Palestine Polytechnic University

College of Engineering & Technology
Electrical & Computer Engineering Department

Graduation Project

Designing Spirometer Using Turbine Sensing Devices

Project Team

Mashhour I.Y Tarayrah

Hadeel A. Arar

Sojoud A. Al-Rjoub

Project Supervisor

Dr. Ghandi Manasra

Hebron – Palestine

2007

Abstract

Functionally, the respiratory system composed of an integrated set of regulated processes that include pulmonary ventilation (breathing) and gas exchange in the lungs. Breathing is the exchange of air between the atmosphere and the lung, as air moves into and out of the lungs it travel from regions of high air pressure to regions of low air pressure.

Spirometers are devices used to measure the volume of air inhaled or exhaled by the lungs. Diagnostic spirometers measure the flow and volume of gas moving into and out of the lungs during inspiratory and expiratory efforts. They help diagnose and monitor disease, as well as assess treatment effectiveness and disability. Comparing spirometric values to the patient's previous values or to normal values for a patient of the same sex, height, weight, and/or age provides information about pulmonary function. Spirometers aid in diagnosing lung diseases, classifying obstructive and restrictive disorders, and measuring the efficacy of subsequent therapies. They also help distinguish between pulmonary anomalies and anomalies of other origin (e.g., neuralgic, chest wall).

There are two types of Spirometers using flow sensing or volume sensing.

In our project we will use the flow sensing techniques working by turbine sensor, when patient performs specific breathing maneuvers the turbine will move and cut infrared sensor a counter will count the plusses and by using computer software it will change it to volume, peak expiratory flow (PEF), functional Residual capacity (FRC), then it can be stored for transfer analysis and testes.

الخلاصة

يعتبر الجهاز التنفسي من الأجهزة الرئيسية و الحيوية للكائن الحي, حيث يقع على عاتقه القيام بالعديد من العمليات الحيوية الضرورية لبقاء الجسم على قيد الحياة. فمن خلاله تأخذ خلايا الجسم المختلفة غاز الاكسجين اللازم لانتاج الطاقة, وتتخلص من غاز ثاني اكسيد الكربون.

يهدف هذا المشروع إلى قياس الحجم الكلي للهواء الذي يمكن اخراجه أثناء الزفير ومعدل تدفق الهواء الخارج,وتتم عملية التشخيص بمقارنة القيم والقراءات الناتجة من الفحص مع قيم معيارية تعتمد على طول وعمر وجنس الشخص والمنطقة الجغرافية التي يعيش بها.

وفي هذا المشروع تم استخدام تقنية Flow Sensing بالاعتماد على Turbine sensor . ففي أثناء عملية التنفس تتحرك لوحات Turbine مما يؤدي إلى قطع الضوء الواصل بين المرسل والمستقبل مما يدي إلى إنشاء pulses . وباستخدام DAQ يتم إدخال الإشارة إلى الكمبيوتر والتعامل معها باستخدام برنامجي LABVIEW, MATLAB

Table of contents

Dedication	iv
Acknowledgments	v
Abstract	vi
Abstract(Arabic)	vii
Table of contents	viii
List of Tables	xii
List of Figures	xiii

Chapter One: - Introduction

1.1 Introduction	2
1.2 Project Objectives	3
1.3 Importance of the Project	4
1.4 Literature Review	4
1.5 Time Plan	5
1.6 Economical Study	6
1.7 Risk Management	6
1.7.1 Hard risk	6
1.7.2 Software risk	7
1.7.3 Group risk	7
1.7.4 Requirement risk	7
1.7.5 Project risk	7
1.8 Project Content	8

Chapter Two: - Physiology of Respiration

2.1 Introduction 10

2.2 Respiratory System Structure 11

2.3 The Mechanism of Respiration 12

2.4 The factors affecting ventilation 16

2.5 Respiratory Control 18

2.6 The lung Volumes 19

2.7 Respiratory Rate 21

2.8 Diseases of the lung 21

Chapter Three: - Theoritical Background

3.1 Introduction 25

3.2 Spirometers, Diagnostic 25

3.3 Spirograms 26

3.4 Types of Spirometers 27

3.4.1 Volume 27

3.4.1.1 Volume Measuring devices 27

3.4.1.1.1 Water-sealed spirometer 28

3.4.1.1.2 The Dry-seal spirometer 28

3.4.1.1.3 The Bellows, or wedge, spirometer 29

3.4.1.2 Advantages of volume displacement Spirometers 29

3.4.1.3 Disadvantages of volume displacement Spirometers 29

3.4.2 Flow Measuring Devices 30

3.4.2.1 Pneumotachograph 30

3.4.2.2 Hot Wire Anemometer 31

3.4.2.3 Sonic Devices 32

3.4.2.4 The Ultrasonic Principle 32

Chapter Four :- Design Concepts
4.1 Measurement of Ventilator Function 35
4.2 General Block Diagram 36
4.3 The Technique - How to do the spirometer? 40
4.3.1 Patient-Related Problems 41
4.3.2 Instrument-Related Problems 42
4.4 Predicted Normal Values 43
4.5 Interpretation of Ventilator Function Tests 43
4.5.1 Classifying Abnormal Ventilator Function 43

Chapter Five:- Hardware System Design
5.1 Structural design 49
5.2 Turbine Flow sensor 50
5.2.1 Gas infrared emitting diode (LED 55C) 50
5.2.2 Phototransistor 54
5.3 DAQ (Data Acquisition System) 56
5.3.1 DAQ specifications and requirements 58

Chapter Six:- Software System Design
6.1 The "LabVIEW" software 60
6.1.1 VI 61
6.1.1.1 Front Panel 62
6.1.1.2 Block Diagram 63
6.1.2 Why use LabVIEW 64
6.2 MATLAB 64
6.2.1 The MATLAB System 66
6.2.1.1 Development Environment 67

6.2.1.2 The MATLAB Mathematical Function Library 67
6.2.1.3 The MATLAB Language 67
6.2.1.4 Graphics 67
6.2.1.5 The MATLAB Application Program Interface (API) 68
6.2.2 GUI (Graphical User Interface) 68
6.2.3 Using MATLAB in this Project 69
6.2.3.1 Spirometer Calculation 71
6.2.3.1.1 Measurements made in Spirometry 71
6.2.3.1.2 Calculated parameters 71
6.2.3.1.2.2 Temperature correction factor 71
6.2.3.1.3 The Predicted Values 72
6.2.3.1.4Calculating the %Result 74
6.2.3.1.5 FEV1 as a Percentage of FVC 75
6.2.3.1.6 Comparing normal with the measured Values 75
6.2.3.2 Change over time 77

Chapter Seven:- System Testing
7.1 Hardware Testing 80
7.2 System Testing 81

Chapter Eight:- Conclusions and Future Works
8.1 Conclusions 87
8.2 problems 88
8.2.1 Hardware problems 88
8.2.2 Software problems 88
8.3 future works 89

Referencec 90
Appendices 92

List of Tables

Table 1.1	Economical Study	6
Table 4.1	Comparison Between Thermister and Turbine spirometers	39
Table 4.2	Classification of Ventilator Abnormalities by Spirometry	46

List of Figures:

Figure1.1 The First Time Planning 5

Figure1.2 The Second Time Planning 5

Figure 2.1 Respiratory System Structure 12

Figure 2.2 Quiet inspiration 13

Figure2.3 Quiet expiration 14

Figure 2.4 Intrapulmonary pressure 14

Figure 2.5 Intraplural Pressure 15

Figure 2.6 Changes of pressure 16

Figure 2.7 Breathe volume 16

Figure 2.8 Elastic fiber like balloon 17

Figure 2.9 General scheme of respiratory control 19

Figure 2.10 Lung capacities and lung volumes 20

Figure 3.1 Typical recording of various lung volumes 26

Figure 3.2 Volume-sensing spirometric device 30

Figure 3.3 Flow-sensing spirometric devices 33

Figure 4.1 General Block diagram 37

Figure 4.2 Turbine flow meter 38

Figure 4.3 Enter Patient Data 42

Figure 4.4 How the spirometer connect with the patient 42

Figure 4.5 Patient-Related Problems 43

Figure 4.6 Schematic diagram illustrating idealized shapes of flow-volume curves and spirograms for obstructing, restrictive and mixed ventilator defects 46

Figure 5.1 General block diagram 49

Figure 5.2 LED55C and Phototransistor Circuit 50
Figure 5.3 Typical light output versus forward current 51
Figure 5.4.a Visible Red 52
Figure 5.4.b Infrared (IR) 53
Figure 5.5 Turbine Sensor 55
Figure 5.6 Phototransistor output 56
Figure 5.7 The DAQ card 57
Figure 6.1 Front Panel window 62
Figure 6.2 Block Diagram window 63
Figure 6.3 Sequence of steps in Software 65
Figure 6.4 MATLAB Sequence 70
Figure 6.5 Temperature Correction 73
Figure 6.6 Diagnosis 76
Figure 6.7 Change over time 78
Figure 7.1 LED55C & Phototransistor test circuit 80
Figure 7.2 Majed Test 81
Figure 7.3 Mashhour Test 82
Figure 7.4 Shaden Test 83
Figure 7.5 Eman Test 84
Figure 7.6 Change Over Time 85

1

Chapter One

Introduction

1.1 Introduction

1.2 Project Objectives

1.3 What is the Importance of the Project?

1.4 Literature Review

1.5 Time Plan

1.6 Economical Study

1.7 Risk management

1.8 Project Content

Chapter One
Introduction

1.1 Introduction

In physiology thus far, you have been introduced to the concept of cellular or internal respiration, which refers to the cellular metabolic processes that break down nutrient molecules, using O_2 and producing CO_2. The respiratory system of the body (lungs, airways and muscles) is not directly involved in this process; rather it is involved in the exchange of O_2 and CO_2 between the blood (brought to the alveoli in the lungs) and the inspired air (filling the alveoli in the lungs).

Respiration is composed of four steps:

1) Ventilation (or breathing).

2) Gas exchange in the lungs.

3) Circulation of blood between the lungs and tissues and.

4) Gas exchange at between the blood and tissues.

Respiratory failure is difficult to predict and can become life threatening in a few minutes. Thus continuous monitoring of respiratory activity should be mandatory in clinical and high risk situations. Several non-invasive methods and devices provide information about respiratory rate or depth, or gas exchange. Methods are categorized into: volume and tissue composition detection, air flow sensing, and blood gas concentration.

Spirometry is Pulmonary Function Tests (PFTs), Measuring lung function, specifically the measurement of the amount (volume) and/or speed (flow) of air that

can be inhaled and exhaled. Spirometry is an important tool used for assessing conditions such as asthma, cystic fibrosis, and COPD.

Since 1987 no mature change has been don on Spiro meter, Most Spiro meters are microprocessor-based devices, although a few require the user to calculate some results manually. There are two types of Spiro meters using flow sensing or volume sensing.

In this project we will use the flow sensing techniques working by turbine sensor, when patient performs specific breathing the turbine will move and cut infrared sensor a counter will count the plusses & then use a computer software to change it to volume, peak expiratory flow (PEF), functional Residual capacity (FRC), then it can be stored by software for transfer analysis and testes.

1.2 Project Objectives

The main objectives of this project are:

1- Design a non-invasive medical device that can be used for medical applications.
2- To use Flow Rate to determine Lung Volumes & Capacitances.
3- Designing a spirometer by using turbine sensing devices.
4- Interface this device to the personal computer (PC).
5- Design diagnostic software, in order to determine Lung volumes & Capacitances such as FEV, VC ...etc. Draw Spirogram and use it in clinical setting such as Obstructive lung disorders such as bronchitis and asthma, Restrictive lung disorders.

1.3 What is the Importance of the Project?

Our project is very important, because it adds a new technique in order to determine lung volumes by using flow rate through turbine sensing devices. The importance of this device comes from the following:

1. It is safe.
2. Simple to use.
3. Test can be repeated.
4. Non-invasive technique.
5. The patient doesn't require special preparation.
6. Low Cost.

1.4 Literature Review

The study of this project depends on some ideas of other projects.

The first project were done by Huda Abufarda, Fatima Battat, and Ayat Sha'aban. This project was done in Palestine Polytechnic University (PPU) to be the graduation project of this team. This project talks about Pulmonary Ventilation Monitoring and Diagnostic (PVMD), (they were use temperature sensor). PVMD use mask contains temperature sensor, fixed on the infant's mouth and nose. From this project we obtained some information about respiratory rate and some Physiological information.

The second project were done by Ali Ahmad Amro & Moayyad Al khatib This project was done in Palestine Polytechnic University (PPU) to be the graduation project of this team. This project was Designing of Respiratory Monitoring System Using Impedance Plethysmography. From this project we obtained some information about respiratory rate and some Physiological information.

1.5 Time Plan

The project follows is following time schedule, which includes the related tasks of study and system analysis.

The time planning consists of two time estimation schedules; the first one demonstrates what is done in the first semester and the second demonstrates the scheduling time of the second semester.

1.5.1 The First Time Planning

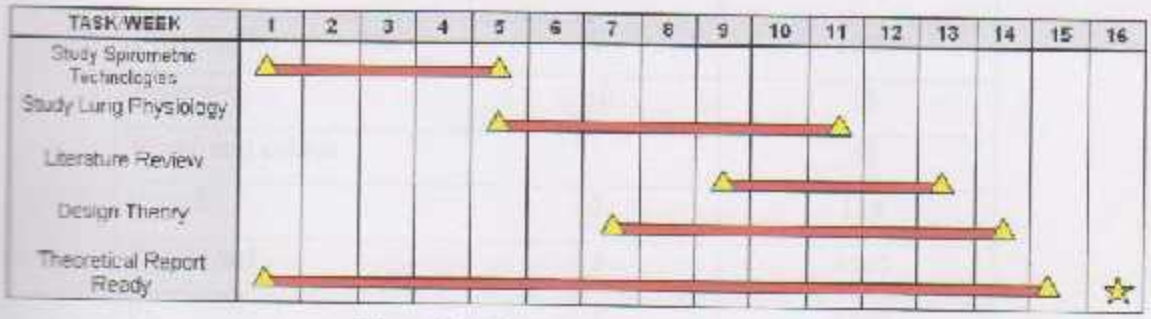


Figure1.1: The First Time Planning

1.5.2 The Second Time Planning

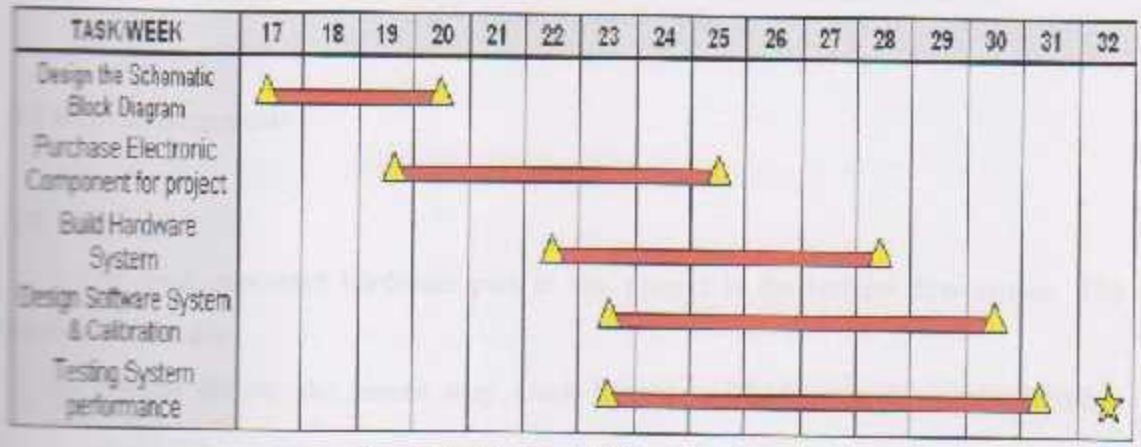


Figure1.2: The Second Time Planning

1.6 Economical Study

This project needs the following accessories that are shown in table (1.2) with their salaries:

Component	Quantity	Cost ($)
Phototransistors	2	40$
Mouthpiece with filter	2	10$
LED55C	2	10$
MATLAB student edition	1	10$
Resistors	10	1$
Wires and cables	-	5$
Board	2	15$
DAQ card	1	450$
PC	1	550$
Total cost		1091$

Table 1.1: Economical Study.

1.7 Risk management

1.7.1 Hard risk

The most important hardware part in this project is the turbine flow sensor. The predicted risks are:

- Device failure: the sensor may crash because of high voltage supply or other problems.
- Giving wrong results from sensor.

- Effect of environmental factors and high air streams would effect also reading correction.
- Wrong DAQ connection and high voltage would born it.

1.7.2 Software risk

There are two software programs to be used in this design. These are lab view and MATLAB. There may be some risks with the software such as:

- Problems in learning the lab view program.
- Make a mistake during writing the program on a m-file in MATLAB.
- Facing problems during constructing user interface.

1.7.3 Group risk

- Illness of one or more of group members.
- Group meeting difficulties.

1.7.4 Requirement risk

- Several requirements may be affected by other requirement due to the high cost of some components.
- The unavailability of DAQ for the team work to work on.

1.7.5 Project risk

- Sum requirements changes and may arise later.
- Schedule not accurate.
- Budget not sufficient.

1.8 Project Content

This report is divided into four chapters; these chapters are described as follows:

Chapter One: In this chapter we describe the importance of the project, literature review, objectives, and economical study.

Chapter Two: In this chapter we talk about respiratory system structure, mechanism of respiration, lung volumes, respiratory rate, and Lung diseases.

Chapter Three: This chapter includes Spirometer Theory & Types.

Chapter Four: This chapter describes general block diagram and the measurement of Ventilator Function.

Chapter Five: This chapter describes the Hardware Design system and the operation principle of hardware component that are used for this project.

Chapter Six: This chapter presents the flowcharts of software system, and then it talks about the software tools that used to implement the system which they are LABVIEW & MATLAB.

Chapter Seven: This chapter includes Testing the system and results of volunteer tests done by this device.

Chapter Eight: This chapter includes what we conclude and problems we face during the project, then what may be done on the device in future.

2

Chapter Two

Physiology of Respiration

2.1 Introduction

2.2 Respiratory System Structure

2.3 The Mechanism of Respiration

2.4 The factors affecting ventilation

2.5 Respiratory Control

2.6 The lung Volumes

2.7 Respiratory Rate

2.8 Diseases of the lung

Chapter Two

Physiology of Respiration

2.1 Introduction

Our cells use oxygen and produce carbon dioxide. The respiratory system brings the needed oxygen into the body and eliminates carbon dioxide from it.

Oxygen and carbon dioxide diffuse between the alveoli and pulmonary capillaries in the lungs, and between the systemic capillaries and cells throughout the body.

The diffusion of these gases, moving in opposite directions, is called gas exchange. When external respiration:

- Carbon dioxide diffuses from pulmonary capillaries into alveoli.
- Oxygen diffuses from alveoli into pulmonary capillaries.

when internal respiration:

- Oxygen diffuses from systemic capillaries into cells.
- Carbon dioxide diffuses from cells into systemic capillaries.

Gas Effect exchange depends on:

- Available surface area, which varies in different tissues.
- Partial pressure gradients
- Rate of blood flow.

The blood transports oxygen and carbon dioxide between the lung and other tissues throughout the body. These gases are carried in several different forms: dissolved in the plasma, chemically combined with hemoglobin, or converted into a different molecule. The basic rhythm of breathing is controlled by respiratory centers located in the brainstem. This rhythm is modified in response to input from sensory receptors and from other regions of the brain.

2.2 Respiratory System Structure

The organs of the respiratory system extend from the nose to the lungs and are divided into the upper and lower respiratory tracts. The upper respiratory tract consists of the nose and the pharynx, or throat. The lower respiratory tract includes the larynx, or voice box; the trachea, or windpipe, which splits into two main branches called bronchi; tiny branches of the bronchi called bronchioles; and the lungs, a pair of saclike, spongy organs. The nose, pharynx, larynx, trachea, bronchi, and bronchioles conduct air to and from the lungs. The lungs interact with the circulatory system to deliver oxygen and remove carbon dioxide.

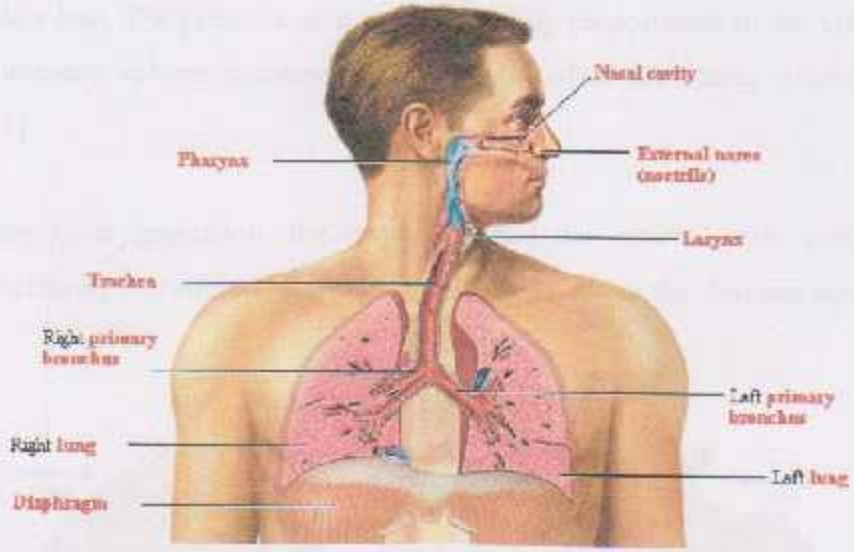


Figure 2.1: Respiratory System Structure [1].

2.3 The Mechanism of Respiration

Breathing is the exchange of air between the atmosphere and the lung, as air moves into and out of the lungs it travel from regions of high air pressure to regions of low air pressure.

The relation ship between pressure and volume by Boyle's law:

When the lung is the larger volume, the gas molecules strike the wall less frequently, thus exerting less pressure (inspiration).

When the lung is the smallest volume the gas molecules strike the wall more frequently, thus exerting more pressure (expiration).

Boyle's law: The pressure of a gas is inversely proportional to the volume when the lung increase volume decreases pressure, and when decreasing volume increases pressure [1].

During quiet inspiration, the diaphragm and the external inter costal muscles contract increasing the volume decreases the pressure within the thoracic cavity and the lung.

Figure 2.2 Quiet inspiration

During quiet expiration is a passive processing which the diaphragm and the external inter costal muscles relax, and the elastic lung and thoracic wall recoil inward.

This decreases the volume and therefore increases the pressure in the thoracic cavity.

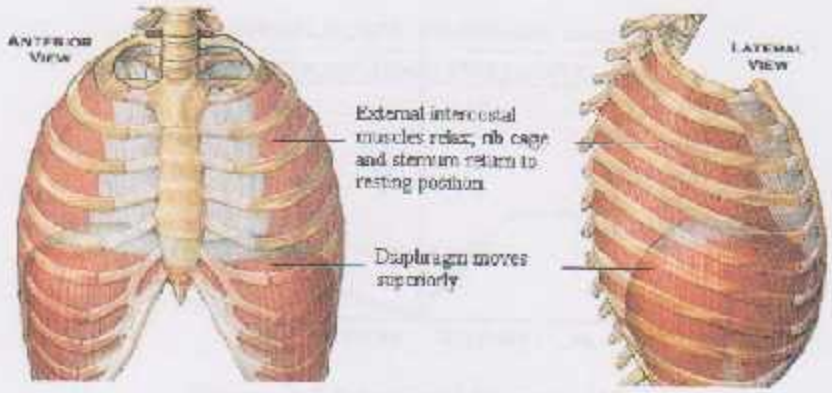


Figure 2.3: Quiet expiration

The mechanism of respiration depends on intrapulmonary pressure change and intrapleural pressure change.

- Intrapulmonary pressure change:

Intrapulmonary (intra – alveolar) pressure is the pressure within the alveoli between breaths, it equal atmospheric pressure (760 mmHg).

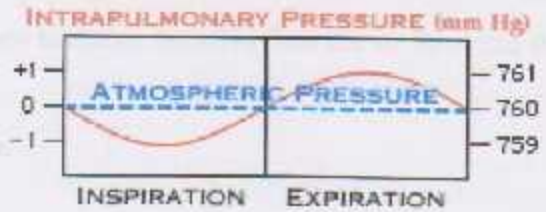


Figure 2.4:Intrapulmonary pressure [1].

