Bluetooth Students' Attendance
Verified by Fingerprint Sensor

by

Noradzni binti Nadzrin

Dissertation submitted in partial fulfilment of
the requirements for the
Bachelor of Engineering (Hons)
(Electrical & Electronics Engineering)

DECEMBER 2006

Universiti Teknologi PETRONAS
Bandar Seri Iskandar
31750 Tronoh
Perak Darul Ridzuan

CERTIFICATION OF APPROVAL

BLUETOOTH STUDENTS' ATTENDANCE

VERIFIED BY FINGERPRINT SENSOR

by

Noradzni binti Nadzrin

A project dissertation submitted to the
Department of Electrical & Electronics Engineering
Universiti Teknologi PETRONAS
in partial fulfilment of the requirement for the
Bachelor of Engineering (Hons)
(Electrical & Electronics Engineering)

Approved:

____
Dr. Abdallah Belal Adam
Project Supervisor

UNIVERSITI TEKNOLOGI PETRONAS
TRONOH, PERAK

December 2006

CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein has not been undertaken or done by unspecified sources or persons.

Noradzni binti Nadzrin

ABSTRACT

Recording students' attendance using the conventional way is tedious and does not prevent any attempts of cheating. Attendance of students in class has been a major component in evaluating students' performance in academic. Thus, the attendance must be recorded correctly and systematically. Students' attendance using Bluetooth is designed not only to avoid any attempts of cheating but to provide comfort for students to mark their attendances and to ease the lecturers in keeping the record. A student needs to enter his or her student's matric number on a wireless keypad which utilizing Bluetooth technology. The input from keypad is processed by a Peripheral Interface Controller (PIC) and displayed through a Liquid Crystal Display (LCD). The data is then sent to the computer via a stand-alone Bluetooth. Visual Basic 6.0 is a software used to process the data received via Bluetooth. Random fingerprint verification using a fingerprint sensor is done after all students have key in their students' matric number wirelessly to ensure no attempts of cheating and the students' matric numbers are entered by designated students only. The scope of study involves keypad, PIC, Bluetooth and Visual Basic. This project was focusing on creating a wireless fingerprint sensor for recording students' attendance but wireless keypad is constructed as an alternative due to cost and technology constraints. However, this new project also serves the same purpose as the original idea and eases the system of recording the students' attendance by adding in some touch of latest technology

ACKNOWLEDGEMENTS

I would like to acknowledge my FYP supervisor, Dr. Abdallah Belal Adam for his advice, motivation, guidance and helpfulness upon completing this project. Thank you to to Mr. Zuki Yusoff, a lecture in Electrical and Electronics Engineering of Universiti Teknologi PETRONAS and to my cousin, Mohd Ariff for helping me a lot in the programming codes. I also would like recognise Mr Low Tan Jung, a lecture in Information Technology for contributing his ideas in this project. Last but not least, to Electrical Technicians, Miss Siti Hawa, Mr Isnani and Miss Siti Fatimah for their help, and to my family and colleagues, thank you for all the encouragement.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Background

Advance and systematic: the two keywords that describe this project. Students' attendance using Bluetooth provides convenience to students and also eases the lectures in recording the students' attendance. This project can be divided into two main parts; the transmitter that transmits wirelessly the students' matric numbers, and the receiver, which processed the information received and display it in user friendly manner.

For the transmitter part, there are two options of getting the students data. One is by using fingerprint sensor and the other one is by using a normal keypad where the students need to enter their students' matric number instead of scanning their fingerprint. Although these two methods are different in term of the advancement of technology used; they serve the same purpose; gathering the students' data. They also share the same goal; to create an advance and more systematic students' attendance system. This can be achieved by making them wireless.

Fingerprint sensor is definitely the best option since it is one of the biggest inventions that utilizing biometric technology. Fingerprint is used as a replacement for user authentication method using password or pin. Fingerprint sensor authenticates a user based on biometrics technology and thus gives a higher security compared to passwords or pin identification [1]. However, fingerprint sensors that are available in the market are not equipped with wireless feature. On the other hand, to create a wireless fingerprint sensor is costly and requires advance technology for construction process.

The least cost solution but serving the same purpose is required. A keypad is replaced as an input to record students' attendance instead of a fingerprint sensor. This keypad is connected to

PIC so that the input from the keypad can be processed, displayed through an LCD and sent to the computer wirelessly.

Since wireless technology definitely offers a great convenience for recording students' attendance, Bluetooth technology is chosen as the wireless solution for this project due to several distinguishing reasons. Bluetooth supports both voice and data applications, able to penetrate solid objects, omni-directional and does not require line-of-sight positioning of connected devices [2].

Even the construction of wireless keypad is inexpensive and simple, one major issue might occur. The students may enter not only their students' matric number, but also others as well. In order to cater this issue, the system is equipped with additional feature which will double check the students' matric numbers entered with their fingerprints. Random fingerprint verification using a fingerprint sensor is done after all students have key in their students' matric number wirelessly. This is to ensure no attempts of cheating and the students' matric numbers are entered by designated students only.

1.2 Problem Statement

A previous project on "Students Class Record Using Fingerprint Identification Sensor" is carried out successfully whereby the system managed to record students' attendance on 10 selected students by utilizing a fingerprint sensor [3].

However, further enhancement is needed to implement this system in real application. In the implemented project, fingerprint sensor was connected directly to the computer. A lot of difficulties were encountered in wired connection. Therefore wireless connection is highly recommended in implementing the project in the lecture halls.

After thorough research on adding wireless feature to the fingerprint sensor, there are a lot of constraints that have put a stop to this idea. One of major issues is the construction of the fingerprint sensor itself. Since it is a Universal Serial Bus (USB) slave device, it needs a master to recognize it and interpret the scanned fingerprint. Nowadays, the available USB master would be the device that has its owned operating system and capable of running the enumeration process like a computer. To add a Bluetooth to the sensor means the sensor must be recognized wirelessly and a wireless USB master must be created. There is one wireless USB master by Silicon Laboratory (Silab) but it only able to recognized simple device such as a mouse and thumb drive. There are also a complete set of stand alone USB master from Cambridge Silicon Radio (CSR) and from Cypress Semiconductor Corporation but there are very costly and the construction is meant for industry used and not for research purpose.

Thus, wireless keypad has been chosen as an alternative to wireless fingerprint and all possibilities of attempting of cheating has taken into consideration.

1.3 Objectives and Scope of Study

1. To create an advance, convenient and more reliable system for recording students' attendance.
2. To ease the lecturers in keeping the record.
3. To avoid any attempts of cheating
4. To enhanced a senior FYP project entitled "Students Class Record Using Fingerprint Identification Sensor"

The scope of study involves keypad, PIC, Bluetooth and Visual Basic. Basically, this project will be applied in UTP lecture halls to record the students' attendance efficiently.

For the first semester, the project focused on constructing circuitry to connect fingerprint sensor and Bluetooth. However, it is agreed that it is very difficult to create a wireless fingerprint sensor due to constraints in term of cost and technical support. Thus, for the second semester, this project focuses on implementation of wireless keypad to record the students' attendance.

CHAPTER 2

LITERATURE REVIEW

2.1 Bluetooth Wireless Technologies

The term Bluetooth refers to a standard defined by a special interest group (SIG). This group consists of five companies; Ericsson, IBM, Intel, Nokia and Toshiba [4]. Bluetooth wireless technology is a short-range communication system to replace the cables that connect electronic devices. The key features of Bluetooth wireless technology are robustness, low power, and low cost. The Bluetooth consists of RF transceiver which operates in unlicensed Industrial, Scientific, Medical (ISM) band at 2.4 GHz to 2.485 GHz. Bluetooth utilizes spread spectrum and frequency hopping or frequency changing sequences at 1600 hops/sec [5]. Range of Bluetooth transmission is divided into three classes [2].

Table 1: Classes of Bluetooth range

Class	**Range**
Class 3	1 meter or 3 feet
Class 2	10 meters or 30 feet
Class 1	100 meters or 300 feet

The system offers services that enable connection between devices and exchange of data between varieties of classes [2]. Piconet is defined by the Bluetooth specification as an ad-hoc and spontaneous clustering of Bluetooth devices. If there are more than one Bluetooth devices, one will act as a master while the other act as slaves. Bluetooth can support up to seven slaves

[5]. Bluetooth protocol stack is divided into three groups; Transport Protocol Group, Middleware Protocol Group and Application group. Transport protocol group supports both asynchronous and synchronous transmission. The Middleware protocol group allows interoperability between existing and new applications over Bluetooth links. The Application group comprises legacy application and also Bluetooth-ware applications [5].

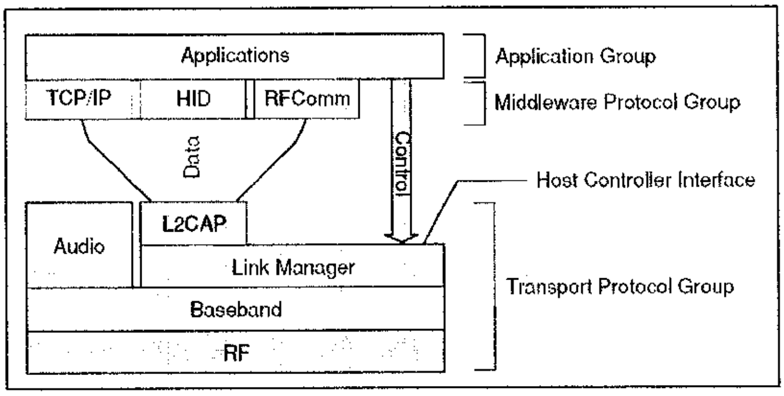


Figure 1: Bluetooth Core Protocol Groups

2.2 Biometric Technology

Security is the biggest challenge in the world today. In line with the modern world, biometric technology is introduced to overcome the security issues. Biometrics authenticates a person based on his/her unique physical body demographics or behavioural characteristics. In other words, a person can be considered as the password. In security sector, biometric technology is integrated with other authentication applications and technologies such as domain access, single sign-on, smart cards, encryption, remote access, and digital signatures. In biometric identification systems, there are several characteristics of physical and behavioural traits required. These include the uniqueness, universality, permanence and measurability [6]. Examples of physical characteristics include fingerprints, eye retinas and irises. Signature, gait and typing patterns fall under behavioural traits.

A basic biometric system consists of a sensor to record the biometric trait, a computer to process and to save the biometric trait and also an application to authenticate the user [6]. The main advantage of biometrics technology is its high assurance in term of security. It is the solutions of tremendous issues faced regarding identification by using passwords.

2.3 Fingerprint

Fingerprint is the oldest biometric signs of identity [7]. A Fingerprint has five main characteristics that made it suitable to be used as biometric method to identify individuals. The characteristics are

- Permanent, whereby it remains the same throughout lifetime.
- Unique, a fingerprint for each individual is different from other individual. The following is the general idea on how a fingerprint sensor works.
- A user places his/her finger on the fingerprint scanning area.
- The fingerprint sensor captures the user's fingerprint and puts it into a template form using a mathematical algorithm
- The sensor compares it to its database or storage of fingerprints to determine if it matches any existing fingerprints. If the fingerprint device finds a match then access is granted [8].

