BUILDING INTERACTIVE TUTORIALS USING VISUAL BASIC

Robert W. Nowlin, Qunying Gao, and Raji Sundararajan
Department of Electronics & Computer Engineering Technology
Arizona State University East - Mesa, AZ – 85212
raji@asu.edu

Abstract

In this computer information age, computers in education play a major role in effective learning. This paper presents the development and the aspects of a graphical, user interactive, Visual Basic tutorial, to learn VHDL via computers.

VHDL, a hardware description language for Very High Speed Integrated Circuits (VHSIC), is used to model digital systems. The digital system can be a simple logic gate or a complicated electronic system. VHDL is an IEEE, as well as an ANSI standard for describing digital designs.

The learning of VHDL can be made more effective by means of an interactive tutorial. Visual Basic (VB), the vehicle to the exciting world of Internet and World Wide Web programming, is used for this purpose. Being a powerful programming language, VB helps students visualize complicated circuits and concepts of VHDL. VB is used to design the graphical user interface and MS Access is used to store the questions and answers the students may need for their practice.

Introduction

Computers, computers, everywhere, used for everything - computers for calculation, computers for communication, computers for games, computers for shopping, and computers for education too. In this information age, use of computers in education is the modern trend of learning, which has expanded into Internet and web-based learning. Traditional and new classes are being taught using computers extensively. One such attempt is made here to learn VHDL effectively, using Visual basic and Access, each being the cutting-edge tool in their own discipline. VHDL is a hardware description language used to simulate, model and synthesize digital systems. Visual Basic is a high level object oriented program language used extensively in presenting information in a pleasing form. Combining the VB and Microsoft (MS) Access to interactively learn VHDL will benefit the students to learn more effectively the subject material.

The tutorial is divided into two modules. Module one, with three sections, deals with the brief introduction of VHDL, and how it works, major modeling features of the VHDL language and a review and summary of VHDL. Module two is the practice model, that lets students practice solving some basic problems of VHDL.

On each page, there are several buttons which students click to choose different topics. It will also support some graphics, diagrams or flowcharts for various topics to help students better understand VHDL. On the practice page the student can choose the questions they need to practice. Each question page will contain a field for students to input their answers. For some questions, students will be allowed to click on respective buttons to get instant feedback.

Visual basic

Visual Basic (VB) is today's most widely used object-oriented programming (OOP) language. This is because it presents the information in a pleasing form, making it is easy to learn the presented material. VB is also the underlying macro engine for all Microsoft products.

It is a high level MS Windows Programming language, developed to provide programmers with a quick and easy method of developing Windows Applications. VB programs are created in an Integrated Development Environment (IDE). The IDE allows the programmer to create, run, and debug VB programs conveniently (without the need to open additional programs) and without being a Windows programming expert. It allows a programmer to create working programs in much less time than it normally takes to code programs without using IDEs. VB is the world's most widely used Rapid Application Development (RAD) language. VB is a distinctly different language providing powerful features such as graphical user interfaces (GUI), event handling, object-oriented features, error handling, and structured programming.

VB provides facilities to create application specific objects. Being an OOP language, it follows the methodology of OOP to model real world objects and other concepts through code. There are a number of ways the OOP can be implemented. They all center around the attempt to produce easily re-usable and robust code on concepts that the programmer will find familiar; objects which have characteristics (properties) and can perform actions (methods).

VB is an event-driven programming language. This means that when the user moves the mouse, or pushes a button, VB generates an event or signal. Based upon the code written into the program, the event is interpreted and an appropriate action is performed. This will help the students to use the tutorial easily and learn the material faster.

When developing a VB application, the user determines what events the program will respond to and a code is written for each of these events. The following flowchart in Fig. 1 shows a simplified model of how a typical VB application operates.

VB provides the programmer with an integral environment where the tools can be used to create a point and click interface and use event driven programming techniques. A developer can quickly and easily create a user interface, then write the code to respond to specific events which occur as a result of user input. The IDE has sophisticated editing and debugging tools which allow the user to attach code quickly to the interface created for each event which is applicable for any type of object on the interface.

