University of Southern Queensland
Faculty of Engineering & Surveying

Interactive Web-Based Signal Processing Tutor

A dissertation submitted by

TOH PEI LING

in fulfilment of the requirements of

ENG4112 Research Project

towards the degree of

Bachelor of Electrical and Electronic Engineering

Submitted: October, 2004

Abstract

This dissertation report details an account of developing an interactive online learning tool on Digital Signal Processing (DSP) content which facilitates students and industry group. Basically, it helps to establish a clear understanding on what the whole project is about, how it was developed, the testing and results, the problem met and lastly the further work recommendations.

Development of the web tool was carried out by first researching on related Internet topics and looking out for suitable software tools and languages to be used. Some brainstorming was also done on possible features which users would prefer. This was followed by the narrowing down of the necessary functions the project would be providing. After which the web tool was build up and coded. Testing was carried out regularly to iron out any errors. The resulting web tool had satisfies the goals of the project although its functionality was faintly imperfect due to insufficient time - thus the web tool usability criterion was pleasing. Nevertheless, opportunities for further work are feasible and it is recommended for future enhancement.

University of Southern Queensland

Faculty of Engineering and Surveying

ENG4111/2 *Research Project*

Limitations of Use

The Council of the University of Southern Queensland, its Faculty of Engineering and Surveying, and the staff of the University of Southern Queensland, do not accept any responsibility for the truth, accuracy or completeness of material contained within or associated with this dissertation.

Persons using all or any part of this material do so at their own risk, and not at the risk of the Council of the University of Southern Queensland, its Faculty of Engineering and Surveying or the staff of the University of Southern Queensland.

This dissertation reports an educational exercise and has no purpose or validity beyond this exercise. The sole purpose of the course pair entitled "Research Project" is to contribute to the overall education within the student's chosen degree program. This document, the associated hardware, software, drawings, and other material set out in the associated appendices should not be used for any other purpose: if they are so used, it is entirely at the risk of the user.

Prof G Baker

Dean

Faculty of Engineering and Surveying

Certification of Dissertation

I certify that the ideas, designs and experimental work, results, analyses and conclusions set out in this dissertation are entirely my own effort, except where otherwise indicated and acknowledged.
I further certify that the work is original and has not been previously submitted for assessment in any other course or institution, except where specifically stated.

TOH PEI LING

D11337628

_____

Signature

_____

Date

Acknowledgments

There are many people that I would like to thank for their contributions to the successful completion of this project, and not all are mentioned here.

Firstly, I would like thank my supervisor, Dr John Leis. He provides resourceful information and suggestions as guidance to the project. His constant encouragement and prompt response help to facilitate the progress of the project too.

Also like to extend my thanks to my friends whom have actively participated in the event of feedback and suggestions for the project. Their kind assistance is greatly appreciated. Last but not least, to my fiance, Daniel, for his great support and understanding during my dissertation.

TOH PEI LING

University of Southern Queensland
October 2004

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Project Overview

Digital Signal Processing or DSP is not an easy subject to learn or master, many students encounter difficulties when studying for it, and even industry engineers sometimes have trouble understanding or remembering the concepts. Hence the idea to develop a learning tool for conducting DSP content through a multimedia platform. In this case, the intended content will be delivered to a personal computer using a combination of text, graphics, and animations in the form of an online website. This will allow it to be accessible by many users.

In the later sections of this dissertation, other issues and aspects involved for developing the project will be further discussed.

1.2 Project Aim

The challenge presented would be to provide a web-based interactive learning tool with selected topics in DSP. The intended audience is primarily students

and industry engineers with basic foundation of DSP. The key idea is to assist them in learning DSP topics in a multimedia environment, or to simply refresh existing concepts. As this tool can be easily accessed from the Internet, users can learn at their own comfortable pace and choose their interested topics with ease. Furthermore, the project also aspired to attract or interest users with more animated illustrations and mini quizzes without the need to install addition software.

In order to achieve the above aims mentioned, several important objectives and specifications of the project were accomplished. They are further discussed in the later sections of this dissertation.

1.3 Nature and Scope of Problems

DSP is actually the algorithms, mathematics, and the techniques used to manipulate the signal such as sound waves, visual images, after they have been converted into digital form. The final intended outputs can have a wide variety of uses, for instance, enhancement of visual images, recognition and generation of speech, compression of data for storage and transmission.

As suggested, the topic DSP has increasingly become an important subject to study and can be interesting and challenging to learn. However, people who have studied this topic know that it is not a subject that can be grasped easily, especially for students who may face difficulties understanding its concepts or even industry engineers who may have forgotten rarely touched areas.

There are two parts to the learning process; learning the general concepts that apply to the field as a whole, and learning specialized techniques for a particular area of interest. As there is a wide range of topics available and not all of them being critical, it would be helpful to narrow down the selection of DSP topics. Also, it would be a crucial task to impart general DSP concepts as it is very

important to grasp the fundamentals since without this, it would be difficult to relate basic concepts or terms to the more specialized sections of DSP.

1.4 Dissertation Outline

This dissertation is divided into an assortment of chapters that describe the different portions of the project. To better facilitate the reader to read this dissertation, this section provides a brief description on what each chapter provides.

Chapter 2 DSP Education and Relevant Literature defines the various concepts and reviews the literature relating to this dissertation. It examines literature on the DSP concept, and its importance. It also introduces the concept of the Internet and its advantages. Other information such as HTML, Dynamic web pages and Scripting Languages are briefly discussed too.

Chapter 3 DSP Educational System: Design and Development Issues was an important phase as the milestones and design decisions are made here. This was to ensure that nothing major is overlooked and that all ideas were consolidated into a set of guidelines for the project to work towards. Topics on this chapter are the research works, critical evaluation on the implementation methods, the user specifications and other planning tasks.

Chapter 4 User Interface Design and Implementation illustrates in detail how tasks such as understanding and constructing DSP concept, learning how to use the new software tools, development of the interactive and animated pieces, and the user interface of the project are done, in a step by step process.

Chapter 5 Testing Methodology and Results examines the methods used for the testing of the project. It details all the steps taken to conduct

the test to ensure the project performed well. Multiple screenshots of the website provides a full view on what actually have been accomplish which are the results.

Chapter 6 Further Work and Conclusion gives a brief description of the problems encountered during the development of the project. It then evaluates if the aims of the project have been met and highlights areas in which this project can be further improved.

Chapter 2

DSP Education and Relevant Concepts

2.1 DSP Education

Digital Signal Processing, or DSP [4], as the term suggests, is the study of signals in a digital representation and the processing of signals by digital means. It is an area of science and engineering that has developed rapidly over the past 30 years with the transformation of data such as signals, images, and video to better transmit or use the information. It is one of the most powerful technologies where revolutionary changes have opened up a broad range of fields such as communications, medical imaging, radar sonar, high fidelity music reproduction, and oil prospecting, just to name a few. Each of these areas makes use of DSP technology; with its own algorithms, mathematics, and specialized techniques.

With advancing DSP technology, a wide range of DSP applications have been developed, such as mobile phones, personal computers, CD players and modems. Figure 2.1 which is on Page 6 illustrates a few of these applications. Modern society places a strong emphasis on them as these applications improve human

living standards, making their life more interesting and comfortable.

Likewise, the world will encounter more advanced technology such as digital cameras and television, multimedia, wireless networks, and speech recognition, as we progress through the information age. These industries are likely to bloom and as a result the high demand in these applications suggests that DSP is an essential subject to study. Knowledge of DSP not only helps in job opportunities but it also serves as basic knowledge to understand how modern applications work behind their fascinating cover.

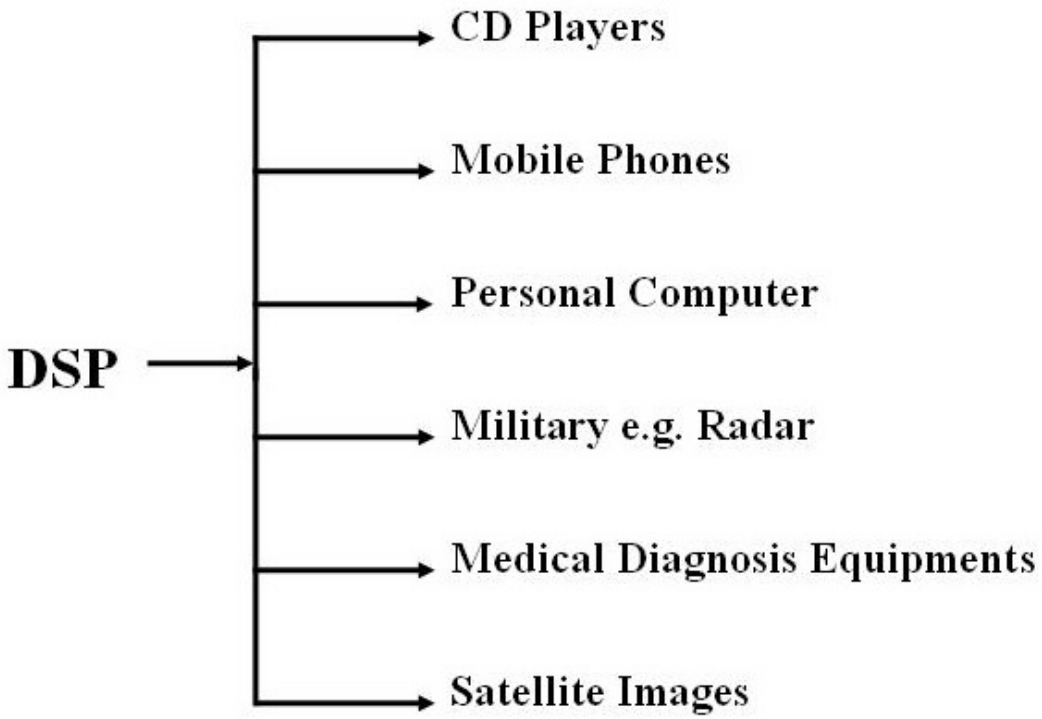

Figure 2.1: DSP has revolutionized many areas in science and engineering. A few of these diverse applications are shown here.

2.2 The Internet Concept

The Internet can be thought of as an abstract space of information, which has been used to share information with millions of users around the world in the

form of text, graphics and video – a channel for communication. Often described as a massive collection of networks linked across the world, the Internet enables communication and sharing of services directly and transparently.

As a powerful tool of communications, it is fast, convenient, widely used throughout the developed world, and it is still growing at a seemingly exponential rate. The most widely used part of the Internet is the World Wide Web, also known as WWW for short. Its outstanding feature is hypertext, a method of instant cross-referencing. In most Web sites, certain words or phrases appear in text of a different colour than the rest; often this text is also underlined. When you select one of these words or phrases, you will be transferred to the site or page that is relevant to this word or phrase. Sometimes there are buttons, images, or portions of images that are "clickable." If you move the pointer over a spot on a Web site and the pointer changes into a hand, this indicates that you can click and be transferred to another site. This interactive multimedia on the World Wide Web (www) is a phenomenon that has been emerged for quite some time. Therefore the widespread use of the Internet suggests that it is an appropriate opportunity to introduce a learning tool such as interactive web-based multimedia to teach DSP concepts and its various algorithms. An online educational tool in the form of a website makes it possible for more people than ever to access materials and to learn things in a new and different environment. It allows education to be brought to users, rather than the other way round. In addition, a website provides flexibility, and does not required users to install any additional software in order to use it.

Using the Web, you have access to millions of pages of information. Web browsing is done with a Web browser, the most popular of which are Microsoft Internet Explorer and Netscape Navigator. The appearance of a particular Web site may vary slightly depending on the browser you use.

2.3 HTML

The web pages that are accessible to us on the Internet are said to be in HTML format but what does this actually mean? HTML, the Hypertext Markup Language, is the principal format used for creating web documents. HTML is a set of platformindependent instructions that indicate to a browser how and where elements in a web page like text, graphics and links are placed.

2.4 Dynamic Web Pages

There are two types of web pages on the Internet. The first type is the static ones meaning that they usually display the same information and/or presentation. The areas of concern here are the age of the data as well as its format. If the information given changed frequently, the validity of the data would then depend on how often the web administrator can update the page, which itself would be a tedious task. Also, the readability of the information presented weighs on the layout of the page.

The other type of web page is known as a dynamic web page which is basically a reactive web page that interacts with the user and allows them to customize their viewing experience. These web pages usually possess some or all of the following qualities:

- Interactive forms.
- Ability to sort and search for data.
- Ability to extract data stored in multiple layers.
- Ability to choose the parts of data to be displayed.

There are roughly 3 methods of generating dynamic web pages:

1. Generation of static HTML pages directly from a database - As the name implies, data is retrieved from the database and formatted in HTML before being displayed in the browser. This is usually used for web sites that require updated information from a database but wish to restrict the format or amount of information that a user can access. This type of web page usually does not allow any form of interaction between the user and the database.

   Advantages

   - Highspeed access.
   - Low development cost.
   - Creation of data generation tools does not require programmer knowledge.

   Disadvantages

   - Low flexibility for user.

2. Generation of individual pages dynamically using a web server each time the user requests the information - The structure of this method is similar to the first one with the difference of increased user flexibility in that format and amount of information can be specified, i.e. the user may choose to display information in a preferred layout or filter out results based on certain criteria. This web page also allows a user to make changes in the database indirectly.

   Advantages

   - Good speed access.
   - Creation of data generation tools requires programmer knowledge.
   - Maximum flexibility.

   Disadvantages

   - High development cost.

3. Generation of individual pages dynamically using the browser. This method does not require the use of the database but instead uses scripting languages such as Javascript and ActionScript which are able to interface with the browser to process information from user input forms. As such, the data can be generated quickly without other programs since everything is stored and processed within the web page but with the added flexibility of user entries. Examples of such web pages would include online metric converters, language converters etc.

   Advantages

   - High speed access.
   - Usually low development cost.
   - May or may not require programmer knowledge depending on if custom or default tools are used respectively.
   - Maximum flexibility.

   Disadvantages

   - Since there is no link with a database, this type of dynamic web page cannot store large amounts of information.

Generating dynamic web pages normally requires the use of additional software besides the standard browser, namely scripting languages to provide interactivity for the web page, server software for hosting web pages and database software to store information that is to remain persistent, i.e. information that does not go away when a user leaves the web site or closes the browser.

For the case of this project, the applicable method lies with method number three. As this project deal with only interactive forms and no database was required.

2.5 Scripting Languages

Scripting languages are similar to conventional ones in that they can both be used to create programs although the ones made by scripting languages are limited in terms of scope and execution efficiency. This is due to the reduced syntax available as well as the need for the code to be executed by an intermediate program. However despite these problems, scripting languages are commonly used for creating dynamic web pages since they're easy to learn and still contain enough functionality to flesh out interactive features and format contents. Some of the more popular ones are listed below. For this project, the scripting language used was ActionScript.

2.6 Chapter Summary

This chapter has provided a general knowledge on DSP concepts. It has introduced some background information on related elements such as the Internet, HTML, dynamic web pages, and the scripting languages. With some understanding of these basics, it can serve as a useful tool on future research or development.

Chapter 3

DSP Educational System: Design and Development Issues

3.1 Methodology

To effectively manage the project in terms of time and resources, a set of guidelines was drawn up together with an accompanying flowchart as shown in Figure 3.1on Page 14. The methodology produced was used for the entire duration of the project, with the different phases being denoted using milestones.

With the basic understanding of the requirements and the scope of problems, research work was carried out by concentrating on information gathering activities, such as reviewing existing websites with similar content or purposes, and to study the implementation methods used.

After this was done, implementation tools or software such as Adobe Photoshop, Macromedia Dreamweaver MX 2004, Macromedia Flash MX 2004, and ActionScript were selected to be evaluated further. These tools were selected for their availability and capabilities based on related materials such as reference books

and online tutorials.

Besides implementation tools, other information and resources such as popular DSP topics and helpful features offered by current existing websites were also taken into consideration.

The next phase to be done was to include the selected resources in the planning and design stage of the program structure, so as to give the project direction in its development path. After the project has been successfully implemented, a testing phase will be carried out to ensure that the project is performing according to the specifications. This phase will consist of seeking feedback from others as well as self-evaluation of the project. Once the testing is completed, final adjustments such as improvements and corrections will be made.

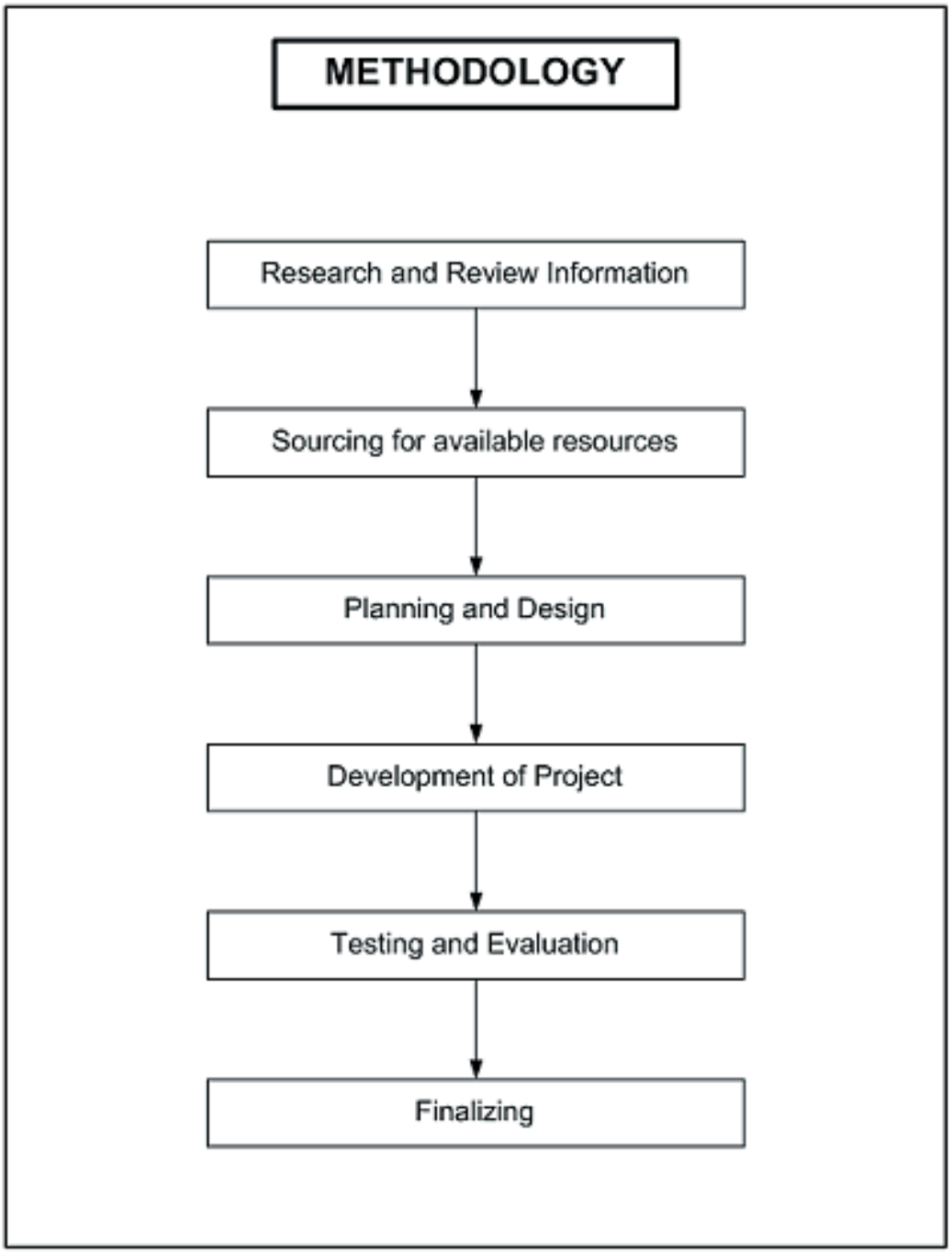

Figure 3.1: Flowchart for methodology

3.2 Existing Works

The research was carried out mainly by looking for similar existing websites posted online. As expected, there were quite a number of DSP websites available. Of the websites visited, some of them possessed good features and information. While others have poor organisations skills with their content scattered around without any logical sequence.

The various websites offers diverse features of teaching. In some areas they are worth to learn from whereas some could be further improved. For example, some websites had interactive illustrations which allowed the user to key in certain parameters and tabulate the required algorithm and diagrams. This is a good feature to resolve users' problems. However, these websites usually conduct very in-depth DSP content and tend to be very technical. Hence, this becomes rather difficult for students or those who were not strong in DSP concepts to understand. On the other side of the spectrum, some of the website targeted a selected number of topics but brief explanations were given on these topics and there was little in-depth discussion on them. Thus, these websites are not very helpful to users too. As a result, those useful and interesting features of the websites visited are noted down for devising the features for the project. Refer to the References section for the list of websites visited.

3.3 User Specifications

As a DSP student myself, there were times where frustrations arises when difficulties were met in the midst of study and help provided was quite limited. Especially for long-distance learning students, a fully-equipped library is not available and seeking for long-distance helps from lecturers sometimes took quite an amount of time.

Hence it is likely for such students to turn to the Internet to look for information by simply searching for the topics of interest using a search engine, e.g. Goggle, Yahoo, and Excite. However the content provided by these websites is sometimes too brief or technical for the average student to understand.

As such, a list of preferred learning choices and methods has to be prepared by gathering them from the users themselves, i.e. the students.

3.3.1 Specifications

To meet the criterion of the users' specifications, firstly, the content for the website should provide a reasonable amount of explanations on the topics. The topics covered in the website provide an overview on DSP concepts which is important for further readings on specialised DSP topics. Beside just static diagrams and text explanations, animated graphics and quizzes were included in the specifications of the project. In this way it would help to increase the interactivity level with the users and to enhance their understanding during the learning process. The presentation of the information was equally important and the website had to be designed to be user-friendly enough to assist users to navigate around the website and absorb the information with ease.

A list of detailed project specifications was formulated for this project for better reference and is illustrated in Appendix A.

3.3.2 System Requirements

Another important factor was the minimum computer requirements the users' need in order to access the website. The website requires the Flash Player for the playback of animated graphics and quizzes.

Many browsers provide support for Netscape and Internet Explorer plug-ins and ActiveX controls. Usually, the Flash Player can be installed in these browsers in the same manner as with Netscape and Internet Explorer. Although support for functionality depends on the browser, users can usually download and install the Macromedia Flash Player directly from the Macromedia website.

The most widely used internet browsers currently are Internet Explorer, Netscape and Mozilla for Linux users. The following tables show the minimum system requirements for each different internet browser respectively.

Table 3.1: Minimum requirements for Internet Explorer Browser. Adapted from `http://www.microsoft.com/windows/ie/evaluation/sysreqs/default.mspx`

tab:IEtable

Computer/Processor	Computer with a 486/66-MHz processor or higher (Pentium processor recommended)
Operating System	Microsoft Windows 98, Windows 98 Second Edition, Windows Millennium Edition (Windows Me), Windows NT 4.0 with the high encryption version of Service Pack 6a (SP6a) and higher, Windows 2000, or Windows XP
Memory	For Internet Explorer 6 SP1: RAM requirements depend on the operating system used Windows 98: 16 MB of RAM minimum Full install size: 11.5 MB Windows 98 Second Edition: 16 MB of RAM minimum Full install size:12.4 MB Windows Me: 32 MB of RAM minimum Full install size: 8.7 MB Windows NT 4.0 with SP 6a and higher: 32 MB of RAM minimum Full install size: 12.7 MB Windows 2000: 32 MB of RAM minimum Full install size: 12 MB Windows XP: 32 MB of RAM minimum Full install size: 12 MB

Table 3.2: Minimum requirements for Netscape Browser. Adapted from `http://channels.netscape.com/ns/browsers/sysreq.jsp`

tab:Nettable

Windows Operating Systems	Windows 98 Windows 98SE Windows ME Windows NT 4.0 Windows 2000 Windows XP
Minimum Hardware	Pentium 233 MHz 64 MB RAM 26 MB of free hard drive space
Mac Operating Systems	Mac OS X 10.1.x Mac OS X 10.2.x and later
Minimum Hardware	PowerPC 400 MHz G3, G4, 256 MB RAM 60 MB of free hard disk space
Linux Operating Systems	Linux kernel -2.2.14 with the following libraries or packages minimums: glibc 2.2.4 gtk +- 1.2.0 (1.2.5 or greater preferred) XFree86-3.3.6 Glib 1.2.x Supported Platforms: Netscape has been certified and is supported on Red Hat Linux 7.0 and greater.
Minimum Hardware	Pentium 233 MHz 64 MB RAM 26 MB of free hard drive space

Table 3.3: Minimum requirements for Mozilla Browser. Adapted from `http://www.mozilla.org/products/mozilla1.x/sysreq.html`

tab:Nettable

Windows	Windows 95
Operating Systems	Windows 98
	Windows 98SE
	Windows ME
	Windows NT 4.0
	Windows 2000
	Windows XP
Minimum Hardware	Pentium 233 MHz
	64 MB RAM
	52 MB of free hard drive space
Mac	Mac OS X 10.1.x
Operating Systems	Mac OS X 10.2.x and later
Minimum Hardware	G3, 266 MHz
	64 MB RAM
	72 MB of free hard disk space
Linux	Linux kernel -2.2.14 with the following
Operating Systems	libraries or packages minimums:
	glibc 2.2.4
	gtk +- 1.2.0 (1.2.5 or greater preferred)
	XFree86-3.3.6
	Mozilla 1.7.3 has been tested on Red Hat Linux 7.0 and later
Minimum Hardware	Pentium 233 MHz
	64 MB RAM
	52 MB of free hard drive space

3.4 Design Choices

With a greater understanding on the users' requirements and notes jotted down from the existing works, the project design choices were drawn up closely to satisfy these requirements. The below sections will discussed in greater details what were the project design choices and why they were selected.

3.4.1 Selection of DSP topics

Beside the features were jotted down, website with good topics arrangement were taken note of. However, for building on the DSP content, books were an excellent alternative to look into. Here were some fruitful findings, which can consider as a guide for building the DSP content.

Three textbooks have influenced the choice of topics and their contents. The first being the University's textbook by John Leis [2] as it provides more MATLAB based introductions to DSP and therefore has a more hands-on approach, which helps the student to understand the concepts of DSP more easily. In addition, the topics arranged in the textbook are in sequence, letting the students build up their foundations properly before moving on to the more difficult concepts. The book also focuses on the application aspects of DSP and thus making it easier to visualize the link between the theory and its practical uses.

The second textbook by Ifeachor and Jervis [1] provides a wide range of DSP topics and in great detail, and provides program samples of DSP concepts in the C language. Several mathematically based examples are also given, and in general the textbook is very technical in nature. Hence the use of this textbook was to compare with other sources to ensure that the information was similar and it was also used as a reference guide.

The last textbook [3] gives an introduction to DSP, and provides a similar mix

of topics to the first book but in greater detail. It is written in a simpler way which is easier to understand especially for students and is used whenever there is difficulty understanding any topics in the first book. The book also provides questions and answers at the end of each chapter and would be helpful when designing the website's quiz sections.

In addition to these, a list of websites was also used as reference material. Both text and online resources can be found in the References section.

The bulk of the content in the project were mainly obtained from the above mentioned reference books, textbooks, online resources and also from friends with DSP background. It was noted that making a proper introduction to DSP and its fundamentals was important; most textbooks start off with an introduction of DSP and feedback from friends also mention that the basic concepts of DSP must be well introduced. This is also helpful to industry personnel who may be interested in a topic but can also make use of this website to refresh their knowledge since most of them would have forgotten what they have learned in the past. On the whole, the project consists of the three chapters outline as below.

Chapter 1 - Introduction to Signal Processing As the title suggests, it explains in layman terms what Signal Processing is about. The advantages of DSP and their applications were also illustrated so that users can better relate DSP to the common applications that they use.

Chapter 2 - Typical real-time DSP system This chapter talks about the concept of DSP and introduces a typical real-time DSP system. It starts off with the explanation of the differences between the analogue and digital signal with the use of animated diagram. The different types of concepts for the conversions of signal from one form to another are also introduced which include: The sampling theorem, Aliasing and Quantization. Students usually start to get confused and uncertain at this point and hence there are some exercises and quizzes used to

test their understanding.

Chapter 3 - Random Signals The last chapter discussed signals in greater detail, in particular random signals. This is a more applicable chapter as most signals are random signals in the real world. This chapter provides an introduction on these types of signals and how to deal with them. It will be of interest for users to analyze something that they may have been dealing with everyday.

3.4.2 Critical evaluation on implementation methods

The following paragraphs describe the critical evaluation on the selected implementation methods and include the functionality of these tools. They were namely, Adobe Photoshop version 6.0, Macromedia Flash MX 2004 with ActionScript, and Macromedia Dreamweaver 2004.

Adobe Photoshop 6.0

The design of illustrations and graphics can be done using a variety of software, such as Adobe Photoshop and Illustrator. However, having prior experience in using Adobe Photoshop was a major factor in the selection criteria, not to mention the fact that it is also well known program used in many website designs.

This software is a popular photograph/ image software editing tool which provides powerful vectors for editing and masking, layer controls and styles, and has a large number of ways to manipulate fonts and text.

Macromedia Flash MX 2004

The majority of the websites visited used Java applets to present their illustrations. Java is actually a multi-platform language. It offers a standard, uniform

programming interface to applets and applications on any hardware, making it suitable for the Internet, where a program should be capable of running on any computer in the world.

Although Java has more powerful functions and is more flexible. It was considered that a non-Java method of implementation would open the system to a potentially much wider audience, and reduce the configuration/ setup problems that might be encountered. Browsers require a plug-in called Java Virtual Machine (JVM) in order to run Java applets. While Java is designed to run on all platforms and operating systems (OS), the version of the API used to program an applet determines how compatible it will be with web browsers; older ones may not be able to run applets that are built on newer APIs such as 1.4.

Also, as graphical functions are difficult to implement in Java, Macromedia Flash was chosen as a replacement as its interface is much more intuitive to use, and that the user can visualize how the graphics come together to form the entire website. This software is able to create effective rich content across desktops and devices and most importantly, adds on interactivity with powerful scripting language and enhances the content with custom effects from third-party extensions. ActionScript which is the powerful scripting language that is inbuilt within Macromedia Flash is the main key software tool to develop the interactive portions of the project.

Another point to define is the difference between Macromedia Flash and Shockwave Players. Macromedia Shockwave and Flash Players are both free Web players from Macromedia. Macromedia Flash Player is used to play back applications created with Macromedia Flash. Macromedia Shockwave Player is used to play back applications created with Macromedia Director Shockwave Studio. These two players are different pieces of software used for viewing different file types. The browser will automatically load or prompt to download the appropriate player when required.

Macromedia Dreamweaver MX 2004

HTML code is created nowadays by using 'what you see is what you get' (WYSIWYG) editors which allows one to lay out the website elements in the form of a Microsoft Word document - the editor then automatically generates the necessary code. Examples of such editors are Microsoft Frontpage and Macromedia Dreamweaver MX.

Both programs are editor tools for making web pages and building web sites. Hence to choose between them, some research was performed and the general idea formed was that primarily, there is no major difference in using either program. The goals and the approaches are the same. The only difference is that they are two different sets of tools.

FrontPage is a simplistic and straight forward program, more appropriate for anyone who wants to build a web site without fancy architecture. It is suitable for small-business owners who want to advertise their merchandise, and those who like to share their experience and ideas in writing with the world.

On other hand, Macromedia Dreamweaver MX is more towards those who wants more professional graphics presentations yet do not even have the slightest ideas how and where to start . Macromedia Dreamweaver MX also comes with specialized templates that allow the user to insert pre-built functions into a page, such as a quiz template. This is also a popular HTML editor available on the market which appeals to all levels of users as it provides a powerful combination of visual layout tools, application development features, and code editing support. This tool keeps all the complicated html code well hidden and works in a design-based view, which is great for a beginner.

To actually gauge the tools for their suitability, a hand-on was carried out for both programs to see which would be more comfortable to use. To save time, a simple page was created to test how user-friendly the software was. The conclusion was

that Macromedia Dreamweaver MX was more comfortable to work with and its functions work hand in hand with Macromedia Flash MX since they are in the same package. Hence the merging process would be easier to manage. Thus, Macromedia Dreamweaver MX was chosen to be the web developer tool for this project.

As a novice to these two software tools and ActionScript programming, some reference books and online resources were utilized to learn their functions and how to use them. These resources are listed in the reference section in the Appendices.

3.4.3 Website Theme Layout

An interesting task was to design a theme for the website. In this case, the website theme refers to a standard design layout, the colour schemes, and the overall interface look of the website. Although the graphical design section may not appear to be a crucial factor in this project, an attractive website will improve the readers' attention as well and stimulate their interest. A website should have a consistent and matching theme throughout as it will provide the user with a sense of belonging and the website will look well organized.

It is important for a website theme to provide the user with a comfortable reading environment and to keep them interested enough to revisit the website. From observation, the appearance of the majority of websites on DSP was not very appealing. Therefore it was important to note down the negative aspects and to avoid them. Besides visiting the related websites, other educational websites were explored too, mainly to take note of their interesting design theme and the different styles of presenting information.

From the current websites and surveys obtained from friends, a preferred website layout was created in a way such that each page was of one screen size so that the users need not to scroll a long way down and to reduce the 'wordiness' feel of

the page. The standard layout size (Screen size) was decided on 1024 by 768 as it was common on most personal computers. Other elements such as the theme, navigation buttons and links, the font type and the graphics that was used for the project was decided during the development as it would be easier to judge if it matches the whole layout.

3.5 Resources Planning

After all the design and development issues were decided, the next thing to begin with was resources planning. Here, the main program structure provided in Figure 3.2on Page 28. was taken into consideration and the requirements for different parts of the program and the most suitable implementation methods are examined. Lastly, the amount of resources needed had to be calculated and divided among the different stages. The issue of having a timeline to keep track of the progress of the whole project was also examined.

The whole program was divided into three major parts, namely design, user interface and the content. It is clearly shown that each individual part concentrates on different technical issues and that their essential resources requirements were different too. Hence, planning for each part is examined and detailed in this section with reference to the Figure 3.2.