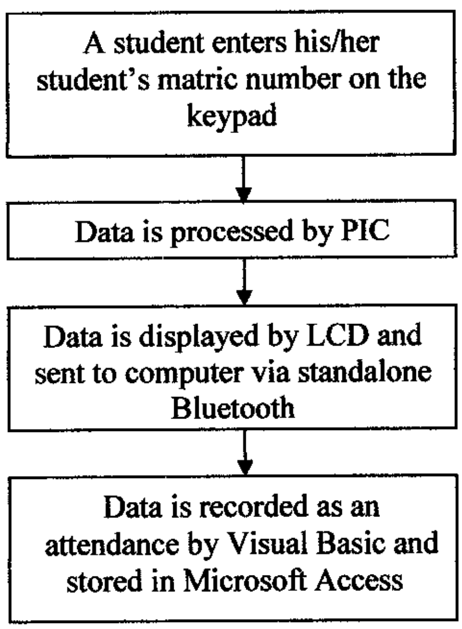
CHAPTER 3

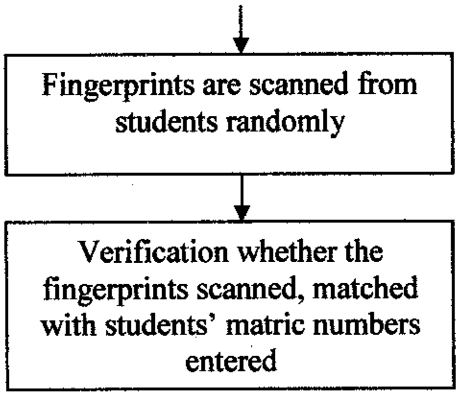
METHODOLOGY

3.1 Tools and Components

This project is divided into two parts: transmitter and receiver. For the transmitter, four main components are involved: a keypad, a PIC, an LCD display and a stand alone Bluetooth. For receiver part, the main components are a Bluetooth adapter, a computer and program to interpret the data received from the transmitter. The software used is the Microsoft Visual Basic 6.0. Apart from interpreting the data received from the transmitter, this software is also used for verification purpose. This project is very significant because fingerprint verification is done to ensure the student's matric number is entered only by the designated student. This can absolutely eliminate any attempts of cheating.

Below is the flow diagram of this project.





3.1.1 PIC 16f877 from Microchip

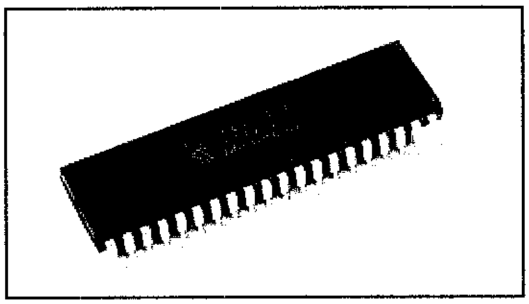


Figure 2: PIC 16f877

PIC is a Reduced Instruction Set Computer (RISC) microprocessor. PIC is perfect for this project since the power consumption is low. Below is the specification for PIC 16f family [9].

Table 2: PIC 16f877 Specifications

Input voltage	2.0V to 5.5V
Low power consumption	< 0.6 mA typical @ 3V, 4 MHz
Total number of pins	40 (33 I/O pins)
High sink / source current	25mA
Standby current	< 1 µA
Memory	Up to 8K x 14 words of FLASH Program Memory Up to 368 x 8 bytes of Data Memory (RAM) Up to 256 x 8 bytes of EEPROM Data Memory

The following depicts the pins diagram for PIC 16f877

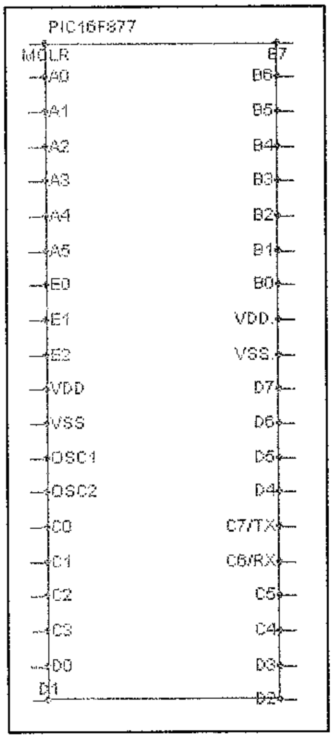


Figure 3: PIC 16f877 Pins Diagram

3.1.2 4x4 Keypad from Farnell

This keypad has input 16 keys (4 columns and 4 rows) and eight output pins. The output pins are connected to the port D of PIC.

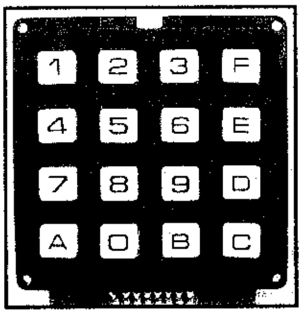


Figure 4: 4x4 Keypad

Table 3: 4x4 Keypad Specifications

Input Voltage	500 mV to 24 V
Input Current	10 mA to 20 mA
E.D.S. protection	5000 V
Operating temperature	-25 to +85°C
Operating force	1.2N ±35%

3.1.3 1 x 16 LCD by Philips

This LCD displays the inputs that have been keyed in on keypad. It has one line of 16 characters.

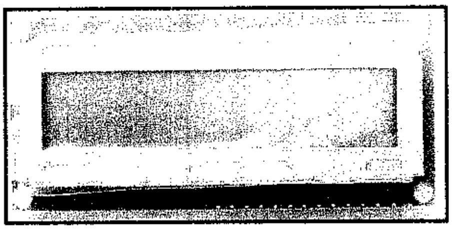


Figure 5: 1x16 LCD

Table 4: LCD Pins Function

Pin	Function	Description	Pin	Function	Description
1	Vss	Ground	8	D1	Data bit 1 (8 bit)
2	Vdd	+5ve supply	9	D2	Data bit 2 (8 bit)
3	Vee	Contrast	10	D3	Data bit 3 (8 bit)
4	RS	Register Select	11	D4	Data bit 4
5	R/W	Read/Write	12	D5	Data bit 5
6	E	Enable	13	D6	Data bit 6
7	D0	Data bit 0 (8 bit)	14	D7	Data bit 7

3.1.4 U.are.U 4000 Fingerprint Sensor

There are many types of fingerprint sensors available in the market. Upon all the sensors, DigitalPersona U.are.U 4000 Fingerprint Sensor is chosen for this project. This sensor is a USB driven sensor. It has a slim design and small form factor. Other distinguish features of this sensor are; on-board electronics control image capture, latent fingerprint rejection, self-calibration, and the Plug-n-Play USB interface [10].

The sensor works in a very convenient way. First, a user just needs to place his or her finger on the glowing reader window. The reader quickly and automatically scans the fingerprint. On-board electronics calibrate the reader and encrypt the scanned data before sending it over the USB interface [10]. This user friendly sensor can authenticate even the most difficult fingerprints accurately and rapidly regardless of placement angle of the fingerprint. For more details please refer to Appendix A.

BioKIT SDK (Software Development Kit) is fingerprint software used for this project. This software enables users to integrate fingerprint authentication into Windows applications.

Table 5: Specifications for U.are.U 4000 Fingerprint Sensor

Specifications
Pixel resolution: - 512 dpi (average x,y over the field) Image capture area : - 6 mm (nominal width at center)18.1 mm (nominal length) 8-bit grayscale (256 levels of gray) Sensor size: approximation - 79 mm x 49 mm x 19 mm Compatible with: - USB specification 1.0, 1.1, 2.0

3.1.5 Bluetooth

Two Bluetooth devices are needed for this project. One will act as a master which initiates the connection and the other one will act as a slave and responses to the connection made by the Bluetooth master.

For the Bluetooth master, it is connected to the PC as its host. A normal Bluetooth USB adapter has been chosen as the Bluetooth master. The Bluetooth adapter used for this project is equipped with software named IVT BlueSoleil. This is generic software that can be associated with other Bluetooth adapters. The main purpose for this software is to detect new Bluetooth devices that are available within the specified ranges.

Table 6: Specifications for Bluetooth Adapter used in this Project

Specifications
Compatible with : • Bluetooth V1.2 • USB V1.0/2.0 Operation range: • 20m (Bluetooth Class 2) Frequency band : • 2.4 GHz unlicensed ISM Band Maximum data rate: • 1MB

In this project, the Bluetooth slave is connected with the fingerprint sensor. However, the data from the fingerprint sensor is not transmitted through it because there is no data communication between both devices. Therefore, a stand-alone Bluetooth is required.

A stand–alone Bluetooth by KC Wirefree has been chosen as the Bluetooth slave. KC Wirefree Bluetooth transceivers are well developed and designed to be robust, flexible and reliable product. All KC Wirefree products are suitable for Original Equipment Manufacturer (OEM) development, manufacturing and also for students' research and development work. The products are Bluetooth V1.2 Compliant [11]. The Bluetooth module selected from KC Wirefree is KC-21 module. Please refer to Appendix B for KC-21 Datasheet

Table 7: Specifications for KC-21

Specifications
• Supply Power:3.3V • Operation range : 20m (Bluetooth Class 2) • High-speed data rates up to 921K baud (UART) • Miniature solution 15mm by 27mm • All Bluetooth data rates (57.6 to 723.2 Kb/sec) • 11 general purpose I/O • Customized firmware • Point-to-point and point-to-multipoint capability • Single chip solution • Host Controller Interface (HCI) and Serial Port Profile (SPP) firmware available

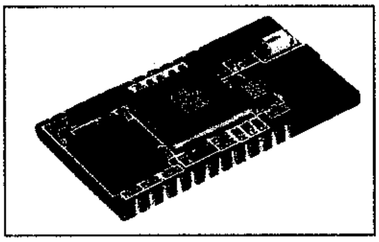


Figure 6: KC-21

3.2 A Review on Previous Project

From the existing system developed by Ms Asiah binti Hanapi entitled project "Students Class Attendance Record Using Fingerprint Identification Sensor," the systems can be enhanced on two areas

- Adding wireless features
- Implementation in real systems – UTP lecture halls

3.3 Designing the Circuit

The circuit design is only required for transmitter part whereas on receiver part, software design is required. For circuit design on the transmitter part, two main aspects are considered: the power supply circuit and also the connection between each component.

For the power supply circuit, the output of the circuit should be +5V and +3.3V. +5V output is used to power up the PIC and also the LCD display while +3.3V output is used to power up the KC-21, the stand alone Bluetooth.

