# Real time identification of local surface properties of material using atomic force microscope - An FPGA based implementation

A THESIS
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Sourav Pradhan

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

Dr. Murti V. Salapaka

February, 2017

# Acknowledgements

I would like to express my earnest gratitude toward my parents for always inspiring me and believing in me. I want to take this opportunity to acknowledge the many sacrifices they have made over the years; and to tell them that I love them and would remain indebted to them forever.

I am grateful to my advisor Dr. Murti V. Salapaka for his constant support at every step of my research work. I consider myself fortunate to have had the opportunity to work under his guidance.

I would like to thank Dr. Greg Haugstad and Dr. Andrew Lamperski for serving on my thesis committee and providing their precious time and useful feedback.

I wish to thank Sayan Ghosal for patiently explaining all the complexities that were pertinent to my work and for providing me insightful ideas throughout the course of my research. I would also like to thank Saurav and Shreyas, for helping me in many ways: both academically and non-academically, not least of which was to make me feel comfortable, and an integral part of the group.

I want to thank my friends: Sudhanya, Pratik, Kartik, Kundan, Sourav, Zaid, Amartya and Poulomi for making my stay in Minneapolis a delightful one.

# Dedication

To those who held me up over the years- my parents, Mr. Krishna Prasad Pradhan and Mrs. Binapani Pradhan and my other family members.

# Abstract

System identification is widely employed for building mathematical models of manifold systems using statistical techniques. In this thesis, the application of system identification to atomic force microscopy using a real-time embedded solution has been reported. Atomic force microscopes are prevalent instruments utilized to explore material properties at the micro/nanometer level. A Field Programmable Gate Array has been chosen to harbor the design of the system identification module. The reported module has been successfully cascaded with an atomic force microscope to estimate local surface mechanical properties of materials. The design layout described in this thesis is not just applicable to commercially available atomic force microscopes, but to a large group of real-time signal processing units. Numerous simulations over multiple platforms and experimental results are presented to validate the accuracy and performance of the designed system identification module.

# Contents

# List of Tables

# List of Figures

# Chapter 1

# Introduction

System identification lays out the mathematical framework for the input-output relationships of a system. Recently, system identification methods have found application in characterizing stochastic and nonlinear systems [1, 2, 3, 4, 5] which are built upon the works of identifying linear systems [6, 7]. Techniques for identifying time varying systems in real-time have also been developed by the research community lately [8, 9, 10, 11] as opposed to the traditional methods of identification by post-processing of data. The embedded implementation of a real-time system identification algorithm with application to characterizing mechanical properties of materials using atomic force microscopes (AFMs) first reported in [12] has been described in this thesis.

Since the invention of AFM [13], it has become a tool of choice for interrogating materials in the micro/nanometer domain. Traditionally, the primary usage of AFMs was to investigate the topography of materials. However recently, the AFM community is also developing techniques to characterize the local surface mechanical properties of materials at the nano regime. Some application areas are polymer science [14, 15, 16], drug delivery [17], biology [18, 19, 20, 21, 22] and opto-electronics [23]. Many of these applications utilize either contact resonance mode methods [24, 25, 26, 22] or forced indentation methods [19, 21] where the AFM micro-cantilever tip is continuously in contact with the sample. While these contact mode techniques have been acclaimed for their simplicity and robustness, they lead to substantial mechanical wear and tear on the sample as well as on the micro-cantilever probe [27, 28]. This poses limitations on investigating soft samples, specially soft polymers and biological samples. To overcome this limitation, The AFM can be operated in the dynamic mode of operation in which the micro-cantilever probe is oscillated typically with sinusoidal external forcing near or at resonance [29, 30, 31]. In the dynamic mode of operation, the micro-cantilever tip interacts intermittently with the sample during each oscillation cycle. Such intermittent interactions create negligible tip-sample wear while scanning the sample surface. Since the dynamic mode AFM is lenient on soft samples, interesting techniques employing dynamic mode AFM with multi-frequency micro-cantilever excitations are developed to extract mechanical properties of materials [16, 32, 33, 34]. With the development of dynamic mode AFM novel system theoretical perspectives laid out in [35, 36, 37, 38] it is possible to view the dynamic mode operation with the perspective of equivalent cantilevers. Such perspective has led to the formulation of a real-time system identification method called the Bias Compensated Exponentially Weighted Recursive Least Squares (BCEWRLS) algorithm for estimating the parameters of equivalent micro-cantilever [12, 35]. The BCEWRLS algorithm with dynamic mode AFM gives the possibility to simultaneously estimate stiffness and dissipative properties of materials in real time. Hence this thesis adopts the BCEWRLS algorithm for AFM application. The parameters of AFM cantilevers, for example, the spring constant and the resonant frequency vary from one micro-cantilever to another. Further, the parameters of the same micro-cantilever tend to change gradually over time due to accumulation of dust and debris while imaging the sample. Thus the embedded system which implements the system identification module should be able to address these parametric changes.

Limited closed loop bandwidth and the exorbitant price of DSP based solutions make them unfavorable to our problem. While ASICs (Application-Specific Integrated Circuits) offer benefits in terms of superior performance and low power consumption, their applications are limited due to overprice, restricted debugging capabilities and long design cycles. Field Programmable Gate Arrays (FPGAs) provide attractive embedded solutions owing to the multitude of benefits they offer including reconfigurable capability, design reuse, high speed of operation and supported protocols [39]. FPGAs find their use in various domains like power electronics, areas requiring high parallel computations, automobiles, and aircrafts to name a few [40, 41, 42, 43]. Modern FPGAs can also cope with high precision data [44, 45] which is essential for the BCEWRLS algorithm. Inspired by these examples, the Xilinx ML605 Evaluation Board was selected which includes Xilinx Virtex 6 generation FPGA as the embedded platform to harbor the design of our system identification module.

This thesis is an extension of a journal article co-authored by me [46]. In this thesis, the design of a real-time system identification module which implements the BCEWRLS algorithm on the chosen FPGA platform has been reported. The whole module is divided into different functional blocks each handling its own computational tasks which are outlined in this thesis. These functional blocks have been designed with basic architectural backbones comprising of computational cores, input and output registers and clocks. The computational cores are responsible for fundamental arithmetic operations (addition, multiplication etc), some special functions (including square root operations and trigonometric functions) and also data type conversions (for example, conversion of fixed point data to floating point data). Applicability of such step by step design method is not limited to the provided example of system identification module but to a wide variety of systems where input data are processed and desired parameters are computed as outputs in real time within a few clock cycles of the sampling frequency. Extensive results are presented to verify the effectiveness of BCEWRLS algorithm and accuracy of the whole design. Further experimental data are presented which elaborates quantitative imaging of stiffness and dissipative properties of a polymer blend with the designed module. The developed FPGA module yields additional channels which provide surface mechanical properties in addition to conventional dynamic mode AFM images. The method is employable to a wide variety of samples, specially soft polymer and biological samples.

- Chapter 2 talks about the system identification algorithm, namely the Bias Compensated Recursive Least Squares (BCEWRLS) algorithm.
- Chapter 3 elaborates the employment of BCEWRLS algorithm for imaging dissipative properties and material stiffness with AFMs.
- Chapter 4 provides the design methodology along with the basic building blocks incorporated in the system identification module implemented on the FPGA.

- Chapter 5 validates the accuracy and performance of the proposed design with simulation and experimental results.

- Chapter 6 gives a brief discussion on possible application areas of the design thus concluding the thesis.

- Appendix A lists some programming codes used during the design

# Chapter 2

# System Identification using Bias Compensated Exponentially Weighted Recursive Least Squares Algorithm

## 2.1 System Dynamics

We consider a continuous time single input single output (SISO) system whose dynamics are given by:

$$\begin{aligned} x(\tau+1) &= Ax(\tau) + Bu(\tau), \\ y(\tau) &= Cx(\tau) + Du(\tau) + \xi(\tau) \end{aligned} \tag{2.1}$$

where $u(\tau)$, $y(\tau)$ and $\xi(\tau)$ are the input, output and the measurement noise of the system respectively. $\xi$ is assumed to be a zero mean white Gaussian process. Reverting back the dynamics given in (2.1) to the discrete-time $\mathbb{Z}$-domain, $y$ can be written as:

$$y(z) = \frac{\mathbb{N}(z^{-1})}{\mathbb{D}(z^{-1})} u(z) + \xi(z), \tag{2.2}$$

Let $\eta(z) := \mathbb{D}(z^{-1})\xi(z)$ denote the filtered noise. Applying Inverse $\mathbb{Z}$ Transform, (2.2) becomes

$$y(n) = \Phi^T(n)\Theta + \Psi^T(n)\Theta + \xi(n) = \Phi^T(n)\Theta + \eta(n), \tag{2.3}$$

where
$\Theta = \left[\mathbb{D}_1 \, \mathbb{D}_2 \, \cdots \, \mathbb{D}_i \, \mathbb{N}_0 \, \mathbb{N}_1 \, \cdots \, \mathbb{N}_j\right]^T$,
$\Phi(n) = \left[ -y(n-1) \; -y(n-2) \; \cdots \; -y(n-i) \, u(n) \, u(n-1) \; \cdots \; u(n-j)\right]^T$ and
$\Psi(n) = [\xi(n-1) \, \xi(n-2) \, \cdots \, \xi(n-i) \, 0 \, 0 \, \cdots 0]^T$
Here, the vector $\Theta$ captures the system parameters which are assumed to be time varying. Therefore, the algorithm which we employ must give us the estimator of the vector $\Theta$ (denoted by $\hat{\Theta}$) that converges before the system parameters change. Hence, this vector becomes the center of our interest and estimating it in real-time becomes the primary objective of our work.

## 2.2 Problem Formulation

We define the problem of minimizing the cost function which results in an exponentially weighted recursive least square estimate of $\Theta$. So, the optimization problem becomes:

$$C(n) = \min_{\hat{\Theta}(n)} \sum_{i=1}^{n} \lambda^{n-i} \left( y(i) - \Phi^T(i)\Theta(n) \right)^2, \tag{2.4}$$

where the exponentially weighted factor (also called as the forgetting factor) $\lambda$ takes into account the past observations for the current estimate of the vector $\Theta$. It has been shown that the solution to (2.4) converges with a bias due to the presence of the filtered noise $\eta(n)$ from (2.3) [12, 47, 48]. To remove this bias, a Bias Compensated Exponentially Weighted Recursive Least Squares (BCEWRLS) algorithm has been taken up in this thesis [12, 48].

The set of update equations which lead to the bias compensated estimator of $\Theta$ are:

$$
\begin{aligned}
&\hat{\Theta}_{LS}(n) = \hat{\Theta}_{LS}(n-1) + P(n)\Phi(n)\left[y(n) - \Phi^T(n)\hat{\Theta}_{LS}(n-1)\right], \\
&P(n) = \frac{1}{\lambda}\left[P(n-1) - \frac{P(n-1)\Phi(n)\Phi^T(n)P(n-1)}{\lambda + \Phi^T(n)P(n-1)\Phi(n)}\right], \\
&J(n) = \lambda\left[J(n-1) + \frac{\left[y(n) - \Phi^T(n)\hat{\Theta}_{LS}(n-1)\right]^2}{\lambda + \Phi^T(n)P(n-1)\Phi(n)}\right], \\
&\hat{\sigma}^2(n) = \frac{\left[\frac{1-\lambda}{1-\lambda^n}\right]J(n)}{1 + \hat{\Theta}_C^T(n-1)\Lambda\hat{\Theta}_{LS}(n)}, \\
&\hat{\Theta}_C(n) = \hat{\Theta}_{LS}(n) + \hat{\sigma}^2(n)\left[\frac{1-\lambda^n}{1-\lambda}\right]P(n)\Lambda\hat{\Theta}_C(n-1)
\end{aligned}
\tag{2.5}
$$

where $\hat{\Theta}_{LS}$ denotes the least squares estimate of $\Theta$, $P(n) := \left[\sum_{i=1}^{n}\lambda^{n-i}\left(\Phi(i)\Phi^T(i)\right)\right]^{-1}$, $\lambda$ is the forgetting factor, $J(n) := \left[\sum_{i=1}^{n}\lambda^{n-i}\left(y(n) - \Phi^T(n)\hat{\Theta}_{LS}(n-1)\right)^2\right]$, $\hat{\sigma}^2$ gives the estimate of the variance of measurement noise $\xi$, $\Lambda = [I_{n\mathbb{X}n}\,0; 0\,0]$ where $I$ is an identity matrix and $n$ denotes the size of the denominator polynomial $\mathbb{D}(z^{-1})$ in (2.2) and $\hat{\Theta}_C$ denotes the bias compensated estimate of $\Theta$. The set of equations given in (2.5) were implemented on the FPGA along with equations given in (3.7), (3.3) and (3.4) relating the connection of the system parameter estimates to the material properties using an AFM.

The application of BCEWRLS with respect to AFM is described in the next chapter.

# Chapter 3

# Identification of system parameters in the Atomic Force Microscope

## 3.1 Atomic Force Microscope Overview

An AFM consists of a micro-cantilever that can interact with a sample in the x, y and z direction. It achieves this with the help of a piezo-electric actuator. In traditional contact mode operation, the micro-cantilever is always in contact with the sample. A constant force is maintained between the sample and micro-cantilever's tip through the actuation of the sample in the $z$-direction by the piezo-actuator. This is accomplished through a feedback mechanism acting on the piezo-actuator, as the micro-cantilever scans the surface of the sample. The height image of the sample is obtained as the piezo-actuator moves in the $z$-direction and hence the topography of the sample is inferred from that. In contrast, in the dynamic mode of operation, owing to high quality factor of the AFM, the base of the micro-cantilever is oscillated with a sinusoid signal having frequency near the resonant frequency of the micro-cantilever. The oscillating micro-cantilever interacts with the sample intermittently only over a small duration of its orbit. Such intermittent tip-sample interactions alter the trajectory of the micro-cantilever which are utilized to infer properties of the sample. The dynamic mode operation considerably reduces the mechanical wear and tear of the sample as well as of the micro-cantilever compared to the contact mode operation [28]. Traditionally, in the dynamic mode, the amplitude and phase of micro-cantilever oscillations are used to interpret the sample properties and as feedback signals to actuate the z-piezo. The dynamics of the micro-cantilever play an important role in interpreting mechanical properties of materials with AFM as explained in the next section.

