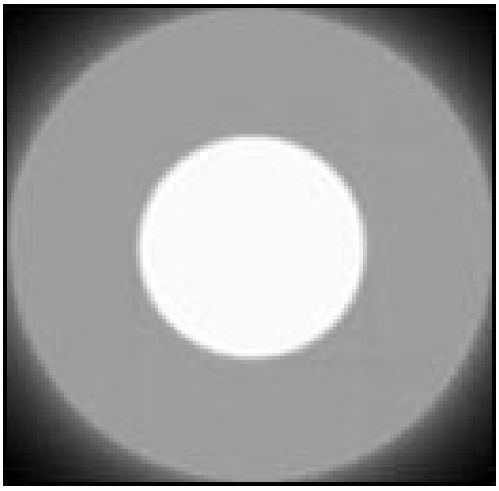

Fig 5.1 Hardware result of Hann data

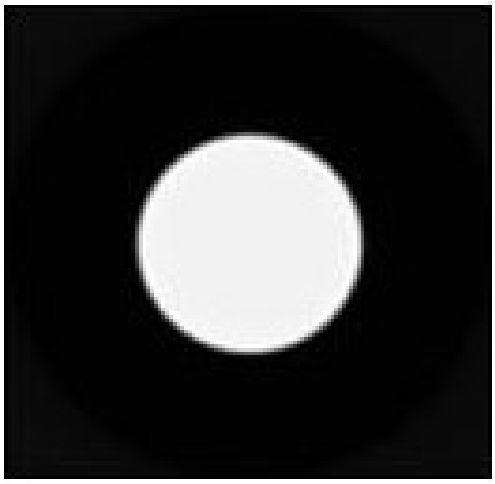

Fig 5.2 Software result of Hann data

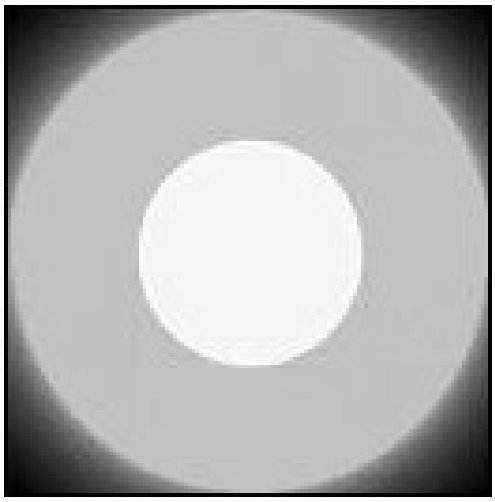

Fig 5.3 Hardware result of Ramp data

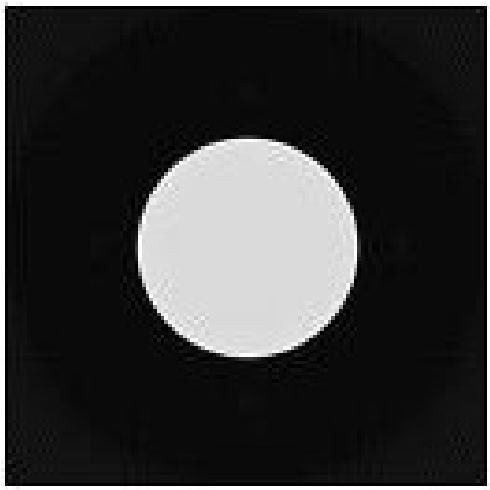

Fig 5.4 Software result of Ramp data

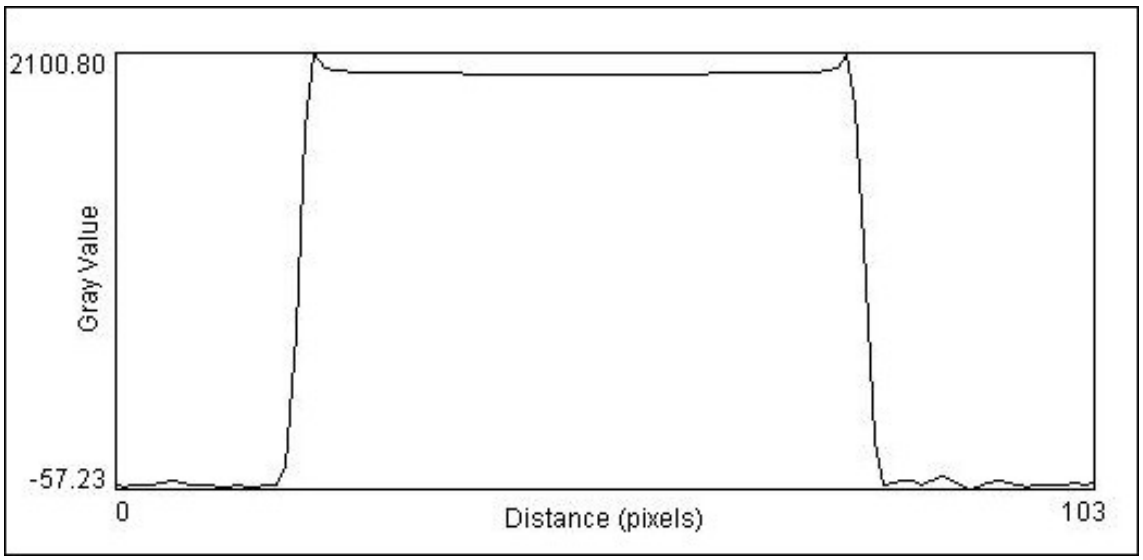

Fig 5.5 Waveform response of Hann data from Hardware Implementation

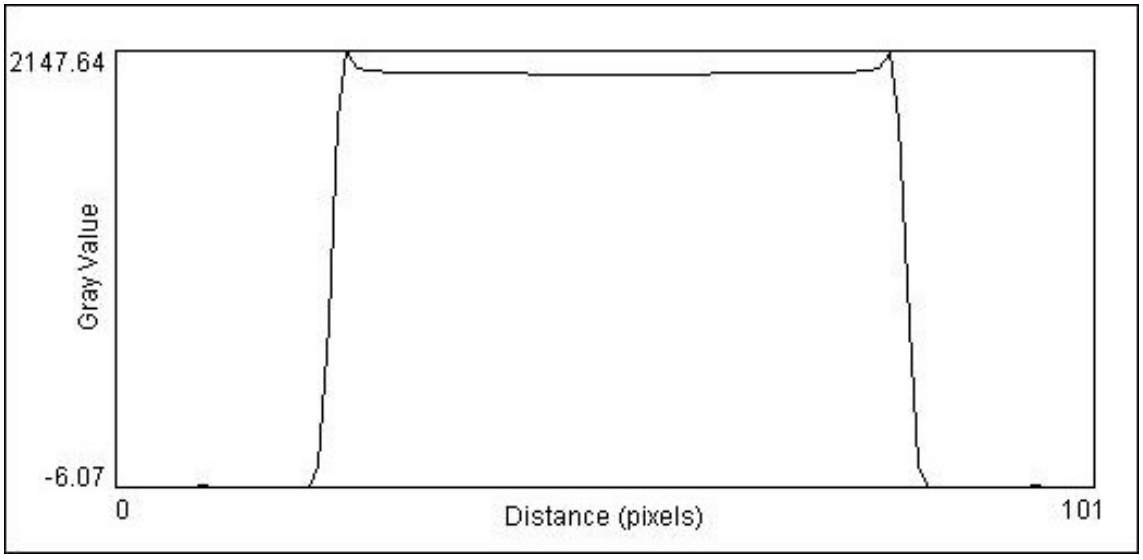

Fig 5.6 Waveform response of Hann data from Software Implementation

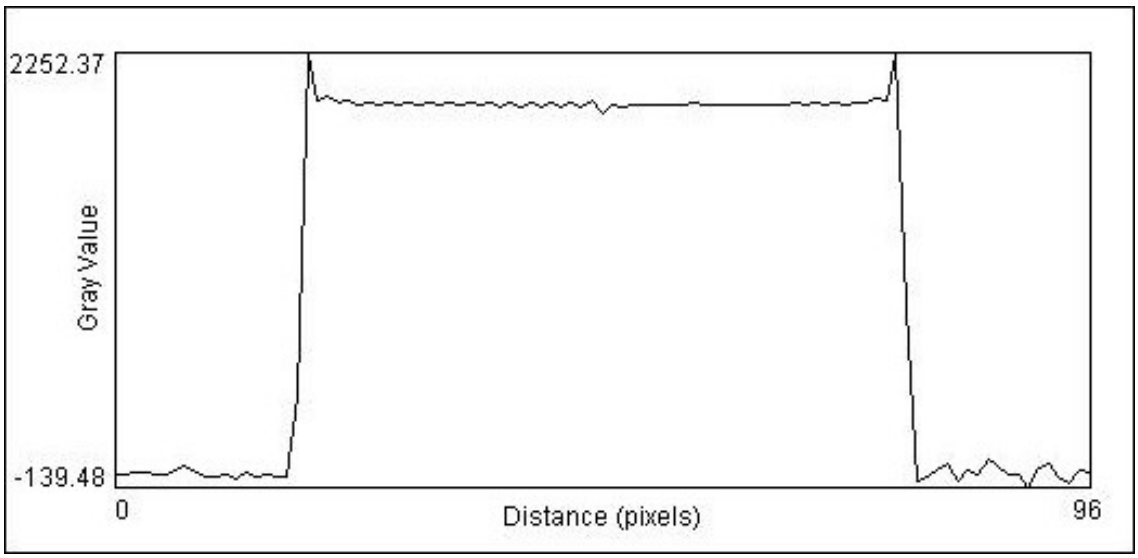

Fig 5.7 Waveform response of Hardware Implementation of Ramp data

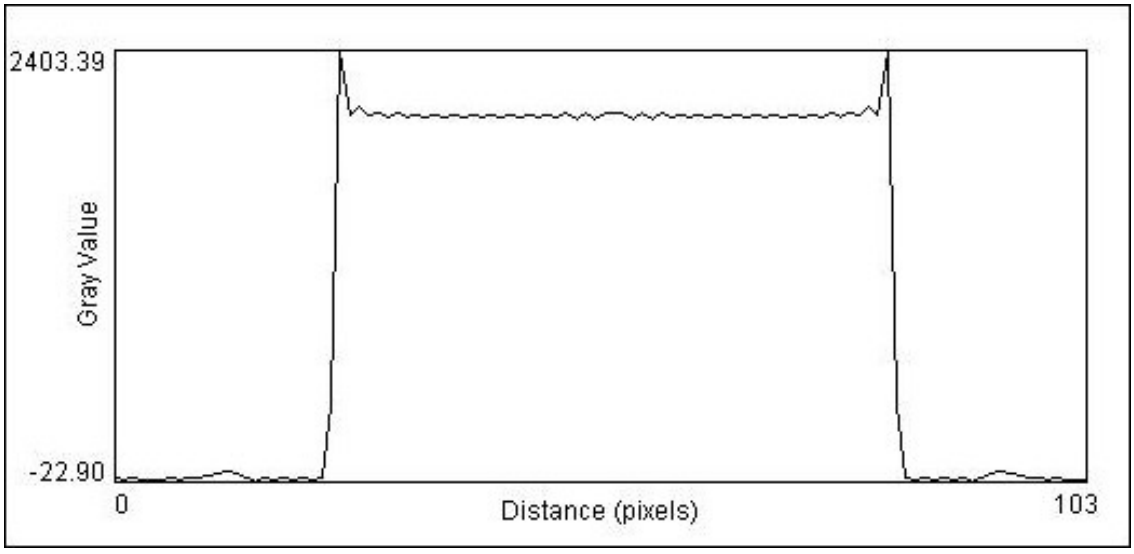

Fig 5.8 Waveform response of Software Implementation of Ramp data

are equivalent from a numeric standpoint. The difference in the images is due to the image viewing program used by the author. The program treated the lowest value as black and the highest value as white. Since the areas around the edges fall to roughly -5000, that area is treated as black, while the center of the image becomes white. The waveforms justify that the hardware backprojection of the image performed backprojection correctly, with