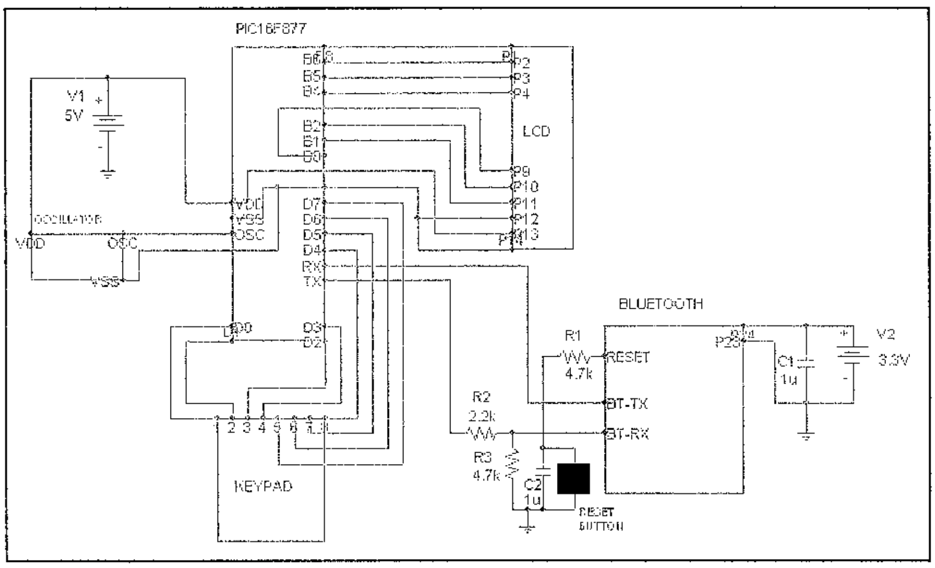


Figure 7: Schematic design for transmitter

The stand-alone Bluetooth is used as the transmitter. Data from PIC is sent through Tx pin, Pin C6 to the Rx pin of the Bluetooth. The stand-alone Bluetooth transmits the data and this data is received by the Bluetooth adapter connected to the PC.

Printed Circuit Board (PCB) layouts for keypad, LCD and also Bluetooth are designed in order to proceed with PCB.

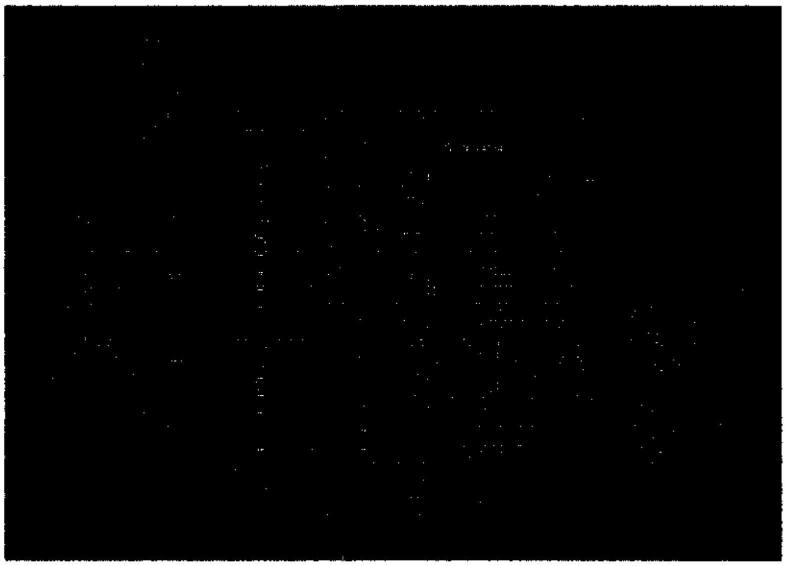


Figure 8: PCB Layout for Keypad and LCD

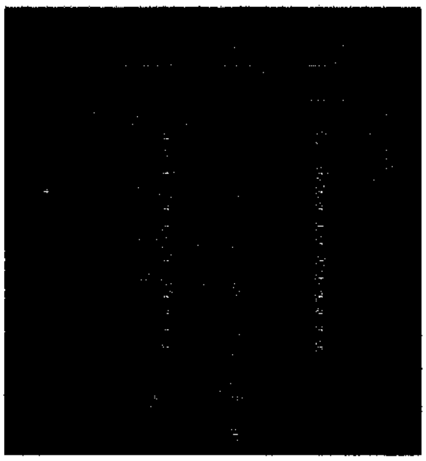


Figure 9: PCB layout for Bluetooth

Visual Basic is used to create a user interface to interpret the data received via the Bluetooth. Based on the data received, a form containing the attendance of the students is automatically created.

3.4 Constructing and Testing the Circuit

The circuit is constructed on the bread board in order to test the workability. After the circuit functioned as expected, PCB design is started followed by PCB fabrication, soldering the components on the PCB and circuit testing.

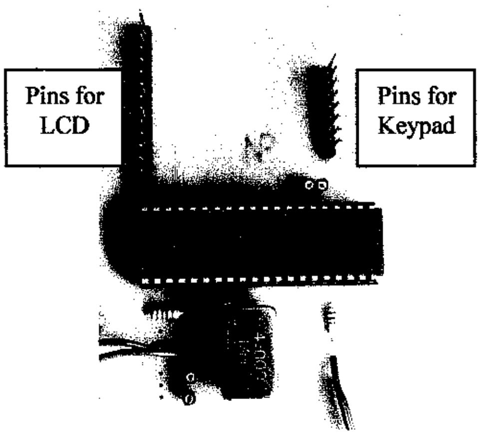


Figure 10 : Keypad and LCD Circuit

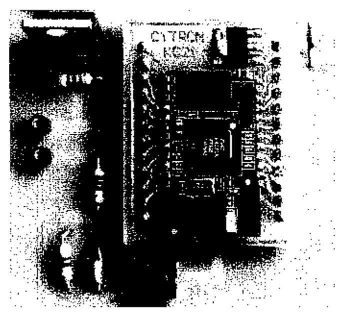


Figure 11: Bluetooth Circuit

Testing is done step by step. First testing is to check the output at LCD. This follows by checking the output from the PIC through its transmitter (Tx) pin.

3.5 Implementing in Real Application– UTP Lecture Halls

This project is targeted to be implemented on the real application, which is in the UTP lecture halls.

CHAPTER 4

RESULTS AND DISCUSSIONS

4.1 Results

4.1.1 Overall Design during First Semester and its Defects

At the initial stage, this project focused on adding a wireless feature to an existing fingerprint sensor by using Bluetooth. The fingerprint sensor was connected to a Bluetooth adapter via its data cables. This Bluetooth adapter acted as a slave Bluetooth. Besides that, a power supply circuit was constructed in order to supply power to the fingerprint sensor and to the Bluetooth adapter. The data which was the image of the fingerprint was supposed to be transmitted via Bluetooth slave and received by another Bluetooth adapter which was connected to the computer. This Bluetooth adapter acted as a Bluetooth master. Then, the data was expected to be displayed on the computer. Below is the overall design in the initial stage.

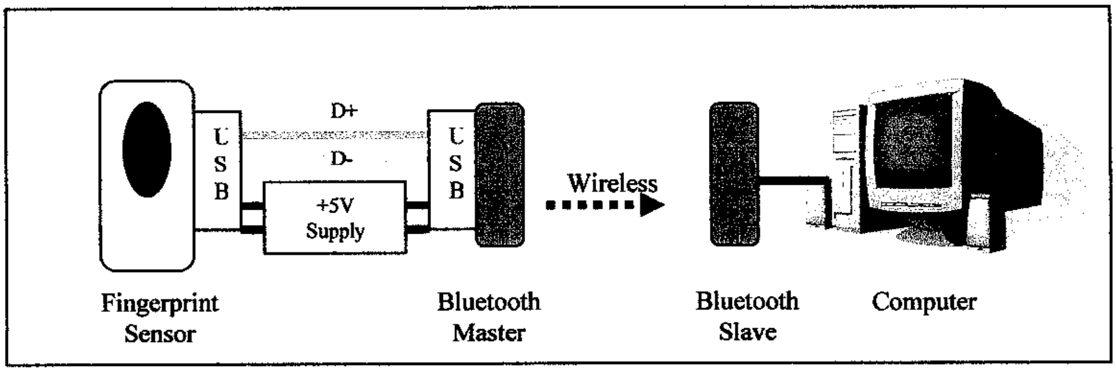


Figure 12: Overall Design in Initial Stage

However, the results were not as expected. The Bluetooth master was not able to detect Bluetooth slave which indicated that there was no communication between Bluetooth master and Bluetooth slave. Thus, there was no data received by the computer.

After a very detail investigation, defects on the overall design were found. First defect was regarding the fingerprint sensor. This fingerprint is a slave device. A slave device needed as master device to detect it so that it can function correctly. Since the sensor is a USB driven device, the USB master devices available are devices that have their own operating systems such as a computer, or a Personal Digital Assistant (PDA). There are a few stand alone USB master devices produced by Cambridge Silicon Radio (CSR) and from Cypress Semiconductor Corporation but there are very costly and the construction was meant for industry and not for research purpose. For Bluetooth slave, it needed to be loaded with software for it to function and thus detected by the Bluetooth master which is connected to the computer. Thus, a stand alone Bluetooth which is loaded with firmware is required. For this project, a stand alone Bluetooth which is produced by KC Wirefree is used.

4.1.2 Change from Wireless Fingerprint Sensor to Wireless Keypad

As mentioned earlier, it is very difficult to create a wireless fingerprint sensor. Thus a wireless keypad is constructed as an alternative to the original idea.

4.1.3 Testing the Output from LCD

For this testing, the output is displayed correctly at the LCD. Please refer to Appendix C for coding on keypad and LCD.

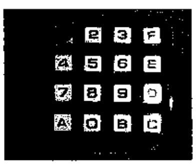
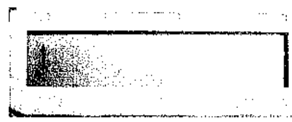
	

Figure 13 : Key Number 1 is pressed	Figure 14: Number 1 is displayed

4.1.4 Checking data transmission through Tx pin of PIC via Hyper Terminal

An additional circuit with MAX 232 is then constructed so that the data transmitted from Tx can be seen on the HyperTerminal. This circuit is connected directly to the computer through RS232.

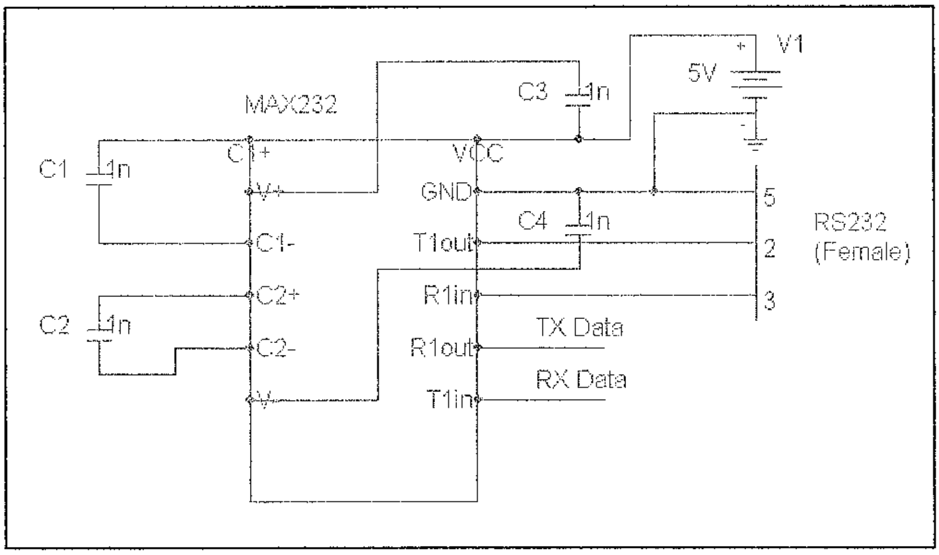


Figure 15: Circuit to connect PIC to the computer through RS232

Below is the output on the Hyper Terminal. The output is displayed as expected.

