WEARABLE DEVICE FOR SPEED AND ENDURANCE ASSESSMENT AND MONITORING USING IMPROVISED STETHOSCOPE AND ACCELEROMETER WITH USB RECEIVER VIA ZIGBEE TECHNOLOGY

by

Jan Mikael C. Estabillo
Mary Anne U. Fabia
Maritoni J. Maculanlan
Darlon Jay C. Mondejar
Maica A. Punsalang

A Design Report Submitted to the School of Electrical Engineering, Electronics Engineering, and Computer Engineering in Partial Fulfilment of the Requirements for the Degree

Bachelor of Science in Computer Engineering

Mapúa Institute of Technology
March 2011

Approval Sheet

Mapua Institute of Technology
School of EECE

This is to certify that we have supervised the preparation of and read the design report prepared by **Jan Mikael C. Estabillo, Mary Anne U. Fabia, Maritoni J. Maculanlan, Darlon Jay C. Mondejar and Maica A. Punsalang** entitled **WEARABLE DEVICE FOR SPEED AND ENDURANCE ASSESSMENT AND MONITORING SYSTEM USING ACCELEROMETER AND IMPROVISED STETHOSCOPE WITH USB RECEIVER VIA ZIGBEE TECHNOLOGY** and that the said report has been submitted for final examination by the Oral Examination Committee.

Prof. Benigno B. Agapito Jr.
Reader

Engr. Cyrel Ontimare-Manlises
Design Adviser

As members of the Oral Examination Committee, we certify that we have examined this design report, presented before the committee on **March 05, 2011**, and hereby recommended that it be accepted in fulfilment of the design requirements for the degree in **Bachelor of Science in Computer Engineering.**

Engr. Dionis A. Padilla
Panel Member

Engr. Jesus M. Martinez Jr.
Panel Member

Engr. Analyn N. Yumang
Panel Member

This design report is hereby approved and accepted by the School of Electrical Engineering, Electronics and Communications Engineering, and Computer Engineering in partial fulfilment of the requirements for the degree in **Bachelor of Science in Computer Engineering.**

Dr. Felicito S. Caluyo
Dean, School of EECE

ACKNOWLEDGEMENT

We are indeed grateful to the people who advised, gave comments, gave assistance and encouraged our group to make this design possible.

We wish to express our sincerest thanks and appreciation to our professor, Prof. Noel Linsangan, for guiding us with the necessary requirements and motivating us for the accomplishment of this design.

We would like to express also our deepest gratitude to our adviser, Engr. Cyrel Ontimare-Manlises, who gave us words of encouragement and for all the support and guidance she had given to our group during consultation regardless of her busy schedule.

We would like to acknowledge with much gratitude Dra. Hiyasmine Mangubat for sharing with us the knowledge she has in Rehabilitation Medicine. Without her help, we wouldn't know how to initiate this design.

We would also like to thank the panel members who agreed to examine and to certify our oral presentation and made this design a success.

We would also sincerely extend our dearest gratitude to our parents for supporting us, morally and financially, as we accomplished this design; and to our friends who also supported us on our endeavors as we finished this design.

For this achievement, we give back all the glory and praises to the omnipotent Father Almighty for giving us strength, wisdom, patience, and guidance in completing this design.

Jan Mikael C. Estabillo
Mary Anne U. Fabia
Maritoni J. Maculanlan
Darlon Jay C. Mondejar
Maica A. Punsalang

ROLES AND RESPONSIBILITIES OF GROUP MEMBERS

Each member contributed on researching the given design project and the development of the prototype.

The following show the list of responsibilities for each member:

Jan Mikael C. Estabillo

- Circuit Design
- Hardware Design
- Program / Software Design
- Documentation

Mary Anne U. Fabia

- Circuit Design
- Hardware Design
- Documentation

Maritoni J. Maculanlan

- Circuit Design
- Hardware Design
- Documentation

Darlon Jay C. Mondejar

- Circuit Design
- Hardware Design
- Documentation

Maica A. Punsalang

- Circuit Design
- Hardware Design
- Documentation

TABLE OF CONTENTS

TITLE PAGE i

APPROVAL SHEET **Error! Bookmark not defined.**

ACKNOWLEDGEMENT iii

ROLES AND RESPONSIBILITIES OF GROUP MEMBERS v

TABLE OF CONTENTS vi

LIST OF TABLES viii

LIST OF FIGURES ix

ABSTRACT x

Chapter 1: DESIGN BACKGROUND AND INTRODUCTION 1

- Background 1
- Statement of the Problem 2
- Objectives of the Design 3
- Significance and Impact of the Design 4
- Scope and Delimitation 6
- Definition of Terms 8

Chapter 2: REVIEW OF RELATED DESIGN LITERATURES AND STUDIES 10

Chapter 3: DESIGN PROCEDURES 22

- Hardware Development 23
- Block Diagram 24
- Software Development 26
- Prototype Development 38

Chapter 4: TESTING, PRESENTATION, AND INTERPRETATION OF DATA 43

Data Transmission Test 43
Time Measurement Test 44
Pulse Rate Measurement Test 43
Endurance Test 48
Speed Test 50

Chapter 5: CONCLUSION AND RECOMMENDATION 52

CONCLUSION 52
RECOMMENDATION 53

BIBLIOGRAPHY 55

APPENDIX 57

APPENDIX A: OPERATION'S MANUAL 58

System Requirements 58
Installation Procedure 59
User's Manual 60
Troubleshooting Guides and Procedures 63

APPENDIX B: PICTURES OF PROTOTYPE 64

Wrist Device 64
Handheld Device 64
Lap Indicator Module 65
Batteries and Charger 65

APPENDIX C: PROGRAM LISTING 66

Microcontroller Source Code 66
Software Application Form Scripts 82

APPENDIX D: DATASHEETS 112

APPENDIX E: SOFTWARE DEVELOPMENT 164

APPENDIX F: SPECIFICATIONS OF THE DIGITAL BLOOD PRESSURE
USED IN THE TESTING 197

LIST OF TABLES

Table 3.10 List of Materials 38

Table 4.1 Data Transmission Test 43

Table 4.2 Time Measurement Test 45

Table 4.3 Pulse Rate Measurement Test 47

Table 4.4 Endurance Test 49

Table 4.5 Speed Computation Using the Time Reading from the Stopwatch and Microcontroller 51

LIST OF FIGURES

Figure 3.1 Conceptual Diagram 22

Figure 3.2 System Block Diagram 24

Figure 3.3 Use Case Diagram 26

Figure 3.4 Data Flow Diagram 28

Figure 3.5 Activity Diagram 30

Figure 3.6 Schematic Diagram of the Wrist Device 32

Figure 3.7 Schematic Diagram of the Handheld Device 36

ABSTRACT

This project concerns runners as its target users because running is the most widely used endurance test for both athletes and non-athletes. Of late, bulky equipments are required to measure speed and pulse rates for endurance assessment. To amend the situation, the designers have incorporated both speed and pulse rate measurements into a single device. This was done by compressing the prototype circuit into a small printed circuit board (PCB), using nickel-metal hydride batteries, and the use of a strap to hold the stethoscope and the circuit along the wrist. The device was able to provide mobility because of the rechargeable batteries and wireless transmission of data using ZigBee. Further, a software application is included to translate data into readable information. This design will have a large impact on those who undergo endurance tests as it will ease condition monitoring, from endurance, by providing graphical interpretation of the activity and portability on devices used.

Keywords: endurance, ZigBee, USB, software application, rechargeable battery

Chapter 1

DESIGN BACKGROUND AND INTRODUCTION

Background

Often times, trainers have a hard time measuring the intensity of the training they impart to their trainees and the performance of the individual throughout the training program. Although some trainers are eased in measuring such quantity, measurements are sometimes, if not are always subjected to approximation. Individuals' tolerance to their exercise is one of the most essential parameters a trainer should consider as it can assess the endurance of an individual to its activity. People often exercise longer and harder so they can improve. But without adequate rest and recovery, these training regimens can backfire, and actually decrease performance.

Given that the design incorporates the use of accelerometer and stethoscope in assessing the conditioning of the person during running activities or exercises. Pulse rate is useful for monitoring the individual's condition on a daily basis.

As stated above, the design uses an accelerometer to determine the hand movements of the corresponding trainee from start to end which will be timed.

From this, we can obtain the overall duration of the individual's exercise. The use of pulse-rate measuring device will provide data on whether the person can tolerate the exercise and recover from the fatigue it can cause. Thus, the endurance of the trainee is assessed using the pulse-rate measurer preventing a course to overtraining that diminishes the performance of an individual. Endurance will be assessed using the Karvonen Formula, one of the most effective methods used to calculate training heart rate and was also advised by the doctor from the Rehabilitation Center that the group had consulted. It also uses ZigBee Technology to transfer training data to the trainer or to a designated individual wirelessly through a USB. A software application is programmed by the proponents to translate the data to readable information.

It has five major components, namely Lap Indicator Module, accelerometer, ZigBee module, improvised stethoscope, and USB storage module.

Statement of the Problem

Pedometers and Pulse Oximeters are the usual equipments of athletes nowadays. Pedometers require a belt and a pair of shoes to function while pulse oximeters can either be worn on the finger-tip, earlobe, or toe to function. From the latter, it can be safely assumed that the said devices will become

cumbersome while using the two functionalities of pedometers and pulse oximeters. Due to this scenario, a solution is considered to lessen the discomfort from wearing many things during workout exercises by combining both pulse and pace measurement into one wrist-worn device. And so the proponents came up with a design having an additional feature of assessing and monitoring one's speed and endurance. This design measures the pulse rate at the wrist along the ulnar artery. The main determinant of running speed is time where the input distance — required to be run by the user — is divided by the time it took to cover the distance. These measured data are wirelessly transmitted to a USB receiver and will eventually be used as an input to a software application for evaluation.

Objectives of the Design

General Objective

The wearable speed and endurance assessment and monitoring system using accelerometer and improvised stethoscope with USB receiver via Zigbee Technology aim to combine two important features, namely pulse and pace measurement to a single wireless device, as these gathered data will be an input to evaluate the user's speed and endurance.

Specific Objectives

1. To measure the pulse rate of the user using an improvised stethoscope with a minimum requirement of percent difference that is not greater than 5% and will use the data to evaluate the individual's endurance along with the inputs namely as age and resting heart rate in a formula.
2. To measure the time the user finished a lap using the microcontroller's interrupt timer and infrared module with a minimum requirement of percent difference that is not greater than 5%.
3. To transmit data wirelessly to a USB receiver using ZigBee Pro and USB VDIP circuit, and to translate data into readable information using program software.
4. To evaluate the pulse rate together with inputs, namely, age and resting heart rate for the user's endurance using Karvonen formula.
5. To evaluate the time along with the input distance for user's speed.

Significance and Impact of the Design

The significance of the design is for the trainers, runners, sport enthusiasts, or simply an individual having exercise / training programs to have a device that assesses and monitors their pulse rate, endurance and speed. This project can help them improve their speed and endurance abilities. They can

monitor their training activities because the data can be transferred and stored at Universal Serial Bus (USB). This design contributes in the advancement of the technology since a pulse rate measurer, accelerometer and ZigBee are all in one device without affecting the runners' carrying weight. Through this design, it is expected that a sufficient training and a maximum health safety will be made reaching every individual's target endurance.

The impact of the design influences three realistic constraints, namely social, ethical, and health and safety. The implementation of the design offers freedom among users that will come to utilize the device which comprises the social constraint. However, appropriate outcome of the device depends a lot on the wearer. Since the device can accept correct but invalid inputs, unethical users may use the device to falsify their training. On the other hand, unethical coaches can come to manipulate the data and cover up possible frauds since the file is in text format. Health and safety constraints concern the use of a pulse rate measurer to monitor an individual's cardiovascular condition.

Scope and Delimitation

Scopes:

The design includes:

- measurement of the time it takes for the user to complete a single lap while hands are in motion using a microcontroller counter.
- recognition of hand movement using an accelerometer.
- pressing of a button to start the timer interrupt of the microcontroller before running.
- infrared transmission by placing lap indicator module on the start line at the same time the finish line of the track to signal the receiver, on the device worn, for the beginning and completion of a lap, respectively.
- measurement of individual's pulse-rate by standing still after completing the final lap using an improvised stethoscope.
- serial transfer of data to a USB interface.
- assessment of speed and endurance by plugging the USB into a computer where the evaluating software application is installed.
- wireless transmission of data through ZigBee Pro which has a maximum range of 1.5 km, but is expected to decrease from various factors such as walls, humidity, temperature, and the like.

- rechargeable batteries to power devices, since running is for an outdoor environment.

Delimitations:

- The design will encounter valid but incorrect outputs, specifically swaying of hands while at rest.
- The user must not be adjacent to the Infrared Sensor at the end of the last lap when measuring the pulse rate.

Definition of Terms

Athletic Attributes or feats are quantifiable skills of an athlete that are shown to be measurable through electronic devices.

Accelerometer measures the static acceleration of gravity in tilt-sensing applications, as well as dynamic acceleration resulting from motion or vibration (Charles, 2007).

Endurance is the time limit of a person's ability to maintain either a specific force or power involving muscular contractions (Morgan, 2001).

Exercise is a physical activity that is planned, structured, and repetitive for the purpose of conditioning any part of the body. Exercise is utilized to improve health, maintain fitness and is important as a means of physical rehabilitation (Bookhout and Grenman, 2001).

Overtraining is a condition that is usually occurring in athletes who are training for competition then exercise outside the body's ability to pull through (Redding, 2001).

Real-time is a representation of the time at which an output is obtained from an input.

Speed is the rate or a measure of the rate of motion, especially distance travelled divided by the time of travel (Young and Freedman, 2004).

Throughput is the interval in which an output is realized from an input.

Universal Serial Bus (USB) is a set of connectivity specifications developed by Intel in collaboration with industry leaders. USB allows high-speed, easy connection of peripherals to a PC. [The Intel Microprocessors-Fifth Edition]

ZigBee is a proprietary set of high level communication protocols designed to use small, low power digital radios based on the IEEE 802.15.4 standard for wireless personal area networking.

Chapter 2

REVIEW OF RELATED DESIGN LITERATURES AND STUDIES

This chapter includes compiled research works and studies that are related to the design. The content of this chapter is used as reference for the development of the design. The group used the following related articles, research works, and inventions as reference which will aid in having an in-depth understanding about the design that the group worked on.

Khalil and Sornanathan (2010) says that a real time system that monitors and analyzes the physical signals during fitness activity was called CaszOxiSys. The system has light-emitting diode and photodiode used to receive signals. Bluetooth was used to transmit these signals wirelessly a netbook or laptop for signal evaluation. The Karvonen formula was also used in this paper in which it calculates the heart rate based on age, gender, resting heart rate, maximum heart rate and fitness level. With this paper, the Karvonen formula acts as a guideline for fitness workout sessions. This paper is related to our design since the CaszOxiSys was used to analyze the physiological data of an individual same as with our software program that evaluates the endurance of a person. The design used the Karvonen formula to make an evaluation of the person's endurance. It also supports the design in using the formula because they also used it as reference. The paper transmits data using Bluetooth, this design uses

a Zigbee Pro. They also used pulse oximeter for measuring the pulse rate while this design is through an improvised stethoscope.

Ohyama et. al. (2007) based their studies on using the Karvonen formula and the rating of their actual actions. The electric cart was installed by a knob that allows the driver to continuously vary the pedal load between the exhausting and assisted modes. A gain-scheduling cart control system is designed that will guarantee the stability of the cart system for any pedal load in the allowed range, and a stability condition is derived using dynamic parallel distributed compensation. They determined the perspective of ergonomics based on the Karvonen formula since it assists the person's physical condition that will be using in the electric cart.

According to the research, "Monitoring Athletes' Physiological Responses to Endurance Training with Genomic-wide Expression Data" , it is a system of fixed effect regression modeling for genome-wide expression data from DNA microarray hybridization. This research was accepted last September 2 2007. In monitoring athletes' physiological responses to endurance training, statistical methods in longitudinal or matched case-control data analysis were discussed. Ilene Chen, Ling-Hong Tseng, Hiroto Homma, Hong Yan and L. Lloyd Keith used the technique called fixed effect regression modeling in identifying the significant differential expressed genes with endurance training-induced muscle contraction.

They also used fixed effect logistic regression modeling to study a gene expression model relating to endurance training-induced vastus lateralis muscle contraction. And they found out that the development of carbohydrate, lipid and energy metabolisms, respectively, the presence of the deleterious effects of oxygen from the metabolic reduction of the reactive oxygen species, and the transcriptional regulations of endurance training-induced vastus lateralis muscle contraction status. The paper also described that it can supply general tools to monitor athletes' physiological responses to endurance training on the genomic scale. This relates to the Wearable Endurance Monitoring using USB via Zigbee Technology of the proponents because it monitors the athletes' physiological responses like their endurance. The project design measures the pulse rate and relates it with the other inputted information like age, gender and weight to assess the endurance of the user.

According to Cheng and Hailes (2008), the main function of the project is to evaluate whether inertial sensors are useful for detecting the smallest details of a rapidly moving (foot) motion of a sprinter, which would be useful for coaching support. Also, in this article, it was shown the on-body wireless inertial sensing system, and was analyzed in three aspects: a) a foot motion analysis using the collected inertial data of sprinters; b) the system's physical characteristics (i.e. weight and operational behaviour); and c) the system's wireless performances. In the proposed design case, to further analyze the

improvement of the performance of an athlete, the proponents will be using an accelerometer sensor and a pulse rate sensor. With this, it will measure the speed of the athlete all throughout the course of his exercise and also the sequence of events the athlete jumped as jumping records a sudden increase in the reading of the accelerometer. The use of pulse-rate measuring device will provide data on whether the athlete can tolerate the exercise and recover from the fatigue it can cause.

He and Jin (2008) talk all about proposing a gesture recognition based on single tri-axis accelerometer mounted on a cell phone for human computer interaction. By using the accelerometer which measures the amount of acceleration of a device in motion enables three kinds of gesture interaction methods: tilt detection, shake detection, and gesture recognition. Thought there are other devices used in gesture recognition, it is much advisable to use the accelerometer because those other devices focus on recognizing the simple gestures only such as Arabic numerals, simple linear movement and directions. With this research, the three feature extraction methods are presented, namely discrete cosine transform, Fast Fourier transform and a hybrid approach which combine wavelet packet decomposition with FFT. Supporting documentaries show that all three proposed feature can recognize the 17 complex gestures based on a single tri-accelerometer. By this information, it is very helpful to the proponents in developing the design. Accelerometer will be used to measure the

gesture of the athletes. With this, it can detect whenever the user starts or stops performing and thus it can be a basis to know how far the user is running. When the accelerometer detects the motion of the user the timer will automatically start, and stop as the accelerometer detects that there is no more motion.

Fu (2008) presents a method to recognize the arm motions performing within a short time for instant interaction called gesture strokes. It also used a computer vision and linear accelerometer. The arm motion was first detected by the accelerometer with a time window. From this window, the information gathered will be used by these two sensory systems which individually estimate the probability mass distribution of the gesture stroke classes. The set of weight exponents are learned by the Nelder-Mead method that minimizes the empirical error rate of classifying all training samples. The experiments show that these two sensory systems encompass with each other and the combination framework improves the recognition correct rate. This research was related to the project design of the proponents because gesture strokes were also the bases in completing this design. The proponents chose this paper because it accounts arm motions for a short time in an instant interaction, accelerometer and sensory systems were used and so for the proposed design.

The paper entitled Signal Processing of The Accelerometer on Handheld Devices, issued last November 2003, reports about treating signals of

accelerometers to recognize user gestures from detected signals from accelerometers after applying small accelerometers to handheld devices, and about how to precisely recognize gestures to detect user gestures. It also accounts how to use handheld devices in recognizing gestures by overheads arising from the process of recognizing gestures should be little and gestures should be effectively recognized in real operational environments. The result states that because gesture-based control is easy to use plus it can reduce preparation process in controlling rapid system reaction then, it is a proper user interface for handheld devices primarily used in mobile environments. This paper relates to the design of the proponents because small accelerometer was used in a handheld device. This project design also uses small accelerometer that comes in a small package plus it is a wearable device. The proponents chose this paper because it has same approach that a small accelerometer detects and recognizes the gesture of the users on a handheld device.

From the article "Evaluation of Neural Networks to Identify Types of Activity Using Accelerometers" they used the ActiGraph accelerators which were placed on each of the subject's hip and ankle. Their purpose here was to develop and evaluate two artificial neural network (ANN) models based on single-sensor accelerometer data and an ANN model based on the data of two accelerometers for the identification of types of physical activity in adults. The subjects were also given a sequence of activities like sitting, standing, using the stairs, and walking

and cycling at two self-paced speeds. The model based on the hip accelerometer data and the model based on the ankle accelerometer data correctly classified the five activities 80.4% and 77.7% of the time respectively, while the model based on the data from both sensors achieved a percentage of 83.0%. The hip model produced a better classification of the activities cycling, using the stairs, and sitting, whereas the ankle model was better able to correctly classify the activities walking and standing still. All three models often misclassified using the stairs and standing still. The accuracy of the models significantly decreased when a distinction was made between regular versus brisk walking or cycling and between going up and going down the stairs. From the study, we can determine what the person is currently doing with the help of the ActiGraph accelerators. The proponents chose this article because it shows how an accelerator can determine the current action of the person and distinguish it from the other actions.

Lee et. al. (2007), implementation of accelerometer sensor module is an application to sports athletes' exercise measurement and pattern analysis. It is about the implementation of wireless accelerometer sensor module and algorithm to determine wearer's posture, activity and fall. According to the article, this research uses wireless RF module which measures accelerometer signal and shows the signal at 'Acceloger' viewer program in PC. Using the same principle, the proponents will be using accelerometer sensor module to

determine the directional movements of the corresponding athlete. From this, it can measure the speed of the athlete all throughout the course of his exercise. And instead of using wireless RF module, the developers come up with the idea of using Zigbee module.

Yangin (2008) discusses the principle of this project which is to describe the latest developments in body sensor networks (BSN) for athletes during training and outline the technical requirement of Sports-BSN hardware design, miniaturisation, packaging, as well as the real-time data processing, sensor fusion, and data visualisation issues. It has motivated the use of wearable devices for sports performance sensing with the need for monitoring athletes under natural training environment. This helps the proponents in developing the proposed design since it aims to miniaturize the packaging of the hardware design which is approximately 5 cm x 7 cm and 2.5 cm thick. It also helps to satisfy every athlete that comes to utilize the device in terms of convenience, portability, and usability. While athletic performances continue to improve, the accurate training prescription and feedback become important to the consistency of the training outcome and maintaining the performance margin.

Cheng et. al. (2010), is all about a practical, cost-effective, user-friendly stride-parameter sensing system - known as the Sensing for Sports And Managed Exercise (SESAME) Integrated System (IS) - which is the first system

for supporting practical and long-term biomechanics research studies in sprinting. According to the article, the device measures some parameters of sprinters, such as split times (i.e. which is speed-related), foot contact times, stance times, stride/step length, and stride/step frequency, etc. which are important factors affecting athletes' performances. The system includes a light-sensor-based split time monitoring system, a radio-based localization athlete tracking system, a stride length monitoring system, and a centralized data repository. It is somehow related to the proposed design since it measures different parameters of different movements of an athlete. The pulse rate sensor and accelerometer sensor can measure an individual's speed, directional movement and pulse-rate. The speed and movement are measured from the motion of the body part to be detected by the accelerometer. It will also use ZigBee module to instantly transfer training data to the coach or to an appropriate person wirelessly. To further increase its portability, it can also transfer data through USB (Universal Serial Bus).

Moron et. al. cite the concept of a wearable device which gives a great impact on medical applications. This paper illustrates the different benefits of a wireless technology. One of these is that it provides freedom of movements since no wires will hinder the user and so it will improve the user's quality of life. Great benefit of a wireless device is that it increments the medical presence in emergency scenarios and makes possible remote diagnosis. This research helps

the proponents to come up with the idea of a wearable endurance detection device which gives a unique quality to other inventions present today. This will serve as a two-in-one device which includes a pulse rate sensor and a gesture recognition system. It will be presented on a medium size package or a "wearable" manner, specifically to be put on wrists.

Zucatto et. al. (2007) deal with the difference between ZigBee and Bluetooth, and the basic concepts of ZigBee. ZigBee, arrived in 2005 in 2.4 GHZ band and has a lower data rate than Bluetooth, spends most of its time snoozing, and targets building automation as the main application. ZigBee is a wireless technology that is a low cost yet long reach (100 m). This study shows that ZigBee represents a promising technology for the development of innovative products. ZigBee battery lifespan can possibly be up to 300 days.

Wireless Sensor Data Collection based on ZigBee Technology that was published last June 2010 envisions a comparison of different configurations of a wireless sensor system for capturing human motion. The researchers discussed the different systems used in this study. The systems consist of sensor elements which wirelessly transfer motion data to a receiver element. The sensor elements consist of a microcontroller, accelerometer(s) and a radio transceiver. The receiver element consists of a radio receiver connected through a microcontroller to a computer for real time sound synthesis. The wireless transmission between

the sensor elements and the receiver element is based on the low rate IEEE 802.15.4/ZigBee standard. A configuration with several accelerometers connected by wire to a wireless sensor element is compared to using multiple wireless sensor elements with only one accelerometer in each. The study shows that it would be feasible to connect 5-6 accelerometers in the given setups. Sensor data processing can be done in either the receiver element or in the sensor element. For various reasons it can be reasonable to implement some sensor data processing in the sensor element. The said research was based on Zigbee Technology as what the project design of the proponents should be carried out. It also states that the data that will be collected were be transmitted through Zigbee module, the said design also uses this information in gathering data that will be sent to USB held by the coaches. This paper was chosen because it gives a background information on how Zigbee module will transmit data to a receiver element. And the device used in this paper consists of an accelerometer, sensors and a radio receiver as the receiver element.

Geer (2005) shows the different applications of ZigBee and other possible application for future use. These applications automate home, building, industrial, agricultural systems, including thermostats and security products, bridges and other structures, automated meter reading, and even in home healthcare, national security, military networks, and routing discovery approach that ad hoc on-demand networks use. With this study, it helps the proponents to

choose ZigBee technology than Bluetooth. Since the field that the athletes are using is really big, ZigBee will be the one to accommodate this scenario. And with the benefits that ZigBee provides it can satisfy the application for our design.

Calabro et. al. (2010) evaluate the validity of energy expenditure estimates from two portable armband devices, the SenseWear Pro3 Armband monitor (SWA) and the SenseWear Mini armband monitor (Mini), under free-living conditions. It concluded that The SenseWearPro3 and the SenseWear Mini armbands show promise for accurately measuring daily energy expenditure under free-living conditions. However, more work is needed to improve the ability of these monitors to accurately measure energy expenditure at higher levels of expenditure. Relating this to the researcher's design, the energy spent can also be measured through the activities the person has done. This can also be added as a feature to the design and for future designs. This was selected because of the accuracy of the device mentioned and how effective it is.

Chapter 3

DESIGN PROCEDURES

This chapter is a step-by-step procedure used in developing the design and in the development of the required program needed to make the proposed design fully functional. The hardware, software and prototype development are discussed.

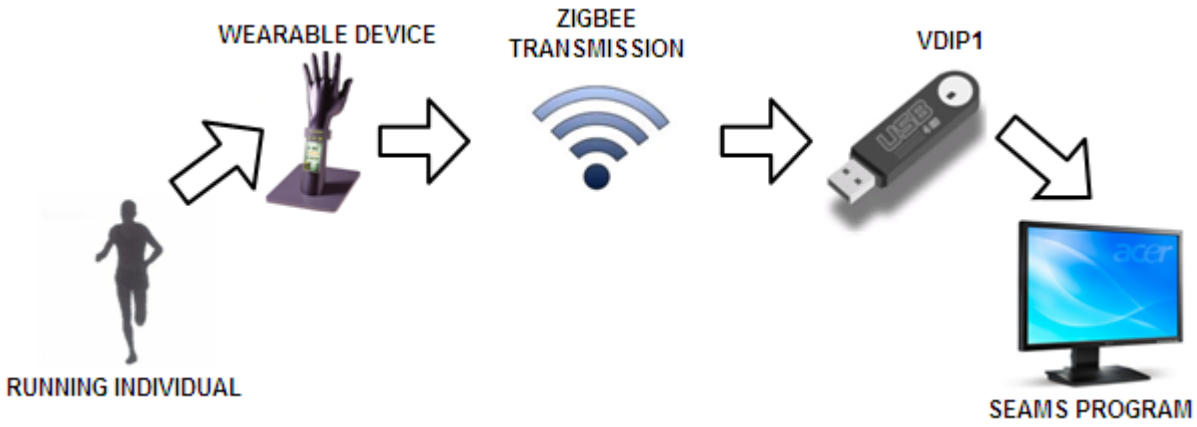


Figure 3.1 Conceptual Diagram

The main concept of the proposed design is that the user will be wearing a device that will measure the pulse rate and the time the user had able to finish a lap. The device will send the information every after the user passes by in the infrared. It will be then transmitted through ZigBee technology. It will be acquired by the receiver which is a USB module. This USB module is connected on a handheld device while the user is performing. After the user performs,

he/she will insert the flash drive on the PC where the SEAMS program is installed and afterwards will run the program and the results gathered will be saved and the user's endurance will be assessed.

Hardware Development

The proposed design which is the Wearable Device for Speed and Endurance Assessment Monitoring System using Improvised Stethoscope and Accelerometer with USB Receiver via Zigbee Technology is divided into 3 hardware parts: transmitter, receiver, and lap indicator module. The transmitter unit is the primary hardware of the proposed design which is composed of the improvised stethoscope, an accelerometer, an MCU, infrared receiver and a ZigBee Pro transmitter module. On the other hand, the receiver component consists of a ZigBee Pro receiver module, serial USB interface and MCU to store data. The last hardware part is the lap indicator module which consists of an infrared transmitter and a microcontroller that complete the system.

Block Diagram

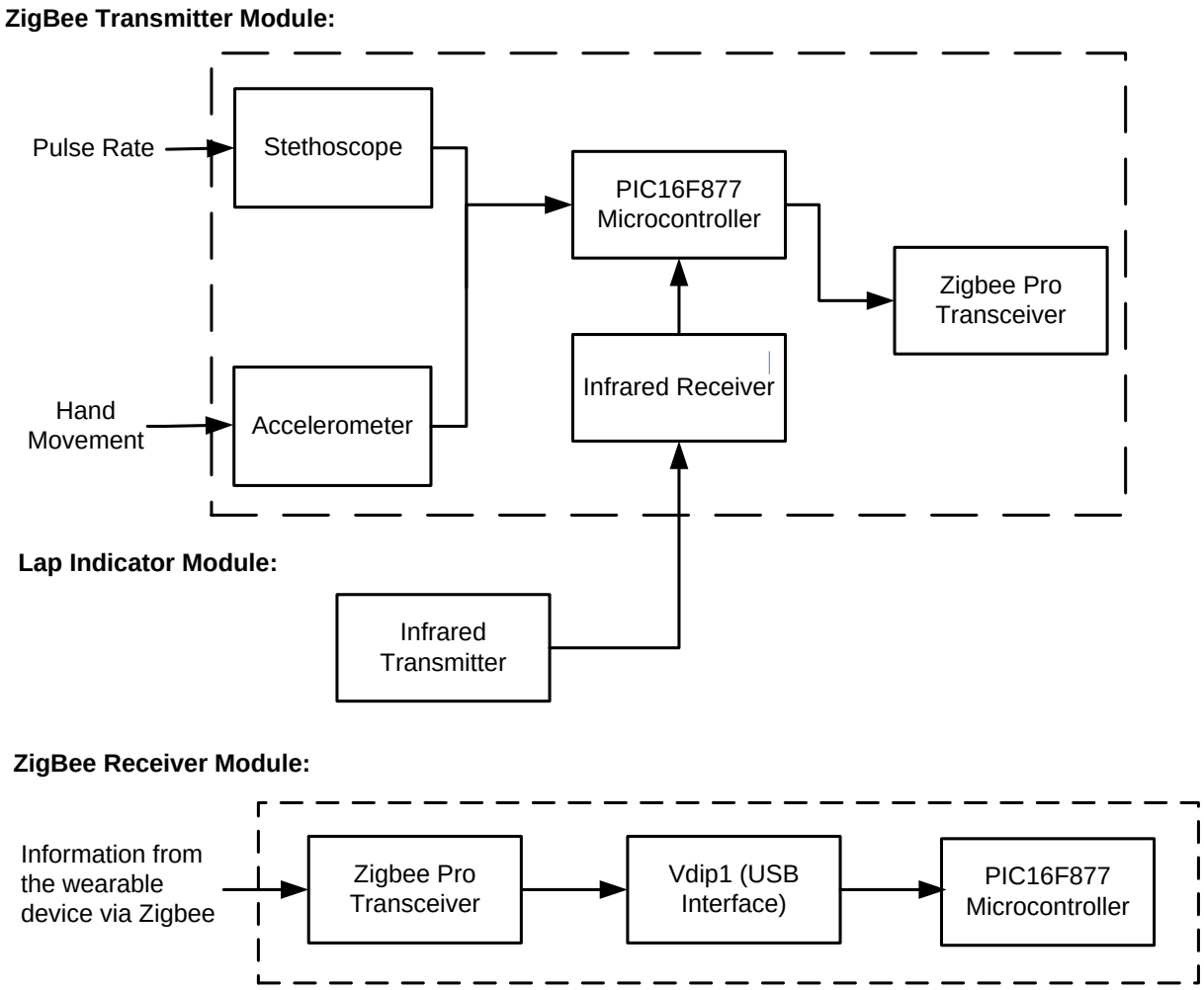


Figure 3.2 System Block Diagram

In Figure 3.2 the transmitter side's inputs are infrared signal, hand movement and pulse rate while the process is composed of the infrared receiver, microcontroller and the ZigBee Pro module. The transmitter side represents the block diagram for the wearable device. It is composed of the accelerometer chip, stethoscope, infrared module, ZigBee Pro module and a microcontroller. An

accelerometer keeps track of the hand motion of the user all throughout the course of his exercise. The microcontroller records individual lap times whenever an infrared signal is received then interprets the time and pulse rate measured to be saved on a text file. The use of stethoscope device is to measure the pulse rate of the user after a while that the user is at rest to indicate the beginning of wireless transmission of data. The ZigBee Pro module will then transmit all the data gathered to the Vdip1 that is connected on the handheld device.

In the receiver portion of the system involves the transmission of the data through the receiver of the ZigBee Pro module. The data will be received by the USB flash drive and can be connected to a laptop or a PC then the user may run the Software Program installed that can assess and monitor their speed and endurance.

On the lap indicator module, an infrared transmitter is made to communicate with the wearable device that consists of infrared receiver. As the name of the component suggests, the infrared communication is made to indicate that a lap has either started or completed.

Use Case

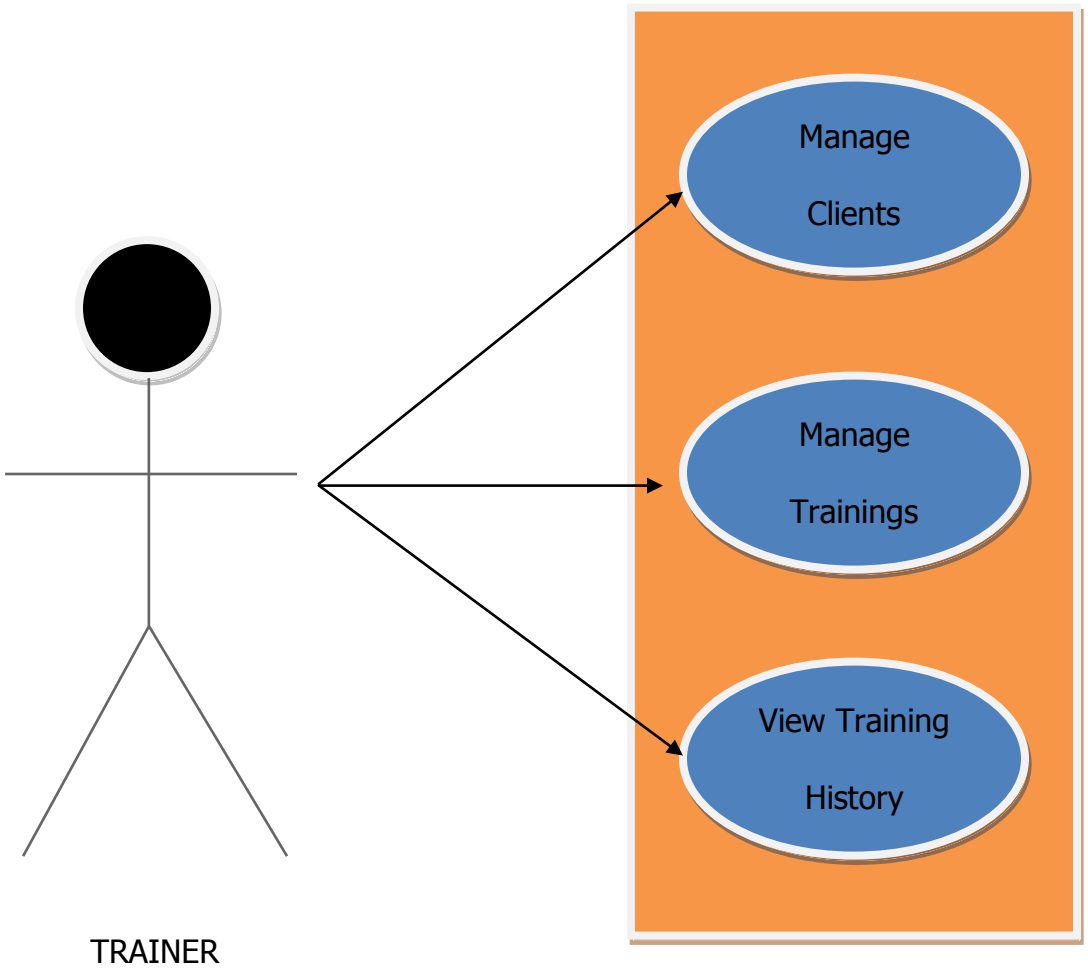


Figure 3.3 Use Case Diagram

Manage Client Use Case

The registration incorporates two validation processes Select and Add/Update that require three fields to be filled before the process prevents a null input during validation process. Fields follow a specific format and must not

deviate from them so as to avoid error encounters. A convenient select mode is also provided to be able to search for existing clients in the record. While adding, check availability is included to inform the user of the existence of the name in the record. The registration can be cancelled at any point during the process.

Manage Trainings Use Case

Revision cannot be opened unless a user has been selected during the registration process. The form prevents errors by providing limitations on input. The resting heart rate can be left blank to signify a no change of data, while a blank track distance is equivalent to 0 meters. An evaluation is incorporated to assess the endurance based on the change of heart rate. A refresh is also included to redraw the graph whenever distorted. A convenient open dialog box is provided to ease file browsing. The Graph can be dragged to view concealed parts of the training easily.

View Training History Use Case

Browsing requires the choice of graph type through radio buttons and a selection within two combo boxes that list all the existing clients and trainings from the corresponding record. It automatically requests information and proceeds immediately to evaluation. Drag capability is also enabled to provide interactive interface on the graph.

Data Flow Diagram

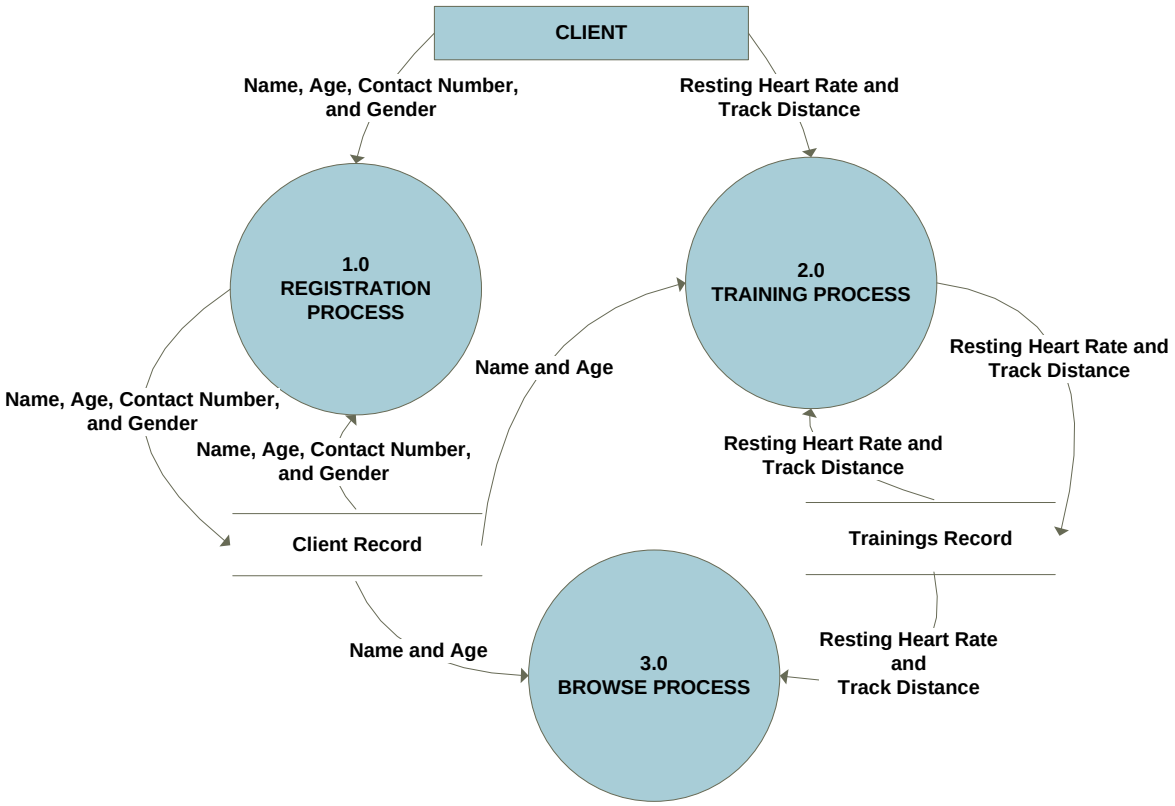


Figure 3.4 Data Flow Diagram

The system accepts input only on the registration and training processes. Registration process accepts name, age, contact number, and gender inputs all being a required field except contact number. All the inputs will immediately be transferred to the client record after clicking the add/update button. Training process accepts two inputs resting heart rate and track distance, and can only modify the current training's resting heart rate. Inputs will then replace the resting heart rate and track distance in the training record. Browse process does not accept inputs as it is for displaying purposes and gets data from the training and client records. Client record transfers all data to registration process and transfers only name and age to training and browse processes. Training Record transfers all data to every process whenever necessary.

Activity Diagram

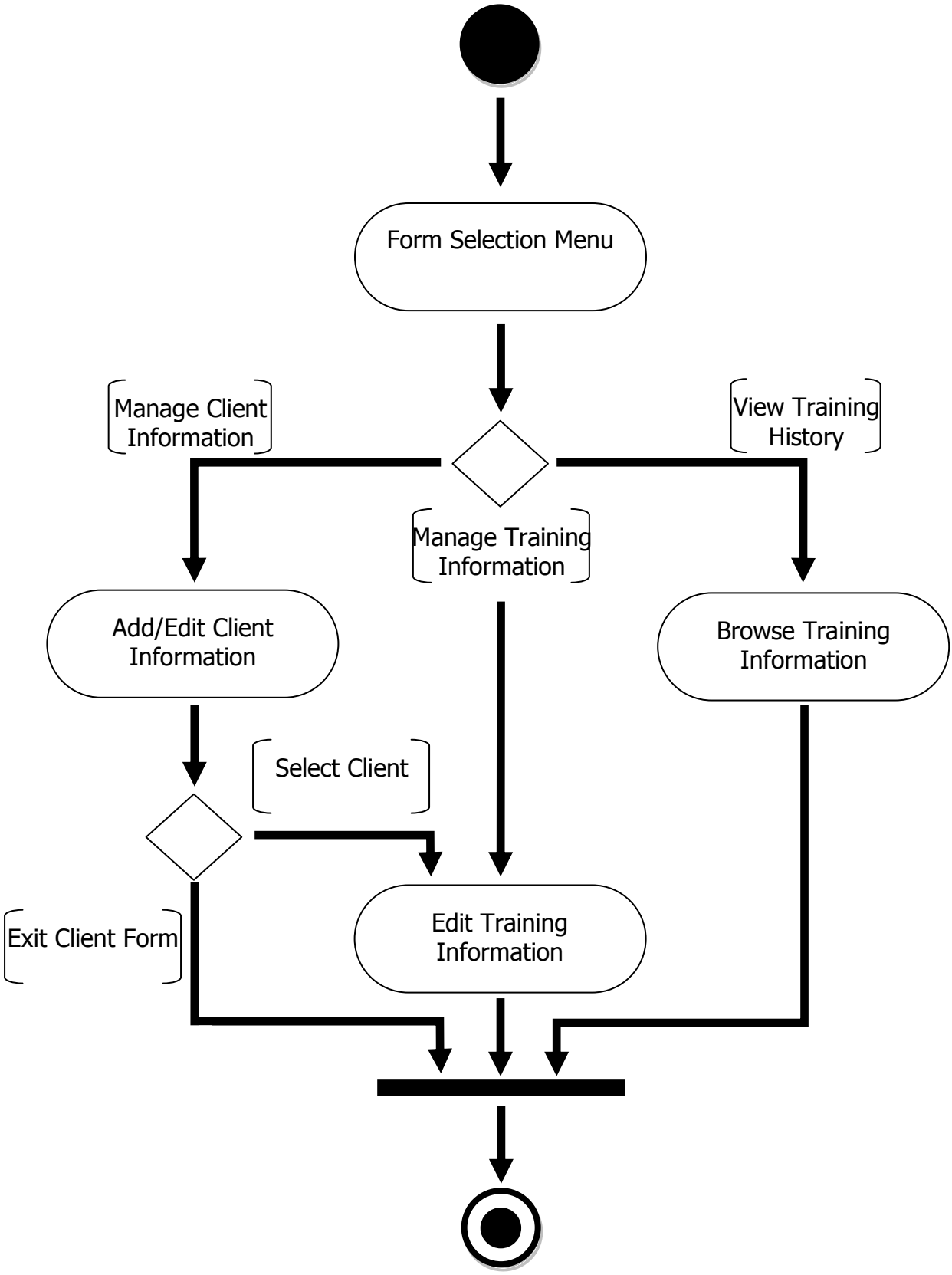


Figure 3.5 Activity Diagram

Before training, the software application must be installed in a PC given together with the hardware. After successfully plugging-in the USB flash drive, the user can choose between managing clients, trainings or training history and then can end the current activity. In the course of managing client information, the user can choose to select or exit the form to end his current activity.