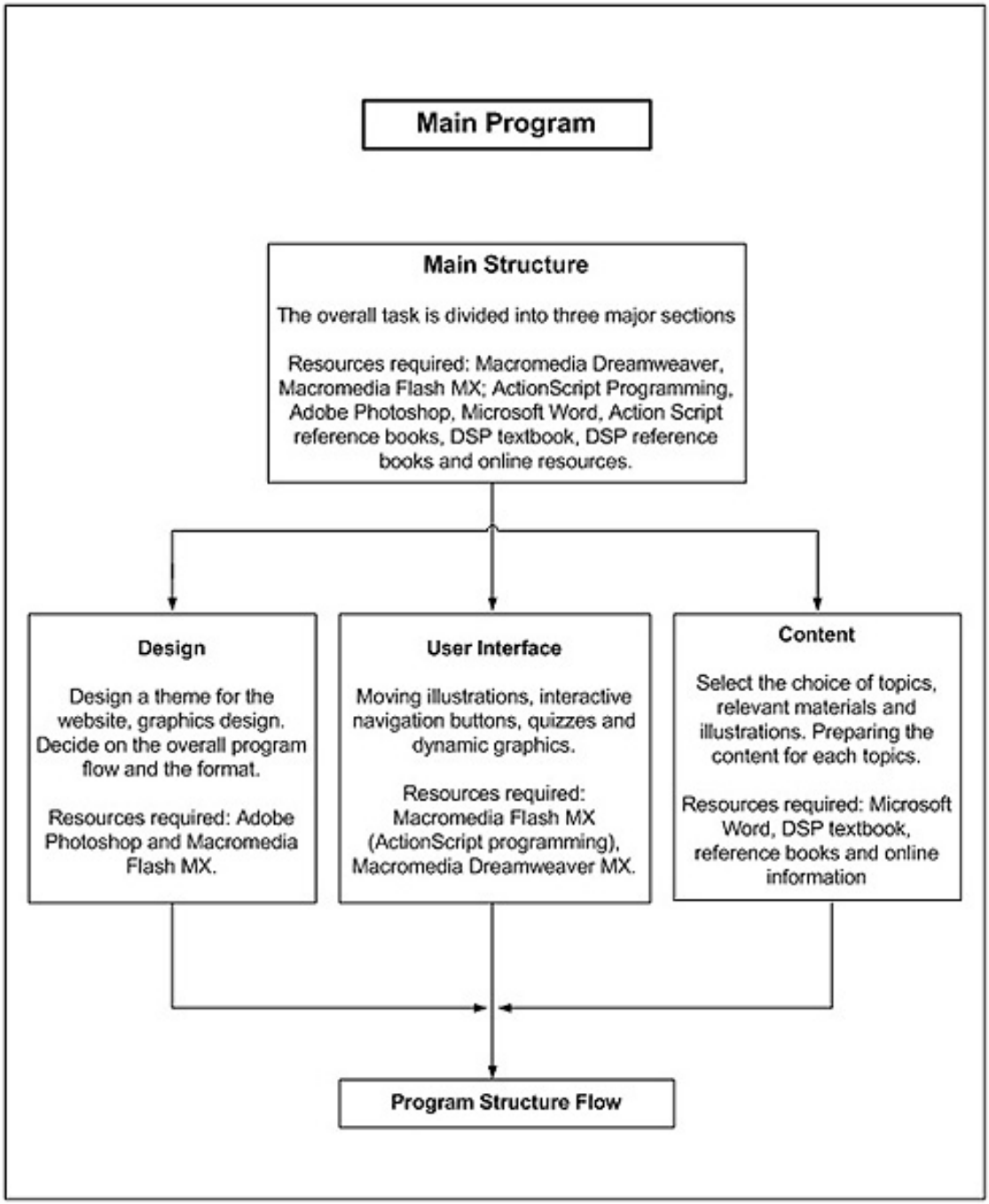

Figure 3.2: Flowchart for the main program.

3.5.1 Design

The design section determines the overall project interface look such as the theme design for the website, navigation buttons, pictures and diagrams for illustrations. Besides this, the overall program flow and the interactivity of the project, such as where should the mini quizzes appear in each chapter, and the appropriate illustrations for the topics were taken into consideration as well. The software resources required for the first section (Design) were namely, Adobe Photoshop 6.0, Macromedia Flash MX 2004 with ActionScript programming.

3.5.2 User Interface

The next section (User interface) can be said to be one of the most important parts of the specification. All the interactive functions such as the moving illustrations for demonstrations, mini quizzes with explained solutions and the positioning of the navigation buttons linking all pages are done here. It requires primarily Macromedia Flash MX 2004; ActionScript programming to create the animated illustrations, interactive mini-quizzes and dynamic graphics such as the interactive learning game. These are then implemented in Macromedia Dreamweaver MX 2004 to form a complete chapter.

3.5.3 Content

The last section (Content) involved more research. For example, reading and understanding of DSP concepts was involved and a choice of topics for the project target audiences was selected. Hence, resources such as DSP texts, reference books, and online resources were essential for building up reliable content and illustrations for the project. This is followed by adjoining the user interface and graphics created in the earlier two sections and by merging them together to create

the website. Appropriate software such as Microsoft Word and Adobe Photoshop 6.0 were used to create illustrations and furnish the interactive demonstration and quizzes with their actual contents.

3.6 Timeline Planning

Another important task is the planning of milestones and objectives to be accomplished for the project duration in the form of a timeline. This is done so as to oversee the progress of the whole project such as important deadlines, and serve as a guideline for estimating the approximate time frame allowed for each task. On the whole, the timeline was scheduled according to the Methodology flowchart in Figure 3.1on Page 14 and the different sections were made into the major milestones. The scheduled timeline for the whole project which is extended for both ENG4111 and ENG4112.

In general, more time was allocated for learning the software tools and developing the project. This is because most of the tools were new and unfamiliar and therefore required more time to learn them. Some of the tasks were planned to run concurrently with one another due to time constraint but for most part, the tasks were usually done sequentially. The timeline schedule details the concurrent processes; the task that was required to be completed first before proceeding to the next one is shown under the "Predecessors" column. These tasks may require more attention as one task delayed may affect the rest behind the line.

3.7 Program Structure Flow

A program structure flow is drawn out as shown in Figure 3.3on Page 31. This flowchart shows the logical process flow of how a user can accesses and navigates around the website in order to obtain its contents with ease. It also provides a clear picture of what the physical project will be providing in each page and how the pages were linked. With the reference of the flowchart, the development of each progress can be easily identified.

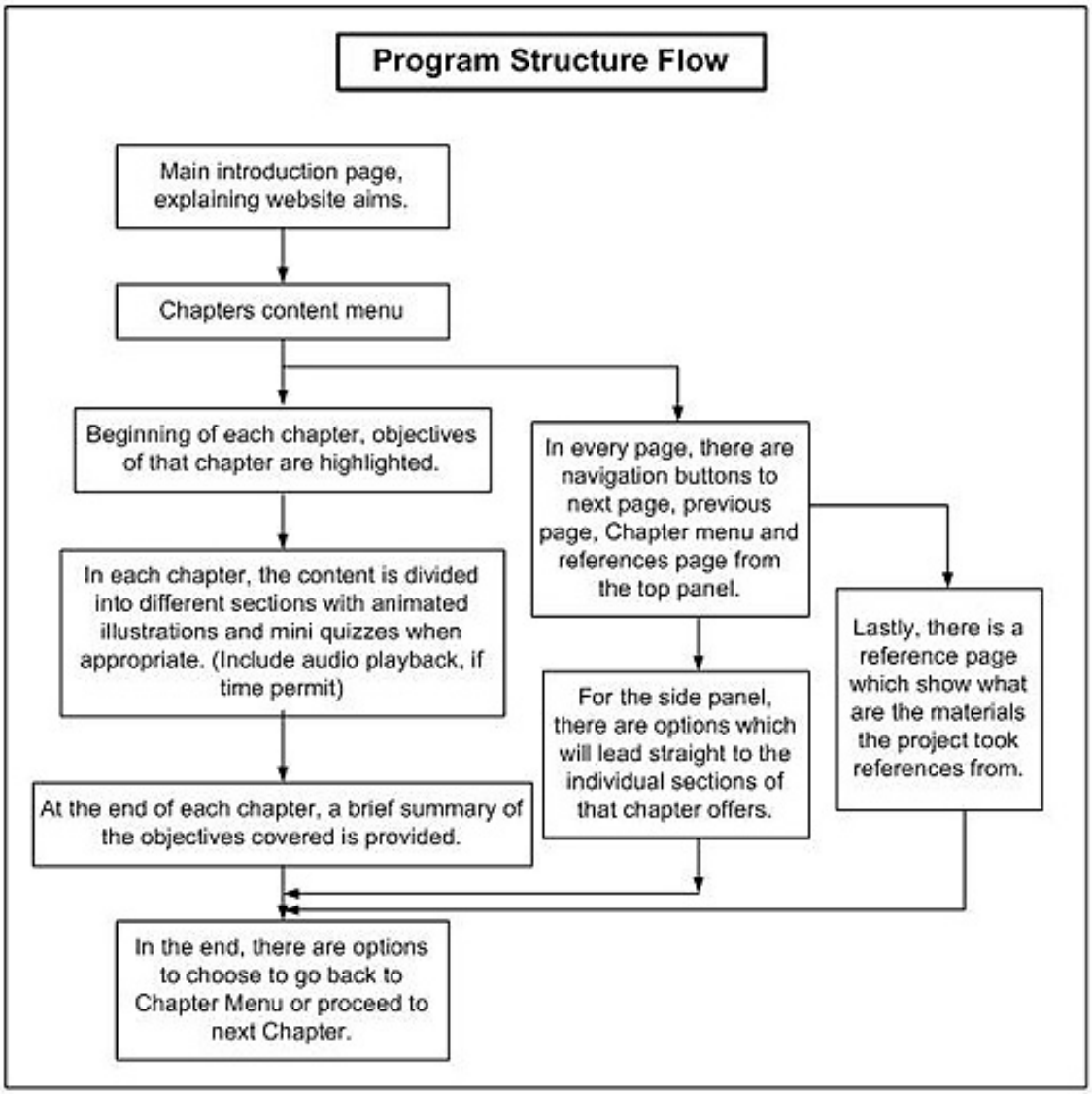

Figure 3.3: Flowchart for the program structure.

3.8 Chapter Summary

By the end of this chapter, the planning and guidelines regarding the development of the project were well defined. The overall methodology employed was drawn out in flowchart to direct the path of project management. In the design and planning section, the researched information was sorted and consolidated into a list of users' preferred design choices. From this list, the resources were then further studied. The implementation methods that were used were given descriptions and critical reviewed. Lastly, the program flow was also designed and prepared to ease the development path.

Chapter 4

User Interface Design and Implementation

4.1 Construction of Content

The first task in the implementation stage was studying and constructing the DSP content for the project. Through the list of the DSP topics selected in the design stage, the content for each topic was then further looked into.

There was a standard procedure for constructing each chapter. Firstly, some efforts were put in on researching for relevant information on the topics of interest. Information was obtained from reference and text books and online resources detailed in section 3.4.1, Project Research section, or refer to References section for the list of references materials. Next was the time consuming task of reading and understanding the required reference materials. Based on this understanding and the support of the DSP theory researched, a set of consolidated DSP content for the project was recomposed onto Microsoft Word. Suitable examples of illustrations, useful exercises and questions were handpicked during this process as well.

The golden rule for building the content was to always ensure that users should be able to absorb the necessary information of the topics through a more applications based or practical explanation. The content was then proof-read several times after being compiled, to ensure that the information provided was correct.

To manage the time efficiently, this task was ran concurrently with other development tasks such as learning how to use the new software tools and designing the website layout theme.

4.2 Website layout template development

Running parallel with the first task was the development of the website layout. The layout was the standard template where all the information would be presented on. Hence, this template acts as a base for the project content and therefore it was needed to be built first before adding the composed content. With the standards set in the chapter 2 and some ideas in mind, the designing of the website theme layout started off immediately and the sections outlined below illustrate how exactly the website templates were designed and implemented.

4.2.1 Graphical Design

This section will explain the physical graphical design of the layout using Adobe Photoshop 6.0. After the template was built and finalized, other parts of the design including the navigation buttons and links, the font type and most importantly, the graphics that was used for illustration purposes were created. Adobe Photoshop allowed the design to be put on screen, the physical graphics of individual sections like the main layout, and the navigation buttons to be created from scratch for the whole project.

For the whole project, there were a total of three templates created as shown in Figure 4.1on Page 35, Figure 4.2on Page 36 and Figure 4.3on Page 37 respectively.

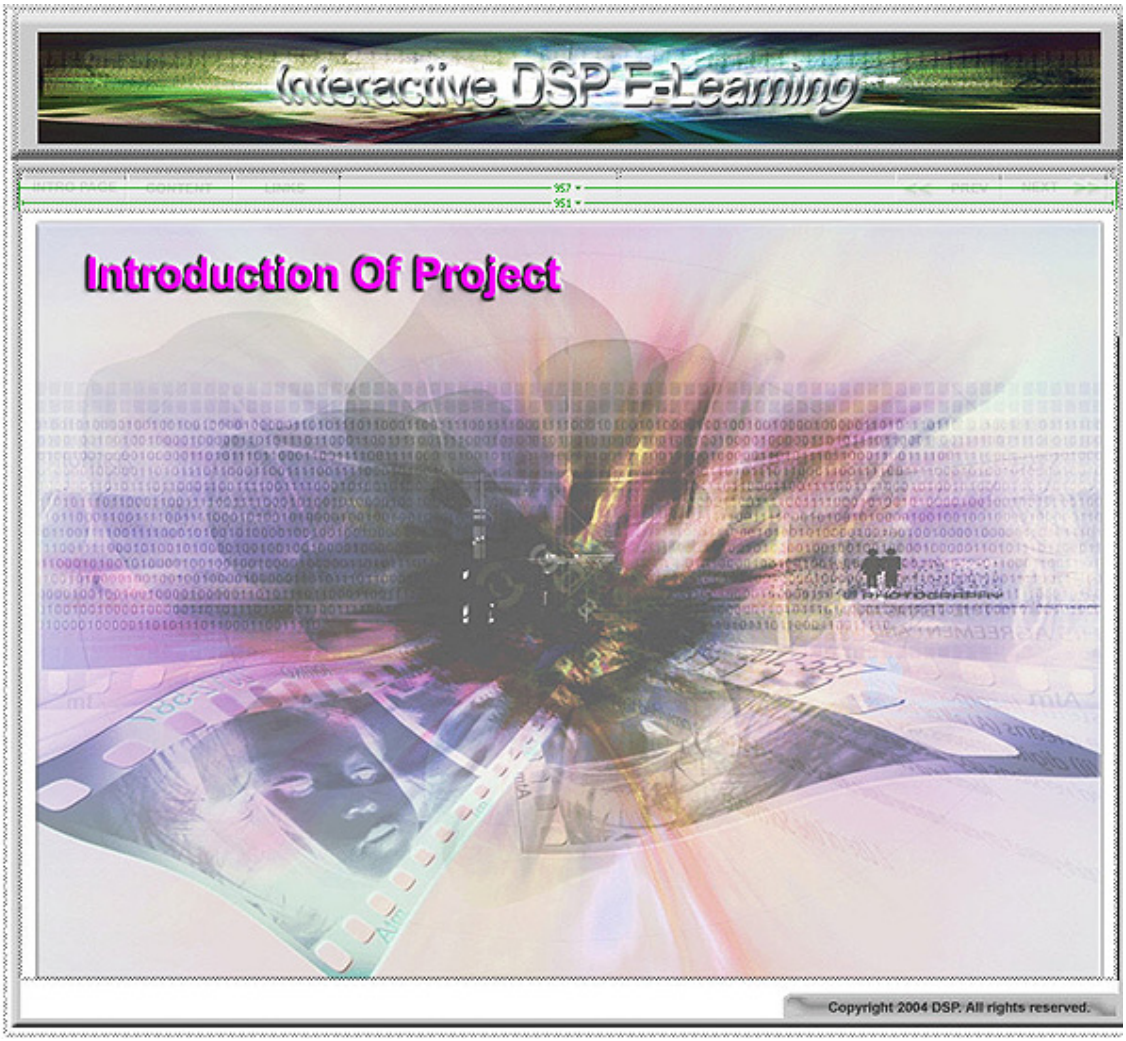

Figure 4.1: Template 1, introductory page of project.

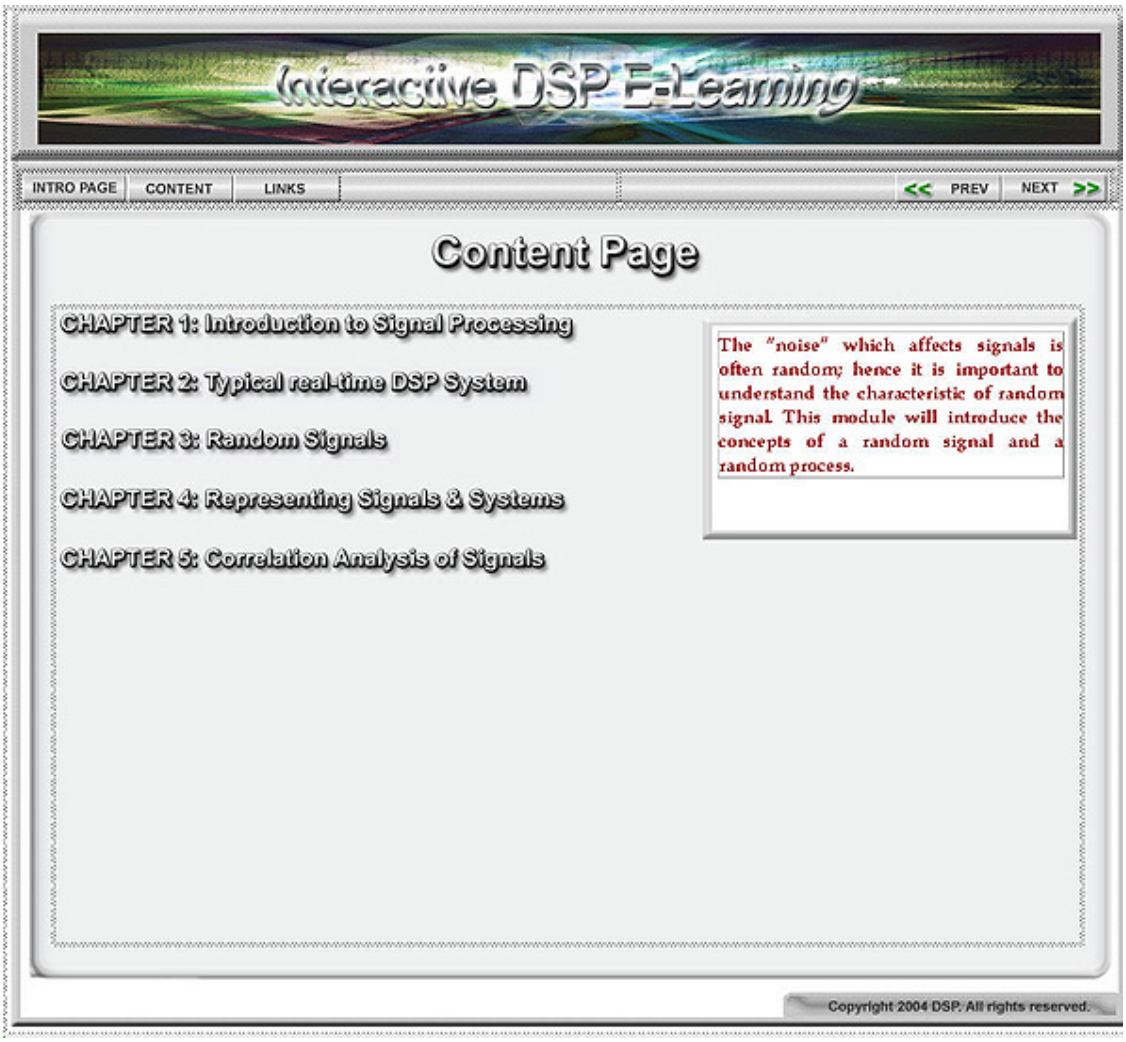

Figure 4.2: Template 2, contents page of the project.

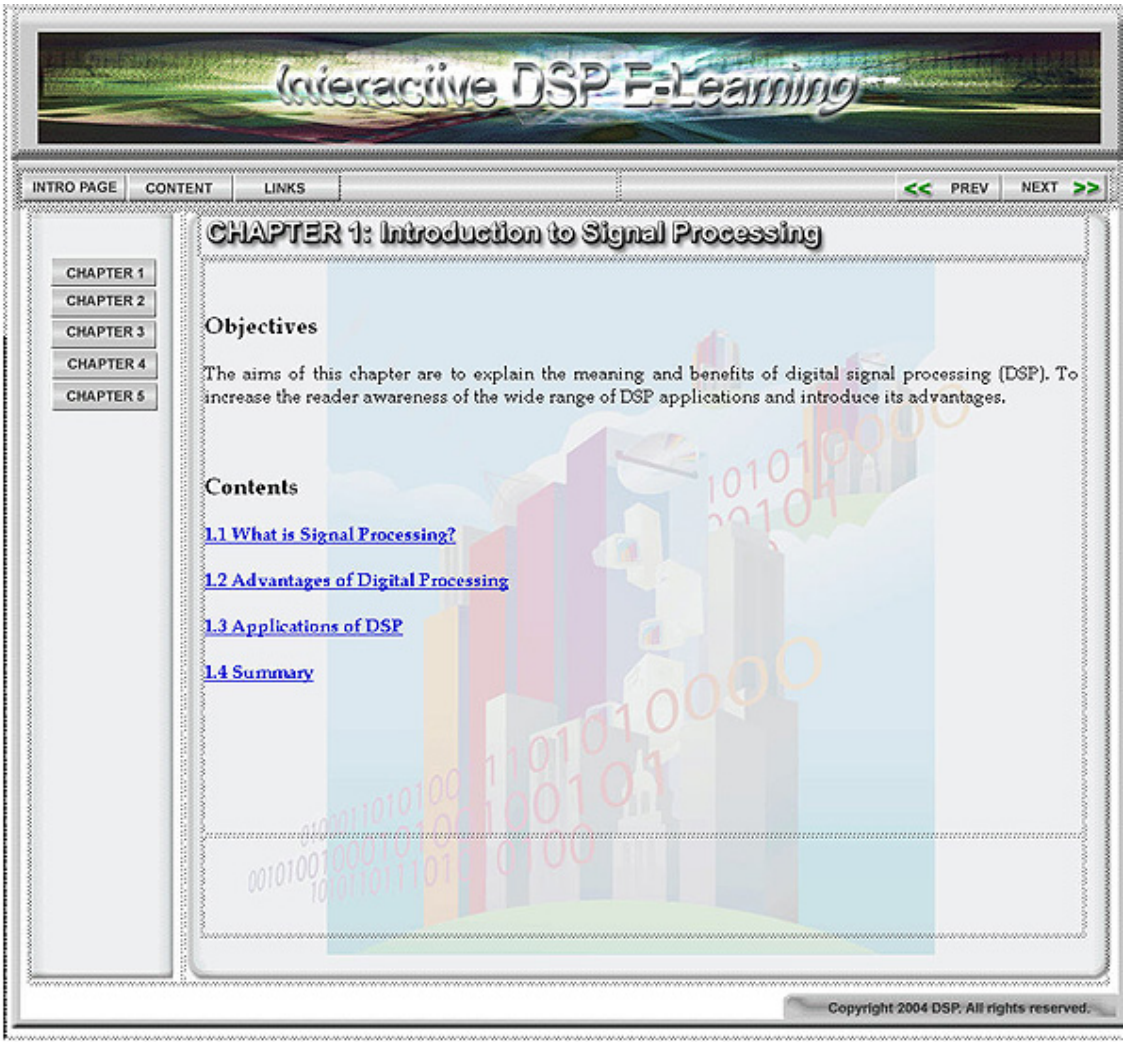

Figure 4.3: Template 3, standard template for main page of every chapter.

The steps involved in developing the templates were almost similar except the graphics used were different. Hence, the following screenshots show how one of the templates was created.

Step 1: Using Adobe Photoshop 6.0, create the standard template size (Screen size), 1024 by 768. The base layer was just a piece of white sheet.

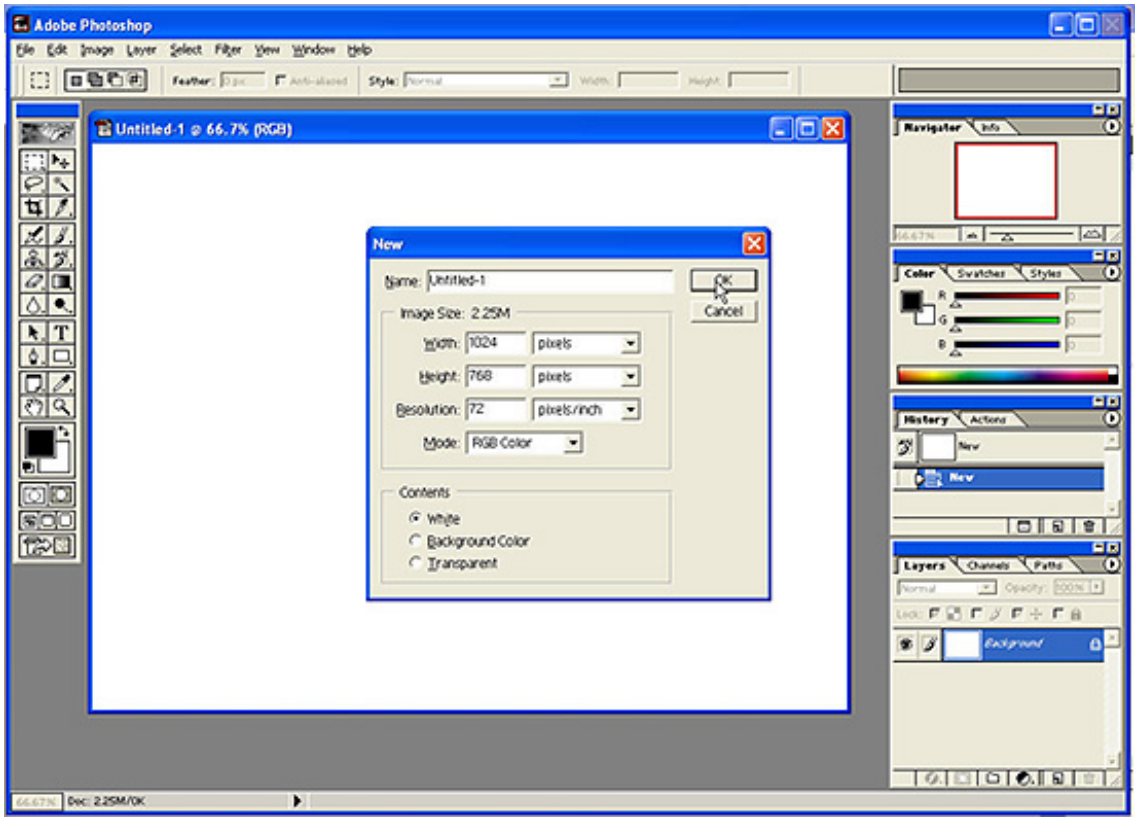

Figure 4.4: Creating template size.

Step 2: The website theme was built on layer by layer on top of the base layer. For instance, the top rectangular frame was one layer. Below the top frame was a bigger square frame, which was another layer and etc. Layers were used because if there was any changes to any of the graphics could be easily done without affecting any other graphics. Each different layers created was illustrated on the bottom right of the snapshot.

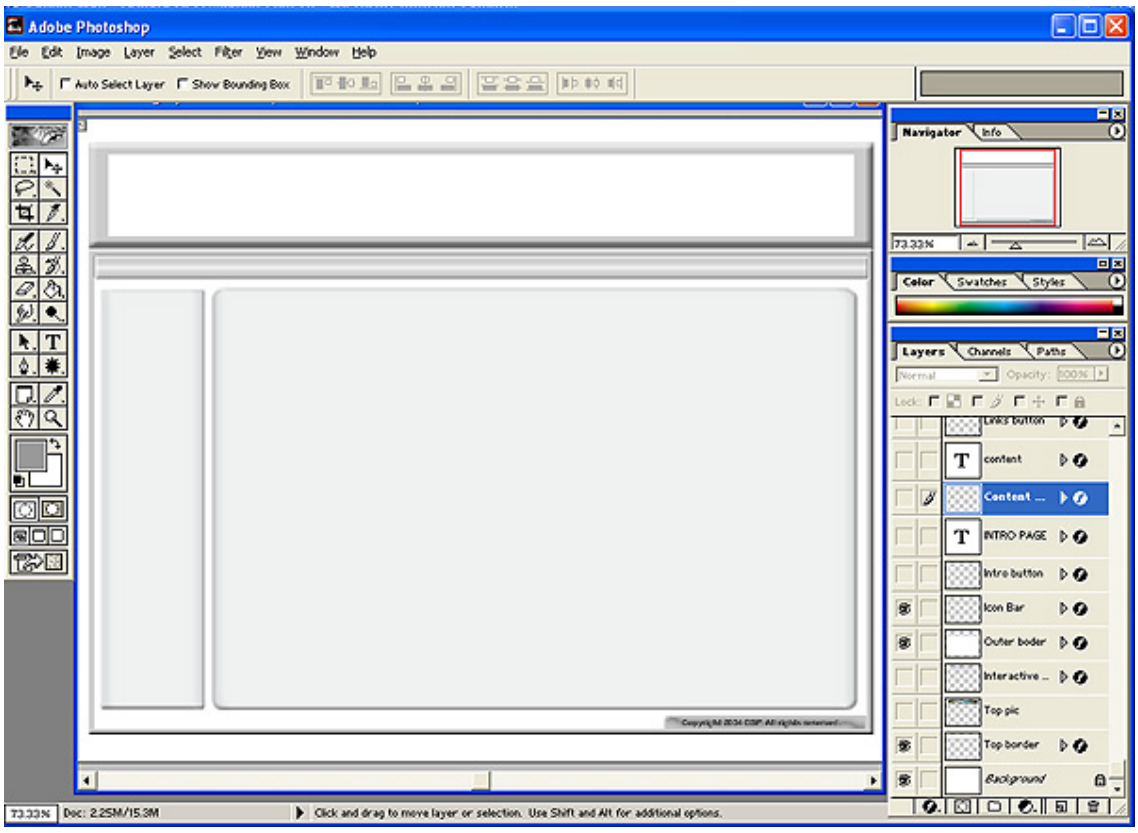

Figure 4.5: Standard template created with layers.

Step 3: The next part involved designing the other graphics components such as navigation buttons, font using, top banner and the project title separately. Once again, each of these graphics was built on individual layers with different layer styles to achieve the appropriate graphical effect. The snapshots below show the different layer styles for the top banner and one of the buttons were selected. The top banner style was meant to provide a 'futuristic' effect while the style for the button was to provide a 3D button effect.

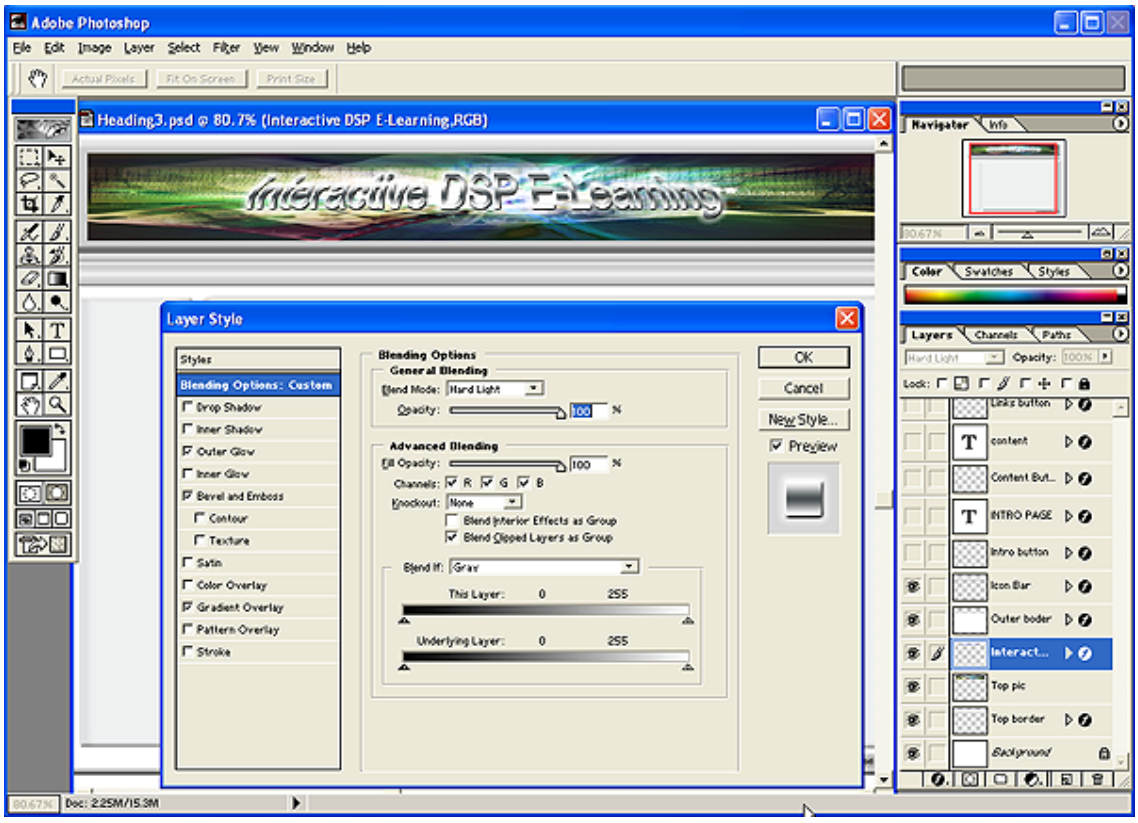

Figure 4.6: Adding on other graphics using different layer styles.

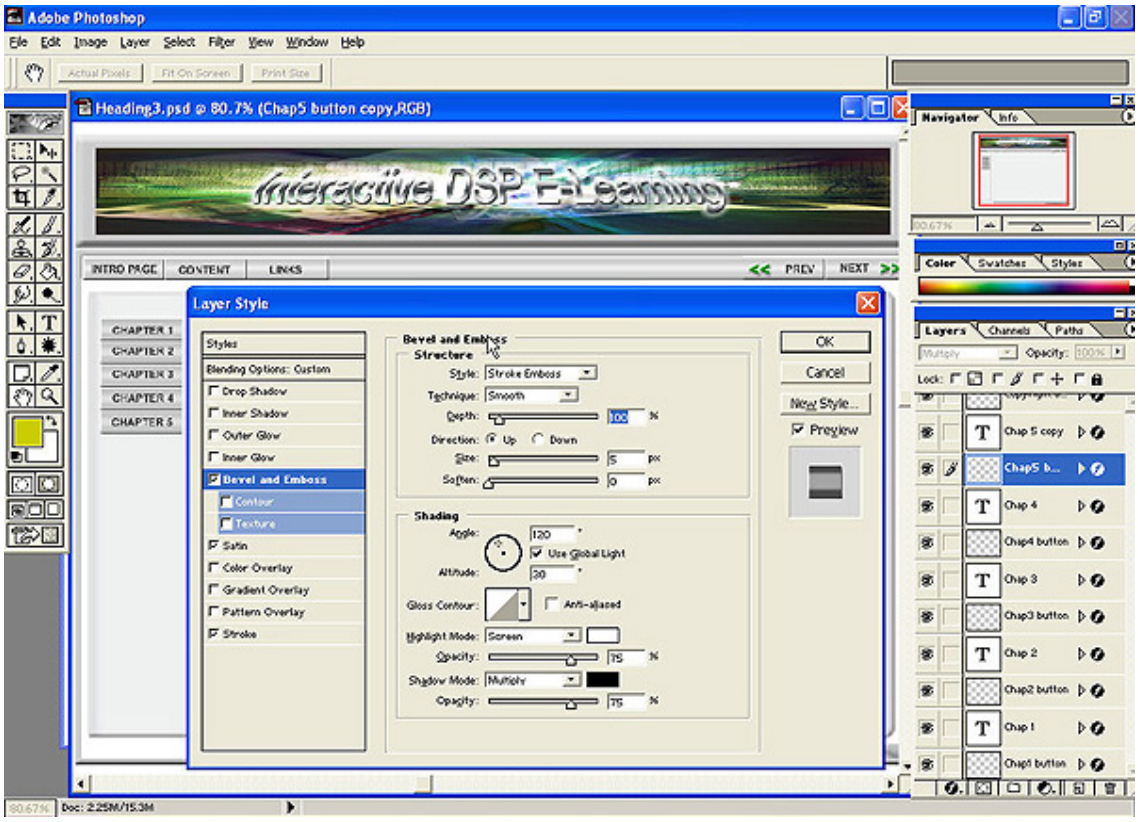

Figure 4.7: Illustrating the button layer style.

Step 4: After several reviews, the finalized template was decided upon as shown in Figure 4.9 on Page 43. This snapshot included the top banner and the navigation buttons. Take note that the navigation buttons shown in Figure 4.7 on Page 41 were actually the normal set; there were two other sets, which were 'over' set (mouse over look) and 'down' (the look when a button is pressed) set. The making of the two other sets were similar except for the changing of the layer styles parameters and buttons colour.