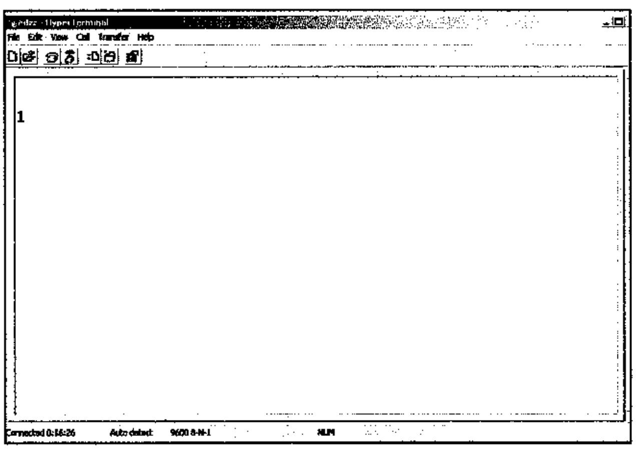


Figure 16: Output at Hyper Terminal when key 1 is pressed

4.1.5 Communication with Bluetooth

After the data is successfully sent when the circuit is directly connected to the computer, wireless connection is implemented. PIC is now connected to the stand alone Bluetooth instead of MAX-232. A Bluetooth adapter is attached with the computer in order to establish a wireless connection with the stand alone Bluetooth.

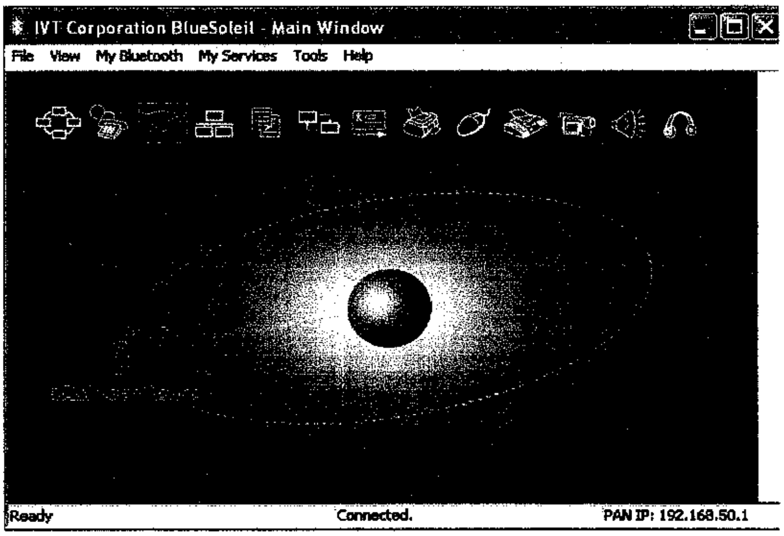


Figure 17: Communication between the stand alone Bluetooth and the Bluetooth adapter is established

4.1.6 Checking wireless data transmission between the stand alone Bluetooth and the Bluetooth adapter via Hyper Terminal.

Figure 16 shows the output on the Hyper Terminal from wireless transmission. The output is displayed at Hyper Terminal is as expected.

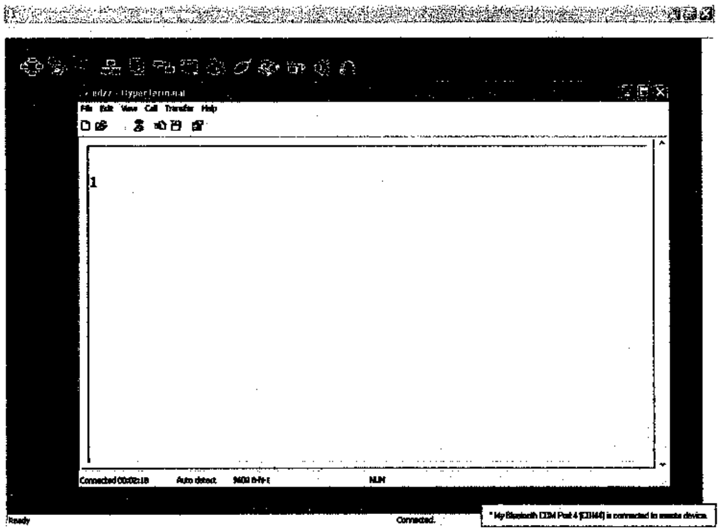


Figure 18: Output at Hyper Terminal when key 1 is pressed

4.1.7 Graphic User Interface using Visual Basic 6.0

Visual Basic is the software that is used to create the GUI students' attendance system. The main window contains two main parts which are for students' registration and also for recording attendance.

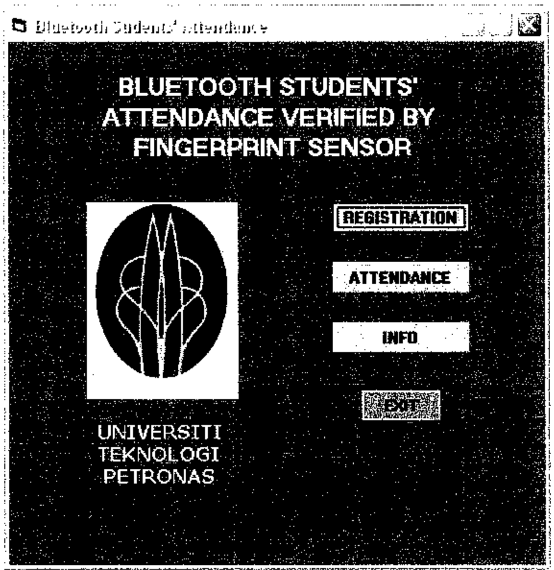


Figure 19: Main Window

4.1.7.1 Registration

For registration part, the students' are required to register their particulars including names, students' matric numbers and programme. Besides, the students are also required to register their fingerprint for attendance verification.

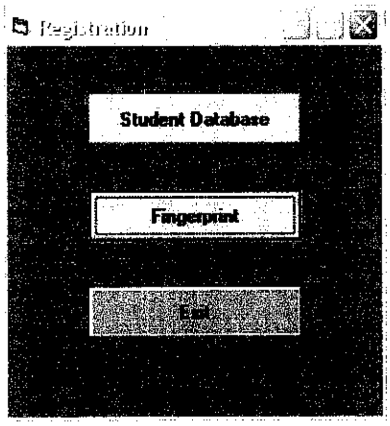


Figure 20: Registration Window

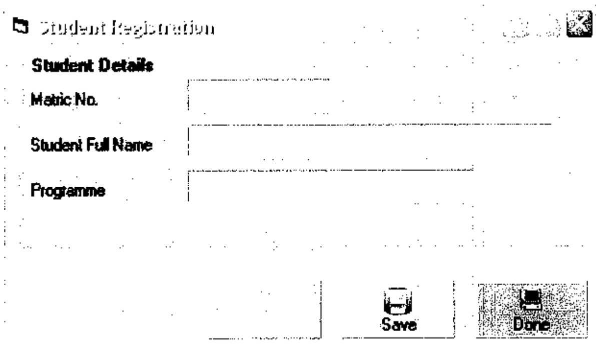


Figure 21: Registration for Student's Matric Number, Name and Programme

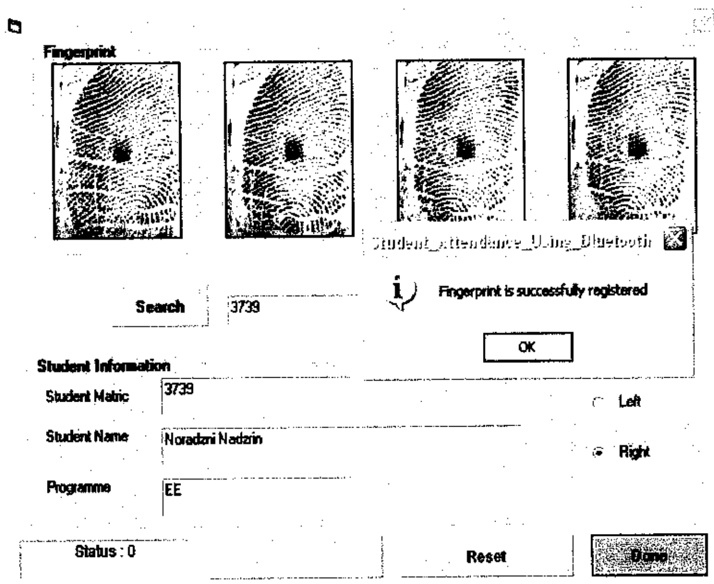


Figure 22: Registration for Fingerprint

4.1.7.2 Recording Attendance

To record the attendance, there are two ways of recording attendance, by students' matric numbers and also fingerprint.

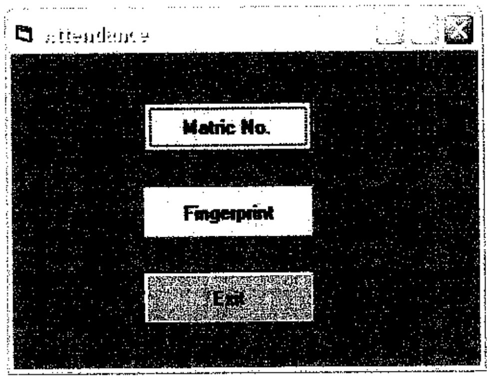


Figure 23: Attendance Window

For recording attendance via students' matric numbers, the ASCII characters that have been sent via Bluetooth to the computer are interpreted as the students' matric numbers. When the students' matric numbers entered matched with the database, the attendance is taken. The "attendance" column for the particular students is automatically ticked and it is indicated by "-1".

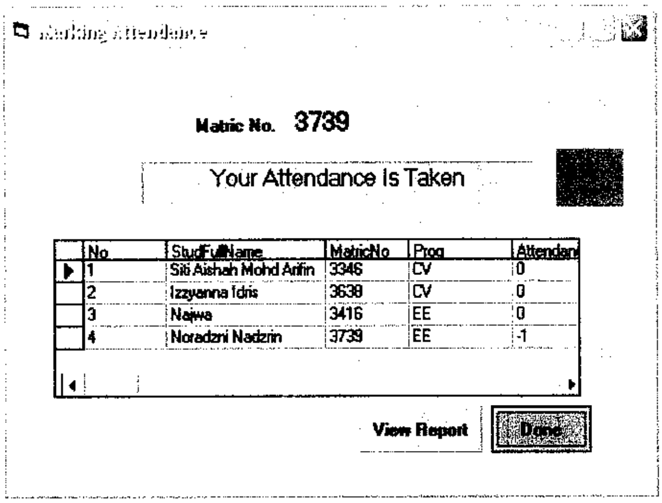


Figure 24: Attendance is taken

Random fingerprint verifications are done after all students have entered their matric numbers. Verified fingerprints are automatically ticked in "FPVerification" column in the database.