SCHEMATIC DIAGRAMS:

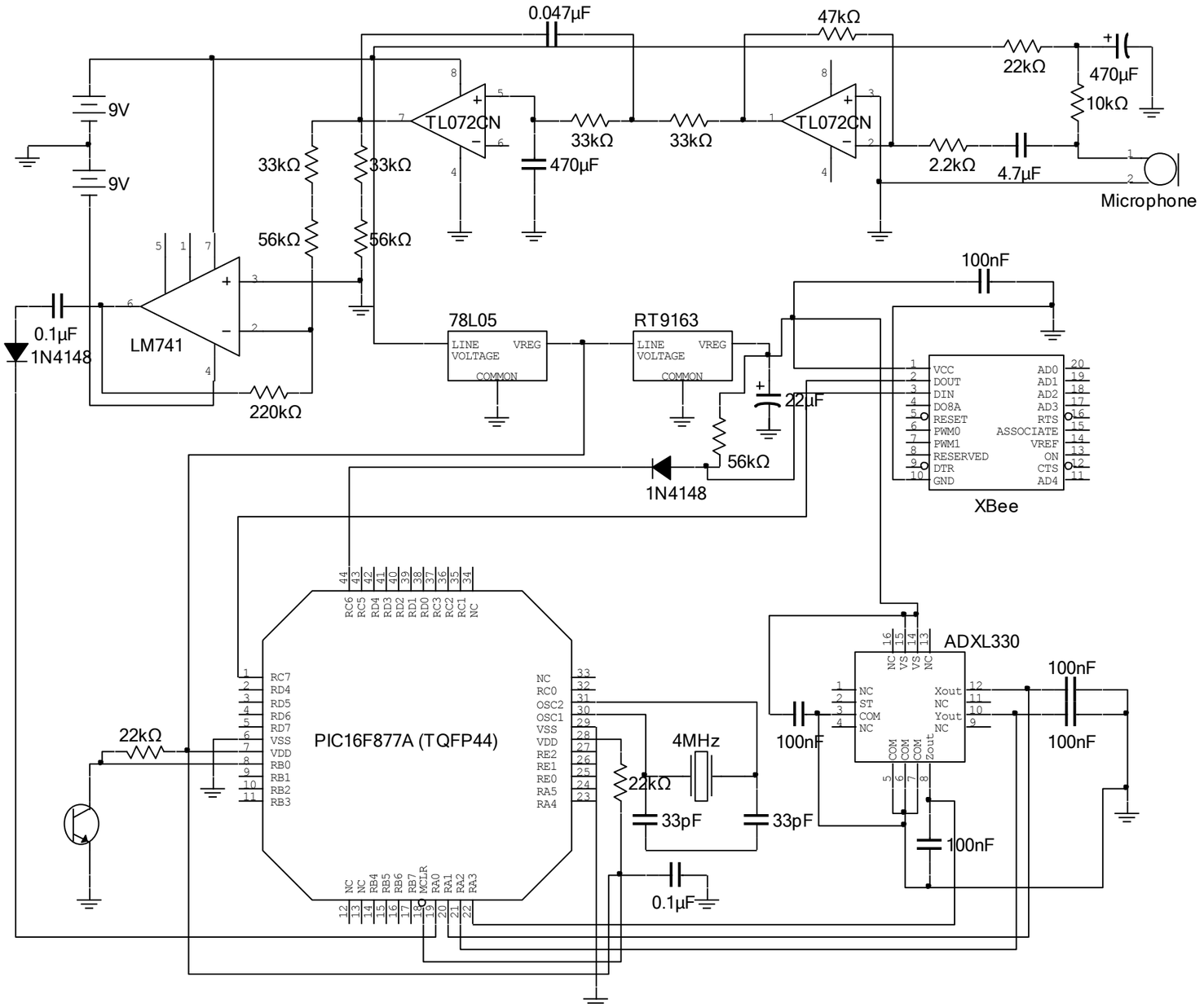


Figure 3.6 Schematic Diagram of the Wrist Device

The stethoscope is connected with a condenser microphone so that the sound it collects will be converted to an AC wave signal. The designers used cascaded JFET operational amplifiers to amplify the sign wave with minimal noise. From the formula,

$$A = R_2 / R_1$$

$$A = 47\ k\Omega / 22\ k\Omega$$

$$A = 2.14$$

the gain will approximately double the input signal of the Low-Noise Microphone Pre-Amplifier. For the Sallen-Key Butterworth Amplifier, we used the formula,

$$fc = 1 / (2\pi\ x\ \sqrt{R1\ x\ R2\ x\ C1\ x\ C2})$$

$$fc = 1 / (2\pi\ x\ \sqrt{33\ k\Omega\ x\ 33k\Omega\ x\ 0.047\ \mu f\ x\ 0.047\ \mu f})$$

$$fc = 102.6144\ Hz$$

to determine the maximum frequency the amplifier will accept since it is a low-pass filter. Thus, any frequency below fc is considered a pulse rate.

A differential operational amplifier is used to further amplify the signal with same resistance on its input and output terminals. A 0.1 μF capacitor is connected on the Sallen-Key Butterworth Amplifier, which can possibly be 1 μF to 0.1 μF on the output to filter noise and prevent oscillation. The 0.1 μF capacitor also serves as a AC to DC converter of the audio signal together with a diode. The diode is used for its fast-switching capability needed for the realtime circuit. The DC audio signal is then fed to the microcontroller for storage and transmission.

The PIC microcontroller supply pins are connected to a positive voltage regulator with an output of 5 Volts. Clock inputs are connected to a 4 MHz crystal oscillator, in which the circuitry needs a maximum of 20 MHz of clock frequency, that allows the operation of the microcontroller. Here is how the resistors are computed:

$V = I \times R$

$R = V / I$

$R = 5V / I_{MIN} \sim I_{AVE} \sim I_{MAX}$

$R = 5V / 50\mu A \sim 200\mu A \sim 400\mu A$

$R = 12.5k\Omega \sim 20k\Omega \sim 100k\Omega$

Thus, a 22kΩ is sufficient for the circuit. Since the load capacitance for parallel resonant crystal C_L is given in the datasheet as 21.5pF, the load capacitor for the crystal oscillator is computed as follows:

$$\mathrm{CL} = \frac{C1\ x\ C2}{C1 + C2} + Cs$$

where C_1 and C_2 are equal and C_S or Stray Capacitor is typically 5pF will give the equation

$$21.5pF = \frac{C\ x\ C}{C + C} + 5pF \sim 16.5pF = \frac{C^2}{2C} \sim 16.5pF\ x\ 2 = C = 33pF$$

obtaining a load capacitor rating of 33pF for the 4MHz crystal oscillator. The 0.1µF capacitor serves as a ripple rejection capacitor since a stable $5V_{DC}$ is required for the MCLR pin.

Vs supply pins need a voltage input of 1.5 V. Output axis data XOUT, YOUT, and ZOUT are connected to their corresponding input pins of the microcontroller R0 (Pin 2), R1 (Pin 3), and R2 (Pin 4), respectively. A 100nF capacitor for each axial output is used to enable a low-pass filtering for antialiasing and noise reduction at 50Hz so that frequencies above 50Hz won't be able to intervene within the circuit.

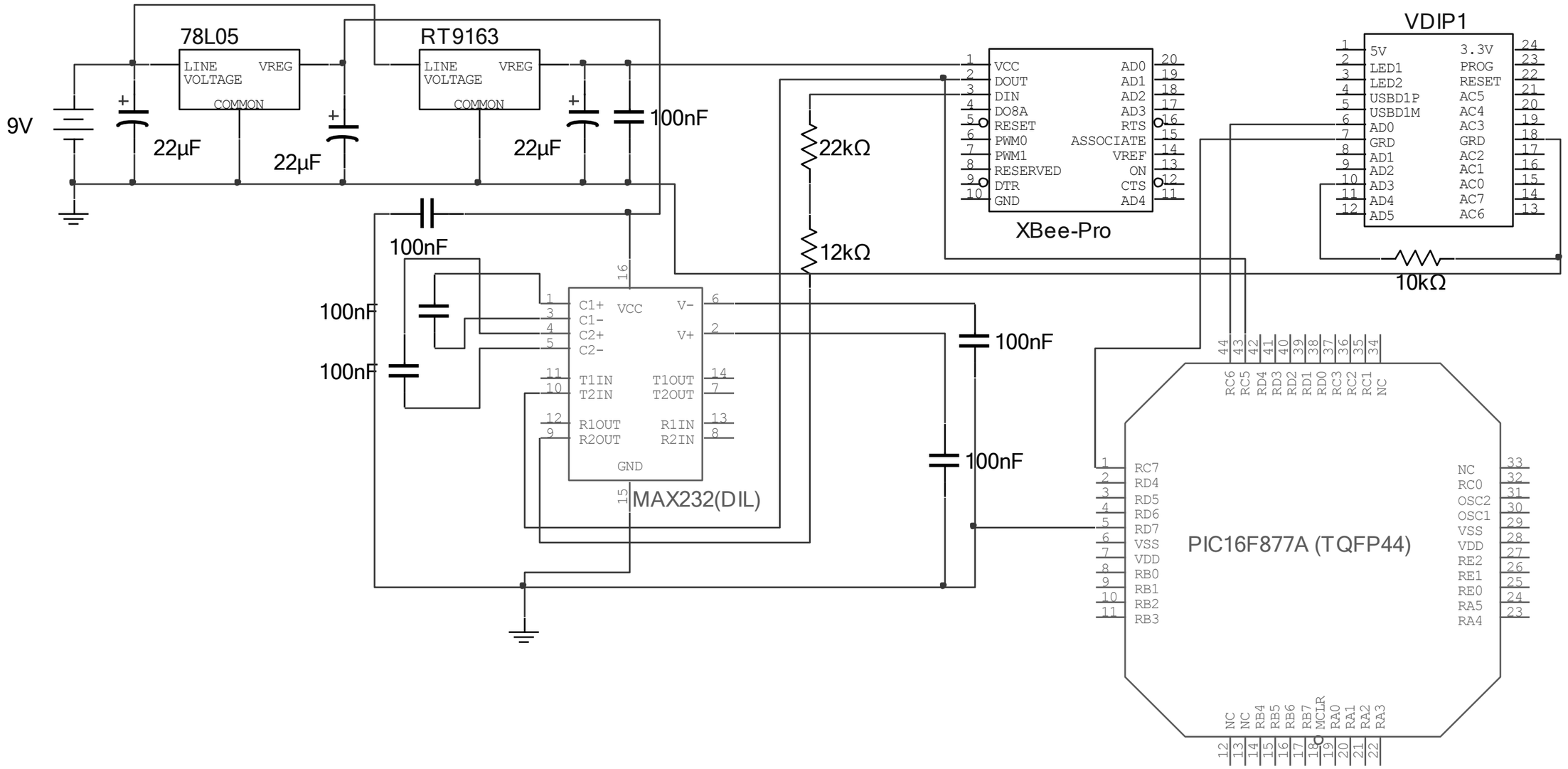


Figure 3.7 Schematic Diagram of the Handheld Device

Initially, a 5 V voltage regulator output is delivered through a low dropout positive voltage regulator with a dropout voltage of 3.3 Volts at 500 mA Amperes specifically for ZigBee supply pins. The input capacitance used suggests that a stable DC voltage will be guaranteed as an input for the regulator.

The 100nF capacitor is also used as a ripple rejection that will guarantee a stable DC voltage for the V_{CC} of the ZigBee circuit.

It is specified in the datasheet of the voltage regulator that an output capacitance of 22µF (electrolytic) or 10µF (tantalum) is required. In this case, the designers used 22µF electrolytic capacitor as higher output capacitance will improve the load transient response but have a maximum capacitance at 100µF.

A positive voltage regulator is used to transform 12 Volts, from two output pins DataOut and Vcc of ZigBee, to 5 Volts that will be used for the supply pins of VDIP1 USB interface.

Prototype Development

This includes the materials and components used and its function in the design.

List of Materials

Component	Quantity	Price per unit	Total Amount
ZigbeePro	2	2,750	5,500
ADXL330 Accelerometer	1	650	650
Stethoscope	1	180	180
VDIP1	1	2,800	2,800
35V – 22	5	5	25
2 pins connector male/female	3	10	30
24 Pins IC Socket	1	12	12
PIC 16F877A	2	320	640
40 Pins IC Socket	1	15	15
Crystal 3.92	1	30	30
Capacitor 33 uF	2	2	4
Capacitor .1	10	2.50	25
¼ W – 22k resistor	5	1	5
16V – 470	1	10	10
7805	1	30	30
7 pins connector male/female	1	22	22
8 pins connector male/female	1	25	25
RT9163	1	50	50
Max 232	1	50	50
16 pins IC socket	1	8	8
LE50	1	10	10
IN4148	1	3	3
¼ W- 3k Resistor	1	1	1
3 pins connector male/female	1	12	12
Casing	1	180	180
9 V Battery	3	300	900
Battery Charger	1	250	250

Table 3.10 List of Materials

Accelerometer

Accelerometer will be used to sense the pulse rate and gestures of the user. The model of the accelerometer used is ADXL330. It is a small, thin, low power, complete 3-axis accelerometer with signal conditioned voltage outputs, all on a single monolithic IC. It measures acceleration and can measure the static acceleration of gravity in tilt-sensing applications, as well as dynamic acceleration resulting from motion, shock, or vibration.

VDIP1

VDIP1 is ideal for developing and rapid prototyping which makes this module suitable for incorporation into low and medium volume finished product designs. It is a module that is an MCU to embedded dual USB host controller IC device. This will be used in the design as the USB interface which serves as the receiver.

PIC16F877

A microcontroller was used to act as a computer to the design. This includes processor core and the programmable input / output peripherals with a maximum of 33 I/O pins. This will automatically control the device that is dependent on how it is programmed using PicBasic Language.

ZigBee Wireless Technology

ZigBee is intended to be simpler and less expensive than other WPANs like Bluetooth. It is targeted at radio-frequency (RF) applications that require a low data rate, long battery life, and secure networking. This will act as the transceiver of the design because of its wide range and low consumption, features that are suitable in the researcher's design.

Improvised Stethoscope

An improvised stethoscope is designed to be in the measurement of the pulse rate in which a transducer converts the Korotkoff sounds into electrical signals. The electrical signals are amplified and fed to a counter in which the detected pulse rate per unit time is calculated and then the result is displayed as a digital pulse rate.

Capacitor

Capacitors are generally used in electronic circuits for blocking direct current, allowing the alternating current to pass in filter networks and for smoothing the output of power supplies. In this design different values of capacitors used are 0.47μF, 470μF, 4.7μF, 22μF, 33pF, 100nF and 0.1μF.

Operational Amplifiers

Amplifiers are used to increase or augment low signals until surpassing a threshold for use as a data. Two types of amplifiers are used in conjunction with the Improvised Stethoscope LM741 and TL072CN.

Diode

Diode allows current in one direction and to block current in the opposite direction. The diodes used are 1N4148.

Resistor

Resistors determine the flow of current in a circuit. The resistors used are 47kΩ, 2.2kΩ, 33kΩ, 56kΩ, 220Ω, 22kΩ, 12kΩ, and 10kΩ.

Battery

Batteries are responsible for energizing the whole circuit. Batteries used are two 9V for the wrist device, one 9V for the handheld device, and two 1.5 V or AAA batteries for the Lap Indicator module.

Voltage Regulator

Voltage regulators are used to sustain the voltage passing through a circuit within an amount the regulator is built for. The voltage regulators used are 78L05 and LM/805RC.

Push Button

A push-button is a simple switch device for controlling some aspects of a machine. It is used to activate or enable the design.

Crystal Oscillator

Crystal oscillator's primary considerations are stability and accuracy of the electronic circuit.

Chapter 4

TESTING, PRESENTATION AND INTERPRETATION OF DATA

This chapter gives the details on how the system was tested in relation to the design objectives.

Data Transmission Test

Since the system is wireless, the gathered data such as the time the user able to finish a lap and the pulse rate can be transmitted to the handheld device using the Zigbee module.

Distance (m)	Red LED	Packet Transmission
0	On	6-Byte Data Written
100	On	6-Byte Data Written
200	On	6-Byte Data Written
300	On	6-Byte Data Written
400	On	6-Byte Data Written
500	On	6-Byte Data Written
600	On	6-Byte Data Written
700	On	8-Byte Data Written
800	On	6-Byte Data Written
900	On	6-Byte Data Written
1000	On	6-Byte Data Written
1100	On	6-Byte Data Written
1200	On	8-Byte Data Written
1300	On	6-Byte Data Written
1400	On	8-Byte Data Written
1500	On	9-Byte Data Written
1600	Blinking	Failed to Write

Table 4.1 Data Transmission Test

Table 4.1 deals on the transmission of the gathered data in relation to the distance between the wearable device and the handheld device. The testing took place in an oval wherein the trainee will run. The trainee should observe proper posture of running. The handheld device can be placed within the range of the Zigbee module given that it is line of sight. On the other hand, the lap indicator module is fixed on the starting point and at the same time it will be on the ending point. Once the user stops from running, the user must refrain from swaying his/her hands in order for the device to measure the pulse rate and to transfer the data to the handheld device. The distance of running will be varied. The distance is then increased at an increment of 100 meters until it reached the maximum distance the Zigbee module can cover. Above 1.5 km, the data could not be transmitted to the handheld device.

If red LED is on, it indicates that a connection is established between the ZigBee devices. Otherwise, the red LED will continuously blink. Data Packet is transmitted and written to the flash drive whenever the flash drive LED blinks, otherwise transmission failed. The packet consists of the lap number and time, but accepts more than 6 bytes whenever there are ASCII conversion errors.

Time Measurement Test

The purpose of this test is to determine if the gathered data are accurate. Time is an essential factor in evaluating the speed of the user. The test will measure the time to finish a lap using the microcontroller's timer interrupt and

lap indicator module. To compare the readings of the microcontroller, a stop watch will be used.

Trial	Lap	Stopwatch Reading (s)	Microcontroller Reading (s)	Percent Difference (%)
1	1	17.66	17	3.81
2	1	18.08	18	0.44
	2	18.62	18	3.39
3	1	18.45	19	2.94
	2	18.92	19	0.42
	3	19.33	19	1.72
4	1	18.73	18	3.97
	2	19.02	19	0.11
	3	19.45	20	2.79
	4	20.05	20	0.25
5	1	17.58	17	3.35
	2	18.02	18	0.11
	3	18.44	19	2.99
	4	19.30	19	1.57
	5	19.87	21	5.53

Table 4.2 Time Measurement Test

Average Percent Difference

$$= \frac{3.81 + 0.44 + 3.39 + 2.94 + 0.42 + 1.72 + 3.97 + 0.11 + 2.79 + 0.25 + 3.35 + 0.11 + 2.99 + 1.57 + 5.53}{15}$$

= 2.23 %

Table 4.2 deals with the comparison of readings between the timer interrupt of the microcontroller and a stop watch. The average percent difference is 2.23% which is less than the set minimum required percent difference by the group (5%). Therefore, it says that the time measurement

reading between the microcontroller and stop watch shows minimal discrepancy. The discrepancy may be due to the reading of time in microcontroller which is whole number while in stop watch is in decimal form.

Pulse Rate Measurement Test

The purpose of this test is to determine if the pulse rate reading of the improvised stethoscope is accurate. Once the accelerometer senses that there is no hand movement it then sends signal to the microcontroller to stop the timer interrupt and the microcontroller will be the one to send signal to the improvised stethoscope to start measuring the pulse rate. Since the pulse rate gathered will be used in evaluating the individual's endurance along with the inputs namely age and resting heart rate using Karvonen formula. To compare the readings of the pulse rate measured by an improvised stethoscope, a digital blood pressure device will be used. At this point, the comparison must be done immediately after the trainee stops because pulse rate varies when a person is at rest or moving. Also, it varies depending on the activity a person is performing.

Trial	Digital Blood Pressure Device (bpm)	Improvised Stethoscope (bpm)	Percent Difference (%)
1	94	98	4.17
2	108	102	5.71
3	101	103	1.96
4	98	100	2.02
5	106	104	1.90

Table 4.3 Pulse Rate Measurement Test

Average Percent Difference

$$= \frac{4.17 + 5.71 + 1.96 + 2.02 + 1.90}{5}$$

=3.15 %

Table 4.3 deals with the comparison of readings between a digital blood pressure device and an improvised stethoscope. The average percent difference is 3.15% which is less than the set minimum required percent difference by the group (5%). The discrepancy may be due to some noise being measured by the improvised stethoscope. The improvised stethoscope based its reading from the sound that the pulse is producing so noise will be a factor if the improvised stethoscope is not properly placed in the wrist. It should be air tight so that it can properly hear the pulse rate.

Endurance Test

To evaluate the individual's endurance, Karvonen formula is one of the most effective methods used to calculate training heart rate. Prior to getting out of bed in the morning, the trainee must take his pulse which is the resting heart rate. Another element in finding the training heart rate zone is determining the intensity level at which the trainee should exercise. As a general rule, the trainee should exercise at an intensity between 50% - 85% of the heart rate reserve. Given all the information needed, it can now pull the information together in the Karvonen Formula:

Maximum Heart Rate = ((220 – Age – Resting Heart Rate) x 85%) + Resting Heart Rate

Minimum Heart Rate = ((220 – Age – Resting Heart Rate) x 50%) + Resting Heart Rate

Training Day No.	Lap	Time (s)	Resting Heart Rate (bpm)	Karvonen Formula		Pulse Rate (bpm)	Result
				Minimum (bpm)	Maximum (bpm)		
1	1	42	65	132.5	179.75	168	Good
	2	40					
	3	39					
	4	37					
	5	39					
2	1	42	71	135.5	180.65	165	Good
	2	36					
	3	36					
	4	40					
	5	34					
3	1	37	76	138	181.4	166	Good
	2	37					
	3	38					
	4	37					
	5	37					
4	1	42	70	135	180.5	170	Good
	2	38					
	3	34					
	4	35					
	5	36					
5	1	33	79	139.5	181.85	159	Good
	2	31					
	3	30					
	4	35					
	5	37					
6	1	33	77	138.5	181.55	160	Good
	2	36					
	3	29					
	4	28					
	5	31					
7	1	31	81	140.5	182.15	166	Good
	2	30					
	3	29					
	4	27					
	5	33					
8	1	31	75	137.5	181.25	155	Good
	2	28					
	3	29					
	4	30					
	5	30					
9	1	30	67	133.5	180.05	162	Good
	2	27					
	3	23					
	4	25					
	5	32					
10	1	31	72	136	180.8	152	Good
	2	31					
	3	24					
	4	26					
	5	25					
11	1	26	83	141.5	182.45	138	Very Good
	2	23					
	3	24					
	4	28					
	5	31					
12	1	25	85	142.5	182.75	141	Very Good
	2	24					
	3	27					
	4	26					
	5	27					

Table 4.4 Endurance Test

The Karvonen formula used in the test produces a maximum and minimum heart rate in which the training heart rate is expected to be within that range. In evaluation, any heart rate inside the Karvonen range is considered good endurance for an individual. Assuming trainees put enough effort in their trainings, heart rates less than the minimum are regarded as very good because the training has a minimal effect on the individual. Heart rates that are greater than the maximum are then considered as bad endurance.

Speed Test

To evaluate the speed of the trainee, the time gathered along with the input distance will be used. Since time will be used as the parameter to evaluate the speed of the trainee, the stopwatch and accelerometer reading will be used to compare the accuracy of the speed. Given the time and distance, the speed can be computed as:

$$\text{Speed} = \text{Distance} / \text{Time}$$

Trial	Distance (m)	Stopwatch Reading (s)	Speed (time reading from Stopwatch) (m/s)	Microcontroller Reading (s)	Speed (time reading from Microcontroller) (m/s)	Percent Difference (%)
1	50	17.66	2.83	17	2.94	3.81
2	100	36.7	2.72	36	2.78	2.18
3	150	56.7	2.65	57	2.63	0.76
4	200	77.25	2.59	77	2.60	0.39
5	250	93.21	2.68	94	2.66	0.75

Table 4.5 Speed Computation Using the Time Reading from the Stopwatch and the Microcontroller

Average Percent Difference

$$= \frac{3.81 + 2.18 + 0.76 + 0.39 + 0.75}{5}$$

=1.58 %

Table 4.5 deals with the computed speed using the measured time of the stopwatch and the microcontroller. Since the speed is dependent on the measured time, therefore the speed is directly proportional to the measured time. The average percent difference is 1.58% which is less than the set minimum required percent difference by the group (5%). Therefore we can say that the time measurement reading between the microcontroller and stop watch shows minimal discrepancy.

Chapter 5

CONCLUSION AND RECOMMENDATION

This chapter lays the overall conclusion of the design by means of answering the objectives of the design problem. In addition to this, it includes the statements that suggest the need for supplementary studies addressing to the delimitations of the design. The recommendation cites what else can be done for the improvement of the design.

CONCLUSION

The group was successful in designing the Wearable Device for Speed and Endurance Assessment and Monitoring System using Improvised Stethoscope and Accelerometer with USB Receiver via ZigBee Technology. With the aid of improvised stethoscope and microcontroller's timer interrupt, it made possible for the device to measure the heart rate and pace of the user respectively.

A software program named "SEAMS" (Speed and Endurance Assessment and Monitoring System) was effectively programmed using the Visual Basic Programming Language. This can assess and monitor the speed and endurance as stated as one of the researcher's design objectives.

The group was also successful in achieving a wireless transmission of data using Zigbee Technology.

Upon testing, the researchers could say that the Zigbee transmission is limited only to a range of 1500 meters. Comparing the test results of the improvised stethoscope and accelerometer with the digital blood pressure device and stopwatch respectively, the difference was really small. Gathering those results, it could be concluded that the design of the researchers made was accurate and would be effective to assess and evaluate the individual's speed and endurance.

RECOMMENDATION

To be able the design to be more efficient, effective and flexible as technology advances, further studies must need to be made. The following are recommended by the group for more improvements of the design.

Further studies should be done to solve the delimitation of the project with regards to having an output although the user is not exercising correctly. Another device must also be studied that will serve as an alternative to the infrared as a result for the user not to be close with the infrared just to be able to send a signal to the handheld to write the data gathered on the USB.

The design can also be improved by adding a database on its application software to help keep records of data for comparison of the results. The

database would allow flexibility by recording or deleting records of trainees for future use or reference.

Lastly, it is recommended that the design may also use a sensor that will be located on the finger of the user on the wearable device as an alternate device to the improvised stethoscope. This can also give accurate measurement of pulse rate because there are also veins in the finger tips that are capable of producing a pulse.

REFERENCES

Bookhout, Mark R., and Grenman, Philip. Principles of Exercise Prescription. Woburn, MA: Butterworth-Heinemann. 2001

Charles S. Electronics principles and applications, 7th edition. USA. 2007.

Chen, Tseng, et al. "Monitoring Athletes' Physiological Responses to Endurance Training with Genomic-wide Expression Data" Citeseerx. September 2, 2007

Cheng, L. "Practical Sensing for Sprint Parameter Monitoring" Sensor Mesh and Ad Hoc Communications and Networks (SECON). June 2010

Cheng, L. Hailes, S. "Analysis of Wireless Inertial Sensing for Athlete Coaching Support" Global Telecommunications Conference. IEEE GLOBECOM. Dec. 4, 2008

Darcy Calabro, et al. "Accuracy of Armband Monitors for Measuring Daily Energy Expenditure in Healthy Adults" Medicine & Science in Sports & Exercise. November 2010: pp 2134-2140

David Geer. "Users Make a Beeline for ZigBee Sensor Technology" Computer. Dec. 2005: p. 16

DE VRIES, SANNE I. et al. "Evaluation of Neural Networks to Identify Types of Activity Using Accelerometers" Medicine & Science in Sports & Exercise. January 2011: pp. 101-107

En Wei, Huang Li, Chen Fu. "Gesture Stroke Recognition Using Computer Vision and Linear Accelerometer" Automatic Face & Gesture Recognition. 19 Sept. 2008

Fabio L. Zucatto, et al. "ZigBee for Building Control Wireless Sensor Networks" Microwave and Optoelectronics Conference. IMOC. Nov. 1 2007: p. 511

Guang-zhong Yang. "Sports Body Sensor Networks" Medical Devices and Biosensors. June 2008: p.14

Jang, I.J. Park, W.B. "Signal Processing of the Accelerometer on Handheld Devices" Robot and Human Interactive Communication. November 2003: p.139

Khalil and Sornanathan. "Fitness monitoring system based on heart rate and SpO2 level". November 2010

Moron, M.J. et al. "A Wireless Monitoring System for Pulse-oximetry Sensors" Systems Communications. 17 Aug. 2005: p. 79

Ohyama, et al. "Electric cart control system for adjustable pedal load using dynamic-parallel-distributed-compensation method". October 2007

Redding, Morgan. Physical Fitness: Concepts and Applications. Dubuque, IA: Kendall/Hunt Publishing. 2001

Young & Freedman, University Physics , 10th Ed. 2004

Youngbum Lee, et al. "Implementation of Accelerometer Sensor Module and Fall Detection Monitoring System based on Wireless Sensor Network" Engineering in Medicine and Biology Society. Aug. 2007: p. 2315

Zhenyu He, Lianwen Jin, et al. "Gesture recognition based on 3D accelerometer for cell phones interaction" Circuits and Systems. APCCAS. IEEE Asia Pacific Conference. Dec. 3, 2008

APPENDIX A

Operation's Manual

System Requirements:

	Minimum	Recommended
Operating System	Windows XP Service Pack 1	Windows 7
Hardware	2 GB HDD 192 MB RAM 1.6 GHz CPU 800 x 600 Display	50 GB HDD 1 GB RAM 4.0 GHz CPU 1280 x 1024 Display

Installation Procedure

The following procedures must be followed carefully to ensure that the installation process is successfully done:

1.) Install the accompanied software on the host PC before the training. The device needed the software to record the information of the trainee and also to evaluate the results gathered from the wearable device.

2.) Place the handheld device on a stable spot wherein it is within the 1.5 km range of the Zigbee module. Be sure that it is line of sight.

User's Manual

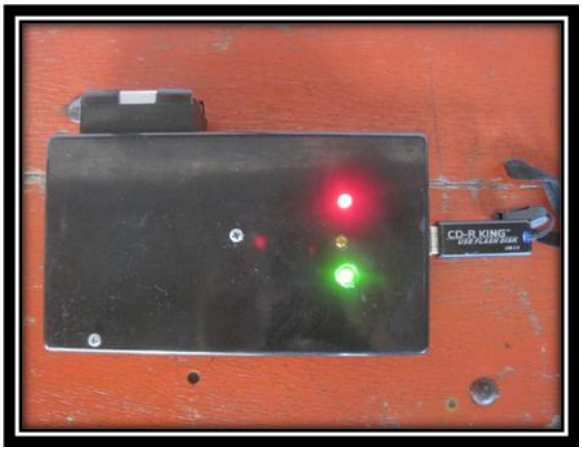
1.) Place the lap indicator module in a fixed position wherein it will be the starting point and at the same time it will be the ending point.

2.) Put the wearable device on your wrist. Make sure that it is tight. Put the battery clip on the batteries.

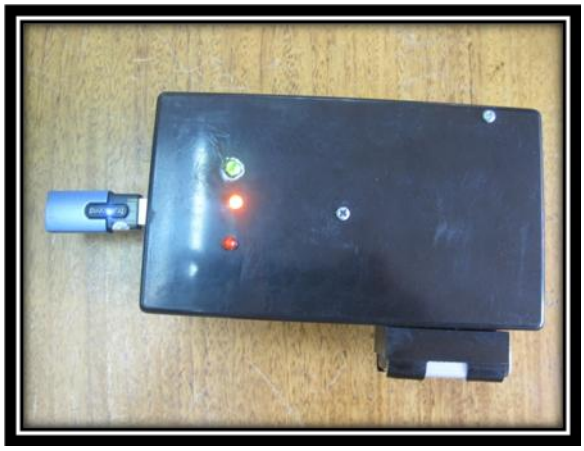
3.) Turn the handheld device on by putting the battery clip on the battery. The red and orange LEDs will indicate if the device is turned on.

4.) Connect the flash drive to a USB port in the handheld device. A green LED will indicate if the flash drive is recognized.

5.) Once the red LED stops from blinking, the user must press the button in the wearable device to indicate the start of running.



6.) Once the user stops from running, the user must not make unnecessary move for 20 seconds in order for the device to measure the pulse rate and transfer the data to the handheld device. An orange LED will indicate that the pulse rate was transmitted to the handheld device.



7.) To evaluate the speed and endurance of the user, connect the flash drive to the host PC. Then run the accompanied SEAMs program which is installed earlier in the host PC.

Troubleshooting Guides and Procedures

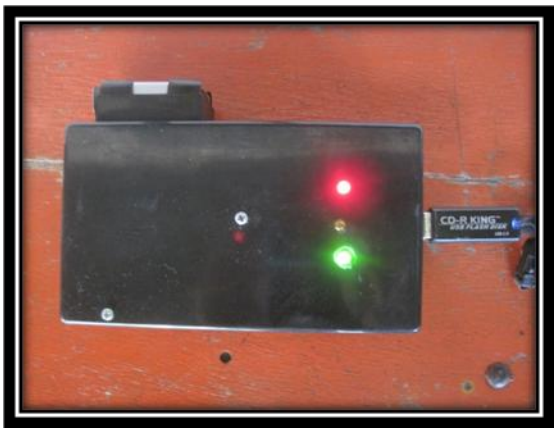
1.) Check if the red LED in the handheld device is lit. This will indicate if the device is on. If not, make sure that the battery is fully charged.
2.) Check if the green LED in the handheld device is lit. This will indicate if the flash drive is detected. If not, replace the flash drive.
3.) If the training graph could not evaluate the file in the software application, this indicates that an error has occurred.
4.) If the handheld device is working properly but the orange LED did not light, consider replacing the batteries in the wearable device.

APPENDIX B

Pictures of Prototype

Wrist Device

Handheld Device



Lap Indicator Module

Batteries and Charger

APPENDIX C

Program Listing

Microcontroller Source Code

- **Heart Rate Measurement Source Code**

```
Device 16F877A
Declare Xtal 20
Declare Watchdog = OFF
Declare FSR_CONTEXT_SAVE = On
Declare Adin_Res        8
Declare Adin_Tad        32_FOSC
Declare Adin_Stime      50
Remarks On
Symbol T0IE = INTCON.5          ' TMR0 Overflow Interrupt Enable
Symbol T0IF = INTCON.2          ' TMR0 Overflow Interrupt Flag
Symbol GIE = INTCON.7           ' Global Interrupt Enable
OPTION_REG = $03
On_Interrupt My_Int        ' Enable software interrupts, and point
to interrupt handler
'TRISA = %00000001              '  Configure  AN0  (PORTA.0)  as  an
input
Hserial_Baud = 9600'115200
Hserial_RCSTA = %10010000
Hserial_TXSTA = %00100100
Hserial_Clear = On
'LCD configuration
Declare  LCD_DTPin PORTC.0
Declare  LCD_RSPin PORTD.0
Declare  LCD_ENPin PORTD.1
Declare LCD_Lines 2
Declare LCD_Interface 4
ADCON1 = $00    ' Set analogue input on PORTA.0
TRISA=$3F
TRISB=$0F
TRISC=$80
TRISD=$00
Dim xCtr As Word
Dim aRead As Byte
Dim MaxADC  As Byte
Dim FTime As Float
Dim PRate As Word
Dim sADC[250] As Byte
Dim x As Byte
Dim ctr As Word
Dim b As Byte
 'On Interrupt
   TMR0= 0
   GIE = 1
   T0IE= 1
 sStartHere:
  GIE = 1
  T0IE= 1
  xCtr=0
  x=0
  MaxADC=0
  FTime=0
  ctr=0
```

```
b=0
PRate=0
xCtr=0
 HSerOut["improper settings",13]
sMain:
'Junk Reference
While 1
         If xCtr > 5000 Then
              If MaxADC < 90 Then 'Default 100
                 GoTo sStartHere
              Else
                     GoTo sNew2
              EndIf
         Else
             aRead = ADIn 0
               If MaxADC > aRead Then
               Else
                    MaxADC=aRead
               EndI
         EndIf
 Wend
 'Check ADC Baseline 0
 sNew2:
 xCtr=0
 While 1
      aRead = ADIn 0
        If aRead <= 45 Then
            Break
        EndIf
 Wend
 MaxADC=MaxADC-10
 'Check ADC Max If Less Than aRead ADC
 sNew3:
 While 1
      aRead = ADIn 0
        If MaxADC < aRead Then
         xCtr=0
            Break
        EndIf
 Wend
 'Delay before reading
 sNew4:
 While 1
          If xCtr > 500 Then 'default 25
              Break
          EndIf
 Wend
 x=0
 ctr=0
 'Get Final Pulse Rate
 sFindADC:
 While 1
          aRead=ADIn
           sADC[x]=aRead
             x=x+1
          sRepeat:
          If ctr >=3 Then
             ctr=0
                 If x >=250 Then
                   Break
                EndIf
             GoTo sFindADC
           Else
                   GoTo  sRepeat
```

```
            EndIf
   Wend
   sSENDUART:
      GIE=0
      T0IE= 0
         sADC[0]=MaxADC
        For x=0 To 249
            'HSEROUT[sADC[x]]
               'HSEROUT[Dec3 sADC[x],32]
         Next x
           For x=1 To 249
             If sADC[x] >= MaxADC Then
                   Break
              Else
              EndIf
          Next x
          If x = 249 Then
          Else
               'HSEROUT[Dec3 x,"OK"]
          EndIf
              'Compute Pulse Rate
                 FTime = x * 2.4
              FTime = FTime + 400
              FTime = FTime / 1000
              FTime = 60 / FTime
                 'Cls
                 'Print at 1,1,"Pulse Rate"
                 'Print at 2,1,Dec5 PRate
                 HSerOut["HEARTRATE:",Dec3 FTime,13]
      GoTo sStartHere
 'Interrupt Routine
 My_Int:
  xCtr=xCtr + 1
  ctr=ctr+1
  PORTD.3=~PORTD.3
  T0IF=0
  Context Restore
End
```

➢ Lap Measurement Source Code

```
Remarks On
Unsigned_Dwords On
Device 16F877A
Declare Xtal 20
Declare Watchdog = OFF
Declare FSR_CONTEXT_SAVE = On

Declare Adin_Res        8
Declare Adin_Tad        32_FOSC
Declare Adin_Stime      50

Hserial_Baud = 9600
Hserial_RCSTA = %10010000
Hserial_TXSTA = %00100100
Hserial_Clear = On

On_Interrupt iHandler

Symbol INTF = INTCON.1 ' RB0 External Interrupt Flag
Symbol T0IF = INTCON.2 ' TMR0 Overflow Interrupt Flag
```

```
Symbol INTE = INTCON.4 ' RB0 External Interrupt Enable
Symbol T0IE = INTCON.5 ' TMR0 Overflow Interrupt Enable
Symbol GIE = INTCON.7  ' Global Interrupt Enable

Dim dsDelay As Word
Dim axCtr As Word
Dim lapCtr As Word
Dim testval As Word
Dim runF As Byte

Dim xCtr As Word
Dim ctr As Word
Dim sc As Word

Dim rndFlag As Byte, rndFClock As Word, sendFlag As Byte
Dim rBPM As Byte

rBPM = 90

GoTo preProg

iHandler:

If T0IF = 1 Then

    axCtr = axCtr + 1

    dsDelay = dsDelay + 1

    lapCtr = lapCtr + 1

    xCtr=xCtr + 1

    ctr=ctr+1

    rndFClock = rndFClock + 1

    If rndFClock > 17647 Then

        If rndFlag = 1 Then
            sendFlag = 1
            rndFlag = 2
        EndIf

    EndIf

    If rBPM > 105 Then

        rBPM = 90
    Else
        rBPM = rBPM + 1

    EndIf

    T0IF=0

EndIf

Context Restore

preProg:

Dim cARead1 As Byte
Dim cARead2 As Byte
```

```
Dim pARead1 As Byte
Dim pARead2 As Byte

Dim sADC[250] As Byte
Dim aRead As Byte
Dim MaxADC  As Byte
Dim FTime As Float
Dim PRate As Word

Dim FTDec As Dword
Dim LapC As Byte

Dim x As Byte
Dim b As Byte

ADCON1 = $00
TMR0 = 0

OPTION_REG = $03
TRISA = $3F
TRISB = $F7
TRISC = $80
TRISD = $00
TRISE = $07

DelayMS 500

T0IF = 0
T0IE = 0
GIE = 0

'while 1 = 1

'    hserout["U"]
'    delayms 500

'wend

startHere:

lapCtr = 0
LapC = 0
runF = 1
rndFlag = 0
sendFlag = 0
PORTB.3 = 1
While 1 = 1

    If PORTB.7 = 0 Then

        DelayMS 50
        While PORTB.7 = 0
        Wend
        DelayMS 50

        'pARead1 = ADIn 1
        pARead2 = ADIn 2
        axCtr = 0
        PORTB.3 = 0
        GoTo startRunnin

    EndIf

Wend
```

```
startRunnin:

T0IF = 0
T0IE = 1
GIE = 1


      testval = 0

      While 1 = 1

        testval = Counter PORTB.0,40

              'caread1 = adin 1
      cARead2 = ADIn 2

      If cARead2 > pARead2 Then

        cARead1 = cARead2 - pARead2

      ElseIf cARead2 < pARead2 Then

        cARead1 = pARead2 - cARead2

      Else

        cARead1 = 0

      EndIf

      If cARead1 < 10 Then

        'not moving

        If axCtr > 2940 Then

            runF = 0

            If rndFlag = 0 Then

                sendFlag = 0
                rndFClock = 0
                rndFlag = 1

            EndIf

            GoSub heartRateMeter

            'HSerOut["GET HEARTRATE!",10,13]

            'stopBPM:
            'GoTo stopBPM

        EndIf

      Else

        'reset movement counter

        axCtr = 0

        'set current reading as previous
```

```
        pARead2 = cARead2

      EndIf

        If testval > 20 And runF = 1 Then

            'hit!
            GIE = 0

            LapC = LapC + 1

            HSerOut["PL", Dec3 LapC, ":", Dec5 lapCtr / 1176 ,10,13]

            lapCtr = 0
            GIE = 1

        EndIf

      Wend

GoTo startHere

heartRateMeter:

 sStartHere:

  GIE = 1
  T0IE= 1

  xCtr=0
  x=0
  MaxADC=0
  FTime=0
  ctr=0
  b=0
  PRate=0

  xCtr=0

  'HSerOut["improper settings",13]

  sMain:

   If sendFlag = 1 Then

      HSerOut["PHBPM:",Dec5 rBPM,10,13]
      sendFlag = 0
      GoTo eHRM

   EndIf

  'Junk Reference
  While 1 = 1

          If xCtr > 5000 Then

               If MaxADC < 90 Then 'Default 100

                  GoTo sStartHere

               Else

                     GoTo sNew2
```

```
             EndIf

         Else

             aRead = ADIn 0

               If MaxADC > aRead Then

               Else
                    MaxADC=aRead

               EndIf

         EndIf

Wend


'Check ADC Baseline 0
sNew2:

xCtr=0


While 1 = 1

If sendFlag = 1 Then

   HSerOut["PHBPM:",Dec5 rBPM,10,13]
   sendFlag = 0
   GoTo eHRM

EndIf


     aRead = ADIn 0

       If aRead <= 45 Then

           Break

       EndIf

Wend

MaxADC=MaxADC-10

'Check ADC Max If Less Than aRead ADC
sNew3:

While 1 = 1

If sendFlag = 1 Then

   HSerOut["PHBPM:",Dec5 rBPM,10,13]
   sendFlag = 0
   GoTo eHRM

EndIf

     aRead = ADIn 0

       If MaxADC < aRead Then
```

```
         xCtr=0
             Break

        EndIf

Wend

'Delay before reading
sNew4:

While 1 = 1

          If xCtr > 500 Then 'default 250


              Break


          EndIf


Wend

x=0
ctr=0

'Get Final Pulse Rate
sFindADC:



While 1 = 1

If sendFlag = 1 Then

   HSerOut["PHBPM:",Dec5 rBPM,10,13]
   sendFlag = 0
   GoTo eHRM

EndIf


          aRead = ADIn 0

           sADC[x]=aRead

             x=x+1

          sRepeat:

          If ctr >=3 Then

             ctr=0


                 If x >=250 Then


                   Break

                EndIf

             GoTo sFindADC
```

```
          Else

                  GoTo  sRepeat

          EndIf

Wend


sSENDUART:

   GIE=0
   T0IE= 0


      sADC[0]=MaxADC

     For x=0 To 249

         'HSEROUT[sADC[x]]
            'HSEROUT[Dec3 sADC[x],32]

      Next x

        For x=1 To 249

          If sADC[x] >= MaxADC Then

                Break

             Else

             EndIf

       Next x

       If x = 249 Then

       Else

            'HSEROUT[Dec3 x,"OK"]

       EndIf

           'Compute Pulse Rate
              FTime = x * 2.4
           FTime = FTime + 400
           FTime = FTime / 1000
           FTime = 60 / FTime

              'Cls

              'Print at 1,1,"Pulse Rate"
              'Print at 2,1,Dec5 PRate

              FTDec = FTime * 100
              FTDec = FTDec / 100

              'HSerOut["H:",Dec3 FTime,13]

              HSerOut["PHBPM:",Dec5 FTDec,10,13]
              rndFlag = 2
```

```
   eHRM:

   Return
      'GoTo sStartHere

End
```