## 3.2 Equivalent Micro-Cantilever Model

The dynamics of the first mode approximate model of the micro-cantilever is given by

$$\begin{aligned} &\frac{d^2p}{dt^2} + \frac{\omega_0}{Q_0}\frac{dp}{dt} + \omega_0^2 p = F(t), \\ &F(t) = g(t) + h, \\ &y(t) = p(t) + \xi \end{aligned} \tag{3.1}$$

where,
$p$: deflection of the micro-cantilever tip
$\frac{dp}{dt}$: velocity of the micro-cantilever tip
$F(t)$: force per unit mass on the micro-cantilever
$g(t)$: dither excitation
$h$: nonlinear interactive forces between the sample and the micro-cantilever tip
$y(t)$: measurement of the position of micro-cantilever tip
$\xi$: measurement noise
$\omega_0$: radial resonant frequency of the micro-cantilever
$Q_0$: quality factor of the micro-cantilever

Generally, the base of the micro-cantilever is forced by a sinusoidal signal with a radial drive frequency $\omega_d = 2\pi f_d$ at or near the radial resonant frequency $\omega_0$ in the dynamic mode of operation. Using asymptotic techniques for weakly nonlinear systems [49, 50], it has been shown that in the dynamic mode of operation, the micro-cantilever tapping the sample periodically can be modeled as an equivalent micro-cantilever whose dynamics are given by [51, 36]

$$\frac{d^2p}{dt^2} + \frac{\omega_{eq}(a)}{Q_{eq}(a)}\frac{dp}{dt} + \omega_{eq}^2(a)p = g(t) \tag{3.2}$$

where $\omega_{eq}$ and $Q_{eq}$ are termed as the equivalent radial resonant frequency and the equivalent quality factor of the micro-cantilever respectively and depend on the amplitude $a$ of the oscillating micro-cantilever. It can be shown that [35]

$$\begin{aligned} \omega_{eq}^2 &= \omega_0^2 - \frac{2}{am}\phi_c, \\ \frac{\omega_{eq}}{2Q_{eq}} &= \frac{\omega_0}{2Q_0} + \frac{1}{am\omega_d}\phi_d \end{aligned} \tag{3.3}$$

where,
$m$: mass of the micro-cantilever
$T = 2\pi/\omega_d$: time period of oscillations and

$$\begin{aligned} \phi_c &= \frac{1}{2\pi}\int_0^{2\pi} \frac{\phi(a\cos\psi, -a\omega_d\sin\psi)}{m}\cos\psi d\psi, \\ \phi_d &= \frac{1}{2\pi}\int_0^{2\pi} \frac{\phi(a\cos\psi, -a\omega_d\sin\psi)}{m}\sin\psi d\psi \end{aligned}$$

The storage power due to tip-sample interactive forces is defined as

$$\phi_C := \frac{\omega_d}{T}\int_0^T \phi(p, \frac{dp}{dt})p\,dt$$

Further, the average power dissipated during each cycle of oscillation due to tip-sample interactive forces is defined as

$$\phi_D := -\frac{1}{T}\int_0^T \phi(p, \frac{dp}{dt})\frac{dp}{dt}\,dt$$

Using the results given in [35], it can be proved that

$$\begin{aligned} \phi_D &= a\omega_d\phi_d, \\ \phi_C &= a\omega_d\phi_c \end{aligned} \tag{3.4}$$

It should be noted that in (3.2) and (3.3), the parameters $\omega_{eq}$ and $Q_{eq}$ are dependent on the deflection amplitude $a$. Thus the time-varying parameter $a$ renders (3.2) to be a nonlinear time-varying system. But since the dither frequency and the time variation of the deflection $p$ is much faster than the time scale at which the deflection amplitude $a$ changes, it allows for a time window where the dynamics given in (3.2) can be assumed to be linear and time invariant, thus opening the scope of utilizing an estimation scheme that can identify parameters $\omega_{eq}$ and $Q_{eq}$ faster than the time-scale at which the deflection amplitude $a$ varies. Therefore, $\phi_C$ and $\phi_D$ can be determined using (3.3) and (3.4) once the estimates of the equivalent parameters $\omega_{eq}$ and $Q_{eq}$ are available. $\omega_{eq} = 2\pi f_{eq}$ and $Q_{eq}$ can be estimated using the BCEWRLS algorithm which is outlined in the next section.

## 3.3 BCEWRLS algorithm with application to AFM

Dynamics of a second order discrete time system corresponding to the continuous time dynamics in (3.2) are given by [12]

$$\begin{aligned} y(n) + b_1 y(n-1) + b_0 y(n-2) &= u(n) + \epsilon(n), \\ u(n) &= a_2 g(n) + a_1 g(n-1) + a_0 g(n-2), \\ \epsilon(n) &= \xi(n) + b_1 \xi(n-1) + b_0 \xi(n-2) \end{aligned} \tag{3.5}$$

where,
$y(n)$: digitally sampled deflection signal
$g(n)$: digitally sampled dither signal
$t = nT_s$: time at which the signals are acquired
$T_s$: sampling interval
$\xi(n)$: zero mean uncorrelated noise with variance $\sigma^2$

The set of parameters $(a_2, a_1, a_0)$ in (3.5) can be evaluated from the discrete time equivalent of the transfer function of the cantilever $G_0(s)$ which is measured free from sample. $G_0(s)$ is determined by applying a frequency sweep method around the first resonant frequency of the cantilever. Subsequently, the values of $a_2$, $a_1$ and $a_0$ can be fixed to these reference values. Thus, it follows from (3.5) that

$$y'(n) \quad = \quad \underbrace{[-y(n-1) \;\; -y(n-2)]}_{\phi^T(n)} \underbrace{\begin{bmatrix} b_1 \\ b_0 \end{bmatrix}}_{\theta} + \epsilon(n) \tag{3.6}$$

where,
$\theta$: vector of unknown parameters and
$y'(n) = y(n) - a_2 g(n) - a_1 g(n-1) - a_0 g(n-2)$

Both $\phi$ and $y'(n)$ are determined from measurements. To guarantee the real-time convergence of $\theta$, two additional sinusoidal excitation signals with frequencies symmetric around the principal excitation frequency are added to the dither signal. It can be proved that the equivalent micro-cantilever model still holds correct with these additional excitation frequencies [35]. The parameters $\omega_{eq}$ and $Q_{eq}$ in (3.2) can be calculated using the set of equations given in [48], once the estimate for $\theta = [b_1 \, b_0]^T$ becomes available. The equations are given by

$$\begin{aligned} s_1 &= \tanh^{-1}\left(\frac{1-b_0}{1+b_0}\right), \\ s_2 &= \tan^{-1}\left(\frac{\sqrt{(4b_0^2 - b_1^2)}}{b_1}\right), \\ \omega'_{eq} &= \sqrt{\left(\frac{s_1}{2}\right)^2 + s_2^2}, \\ \omega_{eq} &= \frac{\omega'_{eq}}{T_s}, \; Q_{eq} = \frac{\omega'_{eq}}{2s_1} \end{aligned} \tag{3.7}$$

The digital system which realizes the BCEWRLS algorithm on an FPGA is discussed in the next chapter.

# Chapter 4

# Hardware implementation of System Identification Algorithm

An interconnection of different hardware units functioning together make it possible for the realisation of our reported system identification algorithm as shown in FIG. 4.1. The next section gives a brief description of each of the hardware units employed in this work along with their respective contribution to the design of the system identification module.

## 4.1 Hardware Units

**1) Atomic Force Microscope:** The input to the Asylum Research MFP 3D AFM is a sum of three sinusoidal signals. These signals are generated using three different Hewlett Packard 33120A Waveform Generators and then added using an adder circuit. The implementation of the adder circuit was done on a bread-board using IC LF351N and resistors chosen to achieve desired signal gains, where the adder circuit was powered using a Hewlett Packard E3630A DC Power Supply. The output signal of the adder circuit $g$ is used to excite the base of the micro-cantilever. The AFM's output signal which is the micro-cantilever tip's deflection $p$ is measured using the controller box of the AFM.

**2) Data Acquisition System:** The analog signals $g$ and $p$ as well as output signals

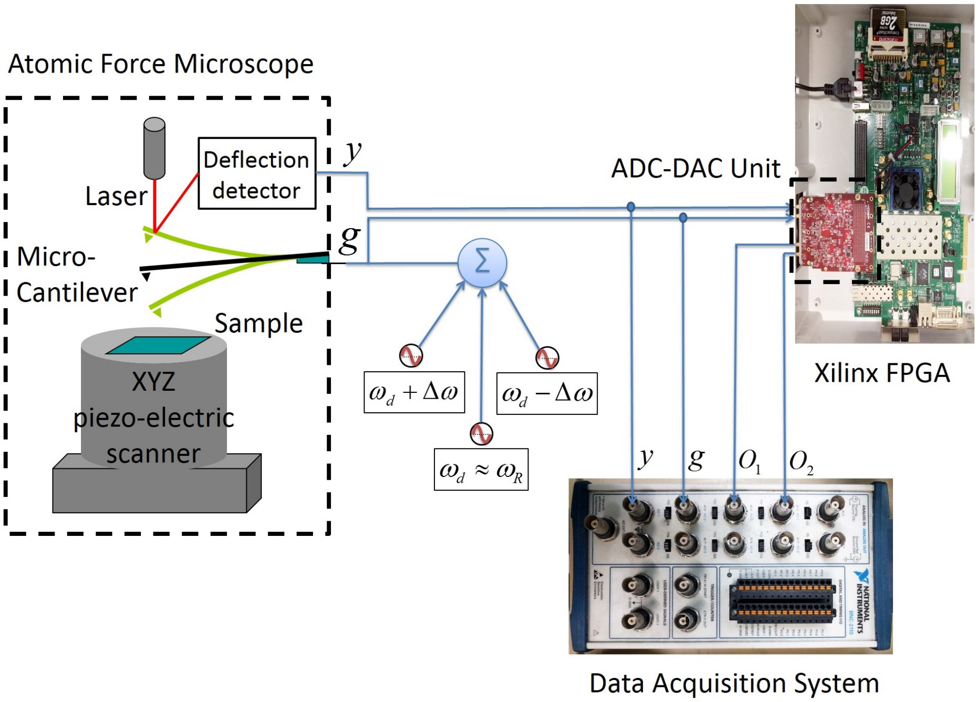

Figure 4.1: Diagram showing the interconnection of hardware units used. In the dynamic mode of operation, the AFM micro-cantilever is oscillated sinusoidally with drive frequency $\omega_d$ near or at resonant frequency $\omega_R$. The additional excitation signals at frequencies $\omega_d - \Delta\omega$ and $\omega_d + \Delta\omega$ are necessary for the system identification module. The system identification module which is implemented on FPGA processes $y$: the measured deflection of micro-cantilever and $g$: the micro-cantilever excitation signal. The outputs $O_1$ and $O_2$ of the FPGA module can be selected to route various equivalent parameters discussed later in this chapter

from the system identification module are acquired at a sampling rate of 2 MHz using a National Instruments BNC-2110 LabVIEW data acquisition system. Continuous monitoring of these signals helps identifying and debugging any errors on the FPGA's front as well on the adder circuit's side.