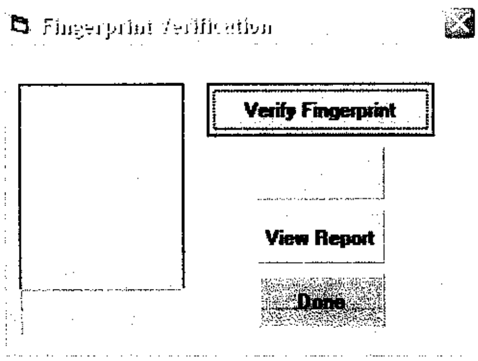


Figure 25: Fingerprint Verification

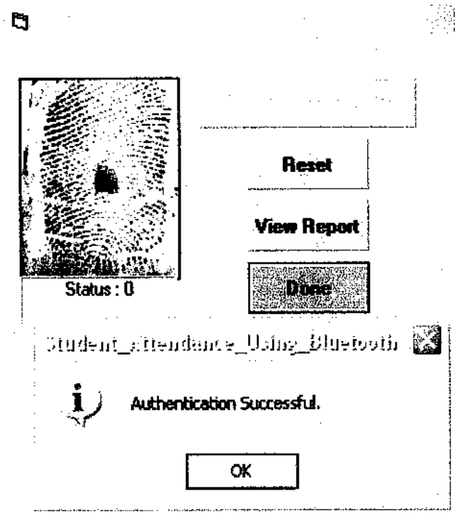


Figure 26: Authentication Successful

4.1.7.3 Microsoft Access

The entire database is automatically recorded in the Microsoft Access. The attendances of the students are automatically ticked after the attendances are successfully recorded and matched with the database.

No	StudFullName	MatricNo	Prog	Attendance	TimeIn	RightFingerPrint	FPVerification
1	Siti Aishah Mohd Arifin	3346	CV	☐		622599651902FE	☐
2	Izzyanna Idris	3638	CV	☐		622599651902FE	☐
3	Najwa	3416	EE	☐		622599651902FE	☐
4	Noradzni Nadzrin	3739	EE	☑	12/2/2006 7:40:34	622599651902FE	☑
er)				■			■

Figure 27: Students Database

4.1.7.4 Print and Save the Attendance Report

A user can choose either to print or to save the attendance report. When "View Report" button is clicked, the user can choose to print the report by clicking the "print" icon, or to save it by clicking the "export" icon. If "export" icon is clicked, the user needs to enter the <filename.doc> and save it as Text(*.txt). The report can be viewed in Microsoft Word

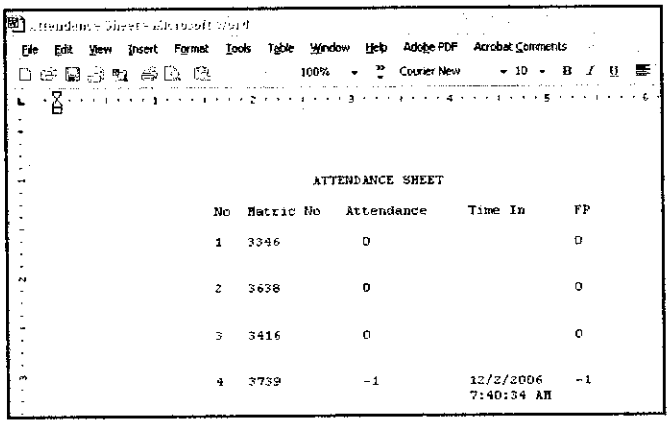


Figure 28: Attendance Report in Microsoft Word

4.2 Discussion

Both Bluetooth devices used in this project must be installed with their particular software. For Bluetooth master, since it is connected to the PC, the software can easily be loaded in the PC. However, for the Bluetooth slave, it needs its own firmware and shall be built-in the Bluetooth itself. For this project, KC-21 has been selected as the most appropriate stand alone Bluetooth module. This Bluetooth module is embedded with the firmware. 16 keys keypad is chosen as alternative to the fingerprint sensor since construction of wireless fingerprint is very difficult due to constraints in term of cost and technical support.

CHAPTER 5

CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

Two semesters that has been allocated to complete the project entitled "Innovation of Fingerprint Sensor Supports Wireless Bluetooth Technology". During the first semester, this project focused on enhancement for the project entitled "Students Class Attendance Record Using Fingerprint Identification Sensor," by adding in a wireless feature to the fingerprint sensor. However, after thorough research on the first semester, it is agreed that is it very difficult to create a wireless fingerprint sensor due to cost and technology constraints. Thus, the title has been changed to "Bluetooth Students' Attendance Verified by Fingerprint Sensor," whereby a wireless keypad is constructed instead of wireless fingerprint. Finally, an advance, convenient and more reliable system for recording students' attendance is created. This project would be very useful in various aspects since wireless feature is added and definitely eases and systematizes the students' attendance system.

5.2 Recommendations

This project can be improved in term of circuit and also the software design. Below are the recommendations for this project.

- Use advanced battery to produce appropriate power for long period.
- Use components that require less power
- Add indications such as LED and LCD commands upon any confirmations

REFERENCES

[1] http://www.chozamtech.com/security/biometrics/biometrics.html

[2] http://www.bluetooth.com

[3] Hanafi A. (2005) Students Class Attendance Record using Fingerprint Identification Sensor. Universiti Teknologi PETRONAS

[4] Bakker,D.M.,Gilster D. M. and Gilster, R. (2002). Bluetooth ™ End to End ™. 1st Edition. Hungry Mind™.

[5] McDermott-Wells, P.What is Bluetooth? Potentials, IEEEVolume 23, Issue 5, Dec 2004-Jan 2005 Page(s):33 - 35

[6] http://www.bromba.com/faq/biofaqe.htm

[7] Jain L.C.,Halici U.,Hayashi I.,Lee S.B., and Tsutsui S.,(1999). Intelligent Biometric Techniques in Fingerprint and Face Recognition. CRC Press LLC

[8] http://www.fingerprints.tk/

[9] Microchip 16f8PIC16F87X Data Sheet 28/40-Pin 8-Bit CMOS FLASH Microcontrollers

[10] http://www.mybiobase.com/uru.htm

[11] KC Wirefree Bluetooth User's Manual (2005). Version 1.4

APPENDICES

APPENDIX A

FINGERPRINT SENSOR (U.ARE.U ® 4000B) DATASHEET

U.are.U® 4000B Reader

USB Fingerprint Reader

Target Applications

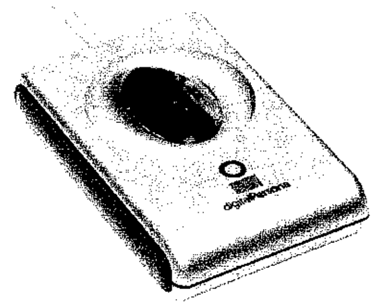
- Desktop PC security
- Mobile PCs
- Custom applications

Features

- Superior ESD resistance
- Small form factor
- Excellent image quality
- Encrypted fingerprint data
- Latent print rejection
- Counterfeit finger rejection
- Rotation invariant
- Rugged
- Works well with dry, moist, or rough fingerprints
- Compatible with Windows® XP, 2000, Me, 98, NT® 4.0 and Windows Server 2000, 2003

Key Specifications

- Pixel resolution: 512 dpi (average x, y over the scan area)
- Scan capture area: 14.6 mm (nom. width at center) 18.1 mm (nom. length)
- 8-bit grayscale (256 levels of gray)
- Reader size (approximate): 79 mm x 49 mm x 19 mm
- Compatible with USB 1.0, 1.1 and 2.0 (Full Speed) specifications
- Indoor, home and office use



Product Description

The U.are.U 4000B Reader is a USB fingerprint reader designed for use with Digital Persona, Inc.'s enterprise software applications and developer tools.

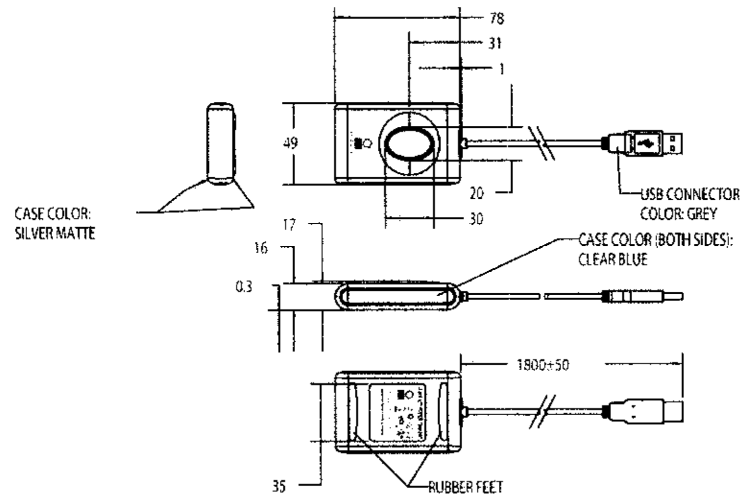
The user simply places their finger on the glowing reader window, and the reader quickly and automatically scans the fingerprint. On-board electronics calibrate the reader and encrypt the scanned data before sending it over the USB interface.

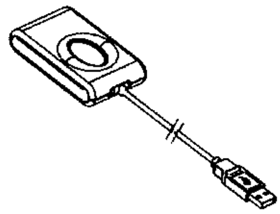
Digital Persona readers utilize optical fingerprint scanning technology to achieve excellent image quality, a large capture area and superior reliability. The U.are.U 4000B Reader and DigitalPersona® Fingerprint Recognition Engine have an unmatched ability to authenticate even the most difficult fingerprints accurately and rapidly regardless of placement angle.

The U.are.U 4000B Reader can be purchased with DigitalPersona Pro Workstation, DigitalPersona Pro Kiosk, DigitalPersona Online or DigitalPersona Integrator packages. Whether you are an enterprise customer or a system integrator, Digital Persona's fingerprint authentication solutions provide a natural extension to your security system and applications.

digitalPersona.

Mechanical Specifications





Ratings

Supply Voltage	5.0V ±5% supplied by USB
Supply Current—scanning	190 mA (Typical)
Supply Current—idle mode	140 mA (Typical)
Supply Current—suspend mode	1.5 mA (Maximum)
ESD Susceptibility	>15 kV, mounted in case
Temperature, Operating	0 - 40 C
Humidity, Operating	20% - 80% non-condensing
Temperature, Storage	-10 - 60 C
Humidity, Storage	20% - 90% non-condensing
Scan data	8-bit grayscale
Standards Compliance	FCC Class B, CE, ICES, BSMI, MIC, USB, WHQL

Data subject to change without notice.

digitalPersona.

Digital Persona, Inc.
720 Bay Road, Suite 100
Redwood City, CA 94063
USA

Tel: +1 650.474.4000
Fax: +1 650.298.8313
E-Mail: info@digitalpersona.com
Web: www.digitalpersona.com

APPENDIX B

KC-21 DATASHEET

KC Wirefree

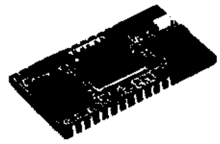
KC21 Bluetooth Class 2 OEM Module

Datasheet

Page 1 of 4

Features

- Bluetooth v1.2 specification
- Complete RF ready module
- Class 2 radio
- High-speed data rates up to 921K baud
- High-performance ARM7 processor up to 48MHz
- High-security 128-bit encryption
- Extended range up to 30m
- Miniature solution 15mm by 27mm
- All Bluetooth data rates (57.6 to 723.2 Kb/sec)
- SPI interface, up to 24MHz
- 11 general purpose I/O
- AT command set
- Dynamic configuration
- Low power sniff mode
- Point-to-point and point-to-multipoint capability
- Multiple device bonding
- USB 2.0 compatible
- CPU, radio, antenna, & firmware on module
- Zeevo 4002 and 4301 Bluetooth chip solutions



Bluetooth

KC21
Bluetooth Module

Description

The KC21 is a surface mount PCB module with an integrated chip antenna that provides ready to use Bluetooth wireless technology. The conveniently pre-programmed flash device contains firmware for serial cable replacement using the Bluetooth SPP profile per the Bluetooth v1.2 specification. Customized firmware is easily pre-loaded into these fully tuned and tested modules so that they are ready to install without any further procedures. The KC21 is a Class-2 Bluetooth Module supplied on a 24 pin surface mount 6 layer PCB with a miniature footprint.