- Intrapleural pressure change:

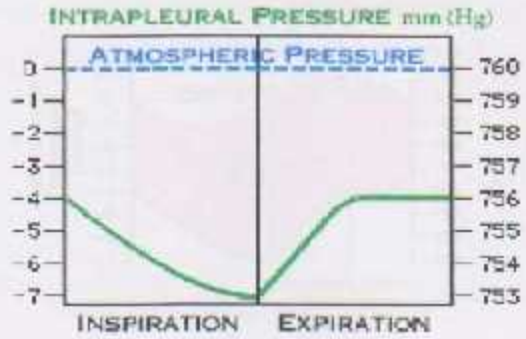


Figure 2.5 Intraplural Pressure [1].

As the thoracic wall moves out ward during inspiration, the intrapleural pressure becomes even more negative, as the thoracic wall recoils during expiration, the pressure return to (-4 mmHg) or (756 mmHg) [1].

-Event during inspiration

Diaphragm and external inter costal muscles contract then the volume of thoracic cavity increases, the intraplural pressure become more negative then the lung expand, the intrapulmonary pressure becomes negative then the air flow into the lung.

-Event during expiration

Diaphragm and external inter costal muscles relax then the volume of thoracic cavity decrease the intraplural pressure becomes less negative the lung recoil then the intrapulmonary pressure rises above atmospheric pressure then the air flows out of the lung.

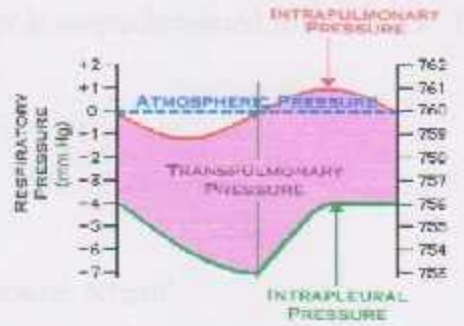


Figure 2.6 Changes of pressure [1].

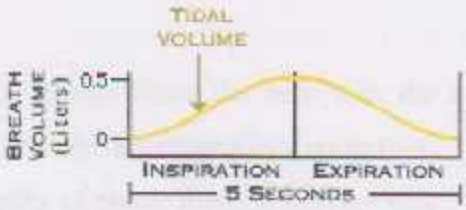


Figure 2.7 Breathe volume [1].

2.4 The factors affecting ventilation

Tow other factors play roles in ventilation:

1- Resistance within the air flow: as air flows into the lung the gas molecules encounter resistance when they strike the walls of the air way. There fore the diameter of the air way effect the resistance when the diameter decrease the resistance increase, because more gas molecules encounter the air way wall. And when the diameters increase the resistance decreases.

Correct air flow is inversely related to resistance. This relation ship is shown by the equation

$$R = P / V \quad \text{.....2.1 [2]}$$

Where R= resistance, N^*s/m^5.

P= pressure, N/m^2.

V=volume flow rate, m^3/s.

In the healthy lungs, the air way typically offers little resistance, so air flow easily into and out of the lungs.

2- lung compliance: elastic fibers the ease with the lung expand is called lung compliance is primarily determined by two factors:

- The stretch ability of elastic fibers within the lung.

Figure 2.8 Elastic fiber like balloon

- The surface tension within the alveoli.

Healthy lungs have high compliance because of their abundant elastic connective tissue.

Compliance is the ratio of lung volume to lung pressure [2]:

$$C = V / P \quad \text{.....2.2}$$

Where C –compliance, m^5/N.

V =lung volume, m^3.

P =pressure, N/m^2.

2.5 Respiratory Control

Control of respiration occurs in many different cerebral structures and regulates many things. Respiration must be controlled to produce the respiratory rhythm, ensure adequate gas exchange, protect against inhalation of poisonous substances, assist in maintenance of body PH, remove irritations, and minimize energy cost. Respiratory control is more complex than cardiac control for at least three reasons:

1- Airways airflow occurs in both directions.
2- The respiratory system interfaces directly with the environment outside the body.
3- Parts of the respiratory system are used for other functions, such as swallowing and speaking.

As a result, respiratory muscular action must be exquisitely coordinated; it must be prepared to protect it self against environmental onslaught and breathing must be temporarily suspended on demand.

All control systems require sensors, controllers, and effectors. [Figure 2.9] presents the general scheme for respiratory control.

Respiratory chemoreceptor exists peripherally in the aortic arch and carotic bodies and centrally in the ventral medulla oblongata of the brain. These receptors are sensitive to partial pressure of CO_2 and O_2 and blood PH.

The respiratory controller is located in several places in the brain. Each location appears to have its own function. Unlike the heart, the basic respiratory rhythm is not generated within the lungs but rather in the brain and is transmitted to respiratory muscles by the phrenic nerve.

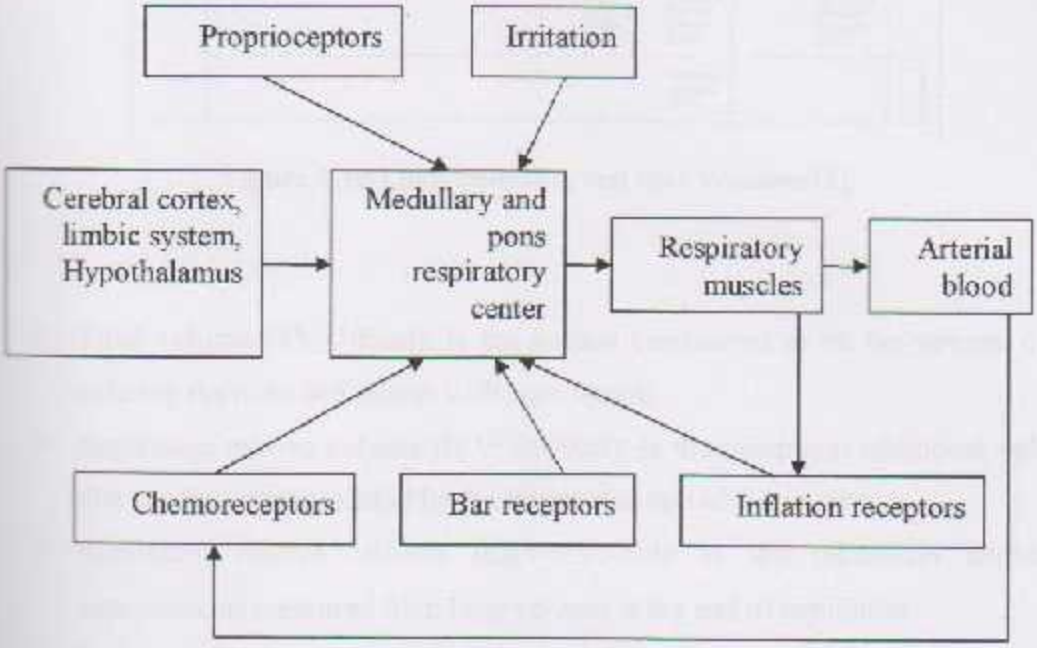


Figure 2.9 General scheme of respiratory control [2].

2.6 The lung Volumes

During respiratory function tests, the Person performs specific breathing maneuvers from which the various volume and flow parameters are determined (Fig. 2.10) [2].

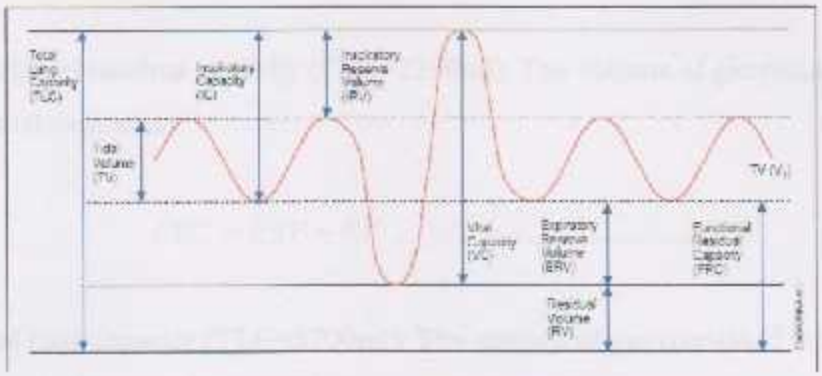


Figure 2.10 Lung capacities and lung volumes [2].

1- Tidal volume (TV=500ml): is the normal considered to be the volume of air entering the nose and mouth with each breath.

2- Inspiration reserve volume (IRV=3000ml): is the maximum additional volume that can be accommodated by the lung at the end of inspiration.

3- Expiratory reserve volume (ERV=1000ml): is the maximum additional expiration, as measured from lung volume at the end of expiration.

4- Residual volume (RV=1200ml): is the amount of gas remaining in the lungs at the end of maximal expiration.

5- Vital capacity (VC=4500ml): The maximal volume of gas that can be forcefully expelled after maximal inspiration.

$$Vc = TV + IRV + ERV \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 2.3$$

6- Inspiratory capacity (IC=3500ml): The maximal volume of gas that can be inspired from the resting expiratory level.

$$IC = TV + IRV \quad \text{.....2.4}$$

7- functional residual capacity (FRC–2200ml): The volume of gas remaining after normal expiration.

$$FRC = ERV + RV \quad \text{.....2.5}$$

8- Total lung capacity (TLC–5700ml): The amount of gas contained in the lung at the end of maximal inspiration.

$$TLC = TV + IRV + ERV + RV \quad \text{.....2.6}$$

Note: these lung volume values for a healthy person.

2.7 Respiratory Rate

The respiration rate is the number of breaths a person takes per minute. The rate is usually measured when a person is at rest and simply involves counting the number of breaths for one minute counting how many times the chest rises.
The respiratory center emits signals 12 to 20 times a minute, causing a person to take 12 to 20 breaths a minute. Newborns breathe at a faster rate, about 30 to 50 breaths a minute.

-The normal Respiratory Rates by Age:

1. Newborns: Average 44 breaths per minute.

2. Infants: 20-40 breaths per minute.
3. Preschool children: 20-30 breaths per minute.
4. Older children: 16-25 breaths per minute.
5. Adults: 14 to 18 breaths per minute [3].

2.8 Diseases of the lung

Pneumoconiosis (plural, pneumoconiosis), a general term for any one of several lung diseases caused by breathing dust from industrial occupations like coal mining, sand blasting, and stone cutting. Years of continual exposure to industrial dust can cause the formation of spots (macules), lumps (nodules), or fibrous growths in lung tissue, causing permanent damage or destruction of these tissues. Smoking can complicate or worsen the conditions. Symptoms of the disease include shortness of breath, labored breathing, coughing, and production of phlegm (mucus secreted in the respiratory system when infections are present). Other, often fatal, illnesses such as cancer, tuberculosis, emphysema, or heart disease may also develop.

Both inorganic dust (from minerals) and organic dust (from plants) can produce pneumoconiosis. For example, inhalation of inorganic irritants such as coal dust, particularly from mining hard coal, or anthracite, causes the condition known as black lung disease, coal worker's pneumoconiosis, or anthracosis. Silica dust from quarrying, mining, or sand blasting causes the disease silicosis. The fine particles and dust from asbestos, a fibrous material commonly used in construction and insulation until its use was curtailed by the Environmental Protection Agency in 1989, causes asbestosis and

mesothelioma, a cancer of the chest lining. The inhalation of organic irritants most often found in textile mills such as the dusts of cotton, flax, hemp, and jute causes byssinosis, or brown lung disease. Another type of pneumoconiosis takes the form of hypersensitivity to irritants, fumes, and vapors in the workplace from substances like cadmium, beryllium, chlorine, and fluorine [3].

3

Chapter Three

Theoretical Background

3.1 Introduction

3.2 Spirometers, Diagnostic

3.3 Spirograms

3.4 Types of Spirometers

Chapter Three

Theoretical Background

3.1 Introduction

It is the most basic and common methods to understand pulmonary lung functions. It may be useful to do the following:

1. To determine how well the lungs receive, hold, and utilize air.
2. To monitor a lung disease.
3. To monitor the effectiveness of treatment.
4. To determine the severity of a lung disease.
5. To determine whether the lung disease is restrictive (decreased airflow) or obstructive (disruption of airflow).

3.2 Spirometers, Diagnostic

Spirometers are devices used to measure the volume of air inhaled or exhaled by the lungs.

Diagnostic spirometers measure the flow and volume of gas moving into and out of the lungs during inspiratory and expiratory efforts. They help diagnose and monitor disease, as well as assess treatment effectiveness and disability. Comparing spirometric values to the patient's previous values or to

normal values for a patient of the same sex, height, weight, and/or age provides information about pulmonary function. Spirometers aid in diagnosing lung diseases, classifying obstructive and restrictive disorders, and measuring the efficacy of subsequent therapies. They also help distinguish between pulmonary anomalies and anomalies of other origin (e.g., neuralgic, chest wall) [4].

3.3-Spirograms

Graphic recordings of expirations are called Spirograms and help to visualize measurements.

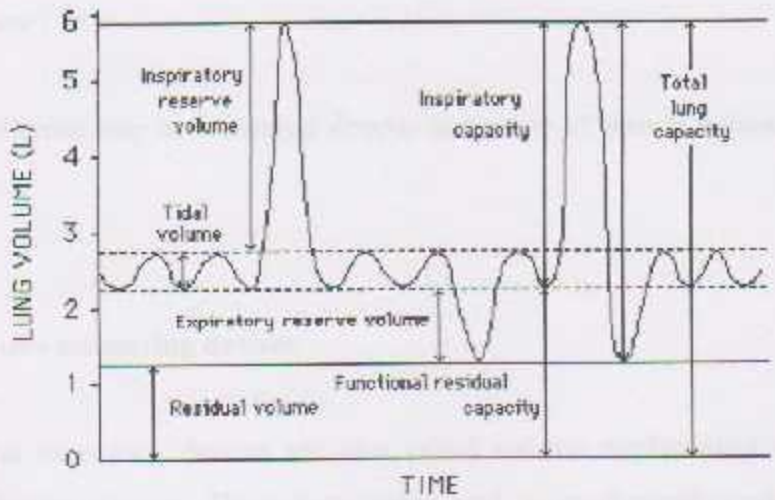


Figure 3.1 Typical recording of various lung volumes from a recording spirometer [5].

3.4 Types of Spirometers

There are broadly two types:

1. Volumetric Spirometers:

Directly measures air volume as a function of time using volume sensing .(Water-sealed, dry rolling seal and bellows spirometer).

2. Flow-Type Spirometer:

Measures airflow during a period of time and integrates the flows to get expired volume using volume sensing (pneumotachometer, turbine, hot wire anemometer spirometers)[4].

3.4.1 Volume

Gas volumes may be measured directly using one of several volume displacement *spirometers*.

3.4.1.1 Volume measuring devices

Volume measuring devices are also called volume displacement spirometers or volume collecting devices. These instruments tend to mechanically collect an expired volume of gas into a leak proof and expandable container; the expansion of the container is designed to be linear, and the container is calibrated for volume [6].

3.4.1.1.1 Water-sealed spirometer

It is the simplest and oldest design of spirometer, Uses a hollow cylinder, or bell, which is inverted and lowered into a bucket of water containing a tube for the movement of gas. The bell rises or lowers as gas moves into or out of the gas space trapped between the bell and the water. In order to prevent excess compression of the gas within the bell, earlier models used a chain and counter-weight, although newer models are designed with lightweight bells in which gas compression is not significant. A pen is often attached to the bell, which may graph the volume-time tracing on an attached *kymograph*, a rotating drum with recording paper.

Many spirometers also incorporate a linear potentiometer attached to the bell for easy electrical recording via a simple amplifier. The basic principle of operation is that the height of the bell is related to its volume by the formula for the volume of a cylinder [6].

$$V = \pi r^2 h \quad \ldots\ldots (3.1)$$

3.4.1.1.2 The Dry-seal spirometer

The bell is sealed to its base with a thin layer of latex (or some other thin and flexible material). As gas is introduced into the bell, the latex prevents its escape and forces the bell to move, as with the water-sealed spirometer. Dry-seal spirometers may be mounted horizontally, and may employ a moving piston instead of a moving bell. Manual and electrical recording are achieved as with the water-sealed spirometer [6].

3.4.1.1.3 The Bellows, or wedge, spirometer

In this device, the gas to be measured is contained within a bellows whose expansion is recorded via a pen or a rotational potentiometer [6].

3.4.1.2 Advantages of volume displacement Spirometers

1- Simple construction and use. They do not require computers or processors for simple volume and time measurements.
2- They are easy to calibrate and do not depend on the composition of gases they are used to measure [7].

3.4.1.3 Disadvantages of volume displacement Spirometers

1- They are bulky.Water-sealed spirometers, in particular, can be heavy when they are filled with water, and they are prone to spillage when tipped.
2- They have a limited frequency response and are not well suited to rapidly changing signals
3- The maximum volume they can measure is limited by their size. Thus, for an experiment in which tidal volume is measured over a period of five minutes, the volume displacement spirometer would be difficult [7].

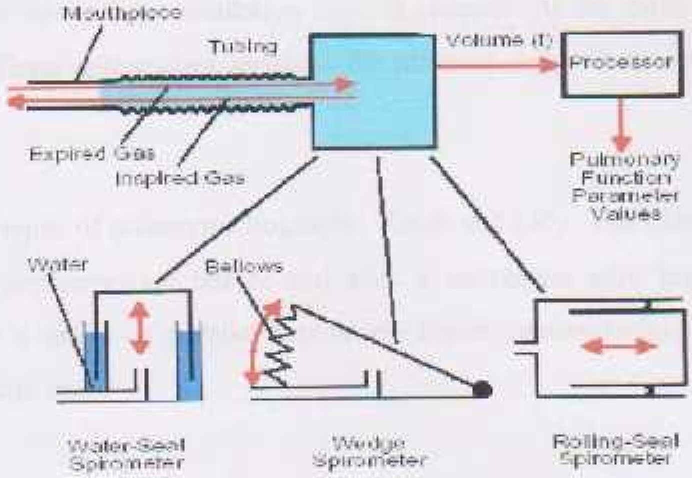


Figure 3.2 Volume-sensing spirometric device [4].

3.4.2 Flow Measuring Devices

Flow is the time-derivative of volume, or

$$F = dv/dt \qquad (3.2)$$

Thus, any device capable of measuring either volume or flow can also report the other, given an appropriate time measurement and the needed processing. For this reason, and others given below, flow measuring devices have become popular methods for measuring both volumes and Flows [7].

3.4.2.1 Pneumotachograph

Pneumotachographs measure the flow according to the Venturi principle. The Venturi principle is the name that was given to the physical phenomenon where gas

particles accelerate when their circulation zone is reduces. At the same time a drop in pressure occurs. These spirometers measure the pressure drop when a patient blows in the device.

There are 2 types of pneumotachographs: Fleish and Lilly. The Lilly type measures the difference in pressure over before and after a membrane with known resistance. Fleisch types use a series of parallel capillaries. Fleish pneumotachographs are more reliable than the Lilly ones.

The biggest disadvantage of this method is that they are very sensitive to temperature, humidity and athmospheric pressure of surrounding air. This means that these spirometers must be calibrated very often: at least daily and after each displacement.

Pneumotachographs without thermostat are not reliable, since they don't hold the change of temperature into account, which is a very important influence on results [8].

3.4.2.2 Hot Wire Anemometer

These spirometers measure the electronic resistance through a hot wire. This resistance is dependant on the temperature of the wire. Temperature in the wire drops when the patient blows air in the spirometer.

These spirometers are not very reliable and do not know the direction of the flow (inspiration ⇔ expiration).

Results are not very precise and calibration is difficult and must be done very often (at least once daily)[6].

3.4.2.3 Sonic Devices

There are two principles of airflow measurement that are related to the production and transmission of sound waves: the vortex principle and the ultrasonic principle.

The sonic principles are not affected by humidity, temperature, gas composition, or viscosity and are ideally suited for monitoring tidal volumes and minute ventilation during continuous mechanical ventilation [6].

3.4.2.4 The Ultrasonic Principle

The ultrasonic principle differs from the vortex principle because there are no struts in the airflow sensor and no vortices are produced. In the ultrasonic airflow sensor, there is an ultrasonic transmitter at one end and a receiver at the other end. The speed of the sound transmission is measured because airflow predictably affects the speed of the ultrasonic wave from transmitter to receiver; ultrasound transit time is decreased with gas flow and increased against flow. The ultrasonic principle is applied in the Perkin-Elmer ventilation monitor [8].

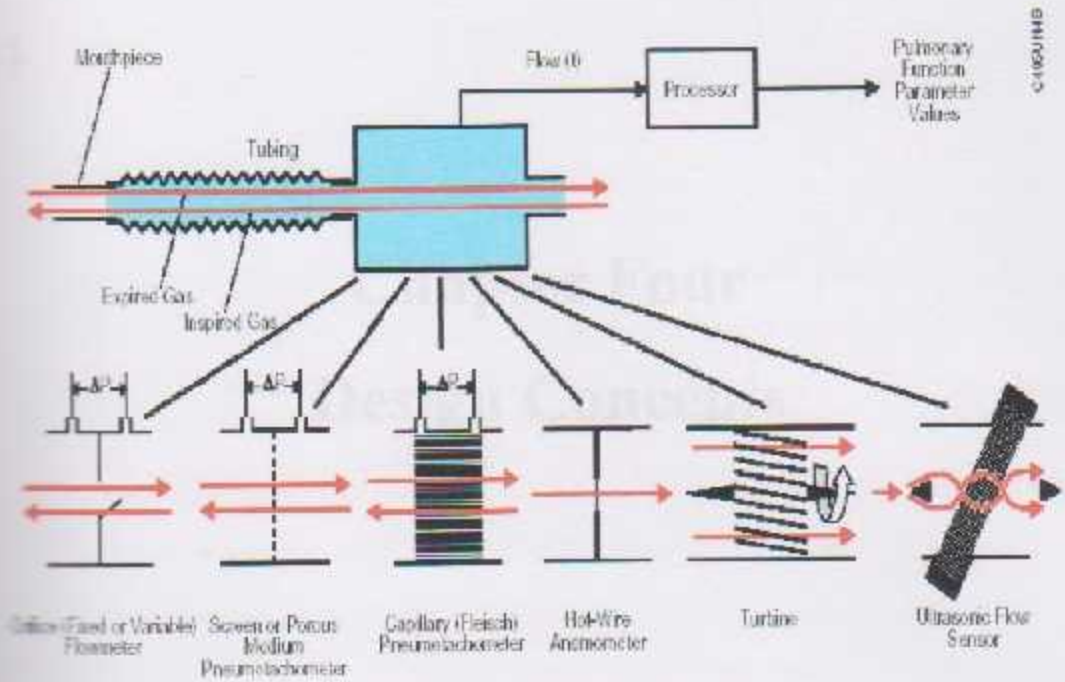


Figure 3.3 - Flow-sensing spirometric devices [4].

Sonic sensor can determine whether there is respiration or not while Turbine sensor determine lung capacity and flow rate.

4

Chapter Four
Design Concepts

4.1 Measurement of Ventilator Function

4.2 General Block Diagram

4.3 The Technique - How to do the spirometer?

4.4 Predicted Normal Values

4.5 Interpretation of Ventilator Function Tests

Chapter Four

Design Concepts

Many types of spirometer where clarified in last chapter. That used for measure timed expired and inspired volumes, and from these we can calculate how effectively and how quickly the lungs can be emptied and filled.

In this project will be focused on one type of these spirometers only. This depends on turbine flow sensor.

This chapter describes designing of a spirometer device. It describes a general block diagram, how to use this device, how to diagnose cases and determine the mistakes which could happen through the examination process, from the patient or the device.

4.1 Measurement of Ventilator Function

Alternatively, measures of flow can be made either absolutely (e.g. peak expiratory flow) or as a function of volume, thus generating a flow-volume curve the shape of which is reproducible for any individual but varies considerably between different lung diseases.

A poorly performed manoeuver is usually characterized by poor reproducibility. The measurements which are usually made are as follows:

1. **VC** (vital capacity) is the maximum volume of air which can be exhaled or inspired during either a forced (FVC) or a slow (VC) manoeuvre.

2. **FEV1** (forced expired volume in one second [9]) is the volume expired in the first second of maximal expiration after a maximal inspiration and is a useful measure of how quickly full lungs can be emptied.

3. **FEV1/VC** is the FEV1 expressed as a percentage of the VC or FVC (whichever volume is larger) and gives a clinically useful index of airflow limitation.

4. **FEF25-75%** is the average expired flow over the middle half of the FVC manoeuvre and is regarded as a more sensitive measure of small airways narrowing than FEV1. Unfortunately FEF25-75% has a wide range of normality, is less reproducible than FEV1, and is difficult to interpret if the VC (or FVC) is reduced or increased.

