DESIGN OF AN ADAPTIVE - PASSIVE DYNAMIC VIBRATION ABSORBER COMPOSED OF A STRING - MASS SYSTEM EQUIPPED WITH NEGATIVE STIFFNESS TENSION ADJUSTING MECHANISM

by

Mustafa Ali ACAR

B.S., Mechanical Engineering, Bogazici University, 2008

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Mechanical Engineering

Boğaziçi University

2011

ACKNOWLEDGEMENTS

First and foremost, I would like to thank Assist. Prof. Çetin Yılmaz for letting me to work on his interesting and innovative idea and supervising me through the journey of evolving this idea and creating this thesis. In every single step of the study, he was always available, encouraging, supportive, understanding and full of ideas. Working with him was always exceptionally inspiring and a pleasure. I learned so much from him. Whenever I seek counsel about innovative design and solving problems in the future, I know from whom I can find the answer.

I also acknowledge Assist. Prof. Arda Deniz Yalçınkaya, Prof. Eşref Eşkinat and Prof. Emre Köse for their precious times to read this thesis and giving critical comments about it.

I was extraordinarily fortunate in working and spending time with Assoc. Prof. Vahan Kalenderoğlu during my research assistantship. He always shared with me his deep experiences in science and engineering and also generally in life. He made great contribution to this thesis with his invaluable ideas and critics during the course of the study. He enriched my growth both in my profession and my personality. I will always consider him as a mentor. I hope I can always find the oppotunity of sharing ideas with him in the future.

I also want to thank to all my colleagues and friends in the University, especially to Kamil Koçak and İsmail Hakkı Şahin for their willingness of sharing ideas about our studies and also for their close friendships. They gave me a pleasant working environment.

I would also acknowledge my closest friends Anıl İlter, Yakup Kale, Gürsoy Yazaroğlu and Osman Yücel for their advice and their willingness to share their bright thoughts with me and for the joyful years we spent so far. Especially, thank you Gürsoy for constantly asking the state of the development and motivating me to keep up.

I convey special acknowledgement to İbrahim Mutlu for his outstanding efforts and teachings during the production of the prototype. Without his skills and knowledge successful realization of the project would be much harder.

Words fail me to express my appreciation to my dear Gizem. While as a colleague her intelligence and skills in science and engineering made great contributions to my thesis, as a girlfriend her dedication, love and persistent confidence in me, has taken the load off my shoulder. I owe her for shining on me even when my motivation of working was obscured by clouds.

My parents deserve special mention for their inseparable support. My father, Ziyat Ebuziya and my mother, Aynur always did their best ever since I was a child for my education, my development of personality and our standards of living. Also, Büşra and Tuğba, thank you for being supportive and caring sisters.

Finally, I would like to thank everybody who was important to the successful realization of this thesis, as well as expressing my apology that I could not mention personally one by one.

ABSTRACT

DESIGN OF AN ADAPTIVE - PASSIVE DYNAMIC VIBRATION ABSORBER COMPOSED OF A STRING - MASS SYSTEM EQUIPPED WITH NEGATIVE STIFFNESS TENSION ADJUSTING MECHANISM

In this study, a new adaptive-passive dynamic vibration absorber design is discussed. This proposed design is composed of a stiff string under tension with a central mass attachment as a dynamic vibration absorber (DVA), a negative stiffness mechanism as a string tension adjustment aid and a tuning controller to make it adaptive. Dynamic properties of adaptive-passive DVA systems are adjusted in real-time by generally varying their stiffness. The adaptive-passive DVA design subject to this thesis uses the string tension as a tuning parameter. The dependency of the natural frequencies of this system on the string tension is analyzed using finite element method and verified analytically. Additionally, a method for adjusting the string tension with almost zero effort is proposed. To achieve this goal, the design incorporated a negative stiffness element to create a quasi-zero stiffness and constant zero-force mechanism when combined with the string. Force-displacement analysis of a system composed of a pre-loaded spring and a rigid link is examined analytically. It was shown that the system can have constant negative stiffness behavior. A string tension adjustment algorithm is created which tunes the DVA system depending on the magnitude and frequency of the most dominant component of the vibration signal. Finally, a prototype of the complete design is built. A series of experimental procedures are conducted on the prototype with the intention of verifying the theoretical calculations. Results obtained from these experiments are also given in the thesis.

ÖZET

NEGATİF DİRENGEN MEKANİZMAYLA GERGİNLİĞİ AYARLANABİLEN KÜTLE EKLENMİŞ GERGİN TEL KULLANARAK UYARLANABİLİR-PASİF DİNAMİK TİTREŞİM SÖNÜMLEYİCİ TASARIMI

Bu çalışmada, yeni bir uyarlanabilir - pasif dinamik titreşim sönümleyici tasarımı anlatılmaktadır. Tasarım kabaca; titreşim sönümleyici olarak kullanılacak gerilim altındaki bir telin ortasına yerleştirilmiş bir kütle sistemi, telin gerginliğini ayarlayacak sisteme yardımcı olması amacıyla kullanılacak negatif direngenliğe sahip bir mekanizma ve sistemi çalışma koşullarına göre ayarlayan sağlayan bir karar mekanizmasından oluşmaktadır. Uyarlanabilir - pasif DTS sistemlerinin dinamik özelliklerinin gerçek zamanlı olarak değiştirilebilmesi gerekmektedir. Tezdeki DTS sisteminde ise bu ayar parametresi telin gerginliği olarak belirlenmiştir. Bu sebeple DTS sisteminin doğal frekansının telin gerginliğine bağlı olarak nasıl değiştiğinin analizi sonlu elemanlar yöntemiyle yapılmış ve sonuçların var olan analitik çözümlerle tutarlı oldukları görülmüştür. DTS sisteminin dinamik özelliklerini ayarlamakta kullandığımız telin gerginliğinin sürekli değiştirilme işini hiç güç sarfetmeden yapabilmek için de bir metot önerilmiştir. Negatif direngen bir mekanizmanın DTS sisteminin teli ile birleştirilmesi sonucu neredeyse hiç direngenliği olmayan bir düzenek yaratmak, dolayısıyla telin gerginliğini ayarlama işindeki gerekli kuvvet için bu mekanizmadan yardım alınmıştır. Önyükleme yapılmış bir yay ve rijit bir koldan oluşan mekanizmanın kuvvet - deformasyon davranışı analitik olarak hesaplanmıştır. Bahsedilen bu sistemin lineer ve negatif eğimli kuvvet - deformasyon davranışı olabileceği gösterilmiştir. Ana yapının titreşiminin baskın frekansını ve genliğini ölçüp telin gerginliğini ayarlayacak bir program akış şeması yaratılmıştır. Son olarak da tasarımı doğrulamak amacıyla bir prototip üretilmiştir. Hesaplamaları doğrulamak adına üretilen sistem üzerinde bir dizi deney yapılmış ve elde edilen sonuçlar tezde verilmiştir.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

c_{a}	Damping of the vibration absorber
E_{ls}	Elastic modulus of the cantilever leaf spring
E_{str}	Elastic modulus of the string material
f_{a}	Normalized excitation frequency with respect to absorber natural frequency
f_{s}	Normalized excitation frequency with respect to structure natural frequency
I_{ls}	Second moment of inertia of the cantilever leaf spring
k_{a}	Stiffness of the vibration absorber
k_{ns}	Stiffness of the negative stiffness mechanism
k_{s}	Stiffness of the primary structure
k_{t}	Total stiffness of the positive and negative stiffness combination
k_{v}	Stiffness of the vertical spring
L_0	Undeformed length of the vertical spring
L_1	Length of the vertical spring at maximum compression
L_1	Position of the DVA mass w.r.t. base hinge
L_2	Position of the primary spring w.r.t. base hinge
L_{link}	Length of the rigid link
L_{ls}	Length the cantilever leaf spring
L_{m}	Length of the attached mass
L_{str}	String length
m_{a}	Mass of the vibration absorber
m_{s}	Mass of the primary structure
r_{str}	String cross-section radius
t	Time
T_{str}	String tension
$u(x,t)$	Transverse displacement of the string at position x and time t

x	Position along string
x_{a}	Displacement of the vibration absorber
x_{b}	Displacement of the base
x_{s}	Displacement of the primary structure
X_{a}	Laplace transform of x_{a}
X_{b}	Laplace transform of x_{b}
X_{s}	Laplace transform of x_{s}
λ_0	Linear mass density of string
$\lambda_{\mathrm{str}}(x)$	Non-homogeneous linear mass density of string
μ	Ratio of absorber and structure masses
Φ	Mode shape of the string
ρ	Density
ω	Excitation frequency
ω_{a}	Natural frequency of the vibration absorber
$\omega_{\mathrm{a}}^{(\mathrm{n})}$	n^{th} natural frequency of the string-mass vibration absorber
ω_{s}	Natural frequency of the primary structure

LIST OF ACRONYMS/ABBREVIATIONS

1D	One Dimensional
2D	Two Dimensional
3D	Three Dimensional
APVA	Adaptive-Passive Vibration Absorber
DOF	Degree of Freedom
DSP	Digital Signal Processing
DVA	Dynamic Vibration Absorber
I^2C	Inter-Integrated Circuit
FEM	Finite Element Method
FFT	Fast Fourier Transform
QZS	Quasi-Zero Stiffness
RMS	Root Mean Square
UART	Universal Asynchronous Receiver/Transmitter
SDOF	Single Degree of Freedom
SPI	Serial Peripheral Interface Bus
TVA	Tuned Vibration Absorber

1. INTRODUCTION

Most of the mechanical and structural systems encounter unwanted vibrations. There have been numerous studies conducted to find ways to reduce the effects of those vibrations, such as damage, fatigue failure, and low human comfort. Traditional treatment methods to reduce the effects of unwanted vibrations generally involve structural modifications, which are often time consuming and expensive. Rather than trying to alter the complete design of structures subjected to vibration, usage of a suitable dynamic vibration absorber (DVA) system can sometimes significantly reduce the costs and complexity of the design. Dynamic vibration absorbers are extensively used in many different application extensively since their invention [1].

1.1. Dynamic Vibration Absorbers (DVA)

A dynamic vibration absorber (DVA) can generally be viewed as a system attached to a structure with the intention of suppressing of its vibration. There are various DVA designs reported in the literature. Each design either directly use a mass-spring system or continuous systems that can be reduced to an equivalent mass-spring system. DVAs have been widely used to suppress structural vibrations. The simplest DVA that can be realized is an absorber having one resonance frequency. A schematic of such a system is shown in Figure 1.1.

In this case, if the natural frequency of the DVA coincides with the frequency of excitation, the vibration of the primary structure, x_s, can be suppressed [2, 3]. This vibration attenuation is achieved by the counteraction to the excitation force with the force applied by the DVA.

Although, the idea of a DVA is a simple attachment of an additional vibratory system to a structure, the designs and the aims of the applications vary [4–7]. Depending on the requirements of the application, DVAs can be constructed as passive, adaptive-passive or active devices. While a passive absorber is generally effective at a

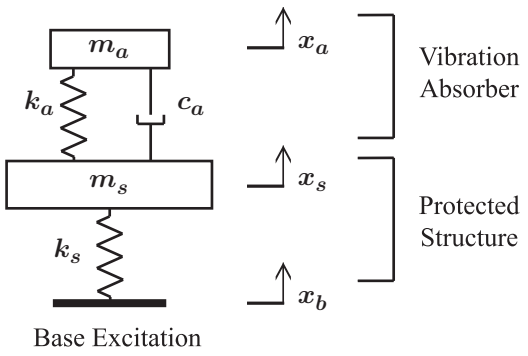


Figure 1.1. A schematic of a simple structure appended with a DVA.

single frequency of vibration, adaptive-passive and active DVAs can be constructed so that vibrations within a range of frequencies can be attenuated. The operation principle behind the active vibration absorbers is that by using an actuator, they emulate the action of a passive DVA as if it is tuned to every frequency of its range. Thus, they are quite effective in many cases. However, using an actuator as a counteracting mechanism to vibration is an energy consuming method and is not always feasible. The adaptive-passive vibration absorbers can be thought of a hybrid version of both passive and active vibration absorbers. They are generally composed of a passive DVA and a self tuning mechanism. During operation, the excitation conditions are constantly monitored and the parameters of the passive DVA is tuned accordingly. Rather than counteracting to the vibration of the protected structure, this type of DVAs only consume energy to vary the parameters, usually the stiffness, that govern their dynamic behavior [4, 8–10].

1.1.1. Example DVA Designs Reported in the Literature

Some designers still favor the simplest mass-spring form, like the system used in [11]. The goal of this design was to minimize wind induced vibrations on a pipeline. This design did not need very accurate tuning, or did not suffer mistuning caused by wind conditions; because the pipeline section on which these DVAs are applied had very dense tonal modes below 5 Hz, hence a DVA tuned to 5 Hz most probably coincided

with or be near to a resonant frequency [4].

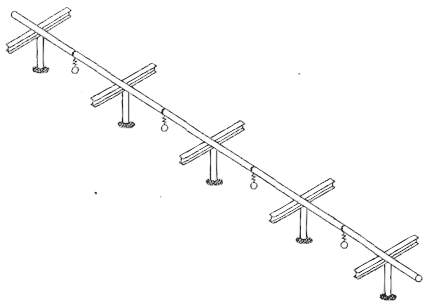


Figure 1.2. Arctic pipeline with one DVA per span [11].

Another common method of realizing passive DVAs is using simple pendulums. In these designs, the natural frequency is determined by the gravity and the pendulum length. The pendulum type absorbers have been used extensively in tall buildings, generally to reduce wind-induced lateral vibrations. The advantage of this type of design is that the setup is simple and generally free of maintenance. There are several damping methods for such absorbers; such as using internal friction of cables of pendulum as damping element, air dampers, or dampers with plastic granular fillings. Some designers had to overcome size problems of the pendulums, for example a 0.12 Hz pendulum has to be about 16 m long. Sometimes such big devices may be inappropriate. In order to overcome the problem, some pendulum DVAs are modified with spring elements to alter their dynamic behavior without increasing their size [3, 12–14].

Additionally in [15] an electromechanical vibration absorber design is proposed, which employed piezoelectric materials connected to some passive electrical circuits. When the connected circuit consists of a resistive element, the overall configuration behaves like a viscoelastic material, thus dissipate mechanical energy. Furthermore when the circuit includes resistive and inductive elements, it introduces a resonant frequency. Therefore the overall configuration behaves like a mechanical vibration absorber. This type of absorber design has the advantage of being solid-state and lightweight.

A basic implementation of an active-passive vibration absorber is designed in [16]. The system uses a force actuator controlled by a feedback algorithm that operates in parallel with a passive DVA. The purpose is to give it the ability to absorb all disturbances in a given frequency band. This can be done by developing a DVA that has a band of resonant frequencies, rather than a single frequency. Since resonance frequencies of DVAs enable absorption of vibrations when appended to a host structure, a band-pass resonance frequency range of the DVA causes a band-stop frequency range for the combined system.

There are also many types of adaptive-passive vibration absorber designs reported in the literature, such as systems having variable stiffness through mechanical mechanisms [9, 10, 17], through controllable smart materials [18–21] and through electric circuits shunted with special transducer elements [15, 22].

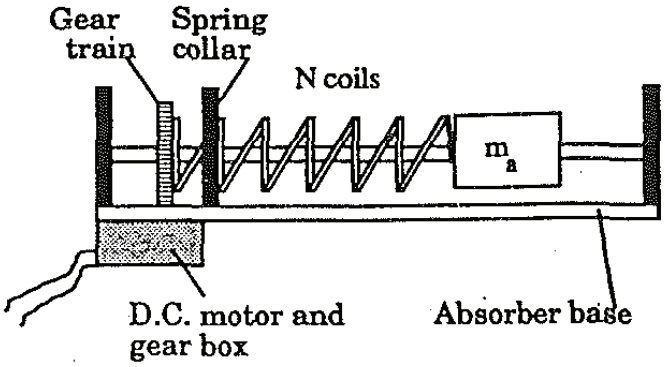


Figure 1.3. A coil spring variable stiffness DVA [9].

A simple mechanism designed to change the stiffness of a DVA is studied in [9]. As shown in the conceptual drawing of the design in Figure 1.3, a DC motor is used to change the effective length of a coil spring during operation. Therefore, the stiffness of the device can be changed in realtime. This design enables an easy to analyze relationship between the spring collar position and the effective stiffness. Therefore, it is a simple-to-model and simple-to-control mechanism. But the reaction time of the

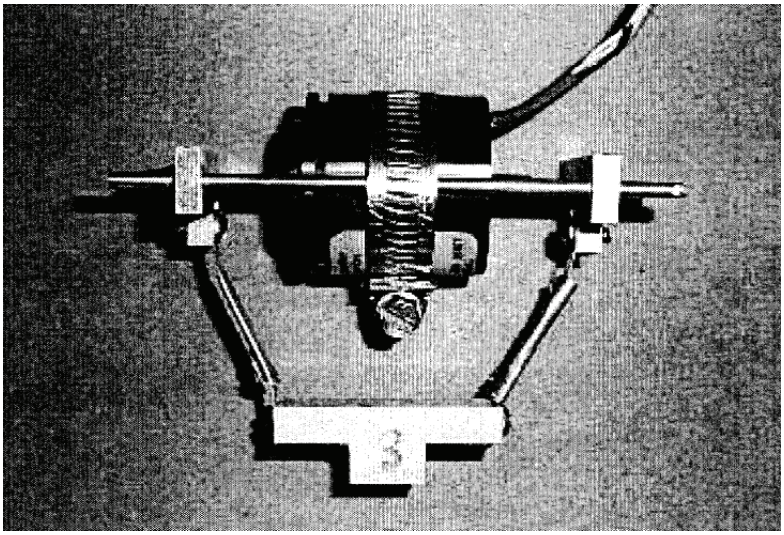


Figure 1.4. Variable stiffness DVA using V-shaped springs [23].

system purely depends on how fast and accurately the collar is moved along the spring, and also the span of the collar may be too long for an adequate range of stiffness values.

Another adaptive DVA design is studied in [23] (see Figure 1.4), which uses a stepper motor to change the angle between two springs placed in a V-shape. The variable angle enables the system to change the resultant spring constant to the desired value. The reactive mass for this design is the stepper motor itself; therefore the designer eliminated the unnecessary weight, which in fact increases the mass of the primary structure [7]. Since the span of the distance between the springs has to be covered every time to alter the stiffness from the maximum to the minimum value, during steep changes of disturbance frequency, the device may not be able to react as rapidly as it should.

Another interesting mechanical adaptive DVA design uses curved cross-section cantilever beams (see Figure 1.5). The system tries to change the stiffness by modifying the curvature of the cross-section of the beam, thus changing the flexural rigidity. Methods offered to alter the shape of the cross-section are either forming the beam using two materials with different thermal expansion coefficients and exposing it to

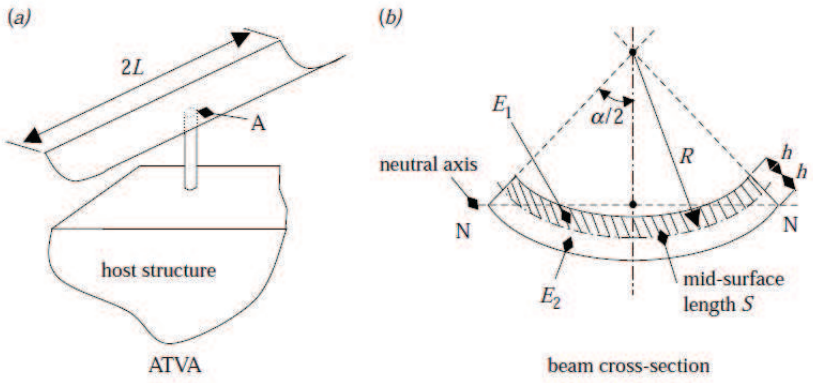


Figure 1.5. Variable stiffness DVA using cross-sectional curvature adjustment of cantilever beams (a) General schematic of the system (b) Cross-section view of the curved beam [17].

temperature changes or putting piezoceramic actuators on one side of the beam. The disadvantage of the design is that the relation between the natural frequency of the beam and its cross-section curvature is highly nonlinear and hard to estimate. Also, both of the methods to change the curvature are either too slow or consumes energy through piezoceramic material [17].

There are also solid-state adaptive DVA designs in the literature. For example, in [22] piezoelectric elements are attached to a passive resonant electric circuit much like in [15]. But the difference from the passive electromechanical DVA is that this system applies a synchronized switching algorithm to the circuit. By continuously monitoring the response of the primary system and piezoelectrical elements, the algorithm determines the timing of switching actions. With this method it was possible to alter the resultant natural frequency of the electromechanical DVA. As a result the system can change its dynamic characteristics without consuming much energy, in fact it only consumes for the switching action. Figure 1.6 shows the concept of both mechanical and electrical parts of the system.

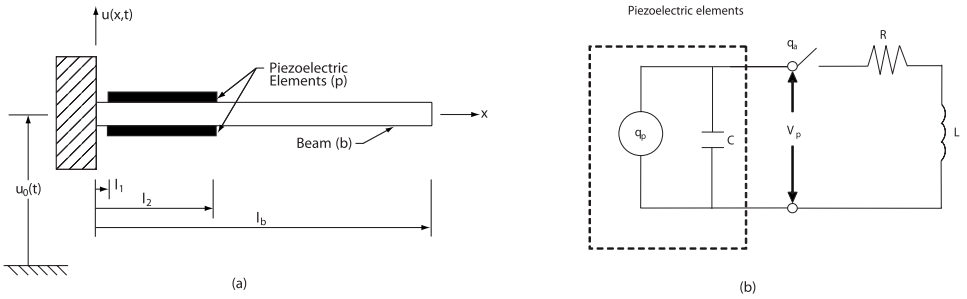


Figure 1.6. (a) Mechanical and (b) Electrical schematics of an adaptive-passive DVA using synchronously switched piezo-materials [22].

1.2. Negative Stiffness Elements

The behavior of ordinary springs is familiar to almost everyone: the more the spring is deformed, the more it resists to deformation. The slope of the reaction force of spring with respect to its deflection is defined as its stiffness. For ordinary springs this slope is positive.

There are, however, mechanisms that can exhibit "negative" stiffness behavior, either partially or over their entire deflection span. Beams at post-buckling conditions, or linkages accompanied with pre-compressed springs are only two of the many ways of creating negative stiffness mechanisms [24]. Negative stiffness can generally occur when the deforming mechanism has stored energy. Therefore, these systems are generally unstable [25].

Despite the fact that negative stiffness systems are unstable, in combination with positive stiffness elements they can be stabilized and various useful mechanisms of different purposes can be synthesized with them. Such as, "quasi-zero stiffness" (QZS) systems for vibration isolation purposes, "bi-stable" mechanisms for latching actuator systems or "constant-force" mechanisms used in robotics can be built with negative stiffness elements.

1.2.1. Applications Reported in the Literature Employing Negative Stiffness Elements

One of the major areas of applications of negative stiffness mechanisms is vibration isolation systems. Structures that need to be supported vertically against gravity, a certain amount of mounting stiffness is required in order to carry the load without causing large static deflections. On the other hand, if isolation of the structure from the vibrations on its base is required, then the mounting stiffness should be as low as possible.

Quasi-zero-stiffness (QZS) mounting systems are widely used in the literature to overcome this basic trade-off of vibration isolation. Basically, QZS springs are adjusted so that they can carry a certain amount of static load. Unlike normal springs, however, finite deflections in the vicinity of this equilibrium point cause almost zero variation in the spring force of these non-linear QZS systems. Thus, both high static stiffness and low dynamic stiffness requirements are satisfied with the same mounting. A conceptual illustration of such force-deflection behavior of QZS systems is shown in Figure 1.7 [26].

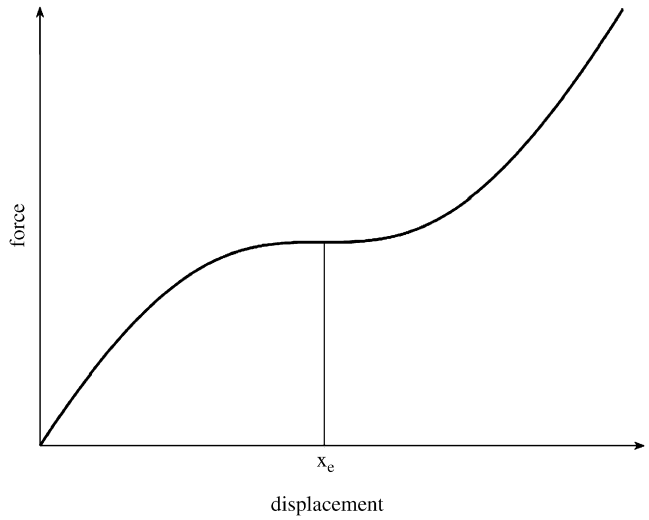


Figure 1.7. A typical force-deflection characteristic of a QZS system [26].

Negative stiffness springs are used in [27] in combination with positive stiffness

springs to create vehicle driver seats that can carry certain static loads whilst exhibiting almost zero dynamic stiffness on a range of deflections. In this system, natural frequency is reduced dramatically, hence a vertical vibration isolation under the effect of gravity is attained. Similar to [27], systems discussed in [28–30] also employed different types of negative stiffness configurations to obtain QZS vibration isolators.

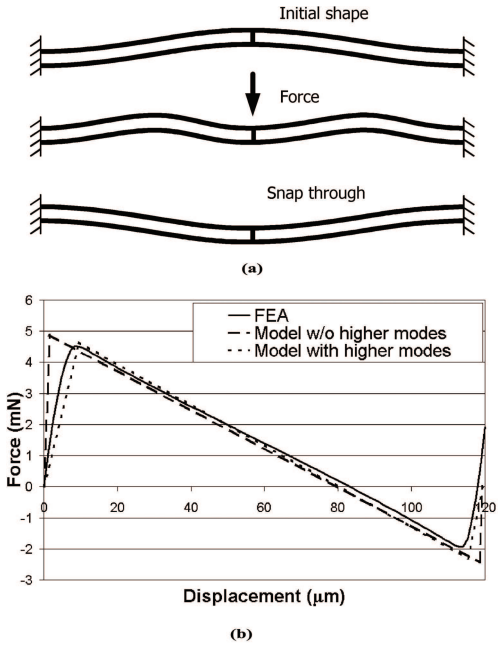


Figure 1.8. Buckled shape bistable beam: (a) schematic (b) forced-deflection curve [31].

In [31], a buckled shaped beam is analyzed to create a mechanically bi-stable structure (see Figure 1.8). Such mechanisms can be used in MEMS devices such as relays, valves, clips, threshold switches and memory cells, etc. A bi-stable system like this shows three different stiffness zones. From static equilibrium position up to a certain peak reaction force, it resists to deflection applied at its mid-point as an ordinary spring. After that peak value, the spring still resists further deformation, however, the resistance tends to decrease by deflection until it reaches to zero, which is an unstable force equilibrium point. Since the mechanism is fairly symmetric, after that unstable equilibrium point, the force-deflection behavior of the system is like a mirror image of the first half. Thus, the beam tends to snap-through the unstable

equilibrium and reach another static equilibrium point. The force-displacement curve of this mechanism between the two peak reaction forces shows a negative stiffness behavior (see Figure 1.8b).

Other common area of application of negative stiffness elements are the constant-force generation mechanisms of different purposes. Systems analyzed in [32, 33] have negative stiffness components, therefore, the designers were able to create mechanisms that can apply constant force over a fairly wide range of deflections. The area of applications of such constant-force systems range from the desk lamps, that can be brought to various positions and still remain in equilibrium, to surgical laser stands.

1.3. Motivation and Research Objective

The main goal of this study is to develop a novel design of variable stiffness adaptive-passive dynamic vibration absorber. The motivations behind this study can be summarized as:

- The potential of adaptive-passive vibration absorbers in the challenge against preventing unwanted vibrations of structures over wide ranges of frequencies
- Introducing a new tunable vibration absorber design having improved properties over current designs in the literature
- Proving that the new design can effectively be used as a DVA
- Having a tuning parameter that can be adjusted accurately and quickly
- Creating a DVA compact in size, and applicable to different structures with ease
- Achieving very low energy requirements for the self-tuning effort of the system
- Developing an autonomous vibration measurement and tuning system

To achieve these goals, the design reported in this study incorporates a stiff string under tension attached with a mass at its center as a passive vibration absorber device. (see Figure 1.9) The tuning parameter for the system is selected as the string tension, thus with very small displacement inputs to the string end, a wide range of natural frequency variation can be achieved. Although there are many different designs of

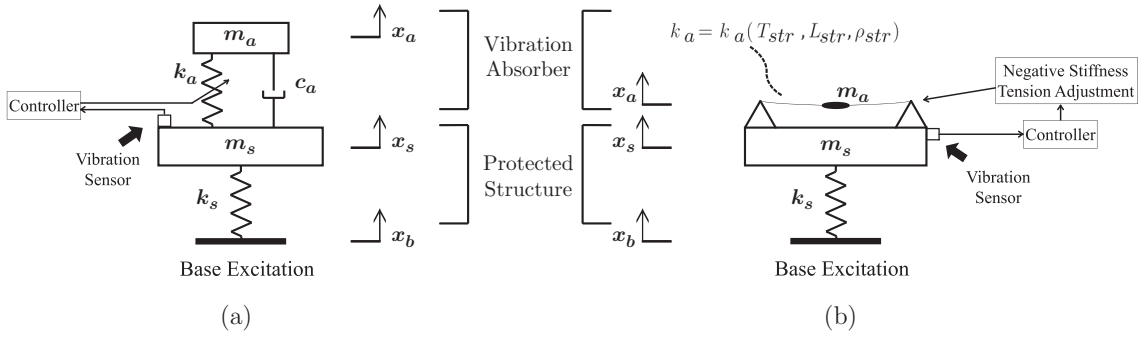


Figure 1.9. A schematic comparison between (a) a generic DVA mechanism and (b) the proposed string with attached mass DVA system.

dynamic vibration absorbers reported in the literature, to the best of our knowledge, this for creating a dynamic vibration absorber has not been applied previously.

The system is equipped with a negative stiffness element in combination with the string so that a resultant quasi-zero-stiffness and zero-constant-force mechanism is formed. With the help of this mechanism, the alterations on the string tension can be carried out with a small actuator, quickly and accurately without consuming much energy. Negative stiffness mechanisms are generally used for different purposes, such as vibration isolation or constant force generation mechanisms. The concept of reducing an actuator's effort in an adaptive DVA system using a negative stiffness mechanism is however never used in the literature.

In addition to this mechanical design; an embedded microcontroller system is developed for data acquisition, signal processing, tuning algorithm execution and actuator driving purposes. With the help of this system, the designed DVA system does not require any computer installation nearby to be able to operate.

After the design and analysis phases, a small prototype of the system is constructed for proof-of-concept purposes.

2. THEORETICAL BACKGROUND AND DESIGN STAGES

2.1. Forced Vibration of an Undamped Single Degree of Freedom (SDOF) System

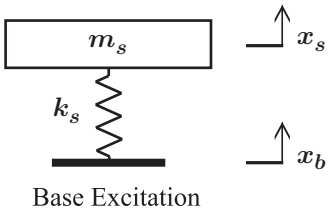


Figure 2.1. Undamped Base Excitation of a SDOF Mass-Spring System.

As a preliminary analysis, let us try to explain the concept of resonance and natural frequency using the undamped single degree of freedom mass-string system shown in Figure 2.1. As illustrated, the system is forced with base excitation. In order to calculate its response to harmonic excitation, let us first write its equation of motion

$$m_{\mathrm{s}}\ddot{x}_{\mathrm{s}} + k_{\mathrm{s}}x_{\mathrm{s}} = k_{\mathrm{s}}x_{\mathrm{b}} \tag{2.1}$$

and if we define $\omega_{\mathrm{s}} \triangleq \sqrt{\frac{k_{\mathrm{s}}}{m_{\mathrm{s}}}}$, the equation becomes

$$\ddot{x}_{\mathrm{s}} + \omega_{\mathrm{s}}^2 x_{\mathrm{s}} = \omega_{\mathrm{s}}^2 x_{\mathrm{b}}. \tag{2.2}$$

Let us take the Laplace transform of both sides of the equation to find the transfer function between the base excitation and the displacement of the mass, $x_{\mathrm{s}}(t)$, with the

assumption of zero initial conditions

$$\mathcal{L}\left\{\ddot{x}_{\mathrm{s}} + \omega_{\mathrm{s}}^2 x_{\mathrm{s}}\right\} = \mathcal{L}\left\{\omega_{\mathrm{s}}^2 x_{\mathrm{b}}\right\} \tag{2.3}$$

$$\Rightarrow (s^2 + \omega_{\mathrm{s}}^2) X_{\mathrm{s}} = \omega_{\mathrm{s}}^2 X_{\mathrm{b}} \tag{2.4}$$

where $X_{\mathrm{s}}(s) = \mathcal{L}\{x_{\mathrm{s}}(t)\}$ and $X_{\mathrm{b}}(s) = \mathcal{L}\{x_{\mathrm{b}}(t)\}$. In order to find the response of the system to harmonic excitation of angular frequency ω, replace the s terms in the transfer function with $i\omega$ where $i = \sqrt{-1}$.

$$\frac{X_{\mathrm{s}}(i\omega)}{X_{\mathrm{b}}(i\omega)} = \frac{\omega_{\mathrm{s}}^2}{\omega_{\mathrm{s}}^2 - \omega^2}$$

$$= \frac{1}{1 - \frac{\omega^2}{\omega_{\mathrm{s}}^2}} \tag{2.5}$$

The magnitude of the Equation 2.5 is defined as the motion transmissibility, TR_{bs} from x_{b} to x_{s}. Now, rewriting the transmissibility using a normalized frequency parameter, $f_{\mathrm{s}} = \frac{\omega}{\omega_{\mathrm{s}}}$, yields

$$TR_{\mathrm{bs}} = \left|\frac{X_{\mathrm{s}}(i\omega)}{X_{\mathrm{b}}(i\omega)}\right|$$

$$= \frac{1}{\sqrt{(1 - f_{\mathrm{s}}^2)^2}}. \tag{2.6}$$

The transmissibility is the ratio of displacement magnitudes of $|X_{\mathrm{s}}|$ and $|X_{\mathrm{b}}|$, thus a value greater than unity means that the displacement magnitude of the mass is greater than the base displacement magnitude. In other words, the displacement input is amplified by the system. Similarly, if the value of the transmissibility is smaller than unity, it is called that the input displacement is attenuated.

If we look at the TR_{bs} vs. f_{s} plot in Figure 2.2, when the value of normal-

ized parameter is one (i.e. $\omega = \omega_s$), the transmissibility, TR_{bs}, goes to infinity. A phenomenon called resonance occurs when the excitation frequency ω is equal to the natural frequency of the system, ω_s [2].

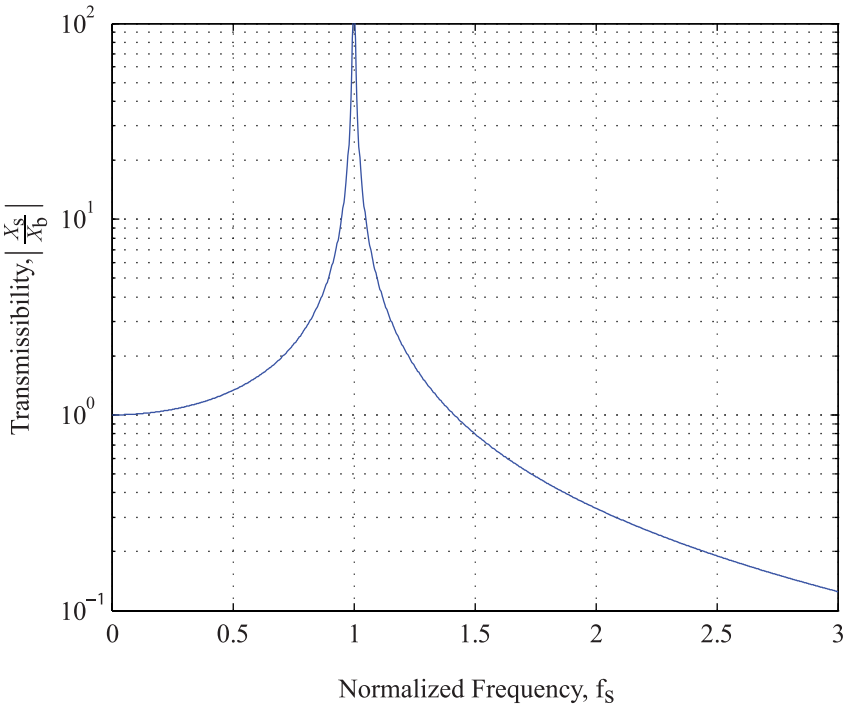


Figure 2.2. Transmissibility of the undamped the SDOF system under base excitation (TR_{bs} vs. f_s).

2.2. Forced Vibration of an Undamped Two Degrees of Freedom System

In the previous section, the resonance and natural frequency concepts are briefly mentioned on the simplest possible vibratory structure. In this section, some of the foundations of a dynamic vibration absorber are explained on an undamped two degrees of freedom mass-spring structure shown in Figure 2.3.

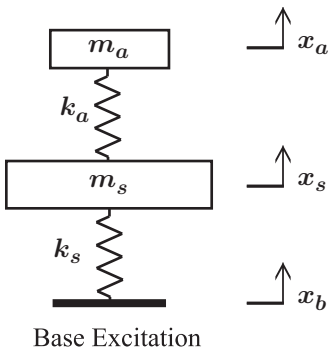


Figure 2.3. Undamped Base Excitation of a Two Degrees of Freedom Mass-Spring System.

Firstly, we write the equation of motion of this system

$$m_{\mathrm{s}}\ddot{x}_{\mathrm{s}} + (k_{\mathrm{s}} + k_{\mathrm{a}})x_{\mathrm{s}} = k_{\mathrm{a}}x_{\mathrm{a}} + k_{\mathrm{s}}x_{\mathrm{b}} \tag{2.7}$$

$$m_{\mathrm{a}}\ddot{x}_{\mathrm{a}} + k_{\mathrm{a}}x_{\mathrm{a}} = k_{\mathrm{a}}x_{\mathrm{s}}. \tag{2.8}$$

Then, we take the Laplace transform of both sides of these equations, again with the assumption of zero initial conditions

$$\mathcal{L}\left\{m_{\mathrm{s}}\ddot{x}_{\mathrm{s}} + (k_{\mathrm{s}} + k_{\mathrm{a}})x_{\mathrm{s}}\right\} = \mathcal{L}\left\{k_{\mathrm{a}}x_{\mathrm{a}} + k_{\mathrm{s}}x_{\mathrm{b}}\right\} \tag{2.9}$$

$$\mathcal{L}\left\{m_{\mathrm{a}}\ddot{x}_{\mathrm{a}} + k_{\mathrm{a}}x_{\mathrm{a}}\right\} = \mathcal{L}\left\{k_{\mathrm{a}}x_{\mathrm{s}}\right\} \tag{2.10}$$

$$\Rightarrow (m_{\mathrm{s}}s^2 + (k_{\mathrm{a}} + k_{\mathrm{s}}))X_{\mathrm{s}} = k_{\mathrm{a}}X_{\mathrm{a}} + k_{\mathrm{s}}X_{\mathrm{b}} \tag{2.11}$$

$$(m_{\mathrm{a}}s^2 + k_{\mathrm{a}})X_{\mathrm{a}} = k_{\mathrm{a}}X_{\mathrm{s}}. \tag{2.12}$$

Now, if we define $\mu \triangleq \frac{m_{\mathrm{a}}}{m_{\mathrm{s}}}$, $\omega_{\mathrm{s}} \triangleq \sqrt{\frac{k_{\mathrm{s}}}{m_{\mathrm{s}}}}$, $\omega_{\mathrm{a}} \triangleq \sqrt{\frac{k_{\mathrm{a}}}{m_{\mathrm{a}}}}$, we get also

$$\frac{k_{\mathrm{a}}}{m_{\mathrm{s}}} = \mu\omega_{\mathrm{a}}^2 \tag{2.13}$$

and the equations above will form into

$$(s^2 + \mu\omega_{\mathrm{a}}^2 + \omega_{\mathrm{s}}^2)X_{\mathrm{s}} = \mu\omega_{\mathrm{a}}^2 X_{\mathrm{a}} + \omega_{\mathrm{s}}^2 X_{\mathrm{b}} \tag{2.14}$$

$$(s^2 + \omega_{\mathrm{a}}^2)X_{\mathrm{a}} = \omega_{\mathrm{a}}^2 X_{\mathrm{s}}. \tag{2.15}$$

With some minor manipulations on the Equations 2.14 and 2.15, the transfer functions between X_{b} and X_{s}, and between X_{b} and X_{a} are obtained as following:

$$\frac{X_{\mathrm{s}}(s)}{X_{\mathrm{b}}(s)} = \frac{\omega_{\mathrm{s}}^2(s^2 + \omega_{\mathrm{a}}^2)}{(s^2 + \mu\omega_{\mathrm{a}}^2 + \omega_{\mathrm{s}}^2)(s^2 + \omega_{\mathrm{a}}^2) - \mu\omega_{\mathrm{a}}^4} \tag{2.16}$$

$$\frac{X_{\mathrm{a}}(s)}{X_{\mathrm{b}}(s)} = \frac{\omega_{\mathrm{a}}^2\omega_{\mathrm{s}}^2}{(s^2 + \mu\omega_{\mathrm{a}}^2 + \omega_{\mathrm{s}}^2)(s^2 + \omega_{\mathrm{a}}^2) - \mu\omega_{\mathrm{a}}^4} \tag{2.17}$$

By replacing the s terms in the above equations with $i\omega$, where ω is again the harmonic excitation angular frequency, and taking the absolute values of them the transmissibilities TR_{bs} and TR_{ba} are obtained.

$$TR_{\mathrm{bs}} = \left|\frac{X_{\mathrm{s}}(i\omega)}{X_{\mathrm{b}}(i\omega)}\right|$$

$$= \left|\frac{\omega_{\mathrm{s}}^2(\omega_{\mathrm{a}}^2 - \omega^2)}{(\mu\omega_{\mathrm{a}}^2 + \omega_{\mathrm{s}}^2 - \omega^2)(\omega_{\mathrm{a}}^2 - \omega^2) - \mu\omega_{\mathrm{a}}^4}\right| \tag{2.18}$$

$$TR_{\mathrm{ba}} = \left|\frac{X_{\mathrm{a}}(i\omega)}{X_{\mathrm{b}}(i\omega)}\right|$$

$$= \left|\frac{\omega_{\mathrm{a}}^2\omega_{\mathrm{s}}^2}{(\mu\omega_{\mathrm{a}}^2 + \omega_{\mathrm{s}}^2 - \omega^2)(\omega_{\mathrm{a}}^2 - \omega^2) - \mu\omega_{\mathrm{a}}^4}\right| \tag{2.19}$$

If we look at the Equation 2.18, it is clearly seen that the value of TR_{bs} becomes zero when the excitation frequency and the natural frequency of the upper mass coin-

cide (i.e. $\omega = \omega_a$). In other words, the vibration of m_s totally stops at that excitation frequency. Also, as can be observed that this behavior does not depend on any parameter of the system, if $\mu > 0$. Note that when $\mu = 0$, it means that no absorber is attached to the structure. Such points, where the transmissibility becomes zero are called antiresonance frequencies.

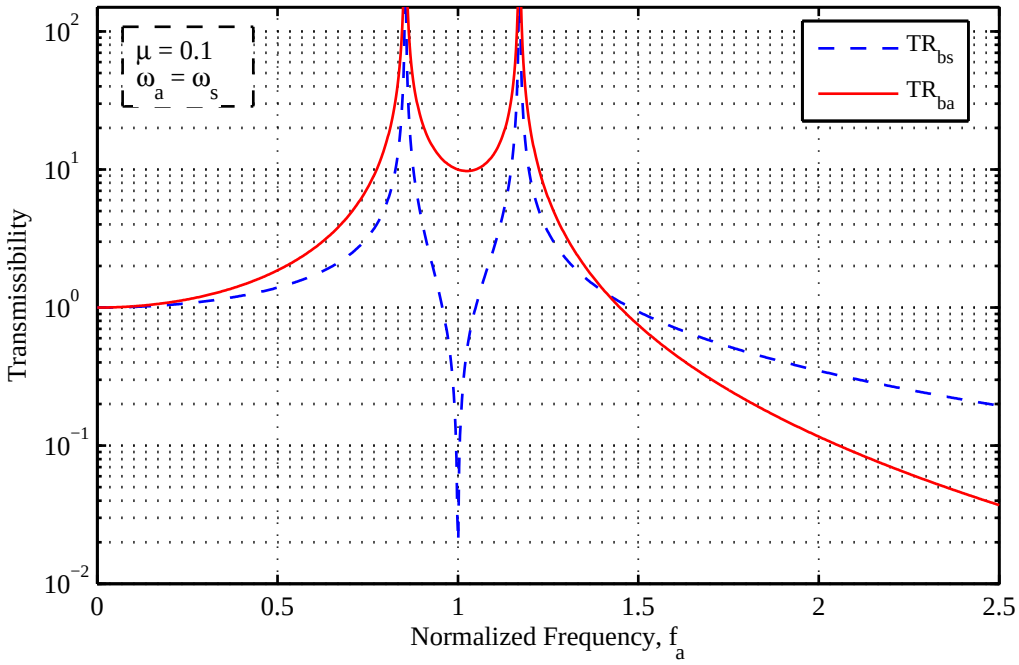


Figure 2.4. Transmissibilities of m_s and m_a (TR_{bs} vs. f_a and TR_{ba} vs. f_a).

This behavior constitute the primary principle of a DVA. When they are attached to a dynamic system, an antiresonance frequency occurs in the transmissibility of the combined system. Therefore, by adjusting the natural frequency of the attached system, ω_a, one can determine the location of the antiresonance. Thus, the frequency at which the vibration of the structure is entirely "absorbed", can be selected.