- **Zigbee Source Code**

```
Device 16F877A
Declare Xtal 4
Declare Watchdog = OFF
Declare FSR_CONTEXT_SAVE = On
All_Digital TRUE
Remarks  On

'Declare LCD_DTPin          PORTC.0
'Declare LCD_RSPin          PORTD.0
'Declare LCD_ENPin          PORTD.1
'Declare LCD_Lines          2
'Declare LCD_Interface      4

Hserial_Baud = 9600
Hserial_RCSTA = %10010000
Hserial_TXSTA = %00100100
Hserial_Clear = On

TRISA = $FF
TRISB = $0F
TRISC = $80
TRISD = $AC
TRISE = $07

Dim gCtr As Byte
Dim strVar[13] As Byte
Dim keytemp As Byte
Dim accelX As Byte
Dim accelY As Byte
Dim accelZ As Byte
Dim HRate As Byte
Dim lapSec As Word
Dim lapCtr As Byte



DelayMS 500


'Cls
'Print At 1,1,"INITIALIZING"
'Print At 2,1,"DEVICE.."

PORTD.0 = 0
PORTD.1 = 0
PORTC.0 = 0

strVar[12]=0

DelayMS 5000
```

```
DRV_ERROR:
'PORTC.0 = 1
'DelayMS 1000
'PORTC.0 = 0
'DelayMS 5000



            PORTD.0 = 0
            PORTD.1 = 0
            DelayMS 500

            PORTD.0 = 1
            PORTD.1 = 1
            DelayMS 500

   SerOut PORTD.6, 84, [13]
   SerIn PORTD.5,84, 2000, DRV_ERROR, [Wait(">")]
   'SerIn PORTD.5,84, 3000, DRV_ERROR, [Str strVar\3]
      'Print At 1,1,Str strVar\3
      GoTo startHere


         'Cls
        'Print At 1,1,"USB SOURCE"
        'Print At 2,1,"MISSING."
   DelayMS 500


startHere:

'Cls
'Print At 1,1,"SELECT GENDER"
'Print At 2,1,"1.MALE 2.FEMALE"

'While 1 = 1

'    GoSub myNumKeypad

'    If keytemp = "1" Or keytemp = "2" Then

'        Break

'    EndIf

'Wend

'Cls
'Print At 1,1,"AGE:"

'gCtr = 0

'While gCtr < 2

'   GoSub myNumKeypad

'    If keytemp <> "*" And keytemp <> "#" Then

'        gCtr = gCtr + 1
'        Print At 1, gCtr + 4, keytemp

'    EndIf

'Wend
```

```
'DelayMS 2000

lapCtr = 0
lapSec = 0

            PORTD.0 = 1
            PORTD.1 = 1
            DelayMS 1500
            PORTD.0 = 0
            PORTD.1 = 0
            DelayMS 1500

            PORTD.0 = 1
            PORTD.1 = 1
            DelayMS 1500
            PORTD.0 = 0
            PORTD.1 = 0
            DelayMS 1500
            PORTD.0 = 0
            PORTD.1 = 1


'Cls
While 1 = 1

'HSerIn            [Wait("X:"),Dec3            accelX,Wait("Y:"),Dec3
accelY,Wait("Z:"),Dec3 accelZ, Wait("H:"),Dec2 HRate ]
'HSerIn [Wait(13),Str strVar\20]
'HSerIn [Wait("LAP"),Dec5 lapSec]

HSerIn [Wait("P"),Str strVar\12]
'lapCtr = lapCtr + 1


       If strVar[0]="L" Then

            PORTD.0 = 0
            PORTD.1 = 1


 '       lapCtr = lapCtr + 1
 '       Print At 1,1,"LAPS: ", Dec3 lapCtr
 '                                  Print    At    2,1,"TIME:    ",
strVar[5],strVar[6],strVar[7],strVar[8],strVar[9]

      Else

 '       Print At 1,1,"LAPS: ", Dec3 lapCtr
 '                                Print    At    2,1,"BPM    :    ",
strVar[5],strVar[6],strVar[7],strVar[8],strVar[9]

            PORTD.0 = 1
            PORTD.1 = 0

      EndIf


    GoSub WriteData


Wend

hrOK:
```

```
'GoSub myNumKeypad


GoTo startHere


WriteData:

          SerOut PORTD.6,84, [13]
          SerIn PORTD.5,84, 1000, DRV_ERROR, [Wait(">")]

          SerOut PORTD.6,84,  ["IPA",13]
          SerIn PORTD.5,84, 1000, DRV_ERROR, [Wait(">")]
          SerOut PORTD.6,84,  ["OPW BPM.TXT",13]
          SerIn PORTD.5,84, 1000, DRV_ERROR, [Wait(">")]

        'SerOut PORTD.6,84,  ["WRF 30",13]

          'SerOut                                              PORTD.6,84,
["HR:",strVar[17],strVar[18],strVar[19]," BPM", " | "]
          'SerOut                                              PORTD.6,84,
["X:",strVar[2],strVar[3],strVar[4],"Y:",strVar[7],strVar[8],strVar[
9],"Z:",strVar[12],strVar[13],strVar[14],13,10]

         SerOut PORTD.6,84,  ["WRF 12",13]
          SerOut PORTD.6,84,  [Str strVar\12]

          SerIn PORTD.5,84, 1000, DRV_ERROR, [Wait(">")]
          SerOut PORTD.6,84,  ["CLF BPM.TXT",13]
          SerIn PORTD.5,84, 1000, DRV_ERROR, [Wait(">")]

          Return


myNumKeypad:

While 1 = 1

    PORTB.4=0
    PORTB.5=1
    PORTB.6=1

                If PORTB.0=0 Then
                     DelayMS 100
                            PORTB.7=1
                     While PORTB.0=0
                            Wend
                            DelayMS 100
                            PORTB.7=0
                            keytemp="1"
                            Break

                ElseIf PORTB.1=0 Then

                            DelayMS 100
                            PORTB.7=1
                     While PORTB.1=0
                            Wend
                            DelayMS 100
                            PORTB.7=0
                            keytemp="4"
                            Break
```

```
                ElseIf PORTB.2=0 Then
                     DelayMS 100
                            PORTB.7=1
                     While PORTB.2=0
                            Wend
                            DelayMS 100
                            PORTB.7=0
                            keytemp="7"
                            Break


                ElseIf PORTB.3=0 Then
                     DelayMS 100
                            PORTB.7=1
                     While PORTB.3=0
                            Wend
                            DelayMS 100
                            PORTB.7=0
                            keytemp="*"
                            Break
                      EndIf


PORTB.4=1
PORTB.5=0
PORTB.6=1

                If PORTB.0=0 Then
                     DelayMS 100
                            PORTB.7=1
                     While PORTB.0=0
                            Wend
                            DelayMS 100
                            PORTB.7=0
                            keytemp="2"
                            Break

                ElseIf PORTB.1=0 Then
                     DelayMS 100
                            PORTB.7=1
                     While PORTB.1=0
                            Wend
                            DelayMS 100
                            PORTB.7=0
                            keytemp="5"
                            Break
                ElseIf PORTB.2=0 Then
                     DelayMS 100
                            PORTB.7=1
                     While PORTB.2=0
                            Wend
                            DelayMS 100
                            PORTB.7=0
                            keytemp="8"
                            Break

                ElseIf PORTB.3=0 Then
                     DelayMS 100
                            PORTB.7=1
                     While PORTB.3=0
                            Wend
                            DelayMS 100
                            PORTB.7=0
```

```
                        keytemp="0"
                        Break

                    EndIf
    PORTB.4=1
    PORTB.5=1
    PORTB.6=0
                If PORTB.0=0 Then
                DelayMS 100
                        PORTB.7=1
                While PORTB.0=0
                        Wend
                        DelayMS 100
                        PORTB.7=0
                        keytemp="3"
                        Break

            ElseIf PORTB.1=0 Then
                DelayMS 100
                        PORTB.7=1
                While PORTB.1=0
                        Wend
                        DelayMS 100
                        PORTB.7=0
                        keytemp="6"
                        Break
            ElseIf PORTB.2=0 Then
                DelayMS 100
                        PORTB.7=1
                While PORTB.2=0
                        Wend
                        DelayMS 100
                        PORTB.7=0
                        keytemp="9"
                        Break


            ElseIf PORTB.3=0 Then
                DelayMS 100
                        PORTB.7=1
                While PORTB.3=0
                        Wend
                        DelayMS 100
                        PORTB.7=0
                        keytemp="#"
                        Break

                    EndIf
    PORTB.4=0
    PORTB.5=1
    PORTB.6=1


Wend

Return


End
```

Software Application Form Scripts

History Form

```
Imports System.IO
Imports System.Drawing
Public Class frmBrowse
    Dim g As Graphics
    'Definition for creating graphics in picture box.
    Dim DataPath As String =
My.Computer.FileSystem.SpecialDirectories.Temp & "\Trainings"
    Dim ListPath As String =
My.Computer.FileSystem.SpecialDirectories.Temp & "\Clients"
    Dim LapTime(255), LapHR(255), HRrest(255) As Single
    'Integer array with one decimal place
    Dim Pos As Integer
    'Determines point of comparison in parsing
    Dim i, x As Integer
    'Declaration of subsitute variables
    Dim IntAdapt As Integer
    'Variable to align integers to the right side
    Dim x_scroll, y_scroll As Integer
    'Form variables for scroll bar movement
    Dim x_coord_new, x_coord As Integer
    Dim x_graph, y_graph, x_end, y_end, x_start, y_start As Single
    'Coordinates for plotting points
    Dim index, LargestTime, HeartRate, InitialDay, CurrentDay,
Training_Length As Single
    'Data-mapping variables
    Dim RestingHR, Age, AverageHR, Speed, Distance, TotalTime As Single
    'Heart Rate Computation Variables
    Dim YScroll As Boolean = False
    Dim response As MsgBoxResult
    'Variable to recieve the response in the message box
    Dim Graph_scale As Single
    Dim Graph_Length As Single
    Const x_allowance = 100, y_allowance = 75, drag_sensitivity = 50
    'Scales for graphing
    Private Sub Load_Graph()
        pcbGraph.Refresh()
        g = pcbGraph.CreateGraphics
        g.FillRectangle(Brushes.Black, -(pcbGraph.Location.X +
x_scroll), 0, x_allowance, LargestTime / graph_scale + y_allowance +
500)
        g.FillRectangle(Brushes.Black, x_allowance, LargestTime /
graph_scale + y_allowance, Graph_Length * Training_Length, 500)
        pcbGraph.BackColor = Color.White
        For Me.x = CurrentDay To 1 Step -1
            If File.Exists(DataPath & "\Day" & x & "_" & cmbName.Text &
cmbTraining.Text & ".end") = True Then
                UpdatePictureBox()
                Dim FileReader As StreamReader
                Dim MainStr, StrTemp As String
                FileReader = New StreamReader(DataPath & "\Day" & x &
"_" & cmbName.Text & cmbTraining.Text & ".end")
                MainStr = FileReader.ReadToEnd()
```

```
                FileReader.Close()
                index = 0
                Pos = 2
                While Pos <> 0
Search:
                    Pos = InStr(MainStr, "L", CompareMethod.Text)
                    If Pos = 0 Then
                        Pos = InStr(MainStr, "H", CompareMethod.Text)
                        If Pos = 0 Then
Invalid:
                            MessageBox.Show("Invalid File Format",
"Deleting Training", 0, _
                                            MessageBoxIcon.Exclamation,
0, 0, False)
                            Dim DayHandler As New FileInfo(DataPath &
"\Day" & x & "_" & _
                                                           cmbName.Text
& cmbTraining.Text & ".end")
                            DayHandler.Delete()
                            GoTo NextDay
                        End If
                        Pos = InStr(MainStr, ":", CompareMethod.Text)
                        StrTemp = Mid(MainStr, Pos, 6)
                        StrTemp = StrTemp.Trim(":")
                        If IsNumeric(StrTemp) Then
                            LapHR(x - 1) = StrTemp
                        Else
                            GoTo Invalid
                        End If
                        Pos = InStr(MainStr, "R", CompareMethod.Text)
                        If Pos <> 0 Then
                            MainStr = Mid(MainStr, Pos + 1)
                            Pos = InStr(MainStr, ":",
CompareMethod.Text)
                            StrTemp = Mid(MainStr, Pos, 4)
                            StrTemp = StrTemp.Trim(":")
                            If IsNumeric(StrTemp) Then
                                HRrest(x - 1) = StrTemp
                            End If
                        End If
                        Exit While
                    End If
                    Pos = InStr(MainStr, ":", CompareMethod.Text)
                    StrTemp = Mid(MainStr, Pos, 6)
                    StrTemp = StrTemp.Trim(":")
                    IntAdapt = StrTemp
                    If IntAdapt > 0 Then
                        LapTime(index) = StrTemp
                    ElseIf IntAdapt = 0 Then
                        MainStr = Mid(MainStr, Pos + 1)
                        GoTo Search
                    ElseIf Not IsNumeric(StrTemp) Then
                        GoTo Invalid
                    End If
                    MainStr = Mid(MainStr, Pos + 1)
                    index += 1
                End While
```

```
                TotalTime = 0
                For Me.i = 0 To index - 1
                    TotalTime += LapTime(i)
                    If LapTime(i) > LargestTime Then
                        LargestTime = LapTime(i)
                    End If
                Next i
                If LargestTime > 5000 Then
                    graph_scale = 10
                ElseIf LargestTime <= 5000 And LargestTime > 1000 Then
                    graph_scale = 5
                ElseIf LargestTime <= 1000 And LargestTime > 500 Then
                    graph_scale = 2
                ElseIf LargestTime <= 500 And LargestTime > 100 Then
                    graph_scale = 1
                ElseIf LargestTime <= 100 And LargestTime > 50 Then
                    graph_scale = 0.5
                ElseIf LargestTime <= 50 And LargestTime >= 0 Then
                    graph_scale = 0.1
                End If
                Speed = Distance / (TotalTime / index)
                Speed = Math.Round(Speed, 2)
                Dim x_reference As Single
                x_reference = (Graph_Length * (x - 1)) + x_allowance
                x_graph = Graph_Length / (index)
                y_graph = LargestTime / graph_scale
                g.DrawString("Time", Font, Brushes.White, 13 -
(pcbGraph.Location.X + x_scroll), pcbGraph.Height / 2 - y_allowance)
                g.DrawString("(Seconds)", Font, Brushes.White, -
(pcbGraph.Location.X + x_scroll), pcbGraph.Height / 2 - 10)
                g.DrawString("Laps", Font, Brushes.White, x_reference +
                             Graph_Length / 2 - 35, y_allowance +
LargestTime / graph_scale + 5)
                g.FillRectangle(Brushes.White, x_reference +
Graph_Length / 2 - 45, y_allowance + LargestTime _
                                / graph_scale + 30, 50, 50)
                g.DrawString("Day " & x, Font, Brushes.Black,
x_reference + Graph_Length / 2 _
                                - 40, y_allowance + LargestTime /
graph_scale + 35)
                g.FillRectangle(Brushes.Black, x_reference, 0, 150, 17)
                g.DrawString("Heart Rate = " & LapHR(x - 1) & "
beats/min", Font, Brushes.White, x_reference + 5, 1)
                g.FillRectangle(Brushes.Blue, x_reference, 17, 150, 17)
                g.DrawString("Resting HR = " & HRrest(x - 1) & "
beats/min", Font, Brushes.White, x_reference + 5, 18)
                g.FillRectangle(Brushes.Gold, x_reference, 34, 150, 17)
                g.DrawString("Speed = " & Speed & " m/s", Font,
Brushes.Black, x_reference + 5, 35)
                g.DrawString("0", Font, Brushes.White, 68 -
(pcbGraph.Location.X + x_scroll), y_allowance + y_graph - 5)
                Update_Graph()
            End If
NextDay:
        Next x
    End Sub
    Private Sub Update_Fields()
```

```
        Dim FileReader As StreamReader
        FileReader = New StreamReader(ListPath & "\" & cmbName.Text &
".end")
        FileReader.ReadLine()
        txtAge.Text = FileReader.ReadLine()
        Age = txtAge.Text
        FileReader.Close()
        Load_Graph()
        Karvonen()
    End Sub
    Private Sub UpdatePictureBox()
        pcbGraph.Height = y_graph + 125
        Me.Height = pcbGraph.Height + 180
        y_scroll = VerticalScroll.Value
        btnRefresh.Location = New Point(11, pcbGraph.Height + 63 -
y_scroll)
        btnZoomIn.Location = New Point(782, pcbGraph.Height + 63 -
y_scroll)
        btnZoomOut.Location = New Point(830, pcbGraph.Height + 63 -
y_scroll)
        btnBack.Location = New Point(370, pcbGraph.Height + 94 -
y_scroll)
        grpGraph.Location = New Point(12, 9 - y_scroll)
        lblDays.Location = New Point(122, 9 - y_scroll)
        txtDays.Location = New Point(125, 25 - y_scroll)
        lblCurrent.Location = New Point(252, 9 - y_scroll)
        txtCurrent.Location = New Point(255, 25 - y_scroll)
        lblAge.Location = New Point(321, 9 - y_scroll)
        txtAge.Location = New Point(324, 25 - y_scroll)
        lblEvaluation.Location = New Point(363, 9 - y_scroll)
        txtEvaluation.Location = New Point(366, 25 - y_scroll)
        lblName.Location = New Point(481, 9 - y_scroll)
        cmbName.Location = New Point(484, 25 - y_scroll)
        lblTraining.Location = New Point(672, 9 - y_scroll)
        cmbTraining.Location = New Point(675, 25 - y_scroll)
        lblDistance.Location = New Point(743, 9 - y_scroll)
        txtDistance.Location = New Point(746, 25 - y_scroll)
        pcbGraph.Location = New Point(pcbGraph.Location.X + x_scroll,
57 - y_scroll)
        If txtDays.Text <> Nothing Then
            Dim x_coordinate As Integer
            Training_Length = txtDays.Text
            x_coordinate = Graph_Length * Training_Length
            pcbGraph.Width = x_coordinate + x_allowance
            Me.Width = pcbGraph.Width + 40
        End If
    End Sub
    Private Sub AdaptInteger(ByVal ind)
        If LapTime(ind) < 1000 And LapTime(ind) >= 100 Then
            IntAdapt = 56
        ElseIf LapTime(ind) < 100 And LapTime(ind) >= 10 Then
            IntAdapt = 62
        ElseIf LapTime(ind) < 10 And LapTime(ind) >= 0 Then
            IntAdapt = 68
        ElseIf LapTime(ind) < 10000 And LapTime(ind) >= 1000 Then
            IntAdapt = 50
        Else
```

```
            IntAdapt = 44
        End If
    End Sub
    Private Sub Update_Graph()
        Dim pen As New Pen(Color.Red, 2)
        pen.DashStyle = Drawing2D.DashStyle.Dot
        g = pcbGraph.CreateGraphics
        For Me.i = 0 To index - 2
            x_start = (Graph_Length * (x - 1)) + x_allowance + (x_graph
* i)
            y_start = y_allowance + y_graph - (LapTime(i) /graph_scale)
            x_end = (Graph_Length * (x - 1)) + x_allowance + (x_graph *
(i + 1))
            y_end = y_allowance + y_graph - (LapTime(i + 1) /
graph_scale)
            If LapTime(i) Mod 5 = 0 Then
                AdaptInteger(i)
                g.DrawString(LapTime(i), Font, Brushes.White, IntAdapt
- (pcbGraph.Location.X + x_scroll), y_start - 5)
            End If
            If LapTime(i + 1) Mod 5 = 0 Then
                AdaptInteger(i + 1)
                g.DrawString(LapTime(i + 1), Font, Brushes.White,
IntAdapt - (pcbGraph.Location.X + x_scroll), y_end - 5)
            End If
            If rdbLine.Checked = True And rdbBar.Checked = False Then
                g.FillEllipse(Brushes.Red, x_start - 5, y_start - 5,
10, 10)
                g.DrawString(i + 1, Font, Brushes.Red, x_start - 4,
y_start - 15)
                g.FillEllipse(Brushes.Red, x_end - 5, y_end - 5, 10,
10)
                g.DrawString(i + 2, Font, Brushes.Red, x_end - 4, y_end
- 15)
                g.DrawLine(pen, x_start, y_start, x_end, y_end)
            ElseIf rdbBar.Checked = True And rdbLine.Checked = False
Then
                g.FillRectangle(Brushes.Green, x_start, y_start,
(x_graph / 2), (LapTime(i) / graph_scale))
                g.FillRectangle(Brushes.Green, x_end, y_end, (x_graph /
2), (LapTime(i + 1) / graph_scale))
                g.DrawString(i + 1, Font, Brushes.Red, x_start -
(x_graph / 20), y_start - 10)
                g.DrawString(i + 2, Font, Brushes.Red, x_end - (x_graph
/ 20), y_end - 10)
            End If
        Next i
    End Sub
    Private Sub Load_Training_Name()
        Dim FileReader As StreamReader
        FileReader = New StreamReader(ListPath & "\Index.end")
        cmbName.Items.Clear()
        While Not FileReader.EndOfStream
            cmbName.Items.Add(FileReader.ReadLine())
        End While
        FileReader.Close()
        cmbName.Text = cmbName.Items.Item(0)
```

```
    End Sub
    Private Sub Load_Training_Number()
        cmbTraining.Items.Clear()
        For Me.x = 1 To 32767
            If File.Exists(DataPath & "\Day1_" & cmbName.Text & x &
".end") = True Then
                cmbTraining.Items.Add(x.ToString)
            Else
                Exit For
            End If
        Next x
        If cmbTraining.Items.Count <> 0 Then
            cmbTraining.Text = cmbTraining.Items.Item(0)
        End If
    End Sub
    Private Sub Load_Fields()
        If File.Exists(DataPath & "\Initial_" & cmbName.Text &
cmbTraining.Text & ".end") = True Then
            Dim FileReader As StreamReader
            FileReader = New StreamReader(DataPath & "\Initial_" &
cmbName.Text & cmbTraining.Text & ".end")
            FileReader.ReadLine()
            InitialDay = FileReader.ReadLine()
            Training_Length = FileReader.ReadLine()
            Distance = FileReader.ReadLine()
            txtDistance.Text = Distance
            FileReader.Close()
            CurrentDay = Today.DayOfYear - InitialDay
            If CurrentDay <> 1 Then
                txtDays.Enabled = False
            End If
            txtDays.Text = Training_Length
            txtCurrent.Text = CurrentDay
        Else
            MessageBox.Show("The selected client has no training record
yet", "Browse Client Record", 0, _
                            MessageBoxIcon.Information, 0, 0, False)
            pcbGraph.Refresh()
        End If
    End Sub
    Private Sub Karvonen()
        Dim MaxHR, MinHR, TotalHR, index As Single
        index = CurrentDay
        While HRrest(index) = 0
            index -= 1
            RestingHR = HRrest(index)
        End While
        MinHR = (220 - Age - RestingHR) * 0.5 + RestingHR
        MaxHR = (220 - Age - RestingHR) * 0.85 + RestingHR
        i = 0
        While i <= Training_Length
            TotalHR += LapHR(i)
            i += 1
        End While
        AverageHR = TotalHR / i
        If AverageHR > MinHR And AverageHR < MaxHR Then
            txtEvaluation.Text = "Good"
```

```
        Else
            If AverageHR > MaxHR Then
                txtEvaluation.Text = "Bad"
            ElseIf AverageHR < MinHR Then
                txtEvaluation.Text = "Very Good"
            End If
        End If
    End Sub
    Private Sub frmBrowse_VisibleChanged(ByVal sender As System.Object,
ByVal e As EventArgs) Handles _
    MyBase.VisibleChanged
        Load_Training_Name()
        Load_Training_Number()
    End Sub
    Private Sub cmbName_SelectedIndexChanged(ByVal sender As
System.Object, ByVal e As EventArgs) Handles
cmbName.SelectedIndexChanged
        cmbName.Text = cmbName.SelectedItem.ToString
        cmbTraining.Items.Clear()
        For Me.x = 1 To 32767
            If File.Exists(DataPath & "\Day1_" & cmbName.Text & x &
".end") = True Then
                cmbTraining.Items.Add(x)
            Else
                Exit For
            End If
        Next x
        If cmbTraining.Items.Count <> 0 Then
            cmbTraining.Text = cmbTraining.Items.Item(0)
            Load_Fields()
            Update_Fields()
        Else
            MessageBox.Show("The selected Client has no training record
yet", "Browse Client Record", 0, _
                            MessageBoxIcon.Information, 0, 0, False)
            pcbGraph.Refresh()
            cmbTraining.Text = Nothing
        End If
    End Sub
    Private Sub cmbTraining_SelectedIndexChanged(ByVal sender As
System.Object, ByVal e As EventArgs) Handles
cmbTraining.SelectedIndexChanged
        pcbGraph.Location = New Point(0, 57 - y_scroll)
        LargestTime = 0
        If cmbName.Text = Nothing Then
            MessageBox.Show("Select an existing Client in our database
before choosing a corresponding training", _
                            "Browse Training Number", 0,
MessageBoxIcon.Information, 0, 0, False)
        Else
            cmbTraining.Text = cmbTraining.SelectedItem.ToString
            Load_Fields()
            Load_Graph()
            Karvonen()
        End If
        btnRefresh.Select()
    End Sub
```

```
    Private Sub Redraw_Graph()
        If cmbTraining.Items.Count = 0 Then
            pcbGraph.Refresh()
        ElseIf cmbTraining.Items.Count > 0 Then
            Load_Graph()
            Karvonen()
        End If
    End Sub
    Private Sub frmBrowse_move(ByVal sender As Object, ByVal e As
EventArgs) Handles Me.Move
        Redraw_Graph()
    End Sub
    Private Sub frmBrowse_MouseWheel(ByVal sender As Object, ByVal e As
EventArgs) Handles _
    Me.MouseWheel
        Redraw_Graph()
    End Sub
    Private Sub frmBrowse_FormClosing(ByVal sender As Object, ByVal e
As EventArgs) Handles _
    Me.FormClosing
        frmMain.Show()
    End Sub
    Private Sub frmBrowse_scroll(ByVal sender As Object, ByVal e As
EventArgs) Handles _
    Me.Scroll
        Redraw_Graph()
    End Sub
    Private Sub frmBrowse_SizeChanged(ByVal sender As Object, ByVal e
As EventArgs) Handles _
    MyBase.SizeChanged
        If Me.WindowState = FormWindowState.Maximized Then
            Redraw_Graph()
        End If
    End Sub
    Private Sub btnRefresh_Click(ByVal sender As System.Object, ByVal e
As EventArgs) Handles _
    btnRefresh.Click
        Redraw_Graph()
    End Sub
    Private Sub btnMain_Click(ByVal sender As System.Object, ByVal e As
EventArgs)
        Me.Close()
        frmMain.Show()
    End Sub
    Private Sub rdbLine_click(ByVal sender As System.Object, ByVal e As
EventArgs) Handles _
    rdbLine.Click
        Radio_Button()
    End Sub
    Private Sub rdbBar_Click(ByVal sender As System.Object, ByVal e As
EventArgs) Handles _
    rdbBar.Click
        Radio_Button()
    End Sub
    Private Sub Radio_Button()
        If File.Exists(DataPath & "\Day1_" & cmbName.Text &
cmbTraining.Text & ".end") = True Then
```

```
            Redraw_Graph()
        End If
    End Sub
    Private Sub btnBack_Click(ByVal sender As System.Object, ByVal e As
EventArgs) Handles btnBack.Click
        Me.Close()
        frmMain.Show()
    End Sub
    Private Sub frmBrowse_Load(ByVal sender As System.Object, ByVal e
As EventArgs) Handles MyBase.Load
        Graph_Length = 200
        graph_scale = 0.1
        rdbLine.Select()
        btnRefresh.Select()
        VScroll = True
    End Sub
    Private Sub frmEndurance_MouseWheel(ByVal sender As System.Object,
ByVal e As MouseEventArgs) Handles MyBase.MouseWheel
        Graph_Length = Graph_Length + e.Delta / 10
        If Graph_Length < 200 Then
            Graph_Length = 200
        End If
    End Sub
    Private Sub pcbGraph_MouseMove(ByVal sender As System.Object, ByVal
e As MouseEventArgs) Handles pcbGraph.MouseMove
        If e.Button = MouseButtons.Left Then
            x_coord_new = MousePosition.X
            If x_coord_new > x_coord Then
                x_scroll = (x_coord_new - x_coord) / drag_sensitivity
                If pcbGraph.Location.X + x_scroll > 0 Then
                    pcbGraph.Location = New Point(0, 57 - y_scroll)
                    Exit Sub
                End If
                Load_Graph()
            ElseIf x_coord_new < x_coord Then
                x_scroll = (x_coord_new - x_coord) / drag_sensitivity
                If pcbGraph.Width < 1000 Then
                    pcbGraph.Location = New Point(0, 57 - y_scroll)
                    Exit Sub
                End If
                If pcbGraph.Location.X + x_scroll <= -(pcbGraph.Width -
900) Then
                    pcbGraph.Location = New Point(-(pcbGraph.Width -
900), 57 - y_scroll)
                    Exit Sub
                End If
                Load_Graph()
            End If
        End If
    End Sub
    Private Sub pcbGraph_MouseDown(ByVal sender As System.Object, ByVal
e As MouseEventArgs) Handles pcbGraph.MouseDown
        Cursor = Cursors.Hand
        x_coord = MousePosition.X
    End Sub
    Private Sub pcbGraph_MouseUp(ByVal sender As System.Object, ByVal e
As MouseEventArgs) Handles pcbGraph.MouseUp
```

```
        Cursor = Cursors.Default
        x_scroll = 0
        Load_Graph()
    End Sub
    Private Sub btnZoomIn_Click(ByVal sender As System.Object, ByVal e
As EventArgs) Handles btnZoomIn.Click
        Graph_Length += 50
        Load_Graph()
    End Sub
    Private Sub btnZoomOut_Click(ByVal sender As System.Object, ByVal e
As EventArgs) Handles btnZoomOut.Click
        Graph_Length -= 50
        If Graph_Length < 200 Then
            Graph_Length = 200
        End If
        Load_Graph()
    End Sub
End Class
```

Training Form

```
Imports System.IO
Imports System.Drawing
Public Class frmEndurance
    Dim g As Graphics
    'Definition for creating graphics in picture box.
    Dim DataPath As String =
My.Computer.FileSystem.SpecialDirectories.Temp & "\Trainings"
    Dim ListPath As String =
My.Computer.FileSystem.SpecialDirectories.Temp & "\Clients"
    Dim LapTime(100), LapHR(100), HRrest(100) As Single
    'Integer array with one decimal place
    Dim Pos As Integer
    'Determines point of comparison in parsing
    Dim i, x, train_no As Integer
    'Declaration of subsitute variables
    Dim IntAdapt As Integer
    'Variable to align integers to the right side
    Dim x_scroll, y_scroll As Integer
    'Form variables for scroll bar movement
    Dim x_coord, x_coord_new As Integer
    Dim x_graph, y_graph, x_end, y_end, x_start, y_start As Single
    'Coordinates for plotting points
    Dim index, LargestTime, HeartRate, InitialDay, CurrentDay,
Training_Length As Single
    'Data-mapping variables
    Dim RestingHR, Age, AverageHR, Speed, Distance, TotalTime As Single
    'Heart Rate Computation Variables
    Dim FileSelected As Boolean = False
    'Flag for determining when file is selected or not (0 if no and 1
if yes)
    Dim ofdGraph As New OpenFileDialog
    'Declaration for the dialog box to open files.
    Dim results As DialogResult
    'Variable to receive the response in the dialog box
    Dim response As MsgBoxResult
```

```
    'Variable to recieve the response in the message box
    Dim Graph_scale As Single
    Dim Graph_Length As Single
    Const x_allowance = 100, y_allowance = 75, drag_sensitivity = 50
    'Scales for graphing
    Private Sub btnOpen_Click(ByVal sender As System.Object, ByVal e As
EventArgs) Handles _
    btnOpen.Click
        If txtAge.Text = Nothing Or txtName.Text = Nothing Then
            MessageBox.Show("Add or Select a Client in 'Client Info'
before selecting a file", _
                            "Browsing File", 0,
MessageBoxIcon.Information, 0, 0, False)
            Me.Close()
            frmMain.Show()
        End If
        Check_Default()
        'Checks options if it has yet to be chosen
        results = ofdGraph.ShowDialog
        'Stores the result of the open file dialog box
        If results = DialogResult.OK Then
            'If activated option is "OK"
            If IO.File.Exists(DataPath & "\Day" & CurrentDay & "_" &
txtName.Text & train_no & ".end") = True Then
                'Checks if the Current Training Day already has an
existing graph copied to a file
                response = MessageBox.Show("You are about to replace
the existing Graph of this day. Continue?", _
                                           "Browsing file",
MessageBoxButtons.OKCancel, MessageBoxIcon.Warning, 0, 0, False)
                'Shows the message box for the replacement scenario
                If response = MsgBoxResult.Ok Then
                    'If "OK" button is selected
                    CopyFile()
                    'Replace the existing with the one inputted on the
form
                    SaveData()
                    'Updates the reference for the length and first day
of training
                    Load_Graph()
                    'Draw the a graph from the current day down to the
first day
                Else
                    Load_Graph()
                End If
            ElseIf CurrentDay > Training_Length And txtDays.Text <>
Nothing Then
                'When the current day is out of range of the recent
training coverage
                response = MessageBox.Show("Today has exceeded the
training length. Do you want to start a " & _
                                           "new training with the
selected file as your initial day?", "Browsing File", _
                                           MessageBoxButtons.YesNo,
MessageBoxIcon.Question, 0, 0, False)
                'Show the message box to prompt for a new training or
browse
```

```
                If response = MsgBoxResult.Yes Then
                    'If "YES" option is selected
                    CurrentDay = 1
                    'New training means the current day is the first
day
                    txtCurrent.Text = CurrentDay
                    txtDays.Text = 1
                    txtDays.Enabled = True
                    'enables control of the textbox to further allow a
new entry
                    For Me.x = 1 To 32767
                        If File.Exists(DataPath & "\Initial_" &
txtName.Text & x & ".end") = False Then
                            train_no = x
                            Exit For
                        End If
                    Next x
                    CopyFile()
                    SaveData()
                    Load_Graph()
                ElseIf response = MsgBoxResult.Cancel Then
                    pcbGraph.Refresh()
                Else
                    Load_Graph()
                    'redraw all the recent training graphs for browsing
                End If
            ElseIf IO.File.Exists(DataPath & "\Day" & CurrentDay & "_"
& txtName.Text & train_no & ".end") = False Then
                'When current day has no existing graph
                CopyFile()
                'Copy graph to a new file
                SaveData()
                Load_Graph()
                'Draw the current graph up to the first day
            Else
                Load_Graph()
            End If
        End If
    End Sub
    Private Sub Clear_All()
        txtDays.Text = Nothing
        txtAge.Text = Nothing
        txtRest.Text = Nothing
        txtEvaluation.Text = Nothing
        txtName.Text = Nothing
    End Sub
    Private Sub Check_Default()
        If txtDays.Text <> Nothing Then
            IntAdapt = txtDays.Text
        End If
        If txtDays.Text = Nothing Or IntAdapt < 12 And txtDays.Enabled
= True Then
            'If training length field is blank
            MessageBox.Show("The minimum number of training days is
12", "Checking program defaults", _
                            0, MessageBoxIcon.Stop, 0, 0, False)
            txtDays.Text = "12"
```

```
        End If
    End Sub
    Private Sub Invalid_Value(ByVal sender)
        If Not IsNumeric(sender.Text) And sender.Text <> Nothing Then
            'If input is not a number and is not empty
            MessageBox.Show("Only numerical inputs are required",
"Invalid Field Value", 0, _
                            MessageBoxIcon.Warning, 0, 0, False)
            'Notify Client of the invalid input
            sender.Text = Nothing
            'Clears the corresponding field
        End If
    End Sub
    Private Sub SaveData()
        If txtDays.Text <> Nothing Or txtRest.Text <> Nothing Then
            If File.Exists(DataPath & "\Day" & CurrentDay & "_" &
txtName.Text & train_no & ".end") = False Then
                MessageBox.Show("Current Day has no graph to record
data", "Saving Training", 0, MessageBoxIcon.Information, _
                                0, 0, False)
                Exit Sub
            End If
            Dim FileWriter As StreamWriter
            InitialDay = Today.DayOfYear - CurrentDay
            Training_Length = txtDays.Text
            FileWriter = New StreamWriter(DataPath & "\Initial_" &
txtName.Text & train_no & ".end")
            FileWriter.WriteLine(txtName.Text)
            FileWriter.WriteLine(InitialDay)
            FileWriter.WriteLine(Training_Length)
            FileWriter.WriteLine(Distance)
            FileWriter.Close()
            Load_Graph()
            Dim RestTemp, MainStr As String
            If txtRest.Text = Nothing Then
                Exit Sub
            End If
            Dim CharTemp(255) As Char
            Dim FileReader As StreamReader
            FileReader = New StreamReader(DataPath & "\Day" &
CurrentDay & "_" & txtName.Text & train_no & ".end")
            MainStr = FileReader.ReadToEnd()
            FileReader.Close()
            Pos = InStr(MainStr, "R", CompareMethod.Text)
            If Pos <> 0 Then
                FileReader = New StreamReader(DataPath & "\Day" &
CurrentDay & "_" & txtName.Text & train_no & ".end")
                FileReader.Read(CharTemp, 0, Pos - 1)
                FileReader.Close()
                FileWriter = New StreamWriter(DataPath & "\Day" &
CurrentDay & "_" & txtName.Text & train_no & ".end")
                FileWriter.Write(CharTemp, 0, Pos - 1)
                FileWriter.Close()
            End If
            If txtRest.TextLength = 2 Then
                RestTemp = "0" & txtRest.Text
            Else
```

```
                RestTemp = txtRest.Text
            End If
            FileWriter = New StreamWriter(DataPath & "\Day" &
CurrentDay & "_" & txtName.Text & train_no & ".end", True)
            FileWriter.Write("RestHR:" & RestTemp)
            FileWriter.Close()
        End If
    End Sub
    Private Sub CopyFile()
        Dim File As New FileInfo(ofdGraph.FileName)
        File.CopyTo(DataPath & "\Day" & CurrentDay & "_" & txtName.Text
& train_no & ".end", True)
        File.Delete()
    End Sub
    Private Sub Load_Graph()
        pcbGraph.Refresh()
        g = pcbGraph.CreateGraphics
        g.FillRectangle(Brushes.Black, -(pcbGraph.Location.X +
x_scroll), 0, x_allowance, LargestTime / Graph_scale + y_allowance +
500)
        g.FillRectangle(Brushes.Black, x_allowance, LargestTime /
Graph_scale + y_allowance, Graph_Length * Training_Length, 500)
        pcbGraph.BackColor = Color.White
        For Me.x = CurrentDay To 1 Step -1
            If File.Exists(DataPath & "\Day" & x & "_" & txtName.Text &
train_no & ".end") = True Then
                UpdatePictureBox()
                Dim FileReader As StreamReader
                Dim MainStr, StrTemp As String
                FileReader = New StreamReader(DataPath & "\Day" & x &
"_" & txtName.Text & train_no & ".end")
                MainStr = FileReader.ReadToEnd()
                FileReader.Close()
                index = 0
                Pos = 2
                While Pos <> 0
Search:
                    Pos = InStr(MainStr, "L", CompareMethod.Text)
                    If Pos = 0 Then
                        Pos = InStr(MainStr, "H", CompareMethod.Text)
                        If Pos = 0 Then
Invalid:
                            MessageBox.Show("Invalid File Format",
"Deleting Training", 0, _
                                            MessageBoxIcon.Exclamation,
0, 0, False)
                            Dim DayHandler As New FileInfo(DataPath &
"\Day" & x & "_" & _
                                                           txtName.Text
& train_no & ".end")
                            DayHandler.Delete()
                            GoTo NextDay
                        End If
                        Pos = InStr(MainStr, ":", CompareMethod.Text)
                        StrTemp = Mid(MainStr, Pos, 6)
                        StrTemp = StrTemp.Trim(":")
                        If IsNumeric(StrTemp) Then
```

```
                            LapHR(x - 1) = StrTemp
                        Else
                            GoTo Invalid
                        End If
                        Pos = InStr(MainStr, "R", CompareMethod.Text)
                        If Pos <> 0 Then
                            MainStr = Mid(MainStr, Pos + 1)
                            Pos = InStr(MainStr, ":",
CompareMethod.Text)
                            StrTemp = Mid(MainStr, Pos, 4)
                            StrTemp = StrTemp.Trim(":")
                            If IsNumeric(StrTemp) Then
                                HRrest(x - 1) = StrTemp
                            End If
                        End If
                        FileSelected = True
                        Exit While
                    End If
                    Pos = InStr(MainStr, ":", CompareMethod.Text)
                    StrTemp = Mid(MainStr, Pos, 6)
                    StrTemp = StrTemp.Trim(":")
                    IntAdapt = StrTemp
                    If IntAdapt > 0 Then
                        LapTime(index) = StrTemp
                    ElseIf IntAdapt = 0 Then
                        MainStr = Mid(MainStr, Pos + 1)
                        GoTo Search
                    ElseIf Not IsNumeric(StrTemp) Then
                        GoTo Invalid
                    End If
                    MainStr = Mid(MainStr, Pos + 1)
                    index += 1
                End While
                TotalTime = 0
                For Me.i = 0 To index - 1
                    TotalTime += LapTime(i)
                    If LapTime(i) > LargestTime Then
                        LargestTime = LapTime(i)
                    End If
                Next i
                If LargestTime > 5000 Then
                    Graph_scale = 10
                ElseIf LargestTime <= 5000 And LargestTime > 1000 Then
                    Graph_scale = 5
                ElseIf LargestTime <= 1000 And LargestTime > 500 Then
                    Graph_scale = 2
                ElseIf LargestTime <= 500 And LargestTime > 100 Then
                    Graph_scale = 1
                ElseIf LargestTime <= 100 And LargestTime > 50 Then
                    Graph_scale = 0.5
                ElseIf LargestTime <= 50 And LargestTime >= 0 Then
                    Graph_scale = 0.1
                End If
                If txtDistance.Text = Nothing Then
                    Distance = 0
                Else
                    Distance = txtDistance.Text
```

```
                End If
                Speed = Distance / (TotalTime / index)
                Speed = Math.Round(Speed, 2)
                Dim x_reference As Single
                x_reference = (Graph_Length * (x - 1)) + x_allowance
                x_graph = Graph_Length / (index)
                y_graph = LargestTime / Graph_scale
                g.DrawString("Time", Font, Brushes.White, 13 -
(pcbGraph.Location.X + x_scroll), pcbGraph.Height / 2 - y_allowance)
                g.DrawString("(Seconds)", Font, Brushes.White, -
(pcbGraph.Location.X + x_scroll), pcbGraph.Height / 2 - 10)
                g.DrawString("Laps", Font, Brushes.White, x_reference +
                             Graph_Length / 2 - 35, y_allowance +
LargestTime / Graph_scale + 5)
                g.FillRectangle(Brushes.White, x_reference +
Graph_Length / 2 - 45, y_allowance + LargestTime _
                               / Graph_scale + 30, 50, 50)
                g.DrawString("Day " & x, Font, Brushes.Black,
x_reference + Graph_Length / 2 _
                             - 40, y_allowance + LargestTime /
Graph_scale + 35)
                g.FillRectangle(Brushes.Black, x_reference, 0, 150, 17)
                g.DrawString("Heart Rate = " & LapHR(x - 1) & "
beats/min", Font, Brushes.White, x_reference + 5, 1)
                g.FillRectangle(Brushes.Blue, x_reference, 17, 150, 17)
                g.DrawString("Resting HR = " & HRrest(x - 1) & "
beats/min", Font, Brushes.White, x_reference + 5, 18)
                g.FillRectangle(Brushes.Gold, x_reference, 34, 150, 17)
                g.DrawString("Speed = " & Speed & " m/s", Font,
Brushes.Black, x_reference + 5, 35)
                g.DrawString("0", Font, Brushes.White, 68 -
(pcbGraph.Location.X + x_scroll), y_allowance + y_graph - 5)
                Update_Graph()
            End If
NextDay:
        Next x
    End Sub
    Private Sub UpdatePictureBox()
        pcbGraph.Height = y_graph + 125
        Me.Height = pcbGraph.Height + 205
        y_scroll = VerticalScroll.Value
        btnOpen.Location = New Point(12, pcbGraph.Height + 77 -
y_scroll)
        btnRefresh.Location = New Point(108, pcbGraph.Height + 77 -
y_scroll)
        btnEvaluate.Location = New Point(602, 36 - y_scroll)
        btnBack.Location = New Point(426, pcbGraph.Height + 101 -
y_scroll)
        btnZoomIn.Location = New Point(852, pcbGraph.Height + 77 -
y_scroll)
        btnZoomOut.Location = New Point(900, pcbGraph.Height + 77 -
y_scroll)
        grpGraph.Location = New Point(13, 20 - y_scroll)
        lblDays.Location = New Point(123, 20 - y_scroll)
        txtDays.Location = New Point(127, 37 - y_scroll)
        lblCurrent.Location = New Point(253, 20 - y_scroll)
        txtCurrent.Location = New Point(256, 37 - y_scroll)
```

```
        lblRest.Location = New Point(322, 20 - y_scroll)
        txtRest.Location = New Point(325, 37 - y_scroll)
        lblAge.Location = New Point(482, 20 - y_scroll)
        txtAge.Location = New Point(482, 37 - y_scroll)
        lblEvaluation.Location = New Point(517, 20 - y_scroll)
        txtEvaluation.Location = New Point(520, 37 - y_scroll)
        lblName.Location = New Point(635, 20 - y_scroll)
        txtName.Location = New Point(635, 37 - y_scroll)
        lblDistance.Location = New Point(825, 20 - y_scroll)
        txtDistance.Location = New Point(828, 37 - y_scroll)
        pcbGraph.Location = New Point(pcbGraph.Location.X + x_scroll,
71 - y_scroll)
        If txtDays.Text <> Nothing Then
            Dim x_coordinate As Integer
            Training_Length = txtDays.Text
            x_coordinate = Graph_Length * Training_Length
            pcbGraph.Width = x_coordinate + x_allowance
            Me.Width = pcbGraph.Width + 128
        End If
    End Sub
    Private Sub AdaptInteger(ByVal ind)
        If LapTime(ind) < 10000 And LapTime(ind) >= 1000 Then
            IntAdapt = 50
        ElseIf LapTime(ind) < 1000 And LapTime(ind) >= 100 Then
            IntAdapt = 56
        ElseIf LapTime(ind) < 100 And LapTime(ind) >= 10 Then
            IntAdapt = 62
        ElseIf LapTime(ind) < 10 And LapTime(ind) >= 0 Then
            IntAdapt = 68
        Else
            IntAdapt = 44
        End If
    End Sub
    Private Sub Update_Graph()
        If FileSelected = True Then
            Dim pen As New Pen(Color.Red, 2)
            pen.DashStyle = Drawing2D.DashStyle.Dot
            g = pcbGraph.CreateGraphics
            For Me.i = 0 To index - 2
                x_start = (Graph_Length * (x - 1)) + x_allowance +
(x_graph * i)
                y_start = y_allowance + y_graph - (LapTime(i) /
Graph_scale)
                x_end = (Graph_Length * (x - 1)) + x_allowance +
(x_graph * (i + 1))
                y_end = y_allowance + y_graph - (LapTime(i + 1) /
Graph_scale)
                If LapTime(i) Mod 5 = 0 Then
                    AdaptInteger(i)
                    g.DrawString(LapTime(i), Font, Brushes.White,
IntAdapt - (pcbGraph.Location.X + x_scroll), y_start - 5)
                End If
                If LapTime(i + 1) Mod 5 = 0 Then
                    AdaptInteger(i + 1)
                    g.DrawString(LapTime(i + 1), Font, Brushes.White,
IntAdapt - (pcbGraph.Location.X + x_scroll), y_end - 5)
                End If
```

```
                If rdbLine.Checked = True And rdbBar.Checked = False
Then
                    g.FillEllipse(Brushes.Red, x_start - 5, y_start -
5, 10, 10)
                    g.DrawString(i + 1, Font, Brushes.Red, x_start - 4,
y_start - 15)
                    g.FillEllipse(Brushes.Red, x_end - 5, y_end - 5,
10, 10)
                    g.DrawString(i + 2, Font, Brushes.Red, x_end - 4,
y_end - 15)
                    g.DrawLine(pen, x_start, y_start, x_end, y_end)
                ElseIf rdbBar.Checked = True And rdbLine.Checked =
False Then
                    g.FillRectangle(Brushes.Green, x_start, y_start,
(x_graph / 2), (LapTime(i) / Graph_scale))
                    g.FillRectangle(Brushes.Green, x_end, y_end,
(x_graph / 2), (LapTime(i + 1) / Graph_scale))
                    g.DrawString(i + 1, Font, Brushes.Red, x_start -
(x_graph / 20), y_start - 10)
                    g.DrawString(i + 2, Font, Brushes.Red, x_end -
(x_graph / 20), y_end - 10)
                End If
            Next i
        End If
    End Sub
    Private Sub Karvonen()
        Dim MaxHR, MinHR, TotalHR, index As Single
        index = CurrentDay
        While HRrest(index) = 0
            index -= 1
            RestingHR = HRrest(index)
        End While
        MinHR = (220 - Age - RestingHR) * 0.5 + RestingHR
        MaxHR = (220 - Age - RestingHR) * 0.85 + RestingHR
        i = 0
        While LapHR(i) <> 0
            TotalHR += LapHR(i)
            i += 1
        End While
        AverageHR = TotalHR / i
        If AverageHR > MinHR And AverageHR < MaxHR Then
            txtEvaluation.Text = "Good"
        Else
            If AverageHR > MaxHR Then
                txtEvaluation.Text = "Bad"
            ElseIf AverageHR < MinHR Then
                txtEvaluation.Text = "Very Good"
            End If
        End If
    End Sub
    Private Sub rdbLine_click(ByVal sender As System.Object, ByVal e As
EventArgs) Handles _
    rdbLine.Click
        Radio_Button()
    End Sub
    Private Sub rdbBar_Click(ByVal sender As System.Object, ByVal e As
EventArgs) Handles _
```

```
    rdbBar.Click
        Radio_Button()
    End Sub
    Private Sub Initialize_Data()
        If Directory.Exists(DataPath) = False Then
            Dim DirHandler As New DirectoryInfo(DataPath)
            DirHandler.Create()
        End If
        Dim FileReader As StreamReader
        If File.Exists(ListPath & "\Current.end") = True Then
            FileReader = New StreamReader(ListPath & "\Current.end")
            txtName.Text = FileReader.ReadLine()
            txtAge.Text = FileReader.ReadLine()
            FileReader.Close()
        End If
        For Me.x = 1 To 32767
            If File.Exists(DataPath & "\Initial_" & txtName.Text & x &
".end") = False Then
                If x > 1 Then
                    train_no = x - 1
                ElseIf x = 1 Then
                    train_no = x
                End If
                Exit For
            End If
        Next x
        If File.Exists(DataPath & "\Initial_" & txtName.Text & train_no
& ".end") = True Then
            FileReader = New StreamReader(DataPath & "\Initial_" &
txtName.Text & train_no & ".end")
            FileReader.ReadLine()
            InitialDay = FileReader.ReadLine()
            Training_Length = FileReader.ReadLine()
            Distance = FileReader.ReadLine()
            FileReader.Close()
            CurrentDay = Today.DayOfYear - InitialDay
            If CurrentDay <> 1 Then
                txtDays.Enabled = False
            End If
            txtDays.Text = Training_Length
            txtCurrent.Text = CurrentDay
            txtDistance.Text = Distance
        Else
            CurrentDay = 1
            txtCurrent.Text = CurrentDay
        End If
        If IO.File.Exists(DataPath & "\Day" & CurrentDay & "_" &
txtName.Text & train_no & ".end") = True Then
            FileSelected = True
        End If
    End Sub
    Private Sub Radio_Button()
        If File.Exists(DataPath & "\Day1_" & txtName.Text & train_no &
".end") = True Then
            Load_Graph()
        Else
```

```
            MessageBox.Show("You currently have no data to graph",
"Loading Graph", 0, _
                                MessageBoxIcon.Stop, 0, 0, False)
            pcbGraph.Refresh()
        End If
    End Sub
    Private Sub frmEndurance_VisibleChanged(ByVal sender As
System.Object, ByVal e As EventArgs) Handles Me.VisibleChanged
        Initialize_Data()
        rdbLine.Select()
        Redraw_Graph()
    End Sub
    Private Sub frmEndurance_FormClosing(ByVal sender As Object, _
        ByVal e As FormClosingEventArgs) Handles Me.FormClosing
        If txtRest.Text <> Nothing Then
            RestingHR = txtRest.Text
        Else
            GoTo Skip
        End If
        If RestingHR < 50 Or RestingHR > 100 Then
            MessageBox.Show("Invalid Resting Heart Rate (must be within
50 and 100 beats/min)", "Saving Training", _
                            0, MessageBoxIcon.Information, 0, 0, False)
            e.Cancel = True
            Exit Sub
        End If
Skip:
        response = MessageBox.Show("Do you want to save the current
training before closing?", "Closing Training Form", _
                                   MessageBoxButtons.YesNoCancel,
MessageBoxIcon.Question, 0, 0, False)
        If response = MsgBoxResult.Yes Then
            SaveData()
            frmMain.Show()
        ElseIf response = MsgBoxResult.No Then
            frmMain.Show()
        ElseIf response = MsgBoxResult.Cancel Then
            e.Cancel = True
        End If
    End Sub
    Private Sub Redraw_Graph()
        If File.Exists(DataPath & "\Day1_" & txtName.Text & train_no &
".end") = True Then
            Load_Graph()
        Else
            pcbGraph.Refresh()
        End If
    End Sub
    Private Sub frmEndurance_move(ByVal sender As Object, ByVal e As
EventArgs) Handles Me.Move
        Redraw_Graph()
    End Sub
    Private Sub frmEndurance_MouseWheel(ByVal sender As Object, ByVal e
As EventArgs) Handles _
    Me.MouseWheel
        Redraw_Graph()
    End Sub
```

```
    Private Sub frmEndurance_scroll(ByVal sender As Object, ByVal e As
EventArgs) Handles _
    Me.Scroll
        Redraw_Graph()
    End Sub
    Private Sub frmEndurance_SizeChanged(ByVal sender As Object, ByVal
e As EventArgs) Handles _
    MyBase.SizeChanged
        If Me.WindowState = FormWindowState.Maximized Then
            Redraw_Graph()
        End If
    End Sub
    Private Sub btnRefresh_Click(ByVal sender As System.Object, ByVal e
As EventArgs) Handles _
    btnRefresh.Click
        Check_Default()
        If File.Exists(DataPath & "\Day1_" & txtName.Text & train_no &
".end") = True Then
            Load_Graph()
        Else
            MessageBox.Show("You do not have anything to reload",
"Reloading Graph", 0, _
                            MessageBoxIcon.Stop, 0, 0, False)
            pcbGraph.Refresh()
        End If
    End Sub
    Private Sub btnEvaluate_Click(ByVal sender As System.Object, ByVal
e As EventArgs) Handles _
    btnEvaluate.Click
        If File.Exists(DataPath & "\Day1_" & txtName.Text & train_no &
".end") = False Then
            MessageBox.Show("No graph to get data from", "Evaluating
Endurance", 0, MessageBoxIcon. _
                            Exclamation, 0, 0, False)
        Else
            Karvonen()
        End If
    End Sub
    Private Sub txtDays_TextChanged(ByVal sender As System.Object,
ByVal e As EventArgs) Handles _
    txtDays.TextChanged
        Invalid_Value(sender)
    End Sub
    Private Sub TextBox1_TextChanged(ByVal sender As System.Object,
ByVal e As EventArgs) Handles txtDistance.TextChanged
        Invalid_Value(sender)
        Load_Graph()
    End Sub
    Private Sub btnBack_Click(ByVal sender As System.Object, ByVal e As
EventArgs) Handles btnBack.Click
        Me.Close()
    End Sub
    Private Sub frmEndurance_Load(ByVal sender As System.Object, ByVal
e As EventArgs) Handles MyBase.Load
        Graph_Length = 200
        Graph_scale = 0.1
        btnRefresh.Select()
```

```
        VScroll = True
    End Sub
    Private Sub frmEndurance_MouseWheel(ByVal sender As System.Object,
ByVal e As MouseEventArgs) Handles MyBase.MouseWheel
        Graph_Length = Graph_Length + e.Delta / 10
        If Graph_Length < 200 Then
            Graph_Length = 200
        End If
    End Sub
    Private Sub txtRest_TextChanged(ByVal sender As System.Object,
ByVal e As EventArgs) Handles txtRest.TextChanged
        Invalid_Value(sender)
    End Sub
    Private Sub pcbGraph_MouseMove(ByVal sender As System.Object, ByVal
e As MouseEventArgs) Handles pcbGraph.MouseMove
        If e.Button = MouseButtons.Left Then
            x_coord_new = MousePosition.X
            If x_coord_new > x_coord Then
                x_scroll = (x_coord_new - x_coord) / drag_sensitivity
                If pcbGraph.Location.X + x_scroll >= 0 Then
                    pcbGraph.Location = New Point(0, 71 - y_scroll)
                    Exit Sub
                End If
                Load_Graph()
            ElseIf x_coord_new < x_coord Then
                x_scroll = (x_coord_new - x_coord) / drag_sensitivity
                If pcbGraph.Location.X + x_scroll <= -(pcbGraph.Width -
900) Then
                    pcbGraph.Location = New Point(-(pcbGraph.Width -
900), 71 - y_scroll)
                    Exit Sub
                End If
                Load_Graph()
            End If
        End If
    End Sub
    Private Sub pcbGraph_MouseDown(ByVal sender As System.Object, ByVal
e As MouseEventArgs) Handles pcbGraph.MouseDown
        Cursor = Cursors.Hand
        x_coord = MousePosition.X
    End Sub
    Private Sub pcbGraph_MouseUp(ByVal sender As System.Object, ByVal e
As MouseEventArgs) Handles pcbGraph.MouseUp
        Cursor = Cursors.Default
        x_scroll = 0
        Load_Graph()
    End Sub
    Private Sub btnZoomIn_Click(ByVal sender As System.Object, ByVal e
As EventArgs) Handles btnZoomIn.Click
        Graph_Length += 50
        Load_Graph()
    End Sub
    Private Sub btnZoomOut_Click(ByVal sender As System.Object, ByVal e
As EventArgs) Handles btnZoomOut.Click
        Graph_Length -= 50
        If Graph_Length < 200 Then
            Graph_Length = 200
```

```
        End If
        Load_Graph()
    End Sub
End Class
```