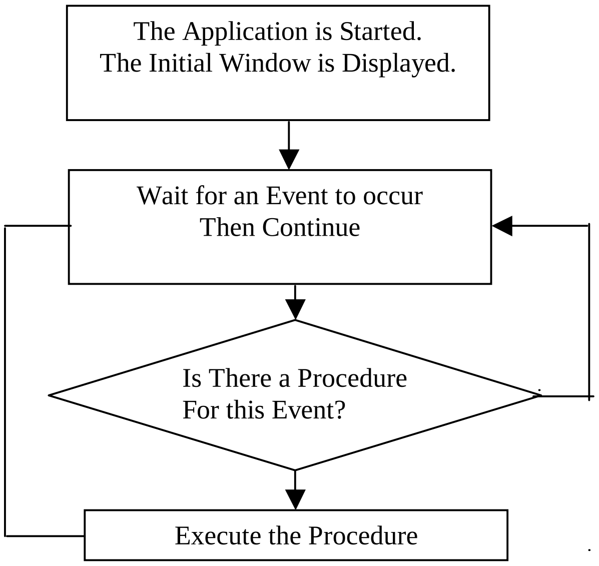

The Application is Started.
The Initial Window is Displayed.
Wait for an Event to occur
Then Continue
Is There a Procedure
For this Event?
Execute the Procedure

Fig. 1: Simplified Model Showing the Operation of Visual Basic Application

When VB is started, the programming environment interface is the first screen to appear. The primary components of this interface are: Title bar, Menu bar, Tool bar, Toolbox, Forms, Code window, Project window, Properties window, Debug window. Forms are central to everything done in VB. A form is a window like any other window. It can have all the standard components of a window-title bar, border, maximize and close buttons, client area and control box. The toolbox contains the tools, which are either controls or insertable objects, which the user can use to build VB applications. When building an application, the user will use form as the background on which to create user interfaces. It is on the form that the user places any of the controls available in the toolbox, such as command buttons, text boxes, and labels. When the program is running, the user sees the forms as normal windows. In this window, the user selects menu options, clicks icons that have been drawn, or enters data into text boxes that have been arranged. A project may contain a single form, many forms, or, in some cases, no forms at all.

Adding controls adds power. One of the strongest features of VB is the way in which the user can heighten its capabilities through the addition of more powerful controls. Some of the most common controls are:
Label, TextBox, CommandButton, Image/PictureBox, CheckBox, OptionButton, just to mention a few.

Working with Objects
Whenever an application using VB is being developed, objects are the medium used. VB provides access to many types of objects. These can be forms and controls or OLE objects, such as Excel Chart. These can also be system objects or you can create your own objects. An object has properties, methods, and events that it responds to, and the user can write code or use the facilities available at design time to configure and manipulate these objects.

Database

A database is a collection of related data. It has the following implicit properties:

1. A database represents some aspect of the real world, like names, locations, and addresses. Changes to the data are reflected in the database.
2. A database is a logically coherent collection of data with some inherent meaning. A random assortment of data cannot correctly be referred to as a database.
3. A database is designed, built, and populated with data for a specific purpose.   It has an intended group of users and some preconceived application in which these users are interested.

A database can be of any size and of varying complexity. A relational database is a collection of relations. Each relation resembles a table, or, to some extent, a simple file. When a relation is thought of as a table of values, each row in the table represents a collection of related data values. These values can be interpreted as facts describing a real word entity or relationship. The table name and column names are used to help in interpreting the meaning of the values in each row of the table. A database management system (DBMS) is a collection of programs that enables users to create and maintain a database. Hence, it is a general-purpose software system.

Access

Access is a relational database management system (RDMS), developed by Microsoft that can be used to store and manipulate large amounts of information. It provides two methods to create a database. The developer can create a blank database and then add the tables, forms, reports, and other objects later. This is the most flexible method, but it requires the developer to define each database element separately. Or, the developer can use a Database Wizard to create in one operation the required tables, forms and reports for the type of database chosen. This is the simple way to start creating the database. In each case, the developer can modify and extend the database at any time after it has been created.

Using Access, the developer can build the tables, forms, queries, reports, macros, and other objects that will make up the database.

Tables

A table is a collection of data about a specific topic, such as products or suppliers. Tables are the primary blocks of Access database. All data is stored in the tables. Every table in the database should focus on one subject, for example, customers, orders, or products. Using a separate table for each topic means that the user stores that data only once. This makes the database more efficient and reduces data-entry errors. Tables organize data into columns (called fields) and rows (called records).