It is also worth noting that, at some certain excitation frequencies the common denominator of the transmissibility relations (Equations 2.18 and 2.19 of the combined system can become zero. In other words, the system can go into resonance. And as

one can expect, the transmissibility relations have two poles for this two DOF system. Therefore, the system, in total, has one antiresonance frequency and two resonance frequencies.

Unlike the antiresonance location, the resonance frequencies of the combined system does depend on parameters like the mass ratio, μ, and natural frequencies of both systems, ω_{s} and ω_{a}. For cases, $\mu > 0$, the resonance frequencies of the combined system is neither equals to ω_{s} nor to ω_{a}. Thus, excitation frequencies that would cause resonance on these two systems separately, do not have resonant effects on the combined system.

The effects of the parameters mentioned above and inclusion of damping in the DVA systems are discussed further in the following sections. However, let us now evaluate the transmissibility functions calculated in this section using an example configuration. But, first we define another normalized frequency value, $f_{\mathrm{a}} = \frac{\omega}{\omega_{\mathrm{a}}}$. Note that this normalized frequency value also indicates the tuning state of the system. When $f_{\mathrm{a}} = 1$, it means that the system is at antiresonance.

In this example, the parameters are selected as; $\mu = 0.1$ and $\omega_{\mathrm{a}} = \omega_{\mathrm{s}}$. The resulting transmissibility curves are shown in Figure 2.4. The antiresonance and the two resonances are clearly shown in the plot. Note that, the value of the transmissibility of the structure, TR_{bs}, increases very sharply around the antiresonance. For instance, in this configuration if $f_{\mathrm{a}} < 0.96$ or $1.06 < f_{\mathrm{a}}$, the transmissibility is even larger than unity. We can define the bandwidth of the system as the range of frequencies around the antiresonance frequency where the vibration suppression is effective. Parameters that determine this range are examined in the following section.

2.3. Dynamic Vibration Absorbers

A dynamic vibration absorber (DVA) can generally be viewed as a system attached to a structure with the intention of suppression of its vibration. This attachment alters the dynamics of the system and increases the degrees of freedom of the system.

As shown previously, the extra DOFs introduced by the addition of the DVA creates extra resonance frequencies, yet not only new resonances occur but also antiresonance frequencies are created by DVAs. At these antiresonances, vibration of the primary structure is significantly reduced.

The simplest DVA application is the system showed in the Section 2.2, in which an absorber mass is attached to a vibratory structure with a spring. The principle of this combination is that the DVA, as a subsystem, counteracts the excitation applied to the structure at a certain frequency, at which the combined system shows antiresonance behavior. This counteraction results in reduced vibration levels compared to the case that does not incorporate a DVA. The antiresonance occurs at a frequency at which the DVA system has a natural frequency, ω_a, when treated separately.

The intention in the incorporation of a DVA can be either to protect the main structure form going into resonance or to reduce vibrations at a certain excitation frequency whether it is a resonant frequency or not. In both cases the natural frequency of the DVA is set to the value of interest. This adjustment, which is also called tuning, is conducted in order to create the antiresonance at the target frequency. Thus, the DVAs are sometimes called as tuned vibration absorbers (TVA) [4].

2.3.1. Bandwidth of Dynamic Vibration Absorbers

The reduction in the vibration levels of a structure by a DVA depends on several factors. First of all, the amount of damping present in a DVA system affects the performance of the system considerably. With an undamped DVA, maximum amount of vibration reduction occurs where the natural frequency of the DVA, ω_a, coincides with the excitation frequency. At frequencies around this value, the performance of the DVA diminishes rapidly. Thus the sensitivity of the system to mistuning becomes bigger compared to damped DVA systems [2, 34]. Secondly, the ratio between the mass of the DVA, m_a, and the mass of the structure, m_s, affects the performance of the DVA on vibration reduction. As the mass ratio, $\mu = m_a/m_s$, increases the two resonant frequencies next to the antiresonance separate further away, thus the effective

region, bandwidth, of the DVA widens.

2.3.1.1. Effect of Damping on the DVA System Bandwidth. As stated previously, one can attain maximum available vibration attenuation performance using a DVA system by not introducing damping to the DVA system. However, this maximum performance is only limited to the point where the excitation frequency exactly coincides with the DVA's natural frequency. And around this maximum performance point, the attenuation reduces sharply.

However, by adding some damping to the DVA system, one can attain a wider and smoother working range for the DVA system. Damped DVA systems cannot reach maximum available attenuation performance, however they also prevent the system to go into resonance at the critical frequencies around the antiresonance point. The damper causes much smoother vibration amplitude changes with respect to the varying frequencies. Thus, the designer can choose to stay on the safe side by sacrificing more available performance

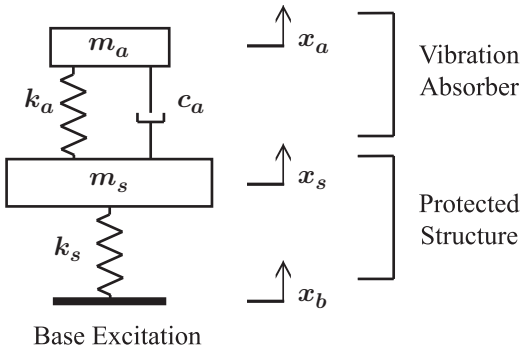


Figure 2.5. Schematic view of a SDOF system appended with a passive damped DVA.

In order to investigate the effect of damping on the dynamical behavior of a DVA system, let us write the transmissibility equation of the system shown in Figure 2.5.

We first define a new dimensionless variable, called damping ratio

$$\zeta = \frac{c_{\mathrm{a}}}{2m_{\mathrm{a}}\omega_{\mathrm{s}}}. \tag{2.20}$$

By adding the damping factor to the Equation 2.8 and following a similar procedure as done previously, one can obtain the transmissibility relation between the base excitation and the structure vibration, TR_{bs} as a function of ω.

$$TR_{\mathrm{bs}} = \left| \frac{\omega_{\mathrm{s}}^2\left(\omega_{\mathrm{a}}^2 + 2i\zeta\omega_{\mathrm{s}}\omega - \omega^2\right)}{\left(\omega^4 - \omega^2\omega_{\mathrm{a}}^2 - \omega^2\omega_{\mathrm{s}}^2 + \omega_{\mathrm{a}}^2\omega_{\mathrm{s}}^2 - \mu\omega^2\omega_{\mathrm{a}}^2\right) + 2i\left(\omega^2\omega_{\mathrm{s}}^3\zeta - \omega^3\omega_{\mathrm{s}}\zeta - \mu\zeta\omega^3\omega_{\mathrm{s}}\right)} \right| \tag{2.21}$$

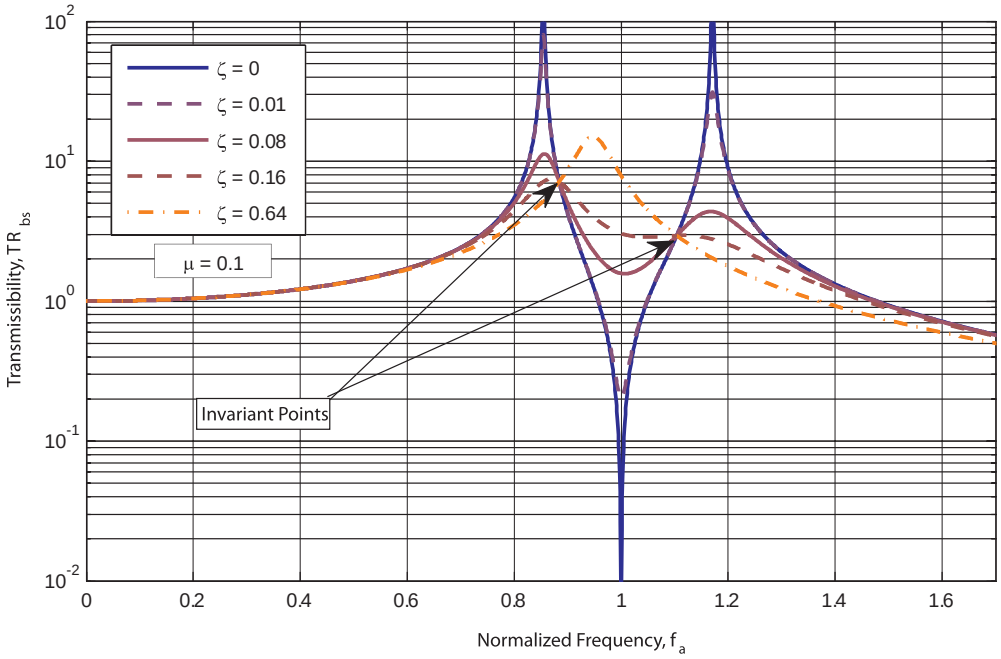


Figure 2.6. Transmissibility of the structure with damped DVA.

The transmissibility Equation 2.21, is evaluated for $\mu = 0.1$, $\omega_{\mathrm{a}} = \omega_{\mathrm{s}}$, $\zeta \in \{0, 0.01, 0.08, 0.16, 0.64\}$. The response of the primary structure is plotted in Figure 2.6 against f_{a}, the dimensionless frequency value.

As shown in the figure, addition of damping to the DVA has two consequences. First, the magnitude of vibration at resonance conditions significantly drops. Thus, the risk of excess vibrations in possible drifts of excitation frequencies is eliminated. Secondly, with increasing damping ratio, the deep and narrow antiresonance region becomes smoother. Therefore, even at perfect tuning condition $f_a = 1$, the vibration attenuation performance becomes lower compared to the undamped case. In addition to these, at damping ratios higher than a certain value (the case where $\zeta = 0.64$ in this configuration, for instance), the antiresonance region completely disappears. The reason is that excess damping between the structure and the absorber masses, prevents them to behave as separate bodies.

2.3.1.2. Effect of Mass Ratio on the DVA System Bandwidth. The ratio of the mass of a DVA system and the primary structure mass is another significant parameter on the performance of the system. As the ratio increases, the range between the two resonant peaks around the antiresonance region widens. Thus the more the absorber mass is added the more effective frequency range can be attained. However, this ratio has no effect on the maximum available DVA performance, which is the case where the damping is zero and the $\omega = \omega_a$.

In order to illustrate the effects of the mass ratio, μ and the damping ratio ζ on the performance of the DVA system, the governing transmissibility (Equation 2.21), is evaluated for various combinations of the two parameters.

The resulting responses for systems having $\mu \in \{0.1, 0.3, 0.9\}$, $\omega_a = \omega_s$, $\zeta \in \{0, 0.01, 0.08, 0.16, 0.64\}$ are plotted in Figure 2.7. As can be seen in the figure, the mass ratio has a significant positive influence on the bandwidth of the DVA system. However, most of the time the increase in the absorber mass is limited by the design constraints.

There are many studies in the literature on the subject of finding an optimum DVA parameter configuration, in terms of damping and mass ratios, in order to meet

certain design constraints, such as sensitivity to mistuning, and requirements, such as maximum possible vibration attenuation [2, 3, 14, 34–38].

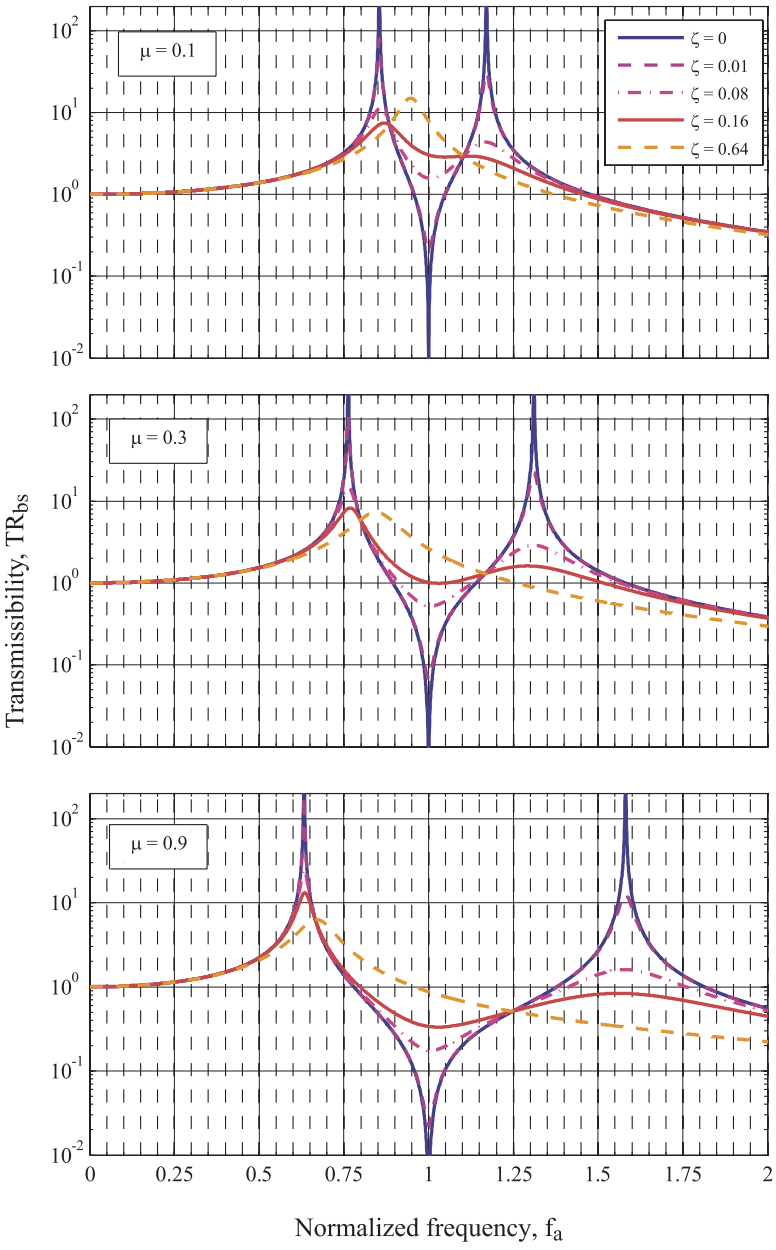


Figure 2.7. Effects of damping and mass ratio on the DVA characteristics.

2.3.2. Main Types of DVA Designs

Another quite important factor of a DVA system performance is the principle it is built, namely the system can be built as passive, adaptive, active or any combination of these [4].

2.3.2.1. Passive DVAs. The simplest examples of DVA systems are built as passive devices (see Figure 2.8). They are tuned to a certain excitation frequency during installation, and no remarkable variation in this frequency is expected. The attenuation bandwidth of such systems is solely determined by the damping and mass ratio of the DVA. Although, such a system is a very simple and cheap solution, it has a very limited usage in practical applications. In a case of variation in the frequency of excitation force to the system, there is always the possibility of coincidence with the resonance frequencies in the vicinity of the antiresonance frequency. And if this case happens, the DVA would cause more harm than it would help.

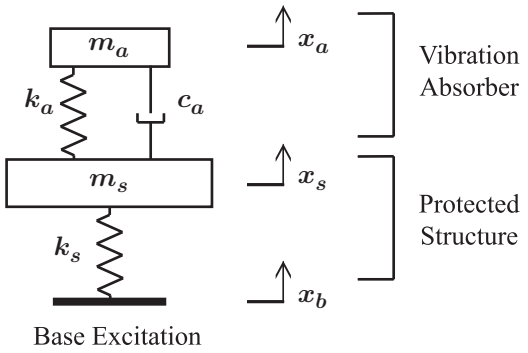


Figure 2.8. Schematic view of a SDOF system appended with a passive DVA.

2.3.2.2. Active DVAs. The most sophisticated dynamic vibration absorbers are active devices that directly counteract the excitation force on a structure. These systems can be built to cooperate with, and assist to a passive DVA system. (see Figure 2.9) However, at the uttermost situations these systems do not even need to incorporate any mass - spring system in their design. Basically, these devices employ vibration sensors,

a control system that decides the action according to the measurements and the model of the primary structure, and finally actuators that directly apply force against the excitation force.

Since these devices are not limited with the natural frequency and bandwidth issues, their performance can generally be superior to passive vibration absorbers. Additionally, since an actuator can apply forces composed of many frequencies rather than a single frequency as in the case of passive systems, they can react against complicated excitation forces. However, the most important drawback of such systems is that they use brute force to attenuate vibration on the system they are appended to. This means, they consume substantial amounts of energy to do their job. Most of the time, in practical applications, the feasibility of employing such systems generally becomes very questionable in terms of energy consumption and cost [39].

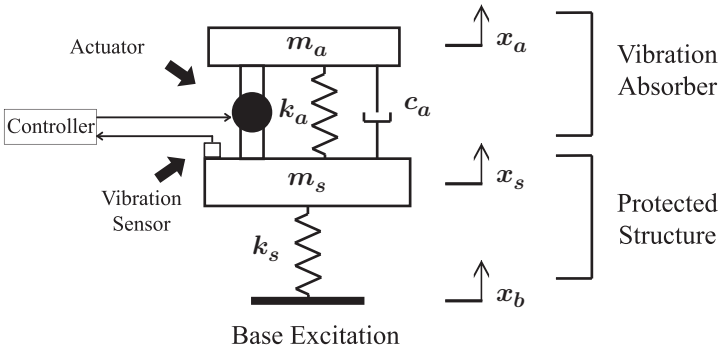


Figure 2.9. Schematic view of a SDOF system appended with an active DVA.

2.3.2.3. Adaptive-passive DVAs (APVA). Generally, adaptive-passive vibration absorbers differ from simple passive DVAs only in terms of online tuning ability. This type of designs employ vibration measurement equipment, an actuation system that can somehow alter the parameters of the system and a controller that runs a tuning algorithm according to the variations in the operation conditions. (see Figure 2.10) Thus, APVAs are basically self tuning passive vibration absorbers. The ability of tuning in real time dramatically reduces the limitations present in the passive DVAs. First of all, the condition for the maximum vibration attenuation performance, $\omega = \omega_a$, can

always be held true within the working range. This situation has two direct results; firstly, the damping value of the DVA can be reduced significantly and secondly, the mass of the DVA can be reduced. On a passive DVA, addition of a damper would be necessary to make it a reliable system, so that the possible detuning situations does not cause unwanted resonances. However, in an adaptive-passive DVA, the controller can shift the antiresonance point to an exact match with the excitation frequency; therefore the resonant frequencies can also be held away from the working conditions. Additionally, the designer can regain the potential attenuation performance sacrificed for the sake of a safe damped DVA. In addition to damping, mass of the DVA can also be reduced in an adaptive-passive system. Normally, for an acceptable DVA performance, as much DVA mass as possible needs to be employed so that the bandwidth of the system can be increased enough. However, considering that the damper of an APVA would be less than a passive DVA, the mass can also be reduced within similar performance requirements.

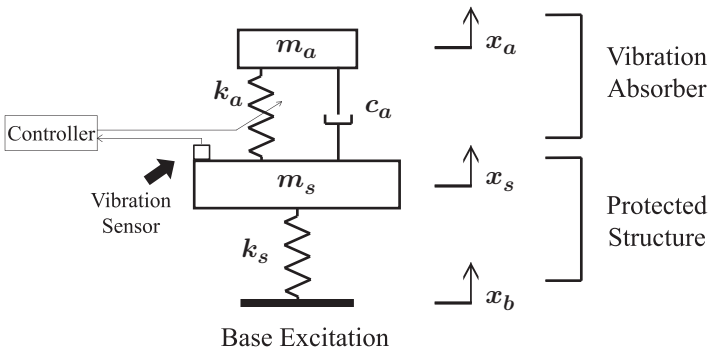


Figure 2.10. Schematic view of a SDOF system appended with an adaptive-passive DVA.

Adaptive-passive vibration absorbers have considerable advantages over passive DVAs. Similar vibration attenuation performances can be achieved using smaller and less damped designs, without the risk of causing unwanted resonance situations over a much larger bandwidth. One of the main drawbacks of Passive DVAs is, therefore, eliminated with this increase in the bandwidth. In general, design and construction processes of adaptive-passive DVA systems are relatively involved and costly compared

to the passive versions. However, adding the ability of self-tuning to a passive DVA, can sometimes enable designers to overcome the limitations imposed by passive devices in certain applications. Therefore, vibration protection problems that cannot be solved by using passive DVAs, can be handled by an adaptive-passive solution.

Active DVAs, on the other hand, are probably suitable in many vibration attenuation applications, if the energy consumption is not an issue. However, such situations are quite scarce in engineering applications, if they ever exist. Therefore, most of the time, utilizing an active DVA would be infeasible in terms of energy consumption.

In many cases, adaptive-passive DVAs can replace active solutions, without sacrificing much in terms of effectiveness but saving substantial amounts of energy.

2.4. Vibration of a String with Concentrated Mass Attachment

As stated previously, the present study introduces a new system to be used as a DVA. This system is composed of a metal string under tension with a central point-like mass attachment, (see Figure 2.11). A general DVA system incorporates a resilient member, an absorber mass and a damper element, if required by the design. All of these components together define the dynamical characteristics of the DVA system. In this present design, the string under tension is intended to be used as the resilient member of the DVA. And the mass at the midpoint of the string is used as the absorber mass. As explained in Section 2.3.1, addition of damping to a DVA system prevents it to reach the maximum possible absorption performance even at the perfect tuning conditions. Since the current design is an adaptive-passive vibration absorber, the possibility of mistuning due to variations in the excitation is not expected. Therefore, damping is not anticipated in the design. However, due to material damping present in the string, and due to the air drag acting on the vibrating components the final system will inevitably have some minor damping.

The string tension is used as a means of tuning the system. In other words, by varying the tension on the string, it is intended to vary the natural frequency of the

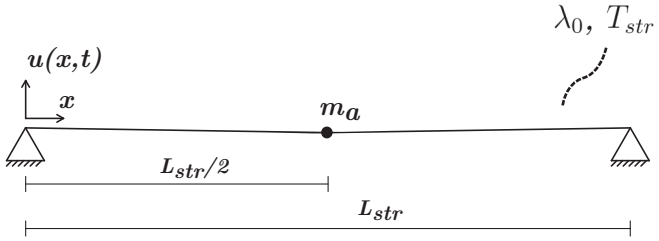


Figure 2.11. Schematic of the string-mass system.

DVA system. Thus, the dynamical properties of the system should be found for the range of attainable string tensions.

As shown in Section 2.3, the antiresonance region created by a DVA system on the combined system is primarily determined by its natural frequency, ω_a. Thus, in the design process, the first step is to calculate the natural frequencies and mode shapes of the system. The calculation is conducted using ABAQUS finite element package. The procedure followed in the construction of the model will be explained in detail in this section. In order to verify the validity of our numerical model, the analytical solution derived in [40] is adapted to our system. Since, in [40] the results are verified experimentally, the solution can be an accurate reference for the numerical model. It should be noted that, unlike a simple mass-spring system, the mass attached string is a continuous system. Although, continuous systems have infinite number of natural frequencies, they can be utilized as a dynamic vibration absorber, especially if the fundamental natural frequency of the system is a dominant mode and the higher order natural frequencies are not in the vicinity of the range of frequencies that the system operate [3].

After calculating the system's natural frequencies and mode shapes, another finite element model is constructed to simulate the string-mass system as a DVA in combination with a primary structure. Unlike the former model, this calculation is a relatively involved process. The dynamic interactions between the primary structure and the DVA system with varying excitation frequencies need to be determined. The

procedures followed in building this model and its results are also presented in the following subsections.

2.4.1. Analytical Calculation of Natural Frequencies of the String-Mass System

Although, solution of dynamical behavior of strings is considered among simple problems of vibration analysis, addition of a concentrated mass to the system requires some modifications in the solution procedure of vibration of homogeneous strings. The parameters of the system in Figure 2.11 are listed in Table 2.1.

Table 2.1. Parameters of the system in Figure 2.11.

m_a	Point mass attached on the midpoint of the homogeneous string
L_{str}	Length of the homogeneous string
T_{str}	Tension on the homogeneous string
λ_0	Linear mass density of the homogeneous string
$u(x,t)$	Transverse displacement of the string

We can model the mass attachment as a sudden jump in the linear mass density of the homogeneous string. Using this approach, one attains a linear mass density relation as

$$\lambda_{\text{str}}(x) = \lambda_0 + m_a \delta\left(x - \frac{L_{\text{str}}}{2}\right) \tag{2.22}$$

where $\delta(x)$ is the Dirac's delta function.

Equation of transverse free vibration of a non-homogeneous string is given by

$$\frac{\partial}{\partial x}\left(T_{\text{str}}\frac{\partial u}{\partial x}\right) = \lambda_{\text{str}}(x)\frac{\partial^2 u}{\partial t^2}. \tag{2.23}$$

If we plug Equation 2.22 into 2.23 and integrate it over $\left[\frac{L_{\text{str}}}{2}-\epsilon, \frac{L_{\text{str}}}{2}+\epsilon\right]$, then take the limit as $\epsilon \to 0$, we obtain the following condition

$$\left(T_{\text{str}}\frac{\partial u}{\partial x}\right)_{\frac{L_{\text{str}}}{2}-0}^{\frac{L_{\text{str}}}{2}+0} = m_{\text{a}}\frac{\partial^2 u(\frac{L_{\text{str}}}{2},t)}{\partial t^2}. \tag{2.24}$$

That represents a jump in the first derivate of the function $u(x,t)$ with respect to x at position where the point mass is attached.

For a solution, that incorporates a mass attachment, to be valid; continuity of displacements at the position of the mass must also be satisfied. This can be expressed as

$$u\left(\frac{L_{\text{str}}}{2}-0,t\right) = u\left(\frac{L_{\text{str}}}{2}+0,t\right). \tag{2.25}$$

Let us now find the solutions of wave equation corresponding to a homogeneous string,

$$\frac{\partial}{\partial x}\left(T_{\text{str}}\frac{\partial u}{\partial x}\right) = \lambda_0\frac{\partial^2 u}{\partial t^2} \tag{2.26}$$

satisfying both the boundary conditions

$$u(0,t) = u(L_{\text{str}},t) = 0 \tag{2.27}$$

and the extra conditions in Equations 2.24 and 2.25.

Assume that a solution to Equation 2.26 will be in the form

$$u(x,t) = \Phi(x)e^{i\omega t} \tag{2.28}$$

where ω is the angular frequency.

By using Equation 2.28 in 2.26, 2.27, 2.24 and 2.25, the following eigenvalue problem is obtained.

$$\frac{\mathrm{d}}{\mathrm{d}x}\left(T_{\mathrm{str}}\frac{\partial\Phi}{\partial x}\right)+\omega^2\lambda_0\Phi=0 \tag{2.29}$$

$$\begin{aligned}\Phi(0)&=0\\ \Phi(L_{\mathrm{str}})&=0\end{aligned} \tag{2.30}$$

$$\begin{aligned}\Phi\left(\frac{L_{\mathrm{str}}}{2}-0\right)&=\Phi\left(\frac{L_{\mathrm{str}}}{2}+0\right)\\ \left(T_{\mathrm{str}}\frac{\partial\Phi}{\partial x}\right)_{\frac{L_{\mathrm{str}}}{2}-0}^{\frac{L_{\mathrm{str}}}{2}+0}&=-m_{\mathrm{a}}\omega^2\Phi\left(\frac{L_{\mathrm{str}}}{2}\right)\end{aligned} \tag{2.31}$$

To solve the above eigenvalue problem, a final assumption is required. That is, the string tension, T_{str}, is considered as constant during the motion of the string. Although, the oscillations cause elongations on the string, if $\partial\Phi/\partial x \ll 1$ then the variations on the string tension caused by vibration can be neglected [41, 42]. In this case, the solution to the Equation 2.29 that satisfies the conditions in 2.30 becomes as following

$$\Phi(x)=\begin{cases}A\sin\left(\frac{\omega}{\sqrt{T_{\mathrm{str}}/\lambda_0}}x\right) & 0\le x\le\frac{L_{\mathrm{str}}}{2}\\ B\sin\left(\frac{\omega}{\sqrt{T_{\mathrm{str}}/\lambda_0}}(x-L_{\mathrm{str}})\right) & \frac{L_{\mathrm{str}}}{2}\le x\le L_{\mathrm{str}}\end{cases} \tag{2.32}$$

And the conditions in Equation 2.31 are satisfied if

$$\left[\sin\left(\frac{L_{\mathrm{str}}}{2}\frac{\omega\sqrt{\lambda_0}}{\sqrt{T_{\mathrm{str}}}}\right)\right]\left[m_{\mathrm{a}}\frac{\omega}{\sqrt{\lambda_0 T_{\mathrm{str}}}}\sin\left(\frac{L_{\mathrm{str}}}{2}\frac{\omega\sqrt{\lambda_0}}{\sqrt{T_{\mathrm{str}}}}\right)-2\cos\left(\frac{L_{\mathrm{str}}}{2}\frac{\omega\sqrt{\lambda_0}}{\sqrt{T_{\mathrm{str}}}}\right)\right]=0. \tag{2.33}$$

The frequency function for the string with concentrated mass attachment system is given in Equation 2.33. The two factors separated by square brackets in this function have different physical meanings. To investigate their behavior, first let us write the natural frequencies obtained from the first factor of the equation.

$$\omega_{\text{a}}^{2n} = \frac{2n\pi\sqrt{T_{\text{str}}}}{L_{\text{str}}\sqrt{\lambda_0}} \tag{2.34}$$

where $n \in \mathbb{Z}$.

Clearly, natural frequencies obtained above are identical to even-numbered natural frequencies of a homogeneous string (i.e. no mass attached). This fact can be explained both mathematically and physically. Firstly, the factor that leads to the above natural frequencies has, obviously, no dependency on the mass attached to the system. In addition to that, the mode shapes, $\Phi_{2n}(x)$, that correspond to the even-numbered natural frequencies of both homogeneous and the non-homogeneous i.e. mass-attached systems have the value zero at the midpoint of the string. In other words, these mode shapes have "nodes" at the mass attachment point, therefore the mass has no effect on the natural frequencies of these modes.

For these natural frequencies, it can be shown that the coefficients, A and B are equal to each other. Thus, the resulting set of mode shapes for even-numbered natural frequencies reduces to

$$\Phi_{2n}(x) = A\sin\left(\frac{2n\pi}{L_{\text{str}}}x\right) \quad 0 \leq x \leq L_{\text{str}} \tag{2.35}$$

The latter factor of the frequency Equation 2.33, however, depends on the mass attached on the midpoint of the string. Moreover, it cannot be solved directly like the case in the first set of natural frequencies. One can find the solutions of this factor by plotting it with respect to ω, the points where the curve crosses zero constitudes the second set of natural frequencies of the system, $\omega_{\text{a}}^{(2n-1)}$ where $n \in \mathbb{Z}$.

For this second set of natural frequencies, it can be shown that the coefficients of the mode shapes are related as $A = -B$. Thus, the resulting set of mode shapes for odd-numbered natural frequencies become

$$\Phi_{2n-1}(x) = \begin{cases} A \sin\left(\frac{\omega_a^{(2n-1)}}{\sqrt{T_{str}/\lambda_0}} x\right) & 0 \leq x \leq \frac{L_{str}}{2} \\ \\ -A \sin\left(\frac{\omega_a^{(2n-1)}}{\sqrt{T_{str}/\lambda_0}} (x - L_{str})\right) & \frac{L_{str}}{2} \leq x \leq L_{str} \end{cases} \tag{2.36}$$

2.4.2. Finite Element Method Calculation of Natural Frequencies of the String-Mass System

In this sub-section, the finite element model built with the intention of calculating the natural frequencies and mode shapes of the proposed DVA design is explained. The model is constructed using ABAQUS finite element package [43].

The accuracy of any finite element model depends on many factors, and they should be selected appropriately. Some of these factors that play role in the solution of our problem can be listed as:

- The model space used (2D or 3D)
- Element types used in the model
- Material properties
- The number of elements used in the model (i.e. mesh density)
- Modeling of boundary conditions
- The analysis method used in the solution (i.e. solver type)
- Types of the analysis steps
- The way the "mass attachment" is modeled

In order to examine the effects of these parameters on the results, different versions of models for the same problem are built. While some of these parameters are

altered in these different models, some of them are kept unchanged.

Let us discuss how these parameters are determined for the models. First of all, the modeling space is selected for the problem. Since the string-mass system under investigation is an axi-symmetrical structure, there is no need of using a 3D space for the model. The vibration characteristics of the system will be identical for both transverse and horizontal vibrations. Thus, in all versions of the model, a 2D modeling space is used.

Secondly, in terms of element types, ABAQUS provides a very rich library. For the type of analysis conducted in this problem, either solid (continuum) elements or structural 1D beam elements can be utilized. As one can consider a string as a very slender beam, the requirements of beam elements for an accurate modeling can easily be satisfied [43]. In addition to this, the complexity of a model built using beam elements is relatively lower than a model that uses continuum elements. In order to use beam elements in an analysis, which are, as stated previously, one dimensional elements; a profile for the cross-section of the beam needs to be defined. Thus, a circular profile for the string cross-section is defined when constructing the model. There are also many variations within the beam elements library, each having different formulations and analysis purposes. For the present problem, the three suitable element types are summarized in Table 2.2. All three of these elements are examined in models built for the problem.

Table 2.2. Suitable Beam Element Types for Modeling the String.

Element	Type	Interpolation Method	Modeling Space
B21	Timoshenko	linear	2D
B22	Timoshenko	quadratic	2D
B23	Euler-Bernoulli	cubic	2D

In ABAQUS, the user can define material properties that will be used in a certain

type of analysis. In our DVA design, a steel music wire will be utilized. And the parameters that need to be defined to solve the problem are: elastic modulus, Poisson's ratio and the density of the material.

Determining the number of elements along a model is a crucial decision for an analyst. Generally as the density of the mesh increases, the accuracy of the model also increases. However, this relationship between the mesh refinement and the numerical error present in the results is not linear. Rather, the results tend to converge to actual values with increased number of elements with a diminishing rate. Thus, further refinement in the mesh only brings more computational burden without any improvements on the results. In order to ensure that the mesh size for a certain model is actually enough for the required accuracy, a mesh convergence analysis should be conducted. This type of analysis can be done by comparing results between successive mesh densities until the change in the results becomes negligible. In this model, initially we used 100 elements along the string. After conducting analyzes for each model version, they are repeated by doubling the number of elements. The corresponding results of the same models with different number of elements are compared, in order to ensure that further increase in the number of elements does not affect the results significantly.

To complete the configuration of the string, the two ends of it should be properly constrained. The constraint on one end of the string is implemented as "pinned" connection to the model ground. On the other end of the string, the boundary condition only constrains the transverse displacement of it. The reason is that this axially unconstrained end is the application point of the string tension load.

The analysis type of ABAQUS used for the calculation of natural frequencies and mode shapes of the system is called "Natural frequency extraction". This method is a "linear perturbation procedure". Linear perturbation procedures conduct the analysis based on the last state of the configuration of the model. This basis configuration can either be the "initial step" of the analysis or the resultant state of any general analysis step, conducted prior to its call. And after the completion of linear perturbation type of analyzes, the state of the model is not altered. Thus, in the course of a multiple

step analysis, a natural frequency extraction procedure can be included anytime after a general analysis. This principle is used in our models to investigate the effect of tension on the natural frequencies of the system. A number of general static and natural frequency extraction analysis pairs are added. Each static analysis step of these pairs increases the string tension by a predetermined amount. After the tension is applied the resulting natural frequencies is calculated by the natural frequency extraction step after it.

To model the mass attached on the center of the string, "point mass" elements provided by ABAQUS are utilized. These point mass type of elements are generally attached on the selected nodes of the model. The insertion of a point mass to a node, alters the dynamic behavior of the system in the allowed degrees of freedom. During construction of our models, two different approaches are followed in terms of mass attachment modeling. Firstly, the mass is inserted only on a single node at the midpoint of the string. The intention of this approach is to obtain a model similar to the analytical solution, in which the addition of the mass is modeled mathematically by Dirac's delta function. By preserving the similarity between the numeric model and the analytical solution, a more accurate comparison between the two can be done.

Even though this comparison reveals that the finite element method can accurately predict the behavior of the system according to the analytical solution, a modification in the model in terms of mass attachment is required. In reality, the attached mass will occupy a certain volume, depending on the density of the material it is made of. Therefore, another approach is followed to reflect this fact into the finite element model. That is, rather than just inserting a point-mass in the middle of the model, the cross-sections of the beam elements at the location of the mass are enlarged. Therefore, a rigid zone of size of a realistic mass dimension at the center of the string is obtained. In addition to that, this new mass attachment approach also takes the rotational inertia of the attached mass into account, which can affect the higher vibration modes of the system.

2.4.3. A Case Study for the Comparison of Natural Frequency Calculation Methods

In this subsection, the analytical and finite element method models discussed in Sections 2.4.1 and 2.4.2 are evaluated for a realistic case study.

First of all, a proper choice for the string is required. Steel music wires with their high tensile strength are quite suitable for the system. There are numerous standard sizes of music wire diameters. For the time being, a wire of radius, $r_{\text{str}} = 0.1$ mm is chosen. For steel music wires, moduli of elasticity, E_{str}, generally range from 200 GPa to 210 GPa. In our calculations it is taken as, $E_{\text{str}} = 205$ GPa. And finally, the density of the wire is taken as, $\rho = 7.8$ g/cm^3.

Steel music wires can have ultimate tensile strengths up to $S_u = 2$ GPa [44]. Before determining the maximum allowed string tension, we should ensure that the system does not suffer from fatigue failure. There are several factors that affect the maximum allowable average stress in our load case. These can be summarized as:

- Surface roughness factor, C_{surf}
- Load type factor, C_{load}
- Reliability factor, C_{rel}

Since music wires are cold drawn, they have a very good surface finish. Thus a value as high as 0.8 can be taken for C_{surf}. Additionally, the load type applied to the string is mostly axial, thus we can take $C_{\text{load}} = 0.9$. Finally, for a reliability of 99%, we need to take $C_{\text{rel}} = 0.814$ [45]. Thus, the stress on the string for infinite life with

99% reliability becomes

$$S_n = \frac{S_u}{2} C_{\text{surf}} C_{\text{load}} C_{\text{rel}}$$

$$= \frac{2 \text{ GPa}}{2} (0.8)(0.9)(0.814)$$

$$\approx 580 \text{ MPa}. \tag{2.37}$$

Thus, the maximum allowable string tension becomes

$$T_{\text{str\_max}} = (580 \text{ MPa})(\pi(0.1 \text{ mm})^2)$$

$$\approx 18 \text{ N}. \tag{2.38}$$

The length of the string is chosen as, $L_{\text{str}} = 300$ mm. And the amount of attached mass, m_{a} is 90 g.

At first, the analytical model and FEM models with three different element types, using both 100 and 200 elements and only single node mass attachment approach, are evaluated, as shown in the Table 2.3. The 1^{st} and 2^{nd} natural frequencies and mode shapes are obtained and compared from these models.

Table 2.4 - 2.10 show results and comparisons obtained from the models summarized in Table 2.3.

As shown in the results, finite element models built with all three types of elements can accurately predict the first two natural frequencies of the string-mass system. The maximum error occurred is found to be less than one per cent. Moreover, differences between results of models built with 100 elements and corresponding refined versions are also quite low. Therefore, it can be concluded that the mesh used in the models is

Table 2.3. First Group of Natural Frequency Calculation Analyzes.

Analysis Type	Element Type	# of Elements	Mass Attachment
Analytical	Not Applicable	Not Applicable	Single Point
FEM	B21	100	
	B22		
	B23		
	B21	200	
	B22		
	B23		

Table 2.4. First and second natural frequencies, analytical solution.

T_{str} (N)	ω_{a}^1 (Hz)	ω_{a}^2 (Hz)
3	3.355	368.822
6	4.744	521.594
9	5.811	638.819
12	6.710	737.645
15	7.501	824.712
18	8.217	903.427

Table 2.5. First and second natural frequencies, B21 elements, single node mass attachment.

	ω_a^1 (Hz)		ω_a^2 (Hz)	
T_{str} (N)	100 Elements	200 Elements	100 Elements	200 Elements
3	3.379	3.379	369.017	369.064
6	4.765	4.767	521.503	521.567
9	5.830	5.831	638.466	638.546
12	6.724	6.727	737.015	737.094
15	7.514	7.517	823.770	823.881
18	8.227	8.230	902.154	902.265

Table 2.6. Error percentages present in FEM results of first and second natural frequency calculations, B21 elements, single node mass attachment.

	Error in ω_a^1 (%)		Error in ω_a^2 (%)	
T_{str} (N)	100 Elements	200 Elements	100 Elements	200 Elements
3	0.697	0.726	0.053	0.066
6	0.442	0.479	-0.017	-0.005
9	0.315	0.354	-0.055	-0.043
12	0.228	0.270	-0.085	-0.075
15	0.161	0.204	-0.114	-0.101
18	0.105	0.150	-0.141	-0.129

Table 2.7. First and second natural frequencies, B22 elements, single node mass attachment.

	ω_a^1 (Hz)		ω_a^2 (Hz)	
T_{str} (N)	100 Elements	200 Elements	100 Elements	200 Elements
3	3.379	3.379	369.080	369.080
6	4.767	4.767	521.598	521.598
9	5.831	5.831	638.577	638.577
12	6.727	6.727	737.126	737.126
15	7.518	7.518	823.913	823.913
18	8.231	8.231	902.313	902.313

Table 2.8. Error percentages present in FEM results of first and second natural frequency calculations, B22 elements, single node mass attachment.

	Error in ω_a^1 (%)		Error in ω_a^2 (%)	
T_{str} (N)	100 Elements	200 Elements	100 Elements	200 Elements
3	0.735	0.735	0.070	0.070
6	0.493	0.493	0.001	0.001
9	0.370	0.370	-0.038	-0.038
12	0.287	0.287	-0.070	-0.070
15	0.225	0.225	-0.097	-0.097
18	0.171	0.171	-0.123	-0.123

Table 2.9. First and second natural frequencies, B23 elements, single node mass attachment.

	ω_a^1 (Hz)		ω_a^2 (Hz)	
T_{str} (N)	100 Elements	200 Elements	100 Elements	200 Elements
3	3.380	3.380	369.303	369.303
6	4.768	4.768	521.758	521.758
9	5.833	5.833	638.705	638.705
12	6.731	6.731	737.238	737.238
15	7.520	7.520	824.009	824.009
18	8.233	8.233	902.409	902.409

Table 2.10. Error percentages present in FEM results of first and second natural frequency calculations, B23 elements, single node mass attachment.

	Error in ω_a^1 (%)		Error in ω_a^2 (%)	
T_{str} (N)	100 Elements	200 Elements	100 Elements	200 Elements
3	0.797	0.797	0.130	0.130
6	0.526	0.526	0.031	0.031
9	0.398	0.398	-0.018	-0.018
12	0.313	0.313	-0.055	-0.055
15	0.248	0.248	-0.085	-0.085
18	0.196	0.196	-0.113	-0.113

sufficient enough for accurate calculations.

Let us now compare first and second mode shapes obtained from analytical and finite element models. As shown in Figure 2.12 - 2.13, normalized mode shapes calculated using both methods are consistent with each other.

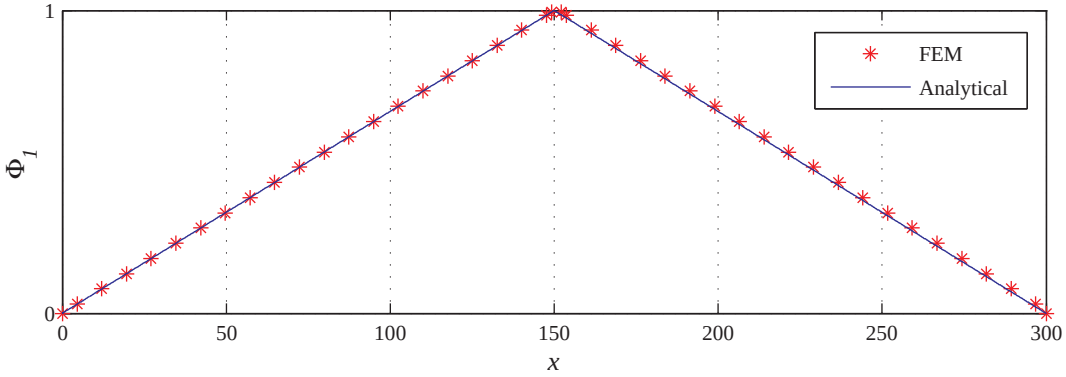


Figure 2.12. 1^{st} mode shape of the string-mass system obtained both analytically and using FEM.

The results of this set of FEM models show that they can predict the dynamic behavior of the string-mass system quite accurately. However, as stated previously, the methodology used in modeling the mass attachment needs to be improved for a more realistic simulation of the system. Both the analytical method and this first set of FEM models, follow a point mass approach for the attachment of the mass. But in reality, the dimensions of the mass attached to the system will affect its dynamical behavior.

In the second set of FEM models, rather than a single node, the attached mass is modeled with its realistic dimensions and rotary intertia. With this approach, a relatively rigid section of length equal to the length of the mass, L_m, is created. All other parameters used in the previous set of analyzes are kept unchanged. Only B21 type elements are utilized, for they showed closest results to the analytical model. Finally, L_m is taken as 60 mm. Note that, the existence of this rigid section in the

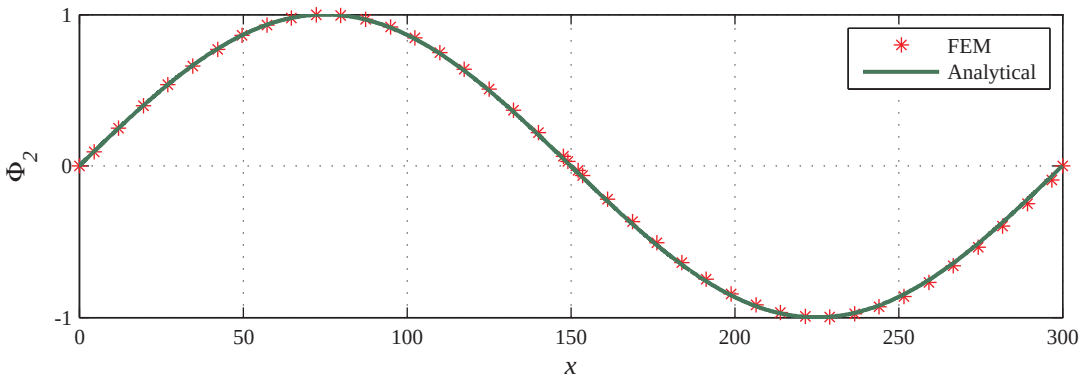


Figure 2.13. 2^{nd} mode shape of the string-mass system obtained both analytically and using FEM.

string model reduces the length of the string. So, in order to make a proper comparison, the analytical model is re-evaluated with a string length $L_{\text{str}} = 240$ mm. (see Table 2.11)

Table 2.11. Modified mass attachment approach, natural frequency calculation analyzes.