Main Form

```
Imports System.IO
Public Class frmMain
    Dim ListPath As String =
My.Computer.FileSystem.SpecialDirectories.Temp & "\Clients"
    Private Sub btnTraining_Click(ByVal sender As System.Object, ByVal
e As EventArgs) Handles btnTraining.Click
        If File.Exists(ListPath & "\Current.end") = True Then
            Me.Hide()
            frmEndurance.Show()
        Else
            MessageBox.Show("Add and Select a Client from 'Clients
Form' before entering 'Trainings Form'",
                "Opening Training Info", 0, MessageBoxIcon.Stop, 0, 0,
False)
        End If
    End Sub
    Private Sub btnBrowse_Click(ByVal sender As System.Object, ByVal e
As EventArgs) Handles btnBrowse.Click
        If File.Exists(ListPath & "\Current.end") = True Then
            Me.Hide()
            frmBrowse.Show()
        Else
            MessageBox.Show("Add and Select a Client from 'Clients
Form' before entering 'History Form'",
              "Opening Browse Trainings", 0, MessageBoxIcon.Stop, 0, 0,
False)
        End If
    End Sub
    Private Sub btnInfo_Click(ByVal sender As System.Object, ByVal e As
EventArgs) Handles btnInfo.Click
        Me.Hide()
        frmInfo.Show()
    End Sub
    Private Sub btnSEAMs_Click(ByVal sender As System.Object, ByVal e
As EventArgs) Handles btnSEAMs.Click
        Me.Hide()
        frmSEAMs.Show()
    End Sub
    Private Sub btnHelp_Click(ByVal sender As System.Object, ByVal e As
EventArgs) Handles btnHelp.Click
        Me.Hide()
        frmHelp.Show()
    End Sub
    Private Sub frmMain_Load(ByVal sender As System.Object, ByVal e As
EventArgs) Handles MyBase.Load
    End Sub
End Class
```

Client Form

```
Imports System.IO
Public Class frmInfo
    Dim ListPath As String =
My.Computer.FileSystem.SpecialDirectories.Temp & "\Clients"
    Dim DataPath As String =
My.Computer.FileSystem.SpecialDirectories.Temp & "\Trainings"
    Dim gender, Client_name, temp As String
    Dim rest As Integer
    Private Sub Invalid_Value(ByVal sender)
        If Not IsNumeric(sender.Text) And sender.Text <> Nothing Then
            'If input is not a number and is not empty
            MessageBox.Show("Invalid Field Value", "Inputting on a
field", 0, _
                            MessageBoxIcon.Warning, 0, 0, False)
            'Notify user of the invalid input
            sender.Text = Nothing
            'Clears the corresponding field
        End If
    End Sub
    Private Sub Check_Name(ByVal cmbName, ByVal txtName)
        If chkSelect.Checked = True Then
            Client_name = cmbName
        ElseIf chkSelect.Checked = False Then
            Client_name = txtName
        End If
    End Sub
    Private Sub List_Clients()
        Dim FileReader As StreamReader
        FileReader = New StreamReader(ListPath & "\Index.end")
        While Not FileReader.EndOfStream
            cmbName.Items.Add(FileReader.ReadLine())
        End While
        FileReader.Close()
    End Sub
    Private Sub Clear_Fields()
        cmbName.Items.Clear()
        cmbName.Text = Nothing
        lblCheck.Text = Nothing
        txtAge.Text = Nothing
        txtName.Text = Nothing
        txtContact.Text = Nothing
        rdbMale.Checked = False
        rdbFemale.Checked = False
    End Sub
    Private Sub btnAdd_Click(ByVal sender As System.Object, ByVal e As
EventArgs) Handles btnAdd.Click
        Check_Name(cmbName.Text, txtName.Text)
        If Client_name = Nothing Or txtAge.Text = Nothing Or
(rdbMale.Checked = False And _
         rdbFemale.Checked = False) Then
            MessageBox.Show("Do not leave required fields (*) blank",
"Client's Info", 0, _
                            MessageBoxIcon.Exclamation, 0, 0, False)
            Exit Sub
```

```
        End If
        If txtContact.TextLength <> 11 And txtContact.TextLength <> 7
And txtContact.TextLength <> 0 Then
            MessageBox.Show("Please follow the format of the contact
number as it will be used for verifications or inquiries", _
                            "Client's Info", 0,
MessageBoxIcon.Information, 0, 0, False)
            txtContact.Text = "09273233227"
            Exit Sub
        End If
        If Client_name.ToString.Length < 9 Or InStr(Client_name, ",",
CompareMethod.Text) = 0 Then
            MessageBox.Show("Please follow the given format for your
name and make sure it's valid", "Client's Info", _
                            0, MessageBoxIcon.Information, 0, 0, False)
            txtName.Text = Nothing
            cmbName.Text = Nothing
            Exit Sub
        End If
        If rdbMale.Checked = True Then
            gender = "Male"
        ElseIf rdbFemale.Checked = True Then
            gender = "Female"
        End If
        Dim FileWriter As StreamWriter
        Dim FileReader As StreamReader
        Dim MainStr As String
        If File.Exists(ListPath & "\" & Client_name & ".end") = True
Then
            MessageBox.Show("Client's Info Updated", "Updating Client",
0, _
                            MessageBoxIcon.Information, 0, 0, False)
        Else
            MessageBox.Show("Client added to the list", "Adding
Client", 0, _
                            MessageBoxIcon.Information, 0, 0, False)
        End If
        Try
            FileWriter = New StreamWriter(ListPath & "\" & Client_name
& ".end", False)
            FileWriter.Close()
        Catch ex As Exception
            MessageBox.Show(ex.Message & " Please enter a valid name",
"Writing to File", 0, MessageBoxIcon.Exclamation, _
                            0, 0, False)
            txtName.Text = Nothing
            cmbName.Text = Nothing
            Exit Sub
        End Try
        FileWriter = New StreamWriter(ListPath & "\" & Client_name &
".end", False)
        FileWriter.WriteLine(Client_name)
        FileWriter.WriteLine(txtAge.Text)
        If txtContact.Text = Nothing Then
            FileWriter.WriteLine("n/a")
        Else
            FileWriter.WriteLine(txtContact.Text)
```

```
        End If
        FileWriter.WriteLine(gender)
        FileWriter.Close()
        FileReader = New StreamReader(ListPath & "\Index.end")
        MainStr = FileReader.ReadToEnd()
        FileReader.Close()
        FileWriter = New StreamWriter(ListPath & "\Index.end", True)
        If InStr(MainStr, Client_name, CompareMethod.Text) = 0 Then
            FileWriter.WriteLine(Client_name)
        End If
        FileWriter.Close()
    End Sub
    Private Sub btnClear_Click(ByVal sender As System.Object, ByVal e
As EventArgs) Handles btnClear.Click
        Clear_Fields()
        List_Clients()
    End Sub
    Private Sub txtAge_TextChanged(ByVal sender As System.Object, ByVal
e As EventArgs) Handles txtAge.TextChanged
        Invalid_Value(sender)
    End Sub
    Private Sub txtRest_TextChanged(ByVal sender As System.Object,
ByVal e As EventArgs)
        Invalid_Value(sender)
    End Sub
    Private Sub txtContact_TextChanged(ByVal sender As System.Object,
ByVal e As EventArgs) Handles txtContact.TextChanged
        Invalid_Value(sender)
    End Sub
    Private Sub frmInfo_FormClosing(ByVal sender As System.Object,
ByVal e As EventArgs) Handles MyBase.FormClosed
        Clear_Fields()
        frmMain.Show()
    End Sub
    Private Sub frmInfo_Load(ByVal sender As System.Object, ByVal e As
EventArgs) Handles MyBase.Load
        Clear_Fields()
        If Directory.Exists(ListPath) = False Then
            Dim DirHandler As New DirectoryInfo(ListPath)
            DirHandler.Create()
        End If
        If File.Exists(ListPath & "\Index.end") = False Then
            Dim FileCreator As New FileInfo(ListPath & "\Index.end")
            FileCreator.Create()
        End If
    End Sub
    Private Sub btnSelect_Click(ByVal sender As System.Object, ByVal e
As EventArgs) Handles btnSelect.Click
        Check_Name(cmbName.Text, txtName.Text)
        If Client_name = Nothing Or txtAge.Text = Nothing Or
(rdbMale.Checked = False And _
        rdbFemale.Checked = False) Then
            MessageBox.Show("Do not leave required fields (*) blank",
"Client's Info", 0, _
                            MessageBoxIcon.Exclamation, 0, 0, False)
            Exit Sub
        End If
```

```
        If txtContact.TextLength <> 11 And txtContact.TextLength <> 7
And txtContact.TextLength <> 0 Then
            MessageBox.Show("Please follow the format of the contact
number as it will be used for verifications or inquiries", _
                            "Client's Info", 0,
MessageBoxIcon.Information, 0, 0, False)
            txtContact.Text = "09273233227"
            Exit Sub
        End If
        If Client_name.ToString.Length < 9 Or InStr(Client_name, ",",
CompareMethod.Text) = 0 Then
            MessageBox.Show("Please follow the given format for your
name and make sure it's valid", "Client's Info", _
                            0, MessageBoxIcon.Information, 0, 0, False)
            txtName.Text = Nothing
            cmbName.Text = Nothing
            Exit Sub
        End If
        Dim FileWriter As StreamWriter
        If File.Exists(ListPath & "\" & Client_name & ".end") = True
Then
            FileWriter = New StreamWriter(ListPath & "\Current.end",
False)
            FileWriter.WriteLine(Client_name)
            FileWriter.WriteLine(txtAge.Text)
            FileWriter.Close()
            Clear_Fields()
            chkSelect.Checked = False
            Me.Hide()
            frmEndurance.Show()
        Else
            MessageBox.Show("No such Client exists in our database",
"Selecting Client", 0, _
                            MessageBoxIcon.Exclamation, 0, 0, False)
        End If
    End Sub
    Private Sub chkSelect_CheckedChanged(ByVal sender As System.Object,
ByVal e As EventArgs) _
    Handles chkSelect.CheckedChanged
        cmbName.Visible = chkSelect.Checked
        btnSelect.Enabled = chkSelect.Checked
        btnDelete.Enabled = chkSelect.Checked
        If chkSelect.Checked = True Then
            txtName.Visible = False
            Clear_Fields()
            List_Clients()
        ElseIf chkSelect.Checked = False Then
            txtName.Visible = True
            Clear_Fields()
        End If
    End Sub
    Private Sub cmbName_SelectedIndexChanged(ByVal sender As
System.Object, ByVal e As EventArgs) _
    Handles cmbName.SelectedIndexChanged
        Dim FileReader As StreamReader
        FileReader = New StreamReader(ListPath & "\" &
cmbName.SelectedItem.ToString & ".end")
```

```
        FileReader.ReadLine()
        txtAge.Text = FileReader.ReadLine()
        temp = FileReader.ReadLine()
        If temp = "n/a" Then
            txtContact.Text = Nothing
        Else
            txtContact.Text = temp
        End If
        gender = FileReader.ReadLine()
        FileReader.Close()
        If gender = "Male" Then
            rdbMale.Checked = True
        ElseIf gender = "Female" Then
            rdbFemale.Checked = True
        End If
    End Sub
    Private Sub txtName_TextChanged(ByVal sender As System.Object,
ByVal e As EventArgs) Handles txtName.TextChanged
        If txtName.TextLength < 7 And txtName.TextLength > 0 Then
            lblCheck.Text = Nothing
            Exit Sub
        ElseIf txtName.Text = Nothing Then
            lblCheck.Text = "Please enter a name"
            Exit Sub
        End If
        If File.Exists(ListPath & "\" & txtName.Text & ".end") = True
And _
        InStr(txtName.Text, ",", CompareMethod.Text) <> 0 And
InStr(txtName.Text, ".", CompareMethod.Text) = 0 Then
            lblCheck.Text = "Name is not available"
        ElseIf File.Exists(ListPath & "\" & txtName.Text & ".end") =
False And _
        InStr(txtName.Text, ",", CompareMethod.Text) <> 0 And
InStr(txtName.Text, ".", CompareMethod.Text) = 0 Then
            lblCheck.Text = "Name is available"
        ElseIf InStr(txtName.Text, ".", CompareMethod.Text) <> 0 Then
            lblCheck.Text = "Period (.) is not needed"
        ElseIf InStr(txtName.Text, ",", CompareMethod.Text) = 0 Then
            lblCheck.Text = "Don't forget the comma (,)"
        End If
    End Sub
    Private Sub btnDelete_Click(ByVal sender As System.Object, ByVal e
As EventArgs) Handles btnDelete.Click
        Check_Name(cmbName.Text, txtName.Text)
        If File.Exists(ListPath & "\" & Client_name & ".end") = True
Then
            Dim FileRemover As New FileInfo(ListPath & "\" &
Client_name & ".end")
            FileRemover.Delete()
            Dim x, y As Integer
            Dim FileReader As StreamReader
            FileReader = New StreamReader(ListPath & "\Index.end")
            y = 0
            While Not FileReader.EndOfStream
                FileReader.ReadLine()
                y += 1
            End While
```

```
            FileReader.Close()
            FileRemover = New FileInfo(ListPath & "\Index.end")
            FileRemover.Delete()
            Dim FileWriter As StreamWriter
            FileWriter = New StreamWriter(ListPath & "\Index.end")
            For x = 0 To y - 1
                If cmbName.Items.Item(x) <> Nothing Then
                    If cmbName.Items.Item(x) <> Client_name Then
                        FileWriter.WriteLine(cmbName.Items.Item(x))
                    End If
                End If
            Next x
            FileWriter.Close()
            If File.Exists(ListPath & "\Current.end") = True Then
                FileReader = New StreamReader(ListPath &
"\Current.end")
                temp = FileReader.ReadLine()
                FileReader.Close()
                If temp = Client_name Then
                    FileRemover = New FileInfo(ListPath &
"\Current.end")
                    FileRemover.Delete()
                End If
            End If
            For x = 1 To 32767
                If File.Exists(DataPath & "\Initial_" & Client_name & x
& ".end") = False Then
                    Exit For
                ElseIf File.Exists(DataPath & "\Initial_" & Client_name
& x & ".end") = True Then
                    FileRemover = New FileInfo(DataPath & "\Initial_" &
Client_name & x & ".end")
                    FileRemover.Delete()
                End If
            Next x
            For x = 1 To 32767
                If File.Exists(DataPath & "\Day1_" & Client_name & x &
".end") = False Then
                    Exit For
                End If
                y = 1
                While y <> 32767
                    If File.Exists(DataPath & "\Day" & y & "_" &
Client_name & x & ".end") = False Then
                        Exit While
                    ElseIf File.Exists(DataPath & "\Day" & y & "_" &
Client_name & x & ".end") = True Then
                        FileRemover = New FileInfo(DataPath & "\Day" &
y & "_" & Client_name & x & ".end")
                        FileRemover.Delete()
                    End If
                    y += 1
                End While
            Next x
            MessageBox.Show("Client and associated records have been
deleted", "Deleting Client", 0, _
                            MessageBoxIcon.Information, 0, 0, False)
```

```
        Else
            MessageBox.Show("No such Client exists in our database",
"Selecting Client", 0, _
                            MessageBoxIcon.Exclamation, 0, 0, False)
        End If
    End Sub
    Private Sub btnBack_Click(ByVal sender As System.Object, ByVal e As
EventArgs) Handles btnBack.Click
        Me.Close()
        frmMain.Show()
    End Sub
End Class
```

Help Form

```
Public Class frmHelp
    Private Sub btnBack_Click(ByVal sender As System.Object, ByVal e As
EventArgs) Handles btnBack.Click
        Me.Close()
    End Sub
    Private Sub Help_FormClosing(ByVal sender As System.Object, ByVal e
As EventArgs) Handles MyBase.FormClosing
        frmMain.Show()
    End Sub
End Class
```

SEAMs Form

```
Public Class frmSEAMs
    Private Sub btnBack_Click(ByVal sender As System.Object, ByVal e As
EventArgs)
        Me.Close()
    End Sub
    Private Sub frmSEAMs_FormClosing(ByVal sender As System.Object,
ByVal e As EventArgs) Handles MyBase.FormClosing
        frmMain.Show()
    End Sub
    Private Sub btnBack_Click_1(ByVal sender As System.Object, ByVal e
As EventArgs) Handles btnBack.Click
        Me.Close()
        frmMain.Show()
    End Sub
End Class
```

APPENDIX D

Data Sheet

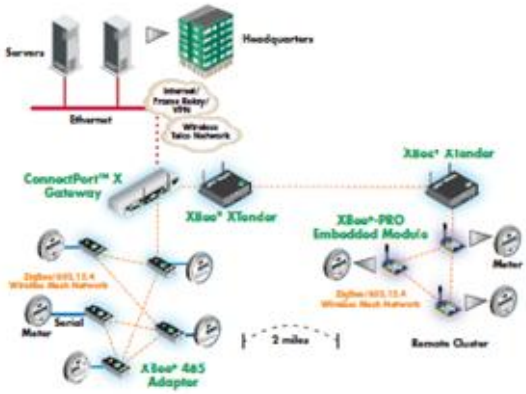
- 78L05
- 7805
- ADXL330
- PIC16F877A
- RT9163T
- L072CN
- VDIP1
- Zigbee Pro

Product Datasheet

XBee® & XBee-PRO® ZB

ZigBee® Embedded RF Modules for OEMs

A critical component to Drop-in Networking, XBee and XBee-PRO ZB embedded RF modules deliver low-cost and low-power wireless end-point connectivity using the ZigBee PRO Feature Set.



Features/Benefits

- ZigBee PRO Feature Set
- Advanced mesh networking
- Over-the-air firmware updates
- ZigBee Certification coming soon
- Low-power/low-cost XBee modules
- Extended-range XBee-PRO modules
- 2.4 GHz for worldwide deployment
- Fully interoperable with other Digi Drop-in Networking products, including gateways, device adapters and extenders
- Common XBee footprint for a variety of RF modules
- Multiple antenna options
- Industrial temperature rating (-40° C to 85° C)

Overview

XBee and XBee-PRO ZB are the most advanced ZigBee modules available in the XBee footprint and are ideal for deployment in ZigBee networks. Available in the low-cost XBee or extended-range XBee-PRO, and utilizing the ZigBee PRO Feature Set, these embedded RF modules are interoperable with ZigBee PRO Feature Set devices from other vendors. With advanced mesh networking functionality, XBee and XBee-PRO ZB modules improve data traffic management, allow for greater node density, and provide OEMs with the ability to change firmware remotely with over-the-air updates.

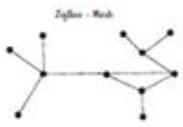
XBee Protocols

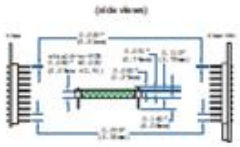
XBee embedded modules are available with different protocols to suit a variety of applications and networking topologies. Supported protocols include IEEE 802.15.4, the ZigBee PRO Feature Set, proprietary long range, and DigiMesh™. XBee embedded RF modules share a common hardware footprint and are modeled after a common software API. Once deployed into an application, OEMs can rapidly change from one protocol to another with minimal time and development risk.

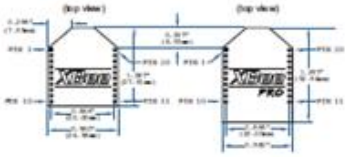
Drop-in Networking Compatibility

XBee embedded RF modules are compatible with Digi's Drop-in Networking adapters, network extenders and gateways that use the same protocol. This allows OEMs to embed XBee solutions into an application and have seamless communication to other devices using USB, RS-232, RS-485, digital I/O, analog I/O, Ethernet, Wi-Fi and and even cellular connections with plug-and-play ease.

Platform	XBee® ZB	XBee-PRO® ZB
Performance		
RF Data Rate	250 Kbps	250 Kbps
Indoor/Urban Range	133 ft (40 m)	300 ft (90 m)
Outdoor/RF Line-of-Sight Range	400 ft (120 m)	1 mi (1.6 km)
Transmit Power	1.25 mW (+1 dBm) / 2 mW (+3 dBm) boost mode	50 mW (+17 dBm) / Int'l 10 mW (+10 dBm)
Receiver Sensitivity (1% PER)	-96 dBm in boost mode	-102 dBm
Features		
Serial Data Interface	3.3V CMOS UART	3.3V CMOS UART
Configuration Method	API or AT commands, local or over-the-air	API or AT commands, local or over-the-air
Frequency Band	2.4 GHz	2.4 GHz
Interference Immunity	DSSS (Direct Sequence Spread Spectrum)	DSSS (Direct Sequence Spread Spectrum)
Serial Data Rate	1200 bps - 1 Mbps	1200 bps - 1 Mbps
ADC Inputs	(4) 10-bit ADC inputs	(4) 10-bit ADC inputs
Digital I/O	10	10
Antenna Options	Chip, Wire Whip, U.FL, RPSMA	Chip, Wire Whip, U.FL, RPSMA
Networking & Security		
Encryption	128-bit AES	128-bit AES
Reliable Packet Delivery	Retries/Acknowledgments	Retries/Acknowledgments
IDs and Channels	PAN ID, 64-bit IEEE MAC, 16 channels	PAN ID, 64-bit IEEE MAC, 13 channels
Power Requirements		
Supply Voltage	2.1 - 3.6VDC	3.0 - 3.4VDC
Transmit Current	35 mA / 45 mA boost mode @ 3.3VDC	295 mA @ 3.3VDC
Receive Current	38 mA / 40 mA boost mode @ 3.3VDC	45 mA @ 3.3VDC
Power-Down Current	<1 uA @ 25° C	<10 uA @ 25° C
Regulatory Approvals		
FCC (USA)	Yes	Yes
IC (Canada)	Yes	Yes
ETSI (Europe)	Yes	Yes (int'l unit only)
C-TICK (Australia)	Yes	Pending
Telec (Japan)	Yes	Pending (int'l unit only)







Please visit www.digi.com for part numbers.

DIGI SERVICE AND SUPPORT - You can purchase with confidence knowing that Digi is here to support you with expert technical support and a one-year warranty. www.digi.com/support

WHEN RELIABILITY MATTERS™

Digi International
11001 Bren Road E.
Minnetonka, MN 55343
U.S.A.
PH: 877-912-3444
952-912-3444
FX: 952-912-4952
email: info@digi.com

Digi International France
31 rue des Poissonniers
92200 Neuilly sur Seine
PH: +33-1-55-61-98-98
FX: +33-1-55-61-98-99
www.digi.fr

Digi International KK
NES Building South 8F
22-14 Sakuragaoka-cho,
Shibuya-ku
Tokyo 150-0031, Japan
PH: +81-3-5428-0261
FX: +81-3-5428-0262
www.digi-intl.co.jp

Digi International (HK) Limited
Suite 1703-05, 17/F,
K Wah Centre
191 Java Road
North Point, Hong Kong
PH: +852-2833-1008
FX: +852-2572-9989
www.digi.cn

Digi International, the leader in device networking for business, develops reliable products and technologies to connect and securely manage local or remote electronic devices over the network or via the web. With over 20 million ports shipped worldwide since 1985, Digi offers the highest levels of performance, flexibility and quality.

www.digi.com

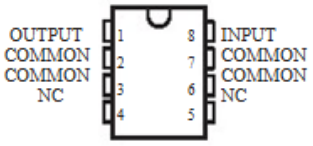
91001471
A1/608

∝A78Λ00 SERIES
POSITIVE-VOLTAGE REGULATORS

SLVS010I – JANUARY 1976 – REVISED JULY 1999

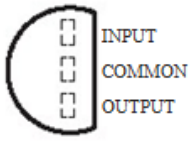
- 3-Terminal Regulators
- Output Current up to 100 mA
- No External Components
- Internal Thermal-Overload Protection
- Internal Short-Circuit Current Limiting
- Direct Replacements for Fairchild ∝A78Λ00 Series

D PACKAGE
(TOP VIEW)



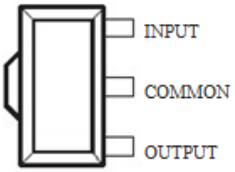
NC – No internal connection

LP PACKAGE
(TOP VIEW)



TO–226AA

PK PACKAGE
(TOP VIEW)



description

This series of fixed-voltage integrated-circuit voltage regulators is designed for a wide range of applications. These applications include on-card regulation for elimination of noise and distribution problems associated with single-point regulation. In addition, they can be used with power-pass elements to make high-current voltage regulators. One of these regulators can deliver up to 100 mA of output current. The internal limiting and thermal-shutdown features of these regulators make them essentially immune to overload. When used as a replacement for a zener diode-resistor combination, an effective improvement in output impedance can be obtained, together with lower bias current.

The ∝A78Λ00X series is characterized for operation over the virtual junction temperature range of 0°C to 125°C.

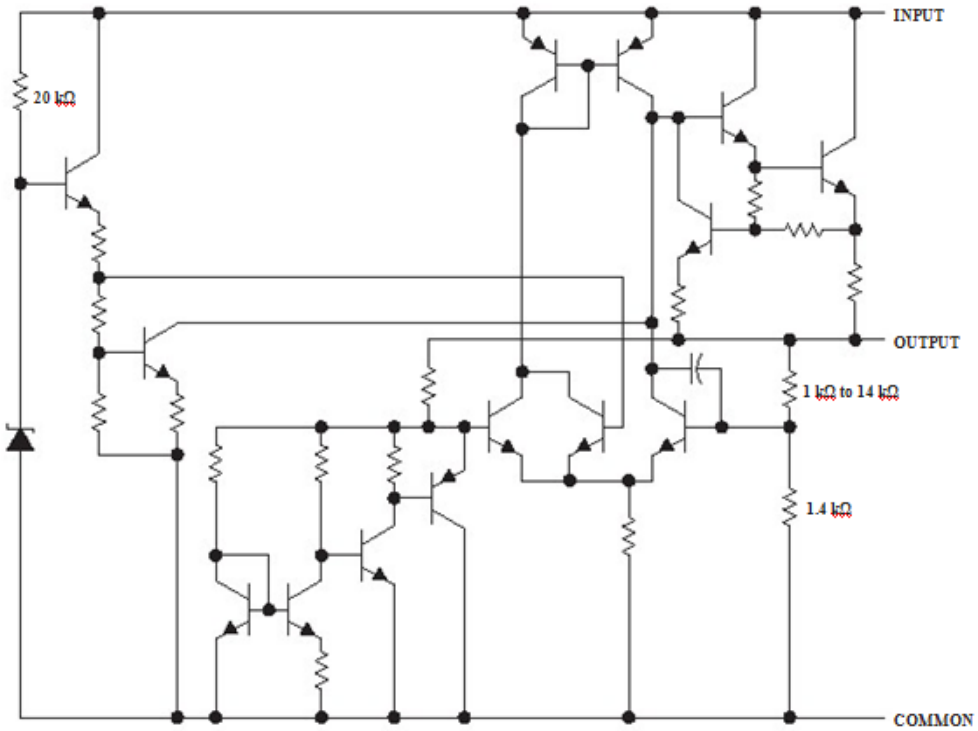
AVAILABLE OPTIONS

TJ	VO(NOM) (V)	PACKAGED DEVICES						CHIP FORM (Y)
		SMALL OUTLINE (D)		PLASTIC CYLINDRICAL (LP)		SOT-89 (PK)		
		OUTPUT VOLTAGE TOLERANCE						
		5%	10%	5%	10%	5%	10%	
0°C to 125°C	2.6	∝A78Λ02AXΔ	–	∝A78Λ02AXΛΠ	∝A78Λ02XΛΠ	∝A78Λ02AXΠK	∝A78Λ02XΠK	∝A78Λ02Ψ
	5	∝A78Λ05AXΔ	∝A78Λ05XΔ	∝A78Λ05AXΛΠ	∝A78Λ05XΛΠ	∝A78Λ05AXΠK	∝A78Λ05XΠK	∝A78Λ05Ψ
	6.2	∝A78Λ06AXΔ	∝A78Λ06XΔ	∝A78Λ06AXΛΠ	∝A78Λ06XΛΠ	∝A78Λ06AXΠK	∝A78Λ06XΠK	∝A78Λ06Ψ
	8	∝A78Λ08AXΔ	∝A78Λ08XΔ	∝A78Λ08AXΛΠ	∝A78Λ08XΛΠ	∝A78Λ08AXΠK	∝A78Λ08XΠK	∝A78Λ08Ψ
	9	∝A78Λ09AXΔ	∝A78Λ09XΔ	∝A78Λ09AXΛΠ	∝A78Λ09XΛΠ	∝A78Λ09AXΠK	∝A78Λ09XΠK	∝A78Λ09Ψ
	10	∝A78Λ10AXΔ	–	∝A78Λ10AXΛΠ	∝A78Λ10XΛΠ	∝A78Λ10AXΠK	∝A78Λ10XΠK	∝A78Λ10Ψ
	12	∝A78Λ12AXΔ	∝A78Λ12XΔ	∝A78Λ12AXΛΠ	∝A78Λ12XΛΠ	∝A78Λ12AXΠK	∝A78Λ12XΠK	∝A78Λ12Ψ
	15	∝A78Λ15AXΔ	∝A78Λ15XΔ	∝A78Λ15AXΛΠ	∝A78Λ15XΛΠ	∝A78Λ15AXΠK	∝A78Λ15XΠK	∝A78Λ15Ψ

D and LP packages are available taped and reeled. Add the suffix R to the device type (e.g., ∝A78Λ05AXΔP). The PK package is only available taped and reeled (e.g., ∝A78Λ02AXΠKP). Chip forms are tested at TA = 25°C.

TEXAS INSTRUMENTS
POST OFFICE BOX 655303 ☐ DALLAS, TEXAS 75265

schematic



NOTE: Resistor values shown are nominal.

electrical characteristics at specified virtual junction temperature, VI = 9 V, IO = 40 mA, TJ = 25°C (unless otherwise noted)

PARAMETER	TEST CONDITIONS†	∝A78Λ02Ψ			UNIT
		MIN	TYP	MAX	
Output voltage			2.6		V
Input voltage regulation	VI = 4.75 V to 20 V		20		mV
	VI = 5 V to 20 V		16		
Ripple rejection	VI = 6 V to 20 V, f = 120 Hz		51		dB
Output voltage regulation	IO = 1 mA to 100 mA		12		mV
	IO = 1 mA to 40 mA		6		
Output noise voltage	f = 10 Hz to 100 kHz		30		∝ς
Dropout voltage			1.7		V
Bias current			3.6		mA

† Pulse-testing techniques maintain TJ as close to TA as possible. Thermal effects must be taken into account separately. All characteristics are measured with a 0.33-µF capacitor across the input and a 0.1-µF capacitor across the output.

electrical characteristics at specified virtual junction temperature, VI = 10 V, IO = 40 mA, TJ = 25°C (unless otherwise noted)

PARAMETER	TEST CONDITIONS†	∝A78Λ05Ψ			UNIT
		MIN	TYP	MAX	
Output voltage			5		V
Input voltage regulation	VI = 7 V to 20 V		32		mV
	VI = 8 V to 20 V		26		
Ripple rejection	VI = 8 V to 18 V, f = 120 Hz		49		dB
Output voltage regulation	IO = 1 mA to 100 mA		15		mV
	IO = 1 mA to 40 mA		8		
Output noise voltage	f = 10 Hz to 100 kHz		42		∝ς
Dropout voltage			1.7		V
Bias current			3.8		mA

† Pulse-testing techniques maintain TJ as close to TA as possible. Thermal effects must be taken into account separately. All characteristics are measured with a 0.33-µF capacitor across the input and a 0.1-µF capacitor across the output.

electrical characteristics at specified virtual junction temperature, VI = 12 V, IO = 40 mA, TJ = 25°C (unless otherwise noted)

PARAMETER	TEST CONDITIONS†	∝A78Λ06Ψ			UNIT
		MIN	TYP	MAX	
Output voltage			6.2		V
Input voltage regulation	VI = 8.5 V to 20 V		35		mV
	VI = 9 V to 20 V		29		
Ripple rejection	VI = 10 V to 20 V, f = 120 Hz		48		dB
Output voltage regulation	IO = 1 mA to 100 mA		16		mV
	IO = 1 mA to 40 mA		9		
Output noise voltage	f = 10 Hz to 100 kHz		46		∝ς
Dropout voltage			1.7		V
Bias current			3.9		mA

† Pulse-testing techniques maintain TJ as close to TA as possible. Thermal effects must be taken into account separately. All characteristics are measured with a 0.33-µF capacitor across the input and a 0.1-µF capacitor across the output.