Typical Cable Replacement Applications

Serial communications
Machine diagnostics and control
Point-of-sale data collection
Remote sensing
Medical device communications
Industrial control
Home automation

KC Wirefree

KC21 Bluetooth Class 2 OEM Module

Datasheet

Page 2 of 4

Software Architecture

Lower Layer Stack

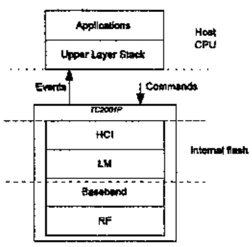
- Full Bluetooth data rate (723.3kbps maximum)
- All ACL (Asynchronous Connection Less) packet types (DM1, DH1, DM3, DH3, DM5, DH5, AUX1)
- SCO (Synchronous Connection Oriented) packet types (HV1, HV2, HV3)
- Point to multipoint and scatternet support—3 master and 7 slave links allowed (10 active links simultaneously)
- Master slave switch—supported during connection and post connection
- Authentication and encryption—encryption key length from 8-bits to 128-bits maximum
- Park, sniff, and hold modes—fully supported to maximum allowed intervals, approximately 50 parked slaves
- Dedicated Inquiry Access Code—for improved inquiry scan performance
- Dynamic packet selection—channel quality driven data rate to optimize link performance
- Dynamic power control—interference reduction and link performance optimization for all device classes
- Bluetooth test modes—per Bluetooth 1.2 specification
- Device power modes—active, sleep and deep sleep
- 802.11b co-existence—AWMA and AFH
- Persistent FLASH memory—for BD Address and radio parameter storage
- Wake on Bluetooth feature—optimized power consumption of host CPU
- SCO over UART, PCM, or SPI interface—application flexibility for host CPU
- Vendor specific HCI commands—to support device configuration and certification test modes

Upper Layer Stack

- SDAP, GAP, SPP, and DUN protocols
- RFComm, SDP, and L2CAP supported

HCI Interface

- Bluetooth 1.2 specification compliant
- HCI USB transport layer (H2)
- HCI UART transport layer (H4)
- Firmware upgrade over UART



HCI Stack

Hardware Specifications

Absolute Maximum Ratings

Rating	Min	Typical	Max	Unit
Storage temperature range	-40	-	+150	°C
Supply voltage, V_{DD}	-0.3	-	+ 3.6	Volts
RF input power	-	-	-5	dBm

Recommended Operating Conditions

Rating	Min	Typical	Max	Unit
Operating Temperature Range	-20	-	85	°C
Supply Voltage V_{DD}	2.7	3.3	3.6	Volts
Signal Pin Voltage	-	3.3	-	Volts
RF Frequency	2400	-	2483.5	MHz

Current Consumption

General Conditions: V_{DD}= 3.3V, temperature = 25 °C, frequency = 2.402 - 2.480 GHz, 50 Ω antenna, 12 MHz ext crystal, and 32 KHz ext sleep crystal.

Modes	Avg	Unit
ACL data 115K Baud UART at max throughput (Master)	35.0	mA
ACL data 115K Baud UART at max throughput (Slave)	35.0	mA
Connection, no data traffic, master	19.0	mA
Connection, no data traffic, slave	29.0	mA
SCO Connection, HV1, master	53.0	mA
SCO Connection, HV1, slave	53.5	mA
SCO Connection, HV3, master	32.0	mA
SCO Connection, HV3, slave	35.0	mA
Connection in sniff (Tsniff=100ms), no data traffic, master	9.2	mA
Connection in sniff (Tsniff=100ms), no data traffic, slave	9.8	mA
Connection in sniff (Tsniff=375ms), no data traffic, master	1.75	mA
Connection in sniff (Tsniff=375ms), no data traffic, slave	2.50	mA
Standby, without deep sleep	16.5	mA
Standby, with deep sleep	0.160	mA
Page/Inquiry scan, deep sleep	2.1	mA
Peak current	90	mA

* All current consumption numbers include on-die regulators and flash memory

I/O Operating Characteristics (V_{DD} = 3.3 V, unless otherwise specified)

Symbol	Parameter	Min	Max	Unit	Conditions
V_{IL}	Low-Level Input Voltage	-	0.8	Volts	
V_{IH}	High-Level Input Voltage	2.0	-	Volts	
V_{OL}	Low-Level Output Voltage	-	0.4	Volts	I_{OL} = 2mA
V_{OH}	High-Level Output Voltage	2.4	-	Volts	I_{OH} = 2mA
I_{OL}	Low -Level Output Current	-	2.2	mA	V_{OL} = 0.4 V
I_{OH}	High-Level Output Current	-	3.1	mA	V_{OH} = 2.4 V
I_I	Input Leakage Current	-1	+1	uA	@V_I = 3.3V or 0V
V_{T+}	Schmitt Trigger Low-High	1.47	1.50	Volts	
V_{T-}	Schmitt Trigger High-Low	0.89	0.95	Volts	
R_{PU}	Pull-up Resistor	53	113	KΩ	Resistor Turned On
R_{PD}	Pull-down Resistor	43	118	KΩ	Resistor Turned On
C_I	Input Capacitance		7.5	pF	

USB I/O Operating Characteristics (V_{DD} = 3.3 V)

Symbol	Parameter	Min	Max	Unit	Conditions
V_{IL}	Low-Level Input Voltage	-	0.8	Volts	
V_{IH}	High-Level Input Voltage	2.0	-	Volts	
V_{OL}	Low-Level Output Voltage	-	0.3	Volts	
V_{OH}	High-Level Output Voltage	2.8	-	Volts	
C_{IN}	Input Capacitance	-	25	pF	DP or DM to GND

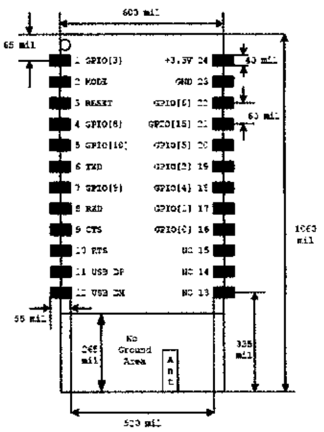
Selected RF Characteristics

General Conditions: V_{DD}= 3.3V, temperature = 25 °C, frequency = 2.402 - 2.480 GHz, 50 Ω antenna, 12 MHz ext crystal, and 32 KHz ext sleep crystal.

Parameters	Conditions	BT Spec	Typical	Unit
Antenna load			50	Ω
Sensitivity level	BER < .001 with DH5	≤ -70	-85	dBm
Maximum usable level	BER < .001 with DH1	≥ -20	-5	dBm
Input VSWR			2.5:1	
Maximum output power	50 Ω load	-6 to +4	1	dBm
Power control range		≥ 16	30	dB
Power control resolution		2 to 8	4	dB
Initial Carrier Frequency Tolerance		± 75	16	kHz
20 dB Bandwidth for modulated carrier		≤ 1000	930	kHz

Pin Assignments by Functional Grouping

Name	Type	Pin #	Description
RESET#	I	3	Reset input (active low for 5 ms); Schmidt triggered
GPIO [0]	I/O	16	General Purpose Input/Output
GPIO [1]	I/O	17	General Purpose Input/Output
GPIO [2]	I/O	19	General Purpose Input/Output
GPIO [3]	I/O	1	General Purpose Input/Output
GPIO [4]	I/O	18	General Purpose Input/Output
GPIO [5]	I/O	20	General Purpose Input/Output
GPIO [6]	I/O	22	General Purpose Input/Output
GPIO [8]	I/O	4	General Purpose Input/Output
GPIO [9]	I/O	7	General Purpose Input/Output
GPIO [10]	I/O	5	General Purpose Input/Output
GPIO [15]	I/O	21	General Purpose Input/Output
RXD	I	8	Receive data
TXD	O	6	Transmit data
CTS#	I	9	Clear to send (active low)
RTS#	O	10	Request to send (active low)
DP	I/O	11	USB data plus
DM	I/O	12	USB data minus
MODE	I	2	Reserved
V_{DD}		24	V_{DD}
GND		23	GND



KC21 Class 2 Bluetooth OEM Module Footprint

Layout Guidelines

The area around the KC21 module should be free of any ground planes, trace routings, or metal for at least 250 mil from the antenna in all directions. Traces should not be routed underneath the module.

APPENDIX C

CODING FOR KEYPAD AND LCD

Main

```
#include <16f877.h>

#USE DELAY(CLOCK=4000000) /* Using a 4 MHz clock */
#FUSES XT,NOWDT,NOPROTECT, NOPUT, NOBROWNOUT,NOLVP

#include <LCD.C>

#include <key_pad.c>


main()
{

  char k;

  lcd_init();

  while (TRUE)
  {
    delay_ms(100);

    k = get_key();

    if(k != 0)
    {
     if(k =='C')
     {
     lcd_putc('\f');
     printf("%c\n", "C");
     }
     else
     {
      lcd_putc(k);
      printf("%c\n", k);
     }
    }
 }
}
```

