AIR QUALITY MONITORING SYSTEM (BACK END) USING FPGA

AMRILLAH BIN NAJMIN
Faculty of Electrical Engineering Universiti Teknologi MARA Malaysia
40450, Shah Alam, Malaysia.

ABSTRACT

This paper is about the development of Air Quality Monitoring System using FPGA. This system is developed in order to monitor air pollution at our country. This system consists of software and hardware part. For the software part, the Graphical User Interface (GUI) is developed using JAVA tools as a medium to activate the sensor through the Field Programmable Gate Array (FPGA) Quartus II Board and the output data from the sensors will be displayed at PC. For the hardware part, a circuit, consists of temperature and humidity sensors, is activated and processed by the program written in VHDL and implemented on Altera CycloneII FPGA. The GUI menu, the FPGA and the sensors circuit are connected together to develop the complete system of the air quality monitoring system.

Keyword: Graphical User Interface, JAVA, Field Programmable Gate Array (FPGA), Air Quality Monitoring System.

I. INTRODUCTION

Human health and environment can be harmed by air pollution in any country. The pollutants can be gases such as carbon monoxide (CO), nitrogen dioxide (NiO_2), sulphur dioxide (SO_2) etc. Any significant changes in the ambient air quality status have to be monitored continuously and manually as implemented by the Department of Environmental in our country. The Department of Environment (DOE) monitors the country's ambient air quality through a network of 51 stations that are strategically located in residential, traffic and industrial [1].

However, some of the system, for example in Malaysia, does not provide an instant analysis where the measurement obtained now is manually collected and delivered to a laboratory for analysis and it is obtained monthly from Alam Sekitar Malaysia Sdn Bhd (ASMA) [2]. Furthermore, people are not aware of the changes in air quality unless they have been informed about it. They just know about the air pollution and IPU index but they do not know the composition of gases in the air. People who have severe allergy problem toward certain level of gases, such as asthmatic people, may expose to the unhealthy environment without any warning and there is no device is to help people to be more alert about quality of air at their surroundings.

By having a user-friendly monitoring system, people will know the amount of "poisonous" gas such as carbon monoxide and sulfur dioxide and ammonia that are emitted from vehicles and factories. Thus, they can avoid themselves from being exposed to this kind of environment, the authorities at factories site will use the information to control the emission of gases from their factories, and this will provide a good environment for the society. Here, we develop a system that can give information automatically such as temperature, CO content and humidity. The results displayed will be warning sign to those who have health problems and warning about the drought and flood season. We will be focusing on the development of the thermal system using thermal sensor and we will be using FPGA as the controller for the sensor and graphical user interface menu at back end of the system

.

II. SCOPE OF WORK

The scope of work of this project consists of seven (7) distinct components, which are as follows:

1. Study the basic concept of developing graphical user interface using JAVA tools.
2. Develop a GUI using Blue J
3. Develop the sensor circuit and other connection to the FPGA
4. Study on specifications and structures of Altera Cyclone II development board.
5. Develop serial communication interface with PC and implement a communication protocol for the system.
6. Implement the algorithm/technique circuit on the Cyclone II chip using Quartus II software.
7. Test the system for usability and accessibility.

III. AIR QUALITY MONITORING SYSTEM MODEL

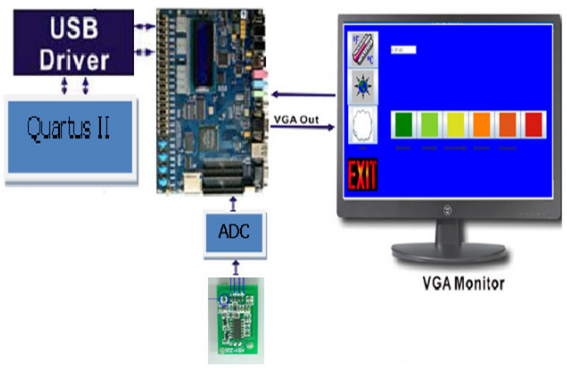

Figure 1: System model

Figure 1 shows the overall model of the system which comprises a computer (PC), two sensors with Analog to Digital Converter (ADC), Quartus II tools equipped with USB Blaster driver and Altera Cyclone II FPGA Board as the controller. Altera Cyclone II board is connected to the PC via serial port (communication port) RS-232, USB Blaster and VGA port. The USB is a two way data paths where the desired program from Quartus II software can be loaded to the FPGA , while, the RS232 is used to transfer information from JAVA to the FPGA and it can transmits back the signals to the computer.