a small measure of data loss. The loss of data can be explained by the manipulation of the data that was done before and after backprojection is performed. The loss of floating precision in decimal numbers when the sinogram data is converted to a fixed point representation will cause data loss that only becomes readily obvious when the final image is reconstructed after the numerous iterations performed during the backprojection.

5.3 Timing Analysis of Design

A timing analysis of the final hardware system was performed to show the relative speed at which the FPGA performs backprojection as compared with the C backprojection. It is asked that the reader keep in mind that the Wildcard system is running at only 30 MHz and the software counterpart that the timing was gathered from was running on an 800 MHz Pentium III processor. However the times can still be compared and the effective speed of the FPGA can be noted. In the analysis it will be seen that in order for a better implementation of this system to be designed, the major bottleneck will be the transferring of data to and from the FPGA system.

To check the timing of the hardware system the system was first run to check the time the system took to perform backprojection on a 128x128 sinogram image. Then a no-op for the VHDL segment was inserted into the system, so that every other function of the process was performed. This second condition includes times for data transfers and for storing of the data in an array on the host computer. The results were as follows:

Elapsed Time for Backprojection – 168 seconds

Elapsed Time with No-Op – 160 seconds

This test indicates that 8 seconds of the total processing time for the system are spent by the FPGA on backprojecting the image. When that time on a 30MHz system is compared to the 0.5 seconds of processing time spent by the software system on backprojecting the same image using an 800 MHz processor, the results of the hardware model are comparable. However as can be seen above, the total process took 168 seconds to perform backprojection on the whole image. Transfer times between the FPGA and the computer are the major bottleneck to this design. Therefore in order to develop a much faster system, the fewer data transfers that can be done, the faster the overall system will operate. Further discussion on speeding up the process will be discussed in Chapter 6.