**3) ADC-DAC Unit:** This is the FPGA Mezzanine Card (FMC) 151 Analog-to-Digital (A/D) and Digital-to-Analog (D/A) Converter unit. This card offers two 14-bit A/D channels and two 16-bit D/A channels with a data transmission limit of 250 and 800 Mega-Samples per second (MSps) respectively. These channels are clocked by an internal clock of frequency 100 MHz. The 14-bit A/D and 16-bit D/A channels give

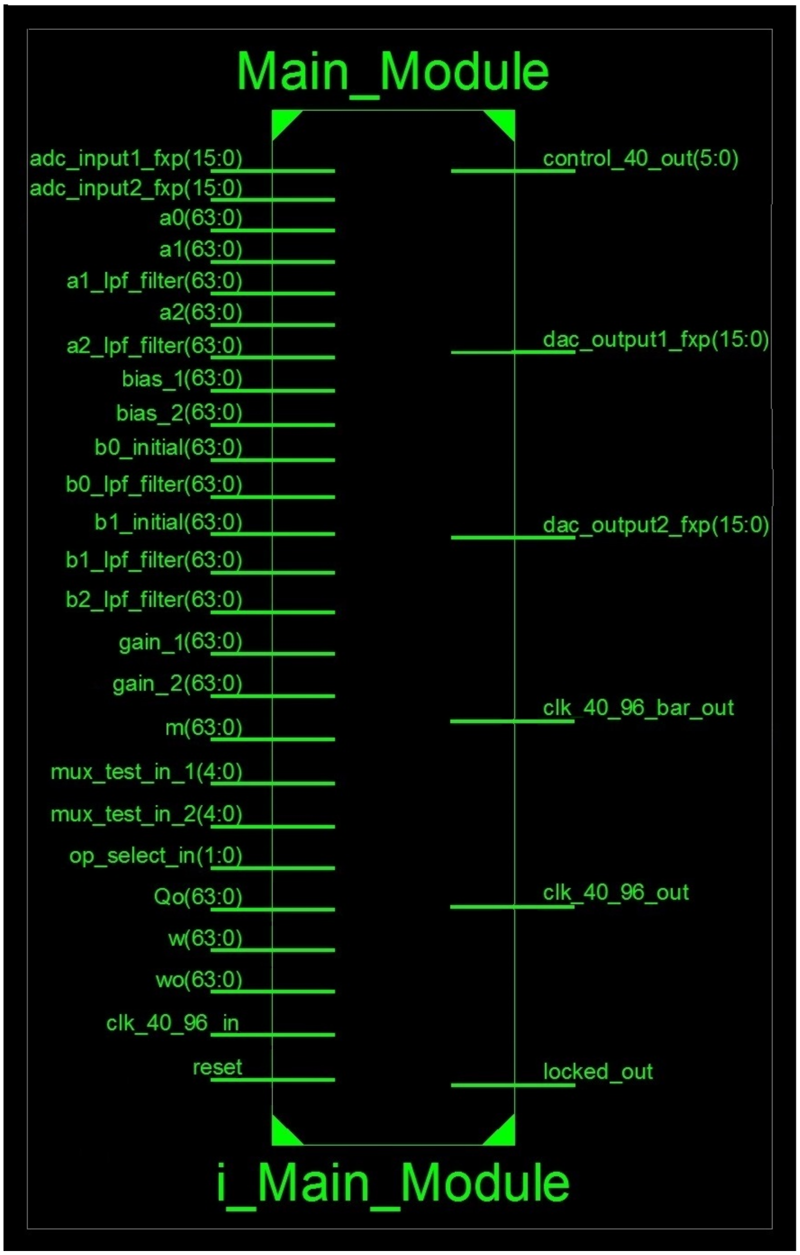

Figure 4.2: RTL schematic of Main Module

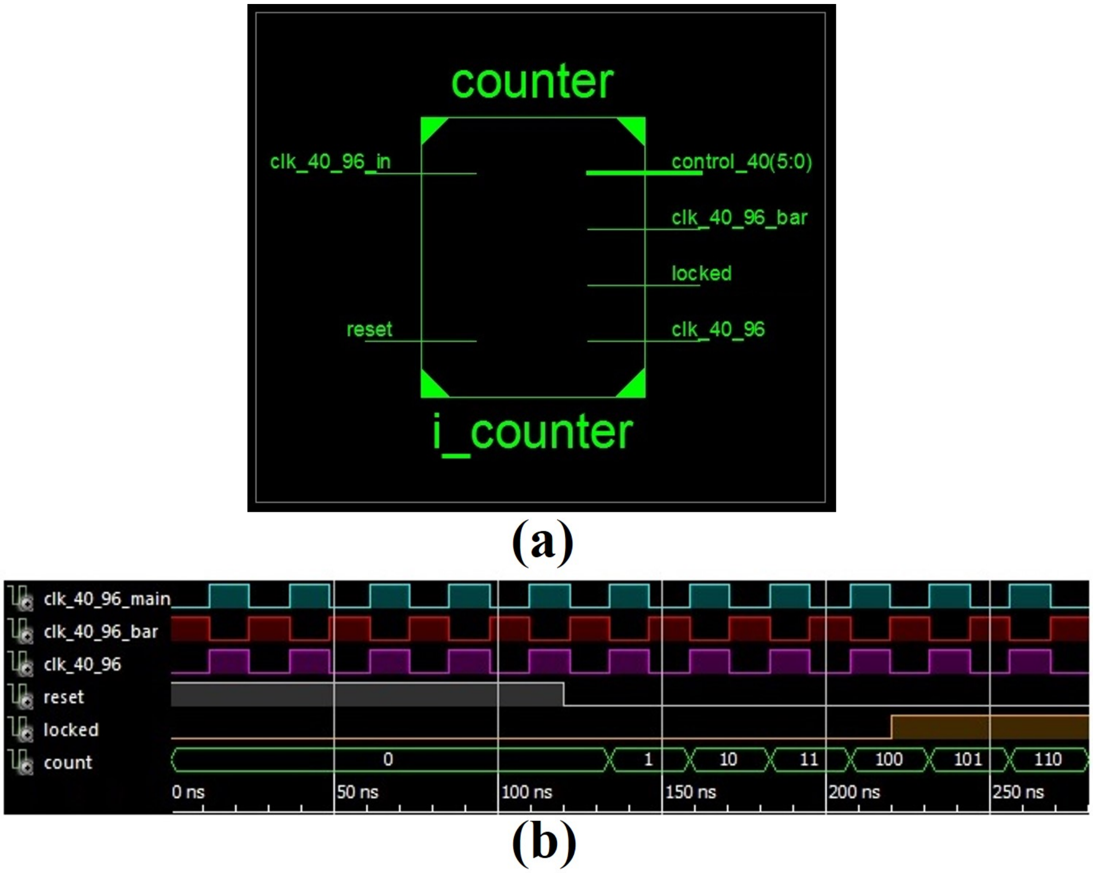

Figure 4.3: (a) RTL schematic of counter module and (b) diagram showing different clock signals generated within the Counter Module

pathway for the input of $g$ and $p$ signals and to the outputs of the system identification module from the FPGA respectively.

**4) Xilinx FPGA:** The implementation of the system identification algorithm has been done on the Xilinx ML605 Digital Signal Processing (DSP) Development Kit. The design process starts with creating a System Stub file which is programmed in VHDL [52]. Under System Stub, four different operational units are created, they are:

*i) Clock Generation Unit:* In this unit, internal clocks are derived using Mixed Mode Clock Manager (MMCM) from the FPGA inbuilt primary clock source of frequency 200

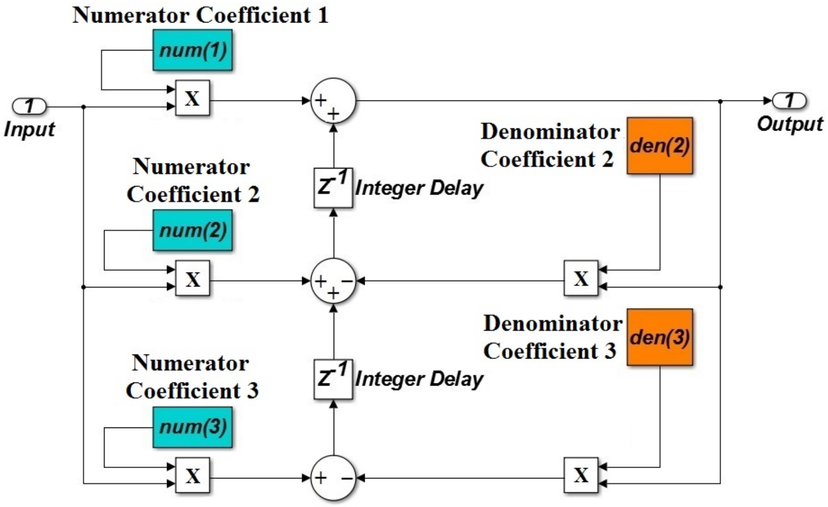

Figure 4.4: Direct Form II Architecture used to implement high-pass and low-pass digital filters inside the Biquads Module

MHz. These internal clocks are used to synchronize other units like the FMC ADC-DAC interface, Main Module and Microblaze Embedded Processor system interface.

*ii) FMC ADC-DAC Interface:* This unit caters with the installation of the FMC ADC-DAC unit on the FPGA. It provides an interface between the FPGA and the input and output signals coming from and going out to the A/D and D/A ports respectively.

*iii) Main Module:* This is the place where the internal clock signals, user-entered variables from the Microblaze Processor and signals from the ADC-DAC unit enter. These signals are routed to different sub-modules inside the Main Module where the system identification algorithm is realized.

*iv) Microblaze Embedded Processor System Interface:* This interface provides a platform to communicate with a personal computer through a USB-to-UART cable. The goal is to access a total of 21 parameters entered by the user during the experiment and route these parameters to the Main Module. The user has the option to update or store these parameters in the configurable memory spaces of the embedded processor system in real time. These parameters are shown in the Register-Transfer-Level (RTL)

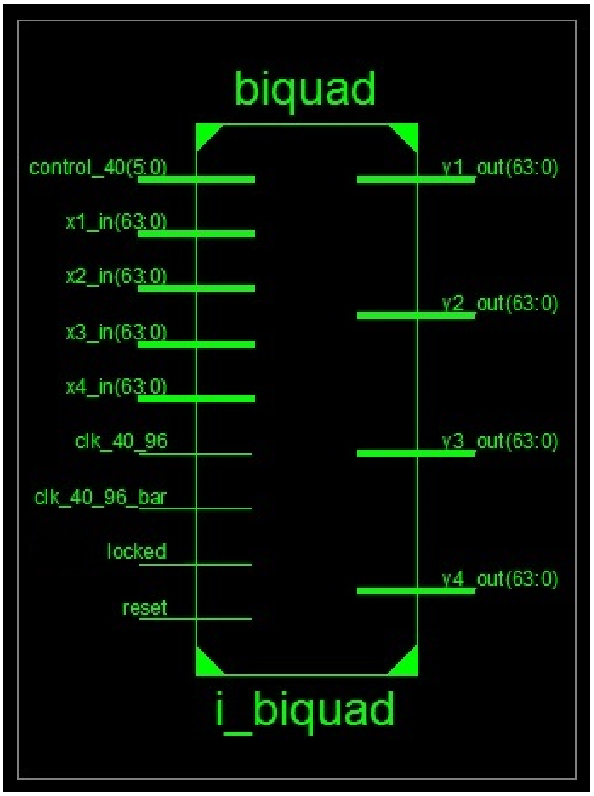

Figure 4.5: RTL schematic of biquads module

schematic of the Main Module in FIG. 4.2. This schematic is produced after the synthesis of the system identification module and opens up as a netlist file called NGR file. The 21 user-entered parameters entering the Main Module from the left side as shown in FIG. 4.2 are initial values (*b0_Initial, b1_Initial*) for the denominator coefficients $b_0$ and $b_1$ in (3.6), the numerator coefficients (*a2, a1, a0*) in (3.5), the mass of the micro-cantilever and driving radial frequency ($m,\ \omega$), the nominal values for quality factor and radial frequency ($Qo,\ \omega o$) and low-pass filter coefficients (*a2_lpf_filter, a1_lpf_filter, a0_lpf_filter, b1_lpf_filter, b2_lpf_filter*) used in demodulation of the deflection signal $p$ to get its amplitude $a$ in (3.3), values (*gain_1, gain_2*) for scaling up/down and terms (*bias_1, bias_2*) for shifting the outputs respectively of the system identification module, vectors (*mux_test_in_1, mux_test_in_2*) which can access different internal signals in the design and route those to the DAC inputs, allows the user to detect any errors in the design implemented within the REEP module and a vector (*op_select_in*) for selecting the

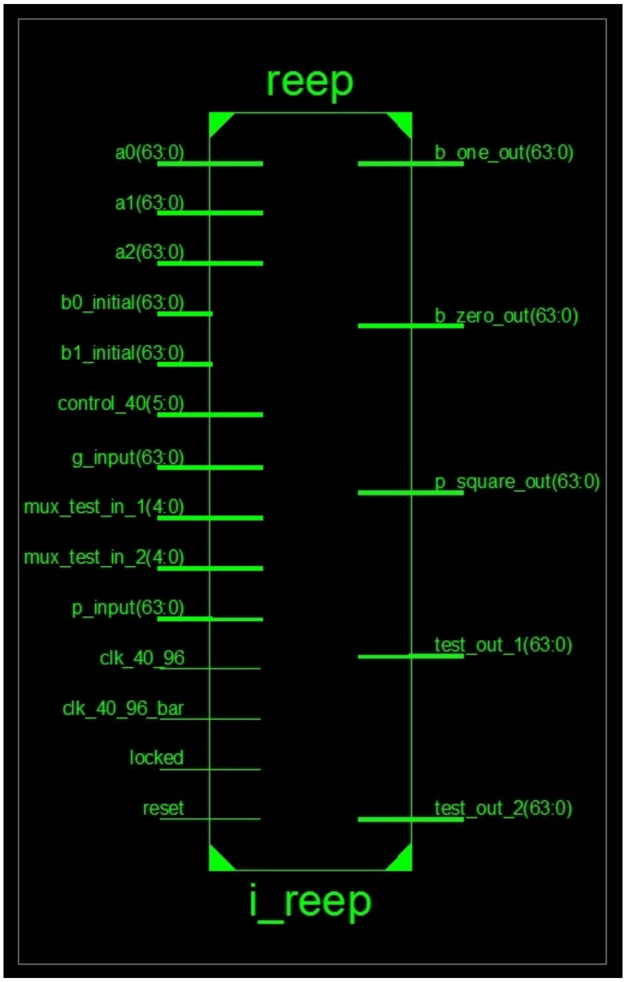

Figure 4.6: RTL schematic of REEP module

pair of outputs exiting the Main Module. The values enclosed within the brackets next to the vectors shown in FIG. 4.2 represent their respective bit length. For e.g. a vector with a representation of (15:0) would mean that it is a 16 bit vector. The explanation for other signals shown in FIG. 4.2 will be provided in the successive sections.