In table Datasheet View, the user can add, edit, or view the data in the table. The user can also check the spelling and print the data, filter or sort the records, change the datasheet's appearance, or change the table structure by adding or deleting the columns. In table Design View, the user can create an entire table from scratch, or add, delete, or customize an existing table's fields.

Forms
The user can use forms for a variety of purposes; create a data-entry form to enter into a table; create a switchboard form to open other forms or reports; create a custom dialog box to accept user input, and then vary out an action based on that input. Most of the information in a form comes from an underlying record source. Other information in the form is stored in the form's design.

Queries
Queries can be used to view, change, and analyze data in different ways. They can be used as the source of records for forms and reports. The most commonly used is a select query. A select query retrieves data from one or more tables using the criteria specified by the user, and then displays it in the order required.

Reports
A report is an effective way to present the data in a printed format. Because the user has control over the size and appearance of everything in a report, the information can be developed the way it is needed. Most of the information in a report comes from an underlying table, query, or SQL statement, which is the source of the report's data. Other information in the report is stored in the report's design.
Macro
A macro is a set of one or more actions that performs a particular operation, such as opening a form or printing a report. Macros can help the user automate common tasks. For example, a macro can be run that prints a report whenever the user clicks a commondbutton.

VHDL

VHDL, a hardware description language for very high speed integrated circuits (VHSIC) is one of the most powerful cutting edge design tools that allows highly complicated digital circuits to be modeled, simulated, and tested before building the actual device. It is used extensively in industry to describe digital systems from very abstract to gate levels. VHDL is a very powerful simulation language. Many constructs in this language are quite useful for modeling digital hardware at high levels of abstraction.   The digital system can also be described hierarchically. Timing can also be explicitly modeled in the same description. VHDL is both an IEEE Std. and an ANSI Std for describing digital designs.

An hardware abstraction of the digital system is called an entity. To describe an entity, VHDL provides five different types of constructs, called design units. They are:

- Entity declaration
- Architecture body
- Configuration declaration
- Package declaration
- Package body.

An entity is modeled using an entity declaration and at least one architecture body. The entity declaration describes the external interface of the entity. It specifies the name of the entity, the name of the interface ports, their mode (i.e. direction), and the type of ports. An example of an entity declaration for an half adder (Fig. 2) is furnished below. It can be viewed in Fig. 13 in the interactive tutorial.

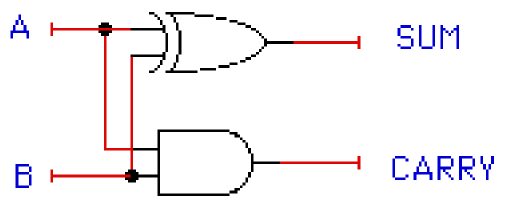


Fig. 2: A Half-Adder Circuit

```
entity HALF_ADDER is
        port (A, B, in BIT; SUM, CARRY, out BIT)
end HALF-ADDER;
-- This is the comment line
```

An Architecture body contains the internal details of the entity. It describes the functionality or the structure of the entity. An entity can modeled in one or many of the three modeling styles:

- Behavioral Modeling
- Dataflow Style
- Structural Style

Behavioral Modeling specifies the behavior of an entity as a set of statements that are executed sequentially in the specified order. A process statement is the core feature used to model the behavior of an entity.
Dataflow style specifies the functionality of the entity without explicitly specifying its structure. The functionality is expressed primarily using concurrent signal assessment statements and block statements.
Structural Style describes an entity as a set of components connected by signals. The component installation statement is the primary mechanism used for describing an entity. Shown below is another example of the modeling of a multiplexer.

```
 -- multiplexer using IF-THEN-ELSE
 multiplexer_1 : PROCESS (sel, d1, d2)
  BEGIN
   IF (sel = ‘1’ ) THEN
         d_out <= d1;
   ELSIF (sel=’0’) THEN
         d_out <= d2;
   ELSE
         d_out <= ‘X’;
   END IF
```

END PROCESS;

A configuration declaration is used to create a configuration for an entity. A package declaration is used to store a set of common declarations, such as components, types, procedures, and functions. Many design units may possibly share these declarations. A package body contains the definitions of subprograms declared in a package declaration. It can be seen that this kind of material can be best learnt, more effectively using a computer interactive tutorial.

Once an entity has been modeled, it needs to be validated by a VHDL system. A typical VHDL system consists ofan analyzer and a simulator. The analyzer reads in one or more design units contained in a single file and complies them into a design library after validating the syntax and performing some static semantic checks. The design library is the place in the host environment where compiled design units are stored.