The sensor circuit will detect the ambient air humidity and temperature and provide the analog output. The relationship between the humidity, temperature and the output voltage is shown in the datasheet. ADC is used in the temperature circuit to convert the analog output to the 8-bit binary value. The FPGA device must be programmed and configured first to implement the circuit. The required configuration file is generated by the Quartus II Compiler's Assembler module. Altera DE2 board allows the configuration to be done in two different ways, which is in JTAG and AS modes. The configuration data is transferred from the host computer, which runs the Quartus II software to the board via a cable that connects a USB port on the host computer. The description of each component in the system is described in the following subsection.

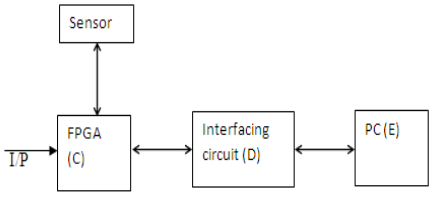

Figure2: Block diagram for air quality monitoring system (back-end).

A. Field Programmable Gate Array (FPGA) – Altera Development Board DE2

An FPGA (Field Programmable Gate Array) is a user-programmable logic device that can be configured to perform a variety of complex logic operations. It is similar to the PLAs (Programmable Logic Arrays) but much more powerful. Figure 5 shows the basic layout of an FPGA, which consist of an array of logic elements (LEs) that can be wired together in a user-defined manner using programmable interconnect switches. These internal logic element blocks communicate with external hardware through I/O blocks on the outer edge of the FPGA. The smallest unit of logic in the Cyclone II architecture, the LE, is compact and provides advanced features with efficient logic utilization. The LE features are:

(a) a four-input look-up table (LUT), which is a function generator that can implement any function of four or fewer variables

(b) A programmable register

(c) A carry chain connection

(d) A register chain connection

(e) The ability to drive all types of interconnects such as local, row, column, register chain, and direct link interconnects

(f) Support for register packing

(g) Support for register feedback.

The switches on Altera Development Board DES2 are used to select which sensor will be activated and measured.

The FPGA is programmed with control and processing algorithm, where the data obtained from the sensor is analyzed and the results will be sent to the PC. The interfacing circuit is also available at the development board but the handshaking or protocol program will be included in the control algorithm.

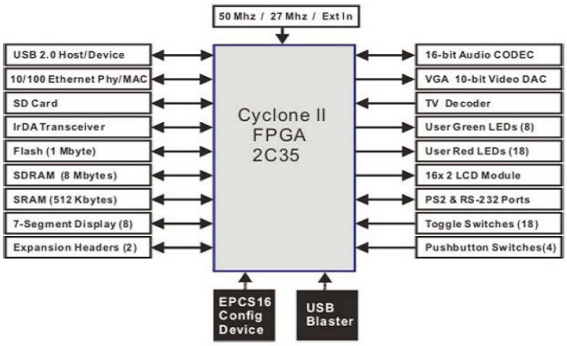

Figure 3: Block diagram of the DE2-70 board

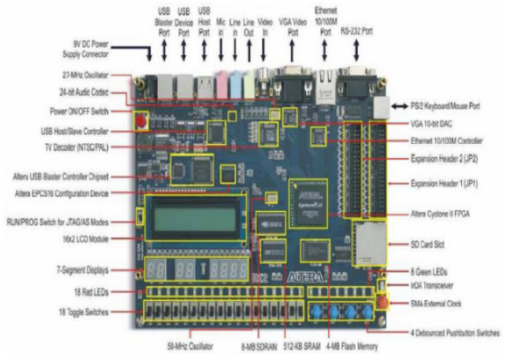

Figure 4: The DE2-70 board

B. Configuring the Cyclone II FPGA

The DE2 board contains a serial EEPROM chip that stores configuration data for the Cyclone II FPGA. This configuration data is automatically loaded from the EEPROM chip into the FPGA each time power is applied to the board. Using the Quartus II software, we can reprogram the FPGA at any time, and can change the non-volatile data that is stored in the serial EEPROM chip. The DE2 board provides two programming modes JTAG and Active Serial (AS) [5].

1. JTAG programming: for this method of programming, its name stands for Joint Test Action Group. The configuration bit stream is downloaded directly into the Cyclone II FPGA. The FPGA will retain this configuration as long as power is applied to the board and the configuration is lost when the power is turned off.