## 4.2 Components inside the Main Module

**1) Counter Module:** This block takes in the reset signal and the 40.96 MHz clock signal ($clk\_40\_96\_in$) as shown in FIG. 4.3 (a) and generates a synchronous 6 bit signal ($control\_40$) that repeats counting from 0 to $N-1$ with $N = 40$, a buffered 40.96 MHz

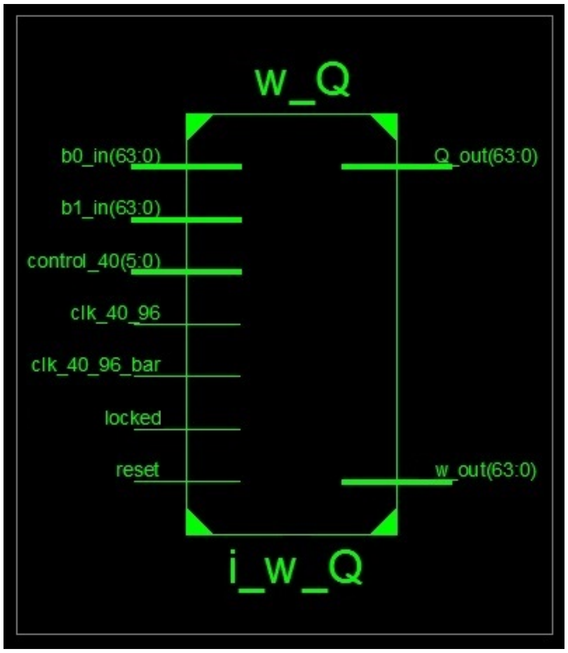

Figure 4.7: RTL schematic of $\omega$ and Q module

clock signal (*Clk_40_96*), a 40.96 MHz signal (*Clk_40_96_bar*) which is phase shifted by 180° from the *Clk_40_96* signal. These four signals drive all the modules as shown in their respective RTL schematics, thus maintaining synchronization throughout the Main Module. The timing scheme of these signals have been shown in FIG. 4.3 (b).The *count* signal captures the value given by *control_40*. The design inside the Main Module gets executed after the *locked* signal becomes 0 to 1. This is to ensure that all the clocking signals get available after the FPGA has been turned on.

**2) Registration Module 1:** The 23 incoming signals to Main Module including digital versions of dither signal (*adc_input1_fxp*) and deflection signal (*adc_input2_fxp*) and other 21 parameters coming from a personal computer through Xilinx Microblaze processor are registered in this module when *control_40* = 0 at the rising edge of *Clk_40_96_bar* clock followed by registering them at the rising edge of *Clk_40_96* clock.

**3) Fixed Point to Floating Point conversion Module:** After exiting Registration Module 1, the 16-bit fixed point signals *adc_input1_fxp* and *adc_input2_fxp* enter this module where they get converted to 64-bit floating point signals. The reason for

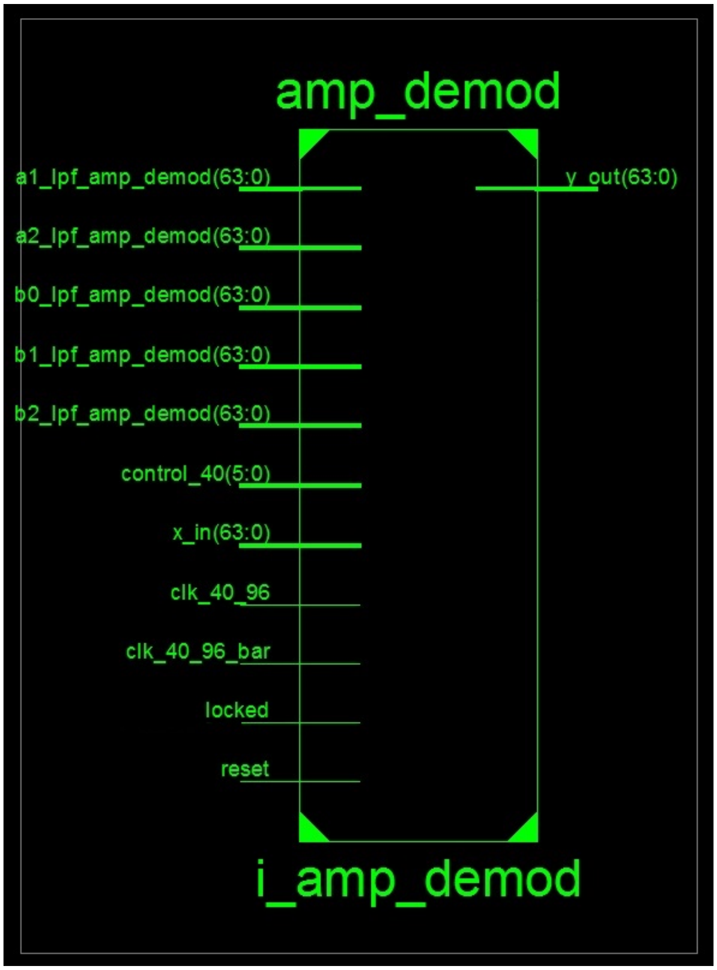

Figure 4.8: RTL schematic of amplitude demodulation module

this conversion is the high computational precision needed for the accuracy of the system identification algorithm. This module operates at the rising edge of *Clk_40_96_bar*.

**4) Registration Module 2:** The outcoming signals from the Fixed Point to Floating Point conversion Module get registered at the rising edge of the *Clk_40_96* clock in this module.

**5) Biquads Module:** This module consists of 2 sets of digital filters- first set has 2 high pass filters (HPFs) and the second one has 2 low pass filters (LPFs). Both of the HPFs and LPFs having a cut-off frequency of 2 kHz are implemented in this using Direct Form II architecture as shown in FIG. 4.4. The HPFs are used to remove biases and

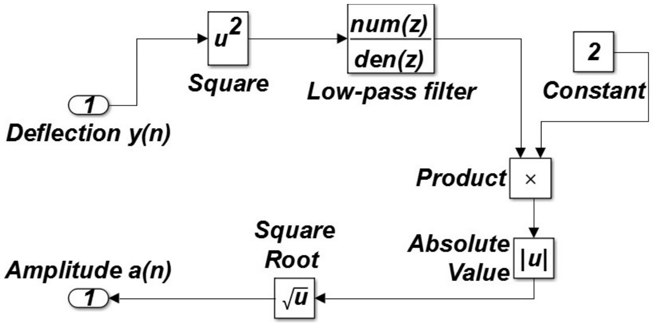

Figure 4.9: Scheme showing the demodulation of the deflection signal to get its amplitude

slow variations present in the input signals $g$ (*x1_in*) and $y$ (*x2_in*) as shown in FIG. 4.5 to yield the outputs *g_input* and *p_input*. The frequency spectrum of both the parameters $b_0$ and $b_1$ is found to be much lower than the excitation frequency which is roughly around 70 kHz, equal to the resonant frequency of the AC240TS micro-cantilever (supplied by Asylum Research). The frequency spectrum of both the estimated parameters $b_0$ (*x3_in*) and $b_1$ (*x4_in*) shown in FIG. 4.5 are mostly contained within the limits 0 kHz to 2-10 kHz. Thus, low pass filters (LPFs) are utilized to filter the high frequency noise present in these estimates. After filtering, these estimates are then registered first at the rising edge of *Clk_40_96_bar* clock followed by registration at the rising edge of *Clk_40_96* clock.

**6) Registration Module 3:** The signals coming out from the Biquads Module are registered first at the rising edge of *Clk_40_96_bar* clock followed by registration at the rising edge of *Clk_40_96* clock.

**7) REEP Module:** The high pass filtered signals *g_input* and *p_input* along with the other 7 parameters received from Registration Module 1 (*a2, a1, a0, b0_Initial, b1_Initial, mux_test_in_1, mux_test_in_2*) enter this module as shown in FIG. 4.6. This module computes the set of recursive update equations required by the BCEWRLS algorithm [12]. The outputs of this module are the bias compensated estimates of discrete time parameters $b_1$ (*b_one_out*) and $b_0$ (*b_zero_out*), the squared deflection signal $p_n^2$ (*p_square_out*) and 2 internal signals of this module (*test_out_1*, *test_out_2*) as shown in (4.6). The signal $p_n^2$ serves as an input to the Amplitude Demodulation Module which

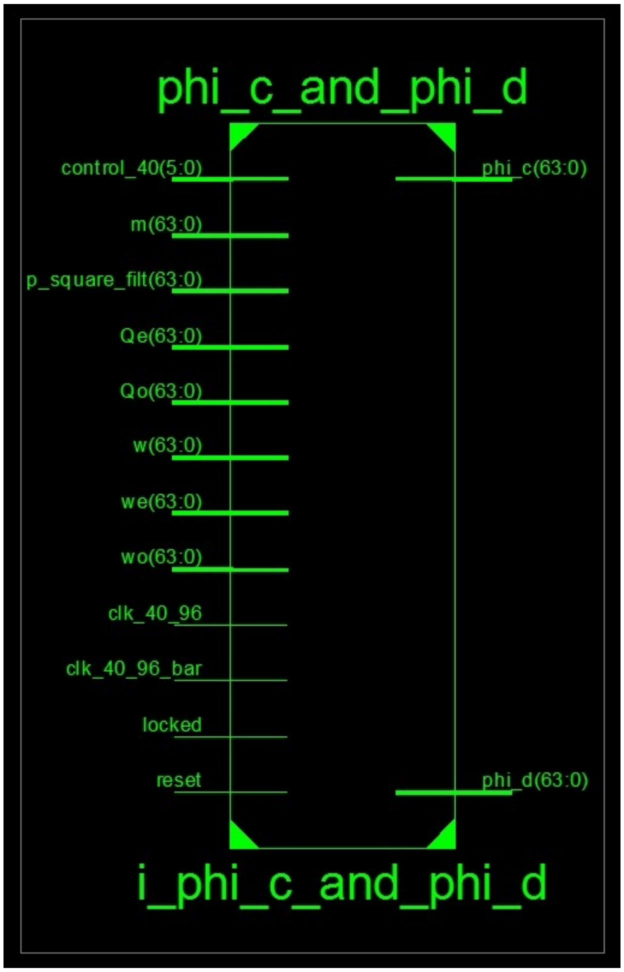

Figure 4.10: RTL schematic of $\phi_C$ and $\phi_D$ module

will be discussed later.

**8) Registration Module 4:** *b_one_out*, *b_zero_out* and *p_square_out* obtained from the REEP Module are sent to this module to be registered first at the rising edge of *Clk_40_96_bar* clock followed by registration at the rising edge of *Clk_40_96* clock.

**9) $\omega$ and Q Module:** The low pass filtered estimates of the discrete time parameters $b_0$ (*b0_in*) and $b_1$ (*b1_in*) are input to this module as shown in FIG. 4.7. This module implements (3.7) to determine the continuous time parameters $\omega_{eq}$ and $Q_{eq}$ in (3.3).

**10) Amplitude Demodulation Module:** It should be noted that in (3.3), it is

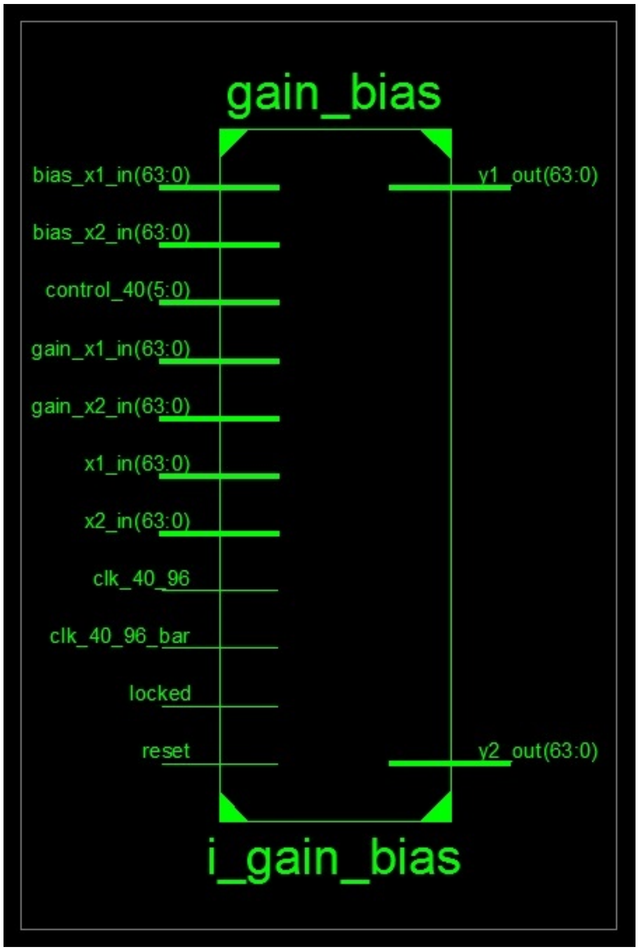

Figure 4.11: RTL schematic of gain and bias module

necessary to know the amplitude $a$ of the oscillating micro-cantilever in real time to determine $\Phi_c$ and $\Phi_d$. An estimate of the amplitude of micro-cantilever oscillations $\hat{a}_n$ can be evaluated in this module. The coefficients of the low pass filter (*a2_lpf_amp_demod, a1_lpf_amp_demod, a0_lpf_amp_demod, b1_lpf_amp_demod, b2_lpf_amp_demod*) entering this module as shown in FIG. 4.8 are obtained from Registration Module 1. The demodulation scheme has been shown in FIG. 4.9.

**11) Registration Module 5:** The parameters received from the $\omega$ and Q Module are registered in this module first at the rising edge of *Clk_40_96_bar* clock followed by registration at the rising edge of *Clk_40_96* clock.