The simulation simulates an entity, represented by an entity-architecture pair or by a configuration, by reading in its compiled description from the design library and then performing the following steps:

- Elaboration
- Initialization
- Simulation

It can be seen that this kind of material can be learnt very effectively using a computer interactive tutorial.

Implementation of Interactive Tutorial

The objective of this paper is to design a user-friendly interactive tutorial for students to better understand VHDL. This can be viewed as another vehicle or additional vehicle the students can access at any time, at their own pace. The material is presented one page at a time. Students have the option to click on different topics that are of interest to them. They can compare abstract models and descriptions with graphs easily visualized. To enhance the student's understanding of the material, exercises with instant feedback are provided.

The tutorial is offered in two modules. One, the instruction module and two, the practice module. The instruction module gives a detailed description of basic VHDL concepts. The practice module contains two different styles of practice for the students, (1) multiple choice type, and (2) writing code. Correct answers to the practice answers can be presented by the tutorial.

This interactive tutorial is implemented with VB programming language and MS Access database system.

Fig. 3 shows the structural flow chart of the interactive tutorial. Each rectangular box in the flowchart represents a window-type page on the screen. Icons in the rectangular shape are clickable buttons that will lead to an action. Arrows in the diagram represent direction of actions.

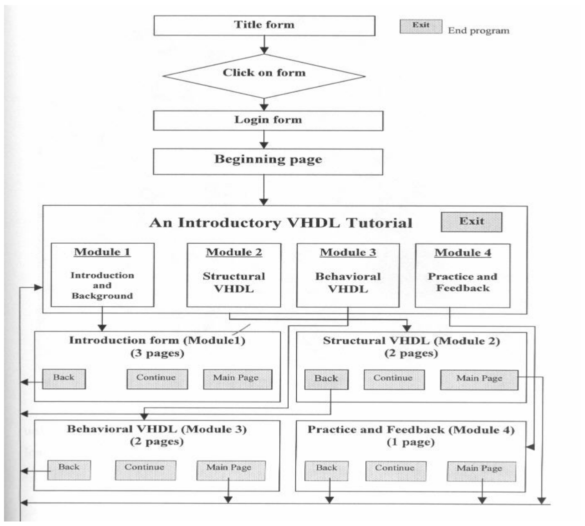


Fig. 3: Structural Flowchart

When the program is started, it pops up the main title page first (Fig. 4). On this page it shows up the title and instructions about how to start this application. This title page contains a form and several label controls with load, unload, and click events. When a title page is clicked, a login form will appear as shown in Fig. 5. The student punches his/her name to log on to the tutorial. This form is implemented with a label, a text box, and a command button. The code for the text box is:

```
Private Sub Text1_LostFocus()
        If Text1.Text = “”Then Text1.setFocus
End Sub
```

The SetFoucs fuction is used to force the users to enter their names.

Once the student logs on to the program, a content page will be shown on the screen (Fig. 6). This form shows the major topics of the tutorial. The title bar of this form gives a tip for how to go to each topic. Students can simply click on different topics to view the details. Each page also contains several buttons, such as back, Exit and Next. These three buttons make the tutorial have a consistent look and make browsing straightforward. At different topics it also supports VHDL code examples with circuit diagrams to help students better understand the instruction on the topic.

For example, if a student clicks on the Introduction of VHDL topic, it will bring up the form shown below (Fig. 7). This form gives a brief introduction of VHDL.

If the student clicks on the VHDL Design Units section, it will bring up the design units page as in Fig. 8. On this form, student can click on different units to get details of the topic. Most of them not only have text information, but also support programs with coding samples. In VB, image/picture box is used to display graphical images, either decorative or active. Image box is a lightweight control. It can not be placed on top of other controls. Picture box is much more functional than image controls. It can work as a container that receives output from graphical methods, or as a container for other controls.

Thus, a user can load a number of different kinds of graphics files into the picture and image controls. This feature really makes the tutorial interactive and user friendly. In this project, all the pictures are incorporated in the bitmap format (.bmp), the traditional Window format for graphics.

The advantage of VB can be seen from this form. For such a big form, the user has to write only a few lines of code. On the design units, students can click on different design units to view details. For example, if a student clicks on the Entity Declaration, it will bring up the entity declaration form (Fig. 9). On this form, the students can view the basic concept of the entity declaration design unit. The learning is enhanced further by the use of examples. Clicking on Example-1 or Example-2 will help view the examples.