2. AS programming: In this method, called Active Serial programming, the configuration bit stream is downloaded into the Altera device serial EEPROM chip. It provides non-volatile storage of the bit stream, so that the information is retained even when the power supply to the board is turned off. When the board's power is turned on, the configuration data in the device is automatically loaded into the Cyclone II FPGA.[7]

C. Graphical User Interface (GUI) Display

GUI is an event-driven programming concepts and object-oriented programming terminology. JAVA is a programming language developed by Sun Microsystems that can be used to develop many types of GUI applications [11]. There are simple text-based programs called console applications. These programs just support text input and output to the computer screen. A GUI component is an object that defines a screen element used to display information or allow the user to interact with a program in certain way. Examples of GUI component include push buttons, text fields, labels and menus.[3]

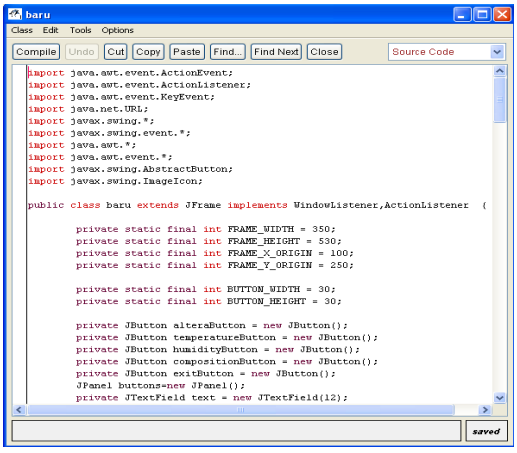

Figure 5: example of JAVA code

D. *SENSOR DESCRIPTION*

The HHT02D is single multi sensors chip which relative to humidity and temperature module comprising a calibrated digital output. The device includes capacitive polymer sensing element for relative humidity and temperature sensor. The main advantages of using this sensor are due to the fast response time and insensitivity to the external disturbance. The 2-wire serial interface and internal voltage regulation allow fast and easy system integration.

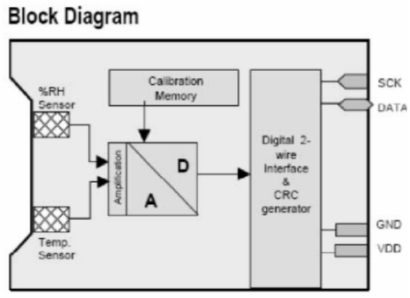

Figure 6: sensor block diagram

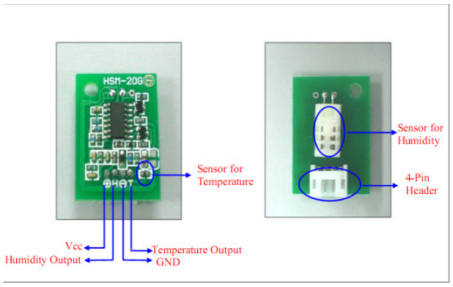

Figure 7: sensor view

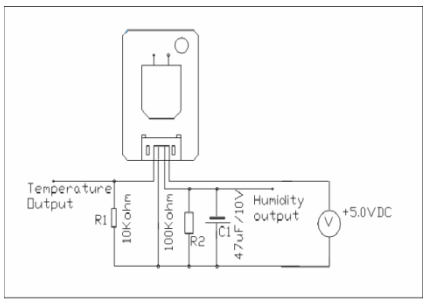

Figure 8: sensor schematic

IV. METHODOLOGY

A. *GUI development*

Java is a case sensitive fully object-oriented language that has rich collection of existing classes in java class libraries (also known as Java APIs). Object-oriented languages give classes, which can be reused in other applications. To do so, this class needs to be well written for further development. The GUI API, which is defined in Java platform, includes functionality to draw graphics/widgets on the output device and to input events from the input devices. It is based on Java Abstract Window Toolkit (AWT).