PIC16F87XA
Data Sheet

28/40/44-Pin Enhanced Flash Microcontrollers

28/40/44-Pin Enhanced Flash Microcontrollers

Devices Included in this Data Sheet:

- PIC16F873A
- PIC16F874A
- PIC16F876A
- PIC16F877A

High-Performance RISC CPU:

- Only 35 single-word instructions to learn
- All single-cycle instructions except for program branches, which are two-cycle
- Operating speed: DC – 20 MHz clock input
  DC – 200 ns instruction cycle
- 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
- Pinout compatible to other 28-pin or 40/44-pin PIC16CXXX and PIC16FXXX microcontrollers

Peripheral Features:

- Timer0: 8-bit timer/counter with 8-bit prescaler
- Timer1: 16-bit timer/counter with prescaler, can be incremented during Sleep via external crystal/clock
- Timer2: 8-bit timer/counter with 8-bit period register, prescaler and postscaler
- Two Capture, Compare, PWM modules
  - Capture is 16-bit, max. resolution is 12.5 ns
  - Compare is 16-bit, max. resolution is 200 ns
  - PWM max. resolution is 10-bit
- Synchronous Serial Port (SSP) with SPI™ (Master mode) and I²C™ (Master/Slave)
- Universal Synchronous Asynchronous Receiver Transmitter (USART/SCI) with 9-bit address detection
- Parallel Slave Port (PSP) – 8 bits wide with external RD, WR and CS controls (40/44-pin only)
- Brown-out detection circuitry for Brown-out Reset (BOR)

Analog Features:

- 10-bit, up to 8-channel Analog-to-Digital Converter (A/D)
- Brown-out Reset (BOR)
- Analog Comparator module with:
  - Two analog comparators
  - Programmable on-chip voltage reference (V_{REF}) module
  - Programmable input multiplexing from device inputs and internal voltage reference
  - Comparator outputs are externally accessible

Special Microcontroller Features:

- 100,000 erase/write cycle Enhanced Flash program memory typical
- 1,000,000 erase/write cycle Data EEPROM memory typical
- Data EEPROM Retention > 40 years
- Self-reprogrammable under software control
- In-Circuit Serial Programming™ (ICSP™) via two pins
- Single-supply 5V In-Circuit Serial Programming
- Watchdog Timer (WDT) with its own on-chip RC oscillator for reliable operation
- Programmable code protection
- Power saving Sleep mode
- Selectable oscillator options
- In-Circuit Debug (ICD) via two pins

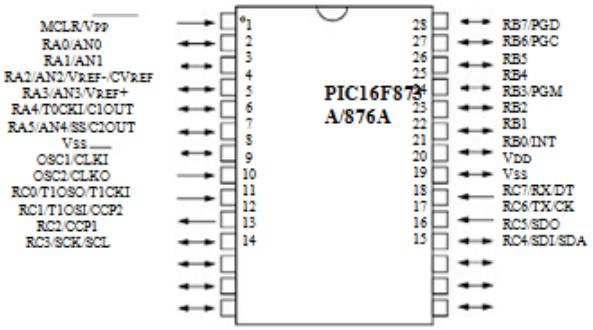
CMOS Technology:

- Low-power, high-speed Flash/EEPROM technology
- Fully static design
- Wide operating voltage range (2.0V to 5.5V)
- Commercial and Industrial temperature ranges
- Low-power consumption

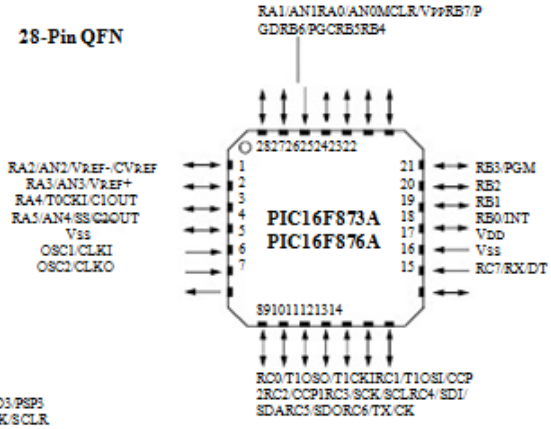
Device	Program Memory # Single Word Instructions	Data SRAM (Bytes)	EEPROM (Bytes)	I/O	A/D (ch)	10-bit PWM	CCP Timers	MSSP Master SPI	MSSP Master I²C	USART	8/16-bit Timers	Comparators
PIC16F873A	7.2K	4096	192	128	22	5	2	Yes	Yes	Yes	2/1	2
PIC16F874A	7.2K	4096	192	128	33	8	2	Yes	Yes	Yes	2/1	2
PIC16F876A	14.3K	8192	368	256	22	5	2	Yes	Yes	Yes	2/1	2
PIC16F877A	14.3K	8192	368	256	33	8	2	Yes	Yes	Yes	2/1	2

Pin Diagrams

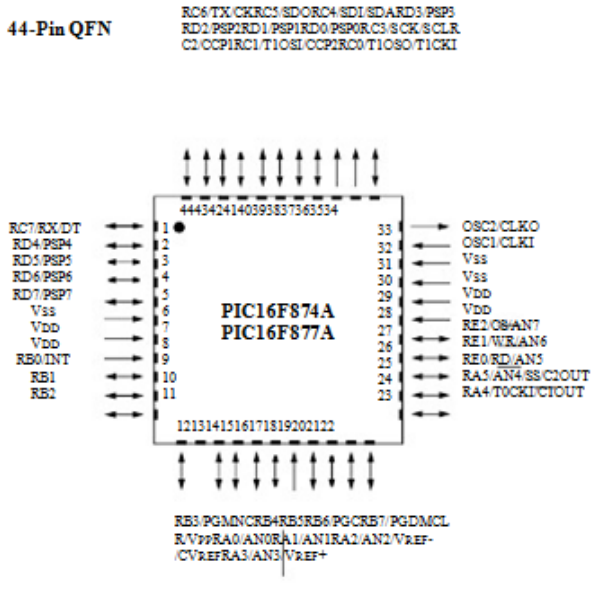
28-Pin PDIP, SOIC, SSOP



28-Pin QFN

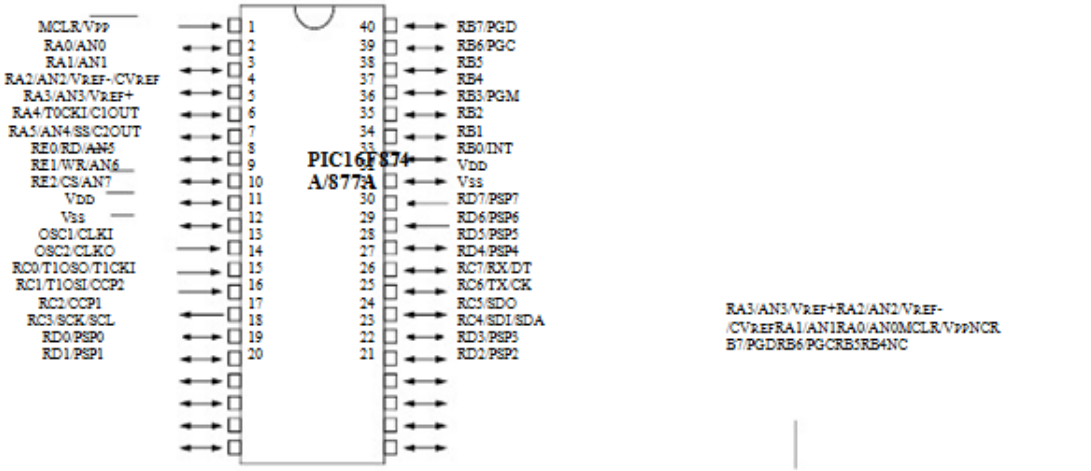


44-Pin QFN

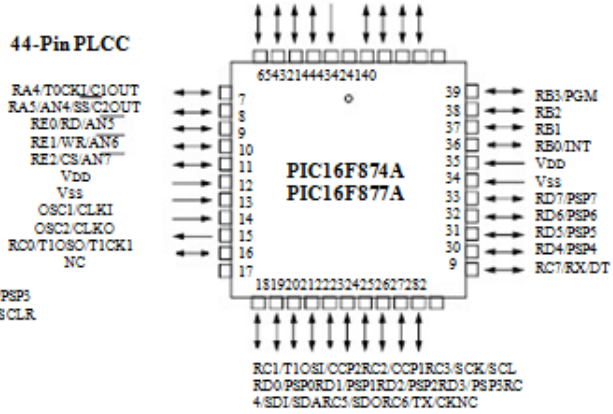


Pin Diagrams (Continued)

40-Pin PDIP



44-Pin PLCC

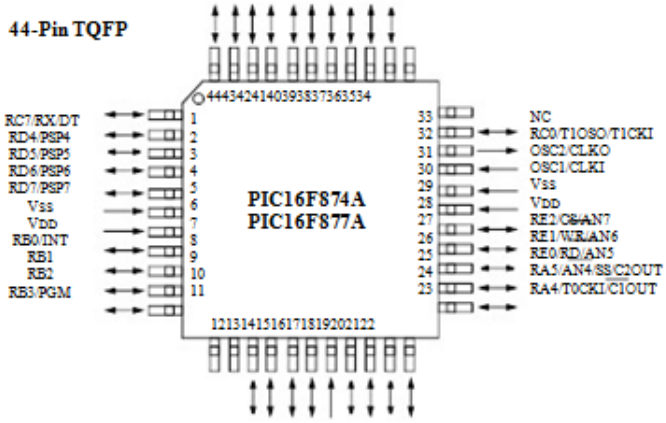


RA3/AN3/VREF+RA2/AN2/VREF-/CVREFRA1/AN1RA0/AN0MCLR/VPPNCRB7/PGDRB6/PGCRB5RB4NC

RC6/TX/CKRC5/SDORC4/SDI/SDARD3/PSP3RD2/PSP2RD1/PSP1RD0/PSP0RC3/SCK/SCLRC2/CCP1RC1/T1OSI/CCP1NC

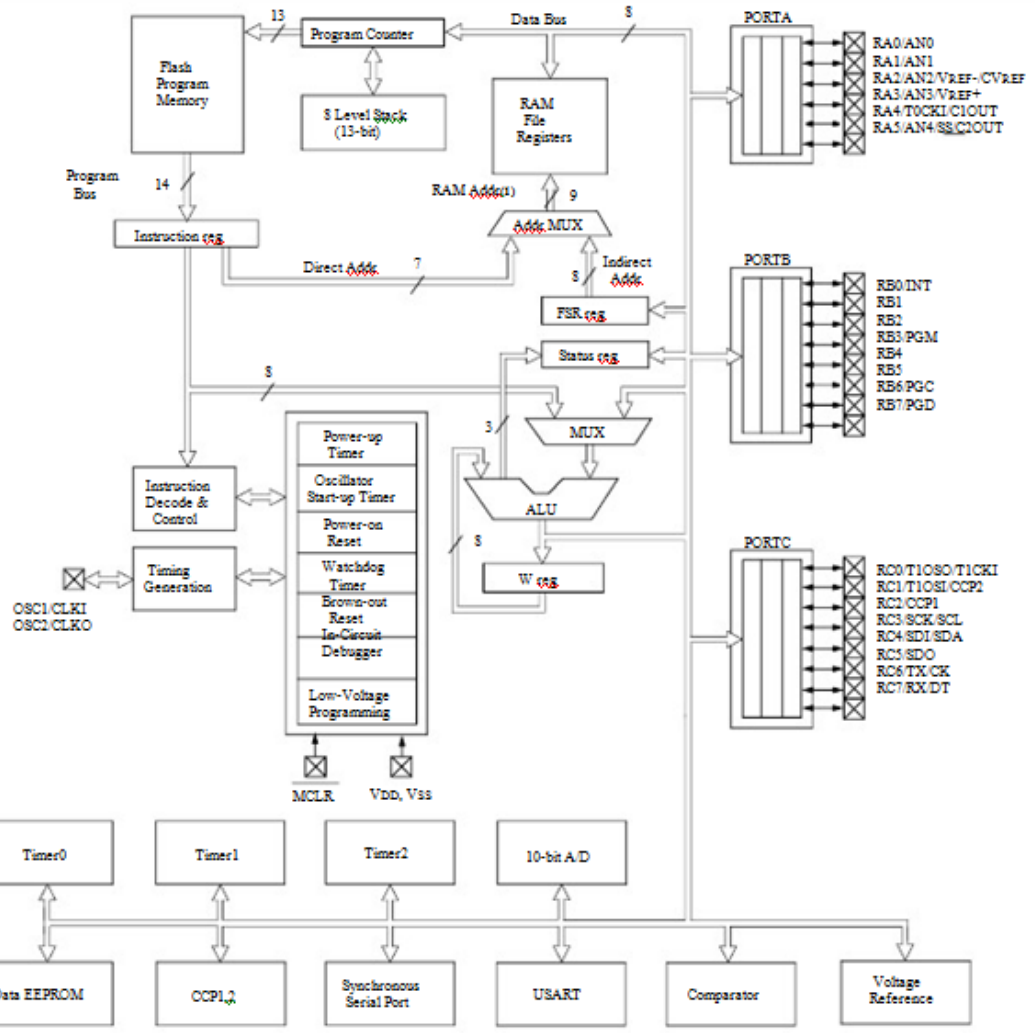
RC1/T1OSI/CCP1RC2/CCP1RC3/SCK/SCLRD0/PSP0RD1/PSP1RD2/PSP2RD3/PSP3RC4/SDI/SDARC5/SDORC6/TX/CKNC

44-Pin TQFP



NCNCRB4RB5RB6/PGCRB7/PGDMCLR/VPPRA0/AN0RA1/AN1RA2/AN2/VREF-/CVREFRA3/AN3/VREF+

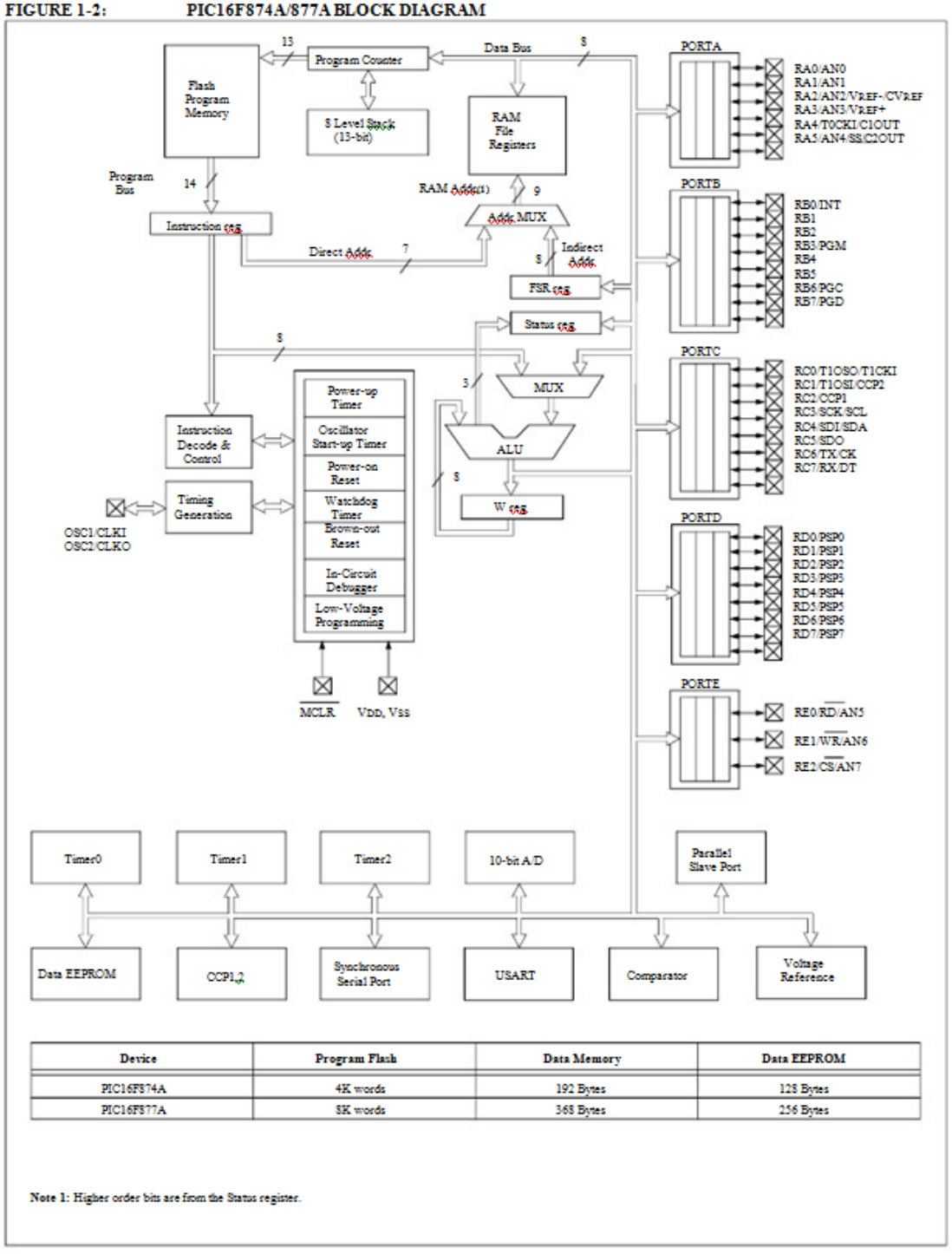
FIGURE 1-1: PIC16F873A/876A BLOCK DIAGRAM



Device	Program Flash	Data Memory	Data EEPROM
PIC16F873A	4K words	192 Bytes	128 Bytes
PIC16F876A	8K words	368 Bytes	256 Bytes

Note 1: Higher order bits are from the Status register.

FIGURE 1-2: PIC16F874A/877A BLOCK DIAGRAM



Device	Program Flash	Data Memory	Data EEPROM
PIC16F874A	4K words	192 Bytes	128 Bytes
PIC16F877A	8K words	368 Bytes	256 Bytes

Note 1: Higher order bits are from the Status register.

TABLE 1-2: PIC16F873A/876A PINOUT DESCRIPTION

Pin Name	PDIP, SOIC, SSOP Pin#	QFN Pin#	I/O/P Type	Buffer Type	Description
OSC1/CLKI	9	6		ST/CMOS(3)	Oscillator crystal or external clock input.
OSC1			I		Oscillator crystal input or external clock source input. ST buffer when configured in RC mode; otherwise CMOS.
CLKI			I		External clock source input. Always associated with pin function OSC1 (see OSC1/CLKI, OSC2/CLKO pins).
OSC2/CLKO	10	7		—	Oscillator crystal or clock output.
OSC2			O		Oscillator crystal output. Connects to crystal or resonator in Crystal Oscillator mode.
CLKO			O		In RC mode, OSC2 pin outputs CLKO, which has 1/4 the frequency of OSC1 and denotes the instruction cycle rate.
MCLR/VPP	1	26		ST	Master Clear (input) or programming voltage (output).
MCLR			I		Master Clear (Reset) input. This pin is an active low Reset to the device.
VPP			P		Programming voltage input.
					PORTA is a bidirectional I/O port.
RA0/AN0	2	27		TTL	
RA0			I/O		Digital I/O.
AN0			I		Analog input 0.
RA1/AN1	3	28		TTL	
RA1			I/O		Digital I/O.
AN1			I		Analog input 1.
RA2/AN2/VREF-/CVREF	4	1		TTL	
RA2			I/O		Digital I/O.
AN2			I		Analog input 2.
VREF-			I		A/D reference voltage (Low) input.
CVREF			O		Comparator VREF output.
RA3/AN3/VREF+	5	2		TTL	
RA3			I/O		Digital I/O.
AN3			I		Analog input 3.
VREF+			I		A/D reference voltage (High) input.
RA4/T0CKI/C1OUT	6	3		ST	
RA4			I/O		Digital I/O – Open-drain when configured as output.
T0CKI			I		Timer0 external clock input.
C1OUT			O		Comparator 1 output.
RA5/AN4/SS/C2OUT	7	4		TTL	
RA5			I/O		Digital I/O.
AN4			I		Analog input 4.
SS			I		SPI slave select input.
C2OUT			O		Comparator 2 output.

Legend: I = input O = output I/O = input/output P = power
— = Not used TTL = TTL input ST = Schmitt Trigger input

Note 1: This buffer is a Schmitt Trigger input when configured as the external interrupt.
2: This buffer is a Schmitt Trigger input when used in Serial Programming mode.
3: This buffer is a Schmitt Trigger input when configured in RC Oscillator mode and a CMOS input otherwise.

TABLE 1-2: PIC16F873A/876A PINOUT DESCRIPTION (CONTINUED)

Pin Name	PDIP, SOIC, SSOP Pin#	QFN Pin#	I/O/P Type	Buffer Type	Description
					PORTB is a bidirectional I/O port. PORTB can be software programmed for internal weak pull-ups on all inputs.
RB0/INT	21	18		TTL/ST(1)	
RB0			I/O		Digital I/O.
INT			I		External interrupt.
RB1	22	19	I/O	TTL	Digital I/O.
RB2	23	20	I/O	TTL	Digital I/O.
RB3/PGM	24	21		TTL	
RB3			I/O		Digital I/O.
PGM			I		Low-voltage (single-supply) ICSP programming enable pin.
RB4	25	22	I/O	TTL	Digital I/O.
RB5	26	23	I/O	TTL	Digital I/O.
RB6/PGC	27	24		TTL/ST(2)	
RB6			I/O		Digital I/O.
PGC			I		In-circuit debugger and ICSP programming clock.
RB7/PGD	28	25		TTL/ST(2)	
RB7			I/O		Digital I/O.
PGD			I/O		In-circuit debugger and ICSP programming data.
					PORTC is a bidirectional I/O port.
RC0/T1OSO/T1CKI	11	8		ST	
RC0			I/O		Digital I/O.
T1OSO			O		Timer1 oscillator output.
T1CKI			I		Timer1 external clock input.
RC1/T1OSI/CCP2	12	9		ST	
RC1			I/O		Digital I/O.
T1OSI			I		Timer1 oscillator input.
CCP2			I/O		Capture2 input, Compare2 output, PWM2 output.
RC2/CCP1	13	10		ST	
RC2			I/O		Digital I/O.
CCP1			I/O		Capture1 input, Compare1 output, PWM1 output.
RC3/SCK/SCL	14	11		ST	
RC3			I/O		Digital I/O.
SCK			I/O		Synchronous serial clock input/output for SPI mode.
SCL			I/O		Synchronous serial clock input/output for I2C mode.
RC4/SDI/SDA	15	12		ST	
RC4			I/O		Digital I/O.
SDI			I		SPI data in.
SDA			I/O		I2C data I/O.
RC5/SDO	16	13		ST	
RC5			I/O		Digital I/O.
SDO			O		SPI data out.
RC6/TX/CK	17	14		ST	
RC6			I/O		Digital I/O.
TX			O		USART asynchronous transmit.
CK			I/O		USART1 synchronous clock.
RC7/RX/DT	18	15		ST	
RC7			I/O		Digital I/O.
RX			I		USART asynchronous receive.
DT			I/O		USART synchronous data.
VSS	8, 19	5, 6	P	—	Ground reference for logic and I/O pins.
VDD	20	17	P	—	Positive supply for logic and I/O pins.

Legend: I = input O = output I/O = input/output P = power
— = Not used TTL = TTL input ST = Schmitt Trigger input

Note 1: This buffer is a Schmitt Trigger input when configured as the external interrupt.
2: This buffer is a Schmitt Trigger input when used in Serial Programming mode.
3: This buffer is a Schmitt Trigger input when configured in RC Oscillator mode and a CMOS input otherwise.

TABLE 1-3: PIC16F874A/877A PINOUT DESCRIPTION

Pin Name	PDIP Pin#	PLCC Pin#	TQFP Pin#	QFN Pin#	I/O/P Type	Buffer Type	Description
OSC1/CLKI	13	14	30	32		ST/CMOS(4)	Oscillator crystal or external clock input.
OSC1					I		Oscillator crystal input or external clock source input. ST buffer when configured in RC mode; otherwise CMOS.
CLKI					I		External clock source input. Always associated with pin function OSC1 (see OSC1/CLKI, OSC2/CLKO pins).
OSC2/CLKO	14	15	31	33		—	Oscillator crystal or clock output.
OSC2					O		Oscillator crystal output. Connects to crystal or resonator in Crystal Oscillator mode.
CLKO					O		In RC mode, OSC2 pin outputs CLKO, which has 1/4 the frequency of OSC1 and denotes the instruction cycle rate.
MCLR/VPP	1	2	18	18		ST	Master Clear (input) or programming voltage (output).
MCLR					I		Master Clear (Reset) input. This pin is an active low Reset to the device.
VPP					P		Programming voltage input.
							PORTA is a bidirectional I/O port.
RA0/AN0	2	3	19	19		TTL	
RA0					I/O		Digital I/O.
AN0					I		Analog input 0.
RA1/AN1	3	4	20	20		TTL	
RA1					I/O		Digital I/O.
AN1					I		Analog input 1.
RA2/AN2/VREF-/CVREF	4	5	21	21		TTL	
RA2					I/O		Digital I/O.
AN2					I		Analog input 2.
VREF-					I		A/D reference voltage (Low) input.
CVREF					O		Comparator VREF output.
RA3/AN3/VREF+	5	6	22	22		TTL	
RA3					I/O		Digital I/O.
AN3					I		Analog input 3.
VREF+					I		A/D reference voltage (High) input.
RA4/T0CKI/C1OUT	6	7	23	23		ST	
RA4					I/O		Digital I/O – Open-drain when configured as output.
T0CKI					I		Timer0 external clock input.
C1OUT					O		Comparator 1 output.
RA5/AN4/SS/C2OUT	7	8	24	24		TTL	
RA5					I/O		Digital I/O.
AN4					I		Analog input 4.
SS					I		SPI slave select input.
C2OUT					O		Comparator 2 output.

Legend: I = inputO = outputI/O = input/outputP = power
— = Not usedTTL = TTL inputST = Schmitt Trigger input

Note 1: This buffer is a Schmitt Trigger input when configured as the external interrupt.
2: This buffer is a Schmitt Trigger input when used in Serial Programming mode.
3: This buffer is a Schmitt Trigger input when configured in RC Oscillator mode and a CMOS input otherwise.

TABLE 1-3: PIC16F874A/877A PINOUT DESCRIPTION (CONTINUED)

Pin Name	PDIP Pin#	PLCC Pin#	TQFP Pin#	QFN Pin#	I/O/P Type	Buffer Type	Description
							PORTB is a bidirectional I/O port. PORTB can be software programmed for internal weak pull-up on all inputs.
RB0/INT	33	36	8	9		TTL/ST(1)	
RB0					I/O		Digital I/O.
INT					I		External interrupt.
RB1	34	37	9	10	I/O	TTL	Digital I/O.
RB2	35	38	10	11	I/O	TTL	Digital I/O.
RB3/PGM	36	39	11	12		TTL	
RB3					I/O		Digital I/O.
PGM					I		Low-voltage ICSP programming enable pin.
RB4	37	41	14	14	I/O	TTL	Digital I/O.
RB5	38	42	15	15	I/O	TTL	Digital I/O.
RB6/PGC	39	43	16	16		TTL/ST(2)	
RB6					I/O		Digital I/O.
PGC					I		In-circuit debugger and ICSP programming clock.
RB7/PGD	40	44	17	17		TTL/ST(2)	
RB7					I/O		Digital I/O.
PGD					I/O		In-circuit debugger and ICSP programming data.

Legend: I = inputO = outputI/O = input/outputP = power
— = Not used TTL = TTL input ST = Schmitt Trigger input

Note 1: This buffer is a Schmitt Trigger input when configured as the external interrupt.
2: This buffer is a Schmitt Trigger input when used in Serial Programming mode.
3: This buffer is a Schmitt Trigger input when configured in RC Oscillator mode and a CMOS input otherwise.

TABLE 1-3: PIC16F874A/877A PINOUT DESCRIPTION (CONTINUED)

Pin Name	PDIP Pin#	PLCC Pin#	TQFP Pin#	QFN Pin#	I/O/P Type	Buffer Type	Description
							PORTD is a bidirectional I/O port or Parallel Slave Port when interfacing to a microprocessor bus.
RD0/PSP0	19	21	38	38		ST/TTL(3)	
RD0					I/O		Digital I/O.
PSP0					I/O		Parallel Slave Port data.
RD1/PSP1	20	22	39	39		ST/TTL(3)	
RD1					I/O		Digital I/O.
PSP1					I/O		Parallel Slave Port data.
RD2/PSP2	21	23	40	40		ST/TTL(3)	
RD2					I/O		Digital I/O.
PSP2					I/O		Parallel Slave Port data.
RD3/PSP3	22	24	41	41		ST/TTL(3)	
RD3					I/O		Digital I/O.
PSP3					I/O		Parallel Slave Port data.
RD4/PSP4	27	30	2	2		ST/TTL(3)	
RD4					I/O		Digital I/O.
PSP4					I/O		Parallel Slave Port data.
RD5/PSP5	28	31	3	3		ST/TTL(3)	
RD5					I/O		Digital I/O.
PSP5					I/O		Parallel Slave Port data.
RD6/PSP6	29	32	4	4		ST/TTL(3)	
RD6					I/O		Digital I/O.
PSP6					I/O		Parallel Slave Port data.
RD7/PSP7	30	33	5	5		ST/TTL(3)	
RD7					I/O		Digital I/O.
PSP7					I/O		Parallel Slave Port data.
							PORTE is a bidirectional I/O port.
RE0/$\overline{RD}$/AN5	8	9	25	25		ST/TTL(3)	
RE0					I/O		Digital I/O.
$\overline{RD}$					I		Read control for Parallel Slave Port.
AN5					I		Analog input 5.
RE1/$\overline{WR}$/AN6	9	10	26	26		ST/TTL(3)	
RE1					I/O		Digital I/O.
$\overline{WR}$					I		Write control for Parallel Slave Port.
AN6					I		Analog input 6.
RE2/$\overline{CS}$/AN7	10	11	27	27		ST/TTL(3)	
RE2					I/O		Digital I/O.
$\overline{CS}$					I		Chip select control for Parallel Slave Port.
AN7					I		Analog input 7.
Vss	12, 31	13, 34	6, 29	6, 30, 31	P	—	Ground reference for logic and I/O pins.
VDD	11, 32	12, 35	7, 28	7, 8, 28, 29	P	—	Positive supply for logic and I/O pins.
NC	—	1, 17, 28, 40	12, 13, 33, 34	13	—	—	These pins are not internally connected. These pins should be left unconnected.

Legend: I = input O = output I/O = input/output P = power
— = Not used TTL = TTL input ST = Schmitt Trigger input

Note 1: This buffer is a Schmitt Trigger input when configured as the external interrupt.
2: This buffer is a Schmitt Trigger input when used in Serial Programming mode.
3: This buffer is a Schmitt Trigger input when configured in RC Oscillator mode and a CMOS input otherwise.

VINCULUM

BINDING USB TECHNOLOGIES

Future Technology Devices International Ltd.

VDIP1

Vinculum VNC1L Module

Datasheet

Document Reference No.: FT_000016

Version 1.01

Issue Date: 2010-05-31

Future Technology Devices International Ltd (FTDI)

Unit 1, 2 Seaward Place, Centurion Business Park, Glasgow, G41 1HH, United Kingdom

Tel.: +44 (0) 141 429 2777 Fax: + 44 (0) 141 429 2758

E-Mail (Support): support1@ftdichip.com

Web: http://www.vinculum.com

1 Introduction

The VDIP1 module is an MCU to embedded USB host controller development module for the VNC1L IC device. The VDIP1 is supplied on a PCB designed to fit into a 24 pin DIP socket, and provides access to the UART, parallel FIFO, and SPI interface pins on the VNC1L device, via its AD and AC bus pins. Not only is it ideal for developing and rapid prototyping of VNC1L designs, but also an attractive quantity discount structure makes this module suitable for incorporation into low and medium volume finished product designs.

Figure 1.1- VDIP1

The Vinculum VNC1L is the first of FTDI's Vinculum family of Embedded USB host controller integrated circuit devices. Not only is it able to handle the USB Host Interface, and data transfer functions but owing to the inbuilt MCU and embedded Flash memory, Vinculum can encapsulate the USB device classes as well. When interfacing to mass storage devices such as USB Flash drives, Vinculum also transparently handles the FAT File structure communicating via UART, SPI or parallel FIFO interfaces via a simple to implement command set. Vinculum provides a new cost effective solution for providing USB Host capability into products that previously did not have the hardware resources available. The VNC1L is available in Pb-free (RoHS compliant) compact 48-Lead LQFP package.

2 Features

The VDIP1 has the following features:

- Uses FTDI's VNC1L embedded dual USB host controller IC device
- USB single 'A' type USB socket to interface with USB peripheral devices
- Second USB interface port available via module pins if required
- Jumper selectable UART, parallel FIFO or SPI MCU interfaces
- Single 5V supply input from USB connection (no external supply necessary)
- Auxiliary 3.3 V / 200 mA power output to external logic.
- Program or update firmware via USB Flash disk or via UART/Parallel FIFO/SPI interface
- Power and traffic indicator LED's
- VNC1L firmware programming control pins PROG# and RESET# brought out onto jumper interface
- VDIP1 is a Pb-free, RoHS complaint development module.
- Schematics, and firmware files available for download from the Vinculum website

3 Pin Out and Signal Description

3.1 Module Pin Out

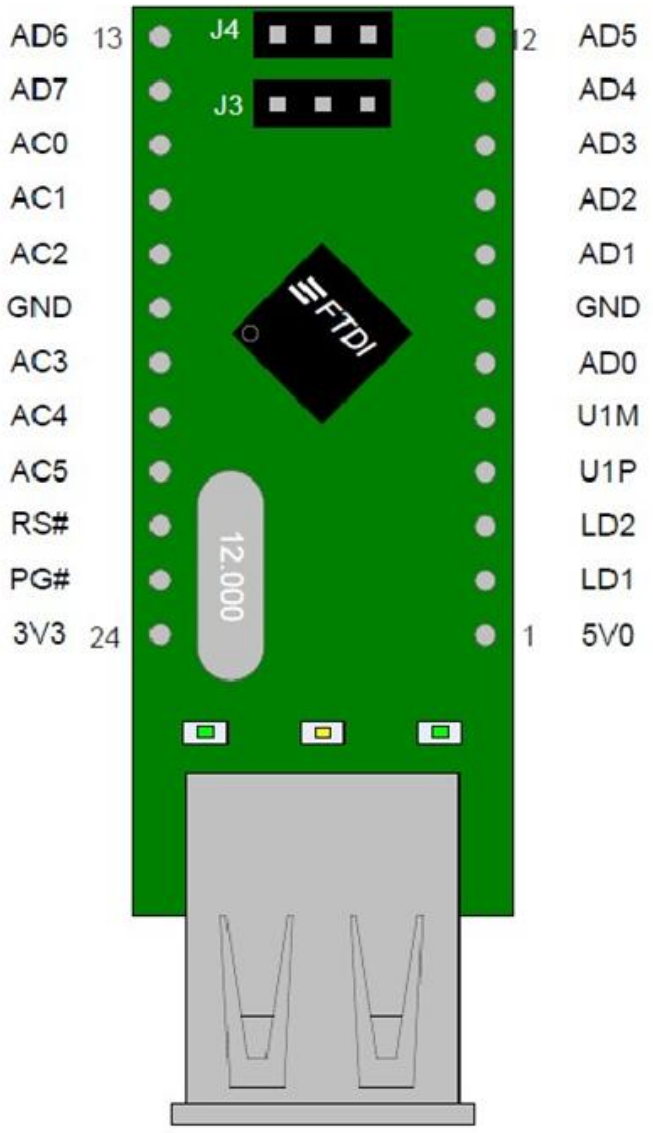


Figure 3.1 - VDIP1 Module Pin Out (Top View)

3.2 Pin Signal Description

Pin No.	Name	Pin Name on PCB	Type	Description
1	5V0	5V0	PWR Input	5.0 V module supply pin. This pin provides the 5.0V output on the USB 'A' type socket, and also the 3.3V supply to VNCL2, via an on-board 3.3 V L.D.O.
2	LED1	LD1	Output	USB port 1 traffic activity indicator LED. This pin is hard wired to a green LED on board the PCB. It is also brought out onto this pin which allows for the possibility of bring- ing out an additional LED traffic indicator out of the VDIP1 board. For example, if the VDIP1 USB connector is brought out onto an instrument front panel, an activity LED could be mounted along side it.
3	LED2	LD2	Output	USB port 2 traffic activity indicator LED. This pin is hard wired to a green LED on board the PCB. It is also brought out onto this pin which allows for the possibility of bring- ing out an additional LED traffic indicator out of the VDIP1 board. For example, if the VDIP1 USB connector is brought out onto an instrument front panel, an activity LED could be mounted along side it.
4	USBD1P	U1P	I/O	USB host / slave port 1 - USB Data Signal Plus with integrated pull up / pull down resistor. Module has on board 27 Ω USB series resistor. This pin can be brought out along with pin 5 to provide a second USB port, if required
5	USBD1M	U1M	I/O	USB host / slave port 1 - USB Data Signal Minus with integrated pull up / pull down resistor. Module has on board 27 Ω USB series resistor. This pin can be brought out along with pin 4 to provide a second USB port, if required
6	ADBUS0	AD0	I/O	5V safe bidirectional data / control bus, AD bit 0
7	GND	GND	PWR	Module ground supply pin
8	ADBUS1	AD1	I/O	5V safe bidirectional data / control bus, AD bit 1
9	ADBUS2	AD2	I/O	5V safe bidirectional data / control bus, AD bit 2
10	ADBUS3	AD3	I/O	5V safe bidirectional data / control bus, AD bit 3
11	ADBUS4	AD4	I/O	5V safe bidirectional data / control bus, AD bit 4
12	ADBUS5	AD5	I/O	5V safe bidirectional data / control bus, AD bit 5
13	ADBUS6	AD6	I/O	5V safe bidirectional data / control bus, AD bit 6
14	ADBUS7	AD7	I/O	5V safe bidirectional data / control bus, AD bit 7
15	ACBUS0	AC0	I/O	5V safe bidirectional data / control bus, AC bit 0
16	ACBUS1	AC1	I/O	5V safe bidirectional data / control bus, AC bit 1
17	ACBUS2	AC2	I/O	5V safe bidirectional data / control bus, AC bit 2
18	GND	GND	PWR	Module Ground Supply Pin
19	ACBUS3	AC3	I/O	5V safe bidirectional data / control bus, AC bit 3
20	ACBUS4	AC4	I/O	5V safe bidirectional data / control bus, AC bit 4
21	ACBUS5	AC5	I/O	5V safe bidirectional data / control bus, AC bit 5
22	RESET#	RS#	Input	Can be used by an external device to reset the VNC1L. This pin can be used in combination with PROG# and the UART / parallel FIFO / SPI interface to program firmware into the Vinculum
23	PROG#	PG#	Input	This pin is used in combination with the RESET# pin and the UART / parallel FIFO / SPI interface to pro gra m fir mware into t he VNC1L.
24	3V3	3V3	PWR	3.3V output from VDIP1's on board 3.3V L.D.O.

Table 3.1 - Pin Signal Descriptions

3.3 I/O Configuration Using The Jumper Pin Header

Two three way jumper pin headers are provided to allow for simple configuration of the I/O on data and control bus pins of the VDIP1. This is done by a combination of pulling up or pulling down the VNC1L ACBUS5 (pin 46) and ACBUS6 (pin 47). The relevant portion of the VDIP1 module schematic is shown in **Figure 3.2**

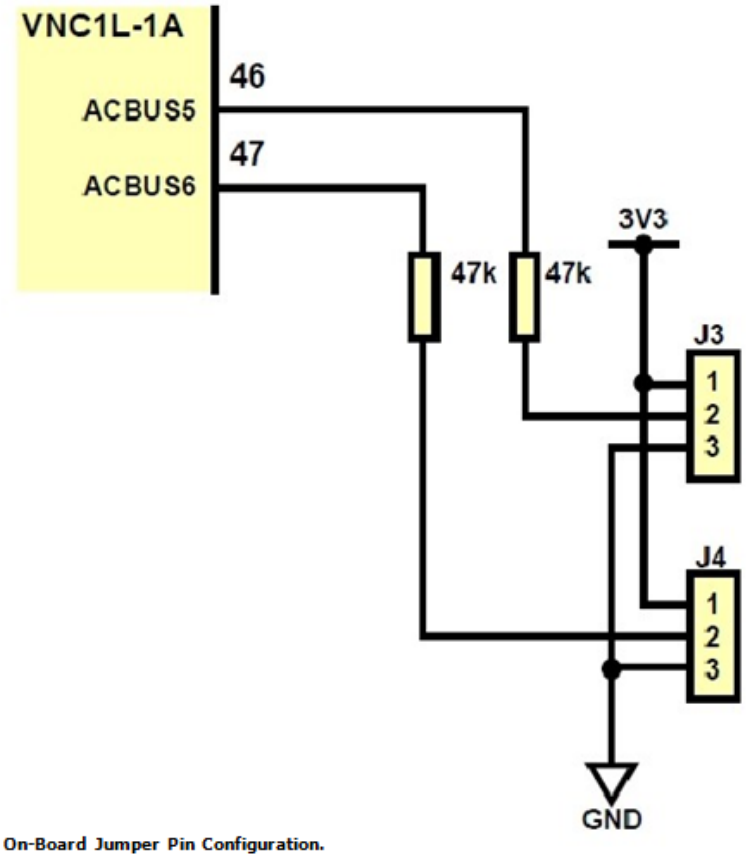


Figure 3.2 – VDIP1 On-Board Jumper Pin Configuration.

ACBUS6 (VNC1L pin 47)	*ACBUS5 (VNC1L pin 46)*	*I/O Mode*
Pull-Up	Pull-Up	Serial UART
Pull-Up	Pull-Down	SPI
Pull-Down	Pull-Up	Parallel FIFO
Pull-Down	Pull-Down	Serial UART

Table 3.2 - VDIP1 Port Selection Jumper Pins

3.4 Default Interface I/O Pin Configuration

The VNC1L device is pre-programmed with default settings for the I/O pins however they can be easily changed to suit a designers needs. The default interface I/O pin configuration of the VNC1L device are shown in **Table 3.3**

Pin No.	Name	Pin Name on PCB	Type	Description	Data and Control Bus Configuration Options			
					UART Interface	Parallel FIFO Interface	SPI Slave Interface	I/O Port
6	ADBUS0	AD0	I/O	5V safe bidirectional data / control bus, AD bit 0	TXD	D0	SCLK	PortAD0
8	ADBUS1	AD1	I/O	5V safe bidirectional data / control bus, AD bit 1	RXD	D1	SDI	PortAD1
9	ADBUS2	AD2	I/O	5V safe bidirectional data / control bus, AD bit 2	RTS#	D2	SDO	PortAD2
10	ADBUS3	AD3	I/O	5V safe bidirectional data / control bus, AD bit 3	CTS#	D3	CS	PortAD3
11	ADBUS4	AD4	I/O	5V safe bidirectional data / control bus, AD bit 4	DTR#	D4		PortAD4
12	ADBUS5	AD5	I/O	5V safe bidirectional data / control bus, AD bit 5	DSR#	D5		PortAD5
13	ADBUS6	AD6	I/O	5V safe bidirectional data / control bus, AD bit 6	DCD#	D6		PortAD6
14	ADBUS7	AD7	I/O	5V safe bidirectional data / control bus, AD bit 7	RI#	D7		PortAD7
15	ACBUS0	AC0	I/O	5V safe bidirectional data / control bus, AC bit 0	TXDEN#	RXF#		PortAC0
16	ACBUS1	AC1	I/O	5V safe bidirectional data / control bus, AC bit 1		TXE#		PortAC1
17	ACBUS2	AC2	I/O	5V safe bidirectional data / control bus, AC bit 2		RD#		PortAC2
19	ACBUS3	AC3	I/O	5V safe bidirectional data / control bus, AC bit 3		WR		PortAC3
20	ACBUS4	AC4	I/O	5V safe bidirectional data / control bus, AC bit 4				PortAC4

Table 3.3 - Default Interface I/O Pin Configuration

3.5 Signal Descriptions - UART Interface

The UART interface I/O pin description of the VNC1L device are shown in **Table 3.4**

Pin No.	Name	Type	Description
6	TXD	Output	Transmit asynchronous data output
8	RXD	Input	Receive asynchronous data input
9	RTS#	Output	Request To Send Control Output / Handshake signal
10	CTS#	Input	Clear To Send Control Input / Handshake signal
11	DTR#	Output	Data Terminal Ready Control Output / Handshake signal
12	DSR#	Input	Data Set Ready Control Input / Handshake signal
13	DCD#	Input	Data Carrier Detect Control Input
14	RI#	Input	Ring Indicator Contro l Input. When the RemoteakeW up option is enabled in the EEPROM, taking RI# low can be used to resume the PC USB Host controller from suspend
15	TXDEN#	Input	Enable Transmit Data for RS485 designs

Table 3.4 - Default I/O Pin Configuration – UART Interface

3.6 Signal Descriptions – Serial Peripheral Interface (SPI)

The SPI I/O pin description of the VNC1L device are shown in **Table 3.5**

Pins No	*Name*	*Type*	*Description*
6	SCLK	Input	SPI Clock input, 12MHz maximum.
8	SDI	Input	SPI Serial Data Input
9	SDO	Output	SPI Serial Data Output
10CSInputSPI Chip Select Input			

Table 3.5 - Data and Control Bus Signal Mode Options – SPI Slave Interface

3.6.1 SPI Slave Data Read Cycle

When in SPI mode, the timing of a read operation is shown in Figure 3.3

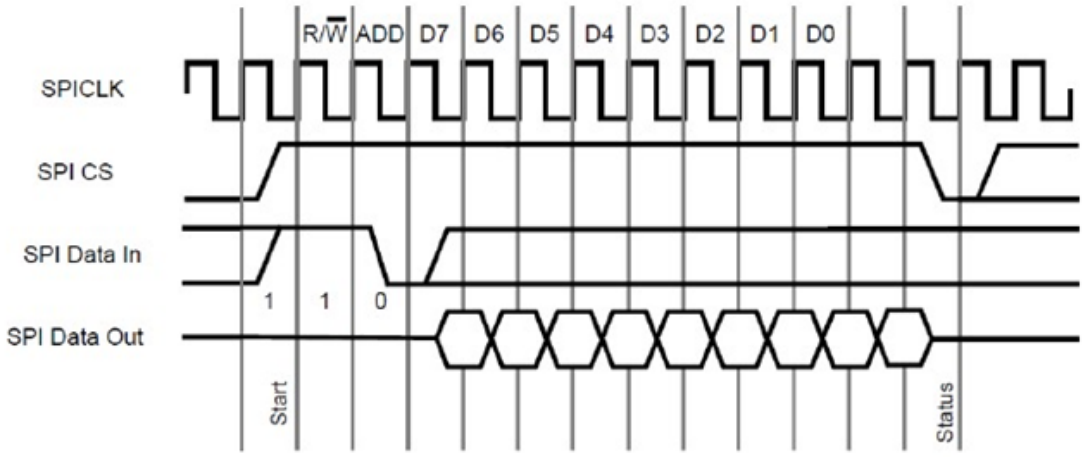


Figure 3.3 – SPI Slave Data Read Cycle.