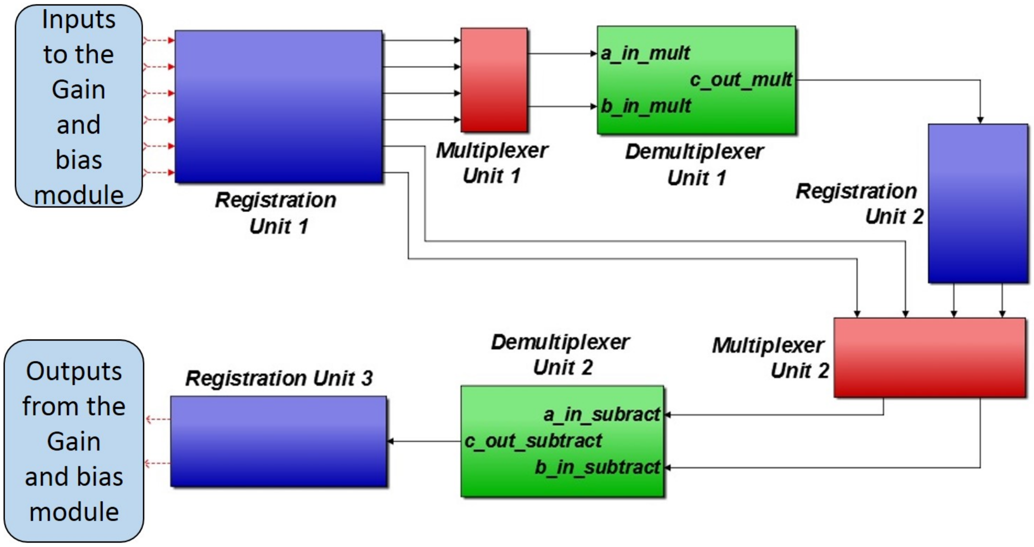

Figure 4.12: Units inside the Gain and Bias module

**12) Registration Module 6:** The amplitude estimate $a$ ($y\_out$) obtained from the Amplitude Demodulation Module is registered at the rising edge of *Clk_40_96_bar* clock followed by registration at the rising edge of *Clk_40_96* clock.

**13) $\phi_C$ and $\phi_D$ Module:** This module determines the storage and dissipative parameters $\Phi_C$ and $\Phi_D$ respectively from $\omega_{eq}$ ($\omega e$) and $Q_{eq}$ ($Qe$) using (3.3) and (3.4). Further, the parameters $\omega_0 = 2\pi f_0$ and $Q_0$ ($\omega o,\ Qo$) which represent the micro-cantilever dynamics free from sample, the drive frequency $\omega_d = 2\pi f_d$ ($\omega$) and the mass of the micro-cantilever $m$ ($m$) are also required in (3.3) and hence these parameters enter this module as shown in FIG. 4.10. $\omega_0$ and $Q_0$ are determined using a frequency sweep based identification of micro-cantilever's frequency response near its first resonant frequency. $m$ is determined before imaging using the thermal response of the cantilever in air. $\omega_d$ is set by the user (typically $\omega_d \approx \omega_0$). Thus the parameters $m$, $\omega_0$, $Q_0$ and $\omega_d$ which are available before imaging are updated on the FPGA using a personal computer through a user interface and hence are routed to this module through Registration Module 1.

**14) Output Selection Module**: The 2-bit vector *op_select_in* obtained from Registration Module 1 is used to select between 4 pair of inputs to this module; $b_0$ & $b_1$ $\{y3\_out,\ y4\_out\}$ from Biquads Module, internal signals $\{test\_out\_1,\ test\_out\_2\}$ from

REEP Module, $\omega_{eq}$ & $Q_{eq}$ $\{\omega\_out, Q\_out\}$ from $\omega$ and Q Module and $\phi_C \& \phi_D$ $\{phi\_c, phi\_d\}$ from $\phi_C$ and $\phi_D$ Module.

**15) Registration Module 7:** The chosen signals from the Output Selection Module are registered at the rising edge of *Clk_40_96_bar* clock followed by registration at the rising edge of *Clk_40_96* clock.

**16) Gain and Bias Module**: This module takes as input the gains values (*gain_x1_in*, *gain_x2_in*) and bias values (*bias_x1_in*, *bias_x2_in*) which are utilized to scale and shift the DAC inputs. These gains and biases are adjustable by the user. The pair of data selected using the signal *op_select_in* is available to the output pair as (*y1_out*, *y2_out*) as shown in FIG. 4.11.

**17) Floating Point to Fixed Point conversion Module:** After exiting the Gain and Bias Module, the 64-bit floating point signals *y1_out* and *y2_out* enter this module where they get converted to 16-bit fixed point signals which are compatible with the DAC ports. This module operates at the rising edge of *Clk_40_96_bar* clock.

**18) Registration Module 8:** The outcoming signals from the Floating Point to Fixed Point conversion Module get registered at the rising edge of *Clk_40_96* clock followed by registration at the rising edge of *Clk_40_96_bar* clock in this module to exit the Main Module as output data pair *dac_output1_fxp* and *dac_output2_fxp* as shown in FIG. 4.2.

## 4.3 Units inside a Module

The scheme employed to create the design of each module is fundamentally the same. This allows for easy customization of one module into another and provides a common ground for effective debugging strategies. The units inside the Gain and Bias Module are shown in FIG. 4.12 and discussed in this section due to its design simplicity.

**1) Registration Unit 1:** The six inputs to this module are the Input 1 (*x1_in*), Input 2 (*x2_in*), gain value for Input 1 (*gain_x1_in*), gain value for Input 2 (*gain_x2_in*), bias value for Input 1 (*bias_x1_in*) and bias value for Input 2 (*bias_x2_in*). The clocking signals namely *Clk_40_96*, *Clk_40_96_bar*, *reset* and *control_40* also enter the Gain and Bias Module. The inputs are first registered at the rising edge of *Clk_40_96_bar* clock

| Count | Mult Core I/P 1 | Mult Core I/P 2 | Mult Core O/P | Sub Core I/P 1 | Sub Core I/P 2 | Sub Core O/P |
|---|---|---|---|---|---|---|
| 0 | | | | | | |
| 1 | $x1\_in\_n$ | $gain\_x1\_in\_n$ | | | | |
| 2 | $x2\_in\_n$ | $gain\_x2\_in\_n$ | | | | |
| 3 | | | | | | |
| 4 | | | $mlt\_4$ | | | |
| 5 | | | $mlt\_5$ | $mlt\_4$ | $bias\_x1\_in\_n$ | |
| 6 | | | | $mlt\_5$ | $bias\_x2\_in\_n$ | |
| 7 | | | | | | $sub\_7$ |
| 8 | | | | | | $sub\_8$ |

Table 4.1: Table showing clock scheme for Gain and bias module

when *control_40* is 0.

**2) Multiplexer Unit 1:** Outputs corresponding to the first four inputs from the Registration Unit 1 are sent to this unit according to the clock signal *control_40.* Inside this unit, an arithmetic computational core is used that performs the multiplication operation. This core requires a specific number of clock periods called latency (3 in this case) to finish a computation while operating synchronously with the clock *Clk_40_96.*

**3) Demultiplexer Unit 1:** The output of Multiplexer Unit 1 is computed at the rising edge of the *Clk_40_96_bar* clock in accordance with the clock signal *control_40.*

**4) Registration Unit 2:** Different outputs obtained from the Demultiplexer Unit 1 at different *control_40* values are registered at the rising edge of the *Clk_40_96* clock.

**5) Multiplexer Unit 2:** The registered outputs from the Registration Unit 2 are sent to this unit according to the *control_40* clock signal and subtraction operation is performed (with a latency of 2) on the inputs by an arithmetic computational core which is located inside this unit.

**6) Demultiplexer Unit 2:** Outgoing signals of the Multiplexer Unit 2 are received here. This unit operates with the help of the clock signal *control_40* and at the rising edge of the *Clk_40_96_bar* clock.

| Module name | Mult cores (3) | Add cores (2) | Sub cores (2) | Rec cores (5) | Sqr root cores (7) | $\tan^{-1}$ cores (35) | $\tanh^{-1}$ cores (35) | Flt pt to fxd pt cores (6) | Fxd pt to flt pt cores (6) |
|---|---|---|---|---|---|---|---|---|---|
| Counter | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Biquads | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| REEP | 5 | 2 | 2 | 1 | 0 | 0 | 0 | 0 | 0 |
| $\omega$ and Q Module | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Amplitude Demodulation | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $\phi_C$ and $\phi_D$ | 2 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| Output Selection | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Gain and Bias | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

Table 4.2: Table showing no. of computational cores and their corresponding latencies (within brackets) used in each module

**7) Registration Unit 3:** The outputs obtained from the Demultiplexer Unit 2 are registered 3 times consecutively by *Clk_40_96* clock, *Clk_40_96_bar* clock and *Clk_40_96* clock when *control_40* is $N = 40$.

Table.4.1 shows the clock scheme for the cores operating inside the Gain and Bias Module. The first column shows the *control_40* values at which the various operations on the two cores are performed. Table.4.2 gives the no. of cores used in all the modules along with the latencies utilized in each of the individual cores in the modules. Some mathematical functions including the trigonometric functions $\tan^{-1}$ and $\tanh^{-1}$ necessary in evaluating (3.7) are implemented through special function cores. Such special function cores can operate only on fixed point data. Besides, the ADC and DAC ports operate on fixed point data as well. Hence, data is converted from fixed point format to floating point format and vice-versa when needed utilizing the data type conversion cores.

## 4.4 Design steps

The design of the submodules of the system identification module are implemented using the hardware description language VHDL. Each block is tested individually through simulation with test input data before being assembled in the final design. The test data is generated using MATLAB and the simulations for VHDL designs are carried out using the simulation tool ISim from Xilinx. The designs are iterated multiple times to meet the space and timing constraints of the design compatible to the Xilinx ML605 FPGA board. Each time a part of the design is updated, it is verified for accuracy by implementating the system identification algorithm in MATLAB. Microblaze embedded processor provides the platform for reading and writing the user-entered configurable parameters in the FPGA. The processor logic is coded using C++ programming language in the Xilinx Software Development Kit (SDK). The VHDL design of system identification module together with the embedded Microblaze processor is implemented on the FPGA through the completion of the following design steps in Xilinx ISE 14.4 design suite:

**Creating Timing Constraints:** This step includes the generation of a User Constraints File (UCF) through which the user can specify timing, placement and input-output constraints on a text editor.

**Synthesizing:** Once the constraints are fixed by the user, netlist files called NGC files are created after the Xilinx Synthesis Technology (XST) synthesizes the VHDL language design. The NGC files contain both the constraints as well as the logical design data.

**Translating:** In this step, a Xilinx native generic database (NGD) file is created after the NGC files from the Synthesizing step are merged.

**Mapping:** This step allocates Input Output Block (IOB) and Configurable Logic Block (CLB) resources for all the primary logic elements in the design based on the NGD file obtained in the Translating step. Checking of the design rule on the final mapped netlist and performing optimizations on the target device are also a part of this step.

**Placing And Routing:** Implementation of timing-driven placments and routing of the mapped netlist from the Mapping step is achieved in this Placing And Routing (PAR) process.

**Generating Programming File:** In this step, the Xilinx bitstream generation program called BitGen is executed, producing a bitstream file for the Xilinx device configuration. This bitstream file when burned on the FPGA using Xilinx SDK, runs the system identification algorithm.

In the next chapter, both simulation and experimental results have been discussed which verify the accuracy and performance of the system identification module.

# Chapter 5

# Simulation and Experimental Results

## 5.1 Behavioral simulation of system identification module

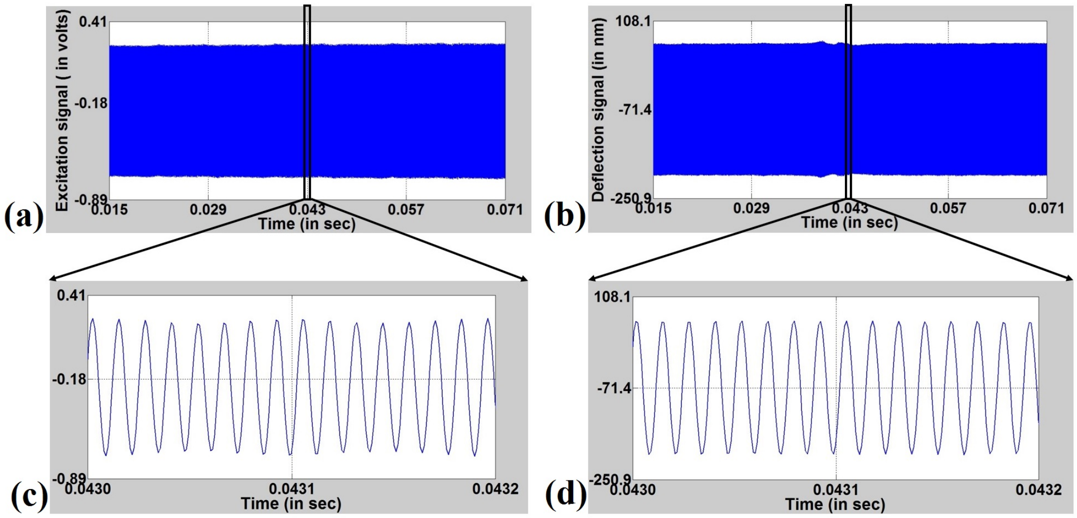

Figure 5.1: (a) Cantilever excitation signal and (b) deflection signal acquired during an experiment performed on a sample of PLMA-PBMA polymer blend, with the dynamic mode AFM. Zoomed version of both the excitation signal (c) and the deflection signal (d) to show their near sinusoidal nature

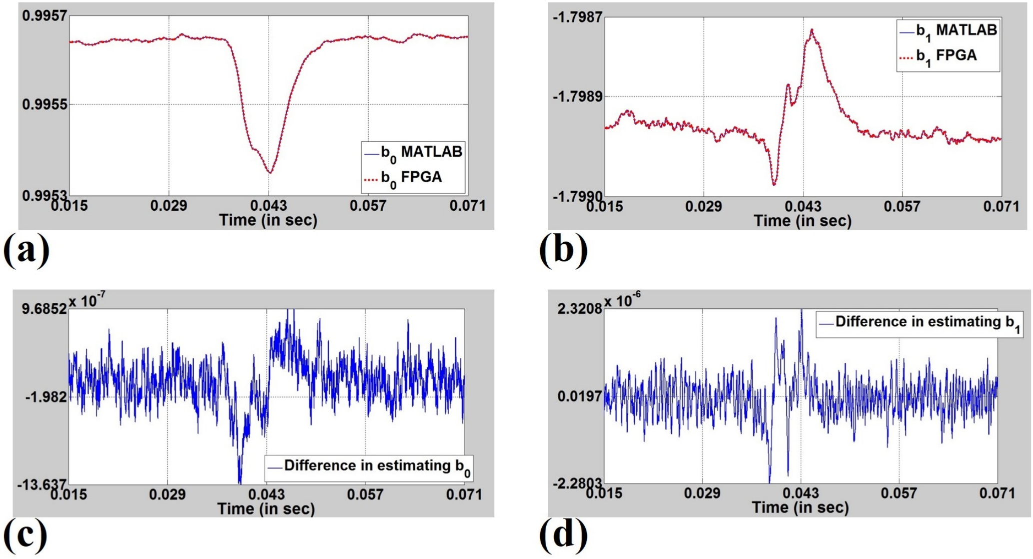

Figure 5.2: (a) $b_0$ and (b) $b_1$ generated by the behavioral simulation of the system identification module in MATLAB and VHDL. Difference between the estimation of (c) $b_0$ and (d) $b_1$ by MATLAB and VHDL

A comparison of the behavioral simulation results is made between the software platforms of MATLAB version R2016a and ISim Simulator, a part of Xilinx ISE Project Navigator version 14.4 to ensure the real-time applicability of the system identification module. The module designed in MATLAB acts as a reference for the testing of the VHDL design.