Analysis Type	Element Type	# of Elements	Remarks
Analytical	Not Applicable	Not Applicable	$L_{\text{str}} = 240$ mm
FEM	B21	200	Modified mass attachment

Table 2.12 and 2.13 show the first two natural frequencies obtained using analytical model and FEM, respectively. First of all, we can see that the first natural frequencies are again quite close to each other. This modified mass attachment approach caused very slight increase in the first natural frequency of the system, compared to the analytical solution. However, we see a remarkable difference between the second natural frequencies of the two different models. This is a direct result of the inclusion of the rotational inertia of the mass into the modified model. It is also clearly seen in

Table 2.12. First and second natural frequencies, analytical solution for $L_{\text{str}} = 240$ mm.

T_{str} (N)	ω_{a}^1 (Hz)	ω_{a}^2 (Hz)
3	3.751	461.024
6	5.305	651.994
9	6.497	798.528
12	7.503	922.048
15	8.387	1030.894
18	9.188	1129.284

Table 2.13. First and second natural frequencies, B21 elements, modified mass attachment.

T_{str} (N)	ω_{a}^1 (Hz)	ω_{a}^2 (Hz)
3	3.788	12.223
6	5.341	17.045
9	6.533	20.754
12	7.538	23.873
15	8.422	26.627
18	9.223	29.109

the second mode shape calculated by the modified model, that the mass attachment affects the second mode of vibration. As we saw before, the addition of a point mass, on the other hand, has no effect on the second mode.

The first two mode shapes obtained with the latter models are given in the Figure 2.14 and 2.15. It is clearly shown in these mode shapes how the attachment of the mass affects the vibration characteristics of the system.

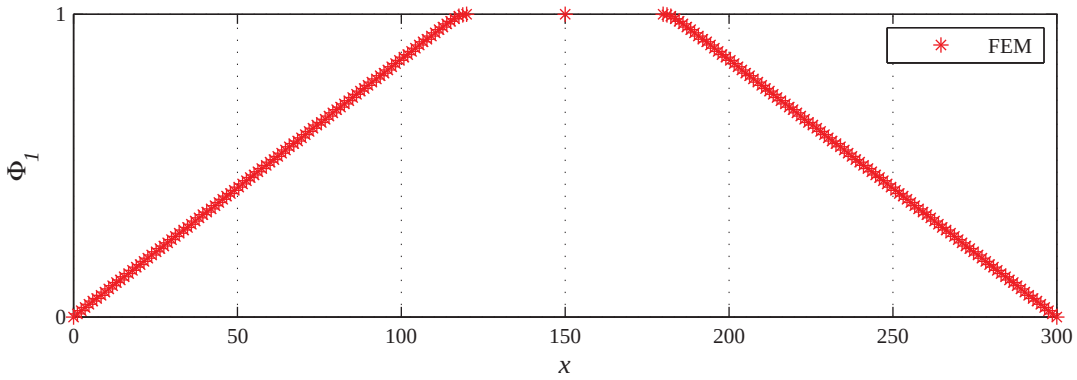


Figure 2.14. 1^{st} mode shape of the string-mass system obtained using modified mass attachment.

In conclusion, first we obtained consistent results with the analytical solution by building FEM models that use similar mass attachment approach. After verifying that using finite element method we can accurately predict the dynamic behavior of the string, a modified finite element model is constructed with the intention of obtaining a more realistic simulation of the actual design. First natural frequencies obtained using the latter approach are slightly higher than the initial models. However, it predicts dramatically reduced second natural frequencies, which is mainly caused by the rotational inertia of the mass model in this model version.

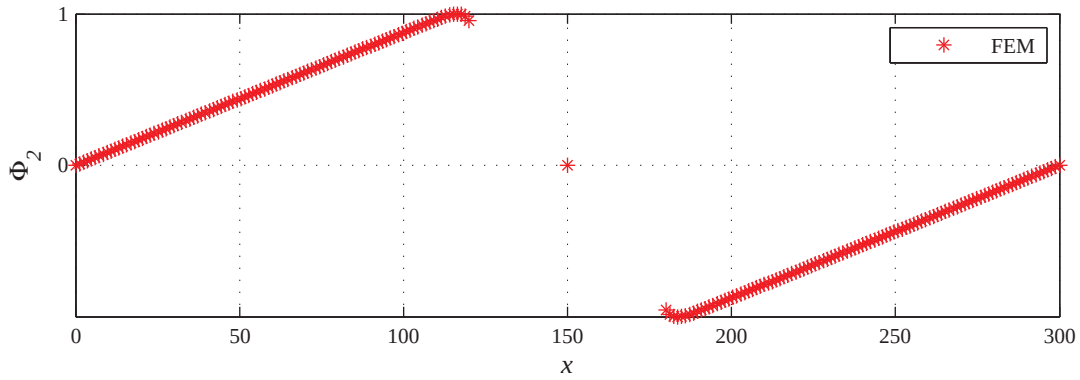


Figure 2.15. 2^{nd} mode shape of the string-mass system obtained using modified mass attachment.

2.4.4. Forced Vibration Analysis of the String-Mass System as a DVA via Finite Element Method

The analyses conducted in the previous subsections only include natural frequency and mode-shape calculations of the designed string-mass system DVA. However, in order to fully understand the performance of this system as a DVA, it should be analyzed in combination with a primary structure over the entire working range of frequencies. In this part of the text, the procedure followed to built a model for such analysis is explained.

In this calculation, only finite element method is utilized and the model is again constructed and solved using ABAQUS. In terms of solution steps, the main difference between this and the previous natural frequency analyses is the inclusion of a new analysis step, called "Steady State Dynamics". The principle of a steady state dynamics step is to apply kinematic base excitation to model and calculate the resulting response based on the natural modes previously obtained. During the calculation of this step, the specified range of frequencies is swept. As a consequence, the frequency response of the model is obtained at the end of the step. The steady state dynamics step

requires that a natural frequency extraction step is executed prior to its call, which is already done in our previous calculations. So, the order of analysis steps used in the natural frequency calculation models is modified with the addition of a steady state dynamics step. Thus, a number of analysis groups composed of a general static, a natural frequency extraction and a steady state dynamics steps are used to investigate the effect of string tension on the frequency response of the model.

In addition to analysis types, the model itself is modified in this calculation. In natural frequency detection analysis, the model was composed of only the string with attached mass. In this procedure, however, the string-mass system is appended to a single degree of freedom primary structure. This combined system is used to reveal the performance of the string-mass system as a dynamic vibration absorber.

Some minor alterations in terms of modeling the string-mass system is required in this analysis. Firstly, the way that the string tension is applied is changed. In the previous analyses, the tension is applied by pulling one of the string ends, which was axially unconstrained for this purpose. In ABAQUS, a force applied to a model is relative to the model ground, which means that the whole structure is pulled towards a fixed point. However, in this combined system, applying the string tension in this way can affect the dynamic behavior. In order to overcome this problem, an alternative method is used. A thermal expansion coefficient definition is added to the material properties of the string. Therefore, the string tension is varied by changing the ambient temperature in the model, while the both ends of the string is pinned to the primary structure. Secondly, in order to prevent numerical errors in the vicinity of resonance conditions, a small material damping coefficient definition is added to the string material definition. To model the attachment of the mass, the second approach discussed in the previous subsection is used.

As a primary structure, we used a single DOF point mass in the model. The mass of this structure is chosen to be as $m_s = 2000$ g, and the stiffness of the spring

between the structure and the base is $k_s = 1.8$ N/mm. Which means that

$$\begin{aligned} \omega_s &= \sqrt{\frac{k_s}{m_s}} \\ &= 4.775 \text{ Hz.} \end{aligned} \tag{2.39}$$

Thus, the natural frequency of the primary structure is within the working range of our DVA (see Table 2.12).

As shown in Figure 2.16, the variations in the string tension directly affects the resulting transmissibility plots of the system.

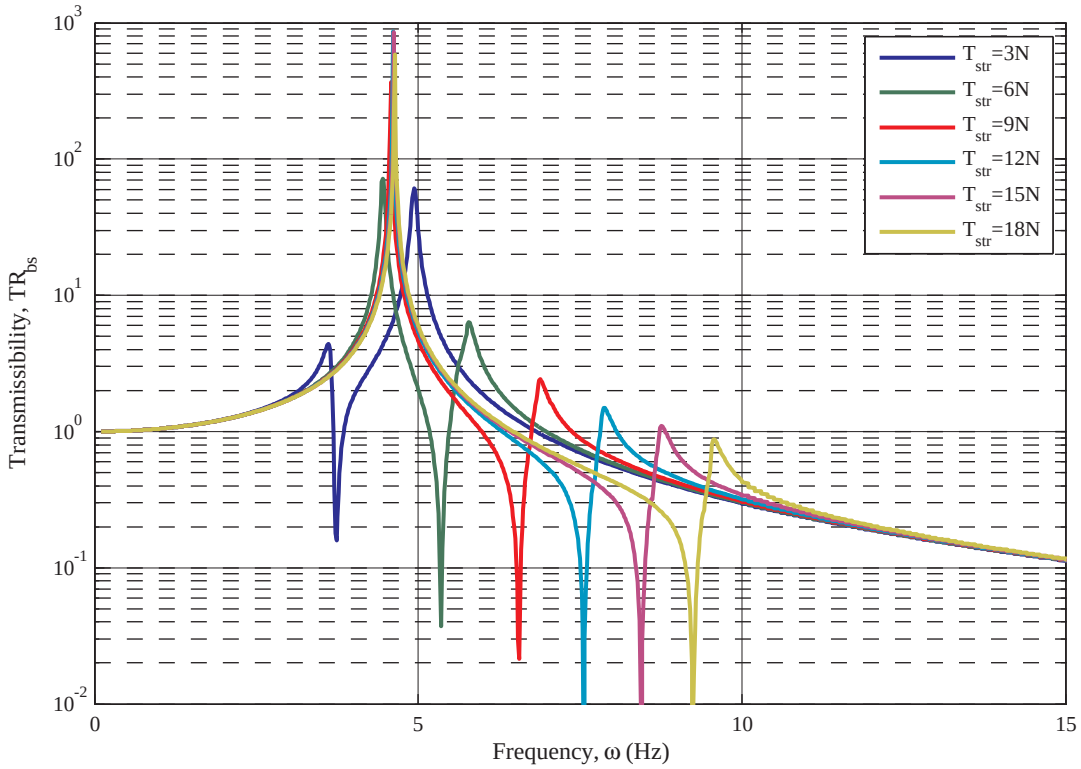


Figure 2.16. Transmissibility of the combined system for various T_{str} values.

2.5. Negative Stiffness Mechanism to Aid String Tension Adjustment

As shown previously, the dynamical characteristics of the string-mass DVA system can be adjusted using the string tension. This method has a key advantage: very small elongations applied to the string will cause large variations in the string tension. Compared to adaptive-passive DVA tuning strategies mentioned in [9] or in [17], for instance, the response time to the tuning commands given by the controller can be significantly reduced.

Although, the ability of varying the tuning parameter with small displacement inputs becomes an advantage in terms of tuning response, a drawback in terms of actuator effort arises in this design. That is, directly connecting an actuator to the string end to alter the tension means that it must deliver all the energy required to increase the strain energy stored in the string or to hold it at a certain value. Since, the load on the actuator will directly be related to the string tension, the operating conditions of the actuator will also vary as the string is tensioned. Therefore, the dynamical characteristics of the actuator and its response to the controller requests will not be the same throughout the range. Moreover, if the position control of the actuator is done in open-loop, like in the case of most stepper-motor actuators, the positioning accuracy will also be affected from these load variations. Considering these problems, an actuation mechanism that is based on directly stretching the string is not a suitable choice for the system.

The simplest solution that can be implemented to overcome this actuator effort problem is the usage of mechanical reduction. Therefore, with actuators of low force (or torque if rotary actuators are used) ratings, the desired string tension adjustment task can be accomplished. However, this approach will solve the problem with the price of loosing sensitivity. When reduction is utilized to ease the actuator‘s job, displacement (or rotations) inputs required to alter the tension will inevitably increase.

Another possible solution for the string tension adjustment with low actuator effort and high sensitivity can be obtained by using a negative stiffness spring in com-

bination with the string. Let us begin the description of this proposed method by explaining the force-deflection characteristics of the string used in the DVA system and a generic negative stiffness spring.

As we stated previously, a steel music wire is to be used in the DVA system as a string. Linearly elastic materials, like this wire, have linear force-deflection curves with a positive slope, until they reach to the material yielding point. Thus we can treat the string as having a constant spring coefficient, k_{str}. This behavior is illustrated in Figure 2.17a.

Mechanisms showing negative stiffness behavior generally have at least one unstable equilibrium point. [24] As the negative stiffness spring is deflected from this equilibrium, it tends to increase its deflection with increasing force. Thus, the slope of its force-deflection curve, $k_{ns}(x)$ has a negative value, and it is not necessarily constant; rather its characteristics depend on how this negative stiffness system is implemented and the parameters used in this implementation. Before continuing to the methods for creating negative stiffness springs, let us illustrate the above mentioned behavior of negative stiffness systems graphically. Note that there is no conventional method for representing negative stiffness system schematics using a single symbol. However, for the sake of simplicity in this introductory illustration, red colored spring symbols are used to represent such systems in Figure 2.17b.

As seen in Figure 2.17, the force-deflection behaviors of these two different kinds of systems are opposite to each other. While positive-stiffness spring (in this case the metal wire) resists to further deflection, the negative stiffness element tries to further deflect from its unstable equilibrium point.

Combining these two types of spring elements in parallel, results in a total stiffness of $k_t(x) = k_{\text{str}} + k_{ns}(x)$. Considering that the sign of the k_{ns} is negative, the stiffness of the combined system will be reduced by its magnitude. So, one can reduce the overall stiffness of the system down to zero where the magnitude of the negative stiffness term is equal to the positive-stiffness value. Moreover, if these springs were connected

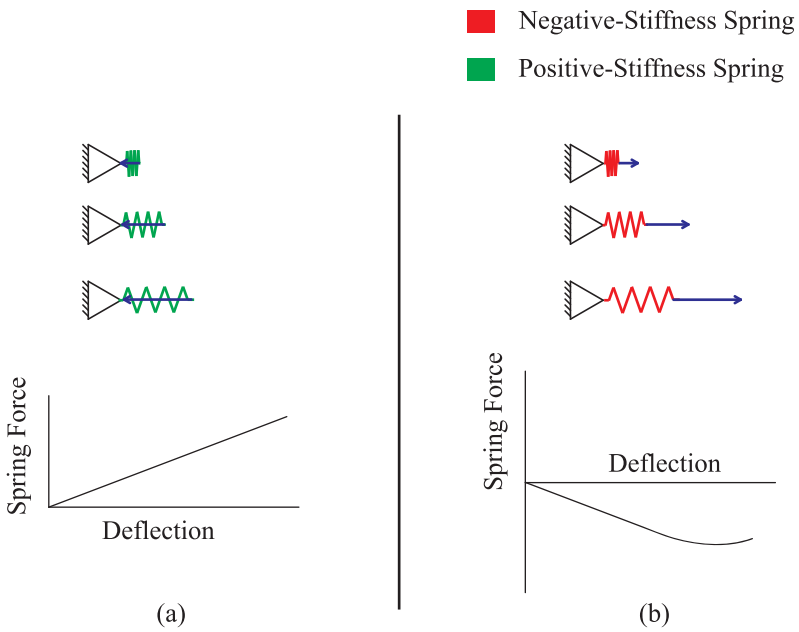


Figure 2.17. Schematic representation of force-displacement behavior of (a) positive and (b) negative springs.

to each other at positions other than their non-deflected states, the stiffness of the combined system would again be $k_t(x) = k_{\text{str}} + k_{ns}(x)$. However, the force-deflection curve of the combined system would be shifted by the amount of force caused by the initial deflection. For example, say a positive-stiffness spring of coefficient, k, vertically carries an object of mass, m. Then the force on this spring becomes, mg, where g: acceleration due to the gravity. If a negative stiffness mechanism having a stiffness about $-k$ around its unstable equilibrium point is attached parallel to this positive-stiffness spring, the combined system would have almost zero stiffness while it can still carry the mass, m. As briefly mentioned in Section 1.2, quasi-zero-stiffness (QZS) mechanisms used in vibration isolation applications or constant-force mechanisms used in a variety of areas are both based on the same principle of operation, contrary to the difference in terminology.

In this study, a special case of this principle is used for a different purpose. The string and a negative stiffness mechanism are combined to give a resultant zero-stiffness

system at constant zero force over the range of possible string tensions. Thus, the linear position actuator, used for stretching of the metal wire in the DVA system, is assisted by a negative stiffness spring element.

One can also explain the behavior of such a combined system in terms of potential energies stored in the springs. When the metal wire is at rest, the potential spring energy stored in it is zero. On the other hand, a negative stiffness element at its unstable equilibrium point has its maximum potential energy value. When they are connected to each other at this state, any deflection applied to the combined system will decrease the potential energy of the negative stiffness structure, while increasing the potential energy of the wire. Thus, all the work an actuator controlling the position of this combination has to do is to supply energy to compensate the hysteretic losses occurring on the combined system.

There are numerous ways of creating negative stiffness springs reported in the literature. In order to obtain a suitable negative stiffness solution to be implemented in the above mentioned string tension adjustment mechanism, two different methods of realizing negative stiffness mechanisms are investigated.

The first option is a buckled shape leaf spring. Force - deflection relations of such systems can show negative stiffness behavior [31]. However, there are several disadvantages of this method. Firstly, force - deflection behavior of systems built with this method are very non-linear and affected by too many parameters. Secondly, production of such a system accurately is hard to achieve. Therefore, there is the possibility of mismatch between the theory and practice. Finally, fine adjustments in the stiffness characteristics of such systems are limited.

Secondly, a system composed of a rigid link and a pre-compressed spring is investigated with the intention of creating a negative stiffness mechanism. The analysis of this system is relatively less complicated compared to the buckled beam system. By properly adjusting parameters of this system, one can build an exact negative of any positive-stiffness spring.

2.5.1. Negative stiffness mechanism using rigid link and pre-compressed spring

Using a mechanism built with a pre-compressed spring and a rigid link, as shown in Figure 2.18, one can obtain a system that shows negative stiffness behavior over a range of deflections.

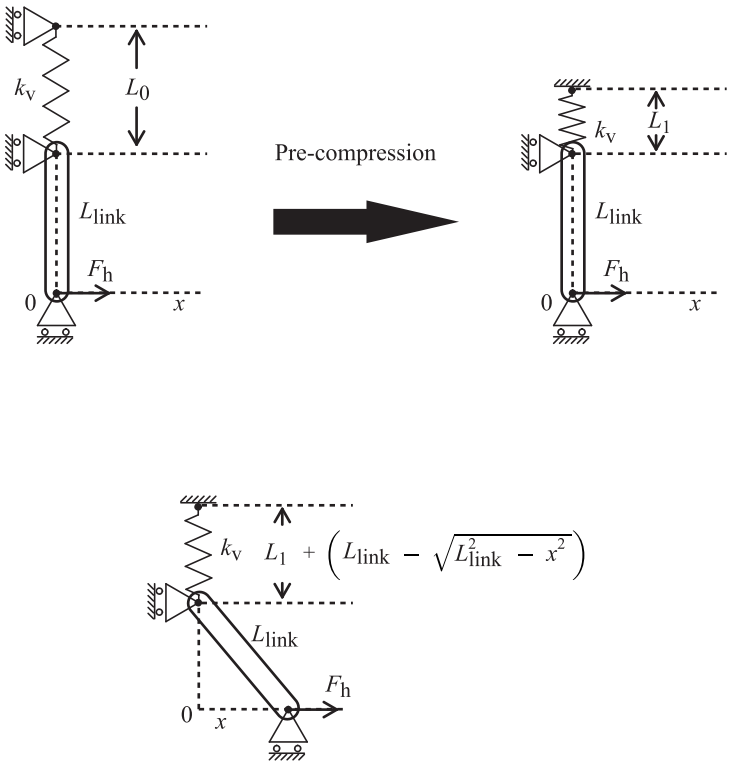


Figure 2.18. Negative stiffness mechanism built with a pre-compressed spring and a rigid link.

Firstly, we write the relationship between the potential energy of the system in terms of the parameters listed in Table 2.14.

$$\Pi(x) = \frac{1}{2}k_v\left(\delta_0 - \left(L_{link} - \sqrt{L_{link}^2 - x^2}\right)^2\right). \tag{2.40}$$

If we differentiate Equation 2.40 with respect to x, we can obtain the equation

Table 2.14. Parameters of the rigid link - pre-compressed spring mechanism

L_0	Undeformed length of the vertical spring
L_1	Length of the vertical spring at maximum compression
δ_0	$L_0 - L_1$, precompression of the spring
k_v	Stiffness of the vertical spring
L_{link}	Length of the rigid link
x	Horizontal deflection coordinate of the combined mechanism
$F_h(x)$	Horizontal spring force of the mechanism
$\Pi(x)$	Total potential energy of the system

for the F_h as a function of x:

$$\begin{aligned} F_h(x) &= \frac{\mathrm{d}}{\mathrm{d}x}\Pi(x) \\ &= k_{\text{v}} x \left(\frac{\delta_0 - L_{\text{link}}}{\sqrt{L_{\text{link}}^2 - x^2}} - 1 \right) \end{aligned} \tag{2.41}$$

In Equation 2.41, we can see that if the pre-compression of the vertical spring, δ_0, is equal to the length of the rigid link, L_{link}, the force-deflection equation of the system reduces to

$$F_h(x) = -k_{\text{v}} x \tag{2.42}$$

which means, the stiffness of the resulting system has a constant negative value equal to the magnitude of the vertical spring stiffness, k_{v}.

2.6. Self-Tuning System for the DVA

Until now, every aspect regarding the mechanical foundations of the proposed adaptive-passive DVA is explained. Without implementing a control system that tunes the DVA according to the operation conditions, one can only say that the DVA is "tunable". In this section, the design of the control system for tuning the DVA is explained.

To begin with, let us discuss about the nature of vibration absorbers, from a control theory perspective. In [46] it is shown that vibration absorbers can be treated as mechanical controllers. Let us illustrate this using a very basic control loop; in Figure 2.19.

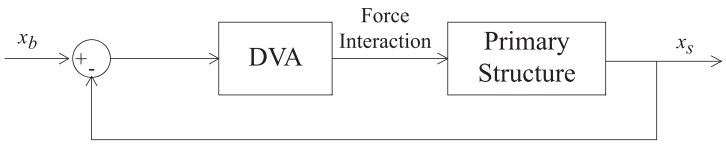


Figure 2.19. Control loop representation of a DVA system.

For SDOF structures appended with SDOF DVAs, we have already derived the closed loop transfer function of this system, in Equation 2.18. As discussed previously, this closed-loop system has two resonance frequencies at which it becomes unstable and also an antiresonance frequency at which the disturbance, x_{b}, is attenuated.

Unlike passive DVAs, active and adaptive passive DVA systems are based on varying their dynamic behavior in real-time, in order to attenuate disturbances of different frequencies. In order to do that, various approaches are studied in the literature.

Let us first briefly discuss the principle of controllers used in active DVAs. Dynamics of systems appended with active DVAs are directly governed by the actions of their controller system. Thus, the form of the control loop of such systems becomes generally as in Figure 2.20.

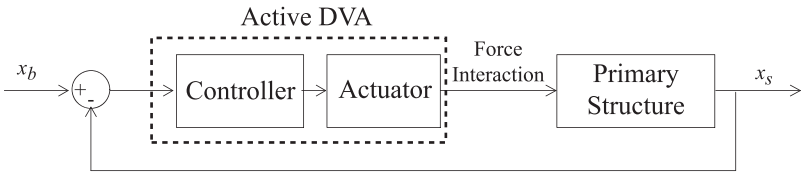


Figure 2.20. Control loop representation of an active DVA system.

Synthesis of such controllers is widely discussed in the literature. Some designers used simple PID systems, while some others use fuzzy logic algorithms and also there are controllers built using H_2 or H_∞ optimizations [47, 48]. On the other hand, controllers used in adaptive-passive DVAs have a different principle. In these systems, the dynamics of the combined system is governed by the passive vibration absorber. And the task of the controller implemented in these systems is only to tune the passive DVA according to the information acquired from the system. This principle is illustrated in Figure 2.21. It is worth noting the functional differences present between the active and adaptive passive ones.

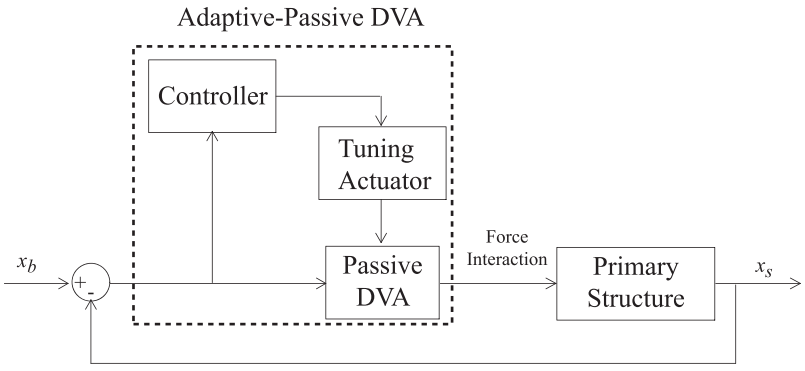


Figure 2.21. Control loop representation of an adaptive-passive DVA system.

Moreover, if the dynamics of the passive DVA as a function of the tuning parameter is previously known, the only duty of the controller of the adaptive-passive system is to adjust the system‘s parameters so that the antiresonance frequency is shifted towards the frequency of excitation. In most of the studies present in the literature,

controllers are developed using this principle. They mainly measure the excitation applied to the system, and analyze its frequency content or phase difference between the motions of the excitation and the absorber mass, and using this information, they adjust the parameters of the passive DVA system [49].

A similar approach is followed in the development of the tuning controller for our adaptive-passive DVA system. Since one of the major goals of this study is to create a flexible solution for various vibration protection systems, an embedded controller system is designed, rather than using a nearby installed computer. Before going into detail, let us summarize the main components of the system.

- Accelerometer for measuring the vibration of the system
- A digital signal processing microcontroller for signal processing, tuning algorithm execution and actuator control
- A linear actuator and its driver for adjusting the string tension

2.6.1. Accelerometer Selection

In an adaptive system, the controller should be able to monitor the response of the system constantly. For the task of vibration measurement, there are many options that can be used such as, linear motion transducers, piezoelectric materials, microphones, inductive or capacitive proximity sensors, optical measurement devices and last but not the least accelerometers.

In this project, we chose an accelerometer for measuring the vibration of the primary structure. An accelerometer is generally a MEMS device that is sensitive to accelerations in its measurement direction. Though, there are many commercially available accelerometers that can measure in three axes. By using such an accelerometer one can measure entire translational motion of an object.

Most of the time, the output of an accelerometer is an analog signal proportional to the acceleration input. However, there are also accelerometers that are combined

with analog to digital conversion and signal conditioning circuitries. And the output of such accelerometers is digital, which can be read using an appropriate communication interface by a computer or an embedded system. One of the main advantages of using digital accelerometers is that, implementing the analog measurement system inside the chip can significantly increase the noise immunity. Because, no matter how an analog output accelerometer is accurate, if the analog measurement system is not designed properly, the overall accuracy and noise immunity of such a system can be dramatically affected.

In addition to single axis or multiple axes and the analog or digital decisions, there are also several parameters that define the performance characteristics of an accelerometer, such as its noise floor, its measurement bandwidth, and maximum level of vibration that can be measured with it. The noise floor is the magnitude of the "white-noise" components present in the output signal of the sensor. It should be as low as possible, depending on the requirements of the applications. The measurement bandwidth of an accelerometer is the range of frequencies that can be accurately sensed by the device. While selecting an accelerometer, the bandwidth required is determined by the application conditions [50].

The above mentioned characteristics of an accelerometer also determine its cost. Thus, the designer should define the required performance for a certain application and make an appropriate decision for the sensor.

In this project, we had chosen a certain triple-axis, digital output accelerometer with suitable noise floor and bandwidth values for our project. However, due to the mistake of the online dealer, we acquired a relatively more expensive, multiple-sensor motion measurement board, called "9 Degrees of Freedom - Razor IMU", at the same price (see Figure 2.22).

This board is composed of a digital output triple-axis accelerometer, *Analog Devices - ADXL345*, a digital output triple-axes gyroscope and a digital output triple-axis magnetometer and a low end microcontroller *ATMEL - ATMega328P* that takes the

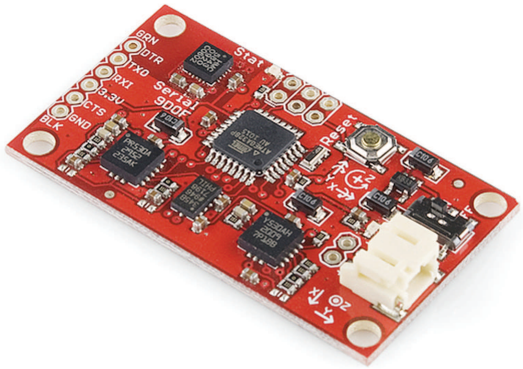


Figure 2.22. Photo of the "9 Degrees of Freedom - Razor IMU" sensor board.

measurements and sends it over an asynchronous serial communications interface.

The properties of the accelerometer implemented in this measurement board was similar to the selection we had, thus by modifying the firmware present in the low-end microcontroller of this device, we managed to use it in our vibration measurement system. For schematic circuit drawing of this measurement board and partial data sheets of the accelerometer and the microcontroller implemented in it, refer to the Appendix A.

2.6.1.1. Modified firmware of the vibration measurement board microcontroller. The block diagram of the system of the "Razor IMU" board is shown in Figure 2.23.

Each three sensors present in the board can communicate with the microcontroller in the board over a synchronous communication protocol called I^2C. In [51], the details of this protocol are given. Using this protocol, the microcontroller can send on-off commands and initial settings to each sensor and also it can retrieve measurement results from them. Additionally, after taking measurements, the microcontroller sends the results using an UART (Universal Asynchronous Receiver/Transmitter) protocol. Therefore, the measurements taken from each sensor can be read using a computer or another microcontroller to process and execute further calculations using them.

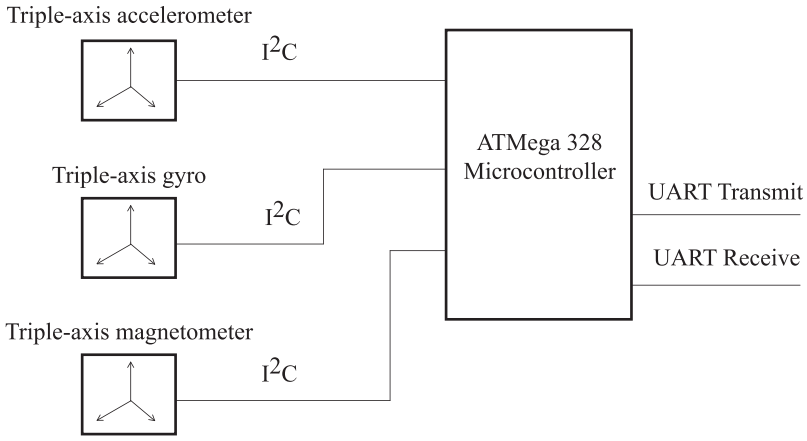


Figure 2.23. Hardware block diagram of the 9 Degrees of Freedom - Razor IMU measurement board.

In order to setup this measurement board according to our control system's requirements, several modifications are done in the firmware of the board. First of all, the other two sensors are shut down, for the measurements of them are not needed in our system. Additionally, the sampling rate and settings for the maximum magnitude of vibration to be measured is sent to the accelerometer. Most important modification done in the firmware is, however, not these settings. The communication protocol used in the original firmware is altered.

The UART protocol is used in the original firmware is an asynchronous protocol. Although, managing this type of connection is very easy, it has several considerable disadvantages. Firstly, the systems communicating over this protocol must have identical timing settings (BAUD RATE) in order to send and receive information. Any slight variation in these timings can result in missing several bits of information, and thus the entire message can be corrupted. In order to eliminate such communication problems, tedious software control routines are required. Secondly, due to strict timing requirements, achievable communication speeds over this protocol are much less than synchronous protocols.

In our project, the communication protocol of the vibration measurement is switched to a synchronous protocol, called SPI. One of the communicating devices

over this protocol is called the master device, and others (if present more than one) are called slaves. In this structure, bits to be sent (either from master to slaves or from a slave to the master) are processed at the instant the master generates a clock pulse. Therefore, the speed of communication is directly determined by the master's clock rate. Moreover, after each byte is sent over SPI, the master has the opportunity to reset the line and, thus, can ensure that no corruption in messages due to erroneous bits occurs. Using this protocol, we achieved very high communication speeds with much higher reliability, compared to the UART system.

With this new firmware (see Appendix B), we managed to measure vibrations using the *Razor IMU* board and to send the results to the signal processing and DVA tuning microcontroller. The final hardware and software diagrams of the vibration measurement part of the whole system are shown in Figure 2.24 and Figure 2.25, respectively.

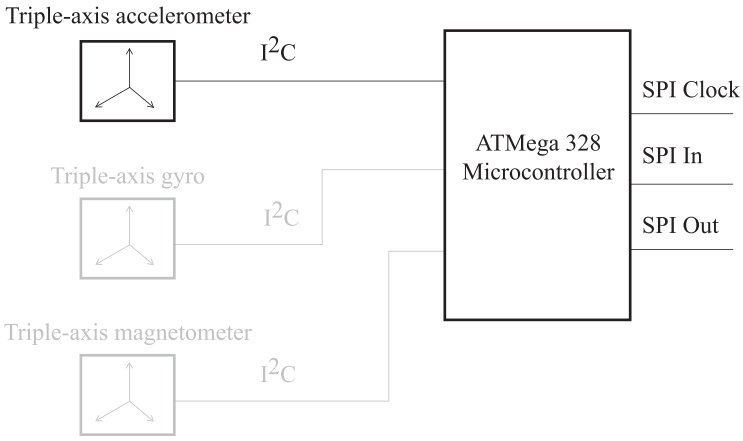


Figure 2.24. Modified hardware block diagram of the 9 Degrees of Freedom - Razor IMU measurement board.

2.6.2. Tuning Microcontroller Selection

In this subsection, the factors taken into account for the selection of the microcontroller that retrieve vibration measurements from the accelerometer, process the

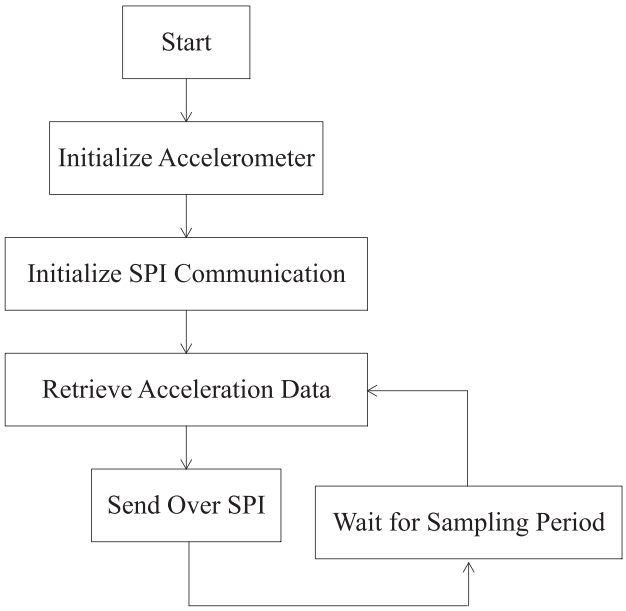


Figure 2.25. Modified software block diagram of the 9 Degrees of Freedom - Razor IMU measurement board.

measured signal, execute the tuning algorithm and finally command the tuning actuator, are discussed.

First of all, it should be stated that if all these operations were to be handled by a computer, there were be much less limitations in terms of complexity and performance. However, since modularity is among our major goals, an embedded platform is preferred.

Modern microcontroller producers offer a wide variety of products in terms of peripheral interaction, computation speeds and capabilities and memory sizes. So, the requirements for this implementation should be determined.

Like any computer system, each task in a microcontroller code is constructed using simple commands called instructions. The usage of these instructions is most of the time determined by compilers. Generally, a more sophisticated microcontroller, or in fact any computer, has wider instruction sets capable of accomplishing complicated

tasks. In order to complete a certain type of operation using fewer instructions, the hardware of that microcontroller must be designed with that capability. Therefore, time-consuming computations due to tedious usage of many different instructions can be completed much faster if certain types of operations are implemented in hardware [52].

In the selection of the microcontroller for our system, the major concern was the above mentioned principle. Since we need to process the vibration signal, and compute the appropriate response in real time, a microcontroller capable of doing mathematical operations fast enough is required. In addition to this, a microcontroller that can communicate over SPI protocol is required. As explained in the Section 2.6.1, the vibration data is sent over SPI protocol. In terms of actuator control, the communication with the motor driver only involves a very simple digital message system which can be handled by most of the microcontroller system. Therefore, this part of the system is not a crucial concern is the selection. Finally, the project does not involve any analog measurement operations, thus it is also not much of a concern in the decision.

By considering all of the above factors, we chose a microcontroller from the dsPIC30F family of Microchip Company, namely dsPIC30F4012. The partial datasheet of this chip can be found in Appendix C. The abbreviation of this microcontroller family, dsPIC is a mixture of "Digital Signal Processing (DSP)" and "Peripheral Interface Contoller (PIC)", a popular family of microcontroller of the same producer.

The advantage of using this type of microcontrollers is that they are equipped with a sophisticated infrastructure that can handle many standard mathematical operations in hardware. For example, consider the calculation of the dot product of two integer vectors.

$$\sum_{i=1}^{n} a_i b_i \tag{2.43}$$

In an ordinary microcontroller, the above operation could be handled by a series of instructions as shown in Figure 2.26.

for $i = 1$ to n **do**
  $a_i \Rightarrow$ Working Register 1
  $b_i \Rightarrow$ Working Register 2
  Working Register 1 $*$ Working Register 2 $\Rightarrow$ Working Register 3
  Accumulation Register $\Rightarrow$ Working Register 4
  Working Register 3 + Working Register 4 $\Rightarrow$ Working Register 4
  Working Register 4 $\Rightarrow$ Accumulation Register
end for

Figure 2.26. Instructions of Dot Product Calculation in an Ordinary Microcontroller.

However, using the DSP engine of the dsPIC microcontrollers, all of the operations above can be completed by executing only a single "multiply and accumulate" instruction n times.

Similar to the above example, the instruction set of the dsPIC family microcontrollers include several other instructions that significantly speed up many digital signal processing operations.

Moreover, the producer of this microcontroller supplies a wide range of sample codes. For instance, an example Fast Fourier Transform (FFT) code, specifically optimized for the dsPIC30F family is implemented in our tuning system.

2.6.3. Linear Position Actuator

In this thesis, the design requires a linear position actuator to vary the string tension. The travel range of the string end point is less than 5 mm in the system. Moreover, the natural frequency of the DVA system is quite sensitive to variations in the string tension. Therefore, the selected actuator system needs to be able to

apply displacements with high accuracy, good resolution as fast as possible. On the other hand, with the help of the designed negative stiffness mechanism, the power output capacity of the actuator system becomes a lesser problem compared to the above requirements. By considering these factors along with the costs of the possible solutions, a stepper motor in combination with a precision lead screw is chosen.

The motor selected for this design is a *MINEBEA 15PM-M*. It has 200 Nmm holding torque output and has 200 natural step positions in a single revolution [53]. By using a technique called microstepping, we were able to divide a single revolution of the motor into 3200 equal microsteps. Therefore, the position resolution is dramatically increased and also the choppy operation of the motor is smoothened. Although, microstepping can affect the positional accuracy of the motor, due to the fact that stepper motors do not accumulate positioning errors over consecutive steps, this does not cause severe problems in the system [54].

2.6.3.1. Microstepping Driver. Driving stepper motors is generally a straightforward operation. They mostly consist of two separate windings. The amount and direction of current flowing across these coils determine the position of the rotor. If these coils are actuated either at zero or full current, the motor aligns itself with its natural step positions. This is the most basic method of driving stepper motors.

If, however, one can control the magnitude of currents in each coil, the motor can be positioned between natural steps. By varying the coil currents in a sinusoidal fashion, one can obtain equally spaced microstep positions [55]. Obtaining microsteps using current control is illustrated in Figure 2.27.

In this thesis, a microstepping driver chip, Allegro A3972 is utilized [56]. It can control the currents on each coil with increments of 1.56% of the maximum current level. This desired value of current is determined by the command messages sent from the tuning microcontroller. Therefore, using predetermined levels of currents, a sinusoidal current variation in each coil can be attained.

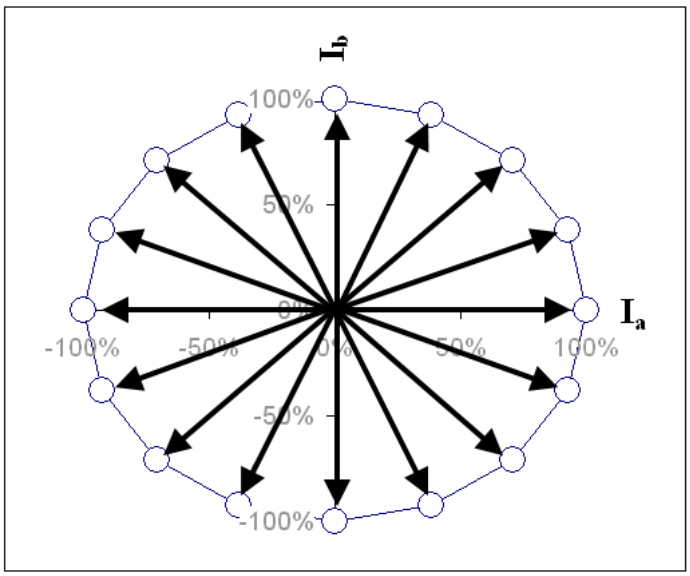


Figure 2.27. Microstepping using sinusoidal control of two motor coil currents, I_a and I_b.

2.6.4. Development of the Signal Processing and Tuning Algorithm

The principle of a tuning system of an adaptive-passive vibration absorber is based on tracking the frequency of excitation and adjusting the parameters of the DVA to match its natural frequency with the measurements.

In order to achieve an efficient program flow, the tuning algorithm has to be able to accomplish the following tasks:

- Acquisition of the vibration data from the accelerometer in time, without losing any information
- Calculation of the Root-Mean-Square (RMS) of the vibration signal for amplitude detection
- Identifying the dominant frequency of the vibration signal by calculating the Fast-Fourier-Transform of it
- Using the natural frequency calibration data of the string-mass system as a func-

tion of the actuator position, calculation of the required motor position for the best vibration suppression performance

- Sending the appropriate motor coil current multiplier commands to the stepper motor driver, with precise timing for the steady stepping period, until the current motor position reaches to the target position.
- Sending important parameters about the code flow, such as the vibration data, program status, calculated RMS and dominant frequency values and motor position, over serial port to the computer for debugging purposes

While doing the above mentioned operations, the following limitations on the microcontroller must be taken into account.

- The memory block of the microcontroller dedicated for the data to be processed in DSP operations is limited to 1 kB, which means it can only store 256 samples of vibration data.
- Due to the memory limitations, it is impossible to execute the FFT routine in a moving window fashion, i.e. the acquisition of the 256 samples must be completed before the FFT function is called.
- The number of samples, the memory limit, in an FFT operation directly affects the resolution of the frequency bins of the resulting frequency domain vector.
- The DSP engine of the microcontroller is optimized to conduct mathematical operations in Q15 fixed point arithmetic [57]. The principle of this format is that the range of numbers stored in 16 bits is $[-1, 1 - 2^{-15}]$ with a 2^{-15} resolution. So, every data that is processed with this approach must be properly normalized to these limits. This type of computer arithmetic has several advantages and disadvantages, such as:
  (i) No matter how many numbers stored in this format are multiplied, the result remains within the limits.
  (ii) Operation on this type of data is extremely fast, compared to floating point arithmetic.
  (iii) Since the numbers are stored in a fixed resolution, loss of information may

be significant especially in operations involving small values.

- Compared to a computer code, timing of tasks in a microcontroller code are much more crucial and attention must be paid during the construction of the code flow.

The code of the tuning algorithm is designed to meet these requirements and to overcome possible problems due to the discussed limitations.

First, in order to strictly control the timing in vibration data acquisition from the sensor, execution of RMS, FFT calculations and tuning operations and motor stepping period, certain interrupt modules of the dsPIC are enabled. For the accelerometer communication, SPI Receive interrupt is set. Therefore, at the instant when a vibration sample is received over SPI protocol, the execution of the code immediately jumps to the routine of storing the sample to the appropriate location in the memory block. Then, the microcontroller resumes from where it was interrupted. Similarly, a timer interrupt with a constant period is set. When the period is completed, the code checks whether a certain task must be executed or not, such as sending new step command to the driver, data sending to the debug computer or execution of certain calculations.