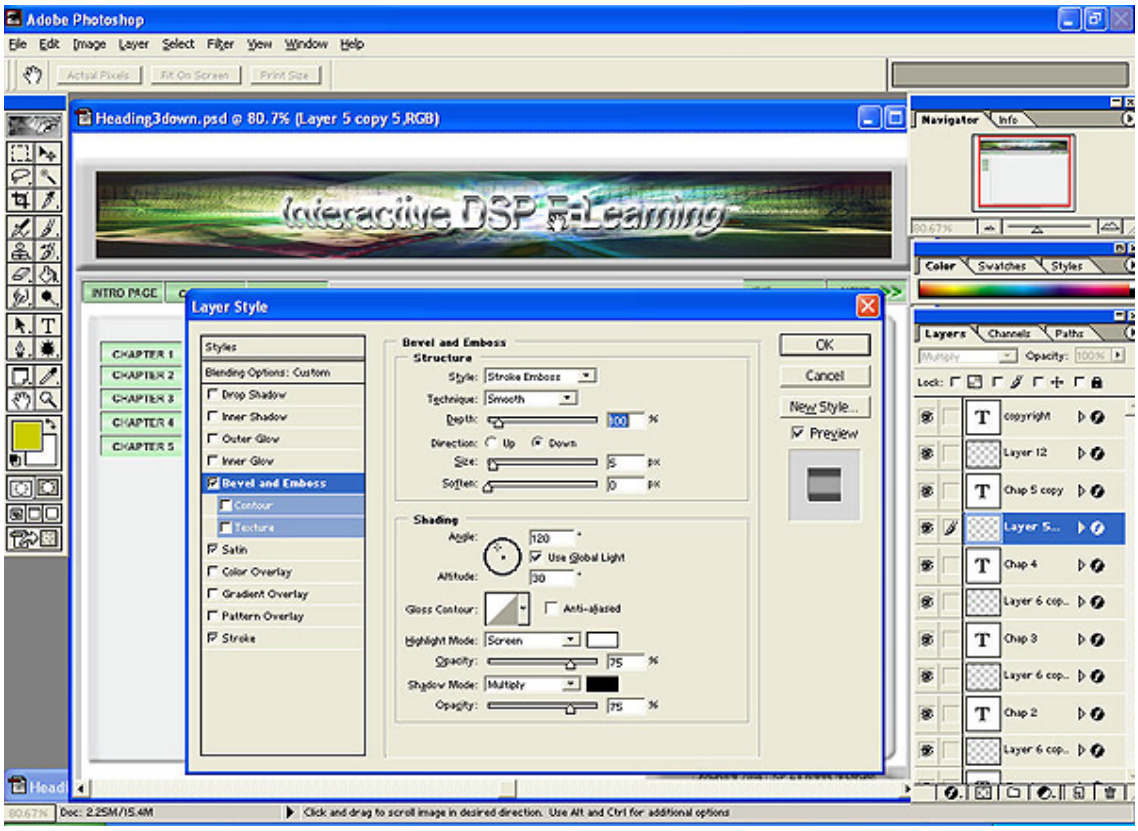

Figure 4.8: Illustrating the 'down' button layer style.

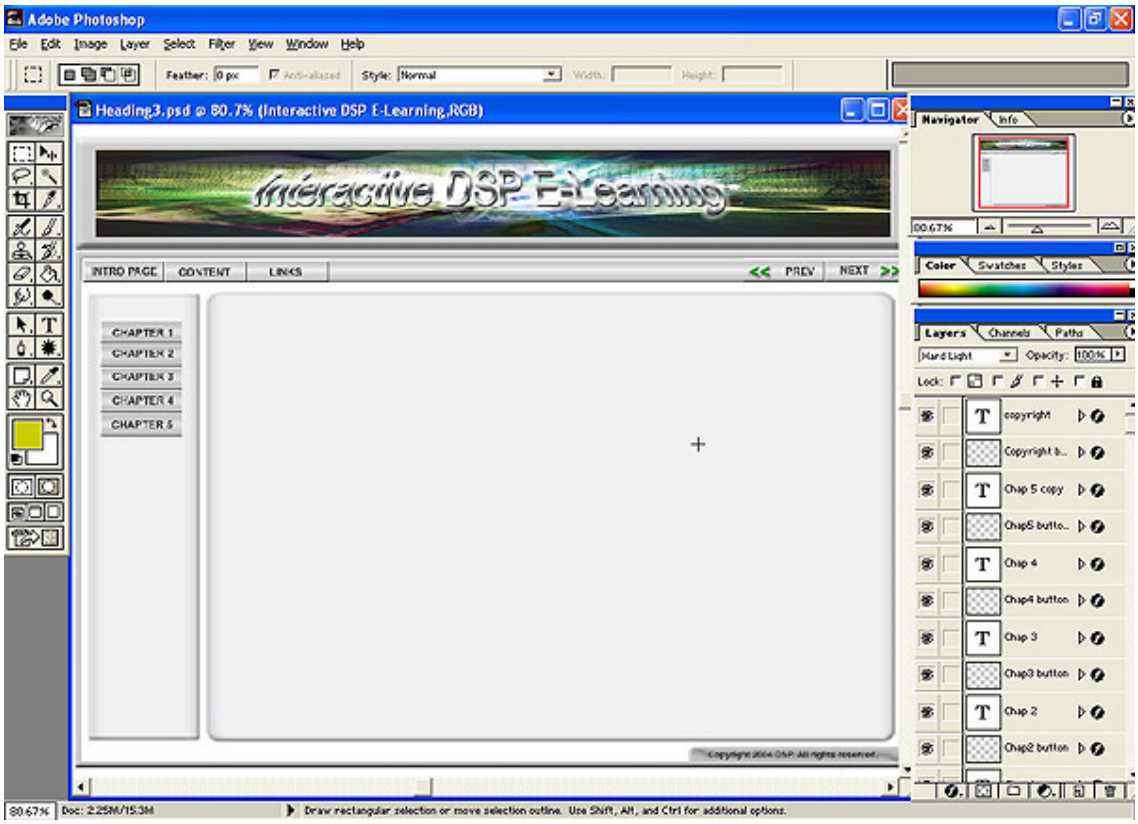

Figure 4.9: Finalized website template.

4.2.2 Screens Setup

The next step was to create a standard template structure to fit the graphics and design layout that was completed in the previous section and the written content on it. As those screenshots shown above were only graphics, they needed to be converted into usable images or structures. Hence, Macromedia Dreamweaver MX 2004 was employed to create this structure and the results are exactly the design of the three standard templates which were illustrated above in Figure 4.1 on Page 35, Figure 4.2 on Page 36 and Figure 4.3 on Page 37 respectively.

In order to present a better picture how these templates were created, the following illustrations show how one of the layout was set up. Note that only part of the website was viewable in the screenshots.

Step 1: First a new html document was created and a table was then inserted and dragged to the specified size of the main website layout. The basic layout graphic was next imported into it; the dotted light grey line along the graphic denotes the template table that was created. The layout graphic seems to have all the buttons missing; the reason being that the graphic acts as a base structure. This base structure will then be introduced in the usable images such as buttons to be added on.

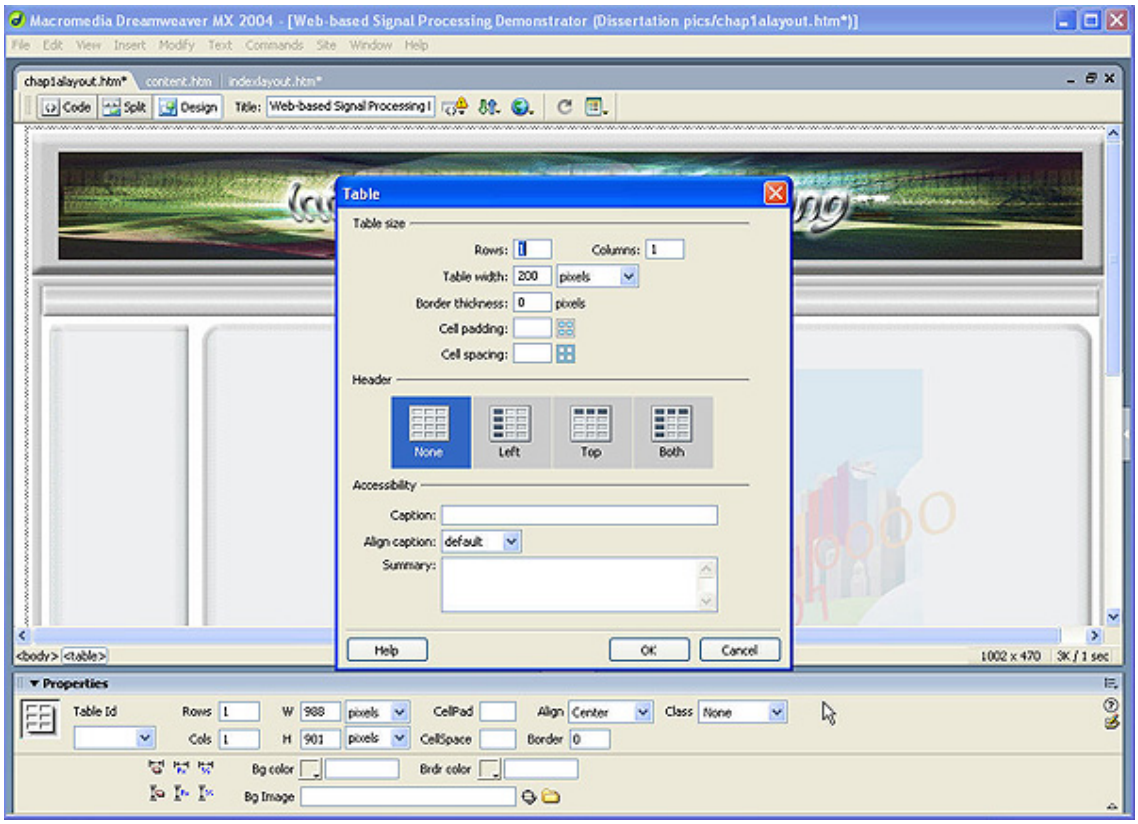

Figure 4.10: Setting a template size for building the base structure.

Step 2: A single table only allows one item to be inserted. Hence in order to insert additional graphics into the template, more tables were needed. While creating the template for the navigation buttons, a table was created for the title banner due to the fact that the navigation bar was in between the project title and the contents. This was followed by a separate table for the navigation bar and another one for the contents of the screen.

There was a need to create a few more tables to fit the navigation buttons into position. There were a total of five navigation buttons; three on the left and two on the right, aligned side by side. Two tables were used inside the navigation bar template to separate the left and right buttons as shown in Figure 4.11 on Page 46.

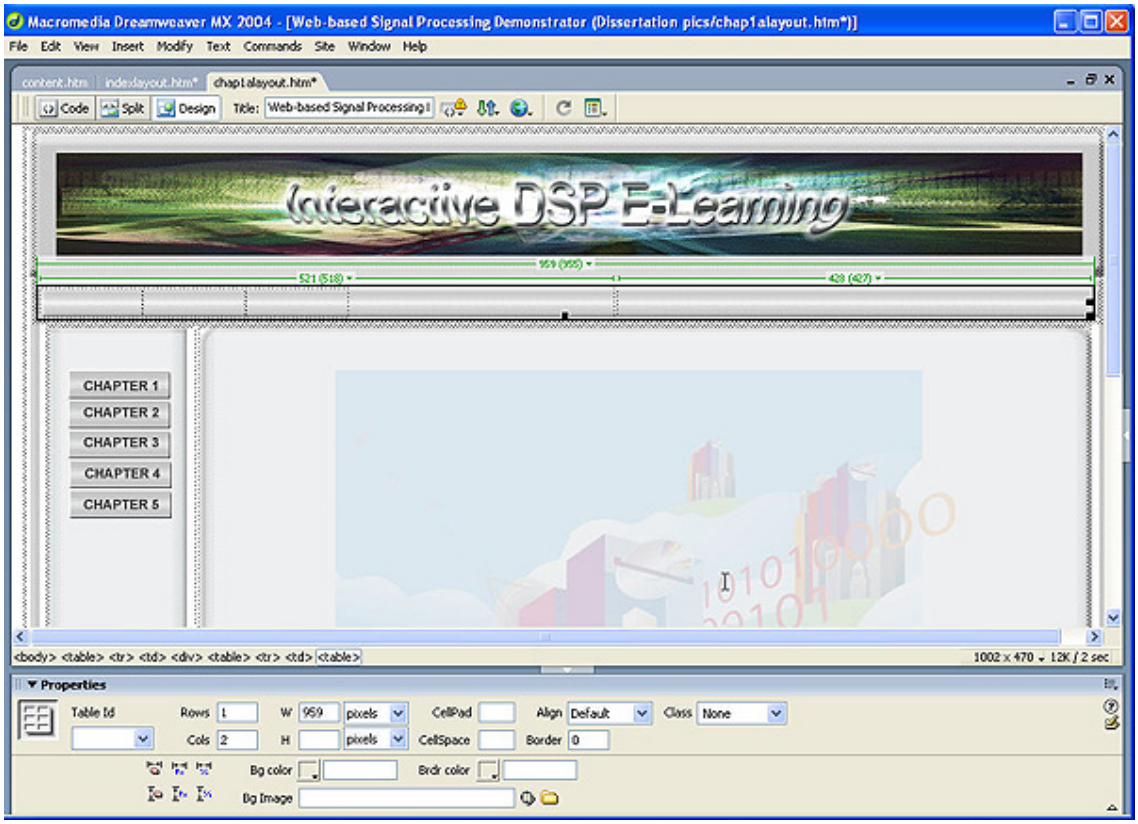

Figure 4.11: Creating tables for adding on of navigation buttons.

Step 3: The next part of the template to create was the side panel that was meant for the chapter buttons. These buttons allowed the users to go to a designated chapter. The table which houses the content (created in Step 2) is now divided into two smaller tables and arranged into their required size and position.

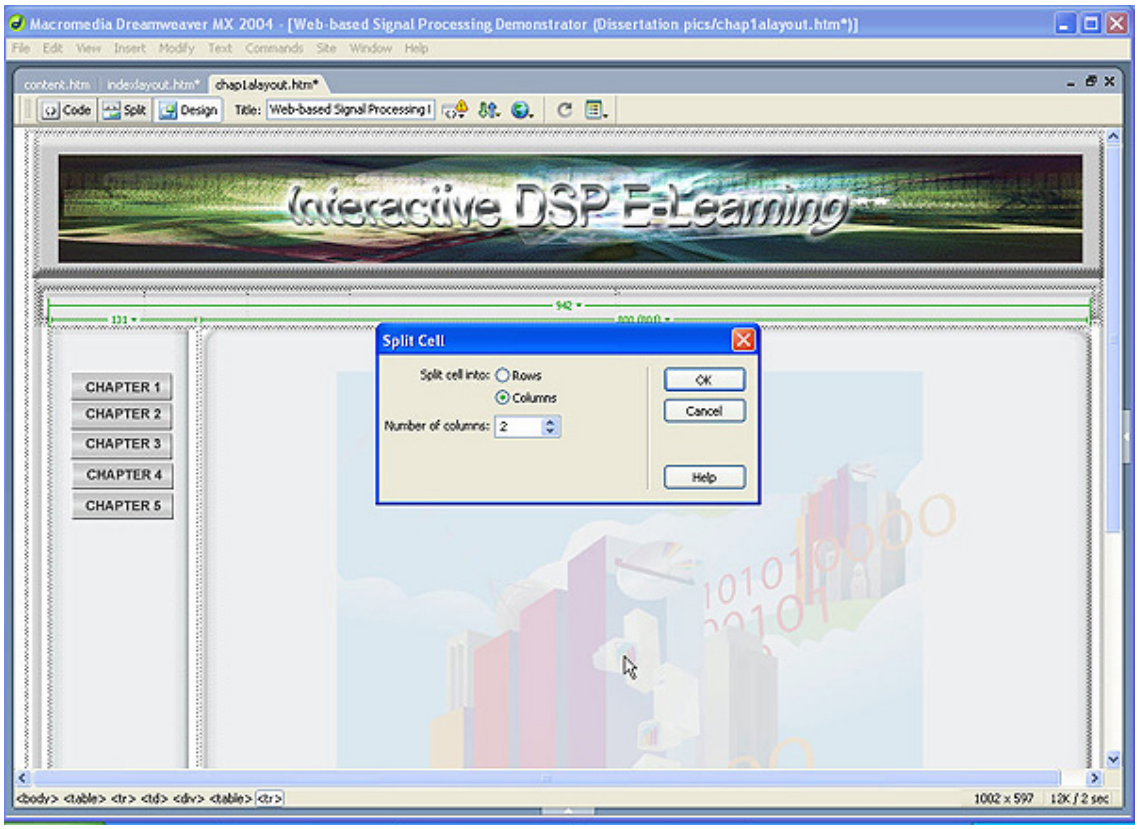

Figure 4.12: Creating tables for adding on of navigation buttons at the side panels.

Step 4: Some slight adjustments to the layout was done here to help align the graphics with the intended layout. Two smaller tables were added in the content table; the small upper table was meant for the chapter title while the larger bottom table was to hold the written content of the page. As for the content of the topic, it can be imported or just copy and paste from the Microsoft Word into the table meant for content. Do the necessary adjustments on the alignment and it was ready. The screenshot below shows the completed layout template of one of the three layout types.

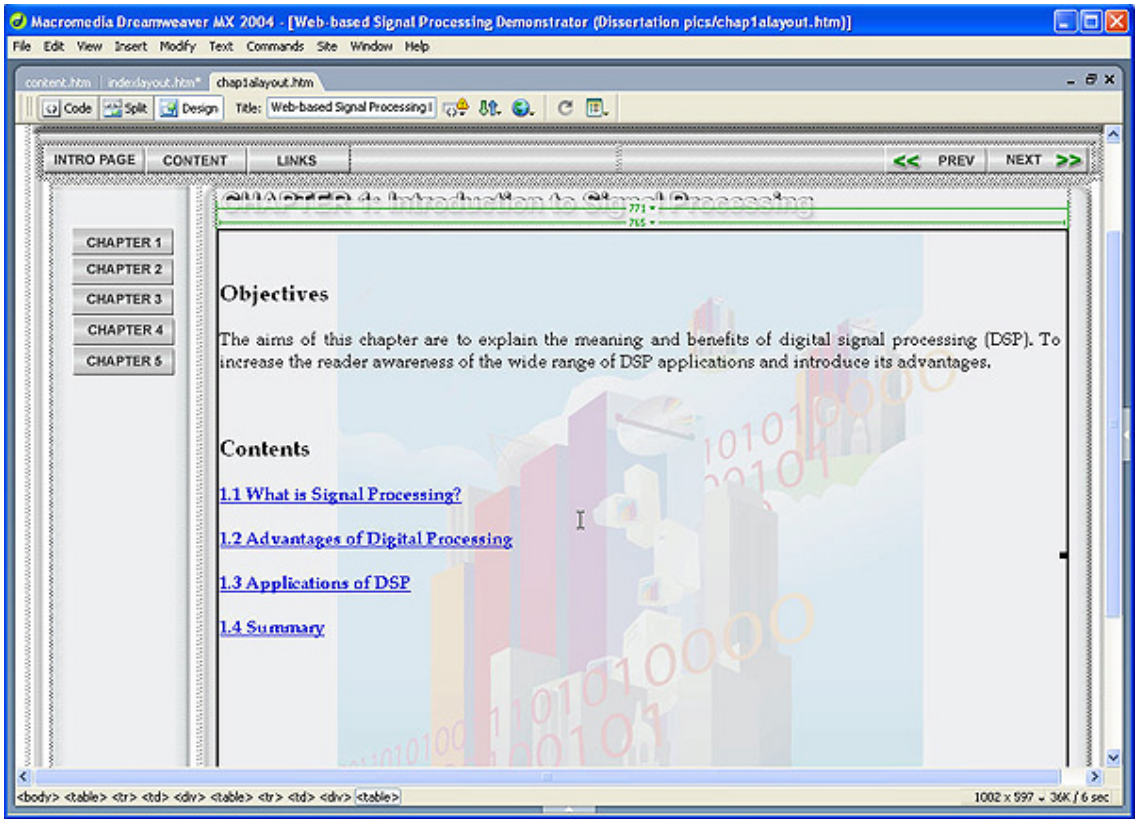

Figure 4.13: Adding on the Chapter title and the topic content.

The above steps looked simple and easy, however some time and effort was taken as it was an unfamiliar tool to work on. Also, the creating of tables and alignment take up some time to adjust to correct position. These steps were meant for putting in images onto the template, to make these images usable such as the clickable buttons, clickable links were introduce in the following sections.

4.3 User Interactive Functions

The term 'Interactive' for computer programs means to accept input from the user while they are running it. For example, a game that waits for the user to take an action, and then responds to that action. The interaction between computer and user may take place through typed commands, mouse clicks, or other types of interfaces. For this project, interaction was mainly through mouse clicks. The interactive functions developed for this website includes navigation buttons, the interactive learning exercise and mini-quizzes. Developing the interactive functions for this project was the one of the main requirements as it helps to improve the website attractiveness and also improve users' interest and understanding on DSP.

This section is divided up into several subsections describing how the different functions were constructed using both Macromedia Dreamweaver and Macromedia Flash MX (ActionScript).

4.3.1 Navigation Buttons and Links

The main purpose of the navigation buttons is to assist the user to navigate around the website with ease. The main buttons which are on the top bar consists of introduction, content, links, the previous and next buttons. Within chapters itself, there are side panel buttons that link directly to the individual chapters as shown in Figure 4.14 on Page 50. With the used of Macromedia Dreamweaver, this can be done.

Figure 4.14: Main navigation buttons and the side chapters buttons.

Referring to Figure 4.15, the 'intro' button which links to the main page of the website was created by selecting the first table space and by selecting the option 'insert image and navigation bar'. The element is given a name called intro', and the three images for the button are also selected. After that, all that is left was to activate the button link, i.e. to allow the 'intro' button to bring the user to the correct site when clicked.

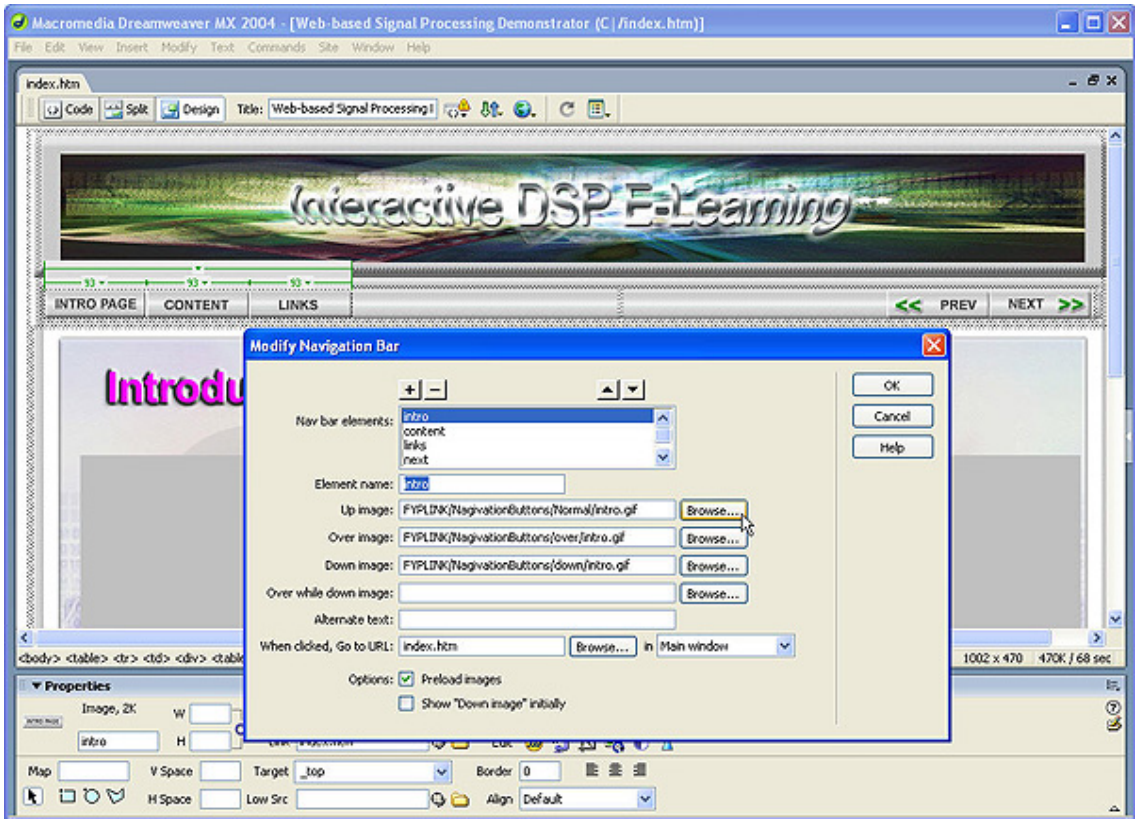

Figure 4.15: Activating the navigation button interaction functions.

The remaining four buttons are created in a similar fashion. Figure 4.16 gives a clearer view on how this was done. This was how the completed screen was linked up together to form one chapter in a continuous sequence by defining the 'prev' and 'next' button in the correct flow.

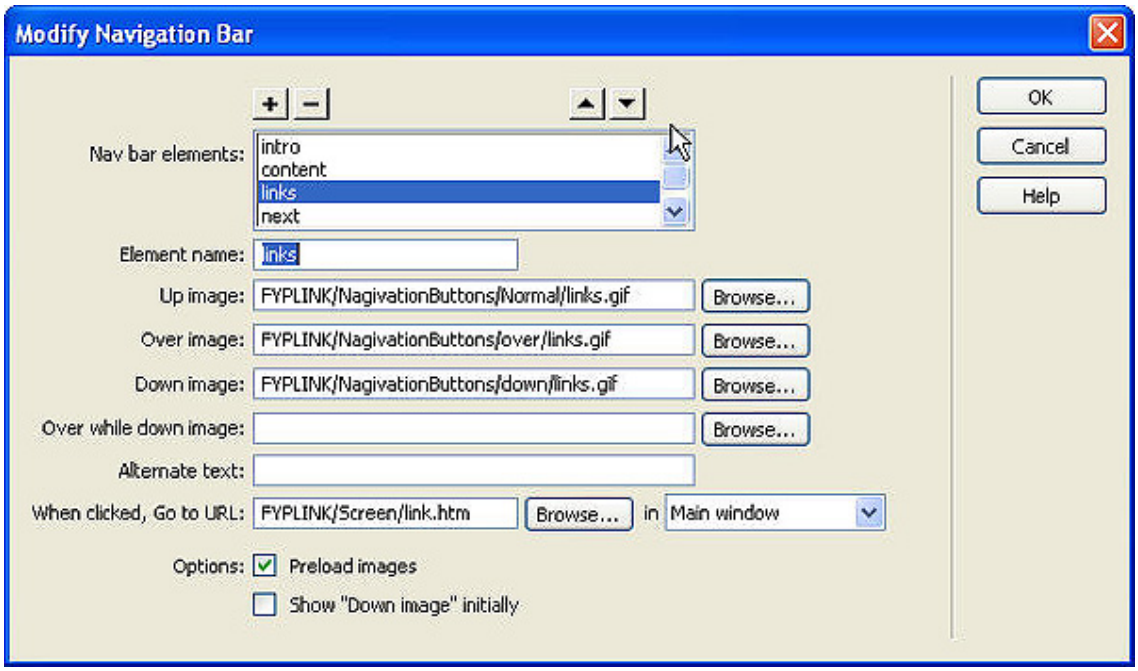

Figure 4.16: Navigation Bar screen for activating the buttons.

The side panel buttons were also similarly created except that they were placed in a different table space.

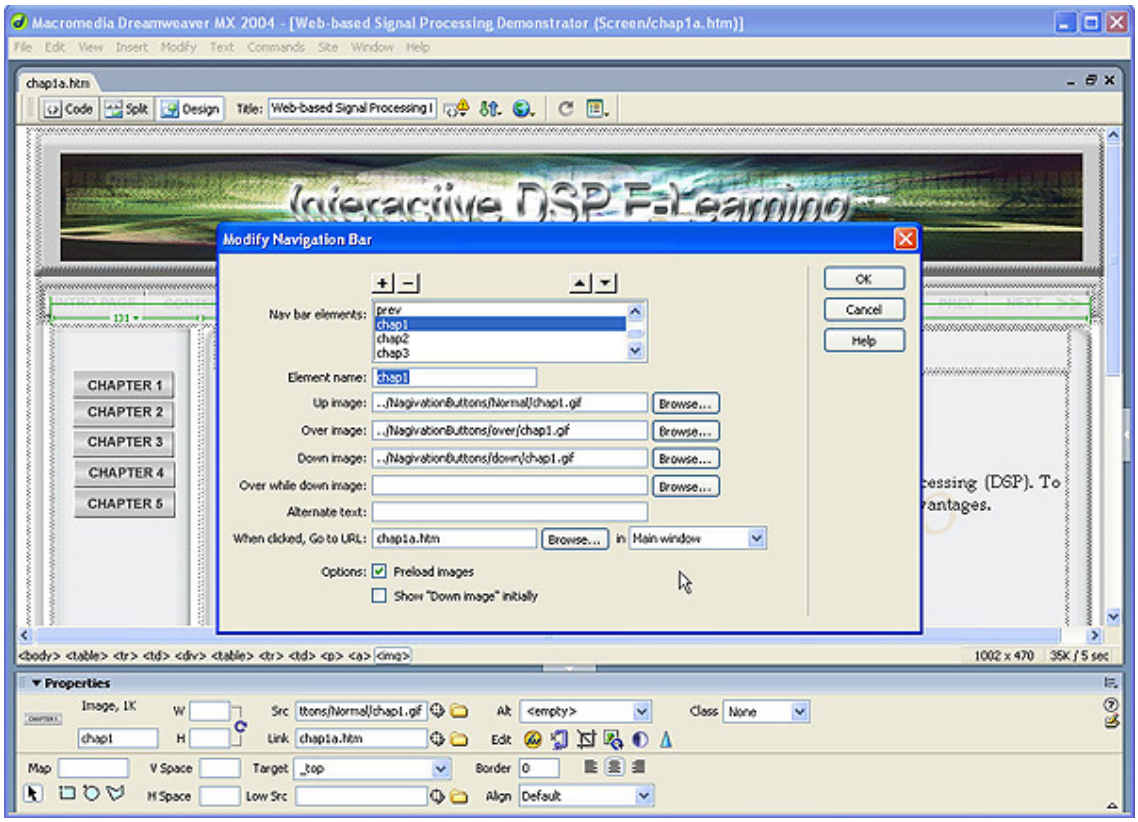

Figure 4.17: For adding the side panel buttons activation.

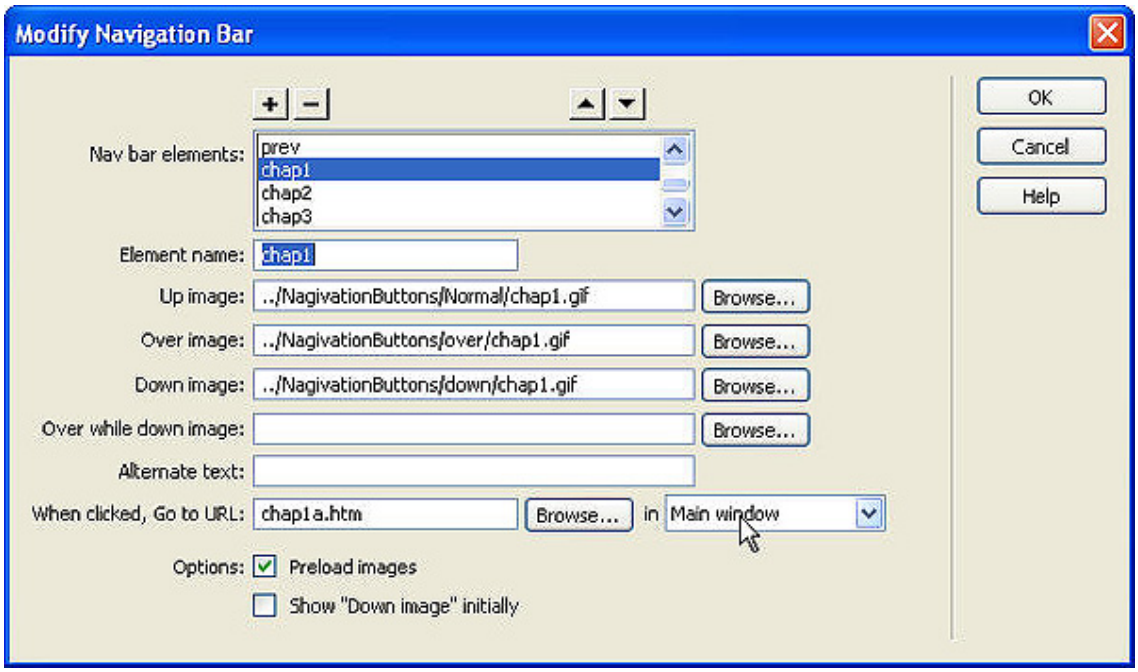

Figure 4.18: Adding on to the navigation bar.

Beside navigation buttons, navigation links were introduced in this project too. These links were only used in the main page of every chapter. The figure below show an example of 'navigation links', i.e. the blue and underlined titles are the navigation links where users can click on them to bring them to the desired section.

Figure 4.19: Navigation Links.

Activating these links was simpler than the navigation buttons as it just required the highlighting of the section name and to select 'make link'. A screen prompt would appear as below and a selection on the page to link the section to has to be made.

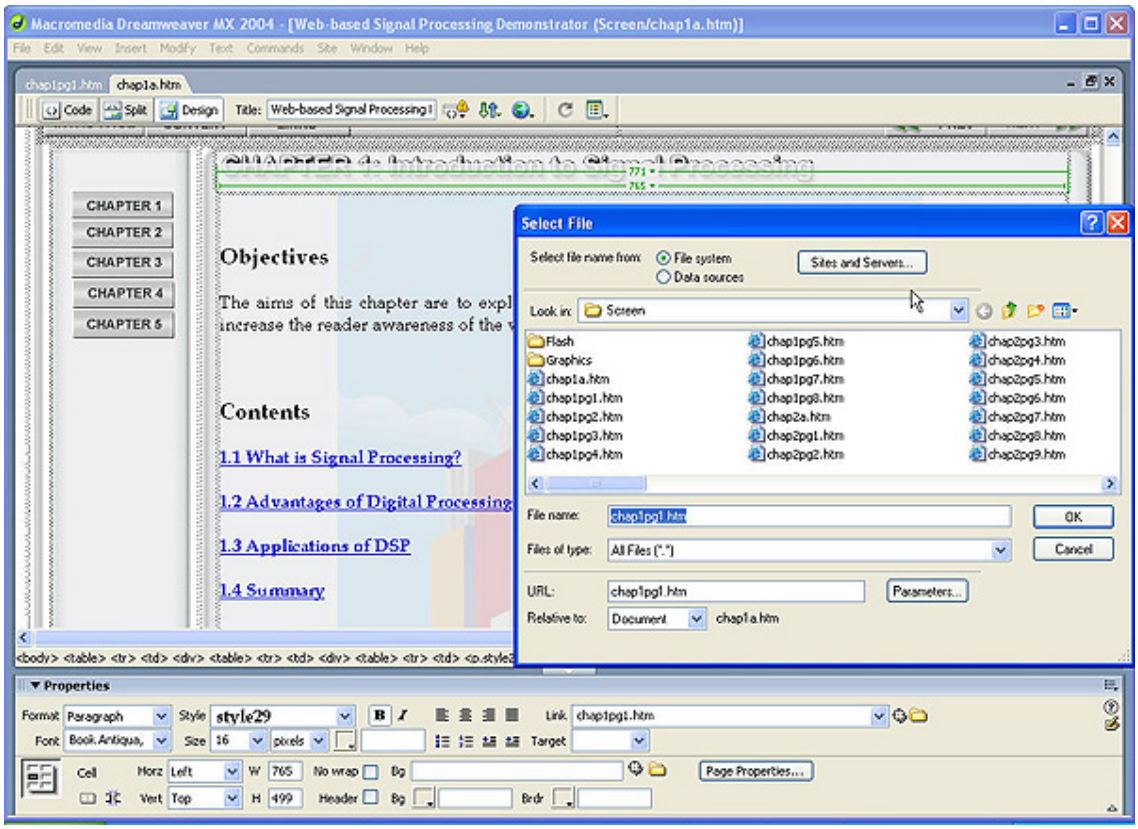

Figure 4.20: Activating the Navigation Links

4.3.2 Animated graphics

This project has incorporated a number of animated graphics that were mainly used for illustrating examples. These animated graphics were developed using the ActionScript language which is actually a programming language built in Macromedia Flash MX. As the language and program was unfamiliar, a huge bulk of time was spent on learning how to use and implement it. A brief explanation on how these animated graphics were done was given as follows.