Fig. 10 shows example-2 appearing when Example-2 is clicked. This form helps students better understand the text information. They can click on Back to move back. This 4-bit comparator circuit schematic was drawn in Logic Works, and transferred as a bitmap file in PhotoShop. Here, image control was used to bring up this circuit image.

The second module was Practice and Feedback. In this module, students will be able to practice what they learned from the first module. There are two sections in this module. (1) Multiple Choice, and (2) Writing Code.

From the design view point, this Practice and Feedback module is how to use VB to access a database. In VB, there are several different ways to access a database. This tutorial uses control to access the database. Each section has its own database. They are all built in MS Access. The database name of the first section on multiple choice is practice1.mdb. It has one table called Mchoice. There are six fields in that database:

Field 1 is Question. This field is used to store questions for multiple choice. The data type s Memo. It can store up to 64000 characters. This field is bound to a label control.
Fields 2-5 are Choices (Option 1, Option 2, Option 3 and Option 4). For each question, there are four different options for students to choose from. The data type is text. They are bound to four option buttons. Students can click on each option and get instant feedback about their choices.
Field 6 is the Answer field. It shows the correct answer of each question. This field data type is number.

Shown in Fig. 11 is a typical illustration of the practice1.mdb.

The data base name of the second section is prectice2.mdb. This is very similar to pracice1.mdb, but has one table with three fields compared to six fields in practice1.mdb.

Field 1 is question (same as in practice1.mdb)
Field 2 is graphic. This field supports the circuit diagram for each question. The data type is memo. It is bound to an image control box.

Field 3 is answer. This field gives the answer for each question. The data type is text and size is 20. It is also bound to an image box control. Fig. 12 shows a record of practice2.mdb.

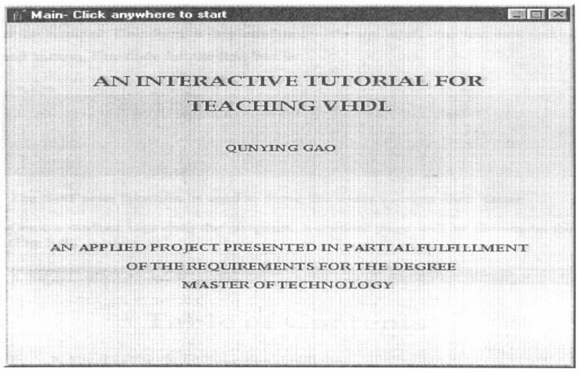
Now look at the application form and examine the details about this design. Consider the Multiple Choice section (Fig. 13). The most important design part is how to bind the database with different controls. Shown below is the coding for Next button, which is used to move to the next record in the database.

```
Private Sub cmdnext_Click()
        Data1.recordset.MoveNext
        Option1(0).Value = False
        Option1(1).Value = False
        Option1(2).Value = False
        Option1(3).Value = False
        IfData1.Recordset.EOF Then
                MagBox “It is finished”
        End if
End Sub
```

In VB, there are a number of built in message box constants that the developer can use to determine which icon and buttons appear on the message box.

The Writing Code section is very similar to the Multiple Choice section. The application form looks like as shown in Fig. 14. The VB coding for answer button in this case is:

```
Private Sub cmdanswer_click()
        Image2.Picture = Loadpicture (Data1.Recordset.Fields(“answer”))
        Image2.Visible = true
End Sub
```