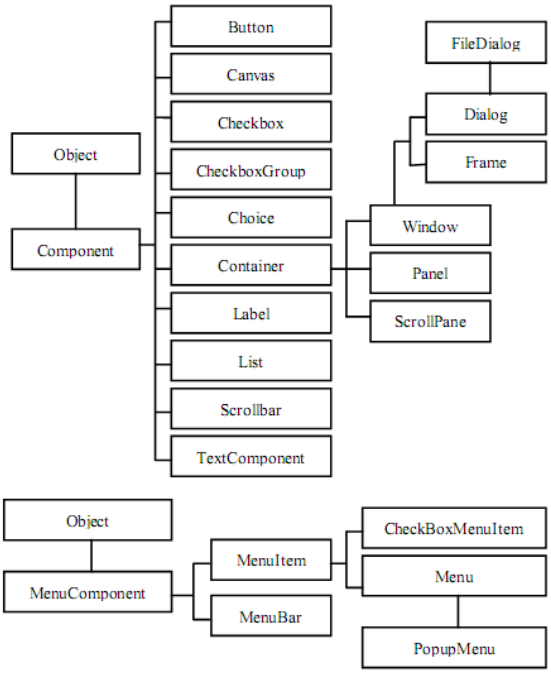

Figure 9: JAVA component

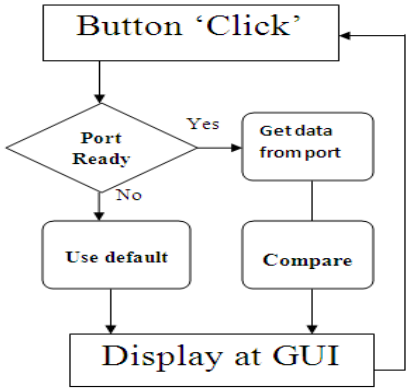

Figure 10: GUI operation flowchart

For this project we have developed an event GUI. We have designed the profiler with the following goals in mind: (1) it should be light-weight so that it does not influence the performance of the application, (2) it should be generic to be applied to any type of GUI applications, and (3) it should be easy to implement.[4]

The java.awt package provides a user interface to allow applications written in Java to generate graphical output and receive user input events[10]. The AWT provides a common set of tools for GUI design that work on a variety of platforms. Amongst the tools in AWT, frame and button are used in this GUI.

A frame is a container that used to display GUI. A frame is defined by the *JFrame class* and contains small buttons in the corner of the frame that allow the frame to be minimized, maximized and closed. A push button is a component that allows the user to initiate an action with a press of the mouse. It is defined by the *JButton class*[8] . A *JButton* generates an action event when it is pushed. The only event of interest occurs when the button is pushed. To respond to the event it should create a listener object for that event and write a class that represent the listener. For this case, we need an *actionevent* listener.

To interface the *ActionListener* with the GUI menu for this case the button, *ActionPerformed* method must be implemented. The button will call the *ActionPerformed* method when the event occurs, passing in an *ActionEvent* object that represents the event[9].

For this project we use 4 different listener objects. When either button is pushed it will invoke its own results. To do that, the button listener must add to all buttons so that when either button is pressed, the *actionperformed* method is invoked. On each invocation, the *action performed* method uses an if-else statement to determine which button generates the event. The *getSource* method is called on the *actionevent* object that the button passes into the *actionperformed* method.[6]

The condition of *the if* statement compares the event source to the reference to the other button[12]. If they don't match then the event must have been generated by the other button.

The *JFrame* for the program is constructed, set up and displayed. The *JFrame* constructor takes a string as a parameter that it displays in the title bar of the frame. When the close button in the corner of the frame is click, the *setDefaultClosedOperation* method determines what will happen[13]. In most cases, that button will terminate the program as indicated by the EXIT_ON_CLOSE constant.

B. Developing the sensor circuit

Temperature and humidity sensor circuit is used to measure the ambient temperature and humidity and then convert the sensor output from analog to digital. The sensor used is HSM-20G which contains both sensors on a circuit board. The electrical output from the sensor is proportional to the temperature and also depends on the humidity. From the datasheet, the temperature sensor output voltage can be calculated by this relationship: for each ^{o}C the output voltage is 10mV or 10mV/^{o}C.

The analog output of the sensor will be converted to 8 bit binary by using ADC. To represent the output binary, LEDs is used. Each LED present one bit of the output. This output then will be sent to the input of the FPGA board.