This sine wave was one of the animated graphic used in Chapter 2 for illustrating as a continuous analogue signal.

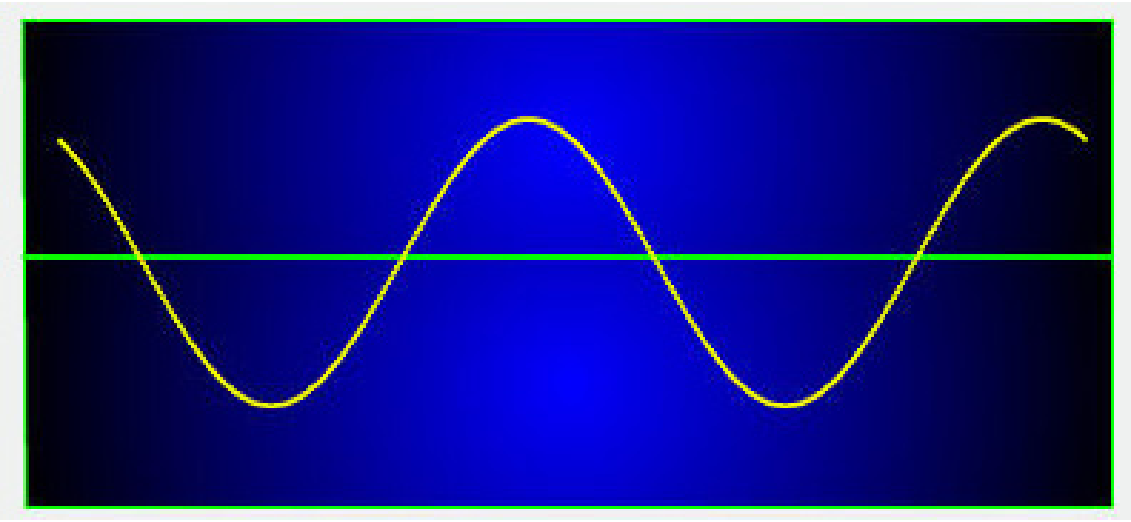

Figure 4.21: Animated sine wave.

Figure 4.22 show two layers in the timelines section. One layer represents the sine wave background while the second layer holds the ActionScript programming. The background layer was created by simply using the drawing tools at the side to draw out and add in the colour. The second layer was more complicated as some programming was involved. Refer back to Figure 4.21 on Page 57 and see that it shows a continuous sine wave. This wave is actually made with the use of a 'yellow dot' movie clip which is stored in the Flash library as shown in Figure 4.23 on Page 59 on the right column.

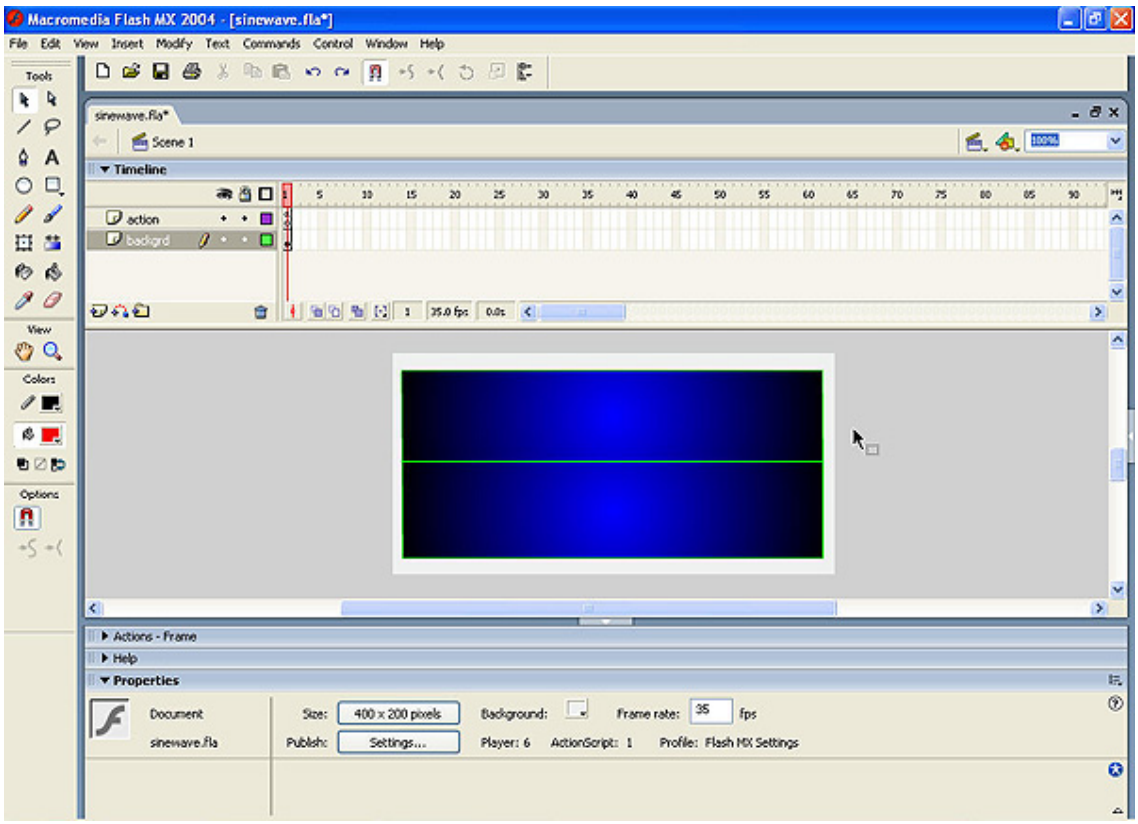

Figure 4.22: Animated sine wave created using Macromedia Flash.

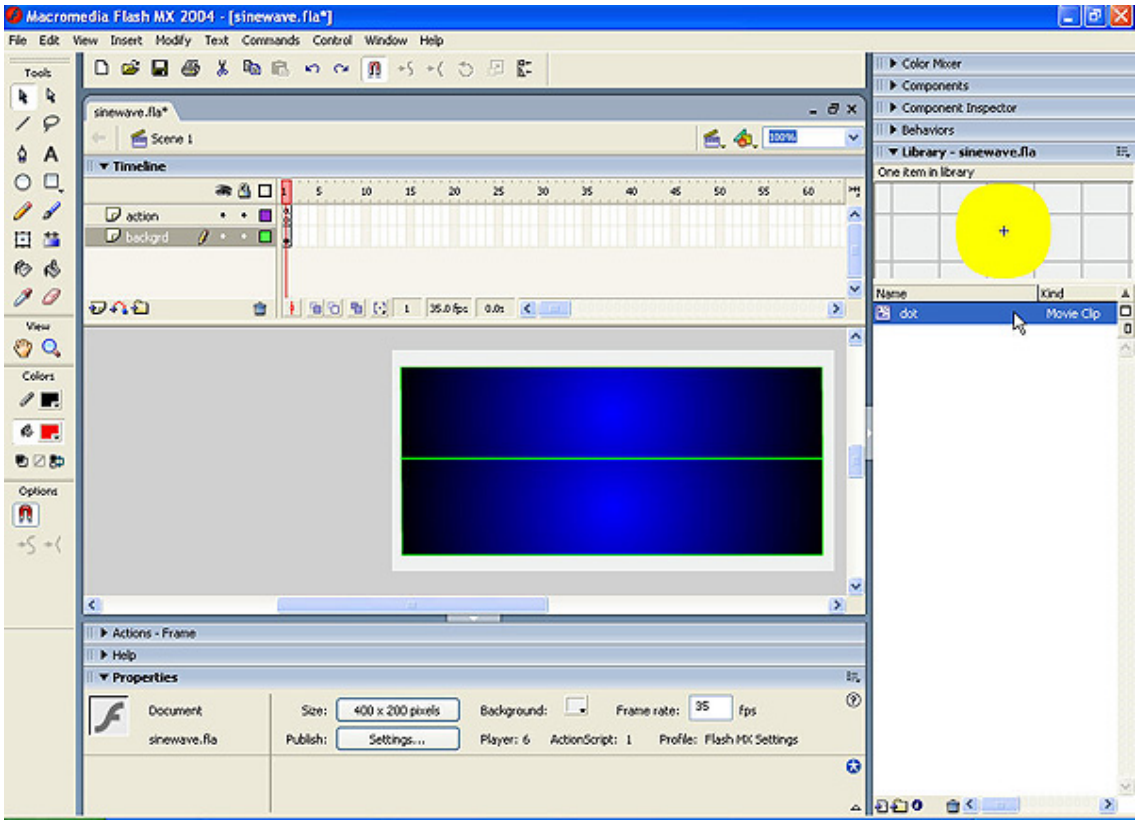

Figure 4.23: Yellow dot movie clip stored inside library.

The yellow dot is just another graphic drawn using the drawing tool. However, the yellow dot is a very small dot, the drawing scale needed to be increased. This dot was then converted into a symbol with movie clip behaviour. For animation, movie clip behaviour must be selected as shown in Figure 4.24.

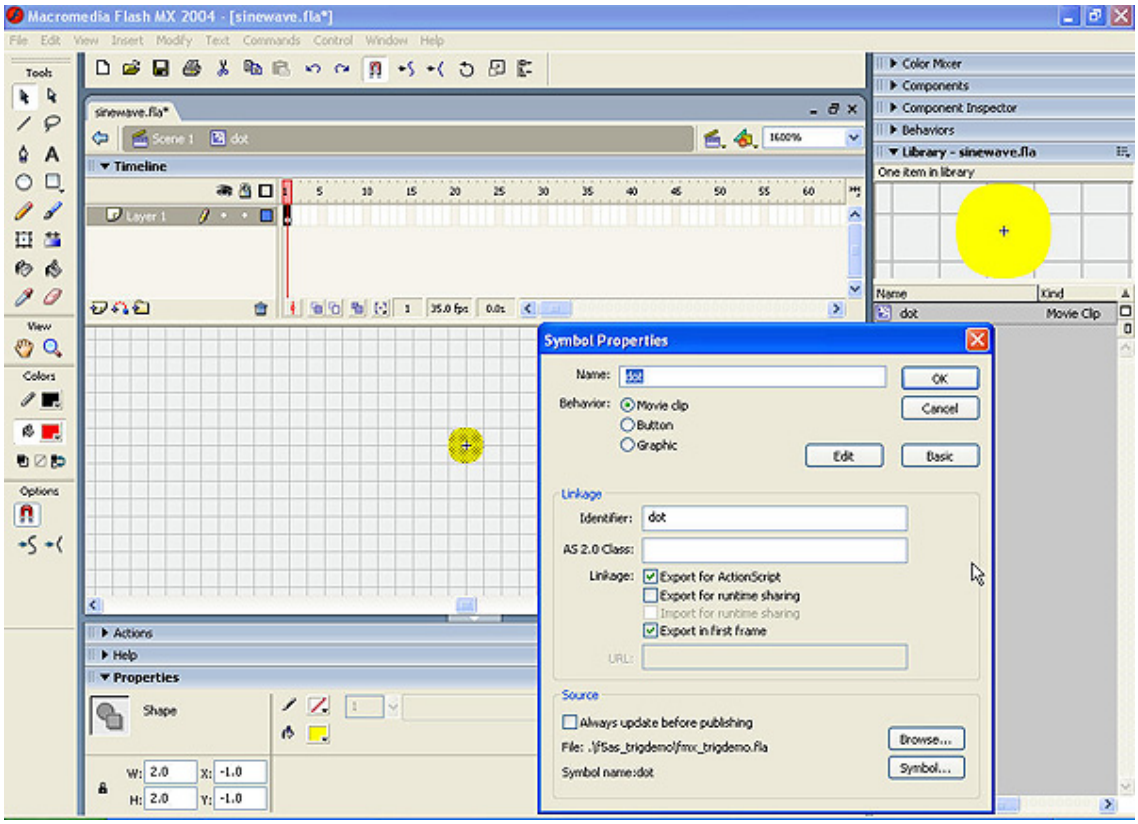

Figure 4.24: Convert the yellow dot to Movie clip.

With the background and the dot available, the next step was use ActionScript and mathematical functions to program it to move in a sinusoidal way. Figure 4.25 show that the programming was written inside the 'actions' workspace.

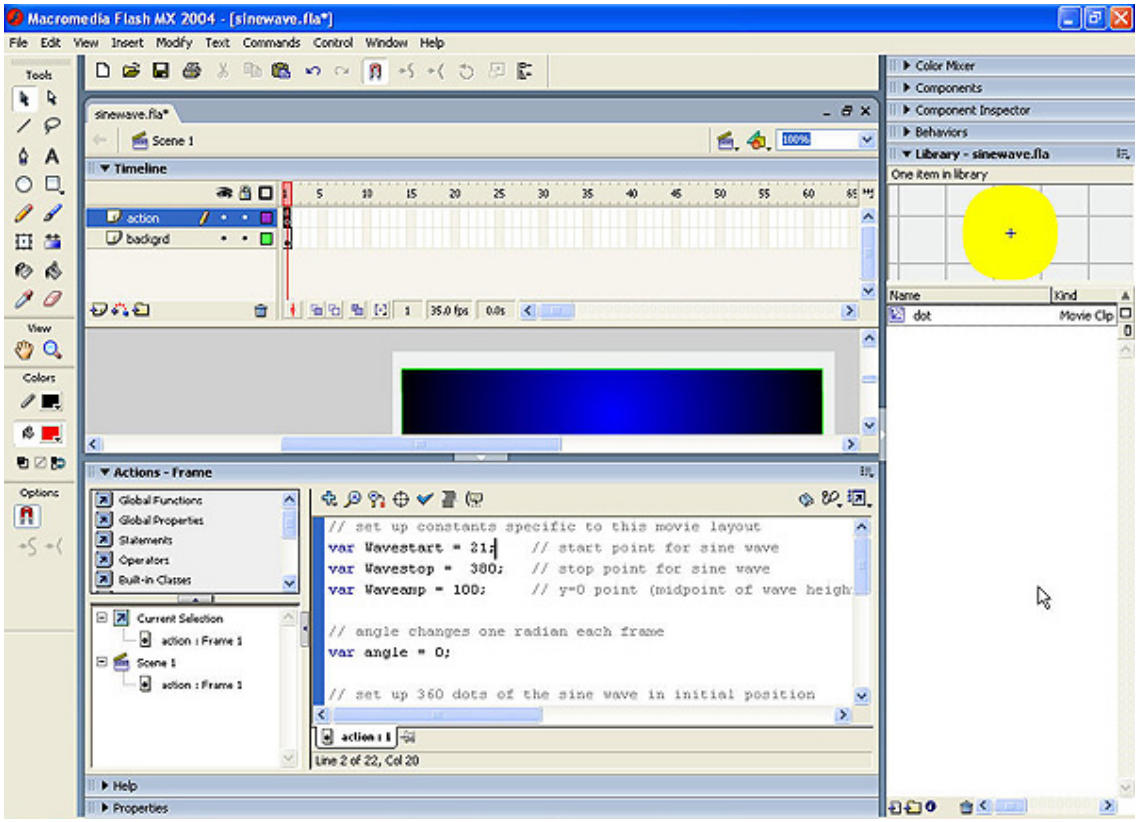

Figure 4.25: Using ActionScript programming to achieve the sine wave

The programming source code for generating the animated sine wave is shown below . Basically, the source code defines the wave starting and ending point as well as the amplitude of the wave. It then runs through a loop which sets up 360 points starting from the wave starting point. Inside this loop, mathematical functions were utilized to calculate the 360 points into forming a sine wave. In order to see this sine wave, the 'dot' movie clip was attached to these functions and the 'dot' would show at the position calculated out by these functions as a sine wave. Hence, the sine wave looks like it is moving.

Programming source code for generating animated sine wave.

```
// set up constants specific to this movie layout
var Wavestart = 21;    // start point for sine wave

var Wavestop=  380;   // stop point for sine wave

Waveamp = 100; //y=0 point (midpoint of wave height). Amplitude
of the waveform

// angle changes one radian each frame
var angle = 0;
// set up 360 dots of the sine wave in initial position


for (var a=0; a<360; a++) {

    angle -= Math.PI/180*2;
    this.attachMovie("dot", "dot"+a, a, {_x:Wavestart+a, _y:Waveamp
    - Math.sin(angle)*50});
    //my_mc.attachMovie(idName, newName, depth [,initObject])
    }
```

```
    this.onEnterFrame = function()
    {
        for (var a=0; a<360; a++)
        {
            this["dot"+a]._x = (this["dot"+a]._x >= Wavestop)
            ? Wavestart : this["dot"+a]._x+
        }
        angle += Math.PI/180*2;
};
```

The following figures illustrate the screenshots of some animated graphics. The source codes for these animations are in the References section.

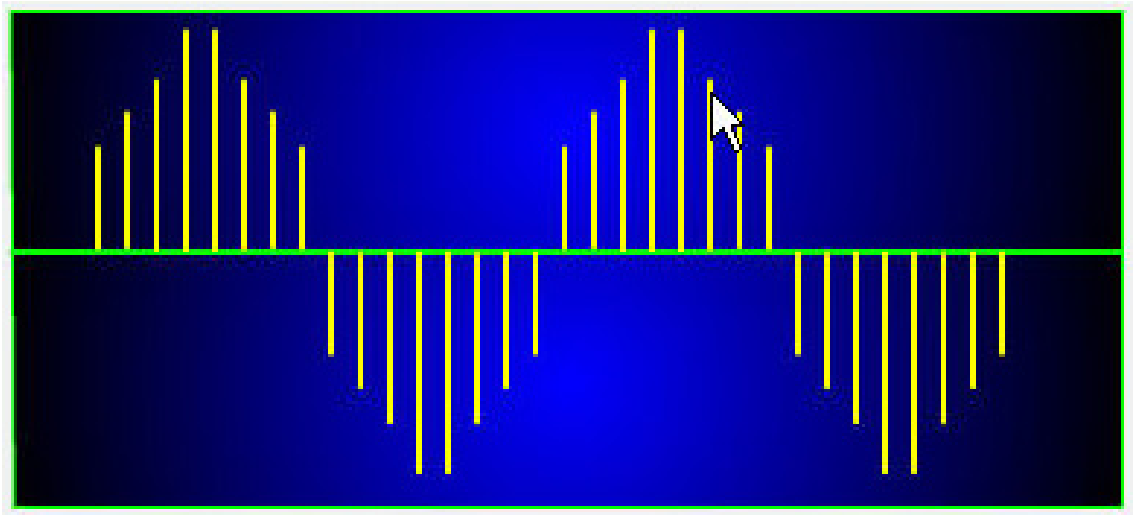

Figure 4.26: Animated Discrete signal.

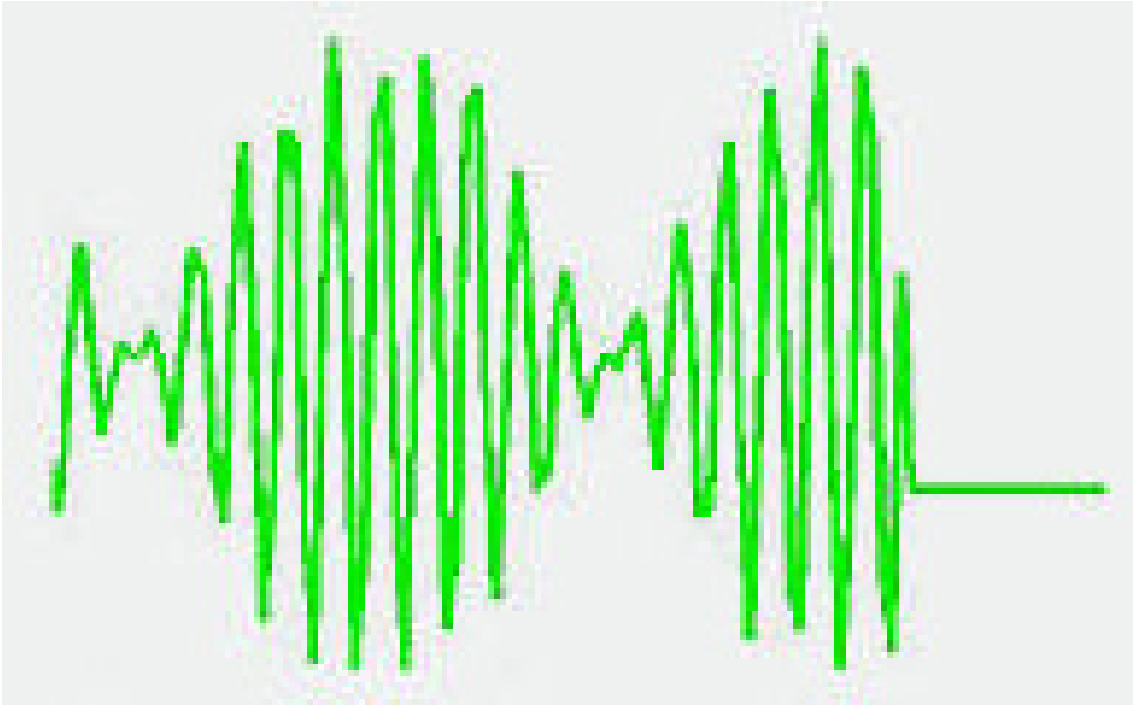

Figure 4.27: Animated signal.

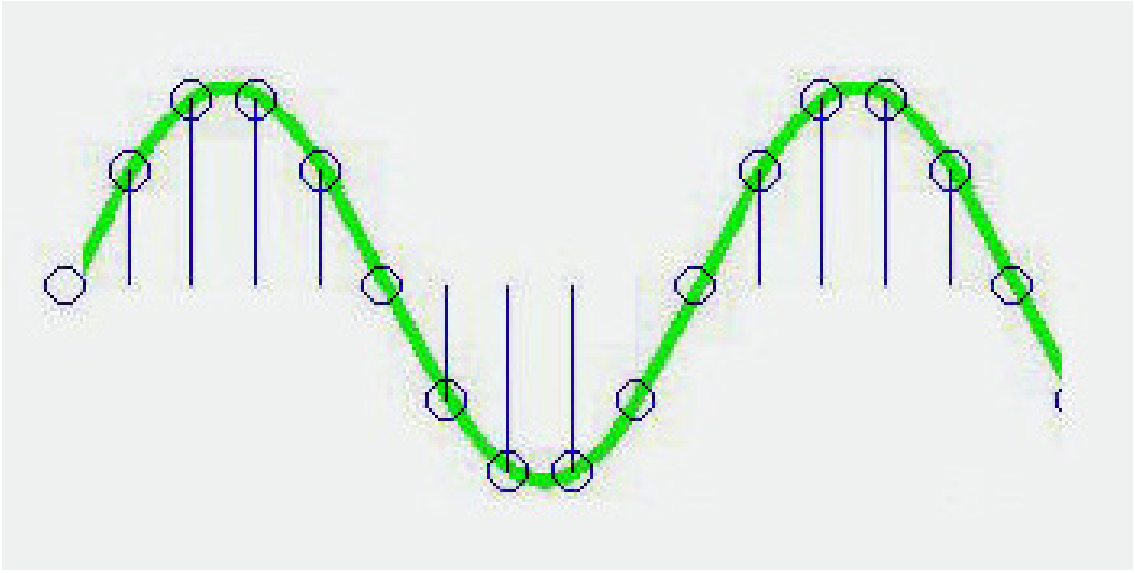

Figure 4.28: Animated sampled signal.

4.3.3 Interactive learning demonstrations

The next thing to develop was the interactive learning demonstrations. Due to time constraints and some difficulties involved, the developments on this section were not as much as intended - only two interactive demonstrations were accomplished. This was the most difficult task to implement due to limited knowledge of how the dragging functions work, especially on defining the correct position for each individual block. Hence a lot of time was spent on researching the functions as well as troubleshooting. Overviews on the processes of the two demonstrations are shown below:

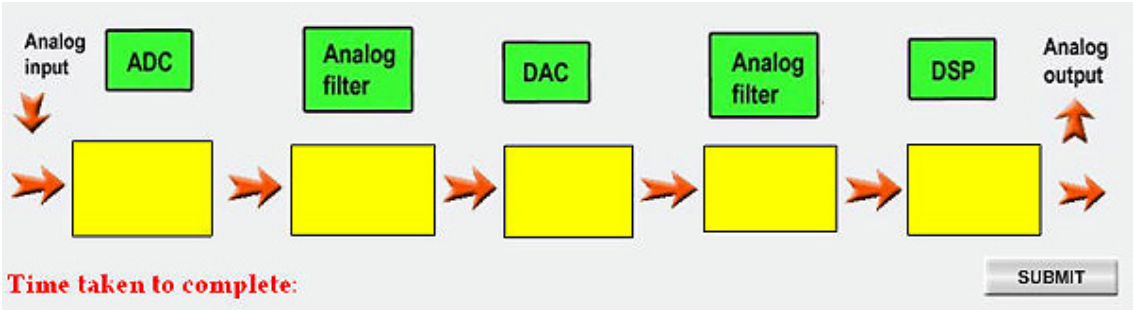

Figure 4.29: Interactive DSP system diagram learning.

Figure 4.29 show a typical real-time DSP system taught in an Interactive way. In this demonstration, users are supposed to drag the individual blocks to their intended position according to their knowledge. This game was designed in such a way that users can only drag and drop the correct blocks to their designated position, otherwise the block would simply bounce back to its original position. Once, they have filled the boxes and pressed the 'submit' button, the time taken for completing the game will be displayed.

The program was written separately for each block and the source code for two of the blocks are shown below. The programming for each block is more or less the same, except for their designated positions. Also the two Analog filter block codes have additional lines because they can be placed at each other's position. The explanation of the source codes is included in the codes itself.

Program code for ADC block

```
/* For ADC block.*/ /*On the event when the game is loading, the
original position (x, y) of ADC are stored into variables of
this._x and this._y, and var noDrag = 0. */

onClipEvent (load) {
    origX = this._x;
    origY = this._y;
    noDrag = 0;
} /* On the event of mouse click on ADC block and recognise that
noDrag = 0, allows dragging function to start*/

onClipEvent (mouseDown){
    if ((this.hitTest(_root._xmouse, _root._ymouse)) and (noDrag == 0))
           {
        this.startDrag();
             }
```

```
}

/*On the event of mouse up from the block, dragging function
stop. Reading on current mouse release position (x, y). Do a
condition check, if current position is same as the designation
position which is _parent.dropZone2._x and _parent.dropZone2._y,
the ADC block will stay at the yellow box and variable noDrag = 1
which disable the dragging function. If the ADC position is not
the same as _parent.dropZone2._x and _parent.dropZone2._y, the
variables this._x and this._y will assigned back to their
original position.*/

onClipEvent (mouseUp) {
    if (this.hitTest(_root._xmouse, _root._ymouse)) {
        this.stopDrag();

        if (_parent.dropZone2.hitTest(this._x, this._y, true)) {
            this._x = _parent.dropZone2._x;
            this._y = _parent.dropZone2._y;
            noDrag = 1;
            }
                        else {
            this._x = origX;
            this._y = origY;
        }
    }
}
```

For the Analog filter blocks, there are two positions which they can be placed. Hence, the code was slightly more complicated and is explained below.

Program code for Analog filter block.

```
/* For Analog filter block.*/ /*On the event when the game is
loading, the original position (x, y) of Analog filter are stored
into variables of this._x and this._y, and var noDrag = 0. */

onClipEvent (load) {
    origX = this._x;
    origY = this._y;
    noDrag = 0;
}

/* On the event of mouse click on the block and recognise that
noDrag = 0, allows dragging function to start*/

onClipEvent (mouseDown) {
    if ((this.hitTest(_root._xmouse, _root._ymouse)) and (noDrag == 0)) {
        this.startDrag();
    }
}

/*On the event of mouse up from the block, dragging function
stop. Reading on current mouse release position (x, y). Do a
condition check, if current position is same as the designation
position which is _parent.dropZone1._x and _parent.dropZone1._y,
or _parent.dropZone5._x and _parent.dropZone5._y the Analog
filter block will stay at the yellow box and variable noDrag = 1
which disable the dragging function. If the block position is not
```

```
the same as either any of the above, the variables this._x and
this._y will assigned back to their original position.*/

onClipEvent (mouseUp) {
    if (this.hitTest(_root._xmouse, _root._ymouse)) {
        this.stopDrag();

        if (_parent.dropZone1.hitTest(this._x, this._y, true)) {

            this._x = _parent.dropZone1._x;
            this._y = _parent.dropZone1._y;
            noDrag = 1;
            }

        else if (_parent.dropZone5.hitTest(this._x, this._y, true)) {

            this._x = _parent.dropZone5._x;
            this._y = _parent.dropZone5._y;
            noDrag = 1;
            }

            else {

            this._x = origX;
            this._y = origY;

        }
    }
}
```

The purpose of the submit button was to provide the time taken when the users finished their game. This button was imported in as graphic and then converted into a button. There were also two invisible dotted blue boxes and grey box named 'mess' and 'secs' as shown in Figure 4.30. These two boxes were selected as 'Input text' so that when the submit button is pressed, the assigned message will appear in the two boxes.

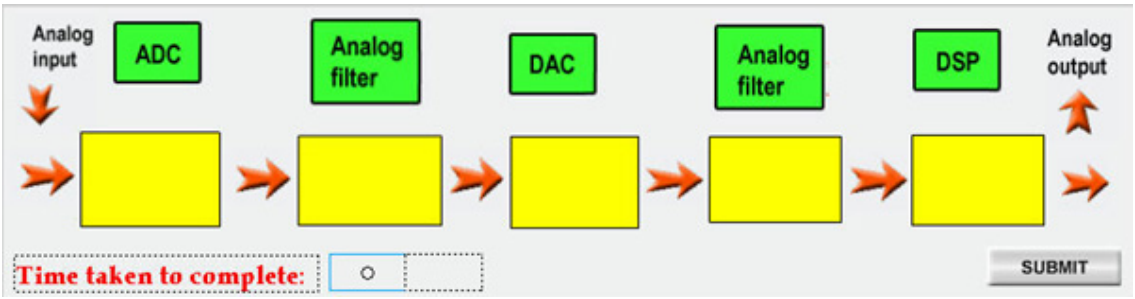

Figure 4.30: Indicting the message box.

Program code for submit button.

```
/* For submit button.*/ /*When the submit button pressed and
released. It stop the timing and get the timing, then print the
timing on to 'mess' and the word "secs" onto 'secs'*/
on(release)
{
    Time=getTimer();
    mess=Time/1000;
    sec="secs"
}
```

The second demonstration is illustrated in Figure 4.31. This learning demonstration serves as a continuation of the previous interactive game. The previous being to test the users' knowledge on DSP system without explaining much on this system while the current demonstration was devised to have users click on the block they wished to understand more about. Each block's purpose is explained in the white screen below, together with any graphic illustrations.

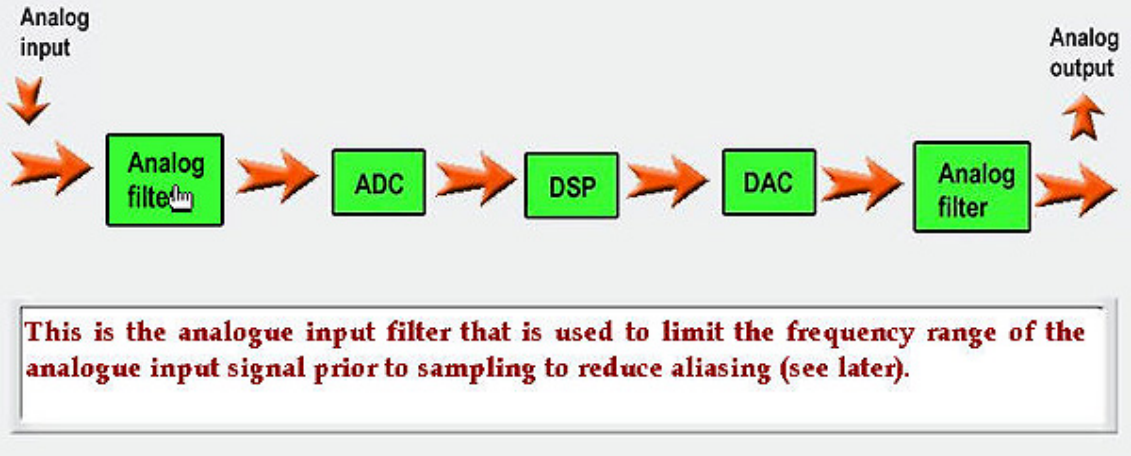

Figure 4.31: Interactive learning demonstration on DSP system.

The development process of this demonstration is as follows. From the figure below, it can be seen that the implementations involved both Timelines and ActionScript programming. The Timelines section will be explained first.

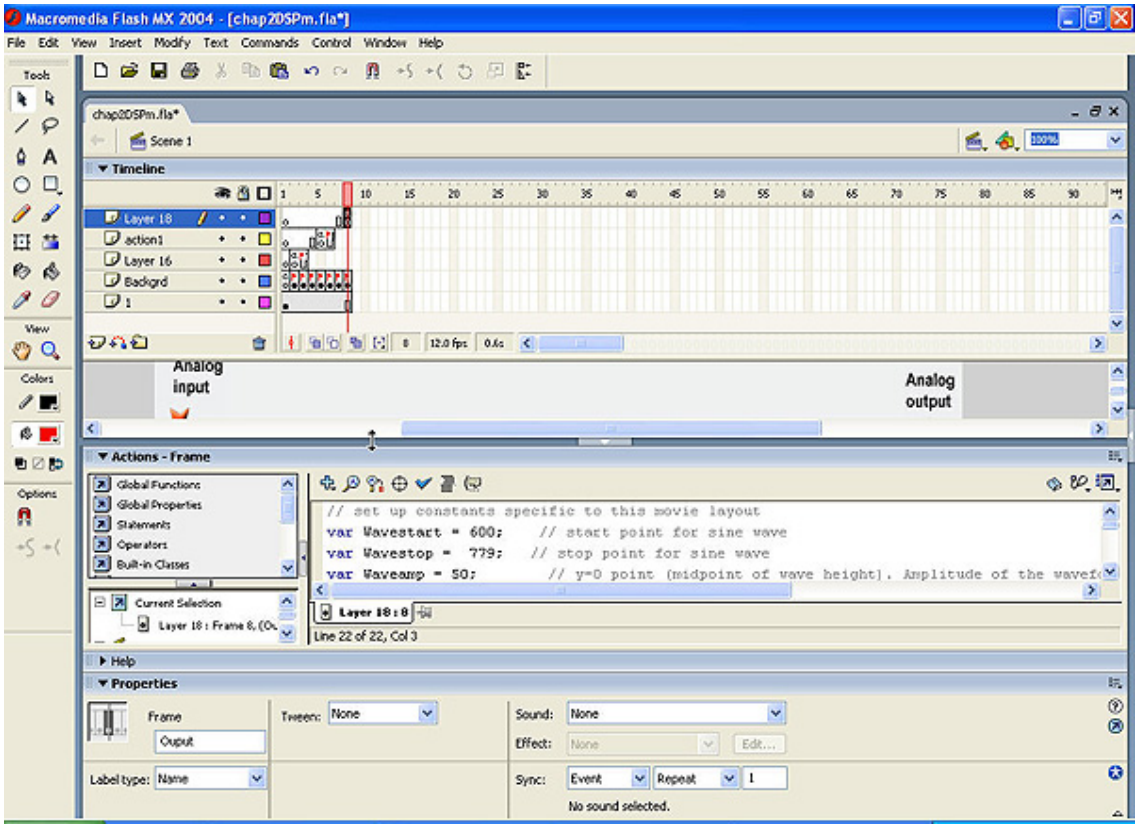

Figure 4.32: Development of DSP demonstration using Flash.