While these interrupts handle the punctual operation of the code, several involved routines handle the main algorithm of the system. These are:

- Calculation of the RMS of the most recent 32 samples
- Updating the maximum value stored in the sample vector, so that the entire vector can be dynamically scaled to minimize the loss of information due to the fixed point arithmetic.
- Fine tuning the string-tension according to the most recent RMS values, thus reducing response time in the adaptation process while waiting for the entire 256 samples to be collected for FFT.
- When these 256 samples are collected:
  (i) Scaling the sample vector according to the maximum value in the vector.
  (ii) Windowing the sample vector with the Hanning Window to enhance the FFT result.

(iii) Calculating the FFT of the sample vector.

(iv) Determining the most dominant frequency bin.

(v) Interpolation of the frequency bin results, so that the limited resolution of the frequency domain data can be reduced [58].

(vi) Interpolation of the look-up calibration table entries to find the target motor position that will shift the antiresonance frequency to the frequency of excitation.

Most of the algorithm is coded in a modified C language, C30, developed for the dsPICs. However, involved mathematical operations are generally coded in assembly language. Only the FFT routine is implemented from the application example of the producer of the microcontroller. The entire code is given in Appendix D.

3. PROTOTYPE DESIGN AND EXPERIMENTAL PROCEDURES

In this section, the design and construction of the prototype of the system subject to this thesis is explained. Additionally, the experimental procedures followed with the intention of verifying the theoretical calculations and evaluating the vibration suppression performance of the designed DVA system are discussed. In the upcoming section, the results obtained from these experiments are presented.

3.1. Design of the Prototype

During the design stage, the prototype is aimed to be both modular and precise. First of all, the setup should enable the construction of the string-mass DVA system using strings of different lengths and diameters. Therefore, using the same structure systems having different dynamical properties and tuning frequency ranges can be obtained. Creating the system with such a requirement not only necessitates modular string mounting systems but also a negative stiffness system with adjustable properties is required. Moreover, a string clamping structure is needed which can tightly hold the string without any slippage while enabling easy replacement of the string.

In addition to the above mentioned features, there are several details in the realization of the design that can influence the effectiveness of the system and cause inconsistencies with the theory. Some of these major details are the propriety of the contact points of the string-mass system, and preventing friction, misalignments, loose fits and backlashes, excess deflections in the structure.

By taking the above mentioned requirements into account, a prototype of the system is constructed. In the following subsections, the details of this realization are explained.

3.1.1. The Base of the System

The prototype is constructed on an aluminum structural framing profile. These profile systems are widely used in many applications of various goals. There is vast number of different cross-sections of these structural beams available in the market. By using nuts, screws and connector elements, specially built for these profiles, construction of many different systems can be accomplished easily [59].

In the current design, a single 45 x 90 mm cross-section profile (see Figure 3.1) is employed as a base to the overall system. Fixtures that fit to this profile are manufactured for mounting the system to the base. Thus, by changing the position of these fixtures along the base, the system can be assembled with various sizes of strings.

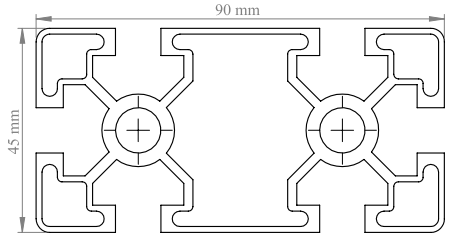


Figure 3.1. Cross-section profile of the 45 x 90 mm aluminum structural beam.

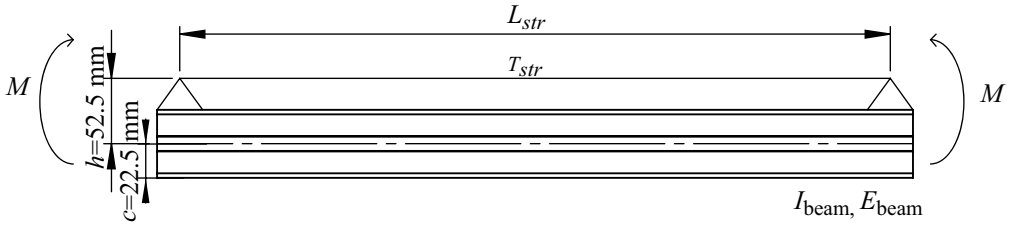


Figure 3.2. Structural load case on the aluminum base due to string tension.

In Figure 3.2, illustration of the load case for this base part is given. The structural beam will bend due to the moment applied by the string tension. The parameters of the structural beam required for the analysis of bending is given in [60]. Namely,

moment of inertia, $I_{\text{beam}} = 73.36\ \text{cm}^4$ and $E_{\text{beam}} = 69$ GPa. The other parameters in Figure 3.2 are taken as in the actual prototype: $L_{\text{str}} = 300$ mm and $h = 52.5$ mm. The bending strain on the outermost surfaces of the beam as a function of the string tension, T_{str}, can be calculated as

$$\begin{aligned} M &= hT_{\text{str}} \\ &= 52.5T_{\text{str}}(\text{N mm}) \end{aligned} \tag{3.1}$$

$$\begin{aligned} \epsilon_{\max} &= \frac{Mc}{E_{\text{beam}}I_{\text{beam}}} \\ &= 23\ 10^{-9}\ T_{\text{str}}. \end{aligned} \tag{3.2}$$

As shown, despite the moderately large moment arm between the string attachment points and the beam's natural axis, the bending strain on the aluminum profile is far below the yielding ($\epsilon_{\max} \approx 0.2\%$) point of the material, even if the string tension reaches up to 1 kN. Due to the modularity requirement, this profile is used, even though one can comment that this profile is overdesigned for this setup. Unlike a prototype, in a final product, every parameter would be determined. Therefore one can omit the usage of such a base part and directly append the system to a primary structure. In addition of being a base part, this structural beam is also used as the protected mass of the primary structure in the experiments.

3.1.2. String Attachment

In theoretical calculations, the boundary conditions of the two ends of the string-mass system are assumed as pinned connections. In order to properly realize these boundary conditions, the string is spanned between two metal pulleys. The string fixtures are then placed slightly lower than the level of the pulleys (see Figure 3.3 - 3.4), so that the string is always in contact with these pulleys. Since the string firmly touches to the pulleys, any frictional losses to cause unexpected levels of damping are minimized.

In addition to these two pulleys, another pulley is placed to guide the string to the level of tension adjustment mechanism, so that the string is kept horizontal and does not apply any vertical force to the tension adjustment mechanism.

Figure 3.3. Pulley system used for string attachment.

The string attachment part should bare large tensions and should enable easy replacement of the string when necessary. In order to achieve this task, miniature drill chucks are used for both attaching the string to the end points and also for attaching the DVA mass in the middle of the string. These drill chucks can tightly grab the string; however, in order to ascertain the grip, remainder string portion is fixed between nuts, as shown in the Figure 3.4 -3.6.

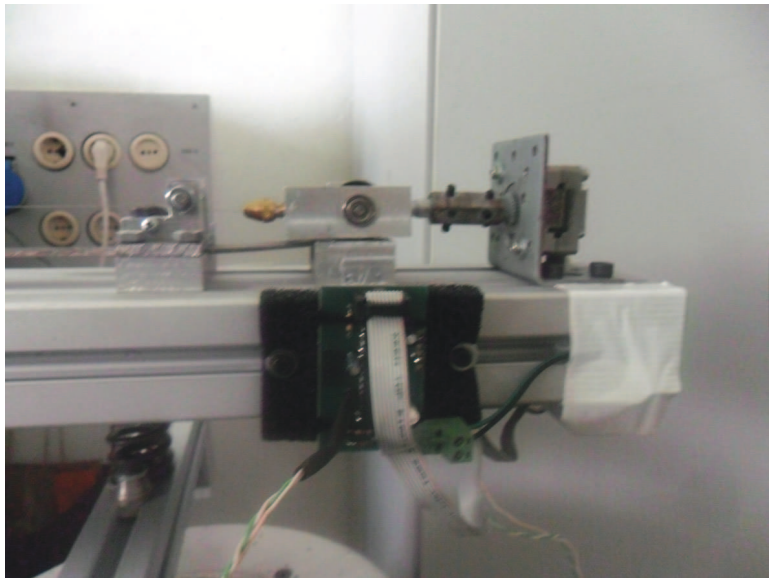


Figure 3.4. Pulley arrangement, moveable string end point connection and the tuning system.

Figure 3.5. Attachment of the mass to the string using drill chucks.

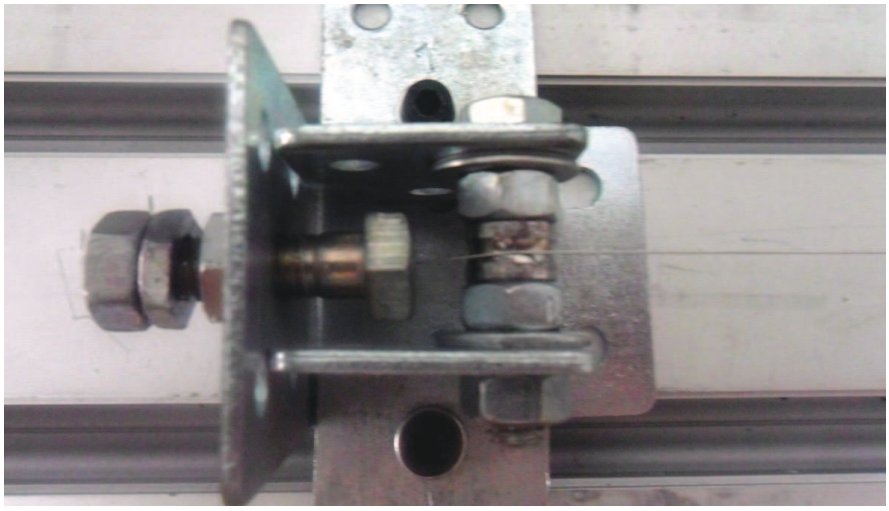


Figure 3.6. Fixed end of the string.

3.1.3. Negative Stiffness Mechanism

In the design of this project, a negative stiffness mechanism is utilized as a tuning actuator aid. Among many different options, a design composed of a rigid link and a pre-compressed spring is chosen. The force-displacement behavior of this design was derived in Equation 2.41, and it was shown that if the pre-compression of the spring is equal to the length of the rigid link, one can obtain a mechanism having a constant negative stiffness of magnitude equal to the stiffness of the pre-compressed spring (Equation 2.42). In other words, the force-deflection relation of the resulting system becomes linear if the spring is pre-compressed with an amount equal to the link length.

Even though the design is composed of standard elements such as links, springs and hinges; proper realization of it is more involved. The solution steps of the realization are as follows.

First of all, the maximum load on the spring of this system occurs when the system is at the unstable equilibrium state, i.e. the beginning of the working range. This means, even if the full deflection range of the system, which is the link length, is not used; the spring has to stand the maximum loading state. This maximum load depends on two factors. The first factor is the spring stiffness. The spring stiffness must match with the stiffness of the string in the DVA system, so it cannot be modified for a given string with the intention of reducing the stress on the pre-loaded spring. The other factor is the pre-compression deflection of the spring. As stated above, this pre-loading must be equal to the link length for a linear operation of the system. Thus, the maximum stress occurring on the spring due to pre-compression is directly affected by the link length. Therefore, it can be minimized if the link length is kept as small as the working range allows. Since the negative stiffness mechanism will undergo deflections less than 5 mm in this prototype, the rigid link length does not have to be larger than this value.

Secondly, the design involves a number of hinges and kinematic constraints. Any

loose fit or excess friction due to improper realization of them would dramatically reduce the precision of the system and affect the resulting force-deflection behavior. So, in order to prevent such problems a miniature crank shaft like assembly composed of a set of ball bearings is designed to construct the mechanism. First, a pair of ball bearings with bore diameters larger than the desired rigid link length is used to act like wheels. Therefore, the resistance to the horizontal movements of the system is reduced while constraining the lower end of the link vertically. In addition to this, these bearings also act as the lower hinge of the link. Second, a small rod eccentrically placed with respect to the axis of the first pair of bearings is used as the upper end of the rigid link. Thus, the eccentricity of this rod determines the link length. Also, with the purpose of mounting the pre-compressed spring, a small block is connected to this rod using another pair of ball bearings. These second set of bearings form the upper hinge of the link. Illustrations of the kinematically equivalent mechanism and the realized version of rigid link part of the system are compared in Figure 3.7.

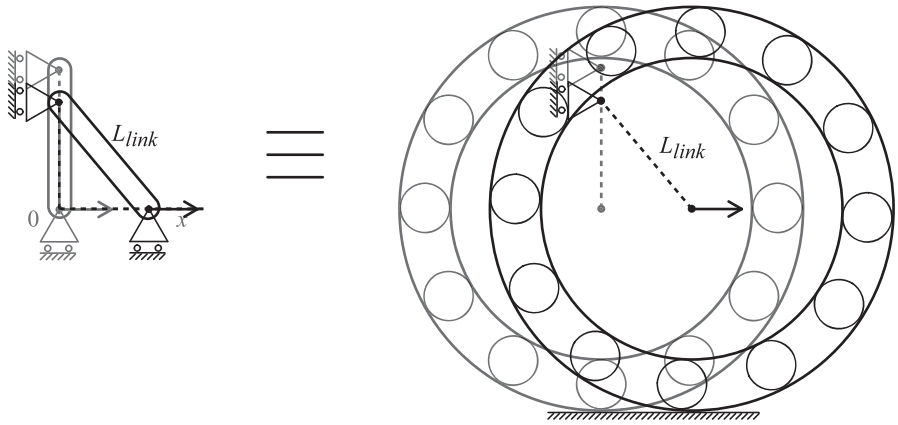


Figure 3.7. Kinematical comparison of the rigid link and equivalent mechanism built using ball bearings.

Thirdly, a suitable spring solution for the negative stiffness mechanism must be determined. One of the most important assets required from the solution is the adjustability of the spring constant. As discussed before, the adjustability of this spring constant enables both the usage of different strings for the DVA system, and correction of small mismatches in the negative and positive stiffness elements. In addition to

that, the spring must operate only in the vertical direction. Helical springs are not very suitable for this mechanism in terms of the above mentioned features. Also, they cannot operate unidirectional unless special guidance is used. On the other hand, by employing a cantilever leaf spring as the spring of the mechanism, these requirements are easily met. Due to the geometry of a cantilever leaf spring, the upper end of the rigid link is forced only to move vertically. Moreover, by changing the effective length of the spring, one can easily adjust the stiffness of it. Using the beam theory, we can calculate the spring force of the cantilever leaf spring used in the prototype as a function of tip deflection.

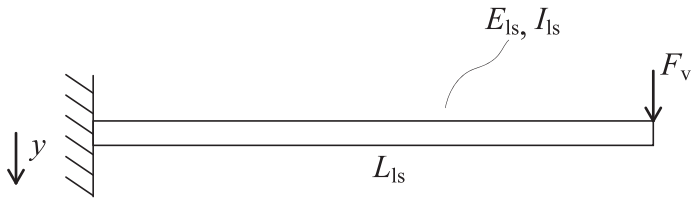


Figure 3.8. Bending stiffness calculation parameters.

The parameters used in the problem are illustrated in Figure 3.8 and defined in Table 3.1.

Table 3.1. Parameter definitions of the leaf spring bending stiffness calculation

E_{ls}	Elastic modulus of the cantilever leaf spring
L_{ls}	Length the cantilever leaf spring
I_{ls}	Second moment of inertia of the cantilever leaf spring
k_v	$\frac{F_v}{y}$ Bending stiffness of the cantilever leaf spring

$$\begin{aligned} k_{\mathrm{v}} &= \frac{F_{\mathrm{v}}}{y} \\ &= \frac{3E_{\mathrm{ls}}I_{\mathrm{ls}}}{L_{\mathrm{ls}}^3} \end{aligned} \tag{3.3}$$

From Equation 2.42, if the tip of the leaf spring in the mechanism is pre-loaded by an amount of L_{link}, then $k_{\mathrm{ns}} = -k_{\mathrm{v}}$.

For a given leaf spring parameters E_{ls}, I_{ls} and string-mass system parameters r_{str}, L_{str}, L_{m} and E_{str}, the equation for L_{ls} that result in a zero-stiffness system, $k_{\mathrm{t}} = 0$, is given below.

$$\begin{aligned} k_{\mathrm{t}} &= k_{\mathrm{str}} + k_{\mathrm{ns}} \\ &= \frac{\pi r_{\mathrm{str}}^2 E_{\mathrm{str}}}{L_{\mathrm{str}} - L_{\mathrm{m}}} - \frac{3E_{\mathrm{ls}}I_{\mathrm{ls}}}{L_{\mathrm{ls}}^3} \\ &= 0 \end{aligned} \tag{3.4}$$

$$L_{\mathrm{ls}} = \sqrt[3]{\frac{3E_{\mathrm{ls}}I_{\mathrm{ls}}(L_{\mathrm{str}} - L_{\mathrm{m}})}{\pi r_{\mathrm{str}}^2 E_{\mathrm{str}}}} \tag{3.5}$$

In the prototype, a clamping fixture for the purposes of beam length adjustments and pre-loading the spring is manufactured. The height of this fixture is set so that when clamped, the deflection of the beam tip with respect to its base becomes equal to the eccentricity of the mechanism, i.e. the rigid link length. Therefore, the pre-compression requirement for a linear negative stiffness behavior is met. Finally, as a means of combining the string of the DVA system, the negative stiffness mechanism and the tuning actuator, a block that encapsulates the negative stiffness structure is connected to the axis of the ball bearings used as wheels. So, the forces applied by the

string tension and by the negative stiffness mechanism are added up. If the stiffness of the pre-compressed spring is adjusted properly, these two forces can cancel out each other over the entire operation range. Both the illustration of the overall negative stiffness mechanism, its operation and the photograph of the realized version is shown in Figure 3.9 and Figure 3.10.

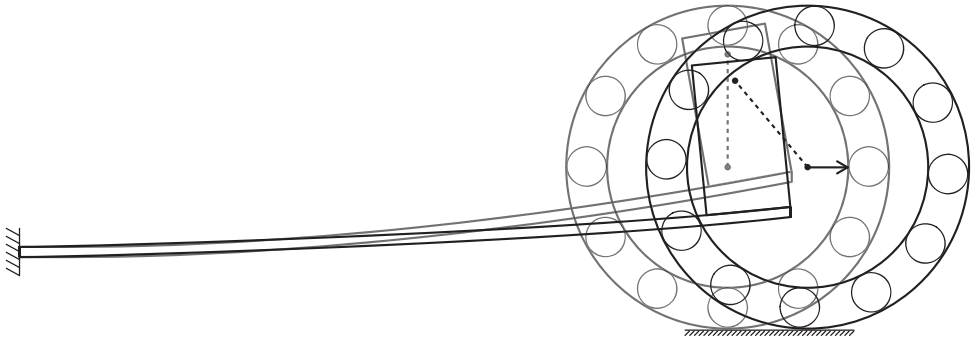


Figure 3.9. Illustration of the realized negative stiffness mechanism operation.

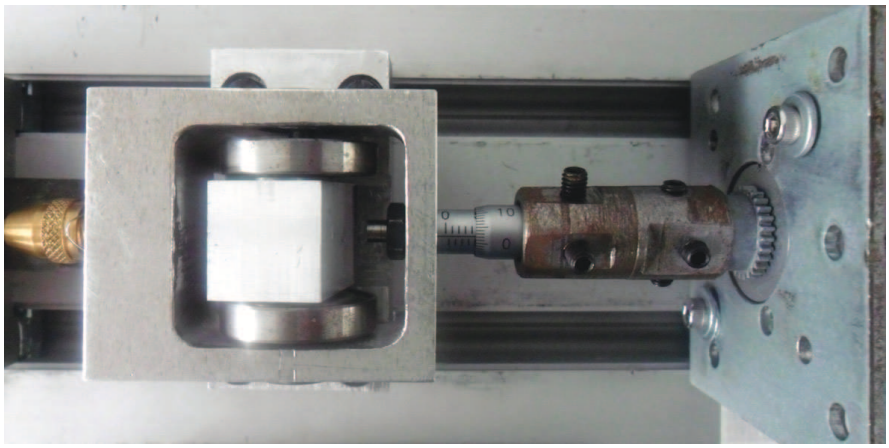


Figure 3.10. Negative stiffness system and the linear position actuator.

3.1.4. Linear Position Actuator

The natural frequency of the string-mass DVA system is tuned by varying the tension of the string. Tension adjustment is conducted by varying the position of the moveable end of the string, which also combined with the negative stiffness mechanism.

Therefore, in the prototype, a linear position actuator is needed. It is very important that this actuator can precisely control the position of the string end. One of the possible solutions is using piezoelectric linear actuators. Although these actuators are extremely precise, accurate and free from any friction, their prices are comparably higher than simpler solutions.

In the realization, a stepper motor driven with open-loop microstepping method is used in combination with a miniature micrometer screw, *Mitutoyo 148-203*, as the linear actuator [61]. The micrometer screw used in this setup has very good precision and accuracy properties. Thus, any mechanical positioning errors caused by the usage of a screw mechanism are minimized. Minor positioning inaccuracies in stepper motors driven by open loop microstepping method are likely to occur due to both mechanical and electrical sources of errors. However, due to the nature of a stepper motor, these positioning errors are not cumulative [55]. Since the tuning algorithm of the system operates according to the vibration suppression performance of the DVA, small positioning errors can be compensated during fine tuning conducted by the controller.

3.1.5. Parameters Used in the Overall Assembly

In this subsection, the system parameters used in the realized prototype are summarized in Table 3.2.

The stiffness of the primary structure spring, k_s, is measured using a tensile testing machine. The result of this test is given in Figure 3.11.

3.2. Experimental Procedures

So far, the theoretical calculations and development of the prototype of the designed system are explained. In terms of theoretical calculations, first the determination of the natural frequency of the string-mass system as a function of string dimensions, attached mass parameters and string tension is discussed. Afterwards, dynamical behavior of this system in combination with a primary structure with the intention of

Table 3.2. Realized prototype parameters

L_{str}	355	mm
L_{m}	80	mm
r_{str}	0.14	mm
m_{a}	84.4	g
m_{s}	2.56	kg
I_{ls}	10.6	mm^4
k_{s}	26.87	N/mm
L_1	700	mm
L_2	850	mm

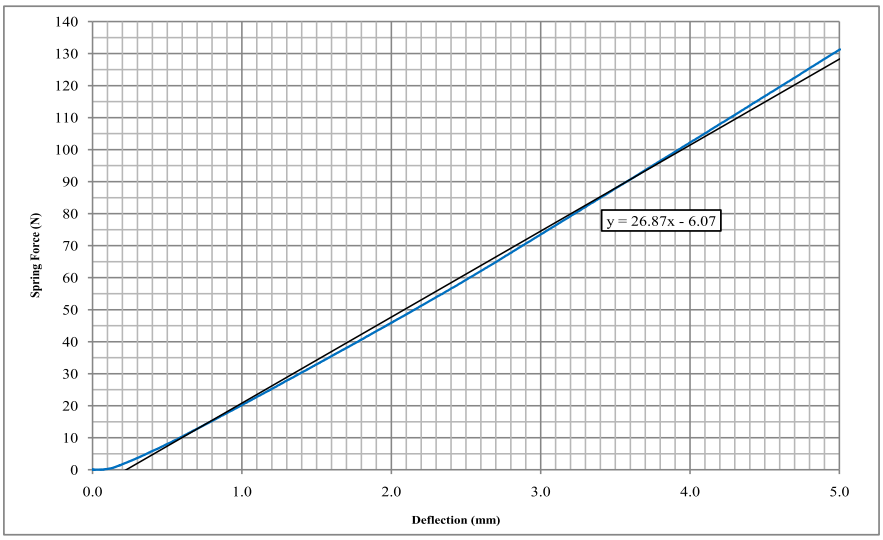


Figure 3.11. Force - deflection plot and linear curve fit of the k_{s}.

vibration suppression is analyzed. Also design of a negative stiffness mechanism that significantly reduces the tuning effort is explained. Finally, the development of a tuning algorithm is discussed. In addition to theory, the development stages of the prototype for proving the validity of calculations and showing that the proposed design can actually operate as an adaptive-passive DVA is explained.

In order to evaluate the behavior of the system, several experimental procedures are determined. In this subsection, these procedures are explained.

3.2.1. Natural Frequency Detection of the String-Mass System

One of the most important elements of the system is the string-mass DVA. In order to verify the calculations and properly operate the tuning algorithm, it is crucial to obtain the relation between its natural frequency and the string tension. During normal operation, this relation is stored as a calibration data in the tuning controller. Hence, this part of the experiment is a calibration operation. The instrumentation used for this purpose will not always stay on the setup.

First of all, a sensor that can measure the vibration of the absorber mass is required. Due to the nature of the mechanism, attachment of a sensor on the mass is problematic. Therefore, a non-contact, *Di-soric DCC 18 M 10/10 AIK-IBS* inductive proximity sensor is utilized [62]. The analog voltage output of this sensor is proportional to its distance to a certain metallic object. Since the distance of the mass to the sensor oscillates as it vibrates, from the output of the sensor the frequency of the vibration can be obtained.

Although obtaining the frequency information using such a sensor is a straightforward procedure, the relation between the sensor output and the amplitude of the vibration is not directly available. Since we are primarily interested in the natural frequency of the system, the information about the vibration amplitudes of the string-mass system is not required.

After selection about the sensor is made, a simple to use data acquisition system, *LabJack U12*, is employed for the purpose of storing the output of the inductive sensor for further calculations [63].

The measurements are carried out over the entire tension range of the string. The independent variable in this experiment is the string tension. Due to practical reasons, the string tension is not directly monitored. Instead of the tension, the position of the string end point is controlled both in this calibration procedure and in normal operation of the tuning system.

In order to detect possible hysteresis and repeatability errors three cycles of data are recorded. As a general practice in calibration operations, the first cycle is started from the middle of the independent variable range. The range is determined to be as 3 mm and this range is divided by 0.1 mm steps. Thus, 60 separate measurements are taken at each cycle.

At each position setting, the DVA mass is hit by a hammer and its vibration is recorded until it dies out significantly. Thus, free vibration behavior of the system for six times at each tension setting is obtained.

Using this recorded time domain data, the frequency content of the free vibration is obtained. The dominant frequency is obtained from each cycle and their mean is assumed as the natural frequency of the system at that particular tension value.

3.2.2. Determination of the Forced Vibration Behavior of the Primary System

It is intended that the beam forming the base of the prototype is also employed as the primary mass. In order to investigate the vibration suppression performance of the DVA system, we first need to observe the behavior of the primary system without a DVA attachment.

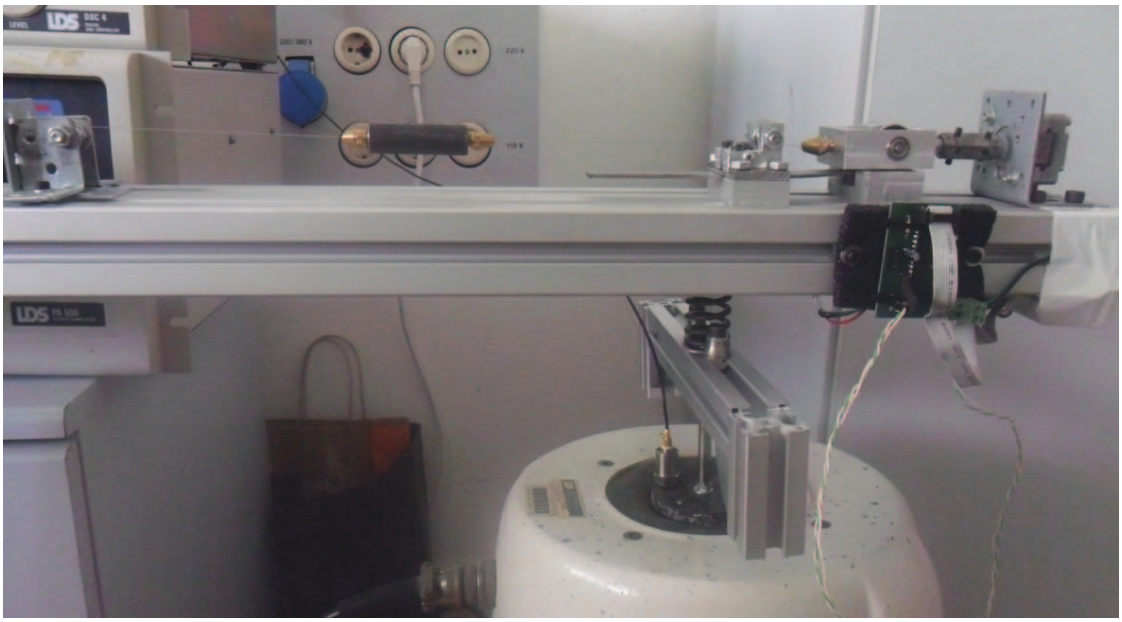


Figure 3.12. The connection of the combined system to the shaker.

An experimental setup, for both this test and the evaluation of the DVA performance, is built as shown in Figure 3.12. The structural beam is connected to the ground with a flexure hinge (see Figure 3.13). This hinge is designed so that, it only allows a SDOF motion to the beam. With the help of this type of hinge, possible frictional losses due to the use of bearings are eliminated. By reducing damping on the primary structure, it is intended to obtain a clear resonant behavior at its natural frequency. Therefore, the vibration suppression performance of the DVA can be clearly seen. After ensuring a SDOF behavior for the system, using a helical spring the structure is connected to the excitation base (see Figure 3.14). By adjusting the position of the connection point of this helical spring, the natural frequency of the primary structure can be varied. After a number of trials, the natural frequency is adjusted to lay within the working range of the DVA system.

The exciting motion to the entire system is applied using a shaker device. This shaker can operate in closed-loop with its controller system. Therefore, no matter in what condition the system is vibrating, a constant amplitude base vibration at

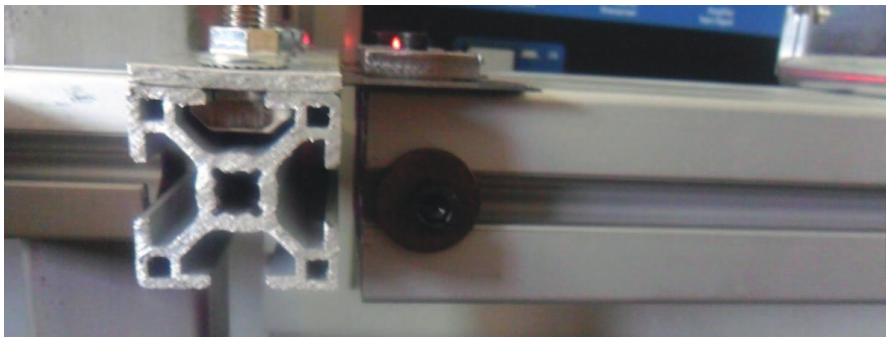


Figure 3.13. Compliant hinge connection of the primary system to the ground.

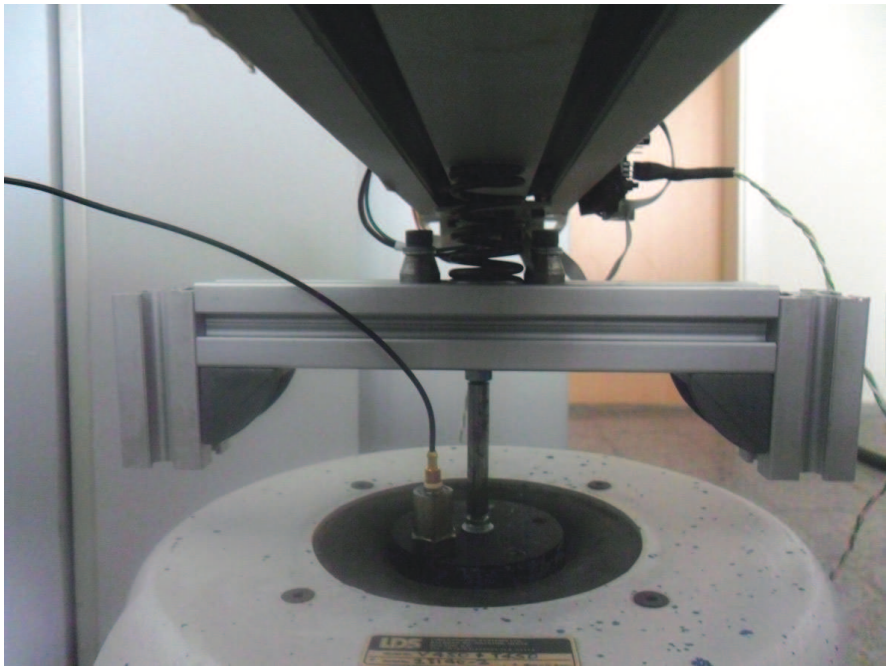


Figure 3.14. Helical spring connection between the shaker and the primary structure.

any desired frequency can be obtained. Moreover, the control system of the shaker enables to apply a sweep of frequencies over a period of time or manual variation of the excitation frequency during operation.

In terms of instrumentation for this test, no extra sensor is used. As part of the tuning system of the overall system, an accelerometer is already present in the primary structure. Therefore, vibration measurements can be accomplished using the information sent by the tuning microcontroller.

To obtain the forced vibration behavior of the primary system, the mass of the DVA is restrained to move. Then a frequency sweep between 13 to 20 Hz of amplitude 0.075 mm and sweep rate of 0.1 Hz/min is applied to the base of the system. For the purpose of checking repeatability, three cycles of sweeps are conducted. Then using the time domain data obtained from the accelerometer and the frequency sweep rate of the shaker system, the frequency response behavior of the primary system is obtained.

3.2.3. Determination of the Forced Vibration Behavior of the Combined System without the Controller

Unlike the unprotected primary structure case, obtaining a frequency response function of the combined system is not a straightforward process. A complete frequency response behavior of the combined mechanical system can be obtained by disabling the tuning controller and applying an excitation sweep at different string tension values similar to the process used in the Section 3.2.2. The results of such an analysis is given in Figure 2.16. However, due to excess vibrations in the DVA mass in mistuned frequencies, this process is only applied at two different tensions for observing the antiresonance region.

Physical setup of this procedure is identical to the test where the frequency response of the primary structure is obtained. Only this time the restraint on the DVA mass is removed.

3.2.4. Determination of the Forced Vibration Behavior of the Combined System in Closed-Loop Mode

To validate the entire system studied in this thesis, its vibration suppression performance and ability to adapt itself to varying frequencies of excitation, a final series of experiments are determined.

As the excitation application scheme, two different strategies are followed. First, similar to the case of the experiment is Section 3.2.2, the frequency is varied over time linearly. While the frequency of vibration is changing, the tuning system is set to automatic tuning mode. Therefore, the obtained time domain data shows the effect of vibration suppression and adaption performance of the overall system. Secondly, the frequency of excitation is varied manually using the control unit of the shaker. With this approach the adaptation performance of the system to arbitrarily varying excitation frequencies are evaluated.

4. RESULTS AND DISCUSSION

4.1. Natural Frequency Detection of the String-Mass System

As explained in Section 3.2.1, the first experiment conducted is the determination of natural frequencies of the string-mass system at different tensions. In Figure 4.1 - 4.31, the frequency response plots of the system for indicated string end point positions are given. Note that, in the experiment the position indicated by the micrometer is used as the independent variable. Hence, the 3 mm range begins from 1.50 mm indication and ends in 4.50 mm. The natural frequencies of the system in each configuration can be obtained from the peak frequencies of the response plots. Note that, the harmonics of the dominant frequencies also appear in the plots. The reason of this apperance is probably the slightly non-linear relation between the sensor output and the DVA mass displacement because of the placement of the sensor on the system. However, since are we mainly interested in the most dominant frequency of the DVA system, these harmonics are not a significant concern for the calibration process. One can also comment on these figures that the system is highly repeatable, no significant drift in the peak frequency value can be observed among different cycles and directions of tension adjustment. Finally, averages of the most dominant frequencies of the system's response at each tension setting is plotted in Figure 4.32.

In order to check the agreement with the theory, the analytical natural frequency calculation method has to be re-evaluated with the parameters of the realized system. By injecting the system parameters given in Section 3.1.5 into Equation 2.33 and solving for ω at different tension values, the analytically predicted natural frequencies are obtained. Note that in analytical calculation, the variable parameter is the string tension, while in the experiments an indirect indicator for the string tension, the end point position, is used. Therefore, in order to compare these two relations, the tension values used in the analytical calculation are divided by the string stiffness and the data is shifted in horizontal axis to match the end point positions. In Figure 4.33, both experimental and theoretical natural frequency vs. string tension relations are

plotted. We can comment that the behavior of the experimental setup can be accurately predicted using the analytical method.

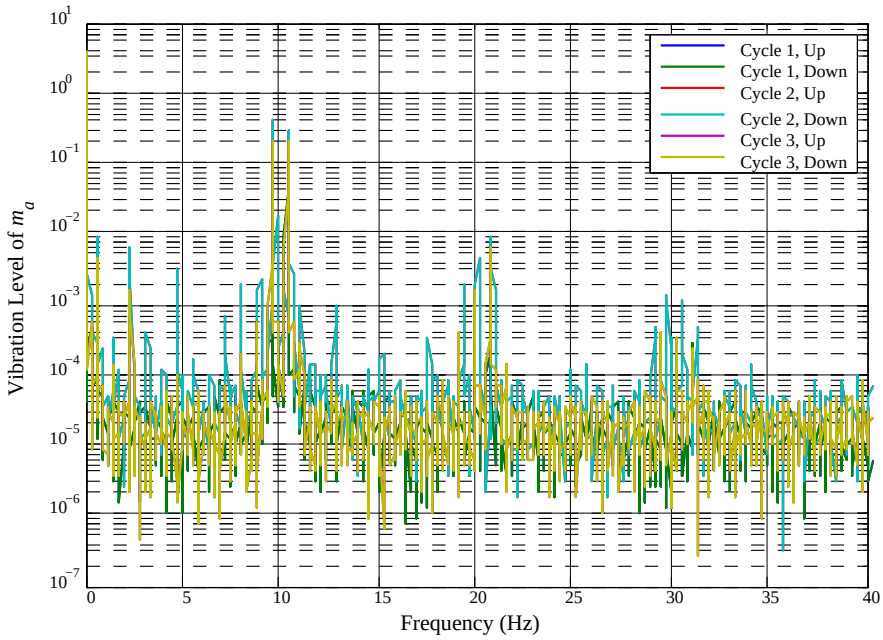


Figure 4.1. Frequency Response of String Mass System at Micrometer Position = 4.50 mm.

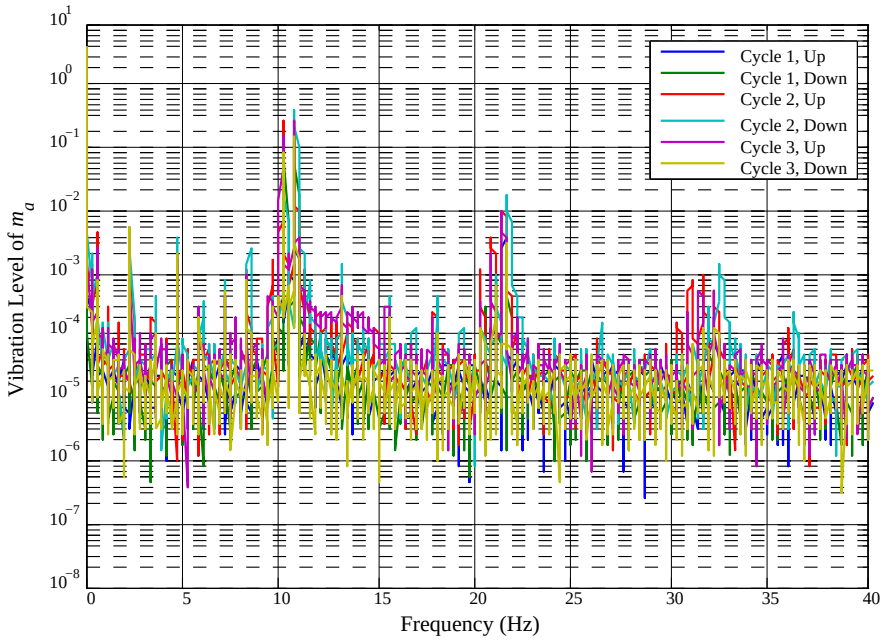


Figure 4.2. Frequency Response of String Mass System at Micrometer Position = 4.40 mm.

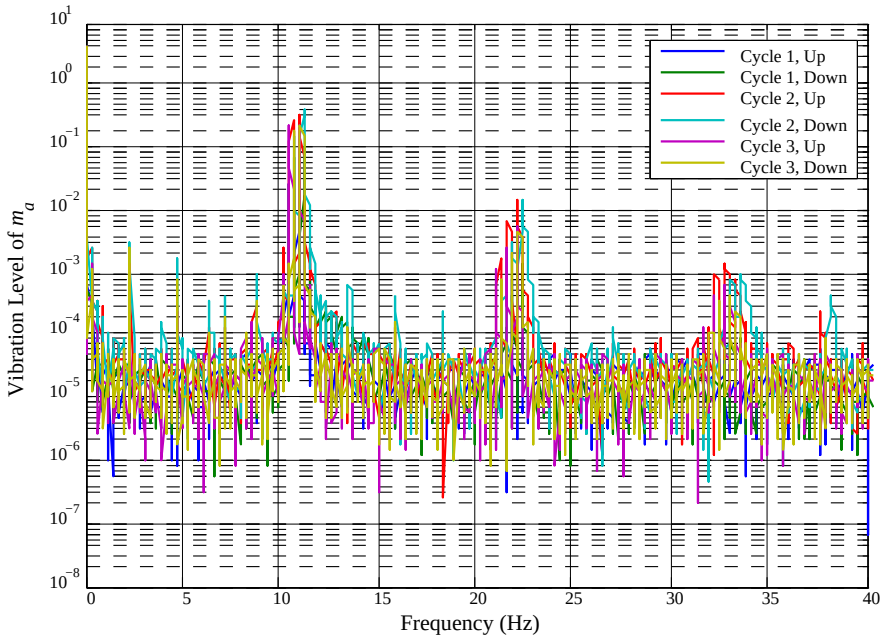


Figure 4.3. Frequency Response of String Mass System at Micrometer Position = 4.30 mm.

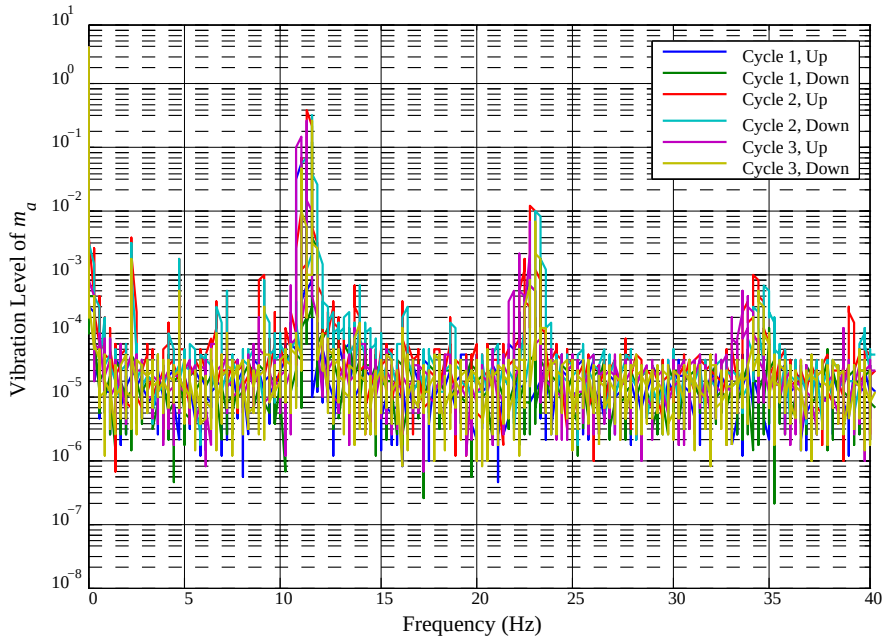


Figure 4.4. Frequency Response of String Mass System at Micrometer Position = 4.20 mm.

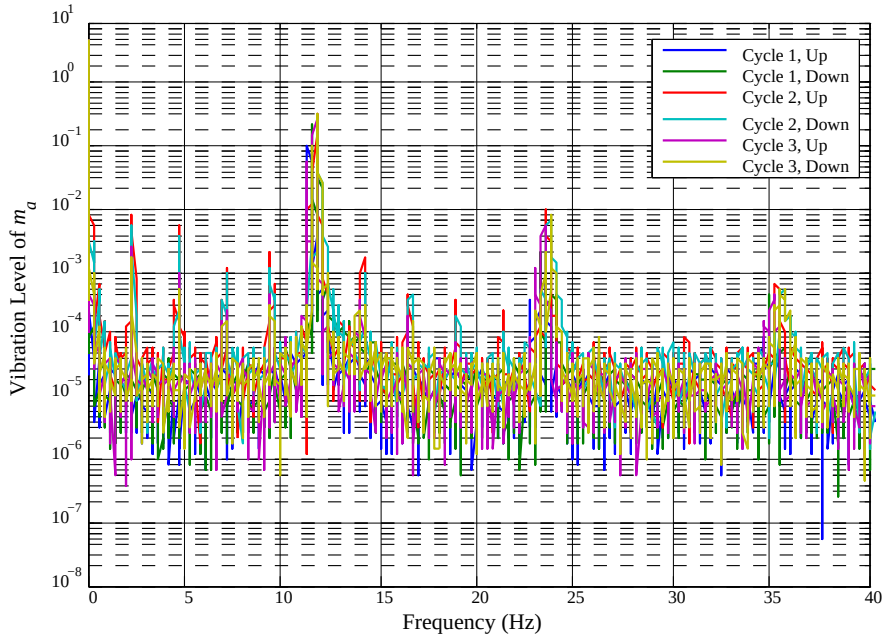


Figure 4.5. Frequency Response of String Mass System at Micrometer Position = 4.10 mm.