C. Interfacing with FPGA's

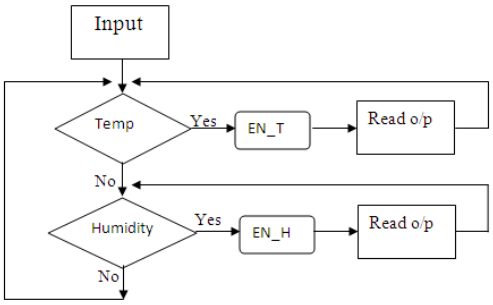

Figure 10: sensor operation flowchart

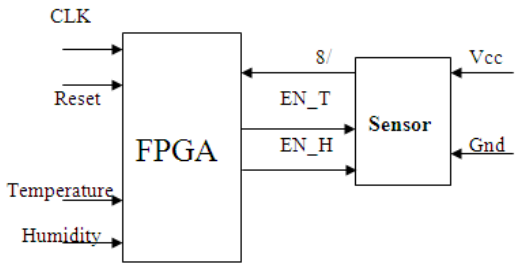

Figure 11: input controlling

For this project, there are two types of interfacing which are the interfacing between the FPGA and the sensor and between the FPGA board and the PC (GUI menu)

V. RESULTS

A. *GUI menu*

Here is the result for the GUI menu developed using JAVA. It contains the menu button, textfield to present the value, separate window to represent the graph of the humidity level and the ozone button that follow the standard colour of API.

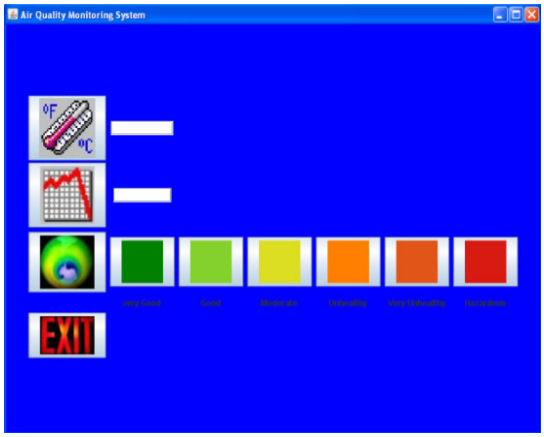

Figure 12: GUI menu

When the temperature button is clicked the present temperature value will be displayed on the *textfield*. If user pushes that button again it will give the value of the temperature at that time.

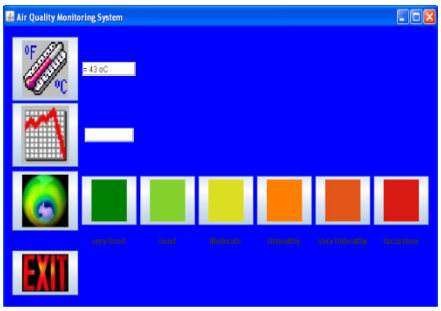

Figure 13: Temperature button pushed

When the ozone button is pushed it will extract data then compare with the range of the exact condition regarding the Air Pollution Index. Then it will indicate the condition whether it's good or hazardous.

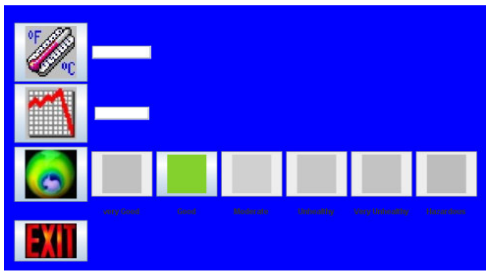

Figure 14: Ozone button pushed

B. *Circuit testing*

For this part, the results are shown as in Figure 15. When there is a changing in temperature or humidity, the output represent by the changing of 8-bit binary at LED.

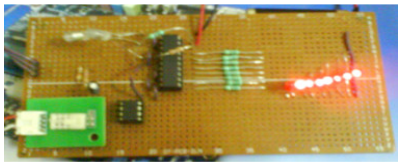

Figure 15: Circuit testing

The reference voltage for this temperature input circuit is 1.33V. Then 1.33/256= 5.195mV. Then the data are collected at different temperature as shown in table 1.

Temperature, ^{O}C	Output voltage, V	Output voltage/5.195mV	Binary
27	0.27	52	110100
28	0.28	54	110110
29	0.29	56	111000
30	0.30	58	111010
31	0.31	60	111100
32	0.32	62	111110
33	0.33	64	1000000
34	0.34	65	1000001
35	0.35	67	1000011

Table 1: temperature output

For the humidity sensor input, the reference voltage for this circuit is 9.888V. Thus, each bit represents 0.0386V. The expected result is shown in Table 2. For every additional 0.0386V output, humidity increases by 1%.