5. **PEF** (peak expiratory flow [9]) is the maximal expiratory flow rate achieved and this occurs very early in the forced expiratory manoeuvre.

6. **FEF50% and FEF75%** (forced expiratory flow at 50% or 75% FVC) is the maximal expiratory flow measured at the point where 50% of the FVC has been expired (FEF50%) and after 75% has been expired (FEF75%). Both indices have a wide range of normality but are usually reproducible in a given subject provided the FVC is reproducible.

4.2 General Block Diagram

There are important criteria's engineers take them in consideration in design spirometer device such as:

- Safely of user and patient
- Easiness in usage and connection with the patient
- Legality & ethics.

- Biocompatibility.
- Easiness of marketing.
- Cost.
- Adaptability issue to face.

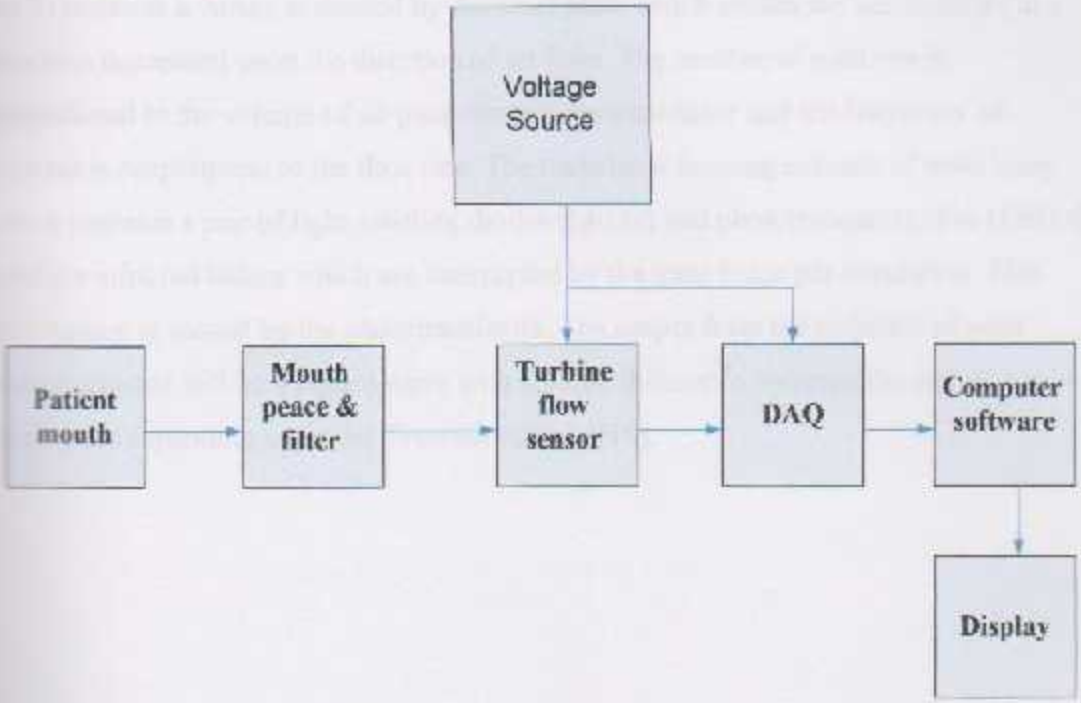


Figure 4.1: General Block diagram

- **Patient mouth:**

During spirometric pulmonary function test patient must exhale / inhale to maximum lung capacity. Usually nose clips to prevent inhalation / exhalation through nose.

- **Mouth peace:**

A tube with filter to avoid infection when using by multiple patient

- **Turbine sensor**:

This transducer consist of an acrylic tube with a vane position between two swirl plates the low internal van is attached to a stainless steel pivot which is free to rotate on two jeweled bearings mounted at the center of the swirl plates. As air is passed through the Transducer a vortex is created by the swirl plate which causes the van to rotate in a direction dependent upon the direction of air flow. The number of rotations is proportional to the volume of air pass through the transducer and the frequency of rotation is proportional to the flow rate. The transducer housing consists of main body which contains a pair of light emitting diodes (LED's) and phototransistors. The (LED's) produce infra red beams which are interrupted by the vane twice per revolution. This interruption is sensed by the phototransistors. The output from the collector of each phototransistor will be a square wave with a phase difference between the two of + or – 90 degrees depending upon the direction of flow [10].

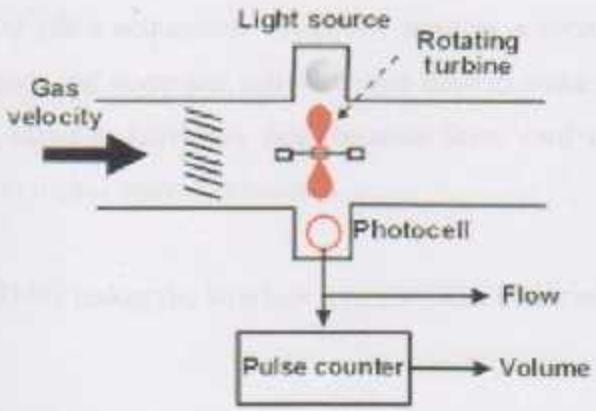


Figure 4.2 Turbine flow meter [10].

Table 4.1 Comparison Between Thermister and Turbine spirometers:

	Thermister	Turbine
Principle of operation	Measure the electronic resistance through a hot wire	Measure number of pulses
Ability	Determine if there is respiration only .	Determine Lung capacity, Flow rate.
Calibration	Difficult Calibration	Easy Calibration
Reliability	not very reliable and do not know the direction of the flow (inspiration ⇔ expiration).	Reliable and can determine the direction of the flow
Precision	not very precise	Very precise

Voltage source:

Will be used the voltage source by battery +5V

- **DAQ:**

DAQ (data acquisition System) used as a connecting process between flow sensor and computer software, also used to make the project construction process easier in hardware side, because these card contains a counters and analog to digital converter inside it.

So that DAQ makes the interface process much easier and saves time

- **Computer software:**

Two programs will be used LABVIEW and MATLAB: LABVIEW used for taking data, then save it into a Text file and show an initial curve before processing operation. After showing this curve we can determine whether it acceptable or not.

If not the test should be repeated, but if it is accepted the file will be loaded in the next program (MATLAB), where the processing operation is done then displaying the results and compare it with the original results taken through certain equations then give the final result which determine weather the patient normal or not.

- **Display:**

  The final results and curves displayed on a monitor then they can be printed by a printer.

The Technique - How to do the spirometer?

To ensure an acceptable result, the FVC manoeuvre must be performed with maximum effort immediately following a maximum inspiration; it should have a rapid start and the spirogram should be a smooth continuous curve.

To achieve good results, carefully explain the procedure to the patient, ensuring that he/she is sitting erect with feet firmly on the floor (the most comfortable position,

though standing gives a similar result in adults, but in children the vital capacity is often greater in the standing position) (see Figure 4.3). Apply a nose clip to the patient's nose (this is recommended but not essential) and urge the patient to: · breathe in fully (must be absolutely full); then the result display on the screen.

- Seal his/her lips around the mouthpiece
- Blast air out 'as fast and as far as you can until the lungs are completely empty.
- Breathe in again as forcibly and fully as possible (if inspiratory curve is required).

If only peak expiratory flow is being measured then the patient need only exhale for a couple of seconds. Essentials are:

- To breathe in fully (must be absolutely full).
- A good seal on the mouthpiece.
- Very vigorous effort right from the start of the manoeuvre and continuing until absolutely no more air can be exhaled.
- No leaning forward during the test.

Figure 4.3 Inter Patient Data

Figure 4.4 How the spirometer connect with the patient.

4.3.1 Patient-Related Problems

The most common patient-related problems when performing the FVC manoeuvre is see Figure 4.2:

1. Sub maximal effort
2. Leaks between the lips and mouthpiece
3. Incomplete inspiration or expiration (prior to or during the forced manoeuvre)
4. Hesitation at the start of the expiration
5. Cough (particularly within the first second of expiration)
6. Glottis closure
7. Obstruction of the mouthpiece by the tongue
8. Vocalization during the forced manoeuvre
9. Poor posture.

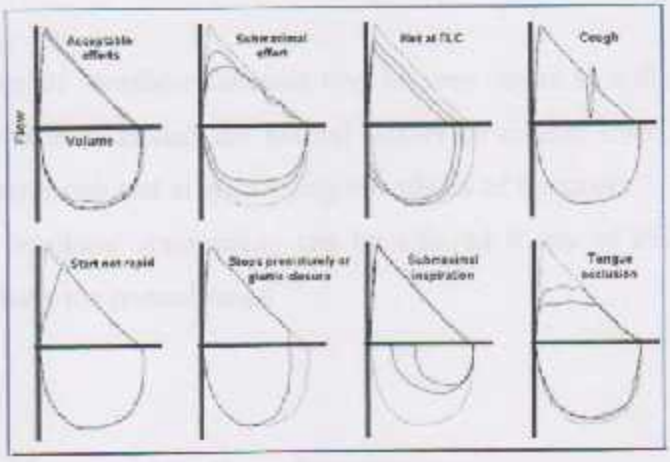


Figure 4.5 Patient-Related Problems [9].

4.3.2 Instrument-Related Problems

These depend largely on the type of spirometer being used. On volume-displacement spirometers look for leaks in the hose connections; on flow-sensing spirometers look for rips and tears in the flow head connector tube; on electronic spirometers be particularly careful about calibration, accuracy and linearity. Standards recommend checking the calibration at least daily and a simple self-test of the spirometer is an additional, useful daily check that the instrument is functioning correctly.

4.4 Predicted Normal Values

To interpret ventilator function tests in any individual, compare the results with reference values obtained from a well defined population of normal subjects matched for sex, age, height and ethnic origin and using similar test protocols; and carefully calibrated and validated instruments [appendix B].

4.5 Interpretation of Ventilator Function Tests

Measurements of ventilator function may be very useful in a diagnostic sense but they are also useful in following the natural history of disease over a period of time, assessing preoperative risk and in quantifying the effects of treatment.

The presence of ventilator abnormality can be inferred if any of FEV1, VC, PEF or FEV1/VC% is outside the normal range.

4.5.1 Classifying Abnormal Ventilator Function

The inter-relationships of the various measurements are also important diagnostically (Figure 4.5 and Table 4.1) [9]. For example:

1. A reduction of FEV1 in relation to the forced vital capacity will result in a low FEV1/FVC% and is typical of obstructive ventilator defects (e.g. asthma and emphysema). The lower limit of normal for FEV1/FVC is around 70-75% but the exact limit is dependent on age. The exact values by age, sex and height are given in the tables in Appendix C. In obstructive lung disease the FVC may be less than the slow VC because of earlier airway closure during the forced manoeuvre. This may lead to an overestimation of the FEV1/FVC%. Thus, the FEV1/VC% may be a more sensitive index of airflow obstruction.

2. The FEV1/FVC% ratio remains normal or high (typically > 80%) with a reduction in both FEV1 and FVC in restrictive ventilator defects (e.g. interstitial lung disease, respiratory muscle weakness, and thoracic cage deformities such as kypho-scoliosis).

3. A reduced FVC together with a low FEV1/FVC% ratio is a feature of a mixed ventilator defect in which a combination of both obstruction and restriction appear to be present, or alternatively may occur in airflow obstruction as a consequence of airway closure resulting in gas trapping, rather than as a result of small lungs. It is necessary to measure the patient's total lung capacity to distinguish between these two possibilities.

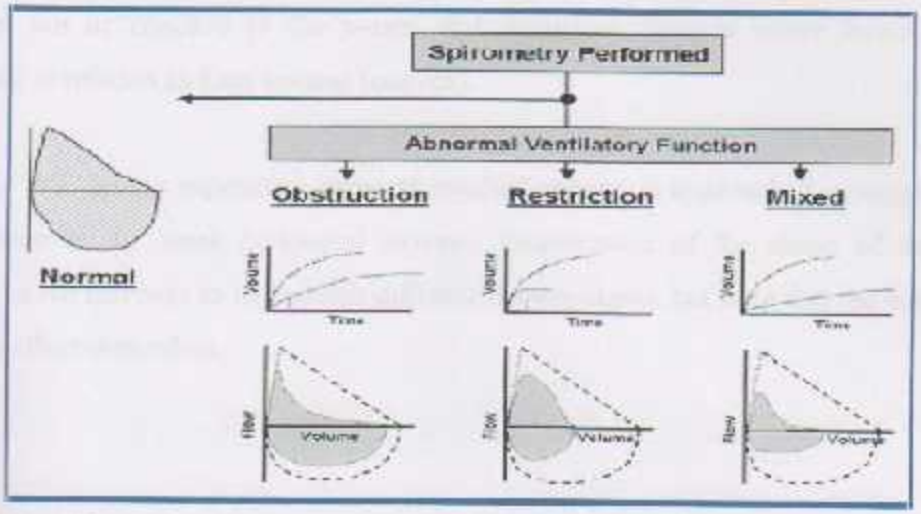


Figure 4.6 Schematic diagram illustrating idealized shapes of flow-volume curves and spirograms for obstructing, restrictive and mixed ventilator defects..

	OBSTRUCTIVE	RESTRICTIVE	MIXED
FEV_1	↓	↓ or Normal	↓
FVC	↓ or Normal	↓	↓
FEV_1/FVC	↓	Normal or ↑	↓

Table 4.2 Classification of Ventilator Abnormalities by Spirometry [9]

The shape of the expiratory flow-volume curve varies between obstructive ventilator defects where maximal flow rates are diminished and the expiratory curve is

scooped out or concave to the x-axis, and restrictive diseases where flows may be increased in relation to lung volume (convex).

A "tail" on the expiratory curve as residual volume is approached is suggestive of obstruction in the small peripheral airways. Examination of the shape of the flow-volume curve can help to distinguish different disease states, but note that the inspiratory curve is effort-dependent.

5

Chapter Five

Hardware System Design

5.1 Structural design

5.2 Turbine Flow sensor

5.3 DAQ (Data Acquisition System)

Chapter Five

Hardware system design

5.1 Structural design:

The following is a block diagram Of the complete system:

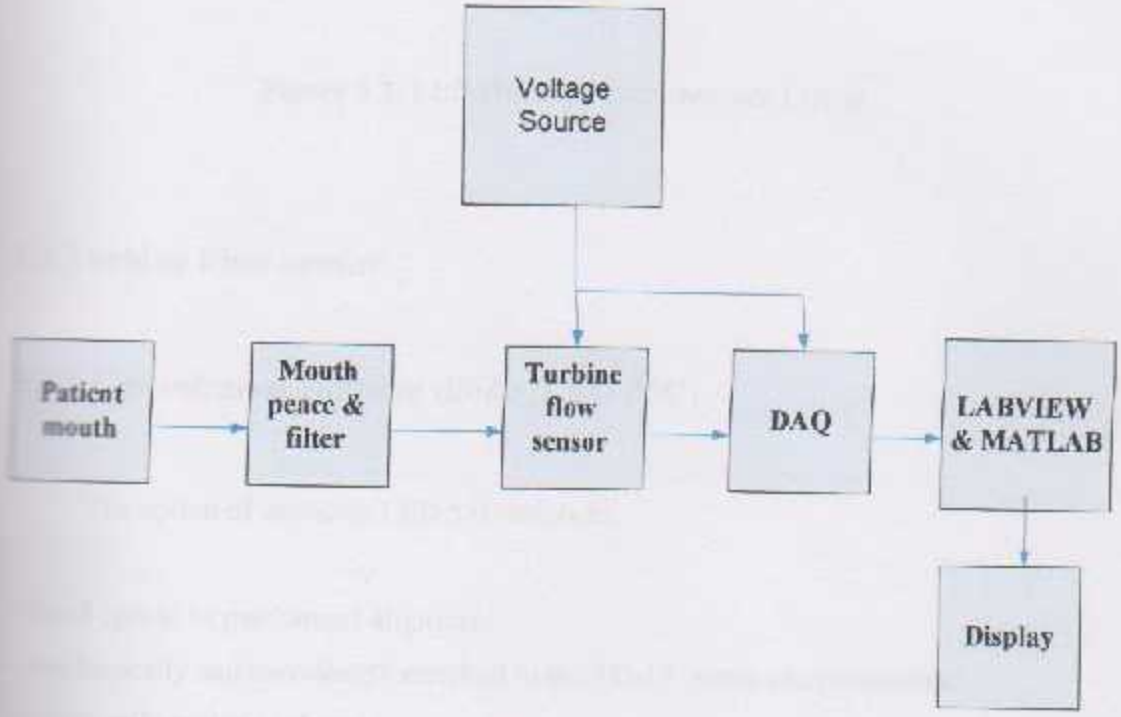


Figure 5-1: General block diagram

In previous block diagram noticed that this device contains many parts such as

- Turbine flow sensor which has been constructed on single board as shown in figure (5-2).

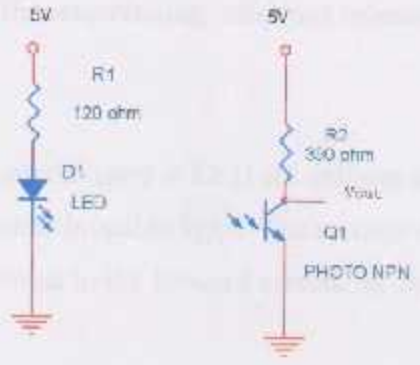


Figure 5.2: LED55C and Phototransistor Circuit

5.2 Turbine Flow sensor

5.2.1 Gas infrared emitting diode (LED 55C)

The option of choosing LED 55C refers to:

- Good optical to mechanical alignment.
- Mechanically and wavelength matched to the TO-18 series phototransistor .
- Hermetically sealed package.
- High irradiance level

The basic operation of the light emitting diode (LED) is as follow: the device operate as the forward biased, the electrons cross the pn junction from the n-type

material and recombine with holes in the p-type material. The free electrons are in the conduction band and a higher energy level than the holes in the valence band. When recombination takes place, the recombining electrons release energy in the form of light.

The semiconductor material used in LED are gallium arsenide (GaAs), that emit infrared (IR) radiation, which is invisible light. The amount of power output translated into light is directly proportional to the forward current, as indicated in figure (5-3).

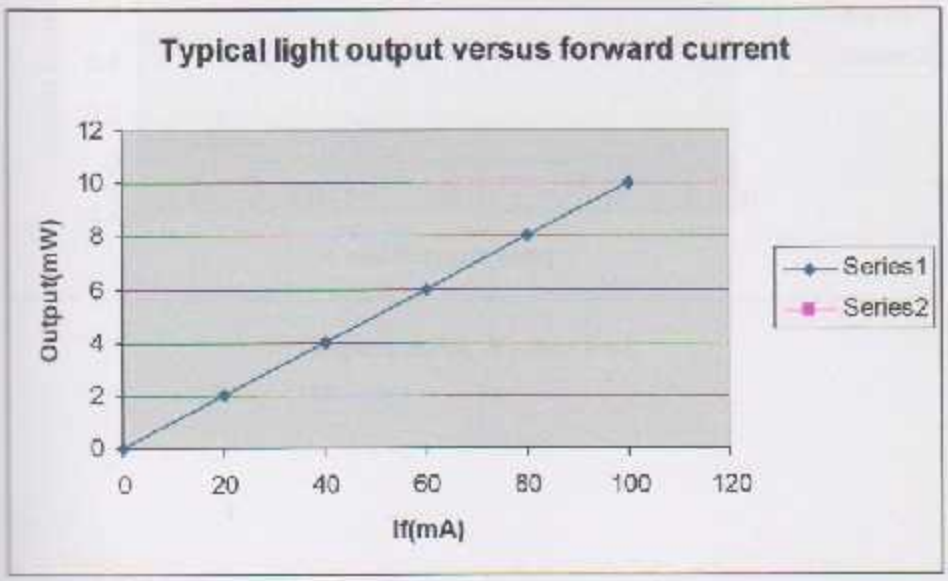


Figure 5.3: Typical light output versus forward current.

The wavelength of light determines whether it is visible or infrared. An LED emits light over a specified range of wavelengths as indicated by the spectral output

curves in figure (5-4). The curve in part (a) represents the light output versus wavelength for a typical visible red LED and the curve in part (b) is for typical infrared LED. The wavelength (λ) is expressed in nanometer (nm). The output of the visible red LED peaks at 660 nm, and output for the infrared LED peaks at 940 nm.

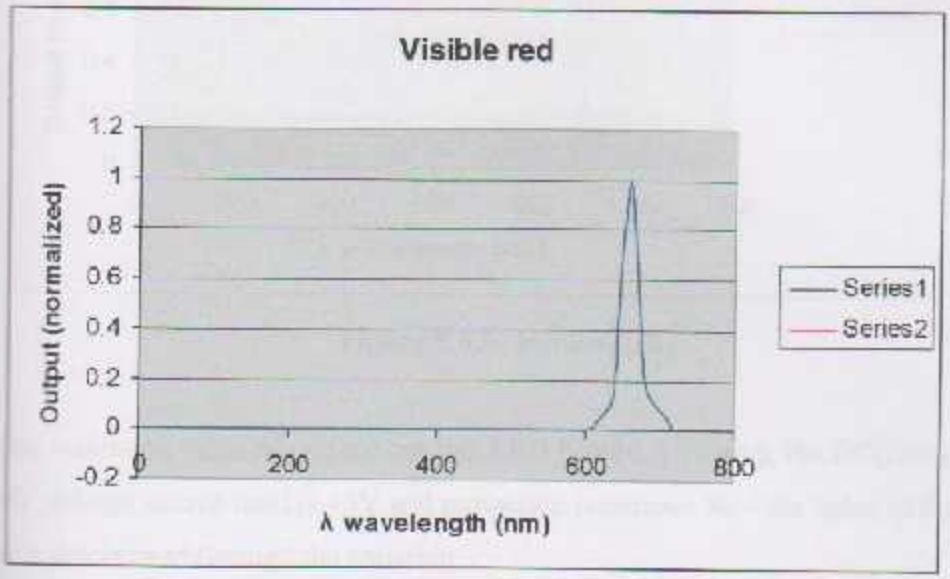


Figure 5.4.a. Visible Red.

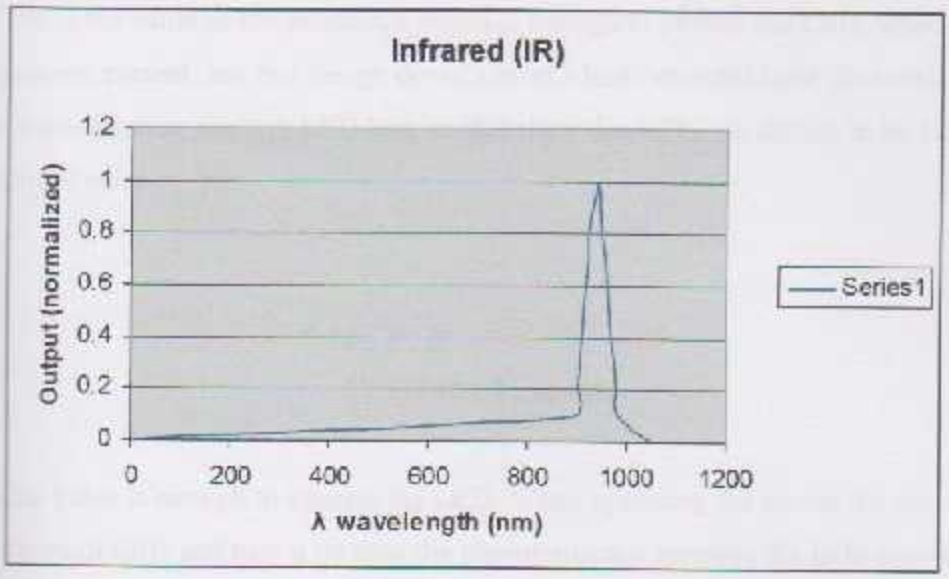


Figure 5.4.b: Infrared (IR).

The maximum value of current can this LED handle is 100mA, the DC(Direct Current) voltage source used is +5V and protection resistance R_1 – the value of the resistance determined through the equation:

$$R = (Vcc - Vd) / I$$
$$R = (5 - 1.7) / 100m = 33\Omega$$

Where are:

R: is a protection resistance (ohm)

V: is a DC voltage source (volt)

I: is a current passing through the LED (mA)

This is the value of the resistance which is enough to protect the LED, when using the maximum current, but this design doesn't need a high intensity light that makes the current value passing through LED less, so that the value of R_1 is chosen to be 120Ω for a current value :

$$I = V / R$$
$$I = 5/120 = 41.66mA$$

This value is enough to operate the LED. When operating the circuit the current passes through LED and turn it on then the phototransistor receives the light coming from the LED, which is the other part of the circuit.