### 5.1.1 Test Data

First, the free air chirp response of the cantilever is recorded using Hewlett Packard 35631 Controls Systems Analyzer. A 2nd order transfer function is fit to the data obtained from the response and the system's nominal parameters like resonant frequency (72.964 kHz), quality factor (205.6) and spring constant (1.75 nN/nm) are recorded.

Second, an approach-retract experiment with dynamic mode MFP-3D AFM is performed on a PLMA-PBMA polymer blend to obtain the test data. In this experiment,the

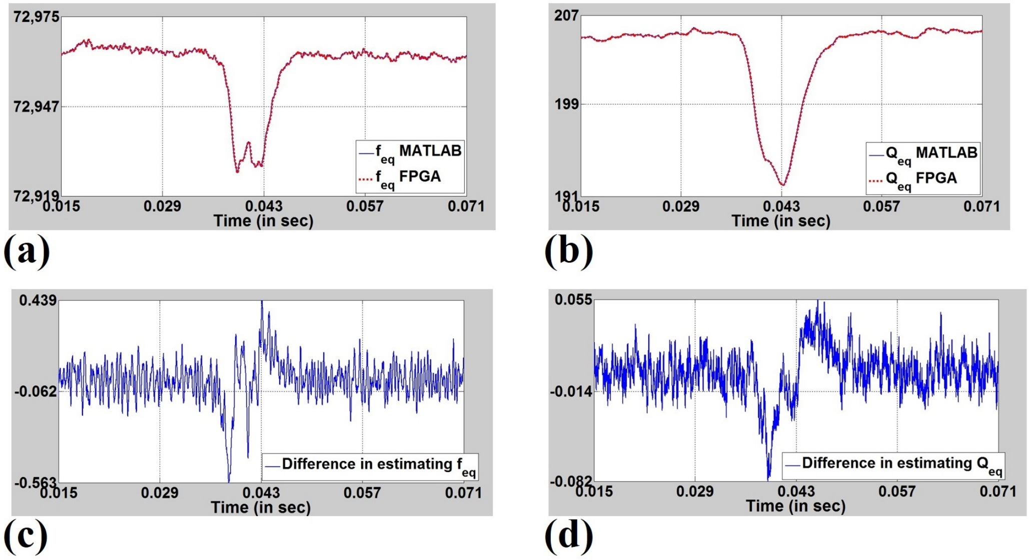

Figure 5.3: A comparison between (a) Equivalent frequency $f_{eq}$ (in Hz) and (b) equivalent quality factor $Q_{eq}$ and their respective plots (c) and (d) showing the difference in their estimation by MATLAB and VHDL

oscillating micro-cantilever is made to approach the surface of the PLMA-PBMA sample till the amplitude of the oscillation decreases due to the micro-cantilever tip-sample interactions and the amplitude of the micro-cantilever attains a specified value (trigger). Once that happens, the micro-cantilever is retracted from the PLMA-PBMA sample till it oscillates free from the sample and the whole process is repeated again. The approach-retract experiment is performed gradually such that at any instant, the cantilever oscillation remains in steady state.

### 5.1.2 Simulation Comparison

The cantilever excitation and the deflection signals which are shown in FIG. 5.1 are acquired at a sampling frequency of 2 MHz using a National Instruments BNC-2110 LabVIEW data acquisition system. In MATLAB, these signals are resampled at 40.96 MHz and converted into 16 bit fixed point signals to serve as input to the system identification module in the ISim Simulator. The behavioral simulation in the ISim Simulator

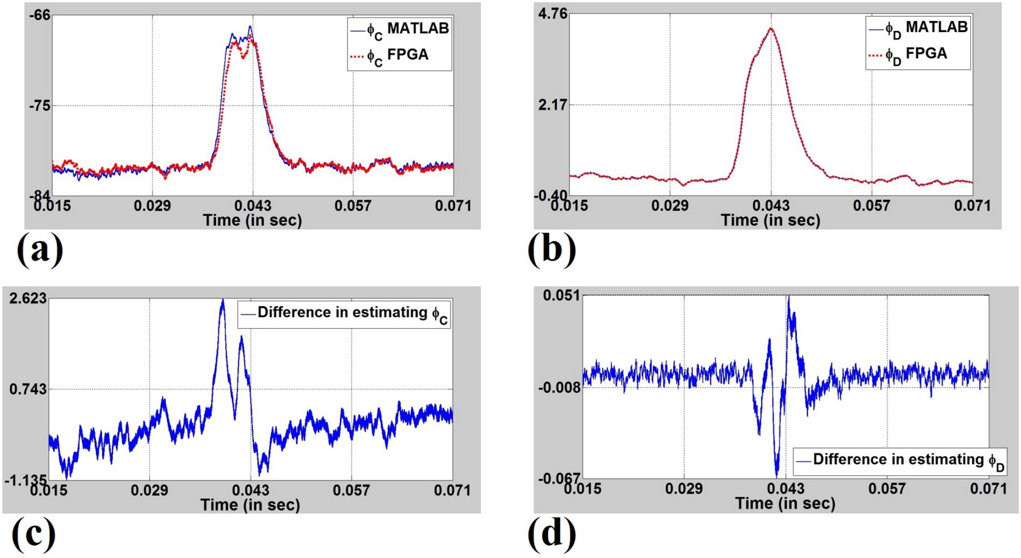

Figure 5.4: (a) Conservative power $\phi_C$ (in Pico-Watt) and (b) dissipative power $\phi_D$ (in Pico-Watt) and their individual plots (c) and (d) depicting the difference in their estimation by MATLAB and VHDL

is run for a simulation time of 75 ms. The Simulator design outputs one of the following data pairs: $\{b_0, b_1\}$ (discrete time equivalent system parameters), $\{f_{eq}, Q_{eq}\}$ (equivalent frequency and quality factor), $\{\phi_C, \phi_D\}$ (storage and dissipative powers) or $\{$*test_out_1*, *test_out_2*$\}$ (internal signals within the REEP Module). These output data pairs are generated using ISim simulation and compared with MATLAB simulation of the system identification module.

Comparison of the signals $\{b_0, b_1\}$, $\{f_{eq}, Q_{eq}\}$ and $\{\phi_C, \phi_D\}$ generated by the MATLAB design and the VHDL design are presented in FIG. 5.2, FIG. 5.3 and FIG. 5.4 respectively. All of these figures demonstrate that the MATLAB implementation and VHDL design produce close match. Thus the efficacy of the VHDL design for FPGA implementation is supported with simulation results.

## 5.2 Experimental implementation of system identification module

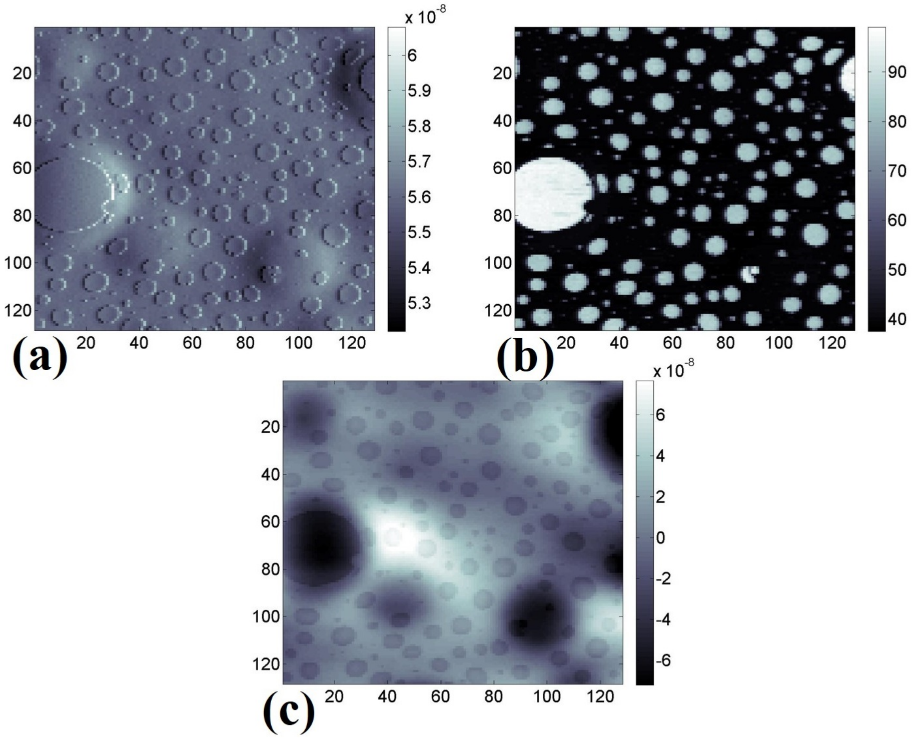

Figure 5.5: 10 $\mu$m × 10 $\mu$m images of (a) amplitude (meter), (b) phase (degree) and (c) height (meter) of a PLMA-PBMA polymer blend, obtained using dynamic mode AFM. The color bars specify the respective ranges. In this case some of 'height' is actually mechanical compliance, such that protrusions are imaged as depressions.

### 5.2.1 Experiment Details

MFP 3D AFM and AC240TS micro-cantilever both from Asylum Research are used for this experiment. The micro-cantilever has a spring constant of 1.75 nN/nm, a quality factor of 146.4 and a resonant frequency of 72.728 kHz. The sample used in this experiment consists of a blend of poly (butyl methacrylate) PBMA and poly(lauryl methacrylate) PLMA polymers which are coated over a 1 $\mu$m thick silica substrate. FIG.

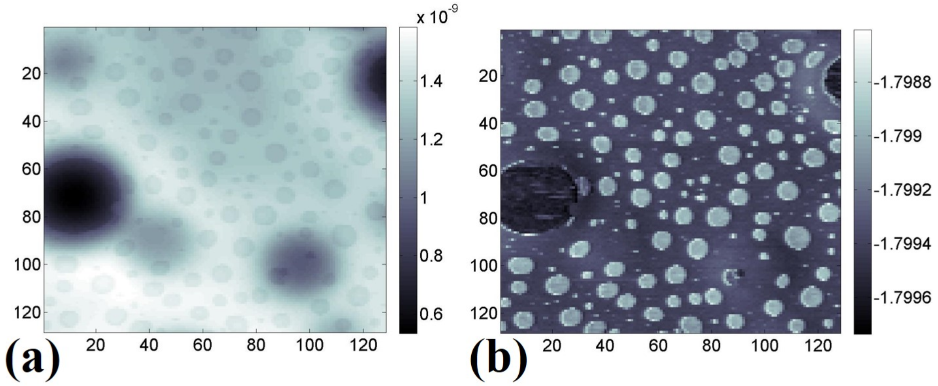

Figure 5.6: 10 $\mu$m × 10 $\mu$m images of estimated (a) $b_0$ and (b) $b_1$ for the PLMA-PBMA polymer blend, obtained using the system identification module implemented on FPGA. The color bars specifying the respective ranges for $b_0$ and $b_1$ are shown with respect to a positive bias of value 0.997121456 and 0 respectively.

5.5 shows the images obtained in the dynamic mode imaging namely the amplitude, phase and height images of a 10 $\mu$m × 10 $\mu$m area of the PBMA-PLMA sample. A scan rate of 0.25 Hz was used to obtain these images. The brighter domains displayed in the phase image in FIG. 5.5 (b) which correspond to the virtual depressions due to mechanical compliance indicated by darker regions in the height image displayed in FIG. 5.5 (c) represent the PLMA regions which are scattered over the PBMA background.