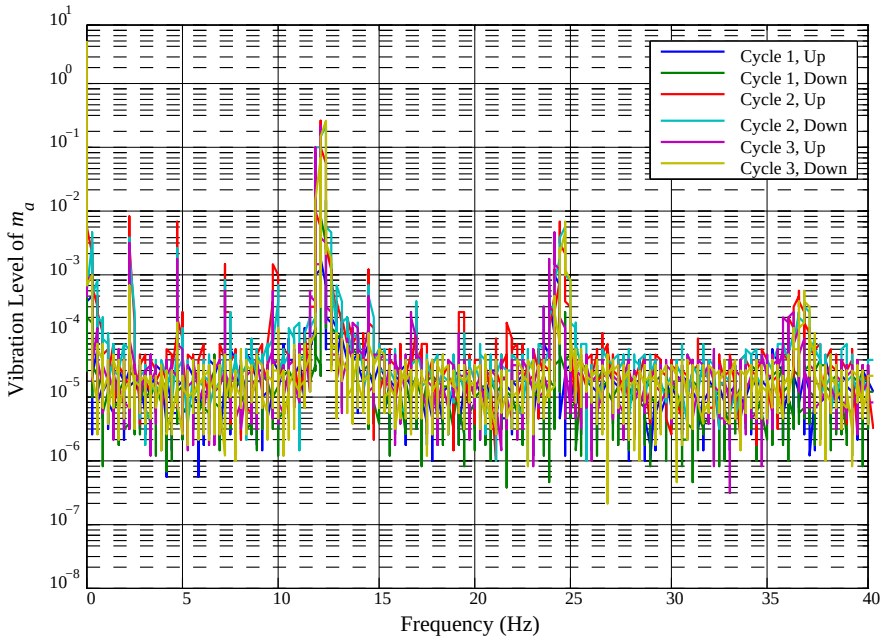


Figure 4.6. Frequency Response of String Mass System at Micrometer Position = 4.00 mm.

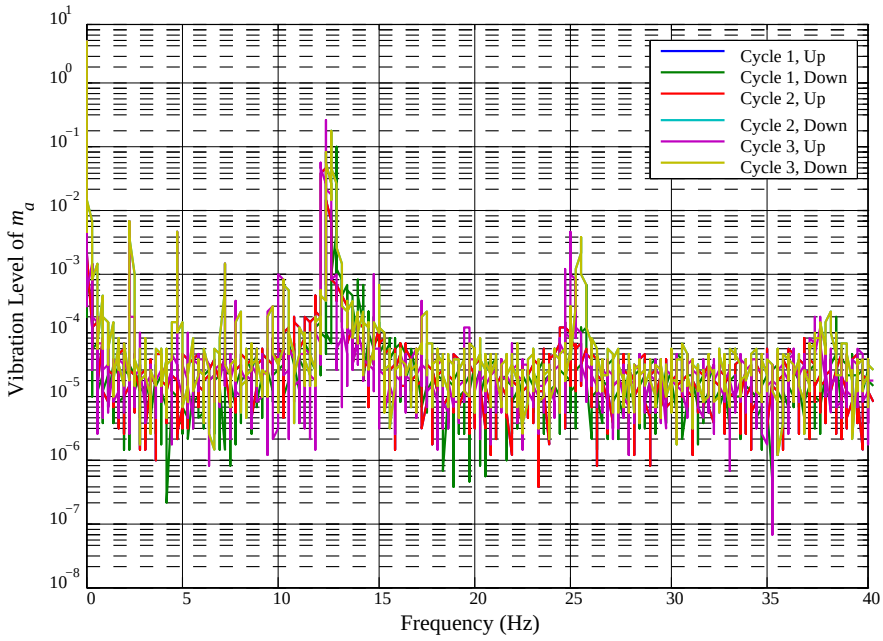


Figure 4.7. Frequency Response of String Mass System at Micrometer Position = 3.90 mm.

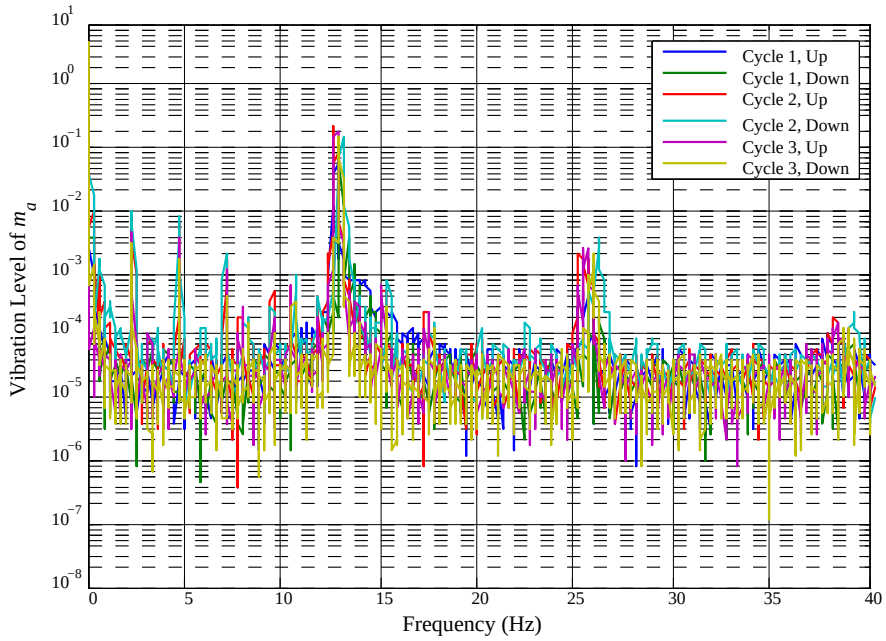


Figure 4.8. Frequency Response of String Mass System at Micrometer Position = 3.80 mm.

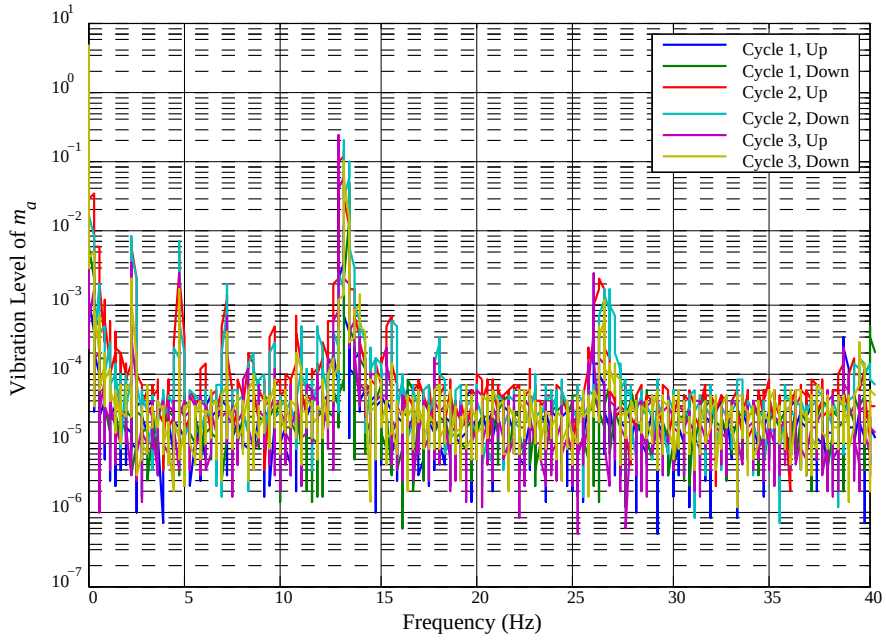


Figure 4.9. Frequency Response of String Mass System at Micrometer Position = 3.70 mm.

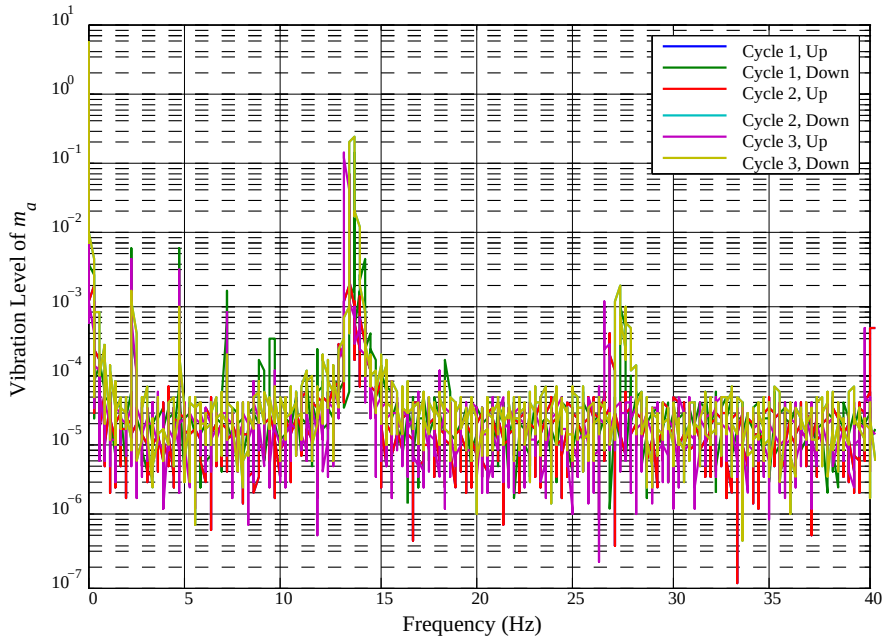


Figure 4.10. Frequency Response of String Mass System at Micrometer Position = 3.60 mm.

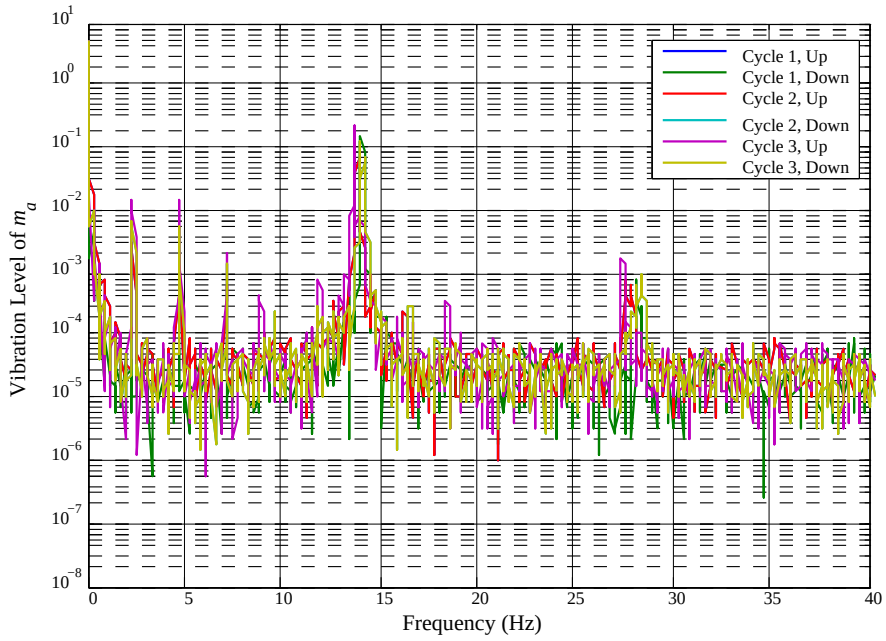


Figure 4.11. Frequency Response of String Mass System at Micrometer Position = 3.50 mm.

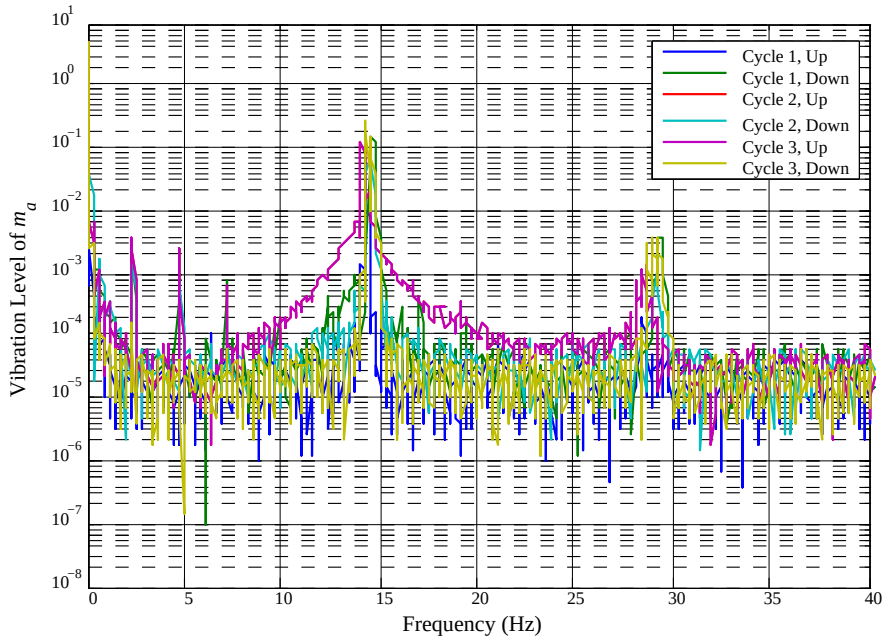


Figure 4.12. Frequency Response of String Mass System at Micrometer Position = 3.40 mm.

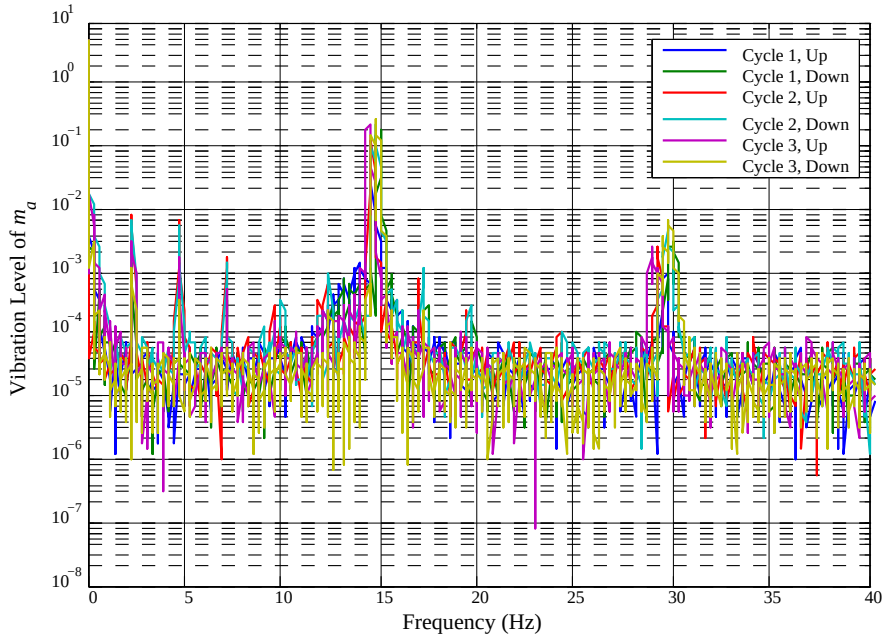


Figure 4.13. Frequency Response of String Mass System at Micrometer Position = 3.30 mm.

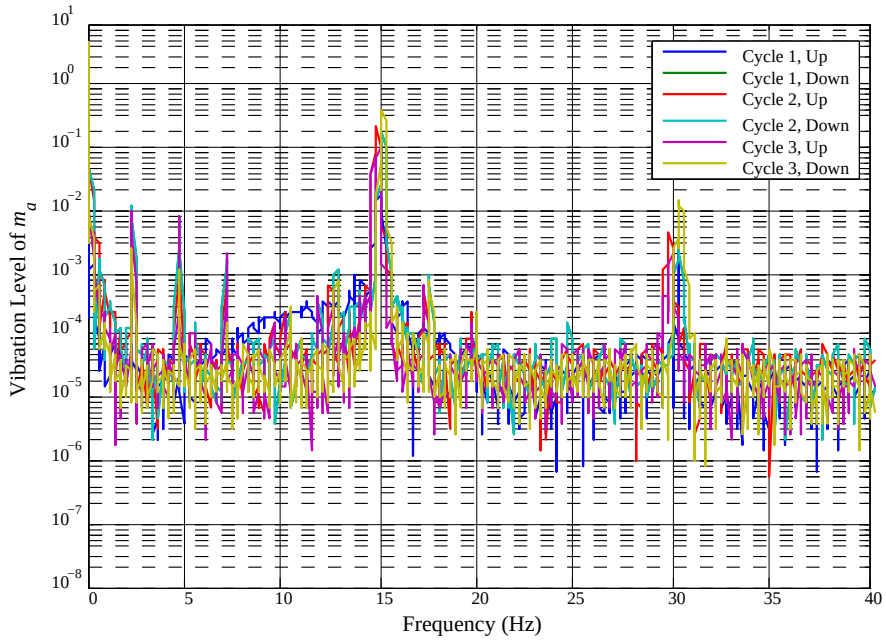


Figure 4.14. Frequency Response of String Mass System at Micrometer Position = 3.20 mm.

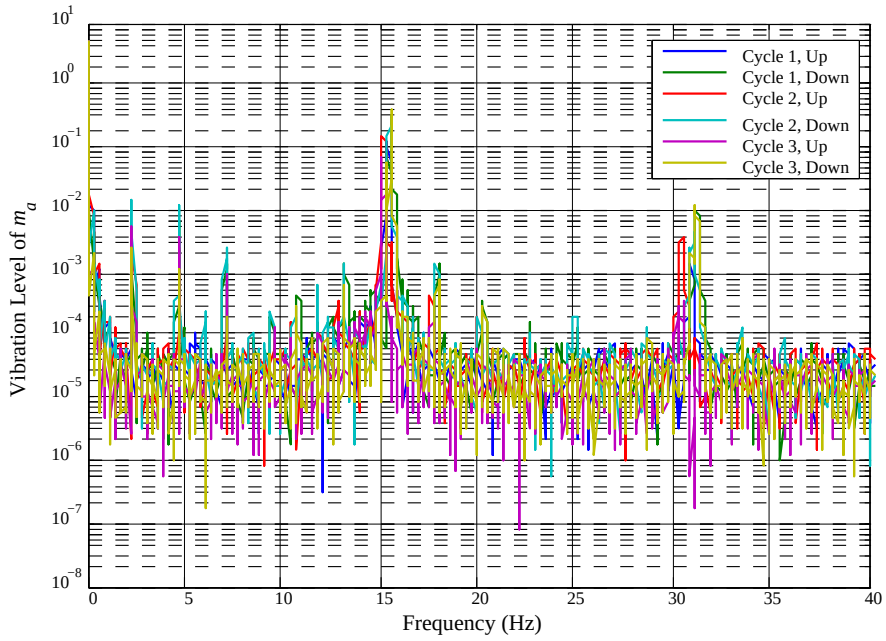


Figure 4.15. Frequency Response of String Mass System at Micrometer Position = 3.10 mm.

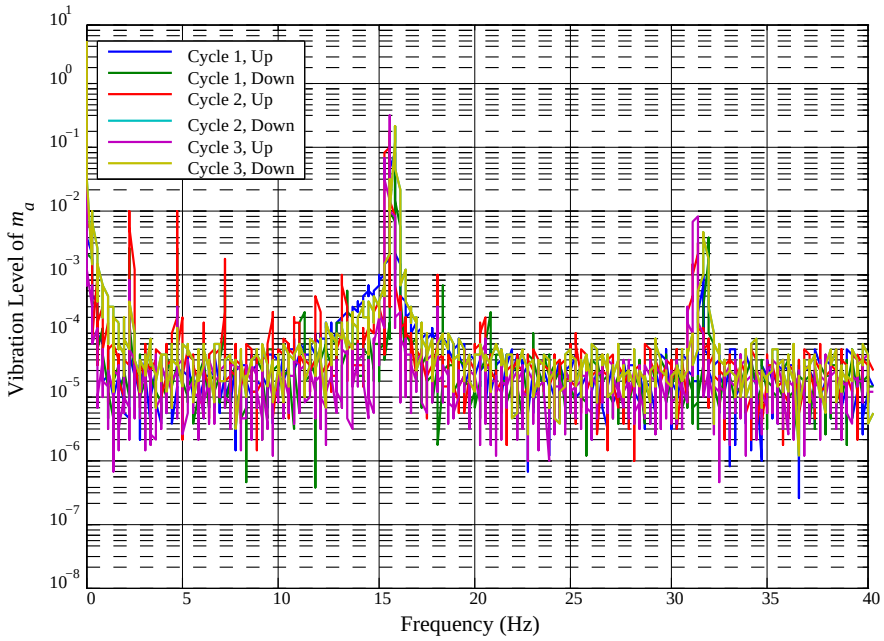


Figure 4.16. Frequency Response of String Mass System at Micrometer Position = 3.00 mm.

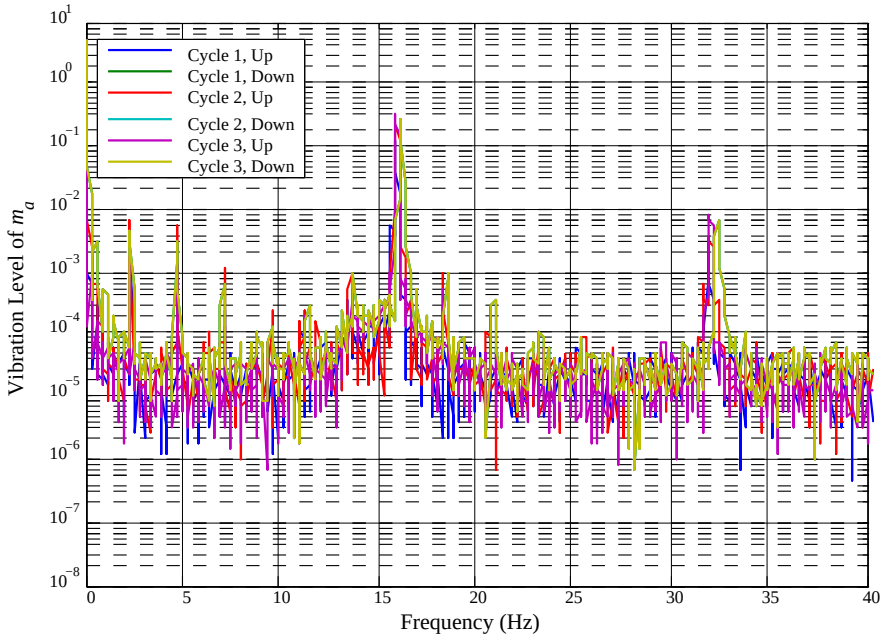


Figure 4.17. Frequency Response of String Mass System at Micrometer Position = 2.90 mm.

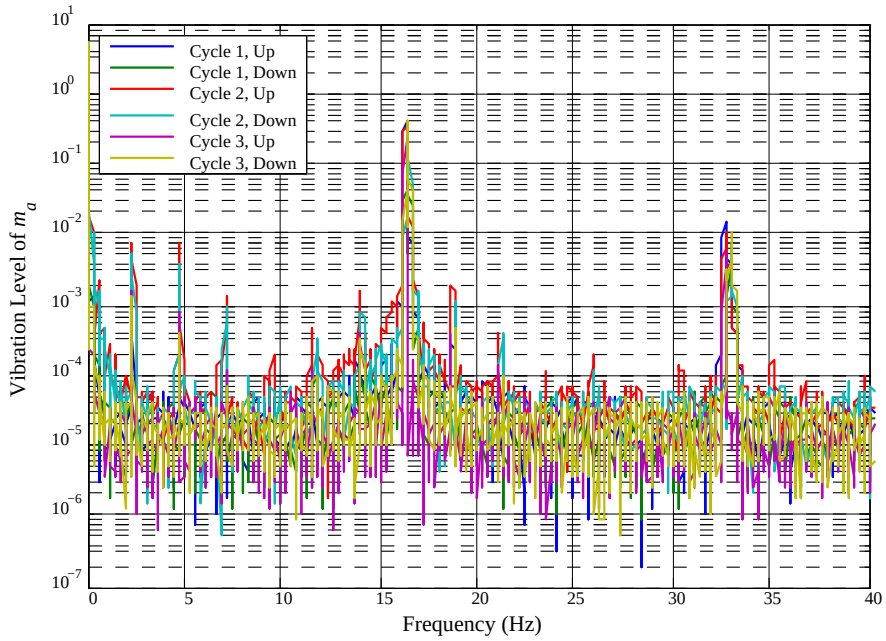


Figure 4.18. Frequency Response of String Mass System at Micrometer Position = 2.80 mm.

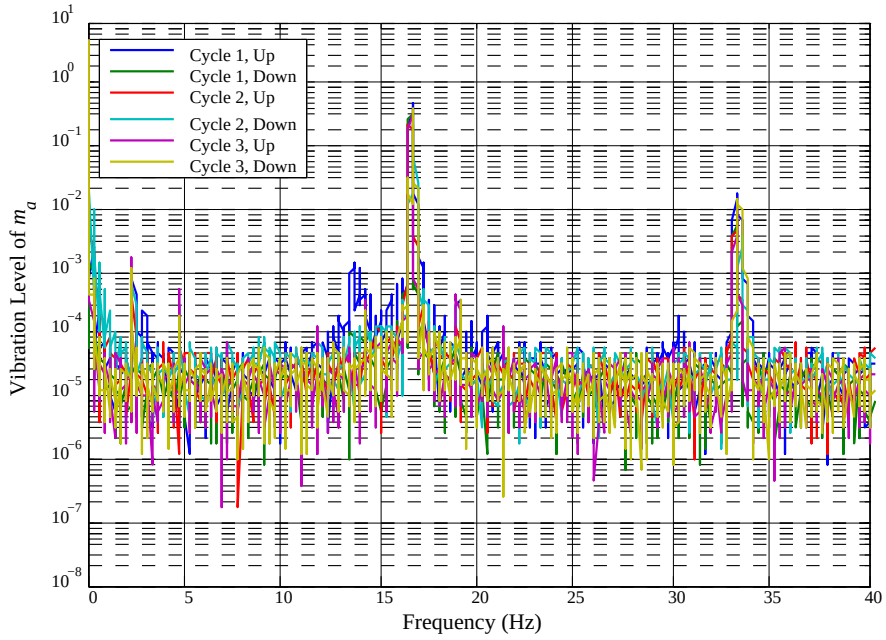


Figure 4.19. Frequency Response of String Mass System at Micrometer Position = 2.70 mm.

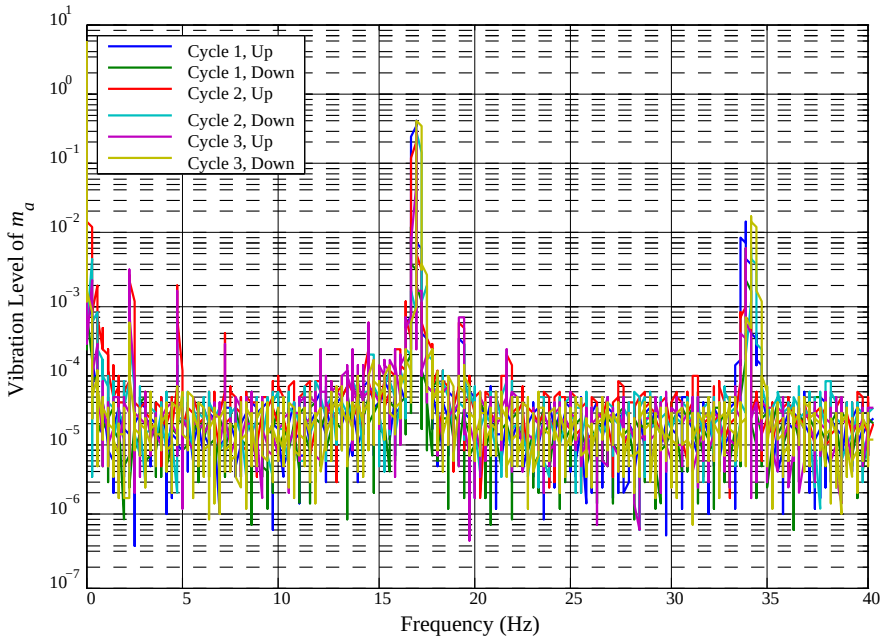


Figure 4.20. Frequency Response of String Mass System at Micrometer Position = 2.60 mm.

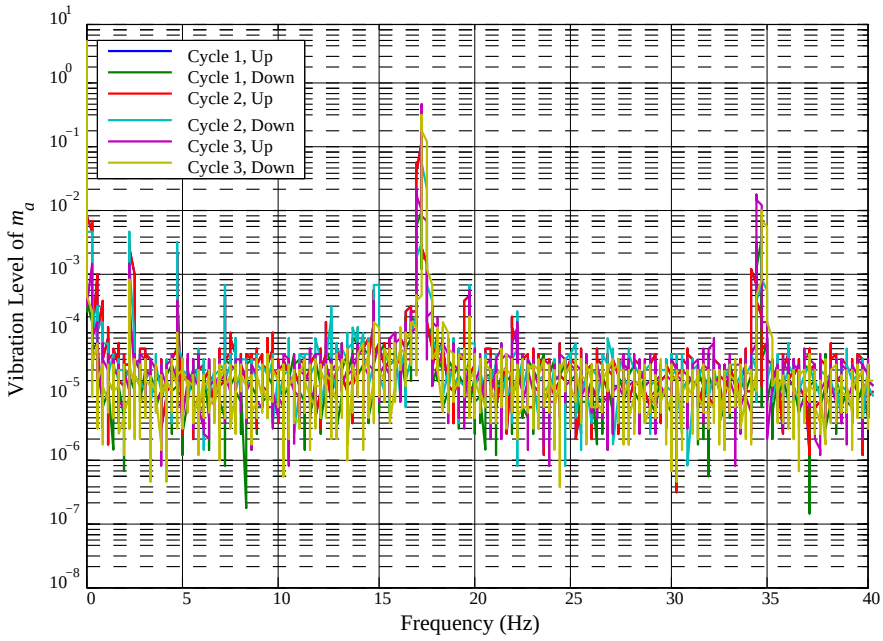


Figure 4.21. Frequency Response of String Mass System at Micrometer Position = 2.50 mm.

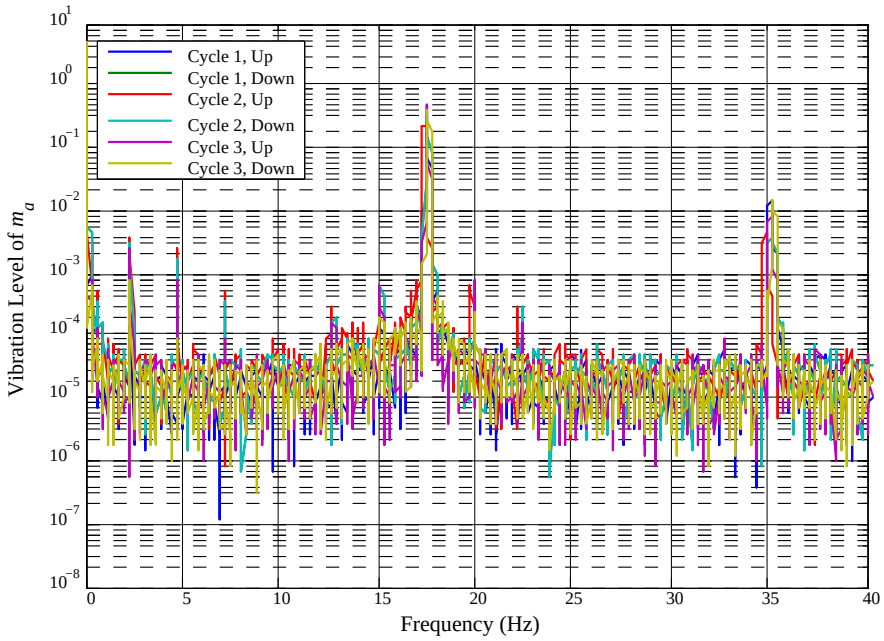


Figure 4.22. Frequency Response of String Mass System at Micrometer Position = 2.40 mm.

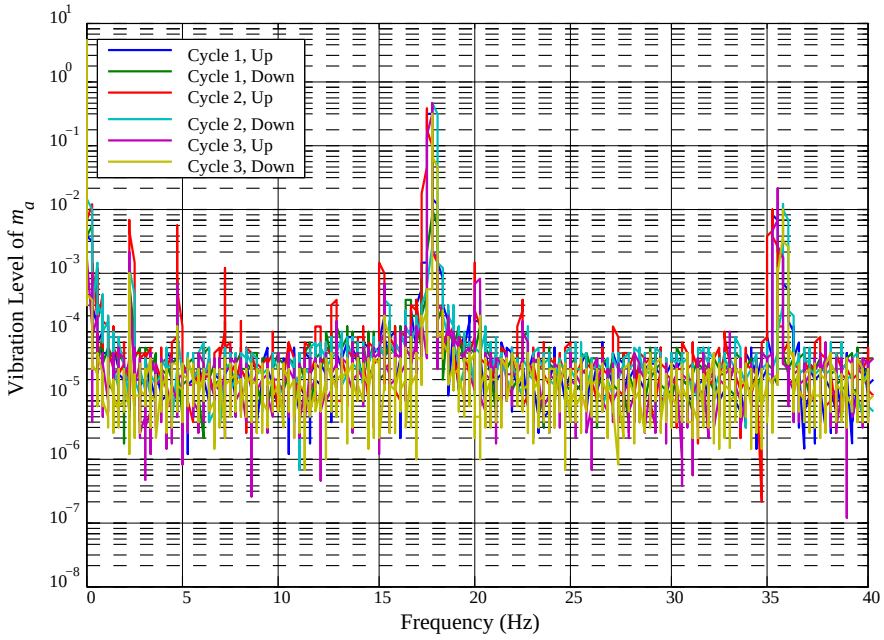


Figure 4.23. Frequency Response of String Mass System at Micrometer Position = 2.30 mm.

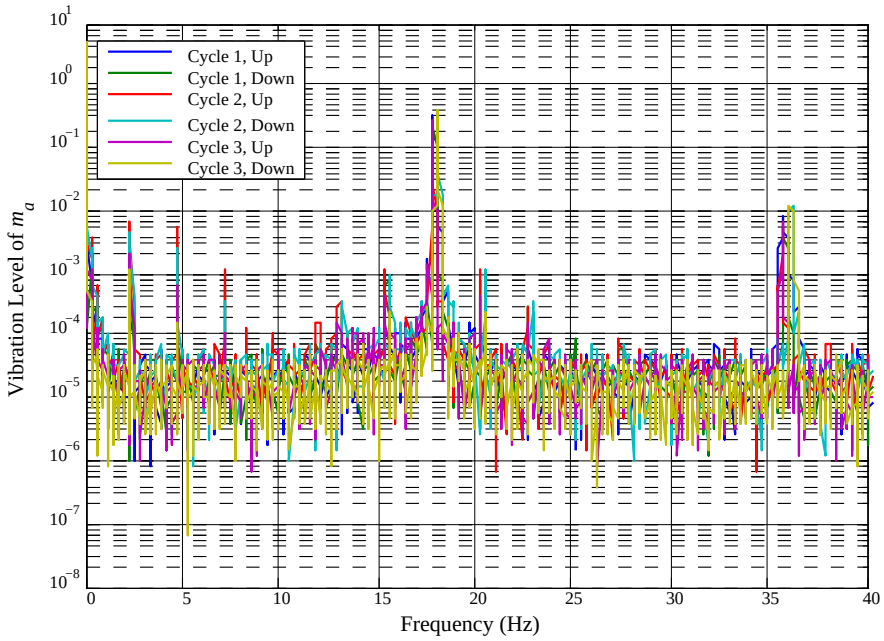


Figure 4.24. Frequency Response of String Mass System at Micrometer Position = 2.20 mm.

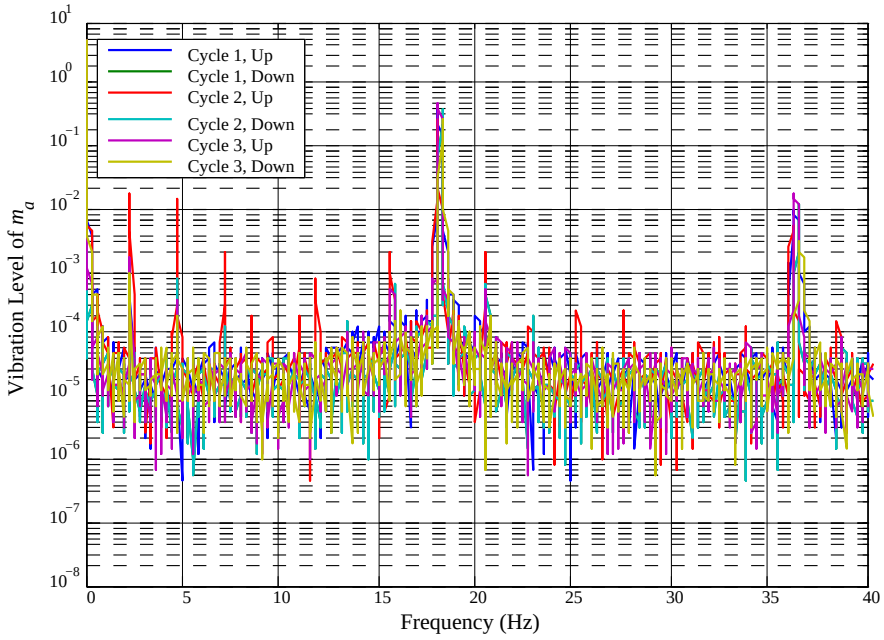


Figure 4.25. Frequency Response of String Mass System at Micrometer Position = 2.10 mm.

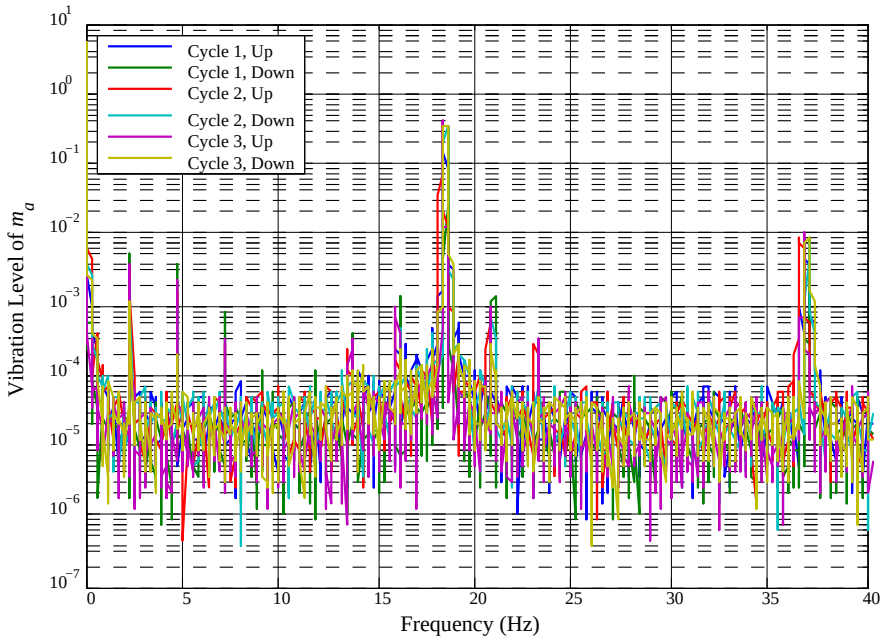


Figure 4.26. Frequency Response of String Mass System at Micrometer Position = 2.00 mm.

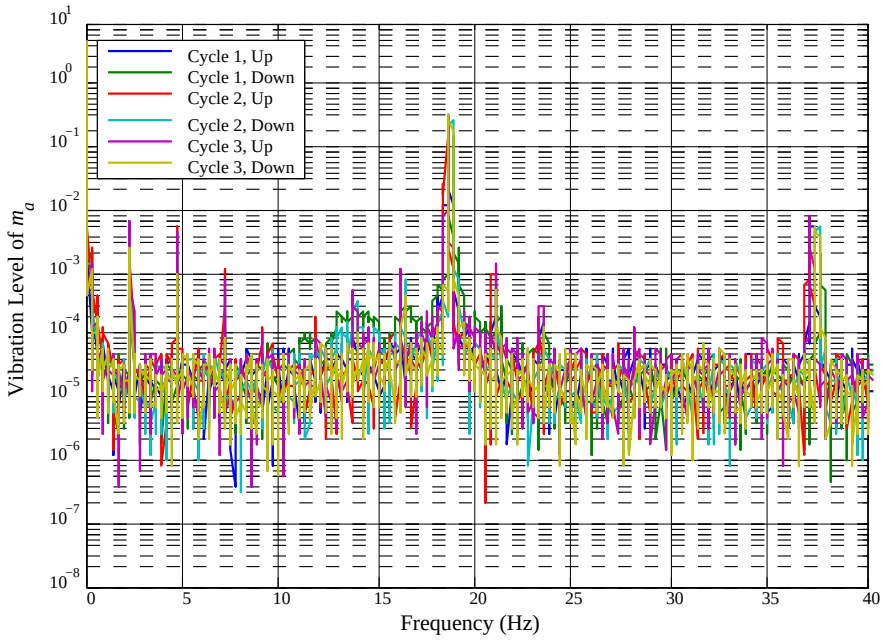


Figure 4.27. Frequency Response of String Mass System at Micrometer Position = 1.90 mm.

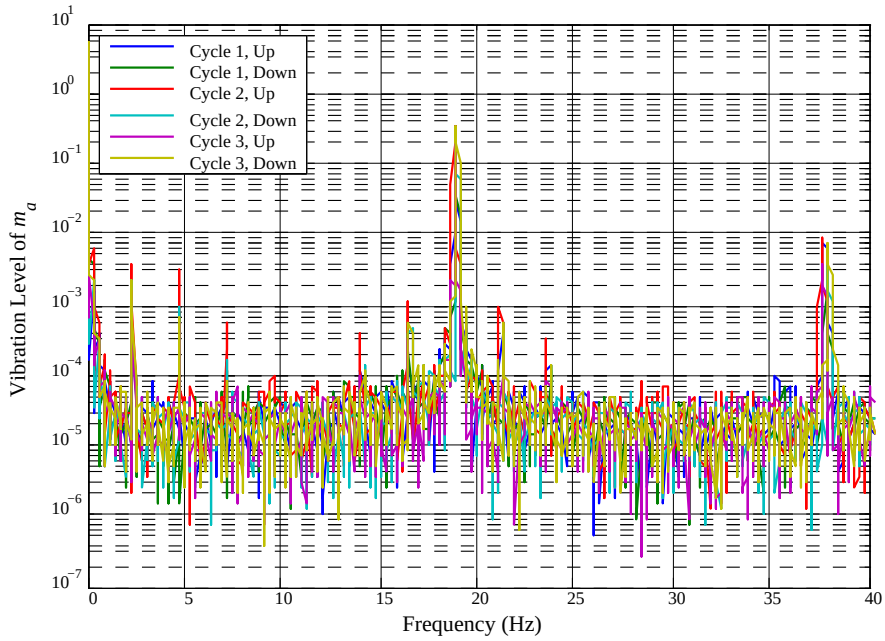


Figure 4.28. Frequency Response of String Mass System at Micrometer Position = 1.80 mm.

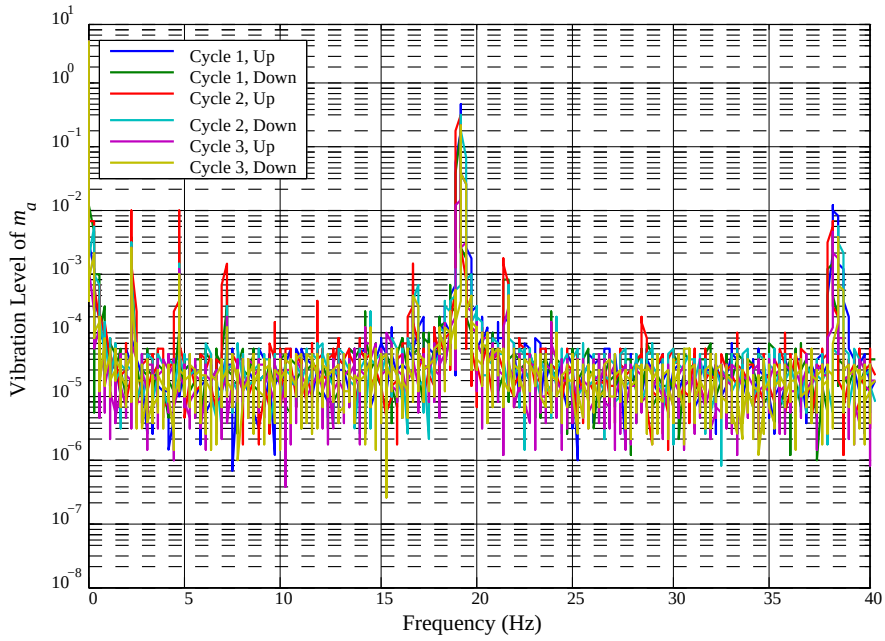


Figure 4.29. Frequency Response of String Mass System at Micrometer Position = 1.70 mm.

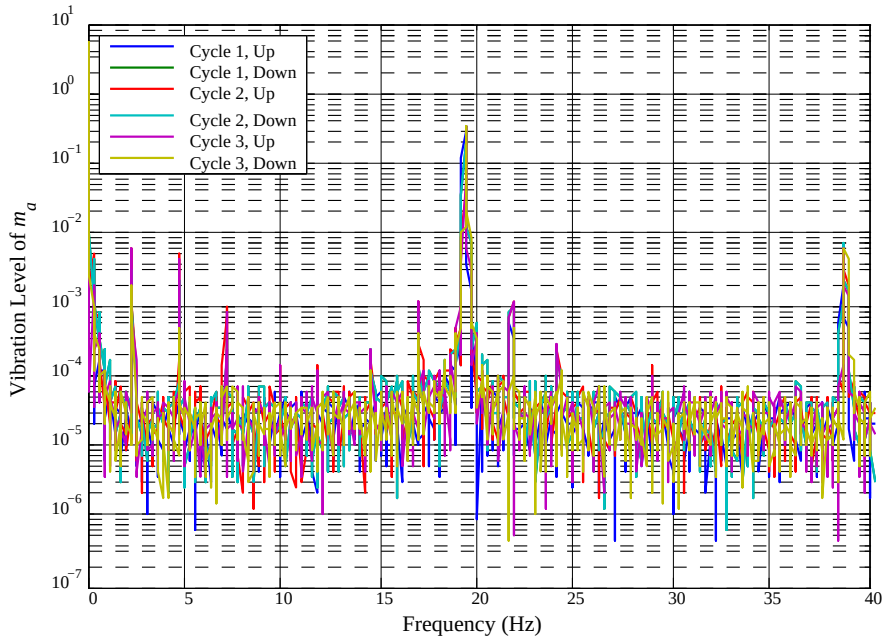


Figure 4.30. Frequency Response of String Mass System at Micrometer Position = 1.60 mm.