5.2.2 Phototransistor(SD5600)

Phototransistors are solid state light detectors that possess in internal gain. This makes them much more sensitive than photodiodes of comparably sized area. These devices can be used to provide either an analog or digital output signal. This family of detectors offers the option of choosing phototransistor refers to.

- Low cost visible and near-IR photo detection
- Available with gains from 100 to over 1500.
- Moderately fast response times.
- Available in a wide range of packages including epoxy coated, transfer molded, cast, hermetic packages, and in chip form.

- Usable with almost any visible or near infrared light source such as IRED, neon, fluorescent, incandescent bulbs, lasers, flame source, sunlight, etc.
- Same general electrical characteristics as familiar signal transistors (except that incident light replaces base drive current).

A phototransistor has been used in this project as shown in figure (5.3).

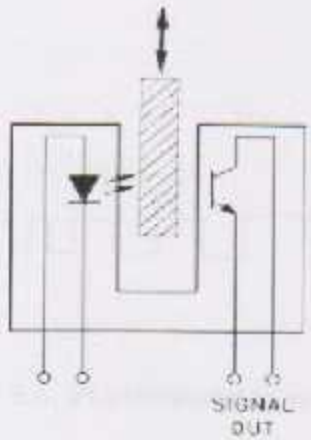


Figure 5.5 : Turbine Sensor

- For Phototransistor circuit the resistance is chosen as the following equation

$R = V / I$

$R = 5/15m = 333\Omega$

For more protection resistance value were used is 390Ω

R is a protection resistance (ohm)

V is a DC voltage source (volt)

I: is a current passing through the Phototransistor (mA)

The principle of operation:

Fan plates is moved between LED and phototransistor by the air out from patient lung when plate interrupts light it records pulse, so result should be like this figure (5.4).

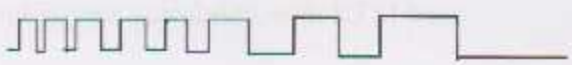


Figure 5.6: Phototransistor output

This signal inters DAQ, then computer software to be processed.

5.3 DAQ (Data Acquisition System):

The DAQ (NI_PCI 6034E)

Figure5.7: The DAQ card.

The NI 6034 DAQ uses the E series technology to deliver high performance, reliable data acquisition capabilities. This card is used in different applications such as:

- Continuous high-speed data logging at up to 200kS/s (Kilo Sample/second).
- Externally timed and/or triggered data acquisition.
- High-voltage and sensor measurements (when used with signal conditioning).
- High-channel-count system scalability with RTSI bus.

The NI 6034E card has many important features such as:

- This card has 16 analog inputs at 200kS/s, with 16 bit resolution.
- 2 analog outputs, each with 16-bit resolution.
- 8 digital input/output lines which are compatible with both 5V TTL(Transistor Transistor Logic) and CMOS(Complementary metal–oxide–semiconductor).
- Two 24-bit counter/timer, with 20MHz frequency.
- Digital triggering.
- 4 analog input signal ranges.

5.3.1 DAQ specifications and requirements

In this project the phototransistor sensor output are analog signal, that are square wave signal and that signal must be summed in two ways using Pulse edge counter and Frequency counter then two analog input channels DAQ is required, for the frequency resulted is about 1kHz , so the DAQ frequency should be at its minimum double the highest frequency present ; but any DAQ in these days has at least more than 1 KHz as its sampling frequency.

So for monitoring these signals the 6034 NI-DAQ have been used, the National Instruments 6034 device is high-performance multifunction analog, digital, and timing I/O device for PCI. The NI 6034 features 16 channels (eight differential) of 16-bit analog input (AI), a 68-pin connector, and eight lines of digital I/O (DIO).

So when using this DAQ, two channels from the differential analog input are used for the one for volume and the other for flow rate.

Finally for monitoring stage a Personal Computer PC is needed to mount the DAQ on it and to run the monitoring program, which is LabView 7.1 or later.

6

Chapter Six

Software System Design

6.1 The "LabVIEW" software

6.2 MATLAB

Chapter Six

Software system design

This chapter describes the types of programs which have been used through this project and description for each one with its flowchart.

6.1 The "LabVIEW" software:

The "LabVIEW" which is an abbreviation of (Laboratory Virtual Instrument Workbench) is a full-featured graphical programming language with all the standard features of a general-purpose programming environment such as data structures, looping structures, and event handling. LabVIEW also has a built-in compiler that compiles all code at edit time. However, unlike other general-purpose programming languages, LabVIEW is specifically designed for engineers and scientists and has built-in tools to meet their needs. These high-level functions, assistants, and tools make LabVIEW much more than a programming language.

National Instruments (NI) LabVIEW is a graphical development environment for creating flexible and scalable design, control, and test applications rapidly and at minimal cost. With NI LabVIEW, engineers and scientists interface with real-world signals, analyze data for meaningful information, and share results through intuitive displays, reports, and the Web. Regardless of experience, LabVIEW makes development fast and easy for all users.

National Instruments LabVIEW delivers a powerful graphical development environment for signal acquisition, measurement analysis, and data presentation, giving

the flexibility of a programming language without the complexity of traditional development tools, and it also uses dataflow programming, where the flow of data through the nodes on the block diagram determines the execution order of the VIs and functions. (VIs are LabVIEW programs that imitate physical instruments).

So the LabVIEW can perform the following functions:

- Acquiring: NI LabVIEW is an open environment designed to make interfacing with any measurement hardware simple. With interactive assistants, code generation, and connectivity to thousands of devices, LabVIEW makes gathering data as simple as possible.

- Analyzing: LabVIEW has more than 500 built-in functions designed specifically for extracting useful information from any set of acquired data and for analyzing measurements and processing signals.

Presenting: LabVIEW provides tools for data visualization, user interface design, Web publishing, report generation, data management, and software connectivity.

6.1.1 VI

VI is a program in LabVIEW that models the appearance and function of a physical instrument is combined from three major components:

1. Front Panel: acts as a user interface.
2. Block Diagram: contains the graphical source code that defines the functionality of the VI.

3. Icon and Connector pane: identifies the VI.

6.1.1.1 Front Panel

Front Panel is the user interface of the VI, it contains two components controls and indicators, which are the interactive input and output terminals of the VI respectively.

Controls such as knobs, push buttons, dials, and other input devices, they simulate instrument input devices and supply data to the block diagram of the VI.

Indicators such as graphs, LEDs, and other displays, they simulate instrument output devices.

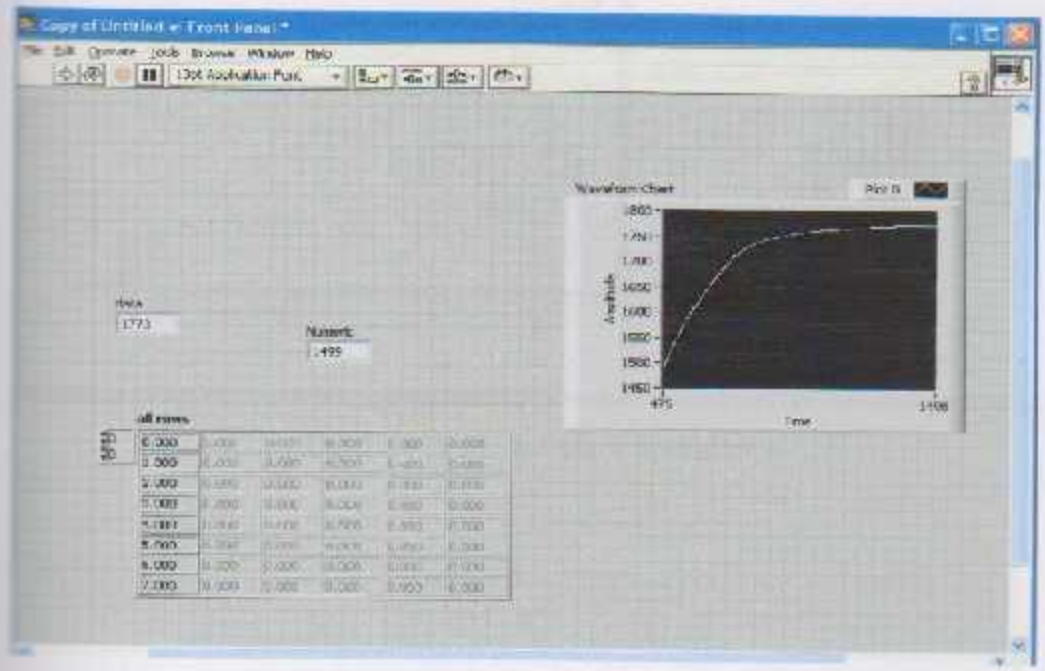


Figure 6.1: Front Panel window

6.1.1.2 Block Diagram

Block Diagram is a window holds the graphical source code of the LabVIEW where Front Panel objects appears as terminals on the block diagram. Also, it contains functions and structures from built-in LabVIEW VI libraries. Wires connect each of the nodes on the block diagram, including control and indicator terminals, functions and structures.

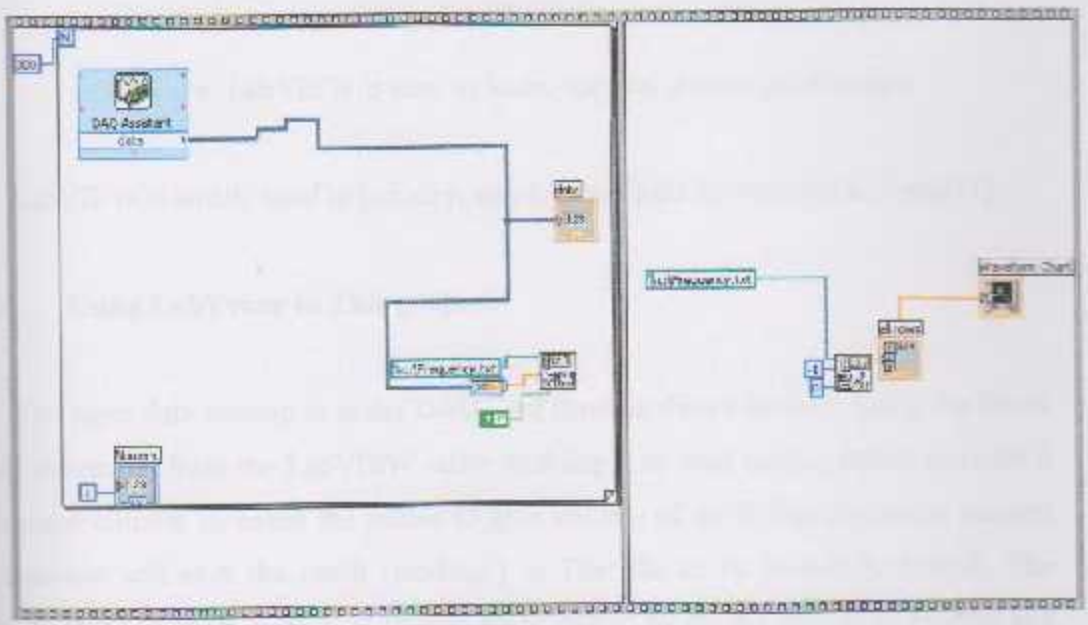


Figure 6.2: *Block Diagram window*

6.1.2 Why use LabVIEW

LabVIEW is easy to use, interactive, so we can write sophisticated programs and applications in a short time.

In addition, LabVIEW presents the following benefits:

- LabVIEW provides open connectivity with other programs such as,C++, Excel ... etc.

- LabVIEW is easy to learn, use and gives a great results.

LabVIEW is widely used in industry, employable skills for students to learn[11].

6.1.3 Using LabVview in This project:

The input data coming in at the DAQ card from hardware devices. Using the Block "DAQ Assistant" from the LabVIEW -after enabling it to read analog input-, once let it Edge pulse counter to count the pulses to give volume of air during expiration process and Labview will save the result (reading) in Text file to be loaded by Matlab. The Second will be Frequency counter to give Flow rate of air during expiration process and Labview will save the result (reading) in another Text file to be loaded by Matlab. The sequence of Project stages will be as describes in the flow chart(Figure 6.3):

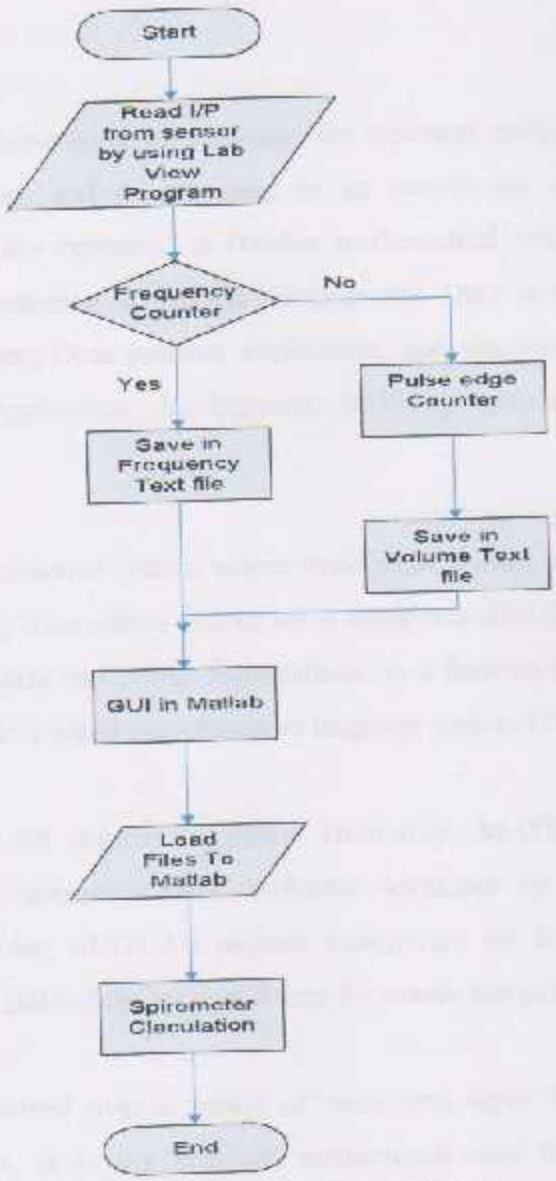


Figure 6.3 : Sequence of steps in Software

MATLAB is a high-performance language for technical computing. It integrates computation, visualization, and programming in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation. Typical uses include Math and computation Algorithm development Data acquisition Modeling, simulation, and prototyping Data analysis, exploration, and visualization Scientific and engineering graphics Application development, including graphical user interface building.

MATLAB is an interactive system whose basic data element is an array that does not require dimensioning. This allows you to solve many technical computing problems, especially those with matrix and vector formulations, in a fraction of the time it would take to write a program in a scalar noninteractive language such as C or Fortran.

The name MATLAB stands for matrix laboratory. MATLAB was originally written to provide easy access to matrix software developed by the LINPACK and EISPACK projects. Today, MATLAB engines incorporate the LAPACK and BLAS libraries, embedding the state of the art in software for matrix computation.

MATLAB has evolved over a period of years with input from many users. In university environments, it is the standard instructional tool for introductory and advanced courses in mathematics, engineering, and science. In industry, MATLAB is the tool of choice for high-productivity research, development, and analysis.

MATLAB features a family of add-on application-specific solutions called toolboxes. Very important to most users of MATLAB, toolboxes allow you to learn and

6.2 MATLAB:

MATLAB is a high-performance language for technical computing. It integrates computation, visualization, and programming in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation. Typical uses include Math and computation Algorithm development Data acquisition Modeling, simulation, and prototyping Data analysis, exploration, and visualization Scientific and engineering graphics Application development, including graphical user interface building.

MATLAB is an interactive system whose basic data element is an array that does not require dimensioning. This allows you to solve many technical computing problems, especially those with matrix and vector formulations, in a fraction of the time it would take to write a program in a scalar noninteractive language such as C or Fortran.

The name MATLAB stands for matrix laboratory. MATLAB was originally written to provide easy access to matrix software developed by the LINPACK and EISPACK projects. Today, MATLAB engines incorporate the LAPACK and BLAS libraries, embedding the state of the art in software for matrix computation.

MATLAB has evolved over a period of years with input from many users. In university environments, it is the standard instructional tool for introductory and advanced courses in mathematics, engineering, and science. In industry, MATLAB is the tool of choice for high-productivity research, development, and analysis.

MATLAB features a family of add-on application-specific solutions called toolboxes. Very important to most users of MATLAB, toolboxes allow you to learn and

apply specialized technology. Toolboxes are comprehensive collections of MATLAB functions (M-files) that extend the MATLAB environment to solve particular classes of problems. Areas in which toolboxes are available include signal processing, control systems, neural networks, fuzzy logic, wavelets, simulation, and many others.

6.2.1 The MATLAB System

The MATLAB system consists of five main parts:

6.2.1.1 Development Environment:

This is the set of tools and facilities that help you use MATLAB functions and files. Many of these tools are graphical user interfaces. It includes the MATLAB desktop and Command Window, a command history, an editor and debugger, and browsers for viewing help, the workspace, files, and the search path.

6.2.1.2 The MATLAB Mathematical Function Library:

This is a vast collection of computational algorithms ranging from elementary functions, like sum, sine, cosine, and complex arithmetic, to more sophisticated functions like matrix inverse, matrix eigenvalues, Bessel functions, and fast Fourier transforms.

6.2.1.3 The MATLAB Language:

This is a high-level matrix/array language with control flow statements, functions, data structures, input/output, and object-oriented programming features. It allows both

"programming in the small" to rapidly create quick and dirty throw-away programs, and "programming in the large" to create large and complex application programs.

6.2.1.4 Graphics:

MATLAB has extensive facilities for displaying vectors and matrices as graphs, as well as annotating and printing these graphs. It includes high-level functions for two-dimensional and three-dimensional data visualization, image processing, animation, and presentation graphics. It also includes low-level functions that allow you to fully customize the appearance of graphics as well as to build complete graphical user interfaces on your MATLAB applications.

6.2.1.5 The MATLAB Application Program Interface (API):

This is a library that allows you to write C and Fortran programs that interact with MATLAB. It includes facilities for calling routines from MATLAB (dynamic linking), calling MATLAB as a computational engine, and for reading and writing MAT-files.

6.2.2 GUI (Graphical User Interface):

GUIDE, the MATLAB® Graphical User Interface development environment, provides a set of tools for creating graphical user interfaces (GUIs). These tools greatly simplify the process of designing and building GUIs. You can use the GUIDE tools to:

•Lay out the GUI

Using the GUIDE Layout Editor, you can lay out a GUI easily by clicking and dragging GUI components — such as panels, buttons, text fields, sliders, menus, and so on — into the layout area.

•Program the GUI

GUIDE automatically generates an M-file that controls how the GUI operates. The M-file initializes the GUI and contains a framework for all the GUI callbacks — the commands that are executed when a user clicks a GUI component. Using the M-file editor, you can add code to the callbacks to perform the functions you want them to [12].

6.2.3 Using MATLAB in this Project:

After initiate GUI to this project it will ask the user to input Patient information Age (years), Height (meters), Gender (Male or Female), Patient name. and Temperature (Celsius).

The program will load Two Text files that was saved by LabView and will plot two curves and do Processing stage as will be describe below. In the flowchart (Figure 6.4) Matlab Sequence.

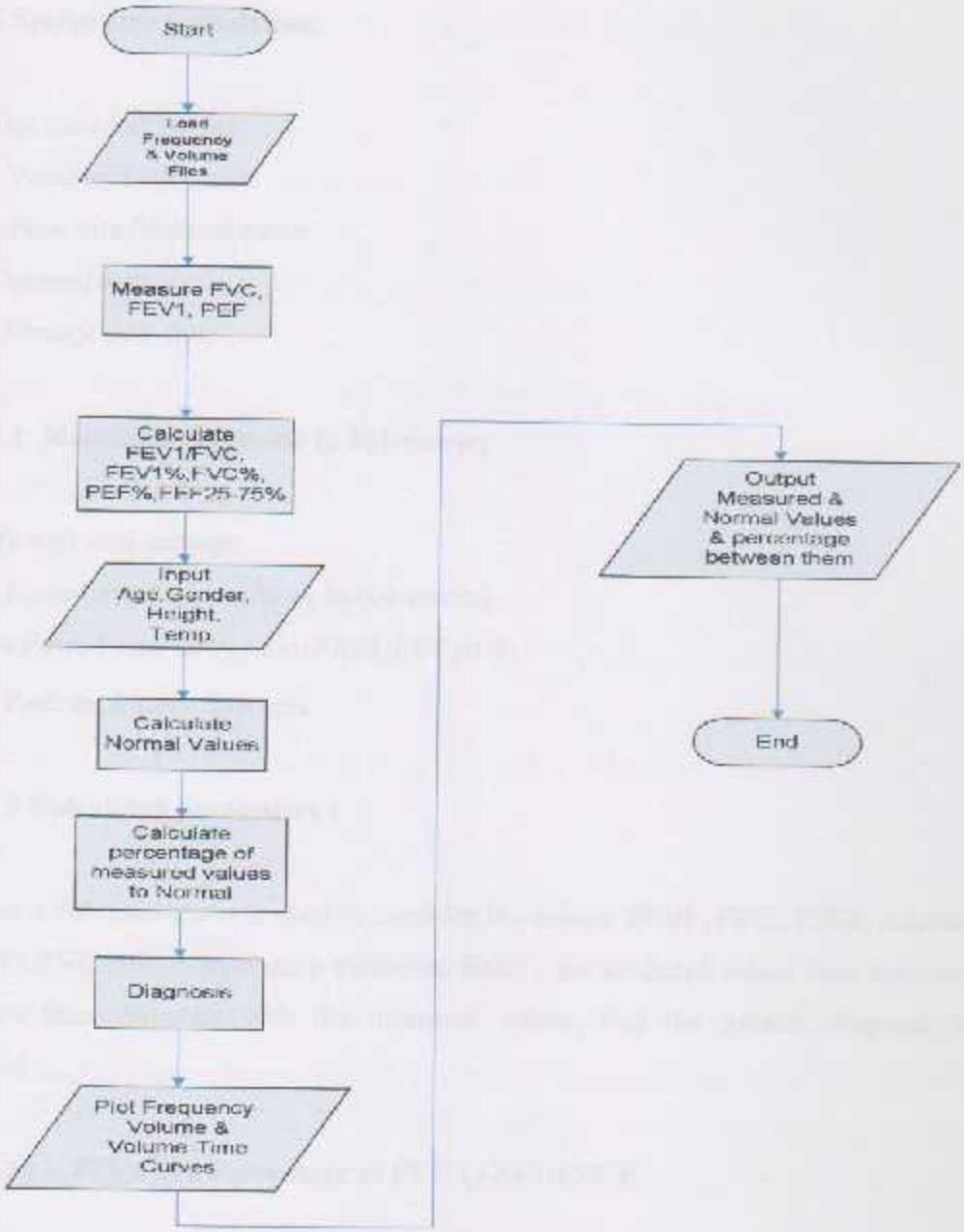


Figure 6.4: MATLAB Sequence

6.2.3.1 Spirometer Calculation:

The main test curves :

- Volume/ Time curve.
- Flow rate /Volume curve.

Optional test:

- Change over time

6.2.3.1.1 Measurements made in Spirometry

FVC: Forced vital capacity.

FEV1: Forced expiratory volume in one second .

FEV1%:Forced expiratory ratio *(FEV1/FVC)*x100

PEFR: Peak expiratory flow rate.

6.2.3.1.2 Calculated parameters :

For a full assessment it need to consider the values: FEV1, FVC, PEFR; calculate the FEV1/FVC ratio, temperature correction factor , the predicted values from the charts and how these compare with the measured values, find the suitable diagnose for measured .

6.2.3.1.2.1 FEV1 as a Percentage of FVC (*FEV1/FVC*):

Definition: The percent of the total observed FVC that is exhaled in the first second (FEV1). This calculation is useful for detecting obstructive disease. A person with healthy lungs can exhale 70-80% of the FVC in the first second, while a person

with airways obstruction may be able to exhale 60% or less of the FVC in the first second[13].

How to calculate:

1. Calculate the largest acceptable FVC and FEV1.
2. Divide the FEV1 by the FVC.
3. Multiply the answer by 100 to obtain the percentage.

6.2.3.1.2.2 Temperature correction factor (Conversion to BTPS):

BTPS Gas (air) at:

-Body Temperature (37°C)

-Ambient Pressure (surrounding air pressure)

-Saturated with water vapor (relative humidity = 100% as is the case in the lungs).