Keypad

```
//#byte port_d = 0x08

char get_key(void)
{
  char t;
    while (1) {
     output_d(input_d() | 0xFF);  /* set RD4 to low to scan the first row */
     output_bit(PIN_D4,0);
         if (input(PIN_D0) ==0){
            delay_us(10);
```

```
        while(input(PIN_D0) ==0)
        {
         return 'A'; /* return the ASCII code of A */
         }
        if (input(PIN_D1) ==0) {
            delay_ms(10);
        while(input(PIN_D1) ==0)
         {
         }
         return '7';  /* return the ASCII code of 7 */
         }
if (input(PIN_D2) ==0) {
delay_ms(10);
      while(input(PIN_D2) ==0)
      {
      }
return '4'; /* return the ASCII code of 4 */
}
if (input(PIN_D3) ==0) {
delay_ms(10);
      while(input(PIN_D3) ==0)
      {
      }
return '1';    /* return the ASCII code of 1 */
}
    output_d(input_d() | 0xFF);    /* set RD5 to low to scan the second row */
    output_bit(PIN_D5,0);
if (input(PIN_D0) ==0) {
delay_ms(10);
     while(input(PIN_D0) ==0)
      {
      }
      return '0';  /* return the ASCII code of 0 */
      }
if (input(PIN_D1) ==0) {
delay_ms(10);
      while(input(PIN_D1) ==0)
      {
      }
return '8';  /* return the ASCII code of 8 */
}
if (input(PIN_D2) ==0) {
delay_ms(10);
      while(input(PIN_D2) ==0)
      {
      }
return '5';     /* return the ASCII code of 5 */
      }
if (input(PIN_D3) ==0) {
delay_ms(10);
      while(input(PIN_D3) ==0)
      {
      }
return '2';     /* return the ASCII code of 2 */
    output_d(input_d() | 0xFF);     /* set RD6 to low to scan the third row */
output_bit(PIN_D6,0);
if (input(PIN_D0) ==0) {
delay_ms(10);
      while(input(PIN_D0) ==0)
      {
      }
return 'B';     /* return the ASCII code of B */
      }
```

```
if (input(PIN_D1) ==0) {
delay_ms(10);
      while(input(PIN_D1) ==0)
      {
      }
return '9';    /* return the ASCII code of 9 */
      }
if (input(PIN_D2) ==0) {
delay_ms(10);
      while(input(PIN_D2) ==0)
      {
      }
return '6';    /* return the ASCII code of 6 */
      }
if (input(PIN_D3) ==0) {
delay_ms(10);
      while(input(PIN_D3) ==0)
      {
      }
return '3';    /* return the ASCII code of 3 */
     }
    output_d(input_d() | 0xFF);      /* set RD7 to low to scan the fourth row */
    output_bit(PIN_D7,0);
if (input(PIN_D0) ==0) {
delay_ms(10);
      while(input(PIN_D0) ==0)
      {
      }
return 'C';    /* return the ASCII code of C */
      }
if (input(PIN_D1) ==0) {
delay_ms(10);
      while(input(PIN_D1) ==0)
      {
      }
return 'D';    /* return the ASCII code of D */
      }
if (input(PIN_D2) ==0) {
delay_ms(10);
      while(input(PIN_D2) ==0)
      {
      }
return 'E';    /* return the ASCII code of E */
      }
if (input(PIN_D3) ==0) {
delay_ms(10);

      while(input(PIN_D3) ==0)
      {
      }
return 'F';    /* return the ASCII code of F */
  }
 }
}
```

LCD

```
#define use_portb_lcd TRUE


struct lcd_pin_map {            // This structure is overlayed
       BOOLEAN enable;          // on to an I/O port to gain
       BOOLEAN rs;              // access to the LCD pins.
       BOOLEAN rw;              // The bits are allocated from
       BOOLEAN unused;          // low order up.  ENABLE will
       int     data : 4;        // be pin B0.
     } lcd;


#if defined(__PCH__)
#if defined use_portb_lcd
  #byte lcd = 0xF81             // This puts the entire structure
#else
  #byte lcd = 0xF83             // This puts the entire structure
#endif
#else
#if defined use_portb_lcd
  #byte lcd = 6                 // on to port B (at address 6)
#else
  #byte lcd = 8                 // on to port D (at address 8)
#endif
#endif

#if defined use_portb_lcd
  #define set_tris_lcd(x) set_tris_b(x)
#else
  #define set_tris_lcd(x) set_tris_d(x)
#endif


#define lcd_type 2         // 0=5x7, 1=5x10, 2=2 lines
#define lcd_line_two 0x40  // LCD RAM address for the second line


BYTE const LCD_INIT_STRING[4] = {0x20 | (lcd_type << 2), 0xc, 1, 6};
                    // These bytes need to be sent to the LCD
                    // to start it up.
```

```
                    // The following are used for setting
                    // the I/O port direction register.

struct lcd_pin_map const LCD_WRITE = {0,0,0,0,0}; // For write mode all pins are out
struct lcd_pin_map const LCD_READ = {0,0,0,0,15}; // For read mode data pins are in



BYTE lcd_read_byte() {
    BYTE low,high;
    set_tris_lcd(LCD_READ);
    lcd.rw = 1;
    delay_cycles(1);
    lcd.enable = 1;
    delay_cycles(1);
    high = lcd.data;
    lcd.enable = 0;
    delay_cycles(1);
    lcd.enable = 1;
    delay_us(1);
    low = lcd.data;
    lcd.enable = 0;
    set_tris_lcd(LCD_WRITE);
    return( (high<<4) | low);
}


void lcd_send_nibble( BYTE n ) {
    lcd.data = n;
    delay_cycles(1);
    lcd.enable = 1;
    delay_us(2);
    lcd.enable = 0;
}


void lcd_send_byte( BYTE address, BYTE n ) {

    lcd.rs = 0;
    while ( bit_test(lcd_read_byte(),7) ) ;
    lcd.rs = address;
```

```
   delay_cycles(1);
   lcd.rw = 0;
   delay_cycles(1);
   lcd.enable = 0;
   lcd_send_nibble(n >> 4);
   lcd_send_nibble(n & 0xf);
}


void lcd_init() {
  BYTE i;
  set_tris_lcd(LCD_WRITE);
  lcd.rs = 0;
  lcd.rw = 0;
  lcd.enable = 0;
  delay_ms(15);
  for(i=1;i<=3;++i) {
    lcd_send_nibble(3);
    delay_ms(5);
  }
  lcd_send_nibble(2);
  for(i=0;i<=3;++i)
    lcd_send_byte(0,LCD_INIT_STRING[i]);
}


void lcd_gotoxy( BYTE x, BYTE y) {
  BYTE address;

  if(y!=1)
   address=lcd_line_two;
  else
   address=0;
  address+=x-1;
  lcd_send_byte(0,0x80|address);
}

void lcd_putc( char c) {
  switch (c) {
   case '\f'  : lcd_send_byte(0,1);
              delay_ms(2);
                             break;
   case '\n'  : lcd_gotoxy(1,2);       break;
```

```
    case '\b'  : lcd_send_byte(0,0x10);  break;
    default    : lcd_send_byte(1,c);     break;
  }
}

char lcd_getc( BYTE x, BYTE y) {
  char value;

   lcd_gotoxy(x,y);
   while ( bit_test(lcd_read_byte(),7) ); // wait until busy flag is low
   lcd.rs=1;
   value = lcd_read_byte();
   lcd.rs=0;
   return(value);
}
```

APPENDIX D

CODING FOR VISUAL BASIC

Students Registration

```
Option Explicit
Dim file_name As String
Dim SQLstmt As String
Dim conn As ADODB.Connection
Dim rs As ADODB.Recordset

Private Sub cmdAdd_Click()
txtStudFullName = ""
txtProgramme = ""
txtMatric = ""
txtMatric.SetFocus

'cmdAdd.Enabled = False
cmdSave.Enabled = True
End Sub

Private Sub cmdDone_Click()
Screen.MousePointer = 0
   Unload Me

End Sub

Private Sub cmdRegister_Click()
frmRegister.Show
End Sub

Private Sub cmdSave_Click()
'Open a new connection.
If txtMatric = "" Or txtStudFullName = "" Or txtProgramme = "" Then
MsgBox "Fill in the Fields"
Else
  ' Get data from the TextBoxes/user.
  With rs
    .AddNew
    !StudFullName = txtStudFullName
    !MatricNo = txtMatric
```

```
    !Prog = txtProgramme
    .Update
  End With
    MsgBox "Done"
    cmdSave.Enabled = False
End If
cmdAdd.Enabled = True
End Sub

Private Sub Form_Load()
file_name = App.Path & "\Attendance.mdb"

 Set conn = New ADODB.Connection

   conn.CursorLocation = adUseClient

   conn.ConnectionString = _
      "Provider=Microsoft.Jet.OLEDB.4.0;" & _
      "Data Source=" & file_name & ";" & _
      "Persist Security Info=False"
   conn.Open
'  Once this connection is open you can use it throughout your application
   SQLstmt = "SELECT * FROM Student"
'  Get the records.
   Set rs = New ADODB.Recordset
   rs.Open SQLstmt, conn, adOpenStatic, adLockOptimistic, adCmdText

cmdAdd.Enabled = False
End Sub
```

Fingerprint Registration

```
Option Explicit
Dim WithEvents RegSample As BioKit.Reg          'SDK Instruction
Dim file_name As String
Dim SQLstmt As String
Dim conn As ADODB.Connection
Dim rs As ADODB.Recordset
Dim txtfingerprint As String
```

```
Private Sub cmdAbort_Click()
   RegSample.Abort                     'SDK Instruction
   'eleminate biokit's instance before can create another new instance
   Set RegSample = Nothing

   cmdStart.Enabled = True
   cmdAbort.Enabled = False
End Sub

Private Sub cmdDone_Click()
   Screen.MousePointer = 0
   Unload Me
   frmAdmin.Show
End Sub

Private Sub cmdSearch_Click()
AdoFinger.Refresh
Do Until frmRegister.AdoFinger.Recordset.EOF
If frmRegister.AdoFinger.Recordset.Fields("MatricNo").Value = txtStudMatr.Text Then
'lblStat.Caption = "Done"
MsgBox "ID is successfully entered", vbInformation
Exit Sub

Else

frmRegister.AdoFinger.Recordset.MoveNext
lblStat.Caption = "Not found"
"MsgBox "Invalid Search Key Entered", vbCritical, "Search Error"
"clearAndFocus
End If

Loop

End Sub

Private Sub cmdStart_Click()
   Dim i As Integer
   'create new instance of biokit
   Set RegSample = New BioKit.Reg
   Dim res As StatusEnum                'SDK Instruction

   RegSample.Initialize                 'SDK Instruction
```

```
  RegSample.SampleWidth = Image1(0).Width          'SDK Instruction
  RegSample.SampleHeight = Image1(0).Height        'SDK Instruction
  res = RegSample.ActivateSensor(Me.hWnd)          'SDK Instruction

  txtfingerprint = ""
  For i = 0 To 3
    Image1(i).Picture = LoadPicture()
  Next i
  RegSample.Execute                     'SDK Instruction
  cmdStart.Enabled = False
  cmdAbort.Enabled = True
End Sub


Private Sub DataGrid1_Click()
With rs
   txtStuName.Text = !StudFullName
   txtStudMat.Text = !MatricNo
   txtProg.Text = !Prog
   End With
End Sub

Private Sub Form_Load()
file_name = App.Path & "\Attendance.mdb"

 Set conn = New ADODB.Connection

 conn.CursorLocation = adUseClient

conn.ConnectionString = _
      "Provider=Microsoft.Jet.OLEDB.4.0;" & _
      "Data Source=" & file_name & ";" & _
      "Persist Security Info=False"
conn.Open
 'Once this connection is open you can use it throughout your application
   SQLstmt = "SELECT * FROM Student"
'  Get the records.
  Set rs = New ADODB.Recordset
  rs.Open SQLstmt, conn, adOpenStatic, adLockOptimistic, adCmdText

  Set DataGrid1.DataSource = rs
   With rs
    .MoveFirst
```

```
  End With
cmdStart.Enabled = True
cmdAbort.Enabled = False
txtStudMat.Text = ""
txtStuName.Text = ""
txtProg.Text = ""
End Sub

Private Sub Option1_Click()

End Sub

Private Sub RegSample_RegDone(HexFingerPrint As String, Status As BioKit.StatusEnum)    'SDK Instruction
  If Status = StatusOK Then                    'SDK Instruction

  If OptRight.Value = True Then
    If AdoFinger.Recordset!RightFingerprint <> "" Then
     If MsgBox("Right finger already registered. Do you want to replace it?", vbOKCancel, "Save") = vbOK Then
      AdoFinger.Recordset!RightFingerprint = HexFingerPrint
        AdoFinger.Recordset.Update
      MsgBox "Fingerprint is successfully registered", vbInformation
      End If
    Else
      AdoFinger.Recordset!RightFingerprint = HexFingerPrint
      AdoFinger.Recordset.Update
      MsgBox "Fingerprint is successfully registered", vbInformation
    End If
  Else
   If AdoFinger.Recordset!LeftFingerPrint <> "" Then
   If MsgBox("Left finger already registered. Do you want to replace it?", vbOKCancel, "Save") = vbOK Then
       AdoFinger.Recordset!LeftFingerPrint = HexFingerPrint
       AdoFinger.Recordset.Update
      MsgBox "Fingerprint is successfully registered", vbInformation
     End If
   Else
     AdoFinger.Recordset!LeftFingerPrint = HexFingerPrint
     AdoFinger.Recordset.Update
     MsgBox "Fingerprint is successfully registered", vbInformation
   End If
  End If

  Else
    MsgBox "Fingerprint is not successfully registered", vbCritical
```

```
    End If
    txtStudMat.Text = ""
    txtStudMatr.Text = ""
    txtStuName.Text = ""
    txtProg.Text = ""

    Call cmdAbort_Click
End Sub

Private Sub RegSample_RegLeaving()
    lblStatus.Caption = "Fingerprint Leaving"
End Sub

Private Sub RegSample_RegOnProcessError(SampleQuality As BioKit.QualityEnum) 'SDK Instruction
    lblStatus.Caption = "Status : " & SampleQuality 'SDK Instruction
End Sub

Private Sub RegSample_RegSample(SamplePicture As stdole.Picture, CurFingerIndex As Integer) 'SDK Instruction
    Image1(CurFingerIndex).Picture = SamplePicture  'SDK Instruction
End Sub

Private Sub RegSample_RegTouching()
    lblStatus.Caption = "Fingerprint Touching"
End Sub

'Clear TextBox and SetFocus on it for fast and efficient result
Private Sub clearAndFocus()
        txtStudMatr.Text = ""
        txtStudMatr.SetFocus
End Sub


'Supporting Function

Private Sub txtStudMatr_Change()

End Sub
```