From the screenshots below, it can be seen that the timelines sections are using a total of five layers. The layers were divided as: the whole demonstration, including the individual block explanation, the three animated illustrations and the graphical block diagrams themselves. The bottom layer acts as the base layer which contained all the graphics while the second layer contained all the individual block explanations which can be identified by the little red flag symbol on each frame. These flags each have their own name and hence when the program calls out the flag name, it will jump straight to that particular flag and play the explanation promptly. This is shown in the next two figures. Notice that the two figures below highlight the different frames and the explanation in the blue box changes. The frame name that appears in the bottom properties section under 'Frame' is different as well.

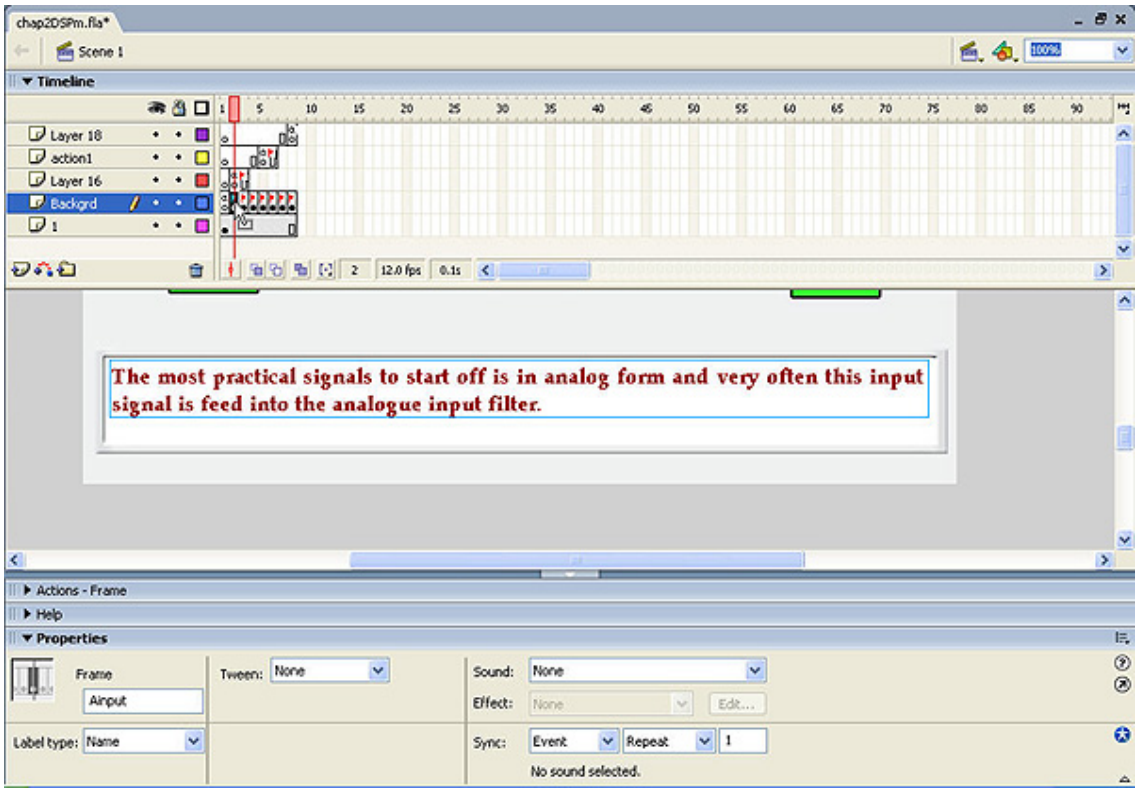

Figure 4.33: Illustrating the Frame name 'Ainput' and its explanation in red.

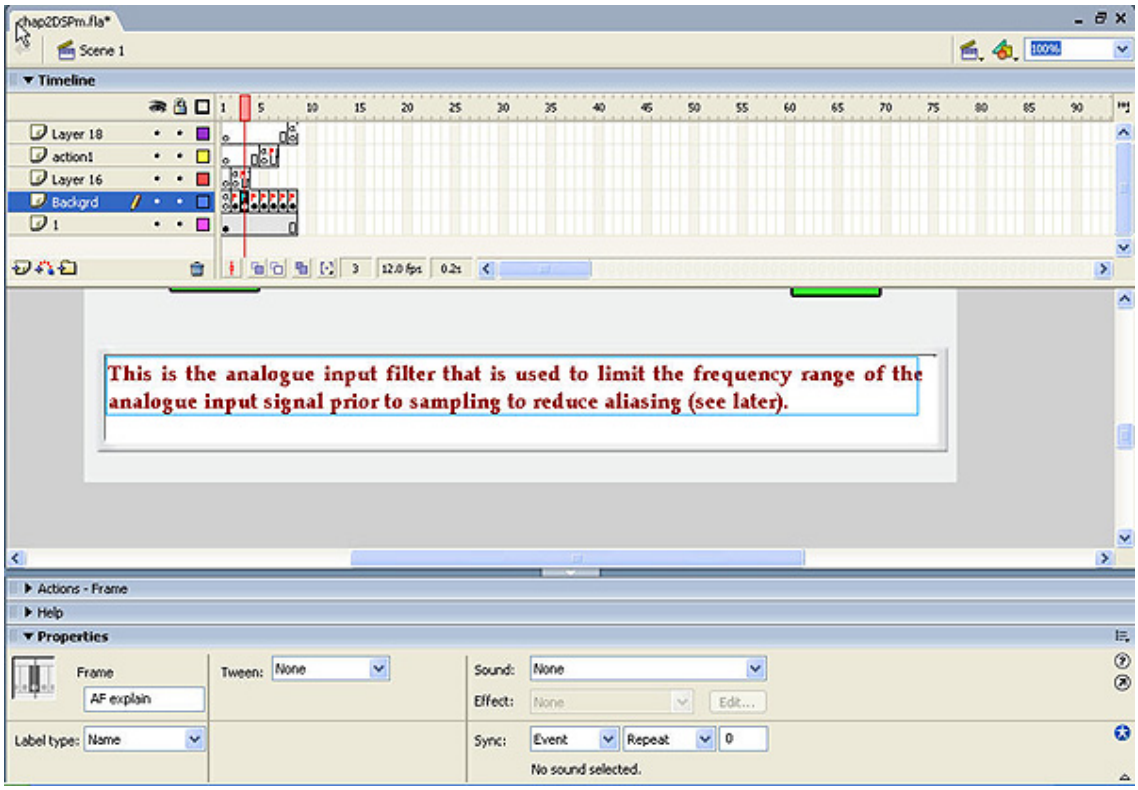

Figure 4.34: Illustrating another Frame name 'AF explain' and its explanation in red.

The next three layers were meant for the wave illustrations which will be shown in the later figures. One layer was used for the Analog input, another for the DSP processed signal and the last one was for the converted Analog output. The ActionScript for the particular frame that is highlighted in black in the timeline section can be seen in Figure 4.35. It has its own frame name 'Input' and it actually produced a sine wave which is shown in Figure 4.36 the one on the left side. This was done similarly for the next two layers on top which illustrated a discrete signal as well as another sine wave as shown in Figure 4.36 the one on the right. The ActionScript programming for these animated signal were recycled from the previously creations except for some changes in the parameters.

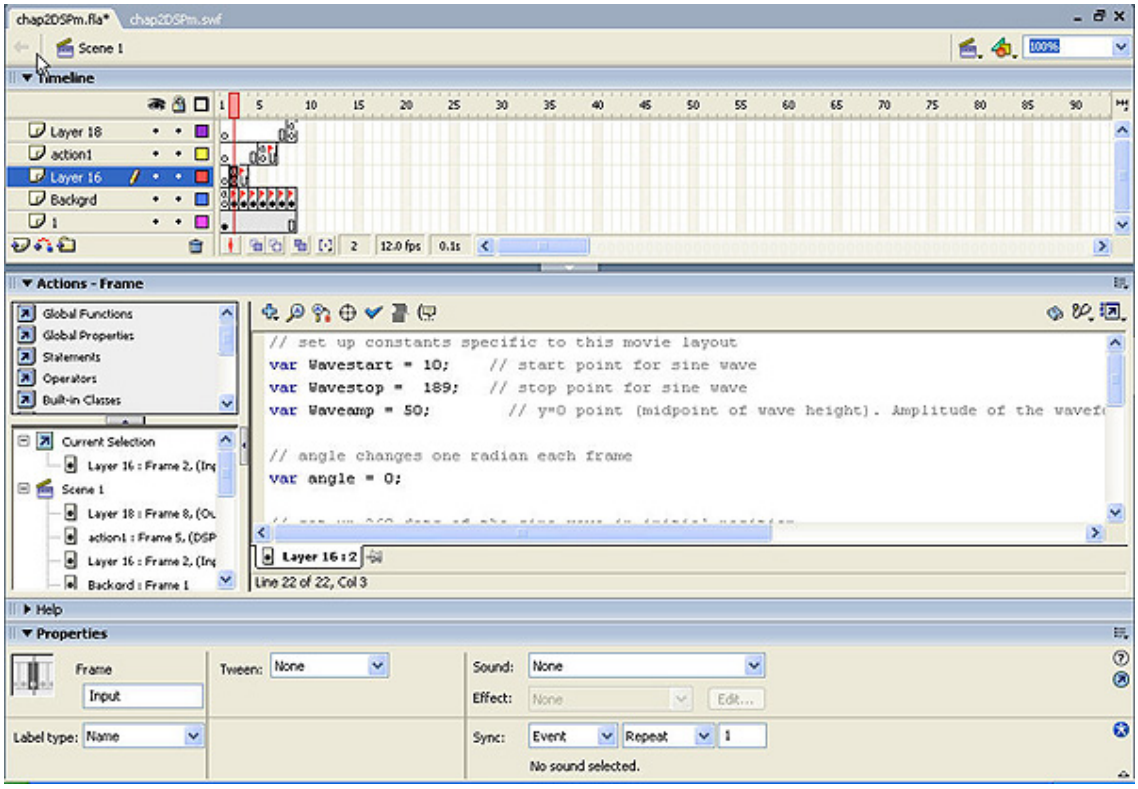

Figure 4.35: Illustrating another Frame name 'Input' and its ActionScript.

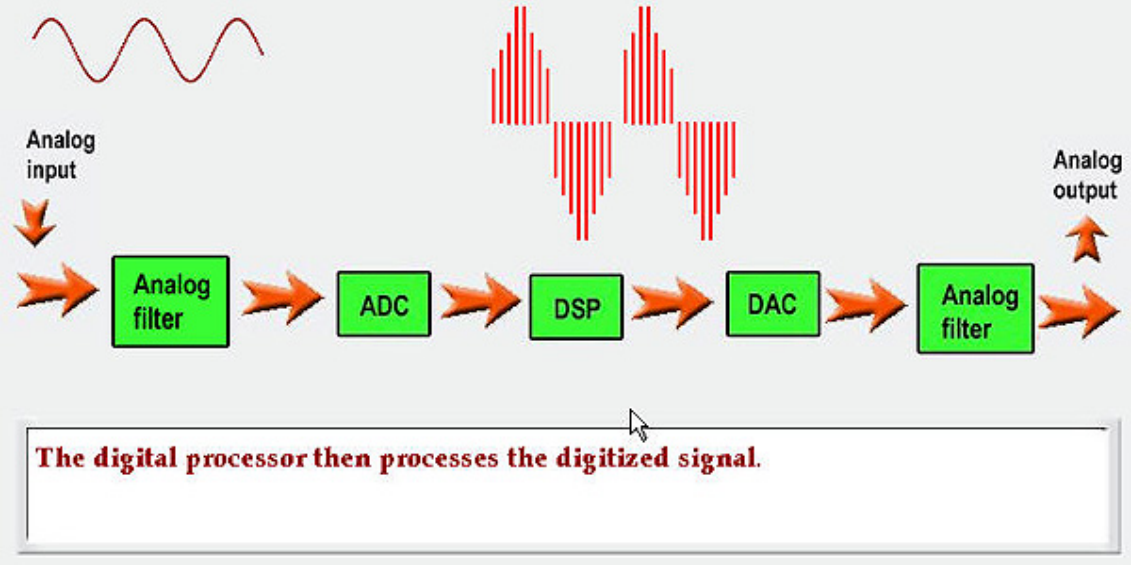

Figure 4.36: Illustrations and its explanations shown.

With the programming of the illustrations and the skeleton of the timeline produced, the next process was to link the pieces together with the individual blocks. First, the blocks were converted into buttons and the ActionScript code was then added to each button as shown and explained below. The programming source codes are similar to the rest of the buttons, the only difference being their frame names.

Programming source codes for clickable blocks.

```
/* For Analog input block*/ /* On the event the mouse press onto
this block, jump to both Frame 'Input' and Frame 'Ainput', play
these frames only. Frame 'Input' was the animated sine wave and
Frame 'Ainput' was the explanation for the block. Once mouse
press on it, it played both Frames.

on(press) {
    gotoAndStop("Input");
    gotoAndStop("Ainput");
}
```

The programming parts were more straight-forward, this round the problem lie with the timelines. Quite some time was taken to find out how to call out the Frame names and the sequence of the frames placing.

4.3.4 Exercises and Mini-quizzes implementations

These implementations come in handy when users wish to test their understanding on their concept. The questions were handpicked beforehand to ensure that they were relevant to the topics and their solutions were worked out too. With experience from the prior sections, this section was developed much faster than expected. An example on how an exercise and quiz were developed is as follows since the concept can be applied to the rest of the exercises and quizzes.

For this exercise, users can key in any input which they may have on the left hand side, pressed the 'submit' button and the matching parameters would be calculated out on the right hand side.

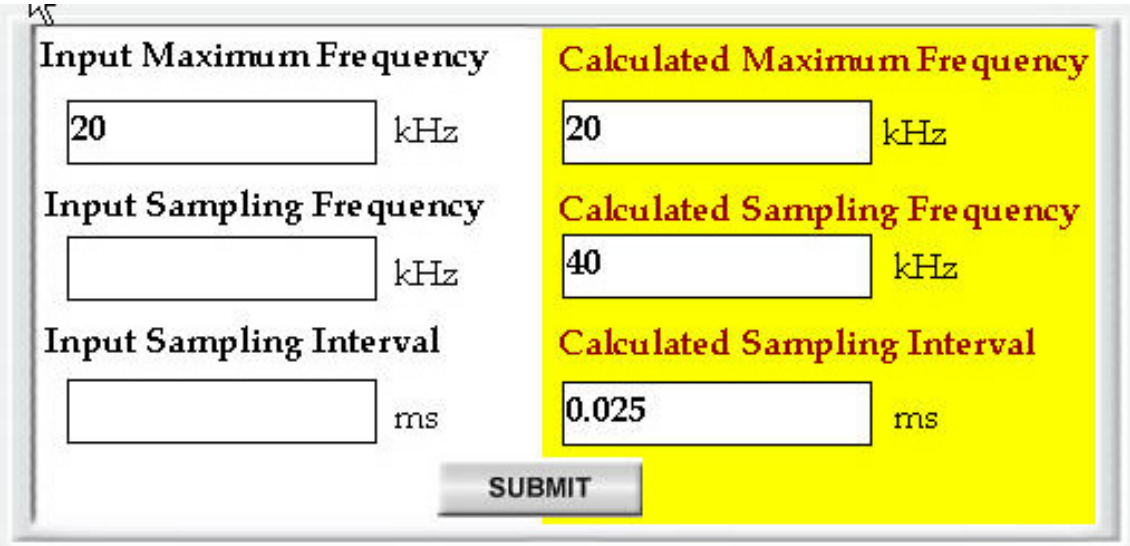

Figure 4.37: One of the exercises in Chapter 2.

The boxes on the left were selected as 'Input text', and these boxes were called different variables names (in red). These variables names were used in the programming to store the user's input. While boxes on the right were selected as 'Dynamic text', for calculated parameters.

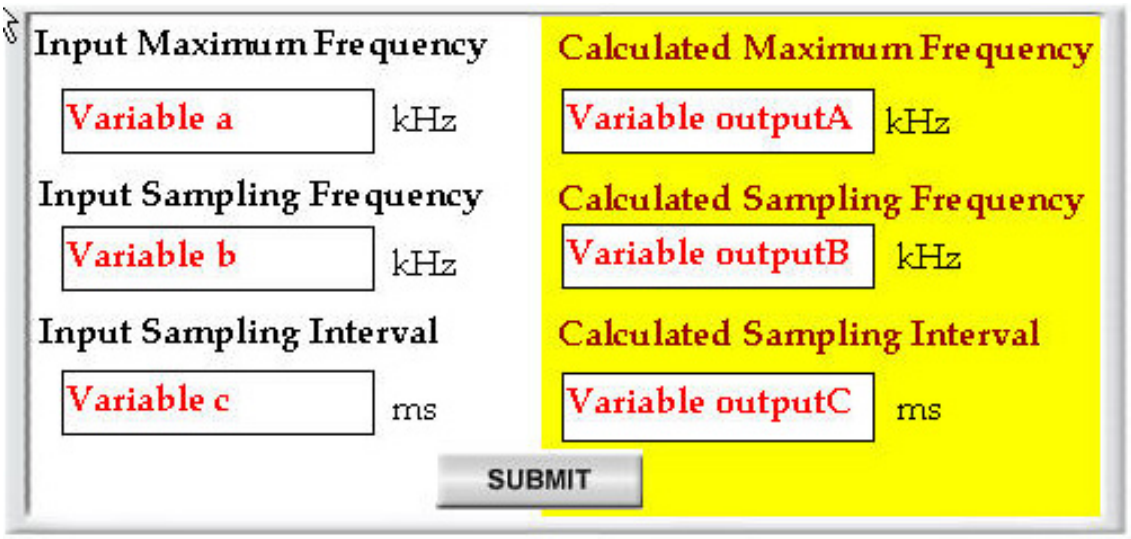

Figure 4.38: With variable name attached.

The source code below was used to program the 'submit' button. It mainly uses condition checking and a formula to calculate the related parameters.

Program code for the exercise.

```
/* For Chapter 2 exercise*/

/* On the event of mouse release, several conditions check were
done in order to perform the correct calculation of the
parameters. */

on (release) {
    if (outputA = a)
    {           outputB = a*2;
        outputC = 1/outputB;
```

```
    }
    else if (outputB = b)
    {           outputA = b*2;
        outputC = 1/b;
     }
     else
     {           outputC=c;
         outputB=1/c;
         outputA=outputB/2;
     }
}

}
```

The user quiz has been designed to be simple to operate. When users complete all the questions and press the 'submit' button, they will be told if they answered correctly or not. If they got the wrong answer, they can either choose to try the question again or press the 'get answer' button to retrieve the explained solution.

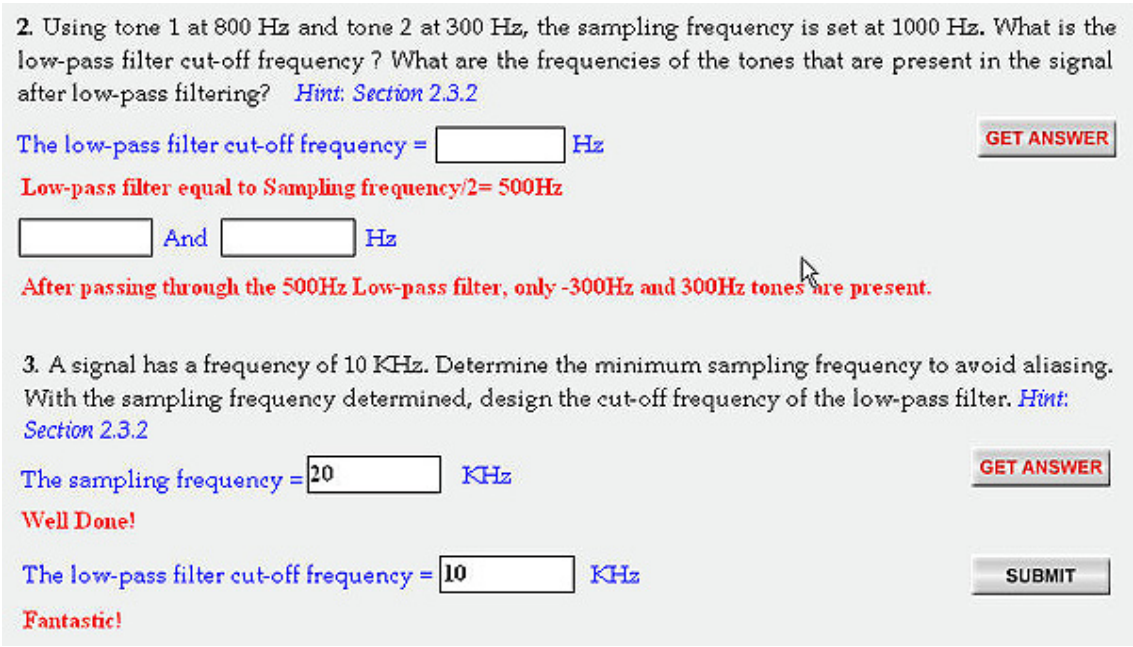

Figure 4.39: An example of quiz for Chapter 2.

Similarly for the quiz, several 'Input text' boxes were created. The variables names for boxes were written in red for easy reference. There are two explained program codes illustrated below are for the quiz shown in the below Figure.

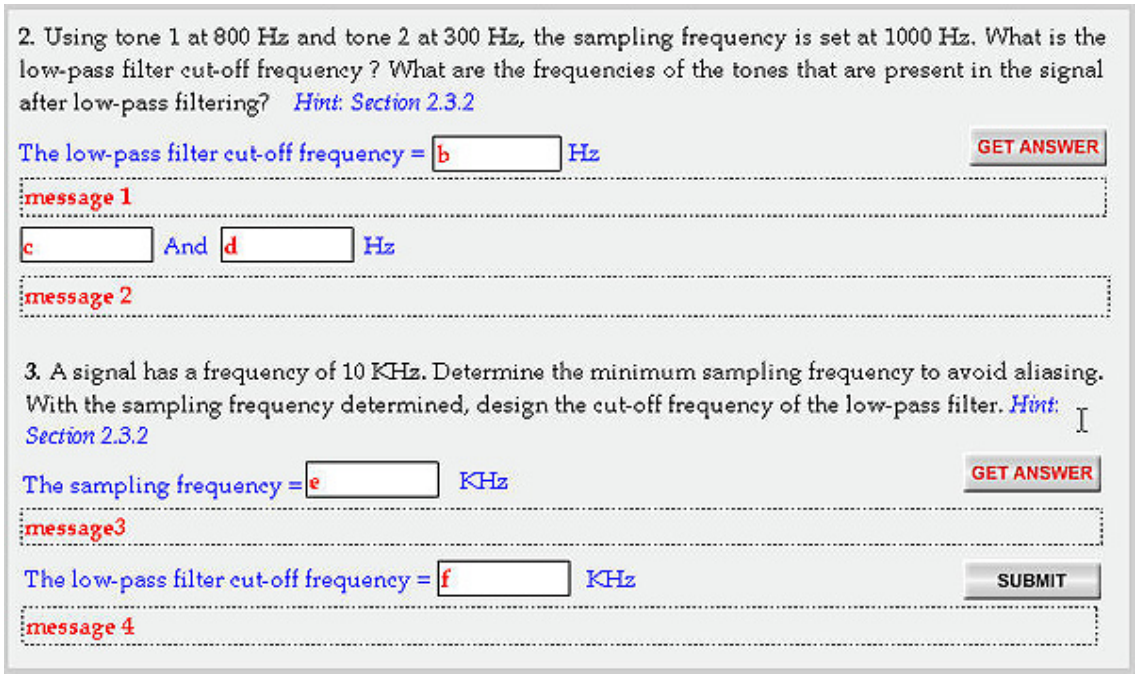

Figure 4.40: An example of quiz for Chapter 2 with variable names in red.

Program code for 'GET ANSWER' button.

```
//When mouse click is release on the "GET ANSWER" button
on (release)
    {
        /*Write the following message in the dynamic input box name
        message1 and message 2.*/

        message1 = "Low-pass filter equal to Sampling frequency/2= 500Hz ";
        message2 = "After passing through the 500Hz Low-pass filter,
        only -300Hz and 300Hz tones are present.";
     }
```

Program code for 'Submit' button.

```
/* On the event of mouse release, several conditions check are
performed. If the answer matches the correct ones, then correct
message would be printed out. Else, other message will printed
out on the correct boxes accordingly. */

on (release) {
    if (a == "no" || a == "No" || a == "NO"){
     message = "Correct!";
     }else {
         message = "Are you sure?";
     }
     if (b == "500"){
     message1 = "Correct!";
     }else {
         message1 = "Try again";
     }
    if (c == "-300" && d == "300" || c == "300" && d == "-300"){
     message2 = "Great work!";
     }else {
         message2 = "You can do better";
     }
      if (e == "20" || e == "20"){
     message3 = "Well Done!";
     }else {
         message3 = "WRONG!";
     }
if (f == "10" || f == "10"){
     message4 = "Fantastic!";
     }else {
```

```
        message4 = "Incorrect!";
    }
}
```

4.4 Upload Chapter online

The above sections have illustrated almost all the content that was developed, and the last step would be to upload them online. This was totally different from just visiting websites and required some time to familiarise oneself with the procedures. Besides that, a free web space had to be found to store the project and to display the web pages - this was located at `www.geocities.com/skylark7831/`.

A problem encountered was that files that were to be uploaded were not allowed to have any spacing in their names and that the first page had to be named index.html. As a result, several files had to be renamed as well as some of the links in the web pages so as to fit the directory structure of the website file manager. This was necessary as the website could not be properly viewed otherwise.

Figure 4.41 shows the GeoCities File manager. This is where the folders were created with the same layout as the C drive. There are three main folders, namely Layouts, NavigationButtons and Screen.

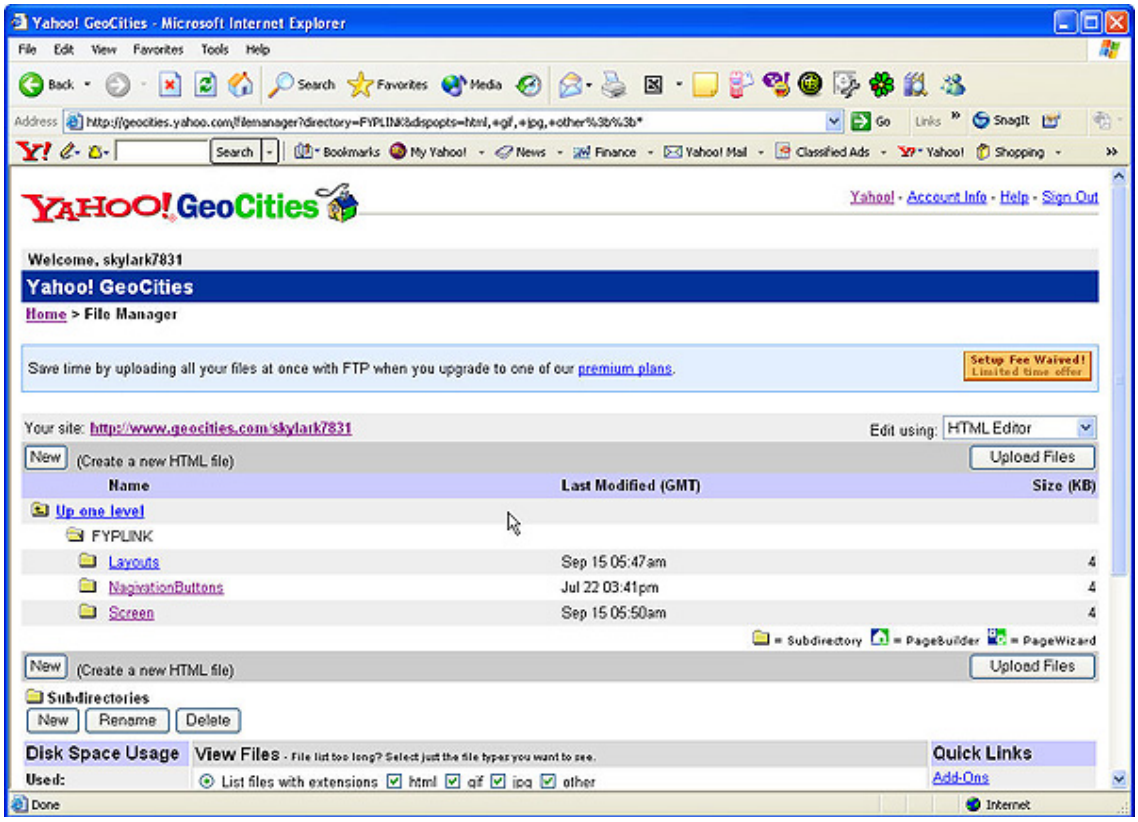

Figure 4.41: Geocities File manager.

Figure 4.42 shows what files were inside the Layout folder. These represent the main layout template for the different page.

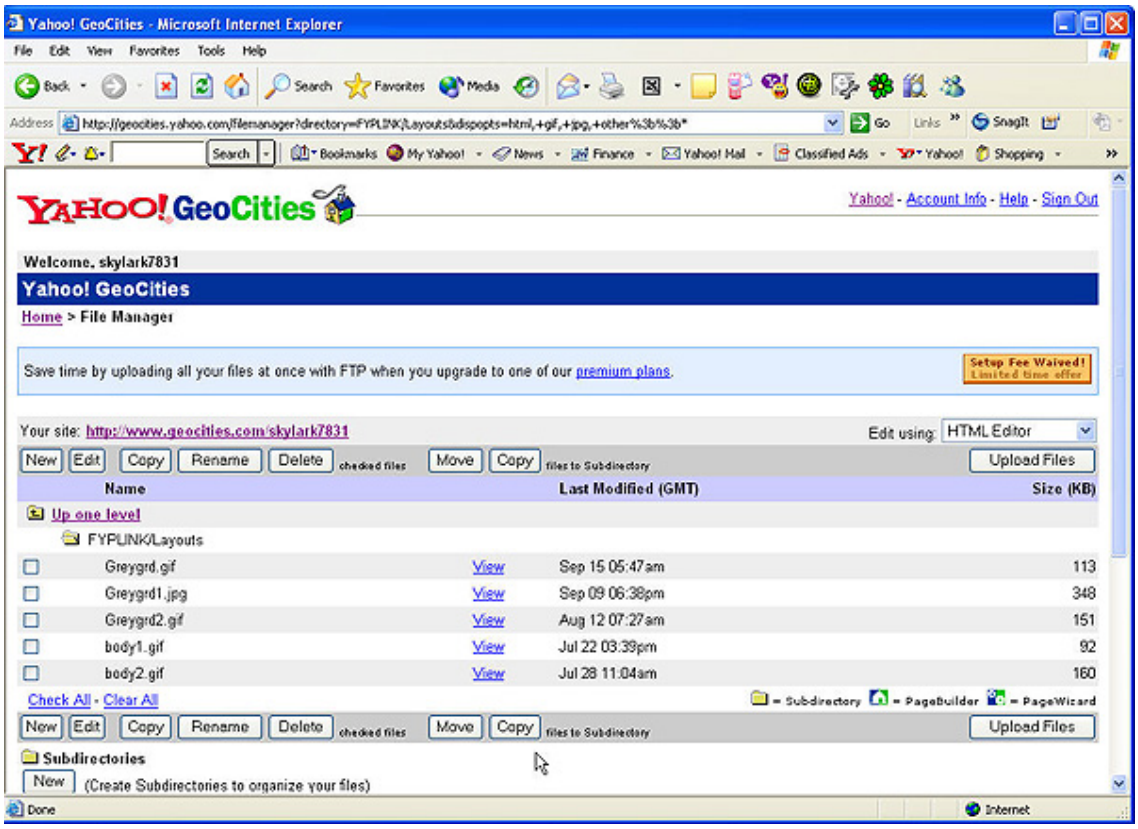

Figure 4.42: The files inside Layouts folder.

The screenshot below shows three subfolders inside the NavigationButtons folder. The three subfolders contained the button images for 'Normal', 'Mouse Down' and 'Mouse Over'.

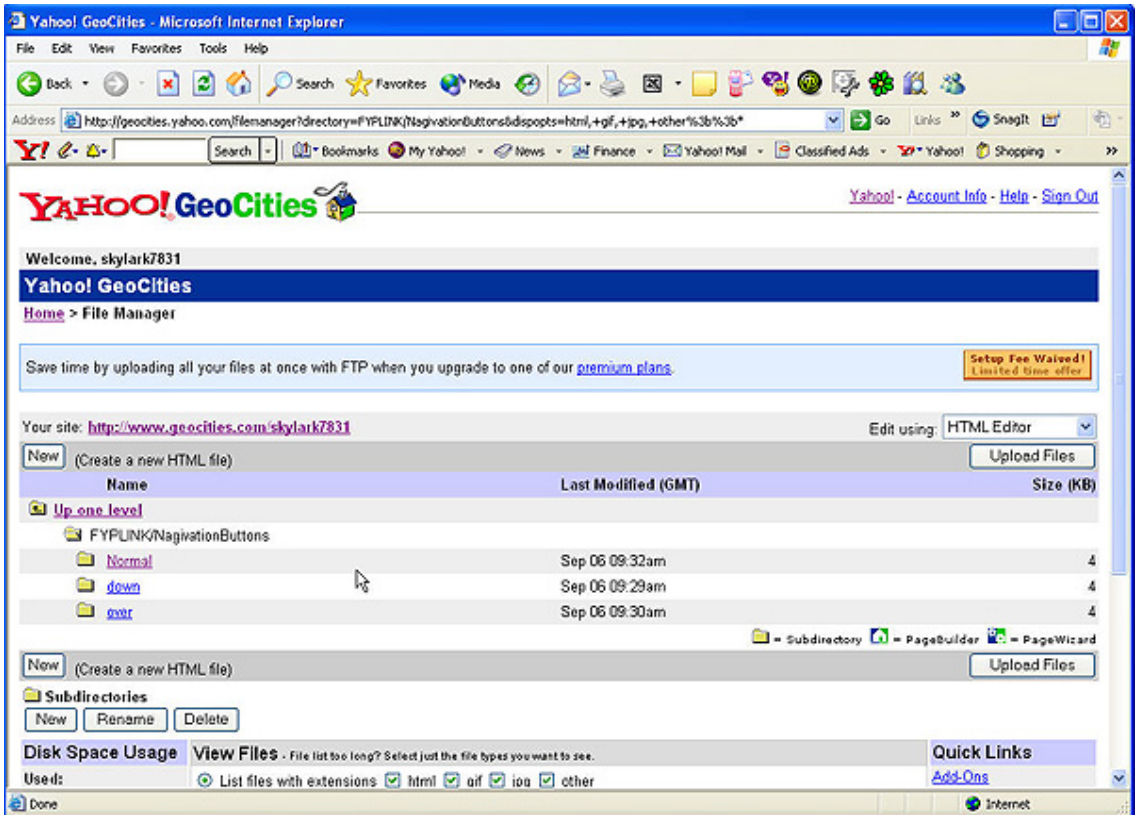

Figure 4.43: The files inside NavigationButtons.