Main- Click anywhere to start

AN INTERACTIVE TUTORIAL FOR
TEACHING VHDL

QUNYING GAO

AN APPLIED PROJECT PRESENTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE
MASTER OF TECHNOLOGY

Fig. 4: Title Form

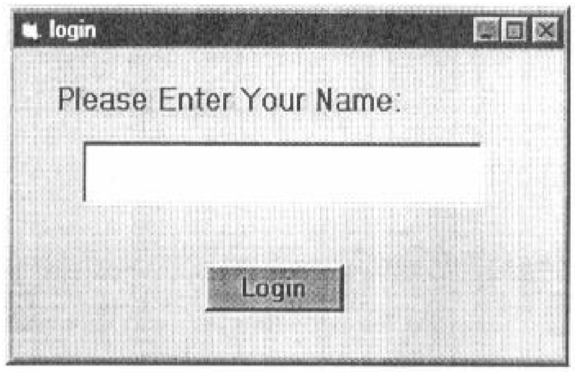


Fig. 5: Login Form

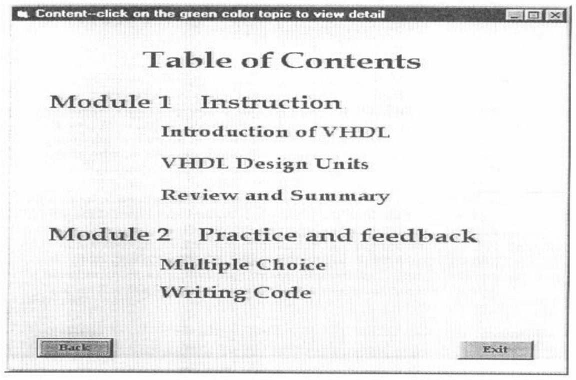

Content--click on the green color topic to view detail

Table of Contents

Module 1 Instruction

Introduction of VHDL

VHDL Design Units

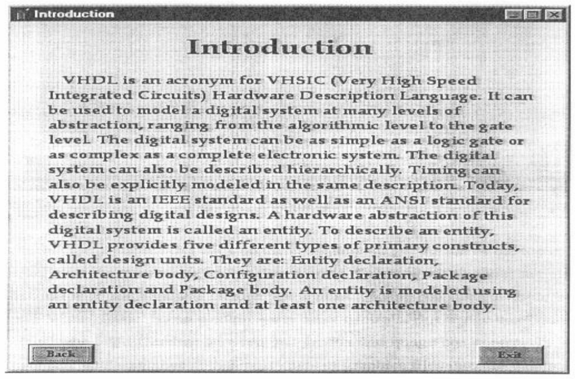
Review and Summary

Module 2 Practice and feedback

Multiple Choice

Writing Code

Fig. 6: Table of Contents Form


Introduction

Introduction

VHDL is an acronym for VHSIC (Very High Speed Integrated Circuits) Hardware Description Language. It can be used to model a digital system at many levels of abstraction, ranging from the algorithmic level to the gate level. The digital system can be as simple as a logic gate or as complex as a complete electronic system. The digital system can also be described hierarchically. Timing can also be explicitly modeled in the same description. Today, VHDL is an IEEE standard as well as an ANSI standard for describing digital designs. A hardware abstraction of this digital system is called an entity. To describe an entity, VHDL provides five different types of primary constructs, called design units. They are: Entity declaration, Architecture body, Configuration declaration, Package declaration and Package body. An entity is modeled using an entity declaration and at least one architecture body.

Fig. 7: Introduction Form

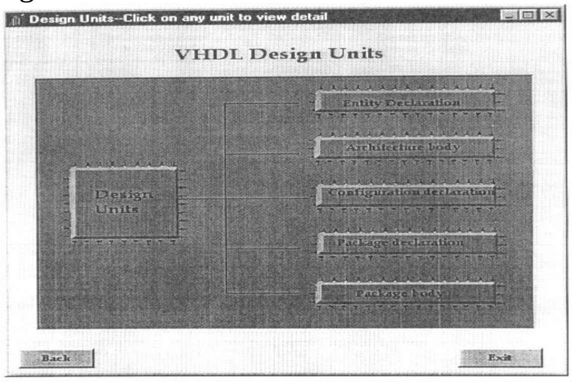
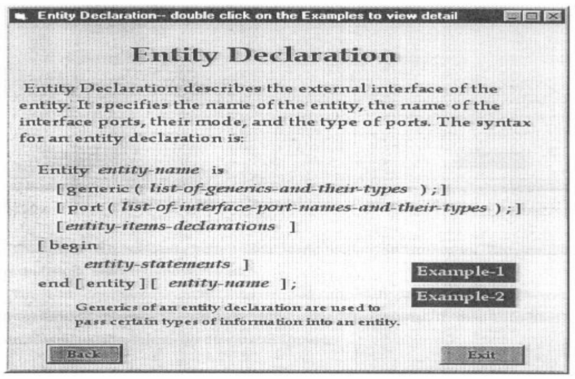