### 5.2.2 System identification module on FPGA

A forgetting factor $\lambda = 0.9995$ has been used on the system identification module which is implemented on the FPGA. The input to the FPGA are the excitation signal to the micro-cantilever $g$ and the measured deflection signal $y$. These signals are filtered with the HPFs with cut-off frequency of 2 kHz, whereas the estimates of the parameters $b_1$ and $b_0$ in (3.6) are filtered with LPFs with cut-off frequency of 2 kHz inside the system identification module. One of the data pairs $\{b_0, b_1\}$, $\{f_{eq}, Q_{eq}\}$, $\{\phi_C, \phi_D\}$ or $\{$*test_out_1*, *test_out_2*$\}$ is chosen using the output selection signal to become the outputs from the FPGA. FPGA generated images of the estimates of discrete time equivalent parameters $b_0$ and $b_1$ are shown in FIG. 5.6 (a) and (b) respectively. From FIG. 5.6, it can

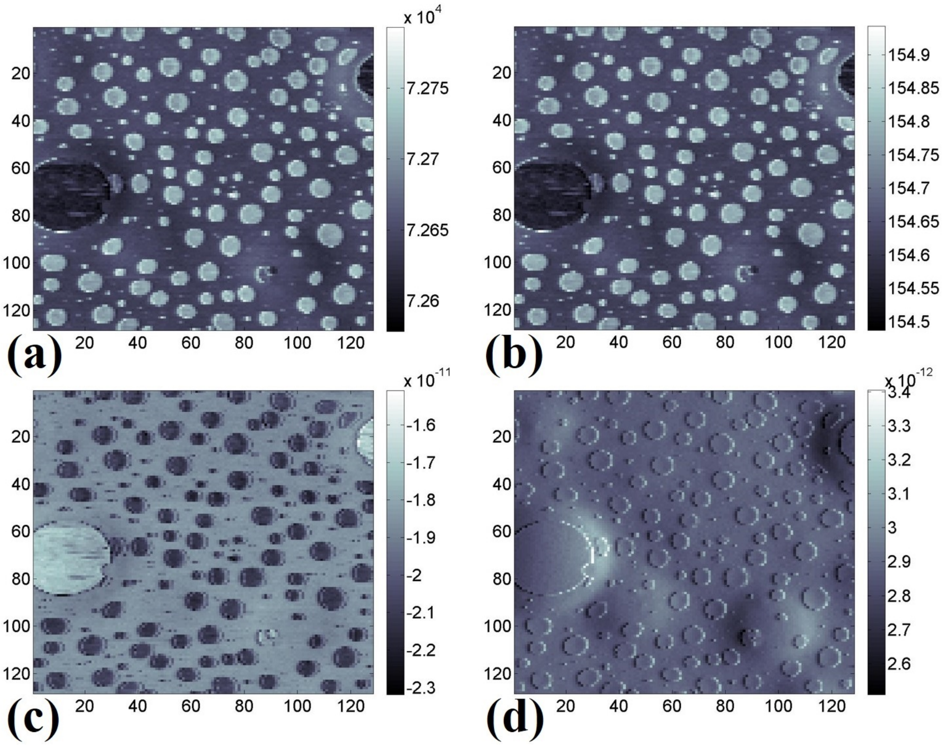

Figure 5.7: 10 $\mu$m × 10 $\mu$m images of estimated (a) equivalent frequency $f_{eq}$ (in Hz), (b) quality factor $Q_{eq}$, (c) conservative power $\phi_C$ (in watt) and (d) dissipative power $\phi_D$ (in watt) for the PLMA-PBMA polymer blend, calculated using the FPGA module. The color bars specify the respective ranges. The opposite contrast for large and small PLMA domains is due to switching of regimes within the image, net repulsive vs net attractive.

be seen that PLMA domains represent lower $b_0$ values than PBMA. A similar trend is observed in the case of $b_1$. FIG.5.7 (a) and FIG. 5.7 (b) show images for the estimates of equivalent frequency $f_{eq}$ and quality factor $Q_{eq}$ respectively. Observing FIG.5.7 (c) and FIG. 5.7 (d), it can be referenced that the conservative power $\phi_C = a\omega_d\phi_c$ and dissipative power component $\phi_D = a\omega_d\phi_d$ due to interactive forces between the cantilever tip and sample are consistently higher on the PLMA domains compared to the PBMA background.

The FPGA based system identification module performed successfully over multiple experimental runs conducted at different times, proves the effectiveness of the module.

# Chapter 6

# Conclusion and Discussion

In this thesis, an FPGA implementation of a system identification module with application to atomic force microscopy has been reported. The BCEWRLS algorithm employed in this thesis is used for estimating discrete time system parameters, followed by estimation of the resonant frequency and quality factor of an equivalent micro-cantilever. The equivalent micro-cantilever is modeled as an AFM micro-cantilever operating in the dynamic mode of operation while interacting with a sample. Within the system identification module, the estimated equivalent resonant frequency and quality factor are employed for the estimation of the conservative and dissipative power components which occur due to micro-cantilever tip-sample interaction. By interfacing the FPGA board with a personal computer, one can update user configurable parameters on the FPGA in real time. The FPGA implementation of the system identification module has been effectively utilized for the quantitative characterization of local surface mechanical properties of a PBMA-PLMA polymer blend sample. The design approach laid out in this thesis serves as a reference for the design of a large variety of real time signal processing units. Hence, applicability of the system identification module is not limited to the PBMA-PLMA sample, but to a wide variety of samples. Some good examples include samples of biological entities like cells, DNA, protein, etc. and different soft polymers.

# References

[1] H. G. Natke, *Application of system identification in engineering.* Springer, 2014, vol. 296.

[2] O. Nelles, *Nonlinear system identification: from classical approaches to neural networks and fuzzy models.* Springer Science & Business Media, 2013.

[3] R. Pintelon and J. Schoukens, *System identification: a frequency domain approach.* John Wiley & Sons, 2012.

[4] T. B. Schön, A. Wills, and B. Ninness, "System identification of nonlinear state-space models," *Automatica*, vol. 47, no. 1, pp. 39–49, 2011.

[5] L. Ljung, "Perspectives on system identification," *Annual Reviews in Control*, vol. 34, no. 1, pp. 1–12, 2010.

[6] L. Ljung and T. Söderström, "Theory and practice of recursive identification," 1983.

[7] K. J. Åström and P. Eykhoff, "System identificationa survey," *Automatica*, vol. 7, no. 2, pp. 123–162, 1971.

[8] T. Hiigglund, "The problem of forgetting old data in recursive estimation," in *Adaptive Systems in Control and Signal Processing 1983: Proceedings of the IFAC Workshop, San Francisco, USA, 20-22 June 1983.* Elsevier, 2014, p. 213.

[9] P. Eykhoff, *Trends and progress in system identification: IFAC Series for Graduates, Research Workers & Practising Engineers.* Elsevier, 2014, vol. 1.

[10] H.-F. Chen, “New approach to recursive identification for armax systems,” *IEEE Transactions on Automatic Control*, vol. 55, no. 4, pp. 868–879, 2010.

[11] A. Vahidi, A. Stefanopoulou, and H. Peng, “Recursive least squares with forgetting for online estimation of vehicle mass and road grade: theory and experiments,” *Vehicle System Dynamics*, vol. 43, no. 1, pp. 31–55, 2005.

[12] P. Agarwal and M. Salapaka, “Real time estimation of equivalent cantilever parameters in tapping mode atomic force microscopy,” *Applied Physics Letters*, vol. 95, no. 8, pp. 083 113–083 113, 2009.

[13] G. Binnig, C. Quate, and C. Gerber, “Atomic force microscope,” *Physical review letters*, vol. 56, no. 9, pp. 930–933, 1986.

[14] H. K. Nguyen, M. Ito, S. Fujinami, and K. Nakajima, “Viscoelasticity of inhomogeneous polymers characterized by loss tangent measurements using atomic force microscopy,” *Macromolecules*, vol. 47, no. 22, pp. 7971–7977, 2014.

[15] D. G. Yablon, J. Grabowski, and I. Chakraborty, “Measuring the loss tangent of polymer materials with atomic force microscopy based methods,” *Measurement Science and Technology*, vol. 25, no. 5, p. 055402, 2014.

[16] R. Garcia and R. Proksch, “Nanomechanical mapping of soft matter by bimodal force microscopy,” *European Polymer Journal*, vol. 49, no. 8, pp. 1897–1906, 2013.

[17] X. Sun, Z. Liu, K. Welsher, J. T. Robinson, A. Goodwin, S. Zaric, and H. Dai, “Nano-graphene oxide for cellular imaging and drug delivery,” *Nano research*, vol. 1, no. 3, pp. 203–212, 2008.

[18] G. Thomas, N. A. Burnham, T. A. Camesano, and Q. Wen, “Measuring the mechanical properties of living cells using atomic force microscopy,” *JoVE (Journal of Visualized Experiments)*, no. 76, pp. e50 497–e50 497, 2013.

[19] J. E. Marturano, J. D. Arena, Z. A. Schiller, I. Georgakoudi, and C. K. Kuo, “Characterization of mechanical and biochemical properties of developing embryonic tendon,” *Proceedings of the National Academy of Sciences*, vol. 110, no. 16, pp. 6370–6375, 2013.

[20] S. Kasas, G. Longo, and G. Dietler, "Mechanical properties of biological specimens explored by atomic force microscopy," *Journal of Physics D: Applied Physics*, vol. 46, no. 13, p. 133001, 2013.

[21] Y. F. Dufrêne, D. Martínez-Martín, I. Medalsy, D. Alsteens, and D. J. Müller, "Multiparametric imaging of biological systems by force-distance curve-based afm," *Nature methods*, vol. 10, no. 9, pp. 847–854, 2013.

[22] M. P. Nikiforov, S. Hohlbauch, W. P. King, S. A. Contera, S. Jesse, S. V. Kalinin, R. Proksch *et al.*, "Temperature-dependent phase transitions in zeptoliter volumes of a complex biological membrane," *Nanotechnology*, vol. 22, no. 5, p. 055709, 2010.

[23] Q. H. Wang, K. Kalantar-Zadeh, A. Kis, J. N. Coleman, and M. S. Strano, "Electronics and optoelectronics of two-dimensional transition metal dichalcogenides," *Nature nanotechnology*, vol. 7, no. 11, pp. 699–712, 2012.

[24] G. Stan, S. W. King, and R. F. Cook, "Nanoscale mapping of contact stiffness and damping by contact resonance atomic force microscopy," *Nanotechnology*, vol. 23, no. 21, p. 215703, 2012.

[25] S. Jesse and S. V. Kalinin, "Band excitation in scanning probe microscopy: sines of change," *Journal of Physics D: Applied Physics*, vol. 44, no. 46, p. 464006, 2011.

[26] A. Gannepalli, D. Yablon, A. Tsou, and R. Proksch, "Mapping nanoscale elasticity and dissipation using dual frequency contact resonance afm," *Nanotechnology*, vol. 22, no. 35, p. 355705, 2011.

[27] H.-J. Kim, S.-S. Yoo, and D.-E. Kim, "Nano-scale wear: a review," *International Journal of Precision Engineering and Manufacturing*, vol. 13, no. 9, pp. 1709–1718, 2012.

[28] B. Gotsmann and M. A. Lantz, "Atomistic wear in a single asperity sliding contact," *Physical review letters*, vol. 101, no. 12, p. 125501, 2008.

[29] G. Haugstad, *Atomic force microscopy: understanding basic modes and advanced applications.* John Wiley & Sons, 2012.

[30] R. García, *Amplitude modulation atomic force microscopy.* John Wiley & Sons, 2011.

[31] R. Garcıa and R. Perez, "Dynamic atomic force microscopy methods," *Surface science reports*, vol. 47, no. 6, pp. 197–301, 2002.

[32] R. Garcia and E. T. Herruzo, "The emergence of multifrequency force microscopy," *Nature nanotechnology*, vol. 7, no. 4, pp. 217–226, 2012.

[33] C. J. Gómez and R. Garcia, "Determination and simulation of nanoscale energy dissipation processes in amplitude modulation afm," *Ultramicroscopy*, vol. 110, no. 6, pp. 626–633, 2010.

[34] R. Garcia, C. Gomez, N. Martinez, S. Patil, C. Dietz, and R. Magerle, "Identification of nanoscale dissipation processes by dynamic atomic force microscopy," *Physical review letters*, vol. 97, no. 1, p. 016103, 2006.

[35] G. Saraswat, P. Agarwal, G. Haugstad, and M. Salapaka, "Real-time probe based quantitative determination of material properties at the nanoscale," *Nanotechnology*, vol. 24, no. 26, p. 265706, 2013.

[36] A. Sebastian, A. Gannepalli, and M. Salapaka, "A review of the systems approach to the analysis of dynamic-mode atomic force microscopy," *Control Systems Technology, IEEE Transactions on*, vol. 15, no. 5, pp. 952–959, 2007.

[37] S. Salapaka, T. De, and A. Sebastian, "Sample-profile estimate for fast atomic force microscopy," *Applied Physics Letters*, vol. 87, no. 5, p. 053112, 2005.

[38] A. Sebastian and S. M. Salapaka, "Design methodologies for robust nano-positioning," *Control Systems Technology, IEEE Transactions on*, vol. 13, no. 6, pp. 868–876, 2005.

[39] A. Malinowski and H. Yu, "Comparison of embedded system design for industrial applications," *Industrial Informatics, IEEE Transactions on*, vol. 7, no. 2, pp. 244–254, 2011.

[40] E. Monmasson, L. Idkhajine, and M. W. Naouar, "Fpga-based controllers," *Industrial Electronics Magazine, IEEE*, vol. 5, no. 1, pp. 14–26, 2011.

[41] F. Baronti, E. Petri, S. Saponara, L. Fanucci, R. Roncella, R. Saletti, P. D. Abramo, and R. Serventi, "Design and verification of hardware building blocks for high-speed and fault-tolerant in-vehicle networks," *Industrial Electronics, IEEE Transactions on*, vol. 58, no. 3, pp. 792–801, 2011.