From Start - SPI CS must be held high for the entire read cycle, and must be taken low for at least one clock period after t he read is co mpleted. The first bit on SPI Data In is the R/W bit - inputting a '1' here a llows data to be read fro m the chip. The next bit is the address bit, ADD, which is used to indicate whether the data register ('0') or the status register ('1') is read from. During the SPI read cycle a byte of data will start being output on SPI Data Out on the next clock cycle after t he address bit, MSBAfterfirst,t he data has been clocked out of the chip, t he status of SPI Data. Out should be checked to see if the data read is new data. A '0' level here on SPI Data Out means that the data read is new data. A '1' indicates that the data read is old data, and the read cycle should be repeated to get new data. Remember that CS must be held low for at least one clock period before being taken high again to continue with the next read or write cycle.

3.6.2 SPI Slave Data Write Cycle

When in SPI mode, the timing of a write operation is shown in

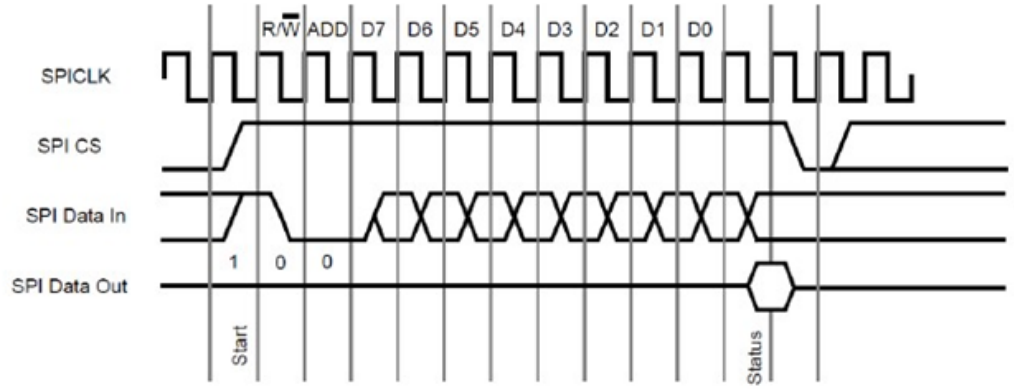


Figure 3.4 – SPI Slave Data Write Cycle.

From Start - SPI CS must be held high for the entire write cycle, and must be taken low for at least one clock period after t he write is co mpletedThe. first bit on SPI Data In is the R/W bit - inputting a '0' here a llows data to be written to the chip. The next bit is the address bit, ADD, which is used to indicate whether the data register ('0') or the status register ('1') is written to. During the SPI write cycle a byte of data can be input to SPI Data In on the next clock cycle after t he address bit, MSBAfterfirst.t he data has been clocked in to the chip, t he status of SPI Data Out should be checked to see if the data read was accepted. A '0' level on SPI Data Out means that the data write was accepted. A '1' indicates that the internal buffer is full, and the write should be repeated. Remember that CS must be held low for at least one clock period before being taken high again to continue with the next read or write cycle.

3.7.2 Timing Diagram - Parallel FIFO Write Transaction

When in parallel FIFO interface mode, the timing of a write operation is shown in **Figure 3.7** and **Table 3.10**

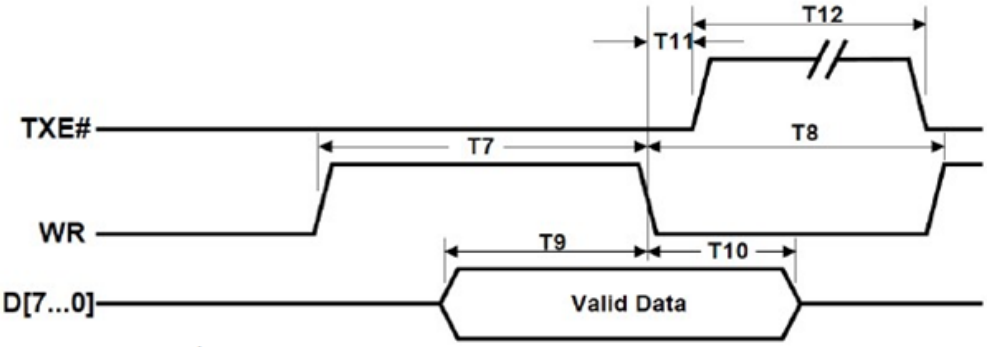


Figure 3.7 - FIFO Write Cycle.

Time	Description	Min	Max	Unit
T7	WR Active Pulse Width	50	-	ns
T8	WR to WR Pre-Charge Time	50	-	ns
T9	WR Active to Valid Data	20	-	ns
T10	Data Hold Time from WR Inactive*	0	-	ns
T11	WR Inactive to TXE#	5	25	ns
T12	TXE# Inactive After WR Cycle	80	-	ns

Table 3.10 - FIFO Write Cycle Timing

6 External circuit Configuration

6.1 Adding a second USB Port

The external circuit configuration for adding second USB host port, with the USB activity LED, is shown below in Figure 6.1

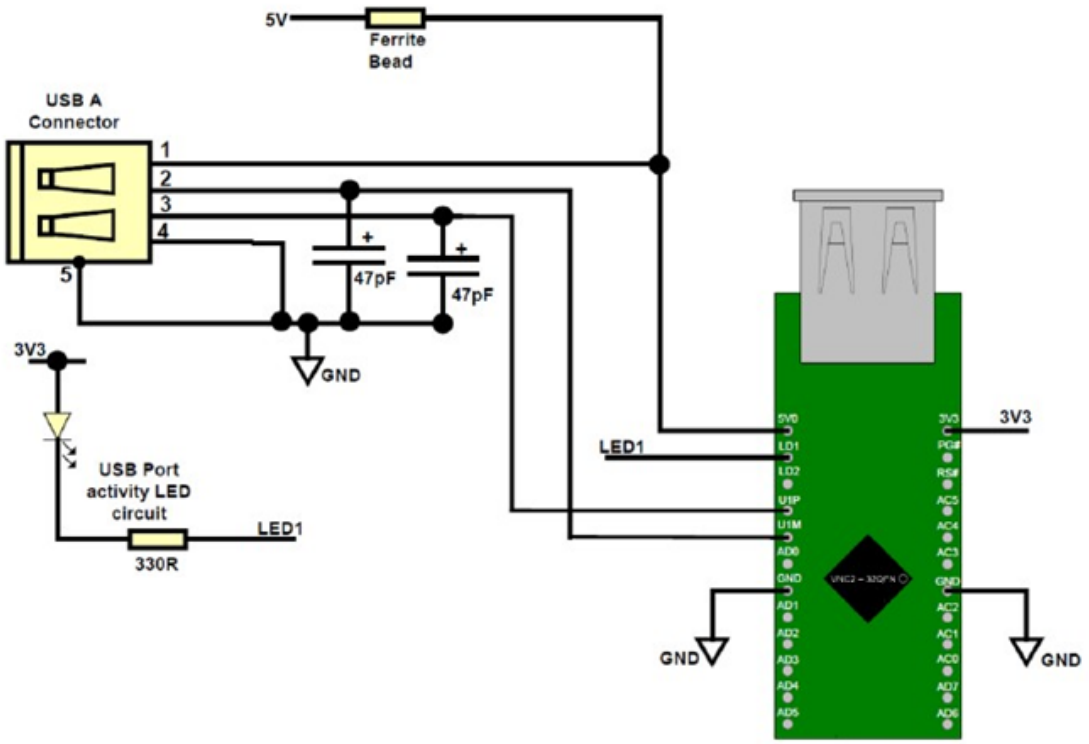


Figure 6.1 Additional USB Port Configuration

7 Schematic Diagram

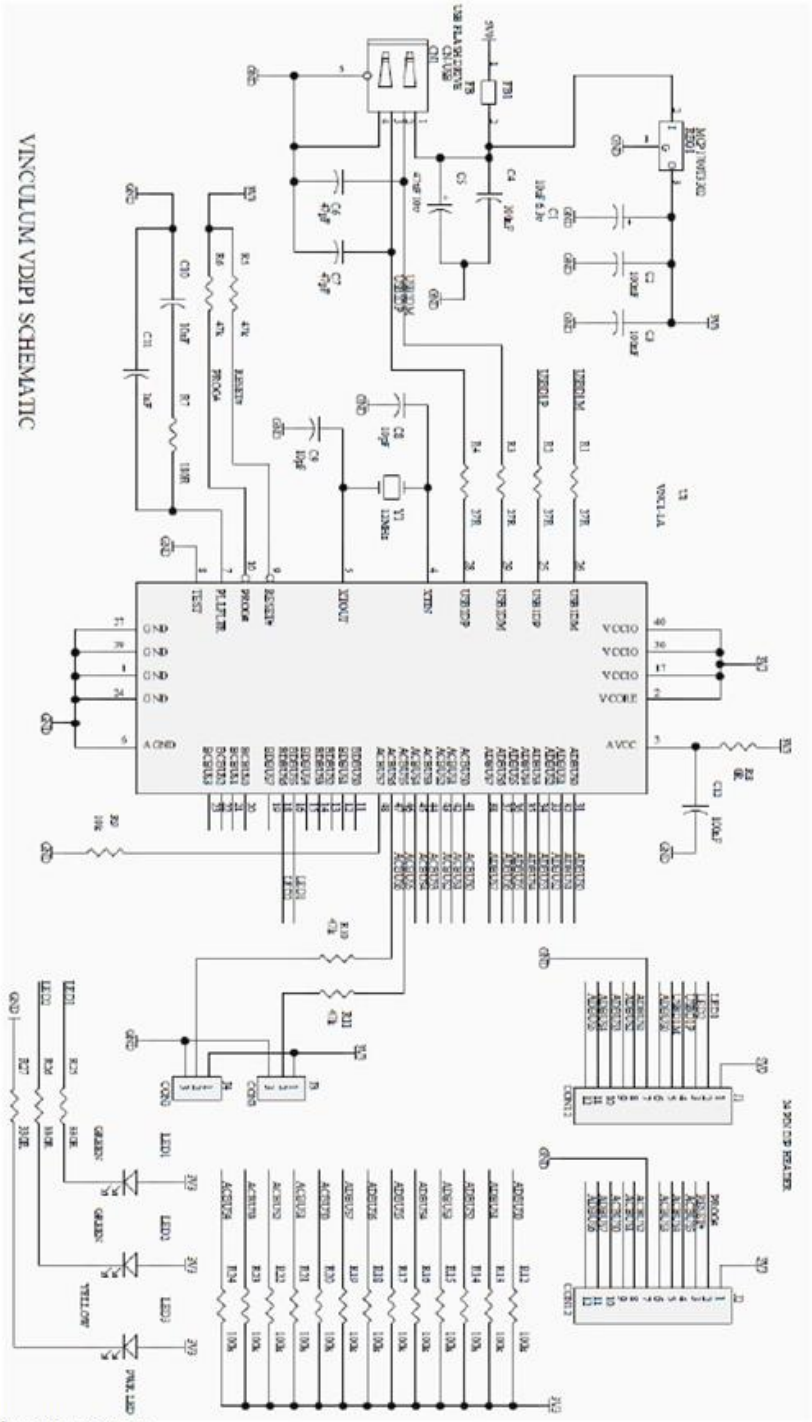


Figure 7.1 - Schematic Diagram

ANALOG DEVICES

Small, Low Power, 3-Axis ±3 g iMEMS® Accelerometer

ADXL330

FEATURES

3-axis sensing
Small, low-profile package
4 mm × 4 mm × 1.45 mm LFCSP
Low power
180 µA at V_S = 1.8 V (typical)
Single-supply operation
1.8 V to 3.6 V
10,000 g shock survival
Excellent temperature stability
BW adjustment with a single capacitor per axis
RoHS/WEEE lead-free compliant

APPLICATIONS

Cost-sensitive, low power, motion- and tilt-sensing applications
Mobile devices
Gaming systems
Disk drive protection
Image stabilization
Sports and health devices

GENERAL DESCRIPTION

The ADXL330 is a small, thin, low power, complete 3-axis accelerometer with signal conditioned voltage outputs, all on a single monolithic IC. The product measures acceleration with a minimum full-scale range of ±3 g. It can measure the static acceleration of gravity in tilt-sensing applications, as well as dynamic acceleration resulting from motion, shock, or vibration.

The user selects the bandwidth of the accelerometer using the C_X, C_Y, and C_Z capacitors at the X_{OUT}, Y_{OUT}, and Z_{OUT} pins. Bandwidths can be selected to suit the application, with a range of 0.5 Hz to 1600 Hz for X and Y axes, and a range of 0.5 Hz to 550 Hz for the Z axis.

The ADXL330 is available in a small, low profile, 4 mm × 4 mm × 1.45 mm, 16-lead, plastic lead frame chip scale package (LFCSP_LQ).

FUNCTIONAL BLOCK DIAGRAM

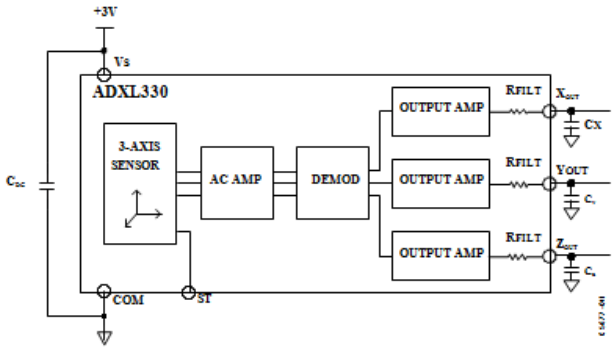


Figure 1.

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

SPECIFICATIONS

T_A = 25°C, V_S = 3 V, C_X = C_Y = C_Z = 0.1 µF, acceleration = 0 g, unless otherwise noted. All minimum and maximum specifications are guaranteed. Typical specifications are not guaranteed.

Table 1.

Parameter	Conditions	Min	Typ	Max	Unit
SENSOR INPUT	Each axis				
Measurement Range		±3	±3.6		g
Nonlinearity	% of full scale		±0.3		%
Package Alignment Error			±1		Degrees
Interaxis Alignment Error			±0.1		Degrees
Cross Axis Sensitivity[1]			±1		%
SENSITIVITY (RATIOMETRIC)[2]	Each axis				
Sensitivity at X_{OUT}, Y_{OUT}, Z_{OUT}	V_S = 3 V	270	300	330	mV/g
Sensitivity Change Due to Temperature[3]	V_S = 3 V		±0.015		%/°C
ZERO g BIAS LEVEL (RATIOMETRIC)	Each axis				
0 g Voltage at X_{OUT}, Y_{OUT}, Z_{OUT}	V_S = 3 V	1.2	1.5	1.8	V
0 g Offset vs. Temperature			±1		mg/°C
NOISE PERFORMANCE					
Noise Density X_{OUT}, Y_{OUT}			280		µg/√Hz rms
Noise Density Z_{OUT}			350		µg/√Hz rms
FREQUENCY RESPONSE[4]					
Bandwidth X_{OUT}, Y_{OUT}[5]	No external filter		1600		Hz
Bandwidth Z_{OUT}[6]	No external filter		550		Hz
R_{FILT} Tolerance			32 ± 15%		kΩ
Sensor Resonant Frequency			5.5		kHz
SELF TEST[7]					
Logic Input Low			+0.6		V
Logic Input High			+2.4		V
ST Actuation Current			+60		µA
Output Change at X_{OUT}	Self test 0 to 1		−150		mV
Output Change at Y_{OUT}	Self test 0 to 1		+150		mV
Output Change at Z_{OUT}	Self test 0 to 1		−60		mV
OUTPUT AMPLIFIER					
Output Swing Low	No load		0.1		V
Output Swing High	No load		2.8		V
POWER SUPPLY					
Operating Voltage Range		1.8		3.6	V
Supply Current	V_S = 3 V		320		µA
Turn-On Time[8]	No external filter		1		ms
TEMPERATURE					
Operating Temperature Range		−25		+70	°C

[1] Defined as coupling between any two axes.
[2] Sensitivity is essentially ratiometric to V_S.
[3] Defined as the output change from ambient-to-maximum temperature or ambient-to-minimum temperature.
[4] Actual frequency response controlled by user-supplied external filter capacitors (C_X, C_Y, C_Z).
[5] Bandwidth with external capacitors = 1/(2 × π × 32 kΩ × C). For C_X, C_Y = 0.003 µF, bandwidth = 1.6 kHz. For C_X = 0.01 µF, bandwidth = 500 Hz. For C_X, C_Y, C_Z = 10 µF, bandwidth = 0.5 Hz.
[6] Self-test response changes cubically with V_S.
[7] Turn-on time is dependent on C_X, C_Y, C_Z and is approximately 160 × C_X or C_Y or C_Z + 1 ms, where C_X, C_Y, C_Z are in µF.

ABSOLUTE MAXIMUM RATINGS

Table 2.

Parameter	Rating
Acceleration (Any Axis, Unpowered)	10,000 g
Acceleration (Any Axis, Powered)	10,000 g
V_s	−0.3 V to +7.0 V
All Other Pins	(COM − 0.3 V) to (V_S + 0.3 V)
Output Short-Circuit Duration (Any Pin to Common)	Indefinite
Temperature Range (Powered)	−55°C to +125°C
Temperature Range (Storage)	−65°C to +150°C

Stresses above those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only; functional operation of the device at these or any other conditions above those indicated in the operational section of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

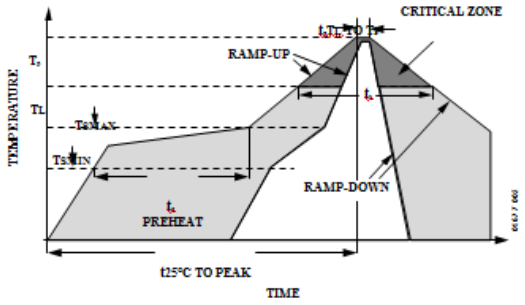


Figure 2. Recommended Soldering Profile

Table 3. Recommended Soldering Profile		
Profile Feature	**Sn63/Pb37**	**Pb-Free**
Average Ramp Rate (T_L to T_P)	3°C/s max	3°C/s max
Preheat		
Minimum Temperature (T_{SMIN})	100°C	150°C
Maximum Temperature (T_{SMAX})	150°C	200°C
Time (T_{SMIN} to T_{SMAX}), t_S	60 s to 120 s	60 s to 180 s
T_{SMAX} to T_L		
Ramp-Up Rate	3°C/s max	3°C/s max
Time Maintained Above Liquidous (T_L)		
Liquidous Temperature (T_L)	183°C	217°C
Time (t_L)	60 s to 150 s	60 s to 150 s
Peak Temperature (T_P)	240°C + 0°C/−5°C	260°C + 0°C/−5°C
Time within 5°C of Actual Peak Temperature (t_P)	10 s to 30 s	20 s to 40 s
Ramp-Down Rate	6°C/s max	6°C/s max
Time 25°C to Peak Temperature	6 minutes max	8 minutes max

ESD CAUTION

ESD (electrostatic discharge) sensitive device. Electrostatic charges as high as 4000 V readily accumulate on the human body and test equipment and can discharge without detection. Although this product features proprietary ESD protection circuitry, permanent damage may occur on devices subjected to high energy electrostatic discharges. Therefore, proper ESD precautions are recommended to avoid performance degradation or loss of functionality.

PIN CONFIGURATION AND FUNCTION DESCRIPTIONS

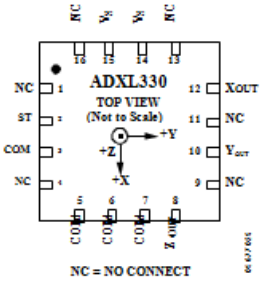


Figure 3. Pin Configuration

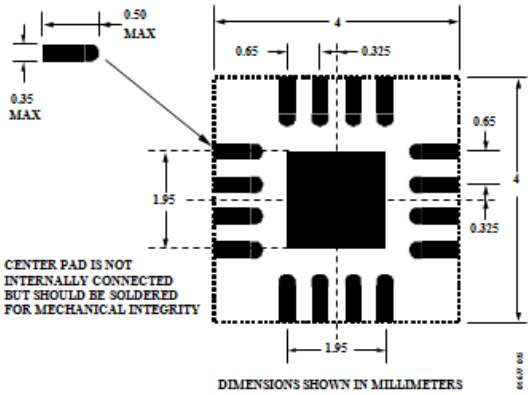


Figure 4. Recommended PCB Layout

Table 4. Pin Function Descriptions

Pin No.	Mnemonic	Description
1	NC	No Connect
2	ST	Self Test
3	COM	Common
4	NC	No Connect
5	COM	Common
6	COM	Common
7	COM	Common
8	Z_{OUT}	Z Channel Output
9	NC	No Connect
10	Y_{OUT}	Y Channel Output
11	NC	No Connect
12	X_{OUT}	X Channel Output
13	NC	No Connect
14	V_S	Supply Voltage (1.8 V to 3.6 V)
15	V_S	Supply Voltage (1.8 V to 3.6 V)
16	NC	No Connect

THEORY OF OPERATION

The ADXL330 is a complete 3-axis acceleration measurement system on a single monolithic IC. The ADXL330 has a measurement range of ±3 g minimum. It contains a polysilicon surface micromachined sensor and signal conditioning circuitry to implement an open-loop acceleration measurement architecture. The output signals are analog voltages that are proportional to acceleration. The accelerometer can measure the static acceleration of gravity in tilt sensing applications as well as dynamic acceleration resulting from motion, shock, or vibration.

The sensor is a polysilicon surface micromachined structure built on top of a silicon wafer. Polysilicon springs suspend the structure over the surface of the wafer and provide a resistance against acceleration forces. Deflection of the structure is measured using a differential capacitor that consists of independent fixed plates and plates attached to the moving mass. The fixed plates are driven by 180° out-of-phase square waves. Acceleration deflects the moving mass and unbalances the differential capacitor resulting in a sensor output whose amplitude is proportional to acceleration. Phase-sensitive demodulation techniques are then used to determine the magnitude and direction of the acceleration.

The demodulator output is amplified and brought off-chip through a 32 kΩ resistor. The user then sets the signal bandwidth of the device by adding a capacitor. This filtering improves measurement resolution and helps prevent aliasing.

MECHANICAL SENSOR

The ADXL330 uses a single structure for sensing the X, Y, and Z axes. As a result, the three axes sense directions are highly orthogonal with little cross axis sensitivity. Mechanical misalignment of the sensor die to the package is the chief source of cross axis sensitivity. Mechanical misalignment can, of course, be calibrated out at the system level.

PERFORMANCE

Rather than using additional temperature compensation circuitry, innovative design techniques ensure high performance is built-in to the ADXL330. As a result, there is neither quantization error nor nonmonotonic behavior, and temperature hysteresis is very low (typically less than 3 mg over the −25°C to +70°C temperature range).

Figure 14, Figure 15, and Figure 16 show the zero g output performance of eight parts (X-, Y-, and Z-axis) soldered to a PCB over a −25°C to +70°C temperature range.

Figure 26, Figure 27, and Figure 28 demonstrate the typical sensitivity shift over temperature for supply voltages of 3 V. This is typically better than ±1% over the −25°C to +70°C temperature range.

APPLICATIONS

POWER SUPPLY DECOUPLING

For most applications, a single 0.1 µF capacitor, C_{DC}, placed close to the ADXL330 supply pins adequately decouples the accelerometer from noise on the power supply. However, in applications where noise is present at the 50 kHz internal clock frequency (or any harmonic thereof), additional care in power supply bypassing is required as this noise can cause errors in acceleration measurement. If additional decoupling is needed, a 100 Ω (or smaller) resistor or ferrite bead can be inserted in the supply line. Additionally, a larger bulk bypass capacitor (1 µF or greater) can be added in parallel to C_{DC}. Ensure that the connection from the ADXL330 ground to the power supply ground is low impedance because noise transmitted through ground has a similar effect as noise transmitted through V_S.

SETTING THE BANDWIDTH USING C_X, C_Y, AND C_Z

The ADXL330 has provisions for band limiting the X_{OUT}, Y_{OUT}, and Z_{OUT} pins. Capacitors must be added at these pins to implement low-pass filtering for antialiasing and noise reduction. The equation for the 3 dB bandwidth is

$$F_{-3\ dB} = 1/(2\pi(32\ k\Omega) \times C_{(X,\ Y,\ Z)})$$

or more simply

$$F_{-3\ dB} = 5\ \mu F/C_{(X,\ Y,\ Z)}$$

The tolerance of the internal resistor (R_{FILT}) typically varies as much as ±15% of its nominal value (32 kΩ), and the bandwidth varies accordingly. A minimum capacitance of 0.0047 µF for C_X, C_Y, and C_Z is recommended in all cases.

Table 5. Filter Capacitor Selection, C_X, C_Y, and C_Z

Bandwidth (Hz)	Capacitor (µF)
1	4.7
10	0.47
50	0.10
100	0.05
200	0.027
500	0.01

SELF TEST

The ST pin controls the self test feature. When this pin is set to V_S, an electrostatic force is exerted on the accelerometer beam. The resulting movement of the beam allows the user to test if the accelerometer is functional. The typical change in output is −500 mg (corresponding to −150 mV) in the X-axis, 500 mg (or 150 mV) on the Y-axis, and −200 mg (or −60 mV) on the Z-axis. This ST pin may be left open circuit or connected to common (COM) in normal use.

Never expose the ST pin to voltages greater than V_S + 0.3 V. If this cannot be guaranteed due to the system design (for instance, if there are multiple supply voltages), then a low V_F clamping diode between ST and V_S is recommended.

DESIGN TRADE-OFFS FOR SELECTING FILTER CHARACTERISTICS: THE NOISE/BW TRADE-OFF

The selected accelerometer bandwidth ultimately determines the measurement resolution (smallest detectable acceleration). Filtering can be used to lower the noise floor to improve the resolution of the accelerometer. Resolution is dependent on the analog filter bandwidth at X_{OUT}, Y_{OUT}, and Z_{OUT}.

The output of the ADXL330 has a typical bandwidth of greater than 500 Hz. The user must filter the signal at this point to limit aliasing errors. The analog bandwidth must be no more than half the analog-to-digital sampling frequency to minimize aliasing. The analog bandwidth can be further decreased to reduce noise and improve resolution.

The ADXL330 noise has the characteristics of white Gaussian noise, which contributes equally at all frequencies and is described in terms of µg/√Hz (the noise is proportional to the square root of the accelerometer bandwidth). The user should limit bandwidth to the lowest frequency needed by the application to maximize the resolution and dynamic range of the accelerometer.

With the single-pole, roll-off characteristic, the typical noise of the ADXL330 is determined by

$$rms\ Noise = Noise\ Density \times (\sqrt{BW} \times 1.6)$$

Often, the peak value of the noise is desired. Peak-to-peak noise can only be estimated by statistical methods. Table 6 is useful for estimating the probabilities of exceeding various peak values, given the rms value.

Table 6. Estimation of Peak-to-Peak Noise

Peak-to-Peak Value	% of Time that Noise Exceeds Nominal Peak-to-Peak Value
2 × rms	32
4 × rms	4.6
6 × rms	0.27
8 × rms	0.006

USE WITH OPERATING VOLTAGES OTHER THAN 3 V

The ADXL330 is tested and specified at V_S = 3 V; however, it can be powered with V_S as low as 1.8 V or as high as 3.6 V. Note that some performance parameters change as the supply voltage is varied.

The ADXL330 output is ratiometric, therefore, the output sensitivity (or scale factor) varies proportionally to the supply voltage. At V_S = 3.6 V, the output sensitivity is typically 360 mV/g. At V_S = 2 V, the output sensitivity is typically 195 mV/g.

The zero g bias output is also ratiometric, so the zero g output is nominally equal to $V_S/2$ at all supply voltages.

The output noise is not ratiometric but is absolute in volts; therefore, the noise density decreases as the supply voltage increases. This is because the scale factor (mV/g) increases while the noise voltage remains constant. At V_S = 3.6 V, the X- and Y-axis noise density is typically 230 µg/√Hz, while at V_S = 2 V, the X- and Y-axis noise density is typically 350 µg/√Hz.

Self test response in g is roughly proportional to the square of the supply voltage. However, when ratiometricity of sensitivity is factored in with supply voltage, the self test response in volts is roughly proportional to the cube of the supply voltage. For example, at V_S = 3.6 V, the self test response for the ADXL330 is approximately −275 mV for the X-axis, +275 mV for the Y-axis, and −100 mV for the Z-axis.

At V_S = 2 V, the self test response is approximately −60 mV for the X-axis, +60 mV for the Y-axis, and −25 mV for the Z-axis.

The supply current decreases as the supply voltage decreases. Typical current consumption at V_S = 3.6 V is 375 µA, and typical current consumption at V_S = 2 V is 200 µA.

AXES OF ACCELERATION SENSITIVITY

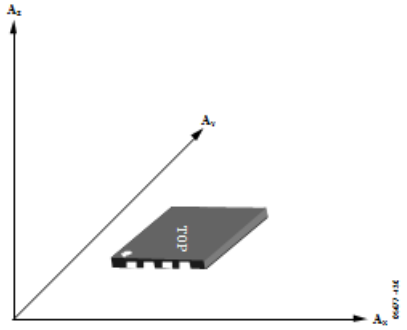


Figure 31. Axes of Acceleration Sensitivity, Corresponding Output Voltage Increases When Accelerated Along the Sensitive Axis

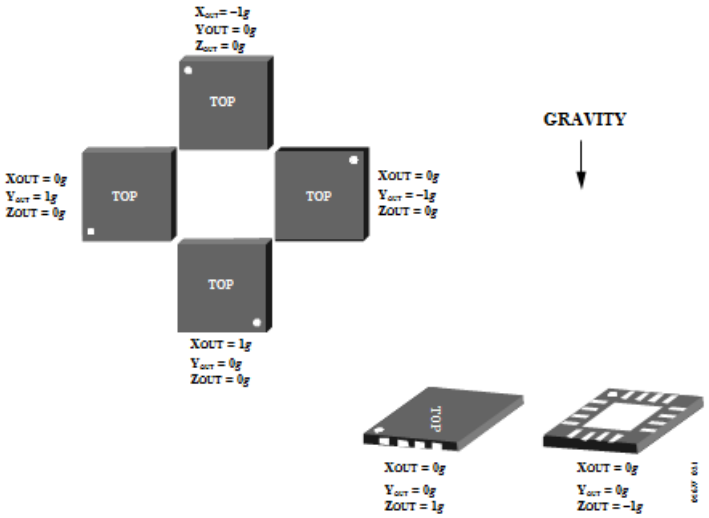


Figure 32. Output Response vs. Orientation to Gravity

500mA Low Dropout Positive Voltage Regulator

General Description

The RT9163 is a positive low dropout regulator designed for applications requiring low dropout performance at full rated current. The device is available in fixed output voltage of 3.3V, 3.5V, and 5.0V. The RT9163 provides excellent regulation over line, load, and temperature variations.

The other features include low dropout performance at a maximum of 1.4V at 500mA, fast transient response, internal current limiting, and thermal shutdown protection of the output devices. The RT9163 is a three-terminal regulator available in surface mount SOT-89, SOT-223, and TO-252 packages.

Applications

- 5V to 3.3V Linear Regulator
- Low Voltage Microcontroller, DSP... etc Power Supply
- Linear Regulator for LAN Card and CD-ROM

Ordering Information

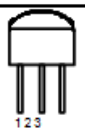
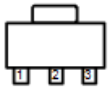
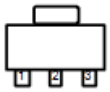
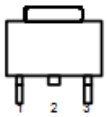
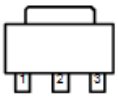
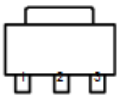
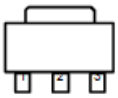
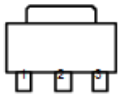
RT9163-□□ □ □

- Package type
  - ZL : TO-92
  - X : SOT-89
  - XL : SOT-89, L Type
  - L : TO-252
  - G : SOT-223 Type I
  - GL : SOT-223 Type II
  - GT : SOT-223 Type III
  - GF : SOT-223 Type IV
- Operating temperature range
  - C: Commercial standard
- Output voltage
  - 33 : 3.3V
  - 35 : 3.5V
  - 50 : 5.0V

Features

- **Low Dropout, Maximum 1.4V at 500mA**
- **Fast Transient Response**
- **±2% Total Output Regulation**
- **0.4% Line Regulation**
- **0.4% Load Regulation**
- **TO-92, SOT-89, SOT-223, and TO-252 Packages**

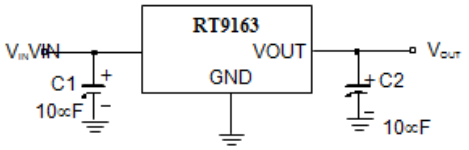
Pin Configurations

Part Number	Pin Configurations
RT9163-□□CZL (Plastic TO-92)	 TOP VIEW 1. VIN 2. GND 3. VOUT
RT9163-□□CX (Plastic SOT-89)	 TOP VIEW 1. VOUT 2. GND (TAB) 3. VIN
RT9163-□□CXL (Plastic SOT-89)	 TOP VIEW 1. GND 2. VIN (TAB) 3. VOUT
RT9163-□□CL (Plastic TO-252)	 TOP VIEW 1. VOUT 2. GND (TAB) 3. VIN
RT9163-□□CG (Plastic SOT-223) Type I	 TOP VIEW 1. VOUT 2. GND (TAB) 3. VIN
RT9163-□□CGL (Plastic SOT-223) Type II	 TOP VIEW 1. VIN 2. GND (TAB) 3. VOUT
RT9163-□□CGT (Plastic SOT-223) Type III	 TOP VIEW 1. GND 2. VOUT (TAB) 3. VIN
RT9163-□□CGF (Plastic SOT-223) Type IV	 TOP VIEW 1. GND 2. VIN (TAB) 3. VOUT

Marking Information

Part Number	Marking
RT9163-33CZL	RTAJ
RT9163-35CZL	RTAK
RT9163-50CZL	RTA1
RT9163-33CX	A8
RT9163-35CX	A9
RT9163-50CX	AT
RT9163-33CXL	CK
RT9163-35CXL	CL
RT9163-50CXL	CM

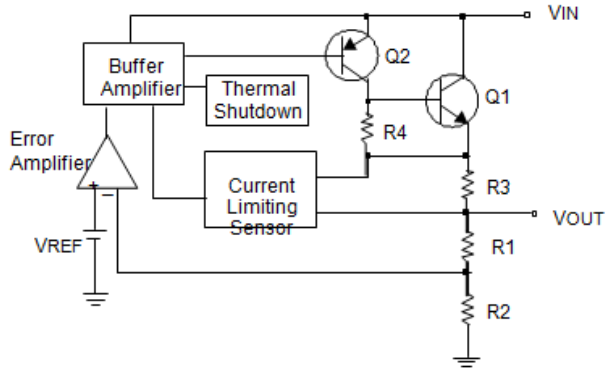
Typical Application Circuit



Pin Description

Pin Name	Pin Function
VOUT	Output Voltage
GND	Ground
VIN	Power Input

Function Block Diagram



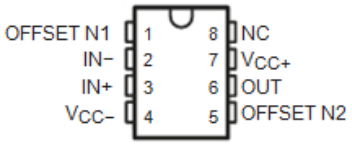
- D Low Power Consumption
- D Wide Common-Mode and Differential Voltage Ranges
- D Low Input Bias and Offset Currents
- D Output Short-Circuit Protection
- D Low Total Harmonic Distortion ...0.003% Typ
- D Low Noise
  V_n = 18 nV/√Hz Typ at f = 1 kHz
- D High Input Impedance . . . JFET Input Stage
- D Internal Frequency Compensation
- D Latch-Up-Free Operation
- D High Slew Rate . . . 13 V/μs Typ
- D Common-Mode Input Voltage Range Includes V_{CC+}

description/ordering information

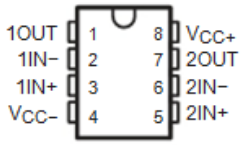
The JFET-input operational amplifiers in the TL07x series are similar to the TL08x series, with low input bias and offset currents and fast slew rate. The low harmonic distortion and low noise make the TL07x series ideally suited for high-fidelity and audio preamplifier applications. Each amplifier features JFET inputs (for high input impedance) coupled with bipolar output stages integrated on a single monolithic chip.

The C-suffix devices are characterized for operation from 0°C to 70°C. The I-suffix devices are characterized for operation from −40°C to 85°C. The M-suffix devices are characterized for operation over the full military temperature range of −55°C to 125°C.

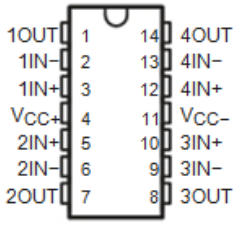
TL071, TL071A, TL071B
D, P, OR PS PACKAGE
(TOP VIEW)



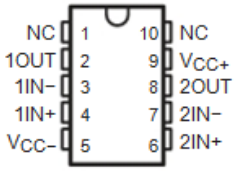
TL072, TL072A, TL072B
D, JG, P, PS, OR PW PACKAGE
(TOP VIEW)



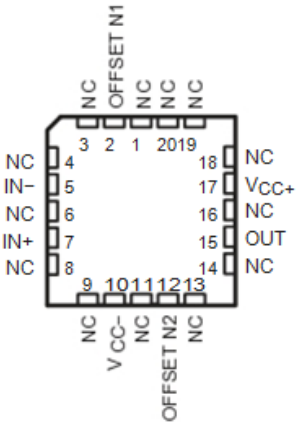
TL074A, TL074B
D, J, N, NS, OR PW PACKAGE
TL074 . . . D, J, N, NS, PW,
OR W PACKAGE
(TOP VIEW)



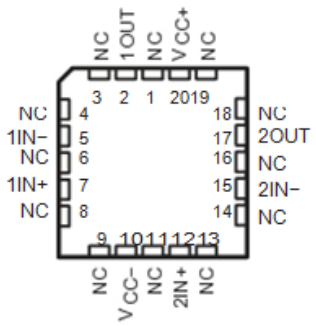
TL072
U PACKAGE
(TOP VIEW)



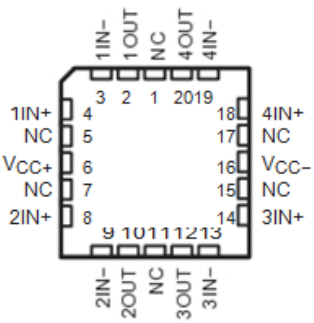
TL071
FK PACKAGE
(TOP VIEW)



TL072
FK PACKAGE
(TOP VIEW)



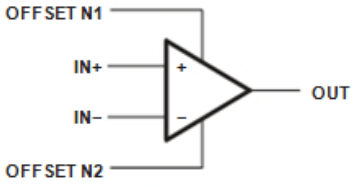
TL074
FK PACKAGE
(TOP VIEW)



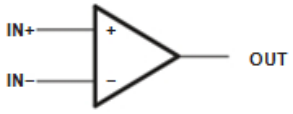
NC – No internal connection

symbols

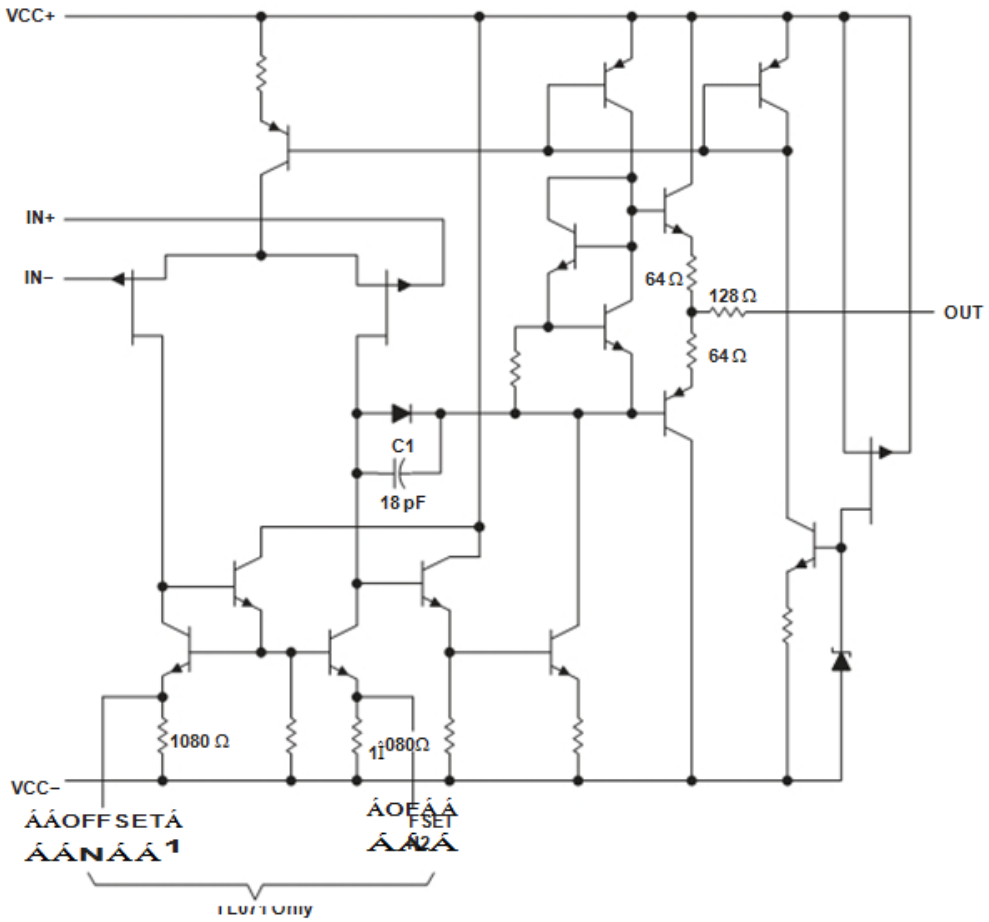
TL071



TL072 (each amplifier)
TL074 (each amplifier)



schematic (each amplifier)



All component values shown are nominal.

COMPONENT COUNT†			
COMPONENT TYPE	TL071	TL072	TL074
Resistors	11	22	44
Transistors	14	28	56
JFET	2	4	6
Diodes	1	2	4
Capacitors	1	2	4
epi-FET	1	2	4

† Includes bias and trim circuitry

absolute maximum ratings over operating free-air temperature range (unless otherwise noted)†

Supply voltage (see Note 1): V_{CC+}		18 V
V_{CC-}		−18 V
Differential input voltage, V_{ID} (see Note 2)		±30 V
Input voltage, V_I (see Notes 1 and 3)		±15 V
Duration of output short circuit (see Note 4)		Unlimited
Package thermal impedance, θ_{JA} (see Notes 5 and 6):	D package (8 pin)	97°C/W
	D package (14 pin)	86°C/W
	N package	80°C/W
	NS package	76°C/W
	P package	85°C/W
	PS package	95°C/W
	PW package (8 pin)	149°C/W
	PW package (14 pin)	113°C/W
	U package	185°C/W
Package thermal impedance, θ_{JC} (see Notes 7 and 8):	FK package	5.61°C/W
	J package	15.05°C/W
	JG package	14.5°C/W
	W package	14.65°C/W
Operating virtual junction temperature, T_J		150°C
Case temperature for 60 seconds: FK package		260°C
Lead temperature 1,6 mm (1/16 inch) from case for 10 seconds: J, JG, or W package		300°C
Storage temperature range, T_{stg}		−65°C to 150°C

† Stresses beyond those listed under "absolute maximum ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated under "recommended operating conditions" is not implied. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.