Fig. 8: Design Units Form


Entity Declaration-- double click on the Examples to view detail

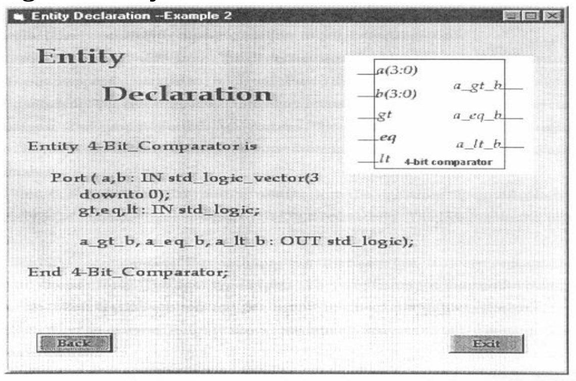
Entity Declaration

Entity Declaration describes the external interface of the entity. It specifies the name of the entity, the name of the interface ports, their mode, and the type of ports. The syntax for an entity declaration is:

```
Entity entity-name is
   [ generic ( list-of-generics-and-their-types ) ; ]
   [ port ( list-of-interface-port-names-and-their-types ) ; ]
   [ entity-items-declarations ]
[ begin
      entity-statements ]
end [ entity ] [ entity-name ] ;
```

Generics of an entity declaration are used to pass certain types of information into an entity.

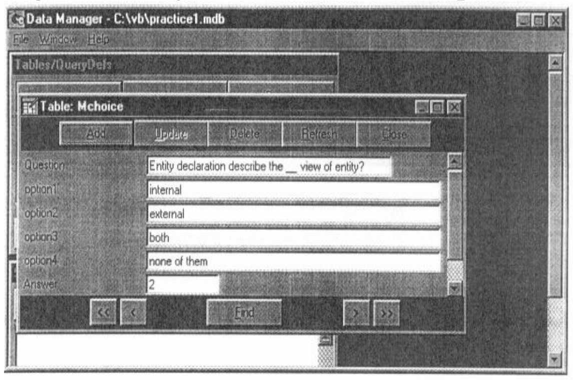
Fig. 9: Entity Declaration Form



Entity Declaration

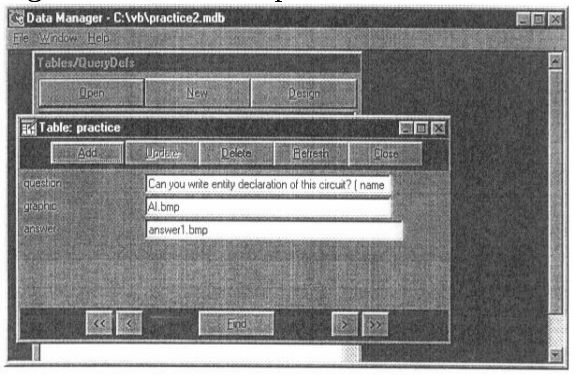
```
Entity 4-Bit_Comparator is

   Port ( a,b : IN std_logic_vector(3
        downto 0);
        gt,eq,lt : IN std_logic;

        a_gt_b, a_eq_b, a_lt_b : OUT std_logic);

End 4-Bit_Comparator;
```

Fig. 10: Entity Declaration – Example2 Form



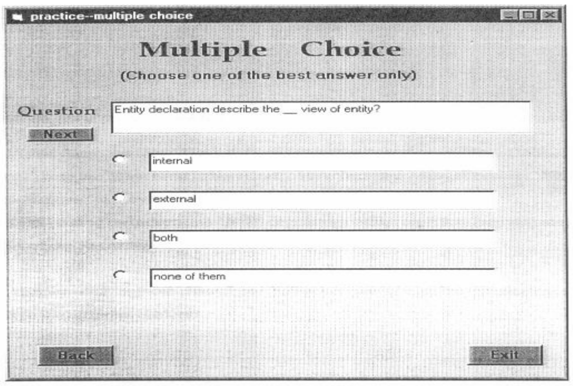
Question	Entity declaration describe the __ view of entity?
option1	internal
option2	external
option3	both
option4	none of them
Answer	2

Fig. 11: A Record of practice1.mdb



question	Can you write entity declaration of this circuit? (name
graphic	AI.bmp
answer	answer1.bmp

Fig. 12: A Record of practice2.mdb


practice--multiple choice

Multiple Choice

(Choose one of the best answer only)

Question

Next

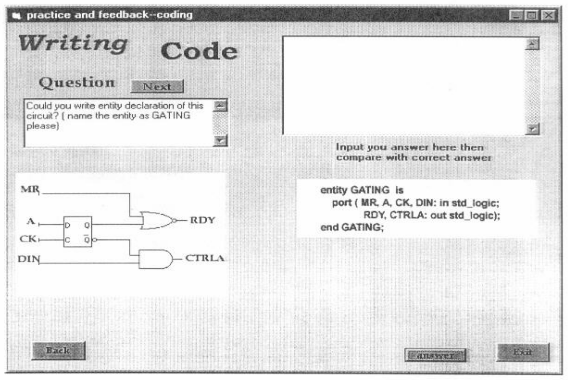
Entity declaration describe the __ view of entity?