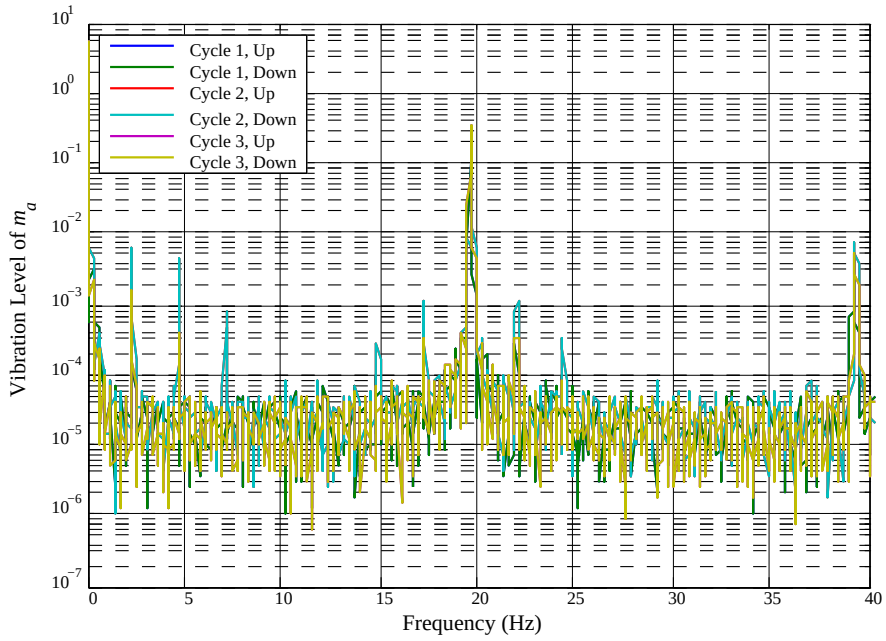


Figure 4.31. Frequency Response of String Mass System at Micrometer Position = 1.50 mm.

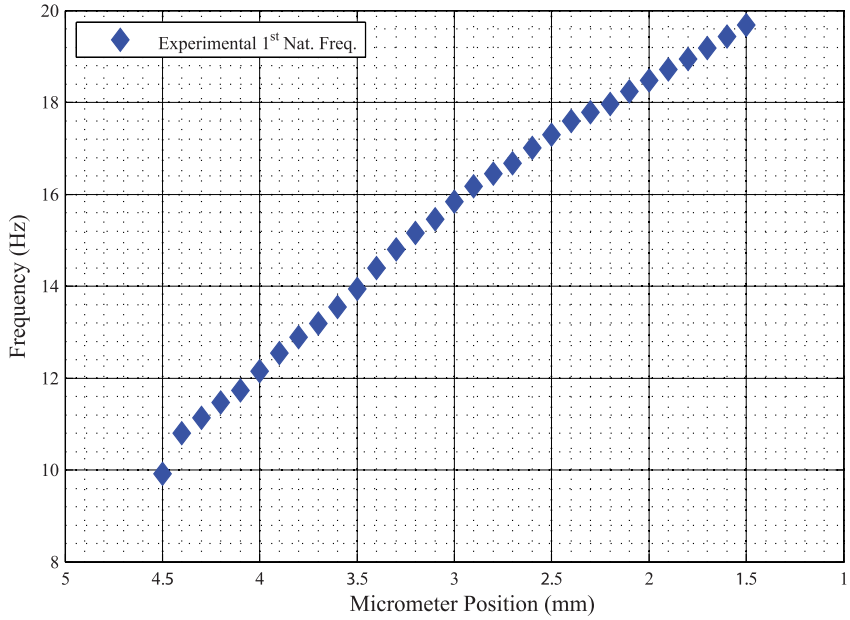


Figure 4.32. Experimental first natural frequencies of the string-mass system.

4.2. Determination of the Forced Vibration Behavior of the Primary System

In this part of the experiments the aim is to determine the frequency response of the primary structure without any DVA protection. The excitation is applied as a constant amplitude base motion. The frequency of the excitation is varied linearly over time. In the realization of the primary structure, the damping of it is kept as low as possible. Therefore, in a frequency sweep experiment like this, it is important to ensure that transient effects due to the light damping die out. A simple step input response of the primary structure is shown in Figure 4.34 to show the decaying vibrations of the system in time. By using the slowest sweep rate allowed by the shaker controller, 0.1 Hz/min, the steady state response of the system to the frequencies within the range is obtained.

The running RMS of acceleration data obtained from this test is given in Figure 4.36. One can see that the primary structure has a severe resonance within the range. Although this acceleration data is given as a function of time, from the frequency sweep rate and limits, the time variable can be converted into excitation frequency values.

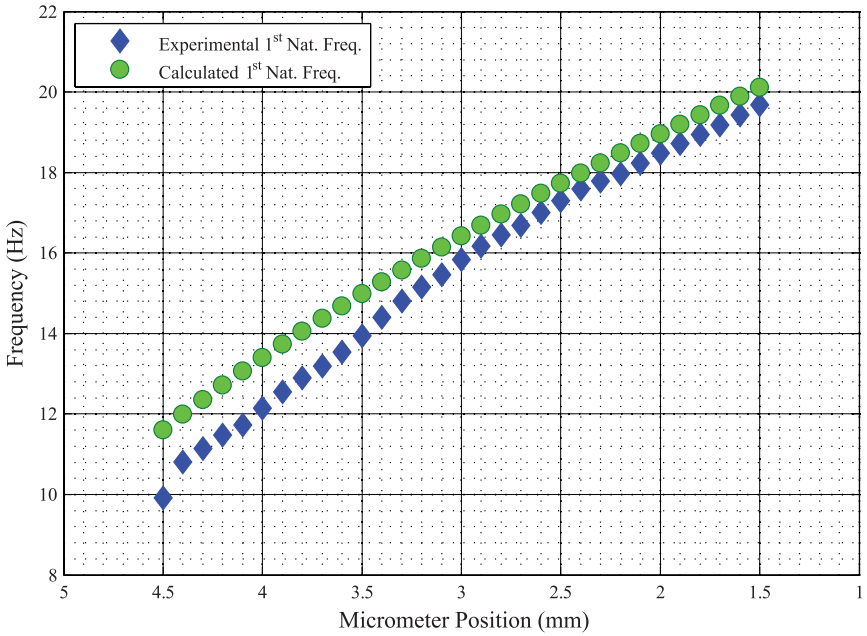


Figure 4.33. Calculated and experimental first natural frequencies of the string-mass system.

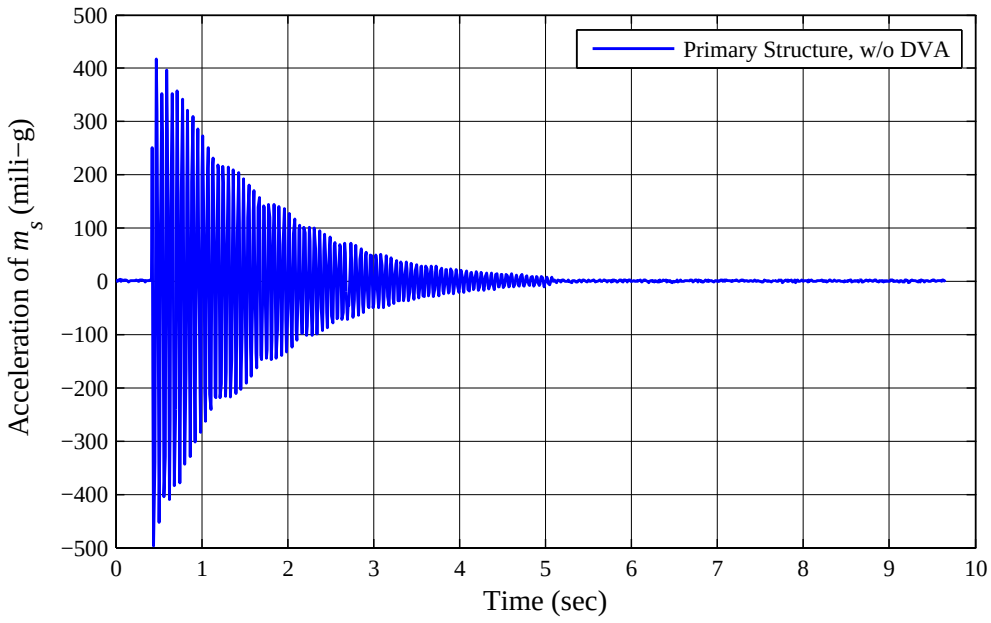


Figure 4.34. Decay of the vibration of the primary mass, after hit by a hammer.

In Figure 4.35, the RMS value of the acceleration of the primary structure is plotted against frequency by using this conversion. It is found that the primary structure's resonance frequency resides around 16.3 Hz.

In fact, the resonance of the primary structure is set to this frequency on purpose. Therefore, it can be shown that even around a severe resonance frequency, the DVA can effectively suppress vibrations of the primary structure.

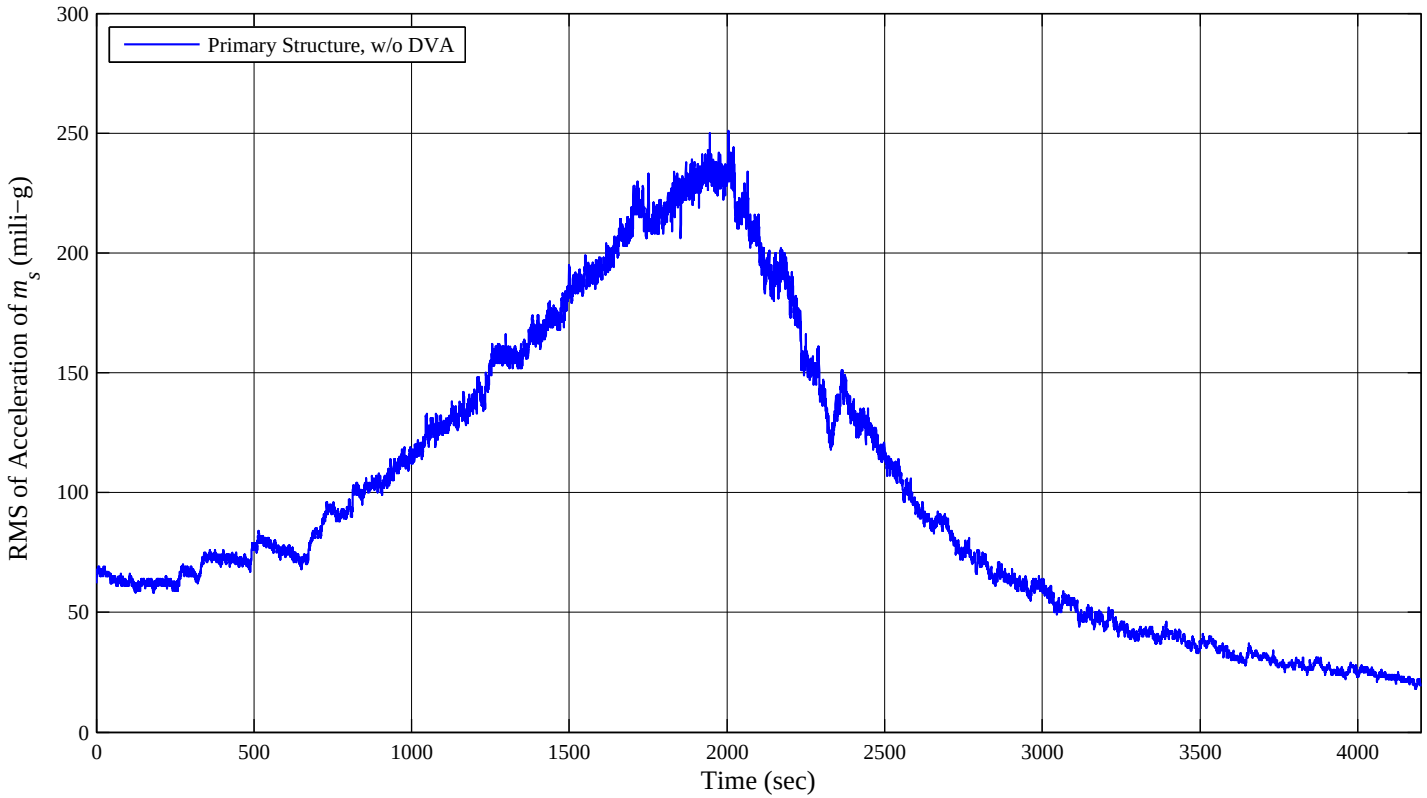


Figure 4.35. RMS of Acceleration History of the Primary Structure without the DVA During 0.075 mm Base Excitation Frequency Sweep between 13-20 Hz at 0.1 Hz/min.

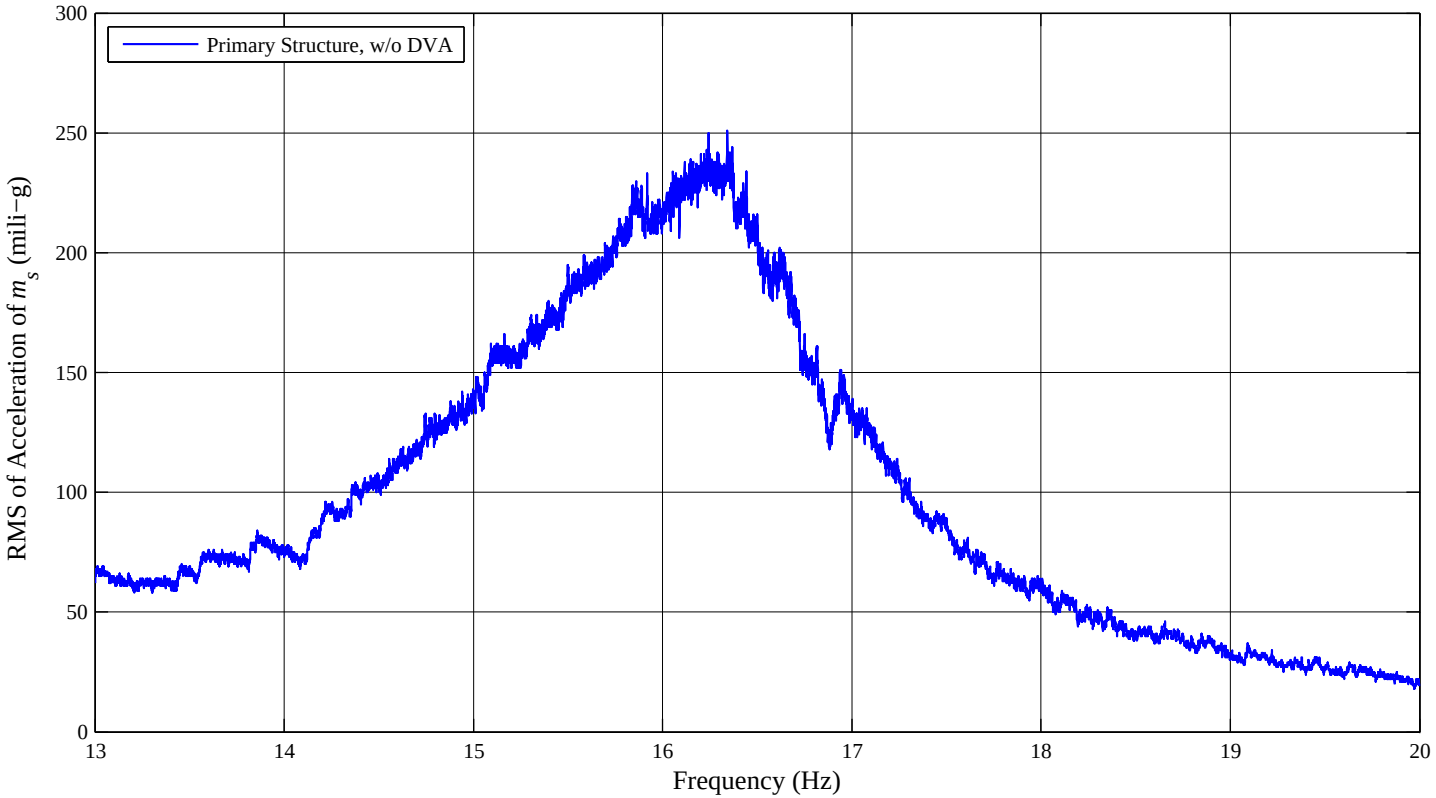


Figure 4.36. Frequency Response Function of the Primary Structure without the DVA.

4.2.1. Determination of the Forced Vibration Behavior of the Combined System without the Controller

In this part, the mechanical response of the primary structure when combined with the DVA system is evaluated for two different string tension values, namely for micrometer indication equals to 1.75 mm and 2.50 mm. The frequency sweep rate used in these tests is again 0.1 Hz/min. However, the fixed base motion amplitude is reduced to 0.05 mm.

Evaluating the system only at two string tensions is not adequate for obtaining the behavior of the system over the entire working range. However, in these tests the DVA mass experiences excess vibrations when the excitation coincides with the resonance frequencies of the combined system, which is not the case when the system is automatically tuned by the controller. Therefore, only for observing the antiresonance region created by the DVA, at two arbitrarily chosen string tensions the frequency response plots of the system are obtained.

In Figure 4.38 - 4.39, the frequency response of the system is plotted for the indicated micrometer positions. The two resonances and one antiresonance are clearly revealed in these plots. The rapid fluctuations around the higher resonance frequencies in both plots are not seperate resonances, rather they occurred during the frequency sweep operation when the DVA mass hit to the stopper used to limit the motion of the DVA. In these plots we see that for 1.75 mm setting, the resonance frequencies are around 16.3 Hz and 19.5 Hz while the antiresonance frequency is around 18.4 Hz, for 2.50 mm setting the resonances occur at 16.5 Hz and 19.4 Hz and the antiresonance is at 17.6 Hz.

The antiresonance frequencies obtained from these figures are in agreement with the natural frequency detection experiment results in Figure 4.33. On the other hand, to predict the theoretical positions of these resonance frequencies, the Equation 2.18 must be re-evaluated using the parameters of the experimental setup. First of all, we need to find the dynamically equivalent 2DOF simple mass spring system of our setup.

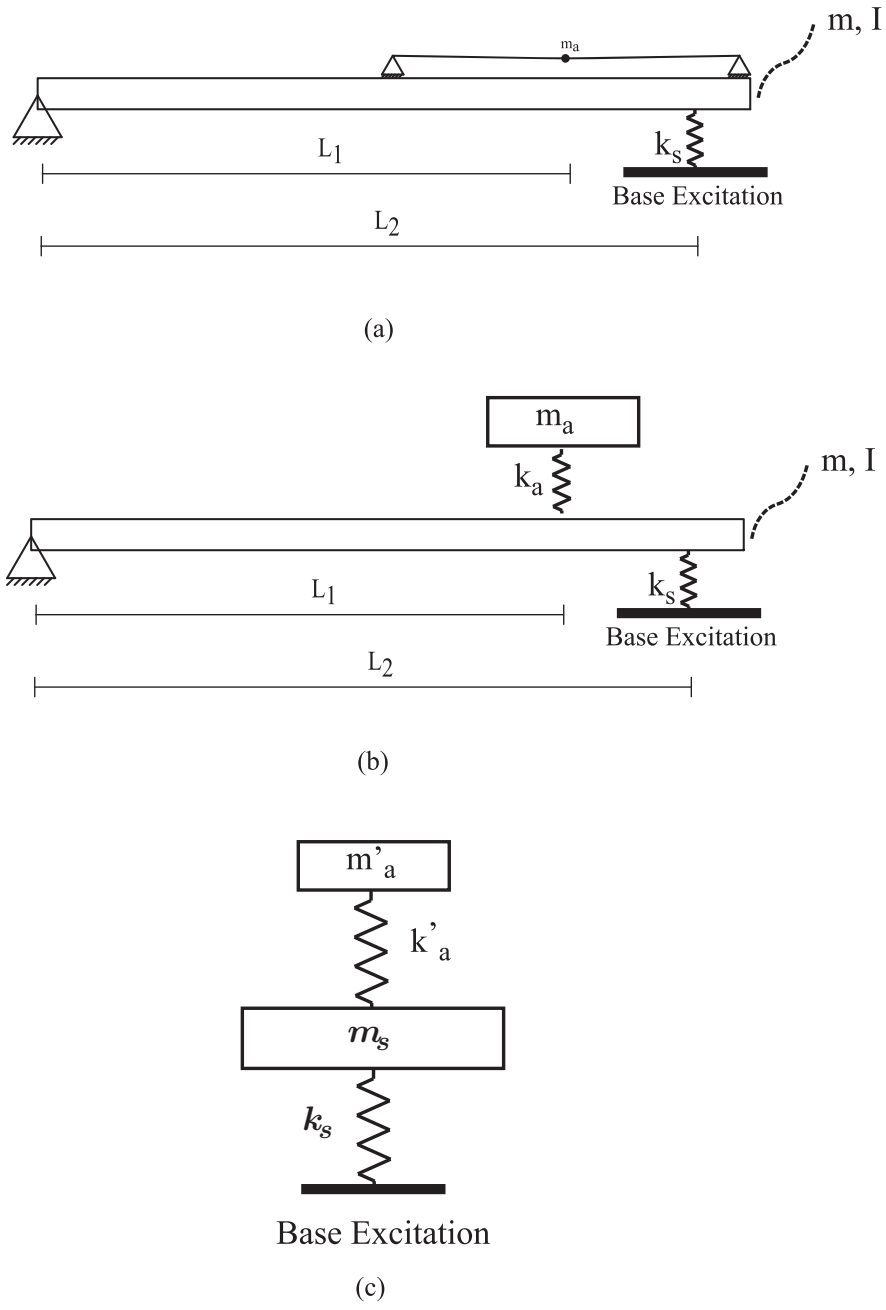


Figure 4.37. Dynamically equivalent 2DOF systems.

The systems illustrated in Figure 4.37a - 4.37c can be dynamically equivalent when

$$k_{\mathrm{a}}^{'} = k_{\mathrm{a}} \left(\frac{L_1}{L_2} \right)^2 \tag{4.1}$$

$$m_{\mathrm{a}}^{'} = m_{\mathrm{a}} \left(\frac{L_1}{L_2} \right)^2 . \tag{4.2}$$

In Equation 2.18, we know the values of ω_{a}, ω_{s} from the experiments. Furthermore, we know that $m_{\mathrm{a}} = 84.4$ g and $k_{\mathrm{s}} = 26.87$ N/mm. Thus, we can calculate μ as

$$\begin{aligned} \mu &= \frac{m_{\mathrm{a}}^{'}}{m_{\mathrm{s}}} \\ &= \frac{m_{\mathrm{a}}}{k_{\mathrm{s}}} \omega_{\mathrm{s}}^2 \left(\frac{L_1}{L_2} \right)^2 . \end{aligned} \tag{4.3}$$

From the above relation, it is found that the mass ratio, μ, of the prototype is 0.0224. By evaluating Equation 2.18 with these parameters it is found that, the analytically predicted resonance frequencies for the 1.75 mm case are 15.67 Hz and 19.12 Hz. Similarly for the 2.50 mm case, the predicted resonances are at 15.52 Hz and 18.42 Hz. Considering the limitations on experimentally obtaining these frequency response behavior of the system, the results are quite similar to the analytically calculated values.

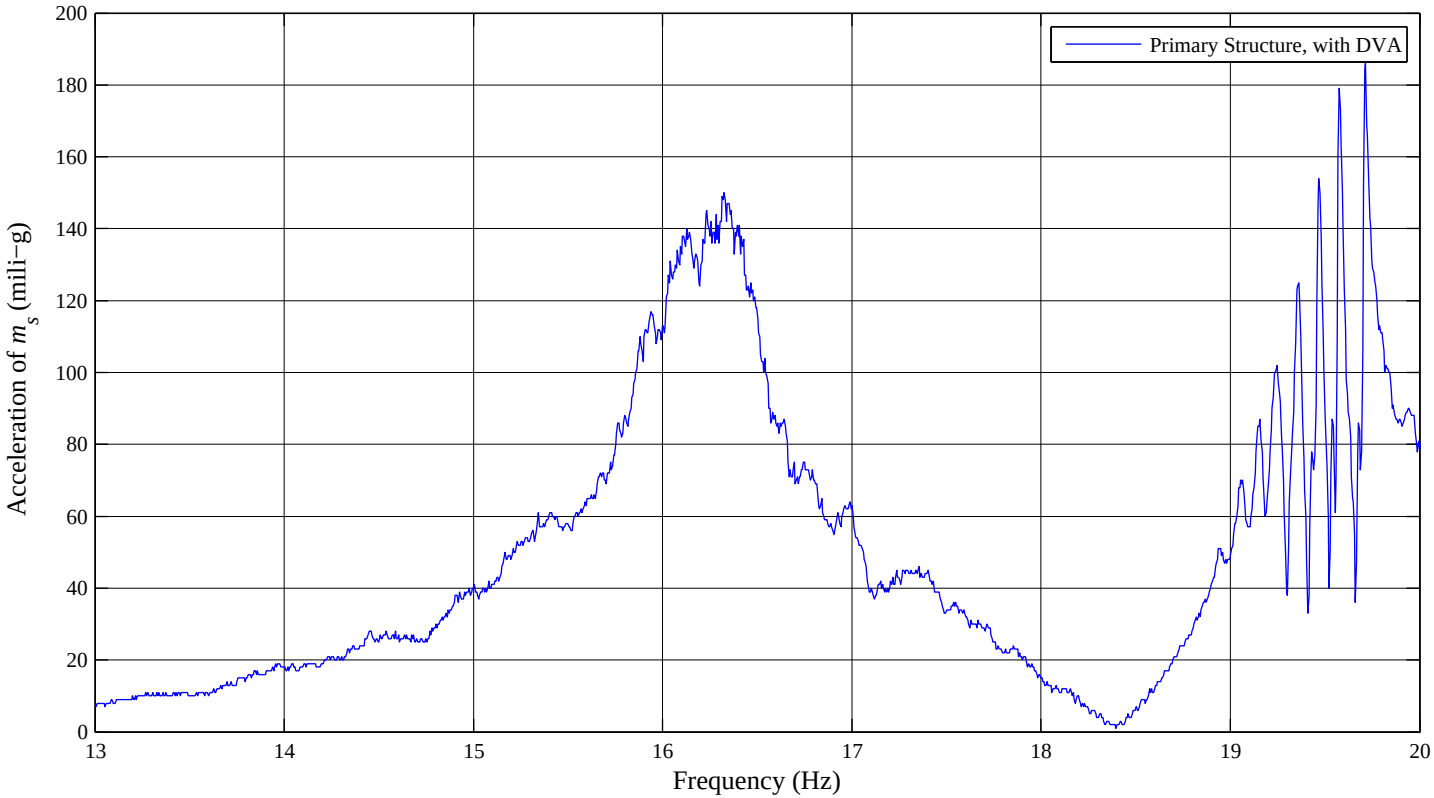


Figure 4.38. Frequency response function of the primary structure with passive DVA for micrometer position = 1.75 mm.

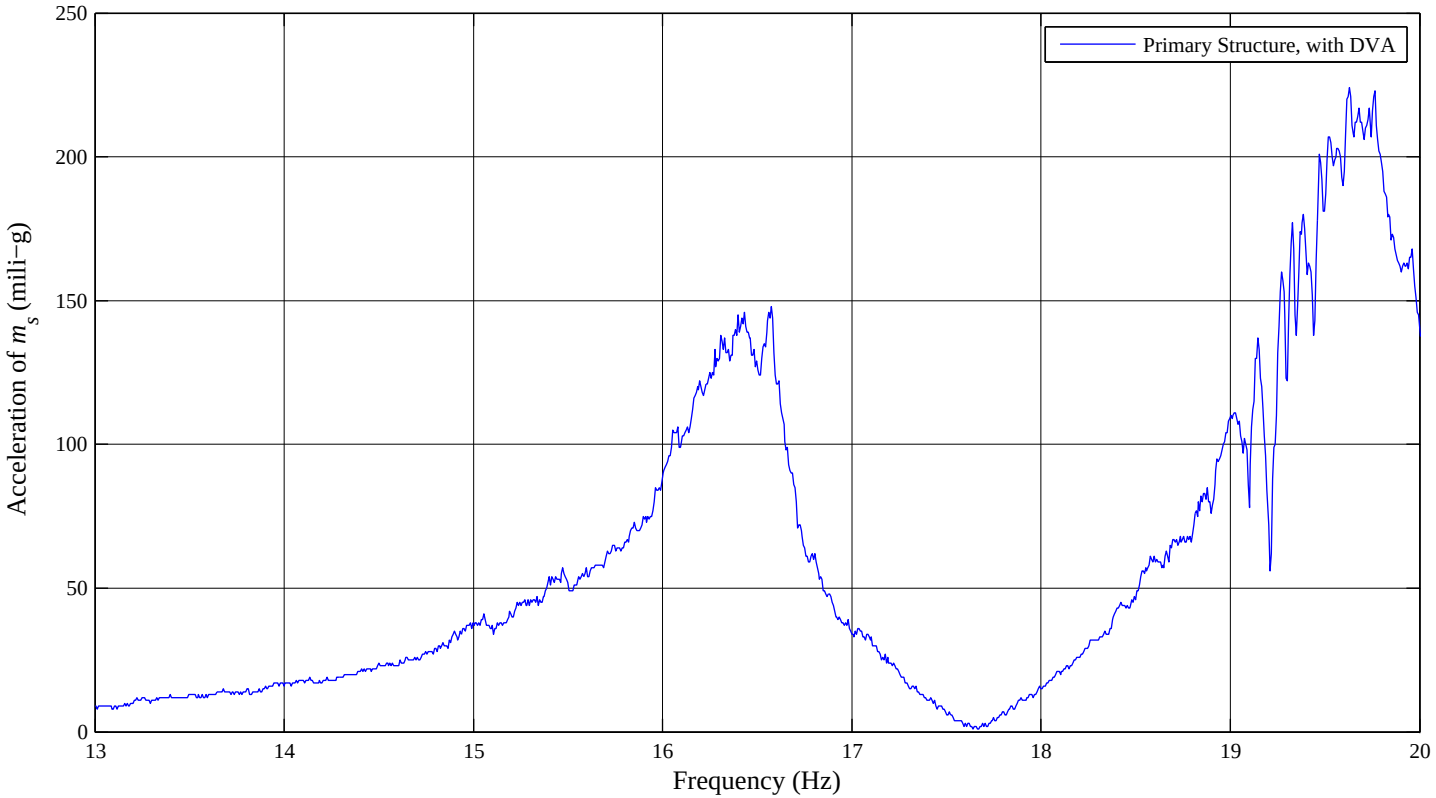


Figure 4.39. Frequency response function of the primary structure with passive DVA for micrometer position = 2.50 mm.

4.3. Determination of the Forced Vibration Behavior of the Combined System in Closed-Loop Mode

In this part of the experiment, the actual performance of the DVA system in adaptive mode is evaluated. When the system is operated in adaptive mode, many parameters like the variation characteristics of the excitation, initial tuning state; affect the response of the overall system. Thus, in this thesis, just two different procedures are determined for the purpose of evaluating the system performance.

The first tests conducted are the application of certain frequency sweep rates and amplitudes as done in the previous section. However, this time the limits of excitation frequencies are kept within 14-19 Hz for the first trial and 13-18 Hz for the second trial. The reason of this reduction in the operation range is that during initial trials, it was observed that at frequencies below 13 Hz and above 19 Hz, the resonance and antiresonance frequencies of the system become too close. Thus, the risk of resonance in case of a possible controller delay is not taken, at least for the sake of protection of the system throughout the experiments.

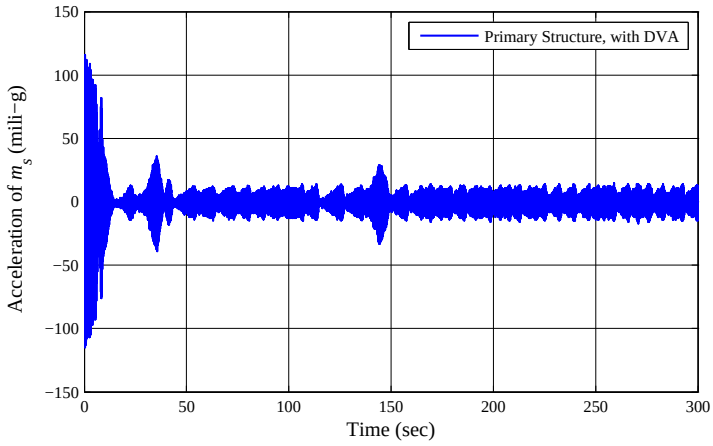


Figure 4.40. Acceleration History of the Primary Structure with the DVA During 0.075 mm Base Excitation Frequency Sweep between 14-19 Hz at 1 Hz/min.

The results obtained from these frequency sweep experiments are given in Figure

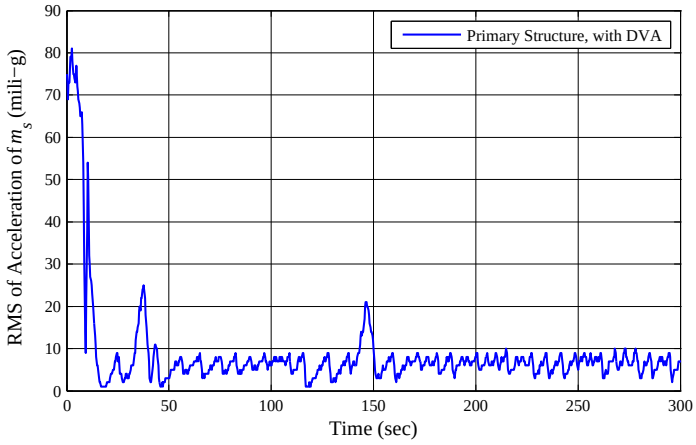


Figure 4.41. RMS of Acceleration History of the Primary Structure with the DVA During 0.075 mm Base Excitation Frequency Sweep between 14-19 Hz at 1 Hz/min.

4.40 - 4.42 and Figure 4.43 - 4.45. In these plots, the acceleration data, the running RMS values of the acceleration data and the position of the tuning actuator set by the controller acquired during the frequency sweep are given, respectively. For a more clear comparison, the acceleration data obtained from these two experiments are combined with the unprotected primary mass frequency response plot in Figure 4.46.

As can be seen in Figure 4.46, the vibration of the primary structure is dramatically reduced throughout the operation range except for the beginning of the test. This initial high levels of vibrations are experienced due the initial adaptation delay of the controller. A judgement about the initial adaptation period of the system can not be stated directly without considering the initial tuning state of the DVA system. In these two experiments, the initial tuning state of the system was at the upper end of the working range. Thus, the actuator had to travel the entire range of tunings to adapt itself to the excitation.

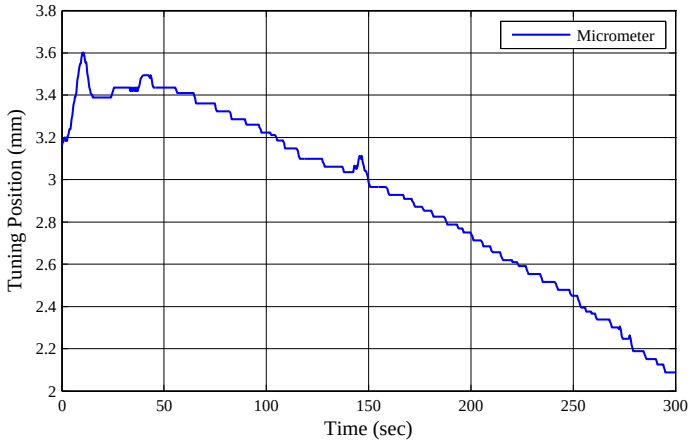


Figure 4.42. Tuning Position Determined by the Controller During 0.075 mm Base Excitation Frequency Sweep between 14-19 Hz at 1 Hz/min.

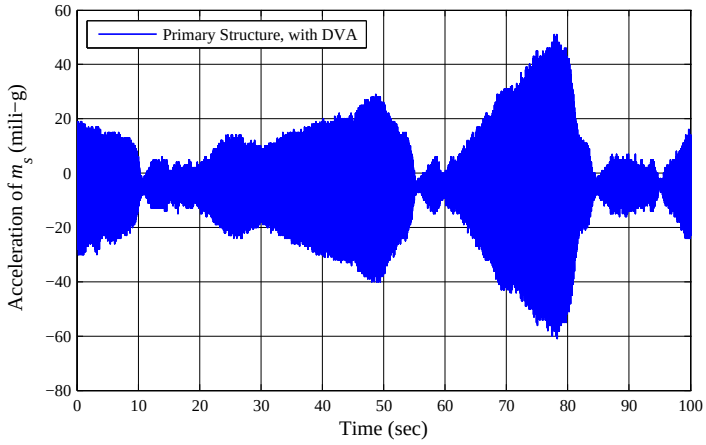


Figure 4.43. Acceleration History of the Primary Structure with the DVA During 0.05 mm Base Excitation Frequency Sweep between 13-18 Hz at 3 Hz/min.

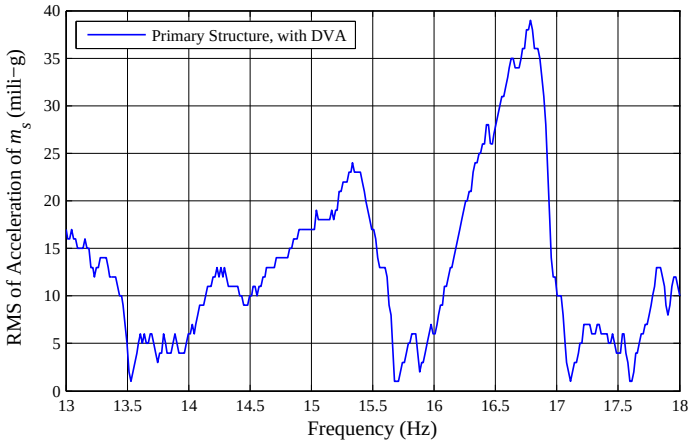


Figure 4.44. RMS of Acceleration History of the Primary Structure with the DVA During 0.05 mm Base Excitation Frequency Sweep between 13-18 Hz at 3 Hz/min.

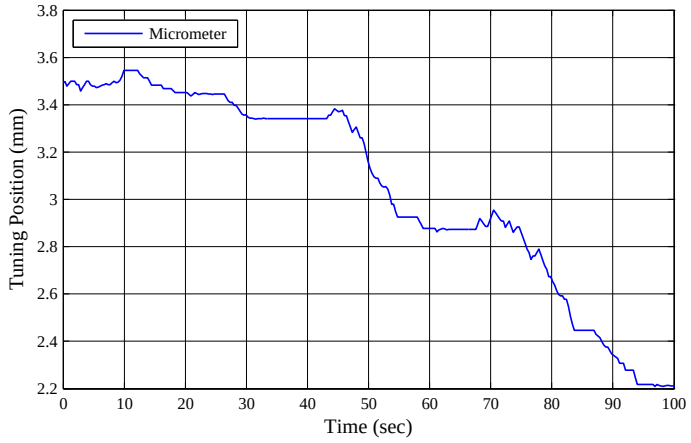


Figure 4.45. Tuning Position Determined by the Controller During 0.05 mm Base Excitation Frequency Sweep between 13-18 Hz at 3 Hz/min.

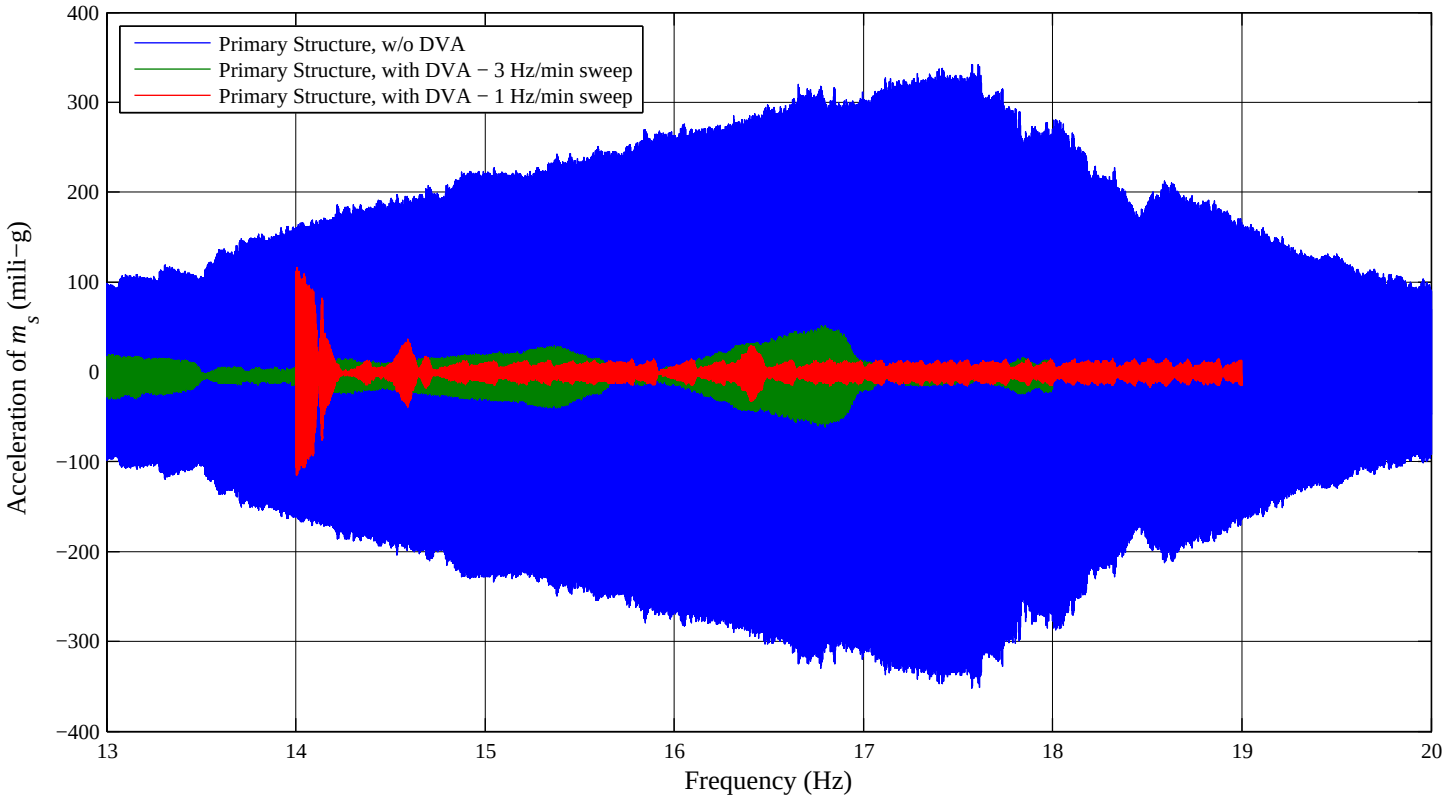


Figure 4.46. Acceleration Histories of the Primary Structure with/without the DVA at Similar Excitation States.

The second set of experiments in this part is the application of certain fixed frequencies to the combined system when it was at a mistuned position. The transitions between excitation frequencies are arbitrarily chosen within the range. In Figure 4.47, the time history of accelerations, RMS values and tuning positions are plotted over time. The indicated points in the plot are the excitation frequency change instants. The values of these points are given in Table 4.1. It can be observed that in each transition, the controller is able to adapt the system parameters and thus reduce the level of vibration significantly. However, as in the case of initial adaptation observed in the previous experiments, the adaptation time is relatively high. The major reason of the slow reaction of the system is the memory limitations in the tuning controller, which prevents more frequent calculations in frequency analysis. Therefore, the reaction time of the system increases.

Table 4.1. Arbitrarily varied excitation frequencies.