Air is at body temperature in the lungs and is saturated with water vapor. The ambient

Temperature is usually much cooler and dryer, thus exhaled air contracts as it leaves the lungs and enters a spirometer. The volume of air as recorded by most spirometers is usually 6-10% less than the actual volume of air exhaled by the test subject. The BTPS correction adjusts the measured result obtained in the spirometer to what the volume originally was in the lungs. In most volume spirometers, recorded volumes that have not been converted are indicated as ATPS (ambient temperature and pressure saturated with water vapor)[13].

Temperature Correction will be as described in FlowChart (Figure 6.5) Temperature Correction:

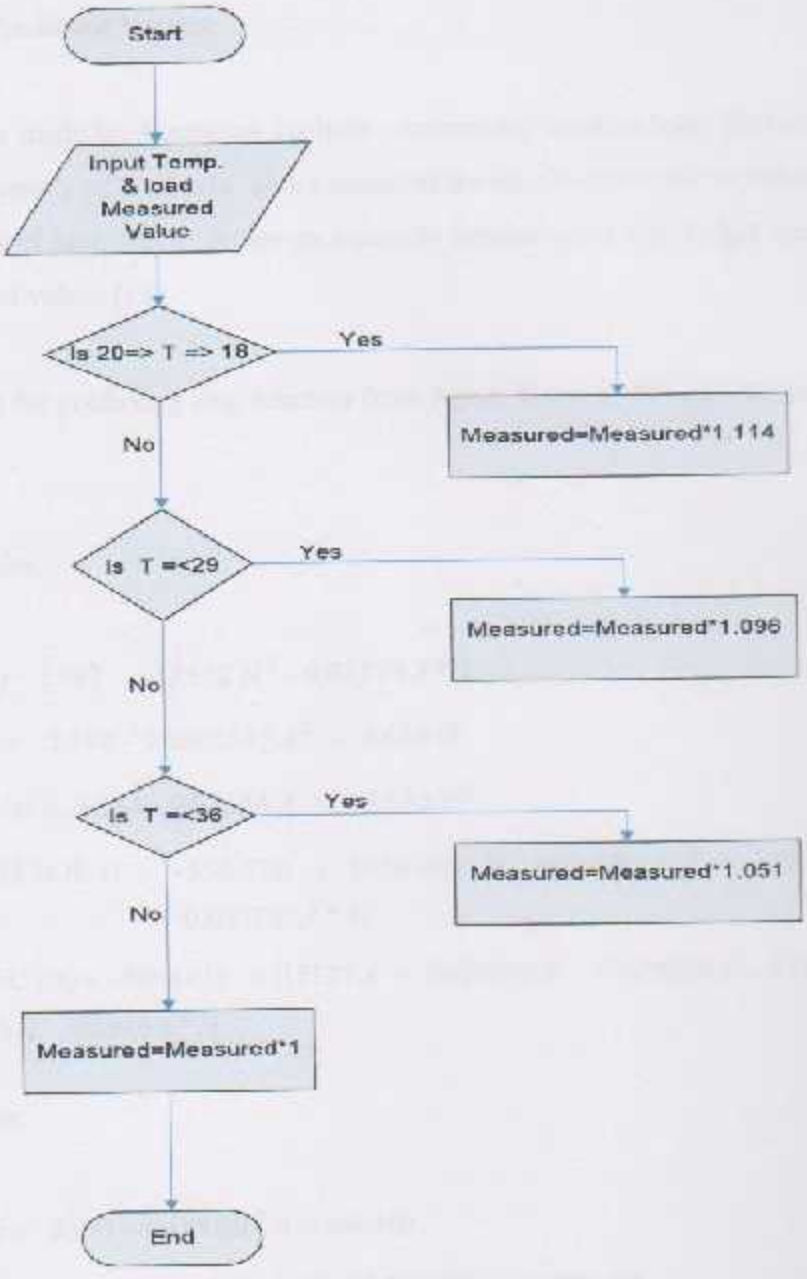


Figure 6.5: Temperature Correction

6.2.3.1.3 The Predicted Values:

A study is made by Australian institute –respiratory function unit, flinders medical center and University of Canberra, about nonsmokers sample of people to determine the normal volumes of lung and to driven an equation depending on age ,height ,sex ,to find normal predicted values [14].

Equations for predicting lung function from Age ((A)yrs) and Height ((H)m):

For Females:

$FEV1\ (L) = 1.597 + 0.5552\,H^3 - 0.01574\,A*H$

$FVC\ (L) = -3.598 - 0.0002525\,A^2 + 4.680\,H$

$PEFR\ (L/s) = 3.364 - 0.02654\,A + 1.036\,H^3$

$FEF\ 2575\%\ (L/s) = -556.706 + 1036.012\,H - 637.715\,H^2 + 131.013\,H^3 - 0.02708\,A*H$

$FEV1/FVC\ (\%) = -4068039 + 0.7137\,A + 0.002234\,A^2 + 7675039\,H - 4719018\,H^2 + 967776\,H^3 - 0.6946\,A*H$

For Males:

$FEV1(L) = 2.081 + 0.5846H^3 - 0.01599\,A*H$

$FVC\ (L) = 12.675 - 0.0002764\,A^2 - 10.736\,H^2 + 4.790\,H^3$

$PEFR\ (L/s) = -6.099 - 0.0003425\,A^2 + 9.708\,H$

$$\log 10\ FEF2575\%\ (l/s) = 0.5707 - 0.00005695\,A^2 + 0.025818\,H^3$$

$$FEV1/FVC\ (\%) = 92.963 + 0.002487\,A^2 - 0.2260\,A * H$$

6.2.3.1.3 Calculating Threshold as percentage with respect to normal:

$$\%\,\mathrm{Result} = [measured\ (read)/calculated\ (normal)] * 100\%$$

Example :

$$FEV1\% = FEV1\ (measured)\ /FEV1(normal)\ * 100\%$$

The lower limit of normality (LLN) for measured % is[15];

$PEFR > 80\%\ Normal$

$FVC\ \% > 80\ \%\ Normal$.

$FEV1\ \% > 80\%\ Normal$

$FEV1/FVC > 80\%\ Normal$

6.2.3.1.5 Comparing normal with the measured Values:

Diagnosis will be As the flowchart (Figure 6.6) Diagnosis:

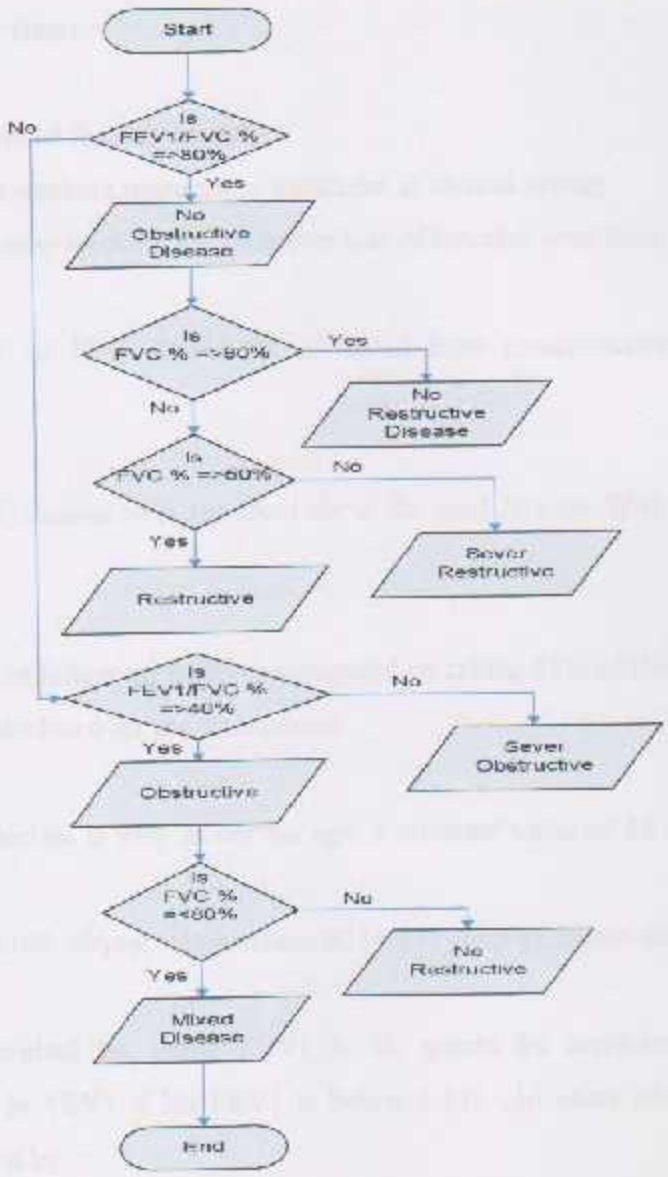


Figure 6.6: Diagnosis

Restrictive and Obstructive Ventilatory Defects

6.2.3.2 Change over time:

Should be examined for two reasons:

a-To evaluate a workers response to treatment in clinical setting

b-To screen healthy workers for excessive loss of function over time

Loss of FEV1 or FVC should be estimated from measurement made over a minimum 4-6 years.

FEV1 and FVC decline with age from about the med 30's on With acceleration of the rate as aging .

The LLN for the follow up FEV1 is computed by taking 85% of the base line value minus the expected decline over the time period.

As expected decline is vary as her/his age, a constant value of 25 ml/year is often recommend

Usually avarge100 ml/year when measured (4-11) years of follow up.

Example: individual his initial FEV1 is 4L would be considered to have an accelerated decline in FEV1 if his FEV1 is below 3.15L ,10 years after the base line value was determined by

*(0.85 *FEV1initial) (years *0.025 L/year)*

*(0.85*4L) - (10 years*0.025 L/year)* = 3.15 *L*

National Live Protection Association Examination Protocol that recommend spirometry testing every 3 years under age 29 ,2 years for people between (30 -39) [16].

And the flow chart below will describe the sequence of Change over Time in MATLAB:

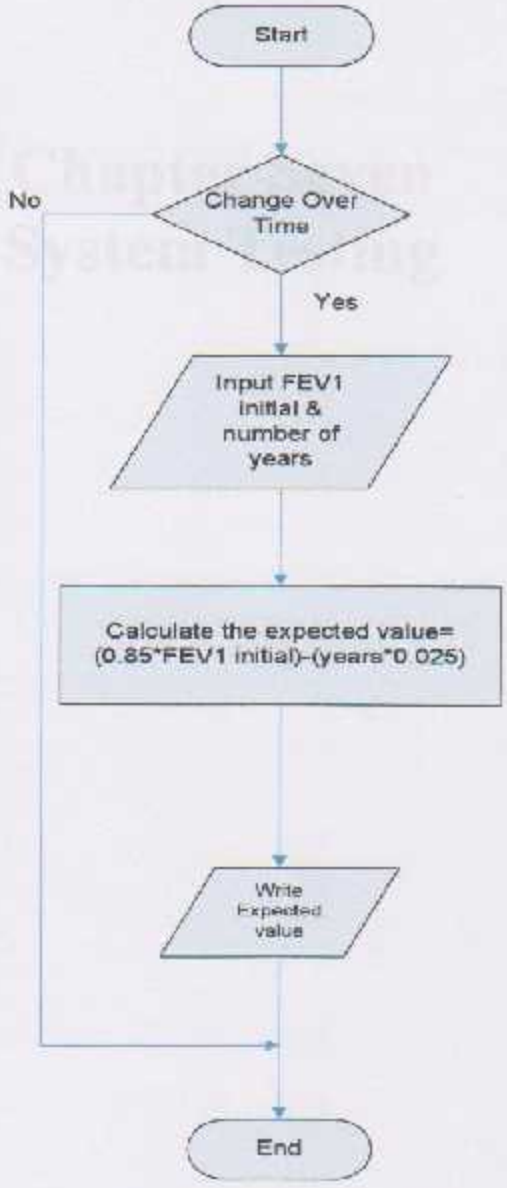


Figure 6.7: Change over time

Note: The M-file of the project will be in appendix (C)

7

Chapter Seven
System Testing

7.1 Hardware Testing
7.2 System Testing

Chapter Seven

System Testing

This chapter demonstrates how the system was tested; it is divided into two sections. In the first section Hardware was tested, then the system was tested finally By taking different samples then the complete system (software and hardware) testing.

7.1 Hardware Testing:

At this testing Transmitting (LED55C) & Receiving (Phototransistor) Devices was Tested. In this part we use Digital Multi meter to measure the voltage at the output of photo transistor when applying +5 V on the led and phototransistor and make them line insight and zero volt if we cut the line between them. as seen in the Figure 7.1.

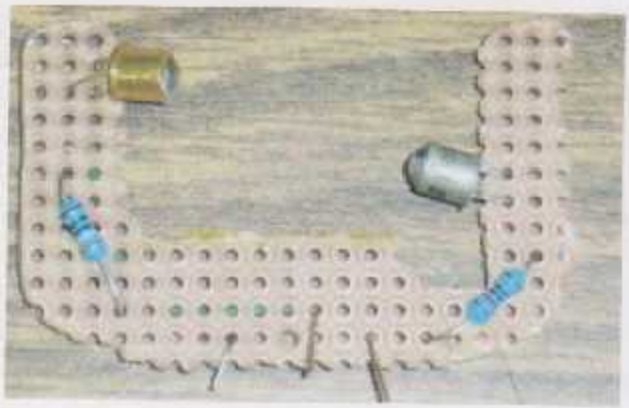


Figure 7.1 LED55C & Phototransistor test circuit

7.2 System Testing

For testing the device we first use Calibration syringe for calibration, secondly we do many tests on different Volunteers and at the same time same Volunteers do it in Al-Ahli Hospital and by comparing two results we find the they are the same. And below there samples of tests we do:

7.2.1 This results for Majed (Male) whose age is 25 , Height is 1.95 m, at 22 C° Room temperature.

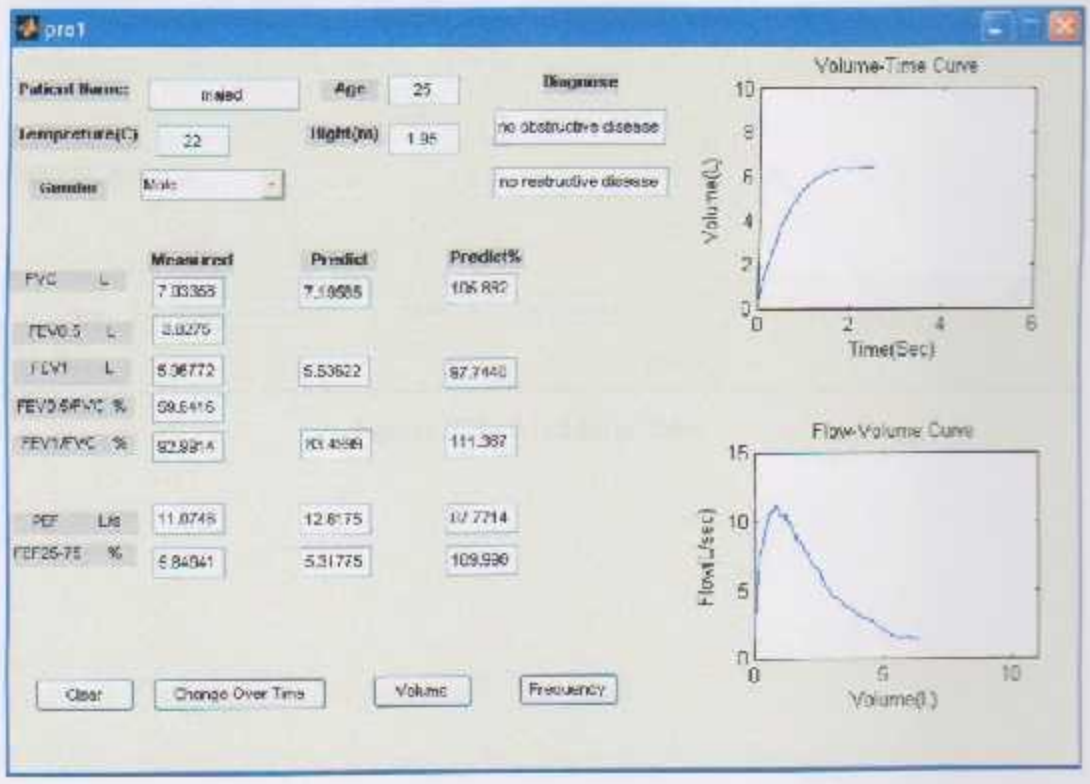


Figure 7.2: Majed Test

7.2.2 This results for Mashhour (Male) whose age is 23 , Hight is 1.65 m, 22 C° Room temperature.

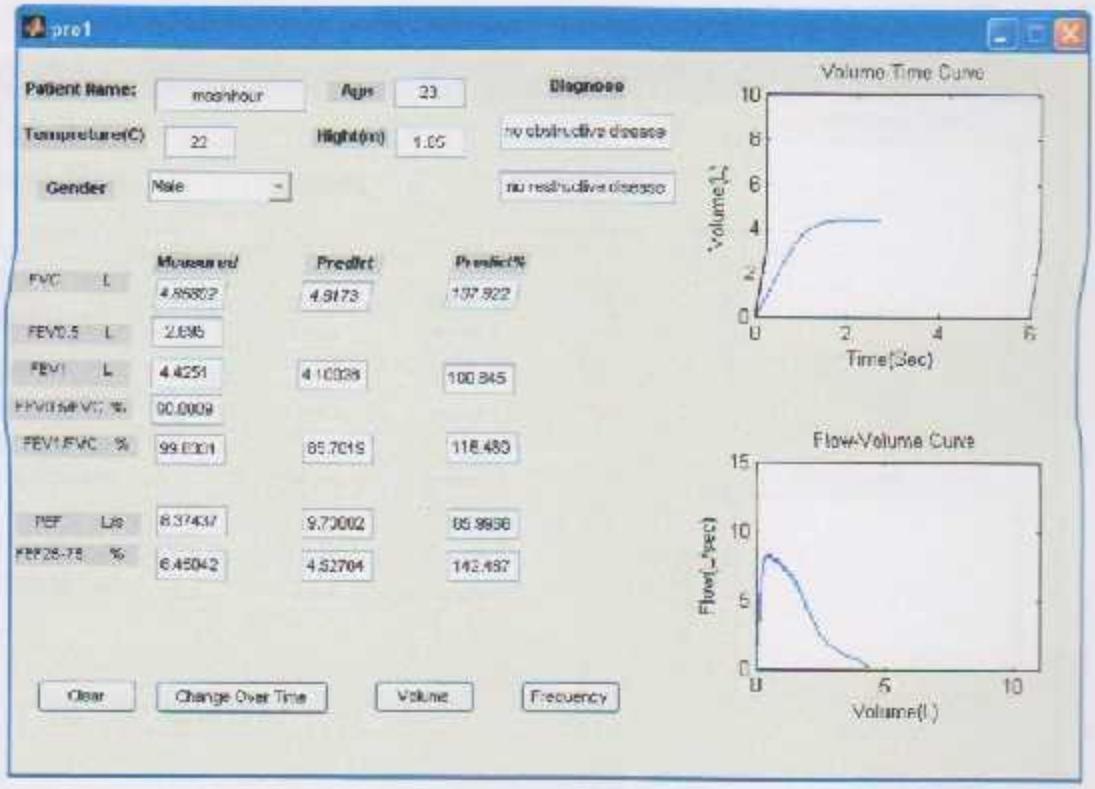


Figure 7.3 : Mashhour Test

7.2.3 This results for Shaden (Female) whose age is 20 , Hight is 1.69 m, 24 C° Room temperature.

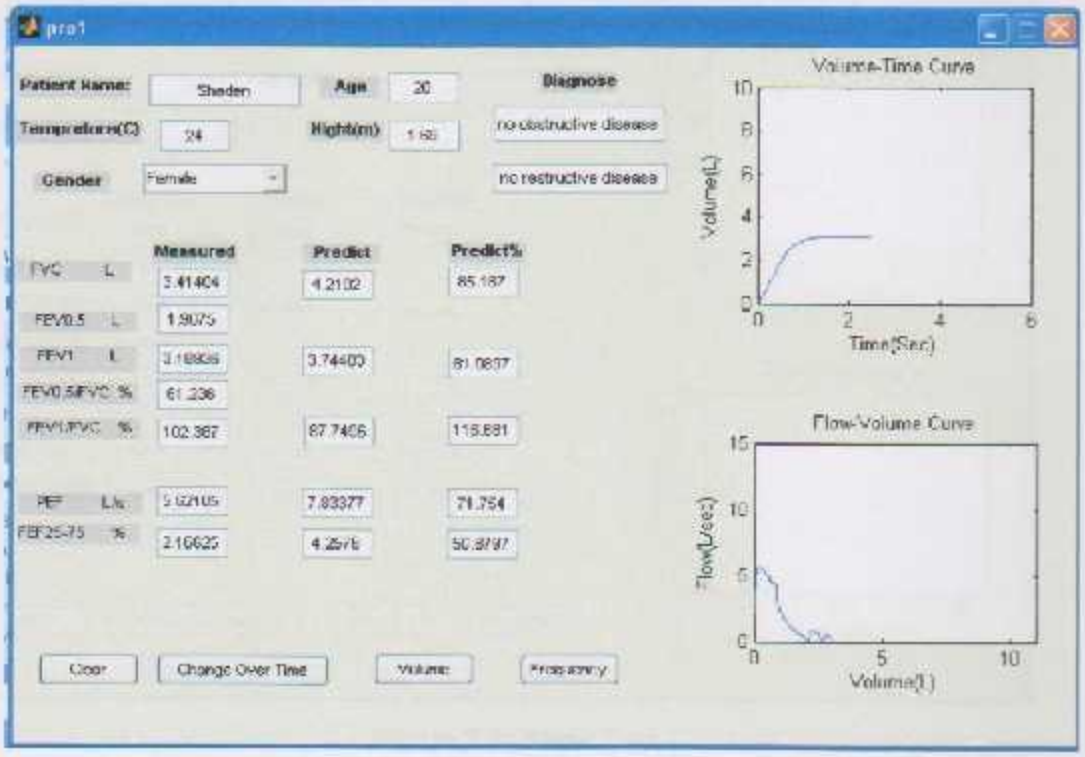


Figure 7.4: Shaden Test

7.2.4 This results for Eman (Female) whose age is 19 , Hight is 1.75 m, 24 C° Room temperature.

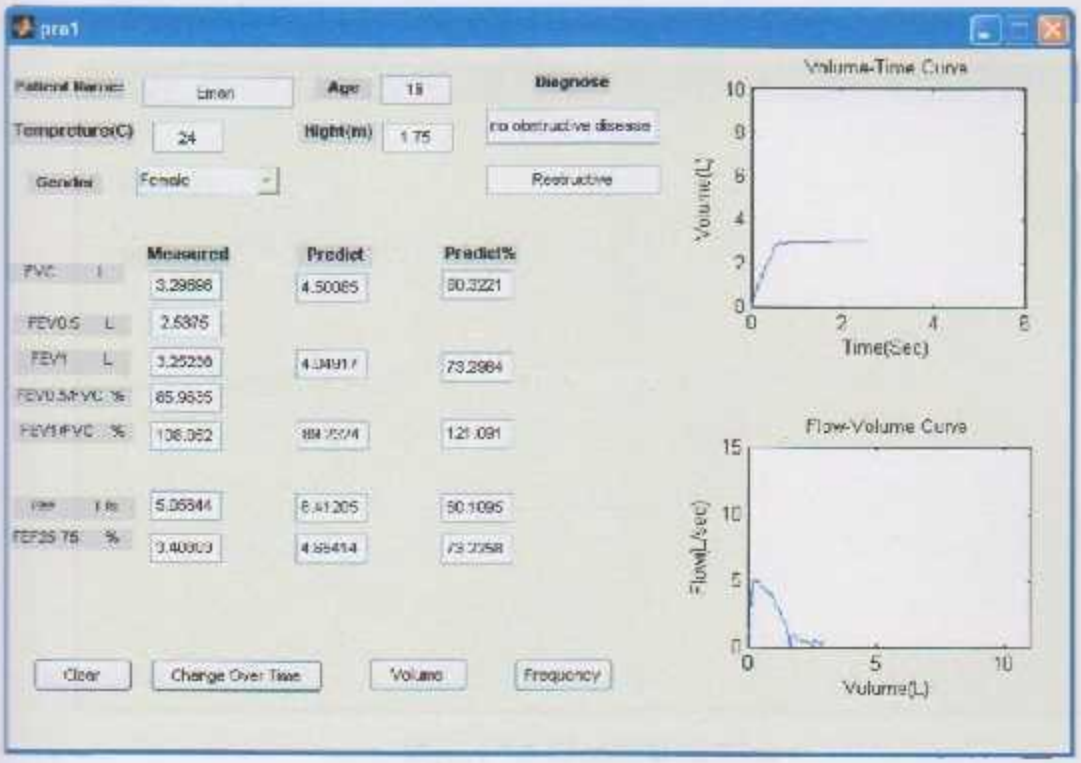


Figure 7.5: Eman Test

7.2.5 Change over Time Testing:

In this test we find Expected Value to 3.81 FEV1 initial after 5 yaers

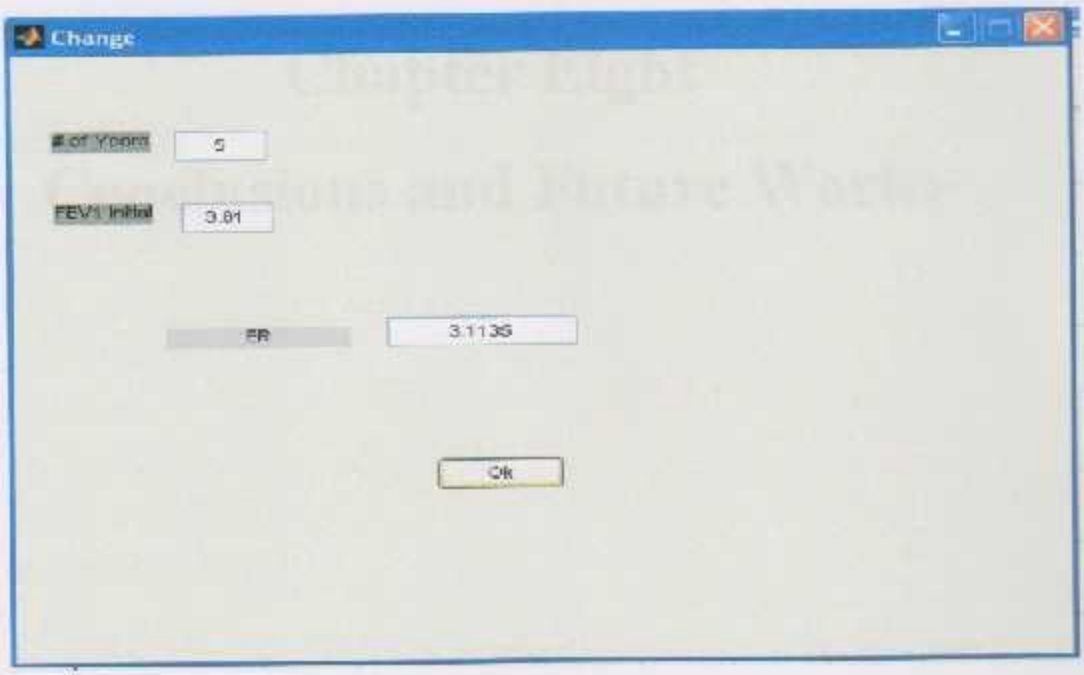


Figure 7.6 Change Over Time

8

Chapter Eight
Conclusions and Future Works

8.1 Conclusions

8.2 problems

8.3 future works

Chapter Eight

Conclusions and Future Works

This chapter describes the most important results which were concluded through constructing this design and discussing the problems which would be faced whether by hardware or software. Also some future studies will be discussed on the same device to develop it. Since this device become one of the devices which should be available in all hospitals and clinics.