CHAPTER 6

Concluding Remarks and Future Works

6.1 Summary

Although the final implementation of the thesis was not at the leading edge of today's standards of high performance parallel computing, the scope of the work was accomplished. Backprojection can be performed in hardware and given a more formidable system, would rival and surpass the speed and capability of the current software counterpart. The driving force for this thesis was to attempt backprojection on a hardware system and see if it could be first and foremost performed correctly and secondly if there would be any real benefit to exploring such avenues for future expansion. With the improving technology for FPGA's and ASIC's, along with upgrades to synthesis tools, applications such as this one may become more widespread. However it seems at the moment the limiting factor in developing more complex algorithms on hardware systems are synthesis tools. Many synthesis tools have no support for some of the more complex features of VHDL, which serves to limit a designer's ability to create more advanced systems utilizing FPGA's. With future enhancements to size, speed and capability, a solid backprojection algorithm for PET could be implemented that would offer users of PET a faster and more effective method for their image reconstruction.

The relative accuracy and attainable speed with which the final images were reconstructed by this thesis show the ability of FPGA's to perform complex algorithm that, up until now, have been performed solely using software systems. However

hardware systems still have the limitation of inflexibility. Many of the aspects of VHDL that allow for the type of flexible programming that can easily be done in software, are as of yet still un-synthesizable. This limitation will make it such that for each different image size that is desired, the same algorithm can be used. However modifications will have to be made to the code. With the new advances to Reconfigurable Computing that are being made, such modifications may soon be unnecessary and a system in which the user simply designates the size of the sinogram and the size of image that is to be reconstructed can be given to the FPGA and backprojection can proceed with no other outside modifications.

6.2 Future Work

One of the major constraints of this thesis was the relatively small size of the Wildcard FPGA. The design utilizes a great deal of routing area in the process and does not require as much logic as was expected. With larger FPGA's available, performing a full backprojection on an FPGA is highly possible. The best possible solution for a hardware application of backprojection would be an ASIC where more of the chip could be designated as routing space and would alleviate the large amounts of extra logic gates that were left over after the initial design.

Utilizing on board RAM and storing all of the sinogram data at one time would allow the design to have access to all of the data. Parallel processing of the algorithm could then be easily performed. Multiple processes could be set to run, each with the task of completing backprojection on a given sinogram. Once all sinograms are complete,

simply adding them up and returning the final image data would complete the process. If a large enough RAM block was allocated to the process this would alleviate much of the needed routing, as the design would decide using the offsets which sinogram values it will need and access them from memory.

One of the largest bottlenecks in this thesis was the use of the data array to load in the sinogram values from the computer. Unfortunately the Wildcard memory functions were not fully completed at the time this design was implemented so use of the on board RAM was limited to a very small degree. However as mentioned in chapter 2, Annapolis Pro Microsystems is developing a DMA for the Wildcard that will allow the system to by-pass the LAD Bus and to store data directly into the RAM. While the DMA is running the host computer and even the PE design can be running processes of their own. As a future development, a design could be implemented in which you would have the system performing reads and writes to the memory without need of involving the PE design, allowing it to work instead of having to delay while memory accesses are taking place. This improvement would greatly enhance the performance and capability of the system.

LIST OF REFERENCES

[1] Bernard Bendriem and David W. Townsend, “The Theory and Practice of 3D PET”, pp. 21 – 37, 1998

[2] Annapolis Micro Systems, Inc., “Wildcard Reference Manual”, pp. 10-1 – 10 – 23, 1999

VITA

Dana Eric Harrah was born in Roanoke, VA on August 10th, 1977. He graduated from East Central High School in Hurley, MS in May of 1995 and enrolled in the University of Tennessee, Knoxville's College of Engineering that fall. The first two years of college were spent in the Air Force ROTC where he gained some flight experience flying Cessna airplanes with the Civil Air Patrol. During his senior year of college, he enrolled in the Satori-Ryu Iaido class offered by Lance England. Shortly after Eric enrolled in sensei England's Isshin-Ryu Karate class and continued his studies of the martial arts. In August of 2001 he attained the rank of Sho-Dan, 1st degree black belt, in Satori-Ryu Iaido and shortly after received his Brown Belt in Isshin-Ryu Karate. After graduating from the UTK with his Bachelor's of Science in Electrical Engineering in May 2000, he began his graduate studies under the tutelage of Dr. Danny F. Newport.