The Figure 4.44 shows all the html files that were inside the Screen folder including the subfolders Flash and Graphics. The Flash folder contained all files created using Flash or ActionScript while the Graphics folder held all the static graphics and images files.

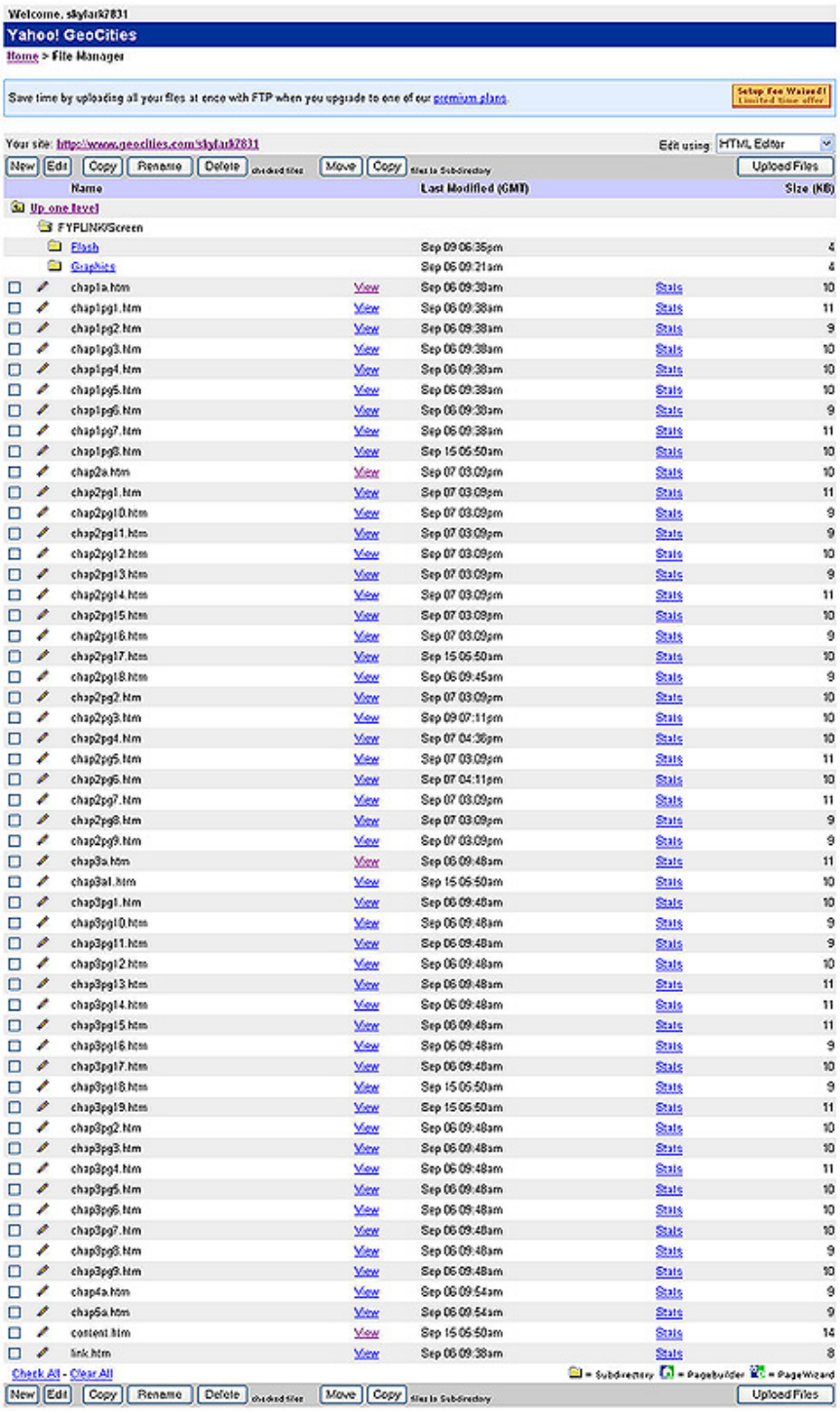

Figure 4.44: Showing the files uploading page.

For uploading the files onto the website, simply select its relevant folder and click the upload button. Then browse for the files to be uploaded as shown below.

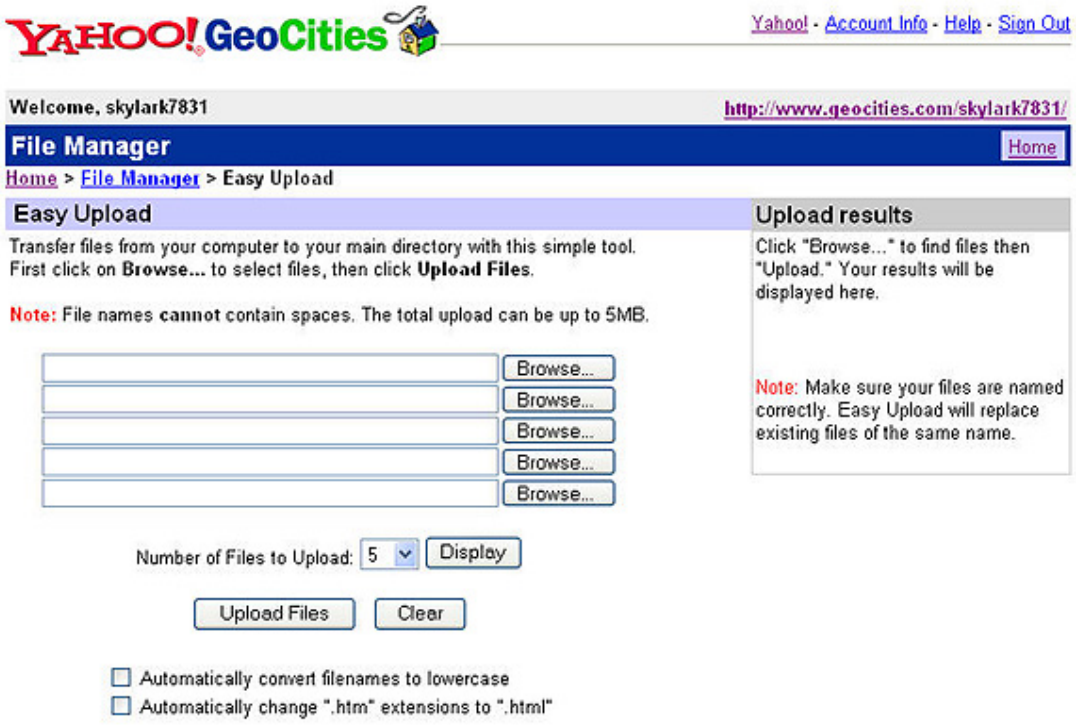

Figure 4.45: Showing all the html pages in Screen folder.

4.5 Chapter Summary

In summary, the chapter introduces the general steps it has been through to accomplish the interactive functions for this website. Although the steps' descriptions were short, there was a good amount of trial and error that was required to achieve the desired looks and effects. A brief overview on how the overall website theme layout and the individual graphics were created using Adobe Photoshop. From the created graphics, they were then transformed into usable graphics such as the clickable buttons, dynamic graphics and exercises. The processes of these transformation using Macromedia Dreamweaver and Macromedia Flash (ActionScript) were also illustrated, followed by how these transformed screens were uploaded online to become a website.

Chapter 5

Testing Methodology and Results

5.1 Testing Methodology

This was carried out according to the methodology plan in Figure 3.1. Although it was planned at the later stage, it was conducted throughout the development process too. This was done so that mistakes could be corrected at an earlier stage. As the project continues to be built on, mistakes and troubleshooting become much harder at a later stage.

To standardise the testing methods for every chapter completed, a flowchart sequence was drawn up for the testing methodology shown in Figure 5.1.

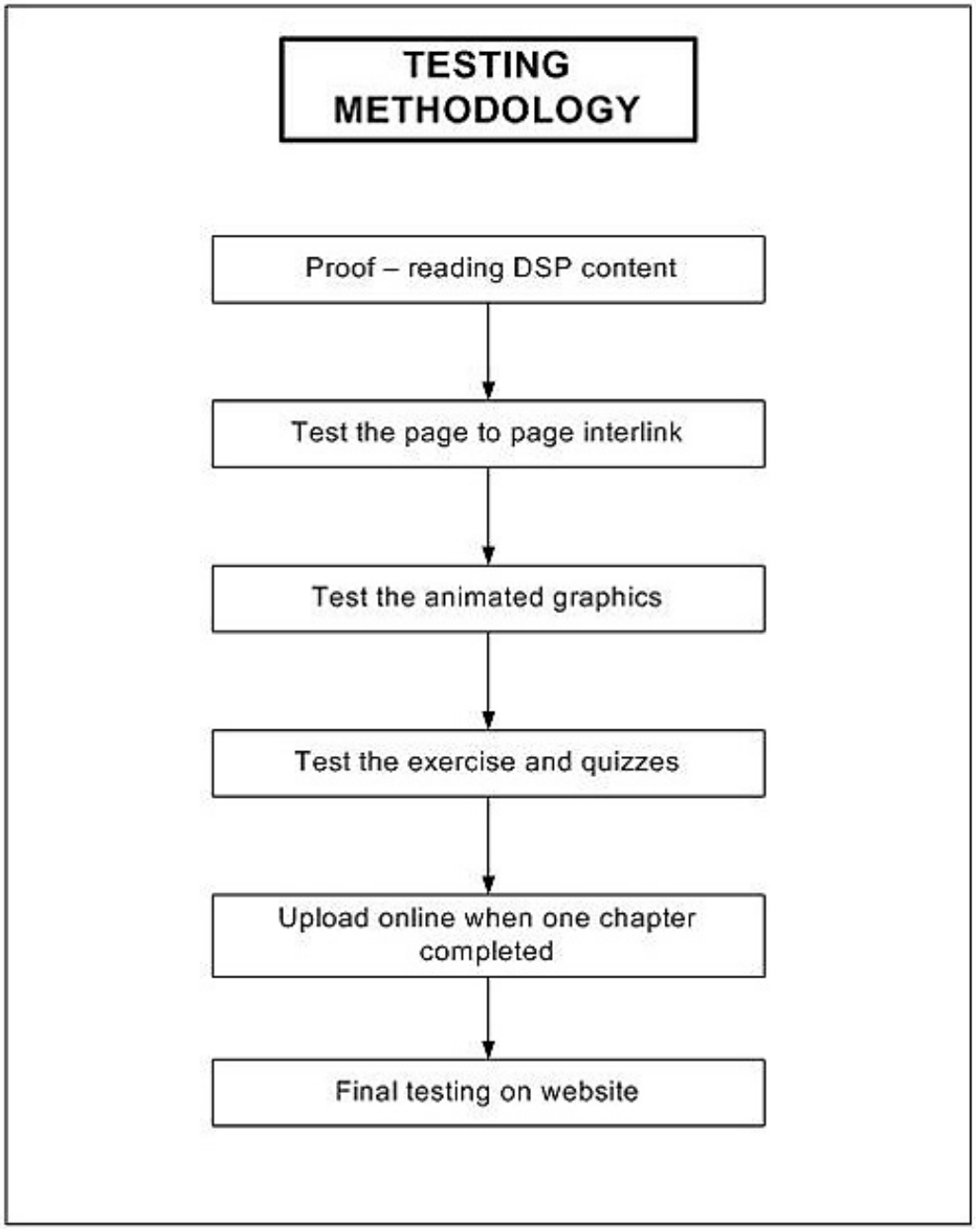

Figure 5.1: Testing Methodology Flowchart.

5.1.1 Proof-reading DSP content

A chapter was considered complete once the content including the text, static images, animated graphics, exercises and quizzes were put together. Before the chapter was uploaded online, it had to go through a series of testing, which is highlighted in Figure 5.1.

One of the tests was to proof-read the text content and the illustrations, even though the composed content was read beforehand. The objective behind this was to ensure that the text for that page was correctly put together with the illustrations and to spot any missing illustrations or misaligned content.

5.1.2 Interactive functions

After confirming the text content was satisfactory. The interactive functions were the next components that were tested. The interactive functions consisted of several components which covered the page to page interlink, the animated graphics and the exercises and quizzes. During the first round of text content testing, the page to page interlink was tested at the same time since the content went hand in hand with the sequence of the pages so once the text was confirmed correct, it meant that the interlink page should be fine too. However, every page was tested on other links such as the content page, introduction page to make sure there was no malfunctioning links for any of the buttons in each page.

The animated graphics testing was carried out visually while going through the pages. It can be safely said that the animated graphics were already tested several times till an acceptable standard was achieved during the development phase before it being imported onto the screens. However the graphics were rechecked again by flipping through the pages ensuring that the animated graphics were loading and moving correctly.

Similarly, the interactive learning demonstrations, exercises and quizzes were thoroughly testing before being imported in. Testing for the exercises and quizzes was performed by trying different ways of playing with the demonstrations and by inputting combinations of different characters, and pressing buttons in different sequences. The testing also ensured that keying in the correct answer would yield the success message.

After validating that the tests had achieved an acceptable standard, the entire chapters were uploaded online, followed by another round of quick testing online by going through the website and ensuring the pages were in sequence, the static, animated graphics, the exercise and quizzes were working correctly.

As a result of prior regular testing during development, it can be said that nothing major was found during the actual testing phase and this helped to save time and effort.

5.1.3 Feedbacks

When the chapter was posted online, emails were sent to friends with and without DSP background. Feedbacks and suggestions obtained were helpful and encouraging. In the early stages, the feedback received was that the overall website was plain and suggestions were given to add more animated illustrations and make the lesson more interactive. Other suggestions were to add more applicable topics and real life DSP applications as simply having illustrations and plain text was too boring. Also suggested was that more colour text be used.

Based on the feedbacks and suggestions above, appropriate changes and improvements were made to better enhance the website. More static and animated illustrations were added in and interactive game and demonstrations were included. Some plain text sections were modified to have appropriate colouring although this was done in moderation so as to maintain readability.

5.2 Results

The objectives and specifications required by the project were all fulfilled and the website is up and running at `www.geocities.com/skylark7831/`.

In order to furnish a full outlook on the working website, various screenshots of the working products are shown in the following pages. However, as the interactive functions on the website cannot be shown here, it would be better to visit the actual website to get an actual feel on the finished product.

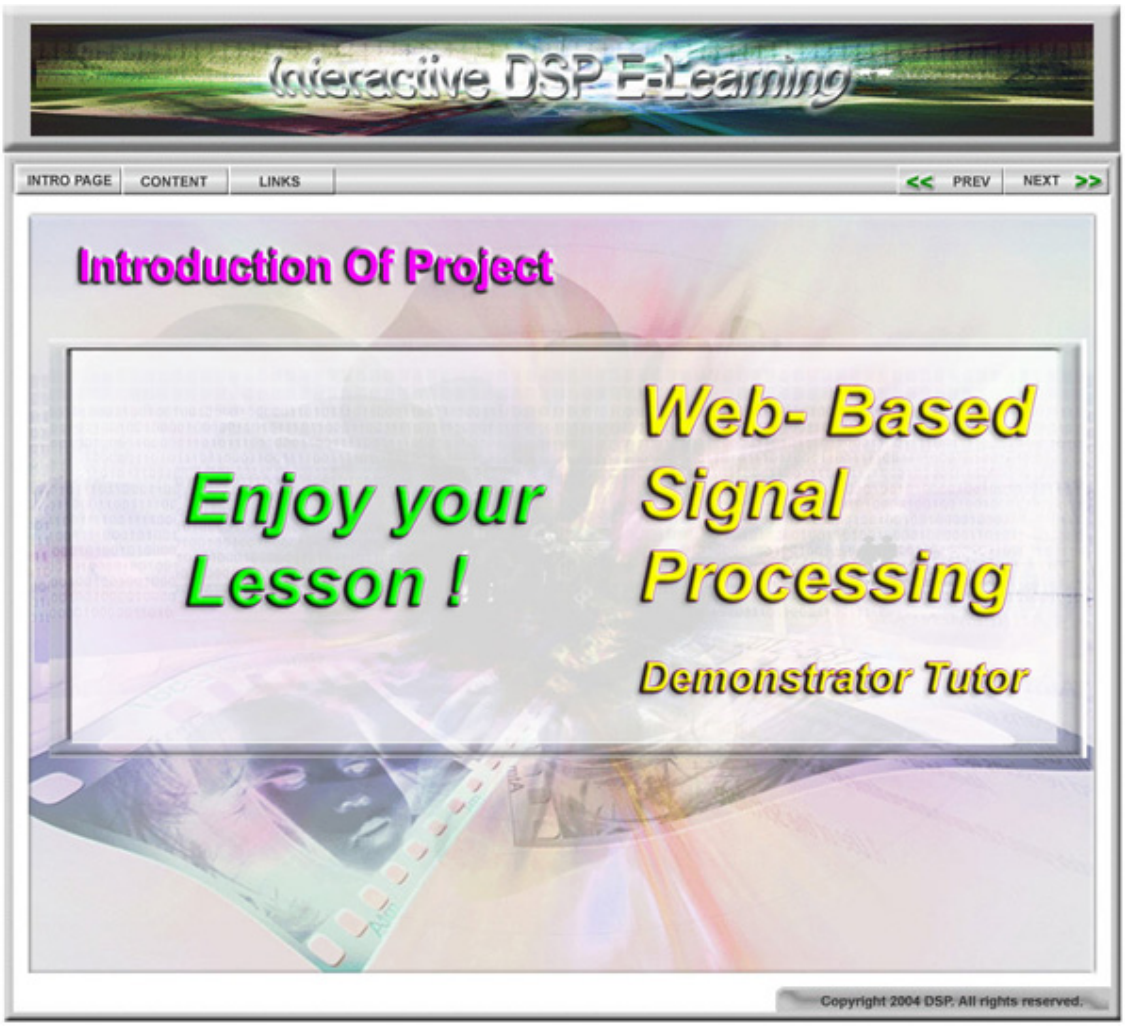

Figure 5.2: Screenshots of Main project page.

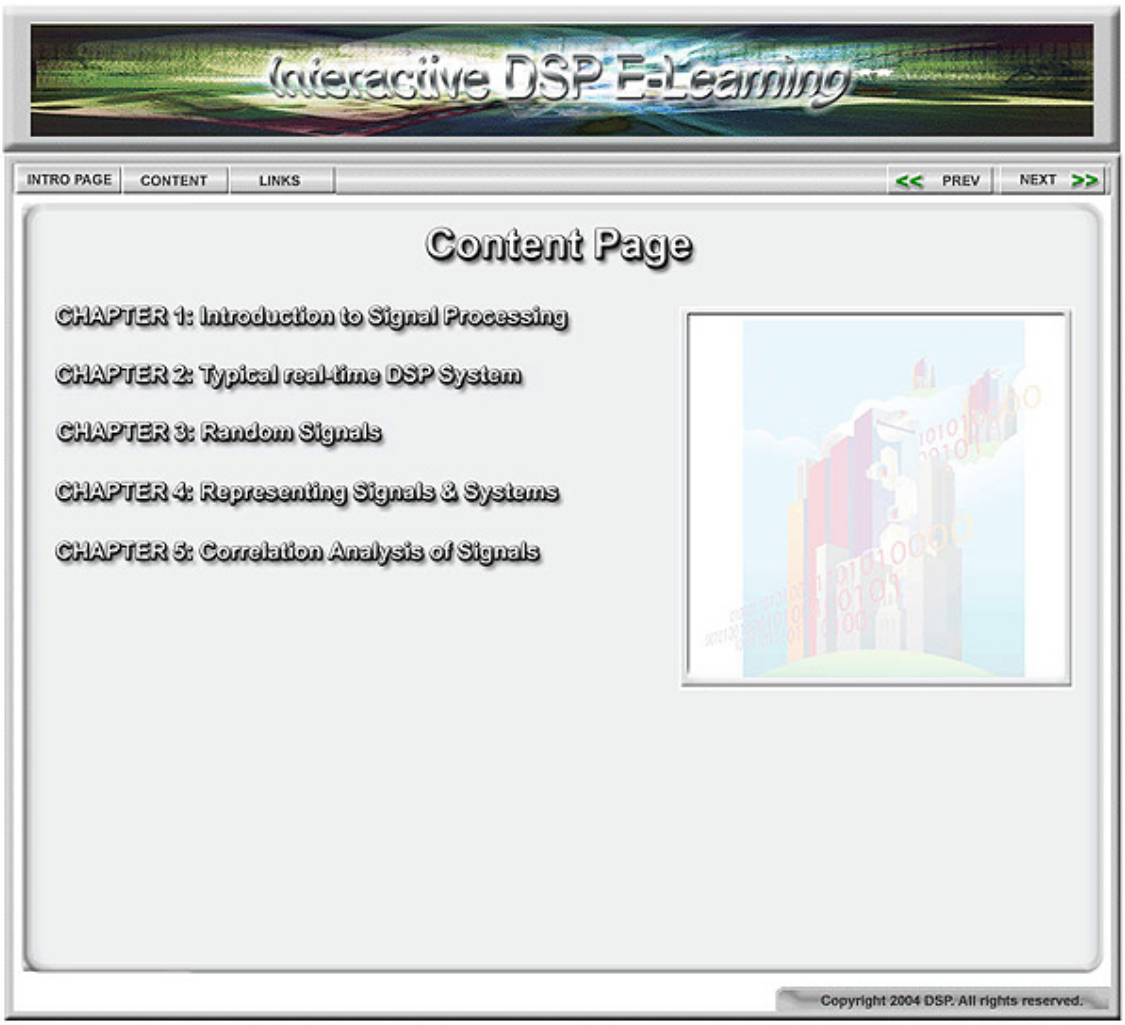

Figure 5.3: Screenshots of project Content Page.

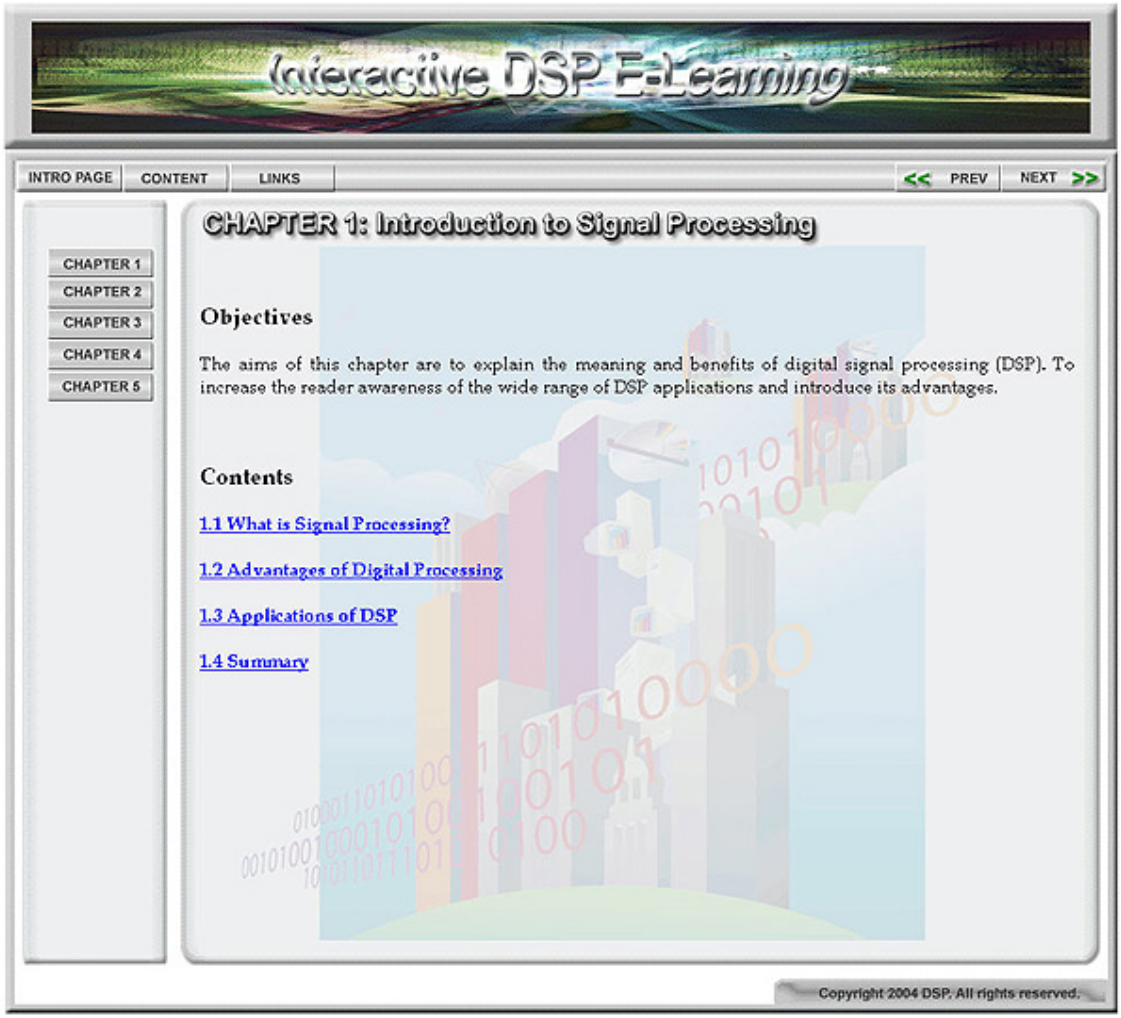

Figure 5.4: Screenshots of Chapter 1 main page.

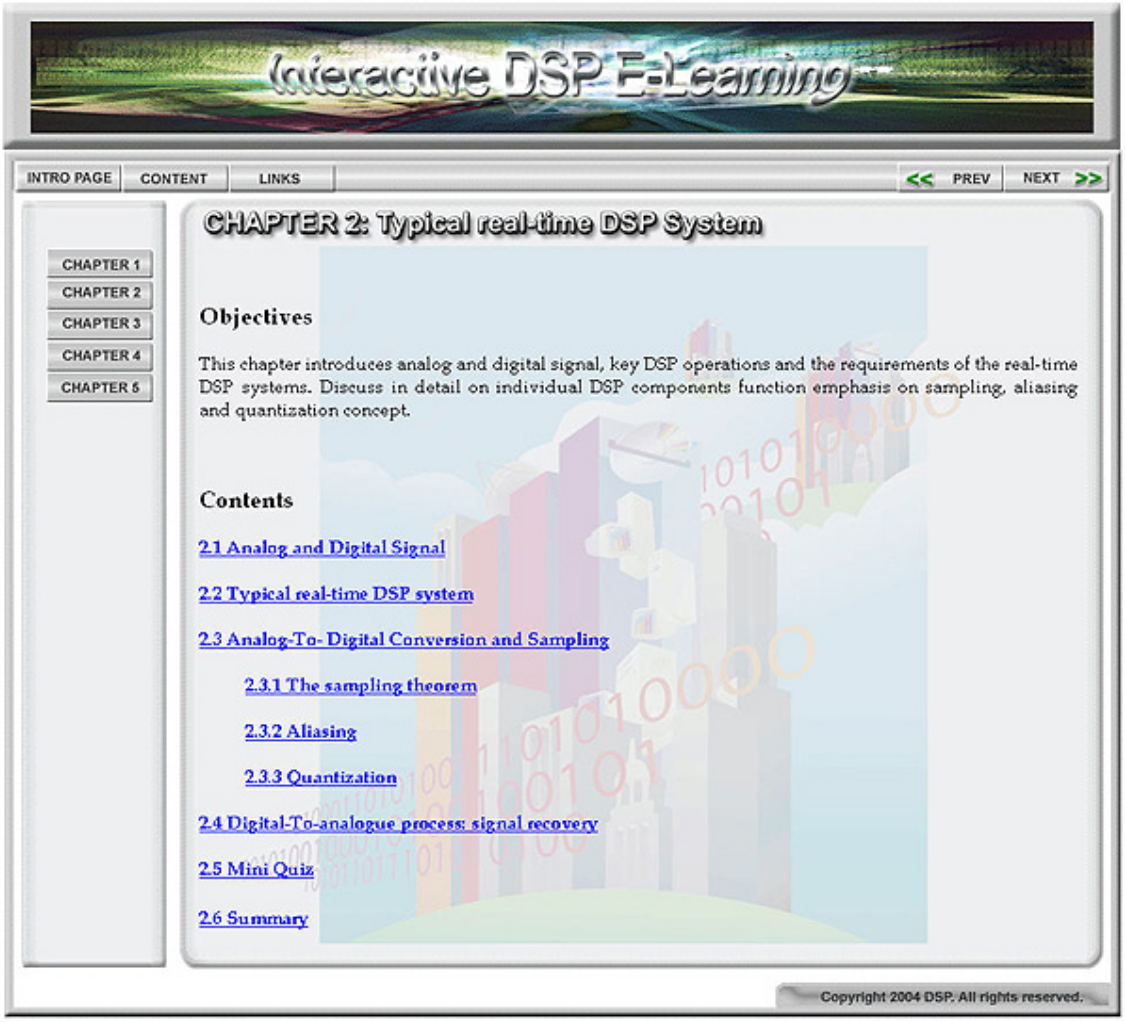

Figure 5.5: Screenshots of Chapter 1 content.

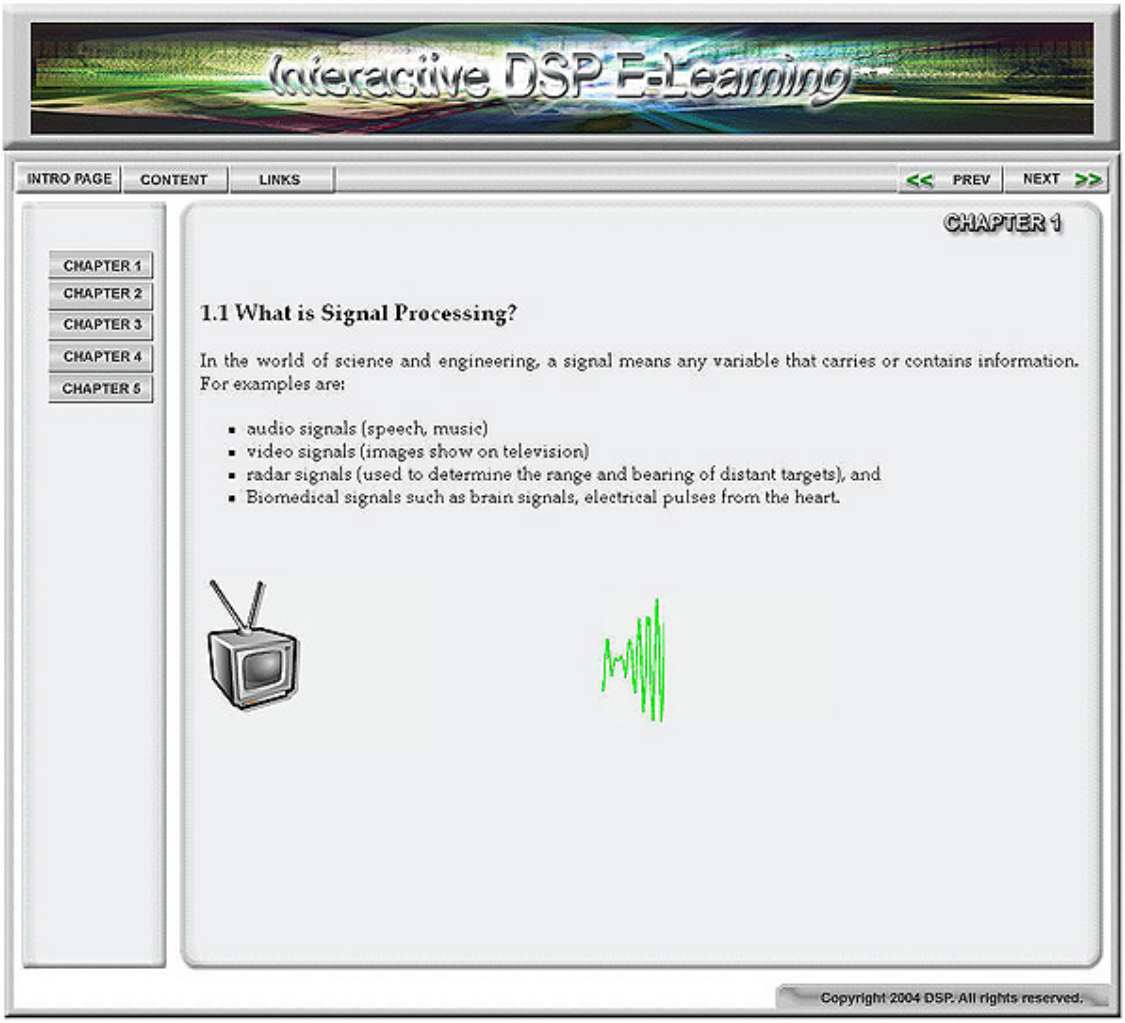

Figure 5.6: Screenshots of Chapter 2 main page.

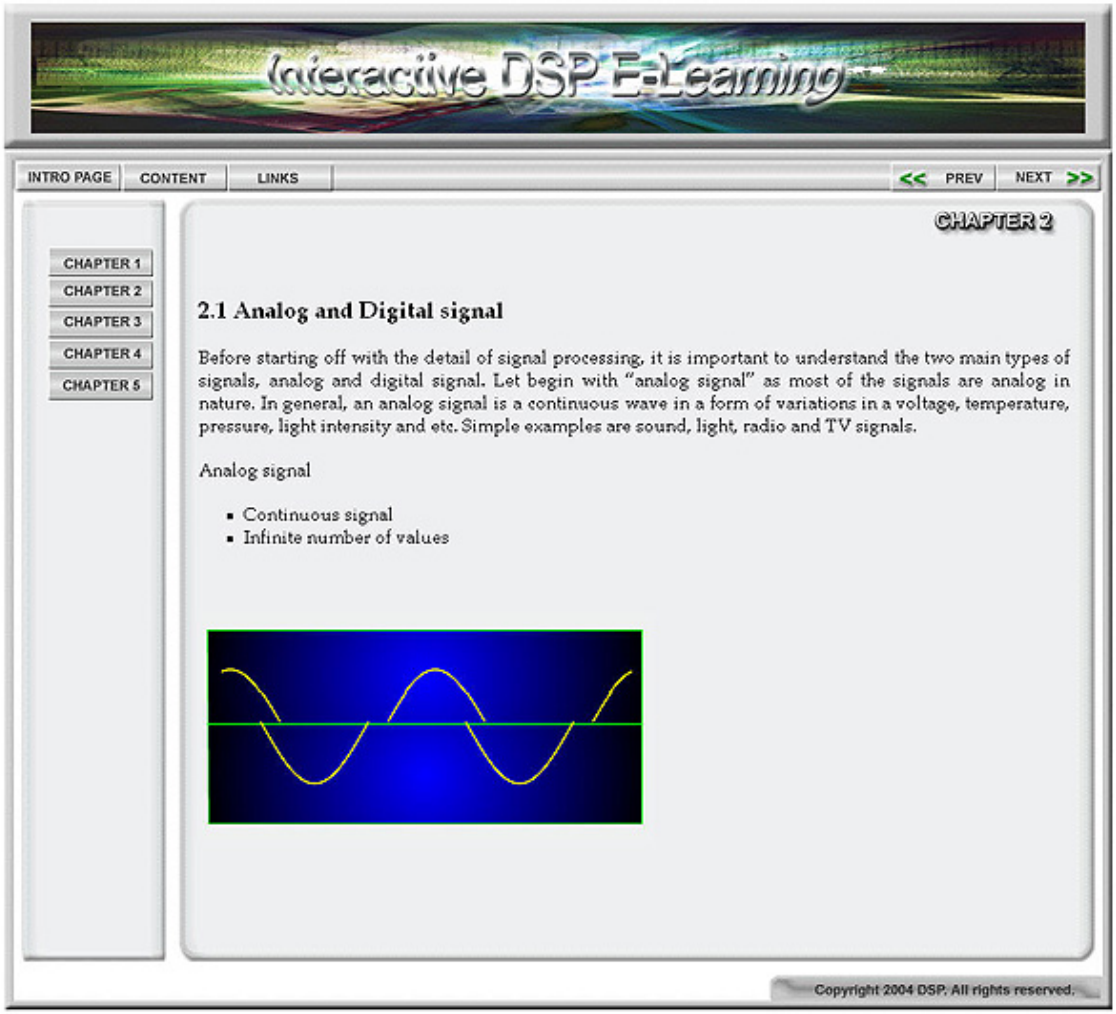

Figure 5.7: Screenshots of Chapter 2 animated sine wave.