8.1 Conclusions

- This project is a good example for the ability to convert the hardware medical devices into software system.
- This Project has low cost.
- Less problems.
- This device has the ability to give indication if there is respiration or not.
- Lung capacity and Flow rate can be determined using this device.
- MATLAB and Labview programs are very strong programs and can do different processing stages.
- Frequency result from the test very high and need sensitive device to measure it.
- Turbine plates are very small and current intensity must be high to be suitable with turbine plates.
- High sampling rate using DAQ.
- Helps doctors to make tests in any place by using it on portable PC.
- This device can help to determine if the person is ill or not.

8.2 problems

8.2.1 Hardware problems:

Although the role of spirometry equipment in the transmission of infections has not been conclusively documented, the mouthpiece, tubing, and other air spaces in a spirometer can provide a warm, moist environment favorable to the growth and transmission of disease-causing microorganisms. Manufacturers have offered bacterial filters, one-way airflow, and disposable components. For additional protection, gloves should be used for equipment handling and disinfection or sterilizing protocol should be implemented and documented. Hospitals should also consider a separate room for infectious patients and a testing room to assess equipment cleanliness to prevent cross-contamination.

8.2.2 Software problems:

Computer software can be a significant source of error, and suppliers should be able to document the accuracy of all software, especially for units whose waveform printouts are not appropriate for manual calculations. In addition, operator's manuals should include all algorithms used to calculate each parameter.

Using LABVIEW Version was not complete in University PC's then it was not helpful in this project for processing stage, for that MATLAB was the program that used.

8.3 Future works:

The researches and studies are advancing these days to develop this device, to make it analysis the gases out of the lungs, find the pollution percentage, types of these gases and diagnostic the lung states and the damages.

References

[1] Beniamin Cumming, Inter active physiology.

Available: Windows NT CD-Rom and www.adam.com.

[2] Joseph D. Bronzino. (2004) *The Biomedical Engineering hand book*. By CRC press.INC. Chapter 7, Respiratory System. p.70-81.

[3] Microsoft Encarta Encyclopedia 2002, Microsoft corporation.

[4] Butler Pike, Plymod Meeting, .(2005) *Health Care Product Compareson System*. USA

[5] *Wanger And Olfert.(2002) Pulmonary Function And Respiratory Regulation*. Spirometry and simple lung mechanism. Chapter 13.

[6] Richard Branson, Dean Hess, Robert Chatburn (eds). (1998) *Flow and Volume Measuring Devices*, J.B. Lippincott Co.. by F. Herbert Douce. Chapter 10 , A chapter in Respiratory Care Equipment. p. 283-303.

[7] David M. Shade, JD. *Design of Respiratory Devices in Biomedical Engineers' Handbook*, Johns Hopkins Pulmonary Laboratory. Research Associate in Medicine, Johns Hopkins School of Medicine Baltimore, MD 21287. University of Maryland College Park. p. 16-26.

[8] *http://en.wikipedia.org/wiki/Body_plethysmography*

[9] Professor Rob pierce MD, FRACP. (2004) *The Measurement and Interpretation of Ventilating Function in clinical*. Practice Pocket Guide to Spirometry.

[10] Philip H. Quanjer etal.(2005) *Lung Function Testing – Turbine Flow meter*. http://www.spirxpert.com.

[11] http://www-ee.eng.buffalo.edu/faculty/paololiu/edtech/rosldi/tutorials/labview.htm

[12] http://www.matlab.com/matlabcentral

[13] Professor Rob pierce MD, FRACP. (2004)*spirometry training guide*. Unit Five.

[14] C.J. Gore*, A.J. Crockett**, D.G. Pederson+, M.L. Booth++, A. Bauman+, N. Owen+,(1995) Spirometric standards for healthy adult life time nonsmokers in Australia . European Respiratory journal ISSN 0903-1936, Printed in UK
DOL 10 .1183/09031936.95.08050773.

[15] Dr Louise Rnewson, Dr Huw THOMAS.(08 may 2006) *Spirometry*. Patientplus © EMIS . www.spirometry -patient -UK.htm.

[16] Mary C Townsend ,Dr.P.H.(2006) *Spirometry in occupational setting*. American College of occupational And Enviromental Medicine ACOEM.

Appendices

Appendix (A)

Appendix (A)

American Thoracic Society (ATS) guidelines

American Thoracic Society (ATS) provides the following guidelines for maneuver performance.

FVC

- Minimum of 3 acceptable blows
- A rapid start is essential: this is defined as a back-extrapolated volume of <5% of FVC or 0.15 L, whichever is greater (See Figure 4)
- At least 6 second expiration
- End of test - no change in volume for at least 1 second after exhalation time of 6 seconds; or FET >15 seconds; or stopped for clinical reasons
- Spirometer temperature between 17 and 40 degrees Celsius; measure spirometer temperature to one degree Celsius
- Use of nose clip is encouraged
- Sitting or standing
- Reproducibility: the highest and second highest FVC should agree to within 0.2L
- Largest VC or FVC is recorded

FEV1

• As for FVC

• Take largest FEV1 even if not from the same curve as the best FVC

• "Zero time" determined by back-extrapolation - extrapolated volume should be <5% of FVC or 0.15 liters,

whichever is greatest (Figure 4)

• Smooth, rapid take off with no: hesitation, cough, leak, tongue obstruction, glottis closure, valsalva or early

termination

• Reproducibility: the highest and second highest FEV1 should agree to within 0.2L

FEF25-75% and Expiratory Flows

• From the single Spiro gram with the largest sum of FEV1 + FVC

PEF (Using a peak flow meter)

• Minimum of 3 acceptable blows

• Standing position is preferred

• Nose clip not necessary

• No cough

• Blow duration 1 to 2 seconds

Appendix (B)

Appendix B

Predicted Normal Values

The use of a fixed percent of predicted (eg 80%) to define the lower limit of normal is widespread despite being shown to be statistically invalid. A more appropriate approach is based on the use of the residual standard deviation (RSD) from regression analyses (e.g. FEV1 versus Age) but this is only possible if the survey population data are normally distributed for subjects of all ages and heights. The addition or subtraction of 1.64 times the RSD from the mean predicted value results in an upper or lower limit of normality with a confidence level such that 95% of the subjects in the survey lie above the lower limit. If the population data is not normally distributed then the 95th percentile may be used. This represents the point at which 95% of the normal population falls. Lower limits of normal are also given at the 95th percentile.

Mean Predicted Normal Values equations :

Table 2. – Equations for predicting lung function from age (yrs) and height (m)

Variable	Equation	ND p-value	Constant (5%)
Females			
FEV_1 L	$1.597 + 0.5552\ H^3 - 0.01574\ AH$	0.68	0.560
FVC L	$-3.598 - 0.0002525\ A^2 + 4.680\ H$	0.13	0.629
PEFR $L{\cdot}s^{-1}$	$3.364 - 0.02654\ A + 1.036\ H^3$	0.83	2.230
$FEF_{25-75\%}$ $L{\cdot}s^{-1}$	$-556.706 + 1036.012\ H - 637.715\ H^2 + 131.013\ H^3 - 0.02708\ AH$	0.32	1.271
FEV_1/FVC %	$-4068.039 + 0.7137\ A + 0.002234\ A^2 + 7675.039\ H - 4719.018\ H^2 + 967.776\ H^3 - 0.6946\ AH$	0.45	8.016
Males			
FEV_1 L	$2.081 + 0.5846\ H^3 - 0.01599\ AH$	0.59	0.798
FVC L	$12.675 - 0.0002764\ A^2 - 10.736\ H^2 + 4.790\ H^3$	0.56	1.035
PEFR $L{\cdot}s^{-1}$	$-6.099 - 0.0003425\ A^2 + 9.708\ H$	0.96	2.896
$\log_{10} FEF_{25-75\%}$ $L{\cdot}s^{-1}$	$0.5707 - 0.00005695\ A^2 + 0.025818\ H^3$	0.44	0.180
FEV_1/FVC %	$92.963 + 0.002487\ A^2 - 0.2260\ AH$	0.86	7.74

BTPS CONVERSION CHART

FACTORS FOR CONVERTING VOLUMES FROM

AMBIENT OR SPIROMETER TEMPERATURE TO BTPS

Gas Temperature Conversion

oF	oC	Factor
64	18	1.114
66	19	1.111
68	20	1.102
70	21	1.096
72	22	1.091
73	23	1.085
75	24	1.080
77	25	1.075
79	26	1.068
81	27	1.063
82	28	1.057
84	29	1.051
86	30	1.045
88	31	1.039
90	32	1.032
91	33	1.026
93	34	1.020
95	35	1.014
97	36	1.007
99	37	1.000

Appendix (C)

Appendix C

M file for spirometer calculation and graphical user interface

```
function varargout = pro1 ( varargin)
% PRO1 M-file for pro1.fig
%      PRO1, by itself, creates a new PRO1 or raises the existing
%      singleton*.
%
%      H = PRO1 returns the handle to a new PRO1 or the handle to
%      the existing singleton*.
%
%      PRO1 ( 'CALLBACK',hObject,eventData,handles,...)  calls the local
%      function named CALLBACK in PRO1.M with the given input arguments.
%
%      PRO1 ( 'Property','Value',...)  creates a new PRO1 or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before pro1_OpeningFunction gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to pro1_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run  ( singleton)  ".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Copyright 2002-2003 The MathWorks, Inc.

% Edit the above text to modify the response to help pro1
```

```
% Last Modified by GUIDE v2.5 17-May-2007 02:51:42

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @pro1_OpeningFcn, ...
                   'gui_OutputFcn',  @pro1_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []) ;
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1}) ;
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:}) ;
else
    gui_mainfcn(gui_State, varargin{:}) ;
end
% End initialization code - DO NOT EDIT


% --- Executes just before pro1 is made visible.
function pro1_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to pro1 (see VARARGIN)

% Choose default command line output for pro1
```

```
handles.output = hObject;

% Update handles structure
guidata (hObject, handles) ;

% UIWAIT makes pro1 wait for user response (see UIRESUME)
% uiwait (handles.figure1) ;


% --- Outputs from this function are returned to the command line.
function varargout = pro1_OutputFcn (hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT) ;
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout {1} = handles.output;


% --- Executes on button press in pushbutton1.
function varargout =pushbutton1_Callback (hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
%geting age,Hight,Tempreture in numerical form
N=get (handles.edit1,'String') ;
A=str2num (get (handles.edit2,'String') ) ;
H=str2num (get (handles.edit4,'String') ) ;
T=str2num (get (handles.edit3,'String') ) ;
%geting age,Hight,Tempreture in numerical form
G=get (handles.popupmenu1,'Value') ;
```

```
%Use equations to get ideal values for both mail and female.
switch (G)
  % 1 is Mail
  case 1
  FVCp = (12.675- (0.0002764*A.^2) - (10.736*H.^2) + (4.790*H.^3) ) ;
  FEV1p = (2.081+0.5846*H.^3-0.01599*A*H) ;
  FEV1FVCp = (92.963+0.002487*A.^2-0.2260*A*H) ;

  % 2 is Female
  case 2
  FVCp= ( 3.598-0.0002525*A.^2+4.680*H) ;
  FEV1p= (1.597+0.5552*H.^3-0.01574*A*H) ;
  FEV1FVCp= (-4068.039+0.7137*A+0.002234*A.^2+7675.039*H-
4719.018*H.^2+967.776*H.^3-0.6946*A*H) ;

end

%Display ideal values
set (handles.edit5,'String',FVCp)
set (handles.edit6,'String',FEV1p)
set (handles.edit7,'String',FEV1FVCp)

%loading Volume text file
a=dlmread ('test2.txt') ;
b=dlmread ('test1.txt') ;
d=b/248;
PEF=max (d) ;
q=d (100) ;
r=d (300) ;
FEF=q-r;
```

```
t1=linspace (0,2.5,1500) ;
c=a/400;
FVC=max (c) ;
FEV1=c (600) ;
FEV05=c (300) ;
FEV1percentage= (FEV1/FVC) *100;
FEV05percentage= (FEV05/FVC) *100;
axes (handles.axes1)
plot (t1,c)
axis ([0 6 0 10])
title ('Volume-Time Curve')
xlabel ('Time (Sec) ') ;
ylabel ('Volume (L) ') ;

%Tempreture Effect

if T >= 18 & T <= 45
  if T<=20
 FVC=FVC*1.114;
 FEV1=FEV1*1.114;
 FEV1percentage=FEV1percentage*1.114;

  elseif T <= 29
     FVC=FVC*1.096;
     FEV1=FEV1*1.096;
     FEV1percentage=FEV1percentage*1.096;

  elseif T <= 36
     FVC=FVC*1.051;
     FEV1=FEV1*1.051;
     FEV1percentage=FEV1percentage*1.051;
```

```
  else
     FVC=FVC*1;
     FEV1=FEV1*1;
     FEV1percentage=FEV1percentage*1;
  end
end

set (handles.edit8,'String',FVC)
set (handles.edit9,'String',FEV05)
set (handles.edit10,'String',FEV1)
set (handles.edit11,'String',FEV05percentage)
set (handles.edit12,'String',FEV1percentage)
%Display Results

%FEV1%
FEV1FEV1p= (FEV1/FEV1p) *100;
%FVC%
FVCFVCp= (FVC/FVCp) *100;
%FEV1per.%
FEV1per= (FEV1percentage/FEV1FVCp) *100;

set (handles.edit13,'String',FEV1FEV1p)
set (handles.edit14,'String',FVCFVCp)
set (handles.edit15,'String',FEV1per)

save ('N','FVC','FEV1','FEV1percentage','FEV1FEV1p','FVCFVCp','PEF','FEF')

%Diagnosis
if FEV1percentage >= 80
   set (handles.edit16,'String','no obstructive disease')
```

```
    if FVCFVCp >= 80
       set (handles.edit17,'String','no restructive disease')
    elseif FVCFVCp >= 50
       set (handles.edit17,'String','Restructive')
    else
       set (handles.edit17,'String','sever Restructive')
    end
elseif    FEV1percentage  >= 40
    set (handles.edit16,'String','Obstructive')
    if FVCFVCp <= 80
       set (handles.edit17,'String','mixed Obstructive & restructive')
    else
       set (handles.edit17,'String','no mixed Obstructive & restructive')
    end
end
return;

function edit1_Callback (hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get (hObject,'String')  returns contents of edit1 as text
%        str2double (get (hObject,'String') )  returns contents of edit1 as a double


% --- Executes during object creation, after setting all properties.
function edit1_CreateFcn (hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
```

```
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), ...
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit2_Callback(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit2 as text
%        str2double(get(hObject,'String')) returns contents of edit2 as a double


% --- Executes during object creation, after setting all properties.
function edit2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), ...
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
```

```
end


function edit3_Callback(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit3 as text
%        str2double(get(hObject,'String')) returns contents of edit3 as a double


% --- Executes during object creation, after setting all properties.
function edit3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function edit4_Callback(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
```

```
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data  (see GUIDATA)

% Hints: get(hObject,'String')  returns contents of edit4 as text
%        str2double(get(hObject,'String'))  returns contents of edit4 as a double


% --- Executes during object creation, after setting all properties.
function edit4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit4  (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in popupmenu1.
function popupmenu1_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu1  (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data  (see GUIDATA)

% Hints: contents = get(hObject,'String')  returns popupmenu1 contents as cell array
%        contents{get(hObject,'Value')}  returns selected item from popupmenu1
```

```
% --- Executes during object creation, after setting all properties.
function popupmenu1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit5_Callback(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit5 as text
%        str2double(get(hObject,'String')) returns contents of edit5 as a double



% --- Executes during object creation, after setting all properties.
function edit5_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
```

```
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal (get (hObject,'BackgroundColor') ,
get (0,'defaultUicontrolBackgroundColor') )
   set (hObject,'BackgroundColor','white') ;
end



function edit6_Callback (hObject, eventdata, handles)
% hObject    handle to edit6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get (hObject,'String') returns contents of edit6 as text
%        str2double (get (hObject,'String') ) returns contents of edit6 as a double


% --- Executes during object creation, after setting all properties.
function edit6_CreateFcn (hObject, eventdata, handles)
% hObject    handle to edit6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal (get (hObject,'BackgroundColor') ,
get (0,'defaultUicontrolBackgroundColor') )
   set (hObject,'BackgroundColor','white') ;
```

```
end


function edit7_Callback(hObject, eventdata, handles)
% hObject    handle to edit7 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit7 as text
%        str2double(get(hObject,'String')) returns contents of edit7 as a double


% --- Executes during object creation, after setting all properties.
function edit7_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit7 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function edit8_Callback(hObject, eventdata, handles)
% hObject    handle to edit8 (see GCBO)
```

```
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data  (see GUIDATA)

% Hints: get(hObject,'String')  returns contents of edit8 as text
%        str2double(get(hObject,'String'))  returns contents of edit8 as a double


% --- Executes during object creation, after setting all properties.
function edit8_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit8  (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit9_Callback(hObject, eventdata, handles)
% hObject    handle to edit9  (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data  (see GUIDATA)

% Hints: get(hObject,'String')  returns contents of edit9 as text
%        str2double(get(hObject,'String'))  returns contents of edit9 as a double
```

```
% --- Executes during object creation, after setting all properties.
function edit9_CreateFcn (hObject, eventdata, handles)
% hObject    handle to edit9 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal (get (hObject,'BackgroundColor') ,
get (0,'defaultUicontrolBackgroundColor') )
   set (hObject,'BackgroundColor','white') ;
end



function edit10_Callback (hObject, eventdata, handles)
% hObject    handle to edit10 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get (hObject,'String') returns contents of edit10 as text
%        str2double (get (hObject,'String') ) returns contents of edit10 as a double



% --- Executes during object creation, after setting all properties.
function edit10_CreateFcn (hObject, eventdata, handles)
% hObject    handle to edit10 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
```

```
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit11_Callback(hObject, eventdata, handles)
% hObject    handle to edit11 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit11 as text
%        str2double(get(hObject,'String')) returns contents of edit11 as a double


% --- Executes during object creation, after setting all properties.
function edit11_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit11 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
```

```
end


function edit12_Callback(hObject, eventdata, handles)
% hObject    handle to edit12 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit12 as text
%        str2double(get(hObject,'String')) returns contents of edit12 as a double


% --- Executes during object creation, after setting all properties.
function edit12_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit12 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function edit13_Callback(hObject, eventdata, handles)
% hObject    handle to edit13 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit13 as text
%        str2double(get(hObject,'String')) returns contents of edit13 as a double


% --- Executes during object creation, after setting all properties.
function edit13_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit13 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit14_Callback(hObject, eventdata, handles)
% hObject    handle to edit14 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit14 as text
```

```
%        str2double(get(hObject,'String')) returns contents of edit14 as a double


% --- Executes during object creation, after setting all properties.
function edit14_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit14 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit15_Callback(hObject, eventdata, handles)
% hObject    handle to edit15 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit15 as text
%        str2double(get(hObject,'String')) returns contents of edit15 as a double


% --- Executes during object creation, after setting all properties.
function edit15_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit15 (see GCBO)
```

```
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit16_Callback(hObject, eventdata, handles)
% hObject    handle to edit16 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit16 as text
%        str2double(get(hObject,'String')) returns contents of edit16 as a double


% --- Executes during object creation, after setting all properties.
function edit16_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit16 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
```

```
%       See ISPC and COMPUTER.
if ispc && isequal (get (hObject,'BackgroundColor') ,
get (0,'defaultUicontrolBackgroundColor') )
   set (hObject,'BackgroundColor','white') ;
end


function edit17_Callback (hObject, eventdata, handles)
% hObject    handle to edit17 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get (hObject,'String')  returns contents of edit17 as text
%        str2double (get (hObject,'String') )  returns contents of edit17 as a double


% --- Executes during object creation, after setting all properties.
function edit17_CreateFcn (hObject, eventdata, handles)
% hObject    handle to edit17 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal (get (hObject,'BackgroundColor') ,
get (0,'defaultUicontrolBackgroundColor') )
   set (hObject,'BackgroundColor','white') ;
end
```

```
% --- Executes on button press in pushbutton2.
function varargout =pushbutton2_Callback (hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

b=dlmread ('test1.txt') ;
a=dlmread ('test2.txt') ;
c=a/400;
FVC=max (c) ;
t2=linspace (0,FVC,300) ;
d=b/248;
axes (handles.axes2)
plot (t2,d)
axis ([0 11 0 15])
title ('Flow-Volume Curve)
xlabel ('Volume (L)') ;
ylabel ('Flow (L/sec)') ;
PEF=max (d) ;
q=d (100) ;
r=d(300),
FEF=q-r;
set(handles.edit19,'String',FEF)
set(handles.edit18,'String',PEF)

N=get(handles.edit1,'String');
A=str2num(get(handles.edit2,'String'));
H=str2num(get(handles.edit4,'String'));
T=str2num(get(handles.edit3,'String'));
G=get(handles.popupmenu1,'Value');

switch (G)
```

```
    case 1
    PEFRp =(-6.099-0.0003425*A.^2+9.708*H);
    r= 0.5707-0.00005695*A.^2-0.025818*H.^3;
    v=2.3*r;
    FEFp=exp(v);

    case 2
    PEFRp=(3.364-0.02654*A+1.036*H.^3);
    FEFp= -556.706+1036.012*H-637.715*H.^2+131.013*H.^3-0.002234*A.^2;
    otherwise
      return
end

set(handles.edit20,'String',PEFRp)
set(handles.edit21,'String',FEFp)

x=(PEF/PEFRp)*100;
y=(FEF/FEFp)*100;
set(handles.edit22,'String',x)
set(handles.edit23,'String',y)
return;

function edit18_Callback(hObject, eventdata, handles)
% hObject    handle to edit18 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit18 as text
%        str2double(get(hObject,'String')) returns contents of edit18 as a double


% --- Executes during object creation, after setting all properties.
function edit18_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit18 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end



function edit19_Callback(hObject, eventdata, handles)
```

```
% hObject    handle to edit19 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit19 as text
%        str2double(get(hObject,'String')) returns contents of edit19 as a double


% --- Executes during object creation, after setting all properties.
function edit19_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit19 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit20_Callback(hObject, eventdata, handles)
% hObject    handle to edit20 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit20 as text
%        str2double(get(hObject,'String')) returns contents of edit20 as a double


% --- Executes during object creation, after setting all properties.
function edit20_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit20 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function edit21_Callback(hObject, eventdata, handles)
% hObject    handle to edit21 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit21 as text
%        str2double(get(hObject,'String')) returns contents of edit21 as a double


% --- Executes during object creation, after setting all properties.
function edit21_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit21 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit22_Callback(hObject, eventdata, handles)
% hObject    handle to edit22 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit22 as text
%        str2double(get(hObject,'String')) returns contents of edit22 as a double


% --- Executes during object creation, after setting all properties.
function edit22_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit22 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function edit23_Callback(hObject, eventdata, handles)
% hObject    handle to edit23 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit23 as text
%        str2double(get(hObject,'String')) returns contents of edit23 as a double


% --- Executes during object creation, after setting all properties.
function edit23_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit23 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



% --- Executes on button press in pushbutton3.
function pushbutton3_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
guide Change



% --- Executes on button press in pushbutton4.
function pushbutton4_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
%Clearing data in Text files and in the user interface
dlmwrite('test1.txt',[]);
dlmwrite('test2.txt',[]);
axes(handles.axes2)
plot(0,0)
```

```
axes(handles.axes1)
plot(0,0)
set(handles.edit1,'String','')
set(handles.edit2,'String','')
set(handles.edit3,'String','')
set(handles.edit4,'String','')
set(handles.edit5,'String','')
set(handles.edit6,'String','')
set(handles.edit7,'String','')
set(handles.edit8,'String','')
set(handles.edit9,'String','')
set(handles.edit10,'String','')
set(handles.edit11,'String','')
set(handles.edit12,'String','')
set(handles.edit13,'String','')
set(handles.edit14,'String','')
set(handles.edit15,'String','')
 set(handles.edit16,'String','')
set(handles.edit17,'String','')
set(handles.edit18,'String','')
set(handles.edit19,'String','')
set(handles.edit20,'String','')
set(handles.edit21,'String','')
set(handles.edit22,'String','')
 set(handles.edit23,'String','')
```