Marker	Tuned Frequency (Hz)	Applied Frequency (Hz)
A	19.00	15.00
B	15.00	16.00
C	16.00	17.00
D	17.00	14.00
E	14.00	18.00
F	18.00	15.50
G	15.50	17.00

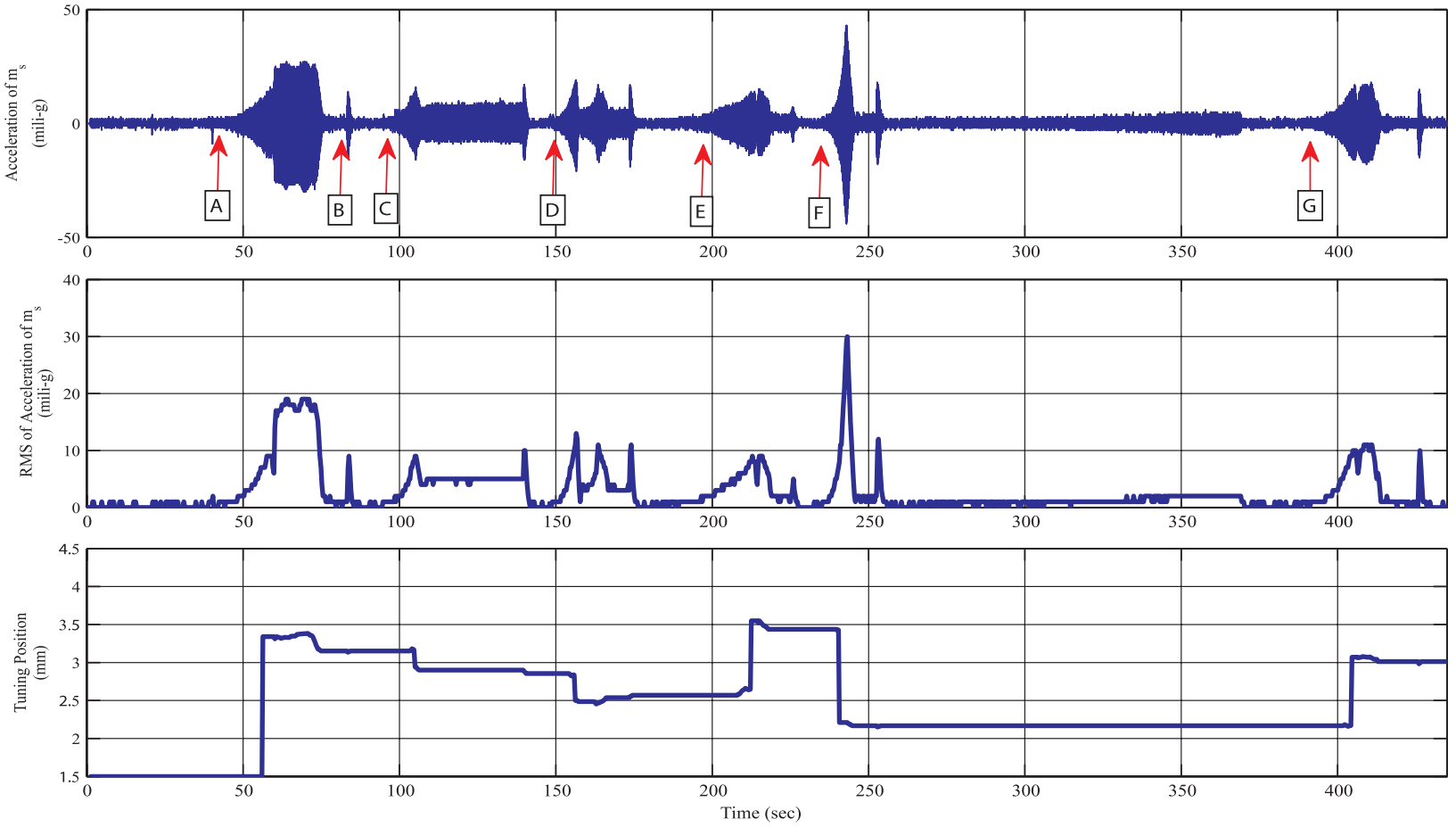


Figure 4.47. Response of the combined system in closed loop operation to arbitrary variations in excitation.

4.3.1. Actuator Power Consumption

In the experiments, it is observed that the power supply connected to the stepper motor driver delivers about 20 mA at 32 V during tuning actions. Thus, the actuator comsumes at most about 0.7 Watts. Moreover, the controller algorithm is designed such that, when a positioning action is completed, the current levels are dropped such that the consumption reduces to 0.1 Watts.

These power consumptions may seem high for the scale of the system. However, most stepper motors exhibit a detent torque, the minimum torque required to move the rotor [55]. In the preliminary tests conducted during the production of the actuator setup, it was observed that, when the motor power is reduced below 0.5 Watts the motor was barely rotating even at no load cases. So, we can roughly say that the power consumption of the actuator system to alter the string tension is about 0.2 Watts. Therefore, we can comment that by using the negative stiffness mechanism, it is possible to operate the tuning system with very low energy consumption. So, one can easily increase the scale of the system with a similar actuator used in this system, and still can run the tuning system with similar consumptions.

On the other hand, if another type of actuator that does not have an minimum torque requirement was used in the system, such as a linear piezo electric actuator, the power consumption could be reduced much more than the current system.

5. CONCLUSIONS

In this thesis, a new adaptive-passive dynamic vibration absorber is developed. Mainly this new design utilizes a tensioned string attached with a concentrated mass as a tunable dynamic vibration absorber.

The foremost property of an adaptive-passive vibration absorber is the tunability of the dynamic properties of the DVA system during operation. The string tension is used in this design as the tuning parameter.

Reducing the time and the effort required for adaptation process of the system to varying excitation conditions was among the main goals of the thesis. Using the string tension as a tuning parameter enables variation of dynamic behavior with very small displacement inputs. Thus, a sensitive system to control input is obtained. In order to reduce the adaptation effort without sacrificing this sensitivity of the tuning system, a negative stiffness mechanism is employed. With the help of this mechanism, the force output requirement from the tuning actuator can be reduced almost to zero.

During the course of the thesis, first, the dynamical behavior of the string-mass system is analyzed. The natural frequencies of the system as a function of the string tension are calculated both analytically, and using finite element method. The analytical solution agrees well with the FEM results.

After validating the FEM model with analytical solution, a forced vibration analysis of the DVA in combination with a primary structure is conducted. In this analysis, the antiresonance frequency at the natural frequency of the vibration absorber is clearly shown in the frequency response function of the combined system. Moreover, the shift in the antiresonance frequency with varying string-tension is demonstrated in the numerical model results.

In addition to the dynamical analyses, the force-deflection behavior of a pre-loaded spring with a rigid link mechanism is analyzed with the intention of obtaining a negative stiffness mechanism. It was shown that for certain configurations, a constant negative stiffness, i.e. linear force-deflection curve with negative slope, mechanism can be built with this approach.

In order to obtain an adaptive system from this tunable dynamic system, a controller is designed. Since modularity was among the design goals of the thesis, a microcontroller based control system is developed. The system consists of an accelerometer, a digital signal processing microcontroller and a stepper motor driver system. A tuning algorithm, which determines the string tension, depending on the amplitude and frequency of the vibration of the protected system and using a look-up table calibration data, is implemented in this microcontroller.

To evaluate the validity of the design, a prototype of the system is constructed. During the prototype design, a modular, precise system is aimed. The steps followed during production of the string-mass system, the negative stiffness mechanism and the tuning actuator system is explained in detail in the thesis.

Finally, a series of experiments are conducted with the intentions of detecting the natural frequency the string-mass system as a function of string tension, obtaining the forced vibration response of the unprotected structure, obtaining vibration suppression and adaptation performances of the protected system, evaluation of the energy consumption of the tuning actuator. In these experiments it was shown that the natural frequency calculation methods discussed in the theoretical background chapter match with the experimental findings.

In the forced vibration tests, it was shown that the designed system can dramatically reduce the vibration level of the protected structure over the range of operation frequencies. Moreover, the ability of the tuning controller to track the varying excitation conditions is validated in these experiments. However, due to some trade-offs imposed by the design goals in the prototype, the controller tracking performance was

not excellent. First of all, the memory limitation in the tuning algorithm forced us a relatively conservative algorithm development with long adjustment periods. If the modularity was a lesser goal in the system, the tuning algorithm could be much more effective on a computer.

Last but not least, it is observed that the tuning actuator could vary the string tension by drawing 0.7 Watts while it is adjusting the string tension. Despite the considerably high detent torque of the stepper motor, the system can operate near no load consumption of the actuator system with the help of the negative stiffness mechanism.

To conclude, in this thesis, the proposed design is accurately analyzed, and most of the design goals are met experimentally, except the adaptation speed of the tuning controller. With a better controller implementation, the system's responsiveness potential could be better revealed. In summary, design of a new vibration protection building block, that can be compact, fast and energy efficient, is thoroughly discussed in this thesis.

APPENDIX A: THE VIBRATION MEASUREMENT BOARD DOCUMENTS

A.1. Circuit Schematics

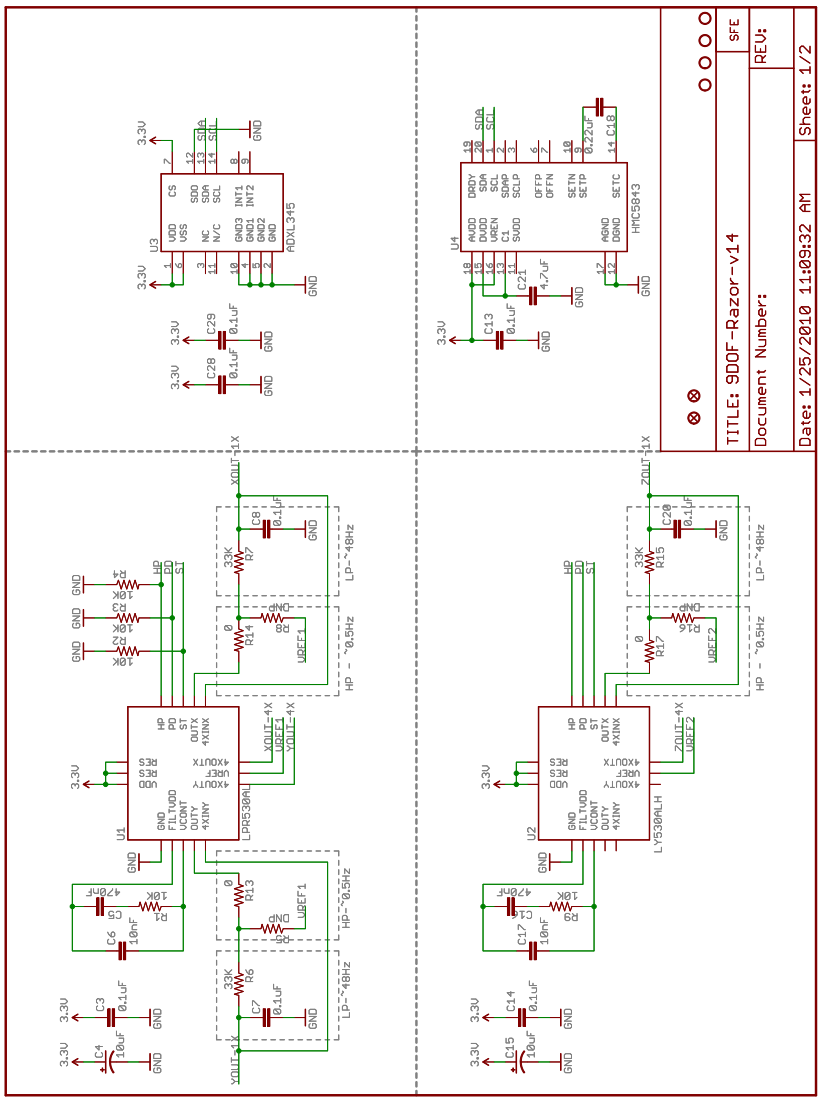


Figure A.1. Circuit schematic of the 9-DOF Razor IMU Board - Part 1 [64].

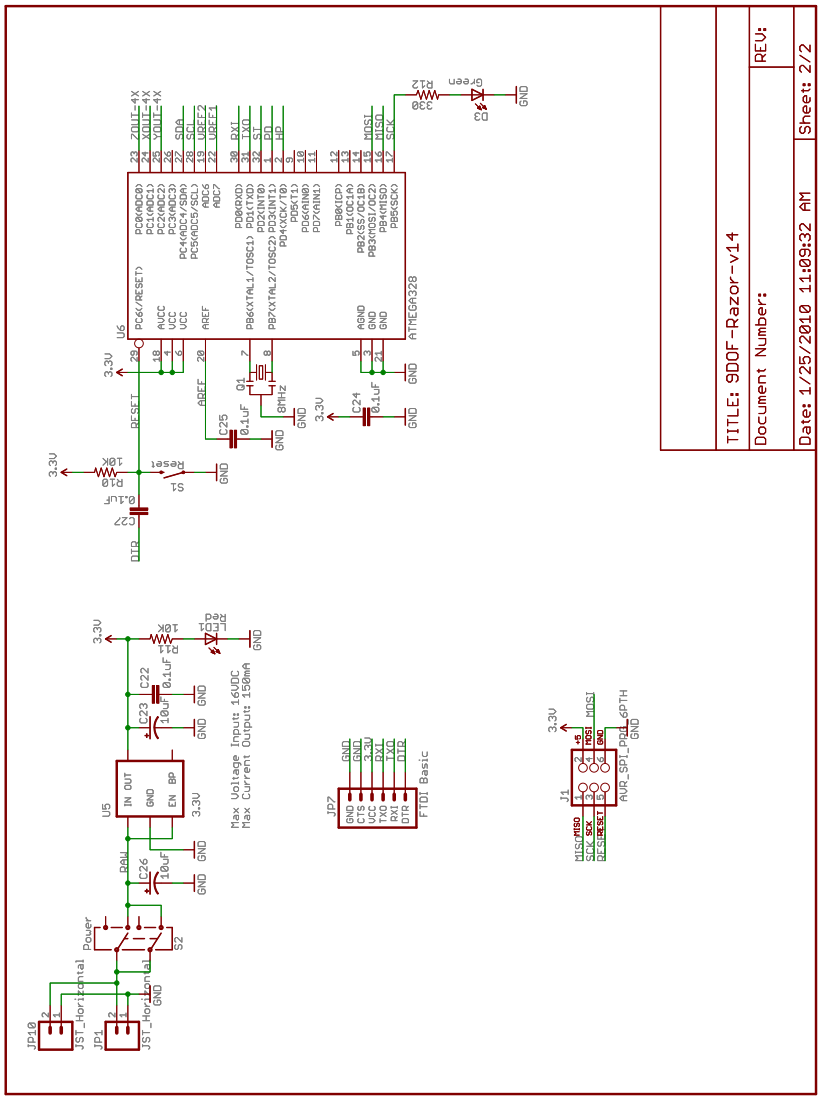


Figure A.2. Circuit schematic of the 9-DOF Razor IMU Board - Part 2 [64].

A.2. Datasheets

Features

- High Performance, Low Power Atmel®AVR® 8-Bit Microcontroller
- Advanced RISC Architecture
  - 131 Powerful Instructions – Most Single Clock Cycle Execution
  - 32 x 8 General Purpose Working Registers
  - Fully Static Operation
  - Up to 20 MIPS Throughput at 20MHz
  - On-chip 2-cycle Multiplier
- High Endurance Non-volatile Memory Segments
  - 4/8/16/32KBytes of In-System Self-Programmable Flash program memory
  - 256/512/512/1KBytes EEPROM
  - 512/1K/1K/2KBytes Internal SRAM
  - Write/Erase Cycles: 10,000 Flash/100,000 EEPROM
  - Data retention: 20 years at 85°C/100 years at 25°C(1)
  - Optional Boot Code Section with Independent Lock Bits
    - In-System Programming by On-chip Boot Program
    - True Read-While-Write Operation
  - Programming Lock for Software Security
- Atmel® QTouch® library support
  - Capacitive touch buttons, sliders and wheels
  - QTouch and QMatrix® acquisition
  - Up to 64 sense channels
- Peripheral Features
  - Two 8-bit Timer/Counters with Separate Prescaler and Compare Mode
  - One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode
  - Real Time Counter with Separate Oscillator
  - Six PWM Channels
  - 8-channel 10-bit ADC in TQFP and QFN/MLF package
    - Temperature Measurement
  - 6-channel 10-bit ADC in PDIP Package
    - Temperature Measurement
  - Programmable Serial USART
  - Master/Slave SPI Serial Interface
  - Byte-oriented 2-wire Serial Interface (Philips I^2C compatible)
  - Programmable Watchdog Timer with Separate On-chip Oscillator
  - On-chip Analog Comparator
  - Interrupt and Wake-up on Pin Change
- Special Microcontroller Features
  - Power-on Reset and Programmable Brown-out Detection
  - Internal Calibrated Oscillator
  - External and Internal Interrupt Sources
  - Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby, and Extended Standby
- I/O and Packages
  - 23 Programmable I/O Lines
  - 28-pin PDIP, 32-lead TQFP, 28-pad QFN/MLF and 32-pad QFN/MLF
- Operating Voltage:
  - 1.8 - 5.5V
- Temperature Range:
  - -40°C to 85°C
- Speed Grade:
  - 0 - 4MHz@1.8 - 5.5V, 0 - 10MHz@2.7 - 5.5.V, 0 - 20MHz @ 4.5 - 5.5V
- Power Consumption at 1MHz, 1.8V, 25°C
  - Active Mode: 0.2mA
  - Power-down Mode: 0.1µA
  - Power-save Mode: 0.75µA (Including 32kHz RTC)

8-bit Atmel Microcontroller with 4/8/16/32K Bytes In-System Programmable Flash

ATmega48A
ATmega48PA
ATmega88A
ATmega88PA
ATmega168A
ATmega168PA
ATmega328
ATmega328P

Summary

Rev. 8271DS–AVR–05/11

Figure A.3. Datasheet summary of the ATMega328P Microcontroller [65].

ANALOG DEVICES

3-Axis, ±2 *g*/±4 *g*/±8 *g*/±16 *g* Digital Accelerometer

ADXL345

FEATURES

Ultralow power: as low as 40 µA in measurement mode and 0.1 µA in standby mode at V_S = 2.5 V (typical)
Power consumption scales automatically with bandwidth
User-selectable resolution
Fixed 10-bit resolution
Full resolution, where resolution increases with *g* range, up to 13-bit resolution at ±16 *g* (maintaining 4 mg/LSB scale factor in all *g* ranges)
Embedded, patent pending FIFO technology minimizes host processor load
Tap/double tap detection
Activity/inactivity monitoring
Free-fall detection
Supply voltage range: 2.0 V to 3.6 V
I/O voltage range: 1.7 V to V_S
SPI (3- and 4-wire) and I^2C digital interfaces
Flexible interrupt modes mappable to either interrupt pin
Measurement ranges selectable via serial command
Bandwidth selectable via serial command
Wide temperature range (−40°C to +85°C)
10,000 *g* shock survival
Pb free/RoHS compliant
Small and thin: 3 mm × 5 mm × 1 mm LGA package

APPLICATIONS

Handsets
Medical instrumentation
Gaming and pointing devices
Industrial instrumentation
Personal navigation devices
Hard disk drive (HDD) protection
Fitness equipment

GENERAL DESCRIPTION

The ADXL345 is a small, thin, low power, 3-axis accelerometer with high resolution (13-bit) measurement at up to ±16 *g*. Digital output data is formatted as 16-bit twos complement and is accessible through either a SPI (3- or 4-wire) or I^2C digital interface.

The ADXL345 is well suited for mobile device applications. It measures the static acceleration of gravity in tilt-sensing applications, as well as dynamic acceleration resulting from motion or shock. Its high resolution (4 mg/LSB) enables measurement of inclination changes less than 1.0°.

Several special sensing functions are provided. Activity and inactivity sensing detect the presence or lack of motion and if the acceleration on any axis exceeds a user-set level. Tap sensing detects single and double taps. Free-fall sensing detects if the device is falling. These functions can be mapped to one of two interrupt output pins. An integrated, patent pending 32-level first in, first out (FIFO) buffer can be used to store data to minimize host processor intervention.

Low power modes enable intelligent motion-based power management with threshold sensing and active acceleration measurement at extremely low power dissipation.

The ADXL345 is supplied in a small, thin, 3 mm × 5 mm × 1 mm, 14-lead, plastic package.

FUNCTIONAL BLOCK DIAGRAM

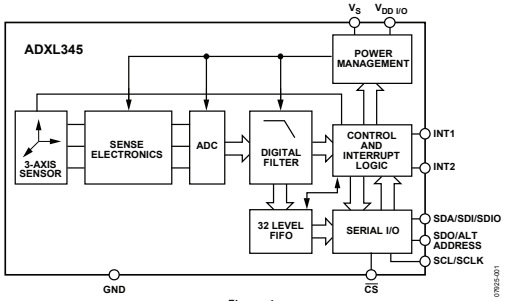


Figure 1.

One Technology Way, P.O. Box 9106, Norwood, MA 02062-9106, U.S.A.
Tel: 781.329.4700 **www.analog.com**
Fax: 781.461.3113

Figure A.4. Datasheet summary of the ADXL345 Accelerometer [65].

APPENDIX B: FIRMWARE OF THE VIBRATION MEASUREMENT BOARD

```
#define F_CPU 8000000UL
#include <stdlib.h>
#include <stdio.h>
#include <avr/io.h>
#include <util/delay.h>
#include "types.h"
#include "defs.h"
#include "i2c.h"


/************MACROS***************/
//#define STATUS_LED 5 //stat LED is on PB5

#define sbi(var, mask)   ((var) |= (uint8_t)(1 << mask))
#define cbi(var, mask)   ((var) &= (uint8_t)~(1 << mask))

#define WRITE_sda() DDRC = DDRC | 0b00010000
#define READ_sda()  DDRC = DDRC & 0b11101111

//#define   x   (*((int16_t *)0x110))


///============Initialize Prototypes=====/////////////////
void init(void);
void timer1_init(void);

void SPI_MasterInit(void);
void SPI_MasterTransmit(uint8_t cDataNr, int16_t cData);

void write_to_accelerometer(unsigned short reg_addr,
unsigned short reg_data,
unsigned char dataFlag);
void accelerometer_init(void);
int16_t accelerometer_read_multibyte(void);

void i2cInit(void);
void i2cSendStart(void);
void i2cSendStop(void);
void i2cWaitForComplete(void);
void i2cSendByte(unsigned char data);
void i2cHz(long uP_F, long scl_F);
```

```
/***Global Variables**/
volatile unsigned char time_to_read_accelerometer = 1;

uint16_t i = 0;
uint8_t sample_nr = 0;
struct sample_mode
{
uint8_t BW_setting;
uint8_t Timer_Reg;
uint16_t Timer_Compare;
} ;


struct sample_mode sample_modes[]
= {0b00001101, (1<<WGM12)|(1<<CS10), 10000, // 800Hz
   0b00001100, (1<<WGM12)|(1<<CS10), 20000, // 400Hz
   0b00001011, (1<<WGM12)|(1<<CS10), 40000, // 200Hz
   0b00001010, (1<<WGM12)|(1<<CS11), 10000, // 100Hz
   0b00001001, (1<<WGM12)|(1<<CS11), 20000 // 50Hz
};

uint8_t sample_mode_nr = 3;

/*******************************
 ****Interrupts*****************
 *******************************/

ISR(TIMER1_COMPA_vect)  //executes at the end of
    { //the measurement period
    time_to_read_accelerometer = 1;
    }


/////===========MAIN=====================////////////////////
int main(void)
    {
    //1 = output, 0 = input
    DDRB = 0b01101100; //PORTB5, B5 output for stat LED
    DDRC = 0b00010000;  //PORTC4 (SDA),
     //PORTC5 (SCL),
     //PORTC all others are inputs
    DDRD = 0b00000010; //PORTD (TX output on PD1)
    PORTC = 0b00110000; //pullups on the I2C bus
    cbi(UCSR0B, TXEN0);


    sbi(PORTD, PIND1);
    cbi(PORTB, PINB5);
```

```
    SPI_MasterInit();

    i2cInit();
    accelerometer_init();
    timer1_init();
    sei();

    while(1)
{
if(time_to_read_accelerometer == 1)
     {
SPI_MasterTransmit( sample_nr,
accelerometer_read_multibyte());
     sample_nr = (sample_nr + 1) & 0xFF;
     time_to_read_accelerometer = 0;
     }
}
}


/****Timer0 Initialize***********/
void timer1_init(void)
    {
    TCCR1B = sample_modes[sample_mode_nr].Timer_Reg;
    OCR1A  = sample_modes[sample_mode_nr].Timer_Compare;
    TIMSK1 = (1<<OCIE1A);
    }

/*****************************
 **************SPI Functions****
 *****************************/
void SPI_MasterInit(void)
    {
    SPSR |= (1<<SPI2X);
    SPCR =  (1<<SPE)|(1<<MSTR)|(1<<SPR0)|
     (0<<SPR1)|(0<<CPHA)|(0<<CPOL)|(0<<DORD);
    }

void SPI_MasterTransmit(uint8_t cDataNr, int16_t cData)
    {
    cbi(PORTD, PIND1);
    SPDR = 0x7F;
    while(!(SPSR & (1<<SPIF)));
    SPDR = cDataNr;
    while(!(SPSR & (1<<SPIF)));
    SPDR = (cData >> 8) & 0xFF;
    while(!(SPSR & (1<<SPIF)));
    SPDR = (cData     ) & 0xFF;
```

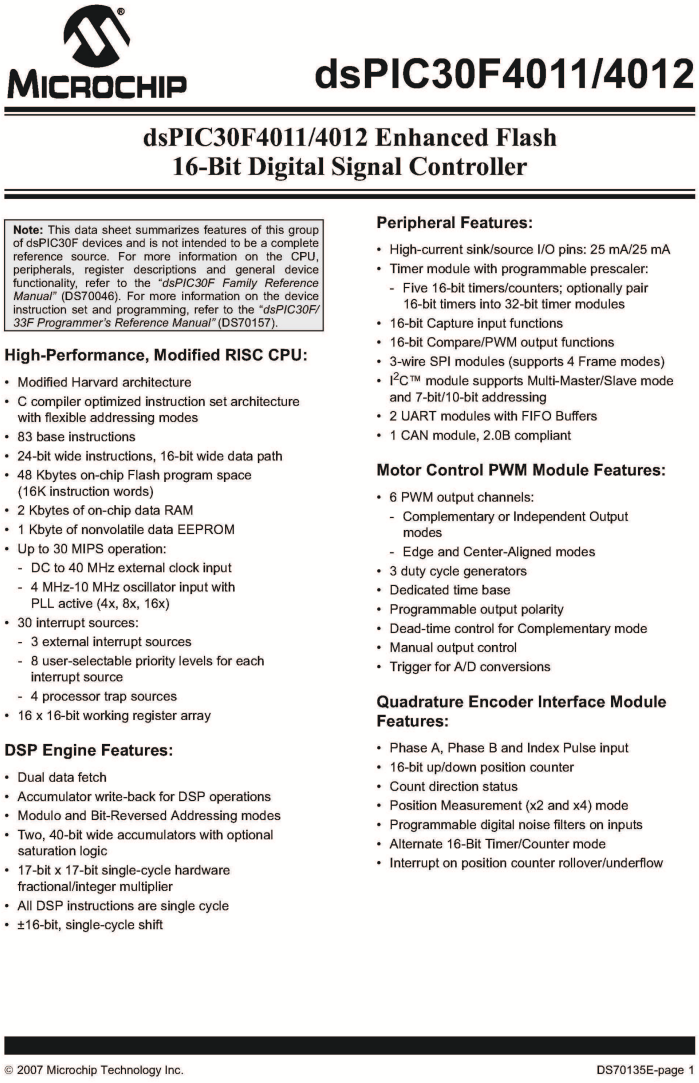
```
    while(!(SPSR & (1<<SPIF)));
    sbi(PORTD, PIND1);
    }

/*******************************
 ****Accelerometer Functions****
 *******************************/
void write_to_accelerometer(unsigned short reg_addr,
unsigned short reg_data,
unsigned char dataFlag)
    {
    i2cSendStart();
    i2cWaitForComplete();
    i2cSendByte(0xA6);    //write to ADXL
    i2cWaitForComplete();
    i2cSendByte(reg_addr);
    i2cWaitForComplete();

    if( dataFlag )
{
i2cSendByte(reg_data);
i2cWaitForComplete();
}
    i2cSendStop();
    }

void accelerometer_init(void)
    {
    //initialize
write_to_accelerometer( 0x2C,
sample_modes[sample_mode_nr].BW_setting,
TRUE);  //BR_RATE: 100Hz Data Rate;
//50Hz Bandwidth
write_to_accelerometer(0x2D, 0x08, TRUE);
//POWER_CTL: measurement mode

write_to_accelerometer(0x31, 0b00001001, TRUE);
//data format: full resolution (13bit),
// +-4g range,
//4mg/LSB
    }

int16_t accelerometer_read_multibyte(void)
    {
    uint8_t low_byte, high_byte;

    //0x32 data registers
    write_to_accelerometer(0x36, 0, FALSE); //Z0 data register
```

```
    i2cSendStart(); //repeat start
    i2cWaitForComplete();
    i2cSendByte(0xA7);    //read from ADXL
    i2cWaitForComplete();

    i2cReceiveByte(TRUE);
    i2cWaitForComplete();
    low_byte = i2cGetReceivedByte(); //
    i2cWaitForComplete();
    i2cReceiveByte(FALSE);
    i2cWaitForComplete();
    high_byte = i2cGetReceivedByte(); //
    i2cWaitForComplete();
    i2cSendStop();
    return ((high_byte << 8) | low_byte);
    }

/*********************
 ****I2C Functions****
 *********************/

void i2cInit(void)
    {
    // set i2c bit rate to 400KHz
    i2cSetBitrate();
    // enable TWI (two-wire interface)
    sbi(TWCR, TWEN);
    }

void i2cSetBitrate()
    {
    cbi(TWSR, TWPS0);
    cbi(TWSR, TWPS1);
    TWBR = 2;
    }

void i2cSendStart(void)
    {
    WRITE_sda();
    // send start condition
    TWCR = (1<<TWINT)|(1<<TWSTA)|(1<<TWEN);
    }

void i2cSendStop(void)
    {
    // transmit stop condition
    TWCR = (1<<TWINT)|(1<<TWEN)|(1<<TWSTO);
```

```
    }

void i2cWaitForComplete(void)
    {
    int i = 0; //time out variable

    // wait for i2c interface to complete operation
    while ((!(TWCR & (1<<TWINT))) && (i < 90))
i++;
    }

void i2cSendByte(unsigned char data)
    {

    WRITE_sda();
    // save data to the TWDR
    TWDR = data;
    // begin send
    TWCR = (1<<TWINT)|(1<<TWEN);
    }

void i2cReceiveByte(unsigned char ackFlag)
    {
    // begin receive over i2c
    if( ackFlag )
{
// ackFlag = TRUE: ACK the recevied data
outb(TWCR, (inb(TWCR)&TWCR_CMD_MASK)|BV(TWINT)|BV(TWEA));
}
    else
{
// ackFlag = FALSE: NACK the recevied data
outb(TWCR, (inb(TWCR)&TWCR_CMD_MASK)|BV(TWINT));
}
    }

unsigned char i2cGetReceivedByte(void)
    {
    // retrieve received data byte from i2c TWDR
    return( inb(TWDR) );
    }

unsigned char i2cGetStatus(void)
    {
    // retieve current i2c status from i2c TWSR
    return( inb(TWSR) );
    }
```

APPENDIX C: TUNING MICROCONTROLLER DOCUMENTS

C.1. Datasheets



MICROCHIP

dsPIC30F4011/4012

dsPIC30F4011/4012 Enhanced Flash 16-Bit Digital Signal Controller

Note: This data sheet summarizes features of this group of dsPIC30F devices and is not intended to be a complete reference source. For more information on the CPU, peripherals, register descriptions and general device functionality, refer to the *"dsPIC30F Family Reference Manual"* (DS70046). For more information on the device instruction set and programming, refer to the *"dsPIC30F/33F Programmer's Reference Manual"* (DS70157).

High-Performance, Modified RISC CPU:

- Modified Harvard architecture
- C compiler optimized instruction set architecture with flexible addressing modes
- 83 base instructions
- 24-bit wide instructions, 16-bit wide data path
- 48 Kbytes on-chip Flash program space (16K instruction words)
- 2 Kbytes of on-chip data RAM
- 1 Kbyte of nonvolatile data EEPROM
- Up to 30 MIPS operation:
  - DC to 40 MHz external clock input
  - 4 MHz-10 MHz oscillator input with PLL active (4x, 8x, 16x)
- 30 interrupt sources:
  - 3 external interrupt sources
  - 8 user-selectable priority levels for each interrupt source
  - 4 processor trap sources
- 16 x 16-bit working register array

DSP Engine Features:

- Dual data fetch
- Accumulator write-back for DSP operations
- Modulo and Bit-Reversed Addressing modes
- Two, 40-bit wide accumulators with optional saturation logic
- 17-bit x 17-bit single-cycle hardware fractional/integer multiplier
- All DSP instructions are single cycle
- ±16-bit, single-cycle shift

Peripheral Features:

- High-current sink/source I/O pins: 25 mA/25 mA
- Timer module with programmable prescaler:
  - Five 16-bit timers/counters; optionally pair 16-bit timers into 32-bit timer modules
- 16-bit Capture input functions
- 16-bit Compare/PWM output functions
- 3-wire SPI modules (supports 4 Frame modes)
- I^2C™ module supports Multi-Master/Slave mode and 7-bit/10-bit addressing
- 2 UART modules with FIFO Buffers
- 1 CAN module, 2.0B compliant

Motor Control PWM Module Features:

- 6 PWM output channels:
  - Complementary or Independent Output modes
  - Edge and Center-Aligned modes
- 3 duty cycle generators
- Dedicated time base
- Programmable output polarity
- Dead-time control for Complementary mode
- Manual output control
- Trigger for A/D conversions

Quadrature Encoder Interface Module Features:

- Phase A, Phase B and Index Pulse input
- 16-bit up/down position counter
- Count direction status
- Position Measurement (x2 and x4) mode
- Programmable digital noise filters on inputs
- Alternate 16-Bit Timer/Counter mode
- Interrupt on position counter rollover/underflow

Figure C.1. Datasheet summary of the dsPIC30F4012 Microcontroller - Part 1 [66].

dsPIC30F4011/4012

Analog Features:

- 10-Bit Analog-to-Digital Converter (A/D) with 4 S/H inputs:
  - 1 Msps conversion rate
  - 9 input channels
  - Conversion available during Sleep and Idle
- Programmable Brown-out Reset

Special Digital Signal Controller Features:

- Enhanced Flash program memory:
  - 10,000 erase/write cycle (min.) for industrial temperature range, 100K (typical)
- Data EEPROM memory:
  - 100,000 erase/write cycle (min.) for industrial temperature range, 1M (typical)
- Self-reprogrammable under software control
- Power-on Reset (POR), Power-up Timer (PWRT) and Oscillator Start-up Timer (OST)

Special Digital Signal Controller Features (Cont.):

- Flexible Watchdog Timer (WDT) with on-chip, low-power RC oscillator for reliable operation
- Fail-Safe Clock Monitor operation detects clock failure and switches to on-chip, low-power RC oscillator
- Programmable code protection
- In-Circuit Serial Programming™ (ICSP™)
- Selectable Power Management modes:
  - Sleep, Idle and Alternate Clock modes

CMOS Technology:

- Low-power, high-speed Flash technology
- Wide operating voltage range (2.5V to 5.5V)
- Industrial and Extended temperature ranges
- Low-power consumption

dsPIC30F Motor Control and Power Conversion Family*

Device	Pins	Program Mem. Bytes/ Instructions	SRAM Bytes	EEPROM Bytes	Timer 16-bit	Input Cap	Output Comp/Std PWM	Motor Control PWM	10-Bit A/D 1 Msps	Quad Enc	UART	SPI	I²C™	CAN
dsPIC30F2010	28	12K/4K	512	1024	3	4	2	6 ch	6 ch	Yes	1	1	1	-
dsPIC30F3010	28	24K/8K	1024	1024	5	4	2	6 ch	6 ch	Yes	1	1	1	-
dsPIC30F4012	28	48K/16K	2048	1024	5	4	2	6 ch	6 ch	Yes	1	1	1	1
dsPIC30F3011	40/44	24K/8K	1024	1024	5	4	4	6 ch	9 ch	Yes	2	1	1	-
dsPIC30F4011	40/44	48K/16K	2048	1024	5	4	4	6 ch	9 ch	Yes	2	1	1	1
dsPIC30F5015	64	66K/22K	2048	1024	5	4	4	8 ch	16 ch	Yes	1	2	1	1
dsPIC30F6010	80	144K/48K	8192	4096	5	8	8	8 ch	16 ch	Yes	2	2	1	2

* This table provides a summary of the dsPIC30F6010 peripheral features. Other available devices in the dsPIC30F Motor Control and Power Conversion Family are shown for feature comparison.

Figure C.2. Datasheet summary of the dsPIC30F4012 Microcontroller - Part 2 [66].