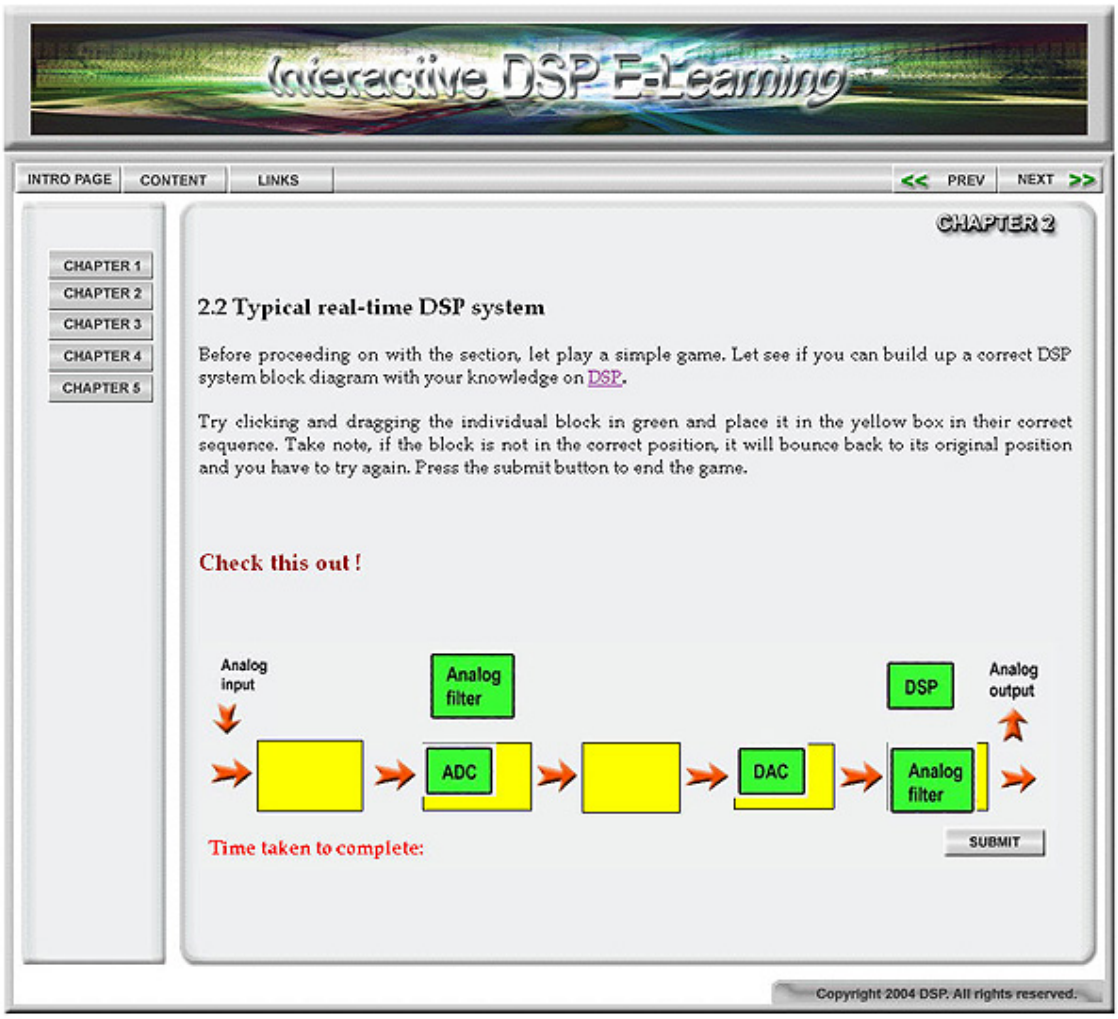

Figure 5.8: Screenshots of Chapter 2, interactive learning DSP system diagrams.

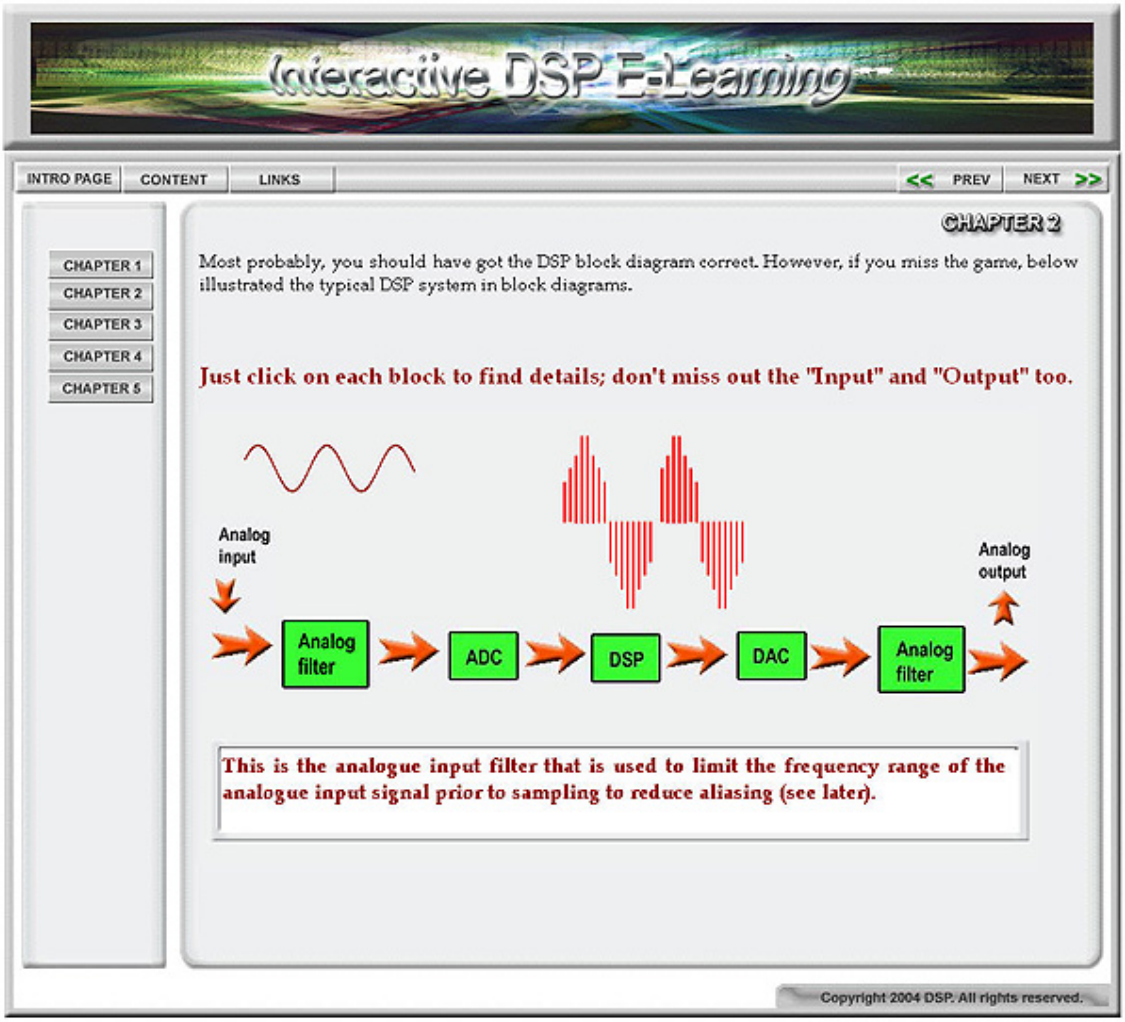

Figure 5.9: Screenshots of Chapter 2, interactive learning by clicking on the block for explanation of its purpose.

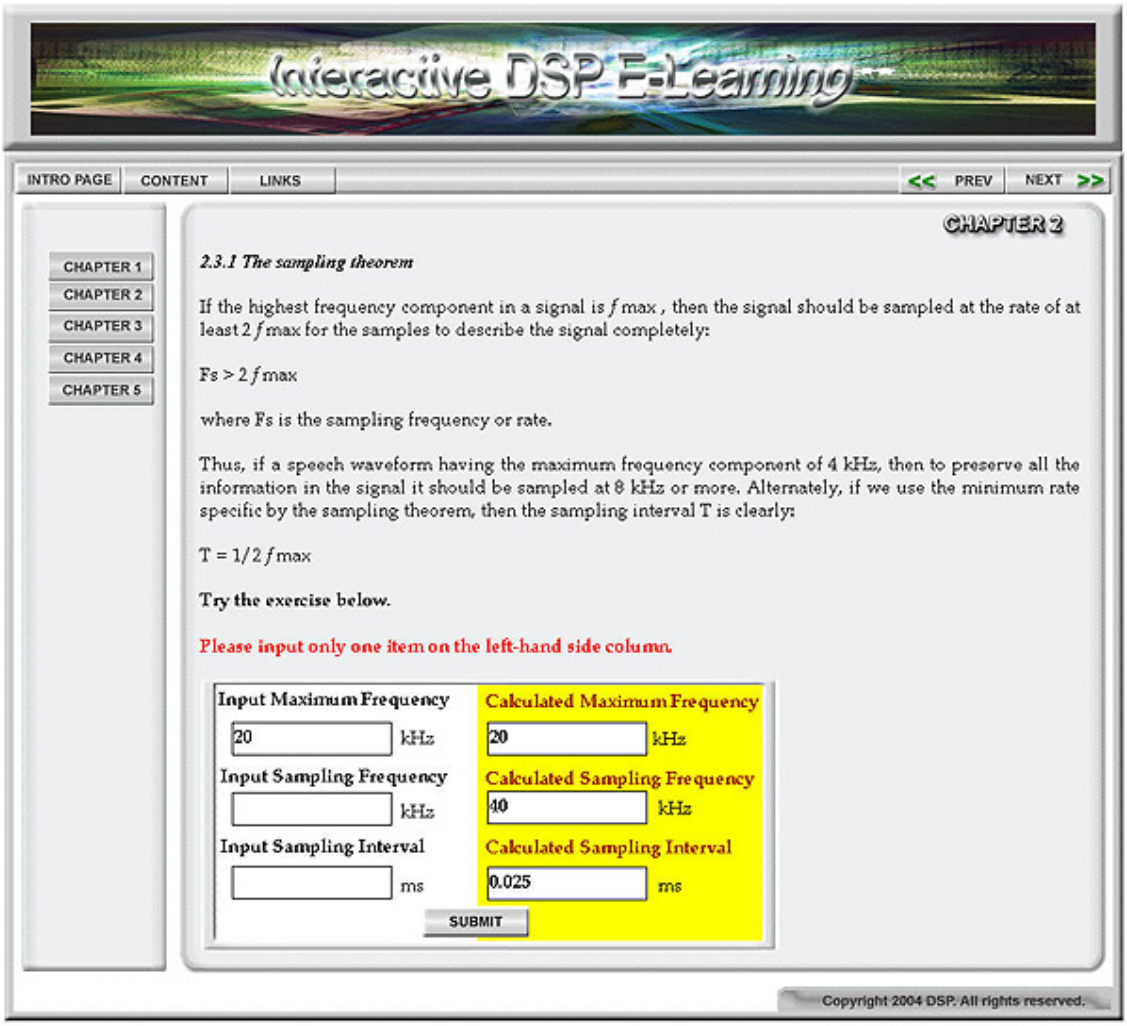

Figure 5.10: Screenshots of Chapter 2, interactive exercise for users to play around with.

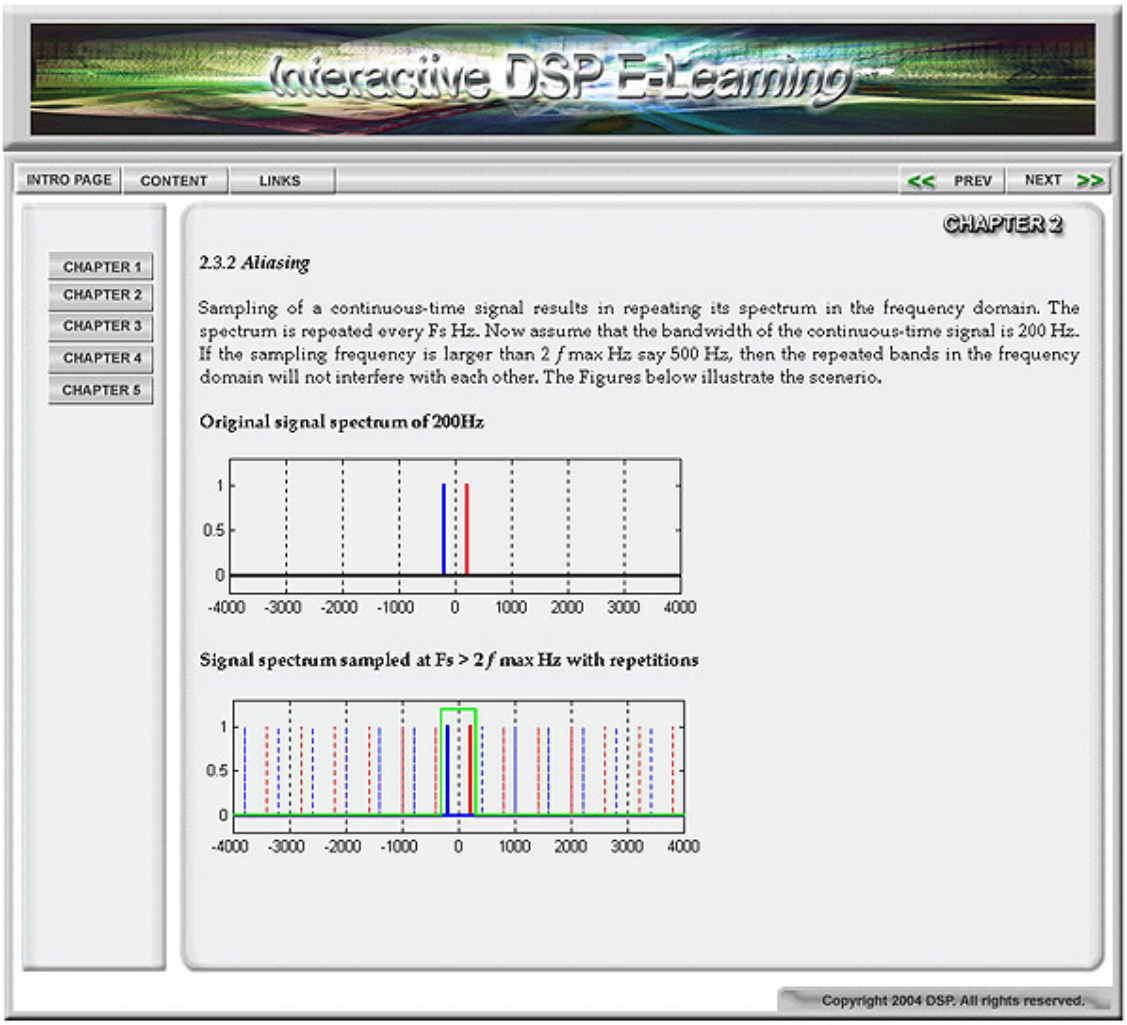

Figure 5.11: Screenshots of Chapter 2, static page illustrating topic on aliasing.

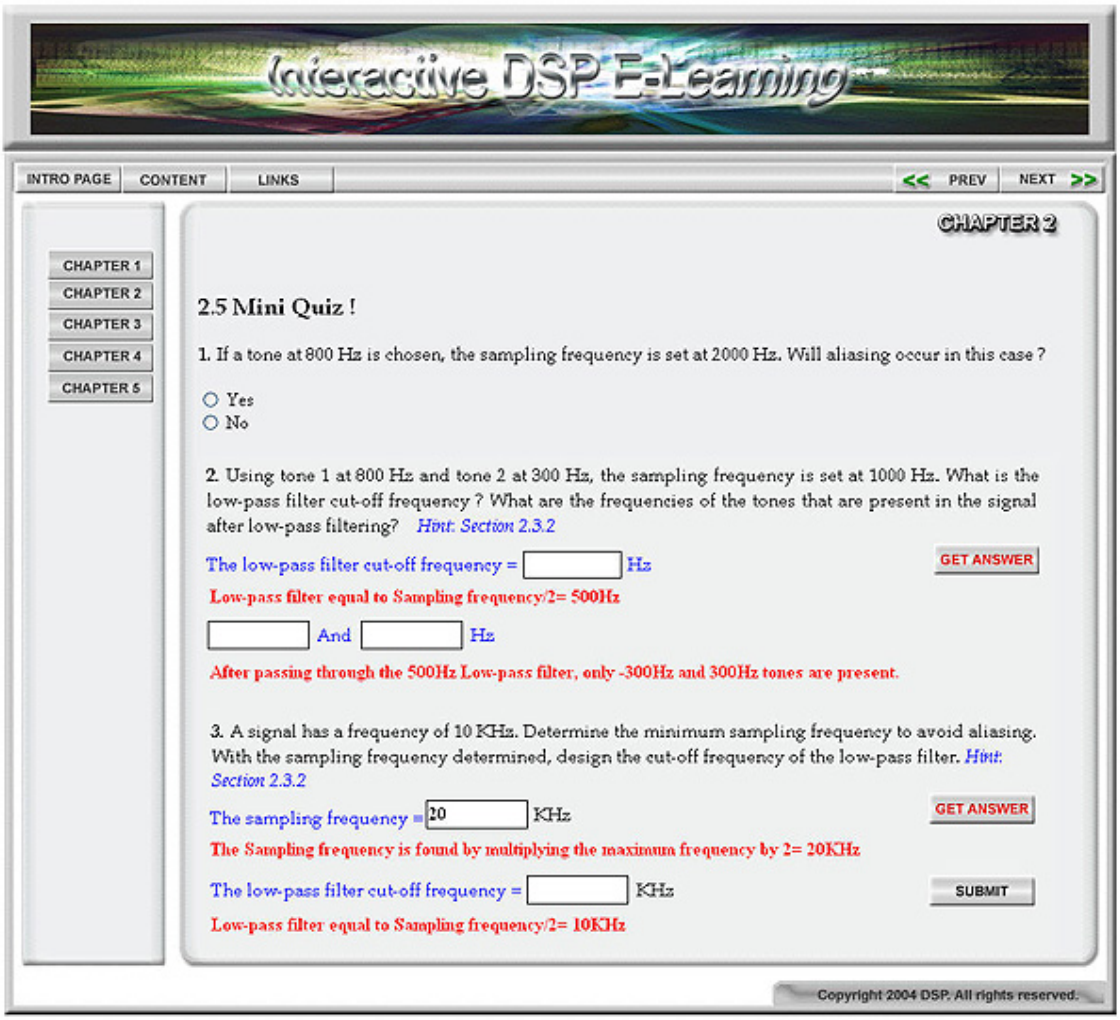

Figure 5.12: Screenshots of Chapter 2, mini-quiz for users to attempt with explained solutions provided.

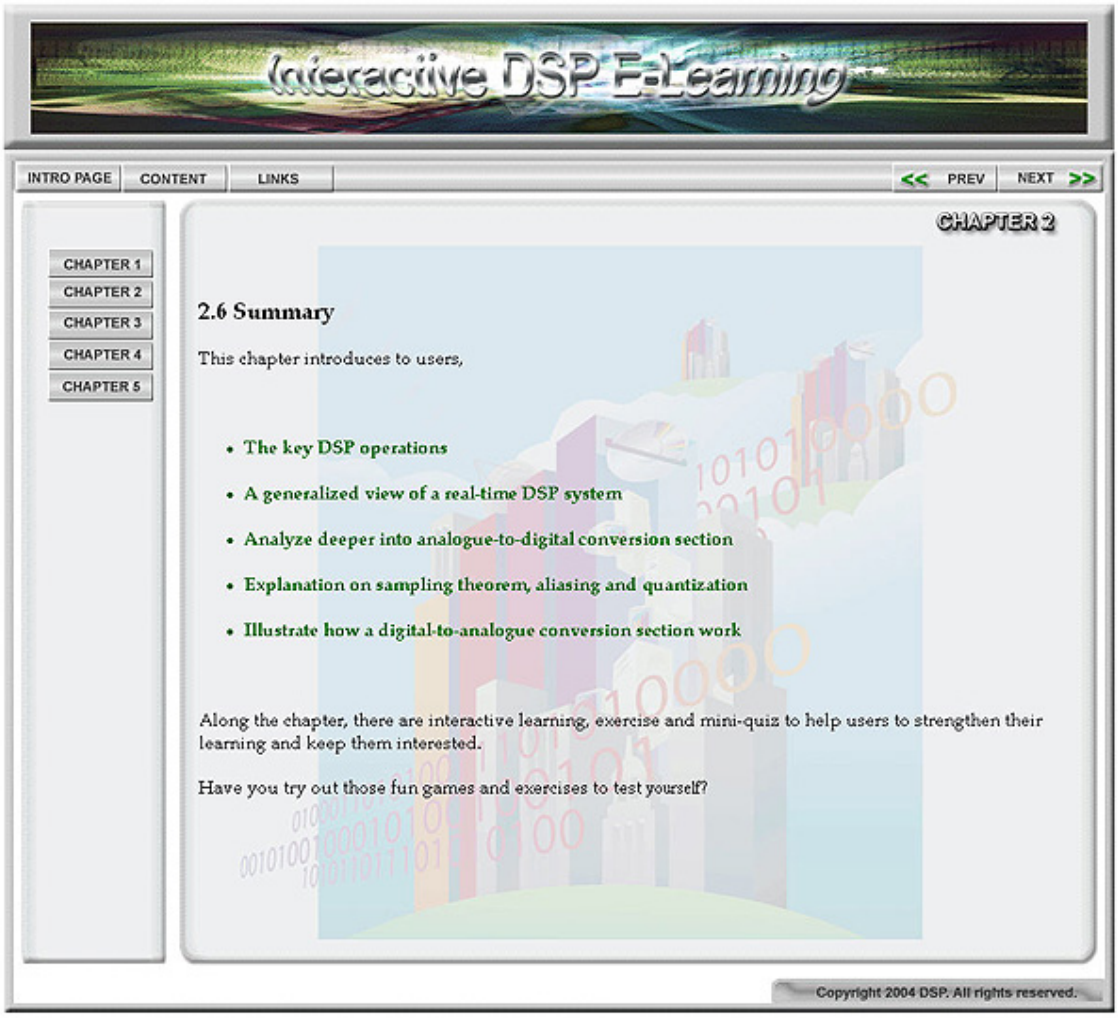

Figure 5.13: Screenshots of Chapter 2, summarize the objectives of the chapter.

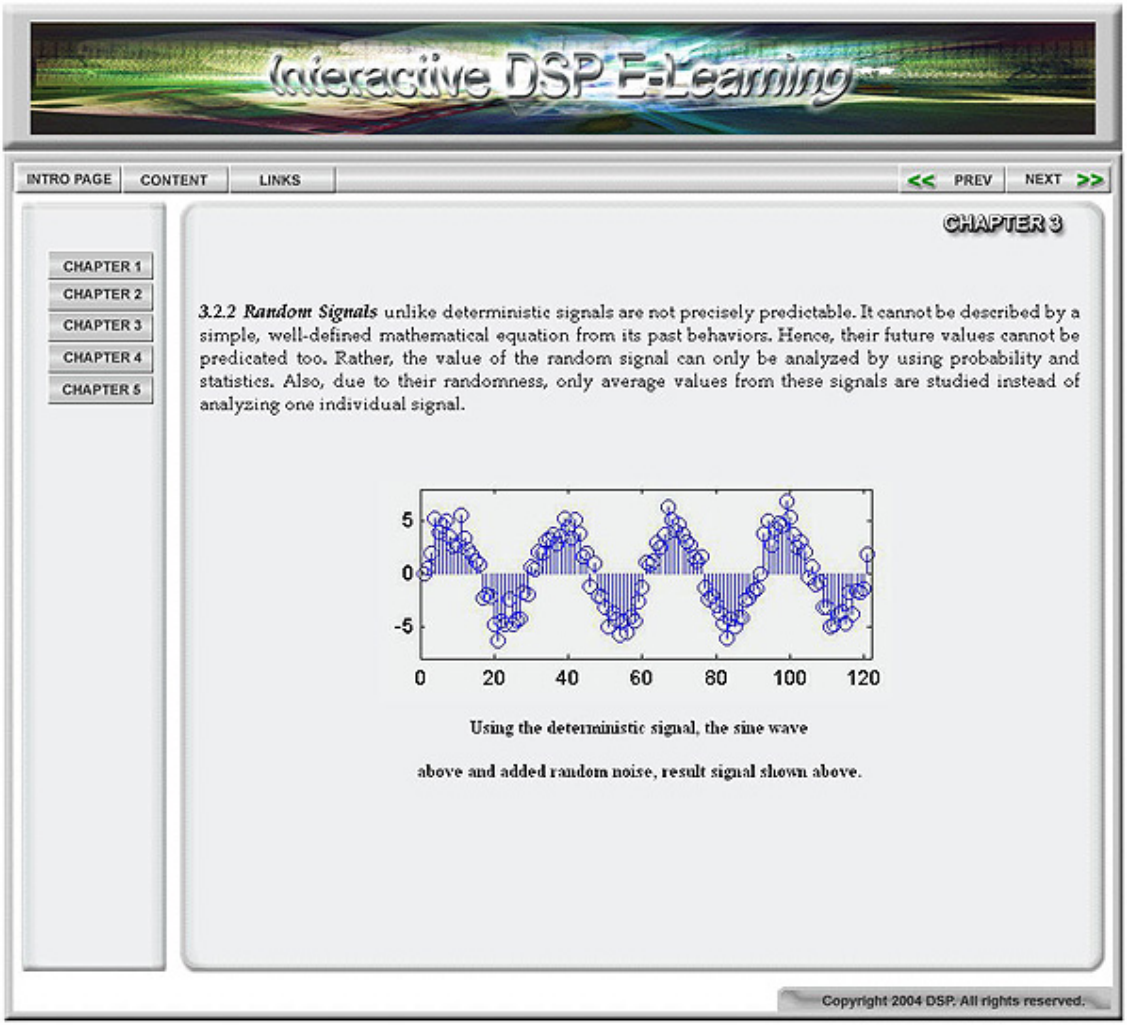

Figure 5.14: Screenshots of Chapter 3, static page on Random Signals.

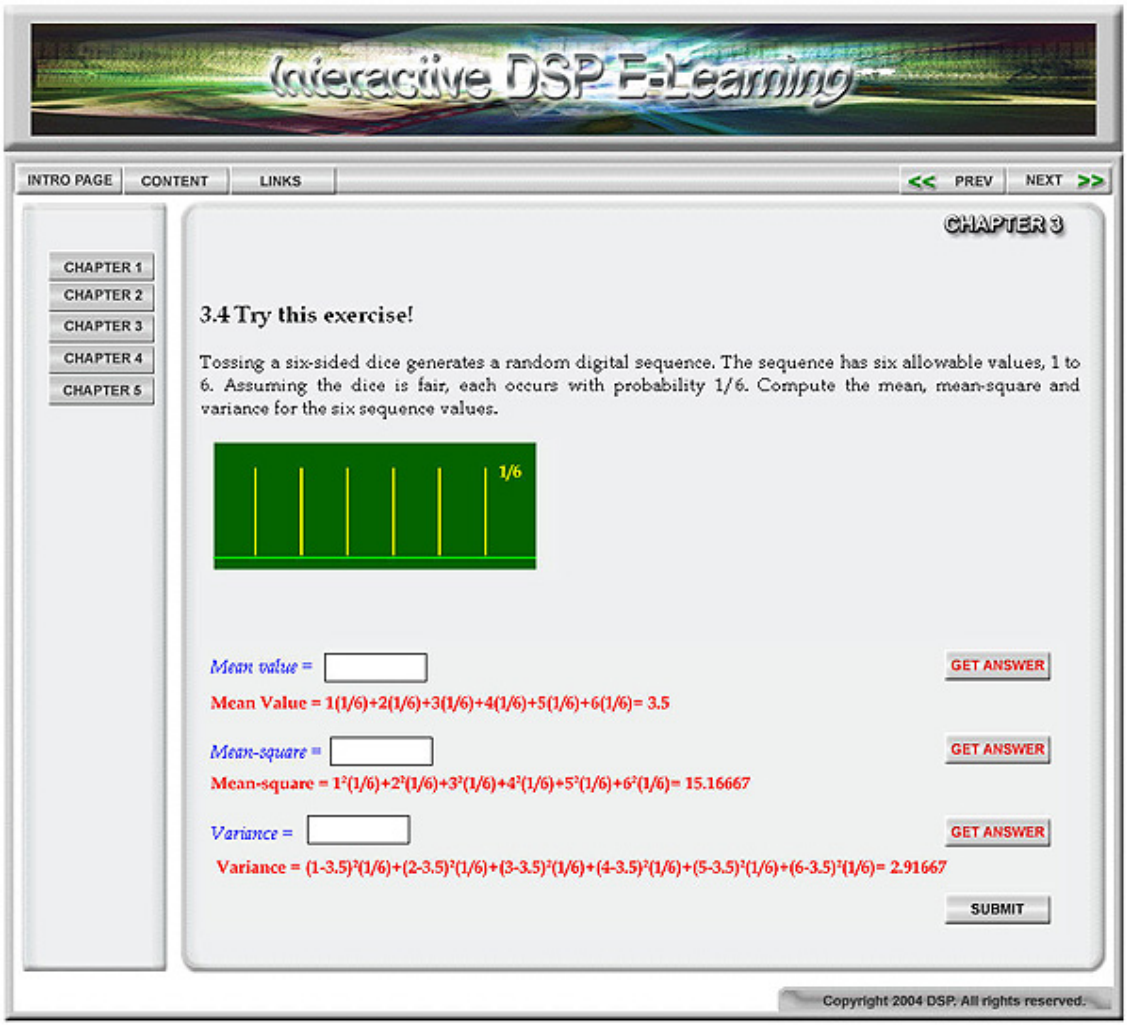

Figure 5.15: Screenshots of Chapter 3, interactive exercise.

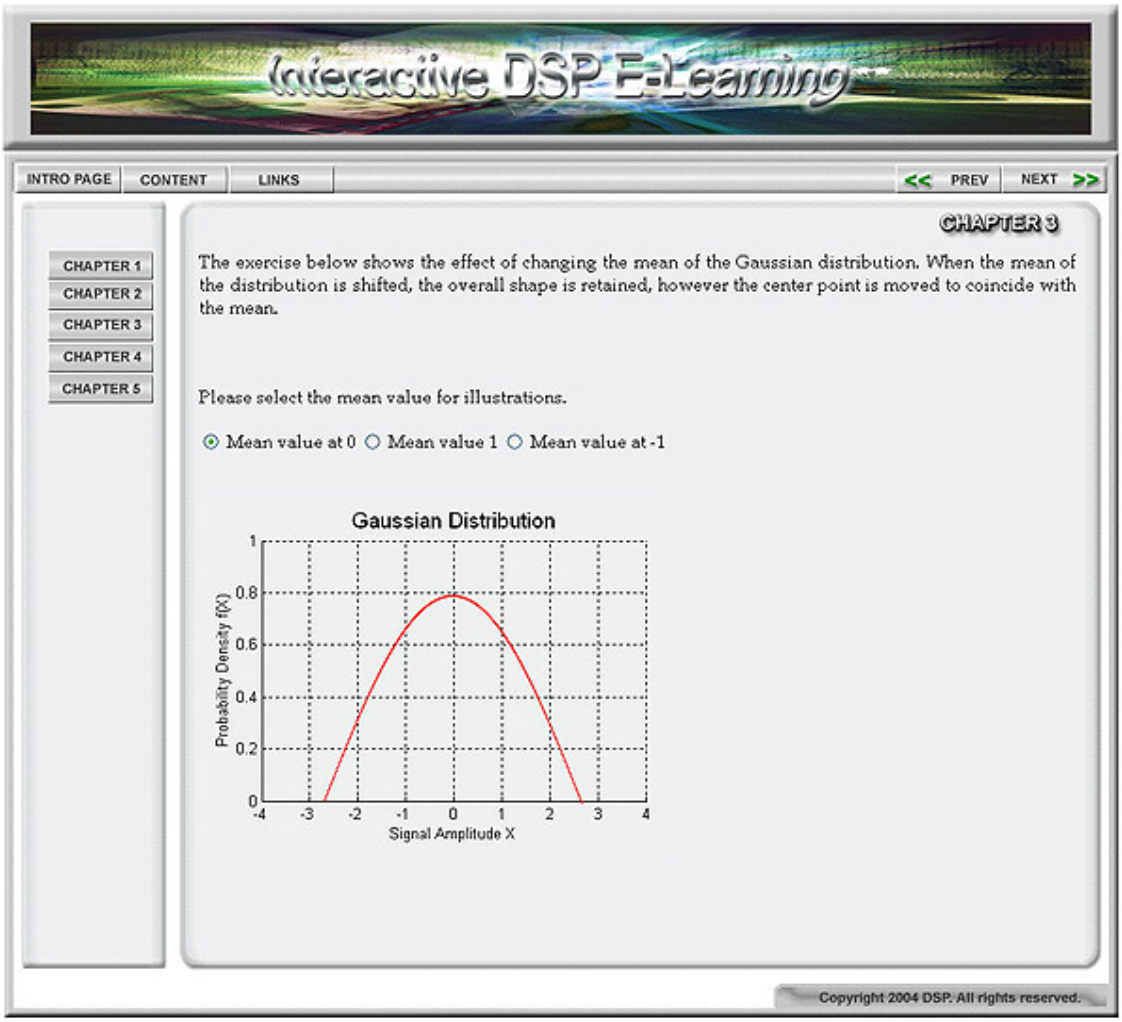

Figure 5.16: Screenshots of Chapter 3, interactive illustrations on Gaussian Distribution.