M File for change overtime calculation :

```
function varargout = Change(varargin)
% CHANGE M-file for Change.fig
%      CHANGE, by itself, creates a new CHANGE or raises the existing
%      singleton*.
%
%      H = CHANGE returns the handle to a new CHANGE or the handle to
%      the existing singleton*.
%
%      CHANGE('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in CHANGE.M with the given input
arguments.
%
%      CHANGE('Property','Value',...) creates a new CHANGE or raises
the
%      existing singleton*.  Starting from the left, property value
pairs are
```

```
%      applied to the GUI before Change_OpeningFunction gets called.
An
%      unrecognized property name or invalid value makes property
application
%      stop.  All inputs are passed to Change_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only
one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Copyright 2002-2003 The MathWorks, Inc.

% Edit the above text to modify the response to help Change

% Last Modified by GUIDE v2.5 13-May-2007 20:18:59

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @Change_OpeningFcn, ...
                   'gui_OutputFcn',  @Change_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before Change is made visible.
function Change_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to Change (see VARARGIN)

% Choose default command line output for Change
handles.output = hObject;
```

```
% Update handles structure
guidata(hObject, handles);

% UIWAIT makes Change wait for user response (see UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = Change_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


function edit1_Callback(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit1 as text
%        str2double(get(hObject,'String')) returns contents of edit1 as
a double


% --- Executes during object creation, after setting all properties.
function edit1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function edit2_Callback(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of edit2 as text
%        str2double(get(hObject,'String')) returns contents of edit2 as
a double

% --- Executes during object creation, after setting all properties.
function edit2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit3_Callback(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit3 as text
%        str2double(get(hObject,'String')) returns contents of edit3 as
a double


% --- Executes during object creation, after setting all properties.
function edit3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
Y=str2num(get(handles.edit1,'String'));
I=str2num(get(handles.edit2,'String'));
Change=(0.85*I)-(Y*0.025);
set(handles.edit3,'String',Change)
```

Appendix (D)

Datasheet for

1-phototransistor

2-infrared LED

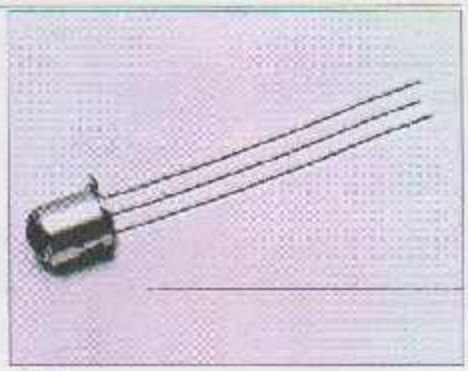
3-DAQ

SD5600/5610

Optoschmitt Detector

FEATURES

- TO-46 metal can package
- 6° (nominal) acceptance angle
- High noise immunity output
- TTL/LSTTL/CMOS compatible
- Buffer (SD5600) or inverting (SD5610) logic available
- Mechanically and spectrally matched to SE3450/5450, SE3455/5455 and SE3470/5470 infrared emitting diodes



DESCRIPTION

The SD5600/5610 series is a family of single chip Optoschmitt IC detectors mounted in a TO-46 metal can package. The photodetector consists of a photodiode, amplifier, voltage regulator, Schmitt trigger and an NPN output transistor with 10 kΩ (nominal) pull-up resistor. Output rise and fall times are independent of the rate of change of incident light. Detector sensitivity has been intensity temperature compensated. The TO-46 package is ideally suited for operation in hostile environments.

Device Polarity

Buffer - Output is HI when incident light intensity is above the turn- on threshold level.

Inverter - Output is LO when incident light intensity is above the turn- on threshold level.

OUTLINE DIMENSIONS in inches (mm)

Tolerance	3 pl c decimals	±0.005(0.12)
	2 pl c decimals	±0.020(0.51)

Honeywell

SD5600/5610

Optoschmitt Detector

ELECTRICAL CHARACTERISTICS (-40°C to +100°C unless otherwise noted)

PARAMETER	SYMBOL	MIN	TYP	MAX	UNITS	TEST CONDITIONS
Operating Supply Voltage	V_{CC}	4.5		16.0	V	T_A=25°C
Turn-on Threshold Irradiance [2]	$E_{E(H)}$				mW/cm²	V_{CC}=5 V
SD5600-001, SD5610-001				2.50		T_A=25°C
Hysteresis [3]	HYST	5		50	%	
Supply Current	I_{CC}				mA	E_E=0 Or 3.0 mW/cm²
				12.0		V_{CC}=5 V
				15.0		V_{CC}=16 V
High Level Output Voltage	V_{OH}				V	V_{CC}=5 V, I_{OH}=0
SD5600		2.4				E_E=0
SD5610		2.4				E_E=3.0 mW/cm²
Low Level Output Voltage	V_{OL}				V	V_{CC}=5 V, I_{SL}=12.8 mA
SD5600				0.4		E_E=0
SD5610				0.4		E_E=3.0 mW/cm²
Internal Pull-Up Resistor	R_{INT}	5.0	10.0	20.0	kΩ	
Operate Point Temperature Coefficient	$OPTO$		-0.76		%/°C	Emitter @ Constant Temperature
Output Rise Time	t_r		50		ns	R_L=390 Ω, C_L=50 pF
Output Fall Time	t_f		15		ns	R_L=390 Ω, C_L=50 pF
Propagation Delay, Low-High, High-Low	t_{PLH}, t_{PHL}		5.0		µs	R_L=390 Ω, C_L=50 pF
Clock Frequency				100	kHz	R_L=390 Ω, C_L=50 pF

Notes

1. It is recommended that a bypass capacitor, 0.1 µF typical, be added between V_{CC} and GND near the device in order to stabilize power supply line.
2. The radiation source is an IRED with a peak wavelength of 938 nm.
3. Hysteresis is defined as the difference between the operating and release threshold irradiances, expressed as a percentage of the operate threshold intensity.

ABSOLUTE MAXIMUM RATINGS

(25°C Free-Air Temperature unless otherwise noted)

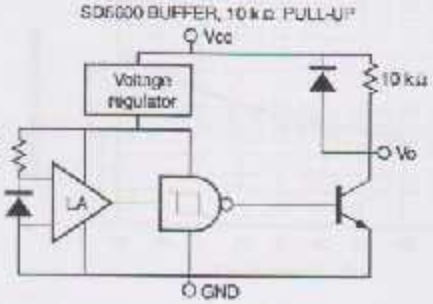
Supply Voltage	16 V [1]
Duration of Output	
Short to V_{CC} or Ground	1.0 sec
Output Current	16 mA
Operating Temperature Range	-40°C to 100°C
Storage Temperature Range	-55°C to 125°C
Soldering Temperature (10 sec)	260°C

Notes

1. Derate linearly from 85°C to 7 V at 100°C.

SCHEMATIC

SD5600 BUFFER, 10 k Ω PULL-UP



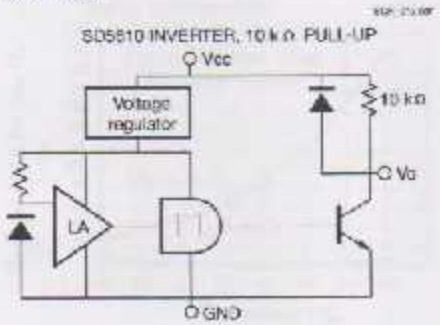
Honeywell reserves the right to make changes in order to improve design and supply the best products possible.

Honeywell

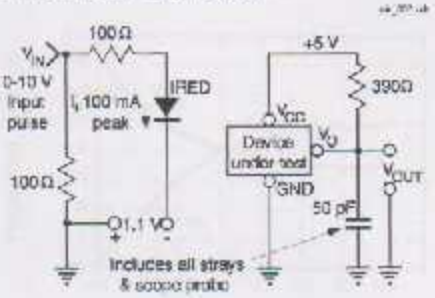
SD5600/5610

Optoschmitt Detector

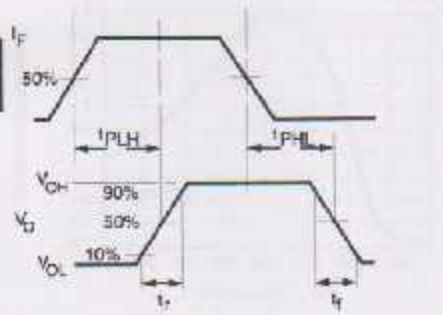
SCHEMATIC



SWITCHING TIME TEST CIRCUIT



SWITCHING WAVEFORM FOR BUFFERS



SWITCHING WAVEFORM FOR INVERTERS

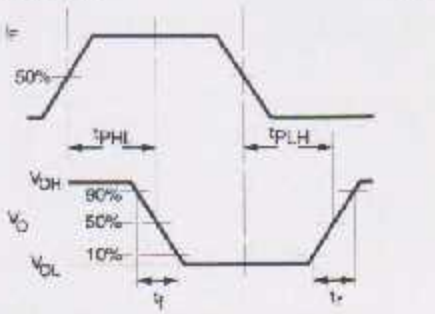


Fig. 1 Responsivity vs. Angular Displacement

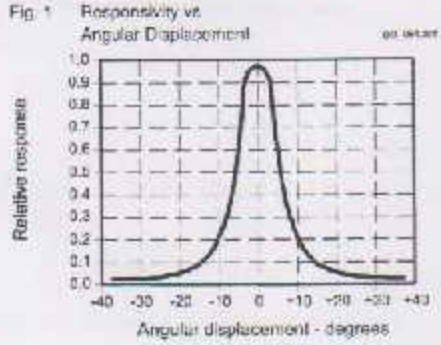
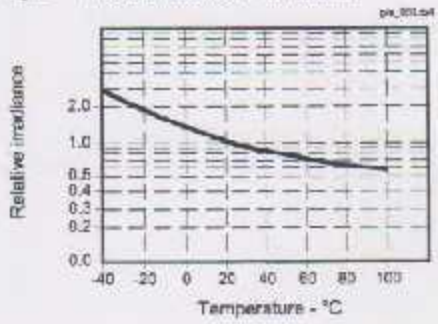


Fig. 2 Threshold Irradiance vs Temperature



Honeywell

SD5600/5610

Optoschmitt Detector

Fig. 3 Output Rise Time (t_R) and Output Fall Time (t_F) vs Temperature

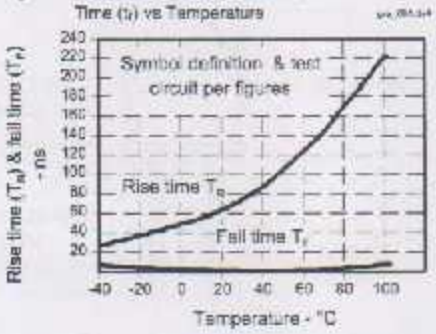


Fig. 4 Delay Time vs Temperature

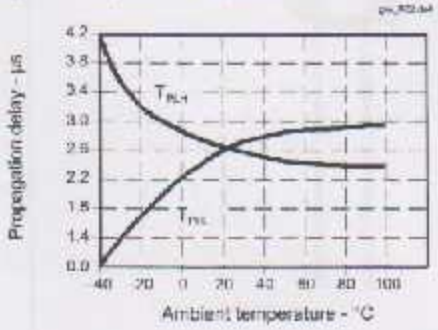
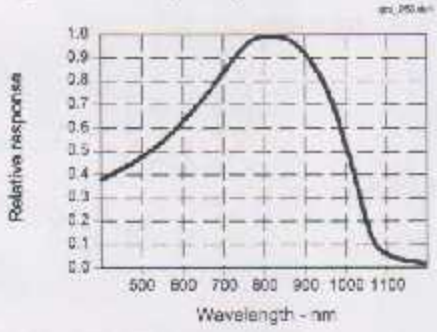


Fig. 5 Spectral Responsivity



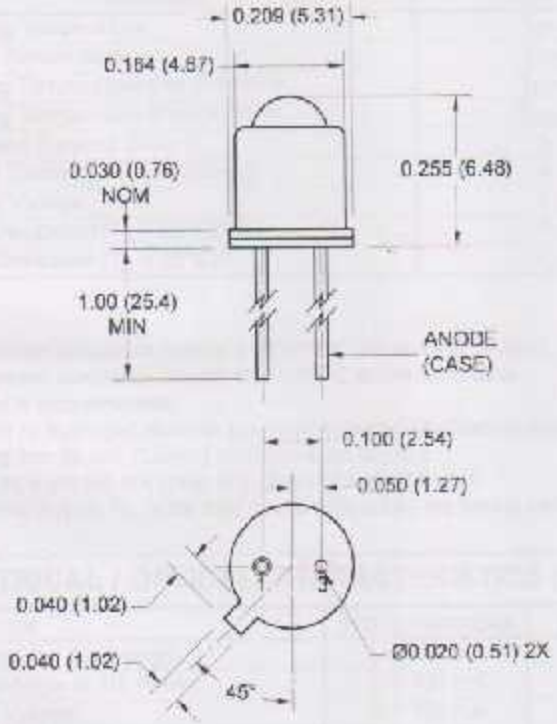
All Performance Curves Show Typical Values

FAIRCHILD
SEMICONDUCTOR®

GaAs INFRARED EMITTING DIODE

LED55B LED55C LED56

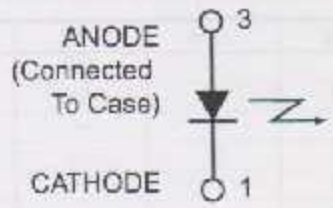
PACKAGE DIMENSIONS



NOTES:

1. Dimensions for all drawings are in inches (mm).
2. Tolerance of ± .010 (.25) on all non-nominal dimensions unless otherwise specified.

SCHEMATIC



DESCRIPTION

The LED55B/LED55C/LED56 are 940 nm LEDs in a narrow angle, TO-46 package.

FEATURES

- Good optical to mechanical alignment
- Mechanically and wavelength matched to the TO-18 series phototransistor
- Hermetically sealed package
- High irradiance level

FAIRCHILD
SEMICONDUCTOR®

GaAs INFRARED EMITTING DIODE

LED55B LED55C LED56

ABSOLUTE MAXIMUM RATINGS (T_A = 25°C unless otherwise specified)

Parameter	Symbol	Rating	Unit
Operating Temperature	T_{OPR}	-65 to +125	°C
Storage Temperature	T_{STG}	-65 to +150	°C
Soldering Temperature (Iron)(3,4,5 and 6)	T_{SOL-I}	240 for 5 sec	°C
Soldering Temperature (Flow)(3,4 and 6)	T_{SOL-F}	260 for 10 sec	°C
Continuous Forward Current	I_F	100	mA
Forward Current (pw, 1µs, 200Hz)	I_F	10	A
Reverse Voltage	V_R	3	V
Power Dissipation (T_A = 25°C)(1)	P_D	170	mW
Power Dissipation (T_C = 25°C)(2)	P_D	1.3	W

NOTE:

1. Derate power dissipation linearly 1.70 mW/°C above 25°C ambient.
2. Derate power dissipation linearly 13.0 mW/°C above 25°C case.
3. RMA flux is recommended.
4. Methanol or isopropyl alcohols are recommended as cleaning agents.
5. Soldering iron tip 1/16" (1.6mm) minimum from housing.
6. As long as leads are not under any stress or spring tension.
7. Total power output, P_O, is the total power radiated by the device into a solid angle of 2 π steradians.

ELECTRICAL / OPTICAL CHARACTERISTICS (T_A =25°C) (All measurements made under pulse conditions)

PARAMETER	TEST CONDITIONS	SYMBOL	MIN	TYP	MAX	UNITS
Peak Emission Wavelength	I_F = 100 mA	λ_P	—	940	—	nm
Emission Angle at 1/2 Power	I_F = 100 mA	θ	—	±8	—	Deg.
Forward Voltage	I_F = 100 mA	V_F	—	—	1.7	V
Reverse Leakage Current	V_R = 3 V	I_R	—	—	10	µA
Total Power LED55B(7)	I_F = 100 mA	P_O	3.5	—	—	mW
Total Power LED55C(7)	I_F = 100 mA	P_O	5.4	—	—	mW
Total Power LED56(7)	I_F = 100 mA	P_O	1.5	—	—	mW
Rise Time 0-90% of output		t_r	—	1.0	—	µs
Fall Time 100-10% of output		t_f	—	1.0	—	µs

GaAs INFRARED EMITTING DIODE

LED55B LED55C LED56

TYPICAL PERFORMANCE CURVES

Figure 1. Power Output vs. Input Current

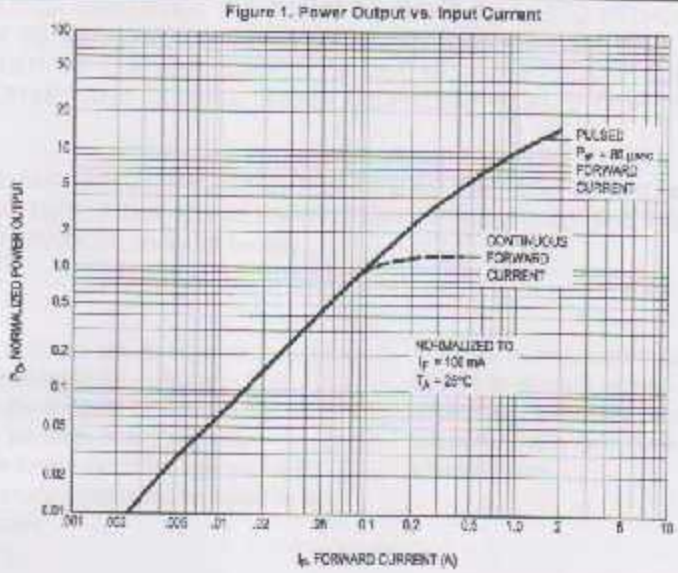


Figure 2. Power Output vs. Temperature

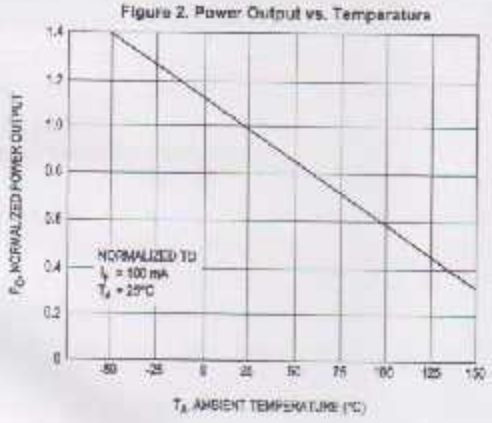


Figure 3. Forward Voltage vs. Forward Current

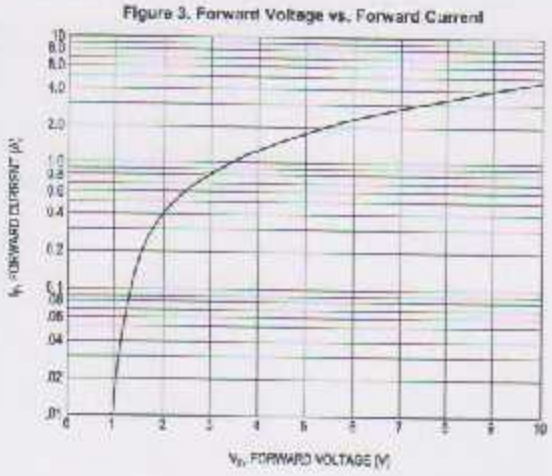


Figure 4. Forward Voltage vs. Forward Current

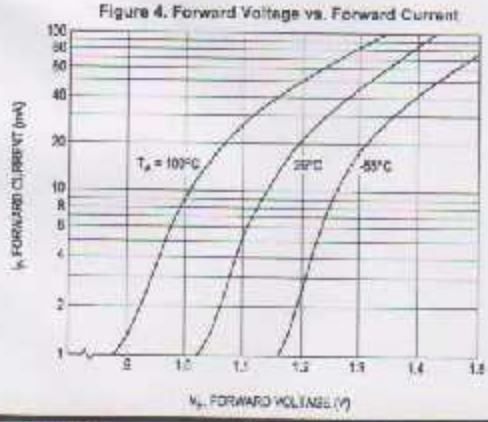
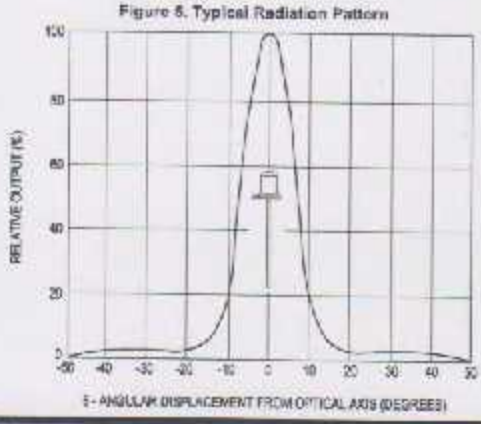


Figure 5. Typical Radiation Pattern



DISCLAIMER

FAIRCHILD SEMICONDUCTOR RESERVES THE RIGHT TO MAKE CHANGES WITHOUT FURTHER NOTICE TO ANY PRODUCTS HEREIN TO IMPROVE RELIABILITY, FUNCTION OR DESIGN. FAIRCHILD DOES NOT ASSUME ANY LIABILITY ARISING OUT OF THE APPLICATION OR USE OF ANY PRODUCT OR CIRCUIT DESCRIBED HEREIN; NEITHER DOES IT CONVEY ANY LICENSE UNDER ITS PATENT RIGHTS, NOR THE RIGHTS OF OTHERS.