Recording Attendance by Using Student's Matric Number

```
Option Explicit
'Dim WithEvents RegSample As BioKit.Reg        'SDK Instruction
Dim file_name As String
Dim SQLstmt As String
Dim conn As ADODB.Connection
Dim rs As ADODB.Recordset
Dim sData As String ' Holds our incoming data
Dim sData1 As Long  ' Holds HighByte value
Dim sData2 As Long   ' Holds LowByte value
Dim sData3 As Long
Dim sData4 As Long
Dim sData5 As Long
Dim StudMatric As Long
Dim StudMatricTxt As String



'Dim txtFingerprint As String



Private Sub cmdDone_Click()
Unload Me
End Sub


Private Sub Command1_Click()
DataReport1.Show
End Sub

Private Sub Form_Load()

  MSComm1.InputMode = 0 'take ascii as input
  MSComm1.CommPort = 1
  MSComm1.Settings = "9600,N,8,1"
  MSComm1.PortOpen = True 'open port
  MSComm1.InputLen = 16 'limitation for input
  'MSComm1.RThreshold = 1
  MSComm1.RThreshold = 16
  'MSComm1.InputLen = 2
```

```
file_name = App.Path & "\Attendance.mdb"

 Set conn = New ADODB.Connection

 conn.CursorLocation = adUseClient

conn.ConnectionString = _
      "Provider=Microsoft.Jet.OLEDB.4.0;" & _
      "Data Source=" & file_name & ";" & _
      "Persist Security Info=False"
conn.Open
 'Once this connection is open you can use it throughout your application
   SQLstmt = "SELECT * FROM Student"
'  Get the records.
  Set rs = New ADODB.Recordset
  rs.Open SQLstmt, conn, adOpenStatic, adLockOptimistic, adCmdText

  Set DataGrid1.DataSource = rs
   With rs
    .MoveFirst
   End With

'txtStudMat.Text = ""
'txtStuName.Text = ""
'txtProg.Text = ""

End Sub

  Private Sub Form_Unload(Cancel As Integer)
MSComm1.PortOpen = False
End Sub

Private Sub Form_Terminate()
   If Comm.PortOpen = True Then 'double check if port still open
      Comm.PortOpen = False 'close port
   End If
   End
End Sub

Private Sub MSComm1_OnComm()
```

```
' If comEvReceive Event then get data and display
If MSComm1.CommEvent = comEvReceive Then

sData = MSComm1.Input ' Get data (4 bytes)

  Text1.Text = sData
  sData1 = Asc(Mid$(sData, 1, 1))  ' get 1st byte
  sData2 = Asc(Mid$(sData, 2, 1))  ' Get 2nd byte
  sData3 = Asc(Mid$(sData, 3, 1))  ' Get 3rd byte
  sData4 = Asc(Mid$(sData, 4, 1))  ' Get 4th byte

  sData1 = sData1 - 48
  sData2 = sData2 - 48
  sData3 = sData3 - 48
  sData4 = sData4 - 48

  StudMatric = sData1 * 1000 + sData2 * 100 + sData3 * 10 + sData4 * 1
StudMatricTxt = CStr(StudMatric)
'Text2.Text = StudMatricTxt

End If

AdoFinger.Refresh
Do Until frmSIDAttendance.AdoFinger.Recordset.EOF
'If frmSIDAttendance.AdoFinger.Recordset.Fields("MatricNo").Value = txtStudMatr.Text Then
If frmSIDAttendance.AdoFinger.Recordset.Fields("MatricNo").Value = StudMatricTxt Then
 frmSIDAttendance.AdoFinger.Recordset.Fields("TimeIn").Value = DateTime.Now 'inser value for start alert
 frmSIDAttendance.AdoFinger.Recordset.Fields("Attendance").Value = "1"
 frmSIDAttendance.AdoFinger.Recordset.Save 'save to database
 frmSIDAttendance.AdoFinger.Recordset.MoveLast

lblStat.Caption = "Your Attendance Is Taken"
Timer1.Enabled = True
Timer2.Enabled = False

'MsgBox "Your attendance is taken", vbInformation

Exit Sub

Else
Timer1.Enabled = False

frmSIDAttendance.AdoFinger.Recordset.MoveNext
```

```
lblStat.Caption = "No Database Found"

"MsgBox "Invalid Search Key Entered", vbCritical, "Search Error"
"clearAndFocus
End If

Loop
End Sub

Private Sub Timer1_Timer()
If lblStatus.BackColor = &HE0E0E0 Then    'change color within interval (blinking)
    lblStatus.BackColor = &H8000&    '&H8000000F&
    Else
    lblStatus.BackColor = &HE0E0E0
End If
End Sub
```

Fingerprint Verification

```
Option Explicit
Dim WithEvents MatchSample As BioKit.Match    'SDK Instruction
Dim file_name As String
Dim SQLstmt As String
Dim conn As ADODB.Connection
Dim rs As ADODB.Recordset
Dim FingerPrint As String

Private Sub cmdAbort_Click()
    MatchSample.Abort                      'SDK Instruction
   'eleminate biokit's instance before can create another new instance
   Set MatchSample = Nothing
   cmdMatch.Enabled = True
   cmdAbort.Enabled = False
End Sub

Private Sub cmdDone_Click()
   Unload Me
   frmSSAR.Show
End Sub
```

```
Private Sub cmdMatch_Click()
   rs.MoveFirst
   FingerPrint = rs!RightFingerprint
    'create new instance of biokit
   Set MatchSample = New BioKit.Match
   Dim res As StatusEnum

   MatchSample.Initialize                       'SDK Instruction
   MatchSample.FAR = 0.005                      'SDK Instruction
   MatchSample.SampleWidth = imgFingerprint.Width     'SDK Instruction
   MatchSample.SampleHeight = imgFingerprint.Height   'SDK Instruction
   res = MatchSample.ActivateSensor(Me.hWnd)

   imgFingerprint.Picture = LoadPicture()

   MatchSample.Execute (FingerPrint)            'SDK Instruction

   cmdMatch.Enabled = False
   cmdAbort.Enabled = True
End Sub

Private Sub initDB()
file_name = App.Path & "\Attendance.mdb"
Set conn = New ADODB.Connection
conn.CursorLocation = adUseClient
conn.ConnectionString = _
      "Provider=Microsoft.Jet.OLEDB.4.0;" & _
      "Data Source=" & file_name & ";" & _
      "Persist Security Info=False"
 conn.Open
'Once this connection is open you can use it throughout your application
   SQLstmt = "SELECT RightFingerPrint FROM Student" ' you should add where to limit the selected students
'Get the records.
 Set rs = New ADODB.Recordset
  rs.Open SQLstmt, conn, adOpenStatic, adLockOptimistic, adCmdText

 'Set DataGrid1.DataSource = rs

End Sub

Private Sub Form_Load()
Call initDB
cmdMatch.Enabled = True
```

```
cmdAbort.Enabled = False
End Sub

Private Sub MatchSample_MatchDone(Status As BioKit.StatusEnum)    'SDK Instruction
    Dim found As Boolean
        found = False

   If Status = StatusOK Then 'SDK Instruction
   AdoFinger.Refresh
      frmFPAttendance.AdoFinger.Recordset.Fields("FPVerification").Value = "1"
      frmFPAttendance.AdoFinger.Recordset.Fields("TimeIn").Value = DateTime.Now
      frmFPAttendance.AdoFinger.Recordset.Save
      MsgBox "Authentication Successful.", vbInformation
      found = True
   Else

       If Not rs.EOF Then
        AdoFinger.Refresh
       Do While (Not rs.EOF And Not found)

        If MatchSample.Compare(rs!RightFingerprint) Then

          frmFPAttendance.AdoFinger.Recordset.Fields("FPVerification").Value = "1"
          frmFPAttendance.AdoFinger.Recordset.Fields("TimeIn").Value = DateTime.Now
          frmFPAttendance.AdoFinger.Recordset.Save
          frmFPAttendance.AdoFinger.Recordset.MoveLast

          found = True

        End If
          rs.MoveNext


      'Loop
      'End If
      If found Then
      'AdoFinger.Refresh
      'frmFPAttendance.AdoFinger.Recordset.Fields("FPVerification").Value = "1"
      'frmFPAttendance.AdoFinger.Recordset.Save

       MsgBox "Authentication Successful.", vbInformation
      Else
         frmFPAttendance.AdoFinger.Recordset.MoveNext
```

```
      'MsgBox "Authentication fail.", vbCritical
      End If
   Loop
    rs.MoveFirst

    cmdAbort_Click
    cmdMatch_Click
   End If

  Call cmdAbort_Click
   End If
End Sub

Private Sub MatchSample_MatchLeaving()
   lblStat.Caption = "Fingerprint Leaving"
End Sub

Private Sub MatchSample_MatchSample(SamplePicture As stdole.Picture, SampleQuality As BioKit.QualityEnum) 'SDK
Instruction
   imgFingerprint.Picture = SamplePicture          'SDK Instruction
   lblStat.Caption = "Status : " & SampleQuality 'SDK Instruction
End Sub

Private Sub MatchSample_MatchTouching()

  lblStat.Caption = "Fingerprint Touching"
End Sub
```