Humidity, %	Output voltage,V	V/0.0386V	Binary
55	2.2	57	111001
58	2.3	60	111100
75	2.8	73	1001001
80	3.0	78	1001110

Table 2: humidity output

C. Interfacing with JAVA

The JAVA and the FPGA interfacing part is still in progress and if it is successful, the real results can be displayed at the GUI menu.

VI. CONCLUSION & FUTURE DEVELOPMENT

One of our goals in the air quality monitoring system project is to implement the user interface by using well-known APIs defined in Java platform. So far, we have successfully developed a demonstration of user interface for several button menus. We also able to process the sensors output using the FPGA. Thus, the interfacing between FPGA and other circuit, which in this case is the sensor circuit, has been successfully developed.

All of these are to ensure that the design has great scalability, reliability and platform independence. Benefit of design using for collaborative FPGA are as follows:

- Access to environment via GUI
- Intuitive graphical user interface
- User managed access to the environment
- Easy implementation of new modules.

However, a lot of implementation works remains to be done in future. Our biggest goal is to integrate the user interface with real software system and make it work in real environment. We plan to embed our system, the hardware and software in a set-top box.

VII. ACKNOWLEDGEMENT

Hereby, author likes to present his sincere appreciation to Dr. Azilah Bt Saparon, for her patience, advice, direction and careful guidance. She had assisted me to quest the information imperative for this project which had been an essential contribution in order to accomplish this project on time.

VIII. REFERENCES

[1] Department of Environment (DOE). 2003. "*Laporan Tahunan Kualiti Environment Malaysia Kuala Lumpur*:" Kementerian Sains Teknologi and Alam Sekitar, Malaysia.

[2] Norela Sulaiman, Hazila Abdul Samat & Maimon Abdullah "*Determination of Air Quality and Noise Levels in the Mixed Industrial and Residential Areas of Nilai, Negeri Sembilan, Malaysia*".
Pusat Pengajian Sains Sekitaran dan Sumber AlamFakulti Sains dan TeknologiUniversiti Kebangsaan Malaysia 43000 UKM Bangi, Selangor D.E. Malaysia

[3] Carey, J. (Ed.) Human Factors in Information Systems: "*The Relationship between User Interface Design and Human Performance*". Ablex: Greenwich, CN, 1997

[4] Weimin Zhang; Devgan, S.S., "Java based graphical user interface for Livingston," *Southeastcon 2000. Proceedings of the IEEE* , vol., no., pp.197-200, 2000

[5] Nino, S.; Mori, T.; YoungHun Ko; Shibata, Y.; Oguri, "*FPGA Implementation of a Statically Reconfigurable Java Environment for Embedded Systems Field-Programmable Technology*", 2007. ICFPT 2007. International Conference on 12-14 Dec. 2007 Page(s): 317-320

[6] Chengyuan Peng and Petri Vuorimaa," *DEVELOPMENT OF JAVA USER INTERFACE FOR DIGITAL TELEVISION*" Telecommunication Software and Multimedia Laboratory,
Helsinki University of Technology, P.O. Box 5400, FI-02015 HUT, Finland.

[7] E. Keller, "*JRoute: A Run-Time Routing API for FPGA Hardware*," 7th Reconfigurable Architectures Workshop, Lecture Notes in Computer Science 1800, pp 874-881, Cancun, Mexico, May, 2000.

[8] David J.Barnes, Micheal Kolling "*Object First With JAVA: A practical Introduction Using BlueJ*" Third Edition. Pearson Education Limited ISBN-10: 0-13-197629-X

[9] David D. Riley "*The Object Of JAVA: Introduction to Programming Using Software Engineering Principle*". Pearson Education Limited. ISBN: 0-321-12173-2.

[10] John Lewis,Peter J.DePasquale and Joseph Chase " *JAVA Foundations: An Introduction to Program Design & Data Structures*" . Addison Wesley. ISBN-10: 0-321-42972-9

[11] Y. Daniel Liang "*Introduction to Java Programming*" Fourth Edition. Pearson Education Limited ISBN : 0-13-120117-4

[12] Peter Van Der Linden "*Just Java2*" Sixth Edition.Sun Microsystems Press Prentice Hall ISBN: 0-13-148211-4

[13] Y. Daniel Liang "*Introduction to Java Programming: Comprehensive Version*" Fifth Edition. Pearson Education Limited ISBN : 0-13-148952-6.