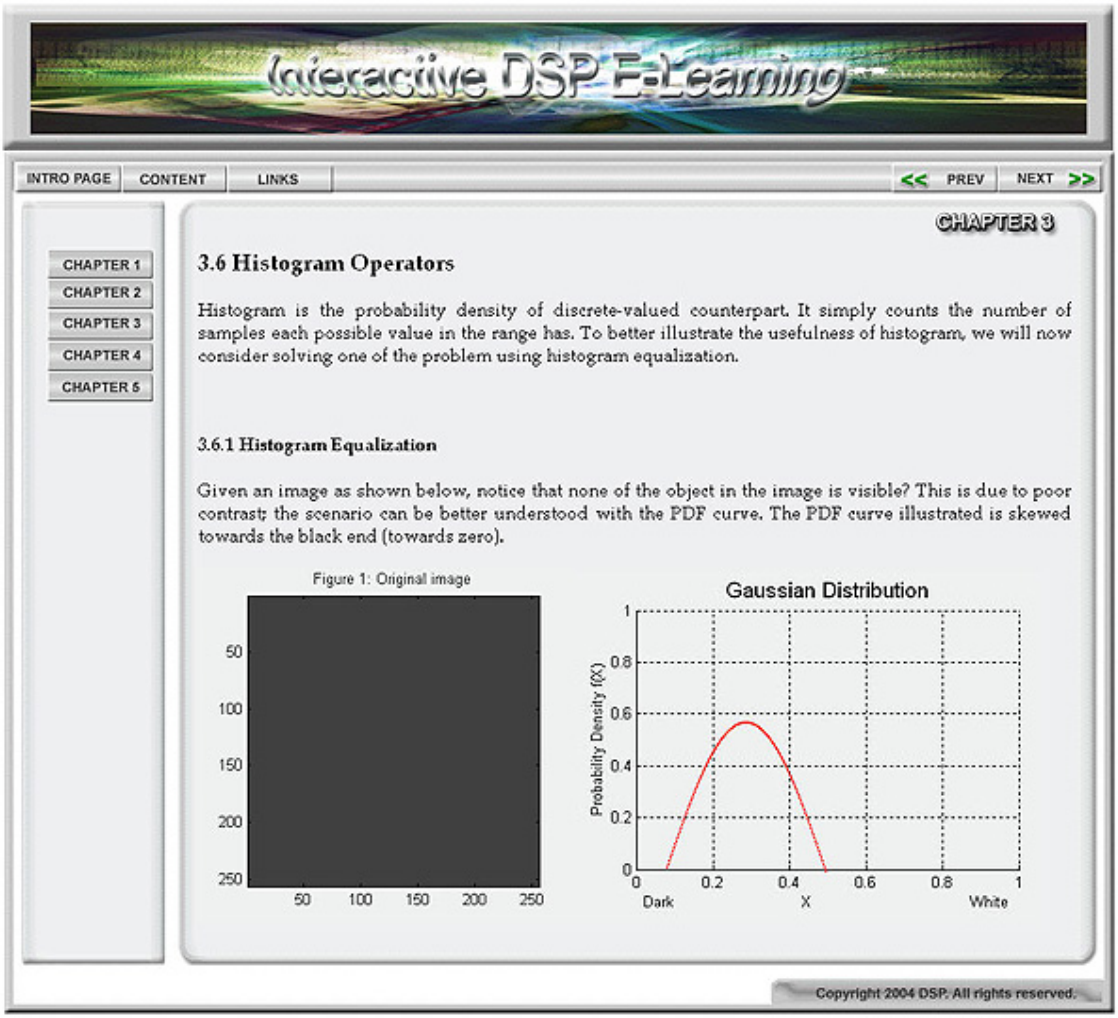

Figure 5.17: Screenshots of Chapter 3, realistic illustrated for users to visualize.

5.3 Chapter Summary

This chapter covered in great details on the testing methodology. A drawn up flowchart on the standard testing was provided, which helped to regulate the procedure. Following the procedures, the steps taken to conduct the testing were described in details. As the testing was conducted throughout the development phase too, problems were resolved at early stage and hence no major problem was faced during the testing phase. Helpful and creative feedbacks and suggestions from friends were taken into considerations and improvements on the project were made. Multiple screenshots were provided to give a full view on the final results of the project. However, the project's integrated animated and interactive functions can only be seen by visiting the website at `www.geocities.com/skylark7831/`.

Chapter 6

Further Work and Conclusion

6.1 Problem Encountered

The major problem faced in developing this project was the handling of totally unfamiliar software tools, namely, Macromedia Dreamweaver, Macromedia Flash MX and ActionScript programming. With no knowledge on any of them, a lot of time was taken to research and learn how to use them. In order to save some time, most of the programming and knowledge on these new tools were learnt through trial and error rather than by reading entire manuals.

Although there was some prior experience using Adobe Photoshop, I am still not a professional graphics designer and the design of the website may look amateurish. However, plenty of effort was put in this section and the results were generally impressive enough.

Also due to time constraints and a lack of in depth knowledge, further enhancement ideas for the website were unable to be fully fulfilled such as converting the current quizzes and exercises into interactive games, and adding audio playback during demonstrations. More specialised DSP topics also were unable to be

produced. Therefore these enhancements will have to be left as future works.

6.2 Achievement of Project Objectives

The main tasks set as the objectives associated with the project have been addressed as follows:

Research for information on the current existing interactive websites. This task was essential to the completion of the project and has been fulfilled. The research and review of this information was discussed in section 3.2, where their features and the various implementation methods were explored.

Critically evaluate the information collected for the suitable software tools and select useful ones for developing the project. This task was essential to the completion of the project and has been satisfied. A brief introduction on the selected implementation methods was given. Also, the evaluated information was detailed in subsection 3.4.2.

Plan and design the detailed structure of the project by breaking it up into multiple stages. At the same time, decide the appropriate software to be used during each stage. This task was completed in section 3.5 with reference to Figure 3.2 flowchart.

Evaluate and select suitable topics to be used in the project, specifically those that are more useful for students and engineers in the industry. Then transform the content into interactive multimedia learning for users. This task is completed in subsection 3.4.1. A short summary was given on each topics selected. The transform of topics into interactive multimedia learning was illustrated across various sections of Chapter 4.

Conduct an in depth study of the selected topics, picking out good illustrations and to compose the content of the DSP topics in a way more easily understood by users. This task was essential to the completion of the project and has been met. The process of the construction of the content was explained in section 4.1.

Construct and program the project according to the detailed structure design to serve as a guideline to work towards. This was accomplished with a drawn up flowchart shown in Figure 3.1 and a brief description was given in section 3.1.

Seek feedback and critical evaluation of others for the project as necessary. Then make necessary amendments to improve on it. This task was accomplished with the help of others actively participated. A summary of the feedbacks and suggestions were documented in section 5.1.3.

Additional Considerations of the project as time permit such as:

Develop interactive quizzes and illustrations which can be included into the project to improve users' understanding of the topics and arouse their interest in learning. This task was completed successful with much effort involved. It fully detailed in section 4.3.

Include audio playback during the demonstrations to enhance the explanation to topics and give more detail. Unfortunately due to time constraint, the already formulated ideas were unable to carry out. Thus this task was unable to be accomplished in time.

Other Objectives were not part of the project aims but was considered accomplished were illustrated as followed.

- Show clear evidence of ability to research and self-study. Also to apply what is being learnt into applications.

- Show clear indications of the ability to articulate findings in a comprehensive manner in the form of a written dissertation.
- Show facts of ability to critically analyze the works of others.

6.3 Further Work

This section discusses further possible ways in which the project could be extended and enhanced. These discussions included in this section are as follows:

- In-depth DSP topics
- Interactive learning games
- Audio playback implementations
- User login and records tracking

6.3.1 In-depth DSP topics

The current DSP topics do not provide more in-depth study of DSP. Topics such as Convolution, Discrete Fourier Transform and etc, are important too. They are considered quite difficult topics to be fully understood by students especially. Hence, further study and offering of these topics help to built on the current topics. This will in turn extend the reach of the website to a wider target, i.e. the advance students and industry engineers

6.3.2 Interactive learning games

Presently, not many interactive learning games were implemented. Some feedback obtained from others that some users may not like to do the interactive exercises

and quizzes. Hence, a better choice was to try to convert the existing interactive exercises and quizzes into interactive learning games. This way, users may find it easier and more fun to learn them.

6.3.3 Audio Playback Implementations

Another great feature for enhancing the project is to implement audio playback during the demonstrations to augment the explanation to topic. This feature can be added in with the use of Macromedia Flash MX. However, the format of the audio file must be converted to Flash Video (FLV) files before it can be imported into Flash. The audio file can then be played by using ActionScript programming. Of course, other considerations need to be taken note such as preparing the explanation speech for the narrator, and ensuring the audio explanations are clear and interesting. Other issues like the development and testing process need to be handled, such as preparations for recording the speech and to manage the size of the audio file; the file can only be as big as dictated by web space constraints. Thus other sources need to be searched or an alternative would be to set up a web server.

6.3.4 User database and record tracking

Other interesting features to be further look into may be to implement a user database and records tracking. It works by storing the user's profile and progress in a database so that when they return to the page, they can resume where they left off. The system can store quiz results and possibly bookmark certain pages for easy reference.

This implementation may be tied to the school database, where students can log in, or simply by letting visitors register themselves to the website. This could prove to be a challenging task to accomplish too since some modifications need to

be made to the existing interface. Also additional scripting languages may need to be learnt so as to develop the features listed above. Some of these languages include .NET, ASP, PHP. There would also be a need to run a web server and database to store all the users' records.

6.4 Conclusion

To reiterate, the project's objectives to develop a web tool in the form of an interactive website to assist students and industry engineers to learn and understand the fundamentals of DSP are achieved through this project. A website on the learning of DSP with the help of interactive functions was created with the knowledge that has been acquired. The basics of the implementation methods and DSP theory used during the developing of this project are fully furnished in this dissertation and can be serve as a reference for any interested parties to promptly acquire the structure and components of this project. In additional, several opportunities for further work have been noted.

Personally, several skills and valuable knowledge were acquired in the duration of this project. It has strengthened my problem-solving and management skills as well as provided exposure to a multimedia environment, and attained the skills to learn useful and fascinating software tools, in particular Macromedia Flash. My knowledge on DSP concepts has improved and the skills and knowledge that I have obtained from this project will certainly come in helpful in my future career.

References

Digital Signal Processing (current Feb 2004).
`http://www.dsptutor.freeuk.com/intro.htm`.

Digital Signal Processing - An Introduction to Digital Signal Processors (current Feb 2004).
`http://www.analog.com/Analog_Root/static/technology/dsp/beginnersGuide/introduction.html`.

Dr.Taft (current June 2004), *DSP Design Performance*.
`http://www.nauticom.net/www/jdtaft/`.

Fisher, T. (current June 2004), *Interactive Digital Filter Design*.
`http://www-users.cs.york.ac.uk/~fisher/mkfilter/`.

Habdus, M. (current March 2004), *Web Page Development Dreamweaver/FrontPage*.
`http://www.computerschool.net/courses/faq.html`.

Homepage of Wikipedia, t. f. E. (current Feb 2004), *Digital Signal Processing*.
`http://en.wikipedia.org/wiki/Digital_signal_processing`.

Kerman, P. (2002), *Sams teach yourself Macromedia Flash MX in 24 hours*, Indianapolis, Ind.

Leis, J. (1996), *Digital Signal Processing - A MATLANB - Based Tutorial Approach*, Institute of Physics, USA.

Lott, J. (2003), *ActionScript Cookbook*, O'Reilly Associates, United States of America.

Lowery, J. (2003), *Dreamweaver MX 2004 killer tips*, New Riders.

Lowery, J. (2004), *Macromedia Dreamweaver MX 2004 : Web application recipes*, New Riders.

Lynn, P. A. (1997), *Introductory Digital Signal Processing with computer Applications*, John Wiley, New York.

Lyons, R. G. (1997), *Understanding digital signal processing*, Addison-Wesley Longman, Inc.

of ActionScript.org, H. (current Feb 2004), *ActionScript*. `http://www.actionscript.org/tutorials.shtml`.

of Bores Signal Processing, H. (current June 2004), *Introducation to DSP*. `http://www.bores.com/courses/intro/index.htm`.

of eDEVcafe, H. (current Feb 2004), *Dynamic Line Generation and Distance Calculation with Macromedia Flash 5*. `http://www.edevcafe.com/viewdoc.php?eid=147`.

of Macromedia, H. (current Feb 2004), *Introduction to Macromedia Flash MX Drawing Methods*. `http://www.macromedia.com/devnet/mx/flash/articles/precision_drawing.html##beginfill`.

of Microsoft Windows, H. (current March 2004), *Internet Explorer 6 SP1 System Requirements*. `http://www.microsoft.com/windows/ie/evaluation/sysreqs/default.mspx`.

of Mozilla, H. (current March 2004), *Mozilla System Requirements*. `http://www.mozilla.org/products/mozilla1.x/sysreq.html`.

of Netsecape network, H. (current March 2004), *Netscape 7.2 System Requirements*.
`http://channels.netscape.com/ns/browsers/sysreq.jsp`.

of WebEng, H. (current March 2004), *Offering Web-based DSP education*.
`http://www.webeng.org/`.

of Webmaster, H. (current March 2004), *Difference between Macromedia Flash MX and Microsoft Frontpage*.
`http://www.webmasterworld.com/forum46/451-2-10.htm`.

Rosenzweig, G. (2003), *Macromedia Flash MX ActionScript for fun games*, Indianapolis, Ind.

Rugh, W. J. (current Feb 2004), *Signals Systems Control*.
`http://www.eas.asu.edu/~midle/jdsp/jdsp.html`.

Spanias, P. A. (current Feb 2004), *JAVA Digital Signal Processing*.
`http://www.eas.asu.edu/~midle/jdsp/jdsp.html`.

W, I. C. . J. (1996), *Digital Signal Processing - A Practical Approach*, Addison-Wesley, Singapore.

Appendix A

Project Specification

University of Southern Queensland

FACULTY OF ENGINEERING AND SURVEYING

ENG 41111/4112 Research Project PROJECT SPECIFICATION

FOR: TOH PEI LING

TOPIC: WEB-BASED SIGNAL PROCESSING
DEMONSTRATOR/INTERACTIVE TUTOR

SUPERVISORS: Dr. John Leis

PROJECT AIM: This project aims to provide a web-based interactive learning tool with selected topics in signal processing. It serves as an easy reference for students and industry engineers that can be easily accessed from the Internet in order to allow users to learn at their own pace.

PROGRAMME: Issue A, 22nd March 2004

1. Research information on the current existing interactive websites and available software tools that are suitable for developing the project.

2. Critically evaluate and select alternatives software tools for developing the interactivity of the project.

3. Plan and design the detailed structures of the project with the respective selected software tools.

4. Consider and choose the suitable content (i) selected topics of DSP, (ii) moving illustrations for DSP students, and engineers in industry.

5. Study and understand the context of the DSP selected topics, examples for illustrations and write out the content.

6. Construct and program according to the detailed structure designed for parts (i) and (ii) above.

7. Seek for feedback and evaluate on the project.

As time permits:

8. Include interactive quizzes and illustrations to improve better understanding of topics.

9. Include audio playback to the demonstrations to improve interactive performance.

AGREED: TOH PEI LING(student) DR. JOHN LEIS(Supervisors)

(Dated) 20 / 03 /04 (Dated) 31 / 03 / 04

Appendix B

How To View the Website Locally

This section provides an idea the way the various classes are organized in the CD and how to view the website locally without the need to go through the internet.

B.1 How the files in the CD are organized

The CDs contain one pdf file and one PROJECT folder.

PROJECT folder contains a subfolder that stored all the files that are required to run the website.

The peilingTOH-2004.pdf file contains this dissertation.

B.2 PROJECT

Notice that the subfolder inside was not named according to 'peilingTOH'. This is because in order to view the website all the folders and files naming must be the same as the original. Hence, these folders were used with it original naming.

Inside this PROJECT folder contains one subfolder and one index.htm.

FYPLINK folder contains another three folders. Namely, Layouts, Nagivation-Buttons and Screen.

index.htm is the main page of the website.

B.3 FYPLINK

Layouts folder contains all the graphical layout files used by the website.

NagivationButtons folder contains another three folders which are Normal, down and over. Each of these folders contain the graphic nagivation buttons that are meant for mouse 'Normal' look, 'down' look and 'over' look respectively.

Screen folder contains another two folders and all the .htm files created for the website. The two folders are Flash and Graphics.

Flash contains all the files created using Flash which mean all the animated graphics, exercises and quizzes, and interactive learning demonstrations were stored in this folder.

Graphics folder contains all the static graphics used for the website.

The .htm files are all the individual screens created for the website.

B.4 How to view the Website

Basically, you can choose to copy the whole peilingTOH-PROJECT folder from the CD to a local hard disk as this may speed up the loading process while viewing the web pages. However, running from the CD itself is also feasible.

To view, simply open up the PROJECT folder and double click on the index.htm file. This bring you directly to the main page of the website and the viewing can start. To choose to view a particular screen directly, you can go to the screen folder and select the wanted screen, provided you know what is the name of that screen. Hence, it advisable to just navigate around the website itself as it easy to move around in there.

Appendix C

Programming Source Code

C.1 Programming source code for generating animated sine wave (sinewave.fla)

```
// set up constants specific to this movie layout
var Wavestart = 21;    // start point for sine wave

var Wavestop=  380;   // stop point for sine wave

Waveamp = 100; //y=0 point (midpoint of wave height). Amplitude
of the waveform

// angle changes one radian each frame
var angle = 0;
// set up 360 dots of the sine wave in initial position

for (var a=0; a<360; a++) {
    angle -= Math.PI/180*2;
    this.attachMovie("dot", "dot"+a, a, {_x:Wavestart+a, _y:Waveamp
    - Math.sin(angle)*50});
    //my_mc.attachMovie(idName, newName, depth [,initObject])
    }
    this.onEnterFrame = function(){
        for (var a=0; a<360; a++)
        {
            this["dot"+a]._x = (this["dot"+a]._x >= Wavestop)
            ? Wavestart : this["dot"+a]._x+
        }angle += Math.PI/180*2; };
```

C.2 Programming source code for generating discrete signal (discrete.fla)

```
//Section 1


_root.attachMovie("Line", "h1",1);
_root.attachMovie("Line1","h2", 2);
_root.attachMovie("Line2","h3", 3); _root.attachMovie("Line 3",
"h4", 4); _root.attachMovie("Line 3","h5",5);
_root.attachMovie("Line 2", "h6", 6); _root.attachMovie("Line 1",
"h7", 7); _root.attachMovie("Line", "h8", 8);


h1._x = 40;
h1._y = 98;
h2._x = 50;
h2._y = 86;
h3._x = 60;
h3._y = 75;
h4._x = 70;
h4._y = 58;
h5._x = 80;
h5._y = 58;
h6._x = 90;
h6._y = 75;
h7._x = 100;
h7._y = 86;
h8._x = 110;
h8._y = 98;
```

```
//Section 2
_root.attachMovie("Line", "hA",9);
_root.attachMovie("Line 1",
"hB", 10);
_root.attachMovie("Line 2", "hC", 11);
_root.attachMovie("Line 3", "hD", 12);
_root.attachMovie("Line
3", "hE",13);
_root.attachMovie("Line 2", "hF", 14);
_root.attachMovie("Line 1", "hG", 15);
_root.attachMovie("Line",
"hH", 16);

hA._x = 120;
hA._y = 134;
hB._x = 130;
hB._y = 134;
hC._x = 140;
hC._y = 134;
hD._x = 150;
hD._y = 134;
hE._x = 160;
hE._y = 134;
hF._x = 170;
hF._y = 134;
hG._x = 180;
hG._y = 134;
hH._x = 190;
hH._y = 134;
```

```
//Section 3
_root.attachMovie("Line", "h9",17);
_root.attachMovie("Line 1",
"h10", 18);
_root.attachMovie("Line 2", "h11", 19);
_root.attachMovie("Line 3", "h12", 20);
_root.attachMovie("Line
3", "h13",21);
_root.attachMovie("Line 2", "h14", 22);
_root.attachMovie("Line 1", "h15", 23);
_root.attachMovie("Line",
"h16", 24);

h9._x = 200;
h9._y = 98;
h10._x = 210;
h10._y = 86;
h11._x = 220;
h11._y = 75;
h12._x = 230;
h12._y = 58;
h13._x = 240;
h13._y = 58;
h14._x = 250;
h14._y = 75;
h15._x = 260;
h15._y = 86;
h16._x =
270;
h16._y = 98;
```

```
//Section 4
_root.attachMovie("Line", "h17",25);
_root.attachMovie("Line 1",
"h18", 26);
_root.attachMovie("Line 2", "h19", 27);
_root.attachMovie("Line 3", "h20", 28);
_root.attachMovie("Line
3", "h21",29);
_root.attachMovie("Line 2", "h22", 30);
_root.attachMovie("Line 1", "h23", 31);
_root.attachMovie("Line",
"h24", 32);

h17._x = 280;
h17._y = 134;
h18._x = 290;
h18._y = 134;
h19._x =300;
h19._y = 134;
h20._x = 310;
h20._y = 134;
h21._x = 320;
h21._y = 134;
h22._x = 330;
h22._y = 134;
h23._x = 340;

h23._y = 134;
h24._x = 350;
h24._y = 134;
```

C.3 Programming source code for ADC block (chap2matchgame.fla)

```
/* For ADC block.*/ /*On the event when the game is loading, the
original position (x, y) of ADC are stored into variables of
this._x and this._y, and var noDrag = 0. */

onClipEvent (load) {
    origX = this._x;
    origY = this._y;
    noDrag = 0;
} /* On the event of mouse click on ADC block and recognise that
noDrag = 0, allows dragging function to start*/

onClipEvent (mouseDown){
    if ((this.hitTest(_root._xmouse, _root._ymouse)) and (noDrag == 0))
           {
        this.startDrag();
            }
}

/*On the event of mouse up from the block, dragging function
stop. Reading on current mouse release position (x, y). Do a
condition check, if current position is same as the designation
position which is _parent.dropZone2._x and _parent.dropZone2._y,
the ADC block will stay at the yellow box and variable noDrag = 1
which disable the dragging function. If the ADC position is not
the same as _parent.dropZone2._x and _parent.dropZone2._y, the
variables this._x and this._y will assigned back to their
original position.*/
```

```
onClipEvent (mouseUp) {
    if (this.hitTest(_root._xmouse, _root._ymouse)) {
        this.stopDrag();

        if (_parent.dropZone2.hitTest(this._x, this._y, true)) {
            this._x = _parent.dropZone2._x;
            this._y = _parent.dropZone2._y;
            noDrag = 1;
            }
                        else {
            this._x = origX;
            this._y = origY;
        }
    }
}
```

C.4 Programming source code for Analog filter block (chap2matchgame.fla)

```
/* For Analog filter block.*/ /*On the event when the game is
loading, the original position (x, y) of Analog filter are stored
into variables of this._x and this._y, and var noDrag = 0. */

onClipEvent (load) {
    origX = this._x;
    origY = this._y;
    noDrag = 0;
}
/* On the event of mouse click on the block and recognise that
noDrag = 0, allows dragging function to start*/

onClipEvent (mouseDown) {
    if ((this.hitTest(_root._xmouse, _root._ymouse)) and (noDrag == 0)) {
        this.startDrag();
    }
}

/*On the event of mouse up from the block, dragging function
stop. Reading on current mouse release position (x, y). Do a
condition check, if current position is same as the designation
position which is _parent.dropZone1._x and _parent.dropZone1._y,
or _parent.dropZone5._x and _parent.dropZone5._y the Analog
filter block will stay at the yellow box and variable noDrag = 1
which disable the dragging function. If the block position is not
the same as either any of the above, the variables this._x and
this._y will assigned back to their original position.*/
```

```
onClipEvent (mouseUp) {
    if (this.hitTest(_root._xmouse, _root._ymouse)) {
        this.stopDrag();

        if (_parent.dropZone1.hitTest(this._x, this._y, true)) {

            this._x = _parent.dropZone1._x;
            this._y = _parent.dropZone1._y;
            noDrag = 1;
            }

        else if (_parent.dropZone5.hitTest(this._x, this._y, true)) {

            this._x = _parent.dropZone5._x;
            this._y = _parent.dropZone5._y;
            noDrag = 1;
            }

            else {

            this._x = origX;
            this._y = origY;

        }
    }
}
```

C.5 Programming source code for DAC block (chap2matchgame.fla)

```
onClipEvent (load) {
    origX = this._x;
    origY = this._y;
    noDrag = 0;
}

onClipEvent (mouseDown) {
    if ((this.hitTest(_root._xmouse, _root._ymouse)) and (noDrag==0)) {
        this.startDrag();
    }
} onClipEvent (mouseUp) {
    if (this.hitTest(_root._xmouse, _root._ymouse)) {
        this.stopDrag();

        if (_parent.dropZone4.hitTest(this._x, this._y, true)) {

            this._x = _parent.dropZone4._x;
            this._y = _parent.dropZone4._y;
            noDrag = 1;

        } else {

            this._x = origX;
            this._y = origY;
        }
    }
}
```

C.6 Programming source code for Analog filter block (chap2matchgame.fla)

```
onClipEvent (load) {
    d1origX = this._x;
    d1origY = this._y;
    noDrag = 0;
}
onClipEvent (mouseDown) {
    if ((this.hitTest(_root._xmouse, _root._ymouse)) and (noDrag == 0)) {
        this.startDrag();    }
}
onClipEvent (mouseUp) {
    if (this.hitTest(_root._xmouse, _root._ymouse)) {
        this.stopDrag();
        if (_parent.dropZone5.hitTest(this._x, this._y, true)) {
            this._x = _parent.dropZone5._x;
            this._y = _parent.dropZone5._y;
            noDrag = 1;}
        else if (_parent.dropZone1.hitTest(this._x, this._y, true)) {
            this._x = _parent.dropZone1._x;
            this._y = _parent.dropZone1._y;
            noDrag = 1;
            }
        else {
            this._x = d1origX;
            this._y = d1origY; }
    }
}
```

C.7 Programming source code for DSP filter block (chap2matchgame.fla)

```
onClipEvent (load) {
    origX = this._x;
    origY = this._y;
    noDrag = 0;
}

onClipEvent (mouseDown) {
    if ((this.hitTest(_root._xmouse, _root._ymouse))and (noDrag ==0)) {
        this.startDrag();
    }
} onClipEvent (mouseUp) {
    if (this.hitTest(_root._xmouse, _root._ymouse)) {
        this.stopDrag();
        if (_parent.dropZone3.hitTest(this._x, this._y, true)) {

            this._x = _parent.dropZone3._x;
            this._y = _parent.dropZone3._y;
            noDrag=1;
        } else {
            this._x = origX;
            this._y = origY;
        }
    }
}
```

C.8 Programming source code for submit button (chap2matchgame.fla)

```
/* For submit button.*/ /*When the submit button pressed and
released. It stop the timing and get the timing, then print the
timing on to 'mess' and the word "secs" onto 'secs'*/ on
(release) {
    Time=getTimer();
    mess=Time/1000;
    sec="secs"
}
```

C.9 Programming source code for clickable blocks (chap2DSPm.fla)

```
/* For Analog input block*/ /* On the event the mouse press onto
this block, jump to both Frame 'Input' and Frame 'Ainput', play
these frames only. Frame 'Input' was the animated sine wave and
Frame 'Ainput' was the explanation for the block. Once mouse
press on it, it played both Frames.*/

on(press) {
    gotoAndStop("Input");
    gotoAndStop("Ainput");
}



/* For Analog filter block*/
on(press) {
    gotoAndStop("AF explain");
}



/* For ADC block*/
 on(press) {
    gotoAndStop("ADC");
}



/* For DSP block*/
on(press) {
    gotoAndStop("DSP");
```

```
    gotoAndStop("DSPsignal");
}



/* For DAC block*/
on(press) {
    gotoAndStop("DAC");
}



/* For DAC block*/
on(press) {
    gotoAndStop("AF2");
}



/* For Analog filter block*/ on(press) {
    gotoAndStop("Output");
    gotoAndStop("Aoutput");
}
```

C.10 Programming source code for illustrations of input signals (chap2DSPm.fla)

```
// set up constants specific to this movie layout
var Wavestart = 10;    // start point for sine wave

var Wavestop=  189;   // stop point for sine wave

var Waveamp = 50;        // y=0 point (midpoint of wave height).
Amplitude of the waveform

// angle changes one radian each frame
var angle = 0;

// set up 360 dots of the sine wave in initial position
for (var a=0; a<180; a++) {
    angle -= Math.PI/60*2;  //Changing "60" number will increase or
    decrease the number of waveforms. (change in set of 0, 60, 120, 180, etc)
    this.attachMovie("dot", "dot"+a, a, {_x:Wavestart+a, _y:Waveamp
    - Math.sin(angle)*20});

    //my_mc.attachMovie(idName, newName, depth [, initObject])
} this.onEnterFrame = function() {
    for (var a=0; a<180; a++) {
    this["dot"+a]._x = (this["dot"+a]._x >= Wavestop) ?
    Wavestart : this["dot"+a]._x + 1;
    }
    angle += Math.PI/60*2;
};
```

C.11 Programming source code for illustrations of DSP signals (chap2DSPm.fla)

```
_root.attachMovie("Line", "h1",1);
_root.attachMovie("Line1","h2", 2);
_root.attachMovie("Line2","h3", 3);
_root.attachMovie("Line3","h4", 4);
_root.attachMovie("Line3","h5",5);
_root.attachMovie("Line2","h6", 6);
_root.attachMovie("Line1","h7", 7);
_root.attachMovie("Line","h8", 8);


h1._x = 640/2;
h1._y = 98;
h2._x = 650/2;
h2._y = 86;


h3._x = 660/2;
h3._y = 75;
h4._x = 670/2;
h4._y = 58;


h5._x = 680/2;
h5._y = 58;
h6._x = 690/2;
h6._y = 75;


h7._x = 700/2;
h7._y = 86;
h8._x = 710/2;
```

```
h8._y = 98;



_root.attachMovie("Line", "hA",9);
_root.attachMovie("Line1","hB", 10);
_root.attachMovie("Line2","hC", 11);
_root.attachMovie("Line3","hD", 12);
_root.attachMovie("Line3","hE",13);
_root.attachMovie("Line2","hF", 14);
_root.attachMovie("Line1","hG", 15);
_root.attachMovie("Line","hH", 16);



hA._x = 720/2;
hA._y = 265/2;
hB._x = 730/2;
hB._y = 265/2;

hC._x = 740/2;
hC._y = 265/2;
hD._x = 750/2;
hD._y = 265/2;

hE._x = 760/2;
hE._y = 265/2;
hF._x = 770/2;
hF._y = 265/2;

hG._x = 780/2;
hG._y = 265/2;
hH._x = 790/2;
```

```
hH._y = 265/2;


_root.attachMovie("Line", "h9",17);
_root.attachMovie("Line1","h10",18);
_root.attachMovie("Line2","h11", 19);
_root.attachMovie("Line3","h12", 20);
_root.attachMovie("Line3","h13",21);
_root.attachMovie("Line2","h14", 22);
_root.attachMovie("Line1","h15", 23);
_root.attachMovie("Line","h16", 24);


h9._x = 700/2;
h9._y = 98;
h10._x = 810/2;
h10._y = 86;

h11._x = 820/2;
h11._y = 75;
h12._x = 830/2;
h12._y = 58;

h13._x = 840/2;
h13._y = 58;
h14._x = 850/2;
h14._y = 75;

h15._x = 860/2;
h15._y = 86;
h16._x = 870/2;
h16._y = 98;
```

```
_root.attachMovie("Line", "h17",25);
_root.attachMovie("Line1","h18",26);
_root.attachMovie("Line2","h19", 27);
_root.attachMovie("Line3","h20", 28);
_root.attachMovie("Line3","h21",29);
_root.attachMovie("Line2","h22", 30);
_root.attachMovie("Line1","h23", 31);
_root.attachMovie("Line","h24", 32);


h17._x = 880/2;
h17._y = 265/2;
h18._x = 890/2;
h18._y = 265/2;
h19._x = 900/2;
h19._y = 265/2;
h20._x = 910/2;
h20._y = 265/2;
h21._x = 920/2;
h21._y = 265/2;
h22._x = 930/2;
h22._y = 265/2;
h23._x = 940/2;
h23._y = 265/2;
h24._x = 950/2;
h24._y = 265/2;
```

C.12 Programming source code for illustrations of Analog output signals (chap2DSPm.fla)

```
// set up constants specific to this movie layout
var Wavestart = 600;    // start point for sine wave

var Wavestop =  779;   // stop point for sine wave

var Waveamp = 50;     // y=0 point (midpoint of wave height).
Amplitude of the waveform

// angle changes one radian each frame
var angle = 0;

// set up 360 dots of the sine wave in initial position
for (var a=0; a<180; a++) {
    angle -= Math.PI/60*2;  //Changing "60" number will increase or decrease
    the number of waveforms. (change in set of 0, 60, 120, 180, etc)
    this.attachMovie("dot1", "dot1"+a, a, {_x:Wavestart+a, _y:Waveamp -
    Math.sin(angle)*20});
    //my_mc.attachMovie(idName, newName, depth [, initObject])
}
    this.onEnterFrame = function() {
    for (var a=0; a<180; a++) {
    this["dot1"+a]._x = (this["dot1"+a]._x >= Wavestop) ?
    Wavestart : this["dot1"+a]._x + 1;
    }
    angle += Math.PI/60*2;
};
```

C.13 Programming source code for the chapter 2 exercise (quiz1.fla)

```
/* For Chapter 2 exercise*/

/* On the event of mouse release, several conditions check were
done in order to perform the correct calculation of the
parameters. */

on (release) {
    if (outputA = a)
    {          outputB = a*2;
        outputC = 1/outputB;
    }
    else if (outputB = b)
    {          outputA = b*2;
        outputC = 1/b;
     }
     else
     {          outputC=c;
         outputB=1/c;
         outputA=outputB/2;
     }
}
}
```

C.14 Programming source code for the chapter 2 quizzes (quiz2.fla)

```
/* For Chapter 2 quizzes*/

//When mouse click is release on the "GET ANSWER" button
on (release)
    {
        /*Write the following message in the dynamic input box name
        message1 and message 2.*/

        message1 = "Low-pass filter equal to Sampling frequency/2= 500Hz ";
        message2 = "After passing through the 500Hz Low-pass filter,
        only -300Hz and 300Hz tones are present.";
     }



/* For Chapter 2 quizzes "SUBMIT BUTTON*/

/* On the event of mouse release, several conditions check are
performed. If the answer matches the correct ones, then correct
message would be printed out. Else, other message will printed
out on the correct boxes accordingly. */

on (release) {
    if (a == "no" || a == "No" || a == "NO"){
     message = "Correct!";
     }else {
         message = "Are you sure?";
     }
```

```
     if (b == "500"){
     message1 = "Correct!";
     }else {
         message1 = "Try again";
     }
    if (c == "-300" && d == "300" || c == "300" && d == "-300"){
     message2 = "Great work!";
     }else {
         message2 = "You can do better";
     }
      if (e == "20" || e == "20"){
     message3 = "Well Done!";
     }else {
         message3 = "WRONG!";
     }
if (f == "10" || f == "10"){
     message4 = "Fantastic!";
     }else {
         message4 = "Incorrect!";
     }
}
```

C.15 Programming source code for the chapter 3 exercise illustrations (quiz4a.fla)

```
// set up constants specific to this movie layout
var Wavestart = 21;    // start point


var Wavestop =  360;//stop point


var Waveamp = 55; // y=0 point (midpoint of wave height).


_root.attachMovie("Line 3", "h7",1);
_root.attachMovie("Line3","h8",2);
_root.attachMovie("Line3","h9",3);
_root.attachMovie("Line3","h10",4);
_root.attachMovie("Line3","h11",5);
_root.attachMovie("Line3","h12",6);


h7._x = 50;
h7._y = 60;
h8._x = 90;
h8._y = 60;
h9._x = 130;
h9._y = 60;
h10._x = 170;
h10._y = 60;
h11._x = 210;
h11._y = 60;
h12._x = 250;
h12._y = 60;
```

C.16 Programming source code for the chapter 3 exercise (quiz4.fla)

```
//For the "GET ANSWER" button
on (release) on (release) {
    quiz4aMC._alpha=100;
}

on (release) {
    quiz4bMC._alpha=100;
}

on (release) {
    quiz4cMC._alpha=100;
}



//For the "SUMBIT" button
on (release) {
    if (a == "3.5")
    {
     message = "Correct!";
    }
    else {
         message = "Are you sure?";
         }
    if (b == "15" || b == "15.2" || b == "15.17" || b == "15.167"
    || b == "15.1667" || b == "15.16667")
    {
     message1 = "Correct!";
```

```
    }
    else {
         message1 = "Wrong!";
         }
    if (c == "3" || c == "2.9" || c == "2.92" || c == "2.917"
    || c == "2.9167" || c == "2.91667")
    {
     message2 = "Brillant!";
    }
    else {
         message2 = "Try again.";
         }
}
```