APPENDIX D: SOURCE CODE OF THE TUNING ALGORITHM

D.1. C Codes

```
#include <p30Fxxxx.h>
#include <dsp.h>
#include "fft.h"
#include "conf.h"

#include <uart.h>
#include <spi.h>
#include <timer.h>
#include <delay.h>

#define  motor_period_low  9000
#define  motor_period_high 25000
#define  motor_period_ramp 300


const unsigned short microstepCosines[8]
   __attribute__ ((space(auto_psv), aligned (16))) =
   {0   ,
   9   ,
   18   ,
   26   ,
   33   ,
   38   ,
   41   ,
   42};


extern fractcomplex sigCmpx[FFT_BLOCK_LENGTH]
            __attribute__ ((section (".ydata, data, ymemory"),
            aligned (FFT_BLOCK_LENGTH * 2 *2)));


extern const fractcomplex twiddleFactors[FFT_BLOCK_LENGTH/2]
            __attribute__ ((space(auto_psv),
            aligned (FFT_BLOCK_LENGTH*2)));


extern fractional fft_mag[FFT_BLOCK_LENGTH/2]
```

```
            __attribute__ ((section (".xbss, bss, xmemory"),
            aligned (FFT_BLOCK_LENGTH)));


extern const fractional hannWindow[FFT_BLOCK_LENGTH]
            __attribute__ ((space(auto_psv),
            aligned (FFT_BLOCK_LENGTH*4)));

extern const fractional testSample[FFT_BLOCK_LENGTH]
            __attribute__ ((space(auto_psv),
            aligned (FFT_BLOCK_LENGTH*4)));


fractional *ptrRegionA       = &sigCmpx[0].real;
fractional *ptrRegionB       = &sigCmpx[64].real;
fractional *ptrRegionC       = &sigCmpx[128].real;
fractional *ptrRegionD       = &sigCmpx[192].real;
fractional *ptrFFTResult    = &fft_mag[0];


volatile int         most_recent_sample_nr = 0;
volatile fractional       most_recent_sample = 0;



fractional       *ptrTestSample;
fractional       *ptrHannwindow;
fractional       *ptrHannedValue;
fractional       hannWindowValue = 0;

fractional     maxSample = 0;
fractional     minSample = 0;
fractional     localMaxSample = 0;
fractional     localMinSample = 0;
volatile fractional   sampleBias = 0;


unsigned int    dynamic_scale_factor = 8;
unsigned int    dummy  = 0;


volatile unsigned int       motorCurrentDivider  = 0;


volatile unsigned long     step_command = 0;
volatile unsigned int      program_status = 0;
```

```
//program status variable
//xxx  motor current mode
//x:   tuning action in process
//x:   routine 1024 * 600usec service is in process
//x:   vibration data in process
//xxx: sample vector section 0-7
//x:   vibration data real/artificial
//x:   tuning mode auto/man
//x:   motor driver up/idle
//x:   uart is up/down
//x:   spi is up/down
//x:   timer is up/down
//x:   system is up/down


volatile unsigned int      program_status2 = 0;
//program status 2 variable
//x
//x:
//x:
//x:
//x:
//x:
//xxx:
//x:
//x:
//x:  bias the accelerometer
//x:  step command is not sent
//x:  uart msg is being sent
//x:  free_wheel_ccw
//x:  free_wheel_cw


volatile unsigned int      txDataCounter = 0;


int          i = 0;
int          j = 0;


volatile unsigned int   UART_Rx_Data[] = {0, 0};
volatile unsigned int   UART_Tx_Data[]
         = {0, 0, 0, 0, 0, 0, 0, 0,
           0, 0, 0, 0, 0, 0, 0, 0,
           0, 0, 0, 0, 0, 0, 0, 0,
           0, 0, 0, 0, 0, 0, 0, 0,
           0, 0, 0, 0, 0, 0, 0, 0,
           0, 0, 0, 0, 0, 0, 0, 0,
```

```
          0, 0, 0, 0, 0, 0, 0, 0,
          0, 0, 0, 0, 0, 0, 0, 0};


fractional      rmsValues[]
   = {   0, 0, 0, 0,
      0, 0, 0, 0,
      0, 0, 0, 0,
      0, 0, 0, 0};


fractional       sumOfRecentRMSValues = 0;
fractional       sumOfRecentRMSValuesOld = 0;
fractional       dRMS1 = 0;
fractional       dRMS2 = 0;

fractional       freqEstimation[] = {0, 0, 0, 0,
                          0, 0, 0, 0,
                          0, 0, 0, 0,
                          0, 0, 0, 0};


fractional      dFreqEstimation = 0;
volatile unsigned int currentMicroStep = 0;
volatile unsigned int targetMicroStep  = 0;
       unsigned int lastFineTuningDirection = 0;
       unsigned int lastFineTuningMag = 0;


volatile unsigned int offsetMicroStep  = 0;
volatile unsigned long       programTimer = 0;


const fractional frequencyLookUp[] =
                  {   8448   ,
                     9214   ,
                     9470   ,
                     10243   ,
                     10499   ,
                     11266   ,
                     11521   ,
                     12032   ,
                     12288   ,
                     12799
                     };


const fractional motorPositionLookUp[] =
```

```
                {
                   16800   ,
                   13440   ,
                   11760   ,
                   10080   ,
                   8400   ,
                   6720   ,
                   5040   ,
                   3360   ,
                   1680   ,
                   0      };


unsigned int lookUpTableSize = 10;


extern void        VectorRevCopy (int numElems,
                          fractional *srcV,
                          fractional *dstV);


extern void        Vector2Cmplx (   int numElems,
                          fractional *srcV,
                          fractional *dstV);


extern fractional   *fractMPY (      fractional *srcV1,
                          fractional *srcV2);


extern void        VectorMultiplyPSV (int numElems,
                             fractional *srcV1,
                             fractional *srcV2,
                             int PSVPage);


extern void        VectorLeftShift (   int numElems,
                             int scaleFactor,
                             fractional *srcV);


extern void        VectorRightShift (   int numElems,
                             int scaleFactor,
                             fractional *srcV);


extern unsigned int    FractSquareRoot  (fractional srcV);
```

```
extern void          CalculateRMS(fractional *srcV,
                        fractional *ptrRMSValues,
                        unsigned int dynamic_rms_scale);


extern void        PeakFFTFrequency(fractional *ptrFFTResults,
                     fractional *ptrFreqEstimation);


extern fractional    interpolateTuningParameter   (
                     fractional freqDifference,
                      fractional freqDifferenceMeasured,
                      fractional positionDifference);

extern void        send_soft_spi_data
                  (unsigned long  lngSPIWord);


extern unsigned int    save_uart_command
                  (unsigned int *ptrUART_Rx_Data);

void initSPI(void);
void initTimer1(void);
void initTimer2(void);
void arrange_sample_array(void);
unsigned int calculateDynamicScaling(fractional maxValue);
void perform_fft_in_place(void);
void initUART(void);


void calculateMicroStep(unsigned int stepNr);
void initSoftSPI();
void sendUARTWord(int UARTWord, int UARTPrefix);



void __attribute__((__interrupt__, auto_psv)) _SPI1Interrupt(void)
   {
   int tmpSPIData = 0;
   IFS0bits.SPI1IF = 0;

   if(SPI1STATbits.SPIRBF)
      {
       SPI1STATbits.SPIROV = 0;
       tmpSPIData = SPI1BUF;

       if((program_status & 0x0040) == 0x0040)
```

```
         {
        if((tmpSPIData >> 8) != 0x007F)
           {
            program_status |= 0x0400;
                     //vibration data in process
            most_recent_sample = ((fractional)tmpSPIData) + 281;
           }
        }
       }
   }


void __attribute__((__interrupt__, auto_psv)) _T1Interrupt(void)
   {
   int tmpSPIData = 0;
    TMR1 = 0;
    IFS0bits.T1IF = 0;
    programTimer++;


    if((programTimer & 0x07FF) == 0x07FF)
       {
      program_status |= 0x0800;
       }

      if(U1STAbits.TRMT)
      {
      if((txDataCounter > 0) && ((program_status2 & 0x0004) == 0))
         {
         program_status2 |= 0x0004;
         U1TXREG = UART_Tx_Data[0];
         for(j=1;j<txDataCounter;j++)
            UART_Tx_Data[j-1] = UART_Tx_Data[j];
         txDataCounter--;
         program_status2 &= 0xFFFB;
         }
      }

   }



void __attribute__((__interrupt__, auto_psv)) _T2Interrupt(void)
   {
    TMR2 = 0;
    IFS0bits.T2IF = 0;
    if(PR2>motor_period_low)
       PR2 = PR2 - motor_period_ramp;
```

```
    program_status2 &= 0xFFF7; //step command is not sent

    }


void __attribute__((__interrupt__, auto_psv)) _U1RXInterrupt(void)
   {
    IFS0bits.U1RXIF = 0;
    if(save_uart_command(&UART_Rx_Data[0]))
       {

       switch(UART_Rx_Data[0])
          {
          case 42:            //reset command
             asm("RESET");
             break;

          case 43:
                //activate motor driver command
            //step_command = 0b10000000100010000101;
            step_command = 0b10000000011011110101;
                  //high fast decay
            program_status |= 0x0010;
             break;

          case 44:
                   //inactivate motor driver command
            step_command = 0b00000000011011110101;
                  //high fast decay
            program_status    &= 0xFFEF;
            program_status2 &= 0xFFFC;
             break;

          case 45:   //tuning mode: auto command
            program_status |= 0x0020;
             break;

          case 46:      //tuning mode: manual command
            program_status &= (0xFFDF);
             break;

          case 47:      //vibration data: real command
            program_status |= 0b1000000;
             break;

          case 48:      //vibration data: test command
            program_status &= (0xFFBF);
             break;
```

```
   case 49:      //set motor position
      if((program_status & 0x0020) == 0)
         {
        PR2 = motor_period_high;
        T2CON |= 0x8000;
        targetMicroStep = UART_Rx_Data[1]<<3;
        }
      break;

   case 51:      //offset motor +8 command
      if((T2CON & 0x8000) == 0)
         {
        PR2 = motor_period_high;
        T2CON |= 0x8000;
        offsetMicroStep += 8;
        }
      break;

   case 52:            //offset motor -8 command
      if((T2CON & 0x8000) == 0)
         {
        PR2 = motor_period_high;
        T2CON |= 0x8000;
         if(offsetMicroStep>7)
            offsetMicroStep -= 8;
        }
      break;

   case 53:            //test sample received
        program_status |= 0x0400;
           //vibration data in process
      most_recent_sample = UART_Rx_Data[1]-2000;
      break;
   case 54:            //free_wheel_cw

      program_status2 = (program_status2 & 0xFFFC) | 0x0001;
      break;
   case 55:            //free_wheel_ccw

      program_status2 = (program_status2 & 0xFFFC) | 0x0002;
      break;

   case 56:            //auto-zero accelerometer output
      program_status2 |= 0x0010;
      break;
   }
}
```

```
    }

void indicate_reset()
   {
   TRISD &= 0b1101;
   asm("BSET 0x02D6, #1");   //set led
   delay_ms(500);
   asm("BCLR 0x02D6, #1");   //clear led
   delay_ms(500);
   program_status |= 0b1;
   }

int main(void)
   {
   ptrHannwindow = (fractional *)
            __builtin_psvoffset(&hannWindow[0]);

   ptrTestSample = (fractional *)
            __builtin_psvoffset(&testSample[0]);


   indicate_reset();
   initSPI();
   initSoftSPI();
   initUART();

   initTimer1();
   initTimer2();

   program_status |=  0x0040; //real data
            step_command = 0b10000000011011110101;
                  //high fast decay
            program_status |= 0x0010;
            program_status |= 0x0020;

   while(1)
      {
      if(step_command != 0)
         {
         send_soft_spi_data (step_command);

         if((step_command & 0x00000001) == 0x00000001)
            delay_us(1200);

         step_command = 0;
         }

      if(((program_status & 0x0010) == 0x0010) &&
```

```
      ((program_status2 & 0x0008) == 0))
{
 if((program_status2 & 0x0003) == 0x0001)
    {
    if(currentMicroStep == 9600)
       {
      PR2 = motor_period_high;
      T2CON &= 0x7FFF;
      targetMicroStep = 0;

      step_command = 0b1000000000011110101;
            //high fast decay

      send_soft_spi_data (step_command);

      delay_ms(1000);
      step_command = 0b1000000011011110101;
            //high fast decay

      send_soft_spi_data (step_command);

      T2CON |= 0x8000;
       program_status2 = (program_status2 & 0xFFFC)
                         | 0x0002;
       }
    }
 else if((program_status2 & 0x0003) == 0x0002)
    {
    if(currentMicroStep == 0)
       {
      PR2 = motor_period_high;
      T2CON &= 0x7FFF;
      targetMicroStep = 9600;

      step_command = 0b1000000000011110101;
            //high fast decay
      send_soft_spi_data (step_command);
      delay_ms(1000);
      step_command = 0b1000000011011110101;
            //high fast decay
      send_soft_spi_data (step_command);

      T2CON |= 0x8000;
       program_status2 = (program_status2 & 0xFFFC)
                       | 0x0001;
       }
    }
```

```
    if(currentMicroStep > targetMicroStep + offsetMicroStep)
       {
      motorCurrentDivider = 2;
       currentMicroStep -= 1;

       calculateMicroStep((currentMicroStep & 0b111111)
                          | 0b1000000);
       }
    else if(currentMicroStep <
             targetMicroStep + offsetMicroStep)
       {
      motorCurrentDivider = 2;

       currentMicroStep += 1;

       calculateMicroStep((currentMicroStep & 0b111111));
       }
    else
       {
      step_command = 0b10000000000011110101;
        //high fast decay
      send_soft_spi_data (step_command);

      delay_us(1200);
      T2CON &= 0x7FFF;
      motorCurrentDivider = 5;
       calculateMicroStep((currentMicroStep & 0b111111));
       }
    program_status2 |= 0x0008;
   }

if(step_command != 0)
   {
   send_soft_spi_data (step_command);

   if((step_command & 0x00000001) == 0x00000001)
      delay_us(1200);
   step_command = 0;
   }

if((program_status & 0x0400) == 0x0400)
   {
   while(txDataCounter > 57);
   sendUARTWord(most_recent_sample, 0x00FA);

   *(ptrRegionA + most_recent_sample_nr)
      = most_recent_sample;
```

```
hannWindowValue
   = (*(ptrHannwindow+most_recent_sample_nr));

ptrHannedValue
   = fractMPY(&most_recent_sample, &hannWindowValue);



if(maxSample < *ptrHannedValue)
   maxSample = *ptrHannedValue;


if(minSample > *ptrHannedValue)
   minSample = *ptrHannedValue;


//rms calculations
if(((most_recent_sample_nr+1) & 0b11111) == 0)
   {
   for(i = 1; i < 16; i++)
      rmsValues[i-1]  = rmsValues[i];


   localMaxSample = 0;
   localMinSample = 0;

   for(i = 0; i < 32; i++)
      {
      if(localMaxSample <
            *(ptrRegionA + most_recent_sample_nr - i))
         localMaxSample
            = *(ptrRegionA + most_recent_sample_nr - i);


      if(localMinSample >
            *(ptrRegionA + most_recent_sample_nr - i))
         localMinSample
            = *(ptrRegionA + most_recent_sample_nr - i);


      }
   if((-1) * localMinSample > localMaxSample)
      localMaxSample = (-1) *localMinSample;

   CalculateRMS(ptrRegionA + most_recent_sample_nr,
            &rmsValues[15],
            calculateDynamicScaling(localMaxSample));
```

```
      dRMS1 = sumOfRecentRMSValues
            - sumOfRecentRMSValuesOld;


      sumOfRecentRMSValues = rmsValues[15]
                           + rmsValues[14]
                           + rmsValues[13]
                           + rmsValues[12];


      sumOfRecentRMSValuesOld = rmsValues[13]
                           + rmsValues[12]
                           + rmsValues[11]
                           + rmsValues[10];



      dRMS2 = sumOfRecentRMSValues
            - sumOfRecentRMSValuesOld;

      while(txDataCounter > 57);
         sendUARTWord(rmsValues[15], 0x00FB);


      program_status = ((program_status & 0xFC7F)
            | ((most_recent_sample_nr & 0xFFE0) << 2))
            | 0x1000;
      }

      most_recent_sample_nr
         = (most_recent_sample_nr+1) & 0x00FF;


   program_status &= 0xFBFF;
   }

if((program_status & 0x1780) == 0x1000)
   {
   asm("BSET 0x02D6, #1");


   if((T2CON &= 0x8000) == 0)
      {
      if(sumOfRecentRMSValues > 30)
         {
         PR2 = motor_period_high;
```

```
            T2CON |= 0x8000;

            if(dRMS2 > 5)
               {
               if(lastFineTuningDirection == 0)
                  lastFineTuningDirection = 1;

               else
                  lastFineTuningDirection = 0;

               lastFineTuningMag = 100;
               }
            else
               {
               if(dRMS1 < 0)
                  {
                  if(dRMS2 > dRMS1)
                     lastFineTuningMag += 50;
                  else
                     lastFineTuningMag -= 50;
                  }
               }

             if(lastFineTuningDirection == 0)
                {
               if(targetMicroStep > lastFineTuningMag)
                  targetMicroStep -= lastFineTuningMag;

               else
                  targetMicroStep = 0;
               }
            else
               {
               if(targetMicroStep + lastFineTuningMag
                     < 16800)
                  targetMicroStep += lastFineTuningMag;

               }
            }
         }


      while(txDataCounter > 57);
      sendUARTWord(targetMicroStep, 247);
      program_status &= 0xEFFF;
      }
   else if((program_status & 0x1780) == 0x1080)
      {
```

```
asm("BCLR 0x02D6, #1");

if((T2CON &= 0x8000) == 0)
   {
   if(sumOfRecentRMSValues > 30)
      {
      PR2 = motor_period_high;
      T2CON |= 0x8000;

      if(dRMS2 > 5)
         {
         if(lastFineTuningDirection == 0)
            lastFineTuningDirection = 1;
         else
            lastFineTuningDirection = 0;
         lastFineTuningMag = 100;
         }
      else
         {
         if(dRMS1 < 0)
            {
            if(dRMS2 > dRMS1)
               lastFineTuningMag += 40;
            else
               lastFineTuningMag -= 40;
            }
         }

       if(lastFineTuningDirection == 0)
          {
         if(targetMicroStep > lastFineTuningMag)
            targetMicroStep -= lastFineTuningMag;

         else
            targetMicroStep = 0;
         }
      else
         {
         if(targetMicroStep + lastFineTuningMag
            < 16800)
            targetMicroStep += lastFineTuningMag;
         }
      }
   }

while(txDataCounter > 57);
sendUARTWord(targetMicroStep, 247);
program_status &= 0xEFFF;
```

```
   }


else if((program_status & 0x1780) == 0x1100)
   {
   asm("BSET 0x02D6, #1");

   if((T2CON &= 0x8000) == 0)
      {

      if(sumOfRecentRMSValues > 30)
         {
         PR2 = motor_period_high;
         T2CON |= 0x8000;



         if(dRMS2 > 5)
            {
            if(lastFineTuningDirection == 0)
               lastFineTuningDirection = 1;
            else
               lastFineTuningDirection = 0;
            lastFineTuningMag = 100;
            }
         else
            {
            if(dRMS1 < 0)
               {
               if(dRMS2 > dRMS1)
                  lastFineTuningMag += 30;
               else
                  lastFineTuningMag -= 30;
               }
            }


          if(lastFineTuningDirection == 0)
             {
            if(targetMicroStep > lastFineTuningMag)
               targetMicroStep -= lastFineTuningMag;
            else
               targetMicroStep = 0;
            }
         else
            {
            if(targetMicroStep + lastFineTuningMag
               < 16800)
               targetMicroStep += lastFineTuningMag;
```

```
            }
         }
      }

   while(txDataCounter > 57);
   sendUARTWord(targetMicroStep, 247);
   program_status &= 0xEFFF;
   }


else if((program_status & 0x1780) == 0x1180)
   {
   asm("BCLR 0x02D6, #1");

   if((T2CON &= 0x8000) == 0)
      {
      if(sumOfRecentRMSValues > 30)
         {
         PR2 = motor_period_high;
         T2CON |= 0x8000;

         if(dRMS2 > 5)
            {
            if(lastFineTuningDirection == 0)
               lastFineTuningDirection = 1;
            else
               lastFineTuningDirection = 0;
            lastFineTuningMag = 100;
            }

         else
            {
            if(dRMS1 < 0)
               {
               if(dRMS2 > dRMS1)
                  lastFineTuningMag += 20;
               else
                  lastFineTuningMag -= 20;
               }
            }

          if(lastFineTuningDirection == 0)
             {
            if(targetMicroStep > lastFineTuningMag)
               targetMicroStep -= lastFineTuningMag;

            else
               targetMicroStep = 0;
```

```
            }

         else
            {
            if(targetMicroStep + lastFineTuningMag
               < 16800)
               targetMicroStep += lastFineTuningMag;
            }
         }
      }



   while(txDataCounter > 57);
   sendUARTWord(targetMicroStep, 247);
   program_status &= 0xEFFF;
   }


else if((program_status & 0x1780) == 0x1200)
   {
   asm("BSET 0x02D6, #1");

   if((T2CON &= 0x8000) == 0)
      {
      if(sumOfRecentRMSValues > 30)
         {
         PR2 = motor_period_high;
         T2CON |= 0x8000;

         if(dRMS2 > 5)
            {
            if(lastFineTuningDirection == 0)
               lastFineTuningDirection = 1;

            else
               lastFineTuningDirection = 0;
            lastFineTuningMag = 100;
            }

         else
            {
            if(dRMS1 < 0)
               {
               if(dRMS2 > dRMS1)
                  lastFineTuningMag += 20;
               else
                  lastFineTuningMag -= 20;
```

```
            }
         }

       if(lastFineTuningDirection == 0)
          {
         if(targetMicroStep > lastFineTuningMag)
            targetMicroStep -= lastFineTuningMag;

         else
            targetMicroStep = 0;
         }

       else
         {
         if(targetMicroStep + lastFineTuningMag
            < 16800)
            targetMicroStep += lastFineTuningMag;
         }
       }
    }

  while(txDataCounter > 57);
  sendUARTWord(targetMicroStep, 247);
  program_status &= 0xEFFF;
  }


else if((program_status & 0x1780) == 0x1280)
  {
  asm("BCLR 0x02D6, #1");

  if((T2CON &= 0x8000) == 0)
    {
    if(sumOfRecentRMSValues > 30)
      {
      PR2 = motor_period_high;
      T2CON |= 0x8000;

      if(dRMS2 > 5)
        {
        if(lastFineTuningDirection == 0)
          lastFineTuningDirection = 1;

        else
          lastFineTuningDirection = 0;
        lastFineTuningMag = 100;
        }
```

```
         else
            {
            if(dRMS1 < 0)
               {
               if(dRMS2 > dRMS1)
                  lastFineTuningMag += 20;
               else
                  lastFineTuningMag -= 20;
               }
            }

          if(lastFineTuningDirection == 0)
             {
            if(targetMicroStep > lastFineTuningMag)
               targetMicroStep -= lastFineTuningMag;

            else
               targetMicroStep = 0;
            }

         else
            {
            if(targetMicroStep + lastFineTuningMag
               < 16800)
               targetMicroStep += lastFineTuningMag;
            }
         }
      }

   while(txDataCounter > 57);
   sendUARTWord(targetMicroStep, 247);
   program_status &= 0xEFFF;
   }


else if((program_status & 0x1780) == 0x1300)
   {
   asm("BSET 0x02D6, #1");

   if((T2CON &= 0x8000) == 0)
      {
      if(sumOfRecentRMSValues > 30)
         {
         PR2 = motor_period_high;
         T2CON |= 0x8000;

         if(dRMS2 > 5)
            {
```

```
            if(lastFineTuningDirection == 0)
               lastFineTuningDirection = 1;

            else
               lastFineTuningDirection = 0;
            lastFineTuningMag = 100;
            }

         else
            {
            if(dRMS1 < 0)
               {
               if(dRMS2 > dRMS1)
                  lastFineTuningMag += 10;

               else
                  lastFineTuningMag -= 10;
               }
            }

          if(lastFineTuningDirection == 0)
             {
            if(targetMicroStep > lastFineTuningMag)
               targetMicroStep -= lastFineTuningMag;

            else
               targetMicroStep = 0;
            }

         else
            {
            if(targetMicroStep + lastFineTuningMag
               < 16800)
               targetMicroStep += lastFineTuningMag;
            }
         }
      }
   while(txDataCounter > 57);
   sendUARTWord(targetMicroStep, 247);
   program_status &= 0xEFFF;
   }


else if((program_status & 0x1780) == 0x1380)
   {
   asm("BCLR 0x02D6, #1");
   if(sumOfRecentRMSValues > 80)
      {
```

```
   arrange_sample_array();
   perform_fft_in_place();

   while(txDataCounter > 57);
   sendUARTWord(freqEstimation[15], 246);

   if((program_status & 0x0020) == 0x0020)
      // if auto tuning is enabled
      {
      if(freqEstimation[15] > freqEstimation[14])
         dFreqEstimation = freqEstimation[15]
                     - freqEstimation[14];

      else
         dFreqEstimation = freqEstimation[14]
                     - freqEstimation[15];


      if(dFreqEstimation < 3000)
         {

         if(dRMS2 > 30 && dRMS1 > 30)
            {

            for(i=0; i<lookUpTableSize; i++)
               {
               if(freqEstimation[15]
                     < frequencyLookUp[i])
                  break;
               }

            PR2 = motor_period_high;
            T2CON |= 0x8000;
            if(i!=0 && i != lookUpTableSize)
               {
targetMicroStep
= motorPositionLookUp[i-1] -
   interpolateTuningParameter(
      frequencyLookUp[i]-frequencyLookUp[i-1],
      freqEstimation[15]-frequencyLookUp[i-1],
      motorPositionLookUp[i-1]-motorPositionLookUp[i]);
                lastFineTuningDirection = 0;
               lastFineTuningMag = 300;
               dRMS2 = -1;
               }
            }
         }
```

```
            }

         while(txDataCounter > 57);
         sendUARTWord(targetMicroStep, 247);
         }

      program_status &= 0xEFFF;
      }


   if((program_status & 0b110000000000) == 0b100000000000)
             //this will execute every sec.
   {          //unless a sample is being processed

      program_status = (program_status & 0x1FFF)
            | (motorCurrentDivider << 13);

      if(txDataCounter < 57)
         sendUARTWord(program_status, 0x00FC);

      program_status &= 0xF7FF; //routine 1sec service
                           //is not in process
      }
   }

}



void arrange_sample_array(void)
   {
   if((-1) * minSample > maxSample)
      maxSample = (-1) * minSample;

   dynamic_scale_factor = calculateDynamicScaling(maxSample);
   maxSample = 0;
   minSample = 0;


   if(dynamic_scale_factor > 0)
      {
      VectorLeftShift (   FFT_BLOCK_LENGTH-1,
                  dynamic_scale_factor,
                  ptrRegionA);

      }
```

```
   VectorMultiplyPSV(   FFT_BLOCK_LENGTH-1,
                        ptrRegionA,
                        ptrHannwindow,
                        (int) __builtin_psvpage(&hannWindow[0]));


     Vector2Cmplx(   FFT_BLOCK_LENGTH-1,
                  ptrRegionC-1,
                  ptrRegionD + FFT_BLOCK_LENGTH/2 - 1);
   }

void perform_fft_in_place(void)
   {
   FFTComplexIP (LOG2_BLOCK_LENGTH,
              &sigCmpx[0],
              (fractcomplex *)
                 __builtin_psvoffset(&twiddleFactors[0]),
              (int) __builtin_psvpage(&twiddleFactors[0]));



   BitReverseComplex(   LOG2_BLOCK_LENGTH,
                    &sigCmpx[0]);


   for(i = 1; i < 16; i++)
      freqEstimation[i-1]  = freqEstimation[i];


   PeakFFTFrequency(   &sigCmpx[0],
                    &freqEstimation[15]);

   }



void sendUARTWord(int UARTWord, int UARTPrefix)
   {
   unsigned long tempProgramTimer = 0;
   tempProgramTimer = programTimer;

   while((program_status2 & 0x0004) == 0x0004);
   program_status2 |= 0x0004;


   UART_Tx_Data[txDataCounter]   = UARTPrefix;
   txDataCounter++;
```

```
   UART_Tx_Data[txDataCounter]   = UARTPrefix;
   txDataCounter++;

   UART_Tx_Data[txDataCounter]   = (tempProgramTimer) & 0x00FF;
   txDataCounter++;

   UART_Tx_Data[txDataCounter]   = (tempProgramTimer >> 8) & 0x00FF ;
   txDataCounter++;

   UART_Tx_Data[txDataCounter]   = (tempProgramTimer >> 16) & 0x00FF ;
   txDataCounter++;

   UART_Tx_Data[txDataCounter]   = (tempProgramTimer >> 24) & 0x00FF ;
   txDataCounter++;


   UART_Tx_Data[txDataCounter]   = (UARTWord) & 0x00FF ;
   txDataCounter++;

   UART_Tx_Data[txDataCounter]   = (UARTWord >> 8) & 0x00FF;
   txDataCounter++;

   program_status2 &= 0xFFFB;
   }


void initTimer1(void)
   {
    ConfigIntTimer1(T1_INT_PRIOR_5 & T1_INT_ON);
    TMR1 = 0;

    OpenTimer1(T1_ON & T1_GATE_OFF & T1_IDLE_STOP &
               T1_PS_1_1 & T1_SYNC_EXT_OFF &
               T1_SOURCE_INT, 14372);

   program_status |= 0b10;
   }


void initTimer2(void)
   {
    ConfigIntTimer2(T2_INT_PRIOR_3 & T2_INT_ON);

    OpenTimer2(T2_OFF & T2_GATE_OFF & T2_IDLE_STOP &
               T1_PS_1_1 & T2_32BIT_MODE_OFF &
               T2_SOURCE_INT, motor_period_high);
   }
```

```
void initSPI(void)
   {
   SPI1BUF = 0x0000;
    CloseSPI1();

    ConfigIntSPI1(SPI_INT_EN &  SPI_INT_PRI_6);

    OpenSPI1(   FRAME_ENABLE_OFF & FRAME_SYNC_INPUT &
              DISABLE_SDO_PIN  & SPI_MODE16_ON &
                SPI_SMP_ON & SPI_CKE_ON &
                SLAVE_ENABLE_OFF  &
                CLK_POL_ACTIVE_HIGH &
                MASTER_ENABLE_OFF &
                SEC_PRESCAL_7_1 &
                PRI_PRESCAL_64_1,
                SPI_ENABLE & SPI_IDLE_STOP &
                SPI_RX_OVFLOW_CLR );

   program_status |= 0b100;
   }


void initUART(void)
   {
   CloseUART1();
    ConfigIntUART1(UART_RX_INT_EN & UART_RX_INT_PR4 &
                   UART_TX_INT_DIS & UART_TX_INT_PR4);

    OpenUART1(   UART_EN & UART_IDLE_STOP &
              UART_ALTRX_ALTTX & UART_DIS_WAKE &
              UART_DIS_LOOPBACK & UART_DIS_ABAUD &
              UART_NO_PAR_8BIT & UART_1STOPBIT,
              UART_INT_TX_BUF_EMPTY & UART_TX_PIN_NORMAL &
              UART_TX_ENABLE & UART_INT_RX_CHAR &
              UART_ADR_DETECT_DIS & UART_RX_OVERRUN_CLEAR,
              15);  // baud = 29.48M/16/(15+1) = 115156.25


   program_status |= 0b1000;
   }

unsigned int calculateDynamicScaling(fractional maxValue)
   {
   unsigned int scale = 0;
   if(maxValue < (0x4000))
      {
      scale = 0;
      if(maxValue < (0x2000))
```

```
        {
        scale = 1;
        if(maxValue < (0x1000))
           {
           scale = 2;
           if(maxValue < (0x0800))
              {
              scale = 3;
              if(maxValue < (0x0400))
                 {
                 scale = 4;
                 if(maxValue < (0x0200))
                    {
                    scale = 5;
                    if(maxValue < (0x0100))
                       {
                       scale = 6;
                       if(maxValue < (0x0080))
                          {
                          scale = 7;
                          if(maxValue < (0x0040))
                             {
                             scale = 8;
                             if(maxValue < (0x0020))
                                {
                                scale = 9;
                                if(maxValue < (0x0010))
                                   {
                                   scale = 10;
                                   }
                                }
                             }
                          }
                       }
                    }
                 }
              }
           }
        }
     }
   return scale;
   }


void calculateMicroStep(unsigned int stepNr)
   {
   unsigned int   divider = 0;
   unsigned short steps[2] = {0,0};
```

```
program_status = (program_status & 0x1FFF)
            | (motorCurrentDivider << 13);


if(motorCurrentDivider == 0)
   {
   step_command = 0b1000000000000000000;
   }

else
   {
   step_command = 0;
   divider = motorCurrentDivider-1;
   }


if((stepNr & 0b111111) < 8)
   {
   steps[0] = microstepCosines[stepNr & 0b111111] >> divider;
   steps[1] = microstepCosines[7-(stepNr & 0b111111)] >> divider;

   if((stepNr>>6) == 0)
      step_command |= 0x10000;

   else
      step_command |= 0x08000;

   step_command |= (steps[0]) << 1 |
               (steps[1]) << 7 ;

   }

else if((stepNr & 0b111111) < 16)
   {
   steps[0] = microstepCosines[15-(stepNr & 0b111111)]
                  >> divider;

   steps[1] = microstepCosines[(stepNr & 0b111111)-8]
                  >> divider;

   if((stepNr>>6) == 0)
      step_command |= 0x08000;

   else
      step_command |= 0x10000;

   step_command |= (0b10)<<13 |
```

```
                (steps[0]) << 1 |
                (steps[1]) << 7 ;

   }

else if((stepNr & 0b111111) < 24)
   {
   steps[0] = microstepCosines[(stepNr & 0b111111) - 16]
                    >> divider;

   steps[1] = microstepCosines[23-(stepNr & 0b111111)]
                    >> divider;

   if((stepNr>>6) == 0)
      step_command |= 0x10000;

   else
      step_command |= 0x08000;

   step_command |= (0b11)<<13 |
                (steps[0]) << 1 |
                (steps[1]) << 7 ;
   }

else
   {
   steps[0] = microstepCosines[31-(stepNr & 0b111111)]
                   >> divider;

   steps[1] = microstepCosines[(stepNr & 0b111111)-24]
                   >> divider;

   if((stepNr>>6) == 0)
      step_command |= 0x08000;

   else
      step_command |= 0x10000;

   step_command |= (0b01)<<13 |
                (steps[0]) << 1 |
                (steps[1]) << 7 ;
   }


//send_soft_spi_data (step_command);
/*
1      : 1/4 current mode
0      : internal vref
```

```
   0      : I1 mixed decay
   0      : I2 mixed decay
   0      : I1 polarity
   0      : I2 polarity
   100000   : I1
   100000   : I2
   0      : word0 (step)
   */
   }



void initSoftSPI()
   {
   TRISE &= ~(0b111);
    LATE  |= 0b1;
    delay_us(60);
//   step_command = 0b0000000100010000101;
   /*
   0      : idle
   00      : xxx
   00      : sync. rectification: active
   00      : internal oscillator
   1000   : (17.75uSec)
      fast decay time: tfd = [(1 + N) x 8/fOSC] - 1/fOSC
   10000   : (33.75uSec)
      fixed off time toff = [(1 + N) x 8/fOSC] - 1/fOSC
   10      : blank time (can be reduced?): 6/fosc
   1      : word1 (config)
   */
   }
```

D.2. Assembly Codes

```
   .nolist
   .include        "dspcommon.inc"
   .list

   .global _PeakFFTFrequency
_PeakFFTFrequency:

    push    CORCON
    fractsetup      w4

    mov     w0, w10
```

```
    mov     w1, w9

    clr     a, [w10]+=2, w4
   clr    b
   clr      w11

    sac.r   a, #0, w6
    sac.r   b, #0, w7


    do      #126, L0
      mpy     w4*w4, a, [w10]+=2, w5
      mac     w5*w5, a, [w10]+=2, w4

      sac.r   a, #0, w6
      cp      w6, w7

      bra      n, L0
      subr   w0, w10, w11
      clr    b

      sub   b
      neg   b
      sac.r  b, #0, w7

L0:      nop
;Last iteration outside the DO loop
   mpy     w4*w4, a, [w10]+=2, w5
   mac     w5*w5, a
   sac.r   a, #0, w6

   mov      #0, w3
   cp      w6, w7

   bra      n, end_of_max_search
   subr   w0, w10, w11

   clr    b
   sub   b
   neg   b

   sac.r  b, #0, w7
   inc2   w11, w11

   mov   #1, w3

end_of_max_search:
```

```
      cp0     w7
      bra     z, peakMagnitudeIsZero

      sub      #6, w11
      sl       w11, #6, w10

      mov     w10, [w9]

peakMagnitudeIsZero:

_exactFrequencyBin:

        pop     CORCON
        return
    .end



 .global _interpolateTuningParameter ; export
_interpolateTuningParameter:

    push CORCON
    fractsetup w4

    mov.w w0, w7
    mov.w w1, w6

    repeat #17       ; Execute DIVF 18 times
    divf   w6, w7    ; Divide W6 by W7

    mov.w w0, w4
    mov.w w2, w5

    mpy  w4*w5,  a
    sac.r a, w0

    pop     CORCON
    return


    .global   _send_soft_spi_data   ; export
_send_soft_spi_data:

      mov.w    w0,    w11
      mov.w    w1,    w10

      bclr     0x02dc, #0  ; low ss
```

```
      do   #2, _WordH   ;do for 3 upper bit of the spi word

         nop
         btsc   w10,   #2    ;if 3rd lsb is clear,
                        ;next instruction will be skipped

         bset   0x02dc,   #2     ;if not skipped this instr.
                        ; will set the sdo pin bit

         btss   w10,   #2    ;if 3rd lsb is set, next
                        ; instruction will be skipped

         bclr   0x02dc,   #2     ;if not skipped this instr.
                        ; will clr the sdo pin bit

         nop

         bset   0x02dc,   #1     ;set clk pin
         nop
         nop
         bclr   0x02dc,   #1     ;clr clk pin


_WordH:      sl       w10,   #1,   w10

      do   #15, _WordL   ;do for 16 lower bit of
                        ; the spi word

         nop
         btsc   w11,   #15    ;if 16th lsb is clear,
                        ; next instruction will be skipped

         bset   0x02dc,   #2     ;if not skipped this
                        ; instr. will set the sdo pin bit

         btss   w11,   #15    ;if 16th lsb is set,
                        ; next instruction will be skipped

         bclr   0x02dc,   #2     ;if not skipped this
                        ; instr. will clr the sdo pin bit

         nop

         bset   0x02dc,   #1     ;set clk pin
         nop
         nop
         bclr   0x02dc,   #1     ;clr clk pin
```

```
_WordL:      sl       w11,   #1,   w11
   nop
   bset     0x02dc,    #0
   return
   .end


 .global _CalculateRMS ; export
_CalculateRMS:

    push CORCON
    fractsetup w4

    mov.w w0, w10
    mov.w [w10--], w4

    clr a

    do #30, _macLoop
            sl w4,w2,w4
    _macLoop : mac w4*w4, a, [w10]-=2, w4

    sl w4,w2,w4
    mac w4*w4, a

    sftac a, #5
    sac.r a, w0   ;sum of squares of
                ;last 32 samples are calculated

    clr  w4

    bset    w4, #14
    mpy     w4*w4, a

    sac.r    a, w5

    cp    w5, w0
    bra    z, _endOfCalculation

    cpslt w5, w0
    bclr w4, #14

    bset w4, #13
    mpy  w4*w4,  a
    sac.r a, w5
```

```
cp    w5, w0

bra    z, _endOfCalculation
cpslt  w5, w0
bclr    w4, #13

bset w4, #12
mpy  w4*w4,  a
sac.r a,   w5
cp    w5, w0

bra    z, _endOfCalculation
cpslt  w5, w0
bclr    w4, #12

bset w4, #11
mpy  w4*w4,  a
sac.r a,   w5
cp    w5, w0

bra    z, _endOfCalculation
cpslt  w5, w0
bclr    w4, #11

bset w4, #10
mpy  w4*w4,  a
sac.r a,   w5
cp    w5, w0

bra    z, _endOfCalculation
cpslt  w5, w0
bclr    w4, #10

bset w4, #9
mpy  w4*w4,  a
sac.r a,   w5
cp    w5, w0

bra    z, _endOfCalculation
cpslt  w5, w0
bclr    w4, #9

bset w4, #8
mpy  w4*w4,  a
sac.r a,   w5
cp    w5, w0

bra    z, _endOfCalculation
```

```
cpslt  w5, w0
bclr    w4, #8

bset w4, #7
mpy  w4*w4,  a
sac.r a,   w5
cp    w5, w0

bra    z, _endOfCalculation
cpslt  w5, w0
bclr    w4, #7

bset w4, #6
mpy  w4*w4,  a
sac.r a,   w5
cp    w5, w0

bra    z, _endOfCalculation
cpslt  w5, w0
bclr    w4, #6

bset w4, #5
mpy  w4*w4,  a
sac.r a,   w5
cp    w5, w0

bra    z, _endOfCalculation
cpslt  w5, w0
bclr    w4, #5

bset w4, #4
mpy  w4*w4,  a
sac.r a,   w5
cp    w5, w0

bra    z, _endOfCalculation
cpslt  w5, w0
bclr    w4, #4

bset w4, #3
mpy  w4*w4,  a
sac.r a,   w5
cp    w5, w0

bra    z, _endOfCalculation
cpslt  w5, w0
bclr    w4, #3
```

```
    bset w4, #2
    mpy  w4*w4,  a
    sac.r a,   w5
    cp    w5, w0

    bra    z, _endOfCalculation
    cpslt  w5, w0
    bclr    w4, #2

    bset w4, #1
    mpy  w4*w4,  a
    sac.r a,   w5
    cp    w5, w0

    bra    z, _endOfCalculation
    cpslt  w5, w0
    bclr    w4, #1

    bset w4, #0
    mpy  w4*w4,  a
    sac.r a,   w5
    cp    w5, w0

    bra    z, _endOfCalculation
    cpslt  w5, w0
    bclr    w4, #0

   _endOfCalculation:
    asr   w4, w2, w4
    mov.w w4, [w1]     ;fractional square root of the sum
         ;is calculated and written to rms array

    pop CORCON
    return
 .end


   .global   _save_uart_command   ; export
_save_uart_command:

   mov.w    w0,    w10
   mov.w    0x0212, w11

   mov.w   #0x00c0, w13
   and.w   w13, w11, w12
   mov.w   #0x0000, w13
   cpsne   w12,     w13
   call   _firstByte
```

```
    mov.w   #0x00c0, w13
    and.w   w13, w11, w12
    mov.w   #0x0040, w13
    cpsne   w12,     w13
    call   _secondByte

    mov.w   #0x00c0, w13
    and.w   w13, w11, w12
    mov.w   #0x00c0, w13
    cpsne   w12,     w13
    call   _thirdByte

    return

_firstByte:

    mov.w   #0x003f, w13
    and.w   w13, w11, w12
    mov.w   w12, [w10]
    mov.w   #0x0001, w0
    return

_secondByte:

    mov.w   #0x003f, w13
    and.w   w13, w11, w12
    sl      w12, #6, w12
    mov.w   w12, [++w10]
    mov.w   #0x0000, w0
    return

_thirdByte:

    mov.w   #0x003f, w13
    and.w   w13, w11, w11
    mov.w   [++w10], w12
    ior.w   w12, w11, w12
    mov.w   w12, [w10]
    mov.w   #0x0000, w0
    return

    .end
```

REFERENCES

1. Frahm, H., "Device for damping vibrations of bodies", US Patent 989,958, 1911.

2. Hartog, J., *Mechanical Vibrations*, Crastre Press, La Vergne, TN USA, 2007.

3. Korenev, B. and L. Reznikov, *Dynamic vibration absorbers: theory and technical applications*, Wiley, New York, 1993.

4. Sun, J., M. Jolly and M. Norris, "Passive, adaptive and active tuned vibration absorbers - a survey", *Transactions of the American Society of Mechanical Engineers*, Vol. 117, pp. 234–242, 1995.

5. Yong, C., D. Zimcik, V. Wickramasinghe and F. Nitzsche, "Development of the smart spring for active vibration control of helicopter blades", *Journal of Intelligent Material Systems and Structures*, Vol. 15, pp. 37–47, 2004.

6. Jacquot, R., "Suppression of random vibrations in plates using vibration absorbers", *Journal of Sound and Vibration*, Vol. 248, No. 4, pp. 585–596, 2001.

7. Wright, R. and M. Kidner, "Vibration Absorbers: A review of applications in interior noise control of propeller aircraft", *Journal of Sound and Vibration*, Vol. 10, pp. 1221–1237, 2004.

8. Kela, L. and P. Vähäoja, "Recent studies of adaptive tuned vibration absorbers/neutralizers", *Applied Mechanics Reviews*, Vol. 62, p. 060801, 2009.

9. Franchek, M. A., M. W. Ryan and R. J. Bernhard, "Adaptive Passive Vibration Control", *Journal of Sound and Vibration*, Vol. 189, No. 5, pp. 565 – 585, 1996.

10. Ryan, M., M. Franchek and R. Bernhard, "Adaptive-passive vibration control of single frequency excitations applied to noise control", *Noise Con 1994: Proceedings*

of the 1994 National Conference on Noise Control Engineering, pp. 461–466, 1994.

11. Hart, J., R. Sause, G. Ford and I. Brown, "Pipeline Vibration Damper", US Patent 5,193,644, 1993.

12. Cuvalci, O. and A. Ertas, "Pendulum as Vibration Absorber for Flexible Structures: Experiments and Theory", *Journal of Vibration and Acoustics*, Vol. 118, No. 4, pp. 558–566, 1996.

13. Setareh, M., J. Ritchey, J. Baxter and T. Murray, "Pendulum Tuned Mass Dampers for Floor Vibration Control", *Journal of Performance of Constructed Facilities*, Vol. 20, No. 1, pp. 64–73, 2006.

14. Chang, C., "Mass dampers and their optimal designs for building vibration control", *Engineering Structures*, Vol. 21, No. 5, pp. 454 – 463, 1999.

15. Hagood, N. and A. von Flotow, "Damping of Structural Vibrations with Piezoelectric Materials and Passive Electrical Networks", *Journal of Sound and Vibration*, Vol. 146, No. 2, pp. 243–268, 1991.

16. Filipovic, D. and D. Schröder, "Bandpass Vibration Absorber", *Journal of Sound and Vibration*, Vol. 214, No. 3, pp. 553–566, 1998.

17. Bonello, P., M. Brennan, S. Elliott, J. Vincent and G. Jeronimidis, "Designs for an adaptive tuned vibration absorber with variable shape stiffness element", *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Science*, Vol. 461, No. 2064, pp. 3955–3976, 2005.

18. Hirunyapruk, C., M. Brennan, B. Mace and W. Li, "A tunable magneto-rheological fluid-filled beam-like vibration absorber", *Smart Materials and Structures*, Vol. 19, No. 5, p. 055020, 2010.

19. Hoang, N., N. Zhang and H. Du, "A dynamic absorber with a soft magnetorheolog-

ical elastomer for powertrain vibration suppression", *Smart Materials and Structures*, Vol. 18, No. 7, p. 074009, 2009.

20. Deng, H., X. Gong and L. Wang, "Development of an adaptive tuned vibration absorber with magnetorheological elastomer", *Smart Materials and Structures*, Vol. 15, No. 5, p. N111, 2006.

21. Williams, K., G. Chiu and R. Bernhard, "Adaptive-Passive Absorbers Using Shape-Memory Alloys", *Journal of Sound and Vibration*, Vol. 249, No. 5, pp. 835 – 848, 2002.

22. Collinger, J., J. Wickert and L. Corr, "Adaptive Piezoelectric Vibration Control With Synchronized Switching", *Journal of Dynamic Systems, Measurement, and Control*, Vol. 131, No. 4, p. 041006, 2009.

23. Charette, F., C. Fuller and J. Carneal, "Adaptive vibration absorbers for control of sound radiation from panels", *Third Joint CEAS/AIAA Aeroacoustics Conference, Atlanta, GA.*, 1997.

24. Alabuzhev, P., A. Gritchin, L. Kim, G. Migirenko, V. Chon and P. Stepanov, *Vibration Protecting and Measuring Systems with Quasi-Zero Stiffness*, Applications of Vibration, Hemisphere Publishing Corporation, New York, 1989.

25. Wang, Y. and R. Lakes, "Extreme stiffness systems due to negative stiffness elements", *American Journal of Physics*, Vol. 72, No. 1, pp. 40–50, 2004.

26. Carrella, A., M. Brennan and T. Waters, "Static analysis of a passive vibration isolator with quasi-zero-stiffness characteristic", *Journal of Sound and Vibration*, Vol. 301, No. 3-5, pp. 678 – 689, 2007.

27. Lee, C., V. Goverdovskiy and A. Temnikov, "Design of springs with "negative" stiffness to improve vehicle driver vibration isolation", *Journal of Sound and Vibration*, Vol. 302, No. 4-5, pp. 865 – 874, 2007.

28. Teter, J. J., "A Discussion of Zero Spring Rate Mechanisms Used for the Active Isolation Mount Experiment", NASA/TM-1999-209723, 1999.

29. Kovacic, I., M. Brennan and T. Waters, "A study of a nonlinear vibration isolator with a quasi-zero stiffness characteristic", *Journal of Sound and Vibration*, Vol. 315, No. 3, pp. 700 – 711, 2008.

30. Robertsona, W., R. Wood, B. Cazzolato and A. Zander, "Zero-stiffness magnetic springs for active vibration isolation", *Proceedings of Active 2006 : the 2006 International Symposium on Active Control of Sound and Vibration*, 2006.

31. Qiu, J., J. Lang and A. Slocum, "A curved-beam bistable mechanism", *Journal of Microelectromechanical Systems*, Vol. 13, No. 2, pp. 137 – 146, April 2004.

32. Nathan, R. and B. Sheva, "A Constant Force Generation Mechanism", *Transactions of ASME*, Vol. 107, 1985.

33. Boyle, C., L. Howell, S. Magleby and M. Evans, "Dynamic modeling of compliant constant-force compression mechanisms", *Mechanism and Machine Theory*, Vol. 38, No. 12, pp. 1469 – 1487, 2003.

34. Liu, K. and J. Liu, "The damped dynamic vibration absorbers: revisited and new result", *Journal of Sound Vibration*, Vol. 284, pp. 1181–1189, June 2005.

35. Nagaya, K., A. Kurusu, S. Ikai and Y. Shitani, "Vibration Control of a Structure by Using a Tunable Absorber and an Optimal Vibration Absorber Under Auto-Tuning Control", *Journal of Sound and Vibration*, Vol. 228, No. 4, pp. 773 – 792, 1999.

36. Jacquot, R., "Optimal dynamic vibration absorbers for general beam systems", *Journal of Sound and Vibration*, Vol. 60, No. 4, pp. 535 – 542, 1978.

37. Thompson, A., "Optimum tuning and damping of a dynamic vibration absorber

applied to a force excited and damped primary system", *Journal of Sound and Vibration*, Vol. 77, No. 3, pp. 403 – 415, 1981.

38. Warburton, G. and E. Ayorinde, "Optimum absorber parameters for simple systems", *Earthquake Engineering & Structural Dynamics*, Vol. 8, No. 3, pp. 197–217, 1980.

39. Sun, H., P. Zhang, X. Gong and H. Chen, "A novel kind of active resonator absorber and the simulation on its control effort", *Journal of Sound and Vibration*, Vol. 300, No. 1-2, pp. 117 – 125, 2007.

40. Gómez, B., C. Repetto, C. Stia and R. Welti, "Oscillations of a string with concentrated masses", *European Journal of Physics*, Vol. 28, No. 5, p. 961, 2007.

41. Crawford, F., *Berkeley physics course: Vol. 3: Waves*, McGraw-Hill, New York, 1968.

42. French, A., *Vibration's and Waves*, The M.I.T. introductory physics series, CBS Publishers & Distributors, New Delhi, 2003.

43. Hibbit, Karlsson and Sorenson Inc., *ABAQUS User's Manual, v6.7*, 2007.

44. ASTM Standard A228, "Standard Specification for Steel Wire, Music Spring Quality", ASTM International, 2007.

45. Marghitu, D., *Mechanical engineer's handbook*, Academic Press series in engineering, Academic Press, San Diego, CA, 2001.

46. Eşkinat, E. and L. Öztürk, "Vibration Absorbers as Controllers", *4th European Control Conference Brussels, Belgium*, 1997.

47. Huang, S. and R. Man, "Active vibration control of a dynamic absorber using fuzzy algorithms", *Mechatronics*, Vol. 6, No. 3, pp. 317 – 336, 1996.

48. Huang, S. and R. Lian, "A Dynamic Absorber With Active Vibration Control", *Journal of Sound and Vibration*, Vol. 178, No. 3, pp. 323 – 335, 1994.

49. Nagarajaiah, S. and N. Varadarajan, "Short time Fourier transform algorithm for wind response control of buildings with variable stiffness TMD", *Engineering Structures*, Vol. 27, No. 3, pp. 431 – 441, 2005.

50. Silva, C., *Vibration monitoring, testing, and instrumentation*, Mechanical engineering series, CRC Press/Taylor & Francis, New York, NY, 2007.

51. Philips Semiconductors, *The I^2C-BUS Specification*, 2.1 edn., Jan. 2000.

52. Bailey, O., *Embedded systems: desktop integration*, Wordware Applications Library, Wordware Pub., Piano, Texas, 2005.

53. Minebea Co. Ltd., *Hybrid Stepper Motor*, `http://www.eminebea.com/content/html/en/hybrid_list/custom.shtml`, accessed at august, 2011.

54. Maske, R. and D. Woods, "Use of digital current ramping to reduce audible noise in stepper motor", US Patent 6208107, 2001.

55. Athani, V., *Stepper motors: fundamentals, applications and design*, New Age International (P) Ltd., Publishers, New Delhi, 2007.

56. Allegro MicroSystems Inc., *A3972 - Dual DMOS Full-Bridge Microstepping PWM Motor Driver*, `http://www.allegromicro.com/en/Products/Part_Numbers/3972/`, accessed at august, 2011.

57. Hyde, R., *Write Great Code: Understanding the machine*, No Starch Press, San Francisco, CA, 2004.

58. Jain, V., W. Collins and D. Davis, "DFT Interpolation for estimation of tone amplitudes and phases", *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP '80.*, Vol. 5, pp. 662 – 665, apr 1980.

59. Rexroth - Bosch Group, *Aluminum Structural Framing*, http://www.boschrexroth-us.com/country_units/america/united_states/sub_websites/brus_dcl/Products/Aluminum_Structural _Framing/, accessed at august, 2011.

60. Bosch Rexroth Corp, *Aluminum Framing Catalog, v6.1*, Linear Motion and Assembly Technologies, Bosch Rexroth Corp, Buchanan, MI, 2010.

61. Mitutoyo Corporation, *Micrometer Head Selection Guide*, Mitutoyo Corporation, Tokyo, 2005.

62. di-soric GmbH, *DCC 18 M 10/10 AIK-IBS Inductive proximity switch*, http://shop.di-soric.de/ shop/navigation/kart_200335/, accessed at august, 2011.

63. LabJack Corporation, *LabJack U12*, http://labjack.com/u12, accessed at august, 2011.

64. Sparkfun Electronics, *9DOF-Razor-v14*, http://www.sparkfun.com/datasheets/ Sensors/IMU/9DOF-Razor-v14.pdf, accessed at august, 2011.

65. Atmel Corporation, *ATmega328P*, http://www.atmel.com/dyn/products/product_card.asp/part_id_4198, accessed at august, 2011.

66. Microchip Technology Inc., *dsPIC30F4011/4012*, http://ww1.microchip.com/downloads/en/ DeviceDoc/70135e.pdf, accessed at august, 2011.