NOTES:
1. All voltage values, except differential voltages, are with respect to the midpoint between VCC+ and VCC−.
2. Differential voltages are at IN+, with respect to IN−.
3. The magnitude of the input voltage must never exceed the magnitude of the supply voltage or 15 V, whichever is less.
4. The output may be shorted to ground or to either supply. Temperature and/or supply voltages must be limited to ensure that the dissipation rating is not exceeded.
5. Maximum power dissipation is a function of $T_J(max)$, θ_{JA}, and T_A. The maximum allowable power dissipation at any allowable ambient temperature is $P_D = (T_J(max) - T_A)/\theta_{JA}$. Operating at the absolute maximum T_J of 150°C can affect reliability.
6. The package thermal impedance is calculated in accordance with JESD 51-7.
7. Maximum power dissipation is a function of $T_J(max)$, θ_{JC}, and T_C. The maximum allowable power dissipation at any allowable case temperature is $P_D = (T_J(max) - T_C)/\theta_{JC}$. Operating at the absolute maximum T_J of 150°C can affect reliability.
8. The package thermal impedance is calculated in accordance with MIL-STD-883.

electrical characteristics, $V_{CC\pm}$ = ±15 V (unless otherwise noted)

PARAMETER		TEST CONDITIONS†	TA‡	TL071M TL072M			TL074M			UNIT
				MIN	TYP	MAX	MIN	TYP	MAX	
VIO	Input offset voltage	VO = 0, RS = 50 Ω	25°C		3	6		3	9	mV
			Full range			9			15	
αV_{IO}	Temperature coefficient of input offset voltage	VO = 0, RS = 50 Ω	Full range		18			18		μV/°C
IIO	Input offset current	VO = 0	25°C		5	100		5	100	pA
			Full range			20			20	nA
IIB	Input bias current‡	VO = 0	25°C		65	200		65	200	pA
						50			50	nA
VICR	Common-mode input voltage range		25°C	±11	−12 to 15		±11	−12 to 15		V
VOM	Maximum peak output voltage swing	RL = 10 kΩ	25°C	±12	±13.5		±12	±13.5		V
		RL ≥ 10 kΩ	Full range	±12			±12			
		RL ≥ 2 kΩ		±10			±10			
AVD	Large-signal differential voltage amplification	VO = ±10 V, RL ≥ 2 kΩ TA = 25°C	25°C	35	200		35	200		V/mV
				15			15			
B1	Unity-gain bandwidth				3			3		MHz
ri	Input resistance	TA = 25°C			1012			1012		Ω
CMRR	Common-mode rejection ratio	VIC = VICRmin, VO = 0, RS = 50 Ω	25°C	80	86		80	86		dB
kSVR	Supply-voltage rejection ratio (ΔVCC±/ΔVIO)	VCC = ±9 V to ±15 V, VO = 0, RS = 50 Ω	25°C	80	86		80	86		dB
ICC	Supply current (each amplifier)	VO = 0, No load	25°C		1.4	2.5		1.4	2.5	mA
VO1/VO2	Crosstalk attenuation	AVD = 100	25°C		120			120		dB

† Input bias currents of an FET-input operational amplifier are normal junction reverse currents, which are temperature sensitive, as shown in Figure 4. Pulse techniques must be used that will maintain the junction temperature as close to the ambient temperature as possible.
‡ All characteristics are measured under open-loop conditions with zero common-mode voltage, unless otherwise specified. Full range is TA = −55°C to 125°C.

operating characteristics, $V_{CC\pm}$ = ±15 V, T_A = 25°C

PARAMETER		TEST CONDITIONS		TL07xM MIN	TL07xM TYP	TL07xM MAX	ALL OTHERS MIN	ALL OTHERS TYP	UNIT
SR	Slew rate at unity gain	V_I = 10 V, C_L = 100 pF,	R_L = 2 kΩ, See Figure 1		13		8	13	V/µs
t_r	Rise-time overshoot factor	V_I = 20 mV, C_L = 100 pF,	R_L = 2 kΩ, See Figure 1		0.1 20%			0.1 20%	µs
V_n	Equivalent input noise voltage	R_S = 20 Ω	f = 1 kHz		18			18	nV/√Hz
			f = 10 Hz to 10 kHz		4			4	µV
I_n	Equivalent input noise current	R_S = 20 Ω,	f = 1 kHz		0.01			0.01	pA/√Hz
THD	Total harmonic distortion	V_{Irms} = 6 V, R_L ≥ 2 kΩ, f = 1 kHz	A_{VD} = 1, R_S ≤ 1 kΩ,		0.003 %			0.003%	

PARAMETER MEASUREMENT INFORMATION

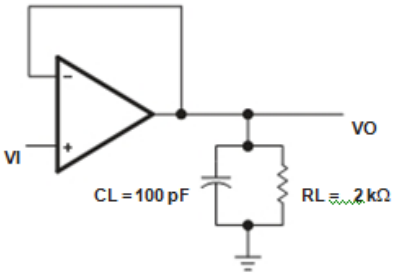


Figure 1. Unity-Gain Amplifier

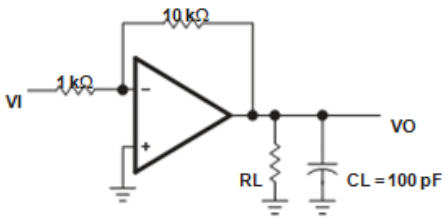


Figure 2. Gain-of-10 Inverting Amplifier

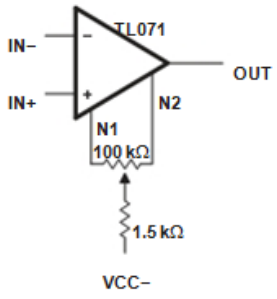


Figure 3. Input Offset-Voltage Null Circuit

Electrical Characteristics (KA7805A)

(Refer to the test circuits. 0°C < T_J < +125°C, I_O =1A, V_I = 10V, C_I=0.33µF, C_O=0.1µF, unless otherwise specified)

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Output Voltage	V_O	T_J =+25°C	4.9	5	5.1	V
		I_O = 5mA to 1A, P_O ≤ 15W V_I = 7.5V to 20V	4.8	5	5.2	
Line Regulation (Note1)	Regline	V_I = 7.5V to 25V I_O = 500mA	-	5	50	mV
		V_I = 8V to 12V	-	3	50	
		T_J =+25°C, V_I= 7.3V to 20V	-	5	50	
		T_J =+25°C, V_I= 8V to 12V	-	1.5	25	
Load Regulation (Note1)	Regload	T_J =+25°C I_O = 5mA to 1.5A	-	9	100	mV
		I_O = 5mA to 1A	-	9	100	
		I_O = 250mA to 750mA	-	4	50	
Quiescent Current	I_Q	T_J =+25°C	-	5.0	6.0	mA
Quiescent Current Change	ΔI_Q	I_O = 5mA to 1A	-	-	0.5	mA
		V_I = 8 V to 25V, I_O = 500mA	-	-	0.8	
		V_I = 7.5V to 20V, T_J =+25°C	-	-	0.8	
Output Voltage Drift	$\Delta V/\Delta T$	I_O = 5mA	-	-0.8	-	mV/°C
Output Noise Voltage	V_N	f = 10Hz to 100KHz T_A =+25°C	-	10	-	µV/Vo
Ripple Rejection	RR	f = 120Hz, I_O = 500mA V_I = 8V to 18V	-	68	-	dB
Dropout Voltage	V_{Drop}	I_O = 1A, T_J =+25°C	-	2	-	V
Output Resistance	r_O	f = 1KHz	-	17	-	mΩ
Short Circuit Current	I_{SC}	V_I= 35V, T_A =+25°C	-	250	-	mA
Peak Current	I_{PK}	T_J= +25°C	-	2.2	-	A

Note:

1. Load and line regulation are specified at constant junction temperature. Change in V_O due to heating effects must be take into account separately. Pulse testing with low duty is used.

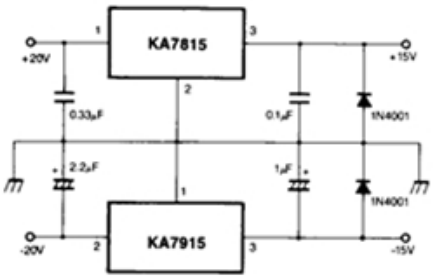


Figure 15. Split Power Supply (±15V-1A)

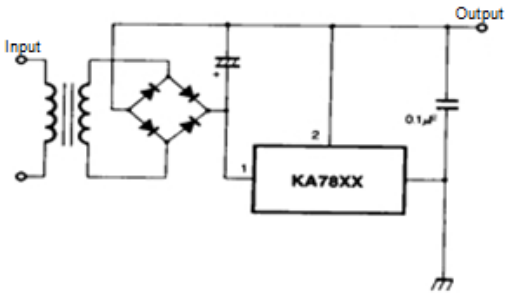


Figure 16. Negative Output Voltage Circuit

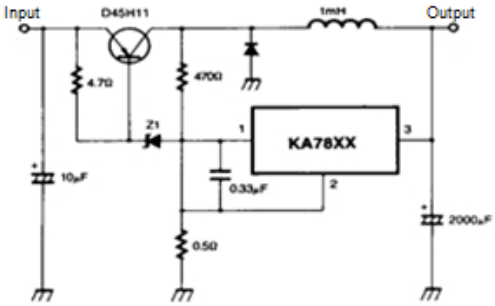


Figure 17. Switching Regulator

KA78XX/KA78XXA

3-Terminal 1A Positive Voltage Regulator

Features

- Output Current up to 1A
- Output Voltages of 5, 6, 8, 9, 10, 12, 15, 18, 24V
- Thermal Overload Protection
- Short Circuit Protection
- Output Transistor Safe Operating Area Protection

Description

The KA78XX/KA78XXA series of three-terminal positive regulator are available in the TO-220/D-PAK package and with several fixed output voltages, making them useful in a wide range of applications. Each type employs internal current limiting, thermal shut down and safe operating area protection, making it essentially indestructible. If adequate heat sinking is provided, they can deliver over 1A output current. Although designed primarily as fixed voltage regulators, these devices can be used with external components to obtain adjustable voltages and currents.

TO-220

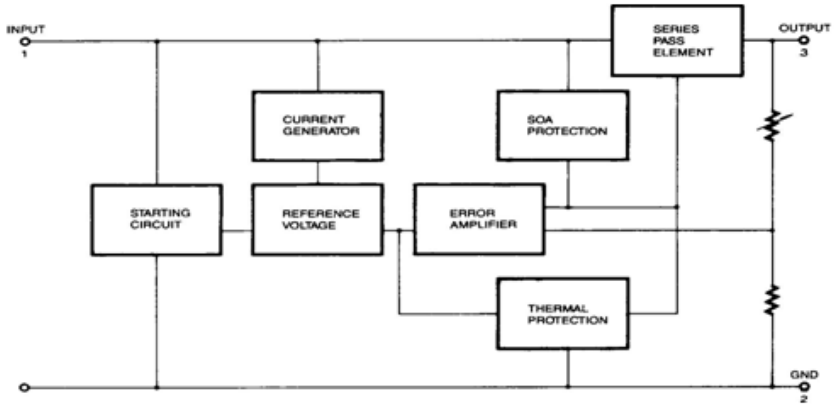
1

D-PAK

1

1. Input 2. GND 3. Output

Internal Block Digram



Rev. 1.0.0

Electrical Characteristics (KA7805A)

(Refer to the test circuits. 0°C < TJ < +125°C, Io =1A, V I = 10V, C I=0.33μF, C O=0.1μF, unless otherwise specified)

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Output Voltage	VO	TJ =+25°C	4.9	5	5.1	V
		IO = 5mA to 1A, PO ≤ 15W VI = 7.5V to 20V	4.8	5	5.2	
Line Regulation (Note1)	Regline	VI = 7.5V to 25V IO = 500mA	-	5	50	mV
		VI = 8V to 12V	-	3	50	
		TJ =+25°C, VI= 7.3V to 20V	-	5	50	
		VI= 8V to 12V	-	1.5	25	
Load Regulation (Note1)	Regload	TJ =+25°C IO = 5mA to 1.5A	-	9	100	mV
		IO = 5mA to 1A	-	9	100	
		IO = 250mA to 750mA	-	4	50	
Quiescent Current	IQ	TJ =+25°C	-	5.0	6.0	mA
Quiescent Current Change	–IQ	IO = 5mA to 1A	-	-	0.5	mA
		VI = 8 V to 25V, IO = 500mA	-	-	0.8	
		VI = 7.5V to 20V, TJ =+25°C	-	-	0.8	
Output Voltage Drift	–V/–T	Io = 5mA	-	-0.8	-	mV/°C
Output Noise Voltage	VN	f = 10Hz to 100KHz TA =+25°C	-	10	-	μV/Vo
Ripple Rejection	RR	f = 120Hz, IO = 500mA VI = 8V to 18V	-	68	-	dB
Dropout Voltage	VDrop	IO = 1A, TJ =+25°C	-	2	-	V
Output Resistance	rO	f = 1KHz	-	17	-	mΩ
Short Circuit Current	ISC	VI= 35V, TA =+25°C	-	250	-	mA
Peak Current	IPK	TJ= +25°C	-	2.2	-	A

Note:

1. Load and line regulation are specified at constant junction temperature. Change in VO due to heating effects must be taken into account separately. Pulse testing with low duty is used.

Typical Perfomance Characteristics

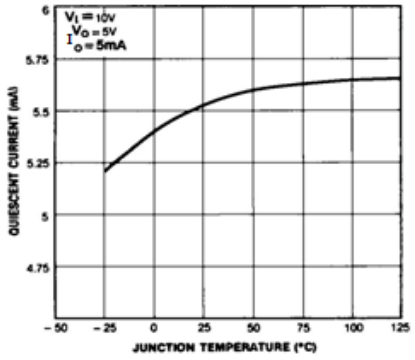


Figure 1. Quiescent Current

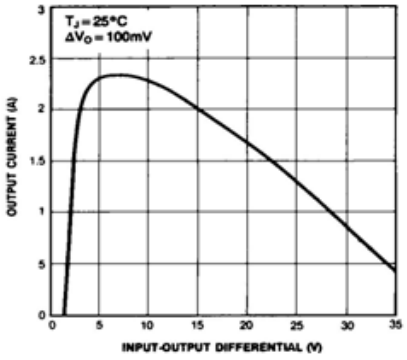


Figure 2. Peak Output Current

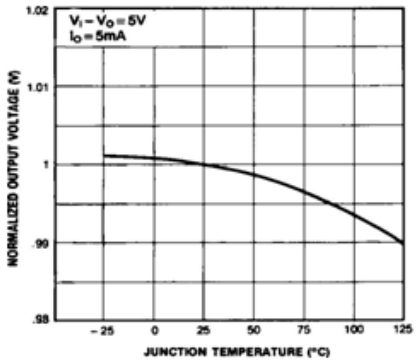


Figure 3. Output Voltage

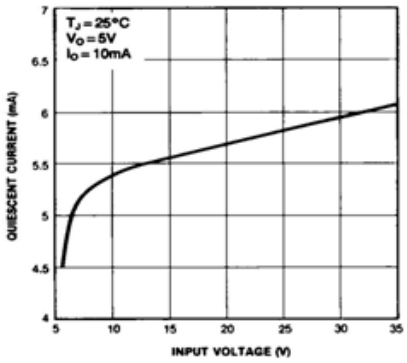


Figure 4. Quiescent Current

Typical Applications

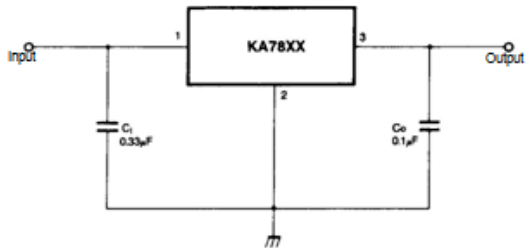


Figure 5. DC Parameters

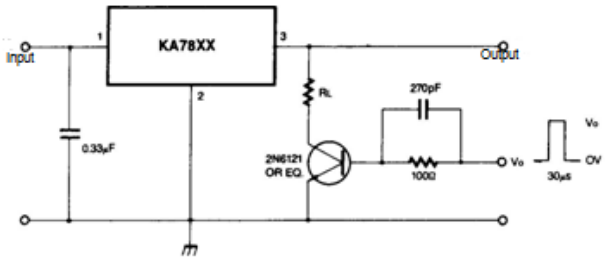


Figure 6. Load Regulation

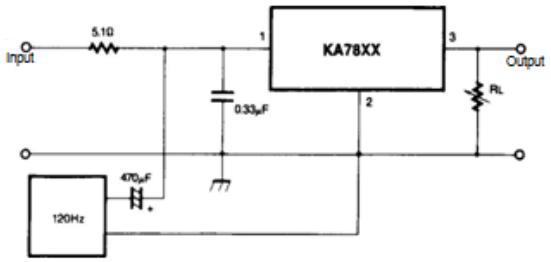


Figure 7. Ripple Rejection

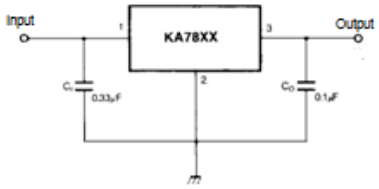
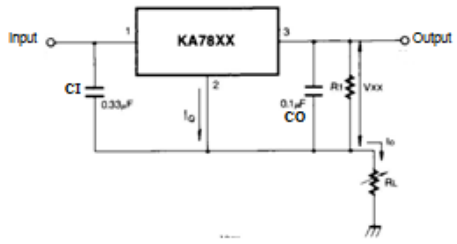


Figure 8. Fixed Output Regulator



$$I_o = \frac{V_{XX}}{R_1} + I_Q$$

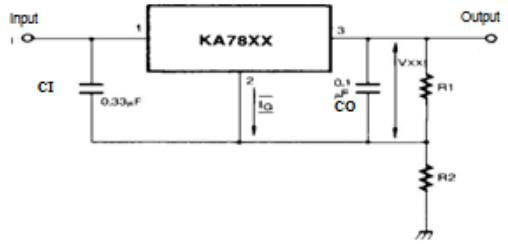
Figure 9. Constant Current Regulator

Notes:

(1) To specify an output voltage. substitute voltage value for "XX." A common ground is required between the input and the Output voltage. The input voltage must remain typically 2.0V above the output voltage even during the low point on the input ripple voltage.

(2) CI is required if regulator is located an appreciable distance from power Supply filter.

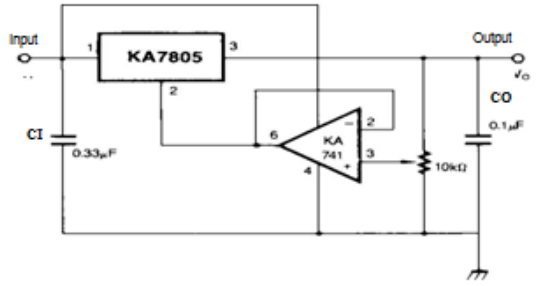
(3) CO improves stability and transient response.



$I_o \varepsilon$ 5IQ

VO = VXX(1+R2/R1)+IQR2

Figure 10. Circuit for Increasing Output Voltage



IRI ε5 IQ

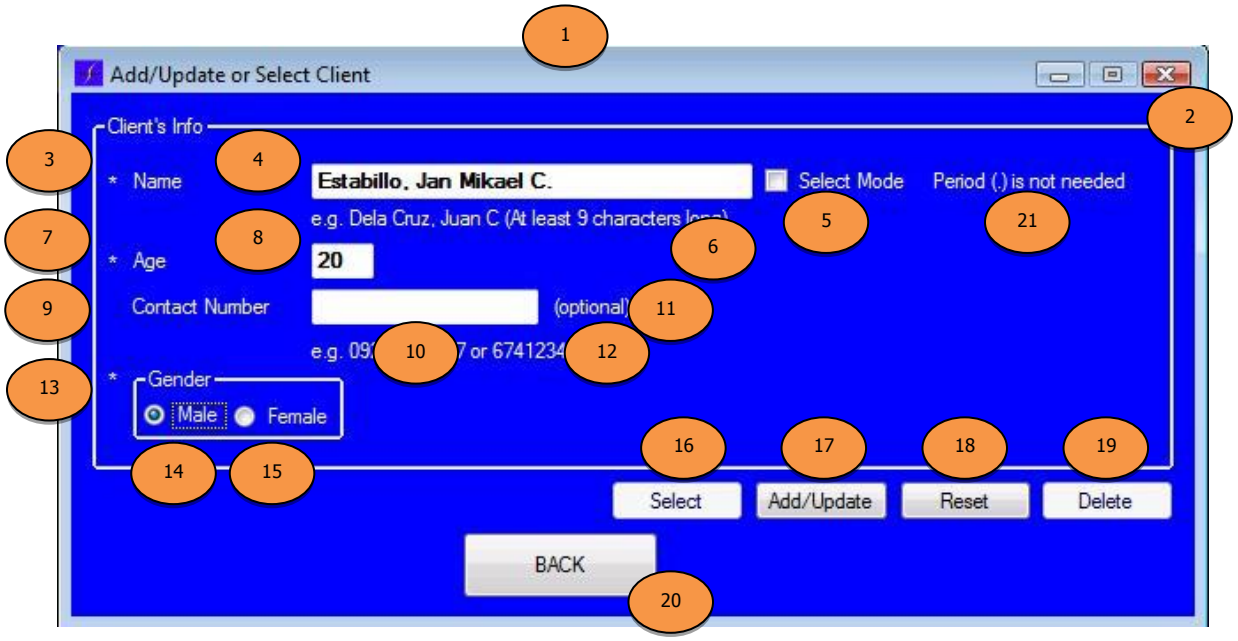
VO = VXX(1+R2/R1)+IQR2

Figure 11. Adjustable Output Regulator (7 to 30V)

APPENDIX E

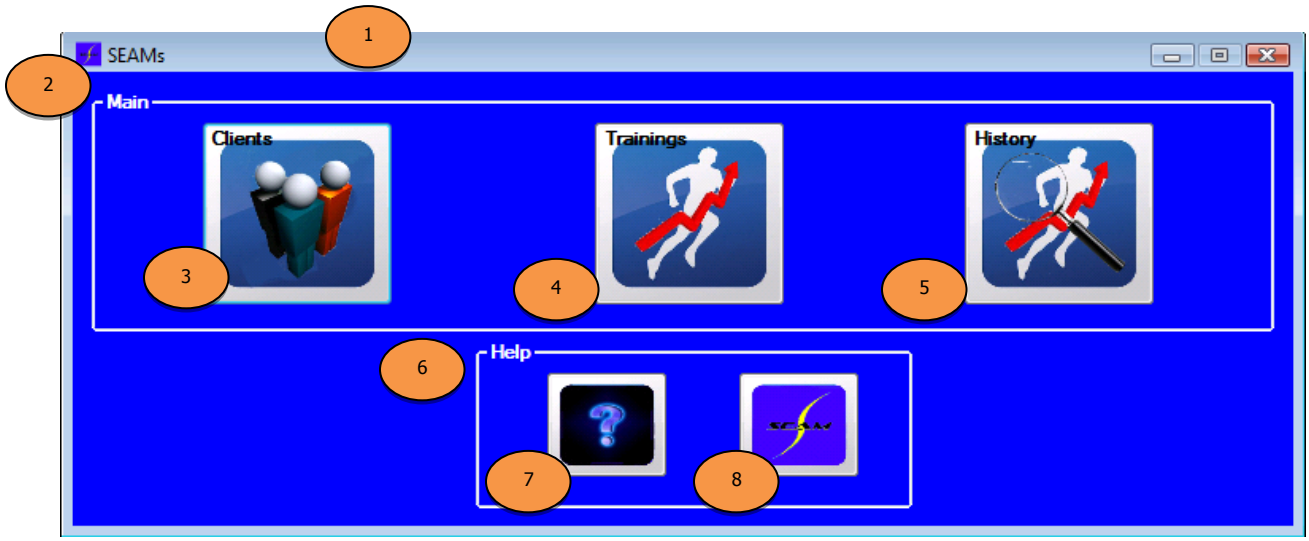
Software Development

Interface Design



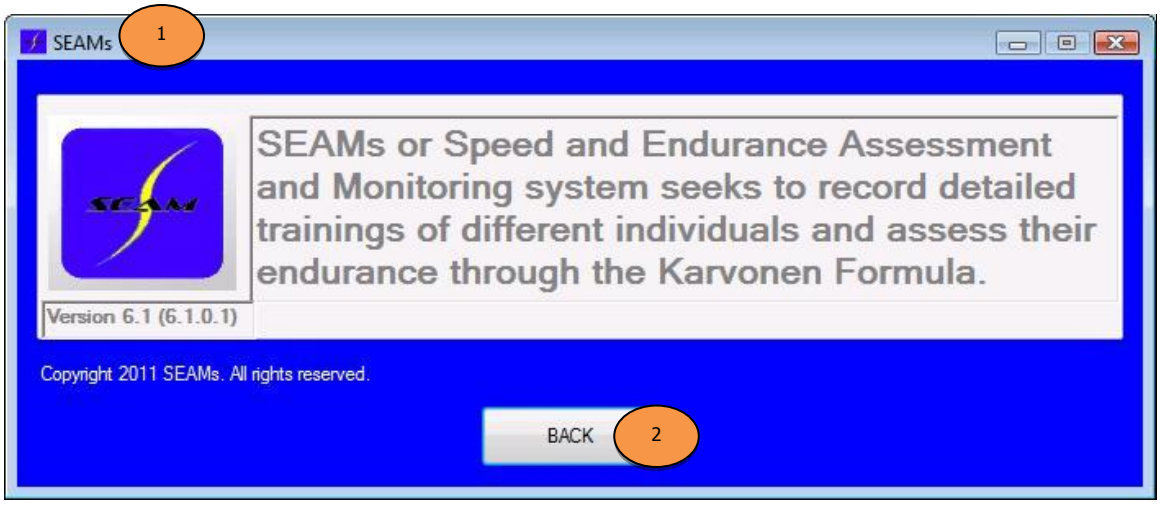
Client Form (1) is the window interface for managing client information. Client Group Box (2) encloses all input fields. Name label (3) is used to describe the adjacent text box which is the Name Text Box (4) or Name Combo Box (hidden behind Name Text Box). Name Text Box accepts any input and needs to follow the format shown in the Name Label to avoid errors. The Name combo box replaces Name Text Box whenever the Select Check Box (5) is checked and automatically accesses the client record for information and indexing listed on the items of the combo box. Altering Select Check Box immediately clears all fields. Age Text Box (8) which is designated by the Age Label (7) accepts only numeric values and has a maximum input of 3 digits. Contact Text Box (10)

attributed by Contact label (9) is similar to Age Text Box except that it only accept 0, 7, and 10-digit long numbers which are empty, landline, and mobile numbers, respectively. Gender group box (13) is responsible for keeping only one radio button checked. The group box consists of the Male Radio Button (14) and Female Radio Button (15) to indicate a male and a female genders, respectively. Select Button (16) is enabled only when select check box is checked which exports the fields to the corresponding text boxes in Training Form and transfers the control to the Training Form after closing the Client Form (1). If the name inputted exists, the Add Button (17) updates fields (8), (10), (14), and (15) and adds the client whenever the inputted name does not exist. Clear Button (18) resets all fields and loads existing clients on the name combo box. Delete Button (19) is enabled only when select check button is checked to avoid errors such as no client exists or name is not valid. Back Button (20) cancels the activity in client form and returns back to Main Form.

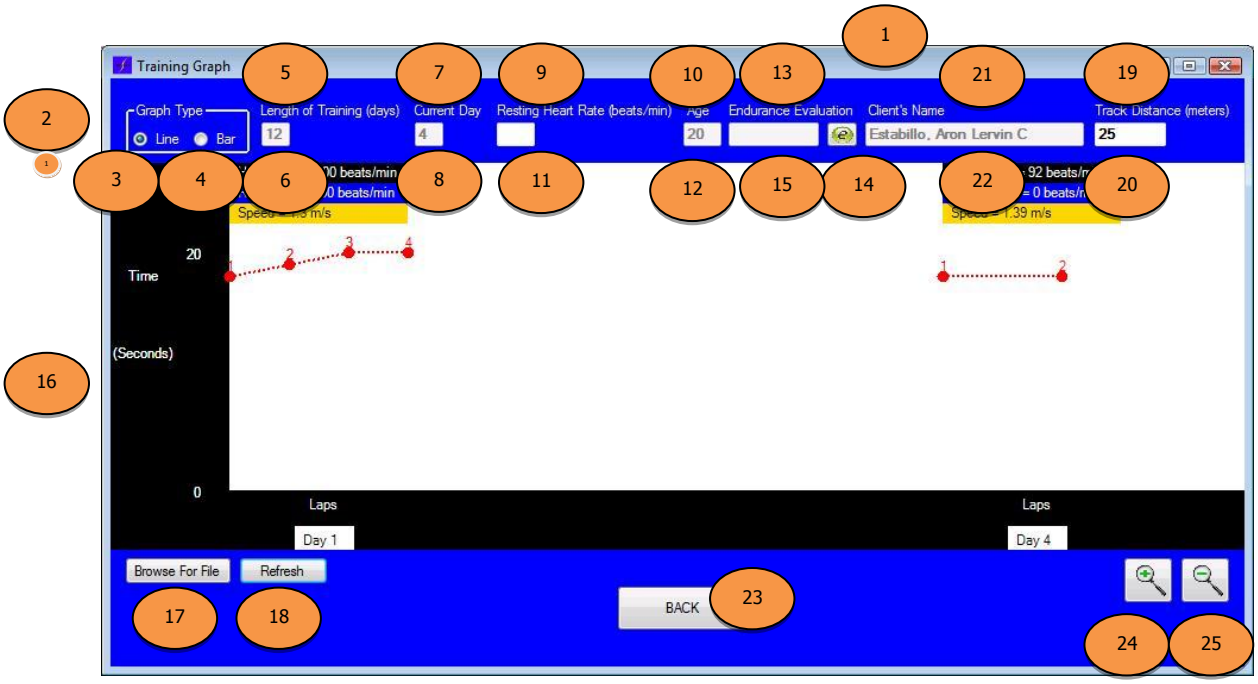


The main form (1) is the root menu of the application and contains all possible activities the user can make. Main Group Box (2) encloses the major activities the user can possibly do and contains Client Button (3), Training Button (4), and Browse Button (5). Help Group Box (6) encloses guides and information of the software containing Help Button (7) and SEAMs Button (8).

Help Form (1) is for giving instructions and guide about the major activities to the user. Back Button (2) closes Help Form and returns to the root menu.

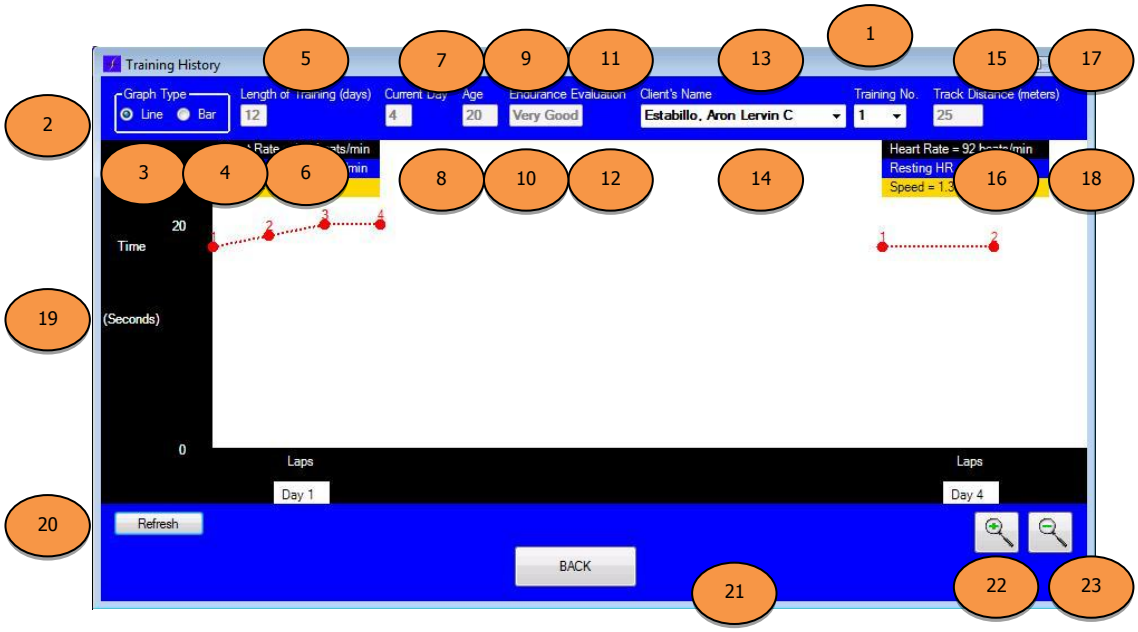


The SEAMs Form (1) is responsible for giving the function and information of the software application. Back Button (2) exits SEAMs Form and returns to the root menu.



The Training Form (1) is responsible for training information management. All labels (5), (7), (9), (10), (13), (21), and (19) are used to describe the text box

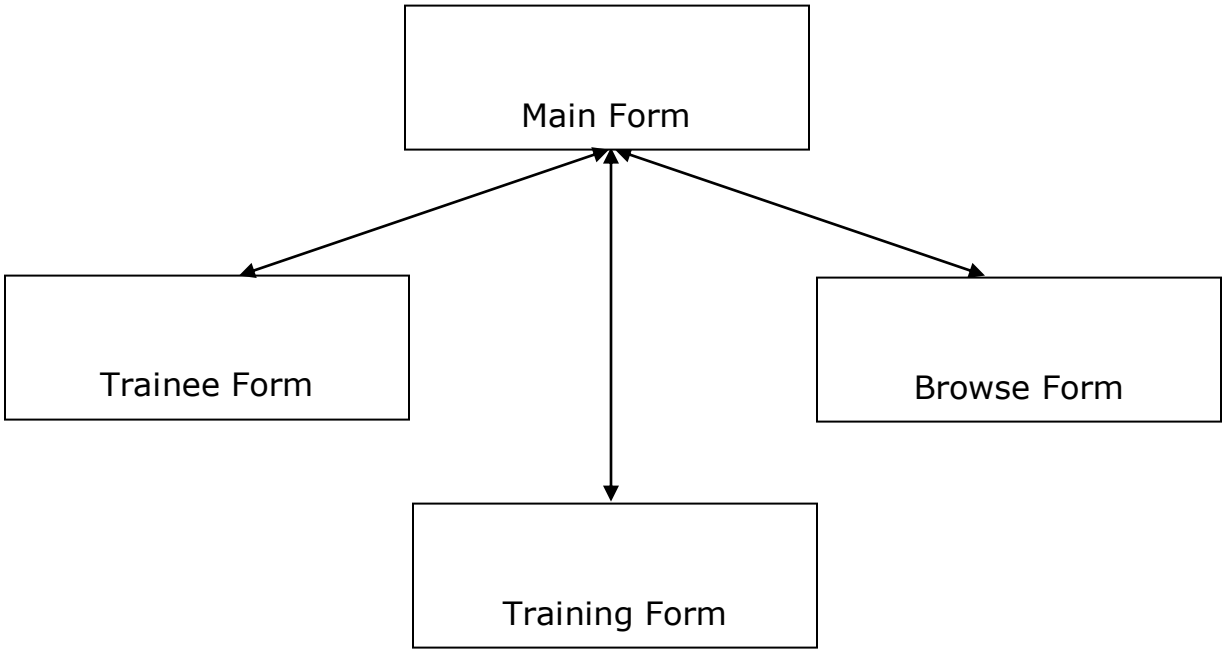
adjacent to them. Line Radio Button (3) and Bar Radio Button (4) specifies the type of graph displayed on the picture box (16) where the Line Radio Button is the default graph type. Text Boxes Training Length (6), Current Day (8), Age (12), and Name (22) are always disabled and gets data from the client record for display. Resting Heart Rate Text Box (11) only accepts numeric values from 50 to 100. Distance Text Box (20) accepts numbers only. Graph Picture Box (16) can be dragged by clicking and dragging to a direction and displays graph, resting heart rate, laps, lap times, days, training heart rate, and speed information, interactively. Open Button (17) uses Windows' open file dialog to search for the data in the plugged-in USB flash drive. Refresh Button (18) redraws the picture box whenever it is distorted. Back Button (23) prompts the user in saving the training information and exits whenever necessary returning to the root menu. Zoom In Button (24) is used to expand the length occupied by each day's graph while Zoom Out Button (25) is used to compress the length of graphs.



Browse Form (1) is similar to the Training Form except that the only enabled fields are the Name Combo Box (14) and Training Number Combo Box (16). Name Combo Box lists existing clients including those that do not have records and notifies the user of the status of the client's records. Training Number Combo Box gets all the existing training record of the selected client on the Name Combo Box. The Graph Picture Box (19) also has a dragging capability to provide interactive feel to the user. Back Button (21) exits the form without prompting the user for saving and returns to the root menu.

BASIC DESIGN SPECIFICATIONS

Program Organization



Major Data Structures

Primitive Types

Boolean — Expressions that are used during a True or False definition.

Characters — Data comprising characters and symbols.

Integers — Collection of signed numbers frequently used for counting and indexing.

String — Sets of characters that are a major element in data storage.

Single — numbers with decimal values required in drawing rectangles in a picturebox.

Arrays

Set of single decimal numbers to store lap times and heart rates during the course of training.

Key Algorithms

Indexed searching — searching all existing trainees using a file that gathers names every time they are added.

Unique counting — keeps track of the days passed during training and the number of trainings a specific trainee has engaged in.

Parsing — reads a specific data format to translate them into readable information.

Major Objects

Form (frm) – are main objects used hold the menus.

Label (lbl) – are texts that describe the purpose of textboxes.

Textbox (txt) – are fields that let user input strings or numbers.

Group Box (grp) – are objects containing several other objects to treat contained objects as one.

Combo Box (cmb) – are objects which comprise of searched strings to implement selection.

Radio Button (rdb) — are objects that serve as a Boolean for specified fields.

Buttons (btn) – fields that serve as action confirmation during form utilization.

Open File Dialog (ofd) - are objects that serve as a dialog box whenever searching for a file.

Error Processing

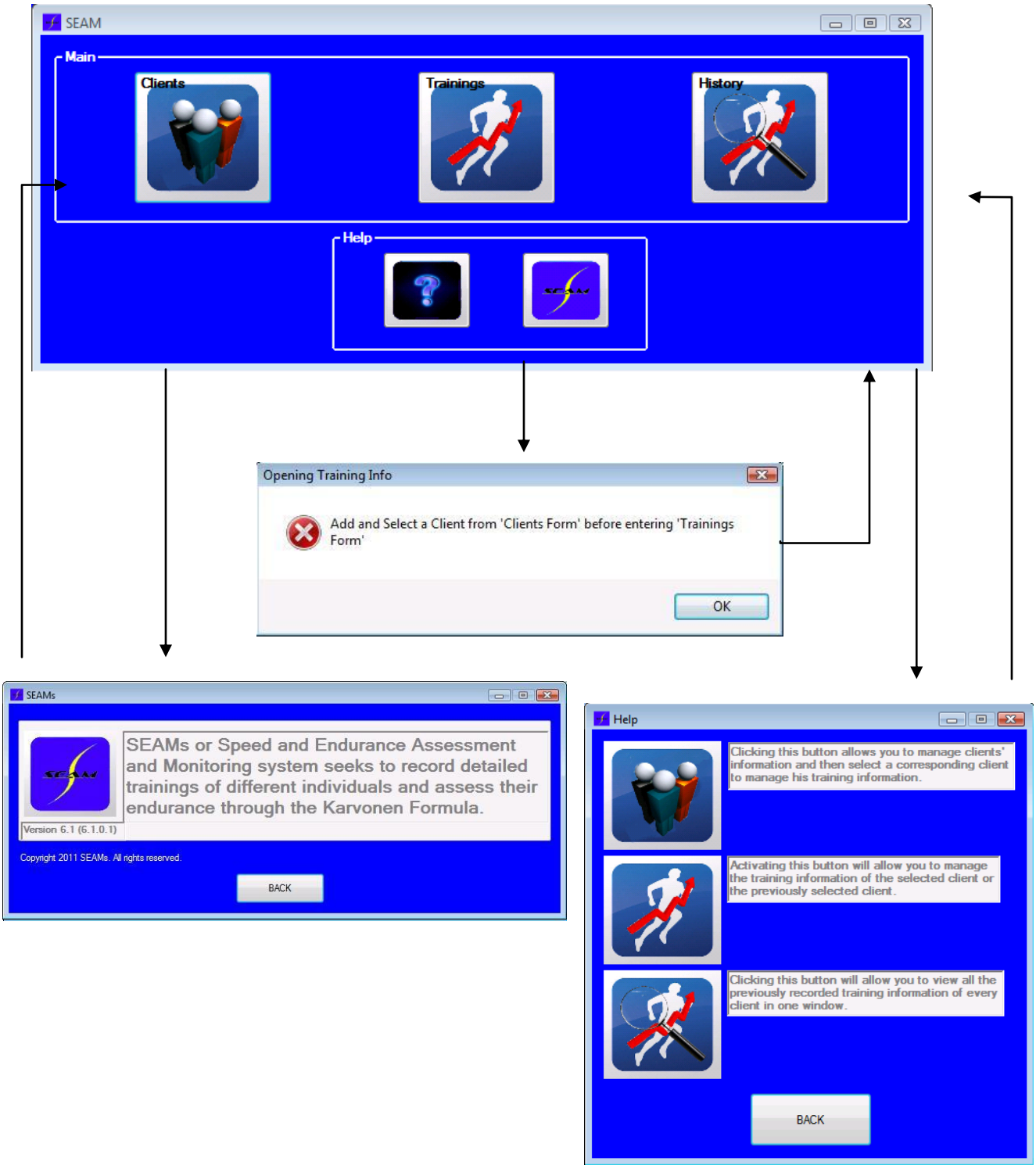
The software prevents the occurrence of errors but miniaturized windows are shown to display errors or shortcomings that bypassed the error prevention scheme. Most of the messages are required to be read and immediately reverses the action whenever prompted.