LIFE SUPPORT POLICY

FAIRCHILD'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF THE PRESIDENT OF FAIRCHILD SEMICONDUCTOR CORPORATION. As used herein:

1. Life support devices or systems are devices or systems which, (a) are intended for surgical implant into the body,or (b) support or sustain life, and (c) whose failure to perform when properly used in accordance with instructions for use provided in labeling, can be reasonably expected to result in a significant injury of the user.

2. A critical component in any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.

Low-Cost E Series Multifunction DAQ
16-Bit, 200 kS/s, 16 Analog Inputs

NI 6034E, NI 6036E

- 16 analog inputs at 200 kS/s, 16-bit resolution
- Up to 2 analog outputs, 16-bit resolution
- 8 digital I/O lines (5 V TTL/CMOS); two 24-bit counter/timers
- Digital triggering
- 4 analog input signal ranges
- NI-DAQ driver simplifies configuration and measurements

Models
NI 6034E
- NI PCI-6034E

NI 6036E
- NI PCI-6036E
- NI DAQCard-6036E ***NEW!***

Operating Systems
- Windows 2000/NT/XP/Me/9x
- Mac OS 9*

Recommended Software
- LabVIEW
- LabWindows/CVI
- Measurement Studio for Visual Basic
- VI Logger

Other Compatible Software
- Visual Basic
- C/C++

Driver Software (included)
- NI-DAQ

Calibration Certificate Included

*See ordering information

NEW

Overview and Applications

NI 6034E and NI 6036E devices use E Series technology to deliver high performance, reliable data acquisition capabilities. These devices enable a broad variety of applications including:

- Continuous high-speed data logging at up to 200 kS/s
- Externally timed and/or triggered data acquisition
- High-voltage and sensor measurements when used with NI signal conditioning
- High channel-count synchronization with RTSI

Features

NI 6034E and NI 6036E devices feature the NI-PGIA, an instrumentation-class amplifier that guarantees settling times at all gains. Typical commercial off-the-shelf amplifier components don't meet the settling time requirements for high-gain measurement applications. Without the NI-PGIA, 16-bit devices with a 100X gain can have an effective resolution of 12 bits. The NI 6034E and NI 6036E devices also offer resolution improvement technologies such as dithering to reduce quantization error. This technology permits NI 16-bit multifunction DAQ devices to perform with an effective input resolution of at least 18 bits. These devices offer several methods in which to connect your signals including differential for eight analog input channels and maximum noise elimination, as well as referenced and nonreferenced single-ended for 16 analog input channels.

NI 6034E and NI 6036E devices feature digital triggering, two 24-bit 20 MHz counter/timers, and eight digital I/O lines compatible with both 5 V TTL and CMOS. NI 6036E devices also feature two 16-bit analog outputs.

Family	Bus	Analog Inputs	Resolution	Sampling Rate S/s	Input Range	Analog Outputs	Resolution	Output Rate	Output Range	Digital I/O	Counter/Timers	Triggers
NI 6034E	PCI	16 SE/8 DI	16 bits	200 kS/s	±0.05 to ±10 V	–	–	–	–	8	2, 24-bit	Digital
NI 6036E	PCI, PCMCIA	16 SE/8 DI	16 bits	200 kS/s	±0.05 to ±10 V	2	16 bits	10 kS/s*	±10 V	8	2, 24-bit	Digital

*10 kS/s maximum when using the single DMA channel for analog output. kS/s system dependent when using the single DMA channel for either analog input or counter/timer operations.

Table 1. NI 6036E and NI 6034E Channel, Speed, and Resolution Specifications

Low-Cost E Series Multifunction DAQ
16-Bit, 200 kS/s, 16 Analog Inputs

Nominal Range (V)		Absolute Accuracy						Relative Accuracy	
		% of Reading		Offset	Noise + Quantization (µV)		Temp Drift (%/°C)	Absolute Accuracy at Full Scale (mV)	Resolution (µV)	
Positive FS	Negative FS	24 Hrs	1 Year	(µV)	Single Pt.	Averaged			Single Pt.	Averaged
10	-10	0.0049	0.0098	1591.4	985.0	31.40	0.0035	3.563	1076.56	122.50
2	-2	0.0045	0.0105	306.2	197.5	28.90	0.0035	1.787	612.69	95.26
0.5	-0.5	0.0049	0.0095	86.5	53.4	4.15	0.0035	0.448	32.73	6.47
0.05	-0.05	0.0070	0.0093	28.4	23.0	4.97	0.0017	0.030	33.83	5.38

Note: Accuracies are valid for measurements following an internal E Series Calibration. Averaged numbers assume dithering and averaging of 100 single-channel readings. Measurement accuracies are listed for operational temperatures within ±1 °C of internal calibration temperature and ±10 °C of external or factory-calibration temperature. One-year calibration interval recommended. The Absolute Accuracy at Full Scale calculations were performed for a maximum range input voltage (for example, 10 V for the ±10 V range) after one year, assuming 100 pt averaging of data.

Table 2. NI PCI-6034E and NI PCI-6036E Analog Input Accuracy Specifications

Nominal Range (V)		Absolute Accuracy						Relative Accuracy	
		% of Reading		Offset	Noise + Quantization (mV)		Temp Drift (%/°C)	Absolute Accuracy at Full Scale (mV)	Resolution (mV)	
Positive FS	Negative FS	24 Hrs	1 Year	(mV)	Single Pt.	Averaged			Single Pt.	Averaged
10	-10	0.0077	0.0314	2.99	0.99	0.078	0.0010	12.104	1.03	0.39
2	-2	0.0272	0.0314	1.46	0.44	0.035	0.0005	3.007	0.51	0.091
0.5	-0.5	0.0077	0.0314	0.354	0.065	0.005	0.0010	0.978	0.063	0.006
0.05	-0.05	0.0272	0.0314	0.055	0.026	0.003	0.0010	0.094	0.034	0.009

Note: Accuracies are valid for measurements following an internal E Series Calibration. Averaged numbers assume dithering and averaging of 100 single-channel readings. Measurement accuracies are listed for operational temperatures within ±1 °C of internal calibration temperature and ±10 °C of external or factory-calibration temperature. One-year calibration interval recommended. The Absolute Accuracy at Full Scale calculations were performed for a maximum range input voltage (for example, 10 V for the ±10 V range) after one year, assuming 100 pt averaging of data.

Table 3. NI DAQCard-6036E Analog Input Accuracy Specifications

Family	Nominal Range (V) FS	Absolute Accuracy					Absolute Accuracy at Full Scale (mV)
		% of Reading			Offset (mV)	Temp Drift (%/°C)	
		24 Hrs.	90 Days	1 Year			
NI PCI-6036E	±10	0.009	0.011	0.013	1.1	0.0005	2.412
NI DAQCard-6036E	±10	0.009	0.011	0.013	1.22	0.0005	2.543

Note: Temp Drift applies only if ambient is greater than ±10 °C of previous external calibration.

Table 4. NI PCI-6036E and NI DAQCard-6036E Analog Output Accuracy Specifications

Low-Cost E Series Multifunction DAQ
16-Bit, 200 kS/s, 16 Analog Inputs

Driver Software

NI-DAQ is the robust driver software included with all National Instruments data acquisition and signal conditioning products. This easy-to-use software tightly integrates the full functionality of your DAQ hardware to LabVIEW, LabWindows/CVI, and Measurement Studio for Visual Basic. High-performance features include multidevice synchronization, networked measurements, and DMA data management. Bundled with NI-DAQ, the Measurement & Automation Explorer utility simplifies the configuration of your measurement hardware with device test panels, interactive measurements, and scaled I/O channels. NI-DAQ also provides numerous example programs for LabVIEW and other application development environments to get you started with your application quickly.

Services and Support/Training

As a complement to your data acquisition and signal conditioning product, consider:

- **Technical Support** – Included in hardware/software purchase through applications engineers worldwide, Web resources with more than 1000 example programs and more than 7000 Knowledge Bases, and Premier Support – *ni.com/support*
- **NI Factory Installation Services (FIS)** – Software and hardware installed in PXI and PXI/SCXI systems, tested and ready to use – *ni.com/advisor*
- **Calibration** – Includes NIST-traceable basic calibration certificate, services for ANSI/NCSL-Z540 and periodic calibration – *ni.com/calibration*
- **Extended Warranty** – Meet project life-cycle requirements and maintain optimal performance in a cost-effective way – *ni.com/services*
- **Data Acquisition Training** – Instructor-led courses – *ni.com/training*
- **Professional Services** – Feasibility, consulting, and integration through our Alliance Program members – *ni.com/alliance*

For more information on NI services and support, visit *ni.com/services*

Related Products

For related products, please refer to:

- SCXI Signal Conditioning
- SCC Signal Conditioning
- Analog Output Multifunction DAQ
- High-speed Digital I/O

Tech Tip

Learn how to reduce your development time and system costs. Visit *ni.com/info* and enter **mreadydaq** to download an interactive white paper on the benefits of Measurement Ready DAQ – measurement quality, software integration, and solutions support.

For more information, visit *ni.com/info* and enter: mreadydaq

Ordering Information

NI PCI-6034E	778035-01
NI PCI-6036E	778466-01
NI DAQCard-6036E	778561-01

Includes NI-DAQ driver software
Compatible with Mac OS 8

Recommended Configurations

Family	DAQ Devices	Accessory	Cable
NI 6034E	NI PCI-6034E	CB-68LP (777105-01)	R6868 (182402-01)
NI 6036E	NI PCI-6036E	CB-68LP (777105-01)	R6868 (182402-01)
NI 6036E	NI DAQCard-6036E	CB-68LP (777105-01)	SH68-68 (187252-01)

16-Bit E Series Multifunction DAQ Specifications

Specifications – 16-Bit E Series for 6052E and N 6023E

These specifications are typical for 25 °C unless otherwise noted.

Analog Input

Accuracy specifications ..... See tables in E Series product pages.

Input Characteristics

Number of channels

6062E 6030E 6032E 6034E 6036E	16 single-ended or 8 differential (software-selectable per channel)
6031E 6033E	64 single-ended or 32 differential (software-selectable per channel)

Resolution ..... 16 bits, 1 in 65,536

Maximum sampling rate

6062E	333 kS/s
6034E 6036E	250 kS/s
6052E 6030E 6031E 6032E 6033E	100 kS/s

Streaming-to-disk rate (system dependent)

6052E	333 kS/s
6034E 6036E	250 kS/s
6030E 6031E 6032E 6033E	100 kS/s

Stream-to-disk rates do not apply to PT Series devices.

Input signal ranges

Device	Range (Software Selectable)	Bipolar Input Range	Unipolar Input Range
6062E	20 V	±10 V	–
	10 V	±5 V	0 to 10 V
	5 V	±2.5 V	0 to 5 V
	2 V	±1 V	0 to 2 V
	1 V	±500 mV	0 to 1 V
	500 mV	±250 mV	0 to 500 mV
	200 mV	±100 mV	0 to 200 mV
	100 mV	±50 mV	0 to 100 mV
6030E 6031E 6032E 6033E	20 V	±10 V	–
	10 V	±5 V	0 to 10 V
	5 V	–	0 to 5 V
	4 V	±2 V	–
	2 V	±1 V	0 to 2 V
	1 V	±500 mV	0 to 1 V
	500 mV	–	0 to 500 mV
	400 mV	±200 mV	–
	200 mV	±100 mV	0 to 200 mV
	100 mV	–	0 to 100 mV
6034E 6036E	20 V	±10 V	–
	10 V	±5 V	–
	1 V	±500 mV	–
	100 mV	±50 mV	–

Input coupling ..... DC

Maximum working voltage (signal + common mode) ..... Each input should remain within ±11 V of ground.

Overvoltage protection

Powered on ..... ±25 V

Powered off ..... ±15 V

Inputs protected

6052E 6030E 6032E 6034E 6036E	ACH<0..15>, SENSE
6031E 6033E	ACH<0..63>, AISENSE AISENSE

FIFO buffer size ..... 512 samples, 4 (1024 samples for DAQCard)

Data transfers

PCI, PXI ..... DMA, interrupts, programmed I/O

DAQCard ..... interrupts, programmed I/O

DMA modes

PCI, PXI ..... Scatter-gather (single-transfer demand transfer)

Configuration memory size ..... 512 words

Transfer Characteristics

Relative accuracy (differential)

Device	Typical	Maximum
6062E 6034E PCI-6036E	±1.5 LSB	±3 LSB
6030E 6031E 6032E 6033E	±0.75 LSB	±1 LSB
DAQCard-6036E	±5.0 LSB	±8 LSB

DNL

Device	Typical	Maximum
6062E 6052E (except DAQCard-6062E)	±0.5 LSB	±1 LSB
DAQCard-6062E	±1.0 LSB	±2 LSB

No missing codes ..... 16 bits, guaranteed

Amplifier Characteristics

Input impedance

Device	Normal Powered On	Powered Off	Overload
6062E 6030E	100 GΩ in parallel with 100 pF	820 Ω	820 Ω

Input bias and offset current

Device	Bias Current	Offset Current
6052E 6034E PCI-6036E	±200 pA	±100 pA
6030E 6031E 6032E 6033E	±1 nA	±2 nA
DAQCard-6036E	±200 pA	±100 pA

16-Bit E Series Multifunction DAQ Specifications

Specifications — 16-Bit (NI 6052E and NI 6036E continued)

CMRR, DC to 60 Hz

Device	Range	CMRR	
		Bipolar (dB)	Unipolar (dB)
6052E	20 V	92	
	10 V	97	97
	5 V	100	101
	2 V	104	104
	100 mV to 1 V	105	105
6030E	20 V	92	–
6031E	10 V	97	92
6032E	5 V	–	97
6033E	4 V	101	
	2 V	104	101
	1 V	105	104
	100 mV to 500 mV	105	105
6034E	20 V	85	
6036E	10 V	85	–
	1 V	95	
	100 mV	95	

Dynamic Characteristics

Bandwidth

Device	Range	Small Signal (-3 dB)
6052E	All ranges	850 kHz
6030E	All ranges	250 kHz
6031E		
6032E		
6033E		
6034E	All ranges	70 kHz
6036E		

System noise (LSB, including quantization)

Device	Range	Bipolar	Unipolar
6052E	2 to 20 V	0.90	0.95
	1 V	1.1	1.1
	500 mV	1.5	1.3
	200 mV	2.3	2.7
	100 mV	5.0	5.0
6030E	2 to 20 V	0.6	0.8
6031E	1 V	0.7	0.8
6032E	500 to 500 mV	1.1	1.1
6033E	200 mV	2.0	2.0
	100 mV	–	3.6
6034E	10 to 20 V	0.9	
6036E	1 V	1.0	–
	100 mV	6.2	

Settling time to full-scale step

Device	Range	Accuracy				
		±0.00076% (±5 LSB)	±0.0015% (±1 LSB)	±0.0031% (±2 LSB)	±0.006% (±4 LSB)	±0.024% (±16 LSB)
6052E	2 to 20 V	–	10 µs max	5 µs max	4 µs max	3 µs max
	1 V		15 µs max	6 µs max	4 µs max	3 µs max
	200 to 500 mV	–	15 µs max	10 µs max	6 µs max	3 µs max
	100 mV		15 µs typical	10 µs typical	6 µs max	3 µs max
6030E	All	40 µs max	25 µs max	–	10 µs max	–
6032E						
6031E	All	140 µs max	25 µs max	–	10 µs max	–
6033E						
6034E	1 to 10 V	–	–	5 µs max	–	–
6036E	200 mV	–	–		5 µs typical	–

Crosstalk

Device	Adjacent Channels	All Other Channels
6052E	-75 dB	-90 dB
6036E		

Analog Output

Output Characteristics

Number of channels

6052E	2 voltage outputs
6030E	
6031E	
6032E	
6033E	
6034E	None

Resolution

6052E	16 bits, 1 in 65,536
6030E	
6036E	
6031E	

Maximum update rate

6052E	833 kS/s
6036E	1 kS/s, system dependent
6030E	100 kS/s
6031E	

Type of DAC ..... Double-buffered, multiplying

FIFO buffer size

6030E	2,048 samples
6032E	
6031E	
6036E	None

Data transfers

PCI, PXI ..... DMA, interrupts, programmed I/O

DaqCard ..... Interrupts, programmed I/O

DMA modes

PCI, PXI ..... Scatter-gather, single transfer, demand transfer

Transfer Characteristics

Relative accuracy

6052E	±0.35 LSB typical, ±1 LSB max
6030E	±0.5 LSB typical, ±1 LSB max
6031E	
6036E	±2 LSB max

DNL ..... <1/2 LSB max

Monotonicity

6052E	16 bits, guaranteed
6030E	
6032E	
6031E	

Voltage Output

Ranges

6052E	±10 V, 0 to 10 V, ±5 V, ±2 V, ±1 V, 0 to 5 V, 0 to 2 V, software selectable
6030E	±10 V, 0 to 10 V, software selectable
6031E	
6036E	±10 V

Output coupling ..... DC

Output impedance ..... 0.1 Ω max

Current drive ..... ±5 mA max

Protection ..... Short-circuit to ground

Power-on state

6052E	0 V (±20 mV)
6030E	
6031E	
PCI-6033E	0 V (±5 mV)
DAQCard-6036E	0 V (±10 mV)

16-Bit E Series Multifunction DAQ Specifications

Specifications – 16-Bit (NI 6052E and NI 605xE four-channel)

External reference input (6052E only)
- Range ..... ±11 V
- Overvoltage protection ..... ±25 V powered on, ±15 V powered off
- Input impedance ..... 1 MΩ
- Bandwidth (3 dB) ..... 3 kHz
- Slew rate ..... 0.3 V/µs

Dynamic Characteristics

Settling time (full-scale step)

Device	Settling Time for Full-Scale Step	Slew Rate
6052E	24 µs to ±1 LSB accuracy	1.5 V/µs
6030E 6031E	10 µs to ±1 LSB accuracy	5 V/µs
PCI-6036E	10 µs to ±0.5 LSB accuracy	15 V/µs
DAQCard-6062E	10 µs to ±0.5 LSB	5 V/µs

Noise

6062E 6030E 6031E	82 μV_{rms} DC to 1 MHz
PCI-6036E	110 μV_{rms} DC to 100 kHz
DAQCard-6036E	160 μV_{rms} DC to 100 kHz

Glitch energy (at mid-scale transition)

Device	Magnitude	Duration
6052E	±10 mV	1 µs
6030E 6031E	N/A	N/A
PCI-6032E	±10 mV	1 µs

Digital I/O

- Number of channels ..... 8 input/output
- Compatibility ..... 5 V TTL/LVCMOS
- Power-on state ..... Input (high impedance)
- Data transfers ..... Programmed I/O

Digital logic levels

Level	Minimum	Maximum
Input low voltage	0 V	0.8 V
Input high voltage	2 V	5 V
Output low voltage (Iout = 24 mA)	–	0.4 V
Output high voltage (Iout = -13 mA)	4.35 V	–

Timing I/O

General-Purpose Up/Down Counter/Timers

- Number of counters 2
- Resolution ..... 24 bits (1 in 16,777,216)
- Compatibility ..... 5 V TTL/LVCMOS

Digital logic levels

Level	Minimum	Max
Input low voltage	0.0 V	0.8 V
Input high voltage	2.0 V	5.0 V
Output low voltage (Iout = 5 mA)	–	0.4 V
Output high voltage (Iout = -3.5 mA)	4.35 V	

- Base clocks available ..... 20 MHz and 100 kHz
- Base clock accuracy ..... ±0.01%
- Maximum source frequency ..... 20 MHz
- External source selections ..... PFI <0..9>, RTSI <0..6>, analog trigger, software selectable
- External gate selections ..... PFI <0..9>, RTSI <0..6>, analog trigger, software selectable
- Minimum source pulse duration ..... 10 ns
- Minimum gate pulse duration ..... 10 ns, edge-detect mode
- Data transfers
  - PCI, PXI ..... DMA, interrupts, programmed I/O
  - DAQCard ..... Interrupts, programmed I/O
- DMA modes
  - PCI, PXI ..... Scatter-gather, single transfer, demand transfer

Frequency Scaler

- Number of channels 1
- Resolution ..... 4 bits
- Compatibility ..... 5 V TTL
- Digital logic levels

Level	Minimum	Max
Input low voltage	0.0 V	0.8 V
Input high voltage	2.0 V	5.0 V
Output low voltage (Iout = 5 mA)	–	0.4 V
Output high voltage (Iout = -3.5 mA)	4.35 V	–

- Base clocks available ..... 20 MHz, 100 kHz
- Base clock accuracy ..... ±0.01%
- Data transfers ..... Programmed I/O

Triggers

Analog Triggers

- Number of triggers

6052E 6030E 6031E 6032E 6033E	1
6034E 6035E	None

- Purpose
  - Analog input ..... Start and stop trigger, gate, clock
  - Analog output ..... Start trigger, gate, clock
  - General-purpose counter/timers ..... Source, gate
- Source

6052E 6030E 6032E	ACH <0..15>, PFI0/TRIG1
6031E 6033E	ACH <0..63>, PFI0/TRIG1

- Level
  - Internal source, ACH <0..15,63> ..... ±Full scale
  - External source, PFI0/TRIG1 ..... ±10 V
- Slope ..... Positive or negative, software-selectable
- Resolution ..... 12 bits, 1 in 4,096
- Hysteresis ..... Programmable
- Bandwidth (-3 dB)

Device	Internal source ACH <0..15,63>	External Source PFI0/TRIG1
6052E	100 kHz	100 kHz
6030E, 6031E, 6032E, 6033E	250 kHz	4 MHz

- Accuracy ..... ±1% of the scale range used

Digital Triggers (all devices)

- Number of triggers 2
- Purpose
  - Analog input ..... Start and stop trigger, gate, clock
  - Analog output ..... Start trigger, gate, clock
  - General-purpose counter/timers ..... Source, gate
- Source ..... PFI <0..9>, RTSI <0..6>
- Slope ..... Positive or negative, software selectable
- Compatibility ..... 5 V TTL
- Response ..... Rising or falling edge
- Pulse width ..... 10 ns minimum

16-Bit E Series Multifunction DAQ Specifications

Specifications – 16-Bit (for 6032E and 6052E boards)

External Input for Digital or Analog Trigger (PFI0/TRIG1)
Impedance ..... 10 kΩ
Coupling ..... DC
Protection
Digital trigger ..... -0.5 to (Vcc + 0.5) V
Analog trigger
OVCM Disabled ..... ±35 V

Calibration

Recommended warm-up time ..... 15 minutes, 30 minutes for DAQCard
Calibration interval ..... 1 year
Onboard calibration reference
DC level

6023E 6024E 6025E 6032E 6033E	5.000 V ±1.0 mV	Over full operating temperature; actual value stored in EEPROM
6034E 6036E	5.000 V ±3.0 mV	

Temperature coefficient

6023E 6024E 6025E 6032E 6033E	±2.5 ppm/°C max
6034E 6035E 6036E	±5.0 ppm/°C max

Long-term stability

6032E 6033E 6034E 6035E 6036E	±8.0 ppm/√1000 h
6023E 6024E	±15.0 ppm/√1000 h

RTSI (PCI only)
Trigger lines 7

PXI Trigger Bus (PXI only)
Trigger lines 8
Star Trigger 1

Bus Interface
PCI, PXI ..... Master, slave
DAQCard ..... Slave

Power Requirements[2]

Device	+5 VDC (±5%)	Power Available at I/O Connector
6023E	1.3 A	+4.65 to +5.25 VDC, 1 A
6024E, 6025E, 6032E, 6033E	1.6 A	+4.65 to +5.25 VDC, 1 A
6034E, 6036E PCI-6036E	0.9 A	+4.65 to +5.25 VDC, 1 A
DAQCard-6062E	300 mA	+4.65 to +5.25 VDC, 0.1 A

Physical[3]

Dimensions (not including connectors)
PCI ..... 17.5 by 10.6 cm (6.9 by 4.2 in.)
PXI ..... 16.0 by 10.6 cm (6.3 by 3.9 in.)
DAQCard ..... Type II PC Card

I/O connectors

6023E 6024E 6025E 6032E 6034E PCI-6036E	68-pin male SCSI-II type
6031E 6033E	100-pin female 0.050 D-type
DAQCard-6062E	68-position VHDCI female

Environment

Operating temperature ..... 0 to 55 °C (DAQCard should not exceed 45 °C while in PCMCIA slot
Storage temperature ..... -55 to 150 °C
Relative humidity ..... 10 to 90%, noncondensing

Certifications and Compliances

CE Mark Compliance CE

See RT Series devices for RT Series power requirements and physical parameters.