- internal
- external
- both
- none of them

Back

Exit

Fig. 13: A Record of practice2.mdb


practice and feedback--coding

Writing Code

Question

Next

Could you write entity declaration of this circuit? (name the entity as GATING please)

Input you answer here then compare with correct answer

```
entity GATING  is
    port ( MR, A, CK, DIN: in std_logic;
          RDY, CTRLA: out std_logic);
end GATING;
```

Back

answer

Exit

Fig. 14: A Record of practice2.mdb

Conclusions

This paper focuses on designing a tutorial for VHDL based on the VB programming language and the Access database program. It uses the various characteristic features of VB, such as forms. Complementing the class lecture, and the text, this tutorial offers the following distinct advantages for learning VHDL effectively.

- The tutorial is user-friendly. A beginner can just click on the first form to start the tutorial. The tutorial is organized into pages. Buttons are designed in each page to enable students to browse the tutorial easily. Students can click on any topic of their interest, and start cruising. Tips and hints are provided to assist students in the case of any hardships encountered during their use.

- The GUI has been emphasized in the design of the tutorial based on the features of VB. All commands and events are controlled by dragging and clicking the mouse. Students need not remember any command to use the program. Wherever possible, graphics are heavily used to help visualize some abstract concepts and entities.

- Interactiveness has been achieved using VB and Access database. Questions stored in Access database can be retrieved using VB to test student understanding of basic concepts. Students can get instant feedback about their choice of each multiple-choice question. They need to choose an answer before the tutorial can give the correct answer.

It is hoped that this tutorial may make leaning VHDL more interesting, effective, convenient, and a rewarding experience. However, additional features need to be added to make it internet/web-based. Also, the feedback from actual use of the students is yet to be done. Additionally, not all the components or entities have been added at this stage. Further work needs to be done to incorporate additional VHDL exercises and input them to the database.

Bibliography

[1] Q. Gao, An Interactive Tutorial for Teaching VHDL, Master's Applied Project, June 1998
[2] Deitel & Deitel, Visual Basic 6 How to Program, Prentice Hall, New Jersey, 1999
[3] J. Bhasker, A VHDL Primer, PTR Prentice Hall, 1995
[4] M. C. Kerman, R. L. Brown, Visual basic 6.0, Addison-Wesley, New York, 2000

Author Biographies

Robert W. Nowlin received the BSEE degree from the university of Washington, Seattle, in 1963, the MSEE degree from San Diego State University, San Diego, in 1969, and the PhDEE degree from Texas Tech University, Lubbock, in 1975.

Dr. Nowlin is a Professor and Chair of the Electronics & Computer Engineering Technology Department at Arizona State University East. He taught at several other universities before coming to ASU and has extensive industrial experience. His current research interests are microcontrollers and VHDL. He is a Senior Member of IEEE and a Member of ASEE.

Qunying Gao earned the BS in Chemistry from the Fuzhou University, Fujian, China, majoring in analytical chemistry in 1991. She worked as an Assistant Engineer at Shuiko Water Power Plant, Minqing, Fujian, China, before coming to United States. She earned her Master of Technology in EET from Arizona State University East, in 1988. Currently she is a RS/6000 Performance Engineer at IBM, Austin.

Raji Sundararajan received the BSEE degree from the University of Madras, India, in 1981, the MSEE degree from Indian Institute of Science, Bangalore, India, in 1988, and the PhDEE degree from Arizona State University, Tempe, AZ, in 1993.

Dr. Sundararajan is an Assistant Professor in the Department of Electronics & Computer Engineering Technology. She served at the Dept. of Space as an electrical engineer and has teaching and research experience from EE Department at Arizona State University, Tempe, AZ. Her research and teaching interests are power engineering and digital systems. She is a member of IEEE and ASEE