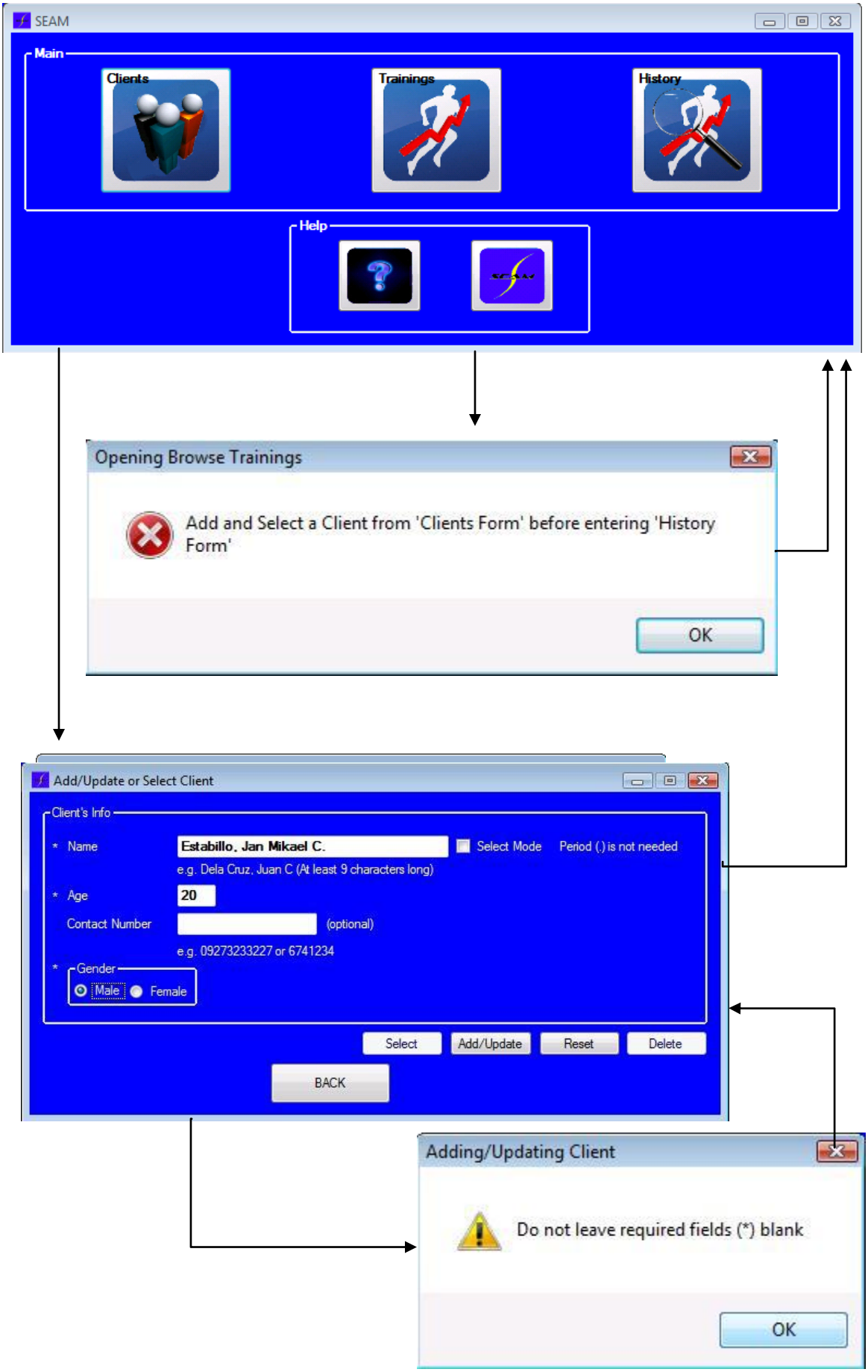
Performance

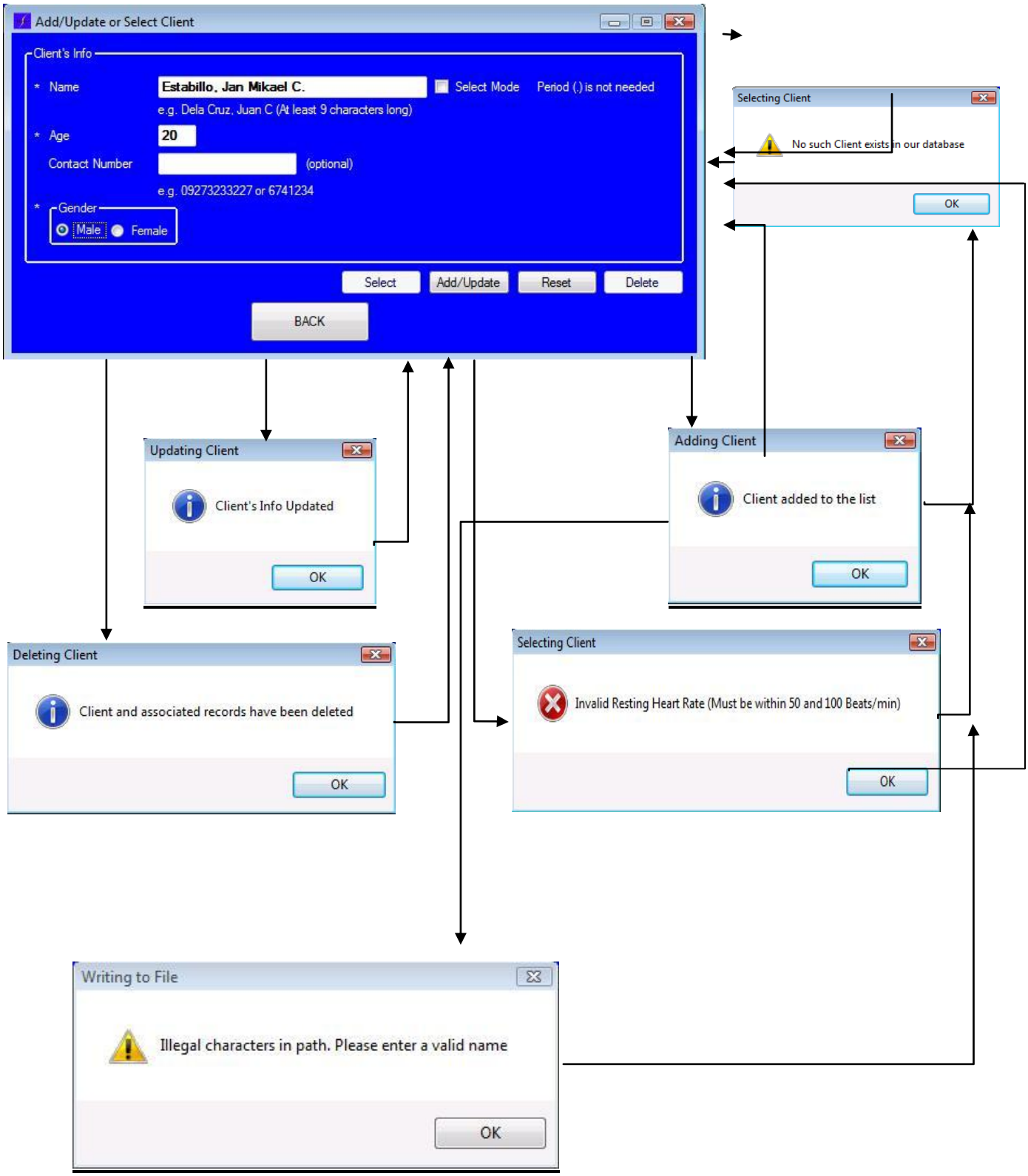
The program is incorporated with integrity using the Visual Basic Programming Language. The overall robustness of the application is guaranteed. Required fields are included to prevent data deficiency and a 'n/a' symbol to the contact number whenever left blank. Windows are closed whenever a new window is requested and entered to release resources and maximize memory used by the unused window.

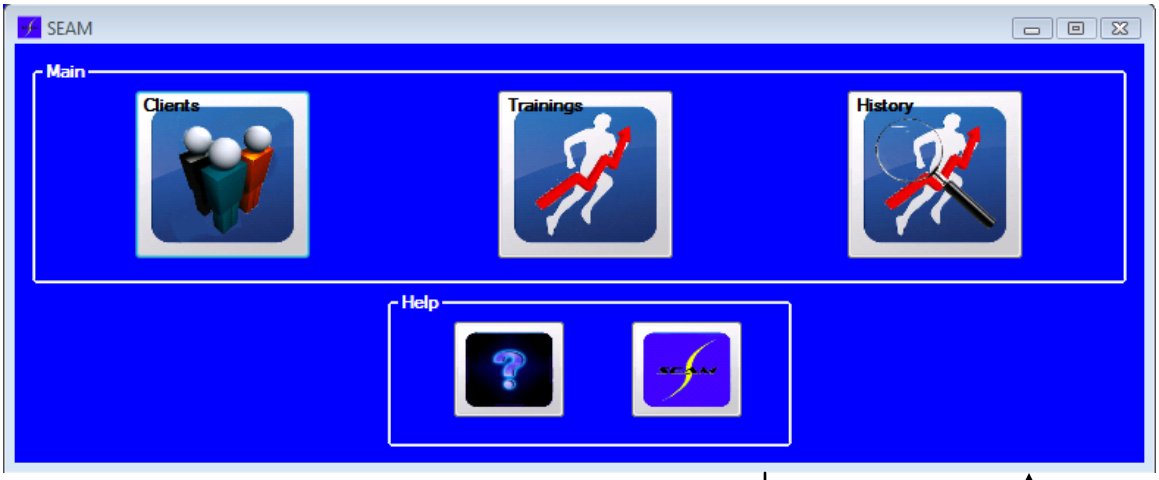
DETAILED DESIGN

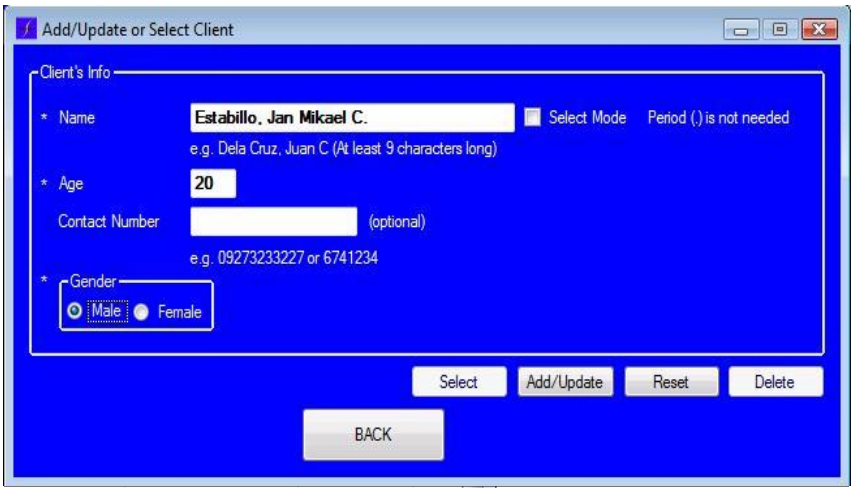
Screen Transition

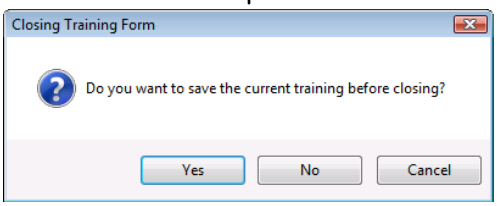



SEAM
Main
Clients
Trainings
History
Help
Opening Browse Trainings
Add and Select a Client from 'Clients Form' before entering 'History Form'
OK
Add/Update or Select Client
Client's Info
* Name
Estabillo, Jan Mikael C.
Select Mode
Period (.) is not needed
e.g. Dela Cruz, Juan C (At least 9 characters long)
* Age
20
Contact Number
(optional)
e.g. 09273233227 or 6741234
* Gender
Male
Female
Select
Add/Update
Reset
Delete
BACK
Adding/Updating Client
Do not leave required fields (*) blank
OK


Add/Update or Select Client
Client's Info
* Name
Estabillo, Jan Mikael C.
Select Mode
Period (.) is not needed
e.g. Dela Cruz, Juan C (At least 9 characters long)
* Age
20
Contact Number
(optional)
e.g. 09273233227 or 6741234
* Gender
Male
Female
Select
Add/Update
Reset
Delete
BACK
Selecting Client
No such Client exists in our database
OK
Updating Client
Client's Info Updated
OK
Adding Client
Client added to the list
OK
Deleting Client
Client and associated records have been deleted
OK
Selecting Client
Invalid Resting Heart Rate (Must be within 50 and 100 Beats/min)
OK
Writing to File
Illegal characters in path. Please enter a valid name
OK

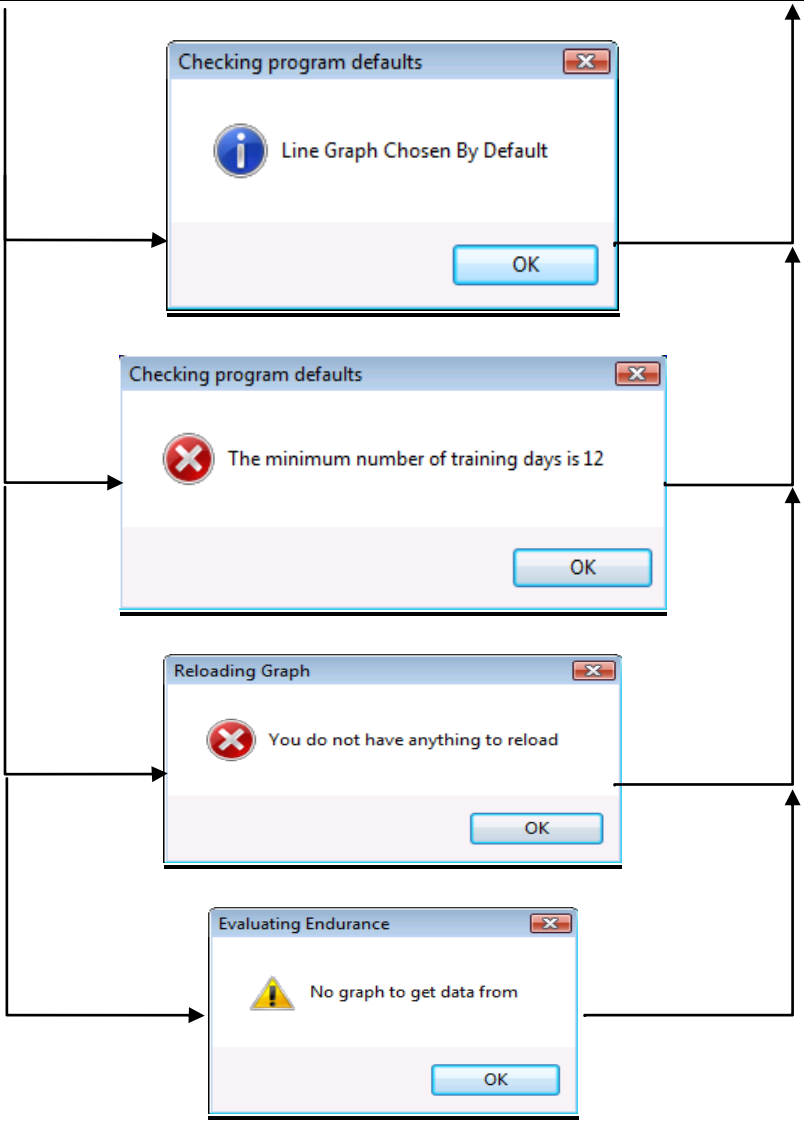

SEAM
Main
Clients
Trainings
History
Help

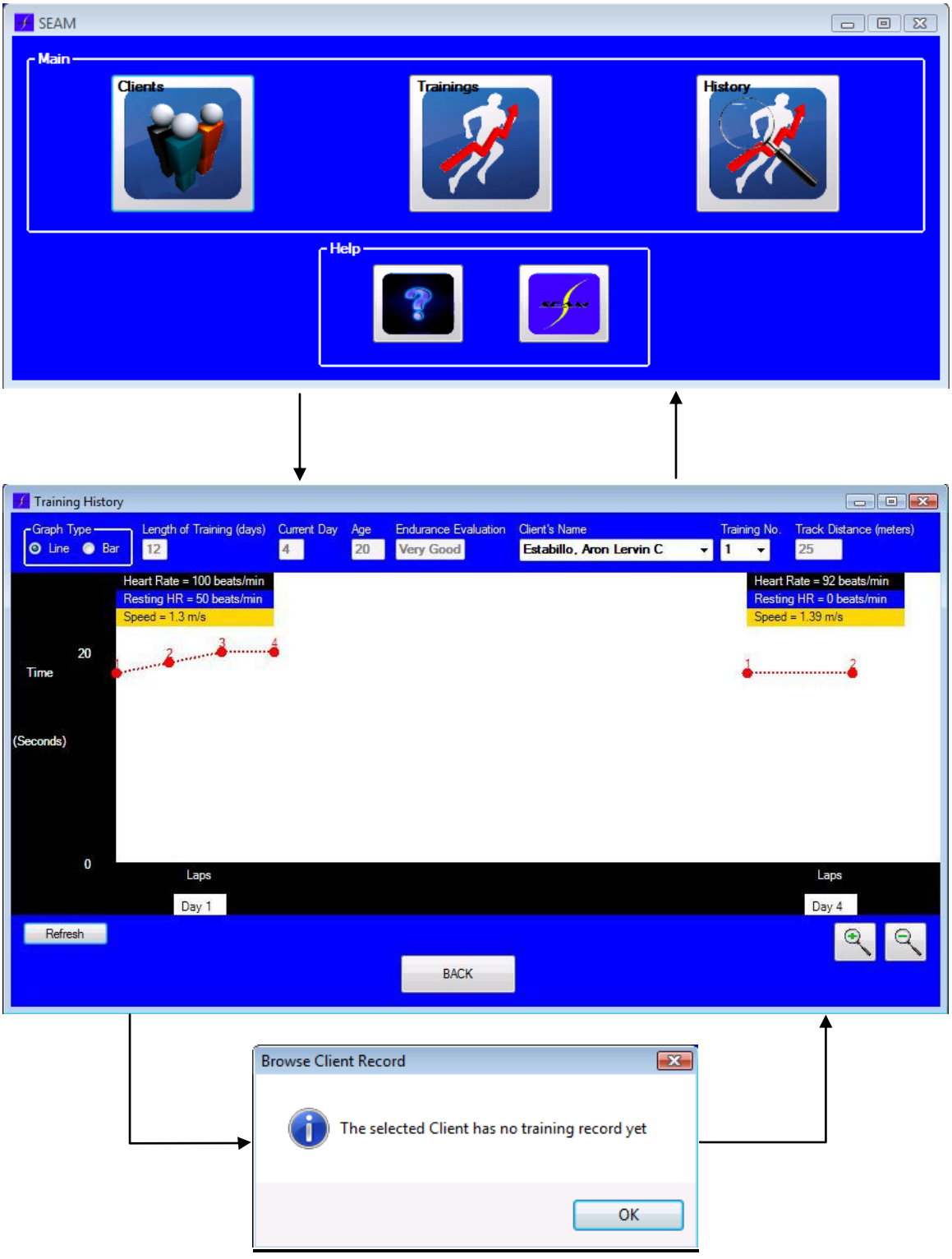

Add/Update or Select Client
Client's Info
* Name
Estabillo, Jan Mikael C.
Select Mode
Period (.) is not needed
e.g. Dela Cruz, Juan C (At least 9 characters long)
* Age
20
Contact Number
(optional)
e.g. 09273233227 or 6741234
* Gender
Male
Female
Select
Add/Update
Reset
Delete
BACK


Closing Training Form
Do you want to save the current training before closing?
Yes
No
Cancel


Training Graph
Graph Type
Line
Bar
Length of Training (days)
12
Current Day
4
Resting Heart Rate (beats/min)
Age
20
Endurance Evaluation
Client's Name
Estabillo, Aron Lervin C
Track Distance (meters)
25
Heart Rate = 100 beats/min
Resting HR = 50 beats/min
Speed = 1.3 m/s
Heart Rate = 92 beats/min
Resting HR = 0 beats/min
Speed = 1.39 m/s
20
Time
(Seconds)
0
Laps
Day 1
Laps
Day 4
Browse For File
Refresh
BACK

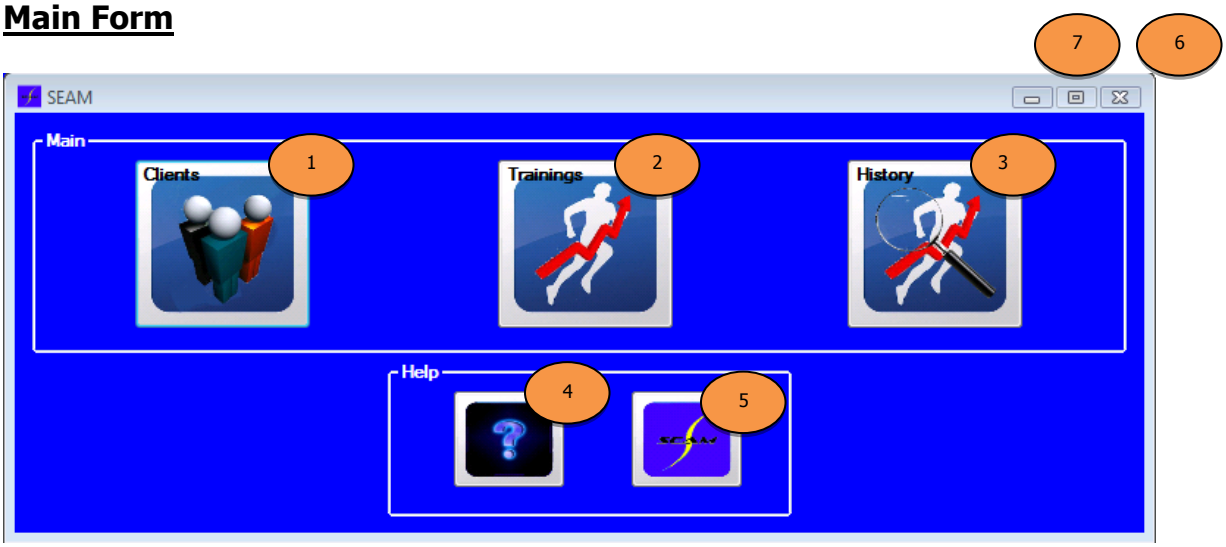

Training Graph
Graph Type
Line
Bar
Length of Training (days)
12
Current Day
4
Resting Heart Rate (beats/min)
Age
20
Endurance Evaluation
Client's Name
Estabillo, Aron Lervin C
Track Distance (meters)
25
Heart Rate = 100 beats/min
Resting HR = 50 beats/min
Speed = 1.3 m/s
Heart Rate = 92 beats/min
Resting HR = 0 beats/min
Speed = 1.39 m/s
20
Time
(Seconds)
0
Laps
Day 1
Laps
Day 4
Browse For File
Refresh
BACK


Checking program defaults
Line Graph Chosen By Default
OK
Checking program defaults
The minimum number of training days is 12
OK
Reloading Graph
You do not have anything to reload
OK
Evaluating Endurance
No graph to get data from
OK


SEAM
Main
Clients
Trainings
History
Help
Training History
Graph Type
Line
Bar
Length of Training (days)
12
Current Day
4
Age
20
Endurance Evaluation
Very Good
Client's Name
Estabillo, Aron Lervin C
Training No.
1
Track Distance (meters)
25
Heart Rate = 100 beats/min
Resting HR = 50 beats/min
Speed = 1.3 m/s
Heart Rate = 92 beats/min
Resting HR = 0 beats/min
Speed = 1.39 m/s
20
Time
(Seconds)
0
Laps
Day 1
Laps
Day 4
Refresh
BACK
Browse Client Record
The selected Client has no training record yet
OK

PDL - Program Description Language

Main Form



On Click – Client Button (1)

Load Client Form

On Click – Training Button (2)

Check if a client is currently selected

If yes

Hide Form

Load Training Form

If no

Cancel Loading

Notify user that no client is selected

On Click – Training History Button (3)

Check if a client is currently selected

If yes

Hide Form

Load History Form

If no

Cancel Loading

Notify user that no client is selected

On Click – Help Button (4)

Hide Form

Load Help Form

On Click – SEAM Button (5)

Hide Form

Load SEAM Form

On Click – Close Button (6)

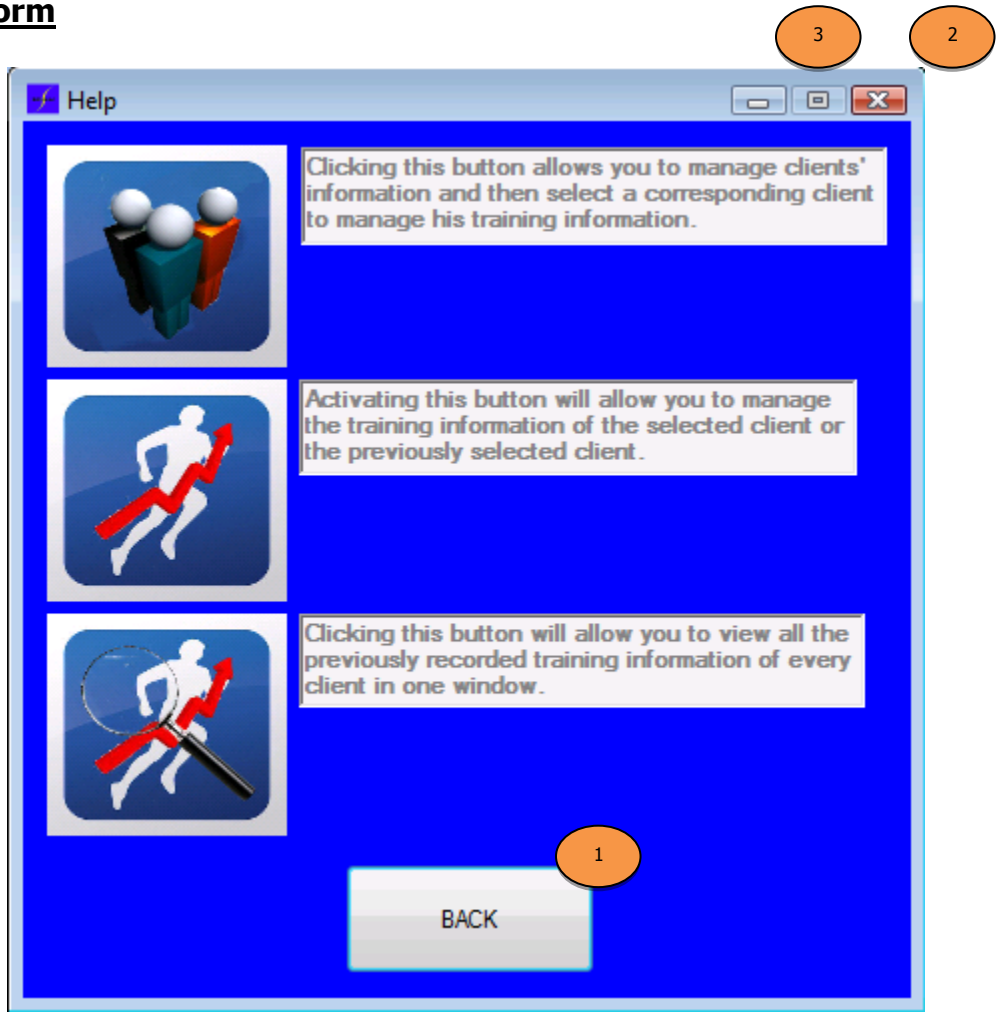
Close Main Form

End Program

On Click – Minimize Button (7)

Minimize Form

Help Form



On Click – Back Button (1) or – Close Button (2)

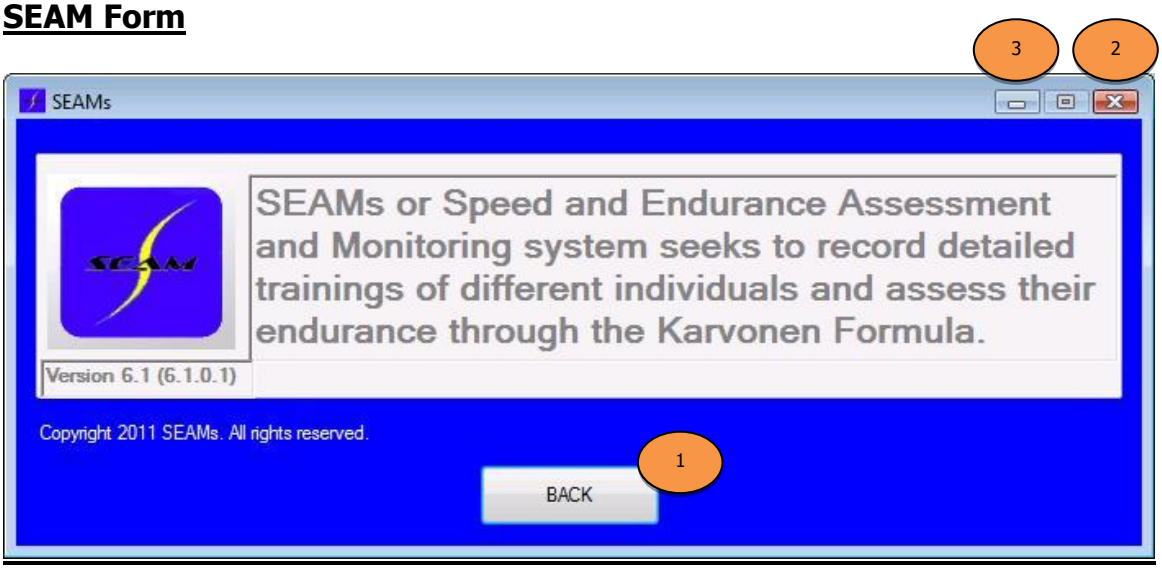
Close Form

Return to Main Form

On Click – Minimize Button (3)

Minimize Form

SEAM Form



On Click – Back Button (1) or – Close Button (2)

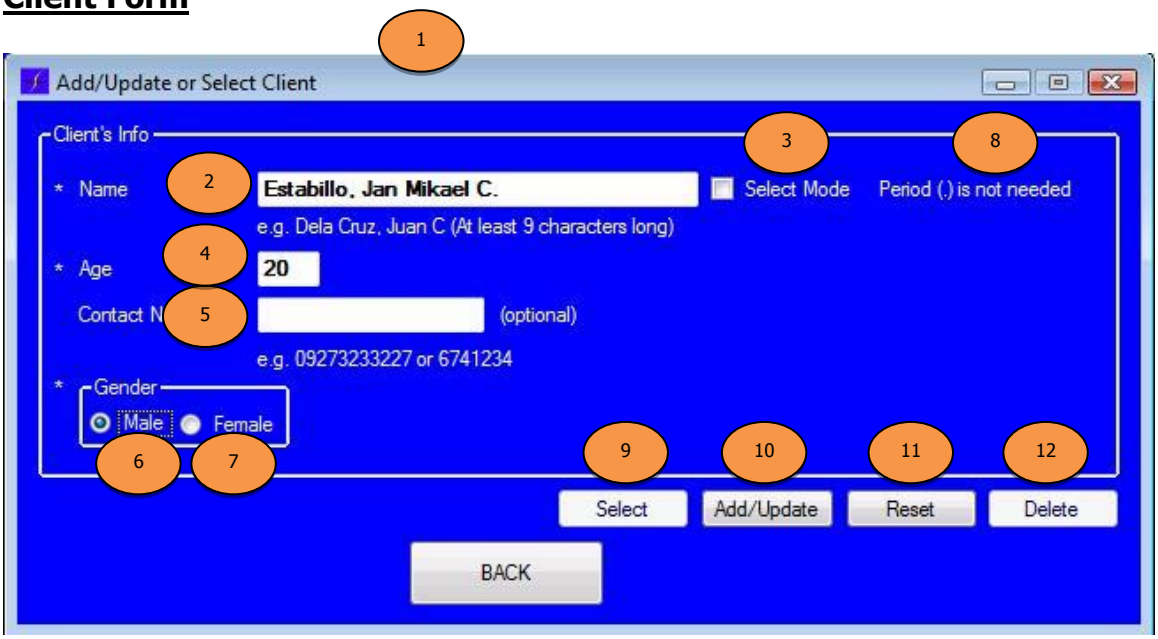
Close Form

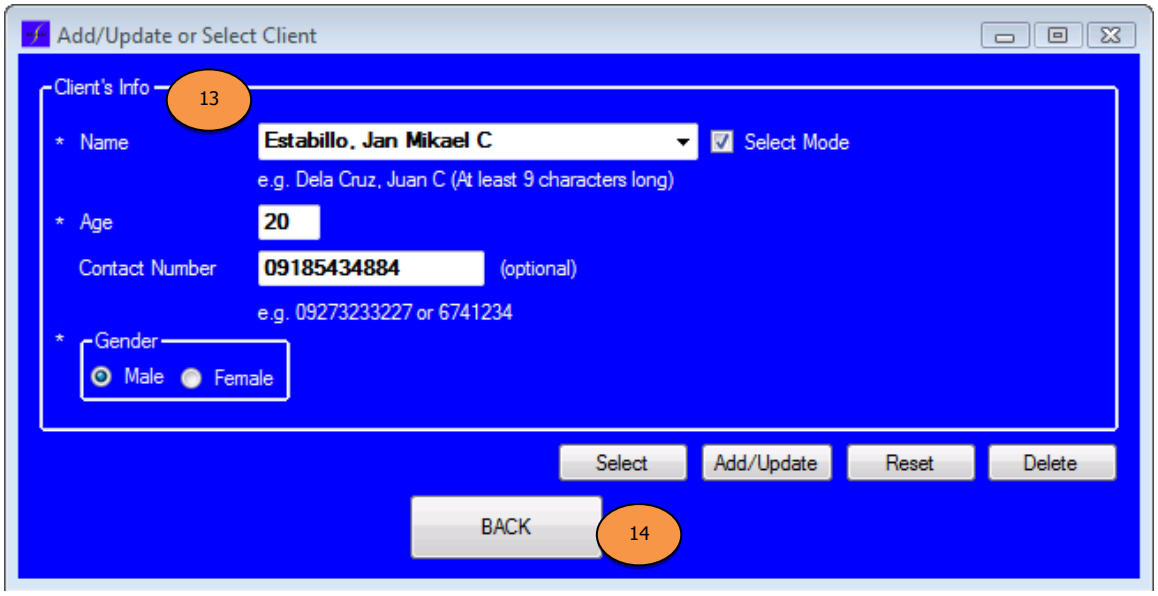
Return to Main Form

On Click – Minimize Button (3)

Minimize Form

Client Form





On Load – Client Form (1)

Clear Fields()

{

Clear Name Text Box (2), Age Text Box (4), Contact Number Text Box (5), and Name Combo Box Text and Items (13). Uncheck Male Radio Button (6) and Female Radio Button (7),

}

Check if client record exists

If not

Create a client record

Check if indexing record exists

If not

Create an indexing record

On Text Change – Name Text Box (2)

Appends and suggests existing clients

Check format of input

Notify user of the status of the input in relation to the client record

On Text Change – Age Text Box (4)

If input is not numeric

Notify user

Clear Age Text Box (4)

On Text Change – Contact Number Text Box (5)

If input is not numeric

Notify user

Clear Contact Number Text Box (5)

On Selected Index Change – Name Combo Box (13)

Load Fields Age Text Box (4), Contact Number Text Box (5), Radio Buttons Male (6) and Female (7)

On Click – Male Radio Button (6)

Check Male Radio Button (6)

Uncheck Female Radio Button (7)

On Click – Female Radio Button (7)

Check Female Radio Button (7)

Uncheck Male Radio Button (6)

On Click – Select Check Box (8)

If Select Check Box (8) is Checked

Show Name Combo Box (14)

Hide Name Text Box (1)

Enable Select Button (9)

Enable Delete Button(12)

Clear Fields()

List Existing Clients in the Name Combo Box (13)

{

Read clients from indexed record

Store them to Name Combo Box (13)

if no client is currently indexed

Clear Name Combo Box (13) Items

}

If Select Check Box (8) is Unchecked

Hide Name Combo Box (13)

Disable Select Button (9)

Disable Delete Button (12)

Show Name Text Box (2)

Clear Fields()

On Click – Select Button (9)

If any of the fields with (*) is left blank

Cancel Selection and notify user

If Resting Heart Rate is not within the range 50 to 100

Cancel Selection and notify user

if client exists in the record

Register the client as the current selected client

Clear Fields()

Uncheck Select Check Box (8)

Hide Form

Load Training Form

If client does not exist

Cancel selection and notify user

On Click – Add/Update Button (10)

If any of the fields with (*) is left blank

Cancel Add/Update and notify user

If Resting Heart Rate is not within the range of 50 to 100

Cancel selection and notify user

If name is invalid

Cancel Adding/Updating and notify user

If not

If client exists

Update the client's record

If not

Add the client to the Record

Index the added client

On Click – Reset Button (11)

Clear Fields()

List existing Clients in the Name Combo Box (13)

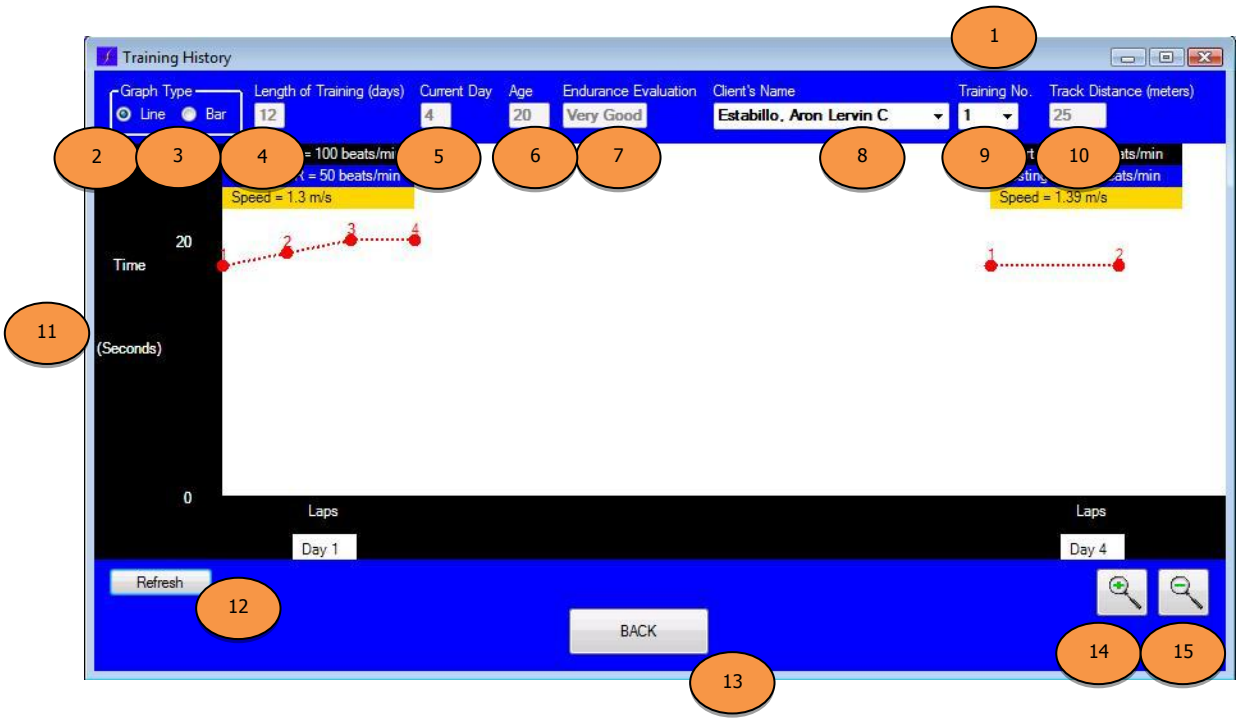
On Click – Delete Button (12)

Delete all items that has the client's name on it including the indexing record and currently selected record then notify user.

On Click – Back Button (14)

Close Form (1)

Return to Main Form



On Horizontal Scroll – History Form (1)

Relocate all elements except Graph Picture Box (11)
Reload Training

On Vertical Scroll – History Form (1)

Relocate all elements
Reload Training

On Load – History Form (1)

Auto-Click Refresh Button (12)
Auto-Click Line Graph Radio Button (2)
Query Indexed Clients
Store Clients to Name Combo Box (8)
Select the first indexed client
Query Training Record of selected client
Store training numbers in corresponding combo box (9)

On Click – Refresh Button (12)

Load Currently Selected Training
Evaluate Endurance using Karvonen Formula

On Selected Index Change – Name Combo Box (8)

Query Training Record of selected client
Store training numbers in corresponding combo box (9)

On Selected Index Change – Training Number Combo Box (9)

Query Training Record
If Training Record exists
Load Training
Update Fields Training Length Text Box (4), Current Day Text Box (5), Age Text Box (6), and Endurance Evaluation Text Box (7)

If Training Record does not exist
Invalidate Graph Picture Box (11)
Notify User

On Distort – Graph Picture Box (11)

Reload Training and Fields (4), (5), (6), and (7)

On MouseWheel – Graph Picture Box (11)

If direction is upward

Expand graph length

Else if downward

Compress graph length

On Click – Back Button (13)

Close Form (1)
Return to Main Form

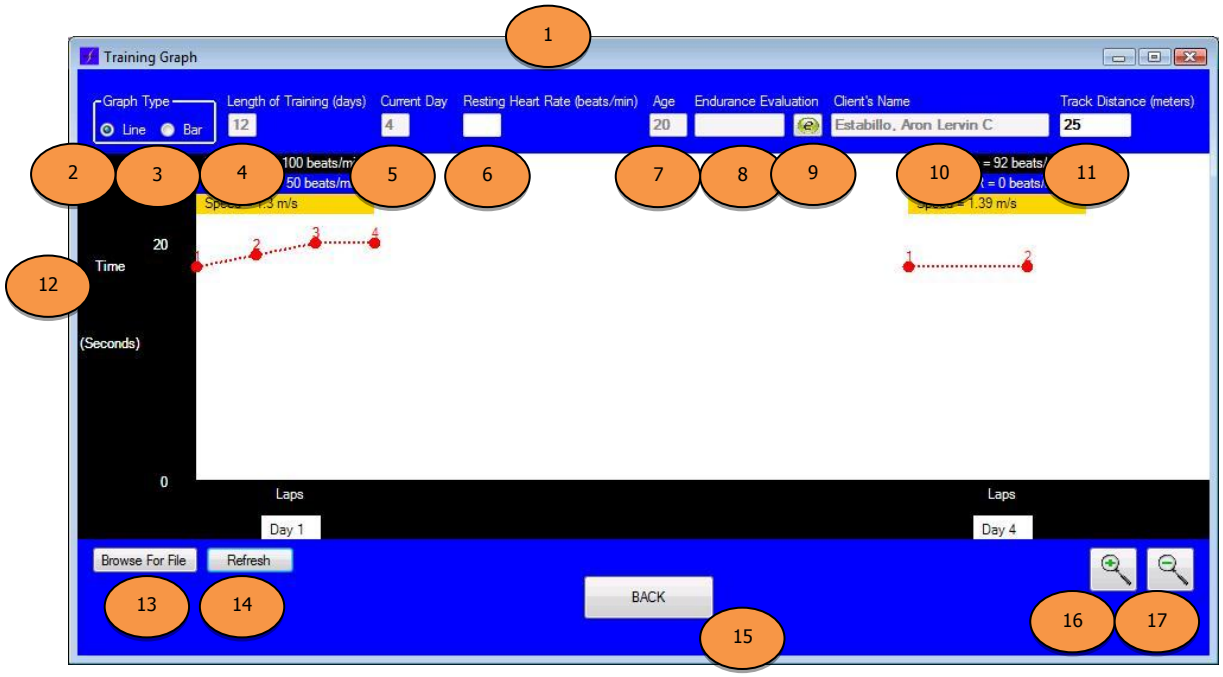
On Click – Zoom In Button (14)

Expand length of graph

On Click – Zoom Out Button (15)

Compress length of graph

Training Form



On Horizontal Scroll – Training Form (1)

Relocate all elements except Graph Picture Box (12)
Reload Training

On Vertical Scroll – Training Form (1)

Relocate all elements
Reload Training

On Load – Training Form (1)

Query Selected Client Record
Update Fields Training Length Text Box (4), Current Day Text Box (5), Resting Heart Rate Text Box (6), Age Text Box (7), Name Text Box (10), Track Distance Text Box (11)
Load Training

On Text Change – Training Length Text Box (4)

If input is not numeric
Notify user
Clear Training Length Text Box (4)

On Text Change – Track Distance Text Box (11)

If input is not numeric
Notify user
Clear Track Distance Text Box (11)

On Distort – Graph Picture Box (12)

Reload Training and Update Fields

On Click – Refresh Button (14)

If Training exists
Reload Training

If not
Notify user and invalidate Graph Picture Box (12)

On Click – Browse Button (11)

If Current Day is greater than Training Length
Ask user if a new training is necessary
If yes
Create new training record

If no
Reload Training

Else if Current day has an existing training
Ask user if replace is necessary
If yes
Replace current day training with selected data
If no
Reload training w/o replacing

Else if Current day has no training
Create Training Record
Load Created Record

On MouseWheel – Graph Picture Box (11)

If direction is upward

Expand graph length

Else if downward

Compress graph length

On Closing – Training Form (1)

Ask user if saving is necessary
If yes
Save training data
If no
Close Form (1)
Return to Main Form
If cancel
Return to Training Form (1)
Reload Training

On Click – Zoom In Button (16)

Expand graph length

On Click – Zoom Out Button (17)

Compress graph length

TEST CASE

Add / Update Clients Form

Test Case	Entry Condition	Expected Output	Actual Output
Name is blank	Check Availability	Message	Please enter a name
Name already exists	Check Availability	Message	Name is not available
Name is valid	Check Availability	Message	Name is available
Name is invalid	Add / Update or Select	Error Message	Client added to the list / Illegal Characters in path. Please enter a valid name.
Name is blank	Add / Update or Select	Error Message	Do not leave required fields (*) blank
Age is blank	Add / Update or Select	Error Message	Do not leave required fields (*) blank
Resting Heart Rate is blank	Add / Update or Select	Error Message	Do not leave required fields (*) blank
Resting Heart Rate Not in Range	Add / Update or Select	Error Message	Invalid Resting Heart Rate (must be within 50 to 100)
Gender is blank	Add / Update or Select	Error Message	Do not leave required fields (*) blank

Delete Clients Form

Test Case	Entry Condition	Expected Output	Actual Output
Clients Name is blank	Select Delete	Error Message	No such Client exists in our database
Client Name is invalid	Select Delete	Error Message	Client and associated records

			have been deleted

Trainings Form

Test Case	Entry Condition	Expected Output	Actual Output
No Data Selected	Select Evaluate	Error Message	No graph to get data from
Length of training days is blank or less than 12	Select Refresh	Error Message	The minimum number of training days is 12
No Data Selected	Select Refresh	Error Message	You do not have anything to reload
Invalid Selected File Format	Select Browse For File	Error Message	Invalid File Format
Invalid Input	Training Length	Error Message	Only numerical inputs are required
Training Length Exceeded	Select Browse For File	Error Message	Today has exceeded the training length. Do you want to start a new training with the selected file as your initial day?

History Form

Test Case	Entry Condition	Expected Output	Actual Output
Training Record of client is empty	Client Name Combo Box Selection	Error Message	The selected Client has no training record yet

DEVELOPMENT STANDARDS

Control Naming

CONTROLS	PREFIX	SAMPLE
Form	frm	frmHelp frmSEAMs frmMain frmInfo frmEndurance frmBrowse
Label	lbl	lblName lblContact lblAge lblRest
Text Box	txt	txtName txtContact txtAge txtRest
Group Box	grp	grpMain grpHelp grpGraph
Combo Box	cmb	cmbName

		cmbTraining
Button	btn	btnAdd
		btnBack
		btnEvaluate
		btnDelete
		btnClear
		btnTraining
		btnBrowse
		btnInfo
Picture Box	pcb	pcbGraph
Radio Button	rdb	rdbMale
		rdbFemale
Check Box	chk	chkSelect

APPENDIX F

Specifications of the Digital Blood Pressure used in the Testing

Omron HEM-773AC Package Contents

Omron HEM-773AC blood pressure monitor, cuff, AC adapter, 4 "AA" batteries, built-in storage case.

- IntelliSense™ Automatic Inflation and Deflation Technology
- Fills arm cuff with air, releases pressure when measurement is complete.
- Pressure valve preset switch allows the correct cuff inflation level to be determined before measurement is taken
- 2 person 21 memory recall date & time stamped for user
- Built-in compartment for convenient cuff storage
- Fuzzy Logic, automatically determines exact inflation level for each user
- Extremely easy to use
- Curved contour ComFit™ cuff provides user with a uniform fit to ensure accuracy
- Cuff fits arms sizes 9" to 17" in circumference
- Systolic and diastolic blood pressure & pulse
- Features angled faceplate and extra large LCD panel for improved visibility
- Includes AC adapter
- Optional operation on 4 "AA" batteries (included)