[42] E. Monmasson, L. Idkhajine, M. N. Cirstea, I. Bahri, A. Tisan, and M. W. Naouar, "Fpgas in industrial control applications," *Industrial Informatics, IEEE Transactions on*, vol. 7, no. 2, pp. 224–243, 2011.

[43] Ó. López, J. Álvarez, J. Doval-Gandoy, and F. D. Freijedo, "Multilevel multiphase space vector pwm algorithm," *Industrial Electronics, IEEE Transactions on*, vol. 55, no. 5, pp. 1933–1942, 2008.

[44] H. Tanaka, K. Ohnishi, H. Nishi, T. Kawai, Y. Morikawa, S. Ozawa, and T. Furukawa, "Implementation of bilateral control system based on acceleration control using fpga for multi-dof haptic endoscopic surgery robot," *Industrial Electronics, IEEE Transactions on*, vol. 56, no. 3, pp. 618–627, 2009.

[45] T. Li and Y. Fujimoto, "Control system with high-speed and real-time communication links," *Industrial Electronics, IEEE Transactions on*, vol. 55, no. 4, pp. 1548–1557, 2008.

[46] S. Ghosal, S. Pradhan, and M. Salapaka, "Design of fpga based system identification module with application to atomic force microscopy," 2017 (Under Review).

[47] F. Ding, T. Chen, and L. Qiu, "Bias compensation based recursive least-squares identification algorithm for miso systems," *Circuits and Systems II: Express Briefs, IEEE Transactions on*, vol. 53, no. 5, pp. 349–353, 2006.

[48] P. Agarwal, "Model mismatch paradigm for probe based nanoscale imaging." Ph.D. dissertation, 2010.

[49] H. K. Khalil, *Nonlinear systems.* Prentice hall Upper Saddle River, 2002, vol. 3.

[50] N. Bogoliubov, *Asymptotic methods in the theory of non-linear oscillations.* Routledge, 1961, vol. 10.

[51] A. Sebastian, A. Gannepalli, and M. Salapaka, “The amplitude phase dynamics and fixed points in tapping-mode atomic force microscopy,” in *American Control Conference, 2004. Proceedings of the 2004*, vol. 3. IEEE, 2004, pp. 2499–2504.

[52] V. A. Pedroni, *Circuit design with VHDL.* MIT press, 2004.

# Appendix A

# Programming Codes

## A.1 C Code used in the Software Development Kit (SDK) for accessing the user-entered parameters through the MicroBlaze Embedded Processor

```
#include < stdio.h >
#include ”xparameters.h”
#include ”platform.h”
#include ”REEP_config.h”
#define XPAR_MYBLOCK_0_BASEADDR XPAR_REEP_0_S_AX_MEM0_BASEADDR

void print(char *str);
int main()
{
unsigned int temp_data, initial_value;
int i, j, sw_input, mem_size;
int wait_on_sw = 1, push_sw_on, LED_loopback_test, loop_count;
int sdram_data_read_error;
unsigned long long sdram_data_read, temp_sdram_data;
unsigned long long* temp;
char buf[8];
char c, d;
char inchar;
int buf1, hum;
unsigned int *my_variable = XPAR_MYBLOCK_0_BASEADDR;
init_platform();
print(”Hello World\n\r”);
cleanup_platform();
print(”entering main\n\r”);
my_variable[0] = 0x10101010;
xil_printf(”%8X \n\r”,my_variable[0]);
print(”Configuration Started...\n\r”);
print(”Configuring the REEP coefficients.... \n\r”);
my_variable[0] = a2_high;
xil_printf(”a2_high configured is %8X \n\r”,my_variable[0]);
my_variable[1] = a2_low;
xil_printf(”a2_low configured is %8X \n\r”,my_variable[1]);
// Similarly other variables are entered
print(” Configuration Completed .. \n\r”);
print(”\n\r”);
mainmenu();
while (1) {
print(”Enter a number (1)>”);
inchar = inbyte();
print(”\n\r”);
switch (inchar) {
case ’1’ : submenu1(); break;
}
mainmenu();
}
return 0;
}
void mainmenu() {
```

```
print(”Main Menu\n\r”);
print(”————————————————————\n\r”);
print(” 1. Configure the REEP signals \n\r”);
}
int read_hex(char buf[8]) {
int i;
int hum = 0x0;
for (i= 0; i¡8; i++) {
switch (buf[i]) {
case ’0’: hum = hum | 0x0 << (7-i)*4;break;
case ’1’: hum = hum | 0x1 << (7-i)*4;break;
// Similarly other variables are entered
}
}
return hum;
}
void submenu1() {
unsigned int *my_variable = XPAR_MYBLOCK_0_BASEADDR;
char c;
int hum;
char buf[8];
print(”Configuring the REEP coefficients ... \n\r”);
c = read(0,buf,8);
hum = read_hex(buf);
xil_printf(”%8X \n\r”,hum);
my_variable[0] = hum;
xil_printf(”a2_high configured is %8X \n\r”,my_variable[0]);
c = read(0,buf,8);
hum = read_hex(buf);
xil_printf(”%8X \n\r”,hum);
my_variable[1] = hum;
xil_printf(”a2_low configured is %8X \n\r”,my_variable[1]);
```

```
// Similarly other variables are entered
}
```

## A.2 VHDL Code used for designing the Gain and Bias Module

```
library IEEE; use IEEE.STD_LOGIC_1164.ALL;
use ieee.std_logic_unsigned.all;
use ieee.std_logic_misc.all;
use ieee.std_logic_arith.all; library UNISIM;
use UNISIM.VCOMPONENTS.ALL;

entity gain_bias is
port
(
reset : in std_logic;
locked_40_96: in std_logic;
clk_40_96: in std_logic; – 40.96 Mhz clock, clock at which mult and add registering is done
clk_40_96_bar: in std_logic; – 40.96 Mhz clock, clock at which mult and add is done
control_40: in std_logic_vector(5 downto 0);
x1_in: in std_logic_vector(63 downto 0);
x2_in: in std_logic_vector(63 downto 0);
gain_x1_in: in std_logic_vector(63 downto 0);
gain_x2_in: in std_logic_vector(63 downto 0);
bias_x1_in: in std_logic_vector(63 downto 0);
bias_x2_in: in std_logic_vector(63 downto 0);
y1_out: out std_logic_vector(63 downto 0);
y2_out: out std_logic_vector(63 downto 0)
);
end gain_bias;
```

```
architecture Behavioral of gain_bias is

COMPONENT mulp_gain_bias
PORT (
aclk : IN STD_LOGIC;
s_axis_a_tvalid : IN STD_LOGIC;
s_axis_a_tdata : IN STD_LOGIC_VECTOR(63 DOWNTO 0);
s_axis_b_tvalid : IN STD_LOGIC;
s_axis_b_tdata : IN STD_LOGIC_VECTOR(63 DOWNTO 0);
m_axis_result_tvalid : OUT STD_LOGIC;
m_axis_result_tdata : OUT STD_LOGIC_VECTOR(63 DOWNTO 0)
);
END COMPONENT;

COMPONENT subtract_gain_bias
PORT (
aclk : IN STD_LOGIC;
s_axis_a_tvalid : IN STD_LOGIC;
s_axis_a_tdata : IN STD_LOGIC_VECTOR(63 DOWNTO 0);
s_axis_b_tvalid : IN STD_LOGIC;
s_axis_b_tdata : IN STD_LOGIC_VECTOR(63 DOWNTO 0);
m_axis_result_tvalid : OUT STD_LOGIC;
m_axis_result_tdata : OUT STD_LOGIC_VECTOR(63 DOWNTO 0)
);
END COMPONENT;

signal x1_in_n, x2_in_n: std_logic_vector(63 downto 0):=(others=>’0’);
signal x1_in_reg, x2_in_reg: std_logic_vector(63 downto 0):=(others=>’0’);
signal gain_x1_in_n, gain_x2_in_n, bias_x1_in_n, bias_x2_in_n : std_logic_vector(63 downto
0):=(others=>’0’);
signal gain_x1_in_reg, gain_x2_in_reg, bias_x1_in_reg, bias_x2_in_reg : std_logic_vector(63
downto 0):=(others=>’0’);
signal y1_out_n, y2_out_n : std_logic_vector(63 downto 0):=(others=>’0’);
signal y1_out_reg, y2_out_reg: std_logic_vector(63 downto 0):=(others=>’0’);
signal a_in_mult, b_in_mult, c_out_mult: std_logic_vector(63 downto 0):=(others=>’0’);
signal m_axis_result_tvalid_mult: std_logic:=’0’;
signal a_in_subtract, b_in_subtract, c_out_subtract: std_logic_vector(63 downto 0):=(others=>’0’);
signal m_axis_result_tvalid_subtract: std_logic:=’0’;
signal mlt_4, mlt_5: std_logic_vector(63 downto 0):=(others=>’0’);
signal sub_7, sub_8: std_logic_vector(63 downto 0):=(others=>’0’);
signal clk_40_96_tmp, clk_40_96_bar_tmp: std_logic:=’0’;
signal control_40_tmp: integer range 0 to 39;

begin control_40_tmp<=conv_integer(control_40);

reg_input2: process(clk_40_96_bar)
begin
if (rising_edge(clk_40_96_bar)) then
if (reset=’1’) or (locked_40_96=’0’)then
x1_in_n<=(others=>’0’);
x2_in_n<=(others=>’0’);
gain_x1_in_n<=(others=>’0’);
gain_x2_in_n<=(others=>’0’);
bias_x1_in_n<=(others=>’0’);
bias_x2_in_n<=(others=>’0’);
else
if (control_40_tmp=0) then
x1_in_n<=x1_in;
x2_in_n<=x2_in;
gain_x1_in_n<=gain_x1_in;
gain_x2_in_n<=gain_x2_in;
bias_x1_in_n<=bias_x1_in;
bias_x2_in_n<=bias_x2_in;
```

```
end if;
end if;
end if;
end process reg_input2;

y1_out_n<=sub_7;
y2_out_n<=sub_8;

reg_y_n_1: process(clk_40_96_bar)
begin
if (rising_edge(clk_40_96_bar)) then
if (reset='1') or (locked_40_96='0') then
y1_out_reg<=(others=>'0');
y2_out_reg<=(others=>'0');
else
if (control_40_tmp=39) then
y1_out_reg<=y1_out_n;
y2_out_reg<=y2_out_n;
end if;
end if;
end if;
end process reg_y_n_1;

reg_y_n_2: process(clk_40_96)
begin
if (rising_edge(clk_40_96)) then
if (reset='1') or (locked_40_96='0') then
y1_out<=(others=>'0');
y2_out<=(others=>'0');
else
y1_out<=y1_out_reg;
y2_out<=y2_out_reg;
```

```
end if;
end if;
end process reg_y_n_2;

mult_in_reg: process(clk_40_96)
begin
if (rising_edge(clk_40_96)) then
if (reset=’1’) or (locked_40_96=’0’) then
a_in_mult<=(others=>’0’);
b_in_mult<=(others=>’0’);
else
case control_40_tmp is
when 0 =>
a_in_mult<=(others=>’0’);
b_in_mult<=(others=>’0’);
when 1 =>
a_in_mult<=x1_in_n;
b_in_mult<=gain_x1_in_n;
when 2 =>
a_in_mult<=x2_in_n;
b_in_mult<=gain_x2_in_n;
when others =>
a_in_mult<=(others=>’0’);
b_in_mult<=(others=>’0’);
end case;
end if;
end if;
end process mult_in_reg;

reg_out_mult: process(clk_40_96)
begin
if (rising_edge(clk_40_96)) then
if (reset=’1’) or (locked_40_96=’0’) then
mlt_4<=(others=>’0’);
mlt_5<=(others=>’0’);
else
case control_40_tmp is
when 4 =>
mlt_4<=c_out_mult;
when 5 =>
mlt_5<=c_out_mult;
when others => null;
end case;
end if;
end if;
end process reg_out_mult;

subtract_in_reg: process(clk_40_96)
begin
if (rising_edge(clk_40_96)) then
if (reset=’1’) or (locked_40_96=’0’) then
a_in_subtract<=(others=>’0’);
b_in_subtract<=(others=>’0’);
else
case control_40_tmp is
when 5 =>
a_in_subtract<=mlt_4;
b_in_subtract<=bias_x1_in_n;
when 6 =>
a_in_subtract<=mlt_5;
b_in_subtract<=bias_x2_in_n;
when others =>
a_in_subtract<=(others=>’0’);
b_in_subtract<=(others=>’0’);
```

```
end case;
end if;
end if;
end process subtract_in_reg;

reg_out_subtract: process(clk_40_96)
begin
if (rising_edge(clk_40_96)) then
if (reset='1') or (locked_40_96='0') then
sub_7<=(others=>'0');
sub_8<=(others=>'0');
else
case control_40_tmp is
when 7 =>
sub_7<=c_out_subtract;
when 8 =>
sub_8<=c_out_subtract;
when others => null;
end case;
end if;
end if;
end process reg_out_subtract;

i_mulp_gain_bias : mulp_gain_bias
PORT MAP (
aclk => clk_40_96_bar,
s_axis_a_tvalid => '1',
s_axis_a_tdata => a_in_mult,
s_axis_b_tvalid => '1',
s_axis_b_tdata => b_in_mult,
m_axis_result_tvalid => m_axis_result_tvalid_mult,
m_axis_result_tdata => c_out_mult
);

i_subtract_gain_bias : subtract_gain_bias
PORT MAP (
aclk => clk_40_96_bar,
s_axis_a_tvalid => ’1’,
s_axis_a_tdata => a_in_subtract,
s_axis_b_tvalid => ’1’,
s_axis_b_tdata => b_in_subtract,
m_axis_result_tvalid => m_axis_result_tvalid_subtract,
m_axis_result_tdata => c_out_subtract
);
end Behavioral;
```